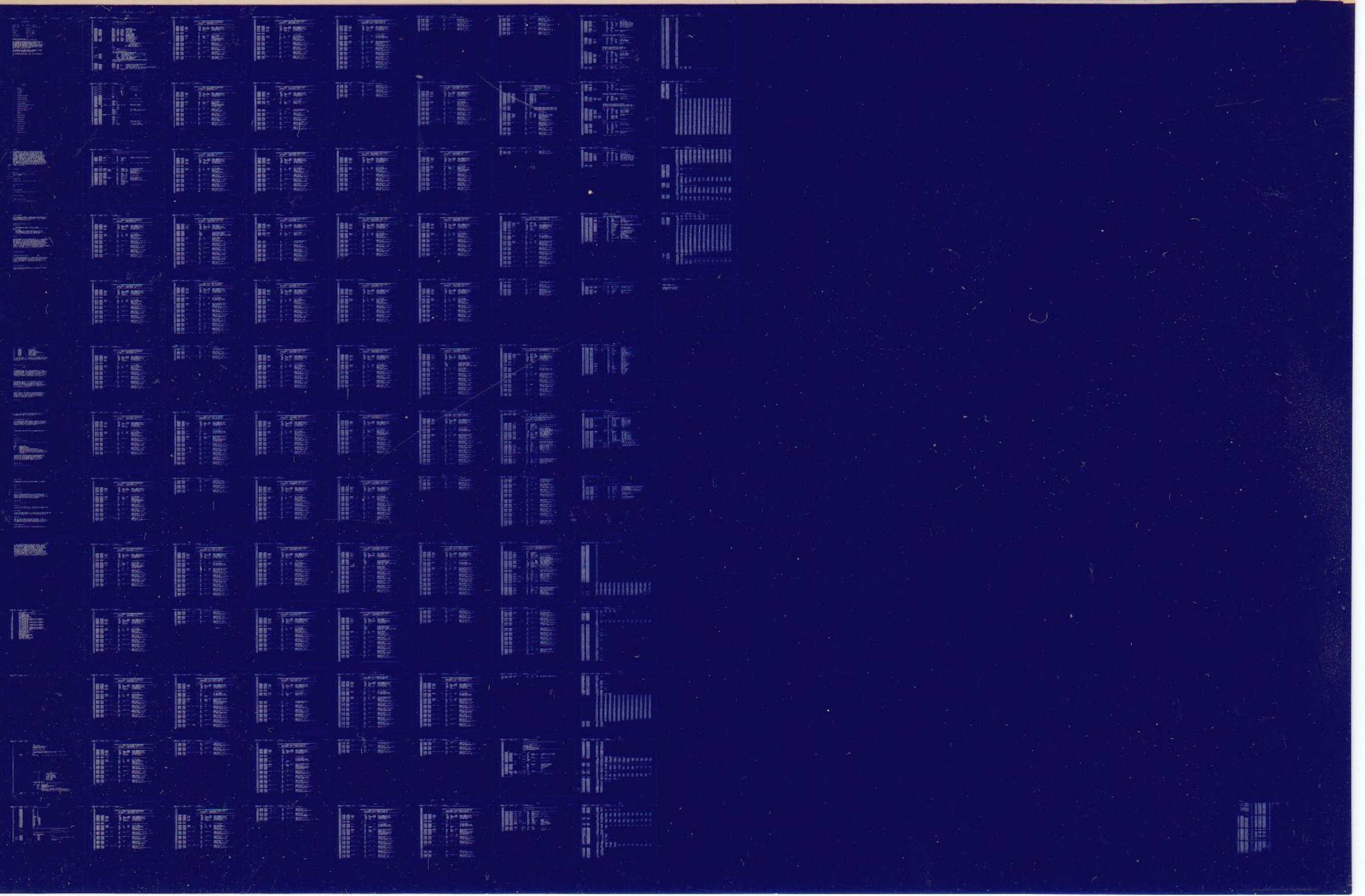


LSI-11

LSI-11 FIS INSTR. TESTS
MD-11-DVKAC-B

EP-DVKAC-B-DL-A
COPYRIGHT © 1977
FICHE 1 OF 1

APR 1977
digital
MADE IN USA



B01

EOF1DZD00000411

00010000

770413

PDP10 411

\$AHDR1DVKACBSEQ

00010000

770413

IDENTIFICATION

SEQ 0001

PRODUCT CODE: MAINDEC-11-DVKAC-B-D
PRODUCT NAME: LSI-11 FIS INSTRUCTION TESTS
DATE: MARCH, 1977
MAINTAINER: DIAGNOSTIC GROUP
AUTHOR: PERVEZ A. ZAKI
MODIFIED BY: ALAN W. BOSTICK
DATE MODIFIED: JANUARY 1977

COPYRIGHT (C) 1975, 1977
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE FOR USE ONLY ON A SINGLE COMPUTER SYSTEM AND MAY BE COPIED ONLY WITH THE INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE, OR ANY OTHER COPIES THEREOF, MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE TERMS. TITLE TO AND OWNERSHIP OF THE SOFTWARE SHALL AT ALL TIMES REMAIN IN DEC.

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.

DEC ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.

CONTENTS

1. ABSTRACT
2. REQUIREMENTS
 - 2.1 EQUIPMENT
 - 2.2 STORAGE
 - 2.3 PRELIMINARY PROGRAMS
3. LOADING PROCEDURE
4. STARTING PROCEDURE
 - 4.1 CONTROL SWITCH SETTINGS
 - 4.2 STARTING ADDRESS
 - 4.3 PROGRAM AND/OR OPERATOR ACTION
5. OPERATING PROCEDURE
 - 5.1 OPERATIONAL SWITCH SETTINGS
 - 5.2 SUBROUTINE ABSTRACTS
6. ERRORS
 - 6.1 ERROR PRINTOUT
 - 6.2 ERROR RECOVERY
 - 6.3 ERROR COUNTER
7. RESTRICTIONS
8. MISCELLANEOUS
 - 8.1 EXECUTION TIME
 - 8.2 STACK POINTER
 - 8.3 PASS COUNTER
 - 8.4 TEST NUMBER
 - 8.5 POWER FAIL
9. PROGRAM DESCRIPTION

1. ABSTRACT

THIS PROGRAM TESTS THE LSI-11 FLOATING INSTRUCTION SET (FADD, FSUB, FMUL, AND FDIV) OPTION WITH FIXED NUMBER PATTERNS, USING EACH REGISTER AT LEAST ONCE AS THE STACK POINTER. IT ALSO CHECKS STACK OVERFLOW AND THAT THE FLOATING INSTRUCTIONS CAN BE INTERRUPTED (BY THE CONSOLE TELETYPE) (HOWEVER THIS TEST WILL NOT BE EXECUTED WHEN BIT 5 OF SENVM BYTE IS HIGH). THE PROGRAM SHOULD BE RUN FOR AT LEAST 2 PASSES WITH ALL SWITCHES LOW. THE PROGRAM IS DESIGNED TO RUN UNDER APT. AND ACT. SYSTEMS. WHEN RUNNING UNDER APT. WITH BIT 5 OF SENVM LOW IT WILL BE REQUIRED TO HAVE A SLU WITH TTY REGISTERS HAVING ADDRESSES OF 176560-66 AND INTERRUPT VECTORS OF 70 FOR RECEIVER AND 74 FOR TRANSMITTER. UNDER SUCH A CONDITION IT WILL ALSO BE REQUIRED TO CHANGE THE RUN TIME OF FIRST PASS FROM 5 SECONDS TO THE TIME GIVEN IN SEC. 8.1, AND THE RUN TIME FOR THE LONGEST TEST FROM 3 SECONDS TO 30 SECONDS.

2. REQUIREMENTS

2.1 EQUIPMENT

LSI-11 STANDARD COMPUTER WITH FIS OPTION
AND 4K OF MEMORY

2.2 STORAGE

PROGRAM STORAGE - THE ROUTINES USE MEMORY 0 - 17500

2.3 PRELIMINARY PROGRAMS

NONE

3. LOADING PROCEDURE

USE STANDARD PROCEDURE FOR ABS TAPES.

4. STARTING PROCEDURE

4.1 CONTROL SWITCH SETTINGS

SEE 5.1.1 (ALL LOW FOR WORST CASE TESTING)

4.2 STARTING ADDRESS

AFTER LOADING THE PROGRAM IT SHOULD ALWAYS BE STARTED AT 200.
IF IT IS DESIRED TO SAVE THE PASS COUNTER THEN THE PROGRAM
SHOULD BE RESTARTED AT LOCATION RESTRT (I.E. 222) OTHERWISE THE
PROGRAM CAN BE RESTARTED AT 200

4.3 PROGRAM AND/OR OPERATOR ACTION

4.3.1 STAND ALONE

- 1) LOAD PROGRAM INTO MEMORY USING ABS LOADER.
- 2) SET SWITCHES (SEE SEC 5.1.1) ALL LOW EXCEPT BIT 7 FOR WORST CASE.
- 3) TYPE 200G
- 4) THE PROGRAM WILL LOOP AND "END PASS" WILL BE TYPED AFTER COMPLETION OF EVERY PASS. HOWEVER TYPE OUT WILL BE SUPPRESSED IF BIT 5 OF LOCATION SENVM IS HIGH
- 5) A MINIMUM OF TWO PASSES SHOULD ALWAYS BE RUN.

4.3.2 UNDER APT

LOAD THE PROGRAM, SET THE SWITCHES (SEE SEC. 5.1.1) AND START. WHEN UNDER APT. WITH BIT 5 OF SENVM LOW IT WILL BE REQUIRED TO HAVE A SLU WITH TTY REGISTERS HAVING ADDRESSES OF 176560-66 AND INTERRUPT VECTORS OF 70 FOR RECEIVER AND 74 FOR TRANSMITTER. UNDER SUCH A CONDITION IT WILL ALSO BE REQUIRED TO CHANGE THE RUN TIME OF FIRST PASS FROM 5 SECONDS TO THE TIME GIVEN IN SEC. 8.1, AND THE RUN TIME FOR THE LONGEST TEST FROM 3 SECONDS TO 30 SECONDS. THE TEST TIMES AND PASS TIMES ARE SUGGESTED WITH BIT 7 OF \$SWREG. LOW, IF IT IS DESIRED TO ENABLE THE ITERATIONS THEN THE TIMES SHOULD BE MULTIPLIED BY A FACTOR OF 256.

5. OPERATING PROCEDURE

5.1 OPERATIONAL SWITCH SETTINGS

ALL SWITCHES LOW EXCEPT SW<11> IS WORST CASE TESTING. WITH BIT 11 OF THE LOCATION \$SWREG (I.E. LOCATION 422), HIGH EACH SUBTEST WILL BE LOOPED UPON UNTIL COMPLETION OF 256 PASSES OF THAT SUBTEST. "END PASS" WILL BE TYPED UPON COMPLETION OF A PASS OF THE ENTIRE PROGRAM. ALTERNATE PASS WILL RUN WITH THE T-BIT SET.

5.1.1 SWITCH SETTINGS ARE:

A 16 BIT LOCATION CALLED \$SWREG (I.E. LOCATION 422) HAS BEEN USED TO GIVE THE FOLLOWING OPTIONS BY INSERTING A 1 IN THEIR RESPECTIVE POSITIONS

F01

BIT #	OCTAL VALUE	FUNCTION
15	100000.....	HALT ON ERROR
14	040000.....	SCOPE LOOP
13	020000.....	INHIBIT PRINTOUT
12	010000.....	INHIBIT TRACE TRAPPING
11	004000.....	ENABLE ITERATIONS OF SUBTEST
10	002000.....	BELL ON ERROR
09	001000.....	LOOP ON ERROR
08	0004XX.....	LOOP ON TEST IN BITS 7 THRU 0

AN 8 BIT BYTE SENVM [I.E. LOCATION 421] HAS BEEN USED TO DEFINE THE OPERATING MODE. ALL TYPEOUTS CAN BE SUPPRESSED BY MAKING BIT 5 OF BYTE SENVM HIGH, IN OTHER WORDS BY PLACING A 20000 IN LOCATION 420

5.2 SUBROUTINE ABSTRACTS

5.2.1 SCOPE

THIS SUBROUTINE CALL (VIA A TRAP INSTRUCTION) IS PLACED BETWEEN EACH SUBTEST IN THE INSTRUCTION SECTION. IT RECORDS THE STARTING ADDRESS OF EACH SUBTEST AS IT IS BEING ENTERED IN LOCATION "LADS". IF A SCOPE LOOP IS REQUESTED, THE CURRENT SUBTEST WILL BE LOOPED UPON. SW<11> IS A 1 INHIBITS ITERATION OF SUBTESTS. THE CONTENTS OF "LADS" MAY BE USED TO DETERMINE THE LAST SUBTEST SUCCESSFULLY COMPLETED.

5.2.2 HLT

THIS ROUTINE (CALLED BY AN EMT INSTRUCTION) PRINTS OUT AN ERROR MESSAGE (SEE 6.1.). IF SW<9> IS A 1 AND A HLT IS EXECUTED, THE SUBTEST WILL BE LOOPED UPON UNTIL 256 CONSECUTIVE GOOD PASSES ARE COMPLETED. TO INHIBIT TYPEOUTS, MAKE SW<13> A 1. TO RING THE BELL ON AN ERROR, MAKE SW<10> A 1. A HIGH BIT 5 IN LOCATION SENVM WILL INHIBIT ANY TYPEOUTS AND RINGING OF BELLS

5.2.3 T BIT TRAP

IF SW<12> IS A 0, THE T-BIT WILL BE SET ON ALTERNATE PASSES. WHEN THE T-BIT IS SET, THE PROCESSOR TRAPS AFTER EACH INSTRUCTION. THE FIRST INSTRUCTION EXECUTED UPON TRAPPING IS AN "RTT" WHICH RETURNS TO THE INTERRUPTED SEQUENCE OF INSTRUCTIONS. THIS SEQUENCE IS CONTINUED UNTIL THE END OF THE PROGRAM IS REACHED.

5.2.4 TRAP CATCHER

A ".+2" - "HALT" SEQUENCE IS REPEATED FROM 0-776 TO CATCH ANY UNEXPECTED TRAPS. THUS ANY UNEXPECTED TRAPS OR INTERRUPTS WILL HALT AT THE VECTOR +2.

5.2.5 FLOATING ERROR TRAP (TO 244)

IF A FLOATING POINT ERROR (OVERFLOW, UNDERFLOW, OR DIVIDE BY ZERO) WAS EXPECTED, THE VECTOR WILL POINT TO A UNIQUE ISR WITHIN THE SUBTEST WHERE THE ERROR OCCURRED WHICH CHECKS THE DATA ON THE STACK(S). IF AN ERROR WAS NOT ANTICIPATED, AN ERRONEOUS TRAP WILL BE DETECTED IN TRAPER.

5.2.6 NOP

A NOP IS PLACED JUST BEFORE EACH FIS INSTRUCTION. THIS ALLOWS THE OPERATOR TO PATCH IN A HALT FOR DEBUGGING PURPOSES

6. ERRORS

6.1 ERROR PRINTOUT

THE FORMAT IS AS FOLLOWS:

```
ERRNM  ADR  PS  SP  ANS1  ANS2  ANS3  ANS4  ANS5  ANS6
```

WHERE:

```
ERRNM  = ERROR NUMBER
ADR     = ADDRESS OF ERROR HLT
PS      = PROCESSOR STATUS
SP      = CONTENTS OF STACK POINTER REGISTER
ANS1-6  = ERROR DATA READ FROM THE STACK(S). FROM 0 TO 6 OF
          THESE MAY BE TYPED DEPENDING ON THE NUMBER
          FOLLOWING THE HLT: E.G., HLT+3 WOULD TYPE ANS1 THRU
          ANS3, HLT (BY ITSELF) WOULD STOP AFTER ERRNM, ADR,
          PS, AND SP.
```

TO FIND THE FAILING TEST, LOOP AT THE LISTING ABOVE THE ADDRESS TYPED. IN MOST CASES THE COMMENT BESIDE THE HLT TELLS WHAT WAS BEING CHECKED AND WHAT WAS EXPECTED. ALL PRINTOUTS WILL BE SUPPRESSED WHEN BIT 5 OF LOCATION SENVM IS HIGH. WHILE RUNNING UNDER APT THE DIAGNOSTIC WILL NOT SUPPORT SPOOLING OF CONSOLE OUTPUTS.

6.2 ERROR RECOVERY

RESTART AT 200 OR 222 (SEE SEC 4.2)

6.3 ERROR COUNTER

AN ERROR COUNT IS KEPT IN LOCATION "ERRORS". IT CAN ONLY BE CLEARED FROM THE CONSOLE OR BY RELOADING THE PROGRAM.

7. RESTRICTIONS

NONE

8. MISCELLANEOUS**8.1 EXECUTION TIME**

DUE TO THE RANDOM CHARACTERISTIC OF THE INTERRUPT TESTS, THE EXECUTION TIME CAN BE HALF A MINUTE OR MORE. HOWEVER, NORMALLY "END PASS" WILL BE TYPED WITHIN 40 SECONDS WITH ALL SWITCHES DOWN. EXECUTION TIME WILL INCREASE BY A FACTOR OF 256 WHEN ITERATIONS OF EACH SUBTEST ARE ENABLED.

8.2 STACK POINTER

STACK IS INITIALLY SET TO 600

8.3 PASS COUNT

A 16 BIT LOCATION "SPASS" (I.E. LOCATION 406) IS USED TO KEEP PASS COUNT. IT CAN BE CLEARED BY RESTARTING THE PROGRAM AT 200

8.4 TEST NUMBER

A 16 BIT LOCATION "STESTN" (I.E. LOCATION 404) IS USED TO KEEP TRACK OF THE TEST NUMBER. UPPER BYTE OF THIS LOCATION GIVES THE ITERATION NUMBER AND THE LOWER BYTE THE TEST THAT WAS BEING EXECUTED

8.5 POWER FAIL

EACH TEST CAN BE POWER FAILED WITH NO ERRORS. TO USE, START THE TEST AS USUAL AND POWER DOWN THEN UP AT ANY TIME. THE PROGRAM SHOULD TYPE "POWER" AND CONTINUE TO RUN FROM WHERE THE POWER FAIL INTERRUPTED WITH NO OTHER ERROR TYPEOUTS.

9. PROGRAM DESCRIPTION

THIS PROGRAM TESTS ALL THE FIS INSTRUCTIONS OF THE LSI-11 (FADD, FSUB, FMUL, AND FDIV). ALL REGISTERS ARE CHECKED TO SEE

J01

SEQ 0008

IF THEY FUNCTION PROPERLY AS THE STACK POINTER. THE PROGRAM HAS MANY SUBTESTS (THE CODE BETWEEN 2 SCOPE STATEMENTS) WHICH ARE RUN ONCE BEFORE CONTINUING TO THE NEXT SUBTEST. SW<11> SET TO A 1 CAUSES EACH SUBTEST TO BE RUN 256 TIMES SW<9> SET TO A 1 ENABLES LOOP ON ERROR. THE LOCATION SICNT CONTAINS THE ITERATION COUNT AND THE LOCATION \$TESTN CONTAINS THE TEST NUMBER. ALL THE SUBTESTS SHOULD BE RUN SEQUENTIALLY BY STARTING AT 200 NOT BY STARTING AT THE BEGINNING OF THE SUBTEST. TO LOOP ON A PARTICULAR SUBTEST, PUT THE TEST NUMBER (SEE LISTING) IN THE RIGHT BYTE OF THE LOCATION \$SMREG AND SW<8> SET TO A 1. THIS TEST WILL BE LOOPED UPON UNTIL SW<8> IS SET TO A 0 OR THE RIGHT BYTE IS CHANGED. IF THE TEST IS NON-EXISTENT, THE PROGRAM WILL BE RUN AS USUAL.

32	SWITCH OPTIONS AND ASSIGNMENTS
81	ACT11 HOOKS
91	VECTOR AREA, STACKS, ANSWER AREA, AND SETUP ROUTINE
96	RPT MAILBOX-ETABLE
126	RPT PARAMETER BLOCK
193	STARTING OF THE PROGRAM
230	FADD TEST SECTION
883	TEST FLOATING ADD INSTRUCTION WITH UNDERFLOW
1028	TEST FLOATING ADD INSTRUCTION WITH OVERFLOW
1173	FSUB TEST SECTION
1741	TEST FLOATING SUB. INSTRUCTION WITH UNDERFLOW
1814	TEST FLOATING SUB. INSTRUCTION WITH OVERFLOW
1887	FML TEST SECTION
2272	TEST FLOATING MUL. INSTRUCTION WITH UNDERFLOW
2345	TEST FLOATING MUL. INSTRUCTION WITH OVERFLOW
2418	FDIV TEST SECTION
2657	TEST FLOATING DIV. INSTRUCTION WITH UNDERFLOW
2730	TEST FLOATING DIV INSTRUCTION WITH OVERFLOW
2803	TEST FLOATING DIV. INSTRUCTION FOR DIVIDE BY ZERO
2953	TEST OF ALL FIS AT ONCE
3016	ADDRESS ERROR TEST
3145	INTERUPT ABORT TEST SECTION
3365	END OF PASS ROUTINE
3404	SCOPE ROUTINE
3429	PUSH AND POP SUBROUTINES
3563	HLT ROUTINE (ERROR TYPEOUT)
3596	USER ERROR ROUTINE
3609	OCTAL WORD & ADDRESS TYPER
3644	POWER DOWN AND UP ROUTINES
3682	ASCIZ TYPE OUT ROUTINE

L01

DVKACB MACY11 27(1006) 17-JAN-77 15:14 PAGE 2
DVKACB.P11 13-JAN-77 15:28

SEQ 0010

1

10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60

```

.ABS
.TITLE DVKACB
.*COPYRIGHT (C) AUGUST 1975
.*DIGITAL EQUIPMENT CORP.
.*MAYNARD, MASS. 01754
.*
.*PROGRAM BY PERVEZ ZAKI
.*
.*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
.*PACKAGE (MAINDEC-11-DZQAC-B), JULY 11, 1975.
.*
$TN=1
$$SWR=160000 ;;HALT ON ERROR, LOOP ON TEST, INHIBIT ERROR TYP0UT

```

000001
160000

SWITCH	USE
8	LOOP ON TEST IN SW<7:0>
9	LOOP ON ERROR
10	1 - BELL ON ERROR
11	INHIBIT ITERATIONS
12	INHIBIT TRACE TRAP
13	INHIBIT ERROR TYPEOUTS
14	LOOP ON TEST
15	HALT ON ERROR

```

:ERROR MESSAGE FORMAT:
ERRNM ADR PSW SP ANS1 ANS2 ANS3 ANS4 ANS5 ANS6

```

```

:WHERE ERRNM= ERROR NUMBER
ADR = ADDRESS OF "HLT" INSTRUCTION + 2
PSW = PROCESSOR STATUS WORD
SP = STACK POINTER
ANS1 THRU ANS6 = DATA OFF THE STACK(S)
NOTE: ANS1 THRU ANS6 ARE NOT ALWAYS TYPED, DEPENDING ON THE
NUMBER ADDED TO THE "HLT". "HLT" ALONE TYPES NONE.
"HLT+1" TYPES ANS1, "HLT+2" TYPES ANS1 AND ANS2, ETC.

```

104000
000000

HLT= EMT
RO= %0

SWITCH OPTIONS AND ASSIGNMENTS

58 000001
59 000002
60 000003
61 000004
62 000005
63 000005
64 000006
65 000007
66 000024
67 104400
68 100000
69 040000
70 020000
71 010000
72 004000
73 002000
74 001000
75 000400
76 000004
77 000001
78 000001
79
80
81 000000
82
83
84
85
86
87 001000
88 000046
89 000046 015650
90 000052 000052
91 000052 000000
92 001000
93
94

R1= %1
R2= %2
R3= %3
R4= %4
R5= %5
TTY= %5
SP= %6
PC= %7
PWRVEC= 24
SCOPE= TRAP
SW15= 100000
SW14= 40000
SW13= 20000
SW12= 10000
SW11= 4000
SW10= 2000
SW09= 1000
SW08= 400
TYPE= IOT
N= 1
SF= 1

. = 0 ;TRAP CATCHER FROM 0 - 776

.SBTTL ACT11 HOOKS
;HOOKS REQUIRED BY ACT11
\$SVPC=
.=46
\$ENDAD
.=52
.WORD 0
.\$SVPC

;SAVE PC
;;1)SET LOC.46 TO ADDRESS OF \$ENDAD IN .SEOP
;;2)SET LOC.52 TO ZERO
;; RESTORE PC

95
96 000400
97
98
99
100
101
102
103
104 000400
105 000400 000000
106 000402 000000
107 000404 000000
108 000406 000000
109 000410 000000
110 000412 000000
111 000414 000000
112 000416 000000
113 000420
114 000420 000
115 000421 000
116 000422 000000
117 000424 000000
118 000426 000000
119
120
121
122
123
124
125 000430
126
127
128
129
130
131
132 000430
133 000024 000024
134 000024 000200
135 000044 000044
136 000044 000430
137 000430
138
139
140
141
142 000430
143 000430 000000
144 000432 000400
145 000434 000003
146 000436 000005
147 000440 000000
148 000442 000014
149 000430
150 000430

```
. = 400  
;*****  
.SBTTL APT MAILBOX-ETABLE  
  
.EVEN  
SMAIL: ; APT MAILBOX  
MSGTY: .WORD AMSGTY ; MESSAGE TYPE CODE  
SFATAL: .WORD AFATAL ; FATAL ERROR NUMBER  
STESTN: .WORD ATESTN ; TEST NUMBER  
SPASS: .WORD APASS ; PASS COUNT  
SDEVCT: .WORD ADEVCT ; DEVICE COUNT  
SUNIT: .WORD AUNIT ; I/O UNIT NUMBER  
SMSGAD: .WORD AMSGAD ; MESSAGE ADDRESS  
SMSGLG: .WORD AMSGLG ; MESSAGE LENGTH  
SETABLE: ; APT ENVIRONMENT TABLE  
SENV: .BYTE AENV ; ENVIRONMENT BYTE  
SENVH: .BYTE AENVH ; ENVIRONMENT MODE BITS  
SSWREG: .WORD ASWREG ; APT SWITCH REGISTER  
SUSWR: .WORD AUSWR ; USER SWITCHES  
SCPUOP: .WORD ACPUOP ; CPU TYPE, OPTIONS  
; BIT 15-11=CPU TYPE  
; 11/04=01, 11/05=02, 11/20=03, 11/40=04, 11/45=05  
; 11/70=06, PDQ=07, Q=10  
; BIT 10=REAL TIME CLOCK  
; BIT 9=FLOATING POINT PROCESSOR  
; BIT 8=MEMORY MANAGEMENT  
SETEND:  
.MEXIT  
;*****  
.SBTTL APT PARAMETER BLOCK  
;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT  
;*****  
.SX= ; SAVE CURRENT LOCATION  
=24 ; SET POWER FAIL TO POINT TO START OF PROGRAM  
200 ; FOR APT START UP  
=44 ; POINT TO APT INDIRECT ADDRESS PNTR.  
$APTHDR ; POINT TO APT HEADER BLOCK  
=.SX ; RESET LOCATION COUNTER  
;*****  
;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC  
;INTERFACE SPEC.  
  
$APTHD: ; TWO HIGH BITS OF 18 BIT MAILBOX ADDR.  
$HIBTS: .WORD 0 ; ADDRESS OF APT MAILBOX (BITS 0-15)  
$MADR: .WORD SMAIL ; RUN TIM OF LONGEST TEST  
$STHM: .WORD 3 ; RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)  
$PASTM: .WORD 5 ; ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT  
$UNITM: .WORD ; SETEND-SMAIL/2 ; LENGTH MAILBOX-ETABLE(WORDS)  
=. $APTHD  
HLTADS:
```



```

194 ;*****
195
196 .SBTTL STARTING OF THE PROGRAM
197
198
199      000200      000001      000330      .=      200
200 000200 012767 000001 000330      MOV      #1,TIMES      ;NUMBER OF ITERATIONS IN THE FIRST PASS=1
201 000206 012700 000410      MOV      #DEVCT,RO      ;PREPARE TO INITIALIZE THE PROGRAM
202 000212 005040      2$:      CLR      -(RO)
203 000214 022700 000400      CMP      #SMAIL,RO
204 000220 001374      BNE      2$
205
206 000222 000167 000352      RESTRT: JMP      BEGIN      ;JUMP TO STARTING ADDRESS OF PROGRAM
207
208
209
210      000600      .=      600
211
212 000600 012706 000600      BEGIN: MOV      #BEGIN, SP      ;INITIALIZE STACK POINTER
213 000604 012737 000542 000014      MOV      #YESRT, #14      ;SET TRACE TRAP VECTOR
214 000612 012737 017154 000020      MOV      #STYPE, #20      ;SET UP VECTOR 20
215 000620 012737 017014 000024      MOV      #SPWRDN, #24      ;SERVICE POWER DOWN ROUTINE FOR ANY FUTURE
216 ;POWER DOWN
217 000626 012700 000030      MOV      #30, RO      ;SET RO TO VECTOR 30
218 000632 012720 016450      MOV      #HLTS, (0)+      ;SET ENT VECTOR
219 000636 012720 000340      MOV      #340, (0)+
220 000642 012720 015702      MOV      #SCOPES, (0)+      ;SET TRAP VECTOR
221 000646 012710 000340      MOV      #340, (0)
222 000652 012737 000006 000004 1$:      MOV      #6, #4      ;RESTORE TIME-OUT VECTOR
223 000660 132737 000001 000420      BITB     #1, #SENV      ;ARE WE UNDER APT ?
224 000666 001410      BEQ      2$      ;IF NOT THEN GO TO 2$
225 000670 012700 000554      MOV      #STPB+2,RO      ;OTHERWISE SET FOR THE OTHER SLU
226 000674 012740 176566      MOV      #176566, -(RO)
227 000700 012740 176564      MOV      #176564, -(RO)
228 000704 012740 000074      MOV      #74, -(RO)
229 000710 005067 177470      2$:      CLR      $TESTN
230 000714 005067 177550      CLR      LADS      ;CLEAR LOOP ADDRESS
231
232
    
```

233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278

```
*****
:TEST 1: FADD (LSI-11 FLOATING ADD INSTRUCTION)
:          000000,000000 + 000000,000000 = 000000,000000
:          PS = 004,          STACK POINTER = RO
*****
```

```
TST1:  SCOPE
      JSR   RS,   PUSH4   ;PUSH 4 WORDS ONTO RO STACK, SET PRIORITY
      .WORD 000000,000000 ;SECOND OPERAND ON TOP
      .WORD 000000,000000 ;FIRST OPERAND ON BOTTOM
      .WORD 000          ;PROCESSOR PRIORITY LEVEL
      .WORD TRAPER,340   ;FIS TRAP VECTOR
      MOV   #STACK0,RO   ;SET UP STACK POINTER

      NOP
      FADD  RO           ;FLOATING ADD ON THE RO STACK

      JSR   PC,   POPR   ;POP THE ANSWER
      MOV   RO,   $SP    ;SAVE "STACK POINTER"
      CMPB #004,  $PSW   ;CHECK PS (EXCEPT T BIT)
      BEQ  .+6          ;BRANCH IF OK
      HLT  .+6          ;PS NOT EQUAL TO 004
                        ;THE ERROR NUMBER IS 1

      CMP   #STACK4,$SP ;CHECK THE STACK POINTER (RO)
      BEQ  .+6          ;BRANCH IF OK
      HLT  .+6          ;STACK POINTER (RO) NOT EQUAL TO #STACK4
                        ;THE ERROR NUMBER IS 2

      TST  ANS1         ;CHECK FIRST HALF OF ANSWER
      BEQ  .+6          ;BRANCH IF OK
      HLT+2 ;ANS1 NOT EQUAL TO 000000
                        ;THE ERROR NUMBER IS 3

      TST  ANS2         ;CHECK SECOND HALF OF ANSWER
      BEQ  .+6          ;BRANCH IF OK
      HLT+2 ;ANS2 NOT EQUAL TO 000000
                        ;THE ERROR NUMBER IS 4

      CMPB #1,   $TESTN ;CHECK THE TEST NUMBER
      BEQ  .+6          ;BRANCH IF OK
      HLT  .+6          ;WRONG TEST! PC MUST HAVE FOULED UP.
                        ;THE ERROR NUMBER IS 5
```

000001 177336 END1:

279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324

```

*****
:TEST 2:      FADD (LSI-11 FLOATING ADD INSTRUCTION)
:      000000,000000 + 152525,052524 = 152525,052524
:      PS = 010,      STACK POINTER = R3
*****
    
```

```

TST2:  SCOPE
        JSR      R5,      PUSHR      ;PUSH 4 WORDS ONTO R3 STACK, SET PRIORITY
        .WORD    152525,052524      ;SECOND OPERAND ON TOP
        .WORD    000000,000000      ;FIRST OPERAND ON BOTTOM
        .WORD    040                ;PROCESSOR PRIORITY LEVEL
        .WORD    TRAPER,340          ;FIS TRAP VECTOR
        MOV      #STACK0,R3         ;SET UP STACK POINTER

        NOP

        FADD     R3                ;FLOATING ADD ON THE R3 STACK

        JSR      PC,      POPR       ;POP THE ANSWER
        MOV      R3,      $SP        ;SAVE "STACK POINTER"
        CMPB    #010,     $PSW       ;CHECK PS (EXCEPT T BIT)
        BEQ     .+6                ;BRANCH IF OK
        HLT     6                  ;PS NOT EQUAL TO 010
        ;THE ERROR NUMBER IS 6

        CMP      #STACK4,$SP        ;CHECK THE STACK POINTER (R3)
        BEQ     .+6                ;BRANCH IF OK
        HLT     7                  ;STACK POINTER (R3) NOT EQUAL TO #STACK4
        ;THE ERROR NUMBER IS 7

        CMP      #152525,ANS1       ;CHECK FIRST HALF OF ANSWER
        BEQ     .+6                ;BRANCH IF OK
        HLT+2   10                 ;ANS1 NOT EQUAL TO 152525
        ;THE ERROR NUMBER IS 10

        CMP      #052524,ANS2       ;CHECK SECOND HALF OF ANSWER
        BEQ     .+6                ;BRANCH IF OK
        HLT+2   11                 ;ANS2 NOT EQUAL TO 052524
        ;THE ERROR NUMBER IS 11

        CMPB    #2,        $TESTN    ;CHECK THE TEST NUMBER
        BEQ     .+6                ;BRANCH IF OK
        HLT     12                 ;WRONG TEST! PC MUST HAVE FOULED UP.
        ;THE ERROR NUMBER IS 12
    
```



```

371
372
373
374
375
376
377
378 001350 104400
379 001352 004567 014614
380 001356 177777 177777
381 001362 077777 177777
382 001366 000100
383 001370 016442 000340
384 001374 012702 000510
385
386 001400 000240
387 001402 075002
388
389 001404 004767 014614
390 001410 010267 177020
391 001414 122767 000004 177010
392 001422 001402
393 001424 104000
394 001426 000020
395
396 001430 022767 000514 176776
397 001436 001402
398 001440 104000
399 001442 000021
400
401 001444 005767 176766
402 001450 001402
403 001452 104002
404 001454 000022
405
406 001456 005767 176756
407 001462 001402
408 001464 104002
409 001466 000023
410
411 001470 122767 000004 176706 END4:
412 001476 001402
413 001500 104000
414 001502 000024
415
416

```

```

*****
TEST 4: FADD (LSI-11 FLOATING ADD INSTRUCTION)
077777,177777 + 177777,177777 = 000000,000000
PS = 004, STACK POINTER = R2
*****

TST4: SCOPE
JSR R5, PUSHR ; PUSH 4 WORDS ONTO R2 STACK, SET PRIORITY
.WORD 177777,177777 ; SECOND OPERAND ON TOP
.WORD 077777,177777 ; FIRST OPERAND ON BOTTOM
.WORD 100 ; PROCESSOR PRIORITY LEVEL
.WORD TRAPER,340 ; FIS TRAP VECTOR
MOV #STACK0,R2 ; SET UP STACK POINTER

NOP
FADD R2 ; FLOATING ADD ON THE R2 STACK

JSR PC, POPR ; POP THE ANSWER
MOV R2, SSP ; SAVE "STACK POINTER"
CMPB #004, SPSW ; CHECK PS (EXCEPT T BIT)
BEQ .+6 ; BRANCH IF OK
HLT ; PS NOT EQUAL TO 004
20 ; THE ERROR NUMBER IS 20

CMP #STACK4,SSP ; CHECK THE STACK POINTER (R2)
BEQ .+6 ; BRANCH IF OK
HLT ; STACK POINTER (R2) NOT EQUAL TO #STACK4
21 ; THE ERROR NUMBER IS 21

TST ANS1 ; CHECK FIRST HALF OF ANSWER
BEQ .+6 ; BRANCH IF OK
HLT+2 ; ANS1 NOT EQUAL TO 000000
22 ; THE ERROR NUMBER IS 22

TST ANS2 ; CHECK SECOND HALF OF ANSWER
BEQ .+6 ; BRANCH IF OK
HLT+2 ; ANS2 NOT EQUAL TO 000000
23 ; THE ERROR NUMBER IS 23

CMPB #4, STESTN ; CHECK THE TEST NUMBER
BEQ .+6 ; BRANCH IF OK
HLT ; WRONG TEST! PC MUST HAVE FOULED UP.
24 ; THE ERROR NUMBER IS 24

```

```

417
418
419
420
421
422
423
424 001504 104400          SCOPE
425 001506 004567 014306  TST5:  JSR      RS      PUSH5  :PUSH 4 WORDS ONTO STACK, SET PRIORITY
426 001512 152525 052524  .WORD  152525,052524  ;SECOND OPERAND ON TOP
427 001516 052525 052525  .WORD  052525,052525  ;FIRST OPERAND ON BOTTOM
428 001522 000217          .WORD  217             ;PROCESSOR PRIORITY LEVEL
429 001524 016442 000340  .WORD  TRAPER,340     ;FIS TRAP VECTOR
430
431 001530 000240          NOP
432 001532 075006          FADD   SP              ;FLOATING ADD ON THE STACK
433
434 001534 004767 014320  JSR    PC,      POPS   ;POP THE ANSWER
435 001540 022706 000600  CMP    #BEGIN, SP    ;CHECK THE STACK POINTER
436 001544 001405          BEQ    TSAS          ;BRANCH IF OK
437 001546 012706 000600  MOV    #BEGIN, SP    ;RESTORE STACK POINTER
438 001552 104000          HLT                    ;STACK POINTER FOULED UP
439 001554 000025          25                     ;THE ERROR NUMBER IS 25
440 001556 000421          BR      ENDS          ;SKIP REST OF TEST
441
442 001560 122767 000200 176644 TSAS:  CMPB   #200,   SPSW   ;CHECK PS (EXCEPT T BIT)
443 001566 001402          BEQ    .+6            ;BRANCH IF OK
444 001570 104000          HLT                    ;PS NOT EQUAL TO 200
445 001572 000026          26                     ;THE ERROR NUMBER IS 26
446
447 001574 022767 044600 176634  CMP    #044600,ANS1  ;CHECK FIRST HALF OF ANSWER
448 001602 001402          BEQ    .+6            ;BRANCH IF OK
449 001604 104002          HLT+2                 ;ANS1 NOT EQUAL TO 044600
450 001606 000027          27                     ;THE ERROR NUMBER IS 27
451
452 001610 005767 176624          TST    ANS2          ;CHECK SECOND HALF OF ANSWER
453 001614 001402          BEQ    .+6            ;BRANCH IF OK
454 001616 104002          HLT+2                 ;ANS2 NOT EQUAL TO 000000
455 001620 000030          30                     ;THE ERROR NUMBER IS 30
456
457 001622 122767 000005 176554 ENDS:  CMPB   #5,     STESTN ;CHECK THE TEST NUMBER
458 001630 001402          BEQ    .+6            ;BRANCH IF OK
459 001632 104000          HLT                    ;WRONG TEST! PC MUST HAVE FOULED UP.
460 001634 000031          31                     ;THE ERROR NUMBER IS 31
461
462

```

```

*****
:TEST 5:      FADD (LSI-11 FLOATING ADD INSTRUCTION)
:      052525,052525 + 152525,052524 = 044600,000000
:      PS = 200,      STACK POINTER = SP
*****

```

463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508

```

*****
TEST 6:      FADD (LSI-11 FLOATING ADD INSTRUCTION)
             125200,000000 + 025177,177777 = 117200,000000
             PS = 210,          STACK POINTER = SP
*****
    
```

```

001636 104400          SCOPE
001640 004567 014154   TST6:  JSR      RS,          PUSHS   ;PUSH 4 WORDS ONTO STACK, SET PRIORITY
001644 025177 177777   .WORD  025177,177777   ;SECOND OPERAND ON TOP
001650 125200 000000   .WORD  125200,000000   ;FIRST OPERAND ON BOTTOM
001654 000307          .WORD  307             ;PROCESSOR PRIORITY LEVEL
001656 016442 000340   .WORD  TRAPER,340      ;FIS TRAP VECTOR

001662 000240          NOP
001664 075006          FADD   SP              ;FLOATING ADD ON THE STACK

001666 004767 014166   JSR    PC,          POPS   ;POP THE ANSWER
001672 022706 000600   CMP    #BEGIN, SP    ;CHECK THE STACK POINTER
001676 001405          BEQ    TSA6          ;BRANCH IF OK
001700 012706 000600   MOV    #BEGIN, SP    ;RESTORE STACK POINTER
001704 104000          HLT                    ;STACK POINTER FOULED UP
001706 000032          32                   ;THE ERROR NUMBER IS 32
001710 000421          BR     END6          ;SKIP REST OF TEST

001712 122767 000210 176512 TSA6:  CMPB   #210,   SPSW    ;CHECK PS (EXCEPT T BIT)
001720 001402          BEQ    .+6          ;BRANCH IF OK
001722 104000          HLT                    ;PS NOT EQUAL TO 210
001724 000033          33                   ;THE ERROR NUMBER IS 33

001726 022767 117200 176502   CMP    #117200,ANS1   ;CHECK FIRST HALF OF ANSWER
001734 001402          BEQ    .+6          ;BRANCH IF OK
001736 104002          HLT+2                ;ANS1 NOT EQUAL TO 117200
001740 000034          34                   ;THE ERROR NUMBER IS 34

001742 005767 176472          TST    ANS2          ;CHECK SECOND HALF OF ANSWER
001746 001402          BEQ    .+6          ;BRANCH IF OK
001750 104002          HLT+2                ;ANS2 NOT EQUAL TO 000000
001752 000035          35                   ;THE ERROR NUMBER IS 35

001754 122767 000006 176422 END6:  CMPB   #6,      $TESTN ;CHECK THE TEST NUMBER
001762 001402          BEQ    .+6          ;BRANCH IF OK
001764 104000          HLT                    ;WRONG TEST! PC MUST HAVE FOULED UP.
001766 000036          36                   ;THE ERROR NUMBER IS 36
    
```

509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554

```

*****
TEST 7: FADD (LSI-11 FLOATING ADD INSTRUCTION)
        135753,024642 + 100125,052525 = 135753,024642
        PS = 210, STACK POINTER = R5
*****
    
```

```

TST7:  SCOPE
        JSR      R5,      PUSHR   ; PUSH 4 WORDS ONTO R5 STACK, SET PRIORITY
        .WORD   100125,052525    ; SECOND OPERAND ON TOP
        .WORD   135753,024642    ; FIRST OPERAND ON BOTTOM
        .WORD   347              ; PROCESSOR PRIORITY LEVEL
        .WORD   TRAPER,340       ; FIS TRAP VECTOR
        MOV     #STACK0,R5       ; SET UP STACK POINTER

        NOP
        FADD   R5                ; FLOATING ADD ON THE R5 STACK

        JSR   PC,      POPR      ; POP THE ANSWER
        MOV  R5,      SSP       ; SAVE "STACK POINTER"
        CMPB #210,    SPSW      ; CHECK PS (EXCEPT T BIT)
        BEQ  .+6              ; BRANCH IF OK
        HLT  .+6              ; PS NOT EQUAL TO 210
        HLT  37              ; THE ERROR NUMBER IS 37

        CMP   #STACK4,SSP      ; CHECK THE STACK POINTER (R5)
        BEQ  .+6              ; BRANCH IF OK
        HLT  40              ; STACK POINTER (R5) NOT EQUAL TO #STACK4
        HLT  40              ; THE ERROR NUMBER IS 40

        CMP   #135753,ANS1     ; CHECK FIRST HALF OF ANSWER
        BEQ  .+6              ; BRANCH IF OK
        HLT+2 41              ; ANS1 NOT EQUAL TO 135753
        HLT  41              ; THE ERROR NUMBER IS 41

        CMP   #024642,ANS2     ; CHECK SECOND HALF OF ANSWER
        BEQ  .+6              ; BRANCH IF OK
        HLT+2 42              ; ANS2 NOT EQUAL TO 024642
        HLT  42              ; THE ERROR NUMBER IS 42

        CMPB #7,      STESTN    ; CHECK THE TEST NUMBER
        BEQ  .+6              ; BRANCH IF OK
        HLT  43              ; WRONG TEST! PC MUST HAVE FOULED UP.
        HLT  43              ; THE ERROR NUMBER IS 43
    
```


601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646

```

*****
:TEST 11: FADD (LSI-11 FLOATING ADD INSTRUCTION)
:      100400,000000 + 000200,000000 = 100200,000000
:      PS = 010,      STACK POINTER = R5
*****
    
```

```

TST11: SCOPE
        JSR      R5,      PUSHR      ;PUSH 4 WORDS ONTO R5 STACK, SET PRIORITY
        .WORD   000200,000000      ;SECOND OPERAND ON TOP
        .WORD   100400,000000      ;FIRST OPERAND ON BOTTOM
        .WORD   140                ;PROCESSOR PRIORITY LEVEL
        .WORD   TRAPER,340         ;FIS TRAP VECTOR
        MOV      #STACK0,R5        ;SET UP STACK POINTER

        NOP
        FADD    R5                ;FLOATING ADD ON THE R5 STACK

        JSR      PC,      POPR       ;POP THE ANSWER
        MOV      R5,      SSP        ;SAVE "STACK POINTER"
        CMPB    #010,      SPSW      ;CHECK PS (EXCEPT T BIT)
        BEQ     .+6                ;BRANCH IF OK
        HLT     .+6                ;PS NOT EQUAL TO 010
        HLT     51                ;THE ERROR NUMBER IS 51

        CMP     #STACK4,SSP         ;CHECK THE STACK POINTER (R5)
        BEQ     .+6                ;BRANCH IF OK
        HLT     .+6                ;STACK POINTER (R5) NOT EQUAL TO #STACK4
        HLT     52                ;THE ERROR NUMBER IS 52

        CMP     #100200,ANS1        ;CHECK FIRST HALF OF ANSWER
        BEQ     .+6                ;BRANCH IF OK
        HLT+2   .+6                ;ANS1 NOT EQUAL TO 100200
        HLT     53                ;THE ERROR NUMBER IS 53

        TST     ANS2                ;CHECK SECOND HALF OF ANSWER
        BEQ     .+6                ;BRANCH IF OK
        HLT+2   .+6                ;ANS2 NOT EQUAL TO 000000
        HLT     54                ;THE ERROR NUMBER IS 54

        CMPB    #11,      STSTN      ;CHECK THE TEST NUMBER
        BEQ     .+6                ;BRANCH IF OK
        HLT     .+6                ;WRONG TEST! PC MUST HAVE FOULED UP.
        HLT     55                ;THE ERROR NUMBER IS 55
    
```

647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692

```

*****
:TEST 12:      FADD (LSI-11 FLOATING ADD INSTRUCTION)
:              000425,052525 + 100252,125252 = 000200,000000
:              PS = 200,      STACK POINTER = R4
*****
    
```

```

TST12:  SCOPE
        JSR      R5,      PUSHR      ;PUSH 4 WORDS ONTO R4 STACK, SET PRIORITY
        .WORD   100252,125252      ;SECOND OPERAND ON TOP
        .WORD   000425,052525      ;FIRST OPERAND ON BOTTOM
        .WORD   217                ;PROCESSOR PRIORITY LEVEL
        .WORD   TRAPER,340         ;FIS TRAP VECTOR
        MOV     #STACK0,R4        ;SET UP STACK POINTER

        NOP
        FADD   R4                ;FLOATING ADD ON THE R4 STACK

        JSR      PC,      POPR      ;POP THE ANSWER
        MOV     R4,      SSP      ;SAVE "STACK POINTER"
        CMPB   #200,     SPSW     ;CHECK PS (EXCEPT T BIT)
        BEQ    .+6              ;BRANCH IF OK
        HLT    .+6              ;PS NOT EQUAL TO 200
        HLT    56              ;THE ERROR NUMBER IS 56

        CMP     #STACK4,$SP      ;CHECK THE STACK POINTER (R4)
        BEQ    .+6              ;BRANCH IF OK
        HLT    .+6              ;STACK POINTER (R4) NOT EQUAL TO #STACK4
        HLT    57              ;THE ERROR NUMBER IS 57

        CMP     #000200,ANS1     ;CHECK FIRST HALF OF ANSWER
        BEQ    .+6              ;BRANCH IF OK
        HLT+2  .+6              ;ANS1 NOT EQUAL TO 000200
        HLT    60              ;THE ERROR NUMBER IS 60

        TST    ANS2             ;CHECK SECOND HALF OF ANSWER
        BEQ    .+6              ;BRANCH IF OK
        HLT+2  .+6              ;ANS2 NOT EQUAL TO 000000
        HLT    61              ;THE ERROR NUMBER IS 61

        CMPB   #12,      STSTN    ;CHECK THE TEST NUMBER
        BEQ    .+6              ;BRANCH IF OK
        HLT    .+6              ;WRONG TEST! PC MUST HAVE FOULED UP.
        HLT    62              ;THE ERROR NUMBER IS 62
    
```

00012 175626 END12:

693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738

002564 104400
002566 004567 013226
002572 000252 125252
002576 100425 052525
002602 000307
002604 016442 000340

002610 000240
002612 075006

002614 004767 013240
002620 022706 000600
002624 001405
002626 012706 000600
002632 104000
002634 000063
002636 000421

002640 122767 000210 175564 TSA13:
002646 001402
002650 104000
002652 000064

002654 022767 100200 175554
002662 001402
002664 104002
002666 000065

002670 005767 175544
002674 001402
002676 104002
002700 000066

002702 122767 000013 175474 END13:
002710 001402
002712 104000
002714 000067

```
*****
TEST 13: FADD (LSI-11 FLOATING ADD INSTRUCTION)
          100425,052525 + 000252,125252 = 100200,000000
          PS = 210, STACK POINTER = SP
*****

SCOPE
TST13: JSR RS PUSH5 ;PUSH 4 WORDS ONTO STACK, SET PRIORITY
        .WORD 000252,125252 ;SECOND OPERAND ON TOP
        .WORD 100425,052525 ;FIRST OPERAND ON BOTTOM
        .WORD 307 ;PROCESSOR PRIORITY LEVEL
        .WORD TRAPER,340 ;FIS TRAP VECTOR

NOP
FADD SP ;FLOATING ADD ON THE STACK

JSR PC, POPS ;POP THE ANSWER
CMP #BEGIN, SP ;CHECK THE STACK POINTER
BEQ TSA13 ;BRANCH IF OK
MOV #BEGIN, SP ;RESTORE STACK POINTER
HLT ;STACK POINTER FOULED UP
63 ;THE ERROR NUMBER IS 63
BR END13 ;SKIP REST OF TEST

CMPB #210, SPSW ;CHECK PS (EXCEPT T BIT)
BEQ .+6 ;BRANCH IF OK
HLT ;PS NOT EQUAL TO 210
64 ;THE ERROR NUMBER IS 64

CMP #100200,ANS1 ;CHECK FIRST HALF OF ANSWER
BEQ .+6 ;BRANCH IF OK
HLT+2 ;ANS1 NOT EQUAL TO 100200
65 ;THE ERROR NUMBER IS 65

TST ANS2 ;CHECK SECOND HALF OF ANSWER
BEQ .+6 ;BRANCH IF OK
HLT+2 ;ANS2 NOT EQUAL TO 000000
66 ;THE ERROR NUMBER IS 66

CMPB #13, $TESTN ;CHECK THE TEST NUMBER
BEQ .+6 ;BRANCH IF OK
HLT ;WRONG TEST! PC MUST HAVE FOULED UP.
67 ;THE ERROR NUMBER IS 67
```

739
 740
 741
 742
 743
 744
 745
 746
 747
 748
 749
 750
 751
 752
 753
 754
 755
 756
 757
 758
 759
 760
 761
 762
 763
 764
 765
 766
 767
 768
 769
 770
 771
 772
 773
 774
 775
 776
 777
 778
 779
 780
 781
 782
 783
 784

```

*****
TEST 14:      FADD (LSI-11 FLOATING ADD INSTRUCTION)
              077652,125252 + 077452,125252 = 077777,177777
              PS = 200,      STACK POINTER = SP
*****
SCOPE
TST14: JSR      RS,      PUSH4      :PUSH 4 WORDS ONTO STACK, SET PRIORITY
        .WORD   077452,125252 :SECOND OPERAND ON TOP
        .WORD   077652,125252 :FIRST OPERAND ON BOTTOM
        .WORD   257           :PROCESSOR PRIORITY LEVEL
        .WORD   TRAPER,340    :FIS TRAP VECTOR

NOP
FADD    SP              :FLOATING ADD ON THE STACK

JSR     PC,      POPS   :POP THE ANSWER
CMP     #BEGIN, SP    :CHECK THE STACK POINTER
BEQ     TSA14      :BRANCH IF OK
MOV     #BEGIN, SP   :RESTORE STACK POINTER
HLT     70          :STACK POINTER FOULED UP
        70          :THE ERROR NUMBER IS 70
BR      END14      :SKIP REST OF TEST

        000200 175432 TSA14: CMPB   #200,   SPSW   :CHECK PS (EXCEPT T BIT)
        003000 001402      BEQ     .+6     :BRANCH IF OK
        003002 104000      HLT     71       :PS NOT EQUAL TO 200
        003004 000071      71       :THE ERROR NUMBER IS 71

        003006 022767 077777 175422 CMPB   #077777,ANS1 :CHECK FIRST HALF OF ANSWER
        003014 001402      BEQ     .+6     :BRANCH IF OK
        003016 104002      HLT+2   72       :ANS1 NOT EQUAL TO 077777
        003020 000072      72       :THE ERROR NUMBER IS 72

        003022 022767 177777 175410 CMPB   #177777,ANS2 :CHECK SECOND HALF OF ANSWER
        003030 001402      BEQ     .+6     :BRANCH IF OK
        003032 104002      HLT+2   73       :ANS2 NOT EQUAL TO 177777
        003034 000073      73       :THE ERROR NUMBER IS 73

        003036 122767 000014 175340 END14: CMPB   #14,    $TESTN :CHECK THE TEST NUMBER
        003044 001402      BEQ     .+6     :BRANCH IF OK
        003046 104000      HLT     74       :WRONG TEST! PC MUST HAVE FOULED UP.
        003050 000074      74       :THE ERROR NUMBER IS 74
  
```

785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830

```
*****
TEST 15:      FADD (LSI-11 FLOATING ADD INSTRUCTION)
              177452,125252 + 177652,125252 = 177777,177777
              PS = 210,      STACK POINTER = R4
*****
```

```
TST15:  SCOPE
        JSR      R5,      PUSHR      ; PUSH 4 WORDS ONTO R4 STACK, SET PRIORITY
        .WORD   177652,125252      ; SECOND OPERAND ON TOP
        .WORD   177452,125252      ; FIRST OPERAND ON BOTTOM
        .WORD   357                ; PROCESSOR PRIORITY LEVEL
        .WORD   TRAPER,340         ; FIS TRAP VECTOR
        MOV      #STACK0,R4       ; SET UP STACK POINTER

        NOP
        FADD     R4                ; FLOATING ADD ON THE R4 STACK

        JSR      PC,      POPR      ; POP THE ANSWER
        MOV      R4,      SSP      ; SAVE "STACK POINTER"
        CMPB    #210,     SPSW     ; CHECK PS (EXCEPT T BIT)
        BEQ     .+6             ; BRANCH IF OK
        HLT     75              ; PS NOT EQUAL TO 210
        75                ; THE ERROR NUMBER IS 75

        CMP     #STACK4,SSP      ; CHECK THE STACK POINTER (R4)
        BEQ     .+6             ; BRANCH IF OK
        HLT     76              ; STACK POINTER (R4) NOT EQUAL TO #STACK4
        76                ; THE ERROR NUMBER IS 76

        CMP     #177777,ANS1     ; CHECK FIRST HALF OF ANSWER
        BEQ     .+6             ; BRANCH IF OK
        HLT+2   77              ; ANS1 NOT EQUAL TO 177777
        77                ; THE ERROR NUMBER IS 77

        CMP     #177777,ANS2     ; CHECK SECOND HALF OF ANSWER
        BEQ     .+6             ; BRANCH IF OK
        HLT+2   100             ; ANS2 NOT EQUAL TO 177777
        100                ; THE ERROR NUMBER IS 100

        CMPB    #15,      $TESTN   ; CHECK THE TEST NUMBER
        BEQ     .+6             ; BRANCH IF OK
        HLT     101             ; WRONG TEST! PC MUST HAVE FOULED UP.
        101                ; THE ERROR NUMBER IS 101
```

```

831
832
833 :*****
834 :TEST 16: FADD (LSI-11 FLOATING ADD INSTRUCTION)
835 : 004000,105004 + 104000,104000 = 000401,000000
836 : PS = 000, STACK POINTER = PC
837 :*****
838 003212 104400
839 003214 004567 013124 TST16: SCOPE JSR RS, PUSH7 ; PUSH 4 WORDS ONTO STACK, SET PRIORITY
840 003220 003244 .WORD STK16 ; TOP OF STACK
841 003222 104000 104000 .WORD 104000,104000 ; SECOND OPERAND ON TOP
842 003226 004000 105004 .WORD 004000,105004 ; FIRST OPERAND ON BOTTOM
843 003232 000144 .WORD 144 ; PROCESSOR PRIORITY LEVEL
844 003234 016442 000340 .WORD TRAPER,340 ; FIS TRAP VECTOR
845
846 003240 000240 NOP
847 003242 075007 FADD PC ; FLOATING ADD ON FOLLOWING 4 WORDS
848 003244 104000 STK16 ; SHOULD CONTAIN 104000
849 003246 104000 104000 ; SHOULD CONTAIN 104000
850 003250 004000 004000 ; BEFORE FADD, 004000; AFTER, 000401
851 003252 105004 105004 ; BEFORE FADD, 105004; AFTER, 000000
852
853 003254 004767 013114 JSR PC, POP7 ; POP THE ANSWER
854 003260 105767 175146 TSTB SPSW ; CHECK PS (EXCEPT T BIT)
855 003264 001402 BEQ .+6 ; BRANCH IF OK
856 003266 104000 HLT ; PS NOT EQUAL TO 000
857 003270 000102 102 ; THE ERROR NUMBER IS 102
858
859 003272 022767 104000 175136 CMP #104000,ANS1 ; CHECK FIRST HALF OF INPUT DATA (STK16)
860 003300 001402 BEQ .+6 ; BRANCH IF OK
861 003302 104002 HLT+2 ; ANS1 NOT EQUAL TO 104000
862 003304 000103 103 ; THE ERROR NUMBER IS 103
863
864 003306 022767 104000 175124 CMP #104000,ANS2 ; CHECK SECOND HALF OF INPUT DATA (STK16+2)
865 003314 001402 BEQ .+6 ; BRANCH IF OK
866 003316 104002 HLT+2 ; ANS2 NOT EQUAL TO 104000
867 003320 000104 104 ; THE ERROR NUMBER IS 104
868
869 003322 022767 000401 175112 CMP #000401,ANS3 ; CHECK FIRST HALF OF ANSWER
870 003330 001402 BEQ .+6 ; BRANCH IF OK
871 003332 104004 HLT+4 ; ANS3 NOT EQUAL TO 000401
872 003334 000105 105 ; THE ERROR NUMBER IS 105
873
874 003336 005767 175102 TST ANS4 ; CHECK SECOND HALF OF ANSWER
875 003342 001402 BEQ .+6 ; BRANCH IF OK
876 003344 104004 HLT+4 ; ANS4 NOT EQUAL TO 000000
877 003346 000106 106 ; THE ERROR NUMBER IS 106
878
879 003350 122767 000016 175026 END16: CMPB #16, $TESTN ; CHECK THE TEST NUMBER
880 003356 001402 BEQ .+6 ; BRANCH IF OK
881 003360 104000 HLT ; WRONG TEST! PC MUST HAVE FOULED UP.
882 003362 000107 107 ; THE ERROR NUMBER IS 107
883
884
885
886 :*****

```

F03

DVKACB MACY11 27(1006) 17-JAN-77 15:14 PAGE 22
 DVKACB.P11 13-JAN-77 15:28

SEQ 0030

```

:TEST 17:      FADD (LSI-11 FLOATING ADD INSTRUCTION)
:              100200,000000 + 000377,177777 ==> UNDERFLOW
:              PS(ON STACK) = 012,      STACK POINTER = R3
:*****
    
```

887									
888									
889									
890									
891									
892	003364	104400							
893	003366	004567	012600	TST17:	JSR	R5,	PUSHR		; PUSH 4 WORDS ONTO R3 STACK, SET PRIORITY
894	003372	000377	177777		.WORD	000377,177777			; SECOND OPERAND ON TOP
895	003376	100200	000000		.WORD	100200,000000			; FIRST OPERAND ON BOTTOM
896	003402	000157			.WORD	157			; PROCESSOR PRIORITY LEVEL
897	003404	003436	000000		.WORD	ISR17, 000			; FIS TRAP VECTOR
898	003410	012703	000510		MOV	#STACK0,R3			; SET UP R3 AS STACK POINTER
899									
900	003414	000240			NOP				
901	003416	075003			FADD	R3			; FLOATING ADD ON THE R3 STACK
902									
903	003420	004767	012600	RTA17:	JSR	PC,	POPR		; POP THE "ANSWER"
904	003424	010367	175004		MOV	R3,	\$\$P		; SAVE STACK POINTER (R3)
905	003430	104002			HLT+2				; FIS TRAP DIDN'T OCCURE!
906	003432	000110			110				; THE ERROR NUMBER IS 110
907	003434	000462			BR	END17			
908									
909	003436	004767	012612	ISR17:	JSR	PC,	POPER		; POP ALL DATA OFF THE STACKS
910	003442	010367	174766		MOV	R3,	\$\$P		; SAVE STACK POINTER (R3)
911	003446	105767	174760		TSTB	\$PSW			; CHECK PS AFTER FIS TRAP
912	003452	001402			BEQ	+.6			; BRANCH IF OK
913	003454	104000			HLT				; PS AFTER FIS TRAP NOT EQUAL TO 000
914	003456	000111			111				; THE ERROR NUMBER IS 111
915									
916	003460	022767	000510 174746		CMP	#STACK0,\$\$P			; CHECK THE STACK POINTER (R3)
917	003466	001402			BEQ	+.6			; BRANCH IF OK
918	003470	104000			HLT				; STACK POINTER (R3) NOT EQUAL TO #STACK0
919	003472	000112			112				; THE ERROR NUMBER IS 112
920									
921	003474	022767	003420 174734		CMP	#RTA17,ANS1			; CHECK FIS TRAP RETURN ADDRESS
922	003502	001402			BEQ	+.6			; BRANCH IF OK
923	003504	104001			HLT+1				; FIS TRAP AT WRONG ADDRESS
924	003506	000113			113				; THE ERROR NUMBER IS 113
925									
926	003510	022767	000012 174722		CMP	#012,ANS2			; CHECK PS BEFORE FIS TRAP
927	003516	001402			BEQ	+.6			; BRANCH IF OK
928	003520	104002			HLT+2				; PS AT FIS TRAP TIME NOT 012
929	003522	000114			114				; THE ERROR NUMBER IS 114
930									
931	003524	022767	000377 174710		CMP	#000377,ANS3			; CHECK DATA FROM THE STACK
932	003532	001402			BEQ	+.6			; BRANCH IF OK
933	003534	104004			HLT+4				; DATA ON STACK (000377) CHANGED
934	003536	000115			115				; THE ERROR NUMBER IS 115
935									
936	003540	022767	177777 174676		CMP	#177777,ANS4			; CHECK DATA FROM STACK
937	003546	001402			BEQ	+.6			; BRANCH IF OK
938	003550	104004			HLT+4				; DATA ON STACK (177777) CHANGED
939	003552	000116			116				; THE ERROR NUMBER IS 116
940									
941	003554	022767	100200 174664		CMP	#100200,ANS5			; CHECK DATA FROM STACK
942	003562	001402			BEQ	+.6			; BRANCH IF OK

TEST FLOATING ADD INSTRUCTION WITH OVERFLOW

1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084

```

*****
:TEST 21:      FADD (LSI-11 FLOATING ADD INSTRUCTION)
:              177452,125253 + 177652,125252 ==> OVERFLOW
:              PS(ON STACK) = 002,      STACK POINTER = R1
*****
    
```

```

TST21:  JSR      R5,      PUSHR      ;PUSH 4 WORDS ONTO R1 STACK, SET PRIORITY
         .WORD   177652,125252      ;SECOND OPERAND ON TOP
         .WORD   177452,125253      ;FIRST OPERAND ON BOTTOM
         .WORD   105                ;PROCESSOR PRIORITY LEVEL
         .WORD   ISR21, 252         ;FIS TRAP VECTOR
         MOV     #STACK0,R1        ;SET UP R1 AS STACK POINTER

         NOP
         FADD   R1                ;FLOATING ADD ON THE R1 STACK

RTA21:  JSR      PC,      POPR       ;POP THE "ANSWER"
         MOV     R1,      SSP       ;SAVE STACK POINTER (R1)
         HLT+2   134              ;FIS TRAP DIDN'T OCCURE!
         BR     END21             ;THE ERROR NUMBER IS 134

ISR21:  JSR      PC,      POPER     ;POP ALL DATA OFF THE STACKS
         MOV     R1,      SSP       ;SAVE STACK POINTER (R1)
         CMPB   #252,      SPSW     ;CHECK PS AFTER FIS TRAP
         BEQ    .+6               ;BRANCH IF OK
         HLT    135               ;PS AFTER FIS TRAP NOT EQUAL TO 252
         HLT    135               ;THE ERROR NUMBER IS 135

         CMP    #STACK0,SSP       ;CHECK THE STACK POINTER (R1)
         BEQ    .+6               ;BRANCH IF OK
         HLT    136               ;STACK POINTER (R1) NOT EQUAL TO #STACK0
         HLT    136               ;THE ERROR NUMBER IS 136

         CMP    #RTA21, ANS1      ;CHECK FIS TRAP RETURN ADDRESS
         BEQ    .+6               ;BRANCH IF OK
         HLT+1  137               ;FIS TRAP AT WRONG ADDRESS
         HLT    137               ;THE ERROR NUMBER IS 137

         CMP    #002,      ANS2     ;CHECK PS BEFORE FIS TRAP
         BEQ    .+6               ;BRANCH IF OK
         HLT+2  140               ;PS AT FIS TRAP TIME NOT 002
         HLT    140               ;THE ERROR NUMBER IS 140

         CMP    #177652,ANS3      ;CHECK DATA FROM THE STACK
         BEQ    .+6               ;BRANCH IF OK
         HLT+4  141               ;DATA ON STACK (177652) CHANGED
         HLT    141               ;THE ERROR NUMBER IS 141

         CMP    #125252,ANS4      ;CHECK DATA FROM STACK
         BEQ    .+6               ;BRANCH IF OK
         HLT+4  142               ;DATA ON STACK (125252) CHANGED
         HLT    142               ;THE ERROR NUMBER IS 142
    
```


TEST FLOATING ADD INSTRUCTION WITH OVERFLOW

```

1101
1102
1103 :*****
1104 :TEST 22:      FADD (LSI-11 FLOATING ADD INSTRUCTION)
1105 :             077652,125253 + 077452,125252 ==> OVERFLOW
1106 :             PS(ON STACK) = 002,      STACK POINTER = SP
1107 :*****
1108 004304 104400
1109 004306 004567 011506      TST22:  SCOPE
1110 004312 077452 125252      JSR    R5,      PUSH5      ;PUSH 4 WORDS ONTO STACK, SET PRIORITY
1111 004316 077652 125253      .WORD 077452,125252      ;SECOND OPERAND ON TOP
1112 004322 000003      .WORD 077652,125253      ;FIRST OPERAND ON BOTTOM
1113 004324 004352 000344      .WORD 003                ;PROCESSOR PRIORITY LEVEL
1114      .WORD ISR22, 344      ;FIS TRAP VECTOR
1115 004330 000240
1116 004332 075006      NOP
1117      FADD    SP          ;FLOATING ADD ON THE STACK
1118 004334 004767 011520      RTA22:  JSR    PC,      POPS      ;POP THE "ANSWER"
1119 004340 104002      HLT+2      ;FIS TRAP DIDN'T OCCURE!
1120 004342 000146      146        ;THE ERROR NUMBER IS 146
1121 004344 012706 000600      MOV    #BEGIN, SP      ;RESTORE THE STACK POINTER
1122 004350 000464      BR      END22
1123
1124 004352 004767 011534      ISR22:  JSR    PC,      POPES      ;POP ALL DATA OFF THE STACK
1125 004356 022706 000600      CMP    #BEGIN, SP      ;CHECK THE STACK POINTER
1126 004362 001405      BEQ    ISA22           ;BRANCH IF OK
1127 004364 012706 000600      MOV    #BEGIN, SP      ;RESTORE THE STACK POINTER
1128 004370 104000      HLT      ;STACK POINTER FOULED UP
1129 004372 000147      147        ;THE ERROR NUMBER IS 147
1130 004374 000452      BR      END22          ;SKIP REST OF TEST
1131
1132 004376 122767 000344 174026  ISA22:  CMPB   #344,   SPSW      ;CHECK PS AFTER FIS TRAP
1133 004404 001402      BEQ    .+6            ;BRANCH IF OK
1134 004406 104000      HLT      ;PS AFTER FIS TRAP NOT EQUAL TO 344
1135 004410 000150      150        ;THE ERROR NUMBER IS 150
1136
1137 004412 022767 004334 174016      CMP    #RTA22, ANS1    ;CHECK FIS TRAP RETURN ADDRESS
1138 004420 001402      BEQ    .+6            ;BRANCH IF OK
1139 004422 104001      HLT+1    ;FIS TRAP AT WRONG ADDRESS
1140 004424 000151      151        ;THE ERROR NUMBER IS 151
1141
1142 004426 022767 000002 174004      CMP    #002,   ANS2    ;CHECK PS BEFORE FIS TRAP
1143 004434 001402      BEQ    .+6            ;BRANCH IF OK
1144 004436 104002      HLT+2    ;PS AT FIS TRAP TIME NOT 002
1145 004440 000152      152        ;THE ERROR NUMBER IS 152
1146
1147 004442 022767 077452 173772      CMP    #077452,ANS3    ;CHECK DATA FROM THE STACK
1148 004450 001402      BEQ    .+6            ;BRANCH IF OK
1149 004452 104004      HLT+4    ;DATA ON STACK (077452) CHANGED
1150 004454 000153      153        ;THE ERROR NUMBER IS 153
1151
1152 004456 022767 125252 173760      CMP    #125252,ANS4    ;CHECK DATA FROM STACK
1153 004464 001402      BEQ    .+6            ;BRANCH IF OK
1154 004466 104004      HLT+4    ;DATA ON STACK (125252) CHANGED
1155 004470 000154      154        ;THE ERROR NUMBER IS 154
1156
    
```


1173
 1174
 1175
 1176
 1177
 1178
 1179
 1180
 1181
 1182
 1183
 1184
 1185
 1186
 1187
 1188
 1189
 1190
 1191
 1192
 1193
 1194
 1195
 1196
 1197
 1198
 1199
 1200
 1201
 1202
 1203
 1204
 1205
 1206
 1207
 1208
 1209
 1210
 1211
 1212
 1213
 1214
 1215
 1216
 1217
 1218

```

*****
TEST 23:      FSUB (LSI-11 FLOATING SUBTRACT INSTRUCTION)
              177520,017552 - 135352,051107 = 177520,017552
              PS = 010,      STACK POINTER = R1
*****
    
```

```

TST23:  SCOPE
        JSR      R5,      PUSHR      ;PUSH 4 WORDS ONTO R1 STACK, SET PRIORITY
        .WORD   135352,051107      ;SECOND OPERAND ON TOP
        .WORD   177520,017552      ;FIRST OPERAND ON BOTTOM
        .WORD   040                ;PROCESSOR PRIORITY LEVEL
        .WORD   TRAPER, 340         ;FIS TRAP VECTOR
        MOV      #STACK0,R1        ;SET UP STACK POINTER

        NOP
        FSUB     R1                ;FLOATING SUBTRACT ON THE R1 STACK

        JSR      PC,      POPR      ;POP THE ANSWER
        MOV      R1,      $SSP      ;SAVE "STACK POINTER"
        CMPB    #010,      $PSW     ;CHECK PS (EXCEPT T BIT)
        BEQ     .+6                ;BRANCH IF OK
        HLT     160                ;PS NOT EQUAL TO 010
        ;THE ERROR NUMBER IS 160

        CMP      #STACK4,$SSP      ;CHECK THE STACK POINTER (R1)
        BEQ     .+6                ;BRANCH IF OK
        HLT     161                ;STACK POINTER (R1) NOT EQUAL TO #STACK4
        ;THE ERROR NUMBER IS 161

        CMP      #177520,ANS1      ;CHECK FIRST HALF OF ANSWER
        BEQ     .+6                ;BRANCH IF OK
        HLT+2  162                ;ANS1 NOT EQUAL TO 177520
        ;THE ERROR NUMBER IS 162

        CMP      #017552,ANS2      ;CHECK SECOND HALF OF ANSWER
        BEQ     .+6                ;BRANCH IF OK
        HLT+2  163                ;ANS2 NOT EQUAL TO 017552
        ;THE ERROR NUMBER IS 163

        CMPB    #23,      $TESTN    ;CHECK THE TEST NUMBER
        BEQ     .+6                ;BRANCH IF OK
        HLT     164                ;WRONG TEST! PC MUST HAVE FOULED UP.
        ;THE ERROR NUMBER IS 164
    
```

```

004536 104400
004540 004567 011426
004544 135352 051107
004550 177520 017552
004554 000040
004556 016442 000340
004562 012701 000510

004566 000240
004570 075011

004572 004767 011426
004576 010167 173632
004602 122767 000010 173622
004610 001402
004612 104000
004614 000160

004616 022767 000514 173610
004624 001402
004626 104000
004630 000161

004632 022767 177520 173576
004640 001402
004642 104002
004644 000162

004646 022767 017552 173564
004654 001402
004656 104002
004660 000163

004662 122767 000023 173514 END23:
004670 001402
004672 104000
004674 000164
    
```



```

1265
1266
1267
1268
1269
1270
1271
1272 005032 104400
1273 005034 004567 010760
1274 005040 100177 177777
1275 005044 002460 123456
1276 005050 000015
1277 005052 016442 000340
1278
1279 005056 000240
1280 005060 075016
1281
1282 005062 004767 010772
1283 005066 022706 000600
1284 005072 001405
1285 005074 012706 000600
1286
1287
1288
1289
1290
1291 005100 104000
1292 005102 000172
1293 005104 000422
1294
1295 005106 122767 000000 173316 TSA25:
1296 005114 001402
1297 005116 104000
1298 005120 000173
1299
1300 005122 022767 002460 173306
1301 005130 001402
1302 005132 104002
1303 005134 000174
1304
1305 005136 022767 123456 173274
1306 005144 001402
1307 005146 104002
1308 005150 000175
1309
1310 005152 122767 000025 173224 END25:
1311 005160 001402
1312 005162 104000
1313 005164 000176
1314
1315

```

```

*****
:TEST 25: FSUB (LSI-11 FLOATING SUBTRACT INSTRUCTION)
: 002460,123456 - 100177,177777 = 002460,123456
: PS = 000, STACK POINTER = SP
*****
TST25: SCOPE
JSR R5, PUSH5 ;PUSH 4 WORDS ONTO STACK, SET PRIORITY
.WORD 100177,177777 ;SECOND OPERAND ON TOP
.WORD 002460,123456 ;FIRST OPERAND ON BOTTOM
.WORD 015 ;PROCESSOR PRIORITY LEVEL
.WORD TRAPER, 340 ;FIS TRAP VECTOR

NOP
FSUB SP ;FLOATING SUBTRACT ON THE STACK

JSR PC, POPS ;POP THE ANSWER
CMP #BEGIN, SP ;CHECK THE STACK POINTER
BEQ TSA25 ;BRANCH IF OK
MOV #BEGIN, SP ;RESTORE STACK POINTER

HLT ;STACK POINTER FOULED UP
172 ;THE ERROR NUMBER IS 172
BR END25 ;SKIP REST OF TEST

CMPB #000, SPSW ;CHECK PS (EXCEPT T BIT)
BEQ .+6 ;BRANCH IF OK
HLT ;PS NOT EQUAL TO 000
173 ;THE ERROR NUMBER IS 173

CMP #002460,ANS1 ;CHECK FIRST HALF OF ANSWER
BEQ .+6 ;BRANCH IF OK
HLT+2 ;ANS1 NOT EQUAL TO 002460
174 ;THE ERROR NUMBER IS 174

CMP #123456,ANS2 ;CHECK SECOND HALF OF ANSWER
BEQ .+6 ;BRANCH IF OK
HLT+2 ;ANS2 NOT EQUAL TO 123456
175 ;THE ERROR NUMBER IS 175

CMPB #25, STESTN ;CHECK THE TEST NUMBER
BEQ .+6 ;BRANCH IF OK
HLT ;WRONG TEST! PC MUST HAVE FOULED UP.
176 ;THE ERROR NUMBER IS 176

```


1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458

```

*****
:TEST 30: FSUB (LSI-11 FLOATING SUBTRACT INSTRUCTION)
: 035152 125252 - 043125 052525 = 143125,052524
: PS = 010, STACK POINTER = R3
*****
    
```

```

TST30: SCOPE
        JSR      R5,      PUSHR      ;PUSH 4 WORDS ONTO R3 STACK, SET PRIORITY
        .WORD   043125,052525      ;SECOND OPERAND ON TOP
        .WORD   035152,125252      ;FIRST OPERAND ON BOTTOM
        .WORD   147                  ;PROCESSOR PRIORITY LEVEL
        .WORD   TRAPER, 340         ;FIS TRAP VECTOR
        MOV     #STACK0,R3         ;SET UP STACK POINTER

        NOP
        FSUB   R3                  ;FLOATING SUBTRACT ON THE R3 STACK

        JSR      PC,      POPR      ;POP THE ANSWER
        MOV     R3,      $SP      ;SAVE "STACK POINTER"
        CMPB   #010,     $PSW     ;CHECK PS (EXCEPT T BIT)
        BEQ    .+6              ;BRANCH IF OK
        HLT    .+6              ;PS NOT EQUAL TO 010
        211                    ;THE ERROR NUMBER IS 211

        CMP     #STACK4,$SP      ;CHECK THE STACK POINTER (R3)
        BEQ    .+6              ;BRANCH IF OK
        HLT    .+6              ;STACK POINTER (R3) NOT EQUAL TO #STACK4
        212                    ;THE ERROR NUMBER IS 212

        CMP     #143125,ANS1     ;CHECK FIRST HALF OF ANSWER
        BEQ    .+6              ;BRANCH IF OK
        HLT+2  .+6              ;ANS1 NOT EQUAL TO 143125
        213                    ;THE ERROR NUMBER IS 213

        CMP     #052524,ANS2     ;CHECK SECOND HALF OF ANSWER
        BEQ    .+6              ;BRANCH IF OK
        HLT+2  .+6              ;ANS2 NOT EQUAL TO 052524
        214                    ;THE ERROR NUMBER IS 214

        END30: CMPB   #30,      $TESTN ;CHECK THE TEST NUMBER
        BEQ    .+6              ;BRANCH IF OK
        HLT    .+6              ;WRONG TEST! PC MUST HAVE FOULED UP.
        215                    ;THE ERROR NUMBER IS 215
    
```

010504
052525
125252
000340
000510
000240
075013
010504
172710
000010 172700
000514 172666
143125 172654
052524 172642
000030 172572

1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504

```
*****
:TEST 31:      FSUB (LSI-11 FLOATING SUBTRACT INSTRUCTION)
:      143325,052525 - 135152,125252 = 143325,052525
:      PS = 210,      STACK POINTER = R0
*****
```

```
TST31:  SCOPE
        JSR      RS,      PUSHR      ;PUSH 4 WORDS ONTO R0 STACK, SET PRIORITY
        .WORD    135152,125252      ;SECOND OPERAND ON TOP
        .WORD    143325,052525      ;FIRST OPERAND ON BOTTOM
        .WORD    243                ;PROCESSOR PRIORITY LEVEL
        .WORD    TRAPER, 340        ;FIS TRAP VECTOR
        MOV      #STACK0,R0        ;SET UP STACK POINTER

        NOP
        FSUB     R0                ;FLOATING SUBTRACT ON THE R0 STACK

        JSR      PC,      POPR      ;POP THE ANSWER
        MOV      R0,      $SP      ;SAVE "STACK POINTER"
        CMPB     #210,     $PSW     ;CHECK PS (EXCEPT T BIT)
        BEQ      .+6              ;BRANCH IF OK
        HLT      216              ;PS NOT EQUAL TO 210
        ;THE ERROR NUMBER IS 216

        CMP      #STACK4,$SP      ;CHECK THE STACK POINTER (R0)
        BEQ      .+6              ;BRANCH IF OK
        HLT      217              ;STACK POINTER (R0) NOT EQUAL TO #STACK4
        ;THE ERROR NUMBER IS 217

        CMP      #143325,ANS1     ;CHECK FIRST HALF OF ANSWER
        BEQ      .+6              ;BRANCH IF OK
        HLT+2   220              ;ANS1 NOT EQUAL TO 143325
        ;THE ERROR NUMBER IS 220

        CMP      #052525,ANS2     ;CHECK SECOND HALF OF ANSWER
        BEQ      .+6              ;BRANCH IF OK
        HLT+2   221              ;ANS2 NOT EQUAL TO 052525
        ;THE ERROR NUMBER IS 221

        CMPB     #31,      $TESTN   ;CHECK THE TEST NUMBER
        BEQ      .+6              ;BRANCH IF OK
        HLT      222              ;WRONG TEST! PC MUST HAVE FOULED UP.
        ;THE ERROR NUMBER IS 222
```

1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550

```
*****  
:TEST 32: FSUB (LSI-11 FLOATING SUBTRACT INSTRUCTION)  
: 135152,125252 - 143325,052525 = 043325,052525  
: PS = 200, STACK POINTER = R5  
:*****
```

```
TST32: SCOPE  
JSR R5, PUSHR ; PUSH 4 WORDS ONTO R5 STACK, SET PRIORITY  
.WORD 143325,052525 ; SECOND OPERAND ON TOP  
.WORD 135152,125252 ; FIRST OPERAND ON BOTTOM  
.WORD 357 ; PROCESSOR PRIORITY LEVEL  
.WORD TRAPER, 340 ; FIS TRAP VECTOR  
MOV #STACK0,R5 ; SET UP STACK POINTER  
  
NOP  
FSUB R5 ; FLOATING SUBTRACT ON THE R5 STACK  
  
JSR PC, POPR ; POP THE ANSWER  
MOV R5, $SP ; SAVE "STACK POINTER"  
CMPB #200, $PSW ; CHECK PS (EXCEPT T BIT)  
BEQ .+6 ; BRANCH IF OK  
HLT ; PS NOT EQUAL TO 200  
223 ; THE ERROR NUMBER IS 223  
  
CMP #STACK4,$SP ; CHECK THE STACK POINTER (R5)  
BEQ .+6 ; BRANCH IF OK  
HLT ; STACK POINTER (R5) NOT EQUAL TO #STACK4  
224 ; THE ERROR NUMBER IS 224  
  
CMP #043325,ANS1 ; CHECK FIRST HALF OF ANSWER  
BEQ .+6 ; BRANCH IF OK  
HLT+2 ; ANS1 NOT EQUAL TO 043325  
225 ; THE ERROR NUMBER IS 225  
  
CMP #052525,ANS2 ; CHECK SECOND HALF OF ANSWER  
BEQ .+6 ; BRANCH IF OK  
HLT+2 ; ANS2 NOT EQUAL TO 052525  
226 ; THE ERROR NUMBER IS 226  
  
END32: CMPB #32, $TESTN ; CHECK THE TEST NUMBER  
BEQ .+6 ; BRANCH IF OK  
HLT ; WRONG TEST! PC MUST HAVE FOULED UP.  
227 ; THE ERROR NUMBER IS 227
```

1551
 1552
 1553
 1554
 1555
 1556
 1557
 1558
 1559
 1560
 1561
 1562
 1563
 1564
 1565
 1566
 1567
 1568
 1569
 1570
 1571
 1572
 1573
 1574
 1575
 1576
 1577
 1578
 1579
 1580
 1581
 1582
 1583
 1584
 1585
 1586
 1587
 1588
 1589
 1590
 1591
 1592
 1593
 1594
 1595
 1596

```

*****
:TEST 33:      FSUB (LSI-11 FLOATING SUBTRACT INSTRUCTION)
:      043125,052525 - 035152,125252 = 043125,052524
:      PS = 000,      STACK POINTER = R2
*****
    
```

```

TST33:  SCOPE
        JSR      R5,      PUSHR      ;PUSH 4 WORDS ONTO R2 STACK, SET PRIORITY
        .WORD   035152,125252      ;SECOND OPERAND ON TOP
        .WORD   043125,052525      ;FIRST OPERAND ON BOTTOM
        .WORD   040                ;PROCESSOR PRIORITY LEVEL
        .WORD   TRAPER, 340        ;FIS TRAP VECTOR
        MOV      #STACK0,R2        ;SET UP STACK POINTER

        NOP
        FSUB    R2                ;FLOATING SUBTRACT ON THE R2 STACK

        JSR      PC,      POPR      ;POP THE ANSWER
        MOV      R2,      $SP      ;SAVE "STACK POINTER"
        TSTB    $PSW            ;CHECK PS (EXCEPT T BIT)
        BEQ     .+6              ;BRANCH IF OK
        HLT     104000          ;PS NOT EQUAL TO 000
        230                ;THE ERROR NUMBER IS 230

        CMP     #STACK4,$SP      ;CHECK THE STACK POINTER (R2)
        BEQ     .+6              ;BRANCH IF OK
        HLT     104000          ;STACK POINTER (R2) NOT EQUAL TO #STACK4
        231                ;THE ERROR NUMBER IS 231

        CMP     #043125,ANS1     ;CHECK FIRST HALF OF ANSWER
        BEQ     .+6              ;BRANCH IF OK
        HLT+2  104002          ;ANS1 NOT EQUAL TO 043125
        232                ;THE ERROR NUMBER IS 232

        CMP     #052524,ANS2     ;CHECK SECOND HALF OF ANSWER
        BEQ     .+6              ;BRANCH IF OK
        HLT+2  104002          ;ANS2 NOT EQUAL TO 052524
        233                ;THE ERROR NUMBER IS 233

        CMPB   #33,      $TESTN   ;CHECK THE TEST NUMBER
        BEQ     .+6              ;BRANCH IF OK
        HLT     104000          ;WRONG TEST! PC MUST HAVE FOULED UP.
        234                ;THE ERROR NUMBER IS 234
    
```

1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642

```
*****
:TEST 34:      FSUB (LSI-11 FLOATING SUBTRACT INSTRUCTION)
:      00000,00000 - 00000,00000 = 00000,00000
:      PS = 204,      STACK POINTER = RO
*****
```

```
TST34:  SCOPE
        JSR      RS,      PUSHR      ;PUSH 4 WORDS ONTO RO STACK, SET PRIORITY
        .WORD    00000,00000        ;SECOND OPERAND ON TOP
        .WORD    00000,00000        ;FIRST OPERAND ON BOTTOM
        .WORD    217                ;PROCESSOR PRIORITY LEVEL
        .WORD    TRAPER, 340        ;FIS TRAP VECTOR
        MOV      #STACK0,RO        ;SET UP STACK POINTER

        NOP
        FSUB     RO                ;FLOATING SUBTRACT ON THE RO STACK

        JSR      PC,      POPR      ;POP THE ANSWER
        MOV      RO,      SSP      ;SAVE "STACK POINTER"
        CMPB     #204,     $PSW     ;CHECK PS (EXCEPT T BIT)
        BEQ     .+6              ;BRANCH IF OK
        HLT     104000          ;PS NOT EQUAL TO 204
        HLT     235             ;THE ERROR NUMBER IS 235

        CMP      #STACK4,$SSP     ;CHECK THE STACK POINTER (RO)
        BEQ     .+6              ;BRANCH IF OK
        HLT     104000          ;STACK POINTER (RO) NOT EQUAL TO #STACK4
        HLT     236             ;THE ERROR NUMBER IS 236

        TST      ANS1           ;CHECK FIRST HALF OF ANSWER
        BEQ     .+6              ;BRANCH IF OK
        HLT+2   104002          ;ANS1 NOT EQUAL TO 000000
        HLT     237             ;THE ERROR NUMBER IS 237

        TST      ANS2           ;CHECK SECOND HALF OF ANSWER
        BEQ     .+6              ;BRANCH IF OK
        HLT+2   104002          ;ANS2 NOT EQUAL TO 000000
        HLT     240             ;THE ERROR NUMBER IS 240

        CMPB     #34,      $TESTN   ;CHECK THE TEST NUMBER
        BEQ     .+6              ;BRANCH IF OK
        HLT     104000          ;WRONG TEST! PC MUST HAVE FOULED UP.
        HLT     241             ;THE ERROR NUMBER IS 241
```

172102

172070

172060

172050

000034 172000 END34:

1643
 1644
 1645
 1646
 1647
 1648
 1649
 1650
 1651
 1652
 1653
 1654
 1655
 1656
 1657
 1658
 1659
 1660
 1661
 1662
 1663
 1664
 1665
 1666
 1667
 1668
 1669
 1670
 1671
 1672
 1673
 1674
 1675
 1676
 1677
 1678
 1679
 1680
 1681
 1682
 1683
 1684
 1685
 1686
 1687
 1688

```

*****
:TEST 35:      FSUB (LSI-11 FLOATING SUBTRACT INSTRUCTION)
:      00000,00000 - 17777,17777 = 07777,17777
:      PS = 000,      STACK POINTER = R2
*****
    
```

```

TST35:  SCOPE
        JSR      R5,      PUSHR      ;PUSH 4 WORDS ONTO R2 STACK, SET PRIORITY
        .WORD   17777,17777        ;SECOND OPERAND ON TOP
        .WORD   00000,00000        ;FIRST OPERAND ON BOTTOM
        .WORD   100                ;PROCESSOR PRIORITY LEVEL
        .WORD   TRAPER, 340        ;FIS TRAP VECTOR
        MOV     #STACK0,R2        ;SET UP STACK POINTER

        NOP
        FSUB   R2                ;FLOATING SUBTRACT ON THE R2 STACK

        JSR      PC,      POPR      ;POP THE ANSWER
        MOV     R2,      SSP        ;SAVE "STACK POINTER"
        TSTB   SPSW              ;CHECK PS (EXCEPT T BIT)
        BEQ    .+6                ;BRANCH IF OK
        HLT    242                ;PS NOT EQUAL TO 000
        ;THE ERROR NUMBER IS 242

        CMP     #STACK4,SSP        ;CHECK THE STACK POINTER (R2)
        BEQ    .+6                ;BRANCH IF OK
        HLT    243                ;STACK POINTER (R2) NOT EQUAL TO #STACK4
        ;THE ERROR NUMBER IS 243

        CMP     #077777,ANS1       ;CHECK FIRST HALF OF ANSWER
        BEQ    .+6                ;BRANCH IF OK
        HLT+2  244                ;ANS1 NOT EQUAL TO 077777
        ;THE ERROR NUMBER IS 244

        CMP     #177777,ANS2       ;CHECK SECOND HALF OF ANSWER
        BEQ    .+6                ;BRANCH IF OK
        HLT+2  245                ;ANS2 NOT EQUAL TO 177777
        ;THE ERROR NUMBER IS 245

        ENO35: CMPB   #35,      $TESTN ;CHECK THE TEST NUMBER
        BEQ    .+6                ;BRANCH IF OK
        HLT    246                ;WRONG TEST! PC MUST HAVE FOULED UP.
        ;THE ERROR NUMBER IS 246
    
```

1689
 1690
 1691
 1692
 1693
 1694
 1695
 1696
 1697
 1698
 1699
 1700
 1701
 1702
 1703
 1704
 1705
 1706
 1707
 1708
 1709
 1710
 1711
 1712
 1713
 1714
 1715
 1716
 1717
 1718
 1719
 1720
 1721
 1722
 1723
 1724
 1725
 1726
 1727
 1728
 1729
 1730
 1731
 1732
 1733
 1734
 1735
 1736
 1737
 1738
 1739

```

*****
TEST 36:      FSUB (LSI-11 FLOATING SUBTRACT INSTRUCTION)
              000000,000000 - 077777,177777 = 177777,177777
              PS = 210,      STACK POINTER = SP
*****
    
```

```

TST36:  SCOPE      RS,      PUSH5      ;PUSH 4 WORDS ONTO STACK, SET PRIORITY
        JSR        077777,177777      ;SECOND OPERAND ON TOP
        .WORD      000000,000000      ;FIRST OPERAND ON BOTTOM
        .WORD      217                ;PROCESSOR PRIORITY LEVEL
        .WORD      TRAPER, 340        ;FIS TRAP VECTOR
    
```

```

        NOP
        FSUB      SP                  ;FLOATING SUBTRACT ON THE STACK
    
```

```

        JSR      PC,      POPS        ;POP THE ANSWER
        CMP      #BEGIN, SP          ;CHECK THE STACK POINTER
        BEQ      TSA36              ;BRANCH IF OK
        MOV      #BEGIN, SP          ;RESTORE STACK POINTER
    
```

```

        HLT      247                ;STACK POINTER FOULED UP
        BR       END36              ;THE ERROR NUMBER IS 247
        BR       END36              ;SKIP REST OF TEST
    
```

```

TSA36:  CMPB     #210,      SPSW      ;CHECK PS (EXCEPT T BIT)
        BEQ     .+6              ;BRANCH IF OK
        HLT     250              ;PS NOT EQUAL TO 210
        HLT     250              ;THE ERROR NUMBER IS 250
    
```

```

        CMP     #177777,ANS1        ;CHECK FIRST HALF OF ANSWER
        BEQ     .+6              ;BRANCH IF OK
        HLT+2   251              ;ANS1 NOT EQUAL TO 177777
        HLT     251              ;THE ERROR NUMBER IS 251
    
```

```

        CMP     #177777,ANS2        ;CHECK SECOND HALF OF ANSWER
        BEQ     .+6              ;BRANCH IF OK
        HLT+2   252              ;ANS2 NOT EQUAL TO 177777
        HLT     252              ;THE ERROR NUMBER IS 252
    
```

```

END36:  CMPB     #36,      STESTN    ;CHECK THE TEST NUMBER
        BEQ     .+6              ;BRANCH IF OK
        HLT     253              ;WRONG TEST! PC MUST HAVE FOULED UP.
        HLT     253              ;THE ERROR NUMBER IS 253
    
```

TEST FLOATING SUB. INSTRUCTION WITH UNDERFLOW

1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795

```

*****
:TEST 37: FSUB (LSI-11 FLOATING SUBTRACT INSTRUCTION)
:         000425,052525 - 000252,125253 ==> UNDERFLOW
:         PS(ON STACK) = 212, STACK POINTER = SP
*****
    
```

```

TST37:  SCOPE
        JSR      R5,     PUSH5      ;PUSH 4 WORDS ONTO STACK, SET PRIORITY
        .WORD   000252,125253      ;SECOND OPERAND ON TOP
        .WORD   000425,052525      ;FIRST OPERAND ON BOTTOM
        .WORD   257                ;PROCESSOR PRIORITY LEVEL
        .WORD   ISR37, 340         ;FIS TRAP VECTOR

RTA37:  NOP
        FSUB     SP                ;FLOATING SUBTRACT ON THE STACK

ISR37:  JSR      PC,     POPS       ;POP THE "ANSWER"
        HLT+2   254          ;FIS TRAP DIDN'T OCCURE!
        MOV     #BEGIN, SP      ;THE ERROR NUMBER IS 254
        BR      END37          ;RESTORE THE STACK POINTER

ISA37:  JSR      PC,     POPES      ;POP ALL DATA OFF THE STACK
        CMP     #BEGIN, SP      ;CHECK THE STACK POINTER
        BEQ    ISA37          ;BRANCH IF OK
        MOV     #BEGIN, SP      ;RESTORE THE STACK POINTER
        HLT    255            ;STACK POINTER FOULED UP
        BR      END37          ;THE ERROR NUMBER IS 255
                                   ;SKIP REST OF TEST

ISA37:  CMPB    #340,     SPSW      ;CHECK PS AFTER FIS TRAP
        BEQ    .+6            ;BRANCH IF OK
        HLT    256            ;PS AFTER FIS TRAP NOT EQUAL TO 340
        BR      .+6            ;THE ERROR NUMBER IS 256

RTA37:  CMP     #RTA37, ANS1      ;CHECK FIS TRAP RETURN ADDRESS
        BEQ    .+6            ;BRANCH IF OK
        HLT+1  257            ;FIS TRAP AT WRONG ADDRESS
        BR      .+6            ;THE ERROR NUMBER IS 257

ISA37:  CMP     #212,     ANS2      ;CHECK PS BEFORE FIS TRAP
        BEQ    .+6            ;BRANCH IF OK
        HLT+2  260            ;PS AT FIS TRAP TIME NOT 212
        BR      .+6            ;THE ERROR NUMBER IS 260

ISA37:  CMP     #000252,ANS3      ;CHECK DATA FROM THE STACK
        BEQ    .+6            ;BRANCH IF OK
        HLT+4  261            ;DATA ON STACK (000252) CHANGED
        BR      .+6            ;THE ERROR NUMBER IS 261

ISA37:  CMP     #125253,ANS4      ;CHECK DATA FROM STACK
        BEQ    .+6            ;BRANCH IF OK
        HLT+4  262            ;DATA ON STACK (125253) CHANGED
        BR      .+6            ;THE ERROR NUMBER IS 262
    
```


1812
 1813
 1814
 1815
 1816
 1817
 1818
 1819
 1820
 1821
 1822
 1823
 1824
 1825
 1826
 1827
 1828
 1829
 1830
 1831
 1832
 1833
 1834
 1835
 1836
 1837
 1838
 1839
 1840
 1841
 1842
 1843
 1844
 1845
 1846
 1847
 1848
 1849
 1850
 1851
 1852
 1853
 1854
 1855
 1856
 1857
 1858
 1859
 1860
 1861
 1862
 1863
 1864
 1865
 1866
 1867

007136 104400
 007140 004567 007026
 007144 177452 125252
 007150 077652 125253
 007154 000015
 007156 007210 000344
 007162 012703 000510
 007166 000240
 007170 075013
 007172 004767 007026
 007176 010367 171232
 007202 104002
 007204 000266
 007206 000464
 007210 004767 007040
 007214 010367 171214
 007220 122767 000344 171204
 007226 001402
 007230 104000
 007232 000267
 007234 022767 000510 171172
 007242 001402
 007244 104000
 007246 000270
 007250 022767 007172 171160
 007256 001402
 007260 104001
 007262 000271
 007264 022767 000002 171146
 007272 001402
 007274 104002
 007276 000272
 007300 022767 177452 171134
 007306 001402
 007310 104004
 007312 000273
 007314 022767 125252 171122
 007322 001402
 007324 104004
 007326 000274

```

*****
:TEST 40: FSUB (LSI-11 FLOATING SUBTRACT INSTRUCTION)
:         077652,125253 - 177452,125252 ==> OVERFLOW
:         PS(ON STACK) = 002, STACK POINTER = R3
*****
    
```

```

TST40: SCOPE
        JSR     R5,     PUSH4    ; PUSH 4 WORDS ONTO R3 STACK, SET PRIORITY
        .WORD  177452,125252    ; SECOND OPERAND ON TOP
        .WORD  077652,125253    ; FIRST OPERAND ON BOTTOM
        .WORD  015              ; PROCESSOR PRIORITY LEVEL
        .WORD  ISR40, 344       ; FIS TRAP VECTOR
        MOV     #STACK0,R3     ; SET UP R3 AS STACK POINTER

        NOP
        FSUB   R3              ; FLOATING SUBTRACT ON THE R3 STACK

RTA40:  JSR     PC,     POPR     ; POP THE "ANSWER"
        MOV     R3,     SSP     ; SAVE STACK POINTER (R3)
        HLT+2   266          ; FIS TRAP DIDN'T OCCURE!
        BR      END40        ; THE ERROR NUMBER IS 266

ISR40:  JSR     PC,     POPER    ; POP ALL DATA OFF THE STACKS
        MOV     R3,     SSP     ; SAVE STACK POINTER (R3)
        CMPB   #344,    SPSW    ; CHECK PS AFTER FIS TRAP
        BEQ    .+6           ; BRANCH IF OK
        HLT    267          ; PS AFTER FIS TRAP NOT EQUAL TO 344
        BR      267        ; THE ERROR NUMBER IS 267

        CMP     #STACK0,SSP    ; CHECK THE STACK POINTER (R3)
        BEQ    .+6           ; BRANCH IF OK
        HLT    270          ; STACK POINTER (R3) NOT EQUAL TO #STACK0
        BR      270        ; THE ERROR NUMBER IS 270

        CMP     #RTA40,ANS1    ; CHECK FIS TRAP RETURN ADDRESS
        BEQ    .+6           ; BRANCH IF OK
        HLT+1  271          ; FIS TRAP AT WRONG ADDRESS
        BR      271        ; THE ERROR NUMBER IS 271

        CMP     #002,ANS2     ; CHECK PS BEFORE FIS TRAP
        BEQ    .+6           ; BRANCH IF OK
        HLT+2  272          ; PS AT FIS TRAP TIME NOT 002
        BR      272        ; THE ERROR NUMBER IS 272

        CMP     #177452,ANS3   ; CHECK DATA FROM THE STACK
        BEQ    .+6           ; BRANCH IF OK
        HLT+4  273          ; DATA ON STACK (177452) CHANGED
        BR      273        ; THE ERROR NUMBER IS 273

        CMP     #125252,ANS4   ; CHECK DATA FROM STACK
        BEQ    .+6           ; BRANCH IF OK
        HLT+4  274          ; DATA ON STACK (125252) CHANGED
        BR      274        ; THE ERROR NUMBER IS 274
    
```


1884
 1885
 1886
 1887
 1888
 1889
 1890
 1891
 1892
 1893
 1894
 1895
 1896
 1897
 1898
 1899
 1900
 1901
 1902
 1903
 1904
 1905
 1906
 1907
 1908
 1909
 1910
 1911
 1912
 1913
 1914
 1915
 1916
 1917
 1918
 1919
 1920
 1921
 1922
 1923
 1924
 1925
 1926
 1927
 1928
 1929

```

*****
:TEST 41:      FMUL (LSI-1) FLOATING MULTIPLY INSTRUCTION)
:      00000,00000 * 00000,00000 = 00000,00000
:      PS = 004,      STACK POINTER = R4
*****
    
```

```

TST41:  SCOPE
        JSR      RS,      PUSHR      ;PUSH 4 WORDS ONTO R4 STACK, SET PRIORITY
        .WORD   00000,00000        ;SECOND OPERAND ON TOP
        .WORD   00000,00000        ;FIRST OPERAND ON BOTOM
        .WORD   111                ;PROCESSOR PRIORITY LEVEL
        .WORD   TRAPER, 340        ;FIS TRAP VECTOR
        MOV     #STACK0,R4        ;SET UP STACK POINTER

        NOP
        FMUL   R4                ;FLOATING MULTIPLY ON THE R4 STACK

        JSR      PC,      POPR      ;POP THE ANSWER
        MOV     R4,      $SP      ;SAVE "STACK POINTER"
        CMPB   #004,     $PSW     ;CHECK PS (EXCEPT T BIT)
        BEQ    .+6              ;BRANCH IF OK
        HLT    300              ;PS NOT EQUAL TO 004
                                ;THE ERROR NUMBER IS 300

        CMP     #STACK4,$SP      ;CHECK THE STACK POINTER (R4)
        BEQ    .+6              ;BRANCH IF OK
        HLT    301              ;STACK POINTER (R4) NOT EQUAL TO #STACK4
                                ;THE ERROR NUMBER IS 301

        TST     ANS1
        BEQ    .+6              ;CHECK FIRST HALF OF ANSWER
        HLT+2 302              ;BRANCH IF OK
                                ;ANS1 NOT EQUAL TO 00000
                                ;THE ERROR NUMBER IS 302

        TST     ANS2
        BEQ    .+6              ;CHECK SECOND HALF OF ANSWER
        HLT+2 303              ;BRANCH IF OK
                                ;ANS2 NOT EQUAL TO 00000
                                ;THE ERROR NUMBER IS 303

        CMPB   #41,      $TESTN   ;CHECK THE TEST NUMBER
        BEQ    .+6              ;BRANCH IF OK
        HLT    304              ;WRONG TEST! PC MUST HAVE FOULED UP.
                                ;THE ERROR NUMBER IS 304
    
```

170764

170752

170742

170732

000041 170662 END41:

1930
 1931
 1932
 1933
 1934
 1935
 1936
 1937
 1938
 1939
 1940
 1941
 1942
 1943
 1944
 1945
 1946
 1947
 1948
 1949
 1950
 1951
 1952
 1953
 1954
 1955
 1956
 1957
 1958
 1959
 1960
 1961
 1962
 1963
 1964
 1965
 1966
 1967
 1968
 1969
 1970
 1971
 1972
 1973
 1974
 1975

```

*****
:TEST 42:      FMUL (LSI-11 FLOATING MULTIPLY INSTRUCTION)
:      140200,000000 * 052345,123456 = 152345,123456
:      PS = 210,      STACK POINTER = R2
*****
    
```

```

TST42:  SCOPE
        JSR      R5,      PUSHR      ;PUSH 4 WORDS ONTO R2 STACK, SET PRIORITY
        .WORD   052345,123456      ;SECOND OPERAND ON TOP
        .WORD   140200,000000      ;FIRST OPERAND ON BOTOM
        .WORD   343                ;PROCESSOR PRIORITY LEVEL
        .WORD   TRAPER, 340        ;FIS TRAP VECTOR
        MOV     #STACK0,R2        ;SET UP STACK POINTER

        NOP
        FMUL   R2                ;FLOATING MULTIPLY ON THE R2 STACK

        JSR      PC,      POPR      ;POP THE ANSWER
        MOV     R2,      SSP      ;SAVE "STACK POINTER"
        CMPB   #210,     SPSW     ;CHECK PS (EXCEPT T BIT)
        BEQ    .+6              ;BRANCH IF OK
        HLT    305              ;PS NOT EQUAL TO 210
        305                    ;THE ERROR NUMBER IS 305

        CMP     #STACK4,SSP      ;CHECK THE STACK POINTER (R2)
        BEQ    .+6              ;BRANCH IF OK
        HLT    306              ;STACK POINTER (R2) NOT EQUAL TO #STACK4
        306                    ;THE ERROR NUMBER IS 306

        CMP     #152345,ANS1     ;CHECK FIRST HALF OF ANSWER
        BEQ    .+6              ;BRANCH IF OK
        HLT+2 307              ;ANS1 NOT EQUAL TO 152345
        307                    ;THE ERROR NUMBER IS 307

        CMP     #123456,ANS2     ;CHECK SECOND HALF OF ANSWER
        BEQ    .+6              ;BRANCH IF OK
        HLT+2 310              ;ANS2 NOT EQUAL TO 123456
        310                    ;THE ERROR NUMBER IS 310

        CMPB   #42,      $TESTN   ;CHECK THE TEST NUMBER
        BEQ    .+6              ;BRANCH IF OK
        HLT    311              ;WRONG TEST! PC MUST HAVE FOULED UP.
        311                    ;THE ERROR NUMBER IS 311
    
```

1976
 1977
 1978
 1979
 1980
 1981
 1982
 1983
 1984
 1985
 1986
 1987
 1988
 1989
 1990
 1991
 1992
 1993
 1994
 1995
 1996
 1997
 1998
 1999
 2000
 2001
 2002
 2003
 2004
 2005
 2006
 2007
 2008
 2009
 2010
 2011
 2012
 2013
 2014
 2015
 2016
 2017
 2018
 2019
 2020
 2021

```

*****
:TEST 43:      FMUL (LSI-11 FLOATING MULTIPLY INSTRUCTION)
:              100125,052525 * 135753,024642 = 000000,000000
:              PS = 004,      STACK POINTER = R5
*****
    
```

```

TST43:  SCOPE
        JSR      R5,      PUSHR      ;PUSH 4 WORDS ONTO R5 STACK, SET PRIORITY
        .WORD   135753,024642      ;SECOND OPERAND ON TOP
        .WORD   100125,052525      ;FIRST OPERAND ON BOTOM
        .WORD   117                ;PROCESSOR PRIORITY LEVEL
        .WORD   TRAPER, 340        ;FIS TRAP VECTOR
        MOV      #STACK0,R5        ;SET UP STACK POINTER

        NOP
        FMUL    R5                ;FLOATING MULTIPLY ON THE R5 STACK

        JSR      PC,      POPR       ;POP THE ANSWER
        MOV      R5,      SSP        ;SAVE "STACK POINTER"
        CMPB    #004,    SPSW       ;CHECK PS (EXCEPT T BIT)
        BEQ     .+6                ;BRANCH IF OK
        HLT     312                ;PS NOT EQUAL TO 004
        312                ;THE ERROR NUMBER IS 312

        CMP     #STACK4,SSP        ;CHECK THE STACK POINTER (R5)
        BEQ     .+6                ;BRANCH IF OK
        HLT     313                ;STACK POINTER (R5) NOT EQUAL TO #STACK4
        313                ;THE ERROR NUMBER IS 313

        TST     ANS1               ;CHECK FIRST HALF OF ANSWER
        BEQ     .+6                ;BRANCH IF OK
        HLT+2   314                ;ANS1 NOT EQUAL TO 000000
        314                ;THE ERROR NUMBER IS 314

        TST     ANS2               ;CHECK SECOND HALF OF ANSWER
        BEQ     .+6                ;BRANCH IF OK
        HLT+2   315                ;ANS2 NOT EQUAL TO 000000
        315                ;THE ERROR NUMBER IS 315

        CMPB    #43,      $TESTN    ;CHECK THE TEST NUMBER
        BEQ     .+6                ;BRANCH IF OK
        HLT     316                ;WRONG TEST! PC MUST HAVE FOULED UP.
        316                ;THE ERROR NUMBER IS 316
    
```

2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067

```
*****
:TEST 44:      FMUL (LSI-11 FLOATING MULTIPLY INSTRUCTION)
:      161616,161616 * 000052,125252 = 000000,000000
:      PS = 204,      STACK POINTER = R3
*****
```

```
TST44:  SCOPE
        JSR      R5,      PUSHR      ;PUSH 4 WORDS ONTO R3 STACK, SET PRIORITY
        .WORD   000052,125252      ;SECOND OPERAND ON TOP
        .WORD   161616,161616      ;FIRST OPERAND ON BOTOM
        .WORD   217                ;PROCESSOR PRIORITY LEVEL
        .WORD   TRAPER, 340        ;FIS TRAP VECTOR
        MOV      #STACK0,R3        ;SET UP STACK POINTER

        NOP
        FMUL    R3                ;FLOATING MULTIPLY ON THE R3 STACK

        JSR      PC,      POPR       ;POP THE ANSWER
        MOV      R3,      SSP       ;SAVE "STACK POINTER"
        CMPB    #204,     SPSW      ;CHECK PS (EXCEPT T BIT)
        BEQ     .+6               ;BRANCH IF OK
        HLT     317              ;PS NOT EQUAL TO 204
                                   ;THE ERROR NUMBER IS 317

        CMP     #STACK4,SSP        ;CHECK THE STACK POINTER (R3)
        BEQ     .+6               ;BRANCH IF OK
        HLT     320              ;STACK POINTER (R3) NOT EQUAL TO #STACK4
                                   ;THE ERROR NUMBER IS 320

        TST     ANS1              ;CHECK FIRST HALF OF ANSWER
        BEQ     .+6               ;BRANCH IF OK
        HLT+2  321              ;ANS1 NOT EQUAL TO 000000
                                   ;THE ERROR NUMBER IS 321

        TST     ANS2              ;CHECK SECOND HALF OF ANSWER
        BEQ     .+6               ;BRANCH IF OK
        HLT+2  322              ;ANS2 NOT EQUAL TO 000000
                                   ;THE ERROR NUMBER IS 322

        CMPB    #44,          STSTN  ;CHECK THE TEST NUMBER
        BEQ     .+6               ;BRANCH IF OK
        HLT     323              ;WRONG TEST! PC MUST HAVE FOULED UP.
                                   ;THE ERROR NUMBER IS 323
```

000044 170232 END44:

2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159

```
*****  
:TEST 46:      FMUL (LSI-11 FLOATING MULTIPLY INSTRUCTION)  
:      124252,125252 * 114100,000001 = 000200,000000  
:      PS = 200,      STACK POINTER = R1  
*****
```

```
TST46:  SCOPE  
JSR     RS,      PUSHR      ;PUSH 4 WORDS ONTO R1 STACK, SET PRIORITY  
.WORD  114100,000001      ;SECOND OPERAND ON TOP  
.WORD  124252,125252      ;FIRST OPERAND ON BOTOM  
.WORD  200              ;PROCESSOR PRIORITY LEVEL  
.WORD  TRAPER, 340        ;FIS TRAP VECTOR  
MOV     #STACK0,R1        ;SET UP STACK POINTER  
  
NOP  
FMUL   R1                ;FLOATING MULTIPLY ON THE R1 STACK  
  
JSR     PC,      POPR       ;POP THE ANSWER  
MOV     R1,      $SP        ;SAVE "STACK POINTER"  
CMPB   #200,    $PSW        ;CHECK PS (EXCEPT T BIT)  
BEQ    .+6          ;BRANCH IF OK  
HLT    ;PS NOT EQUAL TO 200  
331    ;THE ERROR NUMBER IS 331  
  
CMP     #STACK4,$SP        ;CHECK THE STACK POINTER (R1)  
BEQ    .+6          ;BRANCH IF OK  
HLT    ;STACK POINTER (R1) NOT EQUAL TO #STACK4  
332    ;THE ERROR NUMBER IS 332  
  
CMP     #000200,ANS1       ;CHECK FIRST HALF OF ANSWER  
BEQ    .+6          ;BRANCH IF OK  
HLT+2  ;ANS1 NOT EQUAL TO 000200  
333    ;THE ERROR NUMBER IS 333  
  
TST    ANS2              ;CHECK SECOND HALF OF ANSWER  
BEQ    .+6          ;BRANCH IF OK  
HLT+2  ;ANS2 NOT EQUAL TO 000000  
334    ;THE ERROR NUMBER IS 334  
  
CMPB   #46,      $TESTN    ;CHECK THE TEST NUMBER  
BEQ    .+6          ;BRANCH IF OK  
HLT    ;WRONG TEST! PC MUST HAVE FOULED UP.  
335    ;THE ERROR NUMBER IS 335
```

2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213

```
*****
:TEST 47: FSUB (LSI-11 FLOATING SUBTRACT INSTRUCTION)
:104000,105004 - 104000,104000 = 100401,000000
:PS = 210, STACK POINTER = PC
*****
```

```
TST47: SCOPE
JSR RS, PUSH7 ; PUSH 4 WORDS ONTO STACK, SET PRIORITY
.WORD STK47 ; TOP OF STACK
.WORD 104000,104000 ; SECOND OPERAND ON TOP
.WORD 104000,105004 ; FIRST OPERAND ON BOTTOM
.WORD 252 ; PROCESSOR PRIORITY LEVEL
.WORD TRAPER,340 ; FIS TRAP VECTOR

STK47: NOP
FSUB PC ; FLOATING SUBTRACT ON FOLLOWING 4 WORDS
104000 ; SHOULD CONTAIN 104000
104000 ; SHOULD CONTAIN 104000
104000 ; BEFORE FSUB, 104000; AFTER, 100401
105004 ; BEFORE FSUB, 105004; AFTER, 000000

JSR PC, POP7 ; POP THE ANSWER
CMPB #210, SPSW ; CHECK PS (EXCEPT T BIT)
BEQ .+6 ; BRANCH IF OK
HLT ; PS NOT EQUAL TO 210
336 ; THE ERROR NUMBER IS 336

CMP #104000,ANS1 ; CHECK FIRST HALF OF INPUT DATA (STK47)
BEQ .+6 ; BRANCH IF OK
HLT+2 ; ANS1 NOT EQUAL TO 104000
337 ; THE ERROR NUMBER IS 337

CMP #104000,ANS2 ; CHECK SECOND HALF OF INPUT DATA (STK47+2)
BEQ .+6 ; BRANCH IF OK
HLT+2 ; ANS2 NOT EQUAL TO 104000
340 ; THE ERROR NUMBER IS 340

CMP #100401,ANS3 ; CHECK FIRST HALF OF ANSWER
BEQ .+6 ; BRANCH IF OK
HLT+4 ; ANS3 NOT EQUAL TO 100401
341 ; THE ERROR NUMBER IS 341

TST ANS4 ; CHECK SECOND HALF OF ANSWER
BEQ .+6 ; BRANCH IF OK
HLT+4 ; ANS4 NOT EQUAL TO 000000
342 ; THE ERROR NUMBER IS 342

END47: CMPB #47, $TESTN ; CHECK THE TEST NUMBER
BEQ .+6 ; BRANCH IF OK
HLT ; WRONG TEST! PC MUST HAVE FOULED UP.
343 ; THE ERROR NUMBER IS 343
```

```

2214
2215
2216 :*****
2217 :TEST 50: FMUL (LSI-11 FLOATING MULTIPLY INSTRUCTION)
2218 :134600,073601 * 104000,104000 = 000401,000000
2219 :PS = 200, STACK POINTER = PC
2220 :*****
2221 010626 104400
2222 010630 004567 005510 TST50: SCOPE JSR R5, PUSH7 ;PUSH 4 WORDS ONTO STACK, SET PRIORITY
2223 010634 010660 .WORD STK50 ;TOP OF STACK
2224 010636 104000 104000 .WORD 104000,104000 ;SECOND OPERAND ON TOP
2225 010642 134600 073601 .WORD 134600,073601 ;FIRST OPERAND ON BOTTOM
2226 010646 000246 .WORD 246 ;PROCESSOR PRIORITY LEVEL
2227 010650 016442 000340 .WORD TRAPER,340 ;FIS TRAP VECTOR
2228
2229 010654 000240 NOP
2230 010656 075027 FMUL PC ;FLOATING MULTIPLY ON FOLLOWING 4 WORDS
2231 010660 104000 STK50: 104000 ;SHOULD CONTAIN 104000
2232 010662 104000 104000 ;SHOULD CONTAIN 104000
2233 010664 134600 134600 ;BEFORE FMUL, 134600; AFTER, 000401
2234 010666 073601 073601 ;BEFORE FMUL, 073601; AFTER, 000000
2235
2236 010670 004767 005500 JSR PC, POP7 ;POP THE ANSWER
2237 010674 122767 000200 167530 CMPB #200, $PSW ;CHECK PS (EXCEPT T BIT)
2238 010702 001402 BEQ .+6 ;BRANCH IF OK
2239 010704 104000 HLT ;PS NOT EQUAL TO 200
2240 010706 000344 344 ;THE ERROR NUMBER IS 344
2241
2242 010710 022767 104000 167520 CMP #104000,ANS1 ;CHECK FIRST HALF OF INPUT DATA (STK50)
2243 010716 001402 BEQ .+6 ;BRANCH IF OK
2244 010720 104002 HLT+2 ;ANS1 NOT EQUAL TO 104000
2245 010722 000345 345 ;THE ERROR NUMBER IS 345
2246
2247 010724 022767 104000 167506 CMP #104000,ANS2 ;CHECK SECOND HALF OF INPUT DATA (STK50+2)
2248 010732 001402 BEQ .+6 ;BRANCH IF OK
2249 010734 104002 HLT+2 ;ANS2 NOT EQUAL TO 104000
2250 010736 000346 346 ;THE ERROR NUMBER IS 346
2251
2252 010740 022767 000401 167474 CMP #000401,ANS3 ;CHECK FIRST HALF OF ANSWER
2253 010746 001402 BEQ .+6 ;BRANCH IF OK
2254 010750 104004 HLT+4 ;ANS3 NOT EQUAL TO 000401
2255 010752 000347 347 ;THE ERROR NUMBER IS 347
2256
2257 010754 005767 167464 TST ANS4 ;CHECK SECOND HALF OF ANSWER
2258 010760 001402 BEQ .+6 ;BRANCH IF OK
2259 010762 104004 HLT+4 ;ANS4 NOT EQUAL TO 000000
2260 010764 000350 350 ;THE ERROR NUMBER IS 350
2261
2262 010766 122767 000050 167410 END50: CMPB #50, $TESTN ;CHECK THE TEST NUMBER
2263 010774 001402 BEQ .+6 ;BRANCH IF OK
2264 010776 104000 HLT ;WRONG TEST! PC MUST HAVE FOULED UP.
2265 011000 000351 351 ;THE ERROR NUMBER IS 351
2266
2267
2268
2269 :*****

```

TEST FLOATING MUL. INSTRUCTION WITH UNDERFLOW

```

2270          :TEST 51:          FMUL (LSI-11 FLOATING MULTIPLY INSTRUCTION)
2271          :                024252,125252 * 114100,000000 ==> UNDERFLOW
2272          :                PS(ON STACK) = 212,          STACK POINTER = RO
2273          :                *****
2274
2275 011002 104400          TST51: SCOPE
2276 011004 004567 005162 JSR    RS,    PUSHR ;PUSH 4 WORDS ONTO RO STACK, SET PRIORITY
2277 011010 114100 000000 .WORD 114100,000000 ;SECOND OPERAND ON TOP
2278 011014 024252 125252 .WORD 024252,125252 ;FIRST OPERAND ON BOTTOM
2279 011020 000305 .WORD 305 ;PROCESSOR PRIORITY LEVEL
2280 011022 011054 000057 .WORD ISR51, 057 ;FIS TRAP VECTOR
2281 011026 012700 000510 MOV    #STACK0,RO ;SET UP RO AS STACK POINTER
2282
2283 011032 000240          NOP
2284 011034 075020          FMUL    RO          ;FLOATING MULTIPLY ON THE RO STACK
2285
2286 011036 004767 005162 RTA51: JSR    PC,    POPR ;POP THE "ANSWER"
2287 011042 010067 167366 MOV    RO,    SSP ;SAVE STACK POINTER (RO)
2288 011046 104002          HLT+2 ;FIS TRAP DIDN'T OCCURE!
2289 011050 000352          352 ;THE ERROR NUMBER IS 352
2290 011052 000463          BR      ENDS1
2291
2292 011054 004767 005174 ISR51: JSR    PC,    POPER ;POP ALL DATA OFF THE STACKS
2293 011060 010067 167350 MOV    RO,    SSP ;SAVE STACK POINTER (RO)
2294 011064 122767 000057 167340 CMPB  #057,    SPSW ;CHECK PS AFTER FIS TRAP
2295 011072 001402          BEQ    .+6 ;BRANCH IF OK
2296 011074 104000          HLT ;PS AFTER FIS TRAP NOT EQUAL TO 057
2297 011076 000353          353 ;THE ERROR NUMBER IS 353
2298
2299 011100 022767 000510 167326 CMP    #STACK0,SSP ;CHECK THE STACK POINTER (RO)
2300 011106 001402          BEQ    .+6 ;BRANCH IF OK
2301 011110 104000          HLT ;STACK POINTER (RO) NOT EQUAL TO #STACK0
2302 011112 000354          354 ;THE ERROR NUMBER IS 354
2303
2304 011114 022767 011036 167314 CMP    #RTA51, ANS1 ;CHECK FIS TRAP RETURN ADDRESS
2305 011122 001402          BEQ    .+6 ;BRANCH IF OK
2306 011124 104001          HLT+1 ;FIS TRAP AT WRONG ADDRESS
2307 011126 000355          355 ;THE ERROR NUMBER IS 355
2308
2309 011130 022767 000212 167302 CMP    #212,    ANS2 ;CHECK PS BEFORE FIS TRAP
2310 011136 001402          BEQ    .+6 ;BRANCH IF OK
2311 011140 104002          HLT+2 ;PS AT FIS TRAP TIME NOT 212
2312 011142 000356          356 ;THE ERROR NUMBER IS 356
2313
2314 011144 022767 114100 167270 CMP    #114100,ANS3 ;CHECK DATA FROM THE STACK
2315 011152 001402          BEQ    .+6 ;BRANCH IF OK
2316 011154 104004          HLT+4 ;DATA ON STACK (114100) CHANGED
2317 011156 000357          357 ;THE ERROR NUMBER IS 357
2318
2319 011160 005767 167260 TST    ANS4 ;CHECK DATA FROM STACK
2320 011164 001402          BEQ    .+6 ;BRANCH IF OK
2321 011166 104004          HLT+4 ;DATA ON STACK (000000) CHANGED
2322 011170 000360          360 ;THE ERROR NUMBER IS 360
2323
2324 011172 022767 024252 167246 CMP    #024252,ANS5 ;CHECK DATA FROM STACK
2325 011200 001402          BEQ    .+6 ;BRANCH IF OK
    
```


TEST FLOATING MUL. INSTRUCTION WITH OVERFLOW

2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395

```

*****
:TEST 52:      FMUL (LSI-11 FLOATING MULTIPLY INSTRUCTION)
:              076452,125252 * 041500,000001 ==> OVERFLOW
:              PS(ON STACK) = 002,      STACK POINTER = SP
*****
    
```

```

TST52:  SCOPE
        JSR      R5,      PUSH5      ;PUSH 4 WORDS ONTO STACK, SET PRIORITY
        .WORD   041500,000001      ;SECOND OPERAND ON TOP
        .WORD   076452,125252      ;FIRST OPERAND ON BOTTOM
        .WORD   105                ;PROCESSOR PRIORITY LEVEL
        .WORD   ISRS2, 357         ;FIS TRAP VECTOR
    
```

```

        NOP
        FMUL    SP                ;FLOATING MULTIPLY ON THE STACK
    
```

```

RTA52:  JSR      PC,      POPS       ;POP THE "ANSWER"
        HLT+2
        364        ;FIS TRAP DIDN'T OCCURE!
        MOV      #BEGIN, SP        ;THE ERROR NUMBER IS 364
        BR       END52            ;RESTORE THE STACK POINTER
    
```

```

ISRS2:  JSR      PC,      POPES      ;POP ALL DATA OFF THE STACK
        CMP      #BEGIN, SP        ;CHECK THE STACK POINTER
        BEQ      ISA52            ;BRANCH IF OK
        MOV      #BEGIN, SP        ;RESTORE THE STACK POINTER
        HLT
        365        ;STACK POINTER FOULED UP
        BR       END52            ;THE ERROR NUMBER IS 365
        ;SKIP REST OF TEST
    
```

```

ISA52:  CMPB     #357,  $PSW        ;CHECK PS AFTER FIS TRAP
        BEQ      .+6              ;BRANCH IF OK
        HLT
        366        ;PS AFTER FIS TRAP NOT EQUAL TO 357
        ;THE ERROR NUMBER IS 366
    
```

```

        CMP      #RTA52, ANS1     ;CHECK FIS TRAP RETURN ADDRESS
        BEQ      .+6              ;BRANCH IF OK
        HLT+1
        367        ;FIS TRAP AT WRONG ADDRESS
        ;THE ERROR NUMBER IS 367
    
```

```

        CMP      #002,  ANS2      ;CHECK PS BEFORE FIS TRAP
        BEQ      .+6              ;BRANCH IF OK
        HLT+2
        370        ;PS AT FIS TRAP TIME NOT 002
        ;THE ERROR NUMBER IS 370
    
```

```

        CMP      #041500,ANS3     ;CHECK DATA FROM THE STACK
        BEQ      .+6              ;BRANCH IF OK
        HLT+4
        371        ;DATA ON STACK (041500) CHANGED
        ;THE ERROR NUMBER IS 371
    
```

```

        CMP      #000001,ANS4    ;CHECK DATA FROM STACK
        BEQ      .+6              ;BRANCH IF OK
        HLT+4
        372        ;DATA ON STACK (000001) CHANGED
        ;THE ERROR NUMBER IS 372
    
```


2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457

```

*****
:TEST 53:      FDIV (LSI-11 FLOATING DIVIDE INSTRUCTION)
:      167452,125251 / 127652,125252 = 077777,177776
:      PS = 000,      STACK POINTER = R0
*****
    
```

```

011470 104400
011472 004567 004474
011476 127652 125252
011502 167452 125251
011506 000111
011510 016442 000340
011514 012700 000510

011520 000240
011522 075030

011524 004767 004474
011530 010067 166700
011534 105767 166672
011540 001402
011542 104000
011544 000376

011546 022767 000514 166660
011554 001402
011556 104000
011560 000377

011562 022767 077777 166646
011570 001402
011572 104002
011574 000400

011576 022767 177776 166634
011604 001402
011606 104002
011610 000401

011612 122767 000053 166564
011620 001402
011622 104000
011624 000402

TST53: SCOPE
        JSR      R5,      PUSHR      ;PUSH 4 WORDS ONTO R0 STACK, SET PRIORITY
        .WORD   127652,125252      ;SECOND OPERAND ON TOP
        .WORD   167452,125251      ;FIRST OPERAND ON BOTTOM
        .WORD   111                ;PROCESSOR PRIORITY LEVEL
        .WORD   TRAPER, 340        ;FIS TRAP VECTOR
        MOV     #STACK0,R0        ;SHECK STACK POINTER

        NOP
        FDIV   R0                ;FLOATING DIVIDE ON THE R0 STACK

        JSR      PC,      POPR       ;POP THE ANSWER
        MOV     R0,      SSP        ;SAVE "STACK POINTER"
        TSTB   SPSW              ;CHECK PS (EXCEPT T BIT)
        BEQ    .+6               ;BRANCH IF OK
        HLT    376               ;PS NOT EQUAL TO 000
        ;THE ERROR NUMBER IS 376

        CMP     #STACK4,SSP        ;CHECK THE STACK POINTER (R0)
        BEQ    .+6               ;BRANCH IF OK
        HLT    377               ;STACK POINTER (R0) NOT EQUAL TO #STACK4
        ;THE ERROR NUMBER IS 377

        CMP     #077777,ANS1       ;CHECK FIRST HALF OF ANSWER
        BEQ    .+6               ;BRANCH IF OK
        HLT+2  400               ;ANS1 NOT EQUAL TO 077777
        ;THE ERROR NUMBER IS 400

        CMP     #177776,ANS2       ;CHECK SECOND HALF OF ANSWER
        BEQ    .+6               ;BRANCH IF OK
        HLT+2  401               ;ANS2 NOT EQUAL TO 177776
        ;THE ERROR NUMBER IS 401

        CMPB   #53,      $TESTN    ;CHECK THE TEST NUMBER
        BEQ    .+6               ;BRANCH IF OK
        HLT    402               ;WRONG TEST! PC MUST HAVE FOULED UP.
        ;THE ERROR NUMBER IS 402
    
```

2503
2502
2501
2500
2499
2498
2497
2496
2495
2494
2493
2492
2491
2490
2489
2488
2487
2486
2485
2484
2483
2482
2481
2480
2479
2478
2477
2476
2475
2474
2473
2472
2471
2470
2469
2468
2467
2466
2465
2464
2463
2462
2461
2460
2459
2458

```
*****
:TEST 54:      FDIV (LSI-11 FLOATING DIVIDE INSTRUCTION)
:      167452,125252 / 027652,125253 = 177777,177777
:      PS = 210,      STACK POINTER = SP
*****
```

011626	104400				SCOPE				
011630	004567	004164			TST54: JSR	RS,	PUSHS		; PUSH 4 WORDS ONTO STACK, SET PRIORITY
011634	027652	125253			.WORD	027652,	125253		; SECOND OPERAND ON TOP
011640	167452	125252			.WORD	167452,	125252		; FIRST OPERAND ON BOTTOM
011644	000300				.WORD	300			; PROCESSOR PRIORITY LEVEL
011646	016442	000340			.WORD	TRAPER,	340		; FIS TRAP VECTOR
011652	000240				NOP				
011654	075036				FDIV	SP			; FLOATING DIVIDE ON THE STACK
011656	004767	004176			JSR	PC,	POPS		; POP THE ANSWER
011662	022706	000600			CMP	#BEGIN,	SP		; CHECK THE STACK POINTER
011666	001405				BEQ	TSAS4			; BRANCH IF OK
011670	012706	000600			MOV	#BEGIN,	SP		; RESTORE STACK POINTER
011674	104000				HLT				; STACK POINTER FOULED UP
011676	000403				403				; THE ERROR NUMBER IS 403
011700	000422				BR	END54			; SKIP REST OF TEST
011702	122767	000210	166522		TSAS4: CMPB	#210,	SPSW		; CHECK PS (EXCEPT T BIT)
011710	001402				BEQ	+.6			; BRANCH IF OK
011712	104000				HLT				; PS NOT EQUAL TO 210
011714	000404				404				; THE ERROR NUMBER IS 404
011716	022767	177777	166512		CMP	#177777,	ANS1		; CHECK FIRST HALF OF ANSWER
011724	001402				BEQ	+.6			; BRANCH IF OK
011726	104002				HLT+2				; ANS1 NOT EQUAL TO 177777
011730	000405				405				; THE ERROR NUMBER IS 405
011732	022767	177777	166500		CMP	#177777,	ANS2		; CHECK SECOND HALF OF ANSWER
011740	001402				BEQ	+.6			; BRANCH IF OK
011742	104002				HLT+2				; ANS2 NOT EQUAL TO 177777
011744	000406				406				; THE ERROR NUMBER IS 406
011746	122767	000054	166430		END54: CMPB	#54,	STESTN		; CHECK THE TEST NUMBER
011754	001402				BEQ	+.6			; BRANCH IF OK
011756	104000				HLT				; WRONG TEST! PC MUST HAVE FOULED UP.
011760	000407				407				; THE ERROR NUMBER IS 407

2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549

TEST 55: FDIV (LSI-11 FLOATING DIVIDE INSTRUCTION)
125252 125252 / 065252 125252 = 100200,000000
PS = 210 STACK POINTER = R2

011762	104400			SCOPE				
011764	004567	004202		TST55: JSR	R5,	PUSHR		; PUSH 4 WORDS ONTO R2 STACK, SET PRIORITY
011770	065252	125252		.WORD	065252,	125252		; SECOND OPERAND ON TOP
011774	125252	125252		.WORD	125252,	125252		; FIRST OPERAND ON BOTTOM
012000	000217			.WORD	217			; PROCESSOR PRIORITY LEVEL
012002	016442	000340		.WORD	TRAPER,	340		; FIS TRAP VECTOR
012006	012702	000510		MOV	#STACK0,	R2		; CHECK STACK POINTER
012012	000240			NOP				
012014	075032			FDIV	R2			; FLOATING DIVIDE ON THE R2 STACK
012016	004767	004202		JSR	PC,	POPR		; POP THE ANSWER
012022	010267	166406		MOV	R2,	SSP		; SAVE "STACK POINTER"
012026	122767	000210	166376	CMPB	#210,	SPSW		; CHECK PS (EXCEPT T BIT)
012034	001402			BEQ	+.6			; BRANCH IF OK
012036	104000			HLT				; PS NOT EQUAL TO 210
012040	000410			410				; THE ERROR NUMBER IS 410
012042	022767	000514	166364	CMP	#STACK4,	SSP		; CHECK THE STACK POINTER (R2)
012050	001402			BEQ	+.6			; BRANCH IF OK
012052	104000			HLT				; STACK POINTER (R2) NOT EQUAL TO #STACK4
012054	000411			411				; THE ERROR NUMBER IS 411
012056	022767	100200	166352	CMP	#100200,	ANS1		; CHECK FIRST HALF OF ANSWER
012064	001402			BEQ	+.6			; BRANCH IF OK
012066	104002			HLT+2				; ANS1 NOT EQUAL TO 100200
012070	000412			412				; THE ERROR NUMBER IS 412
012072	005767	166342		TST	ANS2			; CHECK SECOND HALF OF ANSWER
012076	001402			BEQ	+.6			; BRANCH IF OK
012100	104002			HLT+2				; ANS2 NOT EQUAL TO 000000
012102	000413			413				; THE ERROR NUMBER IS 413
012104	122767	000055	166272	END55: CMPB	#55,	\$TESTN		; CHECK THE TEST NUMBER
012112	001402			BEQ	+.6			; BRANCH IF OK
012114	104000			HLT				; WRONG TEST! PC MUST HAVE FOULED UP.
012116	000414			414				; THE ERROR NUMBER IS 414

2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595

```

:*****
:TEST 56:      FDIV (LSI-11 FLOATING DIVIDE INSTRUCTION)
:      000000,000000 / 140670,123456 = 000000,000000
:      PS = 004,      STACK POINTER = R3
:*****
    
```

```

TST56:  SCOPE
        JSR      R5,      PUSHR      ;PUSH 4 WORDS ONTO R3 STACK, SET PRIORITY
        .WORD    140670,123456      ;SECOND OPERAND ON TOP
        .WORD    000000,000000      ;FIRST OPERAND ON BOTTOM
        .WORD    105                ;PROCESSOR PRIORITY LEVEL
        .WORD    TRAPER, 340        ;FIS TRAP VECTOR
        MOV      #STACK0,R3        ;CHECK STACK POINTER

        NOP
        FDIV     R3                ;FLOATING DIVIDE ON THE R3 STACK

        JSR      PC,      POPR      ;POP THE ANSWER
        MOV      R3,      SSP      ;SAVE "STACK POINTER"
        CMPB    #004,     SPSW     ;CHECK PS (EXCEPT T BIT)
        BEQ     .+6            ;BRANCH IF OK
        HLT     104000        ;PS NOT EQUAL TO 004
        415                ;THE ERROR NUMBER IS 415

        CMP     #STACK4,SSP      ;CHECK THE STACK POINTER (R3)
        BEQ     .+6            ;BRANCH IF OK
        HLT     104000        ;STACK POINTER (R3) NOT EQUAL TO #STACK4
        416                ;THE ERROR NUMBER IS 416

        TST     ANS1           ;CHECK FIRST HALF OF ANSWER
        BEQ     .+6            ;BRANCH IF OK
        HLT+2  104002        ;ANS1 NOT EQUAL TO 000000
        417                ;THE ERROR NUMBER IS 417

        TST     ANS2           ;CHECK SECOND HALF OF ANSWER
        BEQ     .+6            ;BRANCH IF OK
        HLT+2  104002        ;ANS2 NOT EQUAL TO 000000
        420                ;THE ERROR NUMBER IS 420

        CMPB    #56,         $TESTN ;CHECK THE TEST NUMBER
        BEQ     .+6            ;BRANCH IF OK
        HLT     104000        ;WRONG TEST! PC MUST HAVE FOULED UP.
        421                ;THE ERROR NUMBER IS 421
    
```

2596
2597
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629
2630
2631
2632
2633
2634
2635
2636
2637
2638
2639
2640
2641
2642
2643
2644
2645
2646
2647
2648
2649
2650
2651

```
*****
:TEST 57:      FDIV (LSI-11 FLOATING DIVIDE INSTRUCTION)
:              102500,146000 / 104000,104000 = 036700,000000
:              PS = 200,      STACK POINTER = PC
*****
```

```
TST57:  SCOPE
        JSR   R5,      PUSH7      ;PUSH 4 WORDS ONTO STACK, SET PRIORITY
        .WORD STK57           ;TOP OF STACK
        .WORD 104000,104000      ;SECOND OPERAND ON TOP
        .WORD 102500,146000     ;FIRST OPERAND ON BOTTOM
        .WORD 357              ;PROCESSOR PRIORITY LEVEL
        .WORD TRAPER,340       ;FIS TRAP VECTOR

STK57:  NOP
        FDIV  PC              ;FLOATING DIVIDE ON FOLLOWING 4 WORDS
        104000                ;SHOULD CONTAIN 104000
        104000                ;SHOULD CONTAIN 104000
        102500                ;BEFORE FDIV, 102500; AFTER, 036700
        146000                ;BEFORE FDIV, 146000; AFTER, 000000

        JSR   PC,      POP7      ;POP THE ANSWER
        CMPB #200,    SPSW      ;CHECK PS (EXCEPT T BIT)
        BEQ  .+6           ;BRANCH IF OK
        HLT  422          ;PS NOT EQUAL TO 200
                          ;THE ERROR NUMBER IS 422

        CMP  #104000,ANS1      ;CHECK FIRST HALF OF INPUT DATA (STK57)
        BEQ  .+6           ;BRANCH IF OK
        HLT+2              ;ANS1 NOT EQUAL TO 104000
        423              ;THE ERROR NUMBER IS 423

        CMP  #104000,ANS2      ;CHECK SECOND HALF OF INPUT DATA (STK57+2)
        BEQ  .+6           ;BRANCH IF OK
        HLT+2              ;ANS2 NOT EQUAL TO 104000
        424              ;THE ERROR NUMBER IS 424

        CMP  #036700,ANS3      ;CHECK FIRST HALF OF ANSWER
        BEQ  .+6           ;BRANCH IF OK
        HLT+4              ;ANS3 NOT EQUAL TO 036700
        425              ;THE ERROR NUMBER IS 425

        TST  ANS4            ;CHECK SECOND HALF OF ANSWER
        BEQ  .+6           ;BRANCH IF OK
        HLT+4              ;ANS4 NOT EQUAL TO 000000
        426              ;THE ERROR NUMBER IS 426

ENDS7:  CMPB #57,      $TESTN   ;CHECK THE TEST NUMBER
        BEQ  .+6           ;BRANCH IF OK
        HLT  427          ;WRONG TEST! PC MUST HAVE FOULED UP.
                          ;THE ERROR NUMBER IS 427
```

TEST FLOATING DIV. INSTRUCTION WITH UNDERFLOW

```

2652 ;TEST 60:          FDIV (LSI-11 FLOATING DIVIDE INSTRUCTION)
2653 ;               025252,125251 / 065252,125252 ==> UNDERFLOW
2654 ;               PS(ON STACK) = 012,    STACK POINTER = R1
2655 ;*****
2656 ;
2657 012430 104400          SCOPE
2658 012432 004567 003534  TST60: JSR      R5,    PUSHR  ;PUSH 4 WORDS ONTO R1 STACK, SET PRIORITY
2659 012436 065252 125252   .WORD 065252,125252 ;SECOND OPERAND ON TOP
2660 012442 025252 125251   .WORD 025252,125251 ;FIRST OPERAND ON BOTTOM
2661 012446 000015         .WORD 015          ;PROCESSOR PRIORITY LEVEL
2662 012450 012502 000300   .WORD ISR60, 300    ;FIS TRAP VECTOR
2663 012454 012701 000510   MOV     #STACK0,R1 ;SET UP R1 AS STACK POINTER
2664 ;
2665 012460 000240          NOP
2666 012462 075031         FDIV  R1          ;FLOATING DIVIDE ON THE R1 STACK
2667 ;
2668 012464 004767 003534  RTA60: JSR      PC,    POPR   ;POP THE "ANSWER"
2669 012470 010167 165740   MOV     R1,    SSP    ;SAVE STACK POINTER (R1)
2670 012474 104002         HLT+2  ;FIS TRAP DIDN'T OCCURE!
2671 012476 000430         430     ;THE ERROR NUMBER IS 430
2672 012500 000464         BR      END60
2673 ;
2674 012502 004767 003546  ISR60: JSR      PC,    POPER  ;POP ALL DATA OFF THE STACKS
2675 012506 010167 165722   MOV     R1,    SSP    ;SAVE STACK POINTER (R1)
2676 012512 122767 000300 165712  CMPB   #300,    SPSW   ;CHECK PS AFTER FIS TRAP
2677 012520 001402         BEQ     .+6          ;BRANCH IF OK
2678 012522 104000         HLT     ;PS AFTER FIS TRAP NOT EQUAL TO 300
2679 012524 000431         431     ;THE ERROR NUMBER IS 431
2680 ;
2681 012526 022767 000510 165700  CMP     #STACK0,SSP   ;CHECK THE STACK POINTER (R1)
2682 012534 001402         BEQ     .+6          ;BRANCH IF OK
2683 012536 104000         HLT     ;STACK POINTER (R1) NOT EQUAL TO #STACK0
2684 012540 000432         432     ;THE ERROR NUMBER IS 432
2685 ;
2686 012542 022767 012464 165666  CMP     #RTA60,ANS1   ;CHECK FIS TRAP RETURN ADDRESS
2687 012550 001402         BEQ     .+6          ;BRANCH IF OK
2688 012552 104001         HLT+1  ;FIS TRAP AT WRONG ADDRESS
2689 012554 000433         433     ;THE ERROR NUMBER IS 433
2690 ;
2691 012556 022767 000012 165654  CMP     #012,    ANS2  ;CHECK PS BEFORE FIS TRAP
2692 012564 001402         BEQ     .+6          ;BRANCH IF OK
2693 012566 104002         HLT+2  ;PS AT FIS TRAP TIME NOT 012
2694 012570 000434         434     ;THE ERROR NUMBER IS 434
2695 ;
2696 012572 022767 065252 165642  CMP     #065252,ANS3  ;CHECK DATA FROM THE STACK
2697 012600 001402         BEQ     .+6          ;BRANCH IF OK
2698 012602 104004         HLT+4  ;DATA ON STACK (065252) CHANGED
2699 012604 000435         435     ;THE ERROR NUMBER IS 435
2700 ;
2701 012606 022767 125252 165630  CMP     #125252,ANS4  ;CHECK DATA FROM STACK
2702 012614 001402         BEQ     .+6          ;BRANCH IF OK
2703 012616 104004         HLT+4  ;DATA ON STACK (125252) CHANGED
2704 012620 000436         436     ;THE ERROR NUMBER IS 436
2705 ;
2706 012622 022767 025252 165616  CMP     #025252,ANS5  ;CHECK DATA FROM STACK
2707 012630 001402         BEQ     .+6          ;BRANCH IF OK
    
```



```

2722
2723
2724
2725
2726
2727
2728
2729 012666 104400
2730 012670 004567 003276
2731 012674 127652 125252
2732 012700 067452 125252
2733 012704 000242
2734 012706 012740 000357
2735 012712 012704 000510
2736
2737 012716 000240
2738 012720 075034
2739
2740 012722 004767 003276
2741 012726 010467 165502
2742 012732 104002
2743 012734 000442
2744 012736 000464
2745
2746 012740 004767 003310
2747 012744 010467 165464
2748 012750 122767 000357 165454
2749 012756 001402
2750 012760 104000
2751 012762 000443
2752
2753 012764 022767 000510 165442
2754 012772 001402
2755 012774 104000
2756 012776 000444
2757
2758 013000 022767 012722 165430
2759 013006 001402
2760 013010 104001
2761 013012 000445
2762
2763 013014 022767 000202 165416
2764 013022 001402
2765 013024 104002
2766 013026 000446
2767
2768 013030 022767 127652 165404
2769 013036 001402
2770 013040 104004
2771 013042 000447
2772
2773 013044 022767 125252 165372
2774 013052 001402
2775 013054 104004
2776 013056 000450
2777

```

```

*****
:TEST 61:      FDIV (LSI-11 FLOATING DIVIDE INSTRUCTION)
:              067452,125252 / 127652,125252 ==> OVERFLOW
:              PS(ON STACK) = 202,      STACK POINTER = R4
*****
TST61:  SCOPE
        JSR   R5,      PUSHR      ;PUSH 4 WORDS ONTO R4 STACK, SET PRIORITY
        .WORD 127652,125252      ;SECOND OPERAND ON TOP
        .WORD 067452,125252      ;FIRST OPERAND ON BOTTOM
        .WORD 242                ;PROCESSOR PRIORITY LEVEL
        .WORD ISR61, 357          ;FIS TRAP VECTOR
        MOV   #STACK0,R4        ;SET UP R4 AS STACK POINTER

RTA61:  JSR   PC,      POPR        ;POP THE "ANSWER"
        MOV   R4,      $SP        ;SAVE STACK POINTER (R4)
        HLT+2                ;FIS TRAP DIDN'T OCCURE!
        442                    ;THE ERROR NUMBER IS 442
        BR    END61

ISR61:  JSR   PC,      POPER        ;POP ALL DATA OFF THE STACKS
        MOV   R4,      $SP        ;SAVE STACK POINTER (R4)
        CMPB  #357,     $PSW        ;CHECK PS AFTER FIS TRAP
        BEQ   .+6              ;BRANCH IF OK
        HLT                ;PS AFTER FIS TRAP NOT EQUAL TO 357
        443                    ;THE ERROR NUMBER IS 443

        CMP   #STACK0,$SP        ;CHECK THE STACK POINTER (R4)
        BEQ   .+6              ;BRANCH IF OK
        HLT                ;STACK POINTER (R4) NOT EQUAL TO #STACK0
        444                    ;THE ERROR NUMBER IS 444

        CMP   #RTA61,ANS1        ;CHECK FIS TRAP RETURN ADDRESS
        BEQ   .+6              ;BRANCH IF OK
        HLT+1                ;FIS TRAP AT WRONG ADDRESS
        445                    ;THE ERROR NUMBER IS 445

        CMP   #202,     ANS2        ;CHECK PS BEFORE FIS TRAP
        BEQ   .+6              ;BRANCH IF OK
        HLT+2                ;PS AT FIS TRAP TIME NOT 202
        446                    ;THE ERROR NUMBER IS 446

        CMP   #127652,ANS3        ;CHECK DATA FROM THE STACK
        BEQ   .+6              ;BRANCH IF OK
        HLT+4                ;DATA ON STACK (127652) CHANGED
        447                    ;THE ERROR NUMBER IS 447

        CMP   #125252,ANS4        ;CHECK DATA FROM STACK
        BEQ   .+6              ;BRANCH IF OK
        HLT+4                ;DATA ON STACK (125252) CHANGED
        450                    ;THE ERROR NUMBER IS 450

```



```

2794
2795
2796
2797
2798
2799
2800
2801 013124 104400
2802 013126 004567 003040
2803 013132 100125 125252
2804 013136 052525 052525
2805 013142 000047
2806 013144 013176 000113
2807 013150 012705 000510
2808
2809 013154 000240
2810 013156 075035
2811
2812 013160 004767 003040
2813 013164 010567 165244
2814 013170 104002
2815 013172 000454
2816 013174 000464
2817
2818 013176 004767 003052
2819 013202 010567 165226
2820 013206 122767 000113 165216
2821 013214 001402
2822 013216 104000
2823 013220 000455
2824
2825 013222 022767 000510 165204
2826 013230 001402
2827 013232 104000
2828 013234 000456
2829
2830 013236 022767 013160 165172
2831 013244 001402
2832 013246 104001
2833 013250 000457
2834
2835 013252 022767 000013 165160
2836 013260 001402
2837 013262 104002
2838 013264 000460
2839
2840 013266 022767 100125 165146
2841 013274 001402
2842 013276 104004
2843 013300 000461
2844
2845 013302 022767 125252 165134
2846 013310 001402
2847 013312 104004
2848 013314 000462
2849
    
```

```

*****
:TEST 62:          FDIV (LSI-11 FLOATING DIVIDE INSTRUCTION)
:                052525,052525 / 100125,125252 ==> DIVIDE BY ZERO
:                PS(ON STACK) = 013,          STACK POINTER = R5
*****
    
```

```

TST62:  SCOPE
        JSR      R5,      PUSHR      ;PUSH 4 WORDS ONTO R5 STACK, SET PRIORITY
        .WORD   100125,125252      ;SECOND OPERAND ON TOP
        .WORD   052525,052525      ;FIRST OPERAND ON BOTTOM
        .WORD   047                ;PROCESSOR PRIORITY LEVEL
        .WORD   ISR62, 113         ;FIS TRAP VECTOR
        MOV     #STACK0,R5        ;SET UP R5 AS STACK POINTER
    
```

```

        NOP
        FDIV   R5                ;FLOATING DIVIDE ON THE R5 STACK
    
```

```

RTA62:  JSR      PC,      POPR      ;POP THE "ANSWER"
        MOV     R5,      $SP      ;SAVE STACK POINTER (R5)
        HLT+2  454              ;FIS TRAP DIDN'T OCCURE!
        BR     END62            ;THE ERROR NUMBER IS 454
    
```

```

ISR62:  JSR      PC,      POPER      ;POP ALL DATA OFF THE STACKS
        MOV     R5,      $SP      ;SAVE STACK POINTER (R5)
        CMPB   #113,      $PSW     ;CHECK PS AFTER FIS TRAP
        BEQ    .+6              ;BRANCH IF OK
        HLT    455              ;PS AFTER FIS TRAP NOT EQUAL TO 113
        HLT    455              ;THE ERROR NUMBER IS 455
    
```

```

        CMP     #STACK0,$SP      ;CHECK THE STACK POINTER (R5)
        BEQ    .+6              ;BRANCH IF OK
        HLT    456              ;STACK POINTER (R5) NOT EQUAL TO #STACK0
        HLT    456              ;THE ERROR NUMBER IS 456
    
```

```

        CMP     #RTA62, ANS1     ;CHECK FIS TRAP RETURN ADDRESS
        BEQ    .+6              ;BRANCH IF OK
        HLT+1  457              ;FIS TRAP AT WRONG ADDRESS
        HLT    457              ;THE ERROR NUMBER IS 457
    
```

```

        CMP     #013, ANS2      ;CHECK PS BEFORE FIS TRAP
        BEQ    .+6              ;BRANCH IF OK
        HLT+2  460              ;PS AT FIS TRAP TIME NOT 013
        HLT    460              ;THE ERROR NUMBER IS 460
    
```

```

        CMP     #100125,ANS3    ;CHECK DATA FROM THE STACK
        BEQ    .+6              ;BRANCH IF OK
        HLT+4  461              ;DATA ON STACK (100125) CHANGED
        HLT    461              ;THE ERROR NUMBER IS 461
    
```

```

        CMP     #125252,ANS4    ;CHECK DATA FROM STACK
        BEQ    .+6              ;BRANCH IF OK
        HLT+4  462              ;DATA ON STACK (125252) CHANGED
        HLT    462              ;THE ERROR NUMBER IS 462
    
```


TEST FLOATING DIV. INSTRUCTION FOR DIVIDE BY ZERO

2866
2867
2868
2869
2870
2871
2872
2873
2874
2875
2876
2877
2878
2879
2880
2881
2882
2883
2884
2885
2886
2887
2888
2889
2890
2891
2892
2893
2894
2895
2896
2897
2898
2899
2900
2901
2902
2903
2904
2905
2906
2907
2908
2909
2910
2911
2912
2913
2914
2915
2916
2917
2918
2919
2920
2921

013362 104400
013364 004567 002430
013370 000006 123456
013374 100052 052525
013400 000357
013402 013430 000311
013406 000240
013410 075036
013412 004767 002442
013416 104002
013420 000466
013422 012706 000600
013426 000464
013430 004767 002456
013434 022706 000600
013440 001405
013442 012706 000600
013446 104000
013450 000467
013452 000452
013454 122767 000311 164750
013462 001402
013464 104000
013466 000470
013470 022767 013412 164740
013476 001402
013500 104001
013502 000471
013504 022767 000213 164726
013512 001402
013514 104002
013516 000472
013520 022767 000006 164714
013526 001402
013530 104004
013532 000473

```

*****
:TEST 63:      FDIV (LSI-11 FLOATING DIVIDE INSTRUCTION)
:              100052,052525 / 000006,123456 ==> DIVIDE BY ZERO
:              PS(ON STACK) = 213,      STACK POINTER = SP
*****
    
```

```

TST63:  SCOPE
        JSR      RS,      PUSH5      ;PUSH 4 WORDS ONTO STACK, SET PRIORITY
        .WORD   000006,123456      ;SECOND OPERAND ON TOP
        .WORD   100052,052525      ;FIRST OPERAND ON BOTTOM
        .WORD   357                  ;PROCESSOR PRIORITY LEVEL
        .WORD   ISR63, 311          ;FIS TRAP VECTOR

RTA63:  JSR      PC,      POPS       ;POP THE "ANSWER"
        HLT+2
        466                          ;FIS TRAP DIDN'T OCCURE!
        MOV     #BEGIN, SP          ;THE ERROR NUMBER IS 466
        BR      END63              ;RESTORE THE STACK POINTER

ISR63:  JSR      PC,      POPES      ;POP ALL DATA OFF THE STACK
        CMP     #BEGIN, SP          ;CHECK THE STACK POINTER
        BEQ     ISA63              ;BRANCH IF OK
        MOV     #BEGIN, SP          ;RESTORE THE STACK POINTER
        HLT     467                  ;STACK POINTER FOULED UP
        BR      END63              ;THE ERROR NUMBER IS 467
        BR      END63              ;SKIP REST OF TEST

ISA63:  CMPB    #311,  SPSW         ;CHECK PS AFTER FIS TRAP
        BEQ     .+6                 ;BRANCH IF OK
        HLT     470                  ;PS AFTER FIS TRAP NOT EQUAL TO 311
        BR      END63              ;THE ERROR NUMBER IS 470

        CMP     #RTA63, ANS1        ;CHECK FIS TRAP RETURN ADDRESS
        BEQ     .+6                 ;BRANCH IF OK
        HLT+1  471                  ;FIS TRAP AT WRONG ADDRESS
        BR      END63              ;THE ERROR NUMBER IS 471

        CMP     #213,  ANS2         ;CHECK PS BEFORE FIS TRAP

        BEQ     .+6                 ;BRANCH IF OK
        HLT+2  472                  ;PS AT FIS TRAP TIME NOT 213
        BR      END63              ;THE ERROR NUMBER IS 472

        CMP     #000006,ANS3        ;CHECK DATA FROM THE STACK
        BEQ     .+6                 ;BRANCH IF OK
        HLT+4  473                  ;DATA ON STACK (000006) CHANGED
        BR      END63              ;THE ERROR NUMBER IS 473
    
```

2922	013534	022767	123456	164702	CMP	#123456,ANS4		;CHECK DATA FROM STACK
2923	013542	001402			BEQ	+.6		;BRANCH IF OK
2924	013544	104004			HLT+4			;DATA ON STACK (123456) CHANGED
2925	013546	000474			474			;THE ERROR NUMBER IS 474
2926								
2927	013550	022767	100052	164670	CMP	#100052,ANS5		;CHECK DATA FROMONG TK
2928	013556	001402			BEQ	+.6		;BRANCH IF OK
2929	013560	104006			HLT+6			;DATA ON STACK (100052) CHANGED
2930	013562	000475			475			;THE ERROR NUMBER IS 475
2931								
2932	013564	022767	052525	164656	CMP	#052525,ANS6		;CHECK DATA FROM STACK
2933	013572	001402			BEQ	+.6		;BRANCH IF OK
2934	013574	104006			HLT+6			;DATA ON STACK (052525) CHANGED
2935	013576	000476			476			;THE ERROR NUMBER IS 476
2936								
2937	013600	122767	000063	164576	CMPB	#63, STESTN		;CHECK THE TEST NUMBER
2938	013606	001402			BEQ	+.6		;BRANCH IF OK
2939	013610	104000			HLT			;WREST! PC MUST HAVE FOULED UP.
2940	013612	000477			477			;THE ERROR NUMBER IS 477
2941								
2942								

```

2943
2944
2945
2946
2947
2948
2949
2950
2951
2952 013614 104400
2953 013616 012704 000532
2954 013622 012744 107070
2955 013626 012744 134343
2956 013632 012744 065432
2957 013636 012744 032107
2958 013642 012744 123456
2959 013646 012744 045670
2960 013652 012744 125252
2961 013656 012744 135252
2962 013662 012744 016161
2963 013666 012744 040616
2964 013672
2965 013672 106427
2966
2967 013676 000240
2968 013700 075014
2969 013702 075034
2970 013704 075024
2971 013706 075004
2972
2973 013710
2974 013710 106767
2975 013714 042767 000020 164510
2976 013722 012467 164510
2977 013726 012467 164506
2978 013732 010467 164476
2979 013736 122767 000010 164466
2980 013744 001402
2981 013746 104000
2982 013750 000500
2983
2984 013752 022767 000532 164454
2985 013760 001402
2986 013762 104000
2987 013764 000501
2988
2989
2990 013766 022767 137201 164442
2991 013774 001402
2992 013776 104002
2993 014000 000502
2994
2995 014002 022767 115230 164430
2996 014010 001402
2997 014012 104002
2998 014014 000503

```

```

*****
TEST 64: TEST ALL INSTRUCTION TOGETHER
          032107,065432 * 045670,123456
          134343,107070 + ----- = 137201,115230
                      (135252,125252 - 040616,016161)
          PS=010, STACK POINTER=R4
*****

```

```

TST64: SCOPE
        MOV      #STAK10,R4      ;SET STACK POINTER
        MOV      #107070,-(R4)  ;LOAD DATA ONTO STACK
        MOV      #134343,-(R4)
        MOV      #065432,-(R4)
        MOV      #032107,-(R4)
        MOV      #123456,-(R4)
        MOV      #045670,-(R4)
        MOV      #125252,-(R4)
        MOV      #135252,-(R4)
        MOV      #016161,-(R4)
        MOV      #040616,-(R4)
        MTPS     #144            ;SET PROCESSOR STATUS
        .WORD    106400!..C

        NOP
        FSUB     R4              ;135252,125252-040616,016161=140616,017434
        FDIV     R4              ;045670,123456/140616,017434=145246,047065
        FMUL     R4              ;032107,065432*145246,047065=137201,106137
        FADD     R4              ;134343,107070+137201,106137=137201,115230

        MFPS     $PSW           ;SAVE FINAL PS
        .WORD    106700!..C
        BIC      #20,$PSW      ;CLR T-BIT
        MOV      (R4)+,ANS1     ;SAVE FIRST HALF OF ANSWER
        MOV      (R4)+,ANS2     ;SAVE SECOND HALF OF ANSWER
        MOV      R4,$SP        ;SAVE STACK POINTER
        CMPB     #010,$PSW     ;CHECK PS (EXCEPT T BIT)
        BEQ      .+6           ;BRANCH IF OK
        HLT     ;PS NOT EQUAL TO 010
        500     ;THE ERROR NUMBER IS 500

        CMP      #STAK10,$SP   ;CHECK THE STACK POINTER (R4)
        BEQ      .+6           ;BRANCH IF OK
        HLT     ;STACK POINTER (R4) NOT EQUAL TO THE
        501     ;THE ERROR NUMBER IS 501
        ;ADDRESS OF STAK10

        CMP      #137201,ANS1  ;CHECK FIRST HALF OF ANSWER
        BEQ      .+6           ;BRANCH IF OK
        HLT+2   ;ANS1 NOT EQUAL TO 137201
        502     ;THE ERROR NUMBER IS 502

        CMP      #115230,ANS2  ;CHECK SECOND HALF OF ANSWER
        BEQ      .+6           ;BRANCH IF OK
        HLT+2   ;ANS2 NOT EQUAL TO 115230
        503     ;THE ERROR NUMBER IS 503

```



```

3005
3006
3007
3008
3009
3010
3011 014032 104400
3012 014034 012737 014124 000004 TST65: MOV #ISR65, 2#4 ;SET UP ADDRESS TRAP VECTOR
3013 014042 012737 000340 000006 MOV #340, 2#6
3014 014050 004567 002116 JSR R5, PUSHR ;PUSH 4 WORDS ONTO R2 STACK, SET PRIORITY
3015 014054 070707 016161 .WORD 070707,016161 ;SECOND OPERAND ON TOP
3016 014060 146314 143434 .WORD 146314,143434 ;FIRST OPERAND ON BOTTOM
3017 014064 000143 .WORD 143 ;PROCESSOR PRIORITY LEVEL
3018 014066 016442 000340 .WORD TRAPER, 340 ;FIS TRAP VECTOR
3019 014072 MTPS #143 ;SET PROCESSOR STATUS
3020 014072 106427 .WORD 106400!..C
3021 014076 012702 177777 MOV #177777,R2 ;SET UP R2 AS STACK POINTER
3022
3023 014102 000240 NOP
3024 014104 075002 FADD R2 ;FLOATING ADD ON THE R2 STACK
3025
3026 014106 004767 002112 RTA65: JSR PC, POPR ;POP THE "ANSWER"
3027 014112 010267 164316 MOV R2, SSP ;SAVE STACK POINTER (R2)
3028 014116 104002 HLT+2 ;FIS TRAP DIDN'T OCCURE!
3029 014120 000505 505 ;THE ERROR NUMBER IS 505
3030 014122 000464 BR END65
3031
3032 014124 004767 002124 ISR65: JSR PC, POPER ;POP ALL DATA OFF THE STACKS
3033 014130 010267 164300 MOV R2, SSP ;SAVE STACK POINTER (R2)
3034 014134 122767 000340 164270 CMPB #340, $PSW ;CHECK PS AFTER ADR. ERR. TRAP
3035 014142 001402 BEQ .+6 ;BRANCH IF OK
3036 014144 104000 HLT ;PS AFTER TRAP NOT EQUAL TO 340
3037 014146 000506 506 ;THE ERROR NUMBER IS 506
3038
3039 014150 022767 177777 164256 CMP #177777,$SP ;CHECK THE STACK POINTER (R2)
3040 014156 001402 BEQ .+6 ;BRANCH IF OK
3041 014160 104000 HLT ;STACK POINTER (R2) NOT EQUAL TO #177777
3042 014162 000507 507 ;THE ERROR NUMBER IS 507
3043
3044 014164 022767 014106 164244 CMP #RTA65, ANS1 ;CHECK FIS TRAP RETURN ADDRESS
3045 014172 001402 BEQ .+6 ;BRANCH IF OK
3046 014174 104001 HLT+1 ;FIS TRAP AT WRONG ADDRESS
3047 014176 000510 510 ;THE ERROR NUMBER IS 510
3048
3049 014200 022767 000151 164232 CMP #151, ANS2 ;CHECK PS BEFORE FIS TRAP
3050 014206 001402 BEQ .+6 ;BRANCH IF OK
3051 014210 104002 HLT+2 ;PS AT FIS TRAP TIME NOT 151
3052 014212 000511 511 ;THE ERROR NUMBER IS 511
3053
3054 014214 022767 070707 164220 CMP #070707,ANS3 ;CHECK DATA FROM THE STACK
3055 014222 001402 BEQ .+6 ;BRANCH IF OK
3056 014224 104004 HLT+4 ;DATA ON STACK (070707) CHANGED
3057 014226 000512 512 ;THE ERROR NUMBER IS 512
3058
3059 014230 022767 016161 164206 CMP #016161,ANS4 ;CHECK DATA FROM STACK
3060 014236 001402 BEQ .+6 ;BRANCH IF OK

```



```

3077
3078
3079
3080
3081
3082
3083 014310 104400
3084 014312 012737 014402 000004 TST66: SCOPE
3085 014320 012737 000340 000006 MOV #ISR66, #4 ;SET UP ADDRESS TRAP VECTOR
3086 014326 004567 001640 JSR R5, #6 ;PUSH 4 WORDS ONTO R5 STACK, SET PRIORITY
3087 014332 065432 123456 .WORD 065432,123456 ;SECOND OPERAND ON TOP
3088 014336 037654 032107 .WORD 037654,032107 ;FIRST OPERAND ON BOTTOM
3089 014342 000202 .WORD 202 ;PROCESSOR PRIORITY LEVEL
3090 014344 016442 000340 .WORD TRAPER, 340 ;FIS TRAP VECTOR
3091 014350 MTPS #202 ;SET PROCESSOR STATUS
3092 014350 106427 .WORD 106400!..C
3093 014354 012705 160000 MOV #160000,R5 ;SET UP R5 AS STACK POINTER
3094
3095 014360 000240 NOP
3096 014362 075025 FMUL R5 ;FLOATING MULTIPLY ON THE R5 STACK
3097
3098 014364 RTA66: MFPS $PSW ;SAVE THE PSW
3099 014364 106767 .WORD 106700!..C
3100 014370 010567 164040 MOV R5, $SP ;SAVE STACK POINTER (R5)
3101 014374 104000 HLT ;FIS TRAP DIDN'T OCCURE!
3102 014376 000517 517 ;THE ERROR NUMBER IS 517
3103 014400 000434 BR END66
3104
3105 014402 004767 001646 ISR66: JSR PC, POPER ;POP ALL DATA OFF THE STACKS
3106 014406 010567 164022 MOV R5, $SP ;SAVE STACK POINTER (R5)
3107 014412 122767 000340 164012 CMPB #340, $PSW ;CHECK PS AFTER ADR. ERR. TRAP
3108 014420 001402 BEQ .+6 ;BRANCH IF OK
3109 014422 104000 HLT ;PS AFTER TRAP NOT EQUAL TO 340
3110 014424 000520 520 ;THE ERROR NUMBER IS 520
3111
3112 014426 022767 160000 164000 CMP #160000,$SP ;CHECK THE STACK POINTER (R5)
3113 014434 001402 BEQ .+6 ;BRANCH IF OK
3114 014436 104000 HLT ;STACK POINTER (R5) NOT EQUAL TO #160000
3115 014440 000521 521 ;THE ERROR NUMBER IS 521
3116
3117 014442 022767 014364 163766 CMP #RTA66, ANS1 ;CHECK FIS TRAP RETURN ADDRESS
3118 014450 001402 BEQ .+6 ;BRANCH IF OK
3119 014452 104001 HLT+1 ;FIS TRAP AT WRONG ADDRESS
3120 014454 000522 522 ;THE ERROR NUMBER IS 522
3121
3122 014456 022767 000210 163754 CMP #210, ANS2 ;CHECK PS BEFORE FIS TRAP
3123 014464 001402 BEQ .+6 ;BRANCH IF OK
3124 014466 104002 HLT+2 ;PS AT FIS TRAP TIME NOT 210
3125 014470 000523 523 ;THE ERROR NUMBER IS 523
3126
3127 014472 122767 000066 163704 END66: CMPB #66, $TESTN ;CHECK THE TEST NUMBER
3128 014500 001402 BEQ .+6 ;BRANCH IF OK
3129 014502 104000 HLT ;WRONG TEST! PC MUST HAVE FOULED UP.
3130 014504 000524 524 ;THE ERROR NUMBER IS 524

```

```

3131 014506 012737 000006 000004      MOV    #6,    2#4      ;RESTORE TIME-OUT VECTOR
3132 014514 005037 000006          CLR    2#6
3133 014520 012767 000003 164010      MOV    #3,    TIMES   ;REDUCE NUMBER OF ITERATIONS
3134
3135 ;*****
3136 ;TEST 67:      TEST THAT FIS ABORTS PROPERLY WHEN INTERRUPTED
3137 ;             035700,143235 + 000177,134543 = 035700,143235
3138 ;             PS = .PS,      STACK POINTER = R1
3139 ;*****
3140
3141 014526 104400          SCOPE
3142 014530 132737 000040 000421  TST67:  BITB    #40,2#SENVM
3143 014536 001174          BNE    END67+2      ;EXIT THIS TEST IF BIT 5 OF SENVM IS HIGH
3144 014540 013704 000546          MOV    2#TTYOUT,R4
3145 014544 012724 014646          MOV    #ISR67,(R4)+ ;SET UP TELEPRINTER INTERUPT VECTOR
3146 014550 012714 000340          MOV    #340,(R4)
3147 014554 032737 004000 000422      BIT    #SW11,2#SSWREG ;TEST FOR ITERATIONS
3148 014562 001005          BNE    1$          ;BRANCH TO AVOID HANG UP
3149 014564 000004 000473          TYPE,  RETURN+1    ;RETURN+1 CAN BE REPLACED WITH THE ADDRESS OF RETURN
3150 ;             TO TYPE CARRIAGE RETURN, LINE FEED
3151 014570 012767 014576 163672      MOV    #.+6,    LADS  ;RESET LOOP ADDRESS
3152 014576 004567 001370 1$:      JSR    R5,    PUSHR  ;PUSH 4 WORDS ONTO R1 STACK, SET PRIORITY
3153 014602 000177 134543          .WORD 000177,134543 ;SECOND OPERAND ON TOP
3154 014606 035700 143235          .WORD 035700,143235 ;FIRST OPERAND ON BOTTOM
3155 014612 000143          .WORD 143          ;PROCESSOR PRIORITY LEVEL
3156 014614 016442 000340          .WORD TRAPER, 340  ;FIS TRAP VECTOR
3157 014620 012701 000510          MOV    #STACK0,R1  ;SET UP STACK POINTER
3158 014624 012767 000030 163702      MOV    #30,TEMP
3159 014632 112777 000100 163710      MOVB   #100,    2#STPS ;SET TTY INTERUPT ENABLE
3160
3161 014640 075001          RTA67:  FADD   R1      ;FLOATING ADD ON THE STACK
3162 014642 024141          CMP    -(R1), -(R1) ;RESET THE STACK POINTER FOR NEXT PASS
3163 014644 000775          BR     RTA67      ;REPEAT UNTIL INTERRUPTED
3164
3165 014646 105077 163676          ISR67:  CLRB   2#STPS    ;CLEAR THE INTERUPT ENABLE
3166 014652 022716 014640          CMP    #RTA67,(SP) ;CHECK IF INTERUPT AT FIS INSTR.
3167 014656 001421          BEQ    3$          ;BRANCH IF IT DID
3168 014660 022766 014640 000004      CMP    #RTA67, 4(SP) ;CHECK FOR INTERUPT WITH T-BIT SET
3169 014666 001420          BEQ    4$          ;BRANCH IF IT DID
3170 014670 112777 000015 163654 1$:      MOVB   #15,    2#STPB ;CONTINUE TO TYPE "CR"
3171 014676 105777 163646 2$:      TSTB   2#STPS    ;LOOP HERE UNTILL DONE BIT COMES ON
3172 014702 100375          BPL    2$
3173 014704 112777 000015 163640          MOVB   #15,    2#STPB ;TYPE ANOTHER "CR"
3174 014712 012777 000100 163630          MOV    #100,    2#STPS ;SET TTY INTERUPT ENABLE
3175 014720 000002          RTI
3176
3177 014722 004767 001326 3$:      JSR    PC,    POPER  ;SAVE ALL THE STUFF ON THE STACK
3178 014726 000403          BR
3179
3180 014730 022626          4$:      CMP    (SP)+, (SP)+ ;RESET THE STACK TO IGNORE THE TRACE TRAP
3181 014732 004767 001322          JSR    PC,    POPER1 ;POP ALL THE STUFF OFF THE STACK
3182 014736 005746          5$:      TST   -(SP)      ;SAVE PSW FOR FUTURE RTI
3183 014740 012746 014640          MOV    #RTA67,-(SP) ;PLACE THE RTI ADDRESS BACK IN SP
3184 014744 022706 000574          CMP    #BEGIN-4,SP ;CHECK THE STACK POINTER
3185 014750 001407          BEQ    6$          ;BRANCH IF OK
3186 014752 010667 163456          MOV    SP,    SSP  ;SAVE FOR TYPING
    
```



```

3240
3241
3242
3243
3244
3245
3246
3247 015150 104400
3248 015152 132737 000040 000421 TST70: BITB #40,2#SENVM
3249 015160 001173 BNE END70+2 ;EXIT THIS TEST IF BIT 5 OF SENVM IS HIGH
3250 015162 013704 000546 MOV 2#TTYOUT,R4
3251 015166 012724 015270 MOV #ISR70,(R4)+ ;SET UP TELEPRINTER INTERUPT VECTOR
3252 015172 012714 000340 MOV #340,(R4)
3253 015176 032737 004000 000422 BIT #SW11,2#SSWREG ;TEST FOR ITERATIONS
3254 015204 001005 BNE 1$ ;BRANCH TO AVOID HANG UP
3255 015206 000004 000473 TYPE, RETURN+1 ;RETURN+1 CAN BE REPLACED WITH THE ADDRESS OF RETURN
3256
3257 015212 012767 015220 163250 1$: MOV #.+6, LADS ;RESET LOOP ADDRESS
3258 015220 004567 000746 JSR R5, PUSHR ;PUSH 4 WORDS ONTO R0 STACK, SET PRIORITY
3259 015224 040200 000000 .WORD 040200,000000 ;SECOND OPERAND ON TOP
3260 015230 107070 070707 .WORD 107070,070707 ;FIRST OPERAND ON BOTTOM
3261 015234 000100 .WORD 100 ;PROCESSOR PRIORITY LEVEL
3262 015236 016442 000340 .WORD TRAPER,340 ;FIS TRAP VECTOR
3263 015242 012700 000510 MOV #STACK0,R0 ;SET UP STACK POINTER
3264 015246 012767 000030 163260 MOV #30,TEMP
3265 015254 112777 000100 163266 MOVB #100,2#STPS ;SET TTY INTERUPT ENABLE
3266
3267 015262 075020 RTA70: FMUL R0 ;FLOATING MULTIPLY ON THE STACK
3268 015264 024040 CMP -(R0), -(R0) ;RESET THE STACK POINTER FOR NEXT PASS
3269 015266 000775 BR RTA70 ;REPEAT UNTIL INTERUPTED
3270
3271 015270 105077 163254 ISR70: CLRB 2#STPS ;CLEAR THE INTERUPT ENABLE
3272 015274 022716 015262 CMP #RTA70,(SP) ;CHECK IF INTERUPT AT FIS INSTR.
3273 015300 001421 BEQ 3$ ;BRANCH IF IT DID
3274 015302 022766 015262 000004 CMP #RTA70,4(SP) ;CHECK FOR INTERUPT WITH T-BIT SET
3275 015310 001420 BEQ 4$ ;BRANCH IF IT DID
3276 015312 112777 000015 163232 1$: MOVB #15,2#STPB ;CONTINUE TO TYPE "CR"
3277 015320 105777 163224 2$: TSTB 2#STPS ;LOOP HERE UNTILL DONE BIT COMES ON
3278 015324 100375 BPL 2$
3279 015326 112777 000015 163216 MOVB #15,2#STPB ;TYPE ANOTHER "CR"
3280 015334 012777 000100 163206 MOV #100,2#STPS ;SET TTY INTERUPT ENABLE
3281 015342 000002 RTI
3282
3283 015344 004767 000704 3$: JSR PC, POPER ;SAVE ALL THE STUFF ON THE STACK
3284 015350 000403 BR 5$
3285
3286 015352 022626 4$: CMP (SP)+,(SP)+ ;RESET THE STACK TO IGNORE THE TRACE TRAP
3287 015354 004767 000700 JSR PC, POPER1 ;POP ALL THE STUFF OFF THE STACK
3288 015360 005746 5$: TST -(SP) ;SAVE PSW FOR FUTURE RTI
3289 015362 012746 015262 MOV #RTA70,-(SP) ;PLACE THE RTI ADDRESS BACK IN SP
3290 015366 022706 000574 CMP #BEGIN-4,SP ;CHECK THE STACK POINTER
3291 015372 001407 BEQ 6$ ;BRANCH IF OK
3292 015374 010667 163034 MOV SP,$SP ;SAVE FOR TYPING
3293 015400 012706 000574 MOV #BEGIN-4,SP ;RESTORE THE STACK POINTER
3294 015404 104000 HLT ;STACK POINTER FOULED UP
3295 015406 000536 536 ;THE ERROR NUMBER IS 536

```

K07

DVKACB MACY11 27(1006) 17-JAN-77 15:14 PAGE 79
 DVKACB.P11 13-JAN-77 15:28 INTERUPT ABORT TEST SECTION

SEQ 0087

```

3296 015410 000456          BR      END70          ;SKIP REST OF TEST
3297
3298 015412 010067 163016 65:  MOV      RD,      $SP          ;SAVE STACK POINTER
3299 015416 122767 000344 163006  CMPB    #344,    $PSW        ;CHECK PS AFTER INTERUPT
3300 015424 001402          BEQ     .+6          ;BRANCH IF OK
3301 015426 104000          HLT     ;PS AFTER INTERUPT NOT EQUAL TO LVLA
3302 015430 000537          537          ;THE ERROR NUMBER IS 537
3303
3304 015432 022767 000510 162774  CMP     #STACK0,$SP        ;CHECK THE STACK POINTER (RD)
3305 015440 001402          BEQ     .+6          ;BRANCH IF OK
3306 015442 104000          HLT     ;STACK POINTER (RD) NOT EQUAL TO #STACK0
3307 015444 000540          540          ;THE ERROR NUMBER IS 540
3308
3309 015446 022767 015262 162762  CMP     #RTA70, ANS1       ;CHECK FIS TRAP RETURN ADDRESS
3310 015454 001402          BEQ     .+6          ;BRANCH IF OK
3311 015456 104001          HLT+1  ;FIS TRAP AT WRONG ADDRESS
3312 015460 000541          541          ;THE ERROR NUMBER IS 541
3313
3314
3315 015462 022767 040200 162752  CMP     #040200,ANS3       ;CHECK DATA FROM THE STACK
3316 015470 001402          BEQ     .+6          ;BRANCH IF OK
3317 015472 104004          HLT+4  ;DATA ON STACK (040200) CHANGED
3318 015474 000542          542          ;THE ERROR NUMBER IS 542
3319
3320 015476 005767 162742          TST     ANS4            ;CHECK DATA FROM STACK
3321 015502 001402          BEQ     .+6          ;BRANCH IF OK
3322 015504 104004          HLT+4  ;DATA ON STACK (000000) CHANGED
3323 015506 000543          543          ;THE ERROR NUMBER IS 543
3324
3325 015510 022767 107070 162730  CMP     #107070,ANS5       ;CHECK DATA FROM STACK
3326 015516 001402          BEQ     .+6          ;BRANCH IF OK
3327 015520 104006          HLT+6  ;DATA ON STACK (107070) CHANGED
3328 015522 000544          544          ;THE ERROR NUMBER IS 544
3329
3330 015524 022767 070707 162716  CMP     #070707,ANS6       ;CHECK DATA FROM STACK
3331 015532 001402          BEQ     .+6          ;BRANCH IF OK
3332 015534 104006          HLT+6  ;DATA ON STACK (070707) CHANGED
3333 015536 000545          545          ;THE ERROR NUMBER IS 545
3334
3335 015540 005367 162770          DEC     TEMP           ;STAY IN THE LOOP FOR 30 TIMES
3336 015544 001262          BNE    1$
3337
3338 015546 022626          END70: CMP     (SP)+, (SP)+      ;RESTORE STACK POINTER TO 500
3339 015550 122767 000070 162626  CMPB   #70,      $TESTN    ;CHECK THE TEST NUMBER
3340 015556 001402          BEQ     .+6          ;BRANCH IF OK
3341 015560 104000          HLT     ;WRONG TEST! PC MUST HAVE FOULED UP.
3342 015562 000546          546          ;THE ERROR NUMBER IS 546
3343 015564          MTPS   #340
3344 015564 106427          .WORD  106400!...C
3345

```


INTERUPT ABORT TEST SECTION

```

3350 ;*****
3351
3352 .SBTTL END OF PASS ROUTINE
3353
3354 ;*INCREMENT THE PASS NUMBER ($PASS)
3355 ;*TYPE "END PASS"
3356 ;*IF THERES A MONITOR GO TO IT
3357 ;*IF THERE ISN'T JUMP TO BEGIN
3358 ;*IF IT IS DESIRED TO HAVE A BELL INDICATE THE "END OF PASS" LOCATION
3359 ;*$ENDMG CAN BE CHANGED TO 7.
3360
3361 SEOP:
3362 015604 104400 SCOPE
3363 015606 005267 162574 INC $PASS ;; INCREMENT THE PASS NUMBER
3364 015612 042767 100000 162566 BIC #100000,$PASS ;; DON'T ALLOW A NEG. NUMBER
3365 015620 005327 DEC (PC)+ ;; LOOP?
3366 015622 000001 SEOPCT: .WORD 1
3367 015624 003015 BGT $DOAGN ;; YES
3368 015626 012737 MOV (PC)+,2(PC)+ ;; RESTORE COUNTER
3369 015630 000001 SENDCT: .WORD 1
3370 015632 015622 SEOPCT
3371 015634 000004 015664 TYPE ,SENDMG ;; TYPE "END PASS"
3372 015640 $GET42:
3373
3374 015640 013700 000042 MOV 2#42,R0 ;; GET MONITOR ADDRESS
3375 015644 001405 BEQ $DOAGN ;; BRANCH IF NO MONITOR
3376 015646 000005 RESET ;; CLEAR THE WORLD
3377 015650 004710 SENDAD: JSR PC,(R0) ;; GO TO MONITOR
3378 015652 000240 NOP ;; SAVE ROOM
3379 015654 000240 NOP ;; FOR
3380 015656 000240 NOP ;; ACT11
3381 015660 SDOAGN:
3382 015660 000137 000600 JMP 2#BEGIN ;; RETURN
3383 015664 005015 047105 020104 SENDMG: .ASCII <15><12>/END PASS/
3384 015672 040520 051523
3385 015676 377 377 000 SENULL: .BYTE -1,-1,0 ;; NULL CHARACTER STRING
3386 015702 .EVEN
3387
3388 015630 000001 ENDCT: 1
    
```

```

3389 ;*****
3390
3391 .SBTTL SCOPE ROUTINE
3392
3393 015702 032737 000400 000422 SCOPES: BIT #SW08,2#SSWREG ;KILL LDUB OR LOOP ON SPEC. TEST
3394 015710 001404 BEQ 1$
3395 015712 123767 000422 162464 CMPB 2#SSWREG,$TESTN ;ON RIGHT TEST? *SW7-0*
3396 015720 001431 BEQ OVERS
3397 015722 032737 040000 000422 1$: BIT #SW14,2#SSWREG ;LOOP ON TEST
3398 015730 001023 BNE KITS
3399 015732 032737 004000 000422 BIT #SW11,2#SSWREG ;KILL ITERATIONS
3400 015740 001412 BEQ SVLADS
3401 015742 105767 162535 TSTB $ICNT
3402 015746 001404 BEQ 2$ ;BRANCH IF FIRST
3403 015750 126767 162562 162525 CMPB TIMES,$ICNT ;DONE?
3404 015756 001010 BNE KITS ;BRANCH IF NOT
3405 015760 112767 000001 162515 2$: MOVB #1,$ICNT ;FIRST ITERATION
3406 015766 105267 162412 SVLADS: INCB $TESTN ;COUNT TEST NUMBERS
3407 015772 011667 162472 MOV (6),$LADS ;SAVE LOOP ADDRESS
3408 015776 000002 RTI ;RETURN
3409
3410 016000 105267 162477 KITS: INCB $ICNT
3411 016004 005767 162460 OVERS: TST LADS ;FIRST ONE?
3412 016010 001766 BEQ SVLADS
3413 016012 016716 162452 MOV LADS,(6) ;FUDGE RETURN ADDRESS
3414 016016 000002 RTI ;FIXES PS
3415
    
```

```

3416
3417
3418 ;SUBROUTINE TO PUSH 4 WORDS ONTO THE STACK
3419 016020 005726
3420 016022 062705 000010
3421 016026 014546
3422 016030 014546
3423 016032 014546
3424 016034 014546
3425 016036 062705 000010
3426 016042
3427 016042 106425
3428 016044 005205
3429 016046 012577 162412
3430 016052 012577 162410
3431 016056 000115
3432
3433
3434 ;SUBROUTINE TO POP 2 WORDS OFF THE STACK
3435 ;ALSO SAVES THE PROCESSOR STATUS WORD (EXCEPT T BIT)
3436
3437 016060
3438 016060 106767
3439 016064 042767 000020 162340
3440 016072 012604
3441 016074 012667 162336
3442 016100 012667 162334
3443 016104 010667 162324
3444 016110 000114
3445
3446
3447 ;SUBROUTINE TO POP 6 WORDS OFF THE STACK.
3448 ;THE FIRST TWO WERE PUT ON BY THE ERROR TRAP,
3449 ;THE LAST FOUR WERE THE ORIGINAL INPUT DATA.
3450 ;ALSO SAVES THE PS AND STACK POINTER.
3451
3452 016112
3453 016112 106767
3454 016116 012604
3455 016120 012667 162312
3456 016124 011667 162310
3457 016130 042767 000020 162302
3458 016136 012746 016144
3459 016142 000002
3460 016144 012667 162272
3461 016150 012667 162270
3462 016154 012667 162266
3463 016160 012667 162264
3464 016164 010667 162244
3465 016170 000114
3466
3467 ;SUBROUTINE TO PUSH 4 WORDS ONTO THE STACK
3468
3469 016172 012704 000510
3470 016176 012524
3471 016200 012524

```

```

3472 016202 012524      MOV      (R5)+, (R4)+ ;
3473 016204 012524      MOV      (R5)+, (R4)+ ;
3474 016206             MTPS     (R5)+      ;SET THE PROCESSOR STATUS
3475 016206 106425      .WORD   106400!..C
3476 016210 005205      INC      R5
3477 016212 012577 162246  MOV      (R5)+, @FISVEC ;SET UP FIS ERROR TRAP VECTOR
3478 016216 012577 162244  MOV      (R5)+, @FISLVL ;TRAP STATUS
3479 016222 000205      RTS      R5          ;RETURN

```

```

3480
3481
3482 ;SUBROUTINE TO POP 2 WORDS OFF THE STACK
3483 ;ALSO SAVES THE PROCESSOR STATUS WORD (EXCEPT T BIT)
3484

```

```

3485 016224             POPR:  MFPS     SPSW          ;SAVE PROCESSOR STATUS WORD
3486 016224 106767      .WORD   106700!..C
3487 016230 042767 000020 162174  BIC      #20, SPSW      ;CLEAR T-BIT
3488 016236 016767 162252 162172  MOV      STACK4, ANS1   ;SAVE THE ANSWER
3489 016244 016767 162246 162166  MOV      STACK6, ANS2   ;
3490 016252 000207      RTS      PC

```

```

3491
3492
3493 ;SUBROUTINE TO POP 6 WORDS OFF THE STACKS.
3494 ;THE TWO OFF THE R6 STACK WERE PUT ON BY THE ERROR TRAP
3495 ;THE FOUR OFF THE SOFTWARE STACK WERE THE ORIGINAL INPUT DATA.
3496 ;ALSO SAVES THE PS AND STACK POINTER AFTER THE FIS TRAP.
3497

```

```

3498 016254             POPER:  MFPS     SPSW          ;SAVE PROCESSOR STATUS WORD
3499 016254 106767      .WORD   106700!..C
3500 016260 012667 000056             POPER1: MOV      (SP)+, SAVRTS   ;SAVE RTS ADDRESS
3501 016264 012667 162146             MOV      (SP)+, ANS1      ;SAVE RTI ADDRESS
3502 016270 011667 162144             MOV      (SP), ANS2      ;SAVE RTI STATUS
3503 016274 042767 000020 162136  BIC      #20, ANS2      ;CLEAR THE T-BIT
3504 016302 012746 016310             MOV      #1$, -(SP)
3505 016306 000002      RTI
3506 016310 016767 162174 162124 1$:  MOV      STACK0, ANS3   ;RESTORE PROCESSOR STATUS
3507 016316 016767 162170 162120  MOV      STACK2, ANS4   ;SAVE DATA
3508 016324 016767 162164 162114  MOV      STACK4, ANS5   ;
3509 016332 016767 162160 162110  MOV      STACK6, ANS6   ;
3510 016340 000137      JMP      @PC+          ;SIMULATED RTS
3511 016342 000000      SAVRTS: 0

```

```

3512
3513 ;SUBROUTINE TO PUSH 4 WORDS ONTO THE PC STACK
3514

```

```

3515 016344 012504      PUSH7:  MOV      (R5)+, R4      ;SET R4 TO STACK
3516 016346 012524      MOV      (R5)+, (R4)+   ;PUT DATA ON STACK
3517 016350 012524      MOV      (R5)+, (R4)+   ;
3518 016352 012524      MOV      (R5)+, (R4)+   ;
3519 016354 012524      MOV      (R5)+, (R4)+   ;
3520 016356             MTPS     (R5)+      ;SET THE PROCESSOR STATUS
3521 016356 106425      .WORD   106400!..C
3522 016360 005205      INC      R5
3523 016362 012577 162076  MOV      (R5)+, @FISVEC ;SET UP FIS ERROR TRAP VECTOR
3524 016366 012577 162074  MOV      (R5)+, @FISLVL ;TRAP STATUS
3525 016372 000205      RTS      R5          ;RETURN

```

```

3526
3527 ;SUBROUTINE TO POP 4 WORDS OFF THE PC "STACK"

```

DVKACB MACY11 27(1006) 17-JAN-77 15:14 PAGE 85
 DVKACB.P11 13-JAN-77 15:28 PUSH AND POP SUBROUTINES

SEQ 0093

```

3528 ;ALSO SAVES THE PROCESSOR STATUS WORD (EXCEPT T BIT)
3529
3530 016374 MFP5 $PSW ;SAVE PROCESSOR STATUS WORD
3531 016374 106767 .WORD 106700!..C
3532 016400 042767 000020 162024 BIC #20, $PSW ;CLEAR T-BIT
3533 016406 011600 MOV (SP), RO ;GET RETURN ADDRESS
3534 016410 162700 000014 SUB #14, RO ;POINT TO TOP OF "PC STACK"
3535 016414 012067 162016 MOV (RO)+, ANS1 ;SAVE 1ST HALF INPUT DATA
3536 016420 012067 162014 MOV (RO)+, ANS2 ;SAVE 2ND HALF INPUT DATA
3537 016424 010067 162004 MOV RO, $SP ;SAVE ASSUMED END PC "STACK POINTER"
3538 016430 012067 162006 MOV (RO)+, ANS3 ;SAVE 1ST HALF OF ANSWER
3539 016434 012067 162004 MOV (RO)+, ANS4 ;SAVE 2ND HALF OF ANSWER
3540 016440 000207 RTS PC
3541
3542 ;ERRONIOUS TRAP SERVICE ROUTINE
3543
3544 016442 104000 TRAPER: HLT ;FIS SHOULDN'T HAVE TRAPED
3545 016444 000547 547 ;THE ERROR NUMBER IS 547
3546 016446 000002 RTI
3547

```

```

3548 ;*****
3549 .SBTTL HLT ROUTINE (ERROR TYPEOUT)
3550 016450 032737 002000 000422 HLTS: BIT #SW10,2#SSWREG ; SHOULD IT RING THE BELL ON ERROR?
3551 016456 001402 BEQ 1$ ; NO - SKIP
3552 016460 000004 000504 TYPE $BELL ; RING BELL
3553 016464 005267 161772 15: INC ERRORS ; COUNT THE NUMBER OF ERRORS
3554 016470 032737 020000 000422 BIT #SW13,2#SSWREG ; SKIP TYPEOUT IF SET
3555 016476 001023 BNE 2$ ; SKIP TYPEOUTS
3556 016500 000004 000472 TYPE RETURN
3557 016504 013637 000402 MOV 2(6)+,2#SFATAL ; PLACE THE ERROR NUMBER IN LOCATION SFATAL
3558 016510 014667 161714 MOV -(6),HLTADS ; PUT ADDRESS OF INSTRUCTION ON STACK
3559 016514 162767 000002 161706 SUB #2,HLTADS
3560 016522 017605 000000 MOV 2(6),TTY ; TYPE 2(6) IN OCTAL
3561 016526 004767 000124 JSR %7,PRINTR ; TYPE LEADING ZERO'S
3562 016532 062716 000002 ADD #2,(6) ; ADJUST THE RETURN ADDRESS
3563 016536 000004 000500 TYPE SPACE+3
3564 016542 004767 000046 JSR PC,ERRORS ; GO TO USER ERROR ROUTINE
3565 016546 105767 161646 25: TSTB SENV ; ARE WE RUNNING UNDER APT?
3566 016552 001403 BEQ 4$ ; IF NOT THEN GO TO 4$
3567 016554 005237 000400 INC 2#MSGTY ; OTHERWISE INFORM APT
3568 016560 000777 BR AND LOOP
3569 016562 005737 000422 4$: TST 2#SSWREG ; HALT ON ERROR
3570 016566 100001 BPL .+4 ; SKIP IF CONTINUE
3571 016570 000000 HALT ; HALT ON ERROR!
3572 016572 032737 001000 000422 BIT #SW09,2#SSWREG ; CHECK FOR INHIBIT LOOP ON ERROR
3573 016600 001001 BNE .+4 ; SKIP IF LOOP ON ERROR
3574 016602 000002 RTI
3575 016604 105067 161673 CLRB SICNT
3576 016610 000167 177164 JMP KITS ; LOOP ON TEST UNTIL NO ERRORS
3577
3578
    
```

```

3579
3580 ;*****
3581
3582 .SBTTL USER ERROR ROUTINE
3583
3584 016614 117767 161610 161716 ERRORS: MOVB @HLTADS, TYPCNT ;TYPE COUNT IS LOW BYTE OF HLT
3585 016622 062767 000002 161710 ADD #2, TYPCNT ;TYPE COUNT = X+2
3586 016630 012703 000430 MOV @HLTADS, R3 ;TOP OF DATA TO BE TYPED
3587 016634 ERRIS:
3588 016634 012305 MOV (R3)+, TTY ;TYPE (R3)+ IN OCTAL
3589 016636 004767 000014 JSR %7, PRINTR ;TYPE LEADING ZERO'S
3590 016642 000004 000501 TYPE, SPACE+4 ;SPACE
3591 016646 105367 161666 DECB TYPCNT ;CHECK FOR DONE
3592 016652 100370 BPL ERRIS ;BRANCH IF NOT DONE
3593 016654 000207 RTS PC
3594
    
```

DVKACB MACY11 27(1006) 17-JAN-77 15:14 PAGE 88
 DVKACB.P11 13-JAN-77 15:28 OCTAL WORD & ADDRESS TYPER

SEQ 0096

3595	016656	112767	000001	161660	PRINTR: MOVB	#1, .PR	:SET ZERO FILL SWITCH
3596	016664	000402			BR	+.6	:SKIP
3597	016666	005067	161652		PRINTS: CLR	.PR	:SUPPRESS LEADING ZERO'S
3598	016672	112767	177772	161645	MOVB	#-6, .PR+1	:SET COUNT
3599	016700	010446			MOV	R4, -(6)	:SAVE R4
3600	016702	012704	017004		MOV	#.PRBUF, R4	:SET POINTER TO FIRST ASCII CHAR.
3601	016706	105014			CLRB	(4)	:CLEAR FIRST BYTE
3602	016710	000405			BR	.PRF	:ROTATE FIRST BIT
3603	016712	105014			.PRL: CLRB	(4)	:CLEAR BYTE OF CHARACTER
3604	016714	006105			ROL	TTY	:ROTATE BIT INTO C
3605	016716	106114			ROLB	(4)	:PACK IT
3606	016720	006105			ROL	TTY	:ROTATE BIT INTO C
3607	016722	106114			ROLB	(4)	:PACK IT
3608	016724	006105			.PRF: ROL	TTY	:ROTATE BIT INTO C
3609	016726	106114			ROLB	(4)	:PACK IT
3610	016730	105714			TSTB	(4)	:IS IT ZERO?
3611	016732	001402			BEQ	+.6	:SKIP INC
3612	016734	105267	161604		INCB	.PR	:SET FILL SWITCH
3613	016740	105767	161600		TSTB	.PR	:CHECK FILL SWITCH
3614	016744	001402			BEQ	+.6	:SKIP BITSET
3615	016746	152724	000060		BISB	#'0, (4)+	:MAKE INTO ASCII CHAR
3616	016752	105267	161567		INCB	.PR+1	:INC COUNT
3617	016756	001355			BNE	.PRL	:REPEAT
3618	016760	022704	017004		CMP	#.PRBUF, R4	:EMPTY BUFFER?
3619	016764	001002			BNE	+.6	:SKIP IF NOT
3620	016766	112724	000060		MOVB	#'0, (4)+	:LOAD 1 ZERO
3621	016772	105014			CLRB	(4)	:NULL TERMINATOR
3622	016774	000004	017004		TYPE	.PRBUF	:TYPE IT
3623	017000	012604			MOV	(6)+, R4	:RESTORE R4
3624	017002	000207			RTS	PC	:RETURN
3625							
3626	017004	000004			.PRBUF: .BLKW	4	:OUTPUT BUFFER

```

3627 ;*****
3628
3629
3630 .SBTTL POWER DOWN AND UP ROUTINES
3631 :POWER DOWN ROUTINE
3632 017014 012737 017136 000024 $PWRDN: MOV $SILLUP, @#PWRVEC ;; SET FOR FAST UP
3633 017022 012737 000340 000026 MOV #340, @#PWRVEC+2 ;; PRIO:7
3634 017030 010046 MOV RO, -(SP) ;; PUSH RO ON STACK
3635 017032 010146 MOV R1, -(SP) ;; PUSH R1 ON STACK
3636 017034 010246 MOV R2, -(SP) ;; PUSH R2 ON STACK
3637 017036 010346 MOV R3, -(SP) ;; PUSH R3 ON STACK
3638 017040 010446 MOV R4, -(SP) ;; PUSH R4 ON STACK
3639 017042 010546 MOV R5, -(SP) ;; PUSH R5 ON STACK
3640 017044 010667 000072 MOV SP, $SAVR6 ;; SAVE SP
3641 017050 012737 017062 000024 MOV $PWRUP, @#PWRVEC ;; SET UP VECTOR
3642 017056 000000 HALT
3643 017060 000776 BR .-2 ;; HANG UP
3644
3645 :POWER UP ROUTINE
3646 017062 016706 000054 $PWRUP: MOV $SAVR6, SP ;; GET SP
3647 017066 005067 000050 CLR $SAVR6 ;; WAIT LOOP FOR THE TTY
3648 017072 005267 000044 1$: INC $SAVR6 ;; WAIT FOR THE INC
3649 017076 001375 BNE 1$ ;; OF WORD
3650 017100 012605 MOV (SP)+, R5 ;; POP STACK INTO R5
3651 017102 012604 MOV (SP)+, R4 ;; POP STACK INTO R4
3652 017104 012603 MOV (SP)+, R3 ;; POP STACK INTO R3
3653 017106 012602 MOV (SP)+, R2 ;; POP STACK INTO R2
3654 017110 012601 MOV (SP)+, R1 ;; POP STACK INTO R1
3655 017112 012600 MOV (SP)+, R0 ;; POP STACK INTO R0
3656 017114 012737 017014 000024 MOV $PWRDN, @#PWRVEC ;; SET UP THE POWER DOWN VECTOR
3657 017122 012737 000340 000026 MOV #340, @#PWRVEC+2 ;; PRIO:7
3658 017130 000004 TYPE ;; REPORT THE POWER FAILURE
3659 017132 017144 $PWRMG: .WORD $POWER ;; POWER FAIL MESSAGE POINTER
3660 017134 000002 RTI
3661 017136 000000 $SILLUP: HALT ;; THE POWER UP SEQUENCE WAS STARTED
3662 017140 000776 BR .-2 ;; BEFORE THE POWER DOWN WAS COMPLETE
3663 017142 000000 $SAVR6: 0 ;; PUT THE SP HERE
3664 017144 005015 047520 042527 $POWER: .ASCIZ <15><12>"POWER"
3665 017152 000122
3666 .EVEN
    
```

```

3667
3668          ;*      TYPE OUT ROUTINE
3669          ;*      -----
3670          ;*
3671          ;*
3672          ;*      THIS ROUTINE IS USED TO TYPE ASCIZ MESSAGES
3673          ;*
3674
3675 017154 132737 000040 000421 $TYPE: BITB  #40,2#SENVM ;HAS THE CONSOLE OUTPUTS BEEN SUPPRESSED?
3676 017162 001007          BNE      3$      ;IF SO THEN RETURN FROM THE SUBROUTINE VIA 3$
3677 017164 010046          MOV      RO,-(SP) ;OTHERWISE SAVE RO
3678 017166 017600 000002          MOV      22(SP),RO ;GET THE ADDRESS OF THE ASSCIZ STRING
3679 017172 112046          2$:      MOVB   (RO)+,-(SP) ;PUSH THE CHARACTER TO BE TYPED ONTO STACK
3680 017174 001005          BNE      4$      ;BRANCH IF IT IS NOT THE TERMINATOR
3681 017176 005726          TST     (SP)+
3682 017200 012600          MOV     (SP)+,RO ;OTHERWISE RESTORE THE STACK AND RO
3683 017202 062716 000002          3$:      ADD     #2,(SP) ;ADJUST THE RETURN PC
3684 017206 000002          RTI
3685
3686 017210 105777 161334          4$:      TSTB   2$TPS ;IS THE PRINTER AVAILABLE?
3687 017214 100375          BPL     4$      ;IF NOT THEN LOOP HERE
3688 017216 112677 161330          MOVB   (SP)+,2$TPB ;OUT PUT THE CHARACTER
3689 017222 000763          BR     2$      ;AND GO BACK
3690          .END
    
```


CROSS REFERENCE TABLE -- USER SYMBOLS

	2487#	2491	2492#	2496	2497#	2501	2502#	2527	2528#	2532	2533#	2537	2538#	
	2542	2543#	2547	2548#	2573	2574#	2578	2579#	2583	2584#	2588	2589#	2593	
	2594#	2622	2623#	2627	2628#	2632	2633#	2637	2638#	2642	2643#	2647	2648#	
	2671	2672#	2679	2680#	2684	2685#	2689	2690#	2694	2695#	2699	2700#	2704	
	2705#	2709	2710#	2714	2715#	2719	2720#	2743	2744#	2751	2752#	2756	2757#	
	2761	2762#	2766	2767#	2771	2772#	2776	2777#	2781	2782#	2786	2787#	2791	
	2792#	2815	2816#	2823	2824#	2828	2829#	2833	2834#	2838	2839#	2843	2844#	
	2848	2849#	2853	2854#	2858	2859#	2863	2864#	2885	2886#	2894	2895#	2900	
	2901#	2905	2906#	2915	2916#	2920	2921#	2925	2926#	2930	2931#	2935	2936#	
	2940	2941#	2982	2983#	2987	2988#	2993	2994#	2998	2999#	3003	3004#	3029	
	3030#	3037	3038#	3042	3043#	3047	3048#	3052	3053#	3057	3058#	3062	3063#	
	3067	3068#	3072	3073#	3076	3077#	3102	3103#	3110	3111#	3115	3116#	3120	
	3121#	3125	3126#	3130	3131#	3189	3190#	3196	3197#	3201	3202#	3206	3207#	
	3212	3213#	3217	3218#	3222	3223#	3227	3228#	3236	3237#	3295	3296#	3302	
	3303#	3307	3308#	3312	3313#	3318	3319#	3323	3324#	3328	3329#	3333	3334#	
	3342	3343#	3545	3546#										
SFATAL	000402													
\$GET42	015640	3372#												
\$HD =	000003	14	15											
\$HIBTS	000430	143#												
\$ICNT	000503	174#	3401	3403	3405*	3410*	3575*							
\$ILLUP	017136	3632	3661#											
\$MAIL	000400	104#	144	148	203									
\$MBADR	000432	144#												
\$MSGAD	000414	111#												
\$MSGLC	000416	112#												
\$MSGTY	000400	105#	3567*											
\$PASS	000406	108#	3363*	3364*	3383									
\$PASTM	000436	146#												
\$POWER	017144	3659	3664#											
\$PSM	000432	152#	153	253	299	345	391	442	488	529	575	621	667	718
		764	805	854	911	988	1055	1132	1193	1239	1295	1336	1392	1433
		1479	1525	1571	1617	1663	1719	1771	1838	1904	1950	1996	2042	2093
		2134	2183	2237	2294	2371	2432	2483	2524	2570	2619	2676	2748	2820
		2897	2974*	2975*	2979	3034	3099*	3107	3193	3299	3438*	3439*	3453*	3486*
		3487*	3499*	3531*	3532*									
		3487*	3499*	3531*	3532*									
		215	3632#	3656										
\$PWDRN	017014	215	3632#	3656										
\$PWDRG	017132	3659#												
\$PWDRUP	017062	3641	3646#											
\$SAVR6	017142	3640*	3646	3647*	3648*	3663#								
\$SETUP=	000020	82#	3363											
\$SP	000434	154#	155	252*	258	298*	304	344*	350	390*	396	528*	534	574*
		580	620*	626	666*	672	804*	810	904*	910*	916	1048*	1054*	1060
		1192*	1198	1238*	1244	1335*	1341	1432*	1438	1478*	1484	1524*	1530	1570*
		1576	1616*	1622	1662*	1668	1831*	1837*	1843	1903*	1909	1949*	1955	1997*
		2001	2041*	2047	2133*	2139	2287*	2293*	2299	2431*	2437	2523*	2529	2564*
		2575	2669*	2675*	2681	2741*	2747*	2753	2813*	2819*	2825	2978*	2984	3027*
		3033*	3039	3100*	3106*	3112	3186*	3192*	3198	3292*	3298*	3304	3443*	3464*
		3537*												
\$STUP =	177777	82#												
\$SVPC =	001000	87#	92											
\$SWR =	160000	14	15#	3356	3363	3373	3382	3383	3660					
\$SWREG	000422	116#	3147	3253	3393	3395	3397	3399	3550	3554	3569	3572		
\$TESTN	000404	107#	229*	273	319	365	411	457	503	549	595	641	687	733
		779	825	879	951	1023	1095	1167	1213	1259	1310	1356	1407	1453
		1499	1545	1591	1637	1683	1734	1806	1878	1924	1970	2016	2062	2108

F09

DVKACB MACY11 27(1006) 17-JAN-77 15:14 PAGE 101
DVKACB.P11 13-JAN-77 15:28 CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0108

. ABS. 017224 000

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

DVKACB,DVKACB/SOL/CRF=DVKACB.SML,DVKACB.P11
RUN-TIME: 47 62 4 SECONDS
RUN-TIME RATIO: 267/114=2.3
CORE USED: 41K (82 PAGES)