

# PDP11/23

KEF11-AA DIAGNOSTIC #1  
CJKDCA0

AH-F242A-MC  
COPYRIGHT 1979  
FICHE 1 OF 2

SEP 1979  
**digital**  
MADE IN USA

This microfiche card contains 252 frames of diagnostic data for the PDP11/23 computer system. The frames are arranged in a grid of 14 columns and 18 rows. Each frame displays a small, high-resolution image of a document page, which appears to be a diagnostic report or manual page. The text within the frames is too small to be legible. A small white mark is present at the bottom center of the card.



# PDP11/23

KEF11-AA DIAGNOSTIC #1  
CJKDCA0

AH-F242A-MC  
COPYRIGHT 1979  
FICHE 2 OF 2

SEP 1979  
**digital**  
MADE IN USA

This microfiche strip contains 15 frames of data. The frames are arranged in two columns: the first column has 14 frames and the second column has 1 frame. The frames contain various types of information, including text, tables, and diagrams. The text is too small to read clearly, but it appears to be diagnostic data for the PDP11/23 system. The tables consist of multiple columns of data points, and the diagrams are small schematic-like drawings.



.REM &

IDENTIFICATION

PRODUCT CODE: AC-F241A-MC  
PRODUCT NAME: CJKDCAG KEF11-AA FP DIAG PART 1  
DATE CREATED: 20-JUN-79  
MAINTAINER: DIAGNOSTIC ENGINEERING

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

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

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

COPYRIGHT (C) 1979 BY DIGITAL EQUIPMENT CORPORATION

1  
2  
3  
4  
5  
6  
7  
8  
9  
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  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87

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 OPERATOR INTERACTION
5. OPERATING PROCEDURE
  - 5.1 OPERATIONAL SWITCH SETTINGS
  - 5.3 OPERATOR ACTION
6. ERRORS
  - 6.1 SUMMARY
  - 6.2 ERROR RECOVERY
7. RESTRICTIONS
  - 7.1 STARTING RESTRICTIONS
  - 7.2 OPERATING RESTRICTIONS
8. MISCELLANEOUS
  - 8.1 EXECUTION TIMES
  - 8.2 STACK POINTER
  - 8.3 PASS COUNT
  - 8.4 T-BIT TRAPPING
  - 8.5 SOFTWARE SWITCH REGISTER
  - 8.6 ACT, APT AND XXDP COMPATIBILITY
9. PROGRAM DESCRIPTION
  - 9.1 CJKDCA
10. LISTING
  - 10.1 CJKDCA



88  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100  
101  
102  
103  
104  
105  
106  
107  
108  
109  
110  
111  
112  
113  
114  
115  
116  
117  
118  
119  
120  
121  
122  
123  
124  
125  
126  
127  
128  
129  
130  
131  
132  
133  
134  
135  
136  
137  
138  
139  
140  
141  
142  
143

1.

ABSTRACT  
-----

THE TWO PROGRAMS:

CJKDCA, CJKDDA

ARE DESIGN TO DETECT AND REPORT LOGIC FAULTS IN THE F-11 MMU AND FLOATING POINT CHIP SET. THE DESIGN IS AN ATTEMPT TO REACH ALL MICRO-CODE LOCATIONS. TESTS ARE PARTITIONED INTO TWO STAND-ALONE PROGRAMS DESCRIBED BELOW.

NOTE THAT ERROR REPORTS IN THESE PROGRAMS ARE BASED UPON THE KNOWLEDGE THAT ALL PREVIOUS TESTS (CPU, MMU) HAVE BEEN RUN AND IN MOST CASE THAT THERE IS ONLY A SINGLE POINT FAULT EXISTS. IF THE PROGRAMS OR TESTS ARE NOT RUN IN ORDER THEN ERROR MESSAGES MAY NOT BE ACCURATE.

A. CJKDCA

CJKDCA TESTS: (FLOATING POINT TEST 1)

LDFPS  
STFPS  
CFCC  
SETF, SETD, SETI AND SETL  
STST  
LDF AND LDD (ALL SOURCE MODES)  
STD (MODE 0 AND 1)  
ADDF, ADDD AND SUBD (MOST CONDITIONS)  
ADDF, ADDD AND SUBD (ALL CONDITIONS NOT TESTED IN DFFPA)  
CMPD AND CMPF  
DIVD AND DIVF  
MULD AND MULF  
MODD AND MODF

B. CJKDDA

CJKDDA TESTS: (FLOATING POINT TEST 2)

STF AND STD (ALL MODES)  
STCFD AND STCDF  
CLRD AND CLRF  
NEGF AND NEGD  
ABSF AND ABSD  
TSTF AND TSTD  
NEGF, ABSF AND TSTF (ALL SOURCE MODES)  
NEGF, ABSF AND TSTF (ALL SOURCE MODES)  
LDFPS (ALL SOURCE MODES)



144  
145  
146  
147  
148  
149  
150  
151  
152  
153  
154  
155  
156  
157  
158  
159  
160  
161  
162  
163  
164  
165  
166  
167  
168  
169  
170  
171  
172  
173  
174  
175  
176  
177  
178  
179  
180  
181  
182  
183  
184  
185  
186  
187  
188  
189  
190  
191  
192  
193  
194  
195  
196  
197  
198  
199

LDCIF AND LDCLF  
LDCID AND LDCLD  
LDEXP  
STFPS (ALL DESTINATION MODES)  
STCFL AND STCFI  
STCDL AND STCDI  
STEXP  
STST

2. REQUIREMENTS  
-----

2.1 EQUIPMENT

A PROCESSOR WITH THE DCF11-AA, KTF11-AA AND KEF11-A CHIP SET.

2.2 STORAGE

BOTH PROGRAMS REQUIRE A MEMORY SYSTEM OF AT LEAST 16K TO LOAD AND RUN.

2.3 PRELIMINARY PROGRAMS

THESE TWO DIAGNOSTICS WILL ASSUME THAT THE BASIC CENTRAL PROCESSOR IS FAULTLESS. THEREFORE WHEN IN DOUBT RUN THE DCF11-AA PROCESSOR DIAGNOSTICS BEFORE THESE FLOATING POINT DIAGNOSTICS.

3. LOADING PROCEDURE  
-----

THE PROGRAMS WILL BE SUPPLIED ON THE 11/23 DIAGNOSTIC MEDIA. REFER TO THE XXDP OPERATING MANUAL FOR FURTHER INFORMATION.

4. STARTING PROCEDURE  
-----

4.1 CONTROL SWITCH SETTINGS

SEE SECTION 5.1

4.2 PROGRAM AND OPERATOR ACTION

1. LOAD PROGRAM INTO MEMORY
  2. LOAD ADDRESS 200
  3. SET CONSOLE SWITCHES (IF CONSOLE IS PRESENT)
  4. PRESS START
- ON FIRST PASS THE PROGRAM WILL IDENTIFY ITSELF. NOTE THAT IF THERE IS NO PHYSICAL CONSOLE THE PROGRAM WILL REQUEST THE OPERATOR FOR INITIAL VALUE FOR THE SOFTWARE SWITCH REGISTER (SEE SECTION 8.5). IF RUNNING UNDER ACT, APT OR CHAIN THIS DOES



200  
201  
202  
203  
204  
205  
206  
207  
208  
209  
210  
211  
212  
213  
214  
215  
216  
217  
218  
219  
220  
221  
222  
223  
224  
225  
226  
227  
228  
229  
230  
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

NOT APPLY.  
5. THE PROGRAM WILL LOOP AND AN END OF PASS WILL  
BE TYPED AT THE END OF EVERY PASS.

5. OPERATING PROCEDURE  
-----

5.1 OPERATIONAL SWITCH SETTINGS

THE SWITCH SETTING ARE:

	OCTAL	
SW<15>=1...	100000	HALT ON ERROR
SW<14>=1...	40000	LOOP ON CURRENT TEST
SW<13>=1...	20000	INHIBIT ERROR TYPE OUTS
SW<12>=1...	10000	INHIBIT T-BIT TRAPPING
SW<11>=1...	4000	INHIBIT ITERATIONS
SW<10>=1...	2000	RING TTY BELL ON ERROR
SW<9>=1....	1000	LOOP ON ERROR
SW<8>=1....	400	LOOP ON TEST SPECIFIED IN SW<6> THROUGH SW<0>

6. ERRORS  
-----

6.1 SUMMARIES

WHEN AN ERROR IS ENCOUNTERED, AN ERROR MESSAGE ACCOMPANIED  
BY THE ERROR PC ARE TYPED.  
THERE ARE FOUR STANDARD ERROR MESSAGES USED, DESCRIBING  
THE PROBABLE CAUSE OF FAILURE, SUCH AS: PROBABLY BAD MMU CHIP;  
BAD FP1 CHIP; BAD HYBRID FP CHIP; FLOATING POINT ERROR.

6.2 ERROR RECOVERY

SW<15:9>=0... MOST ERRORS WILL CAUSE EXECUTION TO  
GO TO THE START OF THE NEXT TEST  
AFTER THE MESSAGE IS TYPED. A FEW  
TESTS ARE IN SECTIONS. IN THESE  
TESTS AN ERROR WILL CAUSE EXECUTION  
TO GO TO THE NEXT SECTION AFTER THE  
MESSAGE IS TYPED.

SW<15>=1... THE PROGRAM WILL HALT AFTER TYPING  
THE ERROR MESSAGE. PRESSING THE  
CONSOLE CONTINUE WILL CAUSE THE  
PROGRAM TO CONTINUE AS IF SW<15>=0.

7. RESTRICTIONS  
-----



256  
257  
258  
259  
260  
261  
262  
263  
264  
265  
266  
267  
268  
269  
270  
271  
272  
273  
274  
275  
276  
277  
278  
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

NONE

8. MISCELLANEOUS  
-----

8.1 EXECUTION TIMES

LESS THAN 2 SECONDS FOR EACH PROGRAM ON ANY PASS.

8.2 STACK POINTER

THE STACK POINTER IS INITIALIZED TO 1100 IN EACH OF THE TWO PROGRAMS.

8.3 PASS COUNT

THE PROGRAM MAKES ONE PASS FOR EACH END OF PASS MESSAGE TYPED. THE END OF PASS MESSAGE DESCRIBES THE TOTAL NUMBER OF PASSES COMPLETED.

8.4 T-BIT TRAPPING

IF SW<12>=0 EACH PROGRAM WILL RUN WITH TRACE TRAPS ON EVERY OTHER PASS. FIRST PASS WILL NOT ENABLE TRACE TRAPS. NOTE SW<12>=1 DISABLES T-BIT TRAPS.

8.5 SOFTWARE SWITCH REGISTER

EACH OF THE TWO PROGRAMS WILL RUN WITH OR WITHOUT A CONSOLE SWITCH REGISTER. IF A PHYSICAL CONSOLE SWITCH REGISTER IS PRESENT ON THE SYSTEM, THEN THESE PROGRAMS WILL GO AHEAD AND USE IT FOR THE SWITCH FUNCTIONS DESCRIBED IN 5.1 ABOVE. IF HOWEVER THERE IS NO CONSOLE SWITCH REGISTER ON THE SYSTEM A SOFTWARE SWITCH REGISTER WILL BE USED. THIS SOFTWARE SWITCH REGISTER CAN BE EXAMINED OR MODIFIED AT ANY TIME BY THE USER IF HE TYPES CONTROL G WHILE THE PROGRAM IS RUNNING. THIS CONTROL G WILL CAUSE THE CONTENTS OF THE SOFTWARE SWITCH REGISTER TO BE TYPED ON THE TTY AND ASK THE USER FOR A NEW VALUE. WHEN THE USER TYPES A VALUE AND CARRIAGE RETURN THEN THE PROGRAM WILL RESUME TESTING AT THE SAME POINT AT WHICH IT LEFT OFF WHEN THE USER TYPED CONTROL G. NOTE THAT WHEN NOT RUNNING UNDER ACT, APT OR CHAIN THE USER WILL BE ASKED FOR A SOFTWARE SWITCH REGISTER VALUE AFTER LOADING ADDRESS 200 AND STARTING THE PROGRAM THE FIRST TIME THE PROGRAM IS RUN AFTER LOADING (ONLY IF NO CONSOLE SWITCH REGISTER IS ON THE SYSTEM).

8.6 ACT, APT AND XXDP COMPATIBILITY

THESE PROGRAMS ARE FULLY COMPATIBLE WITH:  
APT



312  
313  
314  
315  
316  
317  
318  
319  
320  
321  
322  
323  
324  
325  
326  
327  
328  
329  
330  
331  
332  
333  
334  
335  
336  
337  
338  
339  
340  
341  
342  
343  
344  
345  
346  
347  
348  
349  
350  
351  
352  
353  
354  
355  
356  
357  
358  
359  
360  
361  
362  
363  
364  
365  
366  
367

ACT  
XXDP MONITOR AND CHAIN PROGRAMS.

9. PROGRAM DESCRIPTION

TEST 1 LDFPS, STFPS AND DATA PATHS TEST

THIS IS A TEST OF THE LDFPS (LOAD FLOATING POINT STATUS) AND STFPS (STORE FLOATING POINT STATUS) INSTRUCTIONS. VARIOUS PATTERN ARE USED AND RUN THROUGH THE FLOATING POINT STATUS REGISTER. ONLY DMO AND SMO ARE USED. NOTE THAT A MASK MUST BE USED BECAUSE SOME OF THE FPS BITS CANNOT BE SET.

TEST 2 CFCC TEST

THIS IS A TEST OF THE COPY CONDITION CODES INSTRUCTION, CFCC.

TEST 3 SETF, SETD, SETI AND SETL TEST

THIS IS A TEST OF THE SETF, SETD, SETI AND SETL INSTRUCTIONS. EACH INSTRUCTION IS EXECUTED WITH THE FPS CONTAINING ALL ONES AND ALSO WITH THE FPS CLEAR. THE RESULT OF EACH SITUATION IS CHECKED.

TEST 4 ILLEGAL FPP OP CODES AND STST TEST

THIS IS A TEST OF THE FPP OPERATION CODES:

170003  
170004  
:  
170010  
170013  
170014

368  
369  
370  
371  
372  
373  
374  
375  
376  
377  
378  
379  
380  
381  
382  
383  
384  
385  
386  
387  
388  
389  
390  
391  
392  
393  
394  
395  
396  
397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420  
421  
422  
423

170077

THESE ARE ILLEGAL INSTRUCTIONS AND WITH INTERRUPTS ENABLED SHOULD CAUSE A TRAP TO 244. ALSO TESTED HERE IS THE INSTRUCTION: STST R1, WHICH SHOULD PUT THE FEC CODE 2 IN R1, AFTER ANY OF THE ABOVE OP CODES IS EXECUTED.

TEST 5 FID, INTERRUPT DISABLE, BIT TEST  
-----

THIS IS A TEST OF FPS BIT 14 (FID) OR FLOATING INTERRUPT DISABLE. AN ILLEGAL INSTRUCTION IS EXECUTED WITH FID=1. NO INTERRUPT SHOULD OCCUR.

TEST 6 LDD AND STD, WITH SRC AND DST MODE 1, TEST  
-----

THIS IS A TEST OF BOTH THE INSTRUCTION:  
LDD (RC),ACO  
AND THE INSTRUCTION:  
STD ACO,(R0) MOST OF THE FAILURES ARE ISOLATED TO THE SRC OR DST FLOWS. NOTE THAT THE INTEGRITY OF ACO HAS NOT BEEN ASSURED. THIS MEANS THAT IN SOME CASES IT WILL BE IMPOSSIBLE TO ISOLATE CERTAIN DATA PATTERN FAILURES TO EITHER THE FLOWS OR THIS ACCUMULATOR.

TEST 7 FSRC MODE 0 TEST  
-----

THIS IS A TEST OF FSRC MODE ZERO USING THE LDD AND LDF INSTRUCTIONS.

TEST 10 FDST MODE 0 TEST  
-----

THIS IS A TEST OF THE STORE INSTRUCTIONS, STD AND STF, WITH FDST MODE 0.

TEST 11 ACCUMULATORS DATA PATTERNS TEST  
-----

THIS IS A TEST OF THE FLOATING POINT PROCESSOR ACCUMULATORS.

EACH ACCUMULATOR IS TESTED IN TWO WAYS:  
1 TEST PATTERN GENERATED BY FLOATING A ONE ACROSS A FIELD OF ZEROES.  
2 TEST PATTERN GENERATED BY FLOATING A ZERO ACROSS A FIELD OF ONES.

TEST 12 EACH OF ACCUMULATORS ACO THROUGH AC5 IS TESTED.  
FPP ACCUMULATORS DUAL ADDRESS TEST  
-----



424  
425  
426  
427  
428  
429  
430  
431  
432  
433  
434  
435  
436  
437  
438  
439  
440  
441  
442  
443  
444  
445  
446  
447  
448  
449  
450  
451  
452  
453  
454  
455  
456  
457  
458  
459  
460  
461  
462  
463  
464  
465  
466  
467  
468  
469  
470  
471  
472  
473  
474  
475  
476  
477  
478  
479

THIS TEST PERFORMS A DUAL ADDRESSING TEST ON THE  
FLOATING ACCUMULATORS. NOTE THAT ACCUMULATOR ZERO  
IS USED TO ACCESS ALL THE OTHERS.

TEST 13 FSRC MODE 0 WITH ILLEGAL ACCUMULATOR TEST  
-----

THIS IS A TEST OF FSRC MODE 0 WITH ACCUMULATORS 6  
AND 7. USE OF EITHER OF THESE NON-EXISTENT  
ACCUMULATORS SHOULD RESULT IN A TRAP TO 244 WITH  
FEC=2 (ILLEGAL FPP INSTRUCTION).

TEST 14 FSRC MODE 2 TEST  
-----

THIS IS A TEST OF FSRC MODE 2, AUTO INCREMENT MODE.

TEST 15 FSRC MODE 4 TEST  
-----

THIS IS A TEST OF FSRC MODE 4, AUTO DECREMENT MODE.

TEST 16 FSRC MODE 2, WITH FD=0, TEST  
-----

THIS IS A TEST OF FSRC MODE 2 WITH FD=0. (AUTO  
INCREMENT)

TEST 17 FSRC MODE 2 WITH GR7, IMMEDIATE MODE, TEST  
-----

THIS IS A TEST OF FSRC MODE 2 USING GR7 (THE PC).  
THIS IS IMMEDIATE MODE.

TEST 20 FSRC MODE 3 TEST  
-----

THIS IS A TEST OF FSRC MODE 3, AUTO INCREMENT  
DEFERRED

TEST 21 FSRC MODE 5 TEST  
-----

THIS IS A TEST OF FSRC MODE 5, AUTO DECREMENT  
DEFERRED.

TEST 22 FSRC MODE 6 TEST  
-----

THIS IS A TEST OF FSRC MODE 6, INDEX MODE

TEST 23 FSRC MODE 7 TEST  
-----

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  
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

THIS IS A TEST OF FSRC MODE 7, INDEX DEFERRED MODE.

TEST 24 (BUT EZBT Y8),(BUT ENBT) AND (BUT FIUV) TEST  
-----

THIS IS A TEST OF THE (BUT EZBT Y8) FORK, THE (BUT ENBT) FORK AND (BUT FIUV) FORK IN THE LOAD INSTRUCTION FLOWS.  
EACH OF THE PATTERNS:

0  
+NUM  
-NUM  
-0

IS LOADED TWICE, ONCE WITH AC>0 THEN WITH AC=0.  
AFTER EACH LOAD THE FPS IS CHECK TO INSURE THAT CONTROL WAS PASSED THROUGH WITH THE FORKS PROPERLY.

TEST 25 ADDF,ADDD,SUBF AND SUBD WITH FSRC=AC=0 TEST  
-----

THIS IS A TEST OF ADD AND SUB WITH FSRC=AC=0

TEST 26 ADDD AND SUB WITH FSRC=0  
-----

THIS IS A TEST OF ADD AND SUB WITH FSRC=0.

TEST 27 SUBD WITH AC=0 TEST  
-----

THIS IS A TEST OF SUBD WITH AC=0. BOTH POSITIVE AND NEGATIVE FSRC'S ARE TRIED.

TEST 30 ADDD WITH AC=0 TEST  
-----

POSITIVE AND NEGATIVE FSRC'S ARE TRIED.

TEST 31 ADDF AND ADDD WITH E(AC)=E(FSRC) AND (BUT FT) TEST  
-----

THIS IS A TEST OF THE ADD INSTRUCTION WITH THE OPERANDS HAVING EQUAL EXPONENTS. THE (BUT FT) FORK IN THE ROUND/TRUNK FLOWS IS ALSO TESTED.

TEST 32 ADDF AND ADDD WITH E(AC) LESS THAN E(FSRC) TEST  
-----

THIS IS A TEST OF THE ADDD AND ADDF INSTRUCTIONS AND THE ALIGN AC ALGORITHM FLOWS. THE CONSTANT (25 FOR FLOATING, 57 FOR DOUBLE) USED IS CHECKED. THEN SIMPLE AND WORST CASE ALIGNMENT SITUATIONS ARE TRIED. NOTE E(AC) IS LESS THEN E(FSRC)



536  
537  
538  
539  
540  
541  
542  
543  
544  
545  
546  
547  
548  
549  
550  
551  
552  
553  
554  
555  
556  
557  
558  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
590  
591

TEST 33 ADDF AND ADDD WITH E(AC) GREATER THAN E(FSRC) TEST  
-----

THIS IS A TEST OF THE ADDD AND ADDF INSTRUCTIONS AND THE ALIGN FSRC ALGORITHM FLOWS. FIRST THE CONSTANT USED IS CHECKED. THEN SIMPLF AND WORST CASE ALIGNMENT SITUATIONS ARE TRIED. NOTE E(AC) IS GREATER THAN E(FSRC).

TEST 34 ADDD WITH NEGATIVE OPRANDS TEST  
-----

THIS IS A TEST OF THE ADDD INSTRUCTION WITH NEGATIVE OPRANDS. EVERY COMBINATION OF OPRAND SIGNS IS TRIED.

TEST 35 SUBD TEST  
-----

THIS IS A TEST OF THE SUBD INSTRUCTION. BOTH A POSITIVE AND A NEGATIVE NUMBER IS SUBTRACTED FROM IT SELF

TEST 36 NORMALIZE ALGORITHM TEST  
-----

THIS IS A TEST OF THE NORMALIZE FLOW ALGORITHM. TWO PATTERNS ARE USED, FIRST THE MINIMUM SITUATION REQUIRING ONE LEFT SHIFT AND THEN THE MAXIMUM SITUATION REQUIRING 56 SHIFTS.

TEST 37 ROUND\TRUNK TEST  
-----

THIS IS A TEST OF THE ROUND\TRUNK FLOWS. IN PARTICULAR TWO THINGS ARE TESTED: FIRST A CONDITION IN WHICH ROUNDING RESULTS IN THE NEED FOR RENORMALIZATION, AND SECOND THE PSW CONDITION CODES N AND Z BIT COMBINATIONS

TEST 40 OVER\UNDER TEST  
-----

THIS IS A PARTIAL TEST OF THE OVER\UNDER FLOWS. ONE OVERFLOW AND TWO UNDERFLOW CONDITIONS ARE CHECKED. THE REMAINING UNDERFLOW COND. AND THE REMAINING OVERFLOW COND. WILL BE CHECKED LATER USING THE XXX INSTRUCTION. HERE EACH CONDITION TESTED IS CHECKED BOTH WITH TRAPS ENABLED (FIU=1 OR FIV=1) AND ALSO WITH TRAPS DISABLED (FIU=0 OR FIV=0).

TEST 41 LDCFD AND LDCDF TEST  
-----

592  
593  
594  
595  
596  
597  
598  
599  
600  
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  
647

THIS IS A TEST OF LDCFD AND LDCDF.

TEST 42                                      
                      CMPD TEST

THIS IS A TEST OF THE CMPD INSTRUCTION. NOTE THAT A SUBROUTINE IS USED TO SET UP OPERANDS, EXECUTE THE INSTRUCTION AND CHECK THE RESULTS

TEST 43                                      
                      DIVD WITH (FSRC=0) AND (BUT FD) TEST

THIS IS A TEST OF THE DIVD INSTRUCTION WITH A ZERO DIVISOR. THE CONDITION IS CHECKED WITH BOTH TRAP ENABLED AND TRAPS DISABLED.

TEST 44                                      
                      DIVF TEST

THIS IS A TEST OF THE DIVF INSTRUCTION. NOTE THAT A SUBROUTINE IS USED TO SET UP THE OPERANDS, EXECUTE THE INSTRUCTION AND CHECK THE RESULTS.

TEST 45                                      
                      DIVD TEST

THIS IS A TEST OF THE DIVD INSTRUCTION. NOTE THAT A SUBROUTINE IS USED TO SET UP THE OPERANDS, EXECUTE THE INSTRUCTION AND CHECK THE RESULTS.

TEST 46                                      
                      MULF TEST

THIS IS A TEST OF THE MULF INSTRUCTION. IT MAKES USE OF A SUBROUTINE TO SET UP THE OPERANDS, EXECUTE THE MULF INSTRUCTION AND CHECK THE RESULTS.

TEST 47                                      
                      MULD TEST

THIS IS A TEST OF THE MULD INSTRUCTION. NOTE THAT A SUBROUTINE IS USED TO SET UP THE OPERANDS, EXECUTE THE MULD INSTRUCTION AND CHECK THE RESULTS.

TEST 50                                      
                      UNDER\OVER FLOW, USING MULF WITH TRAPS DISABLED, TEST

THIS IS A TEST OF THE OVERFLOW AND UNDERFLOW CONDITIONS USING THE MULF INSTRUCTION WITH TRAPS DISABLED. NOTE THAT A SUBROUTINE IS USED TO SET UP THE OPERANDS, EXECUTE THE MULF INSTRUCTION AND CHECK THE RESULTS.

TEST 51                                      
                      UNDER\OVER FLOW, USING MULD WITH TRAPS DISABLED, TEST



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  
693  
694  
695  
696  
697  
698  
699  
700  
701  
702  
703

THIS IS A TEST OF THE OVERFLOW AND UNDERFLOW CONDITIONS THAT CAN ARRISE USING THE MULD INSTRUCTION WITH TRAPS DISABLED. A SUBROUTINE IS USED TO SET UP THE OPERANDS, EXECUTE THE MULD INSTRUCTION AND CHECK THE RESULTS.

TEST 52 UNDER\OVER FLOW, USING MULF WITH TRAPS ENABLED, TEST  
-----

THIS IS A TEST OF THE UNDERFLOW AND OVERFLOW CONDITIONS THAT CAN OCCUR USING THE MULF INSTRUCTION. A SUBROUTINE IS CALLED TO SET UP THE OPERANDS, EXECUTE THE MULF INSTRUCTION AND CHECK THE RESULTS. HERE THE PARTICULAR INTERRUPT, EITHER OVERFLOW OR UNDERFLOW, IS ENABLED SO A TRAP SHOULD OCCUR.

TEST 53 UNDER\OVER FLOW, USING MULD WITH TRAPS ENABLED, TEST  
-----

THIS IS A TEST OF THE OVER FLOW AND UNDER FLOW CONDITIONS USING THE MULD INSTRUCTION WITH TRAPS ENABLED. A SUBROUTINE IS USED TO SET UP THE OPERANDS, EXECUTE THE MULD INSTRUCTION AND CHECK THE RESULTS.

TEST 54 MODF TEST  
-----

THIS IS A TEST OF THE MODF INSTRUCTION, WHICH MAKES USE OF A SUBROUTINE TO SET UP THE OPERANDS, EXECUTE THE MODF INSTRUCTION AND CHECK THE RESULTS.

TEST 55 MODD TEST  
-----

THIS IS A TEST OF THE MODD INSTRUCTION. IT MAKES USE OF A SUBROUTINE TO SET UP THE ARGUMENTS, EXECUTE THE INSTRUCTION AND CHECK THE RESULTS.

TEST 56 UNDER\OVER FLOW, USING MODF WITH TRAPS DISABLED, TEST  
-----

THIS IS A TEST OF THE MODF OVERFLOW AND UNDERFLOW CONDITIONS. IT MAKES USE OF A SUBROUTINE TO SETUP THE OPERANDS, EXECUTE THE MODF INSTRUCTION AND CHECK THE RESULTS. TRAPS ARE DISABLED DURING THIS TEST.

TEST 57 UNDER\OVER FLOW, USING MODD WITH TRAPS DISABLED, TEST  
-----

TEST 60 MORE MICROCODES COVERAGE, TEST  
-----

704  
705  
706  
707  
708  
709  
710  
711

10.  
&

LISTING

C  
C



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  
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

.LIST ME  
.NLIST MD,MC,CND

.ENABL ABS

.TITLE CJKDCA-A KEF11-A FP DIAG PART 1  
;\*COPYRIGHT (C) DEC 1978  
;\*DIGITAL EQUIPMENT CORP.  
;\*MAYNARD, MASS. 01754  
:\*  
;\*PROGRAM BY DIAGNOSTIC ENGINEERING  
:\*  
;\*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC  
;\*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.  
:\*  
\$TN=1  
\$SWR=160000 ;;HALT ON ERROR, LOOP ON TEST, INHIBIT ERROR TYP  
FPVECT=244  
\$SWR=177400  
\$SWRMSK=200  
TAB=11  
CRLF=15  
.SBTTL BASIC DEFINITIONS  
;\*INITIAL ADDRESS OF THE STACK POINTER \*\*\* 1100 \*\*\*  
STACK= 1100  
.EQUIV EMT,ERROR ;;BASIC DEFINITION OF ERROR CALL  
.EQUIV IOT,SCOPE ;;BASIC DEFINITION OF SCOPE CALL

000001  
160000

000244  
177400  
000200  
000011  
000015

001100

```
768
769      : *MISCELLANEOUS DEFINITIONS
770      000011 HT= 11      :: CODE FOR HORIZONTAL TAB
771      000012 LF= 12      :: CODE FOR LINE FEED
772      000015 CR= 15      :: CODE FOR CARRIAGE RETURN
773      000200 CRLF= 200    :: CODE FOR CARRIAGE RETURN-LINE FEED
774      177776 PS= 177776  :: PROCESSOR STATUS WORD
775      .EQUIV PS,PSW
776      177774 STKLMT= 177774 :: STACK LIMIT REGISTER
777      177772 PIRQ= 177772  :: PROGRAM INTERRUPT REQUEST REGISTER
778      177570 DSWR= 177570  :: HARDWARE SWITCH REGISTER
779      177570 DDISP= 177570 :: HARDWARE DISPLAY REGISTER
780
781      : *GENERAL PURPOSE REGISTER DEFINITIONS
782      000000 R0= %0      :: GENERAL REGISTER
783      000001 R1= %1      :: GENERAL REGISTER
784      000002 R2= %2      :: GENERAL REGISTER
785      000003 R3= %3      :: GENERAL REGISTER
786      000004 R4= %4      :: GENERAL REGISTER
787      000005 R5= %5      :: GENERAL REGISTER
788      000006 R6= %6      :: GENERAL REGISTER
789      000007 R7= %7      :: GENERAL REGISTER
790      000006 SP= %6      :: STACK POINTER
791      000007 PC= %7      :: PROGRAM COUNTER
792
793      : *PRIORITY LEVEL DEFINITIONS
794      000000 PR0= 0      :: PRIORITY LEVEL 0
795      000040 PR1= 40     :: PRIORITY LEVEL 1
796      000100 PR2= 100    :: PRIORITY LEVEL 2
797      000140 PR3= 140    :: PRIORITY LEVEL 3
798      000200 PR4= 200    :: PRIORITY LEVEL 4
799      000240 PR5= 240    :: PRIORITY LEVEL 5
800      000300 PR6= 300    :: PRIORITY LEVEL 6
801      000340 PR7= 340    :: PRIORITY LEVEL 7
802
803      : *'SWITCH REGISTER' SWITCH DEFINITIONS
804      100000 SW15= 100000
805      040000 SW14= 40000
806      020000 SW13= 20000
807      010000 SW12= 10000
808      004000 SW11= 4000
809      002000 SW10= 2000
810      001000 SW09= 1000
811      000400 SW08= 400
812      000200 SW07= 200
813      000100 SW06= 100
814      000040 SW05= 40
815      000020 SW04= 20
816      000010 SW03= 10
817      000004 SW02= 4
818      000002 SW01= 2
819      000001 SW00= 1
820      .EQUIV SW09,SW9
821      .EQUIV SW08,SW8
822      .EQUIV SW07,SW7
823      .EQUIV SW06,SW6
```

```

824      .EQUIV SW05,SW5
825      .EQUIV SW04,SW4
826      .EQUIV SW03,SW3
827      .EQUIV SW02,SW2
828      .EQUIV SW01,SW1
829      .EQUIV SW00,SW0
830
831      ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
832      100000 BIT15= 100000
833      040000 BIT14= 40000
834      020000 BIT13= 20000
835      010000 BIT12= 10000
836      004000 BIT11= 4000
837      002000 BIT10= 2000
838      001000 BIT09= 1000
839      000400 BIT08= 400
840      000200 BIT07= 200
841      000100 BIT06= 100
842      000040 BIT05= 40
843      000020 BIT04= 20
844      000010 BIT03= 10
845      000004 BIT02= 4
846      000002 BIT01= 2
847      000001 BIT00= 1
848      .EQUIV BIT09,BIT9
849      .EQUIV BIT08,BIT8
850      .EQUIV BIT07,BIT7
851      .EQUIV BIT06,BIT6
852      .EQUIV BIT05,BIT5
853      .EQUIV BIT04,BIT4
854      .EQUIV BIT03,BIT3
855      .EQUIV BIT02,BIT2
856      .EQUIV BIT01,BIT1
857      .EQUIV BIT00,BIT0
858
859      ;*BASIC "CPU" TRAP VECTOR ADDRESSES
860      000004 ERRVEC= 4      ;; TIME OUT AND OTHER ERRORS
861      000010 RESVEC= 10    ;; RESERVED AND ILLEGAL INSTRUCTIONS
862      000014 TBITVEC=14   ;; 'T' BIT
863      000014 TRTVEC= 14   ;; TRACE TRAP
864      000014 BPTVEC= 14   ;; BREAKPOINT TRAP (BPT)
865      000020 IOTVEC= 20   ;; INPUT/OUTPUT TRAP (IOT) **SCOPE**
866      000024 PWRVEC= 24   ;; POWER FAIL
867      000030 EMTVEC= 30   ;; EMULATOR TRAP (EMT) **ERROR**
868      000034 TRAPVEC=34   ;; 'TRAP' TRAP
869      000060 TKVEC= 60    ;; TTY KEYBOARD VECTOR
870      000064 TPVEC= 64    ;; TTY PRINTER VECTOR
871      000240 PIRQVEC=240  ;; PROGRAM INTERRUPT REQUEST VECTOR
872
873      .SBTTL FPP REGISTER DEFINITIONS
874      000000 AC0      =%0
875      000001 AC1      =%1
876      000002 AC2      =%2
877      000003 AC3      =%3
878      000004 AC4      =%4
879      000005 AC5      =%5
879      000006 AC6      =%6
    
```



```
880          000007          AC7      =%7
881
882          .SBTTL TRAP CATCHER
883
884          000000          .=0
885          ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
886          ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
887          ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
888          000174          .=174
889 000174 000000  DISPREG: .WORD 0          ;;SOFTWARE DISPLAY REGISTER
890 000176 000000  SWREG:   .WORD 0          ;;SOFTWARE SWITCH REGISTER
891          .SBTTL STARTING ADDRESS(ES)
892 000200 000137 001466  JMP      @#START ;;JUMP TO STARTING ADDRESS OF PROGRAM
```

```

893          .SBTTL COMMON TAGS
894
895          ::*****
896          ::*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
897          ::*USED IN THE PROGRAM.
898
899          001100          .=1100
900          001100          $CMTAG:          ;;START OF COMMON TAGS
901          001100          000000          .WORD          0          ;;CONTAINS THE TEST NUMBER
902          001102          000          .BYTE          0          ;;CONTAINS ENRUP FLAG
903          001103          000          .BYTE          0          ;;CONTAINS SUBTEST ITERATION COUNT
904          001104          000000          .WORD          0          ;;CONTAINS SCOPE LOOP ADDRESS
905          001106          000000          .WORD          0          ;;CONTAINS SCOPE RETURN FOR ERRORS
906          001110          000000          .WORD          0          ;;CONTAINS TOTAL ERRORS DETECTED
907          001112          000000          .WORD          0          ;;CONTAINS ITEM CONTROL BYTE
908          001114          000          .BYTE          0          ;;CONTAINS MAX. ERRORS PER TEST
909          001115          001          .BYTE          1          ;;CONTAINS PC OF LAST ERROR INSTRUCTION
910          001116          000000          .WORD          0          ;;CONTAINS ADDRESS OF 'GOOD' DATA
911          001120          000000          .WORD          0          ;;CONTAINS ADDRESS OF 'BAD' DATA
912          001122          000000          .WORD          0          ;;CONTAINS 'GOOD' DATA
913          001124          000000          .WORD          0          ;;CONTAINS 'BAD' DATA
914          001126          000000          .WORD          0          ;;RESERVED--NOT TO BE USED
915          001130          000000          .WORD          0
916          001132          000000          .WORD          0
917          001134          000          .BYTE          0          ;;AUTOMATIC MODE INDICATOR
918          001135          000          .BYTE          0          ;;INTERRUPT MODE INDICATOR
919          001136          000000          .WORD          0
920          001140          177570          .WORD          DSWR          ;;ADDRESS OF SWITCH REGISTER
921          001142          177570          .WORD          DDISP          ;;ADDRESS OF DISPLAY REGISTER
922          001144          177560          .WORD          177560          ;;TTY KBD STATUS
923          001146          177562          .WORD          177562          ;;TTY KBD BUFFER
924          001150          177564          .WORD          177564          ;;TTY PRINTER STATUS REG. ADDRESS
925          001152          177566          .WORD          177566          ;;TTY PRINTER BUFFER REG. ADDRESS
926          001154          000          .BYTE          0          ;;CONTAINS NULL CHARACTER FOR FILLS
927          001155          002          .BYTE          2          ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
928          001156          012          .BYTE          12          ;;INSERT FILL CHARS. AFTER A 'LINE FEED'
929          001157          000          .BYTE          0          ;;'TERMINAL AVAILABLE' FLAG (BIT<07>=0=YES)
930          001160          000000          .WORD          0          ;;CONTAINS THE ADDRESS FROM
931          931          931          .WORD          0          ;;WHICH ($REG0) WAS OBTAINED
932          001162          000000          .WORD          0          ;;CONTAINS (($REGAD)+0)
933          001164          000000          .WORD          0          ;;CONTAINS (($REGAD)+2)
934          001166          000000          .WORD          0          ;;CONTAINS (($REGAD)+4)
935          001170          000000          .WORD          0          ;;CONTAINS (($REGAD)+6)
936          001172          000000          .WORD          0          ;;CONTAINS (($REGAD)+10)
937          001174          000000          .WORD          0          ;;CONTAINS (($REGAD)+12)
938          001176          000000          .WORD          0          ;;CONTAINS (($REGAD)+14)
939          001200          000000          .WORD          0          ;;CONTAINS (($REGAD)+16)
940          001202          000000          .WORD          0          ;;CONTAINS (($REGAD)+20)
941          001204          000000          .WORD          0          ;;CONTAINS (($REGAD)+22)
942          001206          000000          .WORD          0          ;;CONTAINS (($REGAD)+24)
943          001210          000000          .WORD          0          ;;CONTAINS (($REGAD)+26)
944          001212          000000          .WORD          0          ;;CONTAINS (($REGAD)+30)
945          001214          000000          .WORD          0          ;;CONTAINS (($REGAD)+32)
946          001216          000000          .WORD          0          ;;CONTAINS (($REGAD)+34)
947          001220          000000          .WORD          0          ;;CONTAINS (($REGAD)+36)
948          001222          000000          .WORD          0          ;;CONTAINS (($REGAD)+40)
    
```

```
949 001224 000000 $REG21: .WORD 0 ;;CONTAINS (($REGAD)+42)
950 001226 000000 $REG22: .WORD 0 ;;CONTAINS (($REGAD)+44)
951 001230 000000 $REG23: .WORD 0 ;;CONTAINS (($REGAD)+46)
952 001232 000000 $TMP0: .WORD 0 ;;USER DEFINED
953 001234 000000 $TMP1: .WORD 0 ;;USER DEFINED
954 001236 000000 $TMP2: .WORD 0 ;;USER DEFINED
955 001240 000000 $TMP3: .WORD 0 ;;USER DEFINED
956 001242 000000 $TMP4: .WORD 0 ;;USER DEFINED
957 001244 000000 $TMP5: .WORD 0 ;;USER DEFINED
958 001246 000000 $TMP6: .WORD 0 ;;USER DEFINED
959 001250 000000 $TMP7: .WORD 0 ;;USER DEFINED
960 001252 000000 $TMP10: .WORD 0 ;;USER DEFINED
961 001254 000000 $TMP11: .WORD 0 ;;USER DEFINED
962 001256 000000 $TMP12: .WORD 0 ;;USER DEFINED
963 001260 000000 $TMP13: .WORD 0 ;;USER DEFINED
964 001262 000000 $TMP14: .WORD 0 ;;USER DEFINED
965 001264 000000 $TMP15: .WORD 0 ;;USER DEFINED
966 001266 000000 $TMP16: .WORD 0 ;;USER DEFINED
967 001270 000000 $TMP17: .WORD 0 ;;USER DEFINED
968 001272 000000 $TMP20: .WORD 0 ;;USER DEFINED
969 001274 000000 $TMP21: .WORD 0 ;;USER DEFINED
970 001276 000000 $TMP22: .WORD 0 ;;USER DEFINED
971 001300 000000 $TMP23: .WORD 0 ;;USER DEFINED
972 001302 000000 $TIMES: 0 ;;MAX. NUMBER OF ITERATIONS
973 001304 000000 $ESCAPE: 0 ;;ESCAPE ON ERROR ADDRESS
974 001306 177607 000377 $BELL: .ASCIZ <207><377><377> ;;CODE FOR BELL
975 001312 077 $QUES: .ASCII /?/ ;;QUESTION MARK
976 001313 015 $CRLF: .ASCII <15> ;;CARRIAGE RETURN
977 001314 000012 $LF: .ASCIZ <12> ;;LINE FEED
978 ;:*****
979 .SBTTL APT MAILBOX-ETABLE
980
981 ;:*****
982 .EVEN
983 001316 $MAIL: ;;APT MAILBOX
984 001316 000000 $MSGTY: .WORD AMSGTY ;;MESSAGE TYPE CODE
985 001320 000000 $FATAL: .WORD AFATAL ;;FATAL ERROR NUMBER
986 001322 000000 $TESTN: .WORD ATESTN ;;TEST NUMBER
987 001324 000000 $PASS: .WORD APASS ;;PASS COUNT
988 001326 000000 $DEVCT: .WORD ADEVCT ;;DEVICE COUNT
989 001330 000000 $UNIT: .WORD AUNIT ;;I/O UNIT NUMBER
990 001332 000000 $MSGAD: .WORD AMSGAD ;;MESSAGE ADDRESS
991 001334 000000 $MSGLG: .WORD AMSGLG ;;MESSAGE LENGTH
992 001336 $ETABLE: ;;APT ENVIRONMENT TABLE
993 001336 000 $ENV: .BYTE AENV ;;ENVIRONMENT BYTE
994 001337 000 $ENVM: .BYTE AENVM ;;ENVIRONMENT MODE BITS
995 001340 000000 $SWREG: .WORD ASWREG ;;APT SWITCH REGISTER
996 001342 000000 $USWR: .WORD AUSWR ;;USER SWITCHES
997 001344 000000 $CPUOP: .WORD ACPUOP ;;CPU TYPE,OPTIONS
998 ;*
999 ;* BIT 15-11=CPU TYPE
1000 ;* 11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
1001 ;* 11/70=06,PDQ=07,Q=10
1002 ;* BIT 10=REAL TIME CLOCK
1003 ;* BIT 9=FLOATING POINT PROCESSOR
1004 001346 000 $MAMS1: .BYTE AMAMS1 ;;HIGH ADDRESS,M.S. BYTE
```

```

1005 001347 000 $MTYP1: .BYTE AMTYP1 ;;MEM. TYPE,BLK#1
1006 : * MEM.TYPE BYTE -- (HIGH BYTE)
1007 : * 900 NSEC CORE=001
1008 : * 300 NSEC BIPOLAR=002
1009 : * 500 NSEC MOS=003
1010 001350 000000 $MADR1: .WORD AMADR1 ;;HIGH ADDRESS,BLK#1
1011 : * MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF 'TYPE' ABOVE
1012 001352 000 $MAMS2: .BYTE AMAMS2 ;;HIGH ADDRESS,M.S. BYTE
1013 001353 000 $MTYP2: .BYTE AMTYP2 ;;MEM.TYPE,BLK#2
1014 001354 000000 $MADR2: .WORD AMADR2 ;;MEM.LAST ADDRESS,BLK#2
1015 001356 000 $MAMS3: .BYTE AMAMS3 ;;HIGH ADDRESS,M.S.BYTE
1016 001357 000 $MTYP3: .BYTE AMTYP3 ;;MEM.TYPE,BLK#3
1017 001360 000000 $MADR3: .WORD AMADR3 ;;MEM.LAST ADDRESS,BLK#3
1018 001362 000 $MAMS4: .BYTE AMAMS4 ;;HIGH ADDRESS,M.S.BYTE
1019 001363 000 $MTYP4: .BYTE AMTYP4 ;;MEM.TYPE,BLK#4
1020 001364 000000 $MADR4: .WORD AMADR4 ;;MEM.LAST ADDRESS,BLK#4
1021 001366 000000 $VECT1: .WORD AVECT1 ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
1022 001370 000000 $VECT2: .WORD AVECT2 ;;INTERRUPT VECTOR#2BUS PRIORITY#2
1023 001372 000000 $BASE: .WORD ABASE ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
1024 001374 000000 $DEVN: .WORD ADEVN ;;DEVICE MAP
1025 001376 000000 $CDW1: .WORD ACDW1 ;;CONTROLLER DESCRIPTION WORD#1
1026 001400 000000 $CDW2: .WORD ACDW2 ;;CONTROLLER DESCRIPTION WORD#2
1027 001402 000000 $DDW0: .WORD ADDW0 ;;DEVICE DESCRIPTOR WORD#0
1028 001404 000000 $DDW1: .WORD ADDW1 ;;DEVICE DESCRIPTOR WORD#1
1029 001406 000000 $DDW2: .WORD ADDW2 ;;DEVICE DESCRIPTOR WORD#2
1030 001410 000000 $DDW3: .WORD ADDW3 ;;DEVICE DESCRIPTOR WORD#3
1031 001412 000000 $DDW4: .WORD ADDW4 ;;DEVICE DESCRIPTOR WORD#4
1032 001414 000000 $DDW5: .WORD ADDW5 ;;DEVICE DESCRIPTOR WORD#5
1033 001416 000000 $DDW6: .WORD ADDW6 ;;DEVICE DESCRIPTOR WORD#6
1034 001420 000000 $DDW7: .WORD ADDW7 ;;DEVICE DESCRIPTOR WORD#7
1035 001422 000000 $DDW8: .WORD ADDW8 ;;DEVICE DESCRIPTOR WORD#8
1036 001424 000000 $DDW9: .WORD ADDW9 ;;DEVICE DESCRIPTOR WORD#9
1037 001426 000000 $DDW10: .WORD ADDW10 ;;DEVICE DESCRIPTOR WORD#10
1038 001430 000000 $DDW11: .WORD ADDW11 ;;DEVICE DESCRIPTOR WORD#11
1039 001432 000000 $DDW12: .WORD ADDW12 ;;DEVICE DESCRIPTOR WORD#12
1040 001434 000000 $DDW13: .WORD ADDW13 ;;DEVICE DESCRIPTOR WORD#13
1041 001436 000000 $DDW14: .WORD ADDW14 ;;DEVICE DESCRIPTOR WORD#14
1042 001440 000000 $DDW15: .WORD ADDW15 ;;DEVICE DESCRIPTOR WORD#15
1043
1044
1045 001442 $ETEND:
1046
    
```



```
1047 .SBTTL ERROR POINTER TABLE
1048
1049 ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
1050 ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
1051 ;*LOCATION $IITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
1052 ;*NOTE1: IF $IITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
1053 ;*NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
1054
1055 ;* EM ;;POINTS TO THE ERROR MESSAGE
1056 ;* DH ;;POINTS TO THE DATA HEADER
1057 ;* DT ;;POINTS TO THE DATA
1058 ;* DF ;;POINTS TO THE DATA FORMAT
1059
1060
1061 001442 $ERRTB:
1062 ;ITEM NUMBER
1063 001442 062753 .WORD EM1
1064 001444 063001 .WORD EM2
1065 001446 063035 .WORD EM3
1066 001450 063063 .WORD EM4
1067
1068
1069 .SBTTL ACT11 HOOKS
1070
1071 ;*****
1072 ;HOOKS REQUIRED BY ACT11
1073 001452 $SVPC= ;SAVE PC
1074 000046 .=46
1075 000046 057370 $ENDAD ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
1076 000052 000052 .=52
1077 000052 000000 .WORD 0 ;;2)SET LOC.52 TO ZERO
1078 001452 .= $SVPC ;; RESTORE PC
1079
1080 .SBTTL APT PARAMETER BLOCK
1081
1082 ;*****
1083 ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
1084 ;*****
1085 001452 .$X= ;;SAVE CURRENT LOCATION
1086 000024 .=24 ;;SET POWER FAIL TO POINT TO START OF PROGRAM
1087 000024 000200 200 ;;FOR APT START UP
1088 000044 000044 .=44 ;;POINT TO APT INDIRECT ADDRESS PNTR.
1089 001452 $APTHDR ;;POINT TO APT HEADER BLOCK
1090 001452 .= $X ;;RESET LOCATION COUNTER
1091
1092 ;*****
1093 ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
1094 ;INTERFACE SPEC.
1095
1096 $APTHD:
1097 $HIBTS: .WORD 0 ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
1098 $MBADR: .WORD $MAIL ;;ADDRESS OF APT MAILBOX (BITS 0-15)
1099 $TSTM: .WORD 10 ;;RUN TIM OF LONGEST TEST
1100 $PASTM: .WORD 40 ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
1101 $UNITM: .WORD 0 ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
1102 .WORD $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
```

```

1103 001466
1104
1105
1106 001466 012706 001100
1107 001472 005026
1108 001474 022706 001140
1109 001500 001374
1110 001502 012706 001100
1111
1112 001506 012737 057464 000020
1113 001514 012737 000340 000022
1114 001522 012737 057744 000030
1115 001530 012737 000340 000032
1116 001536 012737 062100 000034
1117 001544 012737 000340 000036
1118 001552 012737 062166 000024
1119 001560 012737 000340 000026
1120 001566 016767 055526 055516
1121 001574 005067 177502
1122 001600 005067 177500
1123 001604 112767 000001 177303
1124
1125
1126 001612 012737 057434 000014
1127 001620 012737 000340 000016
1128 001626 012767 000002 055600
1129 001634 012737 001662 000010
1130 001642 005046
1131 001644 012746 001652
1132 001650 000006
1133 001652 012767 000006 055554 64$:
1134 001660 000402
1135 001662 062706 000010 65$:
1136 001666 012737 000012 000010 66$:
1137 001674 005067 055542
1138 001700 012767 001700 177200
1139 001706 012767 001706 177174
1140
1141
1142 001714 013746 000004
1143 001720 012737 001754 000004
1144 001726 012767 177570 177204
1145 001734 012767 177570 177200
1146 001742 022777 177777 177170
1147 001750 001012
1148
1149 001752 000403
1150 001754 012716 001762 67$:
1151 001760 000002
1152 001762 012767 000176 177150 68$:
1153 001770 012767 000174 177144
1154 001776 012637 000004 69$:
1155
1156 002002 005067 177316
1157 002006 132767 000200 177323
1158 002014 001403
    
```

```

START:
.SBTTL INITIALIZE THE COMMON TAGS
::CLEAR THE COMMON TAGS ($CMTAG) AREA
MOV #CMTAG,R6 ::FIRST LOCATION TO BE CLEARED
CLR (R6)+ ::CLEAR MEMORY LOCATION
CMP #SWR,R6 ::DONE?
BNE -6 ::LOOP BACK IF NO
MOV #STACK,SP ::SETUP THE STACK POINTER
::INITIALIZE A FEW VECTORS
MOV #SCOPE,@IOTVEC ::IOT VECTOR FOR SCOPE ROUTINE
MOV #340,@IOTVEC+2 ::LEVEL 7
MOV #ERROR,@EMTVEC ::EMT VECTOR FOR ERROR ROUTINE
MOV #340,@EMTVEC+2 ::LEVEL 7
MOV #TRAP,@TRAPVEC ::TRAP VECTOR FOR TRAP CALLS
MOV #340,@TRAPVEC+2 ::LEVEL 7
MOV #PWRDN,@PWRVEC ::POWER FAILURE VECTOR
MOV #340,@PWRVEC+2 ::LEVEL 7
MOV $ENDCT,$EOPCT ::SETUP END-OF-PROGRAM COUNTER
CLR $TIMES ::INITIALIZE NUMBER OF ITERATIONS
CLR $ESCAPE ::CLEAR THE ESCAPE ON ERROR ADDRESS
MOVB #1,$ERMAX ::ALLOW ONE ERROR PER TEST
::INITIALIZE THE 'T-BIT' TRAP VECTOR. THEN LOAD LOCATION '$RTRN', IN
::THE 'END-OF-PASS' ($EOP) ROUTINE, WITH A 'RTI' OR 'RTT'.
MOV #RTRN,@TBITVEC ::SET 'T' BIT VECTOR TO RTRN
MOV #340,@TBITVEC+2 ::LEVEL 7
MOV #RTI,$RTRN ::SET RTRN TO A RTI
MOV #65,@RESVEC ::TRY TO DO A RTT
CLR -(SP) ::DUMMY PS
MOV #64,-(SP) ::AND PC
RTT ::TRY THE RTT
MOV #RTT,$RTRN ::RTT IS LEGAL--SET RTRN TO A RTT
BR 66$
65$: ADD #10,SP ::RTT ILLEGAL--CLEAN OFF THE STACK
66$: MOV #RESVEC+2,@RESVEC ::RESTORE TRAP CATCHER
CLR $TBIT ::CLEAR 'T' BIT SWITCH
MOV #,$LPADR ::INITIALIZE THE LOOP ADDRESS FOR SCOPE
MOV #,$LPERR ::SETUP THE ERROR LOOP ADDRESS
::SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
::EQUAL TO A '-1', SETUP FOR A SOFTWARE SWITCH REGISTER.
MOV @ERRVEC,-(SP) ::SAVE ERROR VECTOR
MOV #67,@ERRVEC ::SET UP ERROR VECTOR
MOV #DSWR,SWR ::SETUP FOR A HARDWARE SWICH REGISTER
MOV #DDISP,DISPLAY ::AND A HARDWARE DISPLAY REGISTER
CMP #-1,@SWR ::TRY TO REFERENCE HARDWARE SWR
BNE 69$ ::BRANCH IF NO TIMEOUT TRAP OCCURRED
::AND THE HARDWARE SWR IS NOT = -1
BR 68$ ::BRANCH IF NO TIMEOUT
67$: MOV #68$(SP) ::SET UP FOR TRAP RETURN
68$: MOV #SWREG,SWR ::POINT TO SOFTWARE SWR
MOV #DISPREG,DISPLAY
69$: MOV (SP)+,@ERRVEC ::RESTORE ERROR VECTOR
CLR $PASS ::CLEAR PASS COUNT
BITB #APTSIZE,$ENVM ::TEST USER SIZE UNDER APT
BEQ 70$ ::YES,USE NON-APT SWITCH
    
```



```

1215                                     ;ODD ADDRESS TRAP WILL OCCUR.
1216 002252
1217 002252 010004 A1:
1218 002254 042704 030020 A11: MOV R0,R4
1219 002260 170104 LDFPS #30020,R4
1220                                     ;TEST INSTRUCTION.
1221 002262 012701 177777
1222 002266 170201 A12: MOV #-1,R1
1223 002270 012737 062534 000244 STFPS R1
1224 002276 010004 MOV #FPSPUR,@#FPVECT ;TEST INSTRUCTION.
1225 002300 042704 030020 MOV R0,R4 ;SET UP FOR UNEXPECTED TRAPS.
1226 002304 012737 062566 000004 BIC #30020,R4 ;MASK OFF UNSETTABLE BITS.
1227 002312 012737 062604 000010 MOV #CPSPUR,@#ERRVECT
1228 002320 020401 MOV #CPTWO,@#10
1229                                     ;COMPARE DATA EXPECTED WITH
1230 002322 001401 BEQ A2 ;THE DATA READ.
1231 002324 104004 ERROR 4 ;IF NOT EQUAL GO REPORT ERROR.
1232
1233 002326 012700 000001 A2: MOV #1,R0 ;NEXT PATTERN WILL BE ALL ZERO
1234 002332 077031 SOB R0,A1 ;DECREMENT COUNT PATTERN
1235 002334 000443 BR ADONE ;UNTIL IT IS ZERO.
1236
1237
1238 ;UNABLE TO DECODE FPP INSTRUCTION. TRAPPED TO 244.
1239 002336 011637 001236 AERR1: MOV (SP),@#TMP2 ;SAVE PC OF TRAP.
1240 002342 022626 CMP (SP)+,(SP)+
1241 002344 104004 1$: ERROR 4
1242 002346 000436 BR ADONE
1243
1244 ;UNABLE TO DECODE INSTRUCTION. TRAPPED TO 10.
1245 002350 021627 002254 AERR2: CMP (SP),#A11+2 ;DID TRAP OCCUR OF FPP INSTRUCTION?
1246 002354 001405 BEQ 1$
1247 002356 021627 002270 CMP (SP),#A12+2
1248 002362 001402 BEQ 1$
1249 002364 000137 062604 JMP @#CPTWO ;IF NOT FPP INSTRUCTION THEN
1250                                     ;REPORT SPURIOUS TRAP TO 10.
1251
1252 002370 011637 001236 1$: MOV (SP),@#TMP2 ;OTHERWISE REPORT IR DECODE ERROR.
1253 002374 022626 CMP (SP)+,(SP)+
1254 002376 104004 2$: ERROR 4
1255 002400 000421 BR ADONE
1256
1257 ;TRAP TO 4 HANDLER:
1258 002402 021627 002254 AERR3: CMP (SP),#A11+2 ;DID THE TRAP OCCUR ON THE
1259 002406 001405 BEQ 1$ ;LDFPS INSTRUCTION?
1260 002410 021627 002270 CMP (SP),#A12+2 ;OR THE STFPS INSTRUCTION?
1261 002414 001407 BEQ 2$
1262 002416 000137 062566 JMP @#CPSPUR ;IF NEITHER THEN REPORT
1263                                     ;UNEXPECTED TRAP TO 4.
1264
1265 002422 011637 001236 1$: MOV (SP),@#TMP2
1266 002426 022626 CMP (SP)+,(SP)+
1267 002430 104004 15$: ERROR 4
1268 002432 000404 BR ADONE
1269
1270 002434 011637 001236 2$: MOV (SP),@#TMP2
    
```



N 2

```
1271 002440 022626
1272 002442 104004
1273
1274 002444
1275 002444 104413
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288 002446 000004
1289 002450 104414
1290 002452 012700 000017
1291
1292 002456
1293 002456 170100
1294
1295 002460
1296 002460 170000
1297
1298 002462 013703 177776
1299 002466 042703 177760
1300 002472 020003
1301 002474 001002
1302
1303 002476 077011
1304 002500 000422
1305
1306 002502
1307 002502 170201
1308 002504 012737 002460 001236
1309
1310 002512 020001
1311 002514 001006
1312
1313 002516 010337 001240
1314 002522 010037 001242
1315 002526 104001
1316 002530 000762
1317
1318 002532
1319 002532 010037 001240
1320 002536 010137 001242
1321 002542 104001
1322 002544 000754
1323
1324 002546
1325 002546 104413
1326
```

```

                CMP      (SP)+,(SP)+
25$:            ERROR    4
ADONE:         RSETUP
                ;GO INITIALIZE THE FPS AND STACK; AND
                ;SEE IF THE USER HAS EXPRESSED
                ;THE DESIRE TO CHANGE THE SOFTWARE
                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                ;THE USER TYPED CONTROL G?).

:*****
:*TEST 2      CFCC TEST
:*
:*THIS IS A TEST OF THE COPY CONDITION CODES INSTRUCTION, CFCC.
:*
:*****
TST2:          SCOPE
                LPERR
                MOV      #17,R0
                ;SET UP THE LOOP ON ERROR ADDRESS.
                ;R0 CONTAINS TO TEST PATTERN.
B1:            LDFPS    R0
                ;LOAD THE TEST PATTERN
B2:            CFCC
                ;COPY CONDITION CODES.
                MOV      @#PSW,R3
                BIC      #177760,R3
                ;SEE IF PATTERN TRANSFERED.
                CMP      R0,R3
                BNE      BERR
B3:            SOB      R0,B1
                BR       BDONE
BERR:          STFPS    R1
                ;WAS FPS MODIFIED BY CFCC?
                MOV      #B2,@#STMP2
                CMP      R0,R1
                BNE      BERR1
1$:            MOV      R3,@#STMP3
                MOV      R0,@#STMP4
                ERROR    1
                BR       B3
BERR1:         MOV      R0,@#STMP3
                MOV      R1,@#STMP4
1$:            ERROR    1
                BR       B3
BDONE:         RSETUP
                ;GO INITIALIZE THE FPS AND STACK; AND
                ;SEE IF THE USER HAS EXPRESSED
```

;THE DESIRE TO CHANGE THE SOFTWARE  
 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS  
 ;THE USER TYPED CONTROL G?).

```

1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342 002550 000004
1343 002552 104414
1344 002554 012737 000760 001244
1345 002562 012737 000202 001250
1346 002570 005000
1347
1348 002572 170100
1349 002574 012737 002602 001236
1350
1351 002602 170001
1352
1353 002604 170201
1354 002606 005002
1355 002610 020201
1356 002612 001403
1357 002614 010137 001126
1358 002620 104001
1359
1360 002622
1361 002622 104414
1362 002624 012700 147757
1363
1364 002630 170100
1365 002632 012737 002640 001236
1366 002640 170001
1367
1368 002642 170201
1369 002644 012702 147557
1370 002650 020102
1371 002652 001403
1372 002654 010137 001126
1373 002660 104001
1374
1375 002662
1376 002662 104414
1377 002664 012737 000203 001250
1378 002672 012700 147757
1379
1380 002676 170100
1381 002700 012737 002706 001236
1382 002706 170011

;*****
;*TEST 3 SETF, SETD, SETI AND SETL TEST
;*
;*THIS IS A TEST OF THE SETF, SETD, SETI AND SETL INSTRUCTIONS.
;*EACH INSTRUCTION IS EXECUTED WITH THE FPS CONTAINING
;*ALL ONES AND ALSO WITH THE FPS CLEAR. THE RESULT OF EACH
;*SITUATION IS CHECKED.
;*$TEMP2 CONTAINS INSTRUCTION TRIED TO PERFORM
;*$BDDAT CONTAINS BAD FPS CONTAIN.
;*****
TST3: SCOPE
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
MOV #760,@#$TMP5
MOV #202,@#$TMP7
CLR R0
LDFPS R0 ;CLEAR THE FPS.
MOV #C15,@#$TMP2
C15: SETF ;TEST INSTRUCTION.
STFPS R1 ;GET RESULT.
CLR R2
CMP R2,R1 ;DID AN ERROR OCCUR?
BEQ 1$
MOV R1,@#$BDDAT ;STORE BAD DATA
ERROR 1
1$:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
MOV #147757,R0
LDFPS R0 ;PUT 147757 IS FPS
MOV #C25,@#$TMP2
C25: SETF ;CLEAR FD BIT.
STFPS R1 ;GET RESULT
MOV #147557,R2
CMP R1,R2 ;RESULT CORRECT.
BEQ 1$
MOV R1,@#$BDDAT ;STORE BAD DATA
ERROR 1
1$:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
MOV #203,@#$TMP7
MOV #147757,R0
LDFPS R0 ;LOAD 147757 INTO FPS.
MOV #C35,@#$TMP2
C35: SETD ;SETD FD BIT.
    
```

```

1383
1384 002710 170201          STFPS  R1
1385 002712 012702 147757  MOV    #147757,R2
1386 002716 020102          CMP    R1,R2          ;RESULT CORRECT?
1387 002720 001403          BEQ    1$
1388 002722 010137 001126  MOV    R1,@#SBDDAT   ;STORE BAD DATA
1389 002726 104001          ERROR  1
1390
1391 002730          1$:
1392 002730 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
1393 002732 005000          C4:  CLR    R0
1394 002734 170100          LDFPS  R0          ;CLEAR FPS.
1395 002736 012737 002744 001236  MOV    #C45,@#STMP2
1396
1397 002744 170011          C45: SETD          ;SET FD BIT.
1398
1399 002746 170201          STFPS  R1          ;GET RESULT.
1400 002750 012702 000200  MOV    #200,R2
1401 002754 020102          CMP    R1,R2          ;RESULT CORRECT?
1402 002756 001403          BEQ    1$
1403 002760 010137 001126  MOV    R1,@#SBDDAT   ;STORE BAD DATA
1404 002764 104001          ERROR  1
1405
1406 002766          1$:
1407 002766 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
1408 002770 012737 000204 001250  C5:  MOV    #204,@#STMP7
1409 002776 005000          CLR    R0
1410
1411 003000 170100          LDFPS  R0          ;CLEAR FPS
1412 003002 012737 003010 001236  MOV    #C55,@#STMP2
1413
1414 003010 170002          C55: SETI          ;CLEAR FL BIT.
1415
1416 003012 170201          STFPS  R1          ;GET RESULT.
1417 003014 005002          CLR    R2
1418 003016 020201          CMP    R2,R1          ;RESULT CORRECT?
1419 003020 001403          BEQ    1$
1420 003022 010137 001126  MOV    R1,@#SBDDAT   ;STORE BAD DATA
1421 003026 104001          ERROR  1
1422
1423 003030          1$:
1424 003030 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
1425 003032 012700 147757  C6:  MOV    #147757,R0
1426 003036 170100          LDFPS  R0          ;PUT 147757 INTO FPS
1427 003040 012737 003046 001236  MOV    #C65,@#STMP2
1428
1429 003046 170002          C65: SETI          ;CLEAR FL BIT.
1430
1431 003050 170201          STFPS  R1          ;GET THE RESULT.
1432 003052 012702 147657  MOV    #147657,R2
1433 003056 020102          CMP    R1,R2          ;RESULT CORRECT?
1434 003060 001403          BEQ    1$
1435 003062 010137 001126  MOV    R1,@#SBDDAT   ;STORE BAD DATA
1436 003066 104001          ERROR  1
1437
1438 003070          1$:
    
```

```

1439 003070 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
1440 003072 012737 000205 001250 C7: MOV #205,@#STMP7
1441 003100 012700 147757 MOV #147757,R0
1442 003104 170100 LDFPS R0 ;SET FPS TO 147757.
1443 003106 012737 003114 001236 MOV #C75,@#STMP2
1444
1445 003114 170012 C75: SETL ;SET FL BIT.
1446
1447 003116 170201 STFPS R1 ;GET THE RESULT.
1448 003120 012702 147757 MOV #147757,R2
1449 003124 020102 CMP R1,R2 ;RESULT CORRECT?
1450 003126 001403 BEQ 1$
1451 003130 010137 001126 MOV R1,@#SBDDAT ;STORE BAD DATA
1452 003134 104001 ERROR 1
1453
1454 003136 1$: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
1455 003136 104414 C8: CLR R0
1456 003140 005000 LDGPS R0 ;CLEAR FPS.
1457 003142 170100 MOV #C85,@#STMP2
1458 003144 012737 003152 001236
1459
1460 003152 170012 C85: SETL ;SET FL BIT.
1461
1462 003154 170201 STFPS R1
1463 003156 012702 000100 MOV #100,R2
1464 003162 020102 CMP R1,R2 ;RESULT CORRECT.
1465 003164 001403 BEQ 1$
1466 003166 010137 001126 MOV R1,@#SBDDAT ;STORE BAD DATA
1467 003172 104001 ERROR 1
1468
1469 003174 1$: CDONE: RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
1470 003174 ;SEE IF THE USER HAS EXPRESSED
1471 003174 104413 ;THE DESIRE TO CHANGE THE SOFTWARE
1472 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
1473 ;THE USER TYPED CONTROL G?).
1474
1475
1476
1477
1478
1479 *****
1480 ;*TEST 4 ILLEGAL FPP OP CODES AND STST TEST
1481 ;*
1482 ;*THIS IS A TEST OF THE FPP OPERATION CODES:
1483 ;* 170003
1484 ;* 170004
1485 ;* :
1486 ;* 170010
1487 ;* 170013
1488 ;* 170014
1489 ;* :
1490 ;* 170077
1491 ;*THESE ARE ILLEGAL INSTRUCTIONS AND (WITH INTERRUPTS ENABLED)
1492 ;*SHOULD CAUSE A TRAP TO 244.
1493 ;*ALSO TESTED HERE IS THE INSTRUCTION:
1494 ;* STST R1
1495 ;*WHICH SHOULD PUT THE FEC CODE 2 IN R1, AFTER ANY OF THE ABOVE
    
```



```

1495          ;*OP CODES IS EXECUTED.
1496          ;*
1497          ;*****
1498 003176 000004          TST4:  SCOPE
1499 003200 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
1500 003202 012705 170003  MOV      #170003,R5          ;INITIAL OP CODE.
1501 003206 012737 003412 000004  MOV      #DERR2,@#ERRVECT
1502 003214 012737 003316 000244  MOV      #DERR1,@#FPVECT
1503
1504 003222 005000          D1:    CLR      R0
1505 003224 170100          LDFPS   R0          ;CLEAR FPS.
1506 003226 005002          CLR      R2
1507 003230 010537 003246  MOV      R5,@#D2          ;SET UP THE ILLEGAL INSTRUCTION.
1508 003234 010537 001244  MOV      R5,@#$TMP5
1509 003240 012737 003246 001236  MOV      #D2,@#$TMP2
1510 003246 000000          D2:    .WORD  0
1511 003250 170000          D3:    CFCC
1512 003252 005202          INC      R2
1513 003254 005202          D4:    INC      R2
1514
1515 003256 170201          STFPS   R1          ;REPORT FAILURE. DID NOT TRAP.
1516 003260 010137 001240  MOV      R1,@#$TMP3
1517 003264 104001          1$:    ERROR  1
1518
1519 003266 022705 170010  D5:    CMP      #170010,R5          ;COMPUTE NEXT OP CODE
1520 003272 001003          BNE     D6
1521 003274 012705 170013  MOV      #170013,R5
1522 003300 000750          BR      D1
1523
1524 003302 022705 170077  D6:    CMP      #170077,R5
1525 003306 001001          BNE     D7
1526 003310 000452          BR      DDONE
1527 003312 005205          D7:    INC      R5
1528 003314 000742          BR      D1
1529
1530 003316 022716 003250  DERR1: CMP      #D3,(SP)          ;DID TRAP OCCUR ON TEST INSTRUCTION?
1531 003322 001402          BEQ     1$
1532 003324 000137 062534  JMP      @#FPSPUR
1533
1534 003330 022626          1$:    CMP      (SP)+,(SP)+
1535 003332 170201          STFPS   R1          ;GET THE FPS AND SEE IF IT IS
1536 003334 022701 100000  CMP      #100000,R1          ;SET CORRECTLY.
1537 003340 001406          BEQ     3$
1538
1539 003342 012737 100000 001240  MOV      #100000,@#$TMP3
1540 003350 010137 001242  MOV      R1,@#$TMP4
1541 003354 104001          2$:    ERROR  1
1542
1543 003356 012704 000001  3$:    MOV      #1,R4
1544 003362 170304          D8:    STST   R4          ;GET THE FEC CODE. NOTE THAT
1545                                     ;IF THE DESTINATION MODE IS
1546                                     ;IMPROPERLY DECODED AN ODD
1547                                     ;ADDRESS TRAP TO 4 SHOULD OCCUR.
1548 003364 022704 000002          CMP      #2,R4          ;WAS FEC CORRECT?
1549 003370 001001          BNE     D9
1550 003372 000735          BR      D5
    
```

```

1551
1552 003374
1553 003374 012737 003362 001240 D9:          MOV      #D8,@#STMP3          ;REPORT STST FAILURE
1554 003402 010437 001242          MOV      R4,@#STMP4
1555 003406 104001          1$:      ERROR      1
1556 003410 000726          BR       D5
1557
1558 003412 022716 003364 DERR2:    CMP      #D8+2,(SP)          ;DID THE TRAP OCCUR ON THE
1559 003416 001402          BEQ      D10                ;STST INSTRUCTION?
1560 003420 000137 062566          JMP      @#CPSPUR
1561
1562 003424
1563 003424 011637 001236 D10:      MOV      (SP),@#STMP2
1564 003430 022626          CMP      (SP)+,(SP)+
1565 003432 104001          1$:      ERROR      1
1566 003434 000714          BR       D5
1567
1568 003436
1569 003436 104413 DDCNE:    RSETUP                ;GO INITIALIZE THE FPS AND STACK; AND
1570                                     ;SEE IF THE USER HAS EXPRESSED
1571                                     ;THE DESIRE TO CHANGE THE SOFTWARE
1572                                     ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
1573                                     ;THE USER TYPED CONTROL G?).
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584 003440 000004
1585 003442 104414
1586 003444 012737 003546 000244
1587
1588 003452 012700 040000
1589 003456 170100
1590 003460 012737 003466 001236
1591 003466
1592 003466 170020
1593 003470 170000
1594
1595 003472 170201
1596 003474 022701 140000
1597 003500 001005
1598
1599 003502 170304
1600 003504 022704 000002
1601 003510 001010
1602 003512 000431
1603
1604 003514
1605 003514 010137 001240
1606 003520 012737 140000 001242
    ;*****
    ;*TEST 5          FID, INTERRUPT DISABLE, BIT TEST
    ;*
    ;*THIS IS A TEST OF FPS BIT 14 (FID) OR FLOATING INTERRUPT DISABLE.
    ;*AN ILLEGAL INSTRUCTION IS EXECUTED WITH FID=1. NO INTERRUPT SHOULD
    ;*OCCUR.
    ;*
    ;*****
TST5:    SCOPE
          LPERR
          MOV      #EERR2,@#FPVECT ;SET UP THE LOOP ON ERROR ADDRESS.
          ;SETUP FOR THE INTERRUPT.
E1:      MOV      #40000,R0
          LDFPS   R0
          MOV      #E3,@#STMP2 ;SET FID.
E2:
E3:      .WORD   170020 ;ILLEGAL FPP INSTRUCTION.
E4:      CFCC
          STFPS   R1 ;SEE IF ERROR WAS DETECTED.
          CMP     #140000,R1
          BNE     EERR0
          STST    R4 ;SEE IF FEC=2
          CMP     #2,R4
          BNE     EERR1
          BR      EDONE
EERR0:   MOV      R1,@#STMP3 ;REPORT FPS INCORRECTLY SET.
          MOV     #140000,@#STMP4
    
```

```
1607 003526 104001 1$: ERROR 1
1608 003530 000422 BP EDONE
1609
1610 003532 EERR1: ;REPORT FEC NOT 2.
1611 003532 010537 001240 MOV R5,@$TMP3
1612 003536 010437 001242 MOV R4,@$TMP4
1613 003542 104001 1$: ERROR 1
1614 003544 000414 BR EDONE
1615
1616 003546 021627 003470 EERR2: CMP (SP),#E4 ;DID THE ILLLEGAL INSTRUCTION TRAP?
1617 003552 001402 BEQ 1$
1618 003554 000137 062534 JMP @$FSPUR
1619
1620 003560 1$:
1621 003560 011637 001236 MOV (SP),@$TMP2
1622 003564 022626 CMP (SP)+,(SP)+
1623 003566 170201 STFPS R1
1624 003570 010137 001240 MOV R1,$TMP3
1625 003574 104001 2$: ERROR 1
1626
1627 003576 EDONE:
1628 003576 104413 RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
1629 ;SEE IF THE USER HAS EXPRESSED
1630 ;THE DESIRE TO CHANGE THE SOFTWARE
1631 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
1632 ;THE USER TYPED CONTROL G?).
1633
1634
1635 ;*****
1636 ;*TEST 6 LDD AND STD, WITH SRC AND DST MODE 1, TEST
1637 ;*
1638 ;*THIS IS A TEST OF BOTH THE INSTRUCTION:
1639 ;* LDD (R0),ACO
1640 ;*AND THE INSTRUCTION:
1641 ;* STD ACO,(R0)
1642 ;*MOST OF THE FAILURES ARE ISOLATED TO THE SRC OR DST FLOWS. NOTE
1643 ;*THAT THE INTEGRITY OF ACO HAS NOT BEEN ASSURED. THIS MEANS THAT
1644 ;*IN SOME CASES IT WILL BE IMPOSSIBLE TO ISOLATE CERTAIN DATA PATTERN
1645 ;*FAILURES TO EITHER THE FLOWS OR THIS ACCUMULATOR.
1646 ;*
1647 ;*****
1648 003600 000004 TST6: SCOPE
1649
1650 003602 F1:
1651 003602 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
1652 003604 012737 003654 001236 MOV #F3,$TMP2
1653 003612 005000 CLR R0
1654 003614 170100 LDFPS R0
1655 003616 170011 SETD
1656 003620 012701 005372 MOV #FDATIO,R1 ;SET UP THE LOAD DATA.
1657 003624 012702 005436 MOV #FXDAT0,R2
1658 003630 012703 000010 MOV #10,R3
1659
1660 003634 012221 F2: MOV (R2)+,(R1)+
1661 003636 077302 SOB R3,F2
1662
```

H 3

CJKDCA-A KEF11-A FP DIAG PART 1 MACY11 30A(1052) 11-JUN-79 11:50 PAGE 33  
 CJKDCA.P11 08-JUN-79 08:40 T6 LDD AND STD, WITH SRC AND DST MODE 1, TEST SEQ 0033

1663	003640	012700	005402		MOV	#FDAT14,R0		;SETUP R0 FOR THE LDD (R0).ACO.
1664	003644	012737	005072	000004	MOV	#FERR20,@#ERRVECT		;IF THE SRC FLOWS FAIL THEN
1665								;AN ODD ADDRESS MAY OCCUR.
1666	003652	005003			CLR	R3		
1667								
1668	003654	172410			F3:	LDD	(R0).ACO	
1669	003656	005203			F4:	INC	R3	
1670	003660	005203				INC	R3	
1671								
1672	003662	020027	005402		CMP	R0,#FDAT14		;WAS R0 AFFECTED?
1673	003666	001402			BEQ	F5		
1674	003670	000137	004236		JMP	@#FERR1		
1675								
1676	003674	020327	000002		F5:	CMP	R3,#2	;SEE IF THE PC WAS ADVERSELY
1677	003700	001402				BEQ	1\$	;AFFECTED DURING THE INSTRUCTION.
1678	003702	000137	004334		JMP	@#FERR2		
1679								
1680	003706	012701	005372		1\$:	MOV	#FDAT10,R1	;MAKE SURE THE SOURCE DATA WAS
1681	003712	012702	005436			MOV	#FXDAT0,R2	;NOT AFFECTED.
1682	003716	012703	000010			MOV	#10,R3	
1683	003722	022122			2\$:	CMP	(R1)+,(R2)+	
1684	003724	001402				BEQ	3\$	
1685	003726	000137	004200		JMP	@#FERRO		
1686	003732	077305			3\$:	SOB	R3,2\$	
1687								
1688	003734	170201			STFPS	R1		;MAKE SURE THE FPS IS CORRECT.
1689	003736	022701	000200		CMP	#200,R1		
1690	003742	001402			BEQ	F6		
1691	003744	000137	005052		JMP	@#FERR11		
1692								
1693	003750				F6:			
1694	003750	104414			LPERR			;SET UP THE LOOP ON ERROR ADDRESS.
1695	003752	012737	004014	001236	MOV	#F10,@#\$TMP2		
1696								
1697	003760	012703	177777		MOV	#-1,R3		
1698	003764	012704	000010		MOV	#10,R4		
1699	003770	012705	005414		MOV	#FDAT00,R5		;SET UP THE OUTPUT DATA BUFFER.
1700	003774	010325			F7:	MOV	R3,(R5)+	
1701	003776	077402				SOB	R4,F7	
1702								
1703	004000	012700	005424		MOV	#FDAT04,R0		;SET UP R0 FOR DST MODE 1 REG 0.
1704	004004	012737	005232	000004	MOV	#FERR25,@#ERRVECT		;IF THE DST FLOWS FAIL AN ODD
1705								;ADDRESS COULD OCCUR.
1706	004012	005003			CLR	R3		
1707								
1708	004014	174010			F10:	STD	ACO,(R0)	;TEST INSTRUCTION.
1709	004016	005203			F11:	INC	R3	
1710	004020	005203				INC	R3	
1711								
1712	004022	020027	005424		CMP	R0,#FDAT04		;WAS R0 MODIFIED?
1713	004026	001402			BEQ	F12		
1714	004030	000137	004374		JMP	@#FERR3		
1715								
1716	004034	020327	000002		F12:	CMP	R3,#2	;WAS THE PC AFFECTED CORRECTLY?
1717	004040	001402				BEQ	F135	
1718	004042	000167	000320		JMP	FERR4		

```

1719
1720 004046 012701 005414      F135:  MOV    #FDAT00,R1
1721 004052 012702 005436      MOV    #FXDAT0,R2
1722
1723 004056 022122                CMP    (R1)+,(R2)+      ;SEE IF THE DATA WAS OUTPUT
1724 004060 001402                BEQ    F13              ;TO THE TARGET AREA CORRECTLY.
1725 004062 000137 004472      JMP    @#FERR5
1726
1727 004066 022122      F13:   CMP    (R1)+,(R2)+
1728 004070 001402      BEQ    F14
1729 004072 000137 004472      JMP    @#FERR5
1730
1731 004076 022122      F14:   CMP    (R1)+,(R2)+
1732 004100 001402      BEQ    F15
1733 004102 000137 004472      JMP    @#FERR5
1734
1735 004106 022122      F15:   CMP    (R1)+,(R2)+
1736 004110 001402      BEQ    F16
1737 004112 000137 004472      JMP    @#FERR5
1738
1739 004116 022122      F16:   CMP    (R1)+,(R2)+
1740 004120 001402      BEQ    F17
1741 004122 000137 005016      JMP    @#FERR10
1742
1743 004126 022122      F17:   CMP    (R1)+,(R2)+
1744 004130 001402      BEQ    F20
1745 004132 000137 004526      JMP    @#FERR6
1746
1747 004136 022122      F20:   CMP    (R1)+,(R2)+
1748 004140 001402      BEQ    F21
1749 004142 000167 000514      JMP    FERR7
1750
1751 004146 022122      F21:   CMP    (R1)+,(R2)+
1752 004150 001402      BEQ    F22
1753 004152 000137 005016      JMP    @#FERR10
1754
1755 004156 005001      F22:   CLR    R1
1756 004160 170201      STFPS  R1              ;MAKE SURE FPS IS CORRECT.
1757 004162 022701 000200      CMP    #200,R1
1758 004166 001402      BEQ    F23
1759 004170 000137 005052      JMP    @#FERR11
1760 004174 000137 005456      F23:   JMP    @#FDONE
1761
1762 004200      FERRO:
1763 004200 012737 005436 001240      MOV    #FXDAT0,@#STMP3      ;SOURCE DATA AFFECTED BY
1764 004206 012737 005450 001242      MOV    #FXDAT0+12,@#STMP4   ;THE LDD INSTRUCTION.
1765 004214 012737 005372 001244      MOV    #FDAT10,@#STMP5
1766 004222 012737 005404 001246      MOV    #FDAT10+12,@#STMP6
1767 004230 104001      1$:   ERROR 1
1768 004232 000137 005456      JMP    @#FDONE
1769
1770 004236 012737 005402 001242      FERR1: MOV    #FDAT14,@#STMP4      ;FSRC FLOWS FAILURE.
1771 004244 010037 001240      MOV    R0,@#STMP3
1772 004250 012737 000762 001244      MOV    #762,@#STMP5
1773 004256 012737 000321 001250      MOV    #321,@#STMP7
1774
    
```



```

1775 004264 022700 005372          CMP    #FDATIO,R0          ;FSRC MODE 4?
1776 004270 001004          BNE    1$
1777 004272 012737 000324 001246    MOV    #324,@#STMP6
1778 004300 000412          BR     4$
1779
1780 004302 022700 005412          1$:   CMP    #FDATI4+10,R0      ;FSRC MODE 2?
1781 004306 001004          BNE    2$
1782 004310 012737 000322 001246    MOV    #322,@#STMP6
1783 004316 000403          BR     4$
1784
1785 004320          2$:
1786 004320          3$:
1787 004320 104001          ERROR  1                   ;RO WAS MODIFIED, LDD
1788 004322 000137 005456    JMP    @#FDONE
1789 004326          4$:
1790 004326          5$:
1791 004326 104001          ERROR  1                   ;RO WAS MODIFIED, LDD
1792 004330 000137 005456    JMP    @#FDONE
1793
1794
1795 004334 012701 003656    FERR2: MOV    #F4,R1          ;THE PC WAS INCORRECTLY AFFECTED
1796                                ;DURING THE INSTRUCTION.
1797 004340 010137 001242    FER2:  MOV    R1,@#STMP4
1798 004344 162701 000004    SUB    #4,R1
1799 004350 006303    ASL    R3
1800 004352 060301    ADD    R3,R1
1801 004354 010137 001240    MOV    R1,@#STMP3
1802 004360 104001          1$:   ERROR  1
1803 004362 000137 005456    JMP    @#FDONE
1804
1805 004366 012701 004016    FERR4: MOV    #F11,R1
1806 004372 000762    BR     FER2
1807
1808 004374 012737 005424 001242    FERR3: MOV    #FDATO4,@#STMP4      ;FAILURE IN THE FDST FLOWS.
1809 004402 010037 001240    MOV    R0,@#STMP3
1810 004406 012737 000527 001244    MOV    #527,@#STMP5
1811 004414 012737 000641 001250    MOV    #641,@#STMP7
1812
1813 004422 022700 005414          CMP    #FDAT00,R0          ;DST MODE 4?
1814 004426 001004          BNE    1$
1815 004430 012737 000644 001246    MOV    #644,@#STMP6
1816 004436 000412          BR     4$
1817
1818 004440 022700 005434          1$:   CMP    #FDATO4+10,R0      ;DST MODE 2?
1819 004444 001004          BNE    2$
1820 004446 012737 000642 001246    MOV    #642,@#STMP6
1821 004454 000403          BR     4$
1822
1823 004456          2$:
1824 004456 104001          3$:   ERROR  1                   ;RO WAS MODIFIED, STD
1825 004460 000137 005456    JMP    @#FDONE
1826
1827 004464          4$:
1828 004464 104001          5$:   ERROR  1                   ;RO WAS MODIFIED, STD
1829
1830 004466 000137 005456    JMP    @#FDONE
    
```

1831								
1832	004472				FERR5:			;FAILURE OF STD.
1833	004472	010037	001240			MOV	RO,@#STMP3	
1834	004476	012737	005414	001242		MOV	#FDAT00,@#STMP4	
1835	004504	012737	005432	001244		MOV	#FDAT07,@#STMP5	
1836	004512	012737	005436	001246		MOV	#FXDAT0,@#STMP6	
1837	004520	104003			1\$:	ERROR	3	;OUTPUT BAD, STD
1838	004522	000137	005456			JMP	@#FDONE	
1839								
1840	004526	012701	005426		FERR6:	MOV	#FDAT05,R1	;DID (BUT GR7) FAIL IN THE FDST
1841	004532	012702	177777			MOV	#-1,R2	;FLOWS?
1842	004536	012703	000003			MOV	#3,R3	
1843	004542	020221			1\$:	CMP	R2,(R1)+	
1844	004544	001017				BNE	5\$	
1845	004546	077303				SOB	R3,1\$	
1846								
1847								;REPORT FAILURE OF (BUT GR7) IN
1848	004550	010037	001240			MOV	RO,@#STMP3	;THE FDST FLOWS.
1849	004554	012737	000412	001244		MOV	#412,@#STMP5	
1850	004562	012737	000147	001246		MOV	#147,@#STMP6	
1851	004570	012737	000145	001250		MOV	#145,@#STMP7	
1852	004576	104001			2\$:	ERROR	1	
1853	004600	000137	005456			JMP	@#FDONE	
1854								
1855	004604	012701	005426		5\$:	MOV	#FDAT05,R1	;DID (BUT GR7) FAIL IN THE SRC FLOWS?
1856	004610	012703	000003			MOV	#3,R3	
1857	004614	005721			6\$:	TST	(R1)+	
1858	004616	001402				BEG	7\$	
1859	004620	000137	005016			JMP	@#FERR10	
1860	004624	077305			7\$:	SOB	R3,6\$	
1861								
1862								;REPORT FAILURE OF (BUT GR7) IN
1863	004626	010037	001240			MOV	RO,@#STMP3	;THE FSRC FLOWS.
1864	004632	012737	000207	001244		MOV	#207,@#STMP5	
1865	004640	012737	000176	001246		MOV	#176,@#STMP6	
1866	004646	012737	000174	001250		MOV	#174,@#STMP7	
1867								
1868	004654	104001			10\$:	ERROR	1	
1869	004656	000137	005456			JMP	@#FDONE	
1870								
1871	004662	012701	005430		FERR7:	MOV	#FDAT06,R1	;DID (BUT FD) FAIL IN THE FDST FLOWS?
1872	004666	012702	177777			MOV	#-1,R2	
1873	004672	012703	000002			MOV	#2,R3	
1874	004676	020221			1\$:	CMP	R2,(R1)+	
1875	004700	001017				BNE	5\$	
1876	004702	077303				SOB	R3,1\$	
1877								
1878								;REPORT FAILURE OF (BUT FD) IN THE
1879	004704	010037	001240			MOV	RO,@#STMP3	;FDST FLOWS.
1880	004710	012737	000707	001244		MOV	#707,@#STMP5	
1881	004716	012737	000244	001246		MOV	#244,@#STMP6	
1882	004724	012737	000245	001250		MOV	#245,@#STMP7	
1883	004732	104001			2\$:	ERROR	1	
1884	004734	000137	005456			JMP	@#FDONE	
1885								
1886	004740	012701	005430		5\$:	MOV	#FDAT06,R1	;DID (BUT FD) FAIL IN THE FSRC FLOWS?

1887	004744	012703	000002		MOV	#2,R3		
1888	004750	005721		6\$:	TST	(R1)+		
1889	004752	001402			BEQ	7\$		
1890	004754	000137	005016		JMP	@#FERR10		
1891	004760	077305		7\$:	SOB	R3,6\$		
1892								
1893								
1894	004762	010037	001240		MOV	R0,@#STMP3	;REPORT FAILURE OF (BUT FD) IN THE	
1895	004766	012737	000441	001244	MOV	#441,@#STMP5	;FSRC FLOWS.	
1896	004774	012737	000076	001246	MOV	#76,@#STMP6		
1897	005002	012737	000077	001250	MOV	#77,@#STMP7		
1898	005010	104003			10\$:	ERROR	3 ;BAD DATA	
1899	005012	000137	005456		JMP	@#FDONE		
1900								
1901	005016				FERR10:		;REPORT DATA ERROR.	
1902	005016	010037	001240		MOV	R0,@#STMP3		
1903	005022	012737	005424	001242	MOV	#FDAT04,@#STMP4		
1904	005030	012737	005432	001244	MOV	#FDAT07,@#STMP5		
1905	005036	012737	005446	001246	MOV	#FXDAT4,@#STMP6		
1906	005044	104001			1\$:	ERROR	1	
1907	005046	000137	005456		JMP	@#FDONE		
1908								
1909	005052				FERR11:		;REPORT BAD FPS.	
1910	005052	010137	001240		MOV	R1,@#STMP3		
1911	005056	012737	000200	001242	MOV	#200,@#STMP4		
1912	005064	104001			1\$:	ERROR	1	
1913	005066	000137	005456		JMP	@#FDONE		
1914								
1915	005072	012737	062745	001264	FERR20:	MOV	#NULL,@#STMP15	;THE EXECUTION OF THE LDD
1916	005100	005037	001252		CLR	@#STMP10	;CAUSED A TRAP TO 4, BECAUSE	
1917	005104	011637	001236		MOV	(SP),@#STMP2	;A FSRC FLOW FAILURE RESULTED	
1918	005110	012737	005402	001240	MOV	#FDAT14,@#STMP3	;IN AN ODD ADDRESS.	
1919	005116	012737	000321	001250	MOV	#321,@#STMP7		
1920	005124	012737	000762	001244	MOV	#762,@#STMP5		
1921								
1922	005132	021627	003660		CMP	(SP),#F4+2	;SEE IF FSRC MODE 6 OR 7 WAS	
1923	005136	001424			BEQ	FERR21	;EXECUTED.	
1924								
1925	005140	020027	005400		CMP	R0,#FDAT13	;FSRC MODE 5?	
1926	005144	001006			BNF	2\$		
1927								
1928								
1929	005146	012737	000325	001246	MOV	#325,@#STMP6	;REPORT FSRC FLOW FAILURE TO	
1930	005154	022626			CMP	(SP)+,(SP)+	;MODE 5.	
1931	005156	104001			1\$:	ERROR	1	
1932	005160	000536			BR	FDONE		
1933								
1934	005162	020027	005404		2\$:	CMP	R0,#FDAT15	;FSRC MODE 3?
1935	005166	001402			BEQ	3\$		
1936	005170	000137	062566		JMP	@#CPSPUR		
1937								
1938	005174				3\$:		;REPORT FSRC FLOW FAILURE TO	
1939	005174	012737	000323	001246	MOV	#323,@#STMP6	;MODE 3.	
1940	005202	022626			CMP	(SP)+,(SP)+		
1941	005204	104001			4\$:	ERROR	1	
1942	005206	000523			BR	FDONE		

```

1943
1944 005210 022626          FERR21: CMP      (SP)+,(SP)+          ;REPORT FSRC FLOW FAILURE TO
1945                                     ;MODE 6 OR MODE 7.
1946 005212 012737 000326 001246      MOV      #326,@#STMP6
1947 005220 012737 000327 001252      MOV      #327,@#STMP10
1948 005226 104001          1$:      ERROR      1
1949 005230 000512          BR        FDONE
1950
1951 005232 012737 062745 001264      FERR25: MOV      #NULL,@#STMP15          ;THE EXECUTION OF THE STD INSTRUCTION
1952 005240 005037 001252          CLR      @#STMP10          ;TRAPPED TO 4, BECAUSE A FAILURE
1953 005244 012737 005424 001240      MOV      #FDAT04,@#STMP3          ;IN THE FDST FLOWS RESULTED
1954 005252 011637 001236          MOV      (SP),@#STMP2          ;IN AN ODD ADDRESS.
1955 005256 012737 000527 001244      MOV      #527,@#STMP5
1956 005264 012737 000641 001250      MOV      #641,@#STMP7
1957
1958 005272 021627 004016          CMP      (SP),#F10+2          ;FLOW FAILURE TO FDST MODE 6 OR 7?
1959 005276 001424          BEQ      FERR26
1960
1961 005300 020027 005422          CMP      R0,#FDAT03          ;DID FDST FLOW FAIL TO MODE 5?
1962 005304 001006          BNE      2$
1963
1964                                     ;REPORT FLOW FAILURE TO FDST
1965 005306 012737 000645 001246      MOV      #645,@#STMP6          ;MODE 5.
1966 005314 022626          CMP      (SP)+,(SP)+
1967 005316 104001          1$:      ERROR      1
1968 005320 000456          BR        FDONE
1969
1970 005322 020027 005426          2$:      CMP      R0,#FDAT05          ;DID FDST FLOW FAIL TO MODE 3?
1971 005326 001402          BEQ      3$
1972 005330 000137 062566          JMP      @#CPSPUR
1973
1974                                     ;REPORT FDST FLOW FAILED TO MODE 3.
1975 005334 012737 000643 001246      3$:      MOV      #643,@#STMP6
1976 005342 022626          CMP      (SP)+,(SP)+
1977 005344 104001          4$:      ERROR      1
1978 005346 000443          BR        FDONE
1979
1980 005350          FERR26:          ;REPORT FDST FLOW FAILURE TO MODE
1981 005350 012737 000646 001246      MOV      #646,@#STMP6
1982 005356 012737 000647 001252      MOV      #647,@#STMP10
1983 005364 022626          CMP      (SP)+,(SP)+
1984 005366 104001          1$:      ERROR      1
1985 005370 000432          BR        FDONE
1986
1987 005372 177777          FDATA10: -1
1988 005374 177777          FDATA11: -1
1989 005376 177777          FDATA12: -1
1990 005400 177777          FDATA13: -1
1991 005402 177777          FDATA14: -1
1992 005404 177777          FDATA15: -1
1993 005406 177777          FDATA16: -1
1994 005410 177777          FDATA17: -1
1995 005412 177777          -1
1996 005414 177777          FDATA00: -1
1997 005416 177777          FDATA01: -1
1998 005420 177777          FDATA02: -1
    
```

```

1999 005422 177777 FDAT03: -1
2000 005424 177777 FDAT04: -1
2001 005426 177777 FDAT05: -1
2002 005430 177777 FDAT06: -1
2003 005432 177777 FDAT07: -1
2004 005434 177777 -1
2005 005436 177777 FXDAT0: -1
2006 005440 177777 FXDAT1: -1
2007 005442 177777 FXDAT2: -1
2008 005444 177777 FXDAT3: -1
2009 005446 052525 FXDAT4: 052525
2010 005450 031463 FXDAT5: 031463
2011 005452 007417 FXDAT6: 007417
2012 005454 000477 FXDAT7: 000477
2013
2014
2015 005456
2016 005456 104413 F DONE:
2017 R SETUP ;GO INITIALIZE THE FPS AND STACK; AND
2018 ;SEE IF THE USER HAS EXPRESSED
2019 ;THE DESIRE TO CHANGE THE SOFTWARE
2020 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
2021 ;THE USER TYPED CONTROL G?).
2022
2023 ;*****
2024 ;*TEST 7 FSRC MODE 0 TEST
2025 ;*
2026 ;*THIS IS A TEST OF FSRC MODE ZERO USING THE LDD AND LDF INSTRUCTIONS.
2027 ;*
2028 ;*****
2029 005460 000004 TST7: SCOPE
2030 005462 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
2031
2032 005464 I1:
2033 005464 170011 SETD ;SET FD.
2034 005466 012700 006274 MOV #IDAT10,R0
2035 005472 012701 006244 MOV #IPAT10,R1
2036 005476 012702 000004 MOV #4,R2
2037 005502 012120 MOV (R1)+,(R0)+ ;SET UP THE INPUT DATA BUFFER.
2038 005504 077202 SOB R2,I2
2039
2040 005506 012700 006274 MOV #IDAT10,R0 ;LOAD AC1
2041 005512 172510 LDD (R0),AC1
2042
2043 005514 012700 006254 MOV #IPAT20,R0 ;LOAD AC0
2044 005520 172410 LDD (R0),AC0
2045
2046 005522 012701 000001 MOV #1,R1 ;IN CASE THE FSRC FLOWS FAIL
2047 005526 012737 006044 000004 MOV #IERR0,@#ERRVECT ;AN ODD ADDRESS TRAP TO 4 MAY OCCUR.
2048 005534 012737 005542 001236 MOV #13,@#STMP2
2049 005542 172401 I3: LDD AC1,AC0 ;TEST INSTRUCTION.
2050 005544 000240 I4: NOP
2051 005546 000240 I5: NOP
2052
2053 005550 012700 006264 MOV #IDAT00,R0
2054 005554 174010 STD AC0,(R0) ;GET AC0, THE RESULTS.

```



```

2055
2056 005556 012700 006264      MOV      #IDAT00,R0      ;SEE IF DATA IS CORRECT.
2057 005562 012701 006274      MOV      #IDAT10,R1
2058 005566 012702 000004      MOV      #4,R2
2059 005572 022021      16:     CMP      (R0)+,(R1)+
2060 005574 001424      BEQ
2061
2062 005576 012700 006270      MOV      #IDAT02,R0      ;SEE IF (BUT FD) FAILED.
2063 005602 012702 000002      MOV      #2,R2
2064 005606 005720      17:     TST      (R0)+
2065 005610 001413      BEQ      I10
2066
2067 005612 012700 006270      MOV      #IDAT02,R0
2068 005616 012702 000002      MOV      #2,R2
2069 005622 022720 177777      1$:     CMP      #-1,(R0)+
2070 005626 001402      BEQ      2$
2071 005630 000137 006126      JMP      @#IERR1
2072 005634 077206      2$:     SOB      R2,1$
2073 005636 000401      BR       I106
2074 005640 077216      I10:    SOB      R2,17
2075 005642 000137 006146      I106:   JMP      @#IERR2
2076
2077 005646 077227      I105:   SOB      R2,16
2078
2079      ;NOW TEST THE LOAD INSTRUCTION WITH FSRC MODE ZERO AND FD CLEAR.
2080
2081 005650      I11:
2082 005650 104414      LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.
2083
2084 005652 012700 006244      I12:    MOV      #IPAT10,R0
2085 005656 012701 006274      MOV      #IDAT10,R1
2086 005662 012702 000004      MOV      #4,R2
2087 005666 012021      I13:    MOV      (R0)+,(R1)+
2088 005670 077202      SOB      R2,I13
2089
2090 005672 012700 006274      MOV      #IDAT10,R0      ;SET UP AC1
2091 005676 172510      LDD      (R0),AC1
2092
2093 005700 012700 006254      MOV      #IPAT20,R0      ;SET UP AC0
2094 005704 172410      LDD      (R0),AC0
2095
2096 005706 012701 000001      MOV      #1,R1
2097 005712 012737 005722 001236      MOV      #I14,@#STMP2
2098 005720 170001      SETF
2099      ;CLEAR FD.
2100 005722 172401      I14:    LDF      AC1,AC0      ;TEST INSTRUCTION.
2101 005724 000240      I15:    NOP
2102 005726 000240      I16:    NOP
2103
2104 005730 170200      STFPS    R0      ;SEE IF FPS IS STILL CLEAR.
2105 005732 022700 000004      CMP      #4,R0
2106 005736 001402      BEQ      I17
2107 005740 000137 006220      JMP      @#IERR3
2108
2109 005744      I17:
2110 005744 170011      SETD
    
```

```

2111
2112 005746 012700 006264      MOV      #IDAT00,R0
2113 005752 174010              STD      ACO,(R0)          ;GET ACO
2114
2115 005754 012737 177777 006300      MOV      #-1,@#IDATI2
2116 005762 012737 177777 006302      MOV      #-1,@#IDATI3
2117 005770 012700 006264      MOV      #IDAT00,R0
2118 005774 012701 006274      MOV      #IDATIO,R1
2119 006000 012702 000004      MOV      #4,R2
2120 006004 022021              120:    CMP      (R0)+,(R1)+      ;SEE IF ACO WAS CORRECT.
2121 006006 001414              BEQ      I23
2122
2123 006010 023737 006270 006250      CMP      @#IDAT02,@#IPAT12 ;DID (BUT FD) FAIL?
2124 006016 001402              BEQ      I22
2125 006020 000137 006126              121:    JMP      @#IERR1
2126 006024 023737 006272 006252 122:    CMP      @#IDAT03,@#IPAT13
2127 006032 001372              BNE     I21
2128 006034 000137 006174              JMP      @#IERR4
2129
2130 006040 077217              123:    SOB     R2,I20
2131
2132 006042 000520              BR       IDONE            ;NO ERRORS.
2133
2134              ;IF AN ODD ADDRESS TRAP OCCURS COME HERE TO ANALYZE THE FSRC FAILURE.
2135 006044 022716 005544      IERR0:  CMP      #14,(SP)        ;MAKE SURE THE TRAP OCCURRED
2136 006050 001413              BEQ      1$              ;ON THE INSTRUCTION BEING TESTED.
2137 006052 022716 005546      CMP      #15,(SP)
2138 006056 001410              BEQ      1$
2139 006060 022716 005724      CMP      #115,(SP)
2140 006064 001405              BEQ      1$
2141 006066 022716 005726      CMP      #116,(SP)
2142 006072 001402              BEQ      1$
2143 006074 000137 062566      JMP      @#CPSPUR
2144
2145 006100 011637 001236              1$:    MOV      (SP),@#STMP2      ;REPORT FAILURE.
2146 006104 012737 000627 001240      MOV      #627,@#STMP3
2147 006112 012737 000320 001242      MOV      #320,@#STMP4
2148 006120 022626              CMP      (SP)+,(SP)+
2149 006122 104001              2$:    ERROR   1
2150 006124 000467              BR       IDONE
2151
2152              ;REPORT DATA ERROR.
2153 006126              IERR1:
2154 006126 012737 006274 001242      MOV      #IDATIO,@#STMP4
2155 006134 012737 006264 001244      MOV      #IDAT00,@#STMP5
2156 006142 104004              1$:    ERROR   4
2157 006144 000457              BR       IDONE
2158
2159              ;REPORT FAILURE OF (BUT FD)
2160 006146 012737 000153 001244      IERR2:  MOV      #153,@#STMP5
2161 006154 012737 000434 001246      MOV      #434,@#STMP6
2162 006162 012737 000435 001250      MOV      #435,@#STMP7
2163 006170              IERR25:
2164 006170 104001              1$:    ERROR   1
2165 006172 000444              BR       IDONE
2166 006174 012737 000153 001244      IERR4:  MOV      #153,@#STMP5
    
```

```

2167 006202 012737 000435 001246      MOV    #435,@#STMP6
2168 006210 012737 000434 001250      MOV    #434,@#STMP7
2169 006216 000764                BR     IERR25
2170
2171                                ;REPORT INCORRECT FPS AFTER LOAD INSTRUCTION.
2172 006220      IERR3:
2173 006220 012737 005722 001236      MOV    #114,@#STMP2
2174 006226 010037 001240      MOV    R0,@#STMP3
2175 006232 012737 000004 001242      MOV    #4,@#STMP4
2176 006240 104001      1$:    ERROR 1
2177 006242 000420      BR     IDONE
2178
2179
2180 006244 000000      IPAT10: 0
2181 006246 170360      IPAT11: 170360
2182 006250 016161      IPAT12: 016161
2183 006252 052525      IPAT13: 052525
2184
2185 006254 177777      IPAT20: -1
2186 006256 177777      IPAT21: -1
2187 006260 177777      IPAT22: -1
2188 006262 177777      IPAT23: -1
2189
2190 006264 000000      IDAT00: 0
2191 006266 000000      IDAT01: 0
2192 006270 000000      IDAT02: 0
2193 006272 000000      IDAT03: 0
2194
2195 006274 000000      IDAT10: 0
2196 006276 000000      IDAT11: 0
2197 006300 000000      IDAT12: 0
2198 006302 000000      IDAT13: 0
2199
2200 006304      IDONE:
2201 006304 104413      RSETUP                                ;GO INITIALIZE THE FPS AND STACK; AND
2202                                                    ;SEE IF THE USER HAS EXPRESSED
2203                                                    ;THE DESIRE TO CHANGE THE SOFTWARE
2204                                                    ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
2205                                                    ;THE USER TYPED CONTROL G?).
2206
2207
2208                                ;*****
2209                                ;*TEST 10      FDST MODE 0 TEST
2210                                ;*
2211                                ;*THIS IS A TEST OF THE STORE INSTRUCTIONS, STD AND STF, WITH FDST MODE 0.
2212                                ;*
2213                                ;*****
2214 006306 000004      TST10: SCOPE
2215 006310      T1:
2216 006310 104414      LPERR                                ;SET UP THE LOOP ON ERROR ADDRESS.
2217 006312 170011      SETD                                ;SET FD
2218 006314 012700 007036      MOV    #TPAT10,R0
2219 006320 012701 007066      MOV    #TDAT10,R1
2220 006324 012702 000004      MOV    #4,R2
2221 006330 012021      T2:    MOV    (R0)+,(R1)+                ;SET UP THE INPUT DATA BUFFER.
2222 006332 077202      SOB    R2,T2
    
```

```

2223
2224 006334 012700 007066      MOV    #TDAT10,R0      ;LOAD AC0
2225 006340 172410              LDD    (R0),AC0
2226
2227 006342 012700 007046      MOV    #TPAT20,R0      ;LOAD AC1
2228 006346 172510              LDD    (R0),AC1
2229
2230 006350 012701 000001      MOV    #1,R1           ;IF THE (BUT FDST) FORK FAILS
2231 006354 012737 006644 000004  MOV    #TERRO,@#ERRVECT ;AN ODD ADDRESS TRAP COULD RESULT.
2232 006362 012737 006370 001236  MOV    #T3,@#STMP2
2233 006370 174001              MOV    #T3,@#STMP2
2234 006372 000240              STD    AC0,AC1
2235 006374 000240              T3:   NOP
2236
2237 006376 012700 007056      T4:   NOP
2238 006402 174110              T5:   NOP
2239
2240 006404 012703 007056      MOV    #TDAT00,R0      ;GET THE DATA.
2241 006410 012704 007066      STD    AC1,(R0)
2242 006414 012705 000004      ;SEE IF THE DATA IS CORRECT.
2243 006420 022324              MOV    #TDAT00,R3
2244 006422 001413              MOV    #TDAT10,R4
2245
2246 006424 012703 007062      MOV    #4,R5
2247 006430 012705 000002      T6:   CMP    (R3)+,(R4)+
2248 006434 005723              BEQ    T105
2249 006436 001402              ;DID (BUT FD) FAIL?
2250 006440 000137 006726      MOV    #TDAT02,R3
2251 006444 077505              MOV    #2,R5
2252 006446 000137 006746      T7:   TST    (R3)+
2253
2254 006452 077516              BEQ    T10
2255
2256
2257
2258 006454              T10:  JMP    @#TERR1
2259 006454 104414              T11:  SOB    R5,T7
2260
2261 006456 012700 007036      T12:  JMP    @#TERR2
2262 006462 012701 007066      ;NOW TEST THE STF AC0,AC1 INSTRUCTION.
2263 006466 012702 000004      T13:  SOB    R5,T6
2264 006472 012021              ;SET UP THE LOOP ON ERROR ADDRESS.
2265 006474 077202              T14:  LPERR
2266
2267 006476 012700 007066      T15:  MOV    #TPAT10,R0
2268 006502 172410              T16:  MOV    #TDAT10,R1
2269
2270 006504 012700 007046      ;SET UP THE INPUT DATA BUFFER.
2271 006510 172510              MOV    #4,R2
2272
2273 006512 012701 000001      T13:  MOV    (R0)+,(R1)+
2274 006516 012737 006526 001236  SOB    R2,T13
2275 006524 170001              ;SET UP AC0
2276 006526 174001              MOV    #TDAT10,R0
2277 006530 000240              LDD    (R0),AC0
2278 006532 000240              ;SET UP AC1
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
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
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
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
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
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
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
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
2652
2653
2654
2655
2656
2657
2658
2659
2660
2661
2662
2663
2664
2665
2666
2667
2668
2669
2670
2671
2672
2673
2674
2675
2676
2677
2678
2679
2680
2681
2682
2683
2684
2685
2686
2687
2688
2689
2690
2691
2692
2693
2694
2695
2696
2697
2698
2699
2700
2701
2702
2703
2704
2705
2706
2707
2708
2709
2710
2711
2712
2713
2714
2715
2716
2717
2718
2719
2720
2721
2722
2723
2724
2725
2726
2727
2728
2729
2730
2731
2732
2733
2734
2735
2736
2737
2738
2739
2740
2741
2742
2743
2744
2745
2746
2747
2748
2749
2750
2751
2752
2753
2754
2755
2756
2757
2758
2759
2760
2761
2762
2763
2764
2765
2766
2767
2768
2769
2770
2771
2772
2773
2774
2775
2776
2777
2778
2779
2780
2781
2782
2783
2784
2785
2786
2787
2788
2789
2790
2791
2792
2793
2794
2795
2796
2797
2798
2799
2800
2801
2802
2803
2804
2805
2806
2807
2808
2809
2810
2811
2812
2813
2814
2815
2816
2817
2818
2819
2820
2821
2822
2823
2824
2825
2826
2827
2828
2829
2830
2831
2832
2833
2834
2835
2836
2837
2838
2839
2840
2841
2842
2843
2844
2845
2846
2847
2848
2849
2850
2851
2852
2853
2854
2855
2856
2857
2858
2859
2860
2861
2862
2863
2864
2865
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
2922
2923
2924
2925
2926
2927
2928
2929
2930
2931
2932
2933
2934
2935
2936
2937
2938
2939
2940
2941
2942
2943
2944
2945
2946
2947
2948
2949
2950
2951
2952
2953
2954
2955
2956
2957
2958
2959
2960
2961
2962
2963
2964
2965
2966
2967
2968
2969
2970
2971
2972
2973
2974
2975
2976
2977
2978
2979
2980
2981
2982
2983
2984
2985
2986
2987
2988
2989
2990
2991
2992
2993
2994
2995
2996
2997
2998
2999
3000

```

```

2279
2280 006534 005000          CLR      R0
2281 006536 170200          STFPS   R0          ;SEE IF FPS IS CLEAR.
2282 006540 022700 000010  CMP      #10,R0
2283 006544 001401          BEQ     T17
2284 006546 000521          BR      TERR3
2285
2286 006550          T17:
2287 006550 170011          SETD           ;SET FD.
2288
2289 006552 012700 007056  MOV      #TDAT00,R0
2290 006556 174110          STD      AC1,(R0)  ;PICK UP AC1.
2291
2292 006560 012737 177777 007072  MOV      #-1,@#TDAT12
2293 006566 012737 177777 007074  MOV      #-1,@#TDAT13
2294 006574 012703 007056          MOV      #TDAT00,R3
2295 006600 012704 007066          MOV      #TDAT10,R4
2296 006604 012705 000004          MOV      #4,R5
2297 006610 022324          T20:  CMP      (R3)+,(R4)+  ;WAS THE DATA TRANSFERRED CORRECTLY?
2298 006612 001412          BEQ     T23
2299
2300 006614 023737 007062 007042  CMP      @#TDAT02,@#TPAT12  ;DID (BUT FD) FAIL.
2301 006622 001401          BEQ     T22
2302 006624 000440          T21:  BR      TERR1
2303 006626 023737 007064 007044  T22:  CMP      @#TDAT03,@#TPAT13
2304 006634 001373          BNE     T21
2305 006636 000456          BR      TERR4
2306
2307 006640 077515          T23:  SOB     R5,T20
2308 006642 000515          BR      TDONE
2309
2310
2311          ;TRAP HERE THROUGH VECTOR 4 IF AN ODD ADDRESS OCCURS.
2312 006644 022716 006372  TERR0:  CMP      #T4,(SP)          ;MAKE SURE THE TRAP WAS ON
2313 006650 001413          BEQ     1$              ;AN INSTRUCTION BEING TESTED.
2314 006652 022716 006374          CMP      #T5,(SP)
2315 006656 001410          BEQ     1$
2316 006660 022716 006530          CMP      #T15,(SP)
2317 006664 001405          BEQ     1$
2318 006666 022716 006532          CMP      #T16,(SP)
2319 006672 001402          BEQ     1$
2320 006674 000137 062566          JMP     @#CPSPUR
2321
2322 006700 011637 001236          1$:  MOV      (SP),@#STMP2
2323 006704 022626          CMP      (SP)+,(SP)+
2324 006706 012737 000527 001240  MOV      #527,@#STMP3
2325 006714 012737 000640 001242  MOV      #640,@#STMP4
2326 006722 104001          2$:  ERROR   1
2327 006724 000464          BR      TDONE
2328
2329          ;REPORT DATA FAILURE.
2330 006726          TERR1:
2331 006726 012737 007066 001242  MOV      #TDAT10,@#STMP4
2332 006734 012737 007056 001244  MOV      #TDAT00,@#STMP5
2333 006742 104001          1$:  ERROR   1
2334 006744 000454          BR      TDONE
    
```



```
2335
2336
2337 006746 012737 000160 001246 :REPORT FAILURE OF (BUT FD).
2338 006754 012737 000161 001250 TERR2: MOV #160,@#STMP6
2339 006762 012737 000640 001244 TERR25: MOV #161,@#STMP7
2340 006770 104012 1$: ERROR #640,@#STMP5
2341 006772 000441 BR TDONE
2342 006774 012737 000161 001246 TERR4: MOV #161,@#STMP6
2343 007002 012737 000160 001250 MOV #160,@#STMP7
2344 007010 000764 BR TERR25
2345
2346 :REPORT INCORRECT FPS AFTER STORE INSTRUCTION.
2347 007012 TERR3:
2348 007012 012737 006530 001236 MOV #T15,@#STMP2
2349 007020 010037 001240 MOV R0,@#STMP3
2350 007024 012737 000010 001242 MOV #10,@#STMP4
2351 007032 104001 1$: ERROR 1
2352 007034 000420 BR TDONE
2353
2354 007036 000000 TPAT10: 0
2355 007040 170360 TPAT11: 170360
2356 007042 016161 TPAT12: 016161
2357 007044 052525 TPAT13: 052525
2358
2359 007046 177777 TPAT20: -1
2360 007050 177777 TPAT21: -1
2361 007052 177777 TPAT22: -1
2362 007054 177777 TPAT23: -1
2363
2364 007056 000000 TDATA0: 0
2365 007060 000000 TDATA1: 0
2366 007062 000000 TDATA2: 0
2367 007064 000000 TDATA3: 0
2368
2369 007066 000000 TDATA10: 0
2370 007070 000000 TDATA11: 0
2371 007072 000000 TDATA12: 0
2372 007074 000000 TDATA13: 0
2373
2374 007076 TDONE:
2375 007076 104413 RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
2376 ;SEE IF THE USER HAS EXPRESSED
2377 ;THE DESIRE TO CHANGE THE SOFTWARE
2378 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
2379 ;THE USER TYPED CONTROL G?).
2380
2381
2382
2383 :*****
2384 :*TEST 11 ACCUMULATORS DATA PATTERNS TEST
2385 :*
2386 :*THIS IS A TEST OF THE FLOATING POINT PROCESSOR ACCUMULATORS.
2387 :*EACH ACCUMULATOR IS TESTED IN TWO WAYS:
2388 :* 1 TEST PATTERN GENERATED BY FLOATING A ONE ACROSS
2389 :* A FIELD OF ZEROES.
2390 :* 2 TEST PATTERN GENERATED BY FLOATING A ZERO ACROSS
```

```

2391
2392
2393
2394 007100 000004
2395 007102 170011
2396
2397 007104 012737 007134 001236
2398 007112 012700 011002
2399 007116 012701 011042
2400 007122 104414
2401 007124 004737 010650
2402 007130 012703 000102
2403 007134
2404 007134 172410
2405 007136 174000
2406 007140 172400
2407 007142 174011
2408 007144 004737 010746
2409
2410 007150 005737 010776
2411 007154 001004
2412 007156 005137 010776
2413 007162 000261
2414 007164 000401
2415 007166 000241
2416 007170 006160 000006
2417 007174 006160 000004
2418 007200 006160 000002
2419 007204 006110
2420 007206 004737 010726
2421
2422 007212 077330
2423
2424
2425 007214 012737 007244 001236
2426 007222 012700 011012
2427 007226 012701 011042
2428 007232 104414
2429 007234 004737 010650
2430 007240 012703 000102
2431 007244
2432 007244 172410
2433 007246 174000
2434 007250 172400
2435 007252 174011
2436 007254 004737 010746
2437
2438 007260 005737 010776
2439 007264 001004
2440 007266 005137 010776
2441 007272 000241
2442 007274 000401
2443 007276 000261
2444 007300 006160 000006
2445 007304 006160 000004
2446 007310 006160 000002

```

```

;* A FIELD OF ONES.
;*EACH OF ACCUMULATORS ACO THROUGH AC5 IS TESTED.
;*****
TST11: SCOPE
        SETD                                ;SET FD.
;TEST ACCUMULATOR 0 WITH FLOATING ONE
        MOV #G1,@#STMP2
        MOV #GPAT00,R0
        MOV #GDAT00,R1
        LPERR                                ;SET UP THE LOOP ON ERROR ADDRESS.
        JSR PC,@#GSETUP                      ;LOAD TEST PATTERN.
        MOV #102,R3
G1:
        LDD (R0),AC0
        STD AC0,AC0
        LDD AC0,AC0                          ;STORE THE TEST PATTERN.
        STD AC0,(R1)
        JSR PC,@#GCMP                        ;COMPARE THE DATA READ WITH
                                                ;THAT WHICH WAS WRITTEN.
        TST @#GFLAG1
        BNE G2
        COM @#GFLAG1
        SEC
        BR G3
G2:
        CLC
G3:
        ROL 6(R0)                            ;GENERATE THE NEXT TEST PATTERN.
        ROL 4(R0)
        ROL 2(R0)
        ROL (R0)
        JSR PC,@#GRESET                      ;RESET DEFAULT PATTERN IN OUTPUT
                                                ;BUFFER.
        SOB R3,G1
;TEST ACCUMULATOR 0 WITH FLOATING ZERO
        MOV #G4,@#STMP2
        MOV #GPAT10,R0
        MOV #GDAT00,R1
        LPERR                                ;SET UP THE LOOP ON ERROR ADDRESS.
        JSR PC,@#GSETUP                      ;LOAD TEST PATTERN.
        MOV #102,R3
G4:
        LDD (R0),AC0
        STD AC0,AC0
        LDD AC0,AC0                          ;STORE THE TEST PATTERN.
        STD AC0,(R1)
        JSR PC,@#GCMP                        ;COMPARE THE DATA READ WITH
                                                ;THAT WHICH WAS WRITTEN.
        TST @#GFLAG1
        BNE G5
        COM @#GFLAG1
        CLC
        BR G6
G5:
        SEC
G6:
        ROL 6(R0)                            ;GENERATE THE NEXT TEST PATTERN.
        ROL 4(R0)
        ROL 2(R0)

```

```

2447 007314 006110          ROL    (R0)
2448 007316 004737 010726  JSR    PC,@#GRESET          ;RESET DEFAULT PATTERN IN OUTPUT
2449                                ;BUFFER.
2450 007322 077330          SOB    R3,G4
2451
2452                                ;TEST ACCUMULATOR 1 WITH FLOATING ONE
2453 007324 012737 007354 001236  MOV    #G7,@#STMP2
2454 007332 012700 011002          MOV    #GPAT00,R0
2455 007336 012701 011042          MOV    #GDAT00,R1
2456 007342 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
2457 007344 004737 010650  JSR    PC,@#GSETUP          ;LOAD TEST PATTERN.
2458 007350 012703 000102  MOV    #102,R3
2459 007354          G7:
2460 007354 172410          LDD    (R0),AC0
2461 007356 174001          STD    AC0,AC1
2462 007360 172401          LDD    AC1,AC0          ;STORE THE TEST PATTERN.
2463 007362 174011          STD    AC0,(R1)
2464 007364 004737 010746  JSR    PC,@#GCMP          ;COMPARE THE DATA READ WITH
2465                                ;THAT WHICH WAS WRITTEN.
2466 007370 005737 010776          TST    @#GFLAG1
2467 007374 001004          BNE    G10
2468 007376 005137 010776          COM    @#GFLAG1
2469 007402 000261          SEC
2470 007404 000401          BR     G11
2471 007406 000241          G10:  CLC
2472 007410 006160 000006  G11:  ROL    6(R0)          ;GENERATE THE NEXT TEST PATTERN.
2473 007414 006160 000004          ROL    4(R0)
2474 007420 006160 000002          ROL    2(R0)
2475 007424 006110          ROL    (R0)
2476 007426 004737 010726  JSR    PC,@#GRESET          ;RESET DEFAULT PATTERN IN OUTPUT
2477                                ;BUFFER.
2478 007432 077330          SOB    R3,G7
2479
2480                                ;TEST ACCUMULATOR 1 WITH FLOATING ZERO
2481 007434 012737 007464 001236  MOV    #G12,@#STMP2
2482 007442 012700 011012          MOV    #GPAT10,R0
2483 007446 012701 011042          MOV    #GDAT00,R1
2484 007452 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
2485 007454 004737 010650  JSR    PC,@#GSETUP          ;LOAD TEST PATTERN.
2486 007460 012703 000102  MOV    #102,R3
2487 007464          G12:
2488 007464 172410          LDD    (R0),AC0
2489 007466 174001          STD    AC0,AC1
2490 007470 172401          LDD    AC1,AC0          ;STORE THE TEST PATTERN.
2491 007472 174011          STD    AC0,(R1)
2492 007474 004737 010746  JSR    PC,@#GCMP          ;COMPARE THE DATA READ WITH
2493                                ;THAT WHICH WAS WRITTEN.
2494 007500 005737 010776          TST    @#GFLAG1
2495 007504 001004          BNE    G13
2496 007506 005137 010776          COM    @#GFLAG1
2497 007512 000241          CLC
2498 007514 000401          BR     G14
2499 007516 000261          G13:  SEC
2500 007520 006160 000006  G14:  ROL    6(R0)          ;GENERATE THE NEXT TEST PATTERN.
2501 007524 006160 000004          ROL    4(R0)
2502 007530 006160 000002          ROL    2(R0)
    
```

```

2503 007534 006110          ROL    (R0)
2504 007536 004737 010726  JSR    PC,@#GRESET          ;RESET DEFAULT PATTERN IN OUTPUT
2505                                     ;BUFFER.
2506 007542 077330          SOB    R3,G12
2507
2508                                     ;TEST ACCUMULATOR 2 WITH FLOATING ONE
2509 007544 012737 007574 001236  MOV    #G15,@#STMP2
2510 007552 012700 011002      MOV    #GPAT00,R0
2511 007556 012701 011042      MOV    #GDAT00,R1
2512 007562 104414          LPERR
2513 007564 004737 010650      JSR    PC,@#GSETUP          ;SET UP THE LOOP ON ERROR ADDRESS.
2514 007570 012703 000102      MOV    #102,R3              ;LOAD TEST PATTERN.
2515
2516 007574 172410          G15:  LDD    (R0),AC0
2517 007576 174002          STD    AC0,AC2
2518 007600 172402          LDD    AC2,AC0              ;STORE THE TEST PATTERN.
2519 007602 174011          STD    AC0,(R1)
2520 007604 004737 010746      JSR    PC,@#GCMP           ;COMPARE THE DATA READ WITH
2521                                     ;THAT WHICH WAS WRITTEN.
2522 007610 005737 010776      TST    @#GFLAG1
2523 007614 001004          BNE    G16
2524 007616 005137 010776      COM    @#GFLAG1
2525 007622 000261          SEC
2526 007624 000401          BR     G17
2527 007626 000241          G16:  CLC
2528 007630 006160 000006      G17:  ROL    6(R0)           ;GENERATE THE NEXT TEST PATTERN.
2529 007634 006160 000004      ROL    4(R0)
2530 007640 006160 000002      ROL    2(R0)
2531 007644 006110          ROL    (R0)
2532 007646 004737 010726      JSR    PC,@#GRESET          ;RESET DEFAULT PATTERN IN OUTPUT
2533                                     ;BUFFER.
2534 007652 077330          SOB    R3,G15
2535
2536                                     ;TEST ACCUMULATOR 2 WITH FLOATING ZERO
2537 007654 012737 007704 001236  MOV    #G20,@#STMP2
2538 007662 012700 011012      MOV    #GPAT10,R0
2539 007666 012701 011042      MOV    #GDAT00,R1
2540 007672 104414          LPERR
2541 007674 004737 010650      JSR    PC,@#GSETUP          ;SET UP THE LOOP ON ERROR ADDRESS.
2542 007700 012703 000102      MOV    #102,R3              ;LOAD TEST PATTERN.
2543
2544 007704 172410          G20:  LDD    (R0),AC0
2545 007706 174002          STD    AC0,AC2
2546 007710 172402          LDD    AC2,AC0              ;STORE THE TEST PATTERN.
2547 007712 174011          STD    AC0,(R1)
2548 007714 004737 010746      JSR    PC,@#GCMP           ;COMPARE THE DATA READ WITH
2549                                     ;THAT WHICH WAS WRITTEN.
2550 007720 005737 010776      TST    @#GFLAG1
2551 007724 001004          BNE    G21
2552 007726 005137 010776      COM    @#GFLAG1
2553 007732 000241          CLC
2554 007734 000401          BR     G22
2555 007736 000261          G21:  SEC
2556 007740 006160 000006      G22:  ROL    6(R0)           ;GENERATE THE NEXT TEST PATTERN.
2557 007744 006160 000004      ROL    4(R0)
2558 007750 006160 000002      ROL    2(R0)
    
```

```

2559 007754 006110          ROL    (R0)
2560 007756 004737 010726   JSR    PC,@#GRESET          ;RESET DEFAULT.PATTERN IN OUTPUT
2561                               ;BUFFER.
2562 007762 077330          SOB    R3,G20
2563
2564                               ;TEST ACCUMULATOR 3 WITH FLOATING ONE
2565 007764 012737 010014 001236   MOV    #G23,@#STMP2
2566 007772 012700 011002       MOV    #GPAT00,R0
2567 007776 012701 011042       MOV    #GDAT00,R1
2568 010002 104414          LPERR
2569 010004 004737 010650       JSR    PC,@#GSETUP          ;SET UP THE LOOP ON ERROR ADDRESS.
2570 010010 012703 000102       MOV    #102,R3              ;LOAD TEST PATTERN.
2571
2572 010014 172410          G23:  LDD    (R0),AC0
2573 010016 174003          STD    AC0,AC3
2574 010020 172403          LDD    AC3,AC0              ;STORE THE TEST PATTERN.
2575 010022 174011          STD    AC0,(R1)
2576 010024 004737 010746       JSR    PC,@#GCMP
2577                               ;COMPARE THE DATA READ WITH
2578 010030 005737 010776       TST    @#GFLAG1            ;THAT WHICH WAS WRITTEN.
2579 010034 001004          BNE    G24
2580 010036 005137 010776       COM    @#GFLAG1
2581 010042 000261          SEC
2582 010044 000401          BR     G25
2583 010046 000241          G24:  CLC
2584 010050 006160 000006       G25:  ROL    6(R0)            ;GENERATE THE NEXT TEST PATTERN.
2585 010054 006160 000004       ROL    4(R0)
2586 010060 006160 000002       ROL    2(R0)
2587 010064 006110          ROL    (R0)
2588 010066 004737 010726       JSR    PC,@#GRESET          ;RESET DEFAULT PATTERN IN OUTPUT
2589                               ;BUFFER.
2590 010072 077330          SOB    R3,G23
2591
2592                               ;TEST ACCUMULATOR 3 WITH FLOATING ZERO
2593 010074 012737 010124 001236   MOV    #G26,@#STMP2
2594 010102 012700 011012       MOV    #GPAT10,R0
2595 010106 012701 011042       MOV    #GDAT00,R1
2596 010112 104414          LPERR
2597 010114 004737 010650       JSR    PC,@#GSETUP          ;SET UP THE LOOP ON ERROR ADDRESS.
2598 010120 012703 000102       MOV    #102,R3              ;LOAD TEST PATTERN.
2599
2600 010124 172410          G26:  LDD    (R0),AC0
2601 010126 174003          STD    AC0,AC3
2602 010130 172403          LDD    AC3,AC0              ;STORE THE TEST PATTERN.
2603 010132 174011          STD    AC0,(R1)
2604 010134 004737 010746       JSR    PC,@#GCMP
2605                               ;COMPARE THE DATA READ WITH
2606 010140 005737 010776       TST    @#GFLAG1            ;THAT WHICH WAS WRITTEN.
2607 010144 001004          BNE    G27
2608 010146 005137 010776       COM    @#GFLAG1
2609 010152 000241          CLC
2610 010154 000401          BR     G30
2611 010156 000261          G27:  SEC
2612 010160 006160 000006       G30:  ROL    6(R0)            ;GENERATE THE NEXT TEST PATTERN.
2613 010164 006160 000004       ROL    4(R0)
2614 010170 006160 000002       ROL    2(R0)
    
```

```

2615 010174 006110          ROL      (R0)
2616 010176 004737 010726  JSR      PC,@#GRESET      ;RESET DEFAULT PATTERN IN OUTPUT
2617                                ;BUFFER.
2618 010202 077330          SOB      R3,G26
2619
2620                                ;TEST ACCUMULATOR 4 WITH FLOATING ONE
2621 010204 012737 010234 001236  MOV      #G31,@#STMP2
2622 010212 012700 011002      MOV      #GPAT00,R0
2623 010216 012701 011042      MOV      #GDAT00,R1
2624 010222 104414          LPERR
2625 010224 004737 010650      JSR      PC,@#GSETUP      ;SET UP THE LOOP ON ERROR ADDRESS.
2626 010230 012703 000102      MOV      #102,R3          ;LOAD TEST PATTERN.
2627 010234
2628 010234 172410          G31:    LDD      (R0),AC0
2629 010236 174004          STD      AC0,AC4
2630 010240 172404          LDD      AC4,AC0          ;STORE THE TEST PATTERN.
2631 010242 174011          STD      AC0,(R1)
2632 010244 004737 010746      JSR      PC,@#GCMP
2633                                ;COMPARE THE DATA READ WITH
2634 010250 005737 010776      TST      @#GFLAG1        ;THAT WHICH WAS WRITTEN.
2635 010254 001004          BNE      G32
2636 010256 005137 010776      COM      @#GFLAG1
2637 010262 000261          SEC
2638 010264 000401          BR       G33
2639 010266 000241          G32:    CLC
2640 010270 006160 000006      G33:    ROL      6(R0)          ;GENERATE THE NEXT TEST PATTERN.
2641 010274 006160 000004      ROL      4(R0)
2642 010300 006160 000002      ROL      2(R0)
2643 010304 006110          ROL      (R0)
2644 010306 004737 010726      JSR      PC,@#GRESET      ;RESET DEFAULT PATTERN IN OUTPUT
2645                                ;BUFFER.
2646 010312 077330          SOB      R3,G31
2647
2648                                ;TEST ACCUMULATOR 4 WITH FLOATING ZERO
2649 010314 012737 010344 001236  MOV      #G34,@#STMP2
2650 010322 012700 011012      MOV      #GPAT10,R0
2651 010326 012701 011042      MOV      #GDAT00,R1
2652 010332 104414          LPERR
2653 010334 004737 010650      JSR      PC,@#GSETUP      ;SET UP THE LOOP ON ERROR ADDRESS.
2654 010340 012703 000102      MOV      #102,R3          ;LOAD TEST PATTERN.
2655 010344
2656 010344 172410          G34:    LDD      (R0),AC0
2657 010346 174004          STD      AC0,AC4
2658 010350 172404          LDD      AC4,AC0          ;STORE THE TEST PATTERN.
2659 010352 174011          STD      AC0,(R1)
2660 010354 004737 010746      JSR      PC,@#GCMP
2661                                ;COMPARE THE DATA READ WITH
2662 010360 005737 010776      TST      @#GFLAG1        ;THAT WHICH WAS WRITTEN.
2663 010364 001004          BNE      G35
2664 010366 005137 010776      COM      @#GFLAG1
2665 010372 000241          CLC
2666 010374 000401          BR       G36
2667 010376 000261          G35:    SEC
2668 010400 006160 000006      G36:    ROL      6(R0)          ;GENERATE THE NEXT TEST PATTERN.
2669 010404 006160 000004      ROL      4(R0)
2670 010410 006160 000002      ROL      2(R0)
    
```



```

2671 010414 006110          ROL    (R0)
2672 010416 004737 010726  JSR    PC,@#GRESET          ;RESET DEFAULT PATTERN IN OUTPUT
2673                                     ;BUFFER.
2674 010422 077330          SOB    R3,G34
2675
2676                                     ;TEST ACCUMULATOR 5 WITH FLOATING ONE
2677 010424 012737 010454 001236  MOV    #G37,@#STMP2
2678 010432 012700 011002          MOV    #GPAT00,R0
2679 010436 012701 011042          MOV    #GDAT00,R1
2680 010442 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
2681 010444 004737 010650  JSR    PC,@#GSETUP          ;LOAD TEST PATTERN.
2682 010450 012703 000102  MOV    #102,R3
2683 010454
2684 010454 172410          LDD    (R0),AC0
2685 010456 174005          STD    AC0,AC5
2686 010460 172405          LDD    AC5,AC0          ;STORE THE TEST PATTERN.
2687 010462 174011          STD    AC0,(R1)
2688 010464 004737 010746  JSR    PC,@#GCMP          ;COMPARE THE DATA READ WITH
2689                                     ;THAT WHICH WAS WRITEN.
2690 010470 005737 010776          TST    @#GFLAG1
2691 010474 001004          BNE    G40
2692 010476 005137 010776          COM    @#GFLAG1
2693 010502 000261          SEC
2694 010504 000401          BR     G41
2695 010506 000241          G40:  CLC
2696 010510 006160 000006  G41:  ROL    6(R0)          ;GENERATE THE NEXT TEST PATTERN.
2697 010514 006160 000004          ROL    4(R0)
2698 010520 006160 000002          ROL    2(R0)
2699 010524 006110          ROL    (R0)
2700 010526 004737 010726  JSR    PC,@#GRESET          ;RESET DEFAULT PATTERN IN OUTPUT
2701                                     ;BUFFER.
2702 010532 077330          SOB    R3,G37
2703
2704                                     ;TEST ACCUMULATOR 5 WITH FLOATING ZERO
2705 010534 012737 010564 001236  MOV    #G42,@#STMP2
2706 010542 012700 011012          MOV    #GPAT10,R0
2707 010546 012701 011042          MOV    #GDAT00,R1
2708 010552 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
2709 010554 004737 010650  JSR    PC,@#GSETUP          ;LOAD TEST PATTERN.
2710 010560 012703 000102  MOV    #102,R3
2711 010564
2712 010564 172410          LDD    (R0),AC0
2713 010566 174005          STD    AC0,AC5
2714 010570 172405          LDD    AC5,AC0          ;STORE THE TEST PATTERN.
2715 010572 174011          STD    AC0,(R1)
2716 010574 004737 010746  JSR    PC,@#GCMP          ;COMPARE THE DATA READ WITH
2717                                     ;THAT WHICH WAS WRITTEN.
2718 010600 005737 010776          TST    @#GFLAG1
2719 010604 001004          BNE    G43
2720 010606 005137 010776          COM    @#GFLAG1
2721 010612 000241          CLC
2722 010614 000401          BR     G44
2723 010616 000261          G43:  SEC
2724 010620 006160 000006  G44:  ROL    6(R0)          ;GENERATE THE NEXT TEST PATTERN.
2725 010624 006160 000004          ROL    4(R0)
2726 010630 006160 000002          ROL    2(R0)
    
```

```

2727 010634 006110          ROL    (R0)
2728 010636 004737 010726  JSR    PC,@#GRESET          ;RESET DEFAULT PATTERN IN OUTPUT
2729                                ;BUFFER.
2730 010642 077330          SOB    R3,G42
2731
2732
2733 010644 000137 011052      JMP    @#GDONE
2734
2735                                ;USE THIS ROUTINE TO INITIALIZE ALL THE DATA BUFFERS.
2736 010650 012705 010776  GSETUP: MOV    #GFLAG1,R5
2737 010654 012704 000026      MOV    #26,R4
2738 010660 005025          1$:    CLR    (R5)+
2739 010662 077402          SOB    R4,1$
2740
2741 010664 012705 011012      MOV    #GPAT10,R5
2742 010670 012704 000010      MOV    #10,R4
2743 010674 005125          2$:    COM    (R5)+
2744 010676 077402          SOB    R4,2$
2745
2746 010700 020067 000076      GS1:   CMP    R0,GPAT00
2747 010704 001401          BEQ    3$
2748 010706 000207          RTS    PC
2749
2750 010710 012705 011042      3$:   MOV    #GDAT00,R5
2751 010714 012704 000004      MOV    #4,R4
2752 010720 005125          4$:   COM    (R5)+
2753 010722 077402          SOB    R4,4$
2754 010724 000207          RTS    PC
2755
2756 010726 012705 011042      GRESET: MOV   #GDAT00,R5
2757 010732 012704 000004      MOV    #4,R4
2758 010736 005025          1$:   CLR    (R5)+
2759 010740 077402          SOB    R4,1$
2760 010742 000137 010700      JMP    @#GS1
2761
2762                                ;SEE IF THE DATA WRITTEN MATCHES THE DATA READ.
2763 010746 011637 001244      GCMP:  MOV    (SP),@#STMP5      ;STORE ERROR PC
2764 010752 012705 011042      MOV    #GDAT00,R5
2765 010756 012704 000004      MOV    #4,R4
2766 010762 010002          MOV    R0,R2
2767 010764 022225          1$:   CMP    (R2)+,(R5)+
2768 010766 001401          BEQ    2$
2769 010770 104003          ERROR  3
2770 010772 077404          2$:   SOB    R4,1$          ;PROBABLY BAD MMU, OTHERWISE IT IS THE FP CHIP
2771 010774 000207          RTS    PC
2772
2773
2774
2775
2776 010776 000000          GFLAG1: 0
2777 011000 000000          GFLAG2: 0
2778
2779 011002 000000          GPAT00: 0
2780 011004 000000          GPAT01: 0
2781 011006 000000          GPAT02: 0
2782 011010 000000          GPAT03: 0

```

```

2783
2784 011012 177777 GPAT10: -1
2785 011014 177777 GPAT11: -1
2786 011016 177777 GPAT12: -1
2787 011020 177777 GPAT13: -1
2788
2789 011022 177777 GAND0: -1
2790 011024 177777 GAND1: -1
2791 011026 177777 GAND2: -1
2792 011030 177777 GAND3: -1
2793
2794 011032 000000 GOR0: 0
2795 011034 000000 GOR1: 0
2796 011036 000000 GOR2: 0
2797 011040 000000 GOR3: 0
2798
2799 011042 000000 GDAT00: 0
2800 011044 000000 GDAT01: 0
2801 011046 000000 GDAT02: 0
2802 011050 000000 GDAT03: 0
2803
2804
2805 011052 GDONE:
2806 011052 104413 RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
2807 ;SEE IF THE USER HAS EXPRESSED
2808 ;THE DESIRE TO CHANGE THE SOFTWARE
2809 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
2810 ;THE USER TYPED CONTROL G?).
2811
2812
2813 ;:*****
2814 ;*TEST 12 FPP ACCUMULATORS DUAL ADDRESS TEST
2815 ;*
2816 ;*THIS TEST PERFORMS A DUAL ADDRESSING TEST ON THE FLOATING ACCUMULATORS.
2817 ;*NOTE THAT ACCUMULATOR ZERO IS USED TO ACCESS ALL THE OTHERS.
2818 ;*
2819 ;:*****
2820 011054 000004 TST12: SCOPF
2821 011056 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
2822
2823 011060 005037 011604 H1: CLR @#HFLAG
2824 011064 012700 011606 MOV #HA1W,R0 ;INITIALIZE THE LOAD BUFFER DATA.
2825 011070 012701 011726 MOV #HDAT1,R1
2826 011074 012703 000024 MOV #24,R3
2827 011100 012120 H2: MOV (R1)+,(R0)+
2828 011102 077302 SOB R3,H2
2829
2830 011104 004767 000422 JSR PC,HCLR ;CLEAR THE OUTPUT DATA BUFFER.
2831
2832 011110 170011 H3: SETD
2833 ;LOAD ACCUMULATOR 1
2834 011112 012700 011606 MOV #HA1W,R0
2835 011116 172410 LDD (R0),ACC
2836 011120 174001 STD ACC,AC1
2837 ;LOAD ACCUMULATOR 2
2838 011122 012700 011616 MOV #HA2W,R0
    
```

```

2839 011126 172410          LDD    (R0),AC0
2840 011130 174002          STD    AC0,AC2
2841                          ;LOAD ACCUMULATOR 3
2842 011132 012700 011626   MOV    #HA3W,R0
2843 011136 172410          LDD    (R0),AC0
2844 011140 174003          STD    AC0,AC3
2845                          ;LOAD ACCUMULATOR 4
2846 011142 012700 011636   MOV    #HA4W,R0
2847 011146 172410          LDD    (R0),AC0
2848 011150 174004          STD    AC0,AC4
2849                          ;LOAD ACCUMULATOR 5
2850 011152 012700 011646   MOV    #HA5W,R0
2851 011156 172410          LDD    (R0),AC0
2852 011160 174005          STD    AC0,AC5
2853
2854 011162 004737 011416   H4:    JSR    PC,@#HSTD          ;GO READ ALL ACCUMULATORS BACK.
2855
2856 011166 004737 011474          JSR    PC,@#HCMP          ;SEE IF DATA IS CORRECT.
2857
2858                          ;COMPLIMENT EACH WORD OF THE DATA STORED IN ACCUMULATOR 1,
2859                          ;RELOAD THAT ACCUMULATOR, READ ALL THE ACCUMULATORS BACK AND CHECK
2860                          ;THE DATA.
2861 011172 012700 011606   MOV    #HA1W,R0
2862 011176 012702 000004   MOV    #4,R2
2863 011202 010001          MOV    R0,R1
2864 011204 005121          H5:    COM    (R1)+
2865 011206 172410          LDD    (R0),AC0
2866 011210 174001          STD    AC0,AC1
2867 011212 004737 011416   JSR    PC,@#HSTD          ;READ ALL THE ACCUMULATORS BACK.
2868 011216 004737 011474   JSR    PC,@#HCMP          ;CHECK THE DATA.
2869 011222 077210          SOB    R2,H5
2870
2871                          ;COMPLIMENT EACH WORD OF THE DATA STORED IN ACCUMULATOR 2,
2872                          ;RELOAD THAT ACCUMULATOR, READ ALL THE ACCUMULATORS BACK AND CHECK
2873                          ;THE DATA.
2874 011224 012700 011616   MOV    #HA2W,R0
2875 011230 012702 000004   MOV    #4,R2
2876 011234 010001          MOV    R0,R1
2877 011236 005121          H6:    COM    (R1)+
2878 011240 172410          LDD    (R0),AC0
2879 011242 174002          STD    AC0,AC2
2880 011244 004737 011416   JSR    PC,@#HSTD          ;READ ALL THE ACCUMULATORS BACK.
2881 011250 004737 011474   JSR    PC,@#HCMP          ;CHECK THE DATA.
2882 011254 077210          SOB    R2,H6
2883
2884                          ;COMPLIMENT EACH WORD OF THE DATA STORED IN ACCUMULATOR 3,
2885                          ;RELOAD THAT ACCUMULATOR, READ ALL THE ACCUMULATORS BACK AND CHECK
2886                          ;THE DATA.
2887 011256 012700 011626   MOV    #HA3W,R0
2888 011262 012702 000004   MOV    #4,R2
2889 011266 010001          MOV    R0,R1
2890 011270 005121          H7:    COM    (R1)+
2891 011272 172410          LDD    (R0),AC0
2892 011274 174003          STD    AC0,AC3
2893 011276 004737 011416   JSR    PC,@#HSTD          ;READ ALL THE ACCUMULATORS BACK.
2894 011302 004737 011474   JSR    PC,@#HCMP          ;CHECK THE DATA.
    
```

```

2895 011306 077210          SOB      R2,H7
2896
2897          ;COMPLIMENT EACH WORD OF THE DATA STORED IN ACCUMULATOR 4,
2898          ;RELOAD THAT ACCUMULATOR, READ ALL THE ACCUMULATORS BACK AND CHECK
2899          ;THE DATA.
2900 011310 012700 011636      MOV      #HA4W,R0
2901 011314 012702 000004      MOV      #4,R2
2902 011320 010001              MOV      R0,R1
2903 011322 005121          H10:    COM      (R1)+
2904 011324 172410              LDD      (R0),AC0
2905 011326 174004              STD      AC0,AC4
2906 011330 004737 011416      JSR      PC,@#HSTD          ;READ ALL THE ACCUMULATORS BACK.
2907 011334 004737 011474      JSR      PC,@#HCMP         ;CHECK THE DATA.
2908 011340 077210          SOB      R2,H10
2909
2910          ;COMPLIMENT EACH WORD OF THE DATA STORED IN ACCUMULATOR 5,
2911          ;RELOAD THAT ACCUMULATOR, READ ALL THE ACCUMULATORS BACK AND CHECK
2912          ;THE DATA.
2913 011342 012700 011646      MOV      #HA5W,R0
2914 011346 012702 000004      MOV      #4,R2
2915 011352 010001              MOV      R0,R1
2916 011354 005121          H11:    COM      (R1)+
2917 011356 172410              LDD      (R0),AC0
2918 011360 174005              STD      AC0,AC5
2919 011362 007737 011416      JSR      PC,@#HSTD          ;READ ALL THE ACCUMULATORS BACK.
2920 011366 004737 011474      JSR      PC,@#HCMP         ;CHECK THE DATA.
2921 011372 077210          SOB      R2,H11
2922
2923
2924 011374 005737 011604          TST      @#HFLAG
2925 011400 001402          BEQ      H12
2926 011402 000137 011776          JMP      @#HDONE
2927
2928 011406 005137 011604          H12:    COM      @#HFLAG
2929 011412 000137 011110          JMP      @#H3
2930
2931          ;STORE ALL ACCUMULATORS IN THE OUTPUT BUFFERS.
2932 011416 004737 011532          HSTD:   JSR      PC,@#HCLR          ;CLEAR ALL OUTPUT BUFFERS.
2933          ;STORE ACCUMULATOR 1
2934 011422 012704 011656          MOV      #HA1R,R4
2935 011426 172401              LDD      AC1,AC0
2936 011430 174014              STD      AC0,(R4)
2937          ;STORE ACCUMULATOR 2
2938 011432 012704 011666          MOV      #HA2R,R4
2939 011436 172402              LDD      AC2,AC0
2940 011440 174014              STD      AC0,(R4)
2941          ;STORE ACCUMULATOR 3
2942 011442 012704 011676          MOV      #HA3R,R4
2943 011446 172403              LDD      AC3,AC0
2944 011450 174014              STD      AC0,(R4)
2945          ;STORE ACCUMULATOR 4
2946 011452 012704 011706          MOV      #HA4R,R4
2947 011456 172404              LDD      AC4,AC0
2948 011460 174014              STD      AC0,(R4)
2949          ;STORE ACCUMULATOR 5
2950 011462 012704 011716          MOV      #HA5R,R4
    
```

```

2951 011466 172405          LDD    AC5,AC0
2952 011470 174014          STD    AC0,(R4)
2953 011472 000207          RTS    PC
2954
2955          ;COMPARE DATA LOADED WITH DATA READ.
2956 011474 012637 011602  HCMP:  MOV    (SP)+,@#HADR          ;SAVE RETURN ADDRESS.
2957 011500 012703 011606          MOV    #HA1W,R3
2958 011504 012704 011656          MOV    #HA1R,R4
2959 011510 012705 000024          MOV    #24,R5
2960 011514 022324          HCMP1: CMP    (R3)+,(R4)+
2961 011516 001402          BEQ    HCMP2
2962 011520 000137 011550          JMP    @#HERROR
2963 011524 077505          HCMP2: SOB   R5,HCMP1
2964 011526 000177 000050          JMP    @#HADR
2965
2966          ;CLEAR THE DATA OUTPUT BUFFER.
2967 011532 012704 011656  HCLR:  MOV    #HA1R,R4
2968 011536 012705 000024          MOV    #24,R5
2969 011542 005024          HCLR1: CLR    (R4)+
2970 011544 077502          SOB   R5,HCLR1
2971 011546 000207          RTS    PC
2972
2973          ;REPORT ERROR.
2974 011550          HERROR:
2975 011550 012703 011606          MOV    #HA1W,R3
2976 011554 012704 001236          MOV    #STMP2,R4
2977 011560 012705 000012          MOV    #12,R5
2978 011564 010324          1$:   MOV    R3,(R4)+
2979 011566 062703 000010          ADD    #10,R3
2980 011572 077504          SOB   R5,1$
2981 011574 104001          2$:   ERROR  1
2982 011576 000137 011776          JMP    @#HDONE
2983
2984
2985 011602 000000          HADR:  0
2986 011604 000000          HFLAG: 0
2987
2988 011606 000000 000000 000000 HA1W:  .WORD  0,0,0,0
2989 011614 000000
2990 011616 000000 000000 000000 HA2W:  .WORD  0,0,0,0
2991 011624 000000
2992 011626 000000 000000 000000 HA3W:  .WORD  0,0,0,0
2993 011634 000000
2994 011636 000000 000000 000000 HA4W:  .WORD  0,0,0,0
2995 011644 000000
2996 011646 000000 000000 000000 HA5W:  .WORD  0,0,0,0
2997 011654 000000
2998
2999 011656 000000 000000 000000 HA1R:  .WORD  0,0,0,0
3000 011664 000000
3001 011666 000000 000000 000000 HA2R:  .WORD  0,0,0,0
3002 011674 000000
3003 011676 000000 000000 000000 HA3R:  .WORD  0,0,0,0
3004 011704 000000
3005 011706 000000 000000 000000 HA4R:  .WORD  0,0,0,0
3006 011714 000000
    
```



```

3007 011716 000000 000000 000000 HASR: .WORD 0,0,0,0
3008 011724 000000
3009
3010 011726 073567 073567 073567 HDAT1: .WORD 73567,73567,73567,73567
3011 011734 073567
3012 011736 063146 063146 063146 HDAT2: .WORD 63146,63146,63146,63146
3013 011744 063146
3014 011746 010421 010421 010421 HDAT3: .WORD 10421,10421,10421,10421
3015 011754 010421
3016 011756 031463 031463 031463 HDAT4: .WORD 31463,31463,31463,31463
3017 011764 031463
3018 011766 042104 042104 042104 HDAT5: .WORD 42104,42104,42104,42104
3019 011774 042104
3020

```

```

3021 011776 HDONE:
3022 011776 104413 RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
3023 ;SEE IF THE USER HAS EXPRESSED
3024 ;THE DESIRE TO CHANGE THE SOFTWARE
3025 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
3026 ;THE USER TYPED CONTROL G?).
3027
3028
3029

```

```

3030 :*****
3031 :*TEST 13 FSRC MODE 0 WITH ILLEGAL ACCUMULATOR TEST
3032 :*
3033 :*THIS IS A TEST OF FSRC MODE 0 WITH ACCUMULATORS 6 AND 7. USE OF
3034 :*EITHER OF THESE NON-EXISTENT ACCUMULATORS SHOULD RESULT IN A TRAP TO 244
3035 :*WITH FEC=2 (ILLEGAL OPCODE TRAP)
3036 :*

```

```

3037 012000 000004 TST13: SCOPE
3038 012002 S1:
3039 012002 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
3040 012004 170011 SETD ;SET FD
3041 012006 012700 012512 MOV #SPAT10,R0 ;LOAD ACO
3042 012012 172410 LDD (R0),AC0
3043
3044 012014 012737 012212 000244 MOV #SERR0,@#FPVECT ;USE OF THE NON-EXISTENT AC-
3045 ;CUMULATOR SHOULD RESULT IN
3046 ;A TRAP TO 244.
3047 012022 012700 000001 MOV #1,R0 ;A FAILURE IN THE FSRC FLOWS
3048 ;WILL RESULT IN AN ODD ADDRESS
3049 012026 012737 012422 000004 MOV #SERR1,@#ERRVECT ;TRAP TO 4.
3050 012034 005003 CLR R3
3051
3052 012036 172407 S2: LDD AC7,AC0
3053 012040 170000 S3: CFCC
3054 012042 005203 INC R3
3055 012044 005203 S4: INC R3
3056
3057 012046 012701 012522 MOV #SDAT00,R1 ;NO TRAP OCCURRED!!
3058 012052 174011 STD ACO,(R1) ;SEE IF ACO WAS MODIFIED.
3059
3060 012054 012701 012522 MOV #SDAT00,R1
3061 012060 012702 012512 MOV #SPAT10,R2
3062 012064 012703 000004 MOV #4,R3

```

G 5

CJKDCA-A KEF11-A FP DIAG PART 1 MACY11 30A(1052) 11-JUN-79 11:50 PAGE 58  
 CJKDCA.P11 08-JUN-79 08:40 T13 FSRC MODE 0 WITH ILLEGAL ACCUMULATOR TEST SEQ 0058

```

3063 012070 022122          S5:  CMP      (R1)+,(R2)+
3064 012072 001402          BEQ      S6
3065 012074 000137 012352    JMP      @#SERR2          ;BAD DATA
3066 012100 077305          S6:  SOB      R3,S5
3067
3068 012102 000137 012376    JMP      @#SERR3          ;MISSING TRAP
3069
3070          ;NOW TEST AC6.
3071 012106          S7:
3072 012106 104414          LPERR
3073 012110 170011          SETD
3074
3075 012112 012700 012512    MOV      #SPAT10,R0      ;LOAD ACO
3076 012116 172410          LDD      (R0),AC0
3077
3078 012120 012737 012270 000244  MOV      #SERR4,@#FPVECT
3079 012126 012700 000001    MOV      #1,R0
3080 012132 012737 012454 000004  MOV      #SERR5,@#ERRVECT
3081 012140 005003          CLR      R3
3082
3083 012142 172406          S8:  LDD      AC6,AC0
3084 012144 170000          S9:  CFCC
3085 012146 005203          INC      R3
3086 012150 005203          S10: INC      R3
3087
3088 012152 012701 012522    MOV      #SDAT00,R1
3089 012156 174011          STD      AC0,(R1)      ;NO TRAP! GET ACO.
3090
3091 012160 012701 012522    MOV      #SDAT00,R1      ;WAS ACO MODIFIED.
3092 012164 012702 012512    MOV      #SPA110,R2
3093 012170 012703 000004    MOV      #4,R3
3094 012174 022122          S11: CMP      (R1)+,(R2)+
3095 012176 001402          BEQ      S12
3096 012200 000137 012364    JMP      @#SERR6
3097 012204 077305          S12: SOB      R3,S11
3098 012206 000137 012410    JMP      @#SERR7
3099
3100          ;TRAPPED TO 244.
3101 012212 021627 012040    SERR0: CMP      (SP),#S3      ;PC OF TRAP CORRECT?
3102 012216 001402          BEQ      1$
3103 012220 000137 062534    JMP      @#FPSPUR
3104
3105 012224 012737 012106 012506 1$:  MOV      #S7,@#SADR
3106
3107 012232 011637 001236    SERR10: MOV      (SP),@#STMP2
3108 012236 022626          CMP      (SP)+,(SP)+
3109 012240 005004          CLR      R4
3110 012242 170204          STFPS   R4          ;IS FPS CORRECT?
3111 012244 022704 100200    CMP      #100200,R4
3112 012250 001020          BNE     SERR15
3113
3114 012252 005004          CLR      R4
3115 012254 170304          STST   R4          ;IS FEC CORRECT?
3116 012256 022704 000002    CMP      #2,R4
3117 012262 001023          BNE     SERR20
3118 012264 000177 000216    JMP      @#SADR

```

```

3119
3120 012270 021627 012144          SERR4:  (MP      (SP),#S9
3121 012274 001402                   BEQ      1$
3122 012276 000137 062534          JMP      @#FPSPUR
3123 012302 012737 012532 012506 1$:  MOV      #SDONE,@#SADR
3124 012310 000750                   BR       SERR10
3125
3126          ;REPORT FPS FAILURE:
3127 012312 012737 100200 001242 SERR15: MOV      #100200,@#TMP4
3128 012320 010437 001240          MOV      R4,@#TMP3
3129 012324 104001          1$:  ERROR  1
3130 012326 000177 000154          JMP      @SADR
3131
3132          ;REPORT FEC BAD:
3133 012332 012737 000002 001242 SERR20: MOV      #2,@#TMP4
3134 012340 010437 001240          MOV      R4,@#TMP3
3135 012344 104001          1$:  ERROR  1
3136 012346 000177 000134          JMP      @SADR
3137
3138
3139          ;ACO WAS MODIFIED. (BUT FSRC) FORK FAILED.
3140 012352 012737 012036 001236 SERR2:  MOV      #S2,@#TMP2
3141 012360 104001          1$:  ERROR  1
3142 012362 000463                   BR       SDONE
3143 012364 012737 012142 001236 SERR6:  MOV      #S8,@#TMP2
3144 012372 104001          1$:  ERROR  1
3145 012374 000456                   BR       SDONE
3146
3147 012376 012737 012036 001236 SERR3:  MOV      #S2,@#TMP2
3148 012404 104001          1$:  ERROR  1
3149 012406 000451                   BR       SDONE
3150 012410 012737 012142 001236 SERR7:  MOV      #S8,@#TMP2
3151 012416 104001          1$:  ERROR  1
3152 012420 000444                   BR       SDONE
3153
3154          ;FAILURE OF (BUT FSRC) CAUSED AN ODD ADDRESS TRAP TO 4.
3155 012422 021627 012040          SERR1:  CMP      (SP),#S3          ;DID TRAP OCCUR ON TESTED INSTRUCTION?
3156 012426 001405                   BEQ      1$
3157 012430 021627 012044          CMP      (SP),#S4
3158 012434 001402                   BEQ      1$
3159 012436 000137 062566          JMP      @#CPSPUR
3160
3161 012442 011637 001236          1$:  MOV      (SP),@#TMP2
3162 012446 022626          2$:  CMP      (SP)+,(SP)+
3163 012450 104001          ERROR  1
3164 012452 000427                   BR       SDONE
3165
3166 012454 021627 012142          SERR5:  CMP      (SP),#S8          ;DID TRAP OCCUR ON TEST INSTRUCTION?
3167 012460 001405                   BEQ      1$
3168 012462 021627 012144          CMP      (SP),#S9
3169 012466 001402                   BEQ      1$
3170 012470 000137 062566          JMP      @#CPSPUR
3171
3172 012474 011637 001236          1$:  MOV      (SP),@#TMP2
3173 012500 022626          2$:  CMP      (SP)+,(SP)+
3174 012502 104001          ERROR  1
    
```

```

3175 012504 000412          BR      SDONE
3176
3177 012506 000000          SADR:   0
3178 012510 177777          -1
3179 012512 010421          SPAT10: 10421
3180 012514 021042          SPAT11: 21042
3181 012516 031463          SPAT12: 31463
3182 012520 042104          SPAT13: 42104
3183
3184 012522 000000          SDAT00: 0
3185 012524 000000          SDAT01: 0
3186 012526 000000          SDAT02: 0
3187 012530 000000          SDAT03: 0
3188
3189 012532          SDONE:
3190 012532 104413          RSETUP          ;GO INITIALIZE THE FPS AND STACK; AND
3191                                     ;SEE IF THE USER HAS EXPRESSED
3192                                     ;THE DESIRE TO CHANGE THE SOFTWARE
3193                                     ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
3194                                     ;THE USER TYPED CONTROL G?).
3195
3196          ;:*****
3197          ;*TEST 14          FSRC MODE 2 TEST
3198          ;*
3199          ;* THIS IS A TEST OF FSRC MODE 2, AUTO
3200          ;* INCREMENT MODE.
3201          ;*
3202          ;:*****
3203 012534 000004          TST14:  SCOPE
3204 012536 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
3205
3206 012540          J1:
3207 012540 170011          SETD          ;SET DOUBLE MODE
3208
3209 012542 012700 013016          MOV      #JDAT0,R0
3210 012546 172410          LDD      (R0),AC0          ;LOAD AC0=ALL 1
3211
3212 012550 012700 012776          MOV      #JDAT10,R0
3213 012554 005003          CLR      R3
3214 012556 012737 012646 000004          MOV      #JERR0,@#ERRVECT
3215
3216 012564 172420          J2:      LDD      (R0)+,AC0          ;TEST INSTRUCTION
3217 012566 005203          J3:      INC      R3
3218 012570 005203          J4:      INC      R3
3219
3220 012572 012701 013006          MOV      #JDAT00,R1
3221 012576 174011          STD      AC0,(R1)          ;PICK UP RESULTS
3222
3223 012600 020027 012766          CMP      R0,#JBUFO          ;WAS AN AUTO
3224 012604 001001          BNE      1$                ;DECREMENT EXECUTED?
3225 012606 000442          BR       JERR1
3226
3227 012610 012702 012776          1$:     MOV      #JDAT10,R2          ;IS DATA CORRECT?
3228 012614 012703 013006          MOV      #JDAT00,R3
3229 012620 012704 000004          MCV      #4,R4
3230 012624 022223          J5:     CMP      (R2)+,(R3)+

```

```

3231 012626 001401          BEQ    J6
3232 012630 000443          BR     JERR2
3233 012632 077404          J6:   SOB    R4,J5
3234
3235 012634 022700 013006          CMP    #JDAT10+10,RO ;WAS RO INCREM.
3236 012640 001401          BEQ    J7 ;BY 10 (OCTAL)
3237 012642 000424          BR     JERR1
3238
3239 012644 000470          J7:   BR     JDONE
3240
3241          ;IF A TRAP THROUGH 4 OCCURS COME HERE
3242
3243 012646 021627 012566          JERR0: CMP    (SP),#J3 ;SEE IF THE TRAP
3244 012652 001405          BEQ    J10 ;OCCURRED ON THE
3245 012654 021627 012570          CMP    (SP),#J4 ;TESTED INSTRUCTION
3246 012660 001402          BEQ    J10
3247 012662 000137 062566          JMP    @#CPSPUR
3248
3249 012666 012737 000762 001240          J10:  MOV    #762,@#STMP3 ;REPORT FSRC FLOW
3250 012674 012737 000322 001242          MOV    #322,@#STMP4 ;FAILURE
3251 012702 011637 001236          MOV    (SP),@#STMP2
3252 012706 022626          CMP    (SP)+,(SP)+
3253 012710 104001          1$:   ERROR 1
3254 012712 000445          BR     JDONE
3255
3256 012714          JERR1:          ;REPORT, RO NOT
3257 012714 012737 012564 001236          MOV    #J2,@#STMP2 ;CORRECTLY AFFECTED
3258 012722 010037 001240          MOV    RO,@#STMP3
3259 012726 012737 013006 001242          MOV    #JDAT10+10,@#STMP4
3260 012734 104001          1$:   ERROR 1
3261 012736 000433          BR     JDONE
3262
3263          ;REPORT DATA FAILURE
3264
3265 012740          JERR2:
3266 012740 012737 012564 001236          MOV    #J2,@#STMP2
3267 012746 012737 012776 001240          MOV    #JDAT10,@#STMP3
3268 012754 012737 013006 001242          MOV    #JDAT00,@#STMP4
3269 012762 104001          1$:   ERROR 1
3270 012764 000420          BR     JDONE
3271
3272 012766 010421          JBUF0: .WORD 010421
3273 012770 021042          JBUF1:          021042
3274 012772 042104          JBUF2:          042104
3275 012774 031463          JBUF3:          031463
3276
3277 012776 052525          JDAT10:         052525
3278 013000 114631          JDAT11:         114631
3279 013002 063146          JDAT12:         063146
3280 013004 073567          JDAT13:         073567
3281
3282 013006 000000          JDAT00:         0
3283 013010 000000          JDAT01:         0
3284 013012 000000          JDAT02:         0
3285 013014 000000          JDAT03:         0
3286
    
```

```

3287 013016 177777          JDAT0:          -1
3288 013020 177777          JDAT1:          -1
3289 013022 177777          JDAT2:          -1
3290 013024 177777          JDAT3:          -1
3291
3292
3293 013026
3294 013026 104413          JDONE:
3295                                RSETUP
3296                                ;GO INITIALIZE THE FPS AND STACK; AND
3297                                ;SEE IF THE USER HAS EXPRESSED
3298                                ;THE DESIRE TO CHANGE THE SOFTWARE
3299                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
3300                                ;THE USER TYPED CONTROL G?).
3301
3302          ;*****
3303          ;*TEST 15          FSRC MODE 4 TEST
3304          ;*
3305          ;* THIS IS A TEST OF FSRC MODE 4, AUTO
3306          ;* DECREMENT MODE.
3307          ;*****
3308 013030 000004          TST15:  SCOPE
3309 013032 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
3310
3311 013034
3312 013034 170011          K1:
3313                                SETD          ;SET DOUBLE MODE
3314 013036 012700 013310          MOV          #KPATO,R0
3315 013042 172410          LDD          (R0),ACO          ;LOAD A DEFAULT
3316                                ;PATTERN INTO ACO
3317 013044 012700 013270          MOV          #KBUF0,R0
3318 013050 005003          CLR          R3
3319 013052 012737 013142 000004          MOV          #KERR0,@#ERRVECT
3320
3321 013060 172440          K2:  LDD          -(R0),ACO          ;TEST INSTRUCTION
3322 013062 005203          K3:  INC          R3
3323 013064 005203          K4:  INC          R3
3324
3325 013066 012701 013300          MOV          #KDAT00,R1
3326 013072 174011          STD          ACO,(R1)          ;PICK UP THE RESULT
3327
3328 013074 020027 013300          CMP          R0,#KBUF0+10          ;WAS AN AUTO
3329 013100 001001          BNE          1$          ;INCREMENT EXECUTED
3330 013102 000441          BR          KERR1
3331
3332 013104 012702 013260          1$:  MOV          #KDAT10,R2          ;IS DATA CORRECT?
3333 013110 012703 013300          MOV          #KDAT00,R3
3334 013114 012704 000004          MOV          #4,R4
3335 013120 022223          K5:  CMP          (R2)+,(R3)+
3336 013122 001401          BEQ          K6
3337 013124 000442          BR          KERR2
3338 013126 077404          K6:  SOB          R4,K5
3339
3340 013130 022700 013260          CMP          #KBUF0-10,R0          ;WAS R0 DECREMENTED
3341 013134 001401          BEQ          K7          ;PROPERLY?
3342 013136 000423          BR          KERR1
    
```

```

3343
3344 013140 000467      K7:      BR      KDONE
3345
3346      ;TRAP TO HERE ON AN ODD ADDRESS ERROR
3347
3348 013142 021627 013062  KERR0:  CMP      (SP),#K3      ;SEE IF THE ERROR
3349 013146 001405      BEQ      K10          ;OCCURRED AT THE
3350 013150 021627 013064  CMP      (SP),#K4      ;INSTRUCTION TESTED.
3351 013154 001402      BEQ      K10
3352 013156 000137 062566  JMP      @#CPSPUR
3353
3354 013162 012737 000762 001240  K10:    MOV      #762,@#STMP3  ;REPORT FAILURE IN
3355 013170 012737 000324 001242  MOV      #324,@#STMP4  ;FSRC FLOWS
3356 013176 011637 001236      MOV      (SP),@#STMP2
3357 013202 104001      1$:     ERROR    1
3358 013204 000445      BR      KDONE
3359
3360 013206      KERR1:  MOV      #K2,@#STMP2  ;REPORT, R0
3361 013206 012737 013060 001236  MOV      R0,@#STMP3    ;INCORRECTLY AFFECTED.
3362 013214 010037 001240      MOV      #KDAT10,@#STMP4
3363 013220 012737 013260 001242  1$:     ERROR    1
3364 013226 104001      BR      KDONE
3365 013230 000433
3366
3367      ;REPORT DATA FAILURE
3368
3369 013232      KERR2:  MOV      #K2,@#STMP2
3370 013232 012737 013060 001236  MOV      #KDAT10,@#STMP3
3371 013240 012737 013260 001240  MOV      #KDAT00,@#STMP4
3372 013246 012737 013300 001242  1$:     ERROR    1
3373 013254 104001      BR      KDONE
3374 013256 000420
3375
3376 013260 052525      KDAT10: .WORD    052525
3377 013262 114631      KDAT11:      114631
3378 013264 063140      KDAT12:      063140
3379 013266 073567      KDAT13:      073567
3380
3381 013270 010421      KBUF0:       010421
3382 013272 031463      KBUF1:       031463
3383 013274 042104      KBUF2:       042104
3384 013276 021042      KBUF3:       021042
3385
3386 013300 000000      KDAT00:      0
3387 013302 000000      KDAT01:      0
3388 013304 000000      KDAT02:      0
3389 013306 000000      KDAT03:      0
3390
3391 013310 177777      KPAT0:       -1
3392 013312 177777      KPAT1:       -1
3393 013314 177777      KPAT2:       -1
3394 013316 177777      DPAT3:       -1
3395
3396 013320      KDONE:
3397 013320 104413      RSETUP      ;GO INITIALIZE THE FPS AND STACK; AND
3398      ;SEE IF THE USER HAS EXPRESSED
    
```



```

3399                                     ;THE DESIRE TO CHANGE THE SOFTWARE
3400                                     ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
3401                                     ;THE USER TYPED CONTROL G?).
3402
3403
3404                                     ;*****
3405                                     ;*TEST 16      FSRC MODE 2, WITH FD=0, TEST
3406                                     ;*
3407                                     ;* THIS IS A TEST OF FSRC MODE 2 WITH
3408                                     ;* FD=0. (AUTO INCREMENT)
3409                                     ;*
3410                                     ;*****
3411 013322 000004 TST16: SCOPE
3412 013324 104414 LPERR                                     ;SET UP THE LOOP ON ERROR ADDRESS.
3413
3414 013326 L1: SETD                                     ;SET DOUBLE MODE
3415 013326 170011
3416
3417 013330 012700 013576 MOV #LPAT10,R0
3418 013334 172410 LDD (R0),AC0                                     ;LOAD AC0
3419
3420 013336 012700 013620 MOV #LDAT10,R0                                     ;SET UP THE INPUT
3421 013342 012701 013606 MOV #LPAT20,R1                                     ;DATA
3422 013346 012702 000004 MOV #4,R2
3423
3424 013352 012120 1$: MOV (R1)+,(R0)+
3425 013354 077202 SOB R2,1$
3426
3427 013356 012700 013620 MOV #LDAT10,R0
3428 013362 005003 CLR R3
3429 013364 170001 SETF                                     ;CLEAR FD.
3430
3431 013366 172420 L2: LDF (R0)+,AC0
3432 013370 005203 L3: INC R3
3433
3434 013372 L4:
3435 013372 170011 SETD                                     ;SET FD
3436
3437 013374 012701 013632 MOV #LDAT00,R1
3438 013400 174011 STD AC0,(R1)                                     ;PICK UP RESULTS
3439
3440 013402 020027 013624 CMP R0,#LDAT12                                     ;WAS R0 INCREMENTED
3441 013406 001401 BEQ 1$                                     ;CORRECTLY BY 4
3442 013410 000421 BR LERR1
3443
3444 013412 012737 177777 013624 1$: MOV #-1,@#LDAT12
3445 013420 012737 177777 013626 MOV #-1,@#LDAT13
3446 013426 012702 013620 MOV #LDAT10,R2                                     ;IS DATA CORRECT
3447 013432 012703 013632 MOV #LDAT00,R3
3448 013436 012704 000004 MOV #4,R4
3449
3450 013442 022223 L5: CMP (R2)+,(R3)+
3451 013444 001401 BEQ L6
3452 013446 000427 BR LERR2
3453 013450 077404 L6: SOB R4,L5
3454

```

```

3455 013452 000473          BR      LDONE
3456
3457 013454          LERR1:          :REPORT FAILURE
3458 013454 012737 013366 001236      MOV      #L2,@#STMP2      :RO NOT INCREMENTED
3459 013462 010037 001240          MOV      R0,@#STMP3      :BY 4
3460 013466 012737 013624 001242          MOV      #LDAT12,@#STMP4
3461 013474 104001          1$:      ERROR      1
3462 013476 000461          BR      LDONE
3463
3464 013500          LERR3:          :REPORT DATA FAILURE.
3465 013500 012737 013366 001236      MOV      #L2,@#STMP2
3466 013506 012737 013620 001240          MOV      #LDAT10,@#STMP3
3467 013514 012737 013632 001242          MOV      #LDAT00,@#STMP4
3468 013522 104001          1$:      ERROR      1
3469 013524 000446          BR      LDONE
3470
3471 013526 012702 013606          LERR2:          :DID (BUT FD)
3472 013532 012703 013632          MOV      #LDAT00,R3      :FAIL.
3473 013536 012704 000004          MOV      #4,R4
3474 013542 022223          1$:      CMP      (R2)+,(R3)+
3475 013544 001355          BNE     LERR3
3476 013546 077403          SOB     R4,1$
3477 013550 012737 013366 001236          MOV      #L2,@#STMP2
3478 013556 012737 013620 001240          MOV      #LDAT10,@#STMP3
3479 013564 012737 013634 001242          MOV      #LDAT01,@#STMP4
3480 013572 104001          2$:      ERROR      1
3481 013574 000422          BR      LDONE
3482
3483 013576 177777          LPAT10: .WORD    -1
3484 013600 177777          LPAT11:          -1
3485 013602 177777          LPAT12:          -1
3486 013604 177777          LPAT13:          -1
3487
3488 013606 052525          LPAT20:          052525
3489 013610 114631          LPAT21:          114631
3490 013612 063142          LPAT22:          063142
3491 013614 073567          LPAT23:          073567
3492 013616 000001          .WORD    000001
3493 013620 000000          LDAT10:          0
3494 013622 000000          LDAT11:          0
3495 013624 000000          LDAT12:          0
3496 013626 000000          LDAT13:          0
3497 013630 000001          .WORD    000001
3498 013632 000000          LDAT00:          0
3499 013634 000000          LDAT01:          0
3500 013636 000000          LDAT02:          0
3501 013640 000000          LDAT03:          0
3502
3503 013642          LDONE:
3504 013642 104413          RSETUP          :GO INITIALIZE THE FPS AND STACK; AND
3505                                     :SEE IF THE USER HAS EXPRESSED
3506                                     :THE DESIRE TO CHANGE THE SOFTWARE
3507                                     :VIRTUAL CONSOLE SWITCH REGISTER (HAS
3508                                     :THE USER TYPED CONTROL G?).
3509
3510
    
```

```

3511
3512
3513
3514
3515
3516
3517
3518
3519 013644 000004
3520
3521 013646
3522 013646 170011
3523
3524 013650 012700 014142
3525 013654 172410
3526
3527 013656 005004
3528 013660 012737 014102 000004
3529
3530 013666 172427 000000
3531 013670
3532 013670 005204
3533 013672 005204
3534 013674 005204
3535 013676 005204
3536
3537 013700 020427 000003
3538 013704 001401
3539 013706 000443
3540
3541
3542
3543 013710 012700 014162
3544 013714 174010
3545
3546 013716 012700 014162
3547 013722 022720 005204
3548 013726 001401
3549 013730 000451
3550 013732 012701 000003
3551 013736 005720
3552 013740 001002
3553 013742 077103
3554 013744 000512
3555
3556 013746 012700 014162
3557 013752 012701 000004
3558 013756 022720 005204
3559 013762 001401
3560 013764 000433
3561 013766 077105
3562
3563 013770
3564 013770 012737 013666 001236
3565 013776 012737 014152 001240
3566 014004 012737 014162 001242

```

```

*****
*TEST 17 FSRC MODE 2 WITH GR7, IMMEDIATE MODE, TEST
*
* THIS IS A TEST OF FSRC MODE 2
* USING GR7 (THE PC). THIS IS IMMEDIATE
* MODE.
*
*****
TST17: SCOPE

M1:
    SETD
    MOV #MPAT10,R0
    LDD (R0),AC0 ;LOAD BACKGROUND
                    ;PATTERN INTO ACO.
    CLR R4
    MOV #MERR3,@#ERRVECT

M15:
    LDD #0,AC0 ;TEST INSTRUCTION
    .=-2 ;EFFECTIVELY: 05204 IS PUT IN THE FIRST
    .WORD 5204 ;16 BIT WORD, OR THE 'EXP-FRACTION' WORD.

M2:
M3:
M4:
    INC R4 ;NOTE THAT
    INC R4 ;005204=INC R4
    INC R4

    CMP R4,#3 ;SEE IF THE PC
    BEQ 1$ ;WAS INCREMENTED
    BR MERR0 ;BY 2 DURING THE
                ;INSTRUCTION. IF
                ;NOT THEN A BAD
                ;CONSTANT WAS GENERATED

1$:
    MOV #MDAT00,R0
    STD ACO,(R0) ;GET THE DATA

    MOV #MDAT00,R0
    CMP #5204,(R0)+ ;IS THE DATA CORRECT?
    BEQ M5
    BR MERR1

M5:
M6:
    MOV #3,R1
    TST (R0)+
    BNE M7
    SOB R1,M6
    BR MDONE

M7:
    MOV #MDAT00,R0 ;DID (BUT GRM) FAIL?
    MOV #4,R1

M8:
    CMP #5204,(R0)+
    BEQ M9
    BR MERR1

M9:
    SOB R1,M8

MERR2:
    MOV #M15,@#$TMP2 ;REPORT FAILURE
    MOV #MPAT20,@#$TMP3 ;OF (BUT GR7)
    MOV #MDAT00,@#$TMP4

```

```

3567 014012 104001          1$:      ERROR      1
3568 014014 000466          BR          MDONE
3569
3570 014016 012705 013672  MERR0:  MOV      #M2,R5          ;REPORT FAILURE
3571 014022 010537 001242      MOV      R5,@#STMP4      ;PC INCREMENTED
3572 014026 162704 000003      SUB      #3,R4
3573 014032 006304          ASL      R4
3574 014034 160405          SUB      R4,R5
3575 014036 010537 001240      MOV      R5,@#STMP3
3576 014042 012737 013666 001236  MOV      #M15,@#STMP2
3577 014050 104001          1$:      ERROR      1
3578 014052 000447          BR          MDONE
3579
3580 014054          MERR1:          ;REPORT DATA
3581 014054 012737 013666 001236      MOV      #M15,@#STMP2      ;FAILURE
3582 014062 012737 014162 001240      MOV      #MDAT00,@#STMP3
3583 014070 012737 014152 001242      MOV      #MPAT20,@#STMP4
3584 014076 104001          1$:      ERROR      1
3585 014100 000434          BR          MDONE
3586          ;TRAP TO HERE THROUGH 4.
3587 014102 032716 000001  MERR3:  BIT      #1,(SP)      ;SEE IF THE
3588 014106 001002          BNE      1$              ;TRAP TO 4 OCCURRED
3589 014110 000137 062566          JMP      @#CPSPUR        ;BECAUSE OF AN
3590          ;ODD ADDRESS
3591 014114 011637 001240  1$:      MOV      (SP),@#STMP3    ;IF YES REPORT
3592 014120 012737 013672 001242      MOV      #M2,@#STMP4      ;BAD CONSTANT
3593 014126 012737 013666 001236      MOV      #M15,@#STMP2    ;GENERATED
3594 014134 022626          CMP      (SP)+,(SP)+
3595 014136 104001          2$:      ERROR      1
3596 014140 000414          BR          MDONE
3597
3598 014142 177777          MPAT10:      -1
3599 014144 177777          MPAT11:      -1
3600 014146 177777          MPAT12:      -1
3601 014150 177777          MPAT13:      -1
3602
3603 014152 005204          MPAT20:      5204
3604 014154 005204          MPAT21:      5204
3605 014156 005204          MPAT22:      5204
3606 014160 005204          MPAT23:      5204
3607
3608 014162 0C0000          MDAT00:      0
3609 014164 000000          MDAT01:      0
3610 014166 000000          MDAT02:      0
3611 014170 000000          MDAT03:      0
3612
3613 014172          MDONE:
3614 014172 104413          RSETUP      ;GO INITIALIZE THE FPS AND STACK; AND
3615          ;SEE IF THE USER HAS EXPRESSED
3616          ;THE DESIRE TO CHANGE THE SOFTWARE
3617          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
3618          ;THE USER TYPED CONTROL G?).
3619
3620
3621          ;*****
3622          ;*TEST 20          FSRC MODE 3 TEST
    
```

```

3623
3624
3625
3626
3627
3628 014174 000004
3629
3630 014176
3631 014176 170011
3632
3633 014200 012700 014660
3634 014204 172410
3635
3636 014206 012700 014646
3637 014212 005003
3638 014214 012737 014370 000004
3639
3640
3641 014222 172430
3642 014224 005203
3643 014226 005203
3644
3645 014230 012701 014626
3646 014234 174011
3647
3648 014236 020027 014650
3649 014242 001437
3650
3651 014244 020027 014656
3652 014250 001001
3653 014252 000506
3654
3655 014254 020027 014636
3656 014260 001001
3657 014262 000520
3658
3659 014264 020027 014646
3660 014270 001023
3661
3662 014272 012702 014626
3663 014276 012703 000004
3664 014302 022227 177777
3665 014306 001002
3666 014310 077304
3667 014312 000510
3668
3669 014314 012702 014626
3670 014320 012703 014646
3671 014324 012704 000004
3672 014330 022223
3673 014332 001002
3674 014334 077403
3675 014336 000502
3676
3677 014340 000505
3678

: *
: * THIS IS A TEST OF FSRC MODE 3, AUTO INCREMENT
: * DEFERRED
: *
: *****
TST20: SCOPE

N1:      SETD                ;SET FD MODE
          MOV #NPAT10,R0
          LDD (R0),AC0        ;LOAD AC0 WITH A DEFAULT
          ;PATTERN
          MOV #NPAT20,R0
          CLR R3
          MOV #NERR0,@#ERRVECT ;IF A FAILURE OCCURS
          ;IN THE FSRC FLOWS AN
          ;ODD TRAP TO 4 COULD OCCUR
          ;TEST INSTRUCTION.

N2:      LDD @ (R0)+,AC0
N3:      INC R3
N4:      INC R3

          MOV #NDAT00,R1
          STD AC0,(R1)        ;GET THE DATA

          CMP R0,#NPAT20+2    ;WAS R0 INCREMENTED
          BEQ N12             ;BY 2?

N5:      CMP R0,#NPAT20+10   ;FSRC MODE 2?
          BNE N6
          BR NERR1

N6:      CMP R0,#NPAT20-10   ;FSRC MODE 4?
          BNE N7
          BR NERR2

N7:      CMP R0,#NPAT20
          BNE N11

          MOV #NDAT00,R2      ;FSRC MODE 0?
          MOV #4,R3
N8:      CMP (R2)+,#-1
          BNE N9
          SOB R3,N8
          BR NERR3

N9:      MOV #NDAT00,R2      ;FSRC MODE 1
          MOV #NPAT20,R3
          MOV #4,R4
N10:     CMP (R2)+,(R3)+
          BNE N11
          SOB R4,N10
          BR NERR4

N11:     BR NERR5
    
```

```

3679 014342 012702 014626      N12:  MOV    #NDAT00,R2      ;DATA CORRECT?
3680 014346 012703 014670      MOV    #NDAT10,R3
3681 014352 012704 000004      MOV    #4,R4
3682 014356 022223      N13:  CMP    (R2)+,(R3)+
3683 014360 001002      BNE   N14
3684 014362 077403      SOB   R4,N13
3685 014364 000545      BR    NDONE
3686
3687 014366 000504      N14:  BR    NERR6
3688
3689      ;IF AN ODD ADDRESS TRAP OCCURS COME HERE
3690      ;TO SEE IF THE FAILURE WAS IN THE FSRC
3691      ;FLOWS
3692
3693 014370 022716 014226      NERR0: CMP    #N4,(SP)      ;FSRC MODE 6 OR 7?
3694 014374 001412      BEQ   NERR10
3695 014376 022716 014224      CMP    #N3,(SP)
3696 014402 001402      BEQ   1$
3697 014404 000137 062566      JMP   @#CPSPUR
3698 014410 020027 014644      1$:  CMP    R0,#NPAT20-2      ;FSRC MODE 5?
3699 014414 001407      BEQ   NERR11
3700 014416 000137 062566      JMP   @#CPSPUR
3701
3702 014422      NERR10: MOV    (SP),@#STMP2      ;WENT TO FSRC
3703 014422 011637 001236      CMP    (SP)+,(SP)+      ;MODE 6 OR 7.
3704 014426 022626
3705 014430 104001      1$:  ERROR 1
3706 014432 000522      BR    NDONE
3707
3708 014434 011637 001236      NERR11: MOV    (SP),@#STMP2      ;WENT TO FSRC
3709 014440 022626      CMP    (SP)+,(SP)+      ;MODE 5.
3710 014442 012737 000627 001244      MOV    #627,@#STMP5
3711 014450 012737 000323 001250      MOV    #323,@#STMP7
3712 014456 012737 000325 001246      MOV    #325,@#STMP6
3713 014464 104001      1$:  ERROR 1
3714 014466 000504      BR    NDONE
3715 014470 012737 000322 001246      NERR1: MOV    #322,@#STMP6      ;FSRC MODE 2.
3716 014476 012737 000627 001244      NERR20: MOV    #627,@#STMP5
3717 014504 012737 000323 001250      MOV    #323,@#STMP7
3718 014512 012737 014222 001236      MOV    #N2,@#STMP2
3719 014520 104001      1$:  ERROR 1
3720 014522 000466      BR    NDONE
3721 014524 012737 000324 001246      NERR2: MOV    #324,@#STMP6      ;FSRC MODE 4
3722 014532 000761      BR    NERR20
3723 014534 012737 000320 001246      NERR3: MOV    #320,@#STMP6      ;FSRC MODE 0
3724 014542 000755      BR    NERR20
3725 014544 012737 000321 001246      NERR4: MOV    #321,@#STMP6      ;FSRC MODE 1
3726 014552 000751      BR    NERR20
3727
3728 014554 010037 001240      NERR5: MOV    R0,@#STMP3      ;R0 NOT
3729 014560 012737 014650 001242      MOV    #NPAT20+2,@#STMP4      ;INCREMENTED
3730 014566 012737 014222 001236      MOV    #N2,@#STMP2      ;PROPERLY.
3731 014574 104001      1$:  ERROR 1
3732 014576 000440      BR    NDONE
3733
3734 014600      NERR6:      ;DATA FAILURE.
    
```





```

3791
3792 014730 172450          02:   LDD   @-(R0),ACO      ;TRAP TO 4 MAY OCCUR.
3793 014732 005203          03:   INC   R3              ;TEST INSTRUCTION
3794 014734 005203          04:   INC   R3
3795
3796 014736 012701 015332    MOV   #ODAT00,R1
3797 014742 174011          STD   ACO,(R1)          ;GET THE DATA
3798
3799 014744 020027 015350    CMP   R0,#OPAT20       ;WAS R0 DECREMENTED
3800 014750 001436          BEQ   012              ;BY 2?
3801
3802 014752 020027 015362    05:   CMP   R0,#OPAT21+10 ;FSRC MODE 2
3803 014756 001001          BNE   06
3804 014760 000505          BR    OERR1
3805
3806 014762 020027 015342    06:   CMP   R0,#OPAT21-10 ;FSRC MODE 4?
3807 014766 001001          BNE   07
3808 014770 000517          BR    OERR2
3809
3810 014772 020027 015352    07:   CMP   R0,#OPAT21
3811
3812 014776 012702 015334    MOV   #ODAT01,R2       ;FSRC MODE 0?
3813 015002 012703 000004    MOV   #4,R3
3814 015006 022227 177777    08:   CMP   (R2)+,#-1
3815 015012 001002          BNE   09
3816 015014 077304          SOB   R3,08
3817 015016 000510          BR    OERR3
3818
3819 015020 012702 015332    09:   MOV   #ODAT00,R2       ;FSRC MODE 1?
3820 015024 012703 015352    MOV   #OPAT21,R3
3821 015030 012704 000004    MOV   #4,R4
3822 015034 022223          10:   CMP   (R2)+,(R3)+
3823 015036 001002          BNE   011
3824 015040 077403          SOB   R4,010
3825 015042 000502          BR    OERR4
3826
3827 015044 000505          11:   BR    OERR5
3828
3829 015046 012702 015332    12:   MOV   #ODAT00,R2       ;DATA CORRECT?
3830 015052 012703 015374    MOV   #ODAT10,R3
3831 015056 012704 000004    MOV   #4,R4
3832 015062 022223          13:   CMP   (R2)+,(R3)+
3833 015064 001002          BNE   014
3834 015066 077403          SOB   R4,013
3835 015070 000545          BR    ODONE
3836
3837 015072 000504          14:   BR    OERR6
3838
3839          ;IF AN ODD ADDRESS TRAP OCCURS COME
3840          ;HERE TO SEE IF THE FAILURE WAS IN THE
3841          ;FSRC FLOWS:
3842
3843 015074 022716 014734    OERR0: CMP   #04,(SP)       ;FSRC MODE 6 OF 7?
3844 015100 001412          BEQ   OERR10
3845 015102 022716 014732    CMP   #03,(SP)
3846 015106 001402          BEQ   1$
    
```

```

3847 015110 000137 062566          JMP      @#CPSPUR
3848 015114 020027 015354          1$:    CMP      R0,#OPAT21+2      ;FSRC MODE 3?
3849 015120 001425                    BEQ     OERR1
3850 015122 000137 062566          JMP      @#CPSPUR
3851
3852 015126                    OERR10:                ;WENT TO FSRC
3853 015126 011637 001236          MOV     (SP),@#STMP2      ;MODE 6 OR 7
3854 015132 022626                    CMP     (SP)+,(SP)+
3855 015134 104001          1$:    ERROR    1
3856 015136 000522                    BR     ODONE
3857
3858 015140 011637 001240          OERR11: MOV     (SP),@#STMP3      ;WENT TO FSRC MODE
3859 015144 022626                    CMP     (SP)+,(SP)+      ;3
3860 015146 012737 000627 001244          MOV     #627,@#STMP5
3861 015154 012737 000325 001250          MOV     #325,@#STMP7
3862 015162 012737 000323 001246          MOV     #323,@#STMP6
3863 015170 104001          1$:    ERROR    1
3864 015172 000504                    BR     ODONE
3865
3866 015174 012737 000322 001246          OERR1:  MOV     #322,@#STMP6      ;FSRC MODE2
3867 015202 012737 000627 001242          OERR20: MOV     #627,@#STMP4
3868 015210 012737 000325 001250          MOV     #325,@#STMP7
3869 015216 012737 014730 001236          MOV     #02,@#STMP2
3870 015224 104001          1$:    ERROR    1
3871 015226 000466                    BR     ODONE
3872 015230 012737 000324 001246          OERR2:  MOV     #324,@#STMP6      ;FSRC MODE 4
3873 015236 000761                    BR     OERR20
3874 015240 012737 000320 001246          OERR3:  MOV     #320,@#STMP6      ;FSRC MODE 0
3875 015246 000755                    BR     OERR20
3876 015250 012737 000321 001246          OERR4:  MOV     #321,@#STMP6      ;FSRC MODE 1
3877 015256 000751                    BR     OERR20
3878
3879 015260 010037 001240          OERR5:  MOV     R0,@#STMP3      ;R0 NOT DECREMENTED
3880 015264 012737 015350 001242          MOV     #OPAT20,@#STMP4    ;PROPERLY
3881 015272 012737 014734 001236          MOV     #04,@#STMP2
3882 015300 104001          1$:    ERROR    1
3883 015302 000440                    BR     ODONE
3884
3885 015304                    OERR6:                ;DATA FAILURE
3886 015304 012737 014730 001236          MOV     #02,@#STMP2
3887 015312 012737 015332 001240          MOV     #ODAT00,@#STMP3
3888 015320 012737 015374 001242          MOV     #ODAT10,@#STMP4
3889 015326 104001          1$:    ERROR    1
3890 015330 000425                    BR     ODONE
3891
3892 015332 000000          ODAT00: .WORD    0
3893 015334 000000          ODAT01:                0
3894 015336 000000          ODAT02:                0
3895 015340 000000          ODAT03:                0
3896
3897 015342 052525 052525 052525          ODAT00: .WORD    52525,52525,52525
3898 015350 015374          OPAT20: .WORD    ODAT10
3899 015352 070707          OPAT21:                070707
3900 015354 070707          OPAT22:                070707
3901 015356 070707          OPAT23:                070707
3902 015360 070707          OPAT24:                070707
    
```

```

3903 015362 000001          .WORD 1
3904 015364 177777          OPAT10: .WORD -1
3905 015366 177777          OPAT11: -1
3906 015370 177777          OPAT12: -1
3907 015372 177777          OPAT13: -1
3908
3909 015374 073567          ODAT10: .WORD 73567
3910 015376 004210          ODAT11: 004210
3911 015400 114631          ODAT12: 114631
3912 015402 125252          ODAT13: 125252
3913
3914 015404
3915 015404 104413          ODONE:
3916                                     RSETUP
3917                                     ;GO INITIALIZE THE FPS AND STACK; AND
3918                                     ;SEE IF THE USER HAS EXPRESSED
3919                                     ;THE DESIRE TO CHANGE THE SOFTWARE
3920                                     ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
3921                                     ;THE USER TYPED CONTROL G?).
3922
3923 ::*****
3924 *TEST 22          FSRC MODE 6 TEST
3925 *
3926 * THIS IS A TEST OF FSRC MODE 6, INDEX MODE
3927 *
3928 ::*****
3929 TST22: SCOPE
3930
3931 P1:
3932          SETD          ;SET FD MODE
3933          MOV          #PPAT10,R0
3934          LDD          (R0),AC0          ;LOAD A DEFAULT PATTERN
3935          ;INTO AC0
3936          MOV          #PERR0,@#ERRVECT          ;IF THE (BUT FSRC) FORQ
3937          ;FAILS AN ODD ADDRESS TRAP
3938          MOV          #PDAT10-241,R0          ;COULD OCCUR.
3939
3940 P2:
3941          LDD          241(R0),AC0
3942          P3=P2+2
3943
3944 P4:
3945          MOV          #PDAT00,R1
3946          STD          AC0,(R1)          ;GET THE DATA
3947          MOV          #4,R3
3948          MOV          #PDAT10,R2
3949          MOV          #PDAT00,R1
3950          CMP          (R2)+,(R1)+          ;CHECK THE DATA
3951          BNE          P6
3952          SOB          R3,P5
3953          CMP          #PDAT10-241,R0          ;RO CORRECT?
3954          BEQ          1$
3955          BR          PERR21
3956          JMP          @#PDONE
3957
3958 P6:
3959          MOV          #PDAT00,R1
3960          MOV          #4,R3
3961          CMP          #-1,(R1)+          ;WAS IT FSRC MODE 0?
    
```

```

3959 015516 001401          BEQ      P8
3960 015520 000512          BR       PERR1
3961 015522 077305          P8:     SOB      R3,P7
3962 015524 000523          BR       PERR2
3963          ;TRAP TO HERE ON AN ODD ADDRESS
3964 015526 021627 015434      PERR0:  CMP      (SP),#P3
3965 015532 001411          BEQ      PERR11
3966 015534 021627 015436      CMP      (SP),#P4          ;WAS IT FSRC MODE 7?
3967 015540 001402          BEQ      PERR10
3968 015542 000137 062566      JMP      @#CPSPUR
3969
3970 015546 012737 000327 001246      PERR10: MOV      #327,@#STMP6
3971 015554 000443          BR       PERR17
3972 015556 022700 015577      PERR11: CMP      #PDAT10-241,R0 ;WAS IT FSRC MODE 1
3973 015562 001004          BNE      PERR12
3974 015564 012737 000321 001246      MOV      #321,@#STMP6
3975 015572 000434          BR       PERR17
3976 015574 022700 015607      PERR12: CMP      #PDAT10-241+10,R0 ;WAS IT FSRC MODE 2
3977 015600 001004          BNE      PERR13
3978 015602 012737 000322 001246      MOV      #322,@#STMP6
3979 015610 000425          BR       PERR17
3980 015612 022700 015601      PERR13: CMP      #PDAT10-241+2,R0 ;WAS IT FSRC MODE 3
3981 015616 001004          BNE      PERR14
3982 015620 012737 000323 001246      MOV      #323,@#STMP6
3983 015626 000416          BR       PERR17
3984 015630 022700 015567      PERR14: CMP      #PDAT10-241-10,R0 ;WAS IT FSRC MODE 4
3985 015634 001004          BNE      PERR15
3986 015636 012737 000324 001246      MOV      #324,@#STMP6
3987 015644 000407          BR       PERR17
3988 015646 022700 015575      PERR15: CMP      #PDAT10-241-2,R0 ;WAS IT FSRC MODE 5
3989 015652 001401          BEQ      PERR16
3990 015654 000416          BR       PERR20
3991 015656 012737 000325 001246      PERR16: MOV      #325,@#STMP6
3992
3993 015664 012737 000627 001244      PERR17: MOV      #627,@#STMP5 ;REPORT FSRC
3994 015672 012737 000326 001250      MOV      #326,@#STMP7 ;FLOWS FAILURE.
3995 015700 011637 001236      MOV      (SP),@#STMP2
3996 015704 022626          CMP      (SP)+,(SP)+
3997 015706 104001          1$:     ERROR    1
3998 015710 000463          BR       PDONE
3999
4000 015712 011637 001236      PERR20: MOV      (SP),@#STMP2 ;REPORT R0 AFFECTED
4001 015716 022626          CMP      (SP)+,(SP)+
4002 015720 000403          BR       PERR22
4003 015722 012737 015432 001236      PERR21: MOV      #P2,@#STMP2
4004 015730          PERR22:
4005 015730 010037 001240          MOV      R0,@#STMP3
4006 015734 012737 015577 001242      MOV      #PDAT10-241,@#STMP4
4007 015742 104001          1$:     ERROR    1
4008 015744 000445          BR       PDONE
4009
4010 015746          PERR1:          ;DATA FAILURE.
4011 015746 012737 015432 001236      MOV      #P2,@#STMP2
4012 015754 012737 016040 001240      MOV      #PDAT10,@#STMP3
4013 015762 012737 016050 001242      MOV      #PDAT00,@#STMP4
4014 015770 104001          1$:     ERROR    1
    
```

```

4015 015772 000432          BR      PDONE
4016
4017 015774                PERR2:          ;FSRC FAILURE TO
4018 015774 012737 015432 001236      MOV      #P2,@#STMP2      ;MODE 0
4019 016002 012737 000627 001244      MOV      #627,@#STMP5
4020 016010 012737 000326 001250      MOV      #326,@#STMP7
4021 016016 012737 000320 001246      MOV      #320,@#STMP6
4022 016024 104001          1$:      ERROR      1
4023 016026 000414          BR      PDONE
4024
4025 016030 177777          PPAT10: .WORD  -1
4026 016032 177777          PPAT11:          -1
4027 016034 177777          PPAT12:          -1
4028 016036 177777          PPAT13:          -1
4029
4030 016040 010421          PDAT10: .WORD  010421
4031 016042 031463          PDAT11:          031463
4032 016044 052525          PDAT12:          052525
4033 016046 073567          PDAT13:          073567
4034
4035 016050 000000          PDAT00: .WORD  0
4036 016052 000000          PDAT01:          0
4037 016054 000000          PDAT02:          0
4038 016056 000000          PDAT03:          0
4039
4040 016060          PDONE:
4041 016060 104413          RSETUP          ;GO INITIALIZE THE FPS AND STACK; AND
4042                                     ;SEE IF THE USER HAS EXPRESSED
4043                                     ;THE DESIRE TO CHANGE THE SOFTWARE
4044                                     ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
4045                                     ;THE USER TYPED CONTROL G?).
4046
4047
4048                                     ;*****
4049                                     ;*TEST 23      FSRC MODE 7 TEST
4050                                     ;*
4051                                     ;* THIS IS A TEST OF FSRC MODE 7, INDEX
4052                                     ;* DEFERRED MODE.
4053                                     ;*
4054                                     ;*****
4055 016062 000004          TST23: SCOPE
4056
4057 016064          Q1:
4058 016064 170011          SETD
4059
4060 016066 012700 016520          MOV      #QPAT10,R0
4061 016072 172410          LDD      (R0),AC0      ;LOAD A DEFAULT
4062                                     ;PATTERN INTO ACO
4063 016074 012737 016226 000004          MOV      #QERR0,@#ERRVECT      ;IF THE (BUT FSRC)
4064                                     ;FORK FAILS AN
4065                                     ;ODD ADR TRAP COULD
4066                                     ;OCCUR
4067 016102 012700 016267          MOV      #QPAT20-241,R0
4068
4069 016106 172470 000241          Q2:      LDD      @241(R0),AC0
4070 016110          Q3=Q2+2
    
```

```

4071
4072 016112 012701 016540      Q4:  MOV  #QDAT00,R1
4073 016116 174011              STD  AC0,(R1)      ;GET THE DATA
4074
4075 016120 012703 000004      MOV  #4,R3
4076 016124 012704 016540      MOV  #QDAT00,R4
4077 016130 012705 016550      MOV  #QDAT10,R5
4078 016134 022425      Q5:  CMP  (R4)+,(R5)+  ;CHECK THE DATA
4079 016136 001007      BNE  Q6
4080 016140 077303      SOB  R3,Q5
4081
4082 016142 022700 016267      CMP  #QPAT20-241,R0 ;CHECK R0.
4083 016146 001401      BEQ  1$
4084 016150 000514      BR   QERR21
4085 016152 000137 016560      1$:  JMP  @#QDONE
4086
4087 016156 012701 016540      Q6:  MOV  #QDAT00,R1
4088 016162 012703 000004      MOV  #4,R3
4089 016166 022721 177777      Q7:  CMP  #-1,(R1)+    ;WAS IT FSRC MODE 0?
4090 016172 001002      BNE  Q8
4091 016174 077304      SOB  R3,Q7
4092 016176 000513      BR   QERR2
4093
4094 016200 012701 016530      Q8:  MOV  #QPAT20,R1
4095 016204 012702 016540      MOV  #QDAT00,R2
4096 016210 012703 000004      MOV  #4,R3
4097 016214 022122      Q9:  CMP  (R1)+,(R2)+  ;WAS IT FSRC 6
4098 016216 001401      BEQ  Q10           ;OR DATA FAILURE
4099 016220 000524      BR   QERR1
4100 016222 077304      Q10: SOB  R3,Q9
4101 016224 000504      BR   QERR3
4102
4103      ;TRAP TO HERE ON AN ODD ADR FAILURE
4104
4105 016226 021627 015434      QERR0: CMP  (SP),#P3
4106 016232 000137 062566      JMP  @#CPSPUR
4107
4108 016236 022700 016267      QERR11: CMP  #QPAT20-241,R0 ;WAS IT FSRC
4109 016242 001004      BNE  QERR12        ;MODE 1?
4110 016244 012737 000321 001246      MOV  #321,@#$TMP6
4111 016252 000434      BR   QERR17
4112 016254 022700 016277      QERR12: CMP  #QPAT20-241+10,R0 ;WAS IT FSRC
4113 016260 001004      BNE  QERR13        ;MODE 2?
4114 016262 012737 000322 001246      MOV  #322,@#$TMP6
4115 016270 000425      BR   QERR17
4116 016272 022700 016271      QERR13: CMP  #QPAT20-241+2,R0 ;WAS IT FSRC
4117 016276 001004      BNE  QERR14        ;MODE 3?
4118 016300 012737 000323 001246      MOV  #323,@#$TMP6
4119 016306 000416      BR   QERR17
4120 016310 022700 016257      QERR14: CMP  #QPAT20-241-10,R0 ;WAS IT FSRC
4121 016314 001004      BNE  QERR15        ;MODE 4
4122 016316 012737 000324 001246      MOV  #324,@#$TMP6
4123 016324 000407      BR   QERR17
4124
4125 016326 022700 016265      QERR15: CMP  #QPAT20-241-2,R0 ;WAS IT FSRC
4126 016332 001401      BEQ  QERR16        ;MODE 5
    
```

```

4127 016334 000416 BR QERR20
4128
4129 016336 012737 000325 001246 QERR16: MOV #325,@#STMP6
4130
4131 016344 012737 000627 001244 QERR17: MOV #627,@#STMP5 ;REPORT FSRC FAILURE
4132 016352 012737 000327 001250 MOV #327,@#STMP7
4133 016360 011637 001236 MOV (SP),@#STMP2
4134 016364 022626 CMP (SP)+,(SP)+
4135 016366 104001 1$: ERROR 1
4136 016370 000473 BR QDONE
4137
4138 016372 011637 001236 QERR20: MOV (SP),@#STMP2 ;REPORT R0 AFFECTED.
4139 016376 022626 CMP (SP)+,(SP)+
4140 016400 000403 BR QERR22
4141 016402 012737 016106 001236 QERR21: MOV #Q2,@#STMP2
4142 016410 QERR22:
4143 016410 010037 001240 MOV R0,@#STMP3
4144 016414 012737 016267 001242 MOV #QPAT20-241,@#STMP4
4145 016422 104001 1$: ERROR 1
4146 016424 000455 BR QDONE
4147
4148 016426 012737 000320 001246 QERR2: MOV #320,@#STMP6 ;WENT TO FSRC
4149 016434 000403 BR QERR4 ;MODE 0
4150 016436 012737 000326 001246 QERR3: MOV #326,@#STMP6 ;WENT TO FSRC
4151 ;MODE 6
4152 016444 012737 000627 001244 QERR4: MOV #627,@#STMP5
4153 016452 012737 000327 001250 MOV #327,@#STMP7
4154 016460 012737 016106 001236 MOV #Q2,@#STMP2
4155 016466 104001 1$: ERROR 1
4156 016470 000433 BR QDONE
4157
4158 016472 QERR1: ;DATA FAILURE
4159 016472 012737 016106 001236 MOV #Q2,@#STMP2
4160 016500 012737 016550 001240 MOV #QDAT10,@#STMP3
4161 016506 012737 016540 001242 MOV #QDAT00,@#STMP4
4162 016514 104001 1$: ERROR 1
4163 016516 000420 BR QDONE
4164
4165 016520 177777 QPAT10: .WORD -1
4166 016522 177777 QPAT11: -1
4167 016524 177777 QPAT12: -1
4168 016526 177777 QPAT13: -1
4169
4170 016530 016550 QPAT20: .WORD QDAT10
4171 016532 052525 QPAT21: 52525
4172 016534 052525 QPAT22: 52525
4173 016536 052525 QPAT23: 52525
4174
4175 016540 000000 QDAT00: .WORD 0
4176 016542 000000 QDAT01: 0
4177 016544 000000 QDAT02: 0
4178 016546 000000 QDAT03: 0
4179
4180 016550 073567 QDAT10: .WORD 073567
4181 016552 052525 QDAT11: .WORD 052525
4182 016554 031463 QDAT12: .WORD 031463
    
```



4183 016556 010421  
 4184  
 4185 016560  
 4186 016560 104413  
 4187  
 4188  
 4189  
 4190  
 4191  
 4192  
 4193  
 4194  
 4195  
 4196  
 4197  
 4198  
 4199  
 4200  
 4201  
 4202  
 4203  
 4204  
 4205  
 4206  
 4207 016562 000004  
 4208 016564 005037 017662  
 4209 016570 012700 017612  
 4210 016574 012701 000004  
 4211 016600 012720 177777  
 4212 016604 077103  
 4213  
 4214 016606 012737 000033 017664  
 4215 016614 012737 000023 017666  
 4216 016622 012737 017342 000244  
 4217 016630  
 4218 016630 104414  
 4219 016632 012700 000200  
 4220 016636 170100  
 4221 016640 012700 017612  
 4222 016644 172410  
 4223 016646 013737 017664 017670  
 4224 016654 012737 000001 017672  
 4225 016662 012737 000254 017674  
 4226  
 4227 016670 012700 017622  
 4228 016674 172410  
 4229 016676 010037 001252  
 4230 016702 012737 016674 001236  
 4231  
 4232 016710 012704 000204  
 4233 016714 170205  
 4234  
 4235 016716 020405  
 4236 016720 001402  
 4237 016722 000137 017366  
 4238

```

QDATI3: .WORD 010421
QDONE:
RSETUP                                ;GO INITIALIZE THE FPS AND STACK; AND
                                        ;SEE IF THE USER HAS EXPRESSED
                                        ;THE DESIRE TO CHANGE THE SOFTWARE
                                        ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                        ;THE USER TYPED CONTROL G?).

:*****
:*TEST 24 (BUT EZBT Y8),(BUT ENBT) AND (BUT FIUV) TEST
:*
:* LOAD INSTRUCTION FLOWS.
:* EACH OF THE PATTERNS:
:*
:* 0
:* +NUM
:* -NUM
:* -0
:* IS LOADED TWICE, ONCE WITH AC>0 THEN
:* WITH AC=0. AFTER EACH LOAD THE FPS IS
:* CHECK TO INSURE THAT CONTROL WAS PASSED
:* THROUGH WITH THE FORKS PROPERLY.
:*
:*****
TST24: SCOPE
CLR @#UFLAG
MOV #UPAT00,R0 ;SET UP AC#0 DATA.
MOV #4,R1
U0: MOV #-1,(R0)+
SOB R1,U0
MOV #033,@#UTMP1
MOV #023,@#UTMP2
MOV #UERR0,@#FPVECT ;IN CASE (BUT FIUV FAILS)
U1: LPERP ;SET UP THE LOOP ON ERROR ADDRESS.
MOV #200,R0
LDFPS R0
MOV #UPAT00,R0 ;LOAD AC0
LDD (R0),AC0
MOV @#UTMP1,@#UROM1
MOV #001,@#UROM2
MOV #254,@#UROM3
U2: MOV #UPAT10,R0 ;LOAD 0 INTO AC0
LDD (R0),AC0
MOV R0,@#STMP10
MOV #U2,@#STMP2
MOV #204,R4 ;SEE IF FPS IS CORRECT
STFPS R5
CMP R4,R5
BEQ U3
JMP @#UERR1
    
```



```

4295 017210 170205          STFPS  R5
4296 017212 020405          CMP    R4,R5
4297 017214 001402          BEQ    U12
4298 017216 000137 017366    JMP    @#UERR1
4299 017222 005737 017662    U12:  TST    @#UFLAG ;SEE IF ALL THE PATTERNS
4300 017226 001021          BNE    U14                ;HAVE BEEN TEST WITH
4301          ;BOTH AC NOT EQUAL TO 0 AND AC=0
4302 017230 012700 017612    MOV    #UPAT00,R0        ;IF NOT GO BACK AND
4303 017234 012701 000004    MOV    #4,R1             ;CHECK THEM WITH AC=0
4304 017240 005020          U13:  CLR    (R0)+
4305 017242 077102          SOB    R1,U13
4306 017244 012737 177777 017662    MOV    #-1,@#UFLAG
4307 017252 012737 000233 017664    MOV    #233,@#UTMP1
4308 017260 012737 000223 017666    MOV    #223,@#UTMP2
4309 017266 000137 016630    JMP    @#U1
4310 017272
4311 017272 104414          U14:  LPERR
4312          ;NOW SEE IF A TRAP CAN BE FORCED BY SETTING FIUV AND LOADING -0
4313 017274 012737 017536 000244    MOV    #UERR3,@#FPVECT
4314 017302 012700 004200          MOV    #4200,R0         ;SET FD AND FIUV
4315 017306 170100          LDFPS  R0
4316 017310 012700 017612    MOV    #UPAT00,R0        ;SET UP AC0
4317 017314 172410          LDD    (R0),AC0
4318 017316 012700 017652    MOV    #UPAT40,R0        ;LOAD -0
4319 017322 172410          U15:  LDD    (R0),AC0        ;SHOULD TRAP TO 244
4320 017324 170000          U16:  CFCC
4321 017326 000240          NOP
4322 017330 012737 017322 001236    MOV    #U15,@#STMP2     ;REPORT ERROR.
4323          ;DIDN'T TRAP
4324 017336 104001          1$:   ERROR 1           ;(BUT FIUV) FAILED.
4325 017340 000556          BR    UDONE
4326
4327          ;TRAPPED TO 244. DID (BUT FIUV) FAIL?
4328 017342 021627 017170    UERR0: CMP    (SP),#U11
4329 017346 001402          BEQ    1$
4330 017350 000137 062534    JMP    @#FPSPUR
4331 017354 011637 001236    1$:   MOV    (SP),@#STMP2
4332 017360 022626          CMP    (SP)+,(SP)+
4333 017362 104001          2$:   ERROR 1
4334 017364 000544          BR    UDONE
4335
4336          ;COME HERE TO ANALYZE FPS ERRORS
4337
4338 017366 032705 000004    UERR1: BIT    #4,R5
4339 017372 001432          BEQ    UERR20
4340 017374 012737 000443 001244    UERR10: MOV   #443,@#STMP5
4341 017402 013703 017674          MOV    @#UROM3,R3
4342 017406 010337 001250          MOV    R3,@#STMP7
4343 017412 032703 000200          BIT    #200,R3
4344 017416 001403          BEQ    1$
4345 017420 042703 000200          BIC    #200,R3
4346 017424 000402          BR    2$
4347 017426 052703 000200          1$:   BIS    #200,R3
4348 017432 C10337 001246          2$:   MOV    R3,@#STMP6
4349 017436 010537 001240    UERR11: MOV   R5,@#STMP3
4350 017442 010437 001242          MOV    R4,@#STMP4
    
```

```

4351 017446 104001      1$:      ERROR      1
4352 017450 000512      BR          UDONE
4353 017452 032705 000004      UERR2:     BIT          #4,R5
4354 017456 001746      BEQ         UERR10
4355 017460 013737 017670 001244      UERR20:    MOV          @#UROM1,@#STMP5
4356 017466 013703 017672      MOV          @#UROM2,R3
4357 017472 010337 001250      MOV          R3,@#STMP7
4358 017476 032703 000400      BIT          #400,R3
4359 017502 001403      BEQ         1$
4360 017504 042703 000400      BIC          #400,R3
4361 017510 000402      BR          2$
4362 017512 052703 000400      1$:      BIS          #400,R3
4363 017516 010337 001246      2$:      MOV          R3,@#STMP6
4364 017522 010537 001240      UERR21:    MOV          R5,@#STMP3
4365 017526 010437 001242      MOV          R4,@#STMP4
4366 017532 104001      1$:      ERROR      1
4367 017534 000460      BR          UDONE
4368
4369 .
4370 017536 021627 017324      ;INTERRUPT HERE WHEN FIUV SET AND ATTEMPTED TO LOAD-0
4371 017542 001402      UERR3:     CMP          (SP),#U16
4372 017544 000137 062534      BEQ         1$
4373 017550 022626      JMP          @#FPSPUR
4374 017552 005000      1$:      CMP          (SP)+,(SP)+
4375 017554 170300      CLR          R0
4376 017556 022700 000014      STST        R0          ;GET FEC.
4377 017562 001001      CMP          #14,R0      ;CORRECT
4378 017564 000444      BNE         UERR4
4379 017566 012737 017322 001236      UERR4:     BR          UDONE
4380 017574 012737 000012 001242      MOV          #U15,@#STMP2
4381 017602 010037 001240      MOV          #12,@#STMP4
4382 017606 104001      MOV          R0,@#STMP3
4383 017610 000432      1$:      ERROR      1
4384 017612 000000      BR          UDONE
4385 017614 000000      UPAT00:    .WORD      0
4386 017616 000000      UPAT01:    0
4387 017620 000000      UPAT02:    0
4388
4389 017622 000000      UPAT03:    0
4390 017624 000000      UPAT10:    .WORD      0          ;0
4391 017626 000000      UPAT11:    0
4392 017630 000000      UPAT12:    0
4393
4394 017632 010421      UPAT13:    0
4395 017634 114631      UPAT20:    .WORD      010421      ;POS NUM
4396 017636 125252      UPAT21:    114631
4397 017640 177777      UPAT22:    125252
4398
4399 017642 114631      UPAT23:    177777
4400 017644 135673      UPAT30:    114631      ;NEG NUM
4401 017646 146314      UPAT31:    135673
4402 017650 167356      UPAT32:    146314
4403
4404 017652 100000      UPAT33:    167356
4405 017654 000000      UPAT40:    100000      ;NEG ZERO
4406 017656 000000      UPAT41:    0
4406 017656 000000      UPAT42:    0
    
```

```

4407 017660 000000      UPAT43:      0
4408
4409 017662 000000      UFLAG: .WORD 0
4410 017664 000000      UTMP1:      0
4411 017666 000000      UTMP2:      0
4412 017670 000000      UROM1:      0
4413 017672 000000      UROM2:      0
4414 017674 000000      UROM3:      0
4415 017676
4416
4417
4418
4419
4420
4421
4422
4423
4424
4425
4426 017676 000004      TST25: SCOPE
4427 017700      W1:
4428 017700 104414      LPERR
4429 017702 012700 000200      MOV #200,R0 ;SET UP THE LOOP ON ERROR ADDRESS.
4430 017706 170100      LDFPS R0 ;SET DOUBLE MODE
4431 017710 012700 020364      MOV #WPAT00,R0 ;LOAD AC0=:
4432 017714 172410      LDD (R0),AC0
4433 017716 012737 017730 001236      MOV #W2,@#STMP2 ;SAVE FP INSTRUCTION
4434 017724 012700 020364      MOV #WPAT00,R0
4435 017730 172010      W2: ADDD (R0),AC0 ;TEST INSTRUCTION. ADD ITSELF
4436 017732 170205      STFPS R5 ;GET FPS
4437 017734 170011      SETD ;SET DOUBLE MODE
4438 017736 012700 020364      MOV #WPAT00,R0
4439 017742 174010      STD AC0,(R0) ;GET THE RESULT
4440 017744 012701 020364      MOV #WPAT00,R1
4441 017750 012702 000004      MOV #4,R2
4442 017754 022021      W3: CMP (R0)+,(R1)+ ;IS RESULT CORRECT
4443 017756 001403      BEQ W4 ;NO
4444
4445 017760 104002      1$: ERROR 2 ;NO
4446 017762 000137 020404      JMP @#WDONE
4447 017766 077206      W4: SOB R2,W3
4448 017770 022705 000204      CMP #204,R5 ;IS FPS CORRECT
4449 017774 001410      BEQ W5 ;NO
4450
4451 017776 012737 000204 001242      MOV #204,@#STMP4
4452 020004 010537 001240      MOV R5,@#STMP3
4453 020010 104001      1$: ERROR 1
4454 020012 000137 020404      JMP @#WDONE
4455 020016
4456 020016 104414      W5: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
4457 020020 012700 000200      MOV #200,R0
4458 020024 170100      LDFPS R0 ;SET DOUBLE MODE
4459 020026 012700 020364      MOV #WPAT00,R0 ;LOAD AC0=0
4460 020032 172410      LDD (R0),AC0
4461 020034 012737 020052 001236      MOV #W6,@#STMP2
4462 020042 005000      CLR R0
    
```

```

4463 020044 170100          LDFPS R0          ;GO TO FLOATING MODE
4464 020046 012700 020364    MOV #WPAT00,R0
4465 020052 172010          ADDF (R0),AC0     ;TEST INSTRUCTION
4466 020054 170205          STFPS R5         ;GET FPS
4467 020056 170011          SETD            ;RESET TO DOUBLE MODE
4468 020060 012700 020364    MOV #WPAT00,R0
4469 020064 174010          STD AC0,(R0)     ;GET THE RESULT
4470 020066 012701 020364    MOV #WPAT00,R1
4471 020072 012702 000004    MOV #4,R2
4472 020076 022021          W7:  CMP (R0)+,(R1)+ ;WAS THE RESULT
4473 020100 001402          BEQ W10         ;NO. REPORT FAILURE.
4474 020102 104002          1$:  ERROR 2
4475 020104 000537          BR WDONE
4476 020106 077205          W10: SOB R2,W7
4477 020110 022705 000004    CMP #4,R5        ;WAS FPS CORRECT
4478 020114 001407          BEQ W11         ;INCORRECT FPS.
4479
4480 020116 012737 000004 001242    MOV #4,@#STMP4
4481 020124 010537 001240    MOV R5,@#STMP3
4482 020130 104002          1$:  ERROR 2
4483 020132 000524          BR WDONE
4484 020134
4485 020134 104414          W11: LPERR        ;SET UP THE LOOP ON ERROR ADDRESS.
4486 020136 012700 000200    MOV #200,R0
4487 020142 170100          LDFPS R0        ;SET DOUBLE MODE
4488 020144 012700 020364    MOV #WPAT00,R0 ;LOAD AC0=0
4489 020150 172410          LDD (R0),AC0
4490 020152 012737 020164 001236    MOV #W12,@#STMP2
4491 020160 012700 020364    MOV #WPAT00,R0
4492 020164 173010          W12: SUBD (R0),AC0 ;TEST INSTRUCTION
4493 020166 170205          STFPS R5        ;GET FPS
4494 020170 170011          SETD            ;SET DOUBLE MODE
4495 020172 012700 020364    MOV #WPAT00,R0
4496 020176 174010          STD AC0,(R0)   ;GET THE RESULT
4497 020200 012701 020364    MOV #WPAT00,R1
4498 020204 012702 000004    MOV #4,R2
4499 020210 022021          W13: CMP (R0)+,(R1)+ ;IS RESULT CORRECT?
4500 020212 001402          BEQ W14        ;NO.
4501
4502 020214 104002          1$:  ERROR 2
4503 020216 000472          BR WDONE
4504 020220 077205          W14: SOB R2,W13
4505 020222 022705 000204    CMP #204,R5     ;IS FPS CORRECT?
4506 020226 001407          BEQ W15        ;NO.
4507
4508 020230 012737 000204 001242    MOV #204,@#STMP4
4509 020236 010537 001240    MOV R5,@#STMP3
4510 020242 104001          1$:  ERROR 1
4511 020244 000457          BR WDONE
4512 020246
4513 020246 104414          W15: LPERR        ;SET UP THE LOOP ON ERROR ADDRESS.
4514 020250 012700 000200    MOV #200,R0
4515 020254 170100          LDFPS R0        ;SET DOUBLE MODE
4516 020256 012700 020364    MOV #WPAT00,R0 ;LOAD AC0 0
4517 020262 172410          LDD (R0),AC0
4518 020264 012737 020302 001236    MOV #W16,@#STMP2
    
```

```

4519 020272 005000 CLR R0
4520 020274 170100 LDFPS R0 ;ENTER FLOATING MODE.
4521 020276 012700 020364 MOV #WPAT00,R0
4522 020302 173010 W16: SUBF (R0),AC0 ;TEST INSTRUCTION.
4523 020304 170205 STFPS R5 ;GET FPS
4524 020306 170011 SETD ;RESET TO DOUBLE MODE
4525 020310 012700 020364 MOV #WPAT00,R0 ;GET THE RESULT.
4526 020314 174010 STD AC0,(R0)
4527 020316 012701 020364 MOV #WPAT00,R1
4528 020322 012702 000004 MOV #4,R2
4529 020326 022027 W17: CMP (R0)+,(R1)+ ;IS RESULT CORRECT?
4530 020330 001402 BEQ W20 ;NO.
4531 ;NO.
4532 020332 104002 1$: ERROR 2
4533 020334 000423 BR WDONE
4534 020336 077205 W20: SOB R2,W17
4535 020340 022705 000004 CMP #4,R5 ;IS FPS CORRECT?
4536 020344 001417 BEQ WDONE ;NO
4537 ;NO
4538 020346 012737 000004 001242 MOV #4,@#STMP4
4539 020354 010537 001240 MOV R5,@#STMP3
4540 020360 104001 1$: ERROR 1
4541 020362 000410 BR WDONE
4542
4543 020364 000000 WPAT00: .WORD 0
4544 020366 000000 WPAT01: 0
4545 020370 000000 WPAT02: 0
4546 020372 000000 WPAT03: 0
4547
4548 020374 000000 WDAPO0: .WORD 0
4549 020376 000000 WDAT01: 0
4550 020400 000000 WDAT02: 0
4551 020402 000000 WDAT03: 0
4552
4553 020404 WDONE:
4554 020404 104413 RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
4555 ;SEE IF THE USER HAS EXPRESSED
4556 ;THE DESIRE TO CHANGE THE SOFTWARE
4557 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
4558 ;THE USER TYPED CONTROL G?).
4559
4560
4561
4562 ;*****
4563 ;*TEST 26 ADDD AND SUB WITH FSRC=0
4564 ;*
4565 ;* THIS IS A TEST OF ADD AND SUB WITH FSRC=0.
4566 ;*
4567 ;*****
4568 020406 000004 TST26: SCOPE
4569 020410 X1:
4570 020410 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
4571 020412 012700 000200 MOV #200,R0
4572 020416 170100 LDFPS R0 ;SET DOUBLE MODE
4573 020420 012700 021157 MOV #XPAT00,R0 ;SET AC0 TO POSITIVE
4574 020424 010037 021136 MOV R0,@#XTMP ;NUMBER #0

```

```

4575 020430 172410          LDD      (R0),AC0
4576 020432 012737 020444 001236  MOV      #X2,@#STMP2
4577 020440 012700 021160          MOV      #XPAT10,R0          :FSRC=0
4578 020444 172010          ADDD     (R0),AC0          :TEST INSTRUCTION
4579 020446 170205          STFPS   R5
4580 020450 170011          SETD
4581 020452 012700 021140          MOV      #XDAT00,R0          :GET RESULT.
4582 020456 174010          STD     AC0,(R0)
4583 020460 012701 021150          MOV      #XPAT00,R1
4584 020464 012702 000004          MOV      #4,R2
4585 020470 022021          X3:     CMP      (R0)+,(R1)+          :IS RESULT CORRECT?
4586 020472 001401          BEQ     X4
4587 020474 000553          BR      XERR1
4588 020476 077204          X4:     SOB     R2,X3
4589 020500 012704 000200          MOV      #200,R4
4590 020504 020405          CMP     R4,R5          :IS FPS CORRECT?
4591 020506 001402          BEQ     X5
4592 020510 000137 021106          JMP     @#XERR2
4593 020514          X5:
4594 020514 104414          LPERR
4595 020516 012700 000200          MOV      #200,R0          :SET UP THE LOOP ON ERROR ADDRESS.
4596 020522 170100          LDFPS  R0          :SET DOUBLE MODE
4597 020524 012700 021170          MOV      #XPAT20,R0          :SET ACO TO
4598 020530 010037 021136          MOV      R0,@#XTMP          :NEGATIVE NUMBER
4599 020534 172410          LDD     (R0),AC0
4600 020536 012737 020550 001236  MOV      #X6,@#STMP2
4601 020544 012700 021160          MOV      #XPAT10,R0          :FSRC=0
4602 020550 172010          X6:     ADDD     (R0),AC0          :TEST INSTRUCTION
4603 020552 170205          STFPS   R5
4604 020554 170011          SETD
4605 020556 012700 021140          MOV      #XDAT00,R0          :GET RESULT
4606 020562 174010          STD     AC0,(R0)
4607 020564 012701 021170          MOV      #XPAT20,R1
4608 020570 012702 000004          MOV      #4,R2
4609 020574 022021          X7:     CMP      (R0)+,(R1)+          :IS RESULT CORRECT?
4610 020576 001401          BEQ     X10
4611 020600 000511          BR      XERR1
4612 020602 077204          X10:    SOB     R2,X7
4613 020604 012704 000210          MOV      #210,R4
4614 020610 020405          CMP     R4,R5          :IS FPS CORRECT?
4615 020612 001401          BEQ     X11
4616 020614 000534          BR      XERR2
4617 020616          X11:
4618 020616 104414          LPERR
4619 020620 012700 000200          MOV      #200,R0          :SET UP THE LOOP ON ERROR ADDRESS.
4620 020624 170100          LDFPS  R0          :SET DOUBLE MODE
4621 020626 012700 021150          MOV      #XPAT00,R0          :SET ACO TO NON-ZERO
4622 020632 010037 021136          MOV      R0,@#XTMP          :POSITIVE NUMBER
4623 020636 172410          LDD     (R0),AC0
4624 020640 012737 020652 001236  MOV      #X12,@#STMP2
4625 020646 012700 021160          MOV      #XPAT10,R0          :FSRC=0
4626 020652 173010          X12:    SUBD     (R0),AC0          :TEST INSTRUCTION
4627 020654 170205          STFPS   R5
4628 020656 170011          SETD
4629 020660 012700 021140          MOV      #XDAT00,R0          :GET RESULT
4630 020664 174010          STD     AC0,(R0)
    
```



```

4631 020666 012701 021150      MOV      #XPAT00,R1
4632 020672 012702 000004      MOV      #4,R2
4633 020676 022021      X13:    CMP      (R0)+,(R1)+      ;IS RESULT CORRECT?
4634 020700 001401      BEQ      X14
4635 020702 000463      BR      XERR3
4636 020704 077204      X14:    SOB      R2,X13
4637 020706 012704 000200      MOV      #200,R4      ;IS FPS CORRECT?
4638 020712 020405      CMP      R4,R5
4639 020714 001401      BEQ      X15
4640 020716 000501      BR      XERR4
4641 020720      X15:
4642 020720 104414      LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.
4643 020722 012700 000200      MOV      #200,R0
4644 020726 170100      LDFPS     R0      ;SET DOUBLE MODE
4645 020730 012700 021170      MOV      #XPAT20,R0      ;SET ACO=A NEGATIVE
4646 020734 010037 021136      MOV      R0,@#XTMP      ;NUMBER
4647 020740 172410      LDD      (R0),ACO
4648 020742 012737 020754 001236      MOV      #X16,@#STMP2
4649 020750 012700 021160      MOV      #XPAT10,R0      ;FSRC=0
4650 020754 173010      X16:    SUBD     (R0),ACO      ;TEST INSTRUCTION.
4651 020756 170205      STFPS     R5
4652 020760 170011      SETD
4653 020762 012700 021140      MOV      #XDAT00,R0      ;GET RESULT
4654 020766 174010      STD      ACO,(R0)
4655 020770 012701 021170      MOV      #XPAT20,R1
4656 020774 012702 000004      MOV      #4,R2
4657 021000 022021      X17:    CMP      (R0)+,(R1)+      ;IS RESULT CORRECT?
4658 021002 001401      BEQ      X20
4659 021004 000422      BR      XERR3
4660 021006 077204      X20:    SOB      R2,X17
4661 021010 012704 000210      MOV      #210,R4      ;IS FPS CORRECT?
4662 021014 020405      CMP      R4,R5
4663 021016 001401      BEQ      X21
4664 021020 000440      BR      XERR4
4665 021022 000466      X21:    BR      XDONE
4666
4667      ;REPORT DATA ERRORS
4668
4669 021024 012737 021160 001240      XERR1:  MOV      #XPAT10,@#STMP3
4670 021032 013737 021136 001242      MOV      @#XTMP,@#STMP4
4671 021040 012737 021140 001244      MOV      #XDAT00,@#STMP5
4672 021046 104001      1$:    ERROR     1
4673 021050 000453      BR      XDONE
4674 021052 012737 021160 001240      XERR3:  MOV      #XPAT10,@#STMP3
4675 021060 013737 021136 001242      MOV      @#XTMP,@#STMP4
4676 021066 012737 021140 001244      MOV      #XDAT00,@#STMP5
4677 021074 013737 021136 001246      MOV      @#XTMP,@#STMP6
4678 021102 104002      1$:    ERROR     2
4679 021104 000435      BR      XDONE
4680
4681      ;REPORT FPS ERRORS
4682
4683 021106      XERR2:
4684 021106 010537 001240      MOV      R5,@#STMP3
4685 021112 010437 001242      MOV      R4,@#STMP4
4686 021116 104001      1$:    ERROR     1

```

```

4687 021120 000427
4688 021122
4689 021122 010537 001240
4690 021126 010437 001242
4691 021132 104001
4692 021134 000421
4693 021136 000000
4694 021140 000000
4695 021142 000000
4696 021144 000000
4697 021146 000000
4698
4699 021150 010421
4700 021152 021042
4701 021154 031463
4702 021156 042104
4703
4704 021160 000000
4705 021162 000000
4706 021164 000000
4707 021166 000000
4708 021170 104210
4709 021172 114631
4710 021174 125252
4711 021176 135673
4712
4713 021200
4714 021200 104413
4715
4716
4717
4718
4719
4720
4721
4722
4723
4724
4725
4726
4727
4728 021202 000004
4729 021204 005037 021534
4730 021210 012737 021554 021536
4731 021216 012737 021564 021540
4732 021224 012737 000210 021542
4733 021232
4734 021232 104414
4735 021234 012700 000200
4736 021240 170100
4737 021242 012700 021574
4738 021246 172410
4739 021250 013700 021536
4740 021254 173010
4741 021256 170205
4742 021260 170011

XERR4: BR XDONE
MOV R5,@#STMP3
MOV R4,@#STMP4
1$: ERROR 1
BR XDONE
XTMP: .WORD 0
XDAT00: .WORD 0
XDAT01: 0
XDAT02: 0
XDAT03: 0
XPAT00: .WORD 010421
XPAT01: 021042
XPAT02: 031463
XPAT03: 042104
XPAT10: .WORD 0
XPAT11: 0
XPAT12: 0
XPAT13: 0
XPAT20: .WORD 104210
XPAT21: 114631
XPAT22: 125252
XPAT23: 135673
XDONE: RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
;SEE IF THE USER HAS EXPRESSED
;THE DESIRE TO CHANGE THE SOFTWARE
;VIRTUAL CONSOLE SWITCH REGISTER (HAS
;THE USER TYPED CONTROL G?).

:*****
:*TEST 27 SUBD WITH AC=0 TEST
:*
:* THIS IS A TEST OF SUBD WITH AC=0. BOTH POSITIVE
:* AND NEGATIVE FSRC'S ARE TRIED.
:*
:*****
TST27: SCOPE
CLR @#YFLAG
MOV #YPAT00,@#YTMP1 ;P
MOV #YPAT10,@#YTMP2 ;N
MOV #210,@#YTMP3
Y1: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
MOV #200,R0
LDFPS R0 ;SET DOUBLE MODE
MOV #YPAT20,R0 ;SET ACO=0
LDD (R0),AC0
MOV @#YTMP1,R0
Y2: SUBD (R0),AC0 ;TEST INSTRUCTION
STFPS R5
SETD
    
```

```

4743 021262 012700 021544      MOV      #YDAT00,R0      ;GET RESULT
4744 021266 174010      STD      AC0,(R0)
4745 021270 012702 000004      MOV      #4,R2
4746 021274 013701 021540      MOV      @#YTMP2,R1      ;CHECK RESULT.
4747 021300 022021      Y3:     CMP      (R0)+,(R1)+
4748 021302 001026      BNE
4749 021304 077203      SOB     R2,Y3
4750 021306 023705 021542      CMP      @#YTMP3,R5      ;FPS CORRECT?
4751 021312 001401      BEQ     Y4
4752 021314 000475      BR
4753 021316 005737 021534      Y4:     TST      @#YFLAG      ;FINISHED TEST?
4754 021322 001015      BNE     Y5
4755 021324 012737 177777 021534      MOV      #-1,@#YFLAG
4756 021332 012737 021564 021536      MOV      #YPAT10,@#YTMP1
4757 021340 012737 021554 021540      MOV      #YPAT00,@#YTMP2
4758 021346 012737 000200 021542      MOV      #200,@#YTMP3
4759 021354 000726      BR      Y1
4760 021356 000512      Y5:     BR      YDONE
4761 021360 012702 000004      Y6:     MOV      #4,R2
4762 021364 012700 021536      MOV      #YTMP1,R0      ;DID XOR OF SIGN BIT
4763 021370 012701 021544      MOV      #YDAT00,R1      ;FAIL?
4764 021374 022021      Y7:     CMP      (R0)+,(R1)+
4765 021376 001002      BNE     YERR1
4766 021400 077203      SOB     R2,Y7
4767 021402 000421      BR      YERR2
4768 021404      YERR1:  ;DATA FAILURE
4769 021404 012737 021254 001236      MOV      #Y2,@#STMP2
4770 021412 013737 021536 001240      MOV      @#YTMP1,@#STMP3
4771 021420 012737 021574 001242      MOV      #YPAT20,@#STMP4
4772 021426 012737 021544 001244      MOV      #YDAT00,@#STMP5
4773 021434 013737 021540 001246      MOV      @#YTMP2,@#STMP6
4774 021442 104001      1$:     ERROR    1
4775 021444 000457      BR      YDONE
4776 021446      YERR2:  ;XOR OF SIGN BIT
4777 021446 012737 021254 001236      MOV      #Y2,@#STMP2      ;FAILED
4778 021454 013737 021536 001240      MOV      @#YTMP1,@#STMP3
4779 021462 012737 021574 001242      MOV      #YPAT20,@#STMP4
4780 021470 012737 021544 001244      MOV      #YDAT00,@#STMP5
4781 021476 013737 021540 001246      MOV      @#YTMP2,@#STMP6
4782 021504 104002      1$:     ERROR    2
4783 021506 000436      BR      YDONE
4784 021510      YERR3:  ;FPS WRONG.
4785 021510 012737 021254 001236      MOV      #Y2,@#STMP2
4786 021516 010537 001240      MOV      R5,@#STMP3
4787 021522 013737 021542 001242      MOV      @#YTMP3,@#STMP4
4788 021530 104002      1$:     ERROR    2
4789 021532 000424      BR      YDONE
4790
4791 021534 000000      YFLAG:  .WORD    0
4792 021536 000000      YTMP1:  0
4793 021540 000000      YTMP2:  0
4794 021542 000000      YTMP3:  0
4795
4796 021544 000000      YDAT00: .WORD    0
4797 021546 000000      YDAT01: 0
4798 021550 000000      YDAT02: 0
    
```

4799	021552	000000		YDAT03:	0	
4800						
4801	021554	063146		YPAT00:	063146	
4802	021556	052525		YPAT01:	052525	
4803	021560	042104		YPAT02:	042104	
4804	021562	167356		YPAT03:	167356	
4805						
4806	021564	163146		YPAT10:	163146	
4807	021566	052525		YPAT11:	052525	
4808	021570	042104		YPAT12:	042104	
4809	021572	167356		YPAT13:	167356	
4810						
4811	021574	000000		YPAT20:	0	
4812	021576	000000		YPAT21:	0	
4813	021600	000000		YPAT22:	0	
4814	021602	000000		YPAT23:	0	
4815						
4816	021604			YDONE:		
4817	021604	104413		RSETUP		
4818						:GO INITIALIZE THE FPS AND STACK; AND
4819						:SEE IF THE USER HAS EXPRESSED
4820						:THE DESIRE TO CHANGE THE SOFTWARE
4821						:VIRTUAL CONSOLE SWITCH REGISTER (HAS
4822						:THE USER TYPED CONTROL G?).
4823						::*****
4824						:*TEST 30 ADDD WITH AC=0 TEST
4825						
4826						:@
4827						:@ THIS IS A TEST OF ADDD WITH AC=0. BOTH
4828						:* POSITIVE AND NEGATIVE FSRC'S ARE TRIED.
4829						:*
4830						::*****
4831	021606	000004		TST30:	SCOPE	
4832	021610	005067	000224		CLR ZFLAG	
4833	021614	012737	022056		MOV #ZPAT00,@#ZTMP1 ;P	
4834	021622	012737	000200		MOV #200,@#ZTMP2	
4835	021630			Z1:		
4836	021630	104414			LPERR	:SET UP THE LOOP ON ERROR ADDRESS.
4837	021632	012700	000200		MOV #200,R0	
4838	021636	170100			LDFPS R0	:SET DOUBLE MODE
4839	021640	012700	022076		MOV #ZPAT20,R0	:SET ACO=0
4840	021644	172410			LDD (R0),ACO	
4841	021646	013700	022042		MOV @#ZTMP1,R0	
4842	021652	172010		Z2:	ADDD (R0),ACO	:TEST INSTRUCTION
4843	021654	170205			STFPS R5	
4844	021656	170011			SETD	
4845	021660	012700	022046		MOV #ZDAT00,R0	:GET RESULT
4846	021664	174010			STD ACO,(R0)	
4847	021666	012702	000004		MOV #4,R2	
4848	021672	013701	022042		MOV @#ZTMP1,R1	:RESULT CORRECT?
4849	021676	022021		Z3:	CMP (R0)+,(R1)+	
4850	021700	001401			BEQ Z4	
4851	021702	000423			BR ZERR1	
4852	021704	077204		Z4:	SOB R2,Z3	
4853	021706	023705	022044		CMP @#ZTMP2,R5	:FPS CORRECT?
4854	021712	001401			BEQ Z5	

```

4855 021714 000437          BR      ZERR2
4856 021716 005737 022040  Z5:    TST      @#ZFLAG      ;FINISHED TEST?
4857 021722 001012          BNE     Z6
4858 021724 012737 177777 022040  MOV     #-1,@#ZFLAG
4859 021732 012737 022066 022042  MOV     #ZPAT10,@#ZTMP1
4860 021740 012737 000210 022044  MOV     #210,@#ZTMP2
4861 021746 000730          BR      Z1
4862 021750 000456          Z6:    BR      ZDONE
4863 021752          ZERR1:          ;DATA FAILURE
4864 021752 012737 021652 001236  MOV     #Z2,@#STMP2
4865 021760 013737 022042 001240  MOV     @#ZTMP1,@#STMP3
4866 021766 012737 022076 001242  MOV     #ZPAT20,@#STMP4
4867 021774 012737 022046 001244  MOV     #ZDAT00,@#STMP5
4868 022002 013737 022042 001246  MOV     @#ZTMP1,@#STMP6
4869 022010 104002          1$:    ERROR   2
4870 022012 000435          BR      ZDONE
4871 022014          ZERR2:
4872 022014 012737 021652 001236  MOV     #Z2,@#STMP2
4873 022022 010537 001240          MOV     R5,@#STMP3
4874 022026 013737 022044 001242  MOV     @#ZTMP2,@#STMP4
4875 022034 104002          1$:    ERROR   2
4876 022036 000423          BR      ZDONE
4877
4878 022040 000000          ZFLAG:  .WORD   0
4879 022042 000000          ZTMP1:  0
4880 022044 000000          ZTMP2:  0
4881
4882 022046 000000          ZDAT00: .WORD   0
4883 022050 000000          ZDAT01: 0
4884 022052 000000          ZDAT02: 0
4885 022054 000000          ZDAT03: 0
4886
4887 022056 031463          ZPAT00: 031463
4888 022060 010421          ZPAT01: 010421
4889 022062 146314          ZPAT02: 146314
4890 022064 156735          ZPAT03: 156735
4891
4892 022066 156735          ZPAT10: 156735
4893 022070 167356          ZPAT11: 167356
4894 022072 135673          ZPAT12: 135673
4895 022074 146314          ZPAT13: 146314
4896
4897 022076 000000          ZPAT20: 0
4898 022100 000000          ZPAT21: 0
4899 022102 000000          ZPAT22: 0
4900 022104 000000          ZPAT23: 0
4901
4902 022106          ZDONE:
4903 022106 104413          RSETUP      ;GO INITIALIZE THE FPS AND STACK; AND
4904                                     ;SEE IF THE USER HAS EXPRESSED
4905                                     ;THE DESIRE TO CHANGE THE SOFTWARE
4906                                     ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
4907                                     ;THE USER TYPED CONTROL G?).
4908
4909
4910                                     ;*****
    
```

```

4911      ;*TEST 31      ADDF AND ADDD WITH E(AC)=E(FSRC) TEST AND (BUT FT) TEST
4912      ;*
4913      ;* THIS IS A TEST OF THE ADD INSTRUCTION WITH THE
4914      ;* OPERANDS HAVING EQUAL EXPONENTS.
4915      ;* THE ROUND/TRUNK FLOWS IS ALSO TESTED.
4916      ;*
4917      ;*****
4918 022110 000004 TST31: SCOPE
4919 022112 AA1:
4920 022112 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
4921 022114 012700 003240 MOV #3240,R0
4922 022120 170100 LDFPS R0 ;SET FIU FIV FD AND FT
4923 022122 012737 022472 000244 MOV #AAERRO,@#FPVECT ;IN CASE THE OVER/UNDER
4924 022130 012700 023050 MOV #AAPATO,R0 ;FLOWS IN TRAP WILL
4925 ;OCCUR
4926 022134 172410 LDD (R0),AC0 ;SET UP AC0
4927 022136 012737 022150 001236 MOV #AA2,@#STMP2 ;OPERAND
4928 022144 012700 023060 MOV #AAPAT1,R0
4929 022150 172010 AA2: ADDD (R0),AC0 ;TEST INSTRUCTION
4930 ;SHOULD TRUNCATE
4931 022152 012700 023040 AA3: MOV #AADATO,R0
4932 022156 174010 STD AC0,(R0) ;GET THE RESULT
4933 022160 012701 023070 MOV #AAPAT2,R1
4934 022164 012702 000004 MOV #4,R2
4935 022170 022021 AA4: CMP (R0)+,(R1)+ ;CORRECT?
4936 022172 001414 BEQ AA7
4937 022174 012700 023100 MOV #AAPAT3,R0 ;DID (BUT FT) FAIL
4938 022200 012701 023040 MOV #AADATO,R1
4939 022204 012702 000004 MOV #4,R2
4940 022210 022021 AA5: CMP (R0)+,(R1)+
4941 022212 001401 BEQ AA6
4942 022214 000561 BR AAERR1 ;DATA ERROR
4943 022216 077204 AA6: SOB R2,AA5
4944 022220 000137 022614 JMP @#AAERR2 ;(BUT FT) ERROR
4945 022224 077217 AA7: SOB R2,AA4
4946
4947 ;NOW TEST DOUBLE FLOATING ROUND MODE.
4948 ;A 1 SHOULD BE ADDED TO THE LSB ON ROUND MODE.
4949
4950 022226 AA10:
4951 022226 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
4952 022230 012700 003200 MOV #3200,R0 ;SET FD FIV FIV. FT=0
4953 022234 170100 LDFPS R0
4954 022236 012700 023050 MOV #AAPATO,R0
4955 022242 172410 LDD (R0),AC0 ;SET UP AC0 OPERAND
4956 022244 012737 022256 001236 MOV #AA11,@#STMP2
4957 022252 012700 023060 MOV #AAPAT1,R0
4958 022256 172010 AA11: ADDD (R0),AC0 ;TEST INSTRUCTION
4959 ;SHOULD ROUND
4960 022260 012700 023040 AA12: MOV #AADATO,R0
4961 022264 174010 STD AC0,(R0) ;GET THE RESULT
4962 022266 012701 023100 MOV #AAPAT3,R1
4963 022272 012702 000004 MOV #4,R2
4964 022276 022021 AA13: CMP (R0)+,(R1)+ ;CORRECT?
4965 022300 001425 BEQ AA20
4966 022302 012700 023070 MOV #AAPAT2,R0 ;DID (BUT FT) FAIL?
    
```

```

4967 022306 012701 023040      MOV      #AADATO,R1
4968 022312 012702 000004      MOV      #4,R2
4969 022316 022021      AA14:   CMP      (R0)+,(R1)+
4970 022320 001413      BEQ      AA17
4971 022322 012700 023110      MOV      #AAPAT4,R0      ;WAS THE FLOATING
4972 022326 012701 023040      MOV      #AADATO,R1      ;CONSTANT USED
4973 022332 012702 000004      MOV      #4,R2      ;INSTEAD OF THE
4974 022336 022021      AA15:   CMP      (R0)+,(R1)+      ;DOUBLE CONSTANT
4975 022340 001401      BEQ      AA16      ;IN THE ROUND
4976 022342 000542      BR      AAERR3      ;FLOWS?
4977 022344 077204      AA16:   SOB      R2,AA15      ;DATA ERROR
4978 022346 000544      BR      AAERR4      ;CONSTANT ERROR
4979 022350 077216      AA17:   SOB      R2,AA14
4980 022352 000560      BR      AAERR5      ;(BUT FT) ERROR
4981 022354 077230      AA20:   SOB      R2,AA13
4982
4983      ;NOW TEST ADDF WITH FT=0, ROUND MODE
4984
4985 022356      AA21:
4986 022356 104414      LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.
4987 022360 012700 003200      MOV      #3200,R0      ;FIV=1, FIV-1, FT=0
4988 022364 170100      LDFPS     R0
4989 022366 012700 023050      MOV      #AAPATO,R0      ;LOAD ACO OPERAND
4990 022372 172410      LDD      (R0),ACO
4991 022374 170001      SETF     ;ENTER FLOATING MODE
4992 022376 012737 022410 001236      MOV      #AA22,@#STMP2
4993 022404 012700 023120      MOV      #AAPAT5,R0
4994 022410 172010      AA22:   ADDF     (R0),ACO      ;TEST INSTRUCTION
4995      ;SHOULD ROUND
4996 022412      AA23:
4997 022412 170011      SETD     ;RESET TO DOUBLE
4998      ;MODE
4999 022414 012700 023040      MOV      #AADATO,R0      ;GET THE RESULT
5000 022420 174010      STD      ACO,(R0)
5001 022422 012701 023130      MOV      #AAPAT6,R1      ;CORRECT?
5002 022426 012702 000002      MOV      #2,R2
5003 022432 022021      AA24:   CMP      (R0)+,(R1)+
5004 022434 001413      BEQ      AA27
5005 022436 012700 023070      MOV      #AAPAT2,R0      ;WAS THE DOUBLE
5006 022442 012701 023040      MOV      #AADATO,R1      ;CONSTANT USED INSTEAD
5007 022446 012702 000002      MOV      #2,R2      ;OF THE FLOATING
5008 022452 022011      AA25:   CMP      (R0)+,(R1)      ;CONSTANT IN THE
5009 022454 001401      BEQ      AA26      ;ROUND FLOWS?
5010 022456 000534      BR      AAERR6      ;DATA ERROR
5011 022460 077204      AA26:   SOB      R2,AA25
5012 022462 000550      BR      AAERR7      ;CONSTANT ERROR
5013 022464 077216      AA27:   SOB      R2,AA24
5014 022466 000137 023140      JMP      @#AADONE
5015
5016      ;COME HERE IF A TRAP OCCURS TO 244.
5017
5018 022472 013700 001236      AAERRO: MOV      @#STMP2,R0      ;SEE IF THE TRAP WAS
5019 022476 005720      TST      (R0)+      ;AT A TEST INSTRUCTION
5020 022500 020016      CMP      R0,(SP)
5021 022502 001402      BEQ      1$
5022 022504 000137 062534      10$:   JMP      @#FPSPUR
    
```

```

5023 022510          1$:
5024 022510 170300          STST   RO          ;GET FEC
5025 022512 020027 000010    CMP    RO,#10
5026 022516 001405          BEQ    20$         ;OVERFLOW
5027 022520 020027 000012    CMP    RO,#12
5028 022524 001410          BEQ    30$         ;UNDERFLOW
5029 022526 000766          BR     10$
5030 022530 022532          20$:
5031 022532 011637 001236    20$:  MOV    (SP),@#$TMP2 ;REPORT OVERFLOW ERROR
5032 022536 022626          CMP    (SP)+,(SP)+
5033 022540 104002          21$:  ERROR  2
5034 022542 000137 023140    25$:  JMP    @#$ADONE
5035 022546 011637 001236    30$:  MOV    (SP),@#$TMP2 ;REPORT UNDERFLOW
5036 022552 022626          CMP    (SP)+,(SP)+ ;ERROR
5037 022554 104002          31$:  ERROR  2
5038 022556 000771          BR     25$
5039
5040          ;ADD RESULT INCORRECT
5041 022560 012737 023070 001246  AAERR1: MOV   #AAPAT2,@#$TMP6
5042 022566 012737 023050 001242  AAERR10: MOV  #AAPAT0,@#$TMP4
5043 022574 012737 023060 001240          MOV   #AAPAT1,@#$TMP3
5044 022602 012737 023040 001244          MOV   #AADATO,@#$TMP5
5045 022610 104002          1$:   ERROR  2
5046 022612 000552          BR     AADONE
5047 022614 012737 023070 001246  AAERR2: MOV   #AAPAT2,@#$TMP6 ;(BUT FT) FAILED.
5048 022622 012737 023050 001242          MOV   #AAPAT0,@#$TMP4
5049 022630 012737 023060 001240          MOV   #AAPAT1,@#$TMP3
5050 022636 012737 023040 001244          MOV   #AADATO,@#$TMP5
5051 022644 104002          1$:   ERROR  2
5052 022646 000534          BR     AADONE
5053 022650 012737 023100 001246  AAERR3: MOV   #AAPAT3,@#$TMP6 ;DATA ERROR.
5054 022656 000743          BR     AAERR10
5055 022660 012737 023100 001246  AAERR4: MOV   #AAPAT3,@#$TMP6 ;BAD CONSTANT
5056 022666 012737 023050 001242          MOV   #AAPAT0,@#$TMP4
5057 022674 012737 023060 001240          MOV   #AAPAT1,@#$TMP3
5058 022702 012737 023040 001244          MOV   #AADATO,@#$TMP5
5059 022710 104002          1$:   ERROR  2
5060 022712 000512          BR     AADONE
5061 022714 012737 023100 001246  AAERR5: MOV   #AAPAT3,@#$TMP6 ;(BUT FT) FAILED.
5062 022722 012737 023050 001242          MOV   #AAPAT0,@#$TMP4
5063 022730 012737 023060 001240          MOV   #AAPAT1,@#$TMP3
5064 022736 012737 023040 001244          MOV   #AADATO,@#$TMP5
5065 022744 104002          1$:   ERROR  2
5066 022746 000474          BR     AADONE
5067 022750 012737 023120 001240  AAERR6: MOV   #AAPAT5,@#$TMP3 ;FD=0 AND
5068 022756 012737 023050 001242          MOV   #AAPAT0,@#$TMP4 ;DATA ERROR
5069 022764 012737 023040 001244          MOV   #AADATO,@#$TMP5
5070 022772 012737 023130 001246          MOV   #AAPAT6,@#$TMP6
5071 023000 104002          1$:   ERROR  2
5072 023002 000456          BR     AADONE
5073 023004 012737 023120 001240  AAERR7: MOV   #AAPAT5,@#$TMP3 ;CONSTANT ERROR
5074 023012 012737 023050 001242          MOV   #AAPAT0,@#$TMP4
5075 023020 012737 023040 001244          MOV   #AADATO,@#$TMP5
5076 023026 012737 023130 001246          MOV   #AAPAT6,@#$TMP6
5077 023034 104002          1$:   ERROR  2
5078 023036 000440          BR     AADONE
    
```



5079	023040	000000	AADATO:	0
5080	023042	000000		0
5081	023044	000000		0
5082	023046	000000		0
5083	023050	000200	AAPATO:	200
5084	023052	000000		0
5085	023054	000000		0
5086	023056	000000		0
5087	023060	000200	AAPAT1:	200
5088	023062	000000		0
5089	023064	000000		0
5090	023066	000001		1
5091	023070	000400	AAPAT2:	400
5092	023072	000000		0
5093	023074	000000		0
5094	023076	000000		0
5095	023100	000400	AAPAT3:	400
5096	023102	000000		0
5097	023104	000000		0
5098	023106	000001		1
5099	023110	000400	AAPAT4:	400
5100	023112	000000		0
5101	023114	100000		100000
5102	023116	000000		0
5103	023120	000200	AAPAT5:	200
5104	023122	000001		1
5105	023124	000000		0
5106	023126	000000		0
5107	023130	000400	AAPAT6:	400
5108	023132	000001		1
5109	023134	000000		0
5110	023136	000000		0

5111 023140  
 5112 023140 104413 RSETUP ;GO INITIALIZE THE FPS AND STACK; AND  
 5113 ;SEE IF THE USER HAS EXPRESSED  
 5114 ;THE DESIRE TO CHANGE THE SOFTWARE  
 5115 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS  
 5116 ;THE USER TYPED CONTROL G?).

5117 ;\*\*\*\*\*  
 5118 ;\*TEST 32 ADDF AND ADDD WITH E(AC) LESS THAN E(FSRC) TEST  
 5119 ;\*  
 5120 ;\*THIS IS A TEST OF THE ADDD AND ADDF  
 5121 ;\*INSTRUCTIONS AND THE ALIGN AC ALGORITHM  
 5122 ;\*FLOWS. THE CONSTANT (25 FOR FLOATING, 57 FOR  
 5123 ;\*DOUBLE) USED IS CHECKED. THEN SIMPLE  
 5124 ;\*AND WORST CASE ALIGNMENT SITUATIONS ARE  
 5125 ;\*TRIED. NOTE E(AC) IS LESS THEN E(FSRC)  
 5126 ;\*  
 5127 ;\*\*\*\*\*

5128 023142 000004 TST32: SCOPE  
 5129 ;EXPONENT DIFFERENCE=57=71 (OCT) FD=1  
 5130 CC1:  
 5131 023144 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.  
 5132 023146 012704 003200 MOV #3200,R4 ;SET FIV,FIV, AND FD  
 5133 023152 170104 LDFPS R4  
 5134 023154 012737 023174 001236 MOV #CC2,@#TMP2

```

CJKDCA-A KEF11-A FP DIAG PART 1 MACY11 30A(1052) 11-JUN-79 11:50 E 8 PAGE 95
CJKDCA.P11 08-JUN-79 08:40 T32 ADDF AND ADDD WITH E(AC) LESS THAN E(FSRC) TEST SEQ 0095

5135 023162 012700 024576 MOV #CCP0,R0 ;SET ACO OPERAND
5136 023166 172410 LDD (R0),ACO ;ACO
5137 023170 012700 024616 MOV #CCP2,R0
5138 023174 172010 CC2: ADDD (R0),ACO ;TEST INSTRUCTION
5139 023176 170205 STFPS R5 ;GET FPS
5140 023200 012700 024566 MOV #CCDATO,R0 ;GET THE RESULT
5141 023204 174010 STD ACO,(R0)
5142 023206 012701 024616 MOV #CCP2,R1 ;IS IT CORRECT
5143 023212 012702 000004 MOV #4,R2
5144 023216 022021 CC3: CMP (R0)+,(R1)+
5145 023220 001415 BEQ CC6
5146 023222 012700 024566 MOV #CCDATO,R0 ;DID A BAD
5147 023226 012701 024576 MOV #CCP0,R1 ;CONSTANT (NOT 57)
5148 023232 012702 000004 MOV #4,R2 ;GET GENERATED
5149 023236 022021 CC4: CMP (R0)+,(R1)+ ;FOR THE ALIGNMENT
5150 023240 001402 BEQ CC5 ;FLOWS?
5151 023242 000137 024164 JMP @#CCER1 ;DATA ERROR.D
5152 023246 077205 CC5: SOB R2,CC4
5153 023250 000137 024222 JMP @#CCER2 ;BAD CONSTANT.D
5154 023254 077220 CC6: SOB R2,CC3
5155 023256 020405 CMP R4,R5 ;FPS CORRECT?
5156 023260 001402 BEQ CC7
5157 023262 000137 024130 JMP @#CCERO ;BAD FPS.
5158 ;EXPONENT DIFFERENCE=56=70 (OCT) FD=1
5159 CC7: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
5160 023266 104414 MOV #3200,R4 ;SET FIV,FIV, AND FD
5161 023270 012704 003200 LDFPS R4
5162 023274 170104 MOV #CC8,@#STMP2
5163 023276 012737 023316 001236 MOV #CCP0,R0 ;SET ACO OPERAND
5164 023304 012700 024576 LDD (R0),ACO
5165 023310 172410 MOV #CCP1,R0 ;FSRC
5166 023312 012700 024606 ADDD (R0),ACO ;TEST INSTRUCTION
5167 023316 172010 CC8: STFPS R5 ;GET FPS
5168 023320 170205 MOV #CCDATO,R0 ;GET THE RESULT
5169 023322 012700 024566 STD ACO,(R0)
5170 023326 174010 MOV #CCP7,R1 ;IS IT CORRECT
5171 023330 012701 024666 MOV #4,R2
5172 023334 012702 000004 CC9: CMP (R0)+,(R1)+
5173 023340 022021 BEQ CC12
5174 023342 001415 MOV #CCDATO,R0 ;DID A BAD
5175 023344 012700 024566 MOV #CCP1,R1 ;CONSTANT (NOT 57)
5176 023350 012701 024606 MOV #4,P2 ;GET GENERATED
5177 023354 012702 000004 CC10: CMP (R0)+,(R1)+ ;FOR THE ALIGNMENT
5178 023360 022021 BEQ CC11 ;FLOWS?
5179 023362 001402 JMP @#CCER3 ;DATA ERROR.D
5180 023364 000137 024260 CC11: SOB R2,CC10
5181 023370 077205 CC12: SOB R2,CC9 ;BAD CONSTANT.D
5182 023372 000137 024276 CMP R4,R5 ;FPS CORRECT?
5183 023376 077220 BEQ CC13
5184 023400 020405 JMP @#CCERO ;BAD FPS.
5185 023402 001402 ;EXPONENT DIFFERENCE=25=31 (OCT) FD=0
5186 023404 000137 024130 CC13: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
5187 023410 MOV #CC14,@#STMP2
5188 023410 104414
5189 023410 012737 023440 001236
5190 023412 012737 023440 001236

```

5191	023420	012700	024576		MOV	#CCP0,R0		;SET UP ACO OPERAND.
5192	023424	172410			LDD	(R0),ACO		
5193	023426	012704	003000		MOV	#3000,R4		;SET FIV,FIV. CLEAR FD.
5194	023432	170104			LDFPS	R4		
5195	023434	012700	024656		MOV	#CCP6,R0		;FSRC
5196	023440	172010		CC14:	ADDF	(R0),ACO		;TEST INSTRUCTION
5197	023442	170205			STFPS	R5		
5198	023444	170011			SETD			;REENTER DOUBLE MOVE
5199	023446	012700	024566		MOV	#CCDATO,R0		;GET THE RESULT
5200	023452	174010			STD	ACO,(R0)		
5201	023454	012701	024656		MOV	#CCP6,R1		;IS THE RESULT CORRECT?
5202	023460	012702	000002		MOV	#2,R2		
5203	023464	022021		CC15:	CMP	(R0)+,(R1)+		
5204	023466	001415			BEQ	CC18		
5205	023470	012700	024566		MOV	#CCDATO,R0		;WAS A BAD CONSTANT
5206	023474	012701	024626		MOV	#CCP3,R1		;USED (NOT 25) IN
5207	023500	012702	000002		MOV	#2,R2		;THE ALIGN FLOWS?
5208	023504	022021		CC16:	CMP	(R0)+,(R1)+		
5209	023506	001402			BEQ	CC17		
5210	023510	000137	024334		JMP	@#CCER5		;DATA ERROR F
5211	023514	077205		CC17:	SOB	R2,CC16		
5212	023516	000137	024370		JMP	@#CCER6		;BAD CONSTANT F
5213	023522	077220		CC18:	SOB	R2,CC15		
5214	023524	020405			CMP	R4,R5		
5215	023526	001402			BEQ	CC19		
5216	023530	000137	024146		JMP	@#CCER90		;BAD FPS.
5217								;EXPONENT DIFFERENCE=24=30 (OCT) FD=0
5218	023534			CC19:	LPERR			
5219	023534	104414			MOV	#CC20,@#STMP2		;SET UP THE LOOP ON ERROR ADDRESS.
5220	023536	012757	023564	001236	MOV	#CCP3,R0		;SET UP ACO OPERAND.
5221	023544	012700	024626		MOV	#3000,R4		;SET FIV,FIV. CLEAR FD.
5222	023550	172410			LDD	(R0),ACO		
5223	023552	012704	003000		MOV	#3000,R4		;SET FIV,FIV. CLEAR FD.
5224	023556	170104			LDFPS	R4		
5225	023560	012700	024646		MOV	#CCP5,R0		;FSRC
5226	023564	172010		CC20:	ADDF	(R0),ACO		;TEST INSTRUCTION
5227	023566	170205			STFPS	R5		
5228	023570	170011			SETD			;REENTER DOUBLE MOVE
5229	023572	012700	024566		MOV	#CCDATO,R0		;GET THE RESULT
5230	023576	174010			STD	ACO,(R0)		
5231	023600	012701	024676		MOV	#CCP10,R1		;IS THE RESULT CORRECT?
5232	023604	012702	000002		MOV	#2,R2		
5233	023610	022021		CC21:	CMP	(R0)+,(R1)+		
5234	023612	001415			BEQ	CC24		
5235	023614	012700	024566		MOV	#CCDATO,R0		;WAS A BAD CONSTANT
5236	023620	012701	024646		MOV	#CCP5,R1		;USED (NOT 25) IN
5237	023624	012702	000002		MOV	#2,R2		;THE ALIGN FLOWS?
5238	023630	022021		CC22:	CMP	(R0)+,(R1)+		
5239	023632	001402			BEQ	CC23		
5240	023634	000137	024424		JMP	@#CCER7		;DATA ERROR F
5241	023640	077205		CC23:	SOB	R2,CC22		
5242	023642	000137	024442		JMP	@#CCER8		;BAD CONSTANT F
5243	023646	077220		CC24:	SOB	R2,CC21		
5244	023650	020405			CMP	R4,R5		
5245	023652	001402			BEQ	CC25		
5246	023654	000137	024146		JMP	@#CCER90		;BAD FPS.

```

5247      ;EXPONENT DIFFERENCE=1 FD=1
5248 023660      CC25:      LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.
5249 023660 104414      MOV      #3200,R4      ;SET FIV,FIV, AND FD
5250 023662 012704 003200      LDFPS      R4
5251 023666 170104      MOV      #CC26,@#STMP2
5252 023670 012737 023710 001236      MOV      #CCP0,R0      ;SET ACO OPERAND
5253 023676 012700 024576      LDD      (R0),ACO
5254 023702 172410      MOV      #CCP3,R0      ;FSRC
5255 023704 012700 024626      ADDD     (R0),ACO      ;TEST INSTRUCTION
5256 023710 172010      STFPS     R5          ;GET FPS
5257 023712 170205      MOV      #CCDAT0,R0      ;GET THE RESULT
5258 023714 012700 024566      STD      ACO,(R0)
5259 023720 174010      MOV      #CCP11,R1      ;IS IT CORRECT
5260 023722 012701 024706      MOV      #4,R2
5261 023726 012702 000004      CMP      (R0)+,(R1)+
5262 023732 022021      BEQ      CC30
5263 023734 001415      MOV      #CCDAT0,R0      ;DID A BAD
5264 023736 012700 024566      MOV      #CCP3,R1      ;CONSTANT (NOT 57)
5265 023742 012701 024626      MOV      #4,R2          ;GET GENERATED
5266 023746 012702 000004      CMP      (R0)+,(R1)+    ;FOR THE ALIGNMENT
5267 023752 022021      BEQ      CC29          ;FLOWS?
5268 023754 001402      JMP      @#CCER10      ;DATA ERROR.D
5269 023756 000137 024476      SOB      R2,CC28
5270 023762 077205      JMP      @#CCER11      ;BAD CONSTANT.D
5271 023764 000137 024514      SOB      R2,CC27
5272 023770 077220      CMP      R4,R5          ;FPS CORRECT?
5273 023772 020405      BEQ      CC31
5274 023774 001402      JMP      @#CCER0      ;BAD FPS.
5275 023776 000137 024130      ;EXPONENT DIFFERENCE=100=144 (OCT) FD=1
5276      CC31:      LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.
5277 024002      MOV      #3200,R4      ;SET FIV,FIV, AND FD
5278 024002 104414      LDFPS      R4
5279 024004 012704 003200      MOV      #CC32,@#STMP2
5280 024010 170104      MOV      #CCP0,R0      ;SET ACO OPERAND
5281 024012 012737 024032 001236      LDD      (R0),ACO
5282 024020 012700 024576      MOV      #CCP4,R0      ;FSRC
5283 024024 172410      ADDD     (R0),ACO      ;TEST INSTRUCTION
5284 024026 012700 024636      STFPS     R5          ;GET FPS
5285 024032 172010      MOV      #CCDAT0,R0      ;GET THE RESULT
5286 024034 170205      STD      ACO,(R0)
5287 024036 012700 024566      MOV      #CCP4,R1      ;IS IT CORRECT
5288 024042 174010      MOV      #4,R2
5289 024044 012701 024636      CMP      (R0)+,(R1)+
5290 024050 012702 000004      BEQ      CC36
5291 024054 022021      MOV      #CCDAT0,R0      ;DID A BAD
5292 024056 001415      MOV      #CCP4,R1      ;CONSTANT (NOT 57)
5293 024060 012700 024566      MOV      #4,R2          ;GET GENERATED
5294 024064 012701 024636      CMP      (R0)+,(R1)+    ;FOR THE ALIGNMENT
5295 024070 012702 000004      BEQ      CC35          ;FLOWS?
5296 024074 022021      JMP      @#CCER12      ;DATA ERROR.D
5297 024076 001402      SOB      R2,CC34
5298 024100 000137 024532      JMP      @#CCER13      ;BAD CONSTANT.D
5299 024104 077205      SOB      R2,CC33
5300 024106 000137 024550      CMP      R2,CC33
5301 024112 077220      CMP      R4,R5          ;FPS CORRECT?
5302 024114 020405
    
```

5303	024116	001402				BEQ	CC37		
5304	024120	000137	024130			JMP	@#CCERO		;BAD FPS.
5305	024124	000137	024726			CC37:	JMP	@#CCDONE	
5306	024130	010437	001242			CCERO:	MOV	R4,@#STMP4	;FPS ERROR D
5307	024134	010537	001240				MOV	R5,@#STMP3	
5308	024140	104002				1\$:	ERROR	2	
5309	024142	000137	024726				JMP	@#CCDONE	
5310	024146	010437	001242			CCER90:	MOV	R4,@#STMP4	;FPS ERROR F
5311	024152	010537	001240				MOV	R5,@#STMP3	
5312	024156	104002				1\$:	ERROR	2	
5313	024160	000137	024726				JMP	@#CCDONE	
5314	024164	012737	024616	001240		CCER1:	MOV	#CCP2,@#STMP3	;DATA ERROR D
5315	024172	012737	024616	001246			MOV	#CCP2,@#STMP6	
5316	024200	012737	024576	001242		CCER50:	MOV	#CCP0,@#STMP4	
5317	024206	012737	024566	001244			MOV	#CCDAT0,@#STMP5	
5318	024214	104002				1\$:	ERROR	2	
5319	024216	000137	024726				JMP	@#CCDONE	
5320	024222	012737	024616	001240		CCER2:	MOV	#CCP2,@#STMP3	;CONSTANT BAD D(B)
5321	024230	012737	024616	001246			MOV	#CCP2,@#STMP6	
5322	024236	012737	024576	001242		CCER22:	MOV	#CCP0,@#STMP4	
5323	024244	012737	024566	001244			MOV	#CCDAT0,@#STMP5	
5324	024252	104002				1\$:	ERROR	2	
5325	024254	000137	024726				JMP	@#CCDONE	
5326	024260	012737	024606	001240		CCER3:	MOV	#CCP1,@#STMP3	
5327	024266	012737	024666	001246			MOV	#CCP7,@#STMP6	
5328	024274	000741					BR	CCER50	
5329	024276	012737	024606	001240		CCER4:	MOV	#CCP1,@#STMP3	;CONSTANT BAD D(G)
5330	024304	012737	024666	001246			MOV	#CCP7,@#STMP6	
5331	024312	012737	024576	001242		CCER44:	MOV	#CCP0,@#STMP4	
5332	024320	012737	024566	001244			MOV	#CCDAT0,@#STMP5	
5333	024326	104002				1\$:	ERROR	2	
5334	024330	000137	024726				JMP	@#CCDONE	
5335	024334	012737	024656	001240		CCER5:	MOV	#CCP6,@#STMP3	;DATA ERROR F
5336	024342	012737	024656	001246			MOV	#CCP6,@#STMP6	
5337	024350	012737	024576	001242		CCER55:	MOV	#CCP0,@#STMP4	
5338	024356	012737	024566	001244			MOV	#CCDAT0,@#STMP5	
5339	024364	104002				1\$:	ERROR	2	
5340	024366	000557					BR	CCDONE	
5341	024370	012737	024656	001240		CCER6:	MOV	#CCP6,@#STMP3	;CONSTANT BAD F(B)
5342	024376	012737	024656	001246			MOV	#CCP6,@#STMP6	
5343	024404	012737	024576	001242			MOV	#CCP0,@#STMP4	
5344	024412	012737	024566	001244			MOV	#CCDAT0,@#STMP5	
5345	024420	104002				1\$:	ERROR	2	
5346	024422	000541					BR	CCDONE	
5347	024424	012737	024646	001240		CCER7:	MOV	#CCP5,@#STMP3	;DATA ERROR F
5348	024432	012737	024676	001246			MOV	#CCP10,@#STMP6	
5349	024440	000743					BR	CCER55	
5350	024442	012737	024646	001240		CCER8:	MOV	#CCP5,@#STMP3	;CONSTANT BAD F(G)
5351	024450	012737	024676	001246			MOV	#CCP10,@#STMP6	
5352	024456	012737	024566	001244			MOV	#CCDAT0,@#STMP5	
5353	024464	012737	024576	001242			MOV	#CCP0,@#STMP4	
5354	024472	104002				1\$:	ERROR	2	
5355	024474	000514					BR	CCDONE	
5356	024476	012737	024626	001240		CCER10:	MOV	#CCP3,@#STMP3	;DATA ERROR D
5357	024504	012737	024706	001246			MOV	#CCP11,@#STMP6	
5358	024512	000632					BR	CCER50	

```

5359 024514 012737 024626 001240 CCER11: MOV #CCP3,@#STMP3 ;CONSTANT BAD D(G)
5360 024522 012737 024706 001246 MOV #CCP11,@#STMP6
5361 024530 000670 BR CCER44
5362 024532 012737 024636 001240 CCER12: MOV #CCP4,@#STMP3 ;DATA ERROR D
5363 024540 012737 024636 001246 MOV #CCP4,@#STMP6
5364 024546 000614 BR CCER50
5365 024550 012737 024636 001240 CCER13: MOV #CCP4,@#STMP3 ;CONSTANT BAD D(B)
5366 024556 012737 024636 001246 MOV #CCP4,@#STMP6
5367 024564 000624 BR CCER22
5368 024566 000000 CCDATO: 0
5369 024570 000000 0
5370 024572 000000 0
5371 024574 000000 0
5372 024576 000200 CCP0: 200 ;E(AC)=1
5373 024600 000000 0
5374 024602 000000 0
5375 024604 000000 0
5376 024606 016200 CCP1: 16200 ;E(FSRC)=E(AC)+56=57
5377 024610 000000 0 ; =71(OCT)
5378 024612 000000 0
5379 024614 000000 0
5380 024616 016400 CCP2: 16400 ;E(FSRC)=E(AC)+57=58
5381 024620 000000 0 ; =72(OCT)
5382 024622 000000 0
5383 024624 000000 0
5384 024626 000400 CCP3: 400 ;E(FSRC)=E(AC)+1=2
5385 024630 000000 0
5386 024632 000000 0
5387 024634 000000 0
5388 024636 031200 CCP4: 31200 ;E(FSRC)-E(AC)+100=101=145(OCT)
5389 024640 000000 0
5390 024642 000000 0
5391 024644 000000 0
5392 024646 006200 CCP5: 6200 ;E(FSRC)=E(AC)+24=25=31(OCT)
5393 024650 000000 0
5394 024652 000000 0
5395 024654 000000 0
5396 024656 006400 CCP6: 6400 ;E(FSRC)=E(AC)+25=26=32(OCT)
5397 024660 000000 0
5398 024662 000000 0
5399 024664 000000 0
5400 024666 016200 CCP7: 16200 ;CCP1 RES
5401 024670 000000 0
5402 024672 000000 0
5403 024674 000001 1
5404 024676 006200 CCP10: 6200 ;CCP5 RES
5405 024700 000001 1
5406 024702 000000 0
5407 024704 000000 0
5408 024706 000500 CCP11: 500 ;CCP3 RES
5409 024710 000000 0
5410 024712 000000 0
5411 024714 000000 0
5412 024716 000200 CCP12: 200 ;BAD CONSTANT
5413 024720 000000 0 ;RES CCP2,CCP4
5414 024722 000000 0
  
```

5415 024724 000000  
 5416  
 5417 024726  
 5418 024726 104413  
 5419  
 5420  
 5421  
 5422  
 5423  
 5424  
 5425  
 5426  
 5427  
 5428  
 5429  
 5430  
 5431  
 5432  
 5433  
 5434  
 5435 024730 000004  
 5436  
 5437 024732  
 5438 024732 104414  
 5439 024734 012704 003200  
 5440 024740 170104  
 5441 024742 012737 025570 000244  
 5442 024750 012737 024770 001236  
 5443  
 5444 024756 012700 026132  
 5445 024762 172410  
 5446 024764 012700 026122  
 5447 024770 172010  
 5448 024772 170205  
 5449 024774 012700 026102  
 5450 025000 174010  
 5451 025002 012701 026132  
 5452 025006 012702 000004  
 5453 025012 022021  
 5454 025014 001402  
 5455 025016 000137 025630  
 5456 025022 077205  
 5457  
 5458 025024 020405  
 5459 025026 001402  
 5460 025030 000137 025570  
 5461  
 5462 025034  
 5463 025034 104414  
 5464 025036 012704 003200  
 5465 025042 170104  
 5466 025044 012737 025064 001236  
 5467 025052 012700 026152  
 5468 025056 172410  
 5469 025060 012700 026122  
 5470 025064 172010

```

0
CCDONE:
RSETUP          ;GO INITIALIZE THE FPS AND STACK; AND
                ;SEE IF THE USER HAS EXPRESSED
                ;THE DESIRE TO CHANGE THE SOFTWARE
                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                ;THE USER TYPED CONTROL G?).

:*****
:*TEST 33      ADDF AND ADDD WITH E(AC) GREATER THAN E(FSRC) TEST
:*
:*THIS IS A TEST OF THE ADDD AND ADDF
:*INSTRUCTIONS AND THE ALIGN FSRC ALGORITHM
:*FLOWS. FIRST THE CONSTANT USED IS CHECKED.
:*THEN SIMPLE AND WORST CASE ALIGNMENT
:*SITUATIONS ARE TRIED. NOTE E(AC)
:*IS GREATER THAN E(FSRC).
:*
:*****
TST33: SCOPE
;EXPONENT DIFFERENCE=57=71 (OCT) FD=1
BB1:
    LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
    MOV            #3200,R4      ;SET FIV FIV, AND FD
    LDFPS         R4
    MOV            #BBERO,@#FPVECT ;SET UP FOR ERROR
    MOV            #BB2,@#STMP2  ;IN CASE THE OVER\
                                ;UNDER FLOWS FAIL.
    MOV            #BBPAT2,R0    ;SET ACO OPERAND.
    LDD            (R0),ACO
    MOV            #BBPAT1,R0    ;FSRC
    BB2: ADDD      (R0),ACO      ;TEST INSTRUCTION
    STFPS         R5
    BB3: MOV       #BBDAT0,R0    ;GET THE RESULT
    STD           ACO,(R0)
    MOV            #BBPAT2,R1    ;RESULT CORRECT?
    MOV            #4,R2
    BB4: CMP       (R0)+,(R1)+
    BEQ           BB5
    JMP            @#BBER1       ;DATA ERROR D
    BB5: SOB      R2,BB4
                                ;WAS FPS CORRECT?
    CMP           R4,R5
    BEQ           BB6
    JMP            @#BBERO       ;FPS ERROR
;EXPONENT DIFFERENCE=56=70 (OCT) FD=1
BB6:
    LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
    MOV            #3200,R4      ;SE FIV,FIV, AND FD
    LDFPS         R4
    MOV            #BB7,@#STMP2
    MOV            #BBPAT4,R0    ;SET ACO OPERAND
    LDD            (R0),ACO
    MOV            #BBPAT1,R0    ;FSRC
    BB7: ADDD      (R0),ACO      ;TEST INSTRUCTION
    
```

5471	025066	170205			STFPS	R5		:GET FPS
5472	025070	012700	026102		MOV	#BBDATO,R0		:GET THE RESULT
5473	025074	174010			STD	ACO,(R0)		
5474	025076	012701	026212		MOV	#BBP10,R1		:IS IT CORRECT
5475	025102	012702	000004		MOV	#4,R2		
5476	025106	022021		BB10:	CMP	(R0)+,(R1)+		
5477	025110	001415			BEQ	BB13		
5478	025112	012700	026102		MOV	#BBDATO,R0		:DID A BAD
5479	025116	012701	026152		MOV	#BBPAT4,R1		:CONSTANT (NOT 57)
5480	025122	012702	000004		MOV	#4,R2		:GET GENERATED
5481	025126	022021		BB11:	CMP	(R0)+,(R1)+		:FOR THE ALIGNMENT
5482	025130	001402			BEQ	BB12		:FLOWS?
5483	025132	000137	025666		JMP	@#BBER2		:DATA ERROR.D
5484	025136	077205		BB12:	SOB	R2,BB11		
5485	025140	000137	025704		JMP	@#BBER3		:BAD CONSTANT.D
5486	025144	077220		BB13:	SOB	R2,BB10		
5487	025146	020405			CMP	R4,R5		:FPS CORRECT?
5488	025150	001402			BEQ	BB14		
5489	025152	000137	025570		JMP	@#BBERO		:BAD FPS.
5490								:EXPONENT DIFFERENCE=25=31 (OCT) FD=0
5491	025156			BB14:	LPERR			:SET UP THE LOOP ON ERROR ADDRESS.
5492	025156	104414			MOV	#BB15,@#STMP2		
5493	025160	012737	025206	001236	MOV	#BBPAT0,R0		:SET UP ACO OPERAND
5494	025166	012700	026112		LDD	(R0),ACO		
5495	025172	172410			MOV	#3000,R4		:SET FIV AND FIV
5496	025174	012704	003000					:CLEAR FD
5497								
5498	025200	170104			LDFPS	R4		
5499	025202	012700	026122		MOV	#BBPAT1,R0		:FSRC
5500	025206	172010		BB15:	ADDF	(R0),ACO		:TEST INSTRUCTION
5501	025210	170205			STFPS	R5		
5502	025212	170011			SETD			:REENDED DOUBLE MODE.
5503	025214	012700	026102		MOV	#BBDATO,R0		:GET THE RESULT
5504	025220	174010			STD	ACO,(R0)		
5505	025222	012701	026112		MOV	#BBPAT0,R1		:IS THE RESULT
5506	025226	012702	000002		MOV	#2,R2		:CORRECT?
5507	025232	022021		BB16:	CMP	(R0)+,(R1)+		
5508	025234	001402			BEQ	BB17		
5509	025236	000137	025740		JMP	@#BBER4		:DATA ERROR F
5510	025242	077205		BB17:	SOB	R2,BB16		
5511	025244	020405			CMP	R4,R5		:IS FPS CORRECT?
5512	025246	001402			BEQ	BB20		
5513	025250	000137	025610		JMP	@#BBER10		:FPS ERROR.
5514								:EXPONENT DIFFERENCE=24=30 (OCT)
5515	025254			BB20:	LPERR			:SET UP THE LOOP ON ERROR ADDRESS.
5516	025254	104414			MOV	#BB21,@#STMP2		
5517	025256	012737	025304	001236	MOV	#BBPAT3,R0		:SET UP ACO OPERAND.
5518	025264	012700	026142		LDD	(R0),ACO		
5519	025270	172410			MOV	#3000,R4		:SET FIU,FIV. CLEAR FD.
5520	025272	012704	003000					
5521	025276	170104			LDFPS	R4		
5522	025300	012700	026122		MOV	#BBPAT1,R0		:FSRC
5523	025304	172010		BB21:	ADDF	(R0),ACO		:TEST INSTRUCTION
5524	025306	170205			STFPS	R5		
5525	025310	170011			SETD			:REENTER DOUBLE MODE
5526	025312	012700	026102		MOV	#BBDATO,R0		:GET THE RESULT



```

5527 025316 17401C          STD      AC0,(R0)
5528 025320 012701 026202    MOV      #BBP7,R1          ;IS THE RESULT CORRECT?
5529 025324 012702 000002    MOV      #2,R2
5530 025330 022021          BB22:   CMP      (R0)+,(R1)+
5531 025332 001415          BEQ      BB25
5532 025334 012700 026102    MOV      #BBDAT0,R0       ;WAS A BAD CONSTANT
5533 025340 012701 026142    MOV      #BBPAT3,R1       ;USED (NOT ?5) IN
5534 025344 012702 000002    MOV      #2,R2           ;THE ALLIGN FLOWS?
5535 025350 022021          BB23:   CMP      (R0)+,(R1)+
5536 025352 001402          BEQ      BB24
5537 025354 000137 025774    JMP      @#BBER5          ;DATA ERROR F
5538 025360 077205          BB24:   SOB      R2,BB23
5539 025362 000137 026012    JMP      @#BBER6          ;BAD CONSTANT F
5540 025366 077220          BB25:   SOB      R2,BB22
5541 025370 020405          CMP      R4,R5
5542 025372 001402          BEQ      BB26
5543 025374 000137 025610    JMP      @#BBER10        ;BAD FPS.
5544                                ;EXPONENT DIFFERENCE=1
5545 025400          BB26:   LPERR
5546 025400 104414          MOV      #BB27,@#TMP2    ;SET UP THE LOOP ON ERROR ADDRESS.
5547 025402 012737 025430 001236    MOV      #3200,R4
5548 025410 012704 003200    MOV      R4
5549 025414 170104          LDFPS   R4               ;SET UP AC0 OPERAND
5550 025416 012700 026162    MOV      #BBPAT5,R0
5551 025422 172410          LDD     (R0),AC0
5552 025424 012700 026122    MOV      #BBPAT1,R0      ;FSRC
5553 025430 172010          BB27:   ADDD   (R0),AC0    ;TEST INSTRUCTION
5554 025432 170205          STFPS  R5
5555 025434 012700 026102    MOV      #BBDAT0,R0      ;GET THE RESULT.
5556 025440 174010          STD     AC0,(R0)
5557 025442 012701 026222    MOV      #BBP11,R1       ;IS IT CORRECT?
5558 025446 012702 000004    MOV      #4,R2
5559 025452 022021          BB30:   CMP      (R0)+,(R1)+
5560 025454 001402          BEQ      BB31
5561 025456 000137 026046    JMP      @#BBER7          ;DATA ERRCR D
5562 025462 077205          BB31:   SOB      R2,BB30
5563 025464 020405          CMP      R4,R5           ;IS FPS CORRECT
5564 025466 001402          BEQ      BB32
5565 025470 000137 025570    JMP      @#BBER0
5566                                ;EXPONENT DIFFERENCE=100=144 (OCT)
5567 025474          BB32:   LPERR
5568 025474 104414          MOV      #BB33,@#TMP2    ;SET UP THE LOOP ON ERROR ADDRESS.
5569 025476 012737 025524 001236    MOV      #3200,R4
5570 025504 012704 003200    MOV      R4
5571 025510 170104          LDFPS   R4               ;SET FIV,FIV AND FD
5572 025512 012700 026172    MOV      #BBPAT6,R0      ;SET UP AC0 OPERAND.
5573 025516 172410          LDD     (R0),AC0
5574 025520 012700 026122    MOV      #BBPAT1,R0      ;FSRC
5575 025524 172010          BB33:   ADDD   (R0),AC0    ;TEST INSTRUCTION
5576 025526 170205          STFPS  R5
5577 025530 012700 026102    MOV      #BBDAT0,R0      ;GET THE RESULT
5578 025534 174010          STD     AC0,(R0)
5579 025536 012701 026172    MOV      #BBPAT6,R1       ;IS IT CORRECT
5580 025542 012702 000004    MOV      #4,R2
5581 025546 022021          BB34:   CMP      (R0)+,(R1)+
5582 025550 001402          BEQ      BB35
    
```

```

5583 025552 000137 026064          JMP      @#BBER8          ;DATA ERROR D
5584 025556 077205          BB35:   SOB      R2, BB34
5585 025560 020405          CMP      R4, R5          ;IS FPS CORRECT
5586 025562 001002          BNE      BBER0
5587 025564 000167 000442          JMP      BBDONE
5588 025570 010437 001242          BBER0:  MOV      R4, @#$TMP4    ;FPS ERROR D
5589 025574 010537 001240          MOV      R5, @#$TMP3
5590 025600 104001          1$:     ERROR      1
5591 025602 104413          RSETUP
5592                                     ;GO INITIALIZE THE FPS AND STACK; AND
5593                                     ;SEE IF THE USER HAS EXPRESSED
5594                                     ;THE DESIRE TO CHANGE THE SOFTWARE
5595                                     ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
5596                                     ;THE USER TYPED CONTROL G?).
5596 025604 000137 026232          JMP      @#BBDONE
5597 025610 010437 001242          BBER10: MOV      R4, @#$TMP4    ;FPS ERROR F
5598 025614 010537 001240          MOV      R5, @#$TMP3
5599 025620 104002          1$:     ERROR      2
5600 025622 104413          RSETUP
5601                                     ;GO INITIALIZE THE FPS AND STACK; AND
5602                                     ;SEE IF THE USER HAS EXPRESSED
5603                                     ;THE DESIRE TO CHANGE THE SOFTWARE
5604                                     ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
5605                                     ;THE USER TYPED CONTROL G?).
5605 025624 000137 026232          JMP      @#BBDONE
5606 025630 012737 026132 001242          BBER1:  MOV      #BBPAT2, @#$TMP4 ;DATA ERROR D
5607 025636 012737 026132 001246          MOV      #BBPAT2, @#$TMP6
5608 025644 012737 026122 001240          BBER11: MOV      #BBPAT1, @#$TMP3
5609 025652 012737 026102 001244          MOV      #BBDATO, @#$TMP5
5610 025660 104002          1$:     ERROR      2
5611 025662 000137 026232          JMP      @#BBDONE
5612 025666 012737 026152 001242          BBER2:  MOV      #BBPAT4, @#$TMP4
5613 025674 012737 026212 001246          MOV      #BBP10, @#$TMP6
5614 025702 000760          BR
5615 025704 012737 026152 001242          BBER3:  MOV      #BBPAT4, @#$TMP4 ;BAD CONSTANT D
5616 025712 012737 026212 001246          MOV      #BBP10, @#$TMP6
5617 025720 012737 026122 001240          MOV      #BBPAT1, @#$TMP3
5618 025726 012737 026102 001244          MOV      #BBDATO, @#$TMP5
5619 025734 104002          1$:     ERROR      2
5620 025736 000535          BR      BBDONE
5621 025740 012737 026112 001242          BBER4:  MOV      #BBPAT0, @#$TMP4 ;DATA ERROR F
5622 025746 012737 026112 001246          MOV      #BBPAT0, @#$TMP6
5623 025754 012737 026122 001240          BBER40: MOV      #BBPAT1, @#$TMP3
5624 025762 012737 026102 001244          MOV      #BBDATO, @#$TMP5
5625 025770 104002          1$:     ERROR      2
5626 025772 000517          BR      BBDONE
5627 025774 012737 026142 001242          BBER5:  MOV      #BBPAT3, @#$TMP4
5628 026002 012737 026202 001246          MOV      #BBP7, @#$TMP6
5629 026010 000761          BR      BBER40
5630 026012 012737 026142 001242          BBER6:  MOV      #BBPAT3, @#$TMP4 ;CONSTANT ERROR F
5631 026020 012737 026202 001246          MOV      #BBP7, @#$TMP6
5632 026026 012737 026122 001240          MOV      #BBPAT1, @#$TMP3
5633 026034 012737 026102 001244          MOV      #BBDATO, @#$TMP5
5634 026042 104002          1$:     ERROR      2
5635 026044 000472          BR      BBDONE
5636 026046 012737 026162 001242          BBER7:  MOV      #BBPAT5, @#$TMP4
5637 026054 012737 026122 001246          MOV      #BBPAT11, @#$TMP6
5638 026062 000670          BR      BBER11

```

```

5639 026064 012737 026172 001242 BBER8: MOV #BBPAT6,@#$TMP4
5640 026072 012737 026172 001246 MOV #BBPAT6,@#$TMP6
5641 026100 000661 BR BBER11
5642 026102 000000 BBDATO: 0
5643 026104 000000 0
5644 026106 000000 0
5645 026110 000000 0
5646 026112 006400 BBPAT0: 6400 ;F(AC)=E(FSRC)+25=26
5647 026114 000000 0 ; =32(OCT)
5648 026116 000000 0
5649 026120 000000 0
5650 026122 000200 BBPAT1: 200
5651 026124 000000 0 ;E(FSRC)=1
5652 026126 000000 0
5653 026130 000000 0
5654 026132 016400 BBPAT2: 16400
5655 026134 000000 0 ;E(AC)=E(FSRC)+57=58
5656 026136 000000 0 ; =72(OCT)
5657 026140 000000 0
5658 026142 006200 BBPAT3: 6200 ;E(AC)=E(FSRC)+24=25
5659 026144 000000 0 ; =31(OCT)
5660 026146 000000 0
5661 026150 000000 0
5662 026152 016200 BBPAT4: 16200 ;E(AC)=E(FSRC)+56=57
5663 026154 000000 0 ; =71(OCT)
5664 026156 000000 0
5665 026160 000000 0
5666 026162 000400 BBPAT5: 400 ;E(AC)=E(FSRC)+1=2
5667 026164 000000 0
5668 026166 000000 0
5669 026170 000000 0
5670 026172 031200 BBPAT6: 31200 ;E(AC)=E(FSRC)+100=101
5671 026174 000000 0 ; =145(OCT)
5672 026176 000000 0
5673 026200 000000 0
5674 026202 006200 BBP7: 6200 ;BBPAT3 RES
5675 026204 000001 1
5676 026206 000000 0
5677 026210 000000 0
5678 026212 016200 BBP10: 16200 ;BBPAT4 RES
5679 026214 000000 0
5680 026216 000000 0
5681 026220 000001 1
5682 026222 000500 BBP11: 500 ;BBPAT5 RES
5683 026224 000000 0
5684 026226 000000 0
5685 026230 000000 0
5686 026232 BBDONE:
5687 026232 104413 RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
5688 ;SEE IF THE USER HAS EXPRESSED
5689 ;THE DESIRE TO CHANGE THE SOFTWARE
5690 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
5691 ;THE USER TYPED CONTROL G?).
5692 ;*****
5693 ;*TEST 34 ADD WITH NEGATIVE OPRANDS TEST
5694 ;*

```

```

5695 ;*THIS IS A TEST OF THE ADDD INSTRUCTION
5696 ;*WITH NEGATIVE OPERANDS. EVERY COMBINATION OF
5697 ;*OPERAND SIGNS IS TRIED.
5698 ;*
5699 ;*****
5700 026234 000004 TST34: SCOPE
5701 ;BOTH OPERANDS NEGATIVE
5702 026236 DD1:
5703 026236 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
5704 026240 012704 003200 MOV #3200,R4 ;SET FIO, FIV, AND FD
5705 026244 170104 LDFPS R4
5706 026246 012737 026266 001236 MOV #DD2,@$TMP2
5707 026254 012700 030116 MOV #DDP1,R0 ;SET ACO OPERAND
5708 026260 172410 LDD (R0),ACO
5709 026262 012700 030116 MOV #DDP1,R0 ;ESRC
5710 026266 172010 DD2: ADDD (R0),ACO ;TEST INSTRUCTION
5711 026270 170205 STFPS R5 ;GET FPS
5712 026272 012700 030076 MOV #DDDATO,R0 ;GET THE RESULT
5713 026276 174010 STD ACO,(R0)
5714 026300 012701 030216 MOV #DDP9,R1 ;IS IT CORRECT
5715 026304 012702 000004 DD3: MOV #4,R2
5716 026310 022021 CMP (R0)+,(R1)+
5717 026312 001415 BEQ DD6
5718 026314 012700 030076 MOV #DDDATO,R0 ;DID A ADD-SUB
5719 026320 012701 030146 MOV #DDP4,R1 ;FLOW A FAILURE
5720 026324 012702 000004 DD4: MOV #4,R2
5721 026330 022021 CMP (R0)+,(R1)+
5722 026332 001402 BEQ DD5 ;216,442,500
5723 026334 000137 027326 JMP @DDDER1 ;DATA ERROR,D
5724 026340 077205 DD5: SOB R2,DD4
5725 026342 000137 027364 JMP @DDDER2 ;FLOW FAILURE,D
5726 026346 077220 DD6: SOB R2,DD3
5727 026350 052704 000010 BIS #10,R4
5728 026354 020405 CMP R4,R5 ;FPS CORRECT?
5729 026356 001402 BEQ DD7
5730 026360 000137 027310 JMP @DDDERO ;BAD,FPS
5731 ;AC POS FSRC NEG AC=-FSRC
5732 026364 DD7:
5733 026364 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
5734 026366 012704 003200 MOV #3200,R4 ;SET FIO, FIV, AND FD
5735 026372 170104 LDFPS R4
5736 026374 012737 026414 001236 MOV #DD8,@$TMP2
5737 026402 012700 030126 MOV #DDP2,R0 ;SET ACO OPERAND
5738 026406 172410 LDD (R0),ACO
5739 026410 012700 030116 MOV #DDP1,R0 ;FSPC
5740 026414 172010 DD8: ADDD (R0),ACO ;TEST INSTRUCTION
5741 026416 170205 STFPS R5 ;GET FPS
5742 026420 012700 030076 MOV #DDDATO,R0 ;GET THE RESULT
5743 026424 174010 STD ACO,(R0)
5744 026426 012701 030106 MOV #DDP0,R1 ;IS IT CORRECT
5745 026432 012702 000004 DD10: MOV #4,R2
5746 026436 022021 CMP (R0)+,(R1)+
5747 026440 001402 BEQ DD11
5748 026442 000137 027422 JMP @DDDER3 ;FLOW FAILURE
5749 026446 077205 DD11: SOB R2,DD10
5750 026450 052704 000004 BIS #4,R4
    
```

5751	026454	020405			CMP	R4,R5		;FPS CORRECT?
5752	026456	001402			BEQ	DD12		
5753	026460	000137	027310		JMP	@#DDERO		;BAD FPS
5754					;AC NEG	FSRC POS		AC=-FSRC
5755	026464				DD12:			
5756	026464	104414			LPERR			;SET UP THE LOOP ON ERROR ADDRESS.
5757	026466	012704	003200		MOV	#3200,R4		;SET FIU, FIV, AND FD
5758	026472	170104			LDFPS	R4		
5759	026474	012737	026514	001236	MOV	#DD13,@#STMP2		
5760	026502	012700	030116		MOV	#DDP1,R0		;SET ACO OPERAND
5761	026506	172410			LDD	(R0),ACO		
5762	026510	012700	030126		MOV	#DDP2,R0		;FSRC
5763	026514	172010			DD13:	ADDD	(R0),ACO	;TEST INSTRUCTION
5764	026516	170205			STFPS	R5		;GET FPS
5765	026520	012700	030076		MOV	#DDDATO,R0		;GET THE RESULT
5766	026524	174010			STD	ACO,(R0)		
5767	026526	012701	030106		MOV	#DDP0,R1		;IS IT CORRECT
5768	026532	012702	000004		MOV	#4,R2		
5769	026536	022021			DD14:	CMP	(R0)+,(R1)+	
5770	026540	001402			BEQ	DD15		
5771	026542	000137	027460		JMP	@#DDER4		;FLOW FAILURE 216,440,121
5772	026546	077205			DD15:	SOB	R2,DD14	
5773	026550	052704	000004		BIS	#4,R4		
5774	026554	020405			CMP	R4,R5		;EPS CORRECT?
5775	026556	001402			BEQ	DD16		
5776	026560	000137	027310		JMP	@#DDERO		;BAD FPS
5777					;ACO POC	FSRC NEG		/AC/ > /FSRC/
5778	026564				DD16:			
5779	026564	104414			LPERR			;SET UP THE LOOP ON ERROR ADDRESS.
5780	026566	012704	003200		MOV	#3200,R4		;SET FIV, FIV AND FD
5781	026572	170104			LDFPS	R4		
5782	026574	012737	026614	001236	MOV	#DD17,@#STMP2		
5783	026602	012700	030136		MOV	#DDP3,R0		;SET ACO OPERAND
5784	026606	172410			LDD	(R0),ACO		
5785	026610	012700	030166		MOV	#DDP6,R0		;ESPC
5786	026614	172010			DD17:	ADDD	(R0),ACO	;TEST INSTRUCTION
5787	026616	170205			STFPS	R5		;GET FPS
5788	026620	012700	030076		MOV	#DDDATO,R0		;GET THE RESULT
5789	026624	174010			STD	ACO,(R0)		
5790	026626	012701	030176		MOV	#DDP7,R1		;IS IT CORRECT
5791	026632	012702	000004		MOV	#4,R2		
5792	026636	022021			DD18:	CMP	(R0)+,(R1)+	
5793	026640	001415			BEQ	DD21		
5794	026642	012700	030076		MOV	#DDDATO,R0		;FLOWS FAILURE
5795	026646	012701	030206		MOV	#DDP8,R1		;216,440,101
5796	026652	012702	000004		MOV	#4,R2		;GET GENERATED
5797	026656	022021			DD19:	CMP	(R0)+,(R1)+	
5798	026660	001402			BEQ	DD20		
5799	026662	000137	027516		JMP	@#DDERS5		;DATA ERROR.
5800	026666	077205			DD20:	SOB	R2,DD19	
5801	026670	000137	027554		JMP	@#DDER6		
5802	026674	077220			DD21:	SOB	R2,DD18	
5803	026676	020405			CMP	R4,R5		;EPS CORRECT?
5804	026700	001402			BEQ	DD22		
5805	026702	000137	027310		JMP	@#DDERO		;BAD FPS
5806					;AC NEG	FSRC POS		/FSRC/ > /AC/

```

5807 026706 DD22: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
5808 026706 104414 MOV #3200,R4 ;SET FIO,FIV, AND FD
5809 026710 012704 003200 LDFPS R4
5810 026714 170104 MOV #DD23,@#$TMP2
5811 026716 012737 026736 001236 MOV #DDP6,R0 ;SET ACO OPERAND
5812 026724 012700 030166 LDD (R0),ACO
5813 026730 172410 MOV #DDP3,R0 ;FSPC
5814 026732 012700 030136 DD23: ADDD (R0),ACO ;TEST INSTRUCTION
5815 026736 172010 STFPS R5 ;GET FPS
5816 026740 170205 MOV #DDDATO,R0 ;GET THE RESULT
5817 026742 012700 030076 STD ACO,(R0)
5818 026746 174010 MOV #DDP7,R1 ;IS IT CORRECT?
5819 026750 012701 030176 MOV #4,R2
5820 026754 012702 000004 DD24: CMP (R0)+,(R1)+
5821 026760 022021 BEQ DD27
5822 026762 001415 MOV #DDDATO,R0 ;FLO,S FAILURE
5823 026764 012700 030076 MOV #DDP8,R1 ;CONSTANT (NOT 57)
5824 026770 012701 030206 MOV #4,R2 ;216,042,101
5825 026774 012702 000004 DD25: CMP (R0),(R1)
5826 027000 021011 BEQ DD26
5827 027002 001402 DD26: JMP @#DDER7 ;DATA ERROR.
5828 027004 000137 027612 SOB R2,DD25
5829 027010 077205 DD27: JMP @#DDER8
5830 027012 000137 027650 SOB R2,DD24
5831 027016 077220 CMP R4,R5 ;FPS CORRECT?
5832 027020 020405 BEQ DD30
5833 027022 001402 DD30: JMP @#DDERO ;BAD FPS
5834 027024 000137 027310 ;ACO POS FSRC NEG /AC/</FRSRC/
5835 027030 DD30: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
5836 027030 104414 MOV #3200,R4 ;SET FIO,FIV,AND FD
5837 027030 012704 003200 LDFPS R4
5838 027032 012704 003200 MOV #DD31,@#$TMP2
5839 027036 170104 MOV #DDP4,R0 ;SET ACO OPERAND
5840 027040 012737 027060 001236 LDD (R0),ACO
5841 027046 012700 030146 MOV #DDP5,R0 ;FSPC
5842 027052 172410 DD31: ADDD (R0),ACO ;TEST INSTRUCTION
5843 027054 012700 030156 STFPS R5 ;GET FPS
5844 027060 172010 MOV #DDDATO,R0 ;GET THE RESULT
5845 027062 170205 STD ACO,(R0)
5846 027064 012700 030076 MOV #DDP8,R1 ;IS IT CORRECT
5847 027070 174010 MOV #4,R2
5848 027072 012701 030206 DD32: CMP (R0)+,(R1)+
5849 027076 012702 000004 BEQ DD35 ;ADD-SUB
5850 027102 022021 MOV #DDDATO,R0 ;FLOWAS FAILURE
5851 027104 001415 MOV #DDP7,R1 ;CON 216 N440 NOT 141
5852 027106 012700 030076 MOV #4,R2 ;GET GENERATED
5853 027112 012701 030176 DD33: CMP (R0)+,(R1)+ ;FOR THE ALLIGNMENT
5854 027116 012702 000004 BEQ DD34 ;FLOWS?
5855 027122 022021 DD34: JMP @#DDER9 ;DATA ERROR, D
5856 027124 001402 DD35: SOB R2,DD33
5857 027126 000137 027706 JMP @#DDER10
5858 027132 077205 DD35: SOB R2,DD32
5859 027134 000137 027744 BIS #10,R4
5860 027140 077220 CMP R4,R5 ;FPS CORRECT?
5861 027142 052704 000010
5862 027146 020405

```

```

5863 027150 001402          BEQ      #DD36
5864 027152 000137 027310  JMP      @#DDERO          ;BAD FPS
5865          ;ACO NEG          FSRC      POS      /FSRC/</AC/
5866 027156          DD36:
5867 027156 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
5868 027160 012704 003200  MOV      #3200,R4          ;SET FIO, FIV, AND FD
5869 027164 170104          LDFPS        R4
5870 027166 012737 027206 001236  MOV      #DD37,@#STMP2
5871 027174 012700 030156  MOV      #DDP5,R0          ;SET ACO OPERAND
5872 027200 172410          LDD          (R0),ACO
5873 027202 012700 030146  MOV      #DDP4,R0          ;FSPC
5874 027206 172010  DD37:  ADDD   (R0),ACO          ;TEST INSTRUCTION
5875 027210 170205          STFPS        R5          ;GET FPS
5876 027212 012700 030076  MOV      #DDDATO,R0        ;GET THE RESULT
5877 027216 174010          STD          ACO,(R0)
5878 027220 012701 030206  MOV      #DDP8,R1          ;IS IT CORRECT
5879 027224 012702 000004  MOV      #4,R2
5880 027230 022021  DD38:  CMP      (R0)+,(R1)+
5881 027232 001415          BEQ          DD41
5882 027234 012700 030076  MOV      #DDDATO,R0        ;ADD SUB
5883 027240 012701 030176  MOV      #DDP7,R1          ;FLOWS FAILURES
5884 027244 012702 000004  MOV      #4,R2          ;GET 216,042,141
5885 027250 022021  DD39:  CMP      (R0)+,(R1)+        ;FOR THE ALLIGNMENT
5886 027252 001402          BEQ          DD40          ;FLOWS?
5887 027254 000137 030002  JMP      @#DDER11        ;DATA ERROR. D
5888 027260 077205  DD40:  SOB      R2,DD39
5889 027262 000137 030040  JMP      @#DDER12        ;BAD CONSTANT.D
5890 027266 077220  DD41:  SOB      R2,DD38
5891 027270 052704 000010  BIS      #10,R4
5892 027274 020405          CMP          R4,R5          ;FPS CORRECT?
5893 027276 001402          BEQ          DD42
5894 027300 000137 027310  JMP      @#DDERO          ;BAD FPS
5895 027304 000137 030226  DD42:  JMP      @#DDDONE
5896 027310 010437 001242  DDERO:  MOV      R4,@#STMP4        ;FPS ERROR
5897 027314 010537 001240  MOV      R5,@#STMP3
5898 027320 104002          1$:      ERROR          2
5899 027322 000137 030226  JMP      @#DDDONE
5900 027326          DDER1:
5901 027326 012737 030116 001240  MOV      #DDP1,@#STMP3
5902 027334 012737 030116 001242  MOV      #DDP1,@#STMP4
5903 027342 012737 030076 001244  MOV      #DDDATO,@#STMP5
5904 027350 012737 030116 001246  MOV      #DDP1,@#STMP6
5905 027356 104002          1$:      ERROR          2
5906 027360 000137 030226  JMP      @#DDDONE
5907 027364          DDER2:
5908 027364 012737 030116 001240  MOV      #DDP1,@#STMP3
5909 027372 012737 030116 001242  MOV      #DDP1,@#STMP4
5910 027400 012737 030076 001244  MOV      #DDDATO,@#STMP5
5911 027406 012737 030216 001246  MOV      #DDP9,@#STMP6
5912 027414 104002          1$:      ERROR          2
5913 027416 000137 030226  JMP      @#DDDONE
5914 027422          DDER3:
5915 027422 012737 030116 001240  MOV      #DDP1,@#STMP3
5916 027430 012737 030126 001242  MOV      #DDP2,@#STMP4
5917 027436 012737 030076 001244  MOV      #DDDATO,@#STMP5
5918 027444 012737 030106 001246  MOV      #DDP0,@#STMP6
    
```

5919	027452	104002			1\$:	ERROR	2
5920	027454	000137	030226			JMP	@#DDDDONE
5921	027460				DDER4:		
5922	027460	012737	030126	001240		MOV	#DDP2,@#STMP3
5923	027466	012737	030116	001242		MOV	#DDP1,@#STMP4
5924	027474	012737	030076	001244		MOV	#DDDATO,@#STMP5
5925	027502	012737	030106	001246		MOV	#DDPO,@#STMP6
5926	027510	104002			1\$:	ERROR	2
5927	027512	000137	030226			JMP	@#DDDDONE
5928	027516				DDER5:		
5929	027516	012737	030166	001240		MOV	#DDP6,@#STMP3
5930	027524	012737	030136	001242		MOV	#DDP3,@#STMP4
5931	027532	012737	030076	001244		MOV	#DDDATO,@#STMP5
5932	027540	012737	030176	001246		MOV	#DDP7,@#STMP6
5933	027546	104002			1\$:	ERROR	2
5934	027550	000137	030226			JMP	@#DDDDONE
5935	027554				DDER6:		
5936	027554	012737	030166	001240		MOV	#DDP6,@#STMP3
5937	027562	012737	030136	001242		MOV	#DDP3,@#STMP4
5938	027570	012737	030076	001244		MOV	#DDDATO,@#STMP5
5939	027576	012737	030176	001246		MOV	#DDP7,@#STMP6
5940	027604	104002			1\$:	ERROR	2
5941	027606	000137	030226			JMP	@#DDDDONE
5942	027612				DDER7:		
5943	027612	012737	030136	001240		MOV	#DDP3,@#STMP3
5944	027620	012737	030166	001242		MOV	#DDP6,@#STMP4
5945	027626	012737	030076	001244		MOV	#DDDATO,@#STMP5
5946	027634	012737	030176	001246		MOV	#DDP7,@#STMP6
5947	027642	104002			1\$:	ERROR	2
5948	027644	000137	030226			JMP	@#DDDDONE
5949	027650				DDER8:		
5950	027650	012737	030136	001240		MOV	#DDP3,@#STMP3
5951	027656	012737	030166	001242		MOV	#DDP6,@#STMP4
5952	027664	012737	030076	001244		MOV	#DDDATO,@#STMP5
5953	027672	012737	030176	001246		MOV	#DDP7,@#STMP6
5954	027700	104002			1\$:	ERROR	2
5955	027702	000137	030226			JMP	@#DDDDONE
5956	027706				DDER9:		
5957	027706	012737	030156	001240		MOV	#DDP5,@#STMP3
5958	027714	012737	030146	001242		MOV	#DDP4,@#STMP4
5959	027722	012737	030076	001244		MOV	#DDDATO,@#STMP5
5960	027730	012737	030206	001246		MOV	#DDP8,@#STMP6
5961	027736	104002			1\$:	ERROR	2
5962	027740	000137	030226			JMP	@#DDDDONE
5963	027744				DDER10:		
5964	027744	012737	030156	001240		MOV	#DDP5,@#STMP3
5965	027752	012737	030146	001242		MOV	#DDP4,@#STMP4
5966	027760	012737	030076	001244		MOV	#DDDATO,@#STMP5
5967	027766	012737	030206	001246		MOV	#DDP8,@#STMP6
5968	027774	104002			1\$:	ERROR	2
5969	027776	000137	030226			JMP	@#DDDDONE
5970	030002				DDER11:		
5971	030002	012737	030146	001240		MOV	#DDP4,@#STMP3
5972	030010	012737	030156	001242		MOV	#DDP5,@#STMP4
5973	030016	012737	030076	001244		MOV	#DDDATO,@#STMP5
5974	030024	012737	030206	001246		MOV	#DDP8,@#STMP6



```

5975 030032 104002          1$:  ERROR 2
5976 030034 000137 030226      JMP  @#DDDDONE
5977 030040
5978 030040 012737 030146 001240  DDER12:  MOV  #DDP4,@#STMP3
5979 030046 012737 030156 001242      MOV  #DDP5,@#STMP4
5980 030054 012737 030076 001244      MOV  #DDDATO,@#STMP5
5981 030062 012737 030206 001246      MOV  #DDP8,@#STMP6
5982 030070 104002          1$:  ERROR 2
5983 030072 000137 030226      JMP  @#DDDDONE
5984 030076 000000
5985 030100 000000
5986 030102 000000
5987 030104 000000
5988 030106 000000
5989 030110 000000
5990 030112 000000
5991 030114 000000
5992 030116 100200      DDP1: 100200      ; -DDP2
5993 030120 000000
5994 030122 000000
5995 030124 000000
5996 030126 000200      DDP2: 200        ; -DDP1
5997 030130 000000
5998 030132 000000
5999 030134 000000
6000 030136 001100      DDP3: 1100       ; EXP=4
6001 030140 000000      ; FRAC=...110...
6002 030142 000000
6003 030144 000000
6004 030146 000600      DDP4: 600        ; EXP=3
6005 030150 000000      ; FRAC=...100...
6006 030152 000000
6007 030154 000000
6008 030156 101100      DDP5: 101100    ; -DDP3
6009 030160 000000
6010 030162 000000
6011 030164 000000
6012 030166 100600      DDP6: 100600    ; -DDP4
6013 030170 000000
6014 030172 000000
6015 030174 000000
6016 030176 001000      DDP7: 1000      ; DDP3+DDP6
6017 030200 000000
6018 030202 000000
6019 030204 000000
6020 030206 101000      DDP8: 101000    ; DDP5+DDP4
6021 030210 000000
6022 030212 000000
6023 030214 000000
6024 030216 100400      DDP9: 100400    ; DDP1+DDP1
6025 030220 000000
6026 030222 000000
6027 030224 000000
6028 030226 000005
6029
6030
DDDONE:  RESET
;*****
;*TEST 35  SUBD TEST
    
```

```

6031
6032      ;*
6033      ;* THIS IS A TEST OF THE SUBD INSTRUCTION.
6034      ;* BOTH A POSITIVE AND A NEGATIVE NUMBER
6035      ;* IS SUBTRACTED FROM IT SELF
6036      ;*
6037      ;*****
6037 030230 000004 TST35: SCOPE
6038      ; USE POSITIVE OPERANDS
6039 030232 EE1:
6040 030232 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
6041 030234 012704 003200 MOV #3200,R4 ;SET FIU, FIV, AND FD
6042 030240 170104 LDFPS R4
6043 030242 012737 030262 001236 MOV #EE2,@#TMP2
6044 030250 012700 030742 MOV #EEP1,R0 ;SET ACO OPERAND
6045 030254 172410 LDD (R0),ACO
6046 030256 012700 030742 MOV #EEP1,R0 ;FSPC
6047 030262 173010 EE2: SUBD (R0),ACO ;TEST INSTRUCTION
6048 030264 170205 STFPS R5 ;GET FPS
6049 030266 012700 030720 MOV #EEDATO,R0 ;GET THE RESULT
6050 030272 174010 STD ACO,(R0)
6051 030274 012701 030730 MOV #EEO,R1 ;IS IT CORRECT?
6052 030300 012702 000004 MOV #4,R2
6053 030304 022021 EE3: CMP (R0)+,(R1)+
6054 030306 001415 BEQ EE6
6055 030310 012700 030720 MOV #EEDATO,R0 ;DID A BAD
6056 030314 012701 030752 MOV #EEP2,R1 ;CONSTANT (NOT 57)
6057 030320 012702 000004 MOV #4,R2 ;GET GENERATED
6058 030324 022021 EE4: CMP (R0)+,(R1)+ ;FOR THE ALLIGNMENT
6059 030326 001402 BEQ EE5 ;FLOWS?
6060 030330 000137 030530 JMP @#EEER1 ;DATA ERROR.D
6061 030334 077205 EE5: SOB R2,EE4
6062 030336 000137 030566 JMP @#EEER2 ;BAD CONSTANT.D
6063 030342 077220 EE6: SOB R2,EE3
6064 030344 052704 000004 BIS #4,R4
6065 030350 020405 CMP R4,R5 ;FPS CORRECT?
6066 030352 001402 BEQ EE7
6067 030354 000137 030512 JMP @#EEERO ;BAD FPS
6068      ;USE NEGATIVE OPERANDS
6069 030360 EE7:
6070 030360 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
6071 030362 012704 003200 MOV #3200,R4 ;SET FIO, FIV, AND FD
6072 030366 170104 LDFPS R4
6073 030370 012737 030410 001236 MOV #EE8,@#TMP2
6074 030376 012700 030762 MOV #EEP3,R0 ;SET ACO OPERAND
6075 030402 172410 LDD (R0),ACO
6076 030404 012700 030762 MOV #EEP3,R0 ;FSPC
6077 030410 173010 EE8: SUBD (R0),ACO ;TEST INSTRUCTION
6078 030412 170205 STFPS R5 ;GET FPS
6079 030414 012700 030720 MOV #EEDATO,R0 ;GET THE RESULT
6080 030420 174010 STD ACO,(R0)
6081 030422 012701 030730 MOV #EEO,R1 ;IS IT CORRECT?
6082 030426 012702 000004 MOV #4,R2
6083 030432 022021 EE9: CMP (R0)+,(R1)+
6084 030434 001415 BEQ EE12
6085 030436 012700 030720 MOV #EEDATO,R0 ;DID A BAD
6086 030442 012701 030772 MOV #EEP4,R1 ;CONSTANT (NOT 57)
    
```

```

6087 030446 012702 000004
6088 030452 022021
6089 030454 001402
6090 030456 000137 030624
6091 030462 077205
6092 030464 000137 030662
6093 030470 077220
6094 030472 052704 000004
6095 030476 020405
6096 030500 001402
6097 030502 000137 030512
6098 030506 000137 031002
6099 030512 010437 001242
6100 030516 010537 001240
6101 030522 104002
6102 030524 000137 031002
6103 030530
6104 030530 012737 030742 001240
6105 030536 012737 030742 001242
6106 030544 012737 030720 001244
6107 030552 012737 030730 001246
6108 030560 104002
6109 030562 000137 031002
6110 030566
6111 030566 012737 030742 001240
6112 030574 012737 030742 001242
6113 030602 012737 030720 001244
6114 030610 012737 030730 001246
6115 030616 104002
6116 030620 000137 031002
6117 030624
6118 030624 012737 030762 001240
6119 030632 012737 030762 001242
6120 030640 012737 030720 001244
6121 030646 012737 030730 001246
6122 030654 104002
6123 030656 000137 031002
6124 030662
6125 030662 012737 030762 001240
6126 030670 012737 030762 001242
6127 030676 012737 030720 001244
6128 030704 012737 030730 001246
6129 030712 104002
6130 030714 000137 031002
6131 030720 000000
6132 030722 000000
6133 030724 000000
6134 030726 000000
6135 030730 000000
6136 030732 000000
6137 030734 000000
6138 030736 000000
6139 030740 000000
6140 030742 000200
6141 030744 000000
6142 030746 000000

EE10:  MOV #4,R2 ;GET GENERATED
      CMP (R0)+,(R1)+ ;FOR THE ALLIGNMENT
      BEQ EE11 ;FLOWS?
      JMP @#EEER3 ;DATA ERROR.D
EE11:  SOB R2,EE10
      JMP @#EEER4 ;BAD CONSTANT.D
EE12:  SOB R2,EE9
      BIS #4,R4
      CMP R4,R5 ;FPS CORRECT?
      BEQ EE13
      JMP @#EEERO ;BAD FPS.
EE13:  JMP @#EEDONE
EEERO: MOV R4,@#STMP4 ;BAD FPS
      MOV R5,@#STMP3
1$:    ERROR 2
      JMP @#EEDONE
EEER1: MOV #EEP1,@#STMP3
      MOV #EEP1,@#STMP4
      MOV #EEDATO,@#STMP5
      MCV #EEO,@#STMP6
1$:    ERROR 2
      JMP @#EEDONE
EEER2: MOV #EEP1,@#STMP3
      MOV #EEP1,@#STMP4
      MOV #EEDATO,@#STMP5
      MOV #EEO,@#STMP6
1$:    ERROR 2
      JMP @#EEDONE
EEER3: MOV #EEP3,@#STMP3
      MOV #EEP3,@#STMP4
      MOV #EEDATO,@#STMP5
      MOV #EEO,@#STMP6
1$:    ERROR 2
      JMP @#EEDONE
EEER4: MOV #EEP3,@#STMP3
      MOV #EEP3,@#STMP4
      MOV #EEDATO,@#STMP5
      MCV #EEO,@#STMP6
1$:    ERROR 2
      JMP @#EEDONE
EEDATO: 0
      0
      0
      0
EEO:    0
      0
00000  0
      0
EEP1:   200
      0
      0
    
```

6143 030750 000000  
 6144 030752 000400  
 6145 030754 000000  
 6146 030756 000000  
 6147 030760 000000  
 6148 030762 100200  
 6149 030764 000000  
 6150 030766 000000  
 6151 030770 000000  
 6152 030772 100400  
 6153 030774 000000  
 6154 030776 000000  
 6155 031000 000000  
 6156 031002 000000  
 6157 031002 104413  
 6158  
 6159  
 6160  
 6161  
 6162  
 6163  
 6164  
 6165  
 6166  
 6167  
 6168  
 6169  
 6170  
 6171  
 6172 031004 000004  
 6173  
 6174 031006  
 6175 031006 104414  
 6176 031010 012704 003200  
 6177 031014 170104  
 6178 031016 012737 031036 001236  
 6179 031024 012700 031330  
 6180 031030 172410  
 6181 031032 012700 031340  
 6182 031036 172010  
 6183 031040 170205  
 6184 031042 012700 031300  
 6185 031046 174010  
 6186 031050 012701 031350  
 6187 031054 012702 000004  
 6188 031060 022021  
 6189 031062 001401  
 6190 031064 000466  
 6191 031066 077204  
 6192 031070 020405  
 6193 031072 001401  
 6194 031074 000435  
 6195  
 6196  
 6197 031076  
 6198 031076 104414

```

EEP2: 0
      400
      0
      0
      0
EEP3: 100200
      0
      0
      0
EEP4: 100400
      0
      0
      0
EEDONE:
RSETUP
;GO INITIALIZE THE FPS AND STACK; AND
;SEE IF THE USER HAS EXPRESSED
;THE DESIRE TO CHANGE THE SOFTWARE
;VIRTUAL CONSOLE SWITCH REGISTER (HAS
;THE USER TYPED CONTROL G?).
:*****
:*TEST 36      NORMALIZE ALGORITHM TEST
:*
:* THIS IS A TEST OF THE NORMALIZE
:* FLOW ALGORITHM. TWO PATTERNS ARE USED,
:* FIRST THE MINIMUM SITUATION REQUIRING ONE
:* LEFT SHIFT AND THEN THE MAXIMUM SITUATION
:* REQUIRING 56 SHIFTS.
:*
:*****
TST36: SCOPE
;USE DATA PATTERNS THAT REQUIRE ONLY ONE LEFT SHIFT TO NORMALIZE
FF1:
      LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
      MOV #3200,R4 ;SET FIO, FIV, AND FD
      LD FPS R4
      MOV #FF2,@#STMP2
      MOV #FFP2,P0 ;SET ACO OPERAND
      LDD (R0),ACO
      MOV #FFP3,R0 ;FSPC
      ADDD (R0),ACO ;TEST INSTRUCTION
      STFPS R5 ;GET FPS
      MOV #FFDAT0,R0 ;GET THE RESULT
      STD ACO,(R0)
      MOV #FFP4,R1 ;IS IT CORRECT
      MOV #4,R2
      FF3: CMP (R0)+,(R1)+
           BEQ FF4
           BR FFER2 ;BAD DATA
      FF4: SOB R2,FF3
           CMP R4,R5 ;FPS CORRECT?
           BEQ FF5
           BR FFERO ;BAD FPS
;USE DATA PATTERNS WHICH REQUIRE 56 LEFT SHIFTS TO NORMALIZE
;THE RESULT
FF5:
      LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
    
```

6199	031100	012704	003200		MOV	#3200,R4		;SET FIU, FIV, AND FD
6200	031104	170104			LDFPS	R4		
6201	031106	012737	031126	001236	MOV	#FF6,@\$TMP2		
6202	031114	012700	031310		MOV	#FFP0,R0		;SET ACO OPERAND
6203	031120	172410			LDD	(R0),ACO		
6204	031122	012700	031320		MOV	#FFP1,R0		;FSRC
6205	031126	172010			ADDD	(R0),ACO		;TEST INSTRUCTION
6206	031130	170205			STFPS	R5		;GET FPS
6207	031132	012700	031300		MOV	#FFDATO,R0		;GET THE RESULT
6208	031136	174010			STD	ACC,(R0)		
6209	031140	012701	031350		MOV	#FFP4,R1		;IS IT CORRECT
6210	031144	012702	000004		MOV	#4,R2		
6211	031150	022021			FF7:	CMP	(R0)+,(R1)+	
6212	031152	001401			BEQ	FF10		
6213	031154	000413			BR	FFER1		;BATA
6214	031156	077204			FF10:	SOB	R2,FF7	
6215	031160	020405			CMP	R4,R5		;FPS CORRECT?
6216	031162	001401			BEQ	FF11		
6217	031164	000401			BR	FFER0		;BAD FPS
6218	031166	000474			FF11:	BR	FFDONE	
6219								
6220	031170	010537	001240		FFER0:	MOV	R5,@\$TMP3	
6221	031174	010437	001242			MOV	R4,@\$TMP4	
6222	031200	104002			1\$:	ERROR	2	
6223	031202	000466				BR	FFDONE	
6224								
6225	031204				FFER1:			
6226	031204	012737	031320	001240		MOV	#FFP1,@\$TMP3	
6227	031212	012737	031310	001242		MOV	#FFP0,@\$TMP4	
6228	031220	012737	031300	001244		MOV	#FFDATO,@\$TMP5	
6229	031226	012737	031350	001246		MOV	#FFP4,@\$TMP6	
6230	031234	104001			1\$:	ERROR	1	
6231	031236	000137	031360			JMP	@\$FFDONE	
6232								
6233	031242				FFER2:			
6234	031242	012737	031340	001240		MOV	#FFP3,@\$TMP3	
6235	031250	012737	031330	001242		MOV	#FFP2,@\$TMP4	
6236	031256	012737	031300	001244		MOV	#FFDATO,@\$TMP5	
6237	031264	012737	031350	001246		MOV	#FFP4,@\$TMP6	
6238	031272	104001			1\$:	ERROR	1	
6239	031274	000137	031360			JMP	@\$FFDONE	
6240								
6241								
6242	031300	000000			FFDATO:	0		
6243	031302	000000				0		
6244	031304	000000				0		
6245	031306	000000				0		
6246								
6247	031310	016000			FFP0:	16000		
6248	031312	000000				0		
6249	031314	000000				0		
6250	031316	000001				1		
6251	031320	116000			FFP1:	116000		
6252	031322	000000				0		
6253	031324	000000				0		
6254	031326	000000				0		

6255 031330 000500  
 6256 031332 000000  
 6257 031334 000000  
 6258 031336 000000  
 6259 031340 100400  
 6260 031342 000000  
 6261 031344 000000  
 6262 031346 000000  
 6263 031350 000200  
 6264 031352 000000  
 6265 031354 000000  
 6266 031356 000000  
 6267  
 6268 031360  
 6269  
 6270  
 6271 031360  
 6272  
 6273  
 6274  
 6275  
 6276  
 6277  
 6278  
 6279  
 6280  
 6281  
 6282  
 6283  
 6284  
 6285  
 6286  
 6287  
 6288  
 6289  
 6290  
 6291  
 6292  
 6293 031360 000004  
 6294  
 6295  
 6296  
 6297 031362  
 6298 031362 104414  
 6299 031364 012704 003200  
 6300 031370 170104  
 6301 031372 012737 031412 001236  
 6302 031400 012700 033142  
 6303 031404 172410  
 6304 031406 012700 033152  
 6305 031412 172010  
 6306 031414 170205  
 6307 031416 012700 033132  
 6308 031422 174010  
 6309 031424 012701 033162  
 6310 031430 012702 000004

FFP2: 500  
 0  
 0  
 0  
 FFP3: 100400  
 0  
 0  
 0  
 FFP4: 200  
 0  
 0  
 0

;FFP4-FFP0+FFP1  
 ; -FFP3+FFP4

FFDONE:  
 TST37:

\*\*\*\*\*  
 :\*\*\*\*\*  
 :FLOATING POINT SECOND PART  
 :\*\*\*\*\*  
 :\*\*\*\*\*

\*\*\*\*\*  
 :\*TEST 37 ROUND\TRUNK TEST  
 :\*  
 :\* THIS IS A TEST OF THE ROUND\TRUNK  
 :\* FLOWS. IN PARTICULAR TWO THINGS ARE TESTED:  
 :\* FIRST A CONDITION IN WHICH ROUNDING  
 :\* RESULTS IN THE NEED FOR RENORMALIZATION, AND  
 :\* SECOND THE PSW CONDITION CODES N AND  
 :\* Z BIT COMBINATIONS  
 :\*  
 :\*\*\*\*\*

TST37: SCOPE  
 ;ROUND AND NORMALIZE TEST

HH1: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.  
 MOV #3200,R4 ;SET FIU, FIV, AND FD  
 LDFPS R4  
 MOV #HH2,@\$TMP2  
 LDD (R0),ACO ;SET ACO OPERAND  
 MOV #HHP0,R0  
 MOV #HHP1,R0 ;FSPC  
 ADDD (R0),ACO ;TEST INSTRUCTION  
 STFPS R5 ;GET FPS  
 MOV #HHDAT0,R0 ;GET THE RESULT  
 SID ACO,(R0)  
 MOV #HHP2,R1 ;IS IT CORRECT  
 MOV #4,R2

```

6311 031434 022021          HH3:  CMP      (R0)+,(R1)+
6312 031436 001415          BEQ      HH6
6313 031440 012700 033132    MOV      #HHDAT0,R0          ;DID FLOW GO
6314 031444 012701 033172    MOV      #HHP3,R1          ;FROM STATE 663
6315 031450 012702 000004    MOV      #4,R2             ;TO 313 INSTEAD
6316 031454 022021          HH4:  CMP      (R0)+,(R1)+          ;OF TO 353
6317 031456 001402          BEQ      HH5
6318 031460 000137 032152    JMP      @#HHER0
6319 031464 077205          HH5:  SOB      R2,HH4
6320 031466 000137 032220    JMP      @#HHER1
6321 031472 077220          HH6:  SOB      R2,HH3
6322 031474 020405          CMP      R4,R5             ;FPS CORRECT?
6323 031476 001402          BEQ      HH7
6324 031500 000137 032266    JMP      @#HHER00
6325
6326          ;THIS IS A TEST OF THE ABILITY
6327          ;OF NORMALIZE TO PRODUCE A ZERO EXP. AND
6328          ;OF THE R\T ALGORITHM TO PROPERLY SET THE FPS
6329
6330 031504          HH7:
6331 031504 104414          LPERR
6332 031506 012704 043200    MOV      #043200,R4        ;SET UP THE LOOP ON ERROR ADDRESS.
6333          ;SET FIV,FIV,AND FD
6334          ;FID
6334 031512 170104          LDFPS   R4
6335 031514 012737 031542 001236    MOV      #HH8,@#STMP2      ;IN CASE UNDERFLOW
6336 031522 012737 033062 000244    MOV      #HHTRAP,@#FPVECT  ;TRAP OCCURS
6337 031530 012700 033212          MOV      #HHP5,R0          ;SET ACO OPERAND
6338 031534 172410          LDD     (R0),ACO
6339 031536 012700 033222          MOV      #HHP6,R0          ;FSPC
6340 031542 172010          HH8:  ADDD    (R0),ACO        ;TEST INSTRUCTION
6341 031544 170205          STFPS  R5                 ;GET FPS
6342 031546 012700 033132          MOV      #HHDAT0,R0        ;GET THE RESULT
6343 031552 174010          STD     ACO,(R0)
6344 031554 012701 033202          MOV      #HHP4,R1          ;IS IT CORRECT
6345 031560 012702 000004          MOV      #4,R2
6346 031564 022021          HH9:  CMP      (R0)+,(R1)+
6347 031566 001402          BEQ      HH10
6348 031570 000137 032334          JMP      @#HHER2
6349 031574 077205          HH10: SOB      R2,HH9
6350 031576 052704 100004          BIS     #100004,R4         ;FPS CORRECT?
6351 031602 020405          CMP      R4,R5
6352 031604 001402          BEQ      HH11
6353 031606 000137 032402          JMP      @#HHER3
6354          ;THIS IS A TEST OF THE R\T ALGORITHM'S
6355          ;ABILITY TO SET BOTH N AND Z ON A - 0 RESULT.
6356 031612          HH11:
6357 031612 104414          LPERR
6358          ;SET UP THE LOOP ON ERROR ADDRESS.
6359 031614 012704 043200          MOV      #043200,R4        ;SET FIV, FIV, AND FD
6360 031620 170104          LDFPS   R4
6361 031622 012737 031642 001236    MOV      #HH12,@#STMP2
6362 031630 012700 033242          MOV      #HHP8,R0          ;SET ACO OPERAND
6363 031634 172410          LDD     (R0),ACO
6364 031636 012700 033252          MOV      #HHP9,R0          ;FSPC
6365 031642 172010          HH12: ADDD    (R0),ACO        ;TEST INSTRUCTION
6366 031644 170205          STFPS  R5                 ;GET FPS
    
```

```

6367 031646 012700 033132      MOV      #HHDATO,R0      ;GET THE RESULT
6368 031652 174010      STD      ACO,(R0)
6369 031654 012701 033232      MOV      #HHP7,R1      ;IS IT CORRECT
6370 031660 012702 000004      MOV      #4,R2
6371 031664 022021      HH13:   CMP      (R0)+,(R1)+
6372 031666 001415      BEQ      HH16
6373 031670 012700 033132      MOV      #HHDATO,R0
6374 031674 012701 033202      MOV      #HHP4,R1
6375 031700 012702 000004      MOV      #4,R2
6376 031704 022021      HH14:   CMP      (R0)+,(R1)+
6377 031706 001402      BEQ      HH15
6378 031710 000137 032450      JMP      @#HHER4
6379 031714 077205      HH15:   SOB      R2,HH14
6380 031716 000137 032516      JMP      @#HHER5
6381 031722 077220      HH16:   SOB      R2,HH13
6382 031724 052704 100014      BIS      #100014,R4      ;FPS CORRECT?
6383 031730 020405      CMP      R4,R5
6384 031732 001402      BEQ      HH17
6385 031734 000137 032564      JMP      @#HHER6
6386      ;TEST THAT CC ARE CLEARED BY R\T
6387 031740      HH17:
6388 031740 104414      LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.
6389 031742 012704 000200      MOV      #00200,R4      ;SET FIV, FIV, AND FD
6390 031746 170104      LDFPS     R4
6391 031750 012737 031776 001236      MOV      #HH18,@#$TMP2
6392 031756 012737 062534 000244      MOV      #FPSPUR,@#FPVECT
6393 031764 012700 033242      MOV      #HHP8,R0      ;SET ACO OPERAND
6394 031770 172410      LDD      (R0),ACO
6395 031772 012700 033242      MOV      #HHP8,R0      ;FSPC
6396 031776 172010      HH18:   ADDD     (R0),ACO      ;TEST INSTRUCTION
6397 032000 170205      STFPS     R5      ;GET FPS
6398 032002 012700 033132      MOV      #HHDATO,R0      ;GET THE RESULT
6399 032006 174010      STD      ACO,(R0)
6400 032010 012701 033262      MOV      #HHP10,R1      ;IS IT CORRECT
6401 032014 012702 000004      MOV      #4,R2
6402 032020 022021      HH19:   CMP      (R0)+,(R1)+
6403 032022 001402      BEQ      HH20
6404 032024 000137 032632      JMP      @#HHER7
6405 032030 077205      HH20:   SOB      R2,HH19
6406 032032 052704 000000      BIS      #00000,R4      ;FPS CORRECT?
6407 032036 020405      CMP      R4,R5
6408 032040 001402      BEQ      HH21
6409 032042 000137 032700      JMP      @#HHER8
6410      ;TEST THAT N IS SET BY R\T
6411 032046      HH21:
6412 032046 104414      LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.
6413 032050 012704 003200      MOV      #3200,R4      ;SET FIV, FIV, AND FD
6414 032054 170104      LDFPS     R4
6415 032056 012737 032076 001236      MOV      #HH22,@#$TMP2
6416 032064 012700 033212      MOV      #HHP5,R0      ;SET ACO OPERAND
6417 032070 172410      LDD      (R0),ACO
6418 032072 012700 033212      MOV      #HHP5,R0      ;FSPC
6419 032076 172010      HH22:   ADDD     (R0),ACO      ;TEST INSTRUCTION
6420 032100 170205      STFPS     R5      ;GET FPS
6421 032102 012700 033132      MOV      #HHDATO,R0      ;GET THE RESULT
6422 032106 174010      STD      ACO,(R0)
    
```



```

6423 032110 012701 033272          MOV      #HHP11,R1          ;IS IT CORRECT
6424 032114 012702 000004          MOV      #4,R2
6425 032120 022021          HH23:   CMP      (R0)+,(R1)+
6426 032122 001402          BEQ      HH24
6427 032124 000137 032746          JMP      @HHER9
6428 032130 077205          HH24:   SOB      R2,HH23
6429 032132 052704 000010          BIS      #10,R4
6430 032136 020405          CMP      R4,R5          ;FPS CORRECT?
6431 032140 001402          BEQ      HH25
6432 032142 000137 033014          JMP      @HHER10
6433 032146 000137 033302          HH25:   JMP      @HHDONE
6434 032152
6435 032152 010537 001252          HHER0:  MOV      R5,@$STMP10
6436 032156 010437 001254          MOV      R4,@$STMP11
6437 032162 012737 033152 001240          MOV      #HHP1,@$STMP3
6438 032170 012737 033142 001242          MOV      #HHP0,@$STMP4
6439 032176 012737 033132 001244          MOV      #HHDATO,@$STMP5
6440 032204 012737 033162 001246          MOV      #HHP2,@$STMP6
6441 032212 104002          1$:     ERROR  2
6442 032214 000137 033302          JMP      @HHDONE
6443 032220
6444 032220 010537 001252          HHER1:  MOV      R5,@$STMP10
6445 032224 010437 001254          MOV      R4,@$STMP11
6446 032230 012737 033152 001240          MOV      #HHP1,@$STMP3
6447 032236 012737 033142 001242          MOV      #HHP0,@$STMP4
6448 032244 012737 033132 001244          MOV      #HHDATO,@$STMP5
6449 032252 012737 033162 001246          MOV      #HHP2,@$STMP6
6450 032260 104002          1$:     ERROR  2
6451 032262 000137 033302          JMP      @HHDONE
6452 032266
6453 032266 010537 001252          HHER00: MOV      R5,@$STMP10
6454 032272 010437 001254          MOV      R4,@$STMP11
6455 032276 012737 033152 001240          MOV      #HHP1,@$STMP3
6456 032304 012737 033142 001242          MOV      #HHP0,@$STMP4
6457 032312 012737 033132 001244          MOV      #HHDATO,@$STMP5
6458 032320 012737 033162 001246          MOV      #HHP2,@$STMP6
6459 032326 104002          1$:     ERROR  2
6460 032330 000137 033302          JMP      @HHDONE
6461 032334
6462 032334 010537 001252          HHER2:  MOV      R5,@$STMP10
6463 032340 010437 001254          MOV      R4,@$STMP11
6464 032344 012737 033222 001240          MOV      #HHP6,@$STMP3
6465 032352 012737 033212 001242          MOV      #HHP5,@$STMP4
6466 032360 012737 033132 001244          MOV      #HHDATO,@$STMP5
6467 032366 012737 033202 001246          MOV      #HHP4,@$STMP6
6468 032374 104002          1$:     ERROR  2
6469 032376 000137 033302          JMP      @HHDONE
6470 032402
6471 032402 010537 001252          HHER3:  MOV      R5,@$STMP10
6472 032406 010437 001254          MOV      R4,@$STMP11
6473 032412 012737 033222 001240          MOV      #HHP6,@$STMP3
6474 032420 012737 033212 001242          MOV      #HHP5,@$STMP4
6475 032426 012737 033132 001244          MOV      #HHDATO,@$STMP5
6476 032434 012737 033202 001246          MOV      #HHP4,@$STMP6
6477 032442 104002          1$:     ERROR  2
6478 032444 000137 033302          JMP      @HHDONE
    
```

6479	032450				HHER4:	
6480	032450	010537	001252		MOV	R5,@#STMP10
6481	032454	010437	001254		MOV	R4,@#STMP11
6482	032460	012737	033252	001240	MOV	#HHP9,@#STMP3
6483	032466	012737	033242	001242	MOV	#HHP8,@#STMP4
6484	032474	012737	033132	001244	MOV	#HHDATO,@#STMP5
6485	032502	012737	033232	001246	MOV	#HHP7,@#STMP6
6486	032510	104002			1\$:	ERROR
6487	032512	000137	033302		JMP	@#HHDONE
6488	032516				HHER5:	
6489	032516	010537	001252		MOV	R5,@#STMP10
6490	032522	010437	001254		MOV	R4,@#STMP11
6491	032526	012737	033252	001240	MOV	#HHP9,@#STMP3
6492	032534	012737	033242	001242	MOV	#HHP8,@#STMP4
6493	032542	012737	033132	001244	MOV	#HHDATO,@#STMP5
6494	032550	012737	033232	001246	MOV	#HHP7,@#STMP6
6495	032556	104002			1\$:	ERROR
6496	032560	000137	033302		JMP	@#HHDONE
6497	032564				HHER6:	
6498	032564	010537	001252		MOV	R5,@#STMP10
6499	032570	010437	001254		MOV	R4,@#STMP11
6500	032574	012737	033252	001240	MOV	#HHP9,@#STMP3
6501	032602	012737	033242	001242	MOV	#HHP8,@#STMP4
6502	032610	012737	033132	001244	MOV	#HHDATO,@#STMP5
6503	032616	012737	033232	001246	MOV	#HHP7,@#STMP6
6504	032624	104002			1\$:	ERROR
6505	032626	000137	033302		JMP	@#HHDONE
6506	032632				HHER7:	
6507	032632	010537	001252		MOV	R5,@#STMP10
6508	032636	010437	001254		MOV	R4,@#STMP11
6509	032642	012737	033242	001240	MOV	#HHP8,@#STMP3
6510	032650	012737	033242	001242	MOV	#HHP8,@#STMP4
6511	032656	012737	033132	001244	MOV	#HHDATO,@#STMP5
6512	032664	012737	033262	001246	MOV	#HHP10,@#STMP6
6513	032672	104002			1\$:	ERROR
6514	032674	000137	033302		JMP	@#HHDONE
6515	032700				HHER8:	
6516	032700	010537	001252		MOV	R5,@#STMP10
6517	032704	010437	001254		MOV	R4,@#STMP11
6518	032710	012737	033242	001240	MOV	#HHP8,@#STMP3
6519	032716	012737	033242	001242	MOV	#HHP8,@#STMP4
6520	032724	012737	033132	001244	MOV	#HHDATO,@#STMP5
6521	032732	012737	033262	001246	MOV	#HHP10,@#STMP6
6522	032740	104002			1\$:	ERROR
6523	032742	000137	033302		JMP	@#HHDONE
6524	032746				HHER9:	
6525	032746	010537	001252		MOV	R5,@#STMP10
6526	032752	010437	001254		MOV	R4,@#STMP11
6527	032756	012737	033212	001240	MOV	#HHP5,@#STMP3
6528	032764	012737	033212	001242	MOV	#HHP5,@#STMP4
6529	032772	012737	033132	001244	MOV	#HHDATO,@#STMP5
6530	033000	012737	033272	001246	MOV	#HHP11,@#STMP6
6531	033006	104002			1\$:	ERROR
6532	033010	000137	033302		JMP	@#HHDONE
6533	033014				HHER10:	
6534	033014	010537	001252		MOV	R5,@#STMP10

6535	033020	010437	001254		MOV	R4,@#STMP11	
6536	033024	012737	033212	001240	MOV	#HHP5,@#STMP3	
6537	033032	012737	033212	001242	MOV	#HHP5,@#STMP4	
6538	033040	012737	033132	001244	MOV	#HHDATO,@#STMP5	
6539	033046	012737	033272	001246	MOV	#HHP11,@#STMP6	
6540	033054	104002			1\$:	ERROR	2
6541	033056	000137	033302		JMP	@#HHDONE	
6542	033062	013703	001236		HHTRAP:	MOV	@#STMP2,R3 ;WAS THE TRAP TO 244
6543	033066	062703	000002			ADD	#2,R3 ;ON THE INSTRUCTION
6544	033072	020316				CMP	R3,(SP) ;BEING TESTED?
6545	033074	001402				BEQ	1\$
6546	033076	000137	062534		JMP	@#FPSPUR	
6547	033102	011637	001236		1\$:	MOV	(SP),@#STMP2 ;FAILURE OF FPS INTERRUPT
6548							;DISABLE BIT (FID=1)
6549	033106	022626				CMP	(SP)+,(SP)+ ;TO INHIBIT TRAP.
6550	033110	170201				STFPS	R1
6551	033112	010137	001240		MOV	R1,@#STMP3	
6552	033116	170301			STST	R1	
6553	033120	010137	001242		MOV	R1,@#STMP4	
6554	033124	104002			2\$:	ERROR	2
6555	033126	000137	033302		JMP	@#HHDONE	
6556	033132	000000			HHDATO:	0	
6557	033134	000000				0	
6558	033136	000000				0	
6559	033140	000000				0	
6560	033142	000452			HHP0:	452	
6561	033144	125252				125252	
6562	033146	125252				125252	
6563	033150	125253				125253	
6564	033152	000252			HHP1:	252	
6565	033154	125252				125252	
6566	033156	125252				125252	
6567	033160	125252				125252	
6568	033162	000600			HHP2:	600	
6569	033164	000000				0	;HHP0 + HHP1 WITH
6570	033166	000000				0	;PROPER NORMALIZATION
6571	033170	000000				0	
6572	033172	000400			HHP3:	400	
6573	033174	000000				0	;HHP0 + HHP1 WITH
6574	033176	000000				0	;BAD NORMALIZATION
6575	033200	000000				0	
6576	033202	000000			HHP4:	0	
6577	033204	000000				0	
6578	033206	000000				0	
6579	033210	000000				0	
6580	033212	100200			HHP5:	100200	
6581	033214	000000				0	
6582	033216	000000				0	
6583	033220	000000				0	
6584	033222	000300			HHP6:	300	
6585	033224	000000				0	
6586	033226	000000				0	
6587	033230	000000				0	
6588	033232	100000			HHP7:	100000	
6589	033234	000000				0	;HHP7 = HHP8 + HHP9
6590	033236	000000				0	; - HHP5 + HHP6

```

6591 033240 000000
6592 033242 000200
6593 033244 000000
6594 033246 000000
6595 033250 000000
6596 033252 100300
6597 033254 000000
6598 033256 000000
6599 033260 000000
6600 033262 000400
6601 033264 000000
6602 033266 000000
6603 033270 000000
6604 033272 100400
6605 033274 000000
6606 033276 000000
6607 033300 000000
6608 033302
6609 033302 104413
6610
6611
6612
6613
6614
6615
6616
6617
6618
6619
6620
6621
6622
6623
6624
6625
6626
6627
6628
6629
6630 033304 000004
6631
6632
6633 033306
6634 033306 104414
6635 033310 012704 000200
6636 033314 170104
6637 033316 012737 033344 001236
6638 033324 012737 034376 000244
6639 033332 012700 036150
6640 033336 172410
6641 033340 012700 036150
6642 033344 172010
6643 033346 170205
6644 033350 012700 036100
6645 033354 174010
6646 033356 012701 036160

HHP8: 0
        200
        0
        0
HHP9: 100300
        0
        0
HHP10: 400
        0
        0
        0
HHP11: 100400
        0
        0
        0
HHDONE:
        RSETUP

;GO INITIALIZE THE FPS AND STACK; AND
;SEE IF THE USER HAS EXPRESSED
;THE DESIRE TO CHANGE THE SOFTWARE
;VIRTUAL CONSOLE SWITCH REGISTER (HAS
;THE USER TYPED CONTROL G?).

;*****
;*TEST 40 OVER\UNDER TEST
;*
;*THIS IS A PARTIAL TEST OF THE OVER\UNDER
;*FLOWS. ONE OVERFLOW AND TWO UNDERFLOW
;*CONDITIONS ARE CHECKED. THE REMAINING
;*UNDERFLOW COND. AND THE REMAINING OVERFLOW
;*COND. WILL BE CHECKED LATER USING THE
;*XXX INSTRUCTION. HERE EACH CONDITION TESTED
;*IS CHECKED BOTH WITH TRAPS ENABLED
;*(FIU=1 OR FIV=1) AND ALSO WITH TRAPS
;*DISABLED (FIU=0 OR FIV=0).
;*
;*****
TST40: SCOPE

;TEST OVERFLOW CONDITION WITH TRAP DISABLER FIV=0
GG1:
        LPERR
        MOV #200,R4
        LDFPS R4
        MOV #GG2,@#STMP2
        MOV #GGERO,@#FPVECT
        MOV #GGP5,R0
        LDD (R0),ACO
        MOV #GGP5,R0
        ADDD (R0),ACO
        STFPS R5
        MOV #GGDATO,R0
        STD ACO,(R0)
        MOV #GGP6,R1

;SET UP THE LOOP ON ERROR ADDRESS.
;CLEAR FIU, FIV, AND SET FD
;SET ACO OPERAND
;FSRC
;TEST INSTRUCTION
;GET FPS
;GET THE RESULT
;IS IT CORRECT
    
```

```

6647 033362 012702 000004
6648 033366 022021
6649 033370 001402
6650 033372 000137 034474
6651 033376 077205
6652 033400 052704 000006
6653 033404 020405
6654 033406 001402
6655 033410 000137 034542
6656
6657
6658 033414
6659 033414 104414
6660 033416 012704 001200
6661 033422 170104
6662 033424 012737 033452 001236
6663 033432 012737 033470 000244
6664 033440 012700 036150
6665 033444 172410
6666 033446 012700 036150
6667 033452 172010
6668 033454 170000
6669 033456 012700 036100
6670 033462 174010
6671 033464 000137 034610
6672 033470 013703 001236
6673 033474 062703 000002
6674 033500 020316
6675 033502 001402
6676 033504 000137 062534
6677 033510 011637 001236
6678 033514 022626
6679 033516 170205
6680 033520 012700 036100
6681 033524 174010
6682 033526 012701 036160
6683 033532 012702 000004
6684 033536 022021
6685 033540 001402
6686 033542 000137 034656
6687 033546 077205
6688 033550 052704 100006
6689 033554 020405
6690 033556 001402
6691 033560 000137 034726
6692 033564 012704 000010
6693
6694 033570 170305
6695 033572 020405
6696 033574 001402
6697 033576 000137 034660
6698
6699
6700 033602
6701 033602 104414
6702 033604 012704 000200

GG3: MOV #4,R2
      CMP (R0)+,(R1)+
      BEQ GG4
      JMP @GGGER1
GG4: SOB R2,GG3
      BIS #6,R4 ;FPS CORRECT?
      CMP R4,R5
      BEQ GG5
      JMP @GGGER2
      ;TEST OVERFLOW WITH TRAPS ENABLED
      ;FIV = 1
GG5: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
      MOV #1200,R4 ;CLEAR FIU, SET FIV, AND FD
      LDFPS R4
      MOV #GG6,@$TMP2
      MOV #GG7,@$FPVECT
      MOV #GGP5,R0 ;SET ACO OPERAND
      LDD (R0),ACO
      MOV #GGP5,R0 ;FSPC
GG6: ADDD (R0),ACO ;TEST INSTRUCTION
      CFCC ;NO OVERFLOW TRAP OCCURED
      MOV #GGDAT0,R0
      STD ACO,(R0)
      JMP @GGGER3
GG7: MOV @$TMP2,R3
      ADD #2,R3
      CMP R3,(SP) ;CHECK STACK DATA
      BEQ 1$
      JMP @$FPSPUR
1$: MOV (SP),@$TMP2
      CMP (SP)+,(SP)+
      STFPS R5
      MOV #GGDAT0,R0 ;GET THE RESULT
      STD ACO,(R0)
      MOV #GGP6,P1 ;IS IT CORRECT
GG8: MOV #4,R2
      CMP (R0)+,(R1)+
      BEQ GG9
      JMP @GGGER4
GG9: SOB R2,GG8
      BIS #100006,R4 ;EXACT ZERO RESULTED IF OVERFLOW
      CMP R4,R5 ;FPS CORRECT?, CHECK FER, FZ, FV
      BEQ 1$
      JMP @GGGER6
1$: MOV #10,R4
      ;CHECK FEC
      STST R5
      CMP R4,R5
      BEQ GG10
      JMP @GGGER5
GG10: ;CHECK UNDER FLOW CONDITION WITH
      ;TRAPS DISABLED (FIU = 0)
      LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
      MOV #0200,R4 ;SET FIU, FIV, AND FD
    
```

6703	033610	170104			LDFPS	R4		
6704	033612	012737	033640	001236	MOV	#GG11,@#STMP2		
6705	033620	012737	034774	000244	MOV	#GGER7,@#FPVECT		
6706	033626	012700	036120		MOV	#GGP2,R0	:SET ACO OPERAND	
6707	033632	172410			LDD	(R0),ACO	:FSRC	
6708	033634	012700	036130		MOV	#GGP3,R0		
6709	033640	172010			GG11: ADDD	(R0),ACO	:TEST INSTRUCTION	
6710	033642	170205			STFPS	R5	:GET FPS	
6711	033644	012700	036100		MOV	#GGDAT0,R0	:GET THE RESULT	
6712	033650	174010			STD	ACO,(R0)		
6713	033652	012701	036160		MOV	#GGP6,R1	:IS IT CORRECT	
6714	033656	012702	000004		MOV	#4,R2		
6715	033662	022021			GG12: CMP	(R0)+,(R1)+		
6716	033664	001402			BEQ	GG13		
6717	033666	000137	035072		JMP	@#GGER8		
6718	033672	077205			GG13: SOB	R2,GG12		
6719	033674	052704	000004		BIS	#4,R4	:FPS CORRECT?	
6720	033700	020405			CMP	R4,R5		
6721	033702	001402			BEQ	GG14		
6722	033704	000137	035140		JMP	@#GGER9		
6723							:CHECK UNDERFLOW CONDITION WITH	
6724							:TRAP ENABLED (FIU = 1)	
6725	033710				GG14: LPERR		:SET UP THE LOOP ON ERROR ADDRESS.	
6726	033710	104414			MOV	#2200,R4	:SET FIU, FIV, AND FD	
6727	033712	012704	002200					
6728	033716	170104			LDFPS	R4		
6729	033720	012737	033746	001236	MOV	#GG15,@#STMP2		
6730	033726	012737	033764	000244	MOV	#GG16,@#FPVECT		
6731	033734	012700	036120		MOV	#GGP2,R0	:SET ACO OPERAND	
6732	033740	172410			LDD	(R0),ACO	:FSPC	
6733	033742	012700	036130		MOV	#GGP3,R0		
6734	033746	172010			GG15: ADDD	(R0),ACO	:TEST INSTRUCTION	
6735	033750	170000			CFCC			
6736	033752	012700	036100		MOV	#GGDAT0,R0		
6737	033756	174010			STD	ACO,(R0)		
6738	033760	000137	035206		JMP	@#GGER10		
6739	033764	013703	001236		GG16: MOV	@#STMP2,R3		
6740	033770	062703	000002		ADD	#2,R3		
6741	033774	021603			CMP	(SP),R3		
6742	033776	001402			BEQ	1\$		
6743	034000	000137	062534		JMP	@#FPSPUR		
6744	034004	011637	001236		1\$: MOV	(SP),@#STMP2		
6745	034010	022626			CMP	(SP)+,(SP)+		
6746	034012	170205			STFPS	R5	:GET FPS	
6747	034014	012700	036100		MOV	#GGDAT0,R0	:GET THE RFSULT	
6748	034020	174010			STD	ACO,(R0)		
6749	034022	012701	036170		MOV	#GGP7,R1	:IS IT CORRECT	
6750	034026	012702	000004		MOV	#4,R2		
6751	034032	022021			GG17: CMP	(R0)+,(R1)+		
6752	034034	001402			BEQ	GG18		
6753	034036	000137	035254		JMP	@#GGER11		
6754	034042	077205			GG18: SOB	R2,GG17		
6755	034044	052704	100000		BIS	#100000,R4		
6756	034050	020405			CMP	R4,R5	:FPS CORRECT?	
6757	034052	001402			BEQ	2\$		
6758	034054	000137	035322		JMP	@#GGER12		

```

6759 034060
6760 034060 012704 000012
6761
6762 034064 170305
6763 034066 020405
6764 034070 001402
6765 034072 000177 001272
6766
6767
6768 034076
6769 034076 104414
6770 034100 012704 000200
6771 034104 170104
6772 034106 012737 034134 001236
6773 034114 012737 035436 000244
6774 034122 012700 036120
6775 034126 172410
6776 034130 012700 036200
6777 034134 172010
6778 034136 170205
6779 034140 012700 036100
6780 034144 174010
6781 034146 012701 036160
6782 034152 012702 000004
6783 034156 022021
6784 034160 001402
6785 034162 000137 035534
6786 034166 077205
6787 034170 052704 000004
6788 034174 020405
6789 034176 001402
6790 034200 000137 035602
6791
6792
6793 034204
6794 034204 104414
6795 034206 012704 002200
6796 034212 170104
6797 034214 012737 034242 001236
6798 034222 012737 034260 000244
6799 034230 012700 036120
6800 034234 172410
6801 034236 012700 036200
6802 034242 172010
6803 034244 170000
6804 034246 012700 036100
6805 034252 174010
6806 034254 000137 035650
6807 034260 013700 001236
6808 034264 062700 000002
6809 034270 020016
6810 034272 001402
6811 034274 000137 062534
6812 034300 011637 001236
6813 034304 022626
6814 034306 170205

2$:
1$:
;CHECK FEC
    STST R5
    CMP R4,R5
    BEQ GG19
    JMP @GGER13
;CHECK UNDERFLOW CONDITION WITH TRAPS
;DISABLED (FIU = 0)
GG19:
    LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
    MOV #0200,R4 ;SET FIU, FIV, AND FD
    LDFPS R4
    MOV #GG20,@$TMP2
    MOV #GGER14,@$FPVECT
    MOV #GGP2,R0 ;SET ACO OPERAND
    LDD (R0),ACO
    MOV #GGP8,R0 ;FSRC
    ADDD (R0),ACO ;TEST INSTRUCTION
    STFPS R5 ;GET FPS
    MOV #GGDATO,R0 ;GET THE RESULT
    STD ACO,(R0)
    MOV #GGP6,R1 ;IS IT CORRECT
    MOV #4,R2
GG21:
    CMP (R0)+,(P1)+
    BEQ GG22
    JMP @GGER15
GG22:
    SOB R2,GG21
    BIS #4,R4 ;FPS CORRECT?
    CMP R4,R5
    BEQ GG23
    JMP @GGER16
;CHECK UNDERFLOW CONDITION WITH TRAP
;ENABLED (FIU = 1)
GG23:
    LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
    MOV #2200,R4 ;SET FIU, FIV, AND FD
    LDFPS R4
    MOV #GG24,@$TMP2
    MOV #GG25,@$FPVECT
    MOV #GGP2,R0 ;SET ACO OPERAND
    LDD (R0),ACO
    MOV #GGP8,R0 ;FSRC
    ADDD (R0),ACO ;TEST INSTRUCTION
    CFCC
    MOV #GGDATO,R0
    STD ACO,(R0)
    JMP @GGER17
GG24:
    MOV @$TMP2,R0
    ADD #2,R0
    CMP R0,(SP)
    BEQ 1$
    JMP @$FPSPUR
GG25:
    MOV (SP),@$TMP2
    CMP (SP)+,(SP)+
    STFPS R5 ;GET FPS
    1$:
    
```

6815	034310	012700	036100		MOV	#GGDATO,R0	;GET THE RESULT
6816	034314	174010			STD	ACO,(R0)	
6817	034316	012701	036210		MOV	#GGP9,R1	;IS IT CORRECT
6818	034322	012702	000004		MOV	#4,R2	
6819	034326	022021		GG26:	CMP	(R0)+,(R1)+	
6820	034330	001402			BEQ	GG27	
6821	034332	000137	035716		JMP	@#GGER18	
6822	034336	077205		GG27:	SOB	R2,GG26	
6823	034340	052704	100004		BIS	#100004,R4	
6824	034344	020405			CMP	R4,R5	;FPS CORRECT?
6825	034346	001402			BEQ	1\$	
6826	034350	000167	001456		JMP	GGER20	
6827	034354	012704	000012	1\$:	MOV	#12,R4	
6828						:CHECK FEC	
6829	034360	170305			STST	R5	
6830	034362	020405			CMP	R4,R5	
6831	034364	001402			BEQ	GG28	
6832	034366	000167	001372		JMP	GGER19	
6833							
6834	034372	000137	036220	GG28:	JMP	@#GGDONE	
6835							
6836	034376	013701	001236	GGER0:	MOV	@#STMP2,R1	
6837	034402	062701	000002		ADD	#2,R1	
6838	034406	020116			CMP	R1,(SP)	
6839	034410	001402			BEQ	10\$	
6840	034412	000137	062534	5\$:	JMP	@#FPSPUR	
6841	034416			10\$:			
6842	034416	170301			STST	R1	
6843	034420	020127	000010		CMP	R1,#10	
6844	034424	001372			BNE	5\$	
6845	034426	022626			CMP	(SP)+,(SP)+	
6846	034430	012700	036100		MOV	#GGDATO,R0	
6847	034434	174010			STD	ACO,(R0)	
6848	034436	012737	036150	001240	MOV	#GGP5,@#STMP3	
6849	034444	012737	036150	001242	MOV	#GGP5,@#STMP4	
6850	034452	012737	036100	001244	MOV	#GGDATO,@#STMP5	
6851	034460	012737	036160	001246	MOV	#GGP6,@#STMP6	
6852	034466	104001		1\$:	ERROR	1	
6853	034470	000137	036220		JMP	@#GGDONE	
6854							
6855	034474			GGER1:			
6856	034474	010537	001252		MOV	R5,@#STMP10	
6857	034500	010437	001254		MOV	R4,@#STMP11	
6858	034504	012737	036150	001240	MOV	#GGP5,@#STMP3	
6859	034512	012737	036150	001242	MOV	#GGP5,@#STMP4	
6860	034520	012737	036100	001244	MOV	#GGDATO,@#STMP5	
6861	034526	012737	036160	001246	MOV	#GGP6,@#STMP6	
6862	034534	104002		1\$:	ERROR	2	
6863	034536	000137	036220		JMP	@#GGDONE	
6864							
6865	034542			GGER2:			
6866	034542	010537	001252		MOV	R5,@#STMP10	
6867	034546	010437	001254		MOV	R4,@#STMP11	
6868	034552	012737	036150	001240	MOV	#GGP5,@#STMP3	
6869	034560	012737	036150	001242	MOV	#GGP5,@#STMP4	
6870	034566	012737	036100	001244	MOV	#GGDATO,@#STMP5	



6871	034574	012737	036160	001246		MOV	#GGP6,@#STMP6
6872	034602	104002			1\$:	ERROR	2
6873	034604	000137	036220			JMP	@#GGDONE
6874							
6875	034610				GGER3:		
6876	034610	010537	001252			MOV	R5,@#STMP10
6877	034614	010437	001254			MOV	R4,@#STMP11
6878	034620	012737	036150	001240		MOV	#GGP5,@#STMP3
6879	034626	012737	036150	001242		MOV	#GGP5,@#STMP4
6880	034634	012737	036100	001244		MOV	#GGDAT0,@#STMP5
6881	034642	012737	036160	001246		MOV	#GGP6,@#STMP6
6882	034650	104002			1\$:	ERROR	2
6883	034652	000137	036220			JMP	@#GGDONE
6884							
6885	034656	000706			GGER4:	BR	GGER1
6886							
6887	034660				GGER5:		
6888	034660	010537	001252			MOV	R5,@#STMP10
6889	034664	010437	001254			MOV	R4,@#STMP11
6890	034670	012737	036150	001240		MOV	#GGP5,@#STMP3
6891	034676	012737	036150	001242		MOV	#GGP5,@#STMP4
6892	034704	012737	036100	001244		MOV	#GGDAT0,@#STMP5
6893	034712	012737	036160	001246		MOV	#GGP6,@#STMP6
6894	034720	104002			1\$:	ERROR	2
6895	034722	000137	036220			JMP	@#GGDONE
6896							
6897	034726				GGER6:		
6898	034726	010537	001252			MOV	R5,@#STMP10
6899	034732	010437	001254			MOV	R4,@#STMP11
6900	034736	012737	036150	001240		MOV	#GGP5,@#STMP3
6901	034744	012737	036150	001242		MOV	#GGP5,@#STMP4
6902	034752	012737	036100	001244		MOV	#GGDAT0,@#STMP5
6903	034760	012737	036160	001246		MOV	#GGP6,@#STMP6
6904	034766	104002			1\$:	ERROR	2
6905	034770	000137	036220			JMP	@#GGDONE
6906							
6907	034774	013701	001236		GGER7:	MOV	@#STMP2,R1
6908	035000	062701	000002			ADD	#2,R1
6909	035004	020116				CMP	R1,(SP)
6910	035006	001402				BEQ	10\$
6911	035010	000137	062534		5\$:	JMP	@#FPSPUR
6912	035014				10\$:		
6913	035014	170301				STST	R1
6914	035016	020127	000012			CMP	R1,#12
6915	035022	001372				BNE	5\$
6916	035024	022626				CMP	(SP)+,(SP)+
6917	035026	012700	036100			MOV	#GGDAT0,R0
6918	035032	174010				STD	AC0,(R0)
6919	035034	012737	036130	001240		MOV	#GGP3,@#STMP3
6920	035042	012737	036120	001242		MOV	#GGP2,@#STMP4
6921	035050	012737	036100	001244		MOV	#GGDAT0,@#STMP5
6922	035056	012737	036160	001246		MOV	#GGP6,@#STMP6
6923	035064	104002			1\$:	ERROR	2
6924	035066	000137	036220			JMP	@#GGDONE
6925							
6926	035072				GGER8:		

6927	035072	010537	001252		MOV	R5,@#STMP10
6928	035076	010437	001254		MOV	R4,@#STMP11
6929	035102	012737	036130	001240	MOV	#GGP3,@#STMP3
6930	035110	012737	036120	001242	MOV	#GGP2,@#STMP4
6931	035116	012737	036100	001244	MOV	#GGDATO,@#STMP5
6932	035124	012737	036160	001246	MOV	#GGP6,@#STMP6
6933	035132	104002			1\$:	ERROR
6934	035134	000137	036220		JMP	@#GGDONE
6935						
6936	035140				GGER9:	
6937	035140	010537	001252		MOV	R5,@#STMP10
6938	035144	010437	001254		MOV	R4,@#STMP11
6939	035150	012737	036130	001240	MOV	#GGP3,@#STMP3
6940	035156	012737	036120	001242	MOV	#GGP2,@#STMP4
6941	035164	012737	036100	001244	MOV	#GGDATO,@#STMP5
6942	035172	012737	036160	001246	MOV	#GGP6,@#STMP6
6943	035200	104002			1\$:	ERROR
6944	035202	000137	036220		JMP	@#GGDONE
6945						
6946	035206				GGER10:	
6947	035206	010537	001252		MOV	R5,@#STMP10
6948	035212	010437	001254		MOV	R4,@#STMP11
6949	035216	012737	036130	001240	MOV	#GGP3,@#STMP3
6950	035224	012737	036120	001242	MOV	#GGP2,@#STMP4
6951	035232	012737	036100	001244	MOV	#GGDATO,@#STMP5
6952	035240	012737	036170	001246	MOV	#GGP7,@#STMP6
6953	035246	104002			1\$:	ERROR
6954	035250	000137	036220		JMP	@#GGDONE
6955						
6956	035254				GGER11:	
6957	035254	010537	001252		MOV	R5,@#STMP10
6958	035260	010437	001254		MOV	R4,@#STMP11
6959	035264	012737	036130	001240	MOV	#GGP3,@#STMP3
6960	035272	012737	036120	001242	MOV	#GGP2,@#STMP4
6961	035300	012737	036100	001244	MOV	#GGDATO,@#STMP5
6962	035306	012737	036170	001246	MOV	#GGP7,@#STMP6
6963	035314	104002			1\$:	ERRCR
6964	035316	000137	036220		JMP	@#GGDONE
6965						
6966	035322				GGER12:	
6967	035322	010537	001252		MOV	R5,@#STMP10
6968	035326	010437	001254		MOV	R4,@#STMP11
6969	035332	012737	036130	001240	MOV	#GGP3,@#STMP3
6970	035340	012737	036120	001242	MOV	#GGP2,@#STMP4
6971	035346	012737	036100	001244	MOV	#GGDATO,@#STMP5
6972	035354	012737	036170	001246	MOV	#GGP7,@#STMP6
6973	035362	104002			1\$:	ERROR
6974	035364	000137	036220		JMP	@#GGDONE
6975						
6976	035370				GGER13:	
6977	035370	010537	001252		MOV	R5,@#STMP10
6978	035374	010437	001254		MOV	R4,@#STMP11
6979	035400	012737	036130	001240	MOV	#GGP3,@#STMP3
6980	035406	012737	036120	001242	MOV	#GGP2,@#STMP4
6981	035414	012737	036100	001244	MOV	#GGDATO,@#STMP5
6982	035422	012737	036170	001246	MOV	#GGP7,@#STMP6

6983	035430	104002			1\$:	ERROR	2
6984	035432	000137	036220			JMP	@#GGDONE
6985							
6986	035436	013701	001236		GGER14:	MOV	@#STMP2,R1
6987	035442	062701	000002			ADD	#2,R1
6988	035446	020116				CMP	R1,(SP)
6989	035450	001402				BEQ	10\$
6990	035452	000137	062534		5\$:	JMP	@#FPSPUR
6991	035456				10\$:		
6992	035456	170301				STST	R1
6993	035460	020127	000012			CMP	R1,#12
6994	035464	001372				BNE	5\$
6995	035466	022626				CMP	(SP)+,(SP)+
6996	035470	012700	036100			MOV	#GGDATO,R0
6997	035474	174010				STD	ACO,(R0)
6998							
6999	035476	012737	036110	001240		MOV	#GGP1,@#STMP3
7000	035504	012737	036130	001242		MOV	#GGP3,@#STMP4
7001	035512	012737	036100	001244		MOV	#GGDATO,@#STMP5
7002	035520	012737	036160	001246		MOV	#GGP6,@#STMP6
7003	035526	104002			1\$:	ERROR	2
7004	035530	000137	036220			JMP	@#GGDONE
7005							
7006	035534				GGER15:		
7007	035534	010537	001252			MOV	R5,@#STMP10
7008	035540	010437	001254			MOV	R4,@#STMP11
7009	035544	012737	036120	001240		MOV	#GGP2,@#STMP3
7010	035552	012737	036200	001242		MOV	#GGP8,@#STMP4
7011	035560	012737	036100	001244		MOV	#GGDATO,@#STMP5
7012	035566	012737	036160	001246		MOV	#GGP6,@#STMP6
7013	035574	104002			1\$:	ERROR	2
7014	035576	000137	036220			JMP	@#GGDONE
7015							
7016	035602				GGER16:		
7017	035602	010537	001252			MOV	R5,@#STMP10
7018	035606	010437	001254			MOV	R4,@#STMP11
7019	035612	012737	036120	001240		MOV	#GGP2,@#STMP3
7020	035620	012737	036200	001242		MOV	#GGP8,@#STMP4
7021	035626	012737	036100	001244		MOV	#GGDATO,@#STMP5
7022	035634	012737	036160	001246		MOV	#GGP6,@#STMP6
7023	035642	104002			1\$:	ERROR	2
7024	035644	000137	036220			JMP	@#GGDONE
7025							
7026	035650				GGER17:		
7027	035650	010537	001252			MOV	R5,@#STMP10
7028	035654	010437	001254			MOV	R4,@#STMP11
7029	035660	012737	036120	001240		MOV	#GGP2,@#STMP3
7030	035666	012737	036200	001242		MOV	#GGP8,@#STMP4
7031	035674	012737	036100	001244		MOV	#GGDATO,@#STMP5
7032	035702	012737	036210	001246		MOV	#GGP9,@#STMP6
7033	035710	104002			1\$:	ERROR	2
7034	035712	000137	036220			JMP	@#GGDONE
7035							
7036	035716				GGER18:		
7037	035716	010537	001252			MOV	R5,@#STMP10
7038	035722	010437	001254			MOV	R4,@#STMP11

```

7039 035726 012737 036120 001240      MOV      #GGP2,@#STMP3
7040 035734 012737 036200 001242      MOV      #GGP8,@#STMP4
7041 035742 012737 036100 001244      MOV      #GGDAT0,@#STMP5
7042 035750 012737 036210 001246      MOV      #GGP9,@#STMP6
7043 035756 104002          1$:      ERROR  2
7044 035760 000137 036220      JMP      @#GGDONE
7045
7046 035764          GGER19:
7047 035764 010537 001252      MOV      R5,@#STMP10
7048 035770 010437 001254      MOV      R4,@#STMP11
7049 035774 012737 036120 001240      MOV      #GGP2,@#STMP3
7050 036002 012737 036200 001242      MOV      #GGP8,@#STMP4
7051 036010 012737 036100 001244      MOV      #GGDAT0,@#STMP5
7052 036016 012737 036210 001246      MOV      #GGP9,@#STMP6
7053 036024 104002          1$:      ERROR  2
7054 036026 000137 036220      JMP      @#GGDONE
7055
7056 036032          GGER20:
7057 036032 010537 001252      MOV      R5,@#STMP10
7058 036036 010437 001254      MOV      R4,@#STMP11
7059 036042 012737 036120 001240      MOV      #GGP2,@#STMP3
7060 036050 012737 036200 001242      MOV      #GGP8,@#STMP4
7061 036056 012737 036100 001244      MOV      #GGDAT0,@#STMP5
7062 036064 012737 036210 001246      MOV      #GGP9,@#STMP6
7063 036072 104002          1$:      ERROR  2
7064 036074 000137 036220      JMP      @#GGDONE
7065
7066 036100 000000      GGDATA: 0
7067 036102 000000          0
7068 036104 000000          0
7069 036106 000000          0
7070
7071 036110 000300      GGP1:   300
7072 036112 000000          0
7073 036114 000000          0
7074 036116 000000          0
7075 036120 100200      GGP2:  100200
7076 036122 000000          0
7077 036124 000000          0
7078 036126 000000          0
7079 036130 000200      GGP3:   200
7080 036132 000000          0
7081 036134 000000          0
7082 036136 000001          1
7083 036140 010200      GGP4:  10200
7084 036142 000000          0
7085 036144 000000          0
7086 036146 000000          0
7087 036150 077600      GGP5:  77600      ;OVER FLOW = GGP5 + GGP5
7088 036152 000000          0
7089 036154 000000          0
7090 036156 000000          0
7091 036160 000000      GGP6:   0      ;OVERFLOW RESULT
7092 036162 000000          0      ;UNDERFLOW RESULT
7093 036164 000000          0      ;GGP6 = GGP4 + GGP5
7094 036166 000000          0      ;      = GGP3 + GGP2 (FIU = 0)

```

7095  
 7096 036170 062400  
 7097 036172 000000  
 7098 036174 000000  
 7099 036176 000000  
 7100 036200 000340  
 7101 036202 000000  
 7102 036204 000000  
 7103 036206 000000  
 7104 036210 000100  
 7105 036212 000000  
 7106 036214 000000  
 7107 036216 000000  
 7108 036220  
 7109 036220 104413  
 7110  
 7111  
 7112  
 7113  
 7114  
 7115  
 7116  
 7117  
 7118  
 7119  
 7120  
 7121  
 7122 036222 000004  
 7123  
 7124 036224  
 7125 036224 104414  
 7126 036226 012704 000200  
 7127 036232 170104  
 7128 036234 012700 037640  
 7129 036240 172410  
 7130 036242 012700 037650  
 7131 036246 012737 036254 001236  
 7132 036254 177420  
 7133 036256 020027 037654  
 7134 036262 001402  
 7135 036264 000137 037216  
 7136 036270  
 7137 036270 170205  
 7138 036272 012700 037630  
 7139 036276 174010  
 7140 036300 012701 037720  
 7141 036304 012702 000004  
 7142 036310 022120  
 7143 036312 001415  
 7144 036314 012701 037650  
 7145 036320 012700 037630  
 7146 036324 012702 000004  
 7147 036330 022120  
 7148 036332 001402  
 7149 036334 000137 037256  
 7150 036340 077205

```

GGP7: 62400          ; = GGP3 + GGP1
          0          ; GGP7 = GGP3 + GGP2 (FIU = 1)
          0
          0
GGP8: 340
          0
          0
GGP9: 100
          0
          0
GGDONE: RSETIP      ;GO INITIALIZE THE FPS AND STACK; AND
                    ;SEE IF THE USER HAS EXPRESSED
                    ;THE DESIRE TO CHANGE THE SOFTWARE
                    ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                    ;THE USER TYPED CONTROL G?).

;*****
;*TEST 41          LDCFD AND LDCDF TEST
;*
;*THIS IS A TEST OF LDCFD AND LDCDF.
;*
;*****
TST41: SCOPE
;TEST FOR CORRECT AUTO INCREMENT CONSTANT.
HX1:
    LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
    MOV #200,R4    ;SET LONG INTEGER MODE
    LDFPS R4
    MOV #HXP1,R0
    LDD (R0),ACO
    MOV #HXP2,R0
    MOV #HX2,@#STMP2
HX2: LDCFD (R0)+,ACO
    CMP R0,#HXP2+4 ;IS R0 CORRECT
    BEQ HX3
    JMP @#HXER1
HX3:
    STFPS R5      ;GET FPS
    MOV #HXDAT0,R0
    STD ACO,(R0)  ;GET ACO
    MOV #HXP7,R1  ;SEE IF RESULT IS
    MOV #4,R2     ;CORRECT
HX4: CMP (R1)+,(R0)+
    BEQ HX7
    MOV #HXP2,R1  ;DID FD GET
    MOV #HXDAT0,R0 ;COMPLIMENTED?
    MOV #4,R2
HX5: CMP (R1)+,(R0)+
    BEQ HX6
    JMP @#HXER2
HX6: SOB R2,HX5
    
```

7151	036342	000137	037306		JMP	@#HXER3	
7152	036346	077220		HX7:	SOB	R2,HX4	
7153	036350	012704	000200		MOV	#200,R4	;FPS CORRECT?
7154	036354	020405			CMP	R4,R5	
7155	036356	001402			BEQ	HX8	
7156	036360	000137	037354		JMP	@#HXER8	
7157				;NOW	TEST	LDCDF	
7158	036364			HX8:			
7159	036364	104414			LPERR		;SET UP THE LOOP ON ERROR ADDRESS.
7160	036366	012704	000200			MOV #200,R4	
7161	036372	170104				LDFPS R4	
7162							
7163	036374	012700	037640		MOV	#HXP1,R0	
7164	036400	172410			LDD	(R0),AC0	
7165							
7166	036402	012700	037650		MOV	#HXP2,R0	
7167	036406	012737	036416	001236	MOV	#HX9,@#TMP2	
7168							
7169	036414	170001			SETF		
7170							
7171	036416	177420		HX9:	LDCDF	(R0)+,AC0	;TEST INSTRUCTION
7172							
7173	036420	020027	037660		CMP	R0,#HXP2+10	;WAS A GOOD
7174	036424	001402			BEQ	HX10	;CONSTANT USED
7175	036426	000137	037236		JMP	@#HXER5	;TO INCREMENT R0?
7176							
7177	036432			HX10:			
7178	036432	170205			STFPS	R5	
7179	036434	012700	037630		MOV	#HXDAT0,R0	
7180	036440	170011			SETD		
7181	036442	174010			STD	AC0,(R0)	;GET RESULT
7182	036444	012701	037730		MOV	#HXP8,R1	
7183	036450	012702	000004		MOV	#4,R2	
7184	036454	022120		HX11:	CMP	(R1)+,(R0)+	;IS IT CORRECT?
7185	036456	001415			BEQ	HX14	
7186							
7187	036460	012701	037720		MOV	#HXP7,R1	
7188	036464	012700	037630		MOV	#HXDAT0,R0	
7189	036470	012702	000004		MOV	#4,R2	
7190	036474	022110		HX12:	CMP	(R1)+,(R0)	;DID FD FAIL TO GET
7191	036476	001402			BEQ	HX13	;COMPLIMENTED?
7192	036500	000137	037372		JMP	@#HXER6	
7193	036504	077205		HX13:	SOB	R2,HX12	
7194	036506	000137	037422		JMP	@#HXER7	
7195							
7196	036512	077220		HX14:	SOB	R2,HX11	
7197							
7198	036514	012704	000000		MOV	#0,R4	;FPS CORRECT?
7199	036520	020405			CMP	R4,R5	
7200	036522	001402			BEQ	HX15	
7201	036524	000137	037354		JMP	@#HXER8	
7202							
7203							
7204							
7205	036530			HX15:			
7206	036530	104414			LPERR		;SET UP THE LOOP ON ERROR ADDRESS.

;TEST GR7 IMMEDIATE MODE CONSTANT

```

7207
7208 036532 012704 000200      MOV      #200,R4
7209 036536 170104      LDFPS   R4                ;SET FD
7210 036540 012737 036556 001236      MOV      #HX16,@#$TMP2
7211 036546 012737 037452 000004      MOV      #HXER9,@#ERRVECT
7212 036554 005001      CLR      R1
7213 036556 177427 043243      HX16:   LDCFD   #5201,AC0
7214 036562 005201      HX165:  INC      R1
7215 036564 005201      INC      R1
7216 036566 005201      INC      R1
7217 036570 012737 062566 000004      MOV      #CPSPUR,@#ERRVECT
7218 036576 020127 000003      CMP      R1,#3            ;SEE IF PC WAS
7219 036602 001402      BEQ      HX17            ;CORRECT
7220 036604 000137 037506      JMP      @#HXER10
7221
7222 036610      HX17:
7223 036610 104414      LPERR                    ;SET UP THE LOOP ON ERROR ADDRESS.
7224
7225 036612 012704 000200      MOV      #200,R4
7226 036616 170104      LDFPS   R4
7227 036620 012737 036646 001236      MOV      #HX18,@#$TMP2
7228 036626 012700 037710      MOV      #HXP6,R0
7229 036632 172410      LDD     (R0),AC0
7230 036634 012737 062566 000004      MOV      #CPSPUR,@#ERRVECT
7231 036642 012700 037650      MOV      #HXP2,R0
7232 036646 177410      HX18:  LDCFD   (R0),AC0
7233
7234 036650 012700 037630      MOV      #HXDAT0,R0
7235 036654 174010      STD     AC0,(R0)         ;GET RESULT.
7236 036656 012701 037720      MOV      #HXP7,R1
7237 036662 012702 000004      MOV      #4,R2
7238 036666 022021      HX19:  CMP      (R0)+,(R1)+   ;IS RESULT CORRECT?
7239 036670 001402      BEQ      HX20
7240 036672 000137 037256      JMP      @#HXER2
7241 036676 077205      HX20:  SOB     R2,HX19
7242
7243      ;TEST LDCFD WITH NEGATIVE OPERAND
7244 036700      HX21:
7245 036700 104414      LPERR                    ;SET UP THE LOOP ON ERROR ADDRESS.
7246 036702 012704 000200      MOV      #200,R4
7247 036706 170104      LDFPS   R4
7248 036710 012737 036730 001236      MOV      #HX22,@#$TMP2
7249 036716 012700 037710      MOV      #HXP6,R0
7250 036722 172410      LDD     (R0),AC0
7251 036724 012700 037670      MOV      #HXP4,R0
7252 036730 177410      HX22:  LDCFD   (R0),AC0
7253
7254 036732 012700 037630      MOV      #HXDAT0,R0
7255 036736 174010      STD     AC0,(R0)         ;GET RESULT
7256
7257 036740 012701 037700      MOV      #HXP5,R1
7258 036744 012702 000004      MOV      #4,R2
7259 036750 022120      HX23:  CMP      (R1)+,(R0)+
7260 036752 001415      BEQ      HX26
7261
7262 036754 012701 037720      MOV      #HXP7,R1
    
```

7263	036760	012700	037630		MOV	#HXDATO,R0	
7264	036764	012702	000004		MOV	#4,R2	
7265	036770	022120		HX24:	CMP	(R1)+,(R0)+	;WAS SIGN INCORRECT
7266	036772	001402			BEQ	HX25	
7267	036774	000137	037540		JMP	@#HXER11	
7268	037000	077205		HX25:	SOB	R2,HX24	
7269	037002	000137	037560		JMP	@#HXFR12	
7270							
7271	037006	077220		HX26:	SOB	R2,HX23	
7272							
7273					;TEST	LDCFD	0
7274							
7275	037010			HX27:			
7276	037010	104414			LPERR		;SET UP THE LOOP ON ERROR ADDRESS.
7277	037012	012704	000200		MOV	#200,R4	
7278	037016	170104			LDFPS	R4	
7279							
7280	037020	012700	037640		MOV	#HXP1,R0	
7281	037024	172410			LDD	(R0),AC0	
7282	037026	172010			ADDD	(R0),AC0	
7283							
7284	037030	012737	037042	001236	MOV	#HX28,@#STMP2	
7285	037036	012700	037640		MOV	#HXP1,R0	
7286	037042	177410		HX28:	LDCFD	(R0),AC0	
7287							
7288	037044	170205			STFPS	R5	
7289							
7290	037046	012700	037630		MOV	#HXDATO,R0	
7291	037052	174010			STD	AC0,(R0)	;GET RESULT
7292							
7293	037054	012701	037640		MOV	#HXP1,R1	
7294	037060	012702	000004		MOV	#4,R2	
7295	037064	022120		HX29:	CMP	(R1)+,(R0)+	;IS IT 0?
7296	037066	001402			BEQ	HX30	
7297	037070	000137	037610		JMP	@#HXER13	
7298	037074	077205		HX30:	SOB	R2,HX29	
7299							
7300	037076	012704	000204		MOV	#204,R4	;FPS CORRECT
7301	037102	020405			CMP	R4,R5	
7302	037104	001402			BEQ	HX31	
7303	037106	000137	037336		JMP	@#HXER4	
7304							
7305					;TEST	LDCFD	0
7306							
7307	037112			HX31:			
7308	037112	104414			LPERR		;SET UP THE LOOP ON ERROR ADDRESS.
7309	037114	012704	000200		MOV	#200,R4	
7310	037120	170104			LDFPS	R4	
7311							
7312	037122	012700	037710		MOV	#HXP6,R0	
7313	037126	172410			LDD	(R0),AC0	
7314							
7315	037130	012737	037142	001236	MOV	#HX32,@#STMP2	
7316	037136	012700	037640		MOV	#HXP1,R0	
7317	037142	177410		HX32:	LDCFD	(R0),AC0	
7318							



```

7319 037144 170205          STFPS  R5
7320
7321 037146 012700 037630      MOV   #HXDATO,R0
7322 037152 174010          STD   AC0,(R0)          ;GET RESULT
7323
7324 037154 012701 037640      MOV   #HXP1,R1
7325 037160 012702 000004      MOV   #4,R2
7326 037164 022120          HX33: CMP   (R1)+,(R0)+  ;IS IT ZERO?
7327 037166 001402          BEQ   HX34
7328 037170 000137 037610      JMP   @#HXER13
7329 037174 077205          HX34: SOB  R2,HX33
7330
7331 037176 012704 000204      MOV   #204,R4          ;FPS CORRECT?
7332 037202 020405          CMP   R4,R5
7333 037204 001402          BEQ   HX35
7334 037206 000137 037336      JMP   @#HXER4
7335 037212 000137 037740      HX35: JMP   @#HXDONE
7336
7337          ;RO INCORRECT
7338
7339 037216 012737 037654 001242  HXER1: MOV   #HXP2+4,@#STMP4
7340 037224 010037 001240          MOV   R0,@#STMP3
7341 037230 104002          1$:  ERROR 2
7342 037232 000137 037740          JMP   @#HXDONE
7343
7344 037236 012737 037660 001242  HXER5: MOV   #HXP2+10,@#STMP4
7345 037244 010037 001240          MOV   R0,@#STMP3
7346 037250 104002          1$:  ERROR 2
7347 037252 000137 037740          JMP   @#HXDONE
7348          ;REPORT BAD DATA
7349 037256 012737 037650 001244  HXER2: MOV   #HXP2,@#STMP5
7350 037264 012737 037720 001250          MOV   #HXP7,@#STMP7
7351 037272 012737 037630 001246  HXER22: MOV  #HXDATO,@#STMP6
7352 037300 104002          1$:  ERROR 2
7353 037302 000137 037740          JMP   @#HXDONE
7354          ;
7355 037306 012737 037650 001244  HXER3: MOV   #HXP2,@#STMP5
7356 037314 012737 037720 001250          MOV   #HXP7,@#STMP7
7357 037322 012737 037630 001246  HXER33: MOV  #HXDATO,@#STMP6
7358 037330 104002          1$:  ERROR 2
7359 037332 000137 037740          JMP   @#HXDONE
7360
7361 037336 010537 001240          HXER4: MOV   R5,@#STMP3
7362 037342 010437 001242          MOV   R4,@#STMP4
7363 037346 104002          1$:  ERROR 2
7364 037350 000137 037740          JMP   @#HXDONE
7365
7366 037354 010537 001240          HXER8: MOV   R5,@#STMP3
7367 037360 010437 001242          MOV   R4,@#STMP4
7368 037364 104002          1$:  ERROR 2
7369 037366 000137 037740          JMP   @#HXDONE
7370 037372 012737 037650 001244  HXER6: MOV   #HXP2,@#STMP5
7371 037400 012737 037730 001250          MOV   #HXP8,@#STMP7
7372 037406 012737 037630 001246  HXER66: MOV  #HXDATO,@#STMP6
7373 037414 104002          1$:  ERROR 2
7374 037416 000137 037740          JMP   @#HXDONE
    
```

```

7375
7376 037422 012737 037650 001244 HXER7: MOV #HXP2,@#STMP5
7377 037430 012737 037730 001250 MOV #HXP8,@#STMP7
7378 037436 012737 037630 001246 MOV #HXDATO,@#STMP6
7379 037444 104002 1$: ERROR 2
7380 037446 000137 037740 JMP @#HXDONE
7381
7382 037452 032716 000001 HXER9: BIT #1,(SP) ;SEE IF IT
7383 037456 001005 BNE 1$ ;AN ODD ADDRESS
7384 037460 022716 036562 CMP #HX165,(SP)
7385 037464 001402 BEQ 1$
7386 037466 000137 062566 JMP @#CPSPUR
7387
7388 037472 011637 001236 1$: MOV (SP),@#STMP2
7389 037476 022626 CMP (SP)+,(SP)+
7390 037500 104002 2$: ERROR 2
7391 037502 000137 037740 JMP @#HXDONE
7392
7393 037506 162701 000003 HXER10: SUB #3,R1
7394 037512 006301 ASL R1
7395 037514 012702 036562 MOV #HX165,R2
7396 037520 010237 001242 MOV R2,@#STMP4
7397 037524 160102 SUB R1,R2
7398 037526 010237 001240 MOV R2,@#STMP3
7399 037532 104002 1$: ERROR 2
7400 037534 000137 037740 JMP @#HXDONE
7401
7402 037540 012737 037670 001244 HXER11: MOV #HXP4,@#STMP5
7403 037546 012737 037700 001250 MOV #HXP5,@#STMP7
7404 037554 000137 037272 JMP @#HXER22
7405 037560 012737 037670 001244 HXER12: MOV #HXP4,@#STMP5
7406 037566 012737 037700 001250 MOV #HXP5,@#STMP7
7407 037574 012737 037630 001246 MOV #HXDATO,@#STMP6
7408 037602 104002 1$: ERROR 2
7409 037604 000137 037740 JMP @#HXDONE
7410
7411 037610 012737 037640 001244 HXER13: MOV #HXP1,@#STMP5
7412 037616 012737 037640 001250 MOV #HXP1,@#STMP7
7413 037624 000137 037272 JMP @#HXER22
7414
7415 037630 000000 HXDATO: 0
7416 037632 000000 0
7417 037634 000000 0
7418 037636 000000 0
7419
7420 037640 000000 HXP1: 0
7421 037642 000000 0
7422 037644 000000 0
7423 037646 000000 0
7424
7425 037650 000577 HXP2: 577
7426 037652 177776 177776
7427 037654 177777 177777
7428 037656 177776 177776
7429 037660 005201 HXP3: 5201
7430 037662 000000 0
    
```

```

7431 037664 000000
7432 037666 000000
7433 037670 100577
7434 037672 177776
7435 037674 177777
7436 037676 177776
7437 037700 100577
7438 037702 177776
7439 037704 000000
7440 037706 000000
7441 037710 000252
7442 037712 125252
7443 037714 125252
7444 037716 125252
7445
7446 037720 000577
7447 037722 177776
7448 037724 000000
7449 037726 000000
7450 037730 000577
7451 037732 177777
7452 037734 000000
7453 037736 000000
7454
7455 037740
7456 037740 104413
7457
7458
7459
7460
7461
7462
7463
7464
7465
7466
7467
7468
7469
7470
7471
7472
7473 037742 000004
7474
7475
7476 037744
7477 037744 104414
7478 037746 004737 040556
7479 037752 000000 000000 000000 1$:
7480 037760 000000
7481 037762 000000 000000 000000 2$:
7482 037770 000000
7483 037772 000200 3$:
7484 037774 000204
7485 037776 000200
7486 040000 104002 4$:
    
```

```

HXP4: 100577
177776
177777
177776
HXP5: 100577
177776
0
0
HXP6: 252
125252
125252
125252
HXP7: 577
177776
0
0
HXP8: 577
177777
0
0
HXDONE:
    
```

```

;GO INITIALIZE THE FPS AND STACK; AND
;SEE IF THE USER HAS EXPRESSED
;THE DESIRE TO CHANGE THE SOFTWARE
;VIRTUAL CONSOLE SWITCH REGISTER (HAS
;THE USER TYPED CONTROL G?).
    
```

```

*****
*TEST 42      CMPD TEST
*
*THIS IS A TEST OF THE CMPD INSTRUCTION. NOTE THAT A SUBROUTINE
*IS USED TO SET UP OPERANDS, EXECUTE THE INSTRUCTION AND CHECK THE
*RESULTS
*
*****
    
```

```

TST42: SCOPE
;TEST THE CMPD INSTRUCTION WITH (FSRC=AC=0)
AAA1:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,@#CMPSUB
.WORD 0,0,0,0 ;AC0
.WORD 0,0,0,0 ;FSRC
200 ;FPS BEFORE EXECUTION
204 ;FPS AFTER EXECUTION
200 ;ERROR FPS
ERROR 2 ;FPS ERROR
    
```

```

7487
7488
7489
7490 040002 ;TEST CMPD WITH (AC=0) AND FSRC POSITIVE.
AAA2:
7491 040002 104414 LPERR ;SET UP THE LOOP ON FPROR ADDRESS.
7492 040004 004737 040556 JSR PC,@#CMPSUB
7493 040010 000000 000000 000000 1$: .WORD 0,0,0,0 ;AC
7494 040016 000000
7495 040020 025252 2$: 25252 ;FSRC
7496 040022 052525 52525
7497 040024 125252 125252
7498 040026 052525 52525
7499 040030 000200 3$: 200 ;FPS BEFORE EXECUTION
7500 040032 000200 200 ;FPS AFTER EXECUTION
7501 040034 000210 210 ;ERROR FPS
7502 040036 1040C2 4$: ERROR 2
7503
7504 ;TEST CMPD WITH (AC=0) AND FSRC NEGATIVE
AAA3:
7505 040040 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
7506 040040 104414 JSR PC,@#CMPSUB
7507 040042 004737 040556 JSR PC,@#CMPSUB
7508 040046 000000 000000 000000 1$: .WORD 0,0,0,0 ;AC
7509 040054 000000
7510 040056 125252 2$: 125252 ;FSRC
7511 040060 125252 125252
7512 040062 052525 52525
7513 040064 125252 125252
7514 040066 000200 3$: 200 ;FPS BEFORE EXECUTION
7515 040070 000210 210 ;FPS AFTER EXECUTION
7516 040072 000200 200 ;ERROR FPS
7517 040074 104002 4$: ERROR 2
7518
7519 ;TEST CMPD WITH (FSRC=0) AND AC POSITIVE
AAA4:
7520 040076 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
7521 040076 104414 JSR PC,@#CMPSUB
7522 040100 004737 040556 JSR PC,@#CMPSUB
7523 040104 025252 1$: 25252 ;AC
7524 040106 052525 52525
7525 040110 125252 125252
7526 040112 052525 52525
7527 040114 000000 000000 000000 2$: .WORD 0,0,0,0 ;FSRC
7528 040122 000000
7529 040124 000200 3$: 200 ;FPS BEFORE EXECUTION
7530 040126 000210 210 ;FPS AFTER EXECUTION
7531 040130 000200 200 ;ERROR FPS
7532 040132 104002 4$: ERROR 2
7533
7534 ;TEST CMPD WITH (FSRC=0) AND AC NEGATIVE
AAA5:
7535 040134 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
7536 040134 104414 JSR PC,@#CMPSUB
7537 040134 104414 JSR PC,@#CMPSUB
7538 040136 004737 040556 JSR PC,@#CMPSUB
7539 040142 125252 1$: 125252 ;AC
7540 040144 125252 125252
7541 040146 052525 52525
7542 040150 125252 125252
    
```

```

7543 040152 000000 000000 000000 2$: .WORD 0,0,0,0 ;FSRC
7544 040160 000000
7545 040162 000200 3$: 200 ;FPS BEFORE EXECUTION
7546 040164 000200 200 ;FPS AFTER EXECUTION
7547 040166 000210 210 ;ERROR FPS
7548 040170 104002 4$: ERROR 2
7549
7550 ;TEST CMPD WITH AC POSITIVE AND FSRC NEGATIVE
7551 040172 AAA6:
7552 040172 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
7553 040174 004737 040556 JSR PC,@#CMPSUB
7554 040200 052525 1$: 52525 ;AC
7555 040202 125252 125252
7556 040204 052525 52525
7557 040206 125252 125252
7558 040210 125252 2$: 125252 ;:FSRC
7559 040212 052525 52525
7560 040214 125252 125252
7561 040216 052525 52525
7562 040220 000200 3$: 200 ;FPS BEFORE EXECUTION
7563 040222 000210 210 ;FPS AFTER EXECUTION
7564 040224 000200 200 ;ERROR FPS
7565 040226 104002 4$: ERROR 2
7566
7567 ;TEST CMPD WITH AC NEGATIVE AND FSRC POSITIVE
7568 AAA7:
7569 040230 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
7570 040230 104414 JSR PC,@#CMPSUB
7571 040232 004737 040556 125252 1$: 125252 ;AC
7572 040236 125252 52525
7573 040240 052525 125252
7574 040242 125252 52525
7575 040244 052525 2$: 52525 ;FSRC
7576 040246 052525 125252
7577 040250 125252 52525
7578 040252 052525 125252
7579 040254 125252 3$: 200 ;FPS BEFORE EXECUTION
7580 040256 000200 200 ;FPS AFTER EXECUTION
7581 040260 000200 210 ;ERROR FPS
7582 040262 000210 4$: ERROR 2
7583 040264 104002
7584
7585 ;TEST CMPD WITH AC POSITIVE AND FSRC POSITIVE
7586 ;AND EAC LESS THAN EFSRC.
7587 040266 AAA8:
7588 040266 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
7589 040270 004737 040556 JSR PC,@#CMPSUB
7590 040274 012345 1$: 12345 ;AC
7591 040276 067654 67654
7592 040300 032101 32101
7593 040302 023456 23456
7594 040304 023456 2$: 23456 ;FSRC
7595 040306 076543 76543
7596 040310 021012 21012
7597 040312 034567 34567
7598 040314 000200 3$: 200 ;FPS BEFORE EXECUTION

```

```

7599 040316 000200          200          ;FPS AFTER EXECUTION
7600 040320 000210          210          ;ERROR FPS
7601 040322 104002          4$: ERROR 2
7602
7603
7604          ;TEST CMPD WITH AC POSITIVE, FSRC POSITIVE AND EAC GREATER THAN EFSRC
7605 040324 AAA9:
7606 040324 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
7607 040326 004737 040556 JSR PC,@#CMPSUB
7608 040332 045676          1$: 45676          ;AC
7609 040334 054321          54321
7610 040336 012345          12345
7611 040340 067654          67654
7612 040342 034567          2$: 34567          ;FSRC
7613 040344 065432          65432
7614 040346 101234          101234
7615 040350 056765          56765
7616 040352 000200          3$: 200          ;FPS BEFORE EXECUTION
7617 040354 000210          210          ;FPS AFTER EXECUTION
7618 040356 000200          200          ;ERROR FPS
7619 040360 104002          4$: ERROR 2
7620
7621          ;TEST CMPD WITH AC POSITIVE, FSRC POSITIVE AND AC EQUAL TO FSRC
7622 040362 AAA10:
7623 040362 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
7624 040364 004737 040556 JSR PC,@#CMPSUB
7625 040370 012345          1$: 12345          ;AC
7626 040372 067012          67012
7627 040374 034567          34567
7628 040376 012345          012345
7629 04040J 012345          2$: 12345          ;FSRC
7630 040402 067012          67012
7631 040404 034567          34567
7632 040406 012345          012345
7633 040410 000200          3$: 200          ;FPS BEFORE EXECUTION
7634 040412 000204          204          ;FPS AFTER EXECUTION
7635 040414 000200          200          ;ERROR FPS
7636 040416 104002          4$: ERROR 2
7637
7638          ;TEST CMPD WITH AC POSITIVE, FSRC POSITIVE, EAC EQUAL TO EFSRC,
7639          ;AND FSRC GREATER THAN AC.
7640 040420 AAA11:
7641 040420 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
7642 040422 004737 040556 JSR PC,@#CMPSUB
7643 040426 012345          1$: 12345          ;AC
7644 040430 067012          67012
7645 040432 034567          34567
7646 040434 012345          012345
7647 040436 012345          2$: 12345          ;FSRC
7648 040440 070123          70123
7649 040442 045670          45670
7650 040444 123456          123456
7651 040446 000200          3$: 200          ;FPS BEFORE EXECUTION
7652 040450 000200          200          ;FPS AFTER EXECUTION
7653 040452 000210          210          ;ERROR FPS
7654 040454 104002          4$: ERROR 2

```

```

7655
7656 ;TEST CMPD WITH AC POSITIVE, FSRC POSITIVE, EAC EQUAL TO EFSRC,
7657 ;AND AC GREATER THAN FSRC.
7658 040456 AAA12:
7659 040456 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
7660 040460 004737 040556 JSR PC,@#CMPSUB
7661 040464 054321 1$: 54321 ;AC
7662 040466 076543 76543
7663 040470 021076 21076
7664 040472 054321 54321
7665 040474 054321 2$: 54321 ;FSRC
7666 040476 065432 65432
7667 040500 107654 107654
7668 040502 032107 32107
7669 040504 000200 3$: 200 ;FPS BEFORE EXECUTION
7670 040506 000210 210 ;FPS AFTER EXECUTION
7671 040510 000200 200 ;ERROR FPS
7672 040512 104002 4$: ERROR 2
    
```

```

7673
7674 ;TEST CMPD WITH AC NEGATIVE, FSRC NEGATIVE, EAC EQUAL TO EFSRC,
7675 ;AND AC GREATER THAN FSRC
7676 040514 AAA13:
7677 040514 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
7678 040516 004737 040556 JSR PC,@#CMPSUB
7679 040522 112345 1$: 112345 ;AC
7680 040524 043210 43210
7681 040526 076543 76543
7682 040530 021076 21076
7683 040532 112345 2$: 112345 ;FSRC
7684 040534 054321 54321
7685 040536 007654 07654
7686 040540 032107 32107
7687 040542 000200 3$: 200 ;FPS BEFORE EXECUTION
7688 040544 000210 210 ;FPS AFTER EXECUTION
7689 040546 000200 200 ;ERROR FPS
7690 040550 104002 4$: ERROR ?
    
```

```

7691
7692
7693 040552 000137 040746 JMP @#AAADONE ;FINISHED CMPD TEST.
7694
7695
    
```

```

7696 ;THIS SUBROUTINE, CMPSUB, IS CALLED TO SET UP, EXECUTE
7697 ;AND CHECK THE RESULTS OF A CMPD INSTRUCTION.
7698 ;IT IS CALLED THUS:
7699 :
7700 :
7701 : ACARG: JSR PC,@#CMPSUB ;AC OPERAND
7702 : FSRCARG: .WORD X,X,X,X ;FSRC OPERAND
7703 : FPSB: .WORD X ;FPS BEFORE EXECUTION
7704 : FPSA: .WORD X ;FPS AFTER EXECUTION
7705 : FPSE: .WORD X ;ERROR FPS
7706 : ERR: ERROR X ;FPS ERROR
7707 : CONT: ;RETURN ADDRESS
7708 :
7709 ;THE OPERANDS ARE SET UP (USING AC0 FOR THE AC OPERAND). THEN
7710 ;FPSB IS LOADED INTO THE FPS. THE INSTRUCTION, CMPD, IS EXECUTED.
    
```

```

7711 ;AFTER THE EXECUTION THE FPS IS CHECKED AGAINST FPSA. IF IT IS A MATCH
7712 ;THEN THERE WAS NO ERROR AND CONTROL IS RETURNED TO CONT. IF
7713 ;THE FPS IS INCORRECT IT IS COMPARED WITH FPSE IN AN ATTEMPT TO ANALYSE
7714 ;THE FAILURE. IF THE FPS IS THE SAME AS FPSE THEN CONTROL IS
7715 ;RETURNED TO THE ERROR CALL AT LOCATION ERR. IF THE FPS WAS
7716 ;NOT CORRECT BUT DIDN'T MATCH FPSE A GENERAL ERROR IS REPORTED
7717 ;AND CONTROL IS PASSED TO CONT.
7718
7719 040556 012601      CMPSUB: MOV      (SP)+,R1      ;PICK UP A POINTER TO THE
7720                                ;ARGUMENTS.
7721 040560 016100 000020      MOV      20(R1),R0      ;GET THE FPS BEFORE EXECUTION.
7722 040564 170100                                LDFPS   R0              ;LOAD IT INTO THE FPS.
7723
7724 040566 012737 040610 001236      MOV      #1$,@#STMP2      ;SAVE ADDRESS OF CMPD INSTRUCTION.
7725 040574 010100                                MOV      R1,R0          ;GET ADDRESS OF AC OPERAND.
7726 040576 172410                                LDD      (R0),AC0       ;LOAD AC0 OPERAND
7727
7728 040600 010100                                MOV      R1,R0          ;COMPUTE FSRC OPERAND
7729 040602 062700 000010      ADD      #10,R0          ;ADDRESS
7730
7731 040606 000240                                NOP
7732 040610 173410      1$:  CMPD      (R0),AC0      ;FOR SCOPING.
7733                                ;EXECUTE THE TEST INSTRUCTION.
7734 040612 170205                                STFPS   R5              ;SAVE FPS AFTER INSTRUCTION.
7735
7736 040614 016104 000022      MOV      22(R1),R4      ;GET EXPECTED FPS.
7737                                ;IF INCORRECT SET UP FOR
7738 040620 010137 001240      MOV      R1,@#STMP3      ;AN ERROR CALL.
7739 040624 010137 001242      MOV      R1,@#STMP4
7740 040630 062737 000010 001242      ADD      #10,@#STMP4
7741 040636 010537 001244      MOV      R5,@#STMP5
7742 040642 010437 001246      MOV      R4,@#STMP6
7743 040646 020405                                CMP      R4,R5
7744 040650 001410      BEQ      3$              ;WAS FPS CORRECT?
7745                                ;BRANCH IF YES.
7746
7747 040652 026105 000024      CMP      24(R1),R5      ;WAS THE FPS THE SAME
7748                                ;AS THE EXPECTED INCORRECT FPS?
7749 040656 001003      BNE      2$              ;BRANCH IF NO MATCH.
7750
7751 040660 062701 000026      ADD      #26,R1
7752                                ;IF THE EXPECTED INCORRECT
7753 040664 000111      JMP      (R1)            ;FPS MATCHED THE RESULTANT FPS
7754                                ;RETURN TO THE ERROR CALL
7755                                ;IN THE CALLING ROUTINE.
7756 040666 104002      2$:  ERROR   2              ;OTHERWISE REPORT INCORRECT FPS
7757 040670 000411      BR      5$
7758
7759 040672 012700 040736      3$:  MOV      #CMPTMP,R0      ;IF FPS WAS CORRECT MAKE SURE
7760 040676 174010                                STD      AC0,(R0)       ;AC0 WAS NOT AFFECTED BY CMPD.
7761 040700 010102                                MOV      R1,R2
7762 040702 012703 000004      MOV      #4,R3
7763 040706 022220      4$:  CMP      (R2)+,(R0)+
7764 040710 001003                                BNE      6$
7765 040712 077303                                SOB      R3,4$
7766

```



```

7767 040714 000161 000030 5$: JMP 30(R1) ;RETURN
7768
7769 040720 6$: ;REPORT ACO MODIFIED BY CMPD
7770 040720 010137 001240 MOV R1,@#STMP3
7771 040724 012737 040736 001242 MOV #CMP TMP,@#STMP4
7772 040732 104002 7$: ERROR 2
7773 040734 000767 BR 5$ ;RETURN
7774
7775 040736 000000 000000 000000 CMP TMP: .WORD 0,0,0,0
7776 040744 000000
7777
7778
7779
7780 040746 AAADONE:
7781 040746 104413 RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
7782 ;SEE IF THE USER HAS EXPRESSED
7783 ;THE DESIRE TO CHANGE THE SOFTWARE
7784 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
7785 ;THE USER TYPED CONTROL G?).
7786
7787
7788
7789
7790
7791 ;*****
7792 ;*TEST 43 DIVD WITH (FSRC=0) AND (BUT FD) TEST
7793 ;*
7794 ;*THIS IS A TEST OF THE DIVD INSTRUCTION WITH A
7795 ;*ZERO DIVISOR. THE CONDITION IS CHECKED WITH BOTH
7796 ;*TRAP ENABLED AND TRAPS DISABLED.
7797 ;*
7798 ;*****
7798 040750 000004 TST43: SCOPE
7799
7800 ;FIRST TEST DIVD WITH (FSRC-AC 0) AND TRAPS DISABLED.
7801 040752 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
7802 040754 012704 040200 BBB0: MOV #40200,R4 ;SET UP FPS
7803 ;WITH INTERRUPTS
7804 ;DISABLED.
7805 040760 170104 LDFPS R4
7806 040762 012737 041224 000244 MOV #BBBER1,@#FPVECT;SET UP FOR ANY FP INTERRUPTS.
7807 040770 012737 041010 001236 MOV #BBB1,@#STMP2
7808 040776 012700 041430 MOV #BBBP1,R0 ;SET UP ACO = 0
7809 041002 172410 LDD (R0),ACO
7810 041004 012701 041430 MOV #BBBP1,R1 ;FSRC = 0
7811
7812 041010 174411 BBB1: DIVD (R1),ACO ;TEST INSTRUCTION
7813
7814 041012 170205 STFPS R5 ;GET FPS
7815 041014 170303 STST R3 ;GET FEC
7816
7817 041016 012704 140204 MOV #140204,R4 ;EXPECTED FPS.
7818 041022 020405 CMP R4,R5 ;IS FPS CORRECT.
7819 041024 001131 BNE BBBER2 ;IF INCORRECT BRANCH.
7820
7821 041026 012702 000004 MOV #4,R2 ;EXPECTED FEC.
7822 041032 020203 CMP R2,R3 ;IS FEC CORRECT?

```

```

7823 041034 001140          BNE      BBBER3          ;IF INCORRECT BRANCH.
7824
7825          ;TEST DIVD WITH (FSRC=0) AND TRAPS DISABLED.
7826 041036          BBB2:
7827 041036 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
7828 041040 012704 040200    MOV      #40200,R4      ;LOAD FPS WITH TRAPS DISABLED.
7829 041044 170104          LDFPS      R4
7830
7831 041046 012737 041066 001236    MOV      #BBB3,@#STMP2
7832 041054 012700 041440    MOV      #BBBP2,R0      ;SET UP ACO OPERAND (NON ZERO).
7833 041060 172410          LDD      (R0),ACO
7834 041062 012700 041430    MOV      #BBBP1,R0      ;FSRC=0
7835 041066 174410          BBB3: DIVD      (R0),ACO
7836
7837 041070 170205          STFPS      R5          ;GET FPS.
7838 041072 170303          STST      R3          ;GET FEC.
7839
7840 041074 012704 140200    MOV      #140200,R4      ;EXPECTED FPS.
7841 041100 020405          CMP      R4,R5          ;IS FPS CORRECT?
7842 041102 001102          BNE      BBBER2          ;IF INCORRECT BRANCH.
7843
7844 041104 012702 000004    MOV      #4,R2          ;EXPECTED FEC.
7845 041110 020203          CMP      R2,R3          ;WAS FEC CORRECT?
7846 041112 001111          BNE      BBBER3          ;IF INCORRECT BRANCH.
7847
7848          ;TEST DIVD WITH FSRC=0) AND TRAPS ENABLED.
7849 041114          BBB4:
7850 041114 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
7851 041116 012704 000200    MOV      #200,R4        ;SET UP FPS. TRAP ENABLED.
7852 041122 170104          LDFPS      R4
7853
7854 041124 012737 041152 001236    MOV      #BBB5,@#STMP2
7855 041132 012700 041440    MOV      #BBBP2,R0      ;SET UP ACO OPERAND (NON ZERO).
7856 041136 172410          LDD      (R0),ACO
7857
7858 041140 012737 041160 000244    MOV      #BBB6,@#FPVECT ;SET UP FOR THE EXPECTED INTERRUPT.
7859 041146 012700 041430    MOV      #BBBP1,R0      ;FSRC=0
7860
7861 041152 174410          BBB5: DIVD      (R0),ACO      ;TEST INSTRUCTION (SHOULD RESULT IN TRAP).
7862 041154 170000          CFCC
7863
7864 041156 000502          BR      BBBER4          ;GO REPORT FAILURE, NO TRAP.
7865
7866 041160 022716 041154          BBB6: CMP      #BBB5+2,(SP) ;TRAP TO HERE WHEN THE DIVISION BY 0
7867          ;OCCURS. FIRST SEE IF THE ADDRESS OF
7868          ;THE TRAP IS 2+THE ADDRESS OF THE TEST
7869          ;DIVD INSTRUCTION.
7870 041164 001402          BEQ      1$
7871 041166 000137 062534          JMP      @#FPSPUR      ;IF NOT THEN REPORT AN UNEXPECTED
7872          ;FP TRAP.
7873 041172 170205          1$: STFPS      R5          ;GET FPS.
7874 041174 170303          STST      R3          ;GET FEC.
7875 041176 022626          CMP      (SP)+,(SP)+   ;RESET THE STACK.
7876
7877 041200 012704 100200    MOV      #100200,R4      ;EXPECTED FPS.
7878 041204 020405          CMP      R4,R5          ;IS FPS CORRECT?
  
```

```

7879 041206 001040          BNE      BBBER2          ;IF INCORRECT BRANCH.
7880
7881 041210 012702 000004    MOV      #4,R2          ;EXPECTED FEC.
7882 041214 020203          CMP      R2,R3          ;IS FEC CORRECT?
7883 041216 001047          BNE      BBBER3          ;IF INCORRECT BRANCH.
7884
7885 041220 000137 041450    JMP      @#BBBDONE      ;OTHERWISE GO TO NEXT TEST.
7886
7887
7888                          ;TRAP HERE IF AN UNEXPECTED INTERRUPT OCCURS.
7889 041224 062737 000002 001236 BBBER1: ADD      #2,@#STMP2      ;SEE IF THE INTERRUPT OCCURRED
7890                          ;DURING THE EXECUTION OF THE DIVD
7891                          ;INSTRUCTION BEING TESTED.
7892 041232 021637 001236          CMP      (SP),@#STMP2
7893 041236 001402          BEQ      1$
7894 041240 000137 062534          JMP      @#FPSPUR        ;IF NOT REPORT UNEXPECTED FP TRAP.
7895
7896 041244 022626          1$:      CMP      (SP)+,(SP)+      ;RESET THE STACK.
7897 041246 170303          STST     R3              ;GET FEC.
7898 041250 170205          STFPS    R5              ;GET FPS.
7899 041252 012737 000004 001240    MOV      #4,@#STMP3      ;EXPECTED FEC.
7900 041260 010337 001242          MOV      R3,@#STMP4
7901 041264 010537 001244          MOV      R5,@#STMP5
7902 041270 010037 001250          MOV      R0,@#STMP7
7903 041274 012737 140200 001246    MOV      #140200,@#STMP6
7904 041302 104002          2$:      ERROR     2
7905                          ;WITH TRAPS DISABLED.
7906 041304 000137 041450          JMP      @#BBBDONE
7907
7908                          ;REPORT FPS INCORRECT:
7909 041310 010537 001242          BBBER2: MOV      R5,@#STMP4
7910 041314 010437 001244          MOV      R4,@#STMP5
7911 041320 010037 001246          MOV      R0,@#STMP6
7912 041324 010137 001250          MOV      R1,@#STMP7
7913 041330 104002          1$:      ERROR     2
7914 041332 000137 041450          JMP      @#BBBDONE
7915
7916                          ;REPORT FEC INCORRECT:
7917 041336 010337 001242          BBBER3: MOV      R3,@#STMP4
7918 041342 010237 001240          MOV      R2,@#STMP3
7919 041346 010037 001246          MOV      R0,@#STMP6
7920 041352 010137 001250          MOV      R1,@#STMP7
7921 041356 104002          1$:      ERROR     2
7922 041360 000137 041450          JMP      @#BBBDONE
7923
7924                          ;REPORT NO TRAP OCCURRED AFTER TRYING TO DIVIDE
7925                          ;BY ZERO WITH ALL TRAPS ENABLED.
7926 041364 170303          BBBER4: STST     R3              ;GET FEC.
7927 041366 170205          STFPS    R5              ;GET FPS.
7928 041370 012737 000004 001242    MOV      #4,@#STMP4
7929 041376 010337 001240          MOV      R3,@#STMP3
7930 041402 010537 001244          MOV      R5,@#STMP5
7931 041406 012737 100200 001246    MOV      #100200,@#STMP6
7932 041414 010037 001250          MOV      R0,@#STMP7
7933 041420 010137 001252          MOV      R1,@#STMP10
7934 041424 104002          1$:      ERROR     2
    
```

7935 041426 000410  
 7936  
 7937 041430 000000 000000 000000 BBBP1: .WORD 0,0,0,0  
 7938 041436 000000  
 7939 041440 012345 054321 023456 BBBP2: .WORD 12345,54321,23456,76543  
 7940 041446 076543  
 7941  
 7942  
 7943  
 7944 041450  
 7945 041450 1044\*3  
 7946  
 7947  
 7948  
 7949  
 7950  
 7951  
 7952  
 7953  
 7954  
 7955  
 7956  
 7957  
 7958  
 7959  
 7960  
 7961 041452 000004  
 7962  
 7963  
 7964 041454  
 7965 041454 104414  
 7966 041456 004767 000552  
 7967 041462 000000 000000  
 7968 041466 012345 067012  
 7969 041472 000000 000000  
 7970 041476 000000  
 7971 041500 000004  
 7972 041502 012345 067012  
 7973 041506 104002  
 7974  
 7975  
 7976 041510  
 7977 041510 104414  
 7978 041512 004737 042234  
 7979 041516 065652 125252  
 7980 041522 065600 000000  
 7981 041526 040252 125252  
 7982 041532 003000  
 7983 041534 003000  
 7984 041536 040052 125252  
 7985 041542 104002  
 7986  
 7987  
 7988 041544  
 7989 041544 104414  
 7990 041546 004767 000462

BR BBBDONE  
 BBBP1: .WORD 0,0,0,0  
 BBBP2: .WORD 12345,54321,23456,76543  
 BBBDONE:  
 RSETUP ;GO INITIALIZE THE FPS AND STACK; AND  
 ;SEE IF THE USER HAS EXPRESSED  
 ;THE DESIRE TO CHANGE THE SOFTWARE  
 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS  
 ;THE USER TYPED CONTROL G?).  
 ;\*\*\*\*\*  
 ;\*TEST 44 DIVF TEST  
 ;\*  
 ;\*THIS IS A TEST OF THE DIVF INSTRUCTION. NOTE THAT A SUBROUTINE IS  
 ;\*USED TO SET UP THE OPERANDS, EXECUTE THE INSTRUCTION AND CHECK THE  
 ;\*RESULTS.  
 ;\*  
 ;\*\*\*\*\*  
 TST44: SCOPE  
 ;CHECK DIVF WITH (AC=0).  
 CCC1:  
 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.  
 JSR PC,DIVFSUB  
 1\$: .WORD 0,0 ;AC  
 2\$: .WORD 12345,67012 ;FSRC  
 3\$: .WORD 0,0 ;RES  
 4\$: 0 ;FPS BEFORE EXECUTION.  
 4 ;FPS AFTER EXECUTION  
 5\$: .WORD 12345,67012 ;ERROR RESULT  
 6\$: ERROR 2  
 ;TEST DIVF WITH AC POSITIVE, FSRC POSITIVE AND IN ROUND MODE.  
 CCC2:  
 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.  
 JSR PC,DIVFSUB  
 1\$: .WORD 65652,125252 ;AC  
 2\$: .WORD 65600,0 ;FSRC  
 3\$: .WORD 40252,125252 ;RES  
 4\$: 3000 ;FPS BEFORE EXECUTION.  
 3000 ;FPS AFTER EXECUTION.  
 5\$: .WORD 40052,125252 ;ERROR RESULT.  
 6\$: ERROR 2  
 ;TEST DIVF WITH AC POSITIVE, FSRC POSITIVE.  
 CCC3:  
 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.  
 JSR PC,DIVFSUB

```

7991 041552 076400 000000 1$: .WORD 76400,0 ;AC
7992 041556 076400 000000 2$: .WORD 76400,0 ;FSRC
7993 041562 040200 000000 3$: .WORD 40200,0 ;RES
7994 041566 001000 4$: 1000 ;FPS BEFORE EXECUTION.
7995 041570 001000 1000 ;FPS AFTER EXECUTION.
7996 041572 140200 000000 5$: .WORD 140200,0 ;ERROR RES.
7997
7998 041576 104002 6$: ERROR 2 ;SIGN BAD
7999
8000 ;TEST DIVF WITH BOTH OPERANDS POSITIVE.
8001 041600 CCC4:
8002 041600 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
8003 041602 004737 042234 JSR PC,@DIVFSUB
8004 041606 056777 177777 1$: .WORD 56777,177777 ;AC
8005 041612 054200 000000 2$: .WORD 54200,0 ;FSRC
8006 041616 042777 177777 3$: .WORD 42777,177777 ;RES
8007 041622 000000 4$: 0 ;FPS BEFORE EXECUTION.
8008 041624 000000 0 ;FPS AFTER EXECUTION.
8009 041626 002000 002000 5$: .WORD 2000,2000 ;ERROR RES.
8010 041632 104002 6$: ERROR 2
8011
8012 ;TEST THE DIVF INSTRUCTION:
8013 041634 CCC5:
8014 041634 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
8015 041636 004737 042234 JSR PC,@DIVFSUB
8016 041642 012377 177777 1$: .WORD 12377,177777 ;AC
8017 041646 012300 000000 2$: .WORD 12300,0 ;FSRC
8018 041652 040252 125252 3$: .WORD 40252,125252 ;RES
8019 041656 000000 4$: 0 ;FPS BEFORE EXECUTION.
8020 041660 000000 0 ;FPS AFTER EXECUTION.
8021 041662 177777 177777 5$: .WORD -1,-1 ;ERROR RES.
8022 041666 104002 6$: ERROR 2
8023
8024 ;TEST DIVIDE ALGORITHM. TEST ROUND CONSTANT.
8025 041670 CCC6:
8026 041670 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
8027 041672 004737 042234 JSR PC,@DIVFSUB
8028 041676 064600 000001 1$: .WORD 64600,1 ;AC
8029 041702 066600 000000 2$: .WORD 66600,0 ;FSRC
8030 041706 036200 000001 3$: .WORD 36200,1 ;RES
8031 041712 000000 4$: 0 ;FPS BEFORE EXECUTION.
8032 041714 000000 0 ;FPS AFTER EXECUTION.
8033 041716 003000 003000 5$: .WORD 3000,3000 ;ERROR RES.
8034 041722 104002 6$: ERROR 2
8035
8036 ;TEST DIVF.
8037 041724 CCC7:
8038 041724 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
8039 041726 004737 042234 JSR PC,@DIVFSUB
8040 041732 034577 177776 1$: .WORD 34577,177776 ;AC
8041 041736 023400 000000 2$: .WORD 23400,0 ;FSRC
8042 041742 051377 177776 3$: .WORD 51377,177776 ;RES
8043 041746 000017 4$: 17 ;FPS BEFORE EXECUTION.
8044 041750 000000 0 ;FPS AFTER EXECUTION.
8045 041752 003400 003400 5$: .WORD 3400,3400 ;ERROR RES.
8046 041756 104002 6$: ERROR 2
    
```

```

8047
8048
8049 ;DIVF TEST.
8050 CCC8: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
8051 JSR PC,@DIVFSUB
8052 1$: .WORD 67652,125252 ;AC
8053 2$: .WORD 56500,0 ;FSRC
8054 3$: .WORD 51343,107070 ;RES
8055 4$: 0 ;FPS BEFORE EXECUTION.
8056 0 ;FPS AFTER EXECUTION.
8057 5$: .WORD 51543,107070 ;ERROR RES.
8058 6$: ERROR 2 ;DIDN'T INCREMENT THE EXPONENT
8059 ;AFTER DIVID NORMALIZATION.
8060
8061 ;DIVF WITH AC NEGATIVE, FSRC NEGATIVE.
8062 CCC9: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
8063 JSR PC,@DIVFSUB
8064 1$: .WORD 140400,0 ;AC
8065 2$: .WORD 140500,0 ;FSRC
8066 3$: .WORD 040052,125253 ;RES
8067 4$: 0 ;FPS BEFORE EXECUTION.
8068 0 ;FPS AFTER EXECUTION.
8069 5$: .WORD 140052,125253 ;ERROR RES.
8070 6$: ERROR 2 ;BAD SIGN.
8071
8072 ;DIVF WITH AC NEGATIVE AND FSRC POSITIVE.
8073 CCC10: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
8074 JSR PC,@DIVFSUB
8075 1$: .WORD 160077,0 ;AC
8076 2$: .WORD 40277,0 ;FSRC
8077 3$: .WORD 160000,0 ;RES
8078 4$: 7 ;FPS BEFORE EXECUTION.
8079 10 ;FPS AFTER EXECUTION.
8080 5$: .WORD 60000,0 ;ERROR RES.
8081 6$: ERROR 2 ;BAD SIGN.
8082
8083 ;DIVF WITH AC POSITIVE AND FSRC NEGATIVE.
8084 CCC11: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
8085 JSR PC,@DIVFSUB
8086 1$: .WORD 40400,0 ;AC
8087 2$: .WORD 140500,0 ;FSRC
8088 3$: .WORD 140052,125253 ;RES
8089 4$: 17 ;FPS BEFORE EXECUTION.
8090 10 ;FPS AFTER EXECUTION.
8091 5$: .WORD 40052,125253 ;ERROR RES.
8092 6$: ERROR 2 ;BAD SIGN.
8093
8094 ;TEST DIVF BOTH OPERANDS POSITIVE AND TRUNCATE MODE.
8095 CCC12: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
8096 JSR PC,@DIVFSUB
8097
8098
8099
8100
8101
8102

```

```

8103 042146 060100 000001 1$: .WORD 60100,1 ;AC
8104 042152 040300 000000 2$: .WORD 40300,0 ;FSRC
8105 042156 060000 000000 3$: .WORD 60000,0 ;RES
8106 042162 000052 4$: 52 ;FPS BEFORE EXECUTION.
8107 042164 000040 40 ;FPS AFTER EXECUTION.
8108 042166 060000 000001 5$: .WORD 60000,1 ;ERROR RES.
8109 042172 104002 6$: ERROR 2 ;TRUNCATION ERROR

```

8110  
8111 ;DIVF WITH POSITIVE OPERANDS AND ROUND MODE.

```

8112 042174 CCC13: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
8113 042174 104414 JSR PC,DIVFSUB
8114 042176 004767 000032 1$: .WORD 60100,1 ;AC
8115 042202 060100 000001 2$: .WORD 40300,0 ;FSRC
8116 042206 040300 000000 3$: .WORD 60000,1 ;RES
8117 042212 060000 000001 4$: 5 ;FPS BEFORE EXECUTION.
8118 042216 000005 0 ;FPS AFTER EXECUTION.
8119 042220 000000 5$: .WORD 60000,0 ;ERROR RES.
8120 042222 060000 000000 6$: ERROR 2 ;ROUND ERROR.

```

```

8121 042226 104002 JMP @#CCCDONE ;GO TO NEXT TEST.
8122
8123 042230 000137 042460
8124
8125
8126
8127
8128
8129
8130
8131
8132
8133
8134
8135
8136
8137
8138
8139
8140
8141
8142
8143
8144
8145
8146
8147
8148
8149
8150

```

;THIS SUBROUTINE, DIVFSUB, IS CALLED TO SET UP, EXECUTE AND CHECK THE RESULT OF A DIVF INSTRUCTION. IT IS CALLED THUS:

```

:
: JSR PC,@#DIVFSUB
: ACARG: .WORD X,X ;AC OPERAND
: FSRCARG: .WORD X,X ;FSRC OPERAND
: RES: .WORD X,X ;EXPECTED RESULT
: FPSB: .WORD X ;FPS BEFORE EXECUTION
: FPSA: .WORD X ;FPS AFTER EXECUTION
: ERRES: .WORD X,X ;ERROR RESULT
: ERR: ERROR X ;RESULT ERROR
: CONT: ;RETURN ADDRESS

```

```

;THE OPERANDS ARE SET UP (USING ACO FOR THE AC OPERAND). THEN
;FPSB IS LOADED INTO THE FPS. THE INSTRUCTION, DIVF IS EXECUTED.
;AFTER THE EXECUTION THE RESULT IS CHECKED AGAINST THE
;EXPECTED CORRECT RESULT, RES. IF IT IS CORRECT THEN THE FPS
;IS CHECKED WITH THE EXPECTED CORRECT FPS, FPSA. IF THE FPS WAS
;INCORRECT THEN IT IS REPORTED. IF THE RESULT WAS INCORRECT IT
;IS COMPARED WITH ERRES IN AN ATTEMPT TO ANALYSE THE ERROR. IF
;THE INCORRECT RESULT MATCHED ERRES THEN CONTROL IS PASSED TO
;THE ERROR CALL AT ERR. IF THE INCORRECT RESULT DID NOT MATCH ERRES
;THEN THE FAILURE IS REPORTED IN DIVFSUB AND CONTROL IS PASSED TO
;CONT. IF NO ERRORS ARE DETECTED THEN DIVFSUB RETURNS CONTROL
;TO CONT.

```

```

8151 042234 012601 DIVFSUB: MOV (SP)+,R1 ;GET A POINTER TO THE ARGUMENTS.
8152 042236 012700 000200 MOV #200,R0 ;SET FD MODE.
8153 042242 170100 LDFPS R0
8154 042244 010100 MOV R1,R0 ;LOAD THE AC OPERAND.
8155 042246 172410 LDD (R0),ACO
8156 042250 016100 000014 MOV 14(R1),R0 ;LOAD THE FPS
8157 042254 1701C0 LDFPS R0
8158 042256 012737 042272 001236 MOV #1$,@#STMP2

```





```

8215                                     ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
8216                                     ;THE USER TYPED CONTROL G?).
8217
8218
8219
8220
8221      ;*****
8222      ;*TEST 45      DIVD TEST
8223      ;*
8224      ;*THIS IS A TEST OF THE DIVD INSTRUCTION. NOTE THAT A SUBROUTINE IS
8225      ;*USED TO SET UP THE OPERANDS, EXECUTE THE INSTRUCTION AND CHECK THE RESULTS.
8226      ;*
8227      ;*****
8227      042462 000004      TST45: SCOPE
8228
8229      ;DIVD TEST WITH POSITIVE OPERANDS AND IN ROUND MODE.
8230      DDD1:
8231      042464 104414      LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.
8232      042466 004737 043154      JSR      PC,@DIVDSUB
8233      042472 034277 000000 000000 1$: .WORD 34277,0,0,0      ;AC
8234      042500 000000
8235      042502 040277 000000 000000 2$: .WORD 40277,0,0,0      ;FSRC
8236      042510 000000
8237      042512 034200 000000 000000 3$: .WORD 34200,0,0,0      ;RES
8238      042520 000000
8239      042522 000200      4$: 200      ;FPS BEFORE EXECUTION.
8240      042524 000200      200      ;FPS AFTER EXECUTION.
8241      042526 177777 177777 177777 5$: .WORD -1,-1,-1,-1      ;ERROR RES.
8242      042534 177777
8243      042536 104002      6$: ERROR 2
8244
8245      ;DIVD WITH AC NEGATIVE AND FSRC POSITIVE IN TRUNCATE MODE.
8246      DDD2:
8247      042540 104414      LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.
8248      042542 004737 043154      JSR      PC,@DIVDSUB
8249      042546 134277 000000 000000 1$: .WORD 134277,0,0,0      ;AC
8250      042554 000000
8251      042556 040277 000000 000000 2$: .WORD 40277,0,0,0      ;FSRC
8252      042564 000000
8253      042566 134200 000000 000000 3$: .WORD 134200,0,0,0      ;RES
8254      042574 000000
8255      042576 000207      4$: 207      ;FPS BEFORE EXECUTION.
8256      042600 000210      210      ;FPS AFTER EXECUTION.
8257      042602 177777 177777 177777 5$: .WORD -1,-1,-1,-1      ;ERROR RESULT.
8258      042610 177777
8259      042612 104002      6$: ERROR 2
8260
8261      ;DIVD TEST WITH OPERANDS BOTH NEGATIVE AND IN TRUNCATE MODE.
8262      DDD3:
8263      042614 104414      LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.
8264      042616 004767 000332      JSR      PC,DIVDSUB
8265      042622 134300 000000 000000 1$: .WORD 134300,0,0,1      ;AC
8266      042630 000001
8267      042632 140300 000000 000000 2$: .WORD 140300,0,0,0      ;FSRC
8268      042640 000000
8269      042642 034200 000000 000000 3$: .WORD 34200,0,0,0      ;RES
8270      042650 000000
    
```

```

8271 042652 000250          4$: 250          ;FPS BEFORE EXECUTION.
8272 042654 000240          240          ;FPS AFTER EXECUTION.
8273 042656 034200 000000 000000 5$: .WORD 34200,0,0,1 ;ERROR RES.
8274 042664 000001
8275 042666 104002          6$: ERROR 2          ;TRUNCATION ERROR.
8276
8277          ;DIVD WITH AC POSITIVE AND FSRC NEGATIVE IN ROUND MODE.
8278 042670          DDD4:
8279 042670 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
8280 042672 004737 043154          JSR PC, @DIVDSUB
8281 042676 034300 000000 000000 1$: .WORD 34300,0,0,1 ;AC
8282 042704 000001
8283 042706 140300 000000 000000 2$: .WORD 140300,0,0,0 ;FSRC
8284 042714 000000
8285 042716 134200 000000 000000 3$: .WORD 134200,0,0,1 ;RES
8286 042724 000001
8287 042726 000207          4$: 207          ;FPS BEFORE EXECUTION.
8288 042730 000210          210          ;FPS AFTER EXECUTION.
8289 042732 134200 000000 000000 5$: .WORD 134200,0,0,0 ;ERROR RES.
8290 042740 000000
8291 042742 104002          6$: ERROR 2          ;ROUND ERROR.
8292
8293          ;DIVD TEST.
8294 042744          DDD5:
8295 042744 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
8296 042746 004737 043154          JSR PC, @DIVDSUB
8297 042752 100400 000000 000000 1$: .WORD 100400,0,0,0 ;AC
8298 042760 000000
8299 042762 000500 000000 000000 2$: .WORD 500,0,0,0 ;FSRC
8300 042770 000000
8301 042772 140052 125252          3$: .WORD 140052,125252 ;RES
8302 042776 125252 125252          .WORD 125252,125252
8303 043002 007647          4$: 7647          ;FPS BEFORE EXECUTION.
8304 043004 007650          7650          ;FPS AFTER EXECUTION.
8305 043006 177777 177777 177777 5$: .WORD -1,-1,-1,-1 ;ERROR RES.
8306 043014 177777
8307 043016 104002          6$: ERROR 2
8308
8309
8310          ;DIVD TEST WITH AC POSITIVE AND FSRC NEGATIVE IN ROUND MODE.
8311 043020          DDD6:
8312 043020 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
8313 043022 004737 043154          JSR PC, @DIVDSUB
8314 043026 004000 000000 000000 1$: .WORD 400,0,0,0 ;AC
8315 043034 000000
8316 043036 100500 000000 000000 2$: .WORD 100500,0,0,0 ;FSRC
8317 043044 000000
8318 043046 140052 125252          3$: .WORD 140052,125252 ;RES
8319 043052 125252 125253          .WORD 125252,125253
8320 043056 007707          4$: 7707          ;FPS BEFORE EXECUTION.
8321 043060 007710          7710          ;FPS AFTER EXECUTION.
8322 043062 177777 177777 177777 5$: .WORD -1,-1,-1,-1 ;ERROR RES.
8323 043070 177777
8324 043072 104002          6$: ERROR 2
8325
8326          ;DIVD TEST.

```

```

8327 043074          DDD7:
8328 043074 104414   LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
8329 043076 004737 043154 JSR      PC,@#DIVDSUB
8330 043102 170360 170360 1$:      .WORD 170360,170360 ;AC
8331 043106 170360 170360      .WORD 170360,170360
8332 043112 170360 170360 2$:      .WORD 170360,170360 ;FSRC
8333 043114 170360 170360      .WORD 170360,170360
8334 043122 040200 000000 3$:      .WORD 40200,0,0,0 ;RES
8335 043130 000000
8336 043132 007717      4$:      7717          ;FPS BEFORE EXECUTION.
8337 043134 007700      7700          ;FPS AFTER EXECUTION.
8338 043136 177777 177777 5$:      .WORD -1,-1,-1,-1 ;ERROR RES.
8339 043144 177777
8340 043146 104002      6$:      ERROR 2
8341
8342 043150 000137 043414      JMP      @#DDDDONE ;GO TO NEXT TEST.
8343
8344
8345
8346
8347
8348
8349
8350
8351
8352
8353
8354
8355
8356
8357
8358
8359
8360
8361
8362
8363
8364
8365
8366
8367
8368
8369
8370

```

```

;THIS SUBROUTINE, DIVDSUB, IS CALLED TO SET UP, EXECUTE
;AND CHECK THE RESULT OF A DIVD INSTRUCTION. IT IS CALLED THUS:

```

```

:
:      JSR      PC,@#DIVDSUB
:      ACARG: .WORD X,X,X,X          ;AC OPERAND
:      FSRCARG: .WORD X,X,X,X       ;FSRC OPERAND
:      RES:    .WORD X,X,X,X       ;EXPECTED RESULT
:      FPSB:   .WORD X              ;FPS BEFORE EXECUTION
:      FPSA:   .WORD X              ;FPS AFTER EXECUTION
:      ERRES:  .WORD X,X,X,X       ;ERROR RESULT
:      ERR:    ERROR X              ;RESULT ERROR
:      CONT:
:      ;RETURN ADDRESS

```

```

;THE OPERANDS ARE SET UP (USING ACO FOR THE AC OPERAND). THEN
;FPSB IS LOADED INTO THE FPS. THE INSTRUCTION, DIVD IS EXECUTED.
;AFTER THE EXECUTION THE RESULT IS CHECKED AGAINST THE
;EXPECTED CORRECT RESULT, RES. IF IT IS CORRECT THEN THE FPS
;IS CHECKED WITH THE EXPECTED CORRECT FPS, FPSA. IF THE FPS WAS
;INCORRECT THEN IT IS REPORTED. IF THE RESULT WAS INCORRECT IT
;IS COMPARED WITH ERRES IN AN ATTEMPT TO ANALYSE THE ERROR. IF
;THE INCORRECT RESULT MATCHED ERRES THEN CONTROL IS PASSED TO
;THE ERROR CALL AT ERR. IF THE INCORRECT RESULT DID NOT MATCH ERRES
;THEN THE FAILURE IS REPORTED IN DIVDSUB AND CONTROL IS PASSED TO
;CONT. IF NO ERRORS ARE DETECTED THEN DIVDSUB RETURNS CONTROL
;TO CONT.

```

```

8371 043154 012601          DIVDSUB:      MOV      (SP)+,R1 ;GET A POINTER TO THE ARGUMENTS.
8372 043156 012700 000200      MOV      #200,R0 ;SET FD MODE.
8373 043162 170100          LDFPS   R0
8374
8375 043164 010100          MOV      R1,R0 ;SET UP THE ACO OPERAND.
8376 043166 172410          LDD     (R0),ACO
8377 043170 016100 000030      MOV      30(R1),R0 ;LOAD THE FPS.
8378 043174 170100          LDFPS   R0
8379
8380 043176 012737 043212 001236      MOV      #15,@#STMP2
8381 043204 010100          MOV      R1,R0 ;ESTABLISH A POINTER TO FSRC.
8382 043206 062700 000010          ADD     #10,R0

```

```

8383
8384 043212 174410      1$:  DIVD    (R0),ACO      ;EXECUTE THE TEST INSTRUCTION.
8385
8386 043214 170204      STFPS   R4              ;GET THE FPS.
8387 043216 012700 000200  MOV     #200,R0        ;SET FD MODE.
8388 043222 170100      LDFPS   R0
8389
8390 043224 012700 043404  MOV     #DIVDT,R0      ;GET THE RESULT.
8391 043230 174010      STD     ACO,(R0)
8392
8393 043232 010102      MOV     R1,R2          ;SAVE DATA IN CASE OF ERROR.
8394 043234 010237 001240  MOV     R2,@STMP3
8395 043240 062702 000010  ADD     #10,R2
8396 043244 010237 001242  MOV     R2,@STMP4
8397 043250 062702 000010  ADD     #10,R2
8398 043254 010237 001244  MOV     R2,@STMP5
8399 043260 012737 043404 001246  MOV     #DIVDT,@STMP6
8400 043266 010437 001250  MOV     R4,@STMP7
8401 043272 016137 000032, 001252  MOV     32(R1),@STMP10
8402
8403 043300 010102      MOV     R1,R2          ;CHECK THE RESULT.
8404 043302 062702 000020  ADD     #20,R2
8405 043306 012703 043404  MOV     #DIVDT,R3
8406 043312 012705 000004  MOV     #4,R5
8407 043316 022223      2$:  CMP     (R2)+,(R3)+
8408 043320 001006      BNE    10$
8409 043322 077503      SOB    R5,2$
8410
8411 043324 026104 000032  CMP     32(R1),R4      ;IS FPS CORRECT?
8412 043330 001023      BNE    15$
8413 043332 000161 000046  JMP     46(R1)        ;BRANCH IF INCORRECT.
8414
8415 043336 010102      10$:  MOV     R1,R2          ;WAS INCORRECT RESULT ANTICIPATED?
8416 043340 062702 000034  ADD     #34,R2
8417 043344 012703 043404  MOV     #DIVDT,R3
8418 043350 012705 000004  MOV     #4,R5
8419 043354 022223      11$:  CMP     (R2)+,(R3)+
8420 043356 001005      BNE    12$
8421 043360 077503      SOB    R5,11$
8422 043362 010102      MOV     R1,R2
8423 043364 062702 000044  ADD     #44,R2
8424
8425 043370 000112      JMP     (R2)
8426
8427 043372      12$:  ;REPORT RESULT INCORRECT.
8428 043372 104002      13$:  ERROR   2
8429 043374 000161 000046  14$:  JMP     46(R1)
8430
8431 043400      15$:  ;REPORT FPS INCORRECT.
8432 043400 104002      16$:  ERROR   2
8433 043402 000774      BR     14$
8434
8435 043404 000000 000000 000000  DIVDT: .WORD 0,0,0,0
8436 043412 000000
8437
8438 043414      DDDDONE:
    
```

8439 043414 104413 RSETUP ;GO INITIALIZE THE FPS AND STACK; AND  
 8440 ;SEE IF THE USER HAS EXPRESSED  
 8441 ;THE DESIRE TO CHANGE THE SOFTWARE  
 8442 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS  
 8443 ;THE USER TYPED CONTROL G?).  
 8444  
 8445  
 8446  
 8447

```

:*****
:*TEST 46      MULF TEST
:*
:*THIS IS A TEST OF THE MULF INSTRUCTION. IT MAKES USE OF A SUBROUTINE
:*TO SET UP THE OPERANDS, EXECUTE THE MULF INSTRUCTION AND CHECK THE
:*RESULTS.
:*
:*****
    
```

8454  
 8455 043416 000004

TST46: SCOPE

8456  
 8457  
 8458 043420  
 8459 043420 104414  
 8460 043422 004737 044200  
 8461 043426 000000 000000  
 8462 043432 000000 000000  
 8463 043436 000000 000000  
 8464 043442 007517  
 8465 043444 007504  
 8466 043446 177777 177777  
 8467 043452 104002

```

;MULF WITH (FSRC=AC=0)
EEE1:
    LPERR                ;SET UP THE LOOP ON ERROR ADDRESS.
    JSR      PC,@MULFSUB
1$:  .WORD  0,0          ;AC
2$:  .WORD  0,0          ;FSRC
3$:  .WORD  0,0          ;RES
4$:  7517                ;FPS BEFORE EXECUTION.
    7504                ;FPS AFTER EXECUTION.
5$:  .WORD  -1,-1
6$:  ERROR  2
    
```

8468  
 8469  
 8470 043454  
 8471 043454 104414  
 8472 043456 004737 044200  
 8473 043462 071625 034435  
 8474 043466 000000 000000  
 8475 043472 000000 000000  
 8476 043476 000013  
 8477 043500 000004  
 8478 043502 177777 177777  
 8479 043506 104002

```

;MULF WITH (FSRC=0).
EEE2:
    LPERR                ;SET UP THE LOOP ON ERROR ADDRESS.
    JSR      PC,@MULFSUB
1$:  .WORD  71625,34435 ;AC
2$:  .WORD  0,0          ;FSRC
3$:  .WORD  0,0          ;RES
4$:  13                  ;FPS BEFORE EXECUTION.
    4                    ;FPS AFTER EXECUTION.
5$:  .WORD  -1,-1
6$:  ERROR  2
    ;ERROR RES.
    
```

8480  
 8481  
 8482 043510  
 8483 043510 104414  
 8484 043512 004737 044200  
 8485 043516 000000 000000  
 8486 043522 071625 153443  
 8487 043526 000000 000000  
 8488 043532 007500  
 8489 043534 007504  
 8490 043536 177777 177777  
 8491 043542 104002

```

;MULF WITH (AC=0)
EEE3:
    LPERR                ;SET UP THE LOOP ON ERROR ADDRESS.
    JSR      PC,@MULFSUB
1$:  .WORD  0,0          ;AC
2$:  .WORD  071625,153443 ;FSRC
3$:  .WORD  0,0          ;RES
4$:  7500                ;FPS BEFORE EXECUTION.
    7504                ;FPS AFTER EXECUTION.
5$:  .WORD  -1,-1
6$:  ERROR  2
    ;ERROR RES.
    
```

8492  
 8493  
 8494 043544

;MULF WITH AC POSITIVE AND FSRC POSITIVE IN ROUND MODE.  
 EEE4:

```

8495 043544 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
8496 043546 004737 044200   JSR          PC,@MULFSUB
8497 043552 040200 000000   1$: .WORD    40200,0      ;AC
8498 043556 040177 177777   2$: .WORD    40177,-1    ;FSRC
8499 043562 040177 177777   3$: .WORD    40177,-1    ;RES
8500 043566 000017          4$:          17          ;FPS BEFORE EXECUTION.
8501 043570 000000          0          ;FPS AFTER EXECUTION.
8502 043572 140177 177777   5$: .WORD    140177,-1   ;ERROR RES.
8503 043576 104002          6$: ERROR      2          ;BAD SIGN.
8504
8505 ;MULF WITH AC POSITIVE AND FSRC POSITIVE IN TRUNCATE MODE.
8506 043600   FEE5:
8507 043600 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
8508 043602 004767 000372   JSR          PC,MULFSUB
8509 043606 040177 177777   1$: .WORD    40177,-1    ;AC
8510 043612 040200 000000   2$: .WORD    40200,0    ;FSRC
8511 043616 040177 177777   3$: .WORD    40177,-1    ;RES
8512 043622 000040          4$:          40          ;FPS BEFORE EXECUTION.
8513 043624 000040          40          ;FPS AFTER EXECUTION.
8514 043626 037777 177777   5$: .WORD    37777,-1   ;ERROR RES.
8515 043632 104002          6$: ERROR      2          ;ST 252 TO 044 INTO 444 (BUT Y62)
8516 ;MUL. NORMALIZATION FAILURE.
8517
8518 ;MULF WITH BOTH OPERANDS POSITIVE NORMALIZE TEST.
8519 043634   EEE6:
8520 043634 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
8521 043636 004737 044200   JSR          PC,@MULFSUB
8522 043642 040100 000000   1$: .WORD    40100,0    ;AC
8523 043646 040100 000000   2$: .WORD    40100,0    ;FSRC
8524 043652 040020 000000   3$: .WORD    40020,0    ;RES
8525 043656 000012          4$:          12          ;FPS BEFORE EXECUTION.
8526 043660 000000          0          ;FPS AFTER EXECUTION.
8527 043662 042040 000000   5$: .WORD    42040,0    ;ERROR RES.
8528 043666 104002          6$: ERROR      2          ;ST 252 TO 444 INTO 042 (BUT Y62)
8529 ;MUL. NORMALIZATION FAILURE.
8530
8531 ;MULF WITH BOTH OPERANDS POSITIVE IN ROUND MODE.
8532 043670   EEE7:
8533 043670 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
8534 043672 004737 044200   JSR          PC,@MULFSUB
8535 043676 017500 000000   1$: .WORD    17500,0    ;AC
8536 043702 023652 125252   2$: .WORD    23652,125252 ;FSRC
8537 043706 003177 177777   3$: .WORD    3177,-1    ;RES
8538 043712 007417          4$:          7417        ;FPS BEFORE EXECUTION.
8539 043714 007400          7400        ;FPS AFTER EXECUTION.
8540 043716 177777 177777   5$: .WORD    -1,-1      ;ERROR RES.
8541 043722 104002          6$: ERROR      2          ;BAD SIGN.
8542
8543 ;MULF WITH AC POSITIVE AND FSRC NEGATIVE IN ROUND MODE.
8544 043724   EEE8:
8545 043724 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
8546 043726 004737 044200   JSR          PC,@MULFSUB
8547 043732 040342 000000   1$: .WORD    40342,0    ;AC
8548 043736 176542 000000   2$: .WORD    176542,0   ;FSRC
8549 043742 176707 102000   3$: .WORD    176707,102000 ;RES
8550 043746 000007          4$:          7          ;FPS BEFORE EXECUTION.
    
```

```

8551 043750 000010          10          :FPS AFTER EXECUTION.
8552 043752 076507 102000  7$: .WORD 76507,102000 :ERROR RES.
8553 043756 104002          6$: ERROR 2           :BAD SIGN.
8554
8555 ;MULF WITH AC NEGATIVE AND FSRC POSITIVE IN ROUND MODE.
8556 043760          EEE9:
8557 043760 104414          LPERR          :SET UP THE LOOP ON ERROR ADDRESS.
8558 043762 004737 044200  JSR PC,@MULFSUB
8559 043766 140200 000000  1$: .WORD 140200,0      :AC
8560 043772 007417 007417  2$: .WORD 7417,7417    :FSRC
8561 043776 107417 007417  3$: .WORD 107417,7417  :RES
8562 044002 000000          4$: 0              :FPS BEFORE EXECUTION.
8563 044004 000010          :FPS AFTER EXECUTION.
8564 044006 007417 007417  5$: .WORD 7417,7417    :ERROR RES.
8565 044012 104002          6$: ERROR 2           :BAD SIGN.
8566
8567 ;MULF WITH BOTH OPERANDS NEGATIVE IN ROUND MODE.
8568 044014          EEE10:
8569 044014 104414          LPERR          :SET UP THE LOOP ON ERROR ADDRESS.
8570 044016 004737 044200  JSR PC,@MULFSUB
8571 044022 144600 000000  1$: .WORD 144600,0      :AC
8572 044026 154000 000000  2$: .WORD 154000,0     :FSRC
8573 044032 060400 000000  3$: .WORD 60400,0      :RES
8574 044036 000017          4$: 17              :FPS BEFORE EXECUTION.
8575 044040 000000          :FPS AFTER EXECUTION.
8576 044042 160400 000000  5$: .WORD 160400,0     :ERROR RES.
8577 044046 104002          6$: ERROR 2           :BAD SIGN.
8578
8579 ;MULF BOTH OPERANDS NEGATIVE IN ROUND MODE.
8580 044050          EEE11:
8581 044050 104414          LPERR          :SET UP THE LOOP ON ERROR ADDRESS.
8582 044052 004737 044200  JSR PC,@MULFSUB
8583 044056 140300 000000  1$: .WORD 140300,0      :AC
8584 044062 160000 000001  2$: .WORD 160000,1     :FSRC
8585 044066 060100 000002  3$: .WORD 60100,2      :RES
8586 044072 000010          4$: 10              :FPS BEFORE EXECUTION.
8587 044074 000000          :FPS AFTER EXECUTION.
8588 044076 060100 000001  5$: .WORD 60100,1      :ERROR RES.
8589 044102 104002          6$: ERROR 2           :ROUND FAILURE.
8590
8591 ;MULF WITH AC POSITIVE AND FSRC NEGATIVE IN TRUNCATE MODE.
8592 044104          EEE12:
8593 044104 104414          LPERR          :SET UP THE LOOP ON ERROR ADDRESS.
8594 044106 004737 044200  JSR PC,@MULFSUB
8595 044112 060000 000001  1$: .WORD 60000,1      :AC
8596 044116 140300 000000  2$: .WORD 140300,0     :FSRC
8597 044122 160100 000001  3$: .WORD 160100,1     :RES
8598 044126 007547          4$: 7547            :FPS BEFORE EXECUTION.
8599 044130 007550          :FPS AFTER EXECUTION.
8600 044132 160100 000001  5$: .WORD 160100,1     :ERROR RES.
8601 044136 104002          6$: ERROR 2           :TRUNCATION ERROR.
8602
8603 ;MULF WITH AC POSITIVE AND FSRC POSITIVE IN ROUND MODE.
8604 044140          EEE13:
8605 044140 104414          LPERR          :SET UP THE LOOP ON ERROR ADDRESS.
8606 044142 004737 044200  JSR PC,@MULFSUB
    
```

8607 044146 040277 000000  
 8608 044152 060000 000001  
 8609 044156 060077 000001  
 8610 044162 000014  
 8611 044164 000000  
 8612 044166 060077 000002  
 8613 044172 104002  
 8614  
 8615 044174 000167 000224  
 8616  
 8617  
 8618  
 8619  
 8620  
 8621  
 8622  
 8623  
 8624  
 8625  
 8626  
 8627  
 8628  
 8629  
 8630  
 8631  
 8632  
 8633  
 8634  
 8635  
 8636  
 8637  
 8638  
 8639  
 8640  
 8641  
 8642  
 8643 044200 012601  
 8644 044202 012700 000200  
 8645 044206 170100  
 8646 044210 010100  
 8647 044212 172410  
 8648 044214 016100 000014  
 8649 044220 170100  
 8650 044222 012737 044236 001236  
 8651 044230 010100  
 8652 044232 062700 000004  
 8653  
 8654 044236 171010  
 8655  
 8656 044240 170204  
 8657 044242 012700 000200  
 8658 044246 170100  
 8659  
 8660 044250 012700 044414  
 8661 044254 174010  
 8662

1\$: .WORD 40277,0 ;AC  
 2\$: .WORD 60000,1 ;FSRC  
 3\$: .WORD 60077,1 ;RES  
 4\$: 14 ;FPS BEFORE EXECUTION.  
 0 ;FPS AFTER EXECUTION.  
 5\$: .WORD 60077,2 ;ERROR RES.  
 6\$: ERROR 2 ;ROUND FAILURE. CONSTANT BAD.  
 JMP EEEDONE ;GO TO THE NEXT TEST.

; THIS SUBROUTINE, MULFSUB, IS CALLED TO SET UP, EXECUTE  
 ; AND CHECK THE RESULT OF A MULF INSTRUCTION. IT IS CALLED THUS:

```

:
:          JSR      PC,@#MULFSUB
:          ACARG:  .WORD  X,X          ;AC OPERAND
:          FSRCARG: .WORD  X,X          ;FSRC OPERAND
:          RES:    .WORD  X,X          ;EXPECTED RESULT
:          FPSB:   .WORD  X           ;FPS BEFORE EXECUTION
:          FPSA:   .WORD  X           ;FPS AFTER EXECUTION
:          ERRES:  .WORD  X,X         ;ERROR RESULT
:          ERR:    ERROR  X           ;RESULT ERROR
:          CONT:   ;RETURN ADDRESS
  
```

; THE OPERANDS ARE SET UP (USING ACO FOR THE AC OPERAND). THEN  
 ; FPSB IS LOADED INTO THE FPS. THE INSTRUCTION, MULF IS EXECUTED.  
 ; AFTER THE EXECUTION THE RESULT IS CHECKED AGAINST THE  
 ; EXPECTED CORRECT RESULT, RES. IF IT IS CORRECT THEN THE FPS  
 ; IS CHECKED WITH THE EXPECTED CORRECT FPS, FPSA. IF THE FPS WAS  
 ; INCORRECT THEN IT IS REPORTED. IF THE RESULT WAS INCORRECT IT  
 ; IS COMPARED WITH ERRES IN AN ATTEMPT TO ANALYSE THE ERROR. IF  
 ; THE INCORRECT RESULT MATCHED ERRES THEN CONTROL IS PASSED TO  
 ; THE ERROR CALL AT ERR. IF THE INCORRECT RESULT DID NOT MATCH ERRES  
 ; THEN THE FAILURE IS REPORTED IN MULFSUB AND CONTROL IS PASSED TO  
 ; CONT. IF NO ERRORS ARE DETECTED THEN MULFSUB RETURNS CONTROL  
 ; TO CONT.

```

MULFSUB:  MOV      (SP)+,R1          ;GET A POINTER TO THE ARGUMENTS.
          MOV      #200,R0          ;SET FD MODE.
          LDFPS   R0
          MOV      R1,R0           ;LOAD THE AC OPERAND.
          LDD     (R0),ACO
          MOV      14(R1),R0        ;LOAD THE FPS
          LDFPS   R0
          MOV      #1$,@#STMP2
          MOV      R1,R0
          ADD     #4,R0            ;ESTABLISH A POINTER TO FSRC.
1$:      MULF    (R0),ACO          ;TEST INSTRUCTION.
          STFPS   R4               ;GET THE FPS.
          MOV      #200,R0          ;SET FD MODE
          LDFPS   R0
          MOV      #MULFT,R0        ;GET THE RESULT OF THE MULF.
          STD     ACO,(R0)
  
```



```

8663 044256 010102          MOV      R1,R2          ;SAVE THE DATA IN CASE OF ERROR.
8664 044260 010237 001240    MOV      R2,@#STMP3
8665 044264 062702 000004    ADD      #4,R2
8666 044270 010237 001242    MOV      R2,@#STMP4
8667 044274 062702 000004    ADD      #4,R2
8668 044300 010237 001244    MOV      R2,@#STMP5
8669 044304 012737 044414 001246    MOV      #MULFT,@#STMP6
8670 044312 010437 001250    MOV      R4,@#STMP7
8671 044316 016137 000016 001252    MOV      16(R1),@#STMP10
8672
8673 044324 021061 000010          CMP      (R0),10(R1)      ;IS THE RESULT CORRECT?
8674 044330 001011          BNE      10$              ;IF INCORRECT BRANCH.
8675 044332 026061 000002 000012    CMP      2(R0),12(R1)
8676 044340 001005          BNE      10$
8677
8678 044342 026104 000016          CMP      16(R1),R4        ;IS FPS CORRECT?
8679 044346 001020          BNE      15$              ;IF INCORRECT BRANCH.
8680 044350 000161 000026          JMP      26(R1)           ;IF NO ERRORS OCCURRED RETURN.
8681
8682 044354 021061 000020          10$:    CMP      (R0),20(R1)      ;DOES THE INCORRECT RESULT
8683 044360 001010          BNE      11$              ;MATCH THE ANTICIPATED INCORRECT RESULT.
8684 044362 026061 000002 000022    CMP      2(R0),22(R1)
8685 044370 001004          BNE      11$              ;BRANCH IF NO.
8686
8687 044372 010102          MOV      R1,R2          ;IT MATCHED SO RETURN TO THE ERROR
8688                                ;REPORT AT THE CALLING ROUTINE.
8689 044374 062702 000024          ADD      #24,R2
8690 044400 000112          JMP      (R2)
8691
8692 044402          11$:
8693 044402 104002          12$:    ERROR  2              ;REPORT RESULT INCORRECT.
8694 044404 000161 000026          13$:    JMP      26(R1)
8695
8696 044410          15$:
8697 044410 104002          16$:    ERROR  2              ;REPORT FPS INCORRECT.
8698 044412 000774          BR      13$
8699
8700 044414 000000 000000 000000    MULFT:  .WORD  0,0,0,0
8701 044422 000000
8702
8703 044424          EEEDONE:
8704 044424 104413          RSETUP              ;GO INITIALIZE THE FPS AND STACK; AND
8705                                ;SEE IF THE USER HAS EXPRESSED
8706                                ;THE DESIRE TO CHANGE THE SOFTWARE
8707                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
8708                                ;THE USER TYPED CONTROL G?).
8709
8710
8711
8712          ;*****
8713          ;*TEST 47      MULD TEST
8714          ;*
8715          ;*THIS IS A TEST OF THE MULD INSTRUCTION. NOTE THAT A SUBROUTINE IS
8716          ;*USED TO SET UP THE OPERANDS, EXECUTE THE MULD INSTRUCTION AND
8717          ;*CHECK THE RESULTS.
8718          ;*

```

```

8719
8720 044426 000004
8721
8722
8723 044430
8724 044430 104414
8725 044432 004737 044714
8726 044436 040200 000000 000000 1$:
8727 044444 000000
8728 044446 023777 177777 177777 2$:
8729 044454 177777
8730 044456 023777 177777 177777 3$:
8731 044464 177777
8732 044466 000217 4$:
8733 044470 000200
8734 044472 023777 177777 000000 5$:
8735 044500 000000
8736 044502 104002 6$:
8737
8738
8739
8740 044504
8741 044504 104414
8742 044506 004767 000202
8743 044512 065400 000000 000000 1$:
8744 044520 000001
8745 044522 037577 177777 177777 2$:
8746 044530 177776
8747 044532 064777 177777 177777 3$:
8748 044540 177777
8749 044542 000247 4$:
8750 044544 000240
8751 044546 065000 000000 000000 5$:
8752 044554 000000
8753 044556 104002 6$:
8754
8755
8756 044560
8757 044560 104414
8758 044562 004737 044714
8759 044566 137577 177777 177777 1$:
8760 044574 177776
8761 044576 165400 000000 000000 2$:
8762 044604 000001
8763 044606 065000 000000 000000 3$:
8764 044614 000000
8765 044616 007717 4$:
8766 044620 007700
8767 044622 064777 177777 177777 5$:
8768 044630 177777
8769 044632 104002 6$:
8770
8771
8772 044634
8773 044634 104414
8774 044636 004737 044714

```

:\*\*\*\*\*  
TST47: SCOPE  
;MULD TEST WITH AC POSITIVE AND FSRC POSITIVE.  
FFF1:  
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.  
JSR PC,@MULDSUB  
.WORD 40200,0,0,0 ;AC  
.WORD 23777,-1,-1,-1 ;FSRC  
.WORD 23777,-1,-1,-1 ;RES  
217 ;FPS BEFORE EXECUTION.  
200 ;FPS AFTER EXECUTION.  
.WORD 23777,-1,0,0 ;ERROR RES.  
ERROR 2 ;BAD CONSTANT USED IN ALGORITHM  
;USED 24 INSTEAD OF 56.  
;MULD TEST WITH BOTH OPERANDS POSITIVE TRUNCATION TEST.  
FFF2:  
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.  
JSR PC,MULDSUB  
.WORD 65400,0,0,1 ;AC  
.WORD 37577,-1,-1,-2 ;FSRC  
.WORD 64777,-1,-1,-1 ;RES  
247 ;FPS BEFORE EXECUTION.  
240 ;FPS AFTER EXECUTION.  
.WORD 65000,0,0,0 ;ERROR RES.  
ERROR 2 ;TRUNCATION ERROR.  
;MULD TEST WITH BOTH OPERANDS NEGATIVE IN ROUND MODE.  
FFF3:  
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.  
JSR PC,@MULDSUB  
.WORD 137577,-1,-1,-2 ;AC  
.WORD 165400,0,0,1 ;FSRC  
.WORD 65000,0,0,0 ;RES  
7717 ;FPS BEFORE EXECUTION.  
7700 ;FPS AFTER EXECUTION.  
.WORD 64777,-1,-1,-1 ;ERROR RES.  
ERROR 2 ;ROUND ERROR.  
;MULD TEST WITH AC POSITIVE AND FSRC NEGATIVE IN ROUND MODE.  
FFF4:  
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.  
JSR PC,@MULDSUB

```

8775 044642 017500 000000 000000 1$: .WORD 17500,0,0,0 ;AC
8776 044650 000000
8777 044652 123652 125252 2$: .WORD 123652,125252 ;FSRC
8778 044656 125252 125252 .WORD 125252,125252
8779 044662 103177 177777 3$: .WORD 103177,-1,-1,-1 ;RES
8780 044670 177777
8781 044672 000200 4$: 200 ;FPS BEFORE EXECUTION.
8782 044674 000210 210 ;FPS AFTER EXECUTION.
8783 044676 103200 000000 000000 5$: .WORD 103200,0,0,0 ;ERROR RES.
8784 044704 000000
8785 044706 104002 6$: ERROR 2 ;ROUND ERROR (BAD CONSTANT).
8786
8787 044710 000167 000240 JMP FFFDONE
8788

```

```

;THIS SUBROUTINE, MULDSUB, IS CALLED TO SET UP, EXECUTE
;AND CHECK THE RESULT OF A MULD INSTRUCTION. IT IS CALLED THUS:

```

```

:
:
:      JSR      PC,@MULDSUB
:      ACARG:  .WORD  X,X,X,X      ;AC OPERAND
:      FSRCARG: .WORD  X,X,X,X      ;FSRC OPERAND
:      RES:    .WORD  X,X,X,X      ;EXPECTED RESULT
:      FPSB:   .WORD  X              ;FPS BEFORE EXECUTION
:      FPSA:   .WORD  X              ;FPS AFTER EXECUTION
:      ERRES:  .WORD  X,X,X,X      ;ERROR RESULT
:      ERR:    ERROR X              ;RESULT ERROR
:      CONT:   ;RETURN ADDRESS

```

```

;THE OPERANDS ARE SET UP (USING ACO FOR THE AC OPERAND). THEN
;FPSB IS LOADED INTO THE FPS. THE INSTRUCTION, MULD IS EXECUTED.
;AFTER THE EXECUTION THE RESULT IS CHECKED AGAINST THE
;EXPECTED CORRECT RESULT, RES. IF IT IS CORRECT THEN THE FPS
;IS CHECKED WITH THE EXPECTED CORRECT FPS, FPSA. IF THE FPS WAS
;INCORRECT THEN IT IS REPORTED. IF THE RESULT WAS INCORRECT IT
;IS COMPARED WITH ERRES IN AN ATTEMPT TO ANALYSE THE ERROR. IF
;THE INCORRECT RESULT MATCHED ERRES THEN CONTROL IS PASSED TO
;THE ERROR CALL AT ERR. IF THE INCORRECT RESULT DID NOT MATCH ERRES
;THEN THE FAILURE IS REPORTED IN MULDSUB AND CONTROL IS PASSED TO
;CONT. IF NO ERRORS ARE DETECTED THEN MULDSUB RETURNS CONTROL
;TO CONT.

```

```

8815 044714 012601 MULDSUB: MOV (SP)+,R1 ;GET A POINTER TO THE ARGUMENTS.
8816 044716 012700 000200 MOV #200,R0 ;SET FD MODE.
8817 044722 170100 LDFPS R0
8818
8819 044724 010100 MOV R1,R0 ;SET UP THE ACO OPERAND.
8820 044726 172410 LDD (R0),ACO
8821 044730 016100 000030 MOV 30(R1),R0 ;LOAD THE FPS.
8822 044734 170100 LDFPS R0
8823
8824 044736 012737 044752 001236 MOV #1$,@#STMP2
8825 044744 010100 MOV R1,R0 ;ESTABLISH A POINTER TO FSRC.
8826 044746 062700 000010 ADD #10,R0
8827
8828 044752 171010 1$: MULD (R0),ACO ;EXECUTE THE TEST INSTRUCTION.
8829
8830 044754 170204 STFPS R4 ;GET THE FPS.

```

```

8831 044756 012700 000200      MOV    #200,R0      ;SET FD MODE.
8832 044762 170100      LDFPS  R0
8833
8834 044764 012700 045144      MOV    #MULDT,R0   ;GET THE RESULT.
8835 044770 174010      STD    ACO,(R0)
8836
8837 044772 010102      MOV    R1,R2      ;SAVE DATA IN CASE OF ERROR.
8838 044774 010237 001240      MOV    R2,@#STMP3
8839 045000 062702 000010      ADD    #10,R2
8840 045004 010237 001242      MOV    R2,@#STMP4
8841 045010 062702 000010      ADD    #10,R2
8842 045014 010237 001244      MOV    R2,@#STMP5
8843 045020 012737 045144 001246      MOV    #MULDT,@#STMP6
8844 045026 010437 001250      MOV    R4,@#STMP7
8845 045032 016137 000032 001252      MOV    32(R1),@#STMP10
8846
8847 045040 010102      MOV    R1,R2      ;CHECK THE RESULT.
8848 045042 062702 000020      ADD    #20,R2
8849 045046 012703 045144      MOV    #MULDT,R3
8850 045052 012705 000004      MOV    #4,R5
8851 045056 022223      2$:  CMP    (R2)+,(R3)+
8852 045060 001006      BNE    10$
8853 045062 077503      SOB    R5,2$
8854
8855 045064 026104 000032      CMP    32(R1),R4   ;IS FPS CORRECT?
8856 045070 001023      BNE    15$         ;BRANCH IF INCORRECT.
8857 045072 000161 000046      JMP    46(R1)     ;RETURN.
8858
8859 045076 010102      10$: MOV    R1,R2     ;WAS INCORRECT RESULT ANTICIPATED?
8860 045100 062702 000034      ADD    #34,R2
8861 045104 012703 045144      MOV    #MULDT,R3
8862 045110 012705 000004      MOV    #4,R5
8863 045114 022223      11$: CMP    (R2)+,(R3)+
8864 045116 001005      BNE    12$
8865 045120 077503      SOB    R5,11$
8866 045122 010102      MOV    R1,R2
8867 045124 062702 000044      ADD    #44,R2
8868
8869 045130 000112      JMP    (R2)
8870
8871 045132      12$:
8872 045132 104002      13$: ERROR 2
8873 045134 000161 000046      14$: JMP    46(R1)
8874
8875 045140      15$:
8876 045140 104002      16$: ERROR 2
8877 045142 000774      BR    14$
8878
8879 045144 000000 000000 000000 MULDT: .WORD 0,0,0,0
8880 045152 000000
8881
8882 045154      FFFDONE:
8883 045154 104413      RSETUP
8884
8885
8886
;GO INITIALIZE THE FPS AND STACK; AND
;SEE IF THE USER HAS EXPRESSED
;THE DESIRE TO CHANGE THE SOFTWARE
;VIRTUAL CONSOLE SWITCH REGISTER (HAS

```

;THE USER TYPED CONTROL G?).

```

8887
8888
8889
8890
8891
8892
8893
8894
8895
8896
8897
8898
8899
8900 045156 000004
8901
8902
8903 045160
8904 045160 104414
8905 045162 004737 045404
8906 045166 020200 000000
8907 045172 020000 000000
8908 045176 000000 000000
8909 045202 177777 177777
8910 045206 000000
8911 045210 000004
8912 045212 000012
8913 045214 177777
8914 045216 104002
8915 045220 000401
8916 045222 104002
8917 045224
8918
8919
8920 045224
8921 045224 104414
8922 045226 004737 045404
8923 045232 010200 000000
8924 045236 010000 000000
8925 045242 000000 000000
8926 045246 010000 000000
8927 045252 005013
8928 045254 005004
8929 045256 000012
8930 045260 177777
8931 045262 104002
8932
8933 045264 000401
8934 045266 104002
8935 045270
8936
8937
8938 045270
8939 045270 104414
8940 045272 004737 045404
8941 045276 060200 000000
8942 045302 060000 000000
    
```

```

:*****
:*TEST 50 UNDER\OVER FLOW, USING MULF WITH TRAPS DISABLED, TEST
:*
:*THIS IS A TEST OF THE OVERFLOW AND UNDERFLOW CONDITIONS USING
:*THE MULF INSTRUCTION WITH TRAPS DISABLED. NOTE THAT A SUBROUTINE
:*IS USED TO SET UP THE OPERANDS, EXECUTE THE MULF INSTRUCTION AND
:*CHECK THE RESULTS.
:*
:*****
    
```

TST50: SCOPE

;UNDERFLOW, WITH EXPONENT OF RESULT = -129

IIII1:

```

LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,@#OVUNFNT
1$: .WORD 20200,0 ;AC
2$: .WORD 20000,0 ;FSRC
3$: .WORD 0,0 ;RES
4$: .WORD -1,-1 ;ERROR RES.
5$: 0 ;FPS BEFORE EXECUTION.
4 ;FPS AFTER EXECUTION.
6$: 12 ;FEC
-1 ;FLAG
7$: ERROR 2 ;ST 331 TO 155 INTO 115 (BUT FIU)
BR 8$
ERROR 2
8$:
    
```

;UNDERFLOW, WITH EXPONENT OF RESULT = -193

IIII2:

```

LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,@#OVUNFNT
1$: .WORD 10200,0 ;AC
2$: .WORD 10000,0 ;FSRC
3$: .WORD 0,0 ;RES
4$: .WORD 10000,0 ;ERROR RES.
5$: 5013 ;FPS BEFORE EXECUTION.
5004 ;FPS AFTER EXECUTION.
6$: 12 ;FEC
-1 ;FLAG
7$: ERROR 2 ;SETTING FIUV OR FIV CAUSES TRAP
BR 8$ ;WITH FIU CLEAR.
ERROR 2
8$:
    
```

;OVERFLOW, EXPONENT OF RESULT = 128

IIII3:

```

LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,@#OVUNFNT
1$: .WORD 60200,0 ;AC
2$: .WORD 60000,0 ;FSRC
    
```

H 13

```

8943 045306 000000 000000 3$: .WORD 0,0 ;RES
8944 045312 060000 000000 4$: .WORD 60000,0 ;ERROR RES.
8945 045316 000000 5$: 0 ;FPS BEFORE EXECUTION.
8946 045320 000006 6 ;FPS AFTER EXECUTION.
8947 045322 000010 6$: 10 ;FEC
8948 045324 000000 0 ;FLAG
8949 045326 104002 7$: ERROR 2 ;ST 333 TO 136 INTO 116 (BUT FIV).
8950 045330 000401 BR 8$
8951 045332 104002 ERROR 2
8952 045334 8$:
    
```

;OVERFLOW, EXPONENT OF RESULT = 130  
 III4:

```

8955 045334 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
8956 045334 104414 JSR PC,OVUNFNT
8957 045336 004737 045404 1$: .WORD 60200,0 ;AC
8958 045342 060200 000000 2$: .WORD 60200,0 ;FSRC
8959 045346 060200 000000 3$: .WORD 0,0 ;RES
8960 045352 000000 000000 4$: .WORD -1,-1 ;ERROR RES.
8961 045356 177777 177777 5$: 6011 ;FPS BEFORE EXECUTION.
8962 045362 006011 6006 ;FPS AFTER EXECUTION.
8963 045364 006006 6$: 10 ;FEC
8964 045366 000010 0 ;FLAG
8965 045370 000000 7$: ERROR 2 ;SETTING FIUV OR FIU WITH
8966 045372 104002 BR 8$ ;FIV CLEAR CAUSES TRAP.
8967 045374 000401 ERROR 2
8968 045376 104002 8$: JMP IIIDONE ;GO TO NEXT TEST.
8970 045400 000167 000410
    
```

;THIS SUBROUTINE, OVUNFNT, IS USED TO SET UP THE OPERANDS, EXECUTE  
 ;THE MULF INSTRUCTION AND CHECK THE RESULTS OF AN INSTRUCTION WITH  
 ;OPERANDS WHICH SHOULD RESULT IN EITHER OVERFLOW OR UNDERFLOW. A CALL  
 ;TO IT IS MADE THUS:

```

:
: ACARG: .WORD X,X ;AC OPERAND
: FSRCARG: .WORD X,X ;FSRC OPERAND
: RES: .WORD X,X ;EXPECTED RESULT
: ERRES: .WORD X,X ;ERROR RESULT
: FPSB: .WORD X ;FPS BEFORE EXECUTION
: FPSA: .WORD X ;FPS AFTER EXECUTION
: FEC: .WORD X ;EXPECTED FEC
: FLAG: .WORD X ;0/-1,OVER/UNDER FLOW FLAG
: ERR1: ERROR X ;TRAP ERROR.
: BR CONT
: ERR2: ERROR X ;DATA, RESULT ERROR
: CONT: ;RETURN ADDRESS
    
```

;THE OPERANDS ARE SET UP (USING ACO AS THE ACCUMULATOR). THEN  
 ;THE MULF INSTRUCTION IS EXECUTED. IF NO TRAP OCCURS THEN THE  
 ;RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS  
 ;COMPARED WITH FPSA IF THIS TOO IS CORRECT OVUNFNT RETURNS CONTROL  
 ;TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD OVUNFNT  
 ;REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE  
 ;MULF IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE

8971  
8972  
8973  
8974  
8975  
8976  
8977  
8978  
8979  
8980  
8981  
8982  
8983  
8984  
8985  
8986  
8987  
8988  
8989  
8990  
8991  
8992  
8993  
8994  
8995  
8996  
8997  
8998

```

8999 ;ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
9000 ;THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN OVUNFNT
9001 ;WILL TRANSFER CONTROL TO THE ERROR CALL AT FRR2. OTHERWISE THE
9002 ;RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND OVUNFNT WILL
9003 ;REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.
9004 ;IF A TRAP OCCURS (IT SHOULD NOT) THEN OVUNFNT WILL READ THE FEC.
9005 ;SHOULD THE FEC MATCH THE ANTICIPATED FEC OVUNFNT WILL
9006 ;STORE ALL DATA AND TRANSFER CONTROL TO THE ERROR CALL AT ERR1. IF THE
9007 ;FEC IS NOT THE SAME AS THE ANTICIPATED FEC OVUNFNT WILL REPORT
9008 ;THE ERROR AND RETURN TO CONT. NOTE THAT OVUNFNT USES THE FLAG
9009 ;TO TELL WHETHER OR NOT THESE PARTICULAR OPERANDS WILL RESULT IN
9010 ;UNDERFLOW (FLAG=-1) OR OVERFLOW (FLAG=0).
9011
9012 045404 012601 000200 OVUNFNT: MOV (SP)+,R1 ;GET A POINTER TO THE ARGUMENTS.
9013 045406 012700 MOV #200,R0 ;SET FD MODE.
9014 045412 170100 LDFPS R0
9015
9016 045414 010100 MOV R1,R0 ;LOAD ACO, OPERAND.
9017 045416 172410 LDD (R0),ACO
9018
9019 045420 010102 MOV R1,R2 ;SAVE THE DATA PATTERNS IN CASE OF
9020 045422 010237 001240 MOV R2,@#STMP3 ;ERROR.
9021 045426 062702 000004 ADD #4,R2
9022 045432 010237 001242 MOV R2,@#STMP4
9023 045436 062702 000004 ADD #4,R2
9024 045442 010237 001244 MOV R2,@#STMP5
9025 045446 016137 000022 001252 MOV 22(R1),@#STMP10
9026 045454 012737 046004 001246 MOV #OVFNIT,@#STMP6
9027
9028 045462 016100 000020 MOV 20(R1),R0 ;LOAD THE FPS.
9029 045466 170100 LDFPS R0
9030 045470 012737 045512 001236 MOV #1$,@#STMP2
9031 045476 012737 045676 000244 MOV #25$,@#FPVECT ;SET UP THE FP TRAP VECTOR IN CASE
9032 ;OF ERROR.
9033 045504 010100 MOV R1,R0 ;COMPUTE THE ADDRESS OF FSRC.
9034 045506 062700 000004 ADD #4,R0
9035
9036 045512 171010 1$: MULF (R0),ACO ;TEST INSTRUCTION.
9037
9038 045514 170204 2$: STFPS R4 ;GET FPS.
9039 045516 170305 STST R5 ;GET FEC.
9040 045520 012700 000200 MOV #200,R0 ;SET FD MODE.
9041 045524 170100 LDFPS R0
9042 045526 012700 046004 MOV #OVFNIT,R0 ;GET THE RESULT.
9043 045532 174010 STD ACO,(R0)
9044 045534 010437 001250 MOV R4,@#STMP7
9045 045540 010537 001254 MOV R5,@#STMP11
9046
9047 045544 012700 046004 MOV #OVFNIT,R0 ;CHECK THE RESULT.
9048 045550 010102 MOV R1,R2
9049 045552 062702 000010 ADD #10,R2
9050 045556 012703 000002 MOV #2,R3
9051 045562 022022 3$: CMP (R0)+,(R2)+
9052 045564 001015 BNE 15$ ;BRANCH IF INCORRECT.
9053 045566 077303 SOB R3,3$
9054
    
```

J 13

CJKDCA-A KEF11-A FP DIAG PART 1 MACY11 30A(1052) 11-JUN-79 11:50 PAGE 165  
 CJKDCA.P11 08-JUN-79 08:40 T50 UNDERFLOW, USING MULF WITH TRAPS DISABLED, TEST SEQ 0165

```

9055 045570 026104 000022      CMP      22(R1),R4      ;WAS FPS CORRECT?
9056 045574 001002      BNE      10$           ;BRANCH IF FPS IS INCORRECT.
9057
9058 045576 000161 000036      4$:     JMP      36(R1)      ;RETURN, TEST COMPLETED.
9059
9060      ;REPORT INCORRECT FPS.
9061 045602 005761 000026      10$:    TST      26(R1)      ;WAS THE RESULT OVER OR UNDER FLOW?
9062 045606 001002      BNE      12$           ;BRANCH IF UNDERFLOW.
9063
9064      ;REPORT FPS BAD AFTER OVERFLOW.
9065 045610 104002      11$:    ERROR    2
9066 045612 000771      BR      4$
9067
9068 045614      12$:
9069 045614 104002      13$:    ERROR    2      ;REPORT FPS BAD AFTER UNDERFLOW.
9070 045616 000767      BR      4$
9071
9072      ;RESULT INCORRECT.
9073 045620 012700 046004      15$:    MOV      #OVFNTT,R0      ;SEE IF FAILURE IS ANTICIPATED
9074 045624 010102      MOV      R1,R2          ;FAILURE.
9075 045626 062702 000014      ADD      #14,R2
9076 045632 012703 000002      MOV      #2,R3
9077 045636 022022      16$:    CMP      (R0)+,(R2)+
9078 045640 001007      BNE      17$           ;BRANCH IF NOT ANTICIPATED.
9079 045642 077303      SOB      R3,16$
9080
9081 045644 010102      MOV      R1,R2          ;ERROR WAS ANTICIPATED SO RETURN
9082 045646 062702 000034      ADD      #34,R2        ;TO THE ERROR REPORT IN THE CALLING
9083 045652 010237 001236      MOV      R2,@#TMP2     ;ROUTINE.
9084 045656 000112      JMP      (R2)
9085
9086 045660 005761 000026      17$:    TST      26(R1)      ;RESULT WAS NOT ANTICIPATED
9087      ;SO ERROR MUST BE REPORTED HERE.
9088      ;FIRST SEE IF ARGUMENTS SHOULD
9089      ;HAVE RESULTED IN OVERFLOW OR UNDER
9090      ;FLOW BY LOOKING AT THE FLAG.
9091 045664 001002      BNE      19$           ;BRANCH IF UNDERFLOW EXPECTED.
9092
9093      ;REPORT RESULT INCORRECT, EXPECTING
9094 045666 104002      18$:    ERROR    2      ;OVERFLOW.
9095 045670 000742      BR      4$
9096
9097 045672      19$:
9098 045672 104002      20$:    ERROR    2      ;REPORT RESULT INCORRECT, EXPECTING
9099 045674 000740      BR      4$           ;UNDERFLOW.
9100
9101      ;IF AN FP TRAP OCCURS COME HERE.
9102 045676 011602      25$:    MOV      (SP),R2      ;GET ADDRESS OF TRAP.
9103 045700 022702 045514      CMP      #2$,R2        ;WAS THE TRAP DURING THE MULF INSTRUCTION?
9104 045704 001402      BEQ      26$           ;BRANCH IF YES.
9105 045706 000137 062534      JMP      @#FPSPUR      ;OTHERWISE GO REPORT A SPURIOUS
9106      ;FP TRAP.
9107 045712 022626      26$:    CMP      (SP)+,(SP)+  ;RESET THE STACK.
9108 045714 010237 001236      MOV      R2,@#TMP2     ;SAVE DATA FOR ERROR REPORT.
9109 045720 170204      STFPS   R4             ;GET FPS.
9110 045722 170305      STST    R5             ;GET FEC.

```



```

9111 045724 012700 000200      MOV      #200,R0          ;SET FD MODE.
9112 045730 170100      LDFPS   R0
9113 045732 012700 046004      MOV      #OVFNTT,R0      ;GET THE RESULT.
9114 045736 174010      STD     AC0,(R0)
9115 045740 010537 0.1254      MOV      R5,@#STMP11
9116 045744 020561 000024      CMP     R5,24(R1)       ;WAS THE FEC ANTICIPATED?
9117 045750 001004      BNE     27$             ;BRANCH IF NOT ANTICIPATED.
9118
9119 045752 010102      MOV     R1,R2           ;ERROR WAS ANTICIPATED SO
9120 045754 062702 000030      ADD     #30,R2         ;RETURN TO THE ERROR REPORT OF THE
9121                                     ;CALLING ROUTINE.
9122 045760 000112      JMP     (R2)
9123
9124 045762 005761 000026      27$:   TST     26(R1)     ;THE ERROR WAS NOT ANTICIPATED SO
9125                                     ;IT MUST BE REPORTED HERE. FIRST SEE IF EXPECTED
9126                                     ;OVERFLOW OR UNDER FLOW.
9127 045766 001003      BNE     29$             ;BRANCH IF EXPECTING UNDERFLOW
9128
9129                                     ;REPORT TRAPPED ON OVERFLOW WITH FIV=0
9130 045770 104002      28$:   ERROR   2
9131 045772 000161 000036      JMP     36(R1)
9132
9133 045776      29$:                                     ;REPORT TRAPPED ON UNDER FLOW WITH FIU=0
9134 045776 104002      30$:   ERROR   2
9135 046000 000161 000036      JMP     36(R1)
9136
9137 046004 000000 000000 0C0000  OVFNTT: .WORD 0,0,0,0
9138 046012 000000
9139
9140 046014      IIIDGNE:
9141 046014 104413      RSETUP                ;GO INITIALIZE THE FPS AND STACK; AND
9142                                     ;SEE IF THE USER HAS EXPRESSED
9143                                     ;THE DESIRE TO CHANGE THE SOFTWARE
9144                                     ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
9145                                     ;THE USER TYPED CONTROL G?).
9146
9147
9148
9149
9150
9151      ;*****
9152      ;*TEST 51      UNDERFLOW, USING MULF WITH TRAP DISABLED, TEST
9153      ;*
9154      ;*THIS IS A TEST OF THE OVERFLOW AND UNDERFLOW CONDITIONS THAT CAN
9155      ;*ARRISE USING THE MULF INSTRUCTION WITH TRAPS DISABLED. A SUBROUTINE IS
9156      ;*USED TO SET UP THE OPERANDS, EXECUTE THE MULF INSTRUCTION AND
9157      ;*CHECK THE RESULTS.
9158      ;*
9159      ;*****
9160      TST51:  SCOPE
9161
9162      ;UNDERFLOW, EXPONENT OF RESULT=-129
9163      JJJ1:
9164      LPERR      JSR     PC,@#OVUNDNT      ;SET UP THE LOOP ON ERROR ADDRESS.
9165      1$:       .WORD  20200,0           ;AC
9166      .WORD  127272,0
    
```

9167	046036	020000	000000	000000	2\$:	.WORD	20000,0,0,0	;FSRC
9168	046044	000000						
9169	046046	000000	000000	000000	3\$:	.WORD	0,0,0,0	;RES
9170	046054	000000						
9171	046056	000000	000000		4\$:	.WORD	0,0	;ERROR RES.
9172	046062	127272	000000			.WORD	127272,0	
9173	046066	000200			5\$:	200		;FPS BEFORE EXECUTION.
9174	046070	000204				204		;FPS AFTER EXECUTION.
9175	046072	000012			6\$:	12		;FEC
9176	046074	177777				-1		;FLAG
9177	046076	104002			7\$:	ERROR	2	;ST 331 TO 155 INTO 115 (BUT FIU)
9178	046100	000401				BR	8\$	
9179	046102	104002				ERROR	2	;ST 115 (BUT FD)
9180	046104				8\$:			
9181								
9182								
9183	046104							;UNDERFLOW, EXPONENT OF RESULT = -193
9184	046104	104414				LPERR		;SET UP THE LOOP ON ERROR ADDRESS.
9185	046106	004737	046344			JSR	PC, #OVUNDNT	
9186	046112	010200	000000		1\$:	.WORD	10200,0	;AC
9187	046116	123456	000000			.WORD	123456,0	
9188	046122	010000	000000	000000	2\$:	.WORD	10000,0,0,0	;FSRC
9189	046130	000000						
9190	046132	000000	000000	000000	3\$:	.WORD	0,0,0,0	;RES
9191	046140	000000						
9192	046142	000000	000000	123456	4\$:	.WORD	0,0,123456,0	;ERROR RES
9193	046150	000000						
9194	046152	005213			5\$:	5213		;FPS BEFORE EXECUTION.
9195	046154	005204				5204		;FPS AFTER EXECUTION.
9196	046156	000012			6\$:	12		;FEC
9197	046160	177777				-1		;FLAG
9198	046162	104002			7\$:	ERROR	2	;SETTING FIUV OR FIV BAD.
9199	046164	000401				BR	8\$	
9200	046166	104002				ERROR	2	;ST 115 (BUT FD)
9201	046170				8\$:			
9202								
9203								
9204	046170							;OVERFLOW, EXPONENT OF RESULT = 128
9205	046170	104414				LPERR		;SET UP THE LOOP ON ERROR ADDRESS.
9206	046172	004737	046344			JSR	PC, #OVUNDNT	
9207	046176	060200	000000		1\$:	.WORD	60200,0	;AC
9208	046202	065432	000000			.WORD	65432,0	
9209	046206	060000	000000	000000	2\$:	.WORD	60000,0,0,0	;FSRC
9210	046214	000000						
9211	046216	000000	000000	000000	3\$:	.WORD	0,0,0,0	;RES
9212	046224	000000						
9213	046226	000000	000000	065432	4\$:	.WORD	0,0,65432,0	;ERROR RES.
9214	046234	000000						
9215	046236	000200			5\$:	200		;FPS BEFORE EXECUTION.
9216	046240	000206				206		;FPS AFTER EXECUTION.
9217	046242	000010			6\$:	10		;FEC
9218	046244	000000				0		;FLAG
9219	046246	104002			7\$:	ERROR	2	;ST 333 TO 136 INTO 116 (BUT FIV)
9220	046250	000401				BR	8\$	
9221	046252	104002				ERROR	2	;ST 116 (BUT FD)
9222	046254				8\$:			

```

9223
9224
9225      ;OVERFLOW, EXPONENT OF RESULT = 130
9226 046254 JJJ4:
9227 046254 104414      LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.
9228 046256 004737 046344 JSR      PC,@OVUNDNT
9229 046262 060200 000000 1$:      .WORD    60200,0      ;AC
9230 046266 125252 000000      .WORD    125252,0
9231 046272 060200 000000 000000 2$:      .WORD    60200,0,0,0      ;FSRC
9232 046300 000000
9233 046302 000000 000000 000000 3$:      .WORD    0,0,0,0      ;RES
9234 046310 000000
9235 046312 000000 000000 125252 4$:      .WORD    0,0,125252,0      ;ERROR RES.
9236 046320 000000
9237 046322 006211      5$:      6211      ;FPS BEFORE EXECUTION.
9238 046324 006206      ;FPS AFTER EXECUTION.
9239 046326 000010      6$:      10      ;FEC
9240 046330 000000      0      ;FLAG
9241 046332 104002      7$:      ERROR    2      ;SETTING FIUV OR FIV BAD.
9242 046334 000401      BR      8$
9243 046336 104002      ERROR    2      ;ST 116 (BUT FD)
9244 046340 000137 046754      8$:      JMP      @JJJDONE      ;GO TO NEXT TEST.
9245
9246      ;THIS SUBROUTINE, OVUNDNT, IS USED TO SET UP THE OPERANDS, EXECUTE
9247      ;THE MULD INSTRUCTION AND CHECK THE RESULTS OF AN INSTRUCTION WITH
9248      ;OPERANDS WHICH SHOULD RESULT IN EITHER OVERFLOW OR UNDERFLOW. A CALL
9249      ;TO IT IS MADE THUS:
9250      :
9251      :          ACARG: .WORD    X,X,X,X      ;AC OPERAND
9252      :          FSRCARG: .WORD    X,X,X,X      ;FSRC OPERAND
9253      :          RES: .WORD    X,X,X,X      ;EXPECTED RESULT
9254      :          ERRES: .WORD    X,X,X,X      ;ERROR RESULT
9255      :          FPSB: .WORD    X      ;FPS BEFORE EXECUTION
9256      :          FPSA: .WORD    X      ;FPS AFTER EXECUTION
9257      :          FEC: .WORD    X      ;EXPECTED FEC
9258      :          FLAG: .WORD    X      ;0/-1,OVER/UNDER FLOW FLAG
9259      :          ERR1: ERROR    X      ;TRAP ERROR.
9260      :          BR      CONT
9261      :          ERR2: ERROR    X      ;DATA, RESULT ERROR
9262      :          CONT:      ;RETURN ADDRESS
9263      :
9264      ;THE OPERANDS ARE SET UP (USING ACO AS THE ACCUMULATOR). THEN
9265      ;THE MULD INSTRUCTION IS EXECUTED. IF NO TRAP OCCURS THEN THE
9266      ;RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
9267      ;COMPARED WITH FPSA IF THIS TOO IS CORRECT OVUNDNT RETURNS CONTROL
9268      ;TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD OVUNDNT
9269      ;REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE
9270      ;MULD IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
9271      ;ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
9272      ;THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN OVUNDNT
9273      ;WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR2. OTHERWISE THE
9274      ;RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND OVUNDNT WILL
9275      ;REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.
9276      ;IF A TRAP OCCURS (IT SHOULD NOT) THEN OVUNDNT WILL READ THE FEC.
9277      ;SHOULD THE FEC MATCH THE ANTICIPATED FEC OVUNDNT WILL
9278      ;STORE ALL DATA AND TRANSFER CONTROL TO THE ERROR CALL AT ERR1. IF THE
    
```

```

9279 ;FEC IS NOT THE SAME AS THE ANTICIPATED FEC OVUNDNT WILL REPORT
9280 ;THE ERROR AND RETURN TO CONT. NOTE THAT OVUNDNT USES THE FLAG
9281 ;TO TELL WHETHER OR NOT THESE PARTICULAR OPERANDS WILL RESULT IN
9282 ;UNDERFLOW (FLAG=-1) OR OVERFLOW (FLAG=0).
9283
9284 046344 012601 OVUNDNT: MOV (SP)+,R1 ;GET A POINTER TO THE ARGUMENTS.
9285 046346 012700 000200 MOV #200,R0 ;SET FD MODE.
9286 046352 170100 LDFPS R0
9287
9288 046354 010100 MOV R1,R0 ;LOAD ACO, OPERAND.
9289 046356 172410 LDD (R0),ACO
9290
9291 046360 010102 MOV R1,R2 ;SAVE THE DATA PATTERNS IN CASE OF
9292 046362 010237 001240 MOV R2,@#STMP3 ;ERROR.
9293 046366 062702 000010 ADD #10,R2
9294 046372 010237 001242 MOV R2,@#STMP4
9295 046376 062702 000010 ADD #10,R2
9296 046402 010237 001244 MOV R2,@#STMP5
9297 046406 016137 000042 001252 MOV 42(R1),@#STMP10
9298 046414 012737 046744 001246 MOV #OVDNIT,@#STMP6
9299
9300 046422 016100 000040 MOV 40(R1),R0 ;LOAD THE FPS.
9301 046426 170100 LDFPS R0
9302 046430 012737 046452 001236 MOV #1$,@#STMP2
9303 046436 012737 046636 000244 MOV #25$,@#FPVECT ;SET UP THE FP TRAP VECTOR IN CASE
9304 ;OF ERROR.
9305 046444 010100 MOV R1,R0 ;COMPUTE THE ADDRESS OF FSRC.
9306 046446 062700 000010 ADD #10,RC
9307
9308 046452 171010 1$: MULD (R0),ACO ;TEST INSTRUCTION.
9309
9310 046454 170204 2$: STFPS R4 ;GET FPS.
9311 046456 170305 STST R5 ;GET FEC.
9312 046460 012700 000200 MOV #200,R0 ;SET FD MODE.
9313 046464 170100 LDFPS R0
9314 046466 012700 046744 MOV #OVDNIT,R0 ;GET THE RESULT.
9315 046472 174010 STD ACO,(R0)
9316 046474 010437 001250 MOV R4,@#STMP7
9317 046500 010537 001254 MOV R5,@#STMP11
9318
9319 046504 012700 046744 MOV #OVDNIT,R0 ;CHECK THE RESULT.
9320 046510 010102 MOV R1,R2
9321 046512 062702 000020 ADD #20,R2
9322 046516 012703 000004 MOV #4,R3
9323 046522 022022 3$: CMP (R0)+,(R2)+
9324 046524 001015 BNE 15$ ;BRANCH IF INCORRECT.
9325 046526 077303 SOB R3,3$
9326
9327 046530 026104 000042 CMP 42(R1),R4 ;WAS FPS CORRECT?
9328 046534 001002 BNE 10$ ;BRANCH IF FPS IS INCORRECT.
9329
9330 046536 000161 000056 4$: JMP 56(R1) ;RETURN, TEST COMPLETED.
9331
9332 ;REPORT INCORRECT FPS.
9333 046542 005761 000046 10$: TST 46(R1) ;WAS THE RESULT OVER OR UNDER FLOW?
9334 046546 001002 BNE 12$ ;BRANCH IF UNDERFLOW.
    
```

```

9335
9336
9337 046550 104002      11$:  ERROR  2      ;REPORT FPS BAD AFTER OVERFLOW.
9338 046552 000771      BR      4$
9339
9340 046554
9341 046554 104002      12$:
9342 046556 000767      13$:  ERROR  2      ;REPORT FPS BAD AFTER UNDERFLOW.
9343      BR      4$
9344      ;RESULT INCORRECT.
9345 046560 012700 046744 15$:  MOV      #OVDNTT,R0      ;SEE IF FAILURE IS ANTICIPATED
9346 046564 010102      MOV      R1,R2      ;FAILURE.
9347 046566 062702 000030      ADD      #30,R2
9348 046572 012703 000004      MOV      #4,R3
9349 046576 022022      16$:  CMP      (R0)+,(R2)+
9350 046600 001007      BNE     17$      ;BRANCH IF NOT ANTICIPATED.
9351 046602 077303      SOB     R3,16$
9352
9353 046604 010102      MOV      R1,R2      ;ERROR WAS ANTICIPATED SO RETURN
9354 046606 062702 000054      ADD      #54,R2      ;TO THE ERROR REPORT IN THE CALLING
9355 046612 010237 001236      MOV      R2,@#STMP2      ;ROUTINE.
9356 046616 000112      JMP     (R2)
9357
9358 046620 005761 000046 17$:  TST      46(R1)      ;RESULT WAS NOT ANTICIPATED
9359      ;SO ERROR MUST BE REPORTED HERE.
9360      ;FIRST SEE IF ARGUMENTS SHOULD
9361      ;HAVE RESULTED IN OVERFLOW OR UNDER
9362      ;FLOW BY LOOKING AT THE FLAG.
9363 046624 001002      BNE     19$      ;BRANCH IF UNDERFLOW EXPECTED.
9364
9365      ;REPORT RESULT INCORRECT, EXPECTING
9366 046626 104002      18$:  ERROR  2      ;OVERFLOW.
9367 046630 000742      BR      4$
9368
9369 046632      19$:
9370 046632 104002      20$:  ERROR  2      ;REPORT RESULT INCORRECT, EXPECTING
9371 046634 000740      BR      4$      ;UNDERFLOW.
9372
9373      ;IF AN FP TRAP OCCURS COME HFRE.
9374 046636 011602      25$:  MOV      (SP),R2      ;GET ADDRESS OF TRAP.
9375 046640 022702 046454      CMP      #2$,R2      ;WAS THE TRAP DURING THE MULF INSTRUCTION?
9376 046644 001402      BEQ     26$      ;BRANCH IF YES.
9377 046646 000137 062534      JMP      @#FPSPUR      ;OTHERWISE GO REPORT A SPURIOUS
9378      ;FP TRAP.
9379 046652 022626      26$:  CMP      (SP)+,(SP)+      ;RESET THE STACK.
9380 046654 010237 001236      MOV      R2,@#STMP2      ;SAVE DATA FOR ERROR REPORT.
9381 046660 170204      STFPS   R4      ;GET FPS.
9382 046662 170305      STST   R5      ;GET FEC.
9383 046664 012700 000200      MOV      #200,R0      ;SET FD MODE.
9384 046670 170100      LDFPS  R0
9385 046672 012700 046744      MOV      #OVDNTT,R0      ;GET THE RESULT.
9386 046676 174010      STD     A0,(R0)
9387 046700 010537 001254      MOV      R5,@#STMP11
9388 046704 020561 000044      CMP      R5,44(R1)
9389 046710 001004      BNE     27$      ;WAS THE FEC ANTICIPATED?
9390      ;BRANCH IF NOT ANTICIPATED.
    
```

```

9391 046712 010102          MOV    R1,R2          ;ERROR WAS ANTICIPATED SO
9392 046714 062702 000050  ADD    #50,R2        ;RETURN TO THE ERROR REPORT OF THE
9393                                ;CALLING ROUTINE.
9394 046720 000112          JMP    (R2)
9395
9396 046722 005761 000026  27$:  TST    26(R1)    ;THE ERROR WAS NOT ANTICIPATED SO
9397                                ;IT MUST BE REPORTED HERE. FIRST SEE IF EXPECTED
9398                                ;OVERFLOW OR UNDER FLOW.
9399 046726 001003          BNE    29$          ;BRANCH IF EXPECTING UNDERFLOW
9400
9401                                ;REPORT TRAPPED ON OVERFLOW WITH FIV=0
9402 046730 104002 28$:  ERROR  2
9403 046732 000161 000056  JMP    56(R1)
9404
9405 046736 29$:
9406 046736 104002 30$:  ERROR  2          ;REPORT TRAPPED ON UNDER FLOW WITH FIU=0
9407 046740 000161 000056  JMP    56(R1)
9408
9409 046744 000000 000000 000000 OVDNTT: .WORD  0,0,0,0
9410 046752 000000
9411
9412 046754          JJJDONE:
9413 046754 104413          RSETUP          ;GO INITIALIZE THE FPS AND STACK; AND
9414                                ;SEE IF THE USER HAS EXPRESSED
9415                                ;THE DESIRE TO CHANGE THE SOFTWARE
9416                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
9417                                ;THE USER TYPED CONTROL G?).
9418
9419
9420
9421
9422
9423          ;*****
9424          ;*TEST 52          UNDER\OVER FLOW, USING MULF WITH TRAPS ENABLED, TEST
9425          ;*
9426          ;*THIS IS A TEST OF THE UNDERFLOW AND OVERFLOW
9427          ;*CONDITIONS THAT CAN OCCUR USING THE MULF INSTRUCTION.
9428          ;*A SUBROUTINE IS CALLED TO SET UP THE OPERANDS,
9429          ;* EXECUTE THE MULF INSTRUCTION AND CHECK
9430          ;*THE RESULTS. HERE THE PARTICULAR INTERRUPT,
9431          ;*EITHER OVERFLOW OR UNDERFLOW, IS ENABLED SO A TRAP SHOULD
9432          ;*OCCUR.
9433          ;*
9434          ;*****
9435          TST52:  SCOPE
9436          ;UNDERFLOW, EXPONENT OF RESULT = -129
9437          KKK1:
9438          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
9439          JSR    PC,@#OVUNFT
9440          1$:  .WORD  20123,45676          ;AC
9441          2$:  .WORD  20200,0            ;FSRC
9442          3$:  .WORD  123,45676          ;RES
9443          4$:  .WORD  -1,-1            ;ERROR RES.
9444          5$:  2000          ;FPS BEFORE EXECUTION.
9445          102004        ;FPS AFTER EXECUTION.
9446          6$:  12          ;FEC

```

9447 047014 177777  
 9448 047016 104002  
 9449 047020 000401  
 9450 047022 104002  
 9451 047024  
 9452  
 9453  
 9454 047024  
 9455 047024 104414  
 9456 047026 004737 047204  
 9457 047032 010127 127272  
 9458 047036 010200 000000  
 9459 047042 060127 127272  
 9460 047046 177777 177777  
 9461 047052 007017  
 9462 047054 107000  
 9463 047056 000012  
 9464 047060 177777  
 9465 047062 104002  
 9466 047064 000401  
 9467 047066 104002  
 9468 047070  
 9469  
 9470  
 9471 047070  
 9472 047070 104414  
 9473 047072 004737 047204  
 9474 047076 060252 125252  
 9475 047102 060000 000000  
 9476 047106 000052 125252  
 9477 047112 177777 177777  
 9478 047116 001000  
 9479 047120 101006  
 9480 047122 000010  
 9481 047124 000000  
 9482 047126 104002  
 9483 047130 000401  
 9484 047132 104002  
 9485 047134  
 9486  
 9487  
 9488 047134  
 9489 047134 104414  
 9490 047136 004737 047204  
 9491 047142 060345 067654  
 9492 047146 060200 000000  
 9493 047152 000345 067654  
 9494 047156 177777 177777  
 9495 047162 007015  
 9496 047164 107002  
 9497 047166 000010  
 9498 047170 000000  
 9499 047172 104002  
 9500 047174 000401  
 9501 047176 104002  
 9502 047200 000167 000412

```

-1 ;FLAG
7$: ERROR 2 ;ST 331 (BUT FIU) NO TRAP.
BR 8$
ERROR 2
8$:
;UNDERFLOW, EXPONENT OF THE RESULT = -193
KKK3:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC, @#OVUNFT
1$: .WORD 10127,127272 ;AC
2$: .WORD 10200,0 ;FSRC
3$: .WORD 60127,127272 ;RES
4$: .WORD -1,-1 ;ERROR RES.
5$: 7017 ;FPS BEFORE EXECUTION.
107000 ;FPS AFTER EXECUTION.
6$: 12 ;FEC
-1
7$: ERROR 2 ;ST 331 (BUT FIU) NO TRAP.
BR 8$
ERROR 2
8$:
;OVERFLOW, EXPONENT OF THE RESULT = 128
KKK4:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC, @#OVUNFT
1$: .WORD 60252,125252 ;AC
2$: .WORD 60000,0 ;FSRC
3$: .WORD 000052,125252 ;RES
4$: .WORD -1,-1 ;ERROR RES.
5$: 1000 ;FPS BEFORE EXECUTION.
101006 ;FPS AFTER EXECUTION.
6$: 10 ;FEC
0 ;FLAG
7$: ERROR 2 ;ST 333 (BUT FIV) NO TRAP
BR 8$
ERROR 2
8$:
;OVERFLOW, EXPONENT OF RESULT = 130
KKK5:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC, @#OVUNFT
1$: .WORD 60345,67654 ;AC
2$: .WORD 60200,0 ;FSRC
3$: .WORD 345,67654 ;RES
4$: .WORD -1,-1 ;ERROR RES.
5$: 7015 ;FPS BEFORE EXECUTION.
107002 ;FPS AFTER EXECUTION.
6$: 10 ;FEC
0 ;FLAG
7$: ERROR 2 ;ST 133 (BUT FIV) NO TRAP
BR 8$
ERROR 2
8$: JMP KKKDONE
  
```

9503  
9504  
9505  
9506  
9507  
9508  
9509  
9510  
9511  
9512  
9513  
9514  
9515  
9516  
9517  
9518  
9519  
9520  
9521  
9522  
9523  
9524  
9525  
9526  
9527  
9528  
9529  
9530  
9531  
9532  
9533  
9534  
9535  
9536  
9537  
9538  
9539  
9540  
9541  
9542  
9543  
9544  
9545  
9546  
9547  
9548  
9549  
9550  
9551  
9552  
9553  
9554  
9555  
9556  
9557  
9558

047204 012601  
047206 012700 000200  
047212 170100  
  
047214 010100  
047216 172410  
  
047220 010102  
047222 010237 001240  
047226 062702 000004  
047232 010237 001242  
047236 062702 000004  
047242 010237 001244  
047246 016137 000022 001252  
047254 012737 047606 001246  
  
047262 016100 000020  
047266 170100  
047270 012737 047312 001236

:THIS SUBROUTINE, OVUNFT, IS USED TO SET UP THE OPERANDS, EXECUTE  
:THE MULF INSTRUCTION AND CHECK THE RESULTS OF AN INSTRUCTION WITH  
:OPERANDS WHICH SHOULD RESULT IN EITHER OVERFLOW OR UNDERFLOW. A CALL  
:TO IT IS MADE THUS:

```
.....
ACARG: .WORD X,X ;AC OPERAND
FSRCARG: .WORD X,X ;FSRC OPERAND
RES: .WORD Y,X ;EXPECTED RESULT
ERRES: .WORD X,X ;ERROR RESULT
FPSB: .WORD X ;FPS BEFORE EXECUTION
FPSA: .WORD X ;FPS AFTER EXECUTION
FEC: .WORD X ;EXPECTED FEC
FLAG: .WORD X ;0/-1,OVER/UNDER FLOW FLAG
ERR1: ERROR X ;TRAP ERROR.
BR CONT
ERR2: ERROR X ;DATA, RESULT ERROR
CONT: ;RETURN ADDRESS
.....
```

:THE OPERANDS ARE SET UP (USING ACO AS THE ACCUMULATOR). THEN  
:THE MULF INSTRUCTION IS EXECUTED. IF THE TRAP OCCURS THEN THE  
:RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS  
:COMPARED WITH FPSA IF THIS TOO IS CORRECT OVUNFT RETURNS CONTROL  
:TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD OVUNFT  
:REPORTS THIS FAILURE AND THEN RETURNS TO CONT. THE FEC IS TREATED  
:IN THE SAME WAY. IF THE RESULT OF THE  
:MULF IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE  
:ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN  
:THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN OVUNFT  
:WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR2. OTHERWISE THE  
:RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND OVUNFT WILL  
:REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.  
:IF NO TRAP OCCURS CONTROL IS PASSED TO ERR1.  
:NOTE THAT OVUNFT USES THE FLAG  
:TO TELL WHETHER OR NOT THESE PARTICULAR OPERANDS WILL RESULT IN  
:UNDERFLOW (FLAG=-1) OR OVERFLOW (FLAG=0).

```
OVUNFT: MOV (SP)+,R1 ;GET A POINTER TO THE ARGUMENTS.
MOV #200,R0 ;SET FD MODE.
LDFPS R0

MOV R1,R0 ;LOAD ACO, OPERAND.
LDD (R0),ACO

MOV R1,R2 ;SAVE THE DATA PATTERNS IN CASE OF
MOV R2,@#STMP3 ;ERROR.
ADD #4,R2
MOV R2,@#STMP4
ADD #4,R2
MOV R2,@#STMP5
MOV 22(R1),@#STMP10
MOV #OVFTT,@#STMP6

MOV 20(R1),R0 ;LOAD THE FPS.
LDFPS R0
MOV #1$,@#STMP2
```



```

9559 047276 012737 047322 000244      MOV      #50$,@#FPVECT      ;SET UP THE FP TRAP VECTOR IN CASE
9560                                     ;OF ERROR.
9561 047304 010100      MOV      R1,R0              ;COMPUTE THE ADDRESS OF FSRC.
9562 047306 062700 000004      ADD      #4,R0
9563
9564 047312 171010      1$:     MULF      (R0),ACO    ;TEST INSTRUCTION. SHOULD CAUSE TRAP.
9565 047314 170000      2$:     CFCC
9566
9567 047316 000137 047546      JMP      @#25$              ;FAILURE, NO TRAP.
9568
9569 047322 011602      50$:    MOV      (SP),R2      ;TRAP TO HERE AND SEE IF THE PC OF THE
9570 047324 020227 047314      CMP      R2,#2$            ;TRAP WAS THAT OF THE MULF INSTRUCTION.
9571 047330 001402      BEQ      51$              ;BRANCH IF YES.
9572 047332 000137 062534      JMP      @#FPSPUR         ;OTHERWISE REPORT SPURIOUS FP ERROR.
9573
9574 047336 022626      51$:    CMP      (SP)+,(SP)+    ;RESET THE STACK
9575 047340 170204      STFPS   R4                ;GET FPS.
9576 047342 170305      STST   R5                ;GET FEC.
9577 047344 012700 000200      MOV      #200,R0          ;SET FD MODE.
9578 047350 170100      LDFPS  R0
9579 047352 012700 047606      MOV      #OVFTT,R0        ;GET THE RESULT.
9580 047356 174010      STD    ACO,(R0)
9581 047360 010437 001250      MOV      R4,@#STMP7
9582 047364 010537 001254      MOV      R5,@#STMP11
9583
9584 047370 012700 047606      MOV      #OVFTT,R0        ;CHECK THE RESULT.
9585 047374 010102      MOV      R1,R2
9586 047376 062702 000010      ADD      #10,R2
9587 047402 012703 000002      MOV      #2,R3
9588 047406 022022      3$:    CMP      (R0)+,(R2)+    ;BRANCH IF INCORRECT.
9589 047410 001027      BNE     15$
9590 047412 077303      SOB    R3,3$
9591
9592 047414 026104 000022      CMP      22(R1),R4        ;WAS FPS CORRECT?
9593 047420 001014      BNE     10$              ;BRANCH IF FPS IS INCORRECT.
9594
9595 047422 026105 000024      CMP      24(R1),R5        ;IS FEC CORRECT?
9596 047426 001002      BNE     5$              ;IF INCORRECT BRANCH.
9597 047430 000161 000036      4$:    JMP      36(R1)          ;RETURN, TEST COMPLETED.
9598
9599      ;REPORT INCORRECT FEC.
9600 047434 005761 000026      5$:    TST      26(R1)        ;WAS THE RESULT OVERFLOW OR UNDERFLOW?
9601 047440 001002      BNE     7$              ;BRANCH IF UNDERFLOW.
9602
9603      ;REPORT BAD FEC ON EXPECTED OVERFLOW.
9604 047442 104002      6$:    ERROR   2
9605 047444 000771      BR      4$
9606
9607 047446      7$:
9608 047446 104002      8$:    ERROR   2
9609 047450 000767      BR      4$
9610
9611      ;REPORT INCORRECT FPS.
9612 047452 005761 000026      10$:   TST      26(R1)        ;WAS THE RESULT OVER OR UNDER FLOW?
9613 047456 001002      BNE     12$            ;BRANCH IF UNDERFLOW.
9614

```

```

9615                                     ;REPORT FPS BAD AFTER OVERFLOW.
9616 047460 104002 11$: ERROR 2
9617 047462 000762 BR 4$
9618
9619 047464 12$:
9620 047464 104002 13$: ERROR 2 ;REPORT FPS BAD AFTER UNDERFLOW.
9621 047466 000760 BR 4$
9622
9623 ;RESULT INCORRECT.
9624 047470 012700 047606 15$: MOV #OVFTT,R0 ;SEE IF FAILURE IS ANTICIPATED
9625 047474 010102 MOV R1,R2 ;FAILURE.
9626 047476 062702 000014 ADD #14,R2
9627 047502 012703 000002 MOV #2,R3
9628 047506 022022 16$: CMP (R0)+,(R2)+ ;BRANCH IF NOT ANTICIPATED.
9629 047510 001007 BNE 17$
9630 047512 077303 SOB R3,16$
9631
9632 047514 010102 MOV R1,R2 ;ERROR WAS ANTICIPATED SO RETURN
9633 047516 062702 000034 ADD #34,R2 ;TO THE ERROR REPORT IN THE CALLING
9634 047522 010237 001236 MOV R2,@$TMP2 ;ROUTINE.
9635 047526 000112 JMP (R2)
9636
9637 047530 005761 000026 17$: TST 26(R1) ;RESULT WAS NOT ANTICIPATED
9638 ;SO ERROR MUST BE REPORTED HERE.
9639 ;FIRST SEE IF ARGUMENTS SHOULD
9640 ;HAVE RESULTED IN OVERFLOW OR UNDER
9641 ;FLOW BY LOOKING AT THE FLAG.
9642 047534 001002 BNE 19$ ;BRANCH IF UNDERFLOW EXPECTED.
9643
9644
9645 047536 104002 18$: ERROR 2 ;REPORT RESULT INCORRECT, EXPECTING
9646 047540 000733 BR 4$ ;OVERFLOW.
9647
9648 047542 19$:
9649 047542 104002 20$: ERROR 2 ;REPORT RESULT INCORRECT, EXPECTING
9650 047544 000731 BR 4$ ;UNDERFLOW.
9651
9652 ;IF NO FP TRAP OCCURS COME HERE.
9653 047546 170204 25$: STFPS R4 ;GET FPS.
9654 047550 170305 STST R5 ;GET FEC.
9655 047552 012700 000200 MOV #200,R0 ;SET FD MODE.
9656 047556 170100 LDFPS R0
9657 047560 012700 047606 MOV #OVFTT,R0 ;GET THE RESULT.
9658 047564 174010 STD ACO,(R0)
9659 047566 010437 001250 MOV R4,@$TMP7
9660 047572 010537 001254 MOV R5,@$TMP11
9661 047576 010102 MOV R1,R2
9662 047600 062702 000030 ADD #30,R2 ;ERROR WAS ANTICIPATED SO
9663 ;RETURN TO THE ERROR REPORT OF THE
9664 047604 000112 JMP (R2) ;CALLING ROUTINE.
9665
9666 047606 000000 000000 000000 OVFTT: .WORD 0,0,0,0
9667 047614 000000
9668
9669 047616 KKKDONE:
9670 047616 104413 RSETUP ;GO INITIALIZE THE FPS AND STACK; AND

```

9671  
9672  
9673  
9674  
9675  
9676  
9677  
9678  
9679  
9680  
9681  
9682  
9683  
9684  
9685  
9686  
9687  
9688  
9689  
9690  
9691  
9692  
9693  
9694  
9695  
9696  
9697  
9698  
9699  
9700  
9701  
9702  
9703  
9704  
9705  
9706  
9707  
9708  
9709  
9710  
9711  
9712  
9713  
9714  
9715  
9716  
9717  
9718  
9719  
9720  
9721  
9722  
9723  
9724  
9725  
9726

047620 000004  
  
047622  
047622 104414  
047624 004737 050146  
047630 020052 125252  
047634 125252 125252  
047640 020300 000000 U00000  
047646 000000  
047650 000177 177777 177777  
047656 177777  
047660 000177 177777  
047664 125252 125252  
047670 002200  
047672 102204  
047674 000012  
047676 177777  
047700 104002  
047702 000401  
047704 104002  
047706  
  
047706  
047706 104414  
047710 004737 050146  
047714 010327 127272  
047720 036363 045454  
047724 010000 000000 000000  
047732 000000  
047734 060127 127272  
047740 036363 045454  
047744 177777 177777 177777  
047752 177777  
047754 007217  
047756 107200  
047760 000012  
047762 177777  
047764 104002

```
;SEE IF THE USER HAS EXPRESSED  
;THE DESIRE TO CHANGE THE SOFTWARE  
;VIRTUAL CONSOLE SWITCH REGISTER (HAS  
;THE USER TYPED CONTROL G?).  
  
:*****  
:TEST 53 UNDER\OVER FLOW, USING MULD WITH TRAPS ENABLED, TEST  
:*****  
:THIS IS A TEST OF THE OVER FLOW AND UNDER FLOW CONDITIONS USING THE  
:MULD INSTRUCTION WITH TRAPS ENABLED. A SUBROUTINE IS USED TO SET UP  
:THE OPERANDS, EXECUTE THE MULD INSTRUCTION AND CHECK THE RESULTS.  
:*****  
:ST53: SCOPE  
  
;UNDERFLOW, EXPONENT OF RESULT = -29  
LLL1:  
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.  
JSR PC,@#OVUNDT  
1$: .WORD 20052,125252 ;AC  
.WORD 125252,125252  
2$: .WORD 20300,0,0,0 ;FSRC  
3$: .WORD 177,-1,-1,-1 ;RES  
4$: .WORD 177,-1 ;ERROR RES.  
.WORD 125252,125252  
5$: 2200 ;FPS BEFORE EXECUTION.  
102204 ;FPS AFTER EXECUTION.  
6$: 12 ;FEC  
-1 ;FLAG  
7$: ERROR 2 ;ST 331 (BUT FIU) NO TRAP.  
BR 8$  
8$: ERROR 2 ;ST 155 (BUT FD)  
  
;UNDERFLOW, EXPONENT OF THE RESULT = -193  
LLL2:  
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.  
JSR PC,@#OVUNDT  
1$: .WORD 10327,127272 ;AC  
.WORD 36363,45454  
2$: .WORD 10000,0,0,0 ;FSRC  
3$: .WORD 60127,127272 ;RES  
.WORD 36363,45454  
4$: .WORD -1,-1,-1,-1 ;ERROR RES.  
5$: 7217 ;FPS BEFORE EXECUTION.  
107200 ;FPS AFTER EXECUTION.  
6$: 12 ;FEC  
-1 ;FLAG  
7$: ERROR 2 ;ST 137 (BUT FIU) NO TRAP.
```

```

9727 047766 000401          BR      8$
9728 047770 104002          ERROR   2
9729 047772          8$:
9730
9731          ;OVERFLOW, EXPONENT OF THE RESULT = 128
9732 047772          LLL3:
9733 047772 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
9734 047774 004737 050146  JSR      PC,@#OVUNDT
9735 050000 060252 125252  1$:      .WORD   60252,125252      ;AC
9736 050004 125252 125252          .WORD   125252,125252      ;FSRC
9737 050010 160100 000000 000000 2$:      .WORD   160100,0,0,0      ;FSRC
9738 050016 000000
9739 050020 100177 177777 177777 3$:      .WORD   100177,-1,-1,-1 ;RES
9740 050026 177777
9741 050030 100177 177777 4$:      .WORD   100177,-1      ;ERROR RES.
9742 050034 125252 125252          .WORD   125252,125252
9743 050040 001200          5$:      1200          ;FPS BEFORE EXECUTION.
9744 050042 101216          ;FPS AFTER EXECUTION.
9745 050044 000010          6$:      10          ;FEC
9746 050046 000000          ;FLAG
9747 050050 104002          7$:      0          ;ST 333 (BUT FIV) NO TRAP.
9748 050052 000401          ERROR   2
9749 050054 104002          BR      8$
9750 050056          ERROR   2          ;ST 700 (BUT FD).
9751
9752          ;OVERFLOW, EXPONENT OF THE RESULT = 130
9753          LLL4:
9754 050056 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
9755 050060 004737 050146  JSR      PC,@#OVUNDT
9756 050064 060345 067654  1$:      .WORD   60345,67654      ;AC
9757 050070 056765 045676          .WORD   56765,45676
9758 050074 060200 000000 000000 2$:      .WORD   60200,0,0,0      ;FSRC
9759 050102 000000
9760 050104 000345 067654          3$:      .WORD   345,67654      ;RES
9761 050110 056765 045676          .WORD   56765,45676
9762 050114 177777 177777 177777 4$:      .WORD   -1,-1,-1,-1      ;ERROR RES.
9763 050122 177777
9764 050124 007215          5$:      7215          ;FPS BEFORE EXECUTION.
9765 050126 107202          ;FPS AFTER EXECUTION.
9766 050130 000010          6$:      10          ;FEC
9767 050132 000000          ;FLAG
9768 050134 104002          7$:      0          ;ST 133 (BUT FIV) NO TRAP
9769 050136 000401          ERROR   2
9770 050140 104002          BR      8$
9771 050142 000137 050560 8$:      ERROR   2
9772          JMP      @#LLLDONE
9773
9774          ;THIS SUBROUTINE, OVUNDT, IS USED TO SET UP THE OPEPANDS, EXECUTE
9775          ;THE MULD INSTRUCTION AND CHECK THE RESULTS OF AN INSTRUCTION WITH
9776          ;OPERANDS WHICH SHOULD RESULT IN EITHER OVERFLOW OR UNDERFLOW. A CALL
9777          ;TO IT IS MADE THUS:
9778          :
9779          :          ACARG: .WORD   X,Y,X,X          ;AC OPERAND
9780          :          FSRCARG: .WORD   X,X,X,X          ;FSRC OPERAND
9781          :          RES:      .WORD   X,X,X,X          ;EXPECTED RESULT
9782          :          ERRES:  .WORD   X,X,X,X          ;ERROR RESULT
          :          FPSB:   .WORD   X          ;FPS BEFORE EXECUTION

```

```

9783      :          FPSA:  .WORD  X          ;FPS AFTER EXECUTION
9784      :          FEC:   .WORD  X          ;EXPECTED FEC
9785      :          FLAG:  .WORD  X          ;0/-1,OVER/UNDER FLOW FLAG
9786      :          ERR1:  ERROR  X          ;TRAP ERROR.
9787      :          BR     CONT          ;
9788      :          ERR2:  ERROR  X          ;DATA, RESULT ERROR
9789      :          CONT:                ;RETURN ADDRESS
9790      :
9791      :
9792      :
9793      :
9794      :
9795      :
9796      :
9797      :
9798      :
9799      :
9800      :
9801      :
9802      :
9803      :
9804      :
9805      :
9806      :
9807      :
9808      :
    
```

```

;THE OPERANDS ARE SET UP (USING ACO AS THE ACCUMULATOR). THEN
;THE MULD INSTRUCTION IS EXECUTED. IF THE TRAP OCCURS THEN THE
;RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
;COMPARED WITH FPSA IF THIS TOO IS CORRECT OVUNDT RETURNS CONTROL
;TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD OVUNDT
;REPORTS THIS FAILURE AND THEN RETURNS TO CONT. THE FEC IS TREATED
;IN THE SAME WAY. IF THE RESULT OF THE
;MULF IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
;ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
;THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN OVUNDT
;WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR2. OTHERWISE THE
;RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND OVUNDT WILL
;REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.
;IF NO TRAP OCCURS CONTROL IS PASSED TO ERR1.
;NOTE THAT OVUNDT USES THE FLAG
;TO TELL WHETHER OR NOT THESE PARTICULAR OPERANDS WILL RESULT IN
;UNDERFLOW (FLAG=-1) OR OVERFLOW (FLAG=0).
    
```

```

9809 050146 012601          OVUNDT: MOV    (SP)+,R1      ;GET A POINTER TO THE ARGUMENTS.
9810 050150 012700 000200  MOV    #200,R0      ;SET FD MODE.
9811 050154 170100          LDFPS  R0
9812
9813 050156 010100          MOV    R1,R0        ;LOAD ACO, OPERAND.
9814 050160 172410          LDD   (R0),ACO
9815
9816 050162 010102          MOV    R1,R2        ;SAVE THE DATA PATTERNS IN CASE OF
9817 050164 010237 001240  MOV    R2,@#STMP3   ;ERROR.
9818 050170 062702 000010  ADD    #10,R2
9819 050174 010237 001242  MOV    R2,@#STMP4
9820 050200 062702 000010  ADD    #10,R2
9821 050204 010237 001244  MOV    R2,@#STMP5
9822 050210 016137 000042 001252  MOV    42(R1),@#STMP10
9823 050216 012737 050550 001246  MOV    #OVDTT,@#STMP6
9824
9825 050224 016100 000040          MOV    40(R1),R0    ;LOAD THE FPS.
9826 050230 170100          LDFPS  R0
9827 050232 012737 050254 001236  MOV    #1$,@#STMP2
9828 050240 012737 050264 000244  MOV    #50$,@#FPVECT ;SET UP THE FP TRAP VECTOR IN CASE
9829                                ;OF ERROR.
9830 050246 010100          MOV    R1,R0        ;COMPUTE THE ADDRESS OF FSRC.
9831 050250 062700 000010  ADD    #10,R0
9832
9833 050254 171010          1$:   MULD  (R0),ACO    ;TEST INSTRUCTION. SHOULD CAUSE TRAP.
9834 050256 170000          2$:   CFCC
9835
9836 050260 000137 050510          JMP    @#25$        ;FAILURE, NO TRAP.
9837
9838 050264 011602          50$:  MOV    (SP),R2    ;TRAP TO HERE AND SEE IF THE PC OF THE
    
```

```

9839 050266 020227 050256      CMP      R2,#2$      ;TRAP WAS THAT OF THE MULF INSTRUCTION.
9840 050272 001402      BEQ      51$         ;BRANCH IF YES.
9841 050274 000137 062534      JMP      @#FPSPUR   ;OTHERWISE REPORT SPURIOUS FP ERROR.
9842
9843 050300 022626      51$:    CMP      (SP)+,(SP)+ ;RESET THE STACK
9844 050302 170204      STFPS   R4         ;GET FPS.
9845 050304 170305      STST    R5         ;GET FEC.
9846 050306 012700 000200      MOV     #200,R0    ;SET FD MODE.
9847 050312 170100      LDFPS  R0
9848 050314 012700 050550      MOV     #OVDTT,R0 ;GET THE RESULT.
9849 050320 174010      STD    ACO,(R0)
9850 050322 010437 001250      MOV     R4,@#STMP7
9851 050326 010537 001254      MOV     R5,@#STMP11
9852
9853 050332 012700 050550      MOV     #OVDTT,R0 ;CHECK THE RESULT.
9854 050336 010102      MOV     R1,R2
9855 050340 062702 000020      ADD    #20,R2
9856 050344 012703 000004      MOV     #4,R3
9857 050350 022022      3$:    CMP     (R0)+,(R2)+
9858 050352 001027      BNE    15$         ;BRANCH IF INCORRECT.
9859 050354 077303      SOB    R3,3$
9860
9861 050356 026104 000042      CMP     42(R1),R4 ;WAS FPS CORRECT?
9862 050362 001014      BNE    10$         ;BRANCH IF FPS IS INCORRECT.
9863
9864 050364 026105 000044      CMP     44(R1),R5 ;IS FEC CORRECT?
9865 050370 001002      BNE    5$         ;IF INCORRECT BRANCH.
9866 050372 000161 000056      4$:    JMP     56(R1)    ;RETURN, TEST COMPLETED.
9867
9868      ;REPORT INCORRECT FEC.
9869 050376 005761 000046      5$:    TST    46(R1)   ;WAS THE RESULT OVERFLOW OR UNDERFLOW?
9870 050402 001002      BNE    7$         ;BRANCH IF UNDERFLOW.
9871
9872      ;REPORT BAD FEC ON EXPECTED OVERFLOW.
9873 050404 104002      6$:    ERROR  2
9874 050406 000771      BR     4$
9875
9876 050410      7$:
9877 050410 104002      8$:    ERROR  2      ;REPORT BAD FEC ON EXPECTED UNDERFLOW.
9878 050412 000767      BR     4$
9879
9880      ;REPORT INCORRECT FPS.
9881 050414 005761 000046      10$:   TST    46(R1)   ;WAS THE RESULT OVER OR UNDER FLOW?
9882 050420 001002      BNE    12$        ;BRANCH IF UNDERFLOW.
9883
9884      ;REPORT FPS BAD AFTER OVERFLOW.
9885 050422 104002      11$:   ERROR  2
9886 050424 000762      BR     4$
9887
9888 050426      12$:
9889 050426 104002      13$:   ERROR  2      ;REPORT FPS BAD AFTER UNDERFLOW.
9890 050430 000760      BR     4$
9891
9892      ;RESULT INCORRECT.
9893 050432 012700 050550      15$:   MOV     #OVDTT,R0 ;SEE IF FAILURE IS ANTICIPATED
9894 050436 010102      MOV     R1,R2     ;FAILURE.
    
```

```

9895 050440 062702 000030      ADD      #30,R2
9896 050444 012703 000004      MOV      #4,R3
9897 050450 022022      16$:    CMP      (R0)+,(R2)+
9898 050452 001007      BNE      17$
9899 050454 077303      SOB      R3,16$
9900
9901 050456 010102      MOV      R1,R2
9902 050460 062702 000054      ADD      #54,R2
9903 050464 010237 001236      MOV      R2,@#STMP2
9904 050470 000112      JMP      (R2)
9905
9906 050472 005761 000046      17$:    TST      46(R1)
9907
9908
9909
9910
9911 050476 001002      BNE      19$
9912
9913
9914 050500 104002      18$:    ERROR   2
9915 050502 000733      BR       4$
9916
9917 050504      19$:
9918 050504 104002      20$:    ERROR   2
9919 050506 000731      BR       4$
9920
9921
9922 050510 170204      ;IF NO FP TRAP OCCURS COME HERE.
9923 050512 170305      25$:    STFPS   R4
9924 050514 012700 000200      STST    R5
9925 050520 170100      MOV      #200,R0
9926 050522 012700 050550      LDFPS   R0
9927 050526 174010      MOV      #OVDTT,R0
9928 050530 010437 001250      STD     ACO,(R0)
9929 050534 010537 001254      MOV      R4,@#STMP7
9930 050540 010102      MOV      R5,@#STMP11
9931 050542 062702 000050      MOV      R1,R2
9932
9933 050546 000112      ADD      #50,R2
9934
9935 050550 000000 000000 000000 OVDTT: .WORD 0,0,0,0
9936 050556 000000
9937
9938
9939 050560 104413      LLLDONE:
9940
9941
9942
9943
9944
9945
9946
9947
9948
9949
9950
    ;*****
    ;*TEST 54      MODF TEST
    ;GO INITIALIZE THE FPS AND STACK; AND
    ;SEE IF THE USER HAS EXPRESSED
    ;THE DESIRE TO CHANGE THE SOFTWARE
    ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
    ;THE USER TYPED CONTROL G?).
    ;REPORT RESULT INCORRECT, EXPECTING
    ;OVERFLOW.
    ;REPORT RESULT INCORRECT, EXPECTING
    ;UNDERFLOW.
    ;RESULT WAS NOT ANTICIPATED
    ;SO ERROR MUST BE REPORTED HERE.
    ;FIRST SEE IF ARGUMENTS SHOULD
    ;HAVE RESULTED IN OVERFLOW OR UNDER
    ;FLOW BY LOOKING AT THE FLAG.
    ;BRANCH IF UNDERFLOW EXPECTED.
    ;BRANCH IF NOT ANTICIPATED.
    ;ERROR WAS ANTICIPATED SO RETURN
    ;TO THE ERROR REPORT IN THE CALLING
    ;ROUTINE.
    
```

```

9951
9952
9953
9954
9955
9956
9957 050562 000004
9958
9959
9960 050564
9961 050564 104414
9962 050566 004737 051650
9963 050572 000000 000000
9964 050576 000000 000000
9965 050602 000000 000000
9966 050606 000000 000000
9967 050612 177777 177777
9968 050616 177777 177777
9969 050622 000013
9970 050624 000004
9971 050626 104002
9972 050630 000401
9973 050632 104002
9974 050634
9975
9976
9977 050634
9978 050634 104414
9979 050636 004737 051650
9980 050642 123456 076543
9981 050646 000000 000000
9982 050652 000000 000000
9983 050656 000000 000000
9984 050662 123456 076543
9985 050666 177777 177777
9986 050672 000000
9987 050674 000004
9988 050676 104002
9989 050700 000401
9990 050702 104002
9991 050704
9992
9993
9994 050704
9995 050704 104414
9996 050706 004737 051650
9997 050712 000000 000000
9998 050716 076543 021234
9999 050722 000000 000000
10000 050726 000000 000000
10001 050732 000000 000000
10002 050736 177777 177777
10003 050742 000003
10004 050744 000004
10005 050746 104002
10006 050750 000401

```

```

:
:
: * THIS IS A TEST OF THE MODF INSTRUCTION, WHICH MAKES USE OF
: * A SUBROUTINE TO SET UP THE OPERANDS, EXECUTE THE MODF INSTRUCTION
: * AND CHECK THE RESULTS.
:
: *****
TST54: SCOPE

;MODF WITH (FSRC=AC=0)
GGG1:
    LPERR
    JSR PC,@#MODFSUB ;SET UP THE LOOP ON ERROR ADDRESS.
1$: .WORD 0,0 ;AC
2$: .WORD 0,0 ;FSRC
3$: .WORD 0,0 ;FRACTIONAL RES.
4$: .WORD 0,0 ;INTEGER RES.
5$: .WORD -1,-1 ;ERROR FRACTIONAL RES.
6$: .WORD -1,-1 ;ERROR INGETER RES.
7$: 13 ;FPS BEFORE EXECUTION.
    4 ;FPS AFTER EXECUTION.
8$: ERROR 2 ;STORE SINGLE ZERO BAD.
    BR 9$
9$: ERROR 2 ;AC V 1 <= ZERO FAILED.

;MODF TEST, WITH (FSRC=0)
GGG2:
    LPERR
    JSR PC,@#MODFSUB ;SET UP THE LOOP ON ERROR ADDRESS.
1$: .WORD 123456,76543 ;AC
2$: .WORD 0,0 ;FSRC
3$: .WORD 0,0 ;FRACTIONAL RES.
4$: .WORD 0,0 ;INTEGER RESULT.
5$: .WORD 123456,76543 ;ERROR FRACTIONAL RES.
6$: .WORD -1,-1 ;ERROR INTEGER RES.
7$: 0 ;FPS BEFORE EXECUTION.
    4 ;FPS AFTER EXECUTION.
8$: ERROR 2 ;STORE ZERO FAILURE.
    BR 9$
9$: ERROR 2

;MODF TEST WITH (AC=0)
GGG3:
    LPERR
    JSR PC,@#MODFSUB ;SET UP THE LOOP ON ERROR ADDRESS.
1$: .WORD 0,0 ;AC
2$: .WORD 76543,21234 ;FSRC
3$: .WORD 0,0 ;FRACTIONAL RES.
4$: .WORD 0,0 ;INTEGER RES.
5$: .WORD 0,0 ;ERROR FRACTIONAL RES.
6$: .WORD -1,-1 ;ERROR INTEGER RES.
7$: 3 ;FPS BEFORE EXECUTION.
    4 ;FPS AFTER EXECUTION.
8$: ERROR 2 ;RES.BAD
    BR 9$

```



```
10007 050752 104002          ERROR 2
10008 050754          9$:
10009
10010          ;MODF TEST WITH EXPONENT OF THE RESULT = 25
10011 050754          GGG4:
10012 050754 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
10013 050756 004737 051650          JSR          PC,@#MODFSUB
10014 050762 046252 125252          1$: .WORD 46252,125252          ;AC
10015 050766 040300 000000          2$: .WORD 40300,0          ;FSRC
10016 050772 000000 000000          3$: .WORD 0,0          ;FRACTIONAL RES.
10017 050776 046377 177777          4$: .WORD 46377,-1          ;INTEGER RES.
10018 051002 046252 125252          5$: .WORD 46252,125252          ;ERROR FRACTIONAL RES.
10019 051006 040300 000000          6$: .WORD 40300,0          ;ERROR INTEGER RES.
10020 051012 000013          7$: 13          ;FPS BEFORE EXECUTION.
10021 051014 000004          4          ;FPS AFTER EXECUTION.
10022 051016 104002          8$: ERROR 2          ;ST 134
10023 051020 000401          BR 9$
10024 051022 104002          ERROR 2
10025 051024          9$:
10026
10027          ;MODF TEST WITH EXPONENT OF THE RESULT = 127
10028 051024          GGG5:
10029 051024 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
10030 051026 004737 051650          JSR          PC,@#MODFSUB
10031 051032 077652 125252          1$: .WORD 77652,125252          ;AC
10032 051036 040300 000000          2$: .WORD 40300,0          ;FSRC
10033 051042 000000 000000          3$: .WORD 0,0          ;FRACTIONAL RES.
10034 051046 077777 177777          4$: .WORD 77777,-1          ;INTEGER RES.
10035 051052 077652 125252          5$: .WORD 77652,125252          ;ERROR FRACTIONAL RES.
10036 051056 040300 000000          6$: .WORD 40300,0          ;ERROR INTEGER RES.
10037 051062 000000          7$: 0          ;FPS BEFORE EXECUTION.
10038 051064 000004          4          ;FPS AFTER EXECUTION.
10039 051066 104002          8$: ERROR 2
10040 051070 000401          BR 9$
10041 051072 104002          ERROR 2
10042 051074          9$:
10043
10044          ;MODF TEST WITH EXPONENT OF RESULT = 25
10045 051074          GGG6:
10046 051074 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
10047 051076 004737 051650          JSR          PC,@#MODFSUB
10048 051102 046200 000001          1$: .WORD 46200,1          ;AC
10049 051106 040340 000000          2$: .WORD 40340,0          ;FSRC
10050 051112 000000 000000          3$: .WORD 0,0          ;FRACTIONAL RES.
10051 051116 046340 000001          4$: .WORD 46340,1          ;INTEGER RES.
10052 051122 040000 000000          5$: .WORD 40000,0          ;ERROR FRACTIONAL RES.
10053 051126 177777 177777          6$: .WORD -1,-1          ;ERROR INTEGER RES.
10054 051132 000013          7$: 13          ;FPS BEFORE EXECUTION.
10055 051134 000004          4          ;FPS AFTER EXECUTION.
10056 051136 104002          8$: ERROR 2          ;BAD CONSTANT (NOT 24),
10057          ;OR ST 525 TO 050 INTO 150.
10058 051140 000401          BR 9$
10059 051142 104002          ERROR 2
10060 051144          9$:
10061
10062          ;MODF TEST WITH EXPONENT OF THE RESULT = 24
```

```

10063 051144
10064 051144 104414
10065 051146 004737 051650
10066 051152 046000 000001
10067 051156 040340 000000
10068 051162 040100 000000
10069 051166 046140 000001
10070 051172 000000 000000
10071 051176 177777 177777
10072 051202 000000
10073 051204 000000
10074 051206 104002
10075
10076 051210 000401
10077 051212 104002
10078 051214
10079
10080
10081 051214
10082 051214 104414
10083 051216 004737 051650
10084 051222 042577 177777
10085 051226 040200 000000
10086 051232 040177 176000
10087 051236 042577 140000
10088 051242 177777 177777
10089 051246 177777 177777
10090 051252 000000
10091 051254 000000
10092 051256 104002
10093 051260 000401
10094 051262 104002
10095 051264
10096
10097
10098 051264
10099 051264 104414
10100 051266 004737 051650
10101 051272 042577 140001
10102 051276 040200 000000
10103 051302 034600 000000
10104 051306 042577 140000
10105 051312 000000 000000
10106 051316 177777 177777
10107 051322 000000
10108 051324 000000
10109 051326 104002
10110 051330 000401
10111 051332 104002
10112 051334
10113
10114
10115 051334
10116 051334 104414
10117 051336 004737 051650
10118 051342 042377 100000

GGG7:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,@#MODFSUB
1$: .WORD 46000,1 ;AC
2$: .WORD 40340,0 ;FSRC
3$: .WORD 40100,0 ;FRACTIONAL RES.
4$: .WORD 46140,1 ;INTEGER RESULT.
5$: .WORD 0,0 ;ERROR FRACTIONAL RES.
6$: .WORD -1,-1 ;ERROR INTEGER RES.
7$: 0 ;FPS BEFORE EXECUTION.
0 ;FPS AFTER EXECUTION.
8$: ERROR 2 ;BAD CONSTANT USED (NOT 24)
;OR ST 525 TO 150 INTO 050
BR 9$
ERROR 2
9$:

;MODF TEST WITH EXPONENT OF THE RESULT - 10
GGG8:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,@#MODFSUB
1$: .WORD 42577,-1 ;AC
2$: .WORD 40200,0 ;FSRC
3$: .WORD 40177,176000 ;FRACTIONAL RES.
4$: .WORD 42577,140000 ;INTEGER RES.
5$: .WORD -1,-1 ;ERROR FRACTIONAL RES.
6$: .WORD -1,-1 ;ERROR INTEGER RES.
7$: 0 ;FPS BEFORE EXECUTION.
0 ;FPS AFTER EXECUTION.
8$: ERROR 2
BR 9$
ERROR 2
9$:

;MODF TEST WITH THE EXPONENT OF THE RESULT = 10
GGG9:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,@#MODFSUB
1$: .WORD 42577,140001 ;AC
2$: .WORD 40200,0 ;FSRC
3$: .WORD 34600,0 ;FRACTIONAL RES.
4$: .WORD 42577,140000 ;INTEGER RES.
5$: .WORD 0,0 ;ERROR FRACTIONAL RES.
6$: .WORD -1,-1 ;ERROR INTEGER RES.
7$: 0 ;FPS BEFORE EXECUTION.
0 ;FPS AFTER EXECUTION.
8$: ERROR 2 ;ST 532 TO 122 INTO NORMALIZE.
BR 9$
ERROR 2
9$:

;MODF TEST WITH EXPONENT OF THE RESULT = 9
GGG10:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,@#MODFSUB
1$: .WORD 42377,100000 ;AC
    
```

10119	051346	040200	000000	2\$:	.WORD	40200,0	:FSRC
10120	051352	000000	000000	3\$:	.WORD	0,0	:FRACTIONAL RES.
10121	051356	042377	100000	4\$:	.WORD	42377,100000	:INTEGER RES.
10122	051362	177777	177777	5\$:	.WORD	-1,-1	:ERROR FRACTIONAL RES.
10123	051366	177777	177777	6\$:	.WORD	-1,-1	:ERROR INTEGER RES.
10124	051372	000013		7\$:		13	:FPS BEFORE EXECUTION.
10125	051374	000004				4	:FPS AFTER EXECUTION.
10126	051376	104002		8\$:	ERROR	2	
10127	051400	000401			BR	9\$	
10128	051402	104002			ERROR	2	
10129	051404			9\$:			
10130							
10131							
10132	051404						
10133	051404	104414					
10134	051406	004737	051650		LPERR		:SET UP THE LOOP ON ERROR ADDRESS.
10135	051412	040177	177777		JSR	PC,@#MODFSUB	
10136	051416	040200	000000	1\$:	.WORD	40177,-1	:AC
10137	051422	040177	177777	2\$:	.WORD	40200,0	:FSRC
10138	051426	000000	000000	3\$:	.WORD	40177,-1	:FRACTIONAL RES.
10139	051432	000000	000000	4\$:	.WORD	0,0	:INTEGER RES.
10140	051436	040177	177777	5\$:	.WORD	0,0	:ERROR FRACTIONAL RES.
10141	051442	000017		6\$:	.WORD	40177,-1	:ERROR INTEGER RES.
10142	051444	000000		7\$:		17	:FPS BEFORE EXECUTION.
10143	051446	104002				0	:FPS AFTER EXECUTION.
10144	051450	000401		8\$:	ERROR	2	:ST 041 TO 046 INTO 246.
10145	051452	104002			BR	9\$	
10146	051454				ERROR	2	
10147				9\$:			
10148							
10149	051454						
10150	051454	104414					
10151	051456	004737	0516.0		LPERR		:SET UP THE LOOP ON ERROR ADDRESS.
10152	051462	034377	177777		JSR	PC,@#MODFSUB	
10153	051466	040200	000000	1\$:	.WORD	34377,-1	:AC
10154	051472	034377	177777	2\$:	.WORD	40200,0	:FSRC
10155	051476	000000	000000	3\$:	.WORD	34377,-1	:FRACTIONAL RES.
10156	051502	000000	000000	4\$:	.WORD	0,0	:INTEGER RES.
10157	051506	034377	177777	5\$:	.WORD	0,0	:ERROR FRACTIONAL RES.
10158	051512	000000		6\$:	.WORD	34377,-1	:ERROR INTEGER RES.
10159	051514	000000		7\$:		0	:FPS BEFORE EXECUTION.
10160	051516	104002				0	:FPS AFTER EXECUTION.
10161	051520	000401		8\$:	ERROR	2	
10162	051522	104002			BR	9\$	
10163	051524				ERROR	2	
10164				9\$:			
10165							
10166	051524						
10167	051524	104414					
10168	051526	004737	051650		LPERR		:SET UP THE LOOP ON ERROR ADDRESS.
10169	051532	020000	000001		JSR	PC,@#MODFSUB	
10170	051536	040300	000000	1\$:	.WORD	20000,1	:AC
10171	051542	020100	000002	2\$:	.WORD	40300,0	:FSRC
10172	051546	000000	000000	3\$:	.WORD	20100,2	:FRACTIONAL RES.
10173	051552	020100	000001	4\$:	.WORD	0,0	:INTEGER RES.
10174	051556	000000	000000	5\$:	.WORD	20100,1	:ERROR FRACTIONAL RES.
				6\$:	.WORD	0,0	:ERROR INTEGER RES.

10175 051562 000000  
 10176 051564 000000  
 10177 051566 104002  
 10178 051570 000401  
 10179 051572 104002  
 10180 051574  
 10181  
 10182  
 10183 051574  
 10184 051574 104414  
 10185 051576 004737 051650  
 10186 051602 142777 170000  
 10187 051606 040200 000000  
 10188 051612 140000 000000  
 10189 051616 142777 160000  
 10190 051622 040000 000000  
 10191 051626 042777 160000  
 10192 051632 000007  
 10193 051634 000010  
 10194 051636 104002  
 10195 051640 000401  
 10196 051642 104002  
 10197 051644 000167 000366  
 10198  
 10199  
 10200  
 10201  
 10202  
 10203  
 10204  
 10205  
 10206  
 10207  
 10208  
 10209  
 10210  
 10211  
 10212  
 10213  
 10214  
 10215  
 10216  
 10217  
 10218  
 10219  
 10220  
 10221  
 10222  
 10223  
 10224  
 10225  
 10226  
 10227  
 10228  
 10229  
 10230

7\$: 0 ;FPS BEFORE EXECUTION.  
 0 ;FPS AFTER EXECUTION.  
 8\$: ERROR 2 ;ROUND TRUNK, ST 126 INTO ROUND.  
 BR 9\$  
 ERROR 2  
 9\$:  
 ;MODF TEST WITH EXPONENT OF RESULT = 11  
 GGG14:  
 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.  
 JSR PC,@MODFSUB  
 1\$: .WORD 142777,170000 ;AC  
 2\$: .WORD 40200,0 ;FSRC  
 3\$: .WORD 140000,0 ;FRACTIONAL RES.  
 4\$: .WORD 142777,160000 ;INTEGER RES.  
 5\$: .WORD 40000,0 ;ERROR FRACTIONAL RES.  
 6\$: .WORD 42777,160000 ;ERROR INTEGER RES.  
 7\$:  
 7 ;FPS BEFORE EXECUTION.  
 10 ;FPS AFTER EXECUTION.  
 8\$: ERROR 2 ;SIGN OF FRACTION.  
 BR 9\$  
 ERROR 2 ;SIGN OF INTEGER.  
 9\$: JMP GGGDONE ;GO TO NEXT TEST.

;THIS SUBROUTINE, MODFSUB, IS CALLED TO SETUP THE  
 ;OPERANDS, EXECUTE THE MODF INSTRUCTION AND CHECK THE RESULTS.  
 ;IT IS CALLED THUS:

```

ACARG: .WORD X,X ;AC OPERAND
FSRCARG: .WORD X,X ;FSRC OPERAND
FRES: .WORD X,X ;FRACTIONAL RESULT
INTRES: .WORD X,X ;INTEGER RESULT
ERFRES: .WORD X,X ;ERROR FRACTION RESULT
ERINTRES: .WORD X,X ;ERROR INTEGER RESULT
FPSB: .WORD X ;FPS BEFORE EXECUTION
FPSA: .WORD X ;FPS AFTER EXECUTION
ERR1: ERROR X ;FRACTION ERROR
BR CONT
ERR2: ERROR X ;INTEGER ERROR
CONT: ;RETURN ADDRESS
  
```

;THE OPERANDS ARE SET UP (USING ACO FOR THE AC ARGUMENT). THE MODF  
 ;INSTRUCTION IS EXECUTED. THEN THE RESULTS ARE RETRIEVED.  
 ;THE FRACTION PART OF THE RESULT IS COMPARED WITH FRES. IF THIS IS CORRECT  
 ;THEN THE INTEGER PART IS COMPARED WITH INTRES. IF BOTH OF THESE ARE CORRECT  
 ;THEN THE FPS IS COMPARED WITH FPSA. AFTER EXECUTION IF NO ERRORS OCCURRED  
 ;THEN MODFSUB WILL RETURN TO CONT. IF THE FPS WAS INCORRECT  
 ;IT IS REPORTED HERE. IF THE FRACTION IS INCORRECT IT IS COMPARED WITH  
 ;THE ANTICIPATED BAD FRACTION, ERFRES. IF THIS DOESN'T MATCH  
 ;THE TRUE RESULT THEN THE ERROR IS REPORTED HERE. IF THE ANTICIPATED  
 ;FAILURE MATCHES THE TRUE RESULT THEN MODFSUB PASSES CONTROL TO THE  
 ;ERROR CALL AT ERR1. LIKewise IF THE INTEGER PART OF THE RESULT IS  
 ;NOT CORRECT THEN IT IS COMPARED WITH THE ANTICIPATED INTEGER  
 ;FAILURE. IF THIS DOESN'T MATCH THEN THE ERROR IS REPORTED HERE.  
 ;IF A MATCH IS MADE HOWEVER, MODFSUB WILL RETURN CONTROL TO THE ERROR

```

10231 ;CALL AT ERR2.
10232
10233 051650 012601 MODFSUB: MOV (SP)+,R1 ;GET A POINTER TO THE ARGUMENTS
10234 051652 012700 000200 MOV #200,R0 ;SET FD MODE.
10235 051656 170100 LDFPS R0
10236 051660 010100 MOV R1,R0 ;SET UP ACO
10237 051662 172410 LDD (R0),AC0
10238 051664 012700 052226 MOV #MODP1,R0 ;PUT A BACKGROUND PATTERN INTO AC1.
10239 051670 172510 LDD (R0),AC1
10240 051672 016100 000030 MOV 30(R1),R0 ;SET UP THE FPS.
10241 051676 170100 LDFPS R0
10242 051700 012737 051714 001236 MOV #1$,@#STMP2
10243 051706 010100 MOV R1,R0 ;COMPUTE THE ADDRESS OF THE FSRC.
10244 051710 062700 000004 ADD #4,R0
10245
10246 051714 171410 1$: MODF (R0),AC0 ;EXECUTE THE TEST INSTRUCTION.
10247
10248 051716 170204 STFPS R4 ;GET THE FPS.
10249 051720 012700 000200 MOV #200,R0 ;SET FD MODE.
10250 051724 170100 LDFPS R0
10251 051726 012700 052206 MOV #MODFT0,R0 ;GET THE FRACTIONAL RESULT.
10252 051732 174010 STD ACO,(R0)
10253 051734 012700 052216 MOV #MODFT1,R0 ;GET THE INTEGER RESULT.
10254 051740 174110 STD AC1,(R0)
10255
10256 051742 010102 MOV R1,R2 ;SAVE THE DATA IN CASE OF ERROR.
10257 051744 010237 001240 MOV R2,@#STMP3
10258 051750 062702 000004 ADD #4,R2
10259 051754 010237 001242 MOV R2,@#STMP4
10260 051760 062702 000004 ADD #4,R2
10261 051764 010237 001244 MOV R2,@#STMP5
10262 051770 062702 000004 ADD #4,R2
10263 051774 010237 001246 MOV R2,@#STMP6
10264 052000 012737 052206 001250 MOV #MODFT0,@#STMP7
10265 052006 012737 052216 001252 MOV #MODFT1,@#STMP10
10266 052014 010437 001254 MOV R4,@#STMP11
10267 052020 016137 000032 001256 MOV 32(R1),@#STMP12
10268
10269 052026 012702 052206 MOV #MODFT0,R2 ;CHECK THE FRACTIONAL RESULT.
10270 052032 026112 000010 CMP 10(R1),(R2)
10271 052036 001022 BNE 10$ ;BRANCH IF INCORRECT.
10272 052040 026162 000012 000002 CMP 12(R1),2(R2)
10273 052046 001016 BNE 10$
10274
10275 052050 012702 052216 MOV #MODFT1,R2 ;CHECK THE INTEGER RESULT.
10276 052054 026112 000014 CMP 14(R1),(R2)
10277 052060 001026 BNE 15$ ;BRANCH IF INCORRECT.
10278 052062 026162 000016 000002 CMP 16(R1),2(R2)
10279 052070 001022 BNE 15$
10280
10281 052072 026104 000032 CMP 32(R1),R4 ;CHECK THE FPS.
10282 052076 001034 BNE 20$ ;BRANCH IF INCORRECT.
10283
10284 052100 000161 000042 9$: JMP 42(R1) ;RETURN.
10285
10286 ;FRACTIONAL ERROR.
    
```

```

10287 052104 026112 000020      10$:  CMP      20(R1),(R2)      ;WAS THE ERROR ANTICIPATED?
10288 052110 001010                BNE      11$                ;BRANCH IF NOT ANTICIPATED.
10289 052112 026162 000022 000002      CMP      22(R1),2(R2)
10290 052120 001004                BNE      11$
10291 052122 010102                MOV      R1,R2              ;THE ERROR WAS ANTICIPATED SO
10292 052124 062702 000034                ADD      #34,R2            ;RETURN TO THE ERROR REPORT AT THE
10293                                ;CALLING ROUTINE.
10294 052130 000112                JMP      (R2)
10295
10296 052132                11$:
10297 052132 104002                *2$:  ERROR      2          ;THE ERROR WAS NOT ANTICIPATED SO
10298 052134 000761                BR        9$                ;REPORT THE INCORRECT FRACTION HERE.
10299
10300                                ;INTEGER ERROR.
10301 052136 026112 000024      15$:  CMP      24(R1),(R2)      ;WAS THIS ERROR ANTICIPATED?
10302 052142 001010                BNE      16$                ;BRANCH IF NOT.
10303 052144 026162 000026 000002      CMP      26(R1),2(R2)
10304 052152 001004                BNE      16$
10305 052154 010102                MOV      R1,R2              ;THE ERROR WAS ANTICIPATED SO RETURN
10306 052156 062702 000040                ADD      #40,R2            ;TO THE ERROR REPORT IN THE CALLING
10307                                ;ROUTINE.
10308 052162 000112                JMP      (R2)
10309
10310 052164                16$:
10311 052164 104002                17$:  ERROR      2          ;THE ERROR WAS NOT ANTICIPATED SO REPORT
10312 052166 000744                BR        9$                ;THE INTEGER FAILURE HERE.
10313
10314                                ;FPS INCORRECT.
10315 052170 010437 001254 001256      20$:  MOV      R4,@$TMP11      ;REPORT INCORRECT FPS.
10316 052174 016137 000032 001256      MOV      32(R1),@$TMP12
10317 052202 104002                21$:  ERROR      2
10318 052204 000735                BR        9$
10319
10320 052206 000000 000000 000000  MODFT0: .WORD 0,0,0,0
10321 052214 000000
10322
10323 052216 000000 000000 000000  MODFT1: .WORD 0,0,0,0
10324 052224 000000
10325
10326 052226 177777 177777 177777  MODP1: .WORD -1,-1,-1,-1
10327 052234 177777
10328
10329 052236                GGGDONE:
10330 052236 104413                RSFTUP                      ;GO INITIALIZE THE FPS AND STACK; AND
10331                                ;SEE IF THE USER HAS EXPRESSED
10332                                ;THE DESIRE TO CHANGE THE SOFTWARE
10333                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
10334                                ;THE USER TYPED CONTROL G?).
10335
10336
10337
10338
10339                                ;*****
10340                                ;*TEST 55          MODD TEST
10341                                ;*
10342                                ;*THIS IS A TEST OF THE MODD INSTRUCTION. IT MAKES USE OF A SUBROUTINE

```

```

10343      ;*TO SET UP THE ARGUMENTS, EXECUTE THE INSTRUCTION AND CHECK THE
10344      ;*RESULTS.
10345      ;*
10346      ;*****
10347 052240 000004      *ST55: SCOPE
10348
10349      ;MODD WITH (FSRC=AC=0)
10350      HHH1:
10351      LPERR
10352 052242 104414      JSR PC,@MODDSUB      ;SET UP THE LOOP ON ERROR ADDRESS.
10353 052244 004737 053746      .WORD 0,0,0,0      ;AC
10354 052250 000000 000000 000000 1$:      .WORD 0,0,0,0      ;FSRC
10355 052256 000000 000000 000000 2$:      .WORD 0,0,0,0      ;FRACTIONAL RES.
10356 052260 000000 000000 000000 3$:      .WORD 0,0,0,0      ;INTEGER RES.
10357 052266 000000 000000 000000 4$:      .WORD 0,0,0,0      ;ERROR FRACTIONAL RES.
10358 052270 000000 000000 000000 5$:      .WORD 0,0,0,0
10359 052276 000000 000000 000000 6$:      .WORD 0,0,-1,-1    ;ERROR INTEGER RES.
10360 052300 000000 000000 000000 7$:      200                ;FPS BEFORE EXECUTION.
10361 052306 000000 000000 000000 8$:      204                ;FPS AFTER EXECUTION.
10362 052310 000000 000000 000000 9$:      ERROR 2
10363 052316 000000 000000 177777      BR 9$
10364 052320 000000 000000 177777      ERROR 2            ;ST 231 TO 142 INTO 143
10365 052326 177777
10366 052330 000200
10367 052332 000204
10368 052334 104002
10369 052336 000401
10370 052340 104002
10371 052342
10372      ;MODD TEST WITH FSRC=0
10373      HHH2:
10374 052342 104414      LPERR
10375 052344 004737 053746      JSR PC,@MODDSUB      ;SET UP THE LOOP ON ERROR ADDRESS.
10376 052350 012345 067012      .WORD 012345,67012   ;AL
10377 052354 034567 012345      .WORD 34567,012345   ;FSRC
10378 052360 000000 000000 000000 2$:      .WORD 0,0,0,0
10379 052366 000000 000000 000000 3$:      .WORD 0,0,0,0      ;FRACTIONAL RES.
10380 052370 000000 000000 000000 4$:      .WORD 0,0,0,0      ;INTEGER RES.
10381 052376 000000 000000 000000 5$:      .WORD 012345,67012   ;ERROR FRACTIONAL RES.
10382 052400 000000 000000 000000 6$:      .WORD 34567,012345   ;ERROR INTEGER RES.
10383 052406 000000 000000 000000 7$:      .WORD -1,-1,-1,-1
10384 052410 012345 067012
10385 052414 034567 012345
10386 052420 177777 177777 177777      .WORD -1,-1,-1,-1
10387 052426 177777
10388 052430 000213
10389 052432 000204
10390 052434 104002
10391 052436 000401
10392 052440 104002
10393 052442
10394
10395      ;MODD TEST WITH (AC=0)
10396 052442      HHH3:
10397 052442 104414      LPERR
10398 052444 004737 053746      JSR PC,@MODDSUB      ;SET UP THE LOOP ON ERROR ADDRESS.

```

```

10399 052450 000000 000000 000000 1$: .WORD 0,0,0,0 ;AC
10400 052456 000000
10401 052460 072727 127272 2$: .WORD 72727,127272 ;FSRC
10402 052464 072727 127272 .WORD 72727,127272
10403 052470 000000 000000 000000 3$: .WORD 0,0,0,0 ;FRACTIONAL RES.
10404 052476 000000
10405 052500 000000 000000 000000 4$: .WORD 0,0,0,0 ;INTEGER RES.
10406 052506 000000
10407 052510 177777 177777 177777 5$: .WORD -1,-1,-1,-1 ;ERROR FRACTIONAL RES.
10408 052516 177777
10409 052520 177777 177777 177777 6$: .WORD -1,-1,-1,-1 ;ERROR INTEGER RES.
10410 052526 177777
10411 052530 000213 7$: 213 ;FPS BEFORE EXECUTION.
10412 052532 000204 204 ;FPS AFTER EXECUTION.
10413 052534 104002 8$: ERROR 2
10414 052536 000401 BR 9$
10415 052540 104002 ERROR 2
10416 052542 9$:
10417
10418 ;MODD TEST WITH EXPONENT OF THE RESULT = 57
10419 052542 HHH4:
10420 052542 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
10421 052544 004737 053746 JSR PC,@#MODDSUB
10422 052550 056252 125252 1$: .WORD 56252,125252 ;AC
10423 052554 125252 125250 .WORD 125252,125250
10424 052560 040300 000000 000000 2$: .WORD 40300,0,0,0 ;FSRC
10425 052566 000000
10426 052570 000000 000000 000000 3$: .WORD 0,0,0,0 ;FRACTIONAL RES.
10427 052576 000000
10428 052600 056377 177777 177777 4$: .WORD 56377,-1,-1,-4 ;INTEGER RES.
10429 052606 177774
10430 052610 000000 000000 5$: .WORD 0,0 ;ERROR FRACTIONAL RES.
10431 052614 125252 125252 .WORD 125252,125252
10432 052620 056377 177777 177777 6$: .WORD 56377,-1,-1,-1 ;ERROR INTEGER RES.
10433 052626 177777
10434 052630 000213 7$: 213 ;FPS BEFORE EXECUTION.
10435 052632 000204 204 ;FPS AFTER EXECUTION.
10436 052634 104002 8$: ERROR 2 ;ST 526 TO 134 INTO 135
10437 052636 000401 BR 9$
10438 052640 104002 ERROR 2
10439 052642 9$:
10440
10441 ;MODD TEST WITH EXPONENT OF THE RESULT - 79
10442 052642 HHH5:
10443 052642 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
10444 052644 004737 053746 JSR PC,@#MODDSUB
10445 052650 140240 000000 000000 1$: .WORD 140240,0,0,0 ;AC
10446 052656 000000
10447 052660 063714 146314 2$: .WORD 63714,146314 ;FSRC
10448 052664 133572 167737 .WORD 133572,167737
10449 052670 000000 000000 000000 3$: .WORD 0,0,0,0 ;FRACTIONAL RES.
10450 052676 000000
10451 052700 163777 177777 4$: .WORD 163777,-1 ;INTEGER RES.
10452 052704 162531 125726 .WORD 162531,125726
10453 052710 177777 177777 177777 5$: .WORD -1,-1,-1,-1 ;ERROR FRACTIONAL RES.
10454 052716 177777

```



```

10455 052720 063777 177777 6$: .WORD 63777,-1 ;ERROR INTEGER RES.
10456 052724 162531 125726 .WORD 162531,125726
10457 052730 000210 7$: 210 ;FPS BEFORE EXECUTION.
10458 052732 000204 204 ;FPS AFTER EXECUTION.
10459 052734 104002 8$: ERROR 2
10460 052736 000401 BR 9$
10461 052740 104002 ERROR 2 ;ST 526 BAD SIGN
10462 052742 9$:
10463
10464 ;MODD TEST WITH EXPONENT OF THE RESULT = 57
10465 052742 HHH6:
10466 052742 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
10467 052744 004737 053746 JSR PC,@#MODDSUB
10468 052750 056200 000000 000000 1$: .WORD 56200,0,0,1 ;AC
10469 052756 000001
10470 052760 040340 000000 000000 2$: .WORD 40340,0,0,0 ;FSRC
10471 052766 000000
10472 052770 000000 000000 000000 3$: .WORD 0,0,0,0 ;FRACTIONAL RES.
10473 052776 000000
10474 053000 056340 000000 000000 4$: .WORD 56340,0,0,1 ;INTEGER RES.
10475 053006 000001
10476 053010 040000 000000 000000 5$: .WORD 40000,0,0,0 ;ERROR FRACTIONAL RES.
10477 053016 000000
10478 053020 056340 000000 000000 6$: .WORD 56340,0,0,1 ;ERROR INTEGER RES.
10479 053026 000001
10480 053030 000213 7$: 213 ;FPS BEFORE EXECUTION.
10481 053032 000204 204 ;FPS AFTER EXECUTION.
10482 053034 104002 8$: ERROR 2 ;CONSTANT BAD (NOT 56)
10483 ;OR ST 525 TO 050 INTO 150
10484 053036 000401 BR 9$
10485 053040 104002 ERROR 2
10486 053042 9$:
10487
10488 ;MODD TEST WITH EXPONENT OF THE RESULT = 56
10489 053042 HHH7:
10490 053042 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
10491 053044 004737 053746 JSR PC,@#MODDSUB
10492 053050 056000 000000 000000 1$: .WORD 56000,0,0,1 ;AC
10493 053056 000001
10494 053060 040340 000000 000000 2$: .WORD 40340,0,0,0 ;FSRC
10495 053066 000000
10496 053070 040100 000000 000000 3$: .WORD 40100,0,0,0 ;FRACTIONAL RES.
10497 053076 000000
10498 053100 056140 000000 000000 4$: .WORD 56140,0,0,1 ;INTEGER RES.
10499 053106 000001
10500 053110 000000 000000 000000 5$: .WORD 0,0,0,0 ;ERROR FRACTIONAL RES.
10501 053116 000000
10502 053120 056140 000000 000000 6$: .WORD 56140,0,0,1 ;ERROR INTEGER RES.
10503 053126 000001
10504 053130 000213 7$: 213 ;FPS BEFORE EXECUTION.
10505 053132 000200 200 ;FPS AFTER EXECUTION.
10506 053134 104002 8$: ERROR 2 ;BAD CONSTANT (NOT 56) OR
10507 ;ST 525 TO 150 INTO 050
10508 053136 000401 BR 9$
10509 053140 104002 ERROR 2
10510 053142 9$:
    
```

```

10511
10512
10513 053142 ;MODD TEST WITH EXPONENT OF THE RESULT = 36
10514 053142 104414 HHH8:
10515 053144 004737 053746 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
10516 053150 051177 177777 177777 1$: JSR PC,@#MODDSUB
10517 053156 177777 .WORD 51177,-1,-1,-1 ;AC
10518 053160 040200 000000 000000 2$: .WORD 40200,0,0,0 ;FSRC
10519 053166 000000
10520 053170 040177 177760 000000 3$: .WORD 40177,-20,0,0 ;FRACTIONAL RES.
10521 053176 000000
10522 053200 051177 177777 177760 4$: .WORD 51177,-1,-20,0 ;INTEGER RES.
10523 053206 000000
10524 053210 177777 177777 177777 5$: .WORD -1,-1,-1,-1 ;ERROR FRACTIONAL RES.
10525 053216 177777
10526 053220 177777 177777 177777 6$: .WORD -1,-1,-1,-1 ;ERROR INTEGER RES.
10527 053226 177777
10528 053230 000217 7$: 217 ;FPS BEFORE EXECUTION.
10529 053232 000200 200 ;FPS AFTER EXECUTION.
10530 053234 104002 8$: ERROR 2
10531 053236 000401 BR 9$
10532 053240 104002 ERROR 2
10533 053242 9$:
10534
10535 ;MODD TEST WITH EXPONENT OF THE RESULT = 30
10536 053242 HHH9:
10537 053242 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
10538 053244 004737 053746 JSR PC,@#MODDSUB
10539 053250 040200 000000 000000 1$: .WORD 40200,0,0,0 ;AC
10540 053256 000000
10541 053260 047577 177777 2$: .WORD 47577,-1 ;FSRC
10542 053264 176000 000001 .WORD 176000,1
10543 053270 031600 000000 000000 3$: .WORD 31600,0,0,0 ;FRACTIONAL RES.
10544 053276 000000
10545 053300 047577 177777 4$: .WORD 47577,-1 ;INTEGER RES.
10546 053304 176000 000000 .WORD 176000,0
10547 053310 000000 000000 000000 5$: .WORD 0,0,0,0 ;ERROR FRACTIONAL RES.
10548 053316 000000
10549 053320 047577 177777 177777 6$: .WORD 47577,-1,-1,-1 ;ERROR INTEGER RES.
10550 053326 177777
10551 053330 000200 7$: 200 ;FPS BEFORE EXECUTION.
10552 053332 000200 200 ;FPS AFTER EXECUTION.
10553 053334 104002 8$: ERROR 2 ;(NORMALIZE) ST 532 TO 122
10554 ;INTO NORM.
10555 053336 000401 BR 9$
10556 053340 104002 ERROR 2 ;AC V 1 <= X14
10557 ;OR ST 733 TO 156 INTO 157.
10558 053342 9$:
10559
10560 ;MODD TEST WITH EXPONENT OF THE RESULT = 31
10561 053342 HHH10:
10562 053342 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
10563 053344 004737 053746 JSR PC,@#MODDSUB
10564 053350 047777 177777 177777 1$: .WORD 47777,-1 ;AC
10565 053354 177000 000000 .WORD 177000,0
10566 053360 040200 000000 000000 2$: .WORD 40200,0,0,0 ;FSRC
    
```

10567	053366	000000								
10568	053370	000000	000000	000000	3\$:	.WORD	0,0,0,0			;FRACTIONAL RES.
10569	053376	000000								
10570	053400	047777	177777		4\$:	.WORD	47777,-1			;INTEGER RES.
10571	053404	177000	000000			.WORD	177000,0			
10572	053410	000000	000000	177000	5\$:	.WORD	0,0,177000,0			;ERROR FRACTIONAL RES.
10573	053416	000000								
10574	053420	177777	177777	177777	6\$:	.WORD	-1,-1,-1,-1			;ERROR INTEGER RES.
10575	053426	177777								
10576	053430	000213			7\$:		213			;FPS BEFORE EXECUTION.
10577	053432	000204					204			;FPS AFTER EXECUTION.
10578	053434	104002			8\$:	ERROR	2			; (BUT FD) STORE X10
10579	053436	000401				BR	9\$			
10580	053440	104002				ERROR	2			
10581	053442				9\$:					
10582										
10583										
10584	053442									;MODD TEST WITH EXPONENT OF THE RESULT = 0
10585	053442	104414				HHH11:				
10586	053444	004737	053746			LPERR				;SET UP THE LOOP ON ERROR ADDRESS.
10587	053450	040200	000000	000000	1\$:	JSR	PC,@#MODDSUB			
10588	053456	000000				.WORD	40200,0,0,0			;AC
10589	053460	040177	072727		2\$:	.WORD	40177,72727			;FSRC
10590	053464	127272	072727			.WORD	127272,72727			
10591	053470	040177	072727		3\$:	.WORD	40177,72727			;FRACTIONAL RES.
10592	053474	127272	072727			.WORD	127272,72727			
10593	053500	000000	000000	000000	4\$:	.WORD	0,0,0,0			;INTEGER RES.
10594	053506	000000								
10595	053510	177777	177777	177777	5\$:	.WORD	-1,-1,-1,-1			;ERROR FRACTIONAL RES.
10596	053516	177777								
10597	053520	000000	000000	177777	6\$:	.WORD	0,0,-1,-1			;ERROR INTEGER RES.
10598	053526	177777								
10599	053530	000200			7\$:		200			;FPS BEFORE EXECUTION.
10600	053532	000200					200			;FPS AFTER EXECUTION.
10601	053534	104002			8\$:	ERROR	2			
10602	053536	000401				BR	9\$			
10603	053540	104002				ERROR	2			;ST 246 TO 126 INTO 127 (BUT FD)
10604	053542				9\$:					
10605										
10606										
10607	053542									;MODD TEST WITH EXPONENT OF THE RESULT = -115
10608	053542	104414				HHH12:				
10609	053544	004737	053746			LPERR				;SET UP THE LOOP ON ERROR ADDRESS.
10610	053550	003377	177777		1\$:	JSR	PC,@#MODDSUB			
10611	053554	177777	052525			.WORD	3377,-1			;AC
10612	053560	040200	000000	000000	2\$:	.WORD	-1,52525			
10613	053566	000000				.WORD	40200,0,0,0			;FSRC
10614	053570	003377	177777		3\$:	.WORD	3377,-1			;FRACTIONAL RES.
10615	053574	177777	052525			.WORD	-1,52525			
10616	053600	000000	000000	000000	4\$:	.WORD	0,0,0,0			;INTEGER RES.
10617	053606	000000								
10618	053610	177777	177777	177777	5\$:	.WORD	-1,-1,-1,-1			;ERROR FRACTIONAL RES.
10619	053616	177777								
10620	053620	000000	000000	177777	6\$:	.WORD	0,0,-1,-1			;ERROR INTEGER RES.
10621	053626	177777								
10622	053630	000200			7\$:		200			;FPS BEFORE EXECUTION.

```

10623 053632 000200                200                ;FPS AFTER EXECUTION.
10624 053634 104002                8$: ERROR 2
10625 053636 000401                BR 9$
10626 053640 104002                ERROR 2                ;ST 446 TO 126 INTO 127 (BUT FD)
10627 053642
10628
10629 ;MODD TEST WITH EXPONENT OF THE RESULT = -63, IN ROUND MODE.
10630 053642 HHH13:
10631 053642 104414                LPERR                ;SET UP THE LOOP ON ERROR ADDRESS.
10632 053644 004737 053746                JSR PC,@#MODDSUB
10633 053650 040300 000000 000000 1$: .WORD 40300,0,0,0 ;AC
10634 053656 000000
10635 053660 020200 000000 000000 2$: .WORD 20200,0,0,1 ;FSRC
10636 053666 000001
10637 053670 020300 000000 000000 3$: .WORD 20300,0,0,2 ;FRACTIONAL RES.
10638 053676 000002
10639 053700 000000 000000 000000 4$: .WORD 0,0,0,0 ;INTEGER RES.
10640 053706 000000
10641 053710 000000 000000 177777 5$: .WORD 0,0,-1,-1 ;ERROR FRACTIONAL RES.
10642 053716 177777
10643 053720 177777 177777 177777 6$: .WORD -1,-1,-1,-1 ;ERROR INTEGER RES.
10644 053726 177777
10645 053730 000200                7$: 200                ;FPS BEFORE EXECUTION.
10646 053732 000200                200                ;FPS AFTER EXECUTION.
10647 053734 104002                8$: ERROR 2                ;ST 127 INTO RND/TR
10648 053736 000401                BR 9$
10649 053740 104002                ERROR 2
10650 053742 000137 054344                9$: JMP @#HHHDONE ;GO TO THE NEXT TEST.
10651
10652 ;THIS SUBROUTINE, MODDSUB, IS CALLED TO SETUP THE
10653 ;OPERANDS, EXECUTE THE MODD INSTRUCTION AND CHECK THE RESULTS.
10654 ;IT IS CALLED THUS:
10655 :
10656 : ACARG: .WORD X,X,X,X ;AC OPERAND
10657 : FSRCARG: .WORD X,X,X,X ;FSRC OPERAND
10658 : FRES: .WORD X,X,X,X ;FRACTIONAL RESULT
10659 : INTRES: .WORD X,X,Y,X ;INTEGER RESULT
10660 : ERFRES: .WORD X,X,X,X ;ERROR FRACTION RESULT
10661 : ERINTRES: .WORD X,X,X,X ;ERROR INTEGER RESULT
10662 : FPSB: .WORD X ;FPS BEFORE EXECUTION
10663 : FPSA: .WORD X ;FPS AFTER EXECUTION
10664 : ERR1: ERROR X ;FRACTION ERROR
10665 : BR CONT
10666 : ERR2: ERROR X ;INTEGER ERROR
10667 : CONT: ;RETURN ADDRESS
10668
10669 ;THE OPERANDS ARE SET UP (USING ACO FOR THE AC ARGUMENT). THE MODD
10670 ;INSTRUCTION IS EXECUTED. THEN THE RESULTS ARE RETRIEVED.
10671 ;THE FRACTION PART OF THE RESULT IS COMPARED WITH FRES. IF THIS IS CORRECT
10672 ;THEN THE INTEGER PART IS COMPARED WITH INTRES. IF BOTH OF THESE ARE CORRECT
10673 ;THEN THE FPS IS COMPARED WITH FPSA. AFTER EXECUTION IF NO ERRORS OCCURRED
10674 ;THEN MODDSUB WILL RETURN TO CONT. IF THE FPS WAS INCORRECT
10675 ;IT IS REPORTED HERE. IF THE FRACTION IS INCORRECT IT IS COMPARED WITH
10676 ;THE ANTICIPATED BAD FRACTION, ERFRES. IF THIS DOESN'T MATCH
10677 ;THE TRUE RESULT THEN THE ERROR IS REPORTED HERE. IF THE ANTICIPATED
10678 ;FAILURE MATCHES THE TRUE RESULT THEN MODDSUB PASSES CONTROL TO THE
    
```

```

10679          ;ERROR CALL AT ERR1. LIKEWISE IF THE INTEGER PART OF THE RESULT IS
10680          ;NOT CORRECT THEN IT IS COMPARED WITH THE ANTICIPATED INTEGER
10681          ;FAILURE. IF THIS DOESN'T MATCH THEN THE ERROR IS REPORTED HERE.
10682          ;IF A MATCH IS MADE HOWEVER, MODDSUB WILL RETURN CONTROL TO THE ERROR
10683          ;CALL AT ERR2.
10684
10685 053746 012601          MODDSUB:      MOV      (SP)+,R1      ;GET A POINTER TO THE ARGUMENTS
10686 053750 012700 000200      MOV      #200,R0      ;SET FD MODE.
10687 053754 170100          LDFPS   R0
10688 053756 010100          MOV      R1,R0      ;SET UP ACO
10689 053760 172410          LDD     (R0),ACO
10690 053762 012700 052226      MOV      #MODPI,R0      ;PUT A BACKGROUND PATTERN INTO AC1.
10691 053766 172510          LDD     (R0),AC1
10692 053770 016100 000060      MOV      60(R1),R0      ;SET UP THE FPS.
10693 053774 170100          LDFPS   R0
10694 053776 012737 054012 001236      MOV      #1$,@#STMP2
10695 054004 010100          MOV      R1,R0      ;COMPUTE THE ADDRESS OF THE FSRC.
10696 054006 062700 000010          ADD     #10,R0
10697
10698 054012 171410          1$:      MODD     (R0),ACO      ;EXECUTE THE TEST INSTRUCTION.
10699
10700 054014 170204          STFPS   R4      ;GET THE FPS.
10701 054016 012700 000200      MOV      #200,R0      ;SET FD MODE.
10702 054022 170100          LDFPS   R0
10703 054024 012700 054324      MOV      #MODDT0,R0      ;GET THE FRACTIONAL RESULT.
10704 054030 174010          STD     ACO,(R0)
10705 054032 012700 054334      MOV      #MODDT1,R0      ;GET THE INTEGER RESULT.
10706 054036 174110          STD     AC1,(R0)
10707
10708 054040 010102          MOV      R1,R2      ;SAVE THE DATA IN CASE OF ERROR.
10709 054042 010237 001240      MOV      R2,@#STMP3
10710 054046 062702 000010          ADD     #10,R2
10711 054052 010237 001242      MOV      R2,@#STMP4
10712 054056 062702 000010          ADD     #10,R2
10713 054062 010237 001244      MOV      R2,@#STMP5
10714 054066 062702 000010          ADD     #10,R2
10715 054072 010237 001246      MOV      R2,@#STMP6
10716 054076 012737 054324 001250      MOV      #MODDT0,@#STMP7
10717 054104 012737 054334 001252      MOV      #MODDT1,@#STMP10
10718 054112 016137 000062 001256      MCV     62(R1),@#STMP12
10719 054120 010437 001254      MOV      R4,@#STMP11
10720
10721 054124 012702 054324          MOV      #MODDT0,R2      ;CHECK THE FRACTIONAL RESULT.
10722 054130 010103          MOV      R1,R3
10723 054132 062703 000020          ADD     #20,R3
10724 054136 012705 000004          MOV      #4,R5
10725 054142 022223          2$:      CMP      (R2)+,(R3)+
10726 054144 001020          BNE     10$      ;BRANCH IF INCORRECT.
10727 054146 077503          SOB     R5,2$
10728
10729 054150 012702 054334          MOV      #MODDT1,R2      ;CHECK THE INTEGER RESULT.
10730 054154 010103          MOV      R1,R3
10731 054156 062703 000030          ADD     #30,R3
10732 054162 012705 000004          MOV      #4,R5
10733 054166 022223          3$:      CMP      (R2)+,(R3)+
10734 054170 001026          BNE     15$      ;BRANCH IF INCORRECT.
    
```

```

10735 054172 077503          SOB      R5,3$
10736
10737
10738 054174 026104 000062          CMP      62(R1),R4      ;CHECK THE FPS.
10739 054200 001042          BNE      20$           ;BRANCH IF INCORRECT.
10740
10741 054202 000161 000072          9$:     JMP      72(R1)      ;RETURN.
10742
10743          ;FRACTIONAL ERROR.
10744 054206 012702 054324          10$:    MOV      #MODDT0,R2      ;WAS THE FRACTIONAL ERROR ANTICIPATED?
10745 054212 010103          MOV      R1,R3
10746 054214 062703 000040          ADD      #40,R3
10747 054220 012705 000004          MOV      #4,R5
10748 054224 022223          50$:    CMP      (R2)+,(R3)+
10749 054226 001005          BNE      11$           ;BRANCH IF NOT ANTICIPATED.
10750 054230 077503          SOB      R5,50$
10751 054232 010102          MOV      R1,R2
10752 054234 062702 000064          ADD      #64,R2
10753
10754 054240 000112          JMP      (R2)
10755
10756 054242          11$:
10757 054242 104002          12$:    ERROR   2           ;THE ERROR WAS NOT ANTICIPATED SO
10758 054244 000756          BR      9$           ;REPORT THE INCORRECT FRACTION HERE.
10759
10760          ;INTEGER ERROR.
10761 054246 012702 054334          15$:    MOV      #MODDT1,R2      ;WAS THE INTEGER ERROR ANTICIPATED?
10762 054252 010103          MOV      R1,R3
10763 054254 062703 000050          ADD      #50,R3
10764 054260 012705 000004          MOV      #4,R5
10765 054264 022223          60$:    CMP      (R2)+,(R3)+
10766 054266 001005          BNE      17$           ;BRANCH IF NOT ANTICIPATED.
10767 054270 077503          SOB      R5,60$
10768 054272 010102          MOV      R1,R2
10769 054274 062702 000070          ADD      #70,R2
10770
10771 054300 000112          JMP      (R2)
10772
10773 054302          16$:
10774 054302 104002          17$:    ERROR   2           ;THE ERROR WAS NOT ANTICIPATED SO REPORT
10775 054304 000736          BR      9$           ;THE INTEGER FAILURE HERE.
10776
10777          ;FPS INCORRECT.
10778 054306 010437 001254          20$:    MOV      R4,@#STMP11      ;REPORT INCORRECT FPS.
10779 054312 016137 000062 001256          MOV      62(R1),@#STMP12
10780 054320 104002          21$:    ERROR   2
10781 054322 000727          BR      9$
10782
10783 054324 000000 000000 000000 MODDT0: .WORD 0,0,0,0
10784 054332 000000
10785
10786 054334 000000 000000 000000 MODDT1: .WORD 0,0,0,0
10787 054342 000000
10788
10789 054344          HHHDONE:
10790 054344 104413          RSETUP           ;GO INITIALIZE THE FPS AND STACK; AND
    
```

10791  
 10792  
 10793  
 10794  
 10795  
 10796  
 10797  
 10798  
 10799  
 10800  
 10801  
 10802  
 10803  
 10804  
 10805  
 10806  
 10807 054346 000004  
 10808  
 10809  
 10810 054350  
 10811 054350 104414  
 10812 054352 004767 000320  
 10813 054356 020123 045676  
 10814 054362 020200 000000  
 10815 054366 000123 045676  
 10816 054372 000000 000000  
 10817 054376 177777 177777  
 10818 054402 177777 177777  
 10819 054406 042000  
 10820 054410 142004  
 10821 054412 000012  
 10822 054414 104002  
 10823 054416 000401  
 10824 054420 104002  
 10825 054422  
 10826  
 10827  
 10828 054422  
 10829 054422 104414  
 10830 054424 004737 054676  
 10831 054430 010200 000000  
 10832 054434 010000 000000  
 10833 054440 000000 000000  
 10834 054444 000000 000000  
 10835 054450 177777 177777  
 10836 054454 177777 177777  
 10837 054460 005013  
 10838 054462 005004  
 10839 054464 000012  
 10840 054466 000240  
 10841 054470 000401  
 10842 054472 104002  
 10843 054474  
 10844  
 10845  
 10846 054474

;SEE IF THE USER HAS EXPRESSED  
 ;THE DESIRE TO CHANGE THE SOFTWARE  
 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS  
 ;THE USER TYPED CONTROL G?).

\*\*\*\*\*  
 ;TEST 56 UNDER\OVER FLOW, USING MODF WITH TRAPS DISABLED, TEST  
 ;\*  
 ;\*THIS IS A TEST OF THE MODF OVERFLOW AND UNDERFLOW CONDITIONS. IT MAKES  
 ;\*USE OF A SUBROUTINE TO SETUP THE OPERANDS, EXECUTE THE MODF INSTRUCTION  
 ;\*AND CHECK THE RESULTS. TRAPS ARE DISABLED DURING THIS TEST.  
 ;\*

\*\*\*\*\*  
 TST56: SCOPE  
 ;UNDERFLOW TEST, WITH EXPONENT OF THE RESULT -129, FIU - 1, FID = 1  
 MMM1:

	LPERR		;SET UP THE LOOP ON ERROR ADDRESS.
	JSR	PC,MODFOV	
1\$:	.WORD	20123,45676	;AC
2\$:	.WORD	20200,0	;FSRC
3\$:	.WORD	123,45676	;FRACTIONAL RES.
4\$:	.WORD	0,0	;INTEGER RES.
5\$:	.WORD	-1,-1	;ERROR FRACTIONAL RES.
6\$:	.WORD	-1,-1	;ERROR INTEGER RES.
7\$:		42000	;FPS BEFORE EXECUTION.
		142004	;FPS AFTER EXECUTION.
		12	;FEC
8\$:	ERROR	2	;FEC INCORRECT, UNDERFLOW.
	BR	9\$	
	ERROR	2	;AC V 1 (2,3) <= ZERO, ST 126.
9\$:			

;UNDERFLOW EXP OF RESULT - -193, FIU = 0, FID = 1  
 MMM2:

	LPERR		;SET UP THE LOOP ON ERROR ADDRESS.
	JSR	PC,@MODFOV	
1\$:	.WORD	10200,0	;AC
2\$:	.WORD	10000,0	;FSRC
3\$:	.WORD	0,0	;FRACTIONAL RES.
4\$:	.WORD	0,0	;INTEGER RES.
5\$:	.WORD	-1,-1	;ERROR FRACTIONAL RES.
6\$:	.WORD	-1,-1	;ERROR INTEGER RES.
7\$:		5013	;FPS BEFORE EXECUTION.
		5004	;FPS AFTER EXECUTION.
		12	;FEC
8\$:	NOP		
	BR	9\$	
	ERROR	2	
9\$:			

;OVERFLOW TEST WITH EXPONENT OF THE RESULT = 128, FIV = 1, FID = 1  
 MMM3:

```

10847 054474 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
10848 054476 004737 054676    JSR          PC,@#MODFOV
10849 054502 060052 125252    1$: .WORD     60052,125252    ;AC
10850 054506 060200 000000    2$: .WORD     60200,0        ;FSRC
10851 054512 000000 000000    3$: .WORD     0,0           ;FRACTIONAL RES.
10852 054516 000052 125252    4$: .WORD     52,125252     ;INTEGER RES.
10853 054522 000000 000000    5$: .WORD     0,0           ;ERROR FRACTIONAL RES.
10854 054526 000000 000000    6$: .WORD     0,0           ;ERROR INTEGER RES.
10855 054532 041000          7$: 41000             ;FPS BEFORE EXECUTION.
10856 054534 141006          141006          ;FPS AFTER EXECUTION.
10857 054536 000010          10             ;FEC
10858 054540 104002    8$: ERROR      2          ;BAD FEC ON OVERFLOW.
10859 054542 000401          BR          9$
10860 054544 104002          ERROR      2          ;ST 520 TO STORE ZERO TWICE
10861
10862 054546    9$:
10863
10864 ;OVERFLOW TEST WITH EXPONENT OF THE RESULT = 130, FIV = 0, FID - 1
10865 054546 ;MMM4:
10866 054546 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
10867 054550 004737 054676    JSR          PC,@#MODFOV
10868 054554 060345 067654    1$: .WORD     60345,67654    ;AC
10869 054560 060200 000000    2$: .WORD     60200,0        ;FSRC
10870 054564 000000 000000    3$: .WORD     0,0           ;FRACTIONAL RES.
10871 054570 000000 000000    4$: .WORD     0,0           ;INTEGER RES.
10872 054574 000000 000000    5$: .WORD     0,0           ;ERROR FRACTIONAL RES.
10873 054600 000345 067654    6$: .WORD     345,67654     ;ERROR INTEGER RES.
10874 054604 006011          7$: 6011             ;FPS BEFORE EXECUTION.
10875 054606 006006          6006          ;FPS AFTER EXECUTION.
10876 054610 000010          10             ;FEC
10877 054612 000240    8$: NOP
10878 054614 000401          BR          9$
10879 054616 104002          ERROR      2          ;ST 520 TO 162 INTO STORE ZERO TWICE.
10880 054620
10881
10882 ;OVERFLOW TEST WITH EXPONENT OF THE RESULT = 128, RESULT NEGATIVE
10883 ;AND FIV = 1, FID - 1
10884 054620 ;MMM5:
10885 054620 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
10886 054622 004737 054676    JSR          PC,@#MODFOV
10887 054626 160252 125252    1$: .WORD     160252,125252  ;AC
10888 054632 060000 000000    2$: .WORD     60000,0        ;FSRC
10889 054636 000000 000000    3$: .WORD     0,0           ;FRACTIONAL RES.
10890 054642 100052 125252    4$: .WORD     100052,125252 ;INTEGER RES.
10891 054646 000000 000000    5$: .WORD     0,0           ;ERROR FRACTIONAL RES.
10892 054652 000052 125252    6$: .WORD     52,125252     ;ERROR INTEGER RES.
10893 054656 041000          7$: 41000             ;FPS BEFORE EXECUTION.
10894 054660 141006          141006          ;FPS AFTER EXECUTION.
10895 054662 000010          10             ;FEC
10896 054664 104002    8$: ERROR      2
10897 054666 000401          BR          9$
10898 054670 104002          ERROR      2          ;ST 517, BAD SIGN.
10899 054672 000137 055272    9$: JMP          @#MMMDONE    ;GO TO THE NEXT TEST.
10900
10901 ;THIS SUBROUTINE, MODFOV, IS CALLED TO SETUP THE
10902 ;OPERANDS, EXECUTE THE MODF INSTRUCTION AND CHECK THE RESULTS.
    
```



10903  
10904  
10905  
10906  
10907  
10908  
10909  
10910  
10911  
10912  
10913  
10914  
10915  
10916  
10917  
10918  
10919  
10920  
10921  
10922  
10923  
10924  
10925  
10926  
10927  
10928  
10929  
10930  
10931  
10932  
10933  
10934  
10935 054676 012601  
10936 054700 012700 000200  
10937 054704 170100  
10938 054706 010100  
10939 054710 172410  
10940 054712 012700 052226  
10941 054716 172510  
10942 054720 016100 000030  
10943 054724 170100  
10944 054726 012737 054742 001236  
10945 054734 010100  
10946 054736 062700 000004  
10947  
10948 054742 171410  
10949  
10950 054744 170204  
10951 054746 170305  
10952 054750 012700 000200  
10953 054754 170100  
10954 054756 012700 055252  
10955 054762 174010  
10956 054764 012700 055262  
10957 054770 174110  
10958

:IT IS CALLED THUS:

```

ACARG: .WORD X,X ;AC OPERAND
FSRCARG: .WORD X,X ;FSRC OPERAND
FRES: .WORD X,X ;FRACTIONAL RESULT
INTRES: .WORD X,X ;INTEGER RESULT
ERFRES: .WORD X,X ;ERROR FRACTION RESULT
ERINTRES: .WORD X,X ;ERROR INTEGER RESULT
FPSB: .WORD X ;FPS BEFORE EXECUTION
FPSA: .WORD X ;FPS AFTER EXECUTION
FEC: .WORD X ;FEC
ERR1: ERROR X ;FEC ERROR
      BR CONT
ERR2: ERROR X ;INTEGER ERROR
CONT: CONT ;RETURN ADDRESS

```

:THE OPERANDS ARE SET UP (USING ACO FOR THE AC ARGUMENT). THE MODF INSTRUCTION IS EXECUTED. THEN THE RESULTS ARE RETRIEVED. THE FRACTION PART OF THE RESULT IS COMPARED WITH FRES. IF THIS IS CORRECT THEN THE INTEGER PART IS COMPARED WITH INTRES. IF BOTH OF THESE ARE CORRECT THEN THE FPS IS COMPARED WITH FPSA. AFTER EXECUTION IF NO ERRORS OCCURRED THEN MODFOV WILL RETURN TO CONT. IF THE FPS WAS INCORRECT IT IS REPORTED HERE. IF THE FRACTION IS INCORRECT IT IS COMPARED WITH THE ANTICIPATED BAD FRACTION, ERFRES. IF THIS DOESN'T MATCH THE TRUE RESULT THEN THE ERROR IS REPORTED HERE. IF THE ANTICIPATED FAILURE MATCHES THE TRUE RESULT THEN MODFOV PASSES CONTROL TO THE ERROR CALL AT ERR1. LIKEWISE IF THE INTEGER PART OF THE RESULT IS NOT CORRECT THEN IT IS COMPARED WITH THE ANTICIPATED INTEGER FAILURE. IF THIS DOESN'T MATCH THEN THE ERROR IS REPORTED HERE. IF A MATCH IS MADE HOWEVER, MODFOV WILL RETURN CONTROL TO THE ERROR CALL AT ERR2.

```

MODFOV: MOV (SP)+,R1 ;GET A POINTER TO THE ARGUMENTS
        MOV #200,R0 ;SET FD MODE.
        LDFPS R0
        MOV R1,R0 ;SET UP ACO
        LDD (R0),ACO
        MOV #MODP1,R0 ;PUT A BACKGROUND PATTERN INTO AC1.
        LDD (R0),AC1
        MOV 30(R1),R0 ;SET UP THE FPS.
        LDFPS R0
        MOV #1$,@#STMP2
        MOV R1,R0 ;COMPUTE THE ADDRESS OF THE FSRC.
        ADD #4,R0

1$: MODF (R0),ACO ;EXECUTE THE TEST INSTRUCTION.

        STFPS R4 ;GET THE FPS.
        STST R5 ;GET FEC.
        MOV #200,R0 ;SET FD MODE.
        LDFPS R0
        MOV #MODFDO,R0 ;GET THE FRACTIONAL RESULT.
        STD ACO,(R0)
        MOV #MODFD1,R0 ;GET THE INTEGER RESULT.
        STD AC1,(R0)

```

```

10959 054772 010102          MOV      R1,R2          ;SAVE THE DATA IN CASE OF ERROR.
10960 054774 010237 001240    MCV      R2,@#STMP3
10961 055000 062702 000004    ADD      #4,R2
10962 055004 010237 001242    MOV      R2,@#STMP4
10963 055010 062702 000004    ADD      #4,R2
10964 055014 010237 001244    MOV      R2,@#STMP5
10965 055020 062702 000004    ADD      #4,R2
10966 055024 010237 001246    MOV      R2,@#STMP6
10967 055030 012737 055252 001250    MOV      #MODFD0,@#STMP7
10968 055036 012737 055262 001252    MOV      #MODFD1,@#STMP10
10969 055044 010437 001254    MOV      R4,@#STMP11
10970 055050 016137 000032 001256    MOV      32(R1),@#STMP12
10971 055056 010537 001260    MOV      R5,@#STMP13
10972 055062 016137 000034 001262    MOV      34(R1),@#STMP14
10973
10974 055070 012702 055252          MOV      #MODFD0,R2      ;CHECK THE FRACTIONAL RESULT.
10975 055074 026112 000010          CMP      10(R1),(R2)
10976 055100 001025          BNE      10$
10977 055102 026162 000012 000002    CMP      12(R1),2(R2)   ;BRANCH IF INCORRECT.
10978 055110 001021          BNE      10$
10979
10980 055112 012702 055262          MOV      #MODFD1,R2      ;CHECK THE INTEGER RESULT.
10981 055116 026112 000014          CMP      14(R1),(R2)
10982 055122 001016          BNE      15$
10983 055124 026162 000016 000002    CMP      16(R1),2(R2)   ;BRANCH IF INCORRECT.
10984 055132 001012          BNE      15$
10985
10986 055134 026104 000032          CMP      32(R1),R4      ;CHECK THE FPS.
10987 055140 001024          BNE      20$            ;BRANCH IF INCORRECT.
10988
10989 055142 026105 000034          CMP      34(R1),R5      ;CHECK THE FEC.
10990 055146 001030          BNE      25$            ;BRANCH IF INCORRECT.
10991
10992 055150 000161 000044          9$:     JMP      44(R1)      ;RETURN.
10993
10994          ;FRACTIONAL ERROR.
10995 055154          10$:
10996 055154 104002          12$:   ERROR 2          ;THE ERROR WAS NOT ANTICIPATED SO
10997 055156 000774          BR      9$            ;REPORT THE INCORRECT FRACTION HERE.
10998
10999          ;INTEGER ERROR.
11000 055160 026112 000024          15$:   CMP      24(R1),(R2)   ;WAS THIS ERROR ANTICIPATED?
11001 055164 001010          BNE      16$            ;BRANCH IF NOT.
11002 055166 026162 000026 000002    CMP      26(R1),2(R2)
11003 055174 001004          BNE      16$
11004 055176 010102          MOV      R1,R2          ;THE ERROR WAS ANTICIPATED SO RETURN
11005 055200 062702 000042          ADD      #42,R2         ;TO THE ERROR REPORT IN THE CALLING
11006                                     ;ROUTINE.
11007 055204 000112          JMP      (R2)
11008
11009 055206          16$:
11010 055206 104002          17$:   ERROR 2          ;THE ERROR WAS NOT ANTICIPATED SO REPORT
11011 055210 000757          BR      9$            ;THE INTEGER FAILURE HERE.
11012
11013          ;FPS INCORRECT.
11014 055212 010437 001254          20$:   MOV      R4,@#STMP11 ;REPORT INCORRECT FPS.

```

11015 055216 016137 000032 001256  
 11016 055224 104002  
 11017 055226 000750  
 11018  
 11019  
 11020 055230 010537 001260  
 11021 055234 016137 000034 001262  
 11022 055242 010102  
 11023 055244 062702 000036  
 11024 055250 000112  
 11025  
 11026 055252 000000 000000 000000  
 11027 055260 000000  
 11028  
 11029 055262 000000 000000 000000  
 11030 055270 000000  
 11031  
 11032 055272  
 11033 055272 104413  
 11034  
 11035  
 11036  
 11037  
 11038  
 11039  
 11040  
 11041  
 11042  
 11043  
 11044  
 11045  
 11046  
 11047  
 11048  
 11049  
 11050 055274 000004  
 11051  
 11052  
 11053 055276  
 11054 055276 104414  
 11055 055300 004737 055712  
 11056 055304 020252 125252  
 11057 055310 125252 125252  
 11058 055314 020100 000000 000000  
 11059 055322 000000  
 11060 055324 000177 177777 177777  
 11061 055332 177777  
 11062 055334 000000 000000 000000  
 11063 055342 000000  
 11064 055344 020252 125252  
 11065 055350 125252 125252  
 11066 055354 000000 000000 177777  
 11067 055362 177777  
 11068 055364 042200  
 11069 055366 142204  
 11070 055370 000012

21\$: MOV 32(R1),@#STMP12  
 ERROR 2  
 BR 9\$  
 :REPORT FEC ERROR.  
 25\$: MOV R5,@#STMP13  
 MOV 34(R1),@#STMP14  
 MOV R1,R2  
 ADD #36,R2  
 JMP (R2)

MODFDO: .WORD 0,0,0,0  
 MODFD1: .WORD 0,0,0,0

MMMDONE:  
 RSETUP

:GO INITIALIZE THE FPS AND STACK; AND  
 :SEE IF THE USER HAS EXPRESSED  
 :THE DESIRE TO CHANGE THE SOFTWARE  
 :VIRTUAL CONSOLE SWITCH REGISTER (HAS  
 :THE USER TYPED CONTROL G?).

::\*\*\*\*\*  
 :\*TEST 57 UNDER\OVER FLOW, USING MODD WITH TRAPS DISABLED, TEST  
 :\*  
 :\*THIS IS A TEST OF THE MODD INSTRUCTION'S OVER FLOW AND UNDER FLOW  
 :\*CONDITIONS. A SUBROUTINE IS USED TO SET UP THE OPERANDS, EXECUTE THE  
 :\*MODD INSTRUCTION AND CHECK THE RESULTS.  
 :\*  
 :\*\*\*\*\*

TST57: SCOPE

:UNDERFLOW TEST WITH EXPONENT OF THE RESULT = -129, FIU = 1, FID = 1  
 NNN1:

LPERR ;SET UP THE LOOP ON ERROR ADDRESS.  
 JSR PC,@#MODDOV  
 1\$: .WORD 20252,125252 ;AC  
 .WORD 125252,125252  
 2\$: .WORD 20100,0,0,0 ;FSRC  
 3\$: .WORD 177,-1,-1,-1 ;FRACTIONAL RES.  
 4\$: .WORD 0,0,0,0 ;INTEGER RES.  
 5\$: .WORD 20252,125252 ;ERROR FRACTIONAL RES.  
 .WORD 125252,125252  
 6\$: .WORD 0,0,-1,-1 ;ERROR INTEGER RES.  
 7\$: 42200 ;FPS BEFORE EXECUTION.  
 142204 ;FPS AFTER EXECUTION.  
 12 ;FEC

```

11071 055372 104002      8$:      ERROR      2      ;FEC INCORRECT ON UNDERFLOW.
11072 055374 000401      BR          9$
11073 055376 104002      ERROR      2      ;ST 155 (BUT FD)
11074 055400
11075
11076      ;UNDERFLOW TEST WITH EXPONENT OF THE RESULT - -193, FIJ = 0, FID = 1
11077 055400      NNN2:
11078 055400 104414      LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.
11079 055402 004737 055712      JSR      PC, @MODDOV
11080 055406 010000 000000      1$:      .WORD    10000,0      ;AC
11081 055412 123456 000000      .WORD    123456,0
11082 055416 010200 000000 000000      2$:      .WORD    10200,0,0,0      ;FSRC
11083 055424 000000
11084 055426 000000 000000 000000      3$:      .WORD    0,0,0,0      ;FRACTIONAL RES.
11085 055434 000000
11086 055436 000000 000000 000000      4$:      .WORD    0,0,0,0      ;INTEGER RES.
11087 055444 000000
11088 055446 000000 000000 000000      5$:      .WORD    0,0,0,0      ;ERROR FRACTIONAL RES.
11089 055454 000000
11090 055456 000000 000000      6$:      .WORD    0,0      ;ERROR INTEGER RES.
11091 055462 123456 000000      .WORD    123456,0
11092 055466 005213      7$:      5213      ;FPS BEFORE EXECUTION.
11093 055470 005204      5204      ;FPS AFTER EXECUTION.
11094 055472 000012
11095 055474 000240      8$:      NOP
11096 055476 000401      BR          9$
11097 055500 104002      ERROR      2      ;ST 047 (BUT FD).
11098 055502
11099
11100      ;OVERFLOW TEST WITH EXPONENT OF THE RESULT = 128, FIV - 1, FID = 1
11101 055502      NNN3:
11102 055502 104414      LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.
11103 055504 004737 055712      JSR      PC, @MODDOV
11104 055510 060252 125252      1$:      .WORD    60252,125252      ;AC
11105 055514 125252 125252      .WORD    125252,125252
11106 055520 060100 000000 000000      2$:      .WORD    60100,0,0,0      ;FSRC
11107 055526 000000
11108 055530 000000 000000 000000      3$:      .WORD    0,0,0,0      ;FRACTIONAL RES.
11109 055536 000000
11110 055540 000177 177777 177777      4$:      .WORD    177,-1,-1,-1      ;INTEGER RES.
11111 055546 177777
11112 055550 000000 000000 000000      5$:      .WORD    0,0,0,0      ;ERROR FRACTIONAL RES.
11113 055556 000000
11114 055560 000177 177777      6$:      .WORD    177,-1      ;ERROR INTEGER RES.
11115 055564 125252 125252      .WORD    125252,125252
11116 055570 041200      7$:      41200      ;FPS BEFORE EXECUTION.
11117 055572 141206      141206      ;FPS AFTER EXECUTION.
11118 055574 000010      10      ;FEC
11119 055576 104002      8$:      ERROR      2      ;FEC BAD ON OVERFLOW.
11120 055600 000401      BR          9$
11121 055602 104002      ERROR      2
11122 055604      9$:
11123
11124      ;OVERFLOW TEST WITH EXPONENT OF THE RESULT = 130, FIV = 0, FID = 1
11125 055604      NNN4:
11126 055604 104414      LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.

```

```

11127 055606 004737 055712 JSR PC,@#MODDOV
11128 055612 060200 000000 1$: .WORD 60200,0 ;AC
11129 055616 125252 000000 .WORD 125252,0
11130 055622 060200 000000 000000 2$: .WORD 60200,0,0,0 ;FSRC
11131 055630 000000
11132 055632 000000 000000 000000 3$: .WORD 0,0,0,0 ;FRACTIONAL RES.
11133 055640 000000
11134 055642 000000 000000 000000 4$: .WORD 0,0,0,0 ;INTEGER RES.
11135 055650 000000
11136 055652 000000 000000 000000 5$: .WORD 0,0,0,0 ;ERROR FRACTIONAL RES.
11137 055660 000000
11138 055662 000400 000000 6$: .WORD 400,0 ;ERROR INTEGER RES.
11139 055666 125252 000000 .WORD 125252,0
11140 055672 006211 7$: 6211 ;FPS BEFORE EXECUTION.
11141 055674 006206 6206 ;FPS AFTER EXECUTION.
11142 055676 000010 10 ;FEC
11143 055700 000240 8$: NOP
11144 055702 000401 BR 9$
11145 055704 104002 ERROR 2 ;ST 520 TO 162 INTO STORE ZERO TWICE.
11146 055706 000137 056320 9$: JMP @#NNNDONE ;GO TO NEXT TEST.

```

```

;THIS SUBROUTINE, MODDOV, IS CALLED TO SETUP THE
;OPERANDS, EXECUTE THE MODD INSTRUCTION AND CHECK THE RESULTS.
;IT IS CALLED THUS:

```

```

:
: ACARG: .WORD X,X,X,X ;AC OPERAND
: FSRCARG: .WORD X,X,X,X ;FSRC OPERAND
: FRES: .WORD X,X,X,X ;FRACTIONAL RESULT
: INTRES: .WORD X,X,X,X ;INTEGER RESULT
: ERFRES: .WORD X,X,X,X ;ERROR FRACTION RESULT
: ERINTRES: .WORD X,X,X,X ;ERROR INTEGER RESULT
: FPSB: .WORD X ;FPS BEFORE EXECUTION
: FPSA: .WORD X ;FPS AFTER EXECUTION
: ERR1: ERROR X ;FRACTION ERROR
: BR CONT
: ERR2: ERROR X ;INTEGER FRROR
: CONT: ;RETURN ADDRESS

```

```

;THE OPERANDS ARE SET UP (USING ACO FOR THE AC ARGUMENT). THE MODD
;INSTRUCTION IS EXECUTED. THEN THE RESULTS ARE RETRIEVED.
;THE FRACTION PART OF THE RESULT IS COMPARED WITH FRES. IF THIS IS CORRECT
;THEN THE INTEGER PART IS COMPARED WITH INTRES. IF BOTH OF THESE ARE CORRECT
;THEN THE FPS IS COMPARED WITH FPSA. AFTER EXECUTION IF NO ERRORS OCCURRED
;THEN MODDOV WILL RETURN TO CONT. IF THE FPS WAS INCORRECT
;IT IS REPORTED HERE. IF THE FRACTION IS INCORRECT IT IS COMPARED WITH
;THE ANTICIPATED BAD FRACTION, ERFRES. IF THIS DOESN'T MATCH
;THE TRUE RESULT THEN THE ERROR IS REPORTED HERE. IF THE ANTICIPATED
;FAILURE MATCHES THE TRUE RESULT THEN MODDOV PASSES CONTROL TO THE
;ERROR CALL AT ERR1. LIKewise IF THE INTEGER PART OF THE RESULT IS
;NOT CORRECT THEN IT IS COMPARED WITH THE ANTICIPATED INTEGER
;FAILURE. IF THIS DOESN'T MATCH THEN THE ERROR IS REPORTED HERE.
;IF A MATCH IS MADE HOWEVER, MODDOV WILL RETURN CONTROL TO THE ERROR
;CALL AT ERR2.

```

```

11180
11181 055712 012601 MODDOV: MOV (SP)+,R1 ;GET A POINTER TO THE ARGUMENTS
11182 055714 012700 000200 MOV #200,R0 ;SET FD MODE.

```

11183	055720	170100		LDFPS	R0	
11184	055722	010100		MOV	R1,R0	;SET UP ACO
11185	055724	172410		LDD	(R0),ACO	
11186	055726	012700	052226	MOV	#MODP1,R0	;PUT A BACKGROUND PATTERN INTO AC1.
11187	055732	172510		LDD	(R0),AC1	
11188	055734	016100	000060	MOV	60(R1),R0	;SET UP THE FPS.
11189	055740	170100		LDFPS	R0	
11190	055742	012737	055756 001236	MOV	#1\$,@#STMP2	
11191	055750	010100		MOV	R1,R0	;COMPUTE THE ADDRESS OF THE FSRC.
11192	055752	062700	000010	ADD	#10,R0	
11193						
11194	055756	171410		1\$: MODD	(R0),ACO	;EXECUTE THE TEST INSTPUCTION.
11195						
11196	055760	170305		STST	R5	;GET THE FPS.
11197	055762	170204		STFPS	R4	;GET THE FPS.
11198	055764	012700	000200	MOV	#200,R0	;SET FD MODE.
11199	055770	170100		LDFPS	R0	
11200	055772	012700	056300	MOV	#MODDD0,R0	;GET THE FRACTIONAL RESULT.
11201	055776	174010		STD	ACO,(R0)	
11202	056000	012700	056310	MOV	#MODDD1,R0	;GET THE INTEGER RESULT.
11203	056004	174110		STD	AC1,(R0)	
11204						
11205	056006	010102		MOV	R1,R2	;SAVE THE DATA IN CASE OF ERROR.
11206	056010	010237	001240	MOV	R2,@#STMP3	
11207	056014	062702	000010	ADD	#10,R2	
11208	056020	010237	001242	MOV	R2,@#STMP4	
11209	056024	062702	000010	ADD	#10,R2	
11210	056030	010237	001244	MOV	R2,@#STMP5	
11211	056034	062702	000010	ADD	#10,R2	
11212	056040	010237	001246	MOV	R2,@#STMP6	
11213	056044	012737	056300 001250	MOV	#MODDD0,@#STMP7	
11214	056052	012737	056310 001252	MOV	#MODDD1,@#STMP10	
11215	056060	010437	001254	MOV	R4,@#STMP11	
11216	056064	016137	000062 001256	MOV	62(R1),@#STMP12	
11217	056072	010537	001260	MOV	R5,@#STMP13	
11218	056076	016137	000064 001262	MOV	64(R1),@#STMP14	
11219						
11220	056104	012702	056300	MOV	#MODDD0,R2	;CHECK THE FRACTIONAL RESULT.
11221	056110	010103		MOV	R1,R3	
11222	056112	062703	000020	ADD	#20,R3	
11223	056116	012700	000004	MOV	#4,R0	
11224	056122	022223		2\$: CMP	(R2)+,(R3)+	
11225	056124	001023		BNE	10\$	;BRANCH IF INCORRECT.
11226	056126	077003		SOB	R0,2\$	
11227						
11228	056130	012702	056310	MOV	#MODDD1,R2	;CHECK THE INTEGER RESULT.
11229	056134	010103		MOV	R1,R3	
11230	056136	062703	000030	ADD	#30,R3	
11231	056142	012700	000004	MOV	#4,R0	
11232	056146	022223		3\$: CMP	(R2)+,(R3)+	
11233	056150	001013		BNE	15\$	;BRANCH IF INCORRECT.
11234	056152	077003		SOB	R0,3\$	
11235						
11236						
11237	056154	026104	000062	CMP	62(R1),R4	;CHECK THE FPS.
11238	056160	001027		BNE	20\$	;BRANCH IF INCORRECT.

```

11239
11240 056162 026105 000064          CMP    64(R1),R5      :CHECK THE FEC.
11241 056166 001033          BNE    25$
11242
11243 056170 000161 000074          9$:    JMP    74(R1)      :RETURN.
11244
11245          :FRACTIONAL ERROR.
11246 056174          10$:
11247 056174 104002          12$:    ERROR    2      :THE ERROR WAS NOT ANTICIPATED SO
11248 056176 000774          BR     9$             :REPORT THE INCORRECT FRACTION HERE.
11249
11250          :INTEGER ERROR.
11251 056200 012702 056310          15$:    MOV    #MODDD1,R2      :WAS THE INTEGER ERROR ANTICIPATED?
11252 056204 010103          MOV    R1,R3
11253 056206 062703 000050          ADD    #50,R3
11254 056212 012705 000004          MOV    #4,R5
11255 056216 022223          50$:    CMP    (R2)+,(R3)+
11256 056220 001005          BNE    17$             :BRANCH IF NOT ANTICIPATED.
11257 056222 077503          SOB    R5,60$
11258 056224 010102          MOV    R1,R2          :THE ERROR WAS ANTICIPATED SO RETURN
11259 056226 062702 000072          ADD    #72,R2        :TO THE ERROR REPORT IN THE CALLING
11260          :ROUTINE.
11261 056232 000112          JMP    (R2)
11262
11263 056234          16$:
11264 056234 104002          17$:    ERROR    2      :THE ERROR WAS NOT ANTICIPATED SO REPORT
11265 056236 000754          BR     9$             :THE INTEGER FAILURE HERE.
11266
11267          :FPS INCORRECT.
11268 056240 010437 001254          20$:    MOV    R4,@#STMP11      :REPORT INCORRECT FPS.
11269 056244 016137 000062 001256          MOV    62(R1),@#STMP12
11270 056252 104002          21$:    ERROR    2
11271 056254 000745          BR     9$
11272
11273          :REPORT FEC ERROR.
11274 056256 010537 001260          25$:    MOV    R5,@#STMP13
11275 056262 016137 000064 001262          MOV    64(R1),@#STMP14
11276 056270 010102          MOV    R1,R2
11277 056272 062702 000066          ADD    #66,R2
11278 056276 000112          JMP    (R2)
11279
11280 056300 000000 000000 000000 MODDD0: .WORD 0,0,0,0
11281 056306 000000
11282
11283 056310 000000 000000 000000 MODDD1: .WORD 0,0,0,0
11284 056316 000000
11285
11286 056320          NNNDONE:
11287 056320 104413          RSETUP              :GO INITIALIZE THE FPS AND STACK; AND
11288          :SEE IF THE USER HAS EXPRESSED
11289          :THE DESIRE TO CHANGE THE SOFTWARE
11290          :VIRTUAL CONSOLE SWITCH REGISTER (HAS
11291          :THE USER TYPED CONTROL G?).
11292
11293
11294          ;:*****
    
```

```

11295 ;*TEST 60 MORE MICROCODES COVERAGE
11296 ;*THIS TEST WILL PROVIDE ADDITIONAL MICRO-CODE LOCATIONS COVERAGE
11297 ;*IN FPP1 AND FPP2.
11298 ;*
11299 ;*****
11300 056322 000004 TST60: SCOPE
11301 056324 XX1:
11302 056324 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
11303 056326 012737 056342 000244 XT1: MOV #XT1A,@#244
11304 056334 170227 000000 STFPS #0
11305 056340 000401 BR XT2
11306 056342 104001 XT1A: ERROR 1 ;SHOULD NOT TRAP
11307
11308 056344 012700 177777 XT2: MOV #-1,R0
11309 056350 170127 000000 LDFPS #0
11310 056354 170200 STFPS R0
11311 056356 005700 TST R0
11312 056360 001401 BEQ XT2A
11313 056362 104001 ERROR 1 ;FPS IS NOT ZEROED
11314 056364 012700 057124 XT2A: MOV #XPAT0,R0
11315 056370 172440 LDF -(R0),ACO
11316 056372 022700 057120 CMP #XPAT0-4,R0
11317 056376 001401 BEQ XT2B
11318 056400 104001 ERROR 1 ;R0 WAS NOT DECR BY 4
11319 056402 170200 XT2B: STFPS R0
11320 056404 022700 000004 CMP #4,R0 ;CHECK IF FZ IS SET?
11321 056410 001401 BEQ XT3
11322 056412 104001 ERROR 1
11323
11324 056414 170127 000000 XT3: LDFPS #0
11325 056420 012700 057124 MOV #XPAT0,R0
11326 056424 174040 STF ACO,-(R0)
11327 056426 022700 057120 CMP #XPAT0-4,R0
11328 056432 001401 BEQ XT3A ;R0 WAS NOT DECR BY 4
11329 056434 104001 ERROR 1
11330 056436 170200 XT3A: STFPS R0
11331 056440 005700 TST R0
11332 056442 001401 BEQ XT4
11333 056444 104001 ERROR 1
11334
11335 056446 170127 000000 XT4: LDFPS #0
11336 056452 012737 056476 000244 MOV #XT4A,@#244
11337 056460 170127 004000 LDFPS #04000 ;INTRPT ON UNDEFINED VARIABLE
11338 056464 172437 057124 LDF @#XPAT0,ACO
11339 056470 174437 057154 DIVF @#XPAT3,ACO ;GET UNDEFINED VARIABLE, 0
11340 056474 104001 ERROR 1 ;MISSING INTERRUPT TO 244
11341 056476 170200 XT4A: STFPS R0
11342 056500 022700 104004 CMP #104004,R0 ;CHECK: FER,FIUV,FZ ARE SET?
11343 056504 001401 BEQ XT4B
11344 056506 104001 ERROR 1
11345 056510 012700 057114 XT4B: MOV #XBUF,R0
11346 056514 174010 STF ACO,(R0)
11347 056516 005737 057114 TST @#XBUF
11348 056522 001401 BEQ XT5
11349 056524 104001 ERROR 1 ;ACO SHOULD REMAIN ZEROED
11350
    
```



```

11351 056526 012737 056546 000244 XT5:  MOV    #XT5A,@#244
11352 056534 170127 004000          LDFPS  #04000          ;INTRPT ON UNDEFINED VARIBALE
11353 056540 177437 057154          LDCDF  @#XPAT3,ACO    ;GET UNDEFINED VARIABLE, _0
11354 056544 104001          ERROR  1              ;MISSING INTERRUPT TO 244
11355 056546 170200          XT5A:  STFPS  R0
11356 056550 022700 104014          CMP    #104014,R0    ;CHECK: FER,FIUV,FN,FZ ARE SET?
11357 056554 001401          BEQ    XT5B
11358 056556 104001          ERROR  1
11359 056560 012700 057114          XT5B:  MOV    #XBUF,R0
11360 056564 174010          STF    ACO,(R0)
11361 056566 005737 057114          TST   @#XBUF
11362 056572 001401          BEQ    XT6
11363 056574 104001          ERROR  1              ;ACO SHOULD BE ZEROED
11364
11365 056576          XT6:
11366 056576 104414          LPERR
11367 056600 012737 056624 000244          MOV    #XT6A,@#244    ;SET UP THE LOOP ON ERROR ADDRESS.
11368 056606 170127 004000          LDFPS  #04000          ;INTRPT ON UNDEFINED VARIBALE
11369 056612 172437 057124          LDF    @#XPAT0,ACO
11370 056616 172037 057154          ADDF  @#XPAT3,ACO
11371 056622 104002          ERROR  2              ;MISSING INTERRUPT TO 244
11372 056624 170200          XT6A:  STFPS  R0
11373 056626 022700 104004          CMP    #104004,R0    ;CHECK: FER,FIUV,FZ ARE SET?
11374 056632 001401          BEQ    XT6B
11375 056634 104002          ERROR  2
11376 056636 012700 057114          XT6B:  MOV    #XBUF,R0
11377 056642 174010          STF    ACO,(R0)
11378 056644 005737 057114          TST   @#XBUF
11379 056650 001401          BEQ    XT7
11380 056652 104002          ERROR  2              ;ACO SHOULD BE ZEROED
11381
11382 056654 170127 000000          XT7:  LDFPS  #0
11383 056660 172437 057164          LDF    @#XPAT4,ACO
11384 056664 175437 057214          STCFI ACO,@#XPAT0
11385 056670 022737 000002 057214          CMP    #2,@#XPAT0    ;CHECK DATA
11386 056676 001401          BEQ    XT8
11387 056700 104001          ERROR  1
11388
11389 056702 170127 000100          XT8:  LDFPS  #100          ;SET FL
11390 056706 172437 057164          LDF    @#XPAT4,ACO
11391 056712 175467 000276          STCFI ACO,XPAT0
11392 056716 022737 000002 057216          CMP    #2,@#XPAT0+2
11393 056724 001401          BEQ    XT9
11394 056726 104001          ERROR  1
11395
11396          ;START OF FPP2
11397 056730 170127 000000          XT9:  LDFPS  #0
11398 056734 172437 057124          LDF    @#XPAT0,ACO
11399 056740 172037 057164          ADDF  @#XPAT4,ACO
11400 056744 170200          STFPS  R0
11401 056746 005700          TST   R0
11402 056750 001401          BEQ    XT10
11403 056752 104002          ERROR  2
11404
11405 056754 170127 000000          XT10: LDFPS  #0
11406 056760 172437 057164          LDF    @#XPAT4,ACO
    
```

11407	056764	173037	057164			SUBF	@XPAT4,ACO	
11408	056770	170200				STFPS	R0	
11409	056772	022700	000004			CMP	#4,R0	
11410	056776	001401				BEQ	XT11	
11411	057000	104002				ERROR	2	
11412								
11413	057002	170127	000000		XT11:	LDFPS	#0	
11414	057006	172437	057164			LDF	@XPAT4,ACO	
11415	057012	173437	057164			CMPF	@XPAT4,ACO	
11416	057016	170200				STFPS	R0	
11417	057020	022700	000004			CMP	#4,R0	;CHECK IF FZ IS SET?
11418	057024	001401				BEQ	XT12	
11419	057026	104002				ERROR	2	
11420								
11421	057030	170127	000000		XT12:	LDFPS	#0	
11422	057034	172437	057164			LDF	@XPAT4,ACO	
11423	057040	174437	057144			DIVF	@XPAT2,ACO	
11424	057044	012700	057114			MOV	#XBUF,R0	
11425	057050	174010				STF	ACO,(R0)	
11426	057052	022737	040176	057114		CMP	#040176,@XBUF	;CHECK DATA
11427	057060	001401				BEQ	XT13	
11428	057062	104002				ERROR	2	
11429								
11430	057064	170127	000000		XT13:	LDFPS	#0	
11431	057070	172437	057174			LDF	@XPAT5,ACO	
11432	057074	174437	057204			DIVF	@XPAT6,ACO	
11433	057100	170200				STFPS	R0	
11434	057102	022700	000004			CMP	#4,R0	
11435	057106	001401				BEQ	XT13B	
11436	057110	104002				ERROR	2	
11437	057112	000444			XT13B:	BR	XNEXT	
11438								
11439								
11440	057114	000000	000000	000000	XBUF:	.WORD	0,0,0,0	
11441	057122	000000						
11442	057124	000000	000000	000000	XPAT0:	.WORD	0,0,0,0	
11443	057132	000000						
11444	057134	000001	000001	000001	XPAT1:	.WORD	1,1,1,1	
11445	057142	000001						
11446	057144	040401	000000	000000	XPAT2:	.WC	40401,0,0,0	
11447	057152	000000						
11448	057154	100000	000000	000000	XPAT3:	.WORD	100000,0,0,0	
11449	057162	000000						
11450	057164	040400	000000	000000	XPAT4:	.WORD	040400,0,0,0	
11451	057172	000000						
11452	057174	000207	000000	000000	XPAT5:	.WORD	207,0,0,0	
11453	057202	000000						
11454	057204	077007	000000	000000	XPAT6:	.WORD	77007,0,0,0	
11455	057212	000000						
11456	057214	000000	000000	000000	XPAT0:	.WORD	0,0,0,0	
11457	057222	000000						
11458								
11459	057224				XNEXT:			
11460								
11461	057224	012706	001100			MOV	#STACK,SP	;SET UP STACK POINTER
11462	057230	012737	057264	000004		MOV	#1\$,@#4	;SET UP FOR TIMEOUT IF NO MULTI-TESTER

```

11463 057236 012737 000340 000006      MOV    #340,@#6
11464 057244 012737 000001 164000      MOV    #1,@#164000      ;SET 'STOP' ON MULTI-TESTER
11465 057252 012737 000006 000004      MOV    #6,@#4           ;RESTORE TRAP CATCHER
11466 057260 005037 000006
11467 057264
11468
11469
11470 057264      TST61:
11471
11472
11473
11474      .SBTTL  END OF PASS ROUTINE
11475
11476      ;*****
11477      ;*INCREMENT THE PASS NUMBER ($PASS)
1478      ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
11479      ;*TYPE 'END PASS #XXXXX' (WHERE XXXXX IS A DECIMAL NUMBER)
11480      ;*IF SW12=1 INHIBIT TRACE TRAP
11481      ;*IF THERES A MONITOR GO TO IT
11482      ;*IF THERE ISN'T JUMP TO LOOP
11483
11484 057264      $EOP:
11485 057264 000004      SCOPE
11486 057266 005067 121610      CLR    $STNM           ;;ZERO THE TEST NUMBER
11487 057272 005067 122004      CLR    $TIMES         ;;ZERO THE NUMBER OF ITERATIONS
11488 057276 005267 122022      INC    $PASS          ;;INCREMENT THE PASS NUMBER
11489 057302 042767 100000 122014      BIC    #100000,$PASS  ;;DON'T ALLOW A NEG. NUMBER
11490 057310 005327      DEC    (PC)+         ;;LOOP?
11491 057312 000001      $EOPCT: .WORD 1
11492 057314 003031      BGT    $DOAGN        ;;YES
11493 057316 012737      MOV    (PC)+,@(PC)+  ;;RESTORE COUNTER
11494 057320 000001      $ENDCT: .WORD 1
11495 057322 057312      $EOPCT
11496 057324 104401 057447      TYPE  , $ENDMG       ;;TYPE 'END PASS #'
11497 057330 016746 121770      MOV    $PASS,-(SP)   ;;SAVE $PASS FOR TYPEOUT
11498 057334 104405      TYPDS  ;;GO TYPE--DECIMAL ASCII WITH SIGN
11499 057336 104401 057444      TYPE  , $ENULL       ;;TYPE A NULL CHARACTER
11500 057342 013700 000042      $GET42: MOV    @#42,R0  ;;GET MONITOR ADDRESS
11501 057346 001414      BEQ    $DOAGN        ;;BRANCH IF NO MONITOR
11502 057350 005046      CLR    -(SP)         ;;INSURE THE 'T' BIT IS CLEAR
11503 057352 012746 057360      MOV    # $CLR.I,-(SP) ;;SETUP FOR AN RTI OR RTT
11504 057356 000426      BR     $RTRN         ;;GO DO AN RTI OR RTT TO LOAD THE PSW
11505
11506 057360      $CLR.T:
11507 057360 013700 000042      MOV    @#42,R0      ;;INSURE R0 CONTAINS THE MONITORS
11508 057364 001405      BEQ    $DOAGN       ;;RETURN ADDRESS
11509 057366 000005      RESET  ;;CLEAR THE WORLD
11510 057370 004710      $ENDAD: JSR    PC,(R0) ;;GO TO MONITOR
11511 057372 000240      NOP    ;;SAVE ROOM
11512 057374 000240      NOP    ;;FOR
11513 057376 000240      NOP    ;;ACT11
11514 057400      $DOAGN:
11515 057400 104400      TRAP  ;;PUSH OLD PSW AND PC ON STACK
11516 057402 042716 000020      BIC    #20,(SP)     ;;CLEAR THE 'T' BIT
11517 057406 032777 010000 121524      BIT    #BIT12,@SWR  ;;RUN WITH TRACE TRAP?
11518 057414 001005      BNE   1$           ;;BR IF NO

```

```

11519 057416 005167 000020          COM      $TBIT          ;; IS IT TIME FOR TRACE TRAP
11520 057422 100402          BMI      1$          ;; BR IF NO
11521 057424 052716 000020          BIS      #20,(SP)    ;; SET TRACE TRAP
11522 057430 012746 057436          1$:      MOV      #SLOOP,-(SP) ;; JUMP TO START OF TEST
11523 057434 000002          $RTN:    RTI          ;; RETURN--THIS IS CHANGED TO
                                ;; AN 'RTI' IF 'RTI' IS A LEGAL
                                ;; INSTRUCTION
11524
11525
11526 057436          $LOOP:
11527 057436 000137          JMP      @(PC)+      ;; RETURN
11528 057440 002160          $RTNAD: .WORD      LOOP
11529 057442 000000          $TBIT:  .WORD      0          ;; 'T' BIT STATE INDICATOR
11530 057444      377      377      000          $ENULL: .BYTE     -1,-1,0    ;; NULL CHARACTER STRING
11531 057447      015      042412 042116          $ENDMG: .ASCII    <15><12>/END PASS #'
11532 057454 050040 051501 020123
11533 057462 000043
11534
11535          .SBTTL  SCOPE HANDLER ROUTINE
11536
11537          ;; *****
11538          ;; *THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
11539          ;; *AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG (DISPLAY<7:0>)
11540          ;; *AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
11541          ;; *THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
11542          ;; *SW14=1      LOOP ON TEST
11543          ;; *SW11=1      INHIBIT ITERATIONS
11544          ;; *SW09=1      LOOP ON ERROR
11545          ;; *SW08=1      LOOP ON TEST IN SWR<7:0>
11546          ;; *CALL
11547          ;; *      SCOPE          ;; SCOPE-IOT
11548
11549 057464          $SCOPE:
11550 057464 104407          CKSWR
11551 057466 032777 040000 121444          1$:      BIT      #BIT14,@SWR    ;; TEST FOR CHANGE IN SOFT-SWR
11552 057474 001114          BNE      $OVER        ;; LOOP ON PRESENT TEST?
11553          ;; *****START OF CODE FOR THE XOR TESTER*****
11554 057476 000416          $XTSTR: BR      6$      ;; IF RUNNING ON THE 'XOR' TESTER CHANGE
11555          ;; THIS INSTRUCTION TO A 'NOP' (NOP 240)
11556 057500 013746 000004          MOV      @ERRVEC,-(SP) ;; SAVE THE CONTENTS OF THE ERROR VECTOR
11557 057504 012737 057524 000004          MOV      #5,@ERRVEC  ;; SET FOR TIMEOUT
11558 057512 005737 177060          TST      @177060      ;; TIME OUT ON XOR?
11559 057516 012637 000004          MOV      (SP)+,@ERRVEC ;; RESTORE THE ERROR VECTOR
11560 057522 000463          BR      $SVLAD        ;; GO TO THE NEXT TEST
11561 057524 022626          5$:      CMP      (SP)+,(SP)+ ;; CLEAR THE STACK AFTER A TIME OUT
11562 057526 012637 000004          MOV      (SP)+,@ERRVEC ;; RESTORE THE ERROR VECTOR
11563 057532 000423          BR      7$          ;; LOOP ON THE PRESENT TEST
11564 057534          6$:; *****END OF CODE FOR THE XOR TESTER*****
11565 057534 032777 000400 121376          BIT      #BIT08,@SWR    ;; LOOP ON SPEC. TEST?
11566 057542 001404          BEQ      2$          ;; BR IF NO
11567 057544 127767 121370 121330          CMPB    @SWR,$TSTNM    ;; ON THE RIGHT TEST?   SWR<7:0>
11568 057552 001465          BEQ      $OVER        ;; BR IF YES
11569 057554 105767 121323          2$:      TSTB    $ERFLG      ;; HAS AN ERROR OCCURRED?
11570 057560 001421          BEQ      3$          ;; BR IF NO
11571 057562 126767 121327 121313          CMPB    $ERMAX,$ERFLG ;; MAX. ERRORS FOR THIS TEST OCCURRED?
11572 057570 101015          BHI      3$          ;; BR IF NO
11573 057572 032777 001000 121340          BIT      #BIT09,@SWR    ;; LOOP ON ERROR?
11574 057600 001404          BEQ      4$          ;; BR IF NO
  
```

```

11575 057602 016767 121302 121276 7$ MOV $LPERR,$LPADR ;;SET LOOP ADDRESS TO LAST SCOPE
11576 057610 000446 BR $OVER
11577 057612 105067 121265 4$ CLR $ERFLG ;;ZERO THE ERROR FLAG
11578 057616 005067 121460 CLR $TIMES ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
11579 057622 000415 BR 1$ ;;ESCAPE TO THE NEXT TEST
11580 057624 032777 004000 121306 3$: BIT #BIT11,@SWR ;;INHIBIT ITERATIONS?
11581 057632 001011 BNE 1$ ;;BR IF YES
11582 057634 005767 121464 TST $PASS ;;IF FIRST PASS OF PROGRAM
11583 057640 001406 BEQ 1$ ;; INHIBIT ITERATIONS
11584 057642 005267 121236 INC $ICNT ;;INCREMENT ITERATION COUNT
11585 057646 026767 121430 121230 CMP $TIMES,$ICNT ;;CHECK THE NUMBER OF ITERATIONS MADE
11586 057654 002024 BGE $OVER ;;BR IF MORE ITERATION REQUIRED
11587 057656 012767 000001 121220 1$: MOV #1,$ICNT ;;REINITIALIZE THE ITERATION COUNTER
11588 057664 016767 000052 121410 MOV $MXCNT,$TIMES ;;SET NUMBER OF ITERATIONS TO DO
11589 057672 105267 121204 $SVLAD: INCB $STNM ;;COUNT TEST NUMBERS
11590 057676 116767 121200 121416 MOV $STNM,$TESTN ;;SET TEST NUMBER IN APT MAILBOX
11591 057704 011667 121176 MOV (SP),$LPADR ;;SAVE SCOPE LOOP ADDRESS
11592 057710 011667 121174 MOV (SP),$LPERR ;;SAVE ERROR LOOP ADDRESS
11593 057714 005067 121364 CLR $ESCAPE ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
11594 057720 112767 000001 121167 MOV #1,$EMAX ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
11595 057726 016777 121150 121206 $OVER: MOV $STNM,@DISPLAY ;;DISPLAY TEST NUMBER
11596 057734 016716 121146 MOV $LPADR,(SP) ;;FUDGE RETURN ADDRESS
11597 057740 000002 RTI ;;FIXES PS
11598 057742 000001 $MXCNT: 1 ;;MAX. NUMBER OF ITERATIONS
11599
11600
11601 .SBTTL~ ERROR HANDLER ROUTINE
11602
11603 ::*****
11604 ::*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
11605 ::*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
11606 ::*AND GO TO ERTYPE ON ERROR
11607 ::*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
11608 ::*SW15=1 HALT ON ERROR
11609 ::*SW13=1 INHIBIT ERROR TYPEOUTS
11610 ::*SW10=1 BELL ON ERROR
11611 ::*SW09=1 LOOP ON ERROR
11612 ::*CALL
11613 ::* ERROR N ;;ERROR EMT AND N-ERROR ITEM NUMBER
11614 $ERROR:
11615 057744 104407
11616 057746 105267 121131 7$: CKSWR ;;TEST FOR CHANGE IN SOFT-SWR
11617 057752 001775 INCB $ERFLG ;;SET THE ERROR FLAG
11618 057754 016777 121122 121160 BEQ 7$ ;;DON'T LET THE FLAG GO TO ZERO
11619 057762 032777 002000 121150 MOV $STNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
11620 057770 001402 BIT #BIT10,@SWR ;;BELL ON ERROR?
11621 057772 104401 001306 BEQ 1$ ;;NO - SKIP
11622 057776 005267 121110 TYPE $BELL ;;RING BELL
11623 060002 011667 121110 1$: INC $ERTL ;;COUNT THE NUMBER OF ERRORS
11624 060006 162767 000002 121102 MOV (SP),$ERRPC ;;GET ADDRESS OF ERROR INSTRUCTION
11625 060014 117767 121076 121072 SUB #2,$ERRPC
11626 060022 032777 020000 121110 MOV @ERRPC,$ITEMB ;;STRIP AND SAVE THE ERROR ITEM CODE
11627 060030 001004 BIT #BIT13,@SWR ;;SKIP TYPEOUT IF SET
11628 060032 004767 002314 BNE 20$ ;;SKIP TYPEOUTS
11629 060036 104401 001313 JSR PC,ERTYPE ;;GO TO USER ERROR ROUTINE
11630 060042 20$: TYPE $CRLF
  
```

```

11631 060042 122767 000001 121266      CMPB   #APTENV,$ENV      ;;RUNNING IN APT MODE
11632 060050 001007                      BNE    2$                ;;NO,SKIP APT ERROR REPORT
11633 060052 116767 121036 000004      MOVB   $ITEMB,21$       ;;SET ITEM NUMBER AS ERROR NUMBER
11634 060060 004767 001126                      JSR    PC,$ATY4         ;;REPORT FATAL ERROR TO APT
11635 060064      000          21$:   .BYTE   0
11636 060065      000          .BYTE   0
11637 060066 000777          22$:   BR     22$           ;;APT ERROR LOOP
11638 060070 005777 121044          2$:   TST   @SWR         ;;HALT ON ERROR
11639 060074 100002          BPL    3$                ;;SKIP IF CONTINUE
11640 060076 000000          HALT                          ;;HALT ON ERROR'
11641 060100 104407          CKSWR                          ;;TEST FOR CHANGE IN SOFT-SWR
11642 060102 032777 001000 121030      3$:   BIT   #BIT09,@SWR    ;;LOOP ON ERROR SWITCH SET?
11643 060110 001402          BEQ    4$                ;;BR IF NO
11644 060112 016716 120772          MOV    $LPERR,(SP)       ;;FUDGE RETURN FOR LOOPING
11645 060116 005767 121162          4$:   TST   $ESCAPE       ;;CHECK FOR AN ESCAPE ADDRESS
11646 060122 001402          BEQ    5$                ;;BR IF NONE
11647 060124 016716 121154          MOV    $ESCAPE,(SP)     ;;FUDGE RETURN ADDRESS FOR ESCAPE
11648 060130          5$:
11649 060130 022737 057370 000042          CMP    #SENDAD,@#42     ;;ACT-11 AUTO-ACCEPT?
11650 060136 001001          BNE    6$                ;;BRANCH IF NO
11651 060140 000000          HALT                          ;;YES
11652 060142          6$:
11653 060142 000002          RTI                          ;;RETURN
11654
11655      .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
11656
11657      ;*****
11658      ;*SAVE R0-R5
11659      ;*CALL:
11660      ;*   SAVREG
11661      ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
11662      ;*
11663      ;*TOP---(+16)
11664      ;* +2---(+18)
11665      ;* +4---R5
11666      ;* +6---R4
11667      ;* +8---R3
11668      ;*+10---R2
11669      ;*+12---R1
11670      ;*+14---R0
11671
11672      $SAVREG:
11673 060144 010046      MOV    R0,-(SP)          ;;PUSH R0 ON STACK
11674 060146 010146      MOV    R1,-(SP)          ;;PUSH R1 ON STACK
11675 060150 010246      MOV    R2,-(SP)          ;;PUSH R2 ON STACK
11676 060152 010346      MOV    R3,-(SP)          ;;PUSH R3 ON STACK
11677 060154 010446      MOV    R4,-(SP)          ;;PUSH R4 ON STACK
11678 060156 010546      MOV    R5,-(SP)          ;;PUSH R5 ON STACK
11679 060160 016646 000022      MOV    22(SP),-(SP)     ;;SAVE PS OF MAIN FLOW
11680 060164 016646 000022      MOV    22(SP),-(SP)     ;;SAVE PC OF MAIN FLOW
11681 060170 016646 000022      MOV    22(SP),-(SP)     ;;SAVE PS OF CALL
11682 060174 016646 000022      MOV    22(SP),-(SP)     ;;SAVE PC OF CALL
11683 060200 000002      RTI
11684
11685      ;*RESTORE R0-R5
11686      ;*CALL:
    
```

```

11687
11688 060202
11689 060202 012666 000022
11690 060206 012666 000022
11691 060212 012666 000022
11692 060216 012666 000022
11693 060222 012605
11694 060224 012604
11695 060226 012603
11696 060230 012602
11697 060232 012601
11698 060234 012600
11699 060236 000002
11700
11701
11702
11703
11704
11705
11706
11707
11708
11709
11710
11711
11712
11713
11714
11715
11716
11717
11718 060240 105767 120713
11719 060244 100002
11720 060246 000000
11721 060250 000430
11722 060252 010046
11723 060254 017600 000002
11724 060260 122767 000001 121050
11725 060266 001011
11726 060270 132767 000100 121041
11727 060276 001405
11728 060300 010067 000004
11729 060304 004767 000672
11730 060310 000000
11731 060312 132767 000040 121017
11732 060320 001003
11733 060322 112046
11734 060324 001005
11735 060326 005726
11736 060330 012600
11737 060332 062716 000002
11738 060336 000002
11739 060340 122716 000011
11740 060344 001430
11741 060346 122716 000200
11742 060352 001006

; * RESREG
$RESREG:
MOV (SP)+,22(SP) ;;RESTORE PC OF CALL
MOV (SP)+,22(SP) ;;RESTORE PS OF CALL
MOV (SP)+,22(SP) ;;RESTORE PC OF MAIN FLOW
MOV (SP)+,22(SP) ;;RESTORE PS OF MAIN FLOW
MOV (SP)+,R5 ;;POP STACK INTO R5
MOV (SP)+,R4 ;;POP STACK INTO R4
MOV (SP)+,R3 ;;POP STACK INTO R3
MOV (SP)+,R2 ;;POP STACK INTO R2
MOV (SP)+,R1 ;;POP STACK INTO R1
MOV (SP)+,R0 ;;POP STACK INTO R0
RTI

.SBTTL TYPE ROUTINE

;*****
;ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
;THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
;NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
;NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
;NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
;
;CALL:
;1) USING A TRAP INSTRUCTION
; * TYPE ,MESADR ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
;OR
; * TYPE
; * MESADR
; *

$TYPE: TSTB $TPFLG ;;IS THERE A TERMINAL?
BPL 1$ ;;BR IF YES
HALT ;;HALT HERE IF NO TERMINAL
BR 3$ ;;LEAVE
1$: MOV R0,-(SP) ;;SAVE R0
MOV @2(SP),R0 ;;GET ADDRESS OF ASCIZ STRING
CMPB #APTENV,$ENV ;;RUNNING IN APT MODE
BNE 62$ ;;NO,GO CHECK FOR APT CONSOLE
BITB #APTPOOL,$ENVM ;;SPOOL MESSAGE TO APT
BEQ 62$ ;;NO,GO CHECK FOR CONSOLE
MOV R0,61$ ;;SETUP MESSAGE ADDRESS FOR APT
JSR PC,$ATY3 ;;SPOOL MESSAGE TO APT
61$: .WORD 0 ;;MESSAGE ADDRESS
62$: BITB #APTCSUP,$ENVM ;;APT CONSOLE SUPPRESSED
BNE 60$ ;;YES,SKIP TYPE OUT
2$: MOVB (R0)+,-(SP) ;;PUSH CHARACTER TO BE TYPED ONTO STACK
BNE 4$ ;;BR IF IT ISN'T THE TERMINATOR
TST (SP)+ ;;IF TERMINATOR POP IT OFF THE STACK
60$: MOV (SP)+,R0 ;;RESTORE R0
3$: ADD #2,(SP) ;;ADJUST RETURN PC
RTI ;;RETURN
4$: CMPB #HT,(SP) ;;BRANCH IF <HT>
BEQ 8$
CMPB #CRLF,(SP) ;;BRANCH IF NOT <CRLF>
BNE 5$
    
```

```

11743 060354 005726          TST      (SP)+          ;;POP <CR><LF> EQUIV
11744 060356 104401          TYPE                    ;;TYPE A CR AND LF
11745 060360 001313          $CRLF                    ;
11746 060362 105067 000130  CLRB     $CHARCNT        ;;CLEAR CHARACTER COUNT
11747 060366 000755          BR       2$              ;;GET NEXT CHARACTER
11748 060370 004767 000056  5$:     JSR     PC,$TYPEC  ;;GO TYPE THIS CHARACTER
11749 060374 126726 120556  6$:     CMPB   $FILLC,(SP)+ ;;IS IT TIME FOR FILLER CHARS.?
11750 060400 001350          BNE     2$              ;;IF NO GO GET NEXT CHAR.
11751 060402 016746 120546  MOV      $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
11752                                     ;;AND THE NULL CHAR.
11753 060406 105366 000001  7$:     DECB   1(SP)      ;;DOES A NULL NEED TO BE TYPED?
11754 060412 002770          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
11755 060414 004767 000032  JSR     PC,$TYPEC  ;;GO TYPE A NULL
11756 060420 105367 000072  DECB   $CHARCNT        ;;DO NOT COUNT AS A COUNT
11757 060424 000770          BR      7$              ;;LOOP
11758
11759                                     ;HORIZONTAL TAB PROCESSOR
11760
11761 060426 112716 000040  8$:     MOVB   #' ,(SP)   ;;REPLACE TAB WITH SPACE
11762 060432 004767 000014  9$:     JSR     PC,$TYPEC  ;;TYPE A SPACE
11763 060436 132767 000007 000052  BITB   #7,$CHARCNT     ;;BRANCH IF NOT AT
11764 060444 001372          BNE     9$              ;;TAB STOP
11765 060446 005726          TST     (SP)+          ;;POP SPACE OFF STACK
11766 060450 000724          BR      2$              ;;GET NEXT CHARACTER
11767 060452 105777 120472  $TYPEC: TSTB   @$TPS      ;;WAIT UNTIL PRINTER IS READY
11768 060456 100375          BPL    $TYPEC
11769 060460 116677 000002 120464  MOVB   2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
11770 060466 122766 000015 000002  CMPB   #CR,2(SP)      ;;IS CHARACTER A CARRIAGE RETURN?
11771 060474 001003          BNE     1$              ;;BRANCH IF NO
11772 060476 105067 000014  CLRB   $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
11773 060502 000406          BR      $TYPEX
11774 060504 122766 000012 000002  1$:     CMPB   #LF,2(SP)   ;;IS CHARACTER A LINE FEED?
11775 060512 001402          BEQ    $TYPEX          ;;BRANCH IF YES
11776 060514 105227          INCB   (PC)+          ;;COUNT THE CHARACTER
11777 060516 000000          $CHARCNT:=.WORD 0     ;;CHARACTER COUNT STORAGE
11778 060520 000207          $TYPEX: RTS          PC
11779
11780
11781                                     .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
11782
11783                                     ;;*****
11784                                     ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
11785                                     ;*OCTAL (ASCII) NUMBER AND TYPE IT.
11786                                     ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
11787                                     ;*CALL:
11788                                     ;*   MOV     NUM,-(SP)   ;;NUMBER TO BE TYPED
11789                                     ;*   TYPOS                    ;;CALL FOR TYPEOUT
11790                                     ;*   .BYTE  N              ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
11791                                     ;*   .BYTE  M              ;;M=1 OR 0
11792                                     ;*                                     ;;1=TYPE LEADING ZEROS
11793                                     ;*                                     ;;0=SUPPRESS LEADING ZEROS
11794
11795                                     ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
11796                                     ;*$TYPOS OR $TYPOC
11797                                     ;*CALL:
11798                                     ;*   MOV     NUM,-(SP)   ;;NUMBER TO BE TYPED
    
```





```
11855 060744 000 $OCNT: .BYTE 0 ;;OCTAL DIGIT COUNTER
11856 060745 000 $OFILL: .BYTE 0 ;;ZERO FILL SWITCH
11857 060746 000000 $OMODE: .WORD 0 ;;NUMBER OF DIGITS TO TYPE
11858
11859 BITL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
11860
11861 ;:*****
11862 ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
11863 ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
11864 ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
11865 ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
11866 ;*REPLACED WITH SPACES.
11867 ;*CALL:
11868 ;* MOV NUM,-(SP) ;;PUT THE BINARY NUMBER ON THE STACK
11869 ;* TYPDS ;;GO TO THE ROUTINE
11870
11871 060750 $TYPDS:
11872 060750 010046 MOV R0,-(SP) ;;PUSH R0 ON STACK
11873 060752 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
11874 060754 010246 MOV R2,-(SP) ;;PUSH R2 ON STACK
11875 060756 010346 MOV R3,-(SP) ;;PUSH R3 ON STACK
11876 060760 010546 MOV R5,-(SP) ;;PUSH R5 ON STACK
11877 060762 012746 020200 MOV #20200,-(SP) ;;SET BLANK SWITCH AND SIGN
11878 060766 016605 000020 MOV 20(SP),R5 ;;GET THE INPUT NUMBER
11879 060772 100004 BPL 1$ ;;BR IF INPUT IS POS.
11880 060774 005405 NEG R5 ;;MAKE THE BINARY NUMBER POS.
11881 060776 112766 000055 000001 MOVB #'-,1(SP) ;;MAKE THE ASCII NUMBER NEG.
11882 061004 005000 1$: CLR R0 ;;ZERO THE CONSTANTS INDEX
11883 061006 012703 061164 MOV #SDBLK,R3 ;;SETUP THE OUTPUT POINTER
11884 061012 112723 000040 MOVB #' ,(R3)+ ;;SET THE FIRST CHARACTER TO A BLANK
11885 061016 005002 2$: CLR R2 ;;CLEAR THE BCD NUMBER
11886 061020 016001 061154 MOV $DTBL(R0),R1 ;;GET THE CONSTANT
11887 061024 160105 3$: SUB R1,R5 ;;FORM THIS BCD DIGIT
11888 061026 002402 BLT 4$ ;;BR IF DONE
11889 061030 005202 INC R2 ;;INCREASE THE BCD DIGIT BY 1
11890 061032 000774 BR 3$
11891 061034 060105 4$: ADD R1,R5 ;;ADD BACK THE CONSTANT
11892 061036 005702 TST R2 ;;CHECK IF BCD DIGIT=0
11893 061040 001002 BNE 5$ ;;FALL THROUGH IF 0
11894 061042 105716 TSTB (SP) ;;STILL DOING LEADING 0'S?
11895 061044 100407 BMI 7$ ;;BR IF YES
11896 061046 106316 5$: ASLB (SP) ;;MSD?
11897 061050 103003 BCC 6$ ;;BR IF NO
11898 061052 116663 000001 177777 MOVB 1(SP),-1(R3) ;;YES--SET THE SIGN
11899 061060 052702 000060 6$: BIS #'0,R2 ;;MAKE THE BCD DIGIT ASCII
11900 061064 052702 000040 7$: BIS #' ,R2 ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
11901 061070 110223 MOVB R2,(R3)+ ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
11902 061072 005720 TST (R0)+ ;;JUST INCREMENTING
11903 061074 020027 000010 CMP R0,#10 ;;CHECK THE TABLE INDEX
11904 061100 002746 BLT 2$ ;;GO DO THE NEXT DIGIT
11905 061102 003002 BGT 8$ ;;GO TO EXIT
11906 061104 010502 MOV R5,R2 ;;GET THE LSD
11907 061106 000764 BR 6$ ;;GO CHANGE TO ASCII
11908 061110 105726 8$: TSTB (SP)+ ;;WAS THE LSD THE FIRST NON-ZERO?
11909 061112 100003 BPL 9$ ;;BR IF NO
11910 061114 116663 177777 177776 MOVB -1(SP),-2(R3) ;;YES--SET THE SIGN FOR TYPING
```

```
11911 061122 105013          9$: CLR B      (R3)          ;;SET THE TERMINATOR
11912 061124 012605          MOV      (SP)+,R5        ;;POP STACK INTO R5
11913 061126 012603          MOV      (SP)+,R3        ;;POP STACK INTO R3
11914 061130 012602          MOV      (SP)+,R2        ;;POP STACK INTO R2
11915 061132 012601          MOV      (SP)+,R1        ;;POP STACK INTO R1
11916 061134 012600          MOV      (SP)+,R0        ;;POP STACK INTO R0
11917 061136 104401 061164  TYPE      ,SDBLK          ;;NOW TYPE THE NUMBER
11918 061142 016666 000002 000004 MOV      2(SP),4(SP)     ;;ADJUST THE STACK
11919 061150 012616          MOV      (SP)+,(SP)
11920 061152 000002          RTI                          ;;RETURN TO USER
11921 061154 023420          $DTBL: 10000.
11922 061156 001750          1000.
11923 061160 000144          100.
11924 061162 000012          10.
11925 061164 000004          $DBLK: .BLKW 4
11926
11927          .SBTTL  APT COMMUNICATIONS ROUTINE
11928
11929          ;:*****
11930 061174 112767 000001 000236 $ATY1: MOV B      #1,$FFLG          ;;TO REPORT FATAL ERROR
11931 061202 112767 000001 000226 $ATY3: MOV B      #1,$MFLG          ;;TO TYPE A MESSAGE
11932 061210 000403          BR      $ATYC
11933 061212 112767 000001 000220 $ATY4: MOV B      #1,$FFLG          ;;TO ONLY REPORT FATAL ERROR
11934 061220          $ATYC:
11935 061220 010046          MOV      R0,-(SP)        ;;PUSH R0 ON STACK
11936 061222 010146          MOV      R1,-(SP)        ;;PUSH R1 ON STACK
11937 061224 105767 000206          TST B      $MFLG          ;;SHOULD TYPE A MESSAGE?
11938 061230 001450          BEQ      5$              ;;IF NOT: BR
11939 061232 122767 000001 120076          CMP B      #APTENV,$ENV          ;;OPERATING UNDER APT?
11940 061240 001031          BNE      3$              ;;IF NOT: BR
11941 061242 132767 000100 120067          BIT B      #APTPOOL,$ENV M      ;;SHOULD SPOOL MESSAGES?
11942 061250 001425          BEQ      3$              ;;IF NOT: BR
11943 061252 017600 000004          MOV      @4(SP),R0        ;;GET MESSAGE ADDR.
11944 061256 062766 000002 000004          ADD      #2,4(SP)          ;;BUMP RETURN ADDR.
11945 061264 005767 120026          1$: TST      $MSGTYPE          ;;SEE IF DONE W/ LAST XMISSION?
11946 061270 001375          BNE      1$              ;;IF NOT: WAIT
11947 061272 010067 120034          MOV      R0,$MSGAD        ;;PUT ADDR IN MAILBOX
11948 061276 105720          2$: TST B      (R0)+          ;;FIND END OF MESSAGE
11949 061300 001376          BNE      2$
11950 061302 166700 120024          SUB      $MSGAD,R0        ;;SUB START OF MESSAGE
11951 061306 006200          ASR      R0              ;;GET MESSAGE LNGTH IN WORDS
11952 061310 010067 120020          MOV      R0,$MSGGLT        ;;PUT LENGTH IN MAILBOX
11953 061314 012767 000004 117774          MOV      #4,$MSGTYPE        ;;TELL APT TO TAKE MSG.
11954 061322 000413          BR      5$
11955 061324 017667 000004 000016 3$: MOV      @4(SP),4$          ;;PUT MSG ADDR IN JSR LINKAGE
11956 061332 062766 000002 000004          ADD      #2,4(SP)          ;;BUMP RETURN ADDRESS
11957 061340 016746 116432          MOV      177776,-(SP)      ;;PUSH 177776 ON STACK
11958 061344 004767 176670          JSR      PC,$TYPE          ;;CALL TYPE MACRO
11959 061350 000000          4$: .WORD      0
11960 061352          5$:
11961 061352 105767 000062          10$: TST B      $FFLG          ;;SHOULD REPORT FATAL ERROR?
11962 061356 001416          BEQ      12$             ;;IF NOT: BR
11963 061360 005767 117752          TST      $ENV            ;;RUNNING UNDER APT?
11964 061364 001413          BEQ      12$             ;;IF NOT: BR
11965 061366 005767 117724          11$: TST      $MSGTYPE          ;;FINISHED LAST MESSAGE?
11966 061372 001375          BNE      11$             ;;IF NOT: WAIT
```



```

12023 061560 104401 062044
12024 061564 062706 000006
12025 061570 000757
12026
12027
12028 061572 021627 000015
12029 061576 001022
12030 061600 005766 000004
12031 061604 001403
12032 061606 016677 000002 117324
12033 061614 062706 000006
12034 061620 104401 001313
12035 061624 126727 117305 000001
12036 061632 001003
12037 061634 012777 000100 117302
12038 061642 000002
12039 061644 004767 176602
12040 061650 021627 000060
12041 061654 002420
12042 061656 021627 000067
12043 061662 003015
12044 061664 042726 000060
12045 061670 005766 000002
12046 061674 001403
12047 061676 006316
12048 061700 006316
12049 061702 006316
12050 061704 005266 000002
12051 061710 056616 177776
12052 061714 000707
12053 061716 104401 001312
12054 061722 000720
12055
12056
12057
12058
12059
12060
12061
12062
12063
12064
12065
12066 061724 011646
12067 061726 016666 000004 000002
12068 061734 105777 117204
12069 061740 100375
12070 061742 117766 117200 000004
12071 061750 042766 177600 000004
12072 061756 026627 000004 000023
12073 061764 001013
12074 061766 105777 117152
12075 061772 100375
12076 061774 117746 117146
12077 062000 042716 177600
12078 062004 022627 000021

                TYPE      , $CNTLU
20$: ADD      #6, SP      ;; YES, ECHO CONTROL-U (^U)
    BR      19$          ;; IGNORE PREVIOUS INPUT
                                ;; LET'S TRY IT AGAIN

10$: CMP      (SP), #15   ;; IS IT A <CR>?
    BNE     16$          ;; BRANCH IF NO
    TST     4(SP)        ;; YES, IS IT THE FIRST CHAR?
    BEQ     11$          ;; BRANCH IF YES
    MOV     2(SP), @SWR   ;; SAVE NEW SWR
11$: ADD      #6, SP      ;; CLEAR UP STACK
14$: TYPE     , $CRLF     ;; ECHO <CR> AND <LF>
    CMPB    $INTAG, #1    ;; RE-ENABLE TTY KBD INTERRUPTS?
    BNE     15$          ;; BRANCH IF NOT
    MOV     #100, @STKS   ;; RE-ENABLE TTY KBD INTERRUPTS
15$: RTI
16$: JSR     PC, $TYPEC   ;; ECHO CHAR
    CMP     (SP), #60     ;; CHAR < 0?
    BLT     18$          ;; BRANCH IF YES
    CMP     (SP), #67     ;; CHAR > 7?
    BGT     18$          ;; BRANCH IF YES
    BIC     #60, (SP)+    ;; STRIP-OFF ASCII
    TST     2(SP)        ;; IS THIS THE FIRST CHAR
    BEQ     17$          ;; BRANCH IF YES
    ASL     (SP)         ;; NO, SHIFT PRESENT
    ASL     (SP)         ;; CHAR OVER TO MAKE
    ASL     (SP)         ;; ROOM FOR NEW ONE.
17$: INC     2(SP)       ;; KEEP COUNT OF CHAR
    BIS     -2(SP), (SP)  ;; SET IN NEW CHAR
    BR      7$          ;; GET THE NEXT ONE
18$: TYPE     , $QUES     ;; TYPE ?<CR><LF>
    BR      20$          ;; SIMULATE CONTROL-U
.DSABL LSB

*****
*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
*CALL:
*      RDCHR              ;; INPUT A SINGLE CHARACTER FROM THE TTY
*      RETURN HERE       ;; CHARACTER IS ON THE STACK
*                          ;; WITH PARITY BIT STRIPPED OFF
*
$RDCHR: MOV     (SP), -(SP) ;; PUSH DOWN THE PC
    MOV     4(SP), 2(SP)   ;; SAVE THE PS
1$: TSTB    @STKS         ;; WAIT FOR
    BPL     1$            ;; A CHARACTER
    MOVB    @STKB, 4(SP)   ;; READ THE TTY
    BIC     #^C<177>, 4(SP) ;; GET RID OF JUNK IF ANY
    CMP     4(SP), #23     ;; IS IT A CONTROL-S?
    BNE     3$            ;; BRANCH IF NO
2$: TSTB    @STKS         ;; WAIT FOR A CHARACTER
    BPL     2$            ;; LOOP UNTIL ITS THERE
    MOVB    @STKB, -(SP)   ;; GET CHARACTER
    BIC     #^C177, (SP)   ;; MAKE IT 7-BIT ASCII
    CMP     (SP)+, #21     ;; IS IT A CONTROL-Q?
    
```

```

12079 062010 001366          BNE      2$          ;;IF NOT DISCARD IT
12080 062012 000750          BR       1$          ;;YES, RESUME
12081 062014 026627 000004 000140 3$:    CMP     4(SP),#140  ;;IS IT UPPER CASE?
12082 062022 002407          BLT     4$          ;;BRANCH IF YES
12083 062024 026627 000004 000175          CMP     4(SP),#175  ;;IS IT A SPECIAL CHAR?
12084 062032 003003          BGT     4$          ;;BRANCH IF YES
12085 062034 042766 000040 000004          BIC     #40,4(SP)  ;;MAKE IT UPPER CASE
12086 062042 000002          4$:    RTI          ;;GO BACK TO USER
12087 062044 052536 005015 000          $CNTLU: .ASCIZ /^U/<15><12>  ;;CONTROL 'U'
12088 062051 0136 006507 000012 $CNTLG: .ASCIZ /^G/<15><12>  ;;CONTROL 'G'
12089 062056 005015 053523 020122 $MSWR:  .ASCIZ <15><12>/SWR = /
12090 062064 020075 000          12091 062067 040 047040 053505 $MNEW:  .ASCIZ / NEW = /
12092 062074 036440 000040
12093
12094          .SBTTL TRAP DECODER
12095
12096          ;:*****
12097          ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE 'TRAP' INSTRUCTION
12098          ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
12099          ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
12100          ;*GO TO THAT ROUTINE.
12101
12102 062100 010046          $TRAP:  MOV     R0,-(SP)          ;;SAVE R0
12103 062102 016600 000002          MOV     2(SP),R0          ;;GET TRAP ADDRESS
12104 062106 005740          TST     -(R0)          ;;BACKUP BY 2
12105 062110 111000          MOVSB  (R0),R0          ;;GET RIGHT BYTE OF TRAP
12106 062112 006300          ASL     R0          ;;POSITION FOR INDEXING
12107 062114 016000 062134          MOV     $TRPAD(R0),R0  ;;INDEX TO TABLE
12108 062120 000200          RTS     R0          ;;GO TO ROUTINE
12109
12110
12111          ;:THIS IS USE TO HANDLE THE 'GETPRI' MACRO
12112
12113 062122 011646          $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
12114 062124 016666 000004 000002          MOV     4(SP),2(SP)      ;;MOVE THE PSW DOWN
12115 062132 000002          RTI          ;;RESTORE THE PSW
12116
12117          .SBTTL TRAP TABLE
12118
12119          ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
12120          ;*BY THE 'TRAP' INSTRUCTION.
12121
12122          :          ROUTINE
12123          :          -----
12124 062134 062122          $TRPAD: .WORD  $TRAP2
12125 062136 060240          $TYPE   ;;CALL=TYPE      TRAP+1(104401) TTY TYPEOUT ROUTINE
12126 062140 060546          $TYPOC  ;;CALL=TYPOC     TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
12127 062142 060522          $TYPOS  ;;CALL=TYPOS     TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
12128 062144 060562          $TYPON  ;;CALL=TYPON     TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
12129 062146 060750          $TYPDS  ;;CALL=TYPDS     TRAP+5(104405) TYPE DECIMAL NUMBER (WITH SIGN)
12130
12131 062150 061512          $GTSWR  ;;CALL=GTSWR     TRAP+6(104406) GET SOFT-SWR SETTING
12132
12133 062152 061442          $CKSWR  ;;CALL=CKSWR     TRAP+7(104407) TEST FOR CHANGE IN SOFT-SWR
12134 062154 061724          $RDCHR  ;;CALL=RDCHR     TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
    
```

```

12135 062156 060144          $SAVREG ;;CALL=SAVREG TRAP+11(104411) SAVE R0-R5 ROUTINE
12136 062160 060202          $RESREG ;;CALL=RESREG TRAP+12(104412) RESTORE R0-R5 ROUTINE
12137 062162 062630          .RSET  ;;CALL=RSETUP TRAP+13(104413) ROUTINE TO RESET STACK AND FPS
12138 062164 062622          .LPER  ;;CALL=LPERR  TRAP+14(104414) ROUTINE TO SET LOOP ON ERROR ADDRESS
12139 000032
12140
12141          .SBTTL POWER DOWN AND UP ROUTINES
12142
12143          ::*****
12144          :POWER DOWN ROUTINE
12145 062166 012737 062344 000024 $PWRDN: MOV    #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
12146 062174 012737 000340 000026      MOV    #340,@#PWRVEC+2 ;;PRIO:7
12147 062202 010046          MOV    R0,-(SP)      ;;PUSH R0 ON STACK
12148 062204 010146          MOV    R1,-(SP)      ;;PUSH R1 ON STACK
12149 062206 010246          MOV    R2,-(SP)      ;;PUSH R2 ON STACK
12150 062210 010346          MOV    R3,-(SP)      ;;PUSH R3 ON STACK
12151 062212 010446          MOV    R4,-(SP)      ;;PUSH R4 ON STACK
12152 062214 010546          MOV    R5,-(SP)      ;;PUSH R5 ON STACK
12153 062216 017746 116716          MOV    @SWR,-(SP)    ;;PUSH @SWR ON STACK
12154 062222 010667 000122          MOV    SP,$SAVR6    ;;SAVE SP
12155 062226 012737 062240 000024          MOV    #$PWRUP,@#PWRVEC ;;SET UP VECTOR
12156 062234 000000          HALT
12157 062236 000776          BR     .-2          ;;HANG UP
12158
12159          ::*****
12160          :POWER UP ROUTINE
12161 062240 012737 062344 000024 $PWRUP: MOV    #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
12162 062246 016706 000076          MOV    $SAVR6,SP    ;;GET SP
12163 062252 005067 000072          CLR    $SAVR6      ;;WAIT LOOP FOR THE TTY
12164 062256 005267 000066          1$: INC    $SAVR6    ;;WAIT FOR THE INC
12165 062262 001375          BNE    1$          ;;OF WORD
12166 062264 012677 116650          MOV    (SP)+,@SWR   ;;POP STACK INTO @SWR
12167 062270 012605          MOV    (SP)+,R5    ;;POP STACK INTO R5
12168 062272 012604          MOV    (SP)+,R4    ;;POP STACK INTO R4
12169 062274 012603          MOV    (SP)+,R3    ;;POP STACK INTO R3
12170 062276 012602          MOV    (SP)+,R2    ;;POP STACK INTO R2
12171 062300 012601          MOV    (SP)+,R1    ;;POP STACK INTO R1
12172 062302 012600          MOV    (SP)+,R0    ;;POP STACK INTO R0
12173 062304 012737 062166 000024          MOV    #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
12174 062312 012737 000340 000026          MOV    #340,@#PWRVEC+2 ;;PRIO:7
12175 062320 104401          TYPE    ;;REPORT THE POWER FAILURE
12176 062322 062700          $PWRMG: .WORD POWERM ;;POWER FAIL MESSAGE POINTER
12177 062324 012716          MOV    (PC)+,(SP)  ;;RESTART AT START
12178 062326 001466          $PWRAD: .WORD START ;;RESTART ADDRESS
12179 062330 042766 000020 000052          BIC    #20,2(SP)   ;;CLEAR 'T' BIT
12180 062336 005067 175100          CLR    $TBIT      ;;CLEAR THE 'T' BIT FLAG
12181 062342 000002          RTI
12182 062344 000000          $ILLUP: HALT      ;;THE POWER UP SEQUENCE WAS STARTED
12183 062346 000776          BR     .-2          ;; BEFORE THE POWER DOWN WAS COMPLETE
12184 062350 000000          $SAVR6: 0          ;;PUT THE SP HERE
12185
12186          .SBTTL ERROR TYPE OUT ROUTINE
12187          ::*****
12188          :*****
12189          :*THIS ROUTINE IS CALLED TO TYPE AN ERROR MESSAGE WHICH IS INCLUDED
12190          :*IN THE ERROR MESSAGE DATA TABLE. IT IS CALLED BY THE $ERROR ROUTINE
    
```

```

12191
12192
12193
12194
12195 062352 104401
12196 062354 001313
12197 062356 113737 001102 001232
12198 062364 042737 177400 001232
12199 062372 013737 001116 001234
12200 062400 010046
12201
12202 062402 113700 001114
12203 062406 042700 177400
12204 062412 001007
12205 062414 104401 062466
12206
12207 062420 013746 001116
12208 062424 104402
12209 062426 000137 062462
12210 062432 005300
12211
12212 062434 006300
12213 062436 062700 001442
12214
12215 062442 011037 062452
12216 062446 001405
12217 062450 104401
12218 062452 000000
12219 062454 104401
12220 062456 001313
12221 062460 000755
12222
12223 062462 012600
12224 062464 000207
12225
12226 062466 046106 040517 044524
12227 062474 043516 050040 044517
12228 062502 052116 042440 051122
12229 062510 051117 020054 052123
12230 062516 050117 042520 020104
12231 062524 052101 050040 036503
12232 062532 000040
12233
12234
12235
12236
12237
12238
12239
12240
12241
12242
12243
12244 062534 011637 001236
12245 062540 022626
12246 062542 170200

; *OR BY FIRST SETTING $ITEMB EQUAL TO THE ERROR TABLE ITEM TO BE PRINTED
; *OUT AND THEN EXECUTING A:
; *
; * JSR PC,ERTYPE
; *
ERTYPE: TYPE ;TYPE A CRLF
        .WORD $CRLF
        MOVB @#$STNM,@#$TMP0
        BIC #177400,@#$TMP0
        MOV @#$ERRPC,@#$TMP1 ;GET PC OF CALL
        MOV RO,-(SP) ;SAVE RO

        MOVB @#$ITEMB,RO ;GET THE ITEM NUMBER.
        BIC #177400,RO
        BNE MULT
        PCTYP: TYPE ,EMSG

        MOV @#$ERRPC,-(SP) ;IF ZERO THEN JUST
        TYPOC ;PRINT THE PC
        JMP @#$ERT5
        MULT: DEC RO ;OTHERWISE MULT RO BY 2 TO
        ;GET ERROR MESSAGE POINTER

        ASL RO
        ADD @#$ERRTB,RO

        MOV (RO),@#$2$ ;PICK UP THE ADDRESS
        BEQ ERT5 ;OF THE EM, ERROR MESSAGE
        TYPE
        $: .WORD 0
        TYPE
        .WORD $CRLF
        BR PCTYP

ERT5: MOV (SP)+,RO ;RESTORE RO.
      RTS PC ;AND RETURN.

EMSG: .ASCIZ /FLOATING POINT ERROR, STOPPED AT PC /

.EVEN

.SBTTL FPP SPURIOUS TRAP TO 244 HANDLER
;*****
;*****
; *THIS ROUTINE HANDLES UNEXPECTED TRAPS TO THE FPP TRAP VECTOR AT 244.
; *THE LAST FPP INSTRUCTION EXECUTED AND ITS ADDRESS HAS BEEN RECORDED
; *THESE ALONG WITH THE FEC, FPS AND PC OF TRAP ARE REPORTED.
; *
FPPSPUR: MOV (SP),@#$TMP2 ;SAVE PC OF TRAP.
          CMP (SP)+,(SP)+ ;RESTORE SP.
          STFPS RO ;GET FPS
    
```



```
12247 062544 010037 001240      MOV     RO,@#STMP3
12248 062550 170300              STST    RO                ;GET FEC
12249 062552 010037 001242      MOV     RO,@#STMP4
12250 062556 104001              1$:    ERROR 1            ;MOST LIKELY BAD FP1 CHIP
12251 062560 104413              RSETUP                ;GO INITIALIZE THE FPS AND STACK; AND
12252                                     ;SEE IF THE USER HAS EXPRESSED
12253                                     ;THE DESIRE TO CHANGE THE SOFTWARE
12254                                     ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
12255                                     ;THE USER TYPED CONTROL G?).
12256 062562 000137 057264      JMP     @#SEOP
12257
12258
12259                                     .SBTTL CPU SPURIOUS TRAP TO 4 HANDLER
12260                                     :*****
12261                                     :*****
12262                                     :THIS ROUTINE REPORTS UNEXPECTED CPU TRAPS TO VECTOR 4.
12263                                     :*
12264 062566 011637 001236      (PSPUR: MOV    (SP),@#STMP2          ;SAVE PC OF TRAP.
12265 062572 022626              CMP     (SP)+,(SP)+
12266 062574 104004              1$:    ERROR 4
12267 062576 104413              RSETUP                ;GO INITIALIZE THE FPS AND STACK; AND
12268                                     ;SEE IF THE USER HAS EXPRESSED
12269                                     ;THE DESIRE TO CHANGE THE SOFTWARE
12270                                     ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
12271                                     ;THE USER TYPED CONTROL G?).
12272 062600 000137 057264      JMP     @#SEOP
12273
12274
12275                                     .SBTTL CPU SPURIOUS TRAP TO 10 HANDLER
12276                                     :*****
12277                                     :*****
12278                                     :THIS ROUTINE REPORTS UNEXPECTED CPL TRAPS TO VECTOR 10.
12279                                     :*
12280 062604 011637 001236      (PTWO: MOV    (SP),@#STMP2          ;SAVE PC OF TRAP.
12281 062610 022626              CMP     (SP)+,(SP)+
12282 062612 104002              1$:    ERROR 2
12283 062614 104413              RSETUP                ;GO INITIALIZE THE FPS AND STACK; AND
12284                                     ;SEE IF THE USER HAS EXPRESSED
12285                                     ;THE DESIRE TO CHANGE THE SOFTWARE
12286                                     ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
12287                                     ;THE USER TYPED CONTROL G?).
12288 062616 000137 057264      JMP     @#SEOP
12289
12290
12291
12292
12293                                     .SBTTL SET LOOP ON ERROR ADDRESS ROUTINE
12294                                     :*****
12295                                     :*****
12296                                     :*
12297 062622 011637 001110      (LPER: MOV    (SP),@#SLPERR
12298 062626 000002              RTI
12299
12300                                     .SBTTL FLAG RESET AND CONSOLE TEST ROUTINE
12301                                     :*****
12302                                     :*****
```

```

12303 ;*THIS ROUTINE WILL BE CALLED AT THE END OF EACH TEST TO
12304 ;*RESET THE STACK, CLEAR THE FPS AND SEE IF THE USER HAS TYPED
12305 ;* CONTROL G ON THE TERMINAL. IF THE USER HAS TYPED CONTROL G AND
12306 ;*THERE IS NO PHYSICAL CONSOLE SWITCH REGISTER THEN THE CONTENTS
12307 ;*OF THE SOFTWARE SWITCH REGISTER WILL BE TYPED IN OCTAL ON THE
12308 ;*TELETYPE AND THE USER CAN MODIFY IT.
12309 ;*
12310 062630 023727 001140 177570 .RSET: CMP @#SWR,#177570 ;SEE IF THERE IS A PHYSICAL
12311 ;CONSOLE SWITCH REGISTER.
12312 062636 001001 BNE 1$ ;BRANCH IF NO.
12313 062640 104407 (KSWR ;OTHERWISE TYPE THE CONTENTS
12314 ;OF THE PROGRAM VIRTUAL SWITCH REGISTER
12315 ;AND GIVE THE USER A CHANCE TO
12316 ;MODIFY IT.
12317 062642 012737 062534 000244 1$: MOV #FPSPUR,@#FPVECT
12318 062650 012737 062566 000004 MOV #CPSPUR,@#ERRVECT
12319 062656 012737 062604 000010 MOV #CPTWO,@#10
12320 062664 011600 MOV (SP),R0 ;SAVE RETURN ADDRESS.
12321 062666 012706 001100 MOV #STACK,SP ;RESET THE STACK POINTER.
12322 062672 005004 CLR R4 ;CLEAR THE FPS.
12323 062674 170104 LDFPS R4
12324 062676 000110 JMP (R0) ;RETURN.
12325
12326 .NLIST BEX
    
```

;SPECIAL MESSAGES:

```

062700 050200 053517 051105 POWERM: .ASCIZ <CRLF>'POWER FAILURE. PROGRAM RESTARTING.'<CRLF>
062745 000 NULL: .BYTE 0
062746 000011 $TAB: .ASCIZ <TAB>
062750 020040 000 SPACE: .ASCIZ ' '
    
```

;ERROR MESSAGES:

```

062753 120 047522 040502 EM1: .ASCIZ /PROBABLY BAD FP1 CHIP/
063001 120 047522 040502 EM2: .ASCIZ /PROBABLY BAD HYBRID FP CHIP/
063035 120 047522 040502 EM3: .ASCIZ /PROBABLY BAD MMU CHIP/
063063 120 047522 040502 EM4: .ASCIZ /PROBABLY BAD MMU OR HYBRID FP CHIPS/
000001 .END
    
```

AAADON	040746	AA6	022215	AMTYP4=	000000	BBP11	026222	BIT3	= 000010
AAA1	037744	AA7	022224	APASS =	000000	BBP7	026202	BIT4	= 000020
AAA10	040362	ABASE =	000000	APRIOR=	000000	BB1	024732	BIT5	= 000040
AAA11	040420	ACDW1 =	000000	APTCU=	000040	BB10	025106	BIT6	= 000100
AAA12	040456	ACDW2 =	000000	APTENV=	000001	BB11	025126	BIT7	= 000200
AAA13	040514	ACDW2 =	000000	APTSIZ=	000200	BB12	025136	BIT8	= 000400
AAA2	040002	ACFJOP=	000000	APTSPO=	000100	BB13	025144	BIT9	= 001000
AAA3	040040	ACO =%	000000	ASWREG=	000000	BB14	025156	BPTVEC=	000014
AAA4	040076	AC1 =%	0000001	ATES^N=	000000	BB15	025206	B1	002456
AAA5	040134	AC2 =%	0000002	AUNIT =	000000	BB16	025232	B2	002460
AAA6	040172	AC3 =%	0000003	AUSWR =	000000	BB17	025242	B3	002476
AAA7	040230	AC4 =%	0000004	AVECT1=	000000	BB2	024770	CCCDON	042460
AAA8	040266	AC5 =%	0000005	AVECT2=	000000	BB20	025254	CCC1	041454
AAA9	040324	AC6 =%	0000006	A1	002252	BB21	025304	CCC10	042050
AA^ATO	023040	AC7 =%	0000007	A11	002252	BB22	025330	CCC11	042104
AADONE	023140	ADDW0 =	000000	A12	002266	BB23	025350	CCC12	042140
AAERRO	022472	ADDW1 =	000000	A2	002326	BB24	025360	CCC13	042174
AAERR0	022472	ADDW10=	000000	BBBDON	041450	BB25	025366	CCC2	041510
AAERR1	022560	ADDW11=	000000	BBBER1	041224	BB26	025400	CCC3	041544
AAERR2	022614	ADDW12=	000000	BBBER2	041310	BB27	025430	CCC4	041600
AAERR3	022650	ADDW13=	000000	BBBER3	041336	BB3	024774	CCC5	041634
AAERR4	022660	ADDW14=	000000	BBBER4	041364	BB30	025452	CCC6	041670
AAERR5	022714	ADDW15=	000000	BBBP1	041430	BB31	025462	CCC7	041724
AAERR6	022750	ADDW2 =	000000	BBBP2	041440	BB32	025474	CCC8	041760
AAERR7	023004	ADDW3 =	000000	BBB0	040754	BB33	025524	CCC9	042014
AAER10	022566	ADDW4 =	000000	BBB1	041010	BB34	025546	CCDATO	024566
AAPATO	023050	ADDW5 =	000000	BBB2	041036	BB35	025556	CCDONE	024726
AAPAT1	023060	ADDW6 =	000000	BBB3	041066	BB4	025012	CCERO	024130
AAPAT2	023070	ADDW7 =	000000	BBB4	041114	BB5	025022	CCER1	024164
AAPAT3	023100	ADDW8 =	000000	BBB5	041152	BB6	025034	CCER10	024476
AAPAT4	023110	ADDW9 =	000000	BBB6	041160	BB7	025064	CCER11	024514
AAPAT5	023120	ADEVCT=	000000	BBDATO	026102	BDONE	002546	CCER12	024532
AAPAT6	023130	ADEVM =	000000	BBDONE	026232	BERR	002502	CCER13	024550
AA1	022112	ADONE	002444	BBERU	025570	BERR1	002532	CCER2	024222
AA10	022226	AENV =	000000	BBER1	025630	BIT0	= 000001	CCER22	024236
AA11	022256	AENVM =	000000	BBER10	025610	BIT00	= 000001	CCER3	024260
AA12	022260	AERR1	002336	BBER11	025644	BIT01	= 000002	CCER4	024276
AA13	022276	AERR2	002350	BBER2	025666	BIT02	= 000004	CCER44	024312
AA14	022316	AERR3	002402	BBER3	025704	BIT03	= 000010	CCER5	024334
AA15	022336	AFATAL=	000000	BBER4	025740	BIT04	= 000020	CCER50	024200
AA16	022344	AMADR1=	000000	BBER5	025774	BIT05	= 000040	CCER55	024350
AA17	022350	AMADR2=	000000	BBER6	026012	BIT06	= 000100	CCER6	024370
AA2	022150	AMADR3=	000000	BBER7	026046	BIT07	= 000200	CCER7	024424
AA20	022354	AMADR4=	000000	BBER8	026064	BIT08	= 000400	CCER8	024442
AA21	022356	AMAMS1=	000000	BBPAT0	026112	BIT09	= 001000	CCER90	024146
AA22	022410	AMAMS2=	000000	BBPAT1	026122	BIT1	= 000002	CCP0	024576
AA23	022412	AMAMS3=	000000	BBPAT2	026132	BIT10	= 002000	CCP1	024606
AA24	022432	AMAMS4=	000000	BBPAT3	026142	BIT11	= 004000	CCP10	024676
AA25	022452	AMSGAD=	000000	BBPAT4	026152	BIT12	= 010000	CCP11	024706
AA26	022460	AMSGLG=	000000	BBPAT5	026162	BIT13	= 020000	CCP12	024716
AA27	022464	AMSGTY=	000000	BBPAT6	026172	BIT14	= 040000	CCP2	024616
AA3	022152	AMTYP1=	000000	BBP10	026212	BIT15	= 100000	CCP3	024626
AA4	022170	AMTYP2=	000000			BIT2	= 000004	CCP4	024636
AA5	022210	AMTYP3=	000000						

CCP5	024646	C35	002706	DD16	026564	EEDATO	030720	E3	003466
CCP6	024656	C4	002732	DD17	026614	EEDONE	031002	E4	003470
CCP7	024666	C45	002744	DD18	026636	EEEDON	044424	FDAT10	005372
CC1	023144	C5	002770	DD19	026656	EEER0	030512	FDAT11	005374
CC10	023360	C55	003010	DD2	026266	EEER1	030530	FDAT12	005376
CC11	023370	C6	003032	DD20	026666	EEER2	030566	FDAT13	005400
CC12	023376	C65	003046	DD21	026674	EEER3	030624	FDAT14	005402
CC13	023410	C7	003072	DD22	026706	EEER4	030662	FDAT15	005404
CC14	023440	C75	003114	DD23	026736	EEE1	043420	FDAT16	005406
CC15	023464	C8	003140	DD24	026760	EEE10	044014	FDAT17	005410
CC16	023504	C85	003152	DD25	027000	EEE11	044050	FDAT00	005414
CC17	023514	DDDATO	030076	DD26	027010	EEE12	044104	FDAT01	005416
CC18	023522	DDDDON	043414	DD27	027016	EEE13	044140	FDAT02	005420
CC19	023534	DDDONE	030226	DD3	026310	EEE2	043454	FDAT03	005422
CC2	023174	DDD1	042464	DD30	027030	EEE3	043510	FDAT04	005424
CC20	023564	DDD2	042540	DD31	027060	EEE4	043544	FDAT05	005426
CC21	023610	DDD3	042614	DD32	027102	EEE5	043600	FDAT06	005430
CC22	023630	DDD4	042670	DD33	027122	EEE6	043634	FDAT07	005432
CC23	023640	DDD5	042744	DD34	027132	EEE7	043670	FDONE	005456
CC24	023646	DDD6	043020	DD35	027140	EEE8	043724	FERRO	004200
CC25	023660	DDD7	043074	DD36	027156	EEE9	043760	FERR1	004236
CC26	023710	DDERO	027310	DD37	027206	EEO0	030730	FERR10	005016
CC27	023732	DDER1	027326	DD38	027230	EEO1	030742	FERR11	005052
CC28	023752	DDER10	027744	DD39	027250	EEO2	030752	FERR2	004334
CC29	023762	DDER11	030002	DD4	026330	EEO3	030762	FERR20	005072
CC3	023216	DDER12	030040	DD40	027260	EEO4	030772	FERR21	005210
CC30	023770	DDER2	027364	DD41	027266	EERRO	003514	FERR25	005232
CC31	024002	DDER3	027422	DD42	027304	EERR1	003532	FERR26	005350
CC32	024032	DDER4	027460	DD5	026340	EERR2	003546	FERR3	004374
CC33	024054	DDER5	027516	DD6	026346	EE1	030232	FERR4	004366
CC34	024074	DDER6	027554	DD7	026364	EE10	030452	FERR5	004472
CC35	024104	DDER7	027612	DD8	026414	EE11	030462	FERR6	004526
CC36	024112	DDER8	027650	DERR1	003316	EE12	030470	FERR7	004662
CC37	024124	DDER9	027706	DERR2	003412	EE13	030506	FER2	004340
CC4	023236	DDISP =	177570	DISPLA	001142	EE2	030262	FFDATO	031300
CC5	023246	DDONE	003436	DISPRE	000174	EE3	030304	FFDONE	031360
CC6	023254	DDPO	030106	DIVDSU	043154	EE4	030324	FFERO	031170
CC7	023266	DDP1	030116	DIVDT	043404	EE5	030334	FFER1	031204
CC8	023316	DDP2	030126	DIVFSU	042234	EE6	030342	FFER2	031242
CC9	023340	DDP3	030136	DIVFT	042450	EE7	030360	FFFON	045154
CDONE	003174	DDP4	030146	DPAT3	013316	EE8	030410	FFF1	044430
CKSWR =	104407	DDP5	030156	DSWR =	177570	EE9	030432	FFF2	044504
CMPSUB	040556	DDP6	030166	D1	003222	EMSG	062466	FFF3	044560
CMPTMP	040736	DDP7	030176	D10	003424	EMTVEC =	000030	FFF4	044634
CPSPUR	062566	DDP8	030206	D2	003246	EM1	062753	FFP0	031310
CPTWO	062604	DDP9	030216	D3	003250	EM2	063001	FFP1	031320
CR =	000015	DD1	026236	D4	003254	EM3	063035	FFP2	031330
CRLF =	000200	DD10	026436	D5	003266	EM4	063063	FFP3	031340
C1	002562	DD11	026446	D6	003302	ERRVEC =	000004	FFP4	031350
C15	002602	DD12	026464	D7	003312	ERTYPE	062352	FF1	031006
C2	002624	DD13	026514	D8	003362	FRT5	062462	FF10	031156
C25	002640	DD14	026536	D9	003374	E1	003452	FF11	031166
C3	002664	DD15	026546	EDONE	003576	E2	003466	FF2	031036

FF3	031060	GGER12	035322	GG20	034134	G32	010266	HHHDON	054344
FF4	031066	GGER13	035370	GG21	034156	G33	010270	HHH1	052242
FF5	031076	GGER14	035436	GG22	034166	G34	010344	HHH10	053342
FF6	031126	GGER15	035534	GG23	034204	G35	010376	HHH11	053442
FF7	031150	GGER16	035602	GG24	034242	G36	010400	HHH12	053542
FPSPUR	062534	GGER17	035650	GG25	034260	G37	010454	HHH13	053642
FPVECT=	000244	GGER18	035716	GG26	034326	G4	007244	HHH2	052342
FXDAT0	005436	GGER19	035764	GG27	034336	G40	010506	HHH3	052442
FXDAT1	005440	GGER2	034542	GG28	034372	G41	010510	HHH4	052542
FXDAT2	005442	GGER20	036032	GG3	033366	G42	010564	HHH5	052642
FXDAT3	005444	GGER3	034610	GG4	033376	G43	010616	HHH6	052742
FXDAT4	005446	GGER4	034656	GG5	033414	G44	010620	HHH7	053042
FXDAT5	005450	GGER5	034660	GG6	033452	G5	007276	HHH8	053142
FXDAT6	005452	GGER6	034726	CG7	033470	G6	007300	HHH9	053242
FXDAT7	005454	GGER7	034774	GG8	033536	G7	007354	HHP0	033142
F1	003602	GGER8	035072	GG9	033546	HADR	011602	HHP1	033152
F10	004014	GGER9	035140	GOR0	011032	HA1R	011656	HHP10	033262
F11	004016	GGGDON	052236	GOR1	011034	HA1W	011606	HHP11	033272
F12	004034	GGG1	050564	GOR2	011036	HA2R	011666	HHP2	033162
F13	004066	GGG10	051334	GOR3	011040	HA2W	011616	HHP3	033172
F135	004046	GGG11	051404	GPAT00	011002	HA3R	011676	HHP4	033202
F14	004076	GGG12	051454	GPAT01	011004	HA3W	011626	HHP5	033212
F15	004106	GGG13	051524	GPAT02	011006	HA4R	011706	HHP6	033222
F16	004116	GGG14	051574	GPAT03	011010	HA4W	011636	HHP7	033232
F17	004126	GGG2	050634	GPAT10	011012	HA5R	011716	HHP8	033242
F2	003634	GGG3	050704	GPAT11	011014	HA5W	011646	HHP9	033252
F20	004136	GGG4	050754	GPAT12	011016	HCLR	011532	HHTRAP	033062
F21	004146	GGG5	051024	GPAT13	011020	HCLR1	011542	HH1	031362
F22	004156	GGG6	051074	GRESET	010726	HCMP	011474	HH10	031574
F23	004174	GGG7	051144	GSETUP	010650	HCMP1	011514	HH11	031612
F3	003654	GGG8	051214	GS1	010700	HCMP2	011524	HH12	031642
F4	003656	GGG9	051264	GTSWR =	104406	HDAT1	011726	HH13	031664
F5	003674	GGP1	036110	G1	007134	HDAT2	011736	HH14	031704
F6	003750	GGP2	036120	G10	007406	HDAT3	011746	HH15	031714
F7	003774	GGP3	036130	G11	007410	HDAT4	011756	HH16	031722
GANDO	011022	GGP4	036140	G12	007464	HDAT5	011766	HH17	031740
GAND1	011024	GGP5	036150	G13	007516	HDONE	011776	HH18	031776
GAND2	011026	GGP6	036160	G14	007520	HERROR	011550	HH19	032020
GAND3	011030	GGP7	036170	G15	007574	HFLAG	011604	HH2	031412
GCMP	010746	GGP8	036200	G16	007626	HHDAT0	033132	HH20	032030
GDATA00	011042	GGP9	036210	G17	007630	HHDONE	033302	HH21	032046
GDATA01	011044	GG1	033306	G2	007160	HHERO	032152	HH22	032076
GDATA02	011046	GG10	033602	G20	007704	HHERO0	032266	HH23	032120
GDATA03	011050	GG11	033640	G21	007736	HHER1	032220	HH24	032130
GDONE	011052	GG12	033662	G22	007740	HHER10	033014	HH25	032146
GFLAG1	010776	GG13	033672	G23	010014	HHER2	032334	HH3	031434
GFLAG2	011000	GG14	033710	G24	010046	HHER3	032402	HH4	031454
GGDATO	036100	GG15	033746	G25	010050	HHER4	032450	HH5	031464
GGDONE	036220	GG16	033764	G26	010124	HHER5	032516	HH6	031472
GGERO	034376	GG17	034032	G27	010156	HHER6	032564	HH7	031504
GGER1	034474	GG18	034042	G3	007170	HHER7	032632	HH8	031542
GGER10	035206	GG19	034076	G30	010160	HHER8	032700	HH9	031564
GGER11	035254	GG2	033344	G31	010234	HHER9	032746	HSTD	011416

HT	= 000011	HX32	037142	I11	005650	KBUF3	013276	LPAT21	013610
HXDATO	037630	HX33	037164	I12	005652	KDATI0	013260	LPAT22	013612
HXDONE	037740	HX34	037174	I13	005666	KDATI1	013262	LPAT23	013614
HXER1	037216	HX35	037212	I14	005722	KDATI2	013264	LPERR =	104414
HXER10	037506	HX4	036310	I15	005724	KDATI3	013266	L1	013326
HXER11	037540	HX5	036330	I16	005726	KDAT00	013300	L2	013366
HXER12	037560	HX6	036340	I17	005744	KDAT01	013302	L3	013370
HXER13	037610	HX7	036346	I2	005502	KDAT02	013304	L4	013372
HXER2	037256	HX8	036364	I20	006004	KDAT03	013306	L5	013442
HXER22	037272	HX9	036416	I21	006020	KDONE	013320	L6	013450
HXER3	037306	H1	011060	I22	006024	KERR0	013142	MDAT00	014162
HXER33	037322	H10	011322	I23	006040	KERR1	013206	MDAT01	014164
HXER4	037336	H11	011354	I3	005542	KERR2	013232	MDAT02	014166
HXER5	037236	H12	011406	I4	005544	KKKDON	047616	MDAT03	014170
HXER6	037372	H2	011100	I5	005546	KKK1	046760	MDONE	014172
HXER66	037406	H3	011110	I6	005572	KKK3	047024	MERR0	014016
HXER7	037422	H4	011162	I7	005606	KKK4	047070	MERR1	014054
HXER8	037354	H5	011204	JBUF0	012766	KKK5	047134	MERR2	013770
HXER9	037452	H6	011236	JBUF1	012770	KPAT0	013310	MERR3	014102
HXP1	037640	H7	011270	JBUF2	012772	KPAT1	013312	MMMDON	055272
HXP2	037650	IDATI0	006274	JBUF3	012774	KPAT2	013314	MM1	054350
HXP3	037660	IDATI1	006276	JDATI0	012776	K1	013034	MM2	054422
HXP4	037670	IDATI2	006300	JDATI1	013000	K10	013162	MM3	054474
HXP5	037700	IDATI3	006302	JDATI2	013002	K2	013060	MM4	054546
HXP6	037710	IDAT00	006264	JDATI3	013004	K3	013062	MM5	054620
HXP7	037720	IDAT01	006266	JDAT00	013006	K4	013064	MODDD0	056300
HXP8	037730	IDAT02	006270	JDAT01	013016	K5	013120	MODDD1	056310
HX1	036224	IDAT03	006272	JDAT02	013010	K6	013126	MODDOV	055712
HX10	036432	IDONE	006304	JDAT03	013012	K7	013140	MODDSU	053746
HX11	036454	IERR0	006044	JDAT1	013014	LDATI0	013620	MODDT0	054324
HX12	036474	IERR1	006126	JDAT2	013020	LDATI1	013622	MODDT1	054334
HX13	036504	IERR2	006146	JDAT3	013022	LDATI2	013624	MODFD0	055252
HX14	036512	IERR25	006170	JDONE	013024	LDATI3	013626	MODFD1	055262
HX15	036530	IERR3	006220	JERR0	013026	LDAT00	013632	MODFOV	054676
HX16	036556	IERR4	006174	JERR1	012714	LDAT01	013634	MODFSU	051650
HX165	036562	IIDON	046014	JERR2	012740	LDAT02	013636	MODFT0	052206
HX17	036610	III1	045160	JJJDON	046754	LDAT03	013640	MODFT1	052216
HX18	036646	III2	045224	JJJ1	046020	LDONE	013642	MUDP1	052226
HX19	036666	III3	045270	JJJ2	046104	LERR1	013454	MPAT10	014142
HX2	036254	III4	045334	JJJ3	046170	LERR2	013526	MPAT11	014144
HX20	036676	IOTVEC=	000020	JJJ4	046254	LERR3	013500	MPAT12	014146
HX21	036700	IPAT10	006244	J1	012540	LF =	000012	MPAT13	014150
HX22	036730	IPAT11	006246	J10	012666	LLLON	050560	MPAT20	014152
HX23	036750	IPAT12	006250	J2	012564	LLL1	047622	MPAT21	014154
HX24	036770	IPAT13	006252	J3	012566	LLL2	047706	MPAT22	014156
HX25	037000	IPAT20	006254	J4	012570	LLL3	047772	MPAT23	014160
HX26	037006	IPAT21	006256	J5	012624	LLL4	050056	MULDSU	044714
HX27	037010	IPAT22	006260	J6	012632	LOOP	002160	MULDT	045144
HX28	037042	IPAT23	006262	J7	012644	LPAT10	013576	MULFSU	044200
HX29	037064	I1	005464	KBUF0	013270	LPAT11	013600	MULFT	044414
HX3	036270	I10	005640	KBUF1	013272	LPAT12	013602	MULT	062432
HX30	037074	I105	005646	KBUF2	013274	LPAT13	013604	M1	013646
HX31	037112	I106	005642			LPAT20	013606	M15	013666

M2	013672	N8	014302	PDAT10	016040	QDAT00	016540	SERR10	012232
M3	013674	N9	014314	PDAT11	016042	QDAT01	016542	SERR15	012312
M4	013676	ODAT10	015374	PDAT12	016044	QDAT02	016544	SERR2	012352
M5	013732	ODAT11	015376	PDAT13	016046	QDAT03	016546	SERR20	012332
M6	013736	ODAT12	015400	PDAT00	016050	QDONE	016560	SERR3	012376
M7	013746	ODAT13	015402	PDAT01	016052	QERR0	016226	SERR4	012270
M8	013756	ODAT00	015332	PDAT02	016054	QERR1	016472	SERR5	012454
M9	013766	ODAT01	015334	PDAT03	016056	QERR11	016236	SERR6	012364
NDAT10	014670	ODAT02	015336	PDONE	016060	QERR12	016254	SERR7	012410
NDAT11	014672	ODAT03	015340	PERR0	015526	QERR13	016272	SPACE	062750
NDAT12	014674	ODONE	015404	PERR1	015746	QERR14	016310	SPAT10	012512
NDAT13	014676	OERR0	015074	PERR10	015546	QERR15	016326	SPAT11	012514
NDAT00	014626	OERR1	015174	PERR11	015556	QERR16	016336	SPAT12	012516
NDAT01	014630	OERR10	015126	PERR12	015574	QERR17	016344	SPAT13	012520
NDAT02	014632	OERR11	015140	PERR13	015612	QERR2	016426	STALK =	001100
NDAT03	014634	OERR2	015230	PERR14	015630	QERR20	016372	START	001466
NDONE	014700	OERR20	015202	PERR15	015646	QERR21	016402	STKLMT=	177774
NERR0	014370	OERR3	015240	PERR16	015656	QERR22	016410	SWR	001140
NERR1	014470	OERR4	015250	PERR17	015664	QERR3	016436	SWREG	000176
NERR10	014422	OERR5	015260	PERR2	015774	QERR4	016444	SW0 =	000001
NERR11	014434	OERR6	015304	PERR20	015712	QPAT10	016520	SW00 =	000001
NERR2	014524	OPAT10	015364	PERR21	015722	QPAT11	016522	SW01 =	000002
NERR20	014476	OPAT11	015366	PERR22	015730	QPAT12	016524	SW02 =	000004
NERR3	014534	OPAT12	015370	PIRQ =	177772	QPAT13	016526	SW03 =	000010
NERR4	014544	OPAT13	015372	PIRQVE=	000240	QPAT20	016530	SW04 =	000020
NERR5	014554	OPAT20	015350	POWERM	062700	QPAT21	016532	SW05 =	000040
NERR6	014600	OPAT21	015352	PPAT10	016030	QPAT22	016534	SW06 =	000100
NNNDON	056320	OPAT22	015354	PPAT11	016032	QPAT23	016536	SW07 =	000200
NNN1	055276	OPAT23	015356	PPAT12	016034	Q1	016064	SW08 =	000400
NNN2	055400	OPAT24	015360	PPAT13	016036	Q10	016222	SW09 =	001000
NNN3	055502	OVDNTT	046744	PR0 =	000000	Q2	016106	SW1 =	000002
NNN4	055604	OVDTT	050550	PR1 =	000040	Q3 =	016110	SW10 =	002000
NPAT10	014660	OVFNNT	046004	PR2 =	000100	Q4	016112	SW11 =	004000
NPAT11	014662	OVFTT	047606	PR3 =	000140	Q5	016134	SW12 =	010000
NPAT12	014664	OVUNDN	046344	PR4 =	000200	Q6	016156	SW13 =	020000
NPAT13	014666	OVUNDT	050146	PR5 =	000240	Q7	016166	SW14 =	040000
NPAT20	014646	OVUNFN	045404	PR6 =	000300	Q8	016200	SW15 =	100000
NPAT21	014650	OVUNFT	047204	PR7 =	000340	Q9	016214	SW2 =	000004
NPAT22	014652	01	014704	PS =	177776	RDCHR =	104410	SW3 =	000010
NPAT23	014654	010	015034	PSW =	177776	RESRFG=	104412	SW4 =	000020
NULL	062745	011	015044	PWRVEC=	000024	RESVEC=	000010	SW5 =	000040
N1	014176	012	015046	P1	015410	RSETUP=	104413	SW6 =	000100
N10	014330	013	015062	P2	015432	R6 =	%000006	SW7 =	000200
N11	014340	014	015072	P3 =	015434	R7 =	%000007	SW8 =	000400
N12	014342	02	014730	P4	015436	SADR	012506	SW9 =	001000
N13	014356	03	014732	P5	015460	SAVREG=	104411	S1	012002
N14	014366	04	014734	P6	015502	SDAT00	012522	S10	012150
N2	014222	05	014752	P7	015512	SDAT01	012524	S11	012174
N3	014224	06	014762	P8	015522	SDAT02	012526	S12	012204
N4	014226	07	014772	QDAT10	016550	SDAT03	012530	S2	012036
N5	014244	08	015006	QDAT11	016552	SDONE	012532	S3	012040
N6	014254	09	015020	QDAT12	016554	SERR0	012212	S4	012044
N7	014264	PCTYP	062414	QDAT13	016556	SERR1	012422	S5	012070



S6	012100	TST31	022110	T7	006434	WDAT01	020376	XPAT5	057174
S7	012106	TST32	023142	UDONE	017676	WDAT02	020400	XPAT6	057204
S8	012142	TST33	024730	UERR0	017342	WDAT03	020402	XTMP	021136
'9	012144	TST34	026234	UERR1	017366	WDONE	020404	XT1	056326
AB =	000011	TST35	030230	UERR10	017374	WPAT00	020364	XT1A	056342
T. ITVE =	000014	TST36	031004	UERR11	017436	WPAT01	020366	XT10	056754
TDA*10	007066	TST37	031360	UERR2	017452	WPAT02	020370	XT11	057002
TDA1.1	007070	TST4	003176	UERR20	017460	WPAT03	020372	XT12	057030
TDA12	007072	TST40	033304	UERR21	017522	W1	017700	XT13	057064
TDA13	007074	TST41	036222	UERR3	017536	W10	020106	XT13B	057112
TDAT00	007056	TST42	037742	UERR4	017566	W11	020134	XT2	056344
TDAT01	007060	TST43	040750	UFLAG	017662	W12	020164	XT2A	056364
TDAT02	007062	TST44	041452	UPAT00	017612	W13	020210	XT2B	056402
TDAT03	007064	TST45	042462	UPAT01	017614	W14	020220	XT3	056414
TDONE	007076	TST46	043416	UPAT02	017616	W15	020246	XT3A	056436
TERR0	006644	TST47	044426	UPAT03	017620	W16	020302	XT4	056446
TERR1	006726	TST5	003440	UPAT10	017622	W17	020326	XT4A	056476
TERR2	006746	TST50	045156	UPAT11	017624	W2	017730	XT4B	056510
TERR25	006762	TST51	046016	UPAT12	017626	W20	020336	XT5	056526
TERR3	007012	TST52	046756	UPAT13	017630	W3	017754	XT5A	056546
TERR4	006774	TST53	047620	UPAT20	017632	W4	017766	XT5B	056560
1KVEC =	000060	TST54	050562	UPAT21	017634	W5	020016	XT6	056576
TPAT10	007036	TST55	052240	UPAT22	017636	W6	020052	XT6A	056624
TPAT11	007040	TST56	054346	UPAT23	017640	W7	020076	XT6B	056636
TPAT12	007042	TST57	055274	UPAT30	017642	XAPT11	021162	XT7	056654
TPAT13	007044	TST6	003600	UPAT31	017644	XBUF	057114	XT8	056702
TFAT20	007046	TST60	056322	UPAT32	017646	XDAT00	021140	XT9	056730
TPAT21	007050	TST61	057264	UPAT33	017650	XDAT01	021142	XX1	056324
TPAT22	007052	TST7	005460	UPAT40	017652	XDAT02	021144	X1	020410
TPAT23	007054	TYPDS =	104405	UPAT41	017654	XDAT03	021146	X10	020602
TPVEC =	000064	TYPE =	104401	UPAT42	017656	XDONE	021200	X11	020616
TRAPVE =	000034	TYPOC =	104402	UPAT43	017660	XERR1	021024	X12	020652
TRTVEC =	000014	TYPON =	104404	UROM1	017670	XERR2	021106	X13	020676
TST1	002220	TYPOS =	104403	UROM2	017672	XERR3	021052	X14	020704
TST10	006306	T1	006310	UROM3	017674	XERR4	021122	X15	020720
TST11	007100	T10	006444	UTMP1	017664	XNEXT	057224	X16	020754
TST12	011054	T105	006452	UTMP2	017666	XPAT0	057214	X17	021000
TST13	012000	T11	006454	U0	016600	XPAT0	057124	X2	020444
TST14	012534	T12	006456	U1	016630	XPAT00	021150	X20	021006
TST15	013030	T13	006472	U10	017166	XPAT01	021152	X21	021022
TST16	013322	T14	006526	U11	017170	XPAT02	021154	X3	020470
TST17	013644	T15	006530	U12	017222	XPAT03	021156	X4	020476
TST2	002446	T16	006532	U13	017240	XPAT1	057134	X5	020514
TST20	014174	T17	006550	U14	017272	XPAT10	021160	X6	020550
TST21	014702	T2	006330	U15	017322	XPAT12	021164	X7	020574
TST22	015406	T20	006610	U16	017324	XPAT13	021166	YDAT00	021544
TST23	016062	T21	006624	U2	016674	XPAT2	057144	YDAT01	021546
TST24	016562	T22	006626	U3	016726	XPAT20	021170	YDAT02	021550
TST25	017676	T23	006640	U4	016772	XPAT21	021172	YDAT03	021552
TST26	020406	T3	006370	U5	017024	XPAT22	021174	YDONE	021604
TST27	021202	T4	006372	U6	017070	XPAT23	021176	YERR1	021404
TST3	002550	T5	006374	U7	017122	XPAT3	057154	YERR2	021446
TST30	021606	T6	006420	WDAP00	020374	XPAT4	057164	YERR3	021510



YFLAG	021534	\$ATYC	061220	\$ERFLG	001103	\$OMODE	060746	\$TKS	001144
YPAT00	021554	\$ATY1	061174	\$ERMAX	001115	\$OVER	057726	\$TMP0	001232
YPAT01	021556	\$ATY3	061202	\$ERROR	057744	\$PASS	001324	\$TMP1	001234
YPAT02	021560	\$ATY4	061212	\$ERRPC	001116	\$PASTM	001460	\$TMP10	001252
YPAT03	021562	\$AUTOB	001134	\$ERRTB	001442	\$PWRAD	062326	\$TMP11	001254
YPAT10	021564	\$BASE	001372	\$ERTL	001112	\$PWRDN	062166	\$TMP12	001256
YPAT11	021566	\$BDADR	001122	\$ESCAP	001304	\$PWRMG	062322	\$TMP13	001260
YPAT12	021570	\$BDDAT	001126	\$ETABL	001336	\$PWRUP	062240	\$TMP14	001262
YPAT13	021572	\$BELL	001306	\$ETEND	001442	\$QUES	001312	\$TMP15	001264
YPAT20	021574	\$CDW1	001376	\$FATAL	001320	\$RDCHR	061724	\$TMP16	001266
YPAT21	021576	\$CDW2	001400	\$FFLG	061440	\$RDSZ =	000001	\$TMP17	001270
YPAT22	021600	\$CHARC	060516	\$FILLC	001156	\$REGAD	001160	\$TMP2	001236
YPAT23	021602	\$CKSWR	061442	\$FILLS	001155	\$REGO	001162	\$TMP20	001272
YTMP1	021536	\$CLR.T	057360	\$GDADR	001120	\$REG1	001164	\$TMP21	001274
YTMP2	021540	\$CMTAG	001100	\$GDDAT	001124	\$REG10	001202	\$TMP22	001276
YTMP3	021542	\$CM1 =	000024	\$GET42	057342	\$REG11	001204	\$TMP23	001300
Y1	021232	\$CM2 =	000050	\$GTSWR	061512	\$REG12	001206	\$TMP3	001240
Y2	021254	\$CM3 =	000024	\$HD =	000003	\$REG13	001210	\$TMP4	001242
Y3	021300	\$CM4 =	000024	\$HIBTS	001452	\$REG14	001212	\$TMP5	001244
Y4	021316	\$CNTLG	062051	\$ICNT	001104	\$REG15	001214	\$TMP6	001246
Y5	021356	\$CNTLU	062044	\$ILLUP	062344	\$REG16	001216	\$TMP7	001250
Y6	021360	\$CPUOP	001344	\$INTAG	001135	\$REG17	001220	\$TN =	000061
Y7	021374	\$CRLF	001313	\$ITEMB	001114	\$REC2	001166	\$TPB	001152
ZDAT00	022046	\$DBLK	061164	\$LF	001314	\$REG20	001222	\$TPFLG	001157
ZDAT01	022050	\$DDW0	001402	\$LFLG	061437	\$REG21	001224	\$TPS	001150
ZDAT02	022052	\$DDW1	001404	\$LOOP	057436	\$REG22	001226	\$TRAP	062100
ZDAT03	022054	\$DDW10	001426	\$LPADR	001106	\$REG23	001230	\$TRAP2	062122
ZDONE	022106	\$DDW11	001430	\$LPERR	001110	\$REG3	001170	\$TRP =	000015
ZERR1	021752	\$DDW12	001432	\$MADR1	001350	\$REG4	001172	\$TRPAD	062134
ZERR2	022014	\$DDW13	001434	\$MADR2	001354	\$REG5	001174	\$TSTM	001456
ZFLAG	022040	\$DDW14	001436	\$MADR3	001360	\$REG6	001176	\$TSTM	001102
ZPAT00	022056	\$DDW15	001440	\$MADR4	001364	\$REG7	001200	\$TYPDS	060750
ZPAT01	022060	\$DDW2	001406	\$MAIL	001316	\$RESRE	060202	\$TYPE	060240
ZPAT02	022062	\$DDW3	001410	\$MAMS1	001346	\$RTNAD	057440	\$TYPEC	060452
ZPAT03	022064	\$DDW4	001412	\$MAMS2	001352	\$RTRN	057434	\$TYPEX	060520
ZPAT10	022066	\$DDW5	001414	\$MAMS3	001356	\$SAVRE	060144	\$TYPOC	060546
ZPAT11	022070	\$DDW6	001416	\$MAMS4	001362	\$SAVR6	062350	\$TYPON	060562
ZPAT12	022072	\$DDW7	001420	\$MBADR	001454	\$SCOPE	057464	\$TYPOS	060522
ZPAT13	022074	\$DDW8	001422	\$MFLG	061436	\$SETUP=	000137	\$UNIT	001330
ZPAT20	022076	\$DDW9	001424	\$MNEW	062067	\$STUP =	177777	\$UNITM	001462
ZPAT21	022100	\$DEVCT	001326	\$MSGAD	001332	\$SVLAD	057672	\$USWR	001342
ZPAT22	022102	\$DEVN	001374	\$MSGLG	001334	\$SVPC =	001452	\$VECT1	001366
ZPAT23	022104	\$DOAGN	057400	\$MSGTY	001316	\$SWR =	177400	\$VECT2	001370
ZTMP1	022042	\$DTBL	061154	\$MSWR	062056	\$SWREG	001340	\$XTSTR	057476
ZTMP2	022044	\$ENDAD	057370	\$MTYP1	001347	\$SWRMK=	000000	\$SGE14=	000001
Z1	021630	\$ENDCT	057320	\$MTYP2	001353	\$SWRMS=	000200	\$OFILL	060745
Z2	021652	\$ENDMG	057447	\$MTYP3	001357	\$TAB	062746	. =	063127
Z3	021676	\$ENULL	057444	\$MTYP4	001363	\$TBIT	057442	.LPER	062622
Z4	021704	\$ENV	001336	\$MXCNT	057742	\$TERM =	000032	.RSET	062630
Z5	021716	\$ENVM	001337	\$NULL	001154	\$TESTN	001322	.\$X =	001452
Z6	021750	\$EOP	057264	\$NWTST=	000001	\$TIMES	001302		
\$APTHD	001452	\$EOPCT	057312	\$OCNT	060744	\$TKB	001146		

. ABS. 063127 000

CJKDCA-A KEF11-A FP DIAG PART 1 MACY11 30A(1052) 11-JUN-79 11:50 <sup>L 2</sup> PAGE 232  
CJKDCA.P11 08-JUN-79 08:40 SYMBOL TABLE

SEQ 0231

ERRORS DETECTED: 0

MULE:CJKDCA,MULE:CJKDCA.SEQ/SOL/NL·TOC=CJKDCA.P11  
RUN-TIME: 110 93 6 SECONDS  
RUN-TIME RATIO: 563/211=2.6  
CORE USED: 32K (63 PAGES)