

The microfiche card contains a grid of frames, each representing a frame of data. The frames are arranged in approximately 12 rows and 12 columns. Each frame contains a small amount of text, which appears to be diagnostic data for a PDP11/23. The text is very small and difficult to read, but it seems to consist of alphanumeric strings, possibly hexadecimal or binary data, and some more complex patterns. The card is dark blue with a grid of lighter blue frames.

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

.REM &

IDENTIFICATION

PRODUCT CODE: AC-F244A-MC
PRODUCT NAME: CJKDDAO KEF11-AA DIAGNOSTIC PART 2
DATE CREATED: 20-JUN-79
MAINTAINER: DIAGNOSTIC ENGINEERING

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED 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

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, FP1) 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 DFFFA)
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 LDCIF AND LDCLF
145 LDCID AND LDCLD
146 LDEXP
147 STFPS (ALL DESTINATION MODES)
148 STCFL AND STCFI
149 STCDL AND STCDI
150 STEXP
151 STST

2. REQUIREMENTS

2.1 EQUIPMENT

A PROCESSOR USING 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

199

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

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 (R0),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

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

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 SIMPLE 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 JNDERFLOW 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

```
712
713
714      000443      MNUMBER=443
715      000003      PROGNUM=3
716
717
718
719              .LIST  ME
720              .NLIST MD,MC,CND
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735              .ENABL ABS
736
737
738
739      .TITLE  CJKDDA KEF11-A DIAG PART 2
740      :*COPYRIGHT (C) JUNE 1979
741      :*DIGITAL EQUIPMENT CORP.
742      :*MAYA RD, MASS. 01754
743      :*
744      :*PROGRAM BY DIAGNOSTIC ENGINEERING
745      :*
746      :*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
747      :*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
748      :*
749      000001      $TN=1
750      160000      $SWR=160000      ;;HALT ON ERROR, LOOP ON TEST, INHIBIT ERRCR TYP0UT
751
752
753      000244      FPVECT=244
754      177400      $SWR=177400
755      000200      $SWRMSK=200
756      000011      TAB=11
757      000015      CRLF=15
758
759      .SBTTL  BASIC DEFINITIONS
760
761      :*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
762      001100      STACK= 1100
763      .EQUIV  EMT,ERROR      ;;BASIC DEFINITION OF ERROR CALL
764      .EQUIV  IOT,SCOPE      ;;BASIC DEFINITION OF SCOPE CALL
765
766      :*MISCELLANEOUS DEFINITIONS
767      000011      HT- 11      ;;CGDE FOR HORIZONTAL TAB
```


768	000012	LF= 12	::CODE FOR LINE FEED
769	000015	CR= 15	::CODE FOR CARRIAGE RETURN
770	000200	CRLF= 200	::CODE FOR CARRIAGE RETURN-LINE FEED
771	177776	PS= 177776	::PROCESSOR STATUS WORD
772		.EQUIV PS,PSW	
773	177774	STKLMT= 177774	::STACK LIMIT REGISTER
774	177772	PIRQ= 177772	::PROGRAM INTERRUPT REQUEST REGISTER
775	177570	DSWR= 177570	::HARDWARE SWITCH REGISTER
776	177570	DDISP= 177570	::HARDWARE DISPLAY REGISTER
777			
778		:*GENERAL PURPOSE REGISTER DEFINITIONS	
779	000000	R0= %0	::GENERAL REGISTER
780	000001	R1= %1	::GENERAL REGISTER
781	000002	R2= %2	::GENERAL REGISTER
782	000003	R3= %3	::GENERAL REGISTER
783	000004	R4= %4	::GENERAL REGISTER
784	000005	R5= %5	::GENERAL REGISTER
785	000006	R6= %6	::GENERAL REGISTER
786	000007	R7= %7	::GENERAL REGISTER
787	000006	SP= %6	::STACK POINTER
788	000007	PC= %7	::PROGRAM COUNTER
789			
790		:*PRIORITY LEVEL DEFINITIONS	
791	000000	PR0= 0	::PRIORITY LEVEL 0
792	000040	PR1= 40	::PRIORITY LEVEL 1
793	000100	PR2= 100	::PRIORITY LEVEL 2
794	000140	PR3= 140	::PRIORITY LEVEL 3
795	000200	PR4= 200	::PRIORITY LEVEL 4
796	000240	PR5= 240	::PRIORITY LEVEL 5
797	000300	PR6= 300	::PRIORITY LEVEL 6
798	000340	PR7= 340	::PRIORITY LEVEL 7
799			
800		:*'SWITCH REGISTER' SWITCH DEFINITIONS	
801	100000	SW15= 100000	
802	040000	SW14= 40000	
803	020000	SW13= 20000	
804	010000	SW12= 10000	
805	004000	SW11= 4000	
806	002000	SW10= 2000	
807	001000	SW09= 1000	
808	000400	SW08= 400	
809	000200	SW07= 200	
810	000100	SW06= 100	
811	000040	SW05= 40	
812	000020	SW04= 20	
813	000010	SW03= 10	
814	000004	SW02= 4	
815	000002	SW01= 2	
816	000001	SW00= 1	
817		.EQUIV SW09,SW9	
818		.EQUIV SW08,SW8	
819		.EQUIV SW07,SW7	
820		.EQUIV SW06,SW6	
821		.EQUIV SW05,SW5	
822		.EQUIV SW04,SW4	
823		.EQUIV SW03,SW3	

```
824 .EQUIV SW02,SW2
825 .EQUIV SW01,SW1
826 .EQUIV SW00,SW0
827
828 ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
829 100000 BIT15= 100000
830 040000 BIT14= 40000
831 020000 BIT13= 20000
832 010000 BIT12= 10000
833 004000 BIT11= 4000
834 002000 BIT10= 2000
835 001000 BIT09= 1000
836 000400 BIT08= 400
837 000200 BIT07= 200
838 000100 BIT06= 100
839 000040 BIT05= 40
840 000020 BIT04= 20
841 000010 BIT03= 10
842 000004 BIT02= 4
843 000002 BIT01= 2
844 000001 BIT00= 1
845 .EQUIV BIT09,BIT9
846 .EQUIV BIT08,BIT8
847 .EQUIV BIT07,BIT7
848 .EQUIV BIT06,BIT6
849 .EQUIV BIT05,BIT5
850 .EQUIV BIT04,BIT4
851 .EQUIV BIT03,BIT3
852 .EQUIV BIT02,BIT2
853 .EQUIV BIT01,BIT1
854 .EQUIV BIT00,BIT0
855
856 ;*BASIC 'CPU' TRAP VECTOR ADDRESSES
857 000004 ERRVEC= 4 ;;TIME OUT AND OTHER ERRORS
858 000010 RESVEC= 10 ;;RESERVED AND ILLEGAL INSTRUCTIONS
859 000014 TBITVEC=14 ;;'T' BIT
860 000014 TRTVEC= 14 ;;TRACE TRAP
861 000014 BPTVEC= 14 ;;BREAKPOINT TRAP (BPT)
862 000020 IOTVEC= 20 ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
863 000024 PWRVEC= 24 ;;POWER FAIL
864 000030 EMTVEC= 30 ;;EMULATOR TRAP (EMT) **ERROR**
865 000034 TRAPVEC=34 ;;'TRAP' TRAP
866 000060 TKVEC= 60 ;;TTY KEYBOARD VECTOR
867 000064 TPVEC= 64 ;;TTY PRINTER VECTOR
868 000240 PIRQVEC=240 ;;PROGRAM INTERRUPT REQUEST VECTOR
869 .SBTTL FPP REGISTER DEFINITIONS
870 000000 AC0 =%0
871 000001 AC1 =%1
872 000002 AC2 =%2
873 000003 AC3 =%3
874 000004 AC4 =%4
875 000005 AC5 =%5
876 000006 AC6 =%6
877 000007 AC7 =%7
878
879 .SBTTL TRAP CATCHER
```

```
880
881      000000
882      .=0
883      ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
884      ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
885      ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
886      000174 000174
887      000174 000000
888      000176 000000
889      000200 000137 001370
      .=174
      DISPREG: .WORD 0      ;;SOFTWARE DISPLAY REGISTER
      SWREG:   .WORD 0      ;;SOFTWARE SWITCH REGISTER
      .SBTTL  STARTING ADDRESS(ES)
      JMP     @#START ;;JUMP TO STARTING ADDRESS OF PROGRAM
```

```
890 .SBTTL COMMON TAGS
891
892 ;:*****
893 ;:THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
894 ;:USED IN THE PROGRAM.
895
896 001100 .=1100
897 001100 000000 $CMTAG: .START OF COMMON TAGS
898 001102 000 $STSTM: .WORD 0 ;:CONTAINS THE TEST NUMBER
899 001103 000 $SERFLG: .BYTE 0 ;:CONTAINS ERROR FLAG
900 001104 000000 $ICNT: .WORD 0 ;:CONTAINS SUBTEST ITERATION COUNT
901 001106 000000 $LPADR: .WORD 0 ;:CONTAINS SCOPE LOOP ADDRESS
902 001110 000000 $LPERR: .WORD 0 ;:CONTAINS SCOPE RETURN FOR ERRORS
903 001112 000000 $ERTTL: .WORD 0 ;:CONTAINS TOTAL ERRORS DETECTED
904 001114 000 $ITEMB: .BYTE 0 ;:CONTAINS ITEM CONTROL BYTE
905 001115 001 $ERMAX: .BYTE 1 ;:CONTAINS MAX. ERRORS PER TEST
906 001116 000000 $ERRPC: .WORD 0 ;:CONTAINS PC OF LAST ERROR INSTRUCTION
907 001120 000000 $GDADR: .WORD 0 ;:CONTAINS ADDRESS OF 'GOOD' DATA
908 001122 000000 $BDADR: .WORD 0 ;:CONTAINS ADDRESS OF 'BAD' DATA
909 001124 000000 $GDDAT: .WORD 0 ;:CONTAINS 'GOOD' DATA
910 001126 000000 $BDDAT: .WORD 0 ;:CONTAINS 'BAD' DATA
911 001130 000000 .WORD 0 ;:RESERVED--NOT TO BE USED
912 001132 000000 .WORD 0
913 001134 000 $AUTOB: .BYTE 0 ;:AUTOMATIC MODE INDICATOR
914 001135 000 $INTAG: .BYTE 0 ;:INTERRUPT MODE INDICATOR
915 001136 000000 .WORD 0
916 001140 177570 SWR: .WORD DSWR ;:ADDRESS OF SWITCH REGISTER
917 001142 177570 DISPLAY: .WORD DDISP ;:ADDRESS OF DISPLAY REGISTER
918 001144 177560 $TKS: 177560 ;:TTY KBD STATUS
919 001146 177562 $TKB: 177562 ;:TTY KBD BUFFER
920 001150 177564 $TPS: 177564 ;:TTY PRINTER STATUS REG. ADDRESS
921 001152 177566 $TPB: 177566 ;:TTY PRINTER BUFFER REG. ADDRESS
922 001154 000 $NULL: .BYTE 0 ;:CONTAINS NULL CHARACTER FOR FILLS
923 001155 002 $FILLS: .BYTE 2 ;:CONTAINS # OF FILLER CHARACTERS REQUIRED
924 001156 012 $FILLC: .BYTE 12 ;:INSERT FILL CHARS. AFTER A 'LINE FEED'
925 001157 000 $TPFLG: .BYTE 0 ;:'TERMINAL AVAILABLE' FLAG (BIT<07>=0=YES)
926 001160 000000 $REGAD: .WORD 0 ;:CONTAINS THE ADDRESS FROM
927 001162 000000 $REG0: .WORD 0 ;:WHICH ($REG0) WAS OBTAINED
928 001164 000000 $REG1: .WORD 0 ;:CONTAINS (($REGAD)+0)
929 001166 000000 $REG2: .WORD 0 ;:CONTAINS (($REGAD)+2)
930 001170 000000 $REG3: .WORD 0 ;:CONTAINS (($REGAD)+4)
931 001172 000000 $REG4: .WORD 0 ;:CONTAINS (($REGAD)+6)
932 001174 000000 $REG5: .WORD 0 ;:CONTAINS (($REGAD)+10)
933 001176 000000 $REG6: .WORD 0 ;:CONTAINS (($REGAD)+12)
934 001200 000000 $REG7: .WORD 0 ;:CONTAINS (($REGAD)+14)
935 001202 000000 $REG10: .WORD 0 ;:CONTAINS (($REGAD)+16)
936 001204 000000 $REG11: .WORD 0 ;:CONTAINS (($REGAD)+20)
937 001206 000000 $REG12: .WORD 0 ;:CONTAINS (($REGAD)+22)
938 001210 000000 $REG13: .WORD 0 ;:CONTAINS (($REGAD)+24)
939 001212 000000 $REG14: .WORD 0 ;:CONTAINS (($REGAD)+26)
940 001214 000000 $REG15: .WORD 0 ;:CONTAINS (($REGAD)+30)
941 001216 000000 $REG16: .WORD 0 ;:CONTAINS (($REGAD)+32)
942 001220 000000 $REG17: .WORD 0 ;:CONTAINS (($REGAD)+34)
943 001222 000000 $REG20: .WORD 0 ;:CONTAINS (($REGAD)+36)
944 001222 000000 ;:CONTAINS (($REGAD)+40)
945
```

```
946 001224 000000 $REG21: .WORD 0 ;;CONTAINS (($REGAD)+42)
947 001226 000000 $REG22: .WORD 0 ;;CONTAINS (($REGAD)+44)
948 001230 000000 $REG23: .WORD 0 ;;CONTAINS (($REGAD)+46)
949 001232 000000 $TMP0: .WORD 0 ;;USER DEFINED
950 001234 000000 $TMP1: .WORD 0 ;;USER DEFINED
951 001236 000000 $TMP2: .WORD 0 ;;USER DEFINED
952 001240 000000 $TMP3: .WORD 0 ;;USER DEFINED
953 001242 000000 $TMP4: .WORD 0 ;;USER DEFINED
954 001244 000000 $TMP5: .WORD 0 ;;USER DEFINED
955 001246 000000 $TMP6: .WORD 0 ;;USER DEFINED
956 001250 000000 $TMP7: .WORD 0 ;;USER DEFINED
957 001252 000000 $TMP10: .WORD 0 ;;USER DEFINED
958 001254 000000 $TMP11: .WORD 0 ;;USER DEFINED
959 001256 000000 $TMP12: .WORD 0 ;;USER DEFINED
960 001260 000000 $TMP13: .WORD 0 ;;USER DEFINED
961 001262 000000 $TMP14: .WORD 0 ;;USER DEFINED
962 001264 000000 $TMP15: .WORD 0 ;;USER DEFINED
963 001266 000000 $TMP16: .WORD 0 ;;USER DEFINED
964 001270 000000 $TMP17: .WORD 0 ;;USER DEFINED
965 001272 000000 $TMP20: .WORD 0 ;;USER DEFINED
966 001274 000000 $TMP21: .WORD 0 ;;USER DEFINED
967 001276 000000 $TMP22: .WORD 0 ;;USER DEFINED
968 001300 000000 $TMP23: .WORD 0 ;;USER DEFINED
969 001302 000000 $TIMES: 0 ;;MAX. NUMBER OF ITERATIONS
970 001304 000000 $ESCAPE: 0 ;;ESCAPE ON ERROR ADDRESS
971 001306 177607 000377 $BELL: .ASCIZ <207><377><377> ;;CODE FOR BELL
972 001312 077 $QUES: .ASCII /?/ ;;QUESTION MARK
973 001313 015 $CRLF: .ASCII <15> ;;CARRIAGE RETURN
974 001314 000012 $LF: .ASCIZ <12> ;;LINE FEED
975 *****
976 .SBTTL APT MAILBOX-ETABLE
977 *****
978 .EVEN
979 $MAIL: ;;APT MAILBOX
980 001316 $MSGTY: .WORD AMSTGY ;;MESSAGE TYPE CODE
981 001316 000000 $FATAL: .WORD AFATAL ;;FATAL ERROR NUMBER
982 001320 000000 $TESTN: .WORD ATESTN ;;TEST NUMBER
983 001322 000000 $PASS: .WORD APASS ;;PASS COUNT
984 001324 000000 $DEVCT: .WORD ADEVCT ;;DEVICE COUNT
985 001326 000000 $UNIT: .WORD AUNIT ;;I/O UNIT NUMBER
986 001330 000000 $MSGAD: .WORD AMSGAD ;;MESSAGE ADDRESS
987 001332 000000 $MSGLG: .WORD AMSGLG ;;MESSAGE LENGTH
988 001334 000000 $ETABLE: ;;APT ENVIRONMENT TABLE
989 001336 000 $ENV: .BYTE AENV ;;ENVIRONMENT BYTE
990 001336 000 $ENVM: .BYTE AENVM ;;ENVIRONMENT MODE BITS
991 001337 000 $SWREG: .WORD ASWREG ;;APT SWITCH REGISTER
992 001340 000000 $USWR: .WORD AUSWR ;;USER SWITCHES
993 001342 000000 $CPUOP: .WORD ACPUOP ;;CPU TYPE, OPTIONS
994 001344 000000 *
995 * BITS 15-11=CPU TYPE
996 * 11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
997 * 11/70=06,PDQ=07,Q=10
998 * BIT 10=REAL TIME CLOCK
999 * BIT 9=FLOATING POINT PROCESSOR
1000 * BIT 8=MEMORY MANAGEMENT
1001 001346 $ETEND:
```

CJKDDA KEF11-A DIAG PART 2
CJKDDA.P11 20-JUN-79 11:22

MACY11 30A(1052) 20-JUN-79 11:39 ¹ ² PAGE 21
APT MAILBOX-ETABLE

SEQ 0021

1002

.MEXIT

```
1003 .SBTTL ERROR POINTER TABLE
1004
1005 ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
1006 ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
1007 ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
1008 ;*NOTE1: IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
1009 ;*NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
1010
1011 ;* EM ;;POINTS TO THE ERROR MESSAGE
1012 ;* DH ;;POINTS TO THE DATA HEADER
1013 ;* DT ;;POINTS TO THE DATA
1014 ;* DF ;;POINTS TO THE DATA FORMAT
1015
1016
1017 001346 $ERRTB:
1018 ;ITEM NUMBER
1019 001346 036423 .WORD EM1
1020 001350 036452 .WORD EM2
1021 001352 036507 .WORD EM3
1022
1023 .SBTTL ACT11 HOOKS
1024
1025 ;*****
1026 ;HOOKS REQUIRED BY ACT11
1027 001354 $SVPC . ;SAVE PC
1028 000046 .46
1029 000046 033042 $ENDAD ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
1030 000052 000052 .-52
1031 000052 000000 .WORD 0 ;;2)SET LOC.52 TO ZERO
1032 001354 .-$VPC ;; RESTORE PC
1033
1034 .SBTTL APT PARAMETER BLOCK
1035
1036 ;*****
1037 ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
1038 ;*****
1038 001354 . $X- ;;SAVE CURRENT LOCATION
1039 000024 =24 ;;SET POWER FAIL TO POINT TO START OF PROGRAM
1040 000024 000200 200 ;;FOR APT START UP
1041 000044 .-44 ;;POINT TO APT INDIRECT ADDRESS PNTR.
1042 000044 001354 $APTHDR ;;POINT TO APT HEADER BLOCK
1043 001354 . $X ;;RESET LOCATION COUNTER
1044
1045 ;*****
1046 ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
1047 ;INTERFACE SPEC.
1048
1048 001354 $APTHD:
1049 001354 000000 $HIBTS: .WORD 0 ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
1050 001356 001316 $MBADR: .WORD $MAIL ;;ADDRESS OF APT MAILBOX (BITS 0-15)
1051 001360 000002 $STMT: .WORD 2 ;;RUN TIM OF LONGEST TEST
1052 001362 000004 $PASTM: .WORD 4 ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
1053 001364 000000 $UNITM: .WORD 0 ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
1054 001366 000014 .WORD $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE (WORDS)
1055
1056
1057 001370 START:
1058 .SBTTL INITIALIZE THE COMMON TAGS
```

```

1059      ;;CLEAR THE COMMON TAGS (%CMTAG) AREA
1060 001370 012706 001100      MOV      #CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
1061 001374 005026             CLR      (R6)+          ;;CLEAR MEMORY LOCATION
1062 001376 022706 001140      CMP      #SWR,R6      ;;DONE?
1063 001402 001374             BNE     -6              ;;LOOP BACK IF NO
1064 001404 012706 001100      MOV      #STACK,SP    ;;SETUP THE STACK POINTER
1065      ;;INITIALIZE A FEW VECTORS
1066 001410 012737 033136 000020 MOV      #SCOPE,@IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
1067 001416 012737 000340 000022 MOV      #340,@IOTVEC+2 ;;LEVEL 7
1068 001424 012737 033416 000030 MOV      #ERROR,@EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
1069 001432 012737 000340 000032 MOV      #340,@EMTVEC+2 ;;LEVEL 7
1070 001440 012737 035552 000034 MOV      #TRAP,@TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
1071 001446 012737 000340 000036 MOV      #340,@TRAPVEC+2;LEVEL 7
1072 001454 012737 035640 000024 MOV      #SPWRDN,@PWRVEC ;;POWER FAILURE VECTOR
1073 001462 012737 000340 000026 MOV      #340,@PWRVEC+2 ;;LEVEL 7
1074 001470 016767 031276 031266 MOV      $ENDCT,$EOPCT ;;SETUP END-OF-PROGRAM COUNTER
1075 001476 005067 177600      CLR      $TIMES       ;;INITIALIZE NUMBER OF ITERATIONS
1076 001502 005067 177576      CLR      $ESCAPE      ;;CLEAR THE ESCAPE ON ERROR ADDRESS
1077 001506 112767 000001 177401 MOVB    #1,$ERMAX     ;;ALLOW ONE ERROR PER TEST
1078      ;;INITIALIZE THE 'T-BIT' TRAP VECTOR. THEN LOAD LOCATION '$RTRN', IN
1079      ;;THE 'END-OF-PASS' ($EOP) ROUTINE, WITH A 'RTI' OR 'RTT'.
1080 001514 012737 033106 000014 MOV      #RTRN,@TBITVEC ;;SET 'T' BIT VECTOR TO RTRN
1081 001522 012737 000340 000016 MOV      #340,@TBITVEC+2 ;;LEVEL 7
1082 001530 012767 000002 031350 MOV      #RTI,$RTRN    ;;SET RTRN TO A RTI
1083 001536 012737 001564 000010 MOV      #65,$RESVEC   ;;TRY TO DO A RTT
1084 001544 005046             CLR      -(SP)        ;;DUMMY PS
1085 001546 012746 001554      MOV      #64,$-(SP)  ;;AND PC
1086 001552 000006             RTT                     ;;TRY THE RTT
1087 001554 012767 000006 031324 64$: MOV      #RTT,$RTRN   ;;RTT IS LEGAL--SET RTRN TO A RTT
1088 001562 000402             BR      66$
1089 001564 062706 000010      65$: ADD      #10,SP    ;;RTT ILLEGAL--CLEAN OFF THE STACK
1090 001570 012737 000012 000010 66$: MOV      #RESVEC+2,@RESVEC ;;RESTORE TRAP CATCHER
1091 001576 005067 031312      CLR      $TBIT       ;;CLEAR 'T' BIT SWITCH
1092 001602 012767 001602 177276 MOV      #.,$LPADR    ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
1093 001610 012767 001610 177272 MOV      #.,$LPERR    ;;SETUP THE ERROR LOOP ADDRESS
1094      ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
1095      ;;EQUAL TO A '-1', SETUP FOR A SOFTWARE SWITCH REGISTER.
1096 001616 013746 000004      MOV      @ERRVEC,-(SP) ;;SAVE ERROR VECTOR
1097 001622 012737 001656 000004 MOV      #67,@ERRVEC  ;;SET UP ERROR VECTOR
1098 001630 012767 177570 177302 MOV      #DSWR,SWR    ;;SETUP FOR A HARDWARE SWICH REGISTER
1099 001636 012767 177570 177276 MOV      #DDISP,DISPLAY ;;AND A HARDWARE DISPLAY REGISTER
1100 001644 022777 177777 177266 CMP      #-1,@SWR     ;;TRY TO REFERENCE HARDWARE SWR
1101 001652 001012             BNE     69$          ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
1102      ;;AND THE HARDWARE SWR IS NOT = -1
1103 001654 000403             BR      68$          ;;BRANCH IF NO TIMEOUT
1104 001656 012716 001664      67$: MOV      #68,$(SP)  ;;SET UP FOR TRAP RETURN
1105 001662 000002             RTI
1106 001664 012767 000176 177246 68$: MOV      #SWREG,SWR   ;;POINT TO SOFTWARE SWR
1107 001672 012767 000174 177242 MOV      #DISPREG,DISPLAY
1108 001700 012637 000004      69$: MOV      (SP)+,@ERRVEC ;;RESTORE ERROR VECTOR
1109
1110 001704 005067 177414      CLR      $PASS       ;;CLEAR PASS COUNT
1111 001710 132767 000200 177421 BITB    #APTSIZE,$ENVM ;;TEST USER SIZE UNDER APT
1112 001716 001403             BEQ     70$          ;;YES,USE NON-APT SWITCH
1113 001720 012767 001340 177212 MOV      #SSWREG,SWR  ;;NO,USE APT SWITCH REGISTER
1114 001726
    
```



```
1115 .SBTTL TYPE PROGRAM NAME
1116 ::TYPE THE NAME OF THE PROGRAM IF FIRST PASS
1117 001726 005227 177777 INC #-1 ;:FIRST TIME?
1118 001732 001051 BNE 71$ ;:BRANCH IF NO
1119 001734 022737 033042 000042 CMF #SENDAD,@#42 ;:ACT-11?
1120 001742 001445 BEQ 71$ ;:BRANCH IF YES
1121 001744 104401 002012 TYPE ,72$ ;:TYPE ASCIZ STRING
1122 .SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
1123 001750 005737 000042 TST @#42 ;:ARE WE RUNNING UNDER XXDP/ACT?
1124 001754 001012 BNE 73$ ;:BRANCH IF YES
1125 001756 126727 177354 000001 CMPB $ENV,#1 ;:ARE WE RUNNING UNDER APT?
1126 001764 001406 BEQ 73$ ;:BRANCH IF YES
1127 001766 026727 177146 000176 CMP SWR,#SWREG ;:SOFTWARE SWITCH REG SELECTED?
1128 001774 001005 BNE 74$ ;:BRANCH IF NO
1129 001776 104406 GTSWR ;:GET SOFT-SWR SETTINGS
1130 002000 000403 BR 74$
1131 002002 112767 000001 177124 73$: MOVB #1,$AUTOB ;:SET AUTO-MODE INDICATOR
1132 002010 74$:
1133 002010 000422 BR 71$ ;:GET OVER THE ASCIZ
1134 ::72$: .ASCIZ <CRLF>*CJKDDA KEF11-A DIAGNOSTIC PART 2*<CRLF>
1135 71$:
1136
1137 002056 LOOP:
1138
1139
1140
1141
1142
1143 ::*****
1144 *TEST 1 STF WITH ILLEGAL ACCUMULATOR TEST
1145 *
1146 *THIS IS A TEST OF THE STF INSTRUCTION USING ILLEGAL ACCUMULATOR 7, MODE 0.
1147 *
1148 ::*****
1149 002056 000004 TST1: SCOPE
1150
1151 002050 0001:
1152 002060 104414 LPFRR ;:SET JP THE LOOP ON ERROR ADDRESS.
1153 002062 005000 CLR R0 ;:SET THE FPS.
1154 002064 170100 LDFPS R0
1155
1156 002066 012737 002124 000244 MOV #000T,@#FPVECT ;:SET UP FOR FP TRAPS.
1157 002074 012737 002102 001236 MOV #1$,@#TMP2
1158
1159 002102 174007 1$: STF AC0,AC7 ;:THIS TEST INSTRUCTION SHOULD
1160 ;:CAUSE A TRAP.
1161
1162 ;REPORT FAILURE OF USE OF ILLEGAL ACCUMULATOR 7 TO CAUSE AN FPP TRAP.
1163 0002:
1164 002104 170200 STFPS R0 ;:GET FPS.
1165 002106 010037 001240 MOV R0,@#TMP3
1166 002112 170300 STST R0 ;:GET FEC.
1167 002114 010037 001242 MOV R0,@#TMP4
1168 002120 104001 3$: ERROR 1 ;:STF WITH ILLEGAL ACCUMULATOR, MODE
1169 ;:0, DIDN'T TRAP. SI 765 TO ST 537.
1170 002122 000434 BR 000DONE
```

```
1171
1172 ;TRAP TO OOOT, HERE, WHEN THE EXPECTED ERROR OCCURS.
1173 002124 011600 OOOT: MOV (SP),R0 ;MAKE SURE THE ERROR OCCURRED
1174 002126 022700 002104 CMF #0002,R0 ;AT THE CORRECT ADDRESS.
1175 002132 001402 BEQ 0003 ;BRANCH IF TRAP ADDRESS CORRECT.
1176 002134 000137 036140 JMP @WFPSPUR ;IF INCORRECT GO REPORT SPURIOUS
1177 ;FP TRAP.
1178
1179 002140 170204 0003: STFPS R4 ;GET FPS.
1180 002142 170305 STST R5 ;GET FEC.
1181 002144 010437 001240 MOV R4,@#STMP3 ;SAVE DATA INCASE OF ERROR.
1182 002150 010537 001242 MOV R5,@#STMP4
1183 002154 012702 100000 MOV #100000,R2 ;EXPECTED FPS
1184 002160 012703 000002 MGV #2,R3 ;EXPECTED FEC
1185 002164 010237 001244 MOV R2,@#STMP5
1186 002170 010337 001246 MOV R3,@#STMP6
1187 002174 022626 CMP (SP)+,(SP)+ ;RESET THE STACK.
1188
1189 002176 020204 CMP R2,R4 ;WAS FPS CORRECT?
1190 002200 001402 BEQ 0004 ;BRANCH IF YES.
1191 ;OTHERWISE REPORT FPS INCORRECTLY
1192 002202 104001 1$: ERROR 1 ;SET AFTER USE OF ILLEGAL ACC.
1193 002204 000403 BR 000DONE
1194
1195 002206 020305 0004: CMP R3,R5 ;WAS THE FEC CORRECT?
1196 002210 001401 BEQ 000DONE ;BRANCH IF CORRECT.
1197 ;OTHERWISE REPORT INCORRECT FEC
1198 002212 104001 1$: ERROR 1 ;AFTER USE OF ILLEGAL ACC.
1199
1200 002214 000DONE:
1201 002214 104413 RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
1202 ;SEE IF THE USER HAS EXPRESSED
1203 ;THE DESIRE TO CHANGE THE SOFTWARE
1204 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
1205 ;THE USER TYPED CONTROL G?).
1206
1207
1208
1209
1210 ;:*****
1211 ;*TEST 2 FDST MODE 1, FLOATING MODE, TEST
1212 ;*
1213 ;*THIS IS A TEST OF THE STF INSTRUCTION USING FDST MODE 1.
1214 ;*
1215 ;:*****
1216 002216 000004 TST2: SCOPE
1217
1218 002220 PPP1:
1219 002220 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
1220
1221 002222 012700 177777 MOV #-1,R0 ;SET UP A BACKGROUND PATTERN IN THE
1222 002226 012701 002356 MOV #PPBF0,R1 ;INPUT BUFFER.
1223 002232 012702 000014 MOV #14,R2
1224 002236 010021 PPP2: MOV R0,(R1)+
1225 002240 077202 SOB R2,PPP2
1226
```

```

1227 002242 012700 000200      MOV      #200,R0          ;SET FD MODE.
1228 002246 170100      LDFPS   R0
1229 002250 012700 002406      MOV      #PPPTP1,R0      ;PUT TEST DATA INTO ACO.
1230 002254 172410      LDD     (R0),AC0
1231
1232 002256 012700 002372      MOV      #PPPF1,R0       ;FDST ADDRESS.
1233 002262 005002      CLR     R2               ;CLEAR THE FPS.
1234 002264 170102      LDFPS   R2
1235 002266 012737 002300 001236      MOV      #PPP3,@#$TMP2
1236 002274 010037 001240      MOV      R0,@#$TMP3
1237
1238 002300 174010      PPP3:   STF     ACO,(R0)   ;TEST INSTRUCTION.
1239
1240 002302 022700 002372      CMP      #PPPF1,R0       ;WAS R0 MODIFIED DURING EXECUTION?
1241 002306 001404      BEQ     PPP4             ;BRANCH IF R0 NOT MODIFIED, CORRECT.
1242
1243 002310 010037 001242      MOV      R0,@#$TMP4      ;OTHERWISE REPORT ERROR, R0 MODIFIED.
1244 002314 104001      1$:     ERROR   1
1245 002316 000456      BR      PPPDONE         ;GO TO NEXT TEST.
1246
1247 002320 012700 002372      PPP4:   MOV      #PPPF1,R0 ;CHECK THE DATA IN THE OUTPUT BUFFER.
1248 002324 012701 002406      MOV      #PPPTP1,R1
1249 002330 022021      CMP      (R0)+,(R1)+
1250 002332 001031      BNE     PPP10           ;BRANCH IF INCORRECT.
1251 002334 022011      CMP      (R0)+,(R1)
1252 002336 001027      BNE     PPP10           ;BRANCH IF INCORRECT.
1253 002340 022720 177777      CMP      #-1,(R0)+      ;WAS FLOATING MODE USED?
1254 002344 001034      BNE     PPP15           ;BRANCH IF NOT.
1255 002346 022710 177777      CMP      #-1,(R0)
1256 002352 001031      BNE     PPP15
1257 002354 000437      BR      PPPDONE ;GO TO NEXT TEST.
1258
1259 002356 177777 177777 177777 PPPBF0: .WORD  -1,-1,-1,-1,-1,-1
1260 002364 177777 177777 177777
1261
1262 002372 177777 177777 177777 PPPBF1: .WORD  -1,-1,-1,-1,-1,-1
1263 002400 177777 177777 177777
1264
1265 002406 123456 023456      PPPTP1: .WORD  123456,23456
1266 002412 034567 045671      .WORD  34567,45671
1267
1268      ;REPORT DATA IN OUT PUT BUFFER INCORRECT.
1269 002416 012737 002406 001242 PPP10: MOV      #PPPTP1,@#$TMP4
1270 002424 012737 002372 001240      MOV      #PPPF1,@#$TMP3
1271 002432 104001      1$:     ERROR   1          ;BAD DATA.
1272 002434 000407      BR      PPPDONE
1273
1274      ;REPORT FLOATING MODE NOT USED, BUT FD FAILED.
1275 002436 012737 002406 001242 PPP15: MOV      #PPPTP1,@#$TMP4
1276 002444 012737 002372 001240      MOV      #PPPF1,@#$TMP3
1277 002452 104001      1$:     ERROR   1          ;ST 707 TO 245 INTO 244 (BUT FD).
1278
1279 002454      PPPDONE:
1280 002454 104413      RSETUP
1281
1282      ;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?).

```
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295 002456 000004
1296
1297
1298 002460
1299 002460 104414
1300
1301 002462 012700 177777
1302 002466 012701 002620
1303 002472 012702 000014
1304 002476 010021
1305 002500 077202
1306
1307 002502 012700 000200
1308 002506 170100
1309 002510 012700 002650
1310 002514 172410
1311
1312 002516 012700 002634
1313 002522 005002
1314 002524 170102
1315 002526 012737 002534 001236
1316
1317 002534 174020
1318
1319 002536 022700 002640
1320
1321 002542 001407
1322 002544 010037 001242
1323 002550 012737 002640 001240
1324 002556 104001
1325 002560 000526
1326 002562 012700 002634
1327 002566 012701 002650
1328 002572 022021
1329 002574 001031
1330 002576 022021
1331 002600 001027
1332 002602 022027 177777
1333 002606 001024
1334 002610 022027 177777
1335 002614 001021
1336 002616 000430
1337 002620 177777 177777 177777
1338 002626 177777 177777 177777
```

```
*****
*TEST 3          FDST MODE 2 TEST
*
*THIS IS A TEST OF BOTH STF AND STD WITH FDST MODE 2.
*
*****
TST3:  SCOPE

;FIRST TEST STF.
QQQ1:  LPERR                      ;SET UP THE LOOP ON ERROR ADDRESS.

      MOV    #-1,R0              ;SET UP THE OUTPUT BUFFER.
      MOV    #QQQBF0,R1
      MOV    #14,R2
QQQ2:  MOV    R0,(R1)+
      SOB    R2,QQQ2

      MOV    #200,R0             ;SET FD MODE.
      LDFPS  R0
      MOV    #QQQTP1,R0         ;SETUP ACO.
      LDD    (R0),ACO

      MOV    #QQQBF1,R0         ;FDST ADDRESS.
      CLR    R2
      LDFPS  R2
      MOV    #QQQ3,@#$TMP2     ;SET FPS.

QQQ3:  STF    ACO,(R0)+         ;TEST INSTRUCTION.

      CMP    #QQQBF1+4,R0      ;WAS R0 INCREMENTED BY 4 PROPERLY?

      BEQ    QQQ4              ;BRANCH IF R0 CORRECT.
      MOV    R0,@#$TMP4        ;REPORT R0 INCORRECT AFTER FDST MODE 2.
      MOV    #QQQBF1+4,@#$TMP5
1$:    ERROR 1                  ;BAD CONSTANT USED OR DIDN'T GO 527 TO 642
      BR    QQQDONE

QQQ4:  MOV    #QQQBF1,R0        ;WAS THE OUTPUT DATA CORRECT?
      MOV    #QQQTP1,R1
      CMP    (R0)+,(R1)+
      BNE   QQQ10              ;BRANCH IF INCORRECT.
      CMP    (R0)+,(R1)+
      BNE   QQQ10              ;BRANCH IF INCORRECT.
      CMP    (R0)+,#-1         ;SEE IF ANY OTHER DATA BUFFER WORDS WERE MODIFIED.
      BNE   QQQ10              ;BRANCH IF INCORRECT.
      CMP    (R0)+,#-1
      BNE   QQQ10              ;BRANCH IF INCORRECT.
      BR    QQQ20

QQQBF0: .WORD -1,-1,-1,-1,-1,-1
```

```

1339 002634 177777 177777 177777 QQQBF1: .WORD -1,-1,-1,-1,-1,-1
1340 002642 177777 177777 177777
1341 002650 076543 QQQTP1: 76543
1342 002652 065432 65432
1343 002654 054321 54321
1344 002656 043210 43210
1345 ;REPORT OUTPUT DATA INCORRECT:
1346 002660 012737 002650 001240 QQQ10: MOV #QQQTP1,@#STMP3
1347 002666 012737 002634 001242 MOV #QQQBF1,@#STMP4
1348 002670 104001 1$: ERROR 1 ;BAD DATA
1349 002676 000457 BR QQQDONE
1350
1351 ;NOW TEST STD MODE 2.
1352
1353 002700 QQQ20:
1354 002700 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
1355 002702 012700 002620 MOV #QQQBF0,R0 ;SET UP DEFAULT INPUT DATA BUFFER.
1356 002706 010001 MOV R0,R1
1357 002710 012702 000014 MOV #14,R2
1358 002714 010021 QQQ22: MOV R0,(R1)+
1359 002716 077202 SOB R2,QQQ22
1360 002720 012700 000200 MOV #200,R0 ;ENTER FLOATING DOUBLE MODE.
1361 002724 170100 LDFPS R0
1362 002726 012700 002650 MOV #QQQTP1,R0 ;LOAD ACO.
1363 002732 172410 LDD (R0),ACO
1364 002734 012700 002634 MOV #QQQBF1,R0 ;SET DESTINATION ADDRESS.
1365 002740 012737 002746 001236 QQQ23: MOV #QQQ23,@#STMP2
1366 002746 174020 QQQ23: STD ACO,(R0)+ ;TEST INSTRUCTION.
1367 002750 022700 002644 CMP #QQQBF1+10,R0 ;WAS R0 INCREMENTED BY 10 CORRECTLY?
1368 002754 001407 BEQ QQQ24 ;BRANCH IF CORRECT.
1369 002756 010037 001242 MOV R0,@#STMP4 ;REPORT R0 INCORRECTLY INCREMENTED.
1370 002762 012737 002644 001240 QQQ24: MOV #QQQBF1+10,@#STMP3
1371 002770 104001 1$: ERROR 1 ;DO NOT INCREM BY 10 BAD CONSTANT
1372 002772 000421 BR QQQDONE
1373 002774 012700 002634 QQQ24: MOV #QQQBF1,R0 ;DID THE DATA REACH THE OUTPUT BUFFER CORRECTLY?
1374 003000 012701 002650 MOV #QQQTP1,R1
1375 003004 012702 000004 MOV #4,R2
1376 003010 022021 1$: CMP (R0)+,(R1)+
1377 003012 001002 BNE QQQ25 ;BRANCH IF INCORRECT.
1378 003014 077203 SOB R2,1$
1379 003016 000407 BR QQQDONE
1380 ;REPORT DATA INCORRECT.
1381 003020 012737 002650 001240 QQQ25: MOV #QQQTP1,@#STMP3
1382 003026 012737 002634 001242 MOV #QQQBF1,@#STMP4
1383 003034 104001 1$: ERROR 1 ;BAD DATA
1384 003036 QQQDONE:
1385 003036 104413 RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
1386 ;SEE IF THE USER HAS EXPRESSED
1387 ;THE DESIRE TO CHANGE THE SOFTWARE
1388 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
1389 ;THE USER TYPED CONTROL G?).
1390
1391 ;*****
1392 ;*TEST 4 FDST MODE 2, WITH GR7, TEST
1393 ;*
1394 ;*THIS IS A TEST OF STF WITH GR7 MODE 2 OR IMMEDIATE MODE.
    
```

```
1395
1396
1397 003040 000004
1398
1399 003042
1400 003042 104414
1401 003044 012700 003122
1402 003050 012701 003170
1403 003054 012702 000004
1404 003060 012021
1405 003062 077202
1406 003064 012700 000200
1407 003070 170100
1408 003072 012700 003200
1409 003076 172410
1410 003100 012737 003220 000004
1411 003106 012737 003120 001236
1412 003114 005001
1413 003116 005004
1414
1415
1416
1417
1418
1419
1420 003120 174027
1421 003122 005201
1422 003124 005201
1423 003126 005201
1424 003130 005201
1425 003132 012700 003210
1426 003136 012702 003122
1427 003142 012703 000004
1428 003146 022022
1429 003150 001051
1430 003152 077303
1431 003154 005704
1432 003156 001056
1433 003160 022701 000003
1434 003164 001053
1435 003166 000474
1436
1437 003170 005201
1438 003172 005201
1439 003174 005201
1440 003176 005201
1441
1442 003200 005204
1443 003202 005204
1444 003204 005204
1445 003206 005204
1446
1447 003210 005204
1448 003212 005201
1449 003214 005201
1450 003216 005201
```

```

:*****
TST4: SCOPE
RRR1:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
MOV #RRR3,R0 ;SET UP THE DATA BUFFER FOLLOWING THE TEST INSTRUCTION.
MOV #RRRTP1,R1
MOV #4,R2
1$: MOV (R0)+,(R1)+
SOB R2,1$
MOV #200,R0 ;ENTER FLOATING DOUBLE MODE.
LDFPS R0
MOV #RRRTP2,R0 ;SET UP ACO.
LDD (R0),ACO
MOV #RRR10,@#ERRVECT ;SET UP FOR AN ODD ADDRESS.
MOV #RRR2,@#$TMP2
CLR R1
CLR R4
;THIS IS THE TEST INSTRUCTION. IT SHOULD MODIFY THE FIRST LOCATION
;AFTER IT TO BE AN INCREMENT R4, INC R4, INSTRUCTION INSTEAD
;OF AN INCREMENT R1 INSTRUCTION. THE INCREMENT R4 SHOULD NOT BE
;EXECUTED SINCE THE PC SHOULD BE INCREMENTED BY TWO DURING IMMEDIATE
;MODE ADDRESSING. THUS AFTER THE EXECUTION OF THE NEXT 5 INSTRUCTIONS
;R1 SHOULD CONTAIN 3 AND R4 SHOULD CONTAIN 0.
RRR2: STD ACO,(R7)+ ;TEST INSTRUCTION.
RRR3: INC R1 ;THE STD INSTRUCTION SHOULD CHANGE THIS TO INC R4.
INC R1
INC R1
INC R1
MOV #RRREXP,R0 ;SEE IF THE DATA WAS OUTPUT CORRECTLY.
MOV #RRR3,R2
MOV #4,R3
RRR4: CMP (R0)+,(R2)+
BNE RRR25 ;BRANCH IF INCORRECT.
SOB R3,RRR4
TST R4 ;MAKE SURE R4 IS 0.
BNE RRR15 ;BRANCH IF R4 IS INCORRECT.
CMP #3,R1 ;SEE IF R1 IS CORRECT.
BNE RRR15 ;BRANCH IF R1 IS INCORRECT.
BR RRRDONE
;THESE ARE TEST DATA PATTERNS USED TO SET UP THE OUTPUT BUFFER AT RRR3.
RRRTP1: INC R1
INC R1
INC R1
INC R1
;THIS IS THE DATA PUT IN ACO BEFORE EXECUTION OF THE STD.
RRRTP2: INC R4
INC R4
INC R4
INC R4
;THIS IS THE EXPECTED DATA AT RRR3 AFTER EXECUTION OF THE STD.
RRREXP: INC R4
INC R1
INC R1
INC R1
```

```
1451 ;IF A FAILURE IN THE FDST FLOWS RESULTS IN AN ODD ADDRESS TRAP THROUGH
1452 ;4 TO HERE:
1453 003220 011602 RRR10: MOV (SP),R2 ;SEE IF THE TRAP WAS BECAUSE OF AN ODD ADDRESS.
1454 003222 032702 000001 BIT #1,R2
1455 003226 001005 BNE RRR11 ;BRANCH IF YES.
1456 003230 020227 003124 CMP R2,#RRR3+2 ;SEE IF THE TRAP OCCURRED AT THE TEST INSTRUCTION.
1457 003234 001412 BEQ RRR12 ;BRANCH IF YES.
1458 003236 000137 036172 JMP @#CPSPUR ;OTHERWISE REPORT A SPURIOUS TRAP THROUGH VECTOR 4.
1459 ;REPORT A FAILURE IN THE FDST FLOWS RESULTED IN AN ODD ADDRESS TRAP.
1460 003242 010237 001236 RRR11: MOV R2,@#STMP2
1461 003246 012737 003124 001240 MOV #RRR3+2,@#STMP3
1462 003254 022626 CMP (SP)+,(SP)+
1463 003256 104001 1$: ERROR 1 ;BAD CONSTANT #2 + PC ODD ADDR.
1464 003260 000437 BR RRRDONE
1465 003262 010237 001236 RRR12: MOV R2,@#STMP2
1466 003266 022626 CMP (SP)+,(SP)+
1467 003270 104001 1$: ERROR 1 ;ODD ADDRESS TRAP
1468 003272 000432 BR RRRDONE ;WRONG MODE USED.
1469
1470 ;REPORT DATA INCORRECT:
1471 003274 012737 003122 001240 RRR25: MOV #RRR3,@#STMP3
1472 003302 012737 003210 001242 MOV #RRREXP,@#STMP4
1473 003310 104001 1$: ERROR 1 ;BAD DATA BUT GR7 FAIL
1474 003312 000422 BR RRRDONE
1475
1476 ;REPORT PC INCORRECT MODIFIED DURING THE EXECUTION OF FDST IMMEDIATE
1477 ;MODE. THE PC SHOULD HAVE BEEN INCREMENTED BY 2 BUT IT WASN'T.
1478 ;USE R1 AND R4 TO COMPUTE THE ACTUAL ACTION THAT WAS TAKEN ON THE PC.
1479 003314 012737 003124 001240 RRR15: MOV #RRR3+2,@#STMP3
1480 003322 005704 TST R4 ;IS R4 CLEAR.
1481 003324 001404 BEQ 1$
1482 003326 012737 003122 001242 MOV #RRR3,@#STMP4
1483 003334 000410 BR 2$
1484 003336 012702 003124 1$: MOV #RRR3+2,R2
1485 003342 062701 177775 ADD #-3,R1
1486 003346 006301 ASL R1
1487 003350 160102 SUB R1,R2
1488 003352 010237 001242 MOV R2,@#STMP4
1489 003356 2$:
1490 003356 104001 3$: ERROR 1 ;BAD CONSTANT PC+
1491 003360 RRRDONE:
1492 003360 104413 RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
1493 ;SEE IF THE USER HAS EXPRESSED
1494 ;THE DESIRE TO CHANGE THE SOFTWARE
1495 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
1496 ;THE USER TYPED CONTROL G?).
1497
1498 ;*****
1499 ;*TEST 5 FDST MODE 4 TEST
1500 ;*
1501 ;*THIS IS A TEST OF STD WITH FDST MODE 4.
1502 ;*
1503 ;*****
1504 003362 000004 TST5: SCOPE
1505
1506 003364 SSS1:
```

```
1507 003364 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
1508 003366 012700 177777 MOV #-1,R0 ;SET UP THE OUTPUT BUFFER.
1509 003372 012701 003522 MOV #SSSBF0,R1
1510 003376 012702 000010 MOV #10,R2
1511 003402 010021 1$: MOV R0,(R1)+
1512 003404 077202 SOB R2,1$
1513 003406 012700 000200 MC #200,R0 ;ENTER FLOATING DOUBLE MODE.
1514 003412 170100 LDFPS R0
1515 003414 012700 003542 MOV #SSSTP1,R0 ;SET UP ACO.
1516 003420 172410 LDD (R0),AC0
1517 003422 012737 003562 000004 MOV #SSS10,@#ERRVECT ;SET UP FOR A TRAP TO 4.
1518 003430 012737 003442 001236 MOV #SSS2,@#STMP2
1519 003436 012700 003532 MOV #SSSA1,R0 ;SET UP THE DESTINATION ADDRESS.
1520
1521 003442 174040 SSS2: STD ACO,-(R0) ;TEST INSTRUCTION.
1522 003444 005201 INC R1
1523 003446 020027 003522 CMP R0,#SSSBF0 ;SEE IF R0 WAS DECREMENTED PROPERLY.
1524 003452 001060 BNE SSS15 ;BRANCH IF R0 IS INCORRECT.
1525 003454 012700 003522 MOV #SSSBF0,R0 ;WAS THE OUTPUT DATA CORRECT?
1526 003460 012701 003542 MOV #SSSTP1,R1
1527 003464 012702 000004 MOV #4,R2
1528 003470 022021 1$: CMP (R0)+,(R1)+
1529 003472 001057 BNE SSS20 ;BRANCH IF INCORRECT.
1530 003474 077203 SOB R2,1$
1531 003476 012700 177777 MOV #-1,R0 ;IS THE REST OF THE OUTPUT BUFFER CORRECT, -1?
1532 003502 012701 003532 MOV #SSSA1,R1
1533 003506 012702 000004 MOV #4,R2
1534 003512 020021 2$: CMP R0,(R1)+
1535 003514 001056 BNE SSS25 ;BRANCH IF INCORRECT.
1536 003516 077203 SOB R2,2$
1537 003520 000463 BR SSSDONE
1538
1539 ;THIS IS THE OUTPUT DATA BUFFER.
1540 003522 177777 SSSBF0: -1
1541 003524 177777 -1
1542 003526 177777 -1
1543 003530 177777 -1
1544 003532 177777 SSSA1: -1
1545 003534 177777 -1
1546 003536 177777 -1
1547 003540 177777 -1
1548
1549 ;THIS IS THE TEST DATA LOADED INTO ACO:
1550 003542 147250 SSSTP1: 147250
1551 003544 036147 36147
1552 003546 025036 25036
1553 003550 147250 147250
1554 003552 177777 SSSTP2: -1
1555 003554 177777 -1
1556 003556 177777 -1
1557 003560 177777 -1
1558
1559 ;IF AN ODD ADDRESS TRAP OCCURS COME HERE:
1560 003562 011600 SSS10: MOV (SP),R0 ;SEE IF THE TRAP ACCURRED ON THE TEST INSTRUCTION.
1561 003564 020027 003444 CMP R0,#SSS2+2
1562 003570 001405 BEQ SSS11 ;BRANCH IF YES.
```



```
1563 003572 020027 003446          CMP      R0,#SSS2+4
1564 003576 001402                   BEQ      SSS11          ;BRANCH IF YES.
1565 003600 000137 036172          JMP      @#CPSPUR      ;OTHERWISE GO REPORT A SPURIOUS TRAP THROUGH 4.
1566                                     ;REPORT FAILURE IN FDST FLOWS RESULTED IN AN ODD ADDRFS.
1567 003604 010037 001236          SSS11:  MOV      R0,@#STMP2
1568 003610 104001                   2$:      ERROR    1          ;FDST FORK X ODD AD RES.
1569 003612 000426                   BR       SSSDONE
1570
1571                                     ;REPORT R0 INCORRECTLY DECREMENTED.
1572 003614 010037 001242          SSS15:  MOV      R0,@#STMP4
1573 003620 012737 003522 001240    MOV      #SSSBFO,@#STMP3
1574 003626 104001                   1$:      ERROR    1          ;R0 NOT DECRE PROP
1575 003630 000417                   BR       SSSDONE
1576
1577                                     ;REPORT OUTPUT DATA INCORRECT:
1578 003632 012737 003522 001240    SSS20:  MOV      #SSSBFO,@#STMP3
1579 003640 012737 003542 001242    MOV      #SSSTP1,@#STMP4
1580 003646 104001                   1$:      ERROR    1          ;BAD DATA
1581 003650 000407                   BR       SSSDONE
1582 003652 012737 003532 001242    SSS25:  MOV      #SSSA1,@#STMP4
1583 003660 012737 003552 001240    MOV      #SSSTP2,@#STMP3
1584 003666 104001                   1$:      ERROR    1          ;DATA BAD OUTSIDE TARGET AREA
1585 003670                                     SSSDONE:
1586 003670 104413                                     RSETUP          ;GO INITIALIZE THE FPS AND STACK; AND
1587                                     ;SEE IF THE USER HAS EXPRESSED
1588                                     ;THE DESIRE TO CHANGE THE SOFTWARE
1589                                     ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
1590                                     ;THE USER TYPED CONTROL G?).
1591
1592                                     ;*****
1593                                     ;*TEST 6          FDST MODE 3 TEST
1594                                     ;*
1595                                     ;*THIS IS A TEST OF FDST MODE 3 USING STD.
1596                                     ;*
1597                                     ;*****
1598 003672 000004          TST6:   SCOPE
1599
1600 003674          TTT1:
1601 003674 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
1602 003676 012701 004014    MOV      #TTTBFO,R1          ;SET UP THE OUTPUT DATA BUFFER.
1603 003702 012700 177777    MOV      #-1,R0
1604 003706 012702 000013    MOV      #13,R2
1605 003712 010021          1$:      MOV      R0,(R1)+
1606 003714 077202          SOB      R2,1$
1607 003716 012737 004014 004030    MOV      #TTTBFO,@#TTTA2
1608 003724 012700 000200          MOV      #200,R0          ;ENTER DOUBLE FLOATING MODE.
1609 003730 170100          LDFPS     R0
1610 003732 012700 004040          MOV      #TTTTP1,R0          ;SET UP ACO.
1611 003736 172410          LDD      (R0),AC0
1612 003740 012737 004050 000004    MOV      #TTTT10,@#ERRVECT ;SET UP FOR TRAPS TO 4.
1613 003746 016737 000006 001236    MOV      TTT2,@#STMP2
1614 003754 012700 004030          MOV      #TTTA2,R0          ;SET UP THE DESTINATION ADDRESS.
1615
1616 003760 174030          TTT2:   STD      ACO,@(R0)+  ;TEST INSTRUCTION.
1617
1618 003762 020027 004032          CMP      R0,#TTTA2+2          ;SEE IF R0 WAS INCREMENTED CORRECTLY.
```

```
1619 003766 001046          BNE      TTT15          ;BRANCH IF INCORRECT.
1620 003770 012701 004014    MOV      #TTTBFO,R1    ;CHECK THE OUTPUT DATA BUFFER.
1621 003774 012702 004040    MOV      #TTTTP1,R2
1622 004000 012703 000004    MOV      #4,R3
1623 004004 022122          TTT3:   CMP      (R1)+,(R2)+
1624 004006 001045          BNE      TTT20          ;BRANCH IF NOT CORRECT.
1625 004010 077303          SOB     R3,TTT3
1626 004012 000452          BR      TTTDONE
1627
1628          ;THIS IS THHE OUTPUT DATA BUFFER:
1629 004014 177777          TTTBFO: -1
1630 004016 177777          -1
1631 004020 177777          -1
1632 004022 177777          -1
1633 004024 177777          -1
1634 004026 177777          TTTA1:  -1
1635 004030 004014          TTTA2:  TTTBFO
1636 004032 177777          TTTA3:  -1
1637 004034 177777          -1
1638 004036 177777          -1
1639 004040 101213          TTTTP1: 101213
1640 004042 141516          141516
1641 004044 071727          71727
1642 004046 037475          37475
1643
1644          ;TRAP THROUGH VECTOR 4 TO HERE.
1645 004050 011602          TTT10:  MOV      (SP),R2    ;SEE IF THE TRAP ADDRESS IS THAT OF THE TEST INSTRUCTION
1646 004052 020227 003762    CMP      R2,#TTT2+2
1647 004056 001405          BEQ     TTT11          ;BRANCH IF YES.
1648 004060 020227 003764    CMP      R2,#TTT2+4
1649 004064 001402          BEQ     TTT11          ;BRANCH IF YES.
1650 004066 000137 036172    JMP     @#CPSPUR      ;OTHERWISE GO REPORT A SPURIOUS TRAP TO 4.
1651
1652          ;REPORT A FAILURE IN THE FDST FLOWS RESULTED IN AN ODD ADDRESS TRAP.
1653 004072 010237 001236          TTT11:  MOV      R2,@#STMP2
1654 004076 022626          CMP     (SP)+,(SP)+
1655 004100 104001          1$:    ERROR   1          ;SET FDST X ODD ADR
1656 004102 000416          BR      TTTDONE
1657
1658          ;REPORT R0 INCORRECT:
1659 004104 010037 001242          TTT15:  MOV      R0,@#STMP4
1660 004110 012737 004032 001240    MOV     #TTTA2+2,@#STMP3
1661 004116 104001          1$:    ERROR   1          ;R0 NOT INCREMENT PROPERLY
1662 004120 000407          BR      TTTDONE
1663
1664          ;REPORT INCORRECT OUTPUT DATA:
1665 004122 012737 004014 001240          TTT20:  MOV      #TTTBFO,@#STMP3
1666 004130 012737 004040 001242    MOV     #TTTTP1,@#STMP4
1667 004136 104001          1$:    ERROR   1          ;BAD DATA
1668 004140          TTTDONE:
1669 004140 104413          RSETUP          ;GO INITIALIZE THE FPS AND STACK; AND
1670          ;SEE IF THE USER HAS EXPRESSED
1671          ;THE DESIRE TO CHANGE THE SOFTWARE
1672          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
1673          ;THE USER TYPED CONTROL G?).
1674
```

```
1675
1676
1677
1678
1679
1680
1681 004142 000004
1682
1683 004144
1684 004144 104414
1685 004146 012701 004264
1686 004152 012700 177777
1687 004156 012702 000013
1688 004162 010021
1689 004164 077202
1690 004166 012737 004264 004276
1691 004174 012700 000200
1692 004200 170100
1693 004202 012700 004310
1694 004206 172410
1695 004210 012737 004320 000004
1696 004216 016737 000006 001236
1697 004224 012700 004300
1698 004230 174050
1699 004232 020027 004276
1700 004236 001046
1701 004240 012701 004264
1702 004244 012702 004310
1703 004250 012703 000004
1704 004254 022122
1705 004256 001045
1706 004260 077303
1707 004262 000452
1708
1709
1710 004264 177777
1711 004266 177777
1712 004270 177777
1713 004272 177777
1714 004274 177777
1715 004276 004264
1716 004300 177777
1717 004302 177777
1718 004304 177777
1719 004306 177777
1720 004310 020212
1721 004312 023242
1722 004314 026273
1723 004316 031323
1724
1725
1726 004320 011602
1727 004322 020227 004232
1728 004326 001405
1729 004330 020227 004234
1730 004334 001402

:*****
:*TEST 7 FDST MODE 5 TEST
:*
:*THIS IS A TEST OF FDST MODE 5 USING STD.
:*
:*****
TST7: SCOPE

UUU1:
      LPERR ;SET UP THE LOGP ON ERROR ADDRESS.
      MOV #UUUBF0,R1 ;SET UP THE OUTPUT DATA BUFFER.
      MOV #-1,R0
      MOV #13,R2
1$:   MOV R0,(R1)+
      SOB R2,1$
      MOV #UUUBF0,@#UUUA1
      MOV #200,R0 ;ENTER DOUBLE FLOATING MODE.
      LDFPS R0
      MOV #UUUTP1,R0 ;SET UP ACO.
      LDD (R0),AC0
      MOV #UUU10,@#ERRVECT ;GET READY FOR ANY TRAPS TO 4.
      MOV UUU2,@#STMP2
      MOV #UUUA2,R0 ;SET UP THE DESTINATION ADDRESS.
UUU2: STD ACO,@-(R0) ;TEST INSTRUCTION.
      CMP R0,#UUUA2-2 ;WAS R0 DECREMENTED PROPERLY?
      BNE UUU15 ;BRANCH IF R0 IS INCORRECT.
      MOV #UUUBF0,R1 ;WAS THE DATA OUTPUT CORRECTLY?
      MOV #UUUTP1,R2
      MOV #4,R3
UUU3: CMP (R1)+,(R2)+
      BNE UUU20 ;BRANCH IF DATA IS INCORRECT.
      SOB R3,UUU3
      BR UUUDONE

;THIS IS THE OUTPUT DATA BUFFER
UUUBF0: -1
      -1
      -1
      -1
      -1
UUUA1: UUUBF0
UUUA2: -1
UUUA3: -1
      -1
      -1
UUUTP1: 20212
      23242
      26273
      031323

;IF A TRAP TO 4 OCCURS COME HERE.
UUU10: MOV (SP),R2 ;SEE IF THE TRAP OCCURRED ON THE TEST INSTRUCTION.
      CMP R2,#UUU2+2
      BEQ UUU11 ;BRANCH IF YES.
      CMP R2,#UUU2+4
      BEQ UUU11 ;BRANCH IF YES.
```

```
1731 004336 000137 036172          JMP @#CPSPUR ;OTHERWISE REPORT A SPURIOUS TRAP TO 4.
1732          ;REPORT FAILURE OF FDST RESULTED IN AN ODD ADDRESS TRAP TO 4.
1733 004342 010237 001236          UUU11: MOV R2,@#STMP2
1734 004346 022626          CMP (SP)+,(SP)+
1735 004350 104001          1$: ERROR 1 ;BET FDST X ODD ADR
1736 004352 000416          BR UUUDONE
1737
1738          ;REPORT R0 INCORRECT.
1739 004354 010037 001242          UUU15: MOV R0,@#STMP4
1740 004360 012737 004302 001240      MOV #UUUA2+2,@#STMP3
1741 004366 104001          1$: ERROR 1 ;R0 NOT INCREMENT PROPERLY
1742 004370 000407          BR UUUDONE
1743
1744          ;REPORT BAD DATA.
1745 004372 012737 004264 001242      UUU20: MOV #UUUBF0,@#STMP4
1746 004400 012737 004310 001240      MOV #UUUTP1,@#STMP3
1747 004406 104001          1$: ERROR 1 ;BAD DATA
1748 004410          UUUDONE:
1749 004410 104413          RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
1750          ;SEE IF THE USER HAS EXPRESSED
1751          ;THE DESIRE TO CHANGE THE SOFTWARE
1752          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
1753          ;THE USER TYPED CONTROL G?).
1754
1755          ;*****
1756          ;*TEST 10 FDST MODE 6, INDEX MODE, TEST
1757          ;*
1758          ;*THIS IS A TEST OF FDST MODE 6, INDEX MODE, USING STD.
1759          ;*
1760          ;*****
1761 004412 000004          TST10: SCOPE
1762
1763          VVV1:
1764 004414 104414          LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
1765 004416 012700 000200      MOV #200,R0 ;ENTER DOUBLE FLOATING MODE.
1766 004422 170100          LDFPS R0
1767 004424 012701 004534      MOV #VVVBF0,R1 ;SET UP THE OUT PUT DATA BUFFER.
1768 004430 012700 177777      MOV #-1,R0
1769 004434 012702 000004      MOV #4,R2
1770 004440 010021          1$: MOV R0,(R1)+
1771 004442 077202          SOB R2,1$
1772 004444 012737 004554 000004      MOV #VVV10,@#ERRVECT ;SET UP VECTOR 4 INCASE OF ERROR.
1773 004452 012700 004544      MOV #VVVTP1,R0 ;SET UP ACO.
1774 004456 172410          LDD (R0),AC0
1775 004460 012737 004476 001236      MOV #VVV2,@#STMP2
1776 004466 012700 176633      MOV #VVVBF0-5701,R0 ;SET UP THE DESTINATION ADDRESS.
1777 004472 012701 000001      MOV #1,R1
1778 004476 174060 005701          VVV2: STD AC0,5701(R0) ;TEST INSTRUCTION.
1779
1780          CMP R0,#VVVBF0-5701 ;SEE IF R0 WAS MODIFIED.
1781 004506 001040          BNE VVV15 ;BRANCH IF INCORRECT.
1782 004510 012702 004534      MOV #VVVBF0,R2 ;WAS THE OUTPUT DATA CORRECT.
1783 004514 012703 004544      MOV #VVVTP1,R3
1784 004520 012704 000004      MOV #4,R4
1785 004524 022223          1$: CMP (R2)+,(R3)+
1786 004526 001037          BNE VVV20 ;BRANCH IF INCORRECT DATA.
```

```
1787 004530 077403
1788 004532 000444
1789 004534 177777
1790 004536 177777
1791 004540 177777
1792 004542 177777
1793 004544 030313
1794 004546 023334
1795 004550 035363
1796 004552 074041
1797
1798
1799 004554 011602
1800 004556 020227 004500
1801 004562 001405
1802 004564 020227 004502
1803 004570 001402
1804 004572 000137 036140
1805
1806 004576 010237 001236
1807 004602 022626
1808 004604 104001
1809 004606 000416
1810
1811
1812 004610 010037 001242
1813 004614 012737 176633 001240
1814 004622 104001
1815 004624 000407
1816
1817
1818 004626 012737 004534 001240
1819 004634 012737 004544 001242
1820 004642 104001
1821 004644
1822 004644 104413
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834 004646 000004
1835
1836 004650
1837 004650 104414
1838 004652 012700 000200
1839 004656 170100
1840 004660 012701 004776
1841 004664 012700 177777
1842 004670 012702 000004
```

SOB R4,1\$
BR VVVDONE
VVVBF0: -1
-1
-1
-1
VVVTP1: 30313
23334
35363
74041

;COME HERE AFTER A TRAP THROUGH VECTOR 4.
VVV10: MOV (SP),R2 ;SEE IF THE TRAP OCCURRED ON THE TEST INSTR.
CMP R2,#VVV2+2
BEQ VVV11 ;BRANCH IF YES.
CMP R2,#VVV2+4
BEQ VVV11 ;BRANCH IF YES.
JMP @#FSPUR ;OTHERWISE GO REPORT SPURIOUS TRAP TO 4.
;REPORT FAILURE OF FDST RESULTED IN AN ODD ADDRESS TRAP TO 4.
VVV11: MOV R2,@#STMP2
CMP (SP)+,(SP)+
1\$: ERROR 1 ;FDST FORK X ODD ADD
BR VVVDONE

;REPORT R0 MODIFIED.
VVV15: MOV R0,@#STMP4
MOV #VVVBF0-5701,@#STMP3
1\$: ERROR 1 ;R0 MODIFIED!
BR VVVDONE

;REPORT INCORRECT DATA.
VVV20: MOV #VVVBF0,@#STMP3
MOV #VVVTP1,@#STMP4
1\$: ERROR 1 ;BAD DATA
VVVDONE: 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 11 FDST MODE 7, INDEX DEFERRED MODE, TEST
;*
;*THIS IS A TEST OF FDST MODE 7, INDEX DEFERRED MODE, USING STD.
;*

TST11: SCOPE

WWW1: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
MOV #200,R0 ;ENTER DOUBLE FLOATING MODE.
LDFPS R0
MOV #WWWBF0,R1 ;SET UP THE OUTPUT DATA BUFFER.
MOV #-1,R0
MOV #4,R2

```

1843 004674 010021          1$:  MOV    RC,(R1)+
1844 004676 077202          SOB    R2,1$
1845 004700 012737 005026 000004  MOV    #WWW10,@#ERRVECT ;SET UP FOR TRAPS TO 4.
1846 004706 012700 005006          MOV    #WWWTP1,R0      ;SET UP ACO.
1847 004712 172410          LDD    (R0),AC0
1848 004714 012737 004740 001236  MOV    #WWW2,@#STMP2
1849 004722 012700 177115          MOV    #WWWBF1-5701,R0 ;SET UP THE DESTINATION ADDRESS.
1850 004726 012701 000001          MOV    #1,R1
1851 004732 012737 004776 005016  MOV    #WWWBF0,@#WWWBF1
1852 004740 174070 005701  WWW2:  STD    ACO,@5701(R0) ;TEST INSTRUCTION.
1853
1854 004744 020027 177115          CMP    R0,#WWWBF1-5701 ;IS R0 CORRECT?
1855 004750 001044          BNE    WWW15           ;BRANCH IF INCORRECT.
1856 004752 012702 004776          MOV    #WWWBF0,R2      ;WAS THE DATA OUTPUT CORRECTLY?
1857 004756 012703 005006          MOV    #WWWTP1,R3
1858 004762 012704 000004          MOV    #4,R4
1859 004766 022223          1$:  CMP    (R2)+,(R3)+
1860 004770 001043          BNE    WWW20           ;BRANCH IF DATA IS INCORRECT.
1861 004772 077403          SOB    R4,1$
1862 004774 000450          BR     WWWDONE
1863 004776 177777  WWWBF0: -1
1864 005000 177777          -1
1865 005002 177777          -1
1866 005004 177777          -1
1867 005006 041424  WWWTP1: 41424
1868 005010 034445          34445
1869 005012 046475          46475
1870 005014 051525          051525
1871 005016 177777  WWWRF1: -1
1872 005020 177777          -1
1873 005022 177777          -1
1874 005024 177777          -1
1875
1876          ;TRAP THROUGH 4 TO HERE.
1877 005026 011602  WWW10:  MOV    (SP),R2      ;SEE IF THE TRAP OCCURRED ON THE TEST INSTR.
1878 005030 020227 004742          CMP    R2,#WWW2+2
1879 005034 001405          BEQ    WWW11           ;BRANCH IF YES.
1880 005036 020227 004744          CMP    R2,#WWW2+4
1881 005042 001402          BEQ    WWW11           ;BRANCH IF YES.
1882 005044 000137 036140          JMP    @#FSPUR        ;OTHERWISE GO REPORT SPURIOUS TRAP TO 4.
1883          ;REPORT FAILURE OF FDST FORK RESULTED IN AN ODD ADDRESS TRAP TO 4.
1884 005050 010237 001236  WWW11:  MOV    R2,@#STMP2
1885 005054 022626          CMP    (SP)+,(SP)+
1886 005056 104001          1$:  ERROR 1      ;FDST FORK X ODD ADD
1887 005060 000416          BR     WWWDONE
1888
1889          ;REPORT R0 MODIFIED.
1890 005062 010037 001242  WWW15:  MOV    R0,@#STMP4
1891 005066 012737 177075 001240  MOV    #WWWBF0-5701,@#STMP3
1892 005074 104001          1$:  ERROR 1      ;R0 MODIFIED!
1893 005076 000407          BR     WWWDONE
1894
1895          ;REPORT DATA INCORRECT
1896 005100 012737 004776 001240  WWW20:  MOV    #WWWBF0,@#STMP3
1897 005106 012737 005006 001242  MOV    #WWWTP1,@#STMP4
1898 005114 104001          1$:  ERROR 1      ;BAD DATA

```

```
1899 005116  
1900 005116 104413  
1901  
1902  
1903  
1904  
1905  
1906  
1907  
1908  
1909  
1910  
1911  
1912 005120 000004  
1913  
1914  
1915 005122  
1916 005122 104414  
1917 005124 004767 000330  
1918 005130 000000  
1919 005132 000000  
1920 005134 000000  
1921 005136 000000  
1922 005140 000000  
1923 005142 000000  
1924 005144 000000  
1925 005146 000000  
1926 005150 000000  
1927 005152 000000  
1928 005154 177777  
1929 005156 177777  
1930 005160 047000  
1931 005162 047004  
1932 005164 177777  
1933 005166 147004  
1934 005170 104001  
1935 005172 000401  
1936 005174 104001  
1937 005176  
1938  
1939 005176  
1940 005176 104414  
1941 005200 004767 000254  
1942 005204 017203  
1943 005206 142536  
1944 005210 047506  
1945 005212 172031  
1946 005214 017203  
1947 005216 142536  
1948 005220 000000  
1949 005222 000000  
1950 005224 017203  
1951 005226 142536  
1952 005230 047506  
1953 005232 172031  
1954 005234 040000
```

WWWDONE:
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 12 STCFD TEST
*
*THIS IS A TEST OF THE STCFD INSTRUCTION.
*

TST12: SCOPE

;AC=0
XXX1:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,STCFDS
1\$: 0 ;AC
0
0
0
2\$: 0 ;RES
0
0
3\$: 0 ;ERROR RES.
0
-1
-1
4\$: 47000 ;FPS BEFORE EXECUTION.
47004 ;FPS AFTER EXECUTION.
-1 ;FEC
147004 ;ERROR FPS.
5\$: ERROR 1 ;FDFL<---FDFLXST 767
BR 6\$
ERROR 1 ;BUT EZBT X ST560 TO 061 INTO 261
6\$:
XXX2:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,STCFDS
1\$: 17203 ;AC
142536
47506
172031
2\$: 17203 ;RES
142536
0
0
3\$: 17203 ;ERROR RES.
142536
47506
172031
4\$: 40000 ;FPS BEFORE EXECUTION.

1955	005236	040000		40000			;FPS AFTER EXECUTION.
1956	005240	177777		-1			;FEC
1957	005242	177777		-1			;ERROR FPS.
1958	005244	104001	5\$:	ERROR	1		;X11(1,0)<---0 X ST766
1959	005246	000401		BR	6\$		
1960	005250	104001		ERROR	1		
1961	005252		6\$:				
1962			:				
1963	005252		XXX3:				
1964	005252	104414		LPERR			;SET UP THE LOOP ON ERROR ADDRESS.
1965	005254	004767	000200	JSR		PC,STCFDS	
1966	005260	050717		50717			;AC
1967	005262	027374	1\$:	27374			
1968	005264	075767		75767			
1969	005266	077071		77071			
1970	005270	050717	2\$:	50717			;RES
1971	005272	027374		27374			
1972	005274	000000		0			
1973	005276	000000		0			
1974	005300	000000	3\$:	0			;ERROR RES.
1975	005302	000000		0			
1976	005304	000000		0			
1977	005306	000000		0			
1978	005310	047000	4\$:	47000			;FPS BEFORE EXECUTION.
1979	005312	047000		47000			;FPS AFTER EXECUTION.
1980	005314	177777		-1			;FEC
1981	005316	174002		174002			;ERROR FPS.
1982	005320	104001	5\$:	ERROR	1		;BUT OPIC X ST251
1983	005322	000401		BR	6\$		
1984	005324	104001		ERROR	1		;BUT EZBT X ST421
1985	005326		6\$:				
1986			:				
1987	005326		XXX4:				
1988	005326	104414		LPERR			;SET UP THE LOOP ON ERROR ADDRESS.
1989	005330	004767	000124	JSR		PC,STCFDS	
1990	005334	020212		20212			;AC
1991	005336	032425	1\$:	32425			
1992	005340	026272		26272			
1993	005342	002123		02123			
1994	005344	020212	2\$:	20212			;RES
1995	005346	032425		32425			
1996	005350	000000		0			
1997	005352	000000		0			
1998	005354	020212	3\$:	20212			;ERROR RES.
1999	005356	032425		32425			
2000	005360	100000		100000			
2001	005362	000000		0			
2002	005364	040000	4\$:	40000			;FPS BEFORE EXECUTION.
2003	005366	040000		40000			;FPS AFTER EXECUTION.
2004	005370	177777		-1			;FEC
2005	005372	177777		-1			;ERROR FPS.
2006	005374	104001	5\$:	ERROR	1		;BUT FD IN ROUND X ST113
2007	005376	000401		BR	6\$		
2008	005400	104001		ERROR	1		
2009	005402		6\$:				
2010			:				

2011 005402
 2012 005402 104414
 2013 005404 004767 000050
 2014 005410 121314
 2015 005412 151617
 2016 005414 101112
 2017 005416 131415
 2018 005420 121314
 2019 005422 151617
 2020 005424 000000
 2021 005426 000000
 2022 005430 021314
 2023 005432 151617
 2024 005434 000000
 2025 005436 000000
 2026 005440 040000
 2027 005442 040010
 2028 005444 177777
 2029 005446 177777
 2030 005450 104001
 2031 005452 000401
 2032 005454 104001
 2033 005456 000535

xxx5:
 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
 JSR PC,STCFDS
 1\$: 121314 ;AC
 151617
 101112
 131415
 2\$: 121314 ;RES
 151617
 0
 0
 3\$: 21314 ;ERROR RES.
 151617
 0
 0
 4\$: 40000 ;FPS BEFORE EXECUTION.
 40010 ;FPS AFTER EXECUTION.
 -1 ;FEC
 -1 ;ERROR FPS.
 5\$: ERROR 1 ;BUT ENBT X ST567 OR BAD SIGN ST460
 BR 6\$
 ERROR 1
 6\$: BR XXXDONE

2034
 2035
 2036
 2037
 2038
 2039
 2040
 2041
 2042
 2043
 2044
 2045
 2046
 2047
 2048
 2049
 2050
 2051
 2052
 2053
 2054
 2055
 2056
 2057
 2058
 2059
 2060
 2061
 2062
 2063
 2064
 2065
 2066

: THIS SUBROUTINE, STCFDS, IS USED TO SET UP THE OPERANDS, EXECUTE
 : THE STCFD INSTRUCTION AND CHECK THE RESULTS. A CALL
 : TO IT IS MADE THUS:

```

:
: JSR PC,STCFDS
: ACARG: .WORD X,X,X,X ;AC OPERAND
: RES: .WORD X,X,X,X ;EXPECTED RESULT
: ERRES: .WORD X,X,X,X ;ERROR RESULT
: FPSB: .WORD X ;FPS BEFORE EXECUTION
: FPSA: .WORD X ;FPS AFTER EXECUTION
: FEC: .WORD X ;EXPECTED FEC
: ERFPS: .WORD X ;ERROR FPS.
: ERR1: ERROR 1 ;DATA ERROR.
: BR CONT
: ERR2: ERROR 1 ;FPS ERROR.
: CONT: ;RETURN ADDRESS

```

: THE OPERANDS ARE SET UP (USING ACO AS THE ACCUMULATOR). THEN
 : THE STCFD INSTRUCTION IS EXECUTED.
 : THE RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
 : COMPARED WITH FPSA IF THIS TOO IS CORRECT STCFDS RETURNS CONTROL
 : TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD STCFDS
 : COMPARE IT TO ERROR FPS. IF THIS MATCHES THEN STCFDS WILL RETURN
 : TO THE ERROR CALL AT ERR2, OTHERWISE STCFDS ITSELF
 : REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE
 : STCFD 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 STCFDS
 : WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR1. OTHERWISE THE

```
2067 ;RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND STCFDS WILL
2068 ;REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.
2069
2070 005460 012601 STCFDS: MOV (SP)+,R1 ;PICK UP THE POINTER TO THE OPERANDS.
2071 005462 012700 000200 MOV #200,R0 ;ENTER DOUBLE FLOATING MODE.
2072 005466 170100 LDFPS R0
2073 005470 010100 MOV R1,R0 ;LOAD ACO.
2074 005472 172410 LDD (R0),ACO
2075 005474 012700 177777 MOV #-1,R0 ;FILL THE OUTPUT BUFFER WITH -1'S.
2076 005500 012702 005742 MOV #STCFT,R2
2077 005504 012703 000004 MOV #4,R3
2078 005510 010022 1$: MOV R0,(R2)+
2079 005512 077302 SOB R3,1$
2080 005514 016100 000030 MOV 30(R1),R0 ;LOAD THE FPS.
2081 005520 170100 LDFPS R0
2082 005522 012737 005534 001236 MOV #2$,@#STMP2
2083 005530 012700 005742 MOV #STCFT,R0 ;SET UP THE DESTINATION ADDRESS.
2084 005534 176010 2$: STCFD ACO,(R0) ;TEST INSTRUCTION.
2085
2086 005536 170204 STFPS R4 ;GET THE FPS.
2087 005540 170305 STST R5 ;GET THE FEC.
2088 005542 010102 MOV R1,R2 ;SAVE THE DATA IN CASE OF ERROR.
2089 005544 010237 001240 MOV R2,@#STMP3
2090 005550 062702 000010 ADD #10,R2
2091 005554 010237 001244 MOV R2,@#STMP5
2092 005560 012737 005742 001242 MOV #STCFT,@#STMP4
2093 005566 010437 001250 MOV R4,@#STMP7
2094 005572 016137 000032 001252 MOV 32(R1),@#STMP10
2095
2096 005600 010102 MOV R1,R2 ;CHECK THE RESULT.
2097 005602 062702 000010 ADD #10,R2
2098 005606 012703 005742 MOV #STCFT,R3
2099 005612 012700 000004 MOV #4,R0
2100 005616 022223 3$: CMP (R2)+,(R3)+
2101 005620 001014 BNE 15$ ;BRANCH IF INCORRECT.
2102 005622 077003 SOB R0,3$
2103
2104 005624 016102 000032 MOV 32(R1),R2
2105 005630 020204 CMP R2,R4 ;IS THE FPS CORRECT?
2106 005632 001025 BNE 20$ ;BRANCH IF FPS INCORRECT.
2107 005634 005702 TST R2 ;IF EXPECTED FPS IS NEGATIVE, THEN
2108 005636 100003 BPL 4$ ;GO AHEAD AND CHECK THE FEC.
2109 005640 026105 000036 CMP 36(R1),R5
2110 005644 001027 BNE 25$ ;BRANCH IF FEC IS INCORRECT.
2111 005646 000161 000046 4$: JMP 46(R1) ;RETURN.
2112
2113 ;RESULT INCORRECT:
2114 005652 010102 15$: MOV R1,R2 ;SEE IF ERROR WAS ANTICIPATED.
2115 005654 062702 000020 ADD #20,R2
2116 005660 012703 005742 MOV #STCFT,R3
2117 005664 012700 000004 MOV #4,R0
2118 005670 022223 16$: CMP (R2)+,(R3)+
2119 005672 001003 BNE 17$ ;BRANCH IF NOT ANTICIPATED.
2120 005674 077003 SOB R0,16$
2121 005676 000161 000040 JMP 40(R1) ;IF ERROR WAS ANTICIPATED RETURN.
2122 ;OTHERWISE REPORT RESULT INCORRECT HERE.
```

```
2123 005702 17$:
2124 005702 104001 18$: ERROR 1 ;DATA ERROR
2125 005704 000760 BR 4$
2126
2127 ;FPS INCORRECT:
2128 005706 020461 000034 20$: CMP R4,34(R1) ;WAS THE ERROR ANTICIPATED.
2129 005712 001002 BNE 21$ ;BRANCH IF NOT ANTICIPATED.
2130 005714 000161 000044 JMP 44(R1) ;IF IT WAS ANTICIPATED RETURN.
2131
2132 ;THE FPS ERROR WAS NOT ANTICIPATED SO REPORT FPS INCORRECT HERE.
2133 005720 21$:
2134 005720 104001 22$: ERROR 1 ;FPS X
2135 005722 000751 BR 4$
2136
2137 ;REPORT FEC INCORRECT:
2138 005724 016137 000036 001256 25$: MOV 36(R1),@#$TMP12
2139 005732 010537 001254 MOV R5,@#$TMP11
2140 005736 104001 26$: ERROR 1 ;FEC X
2141 005740 000742 BR 4$
2142 005742 177777 177777 STCFT: -1,-1,-1,-1
2143 005750 177777
2144 005752
2145 005752 104413 XXXDONE: RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
2146 ;SEE IF THE USER HAS EXPRESSED
2147 ;THE DESIRE TO CHANGE THE SOFTWARE
2148 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
2149 ;THE USER TYPED CONTROL G?).
2150
2151 ;*****
2152 ;*TEST 13 STCDF TEST
2153 ;*
2154 ;*THIS IS A TEST OF THE STCDF INSTRUCTION.
2155 ;*
2156 ;*****
2157 005754 000004 TST13: SCOPE
2158
2159 ;AC=0
2160 005756 YYY1:
2161 005756 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
2162 005760 004767 000330 JSR PC,STCDF5
2163 005764 000000 1$: 0 ;AC
2164 005766 000000 0
2165 005770 000000 0
2166 005772 000000 0
2167 005774 000000 2$: 0 ;RES
2168 005776 000000 0
2169 006000 177777 -1
2170 006002 177777 -1
2171 006004 000000 3$: 0 ;ERROR RES.
2172 006006 000000 0
2173 006010 000000 0
2174 006012 000000 0
2175 006014 047200 4$: 47200 ;FPS BEFORE EXECUTION.
2176 006016 047204 47204 ;FPS AFTER EXECUTION.
2177 006020 177777 -1 ;FEC
2178 006022 177777 -1 ;ERROR FPS.
```

```

2179 006024 104001      5$: ERROR 1          ;FDFL<---FDFL X ST767
2180 006026 000401      BR      6$
2181 006030 104001      ERROR 1          ;FPS INCORRECT.
2182 006032
2183
2184 006032
2185 006032 104414      YYY2: LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
2186 006034 004767 000254 JSR      PC,STCDF5
2187 006040 067574      1$: 67574
2188 006042 073727      73727          ;ACO
2189 006044 170777      170777
2190 006046 067574      67574
2191 006050 067574      2$: 67574          ;RES
2192 006052 073730      73730
2193 006054 177777      -1
2194 006056 177777      -1
2195 006060 067574      3$: 67574          ;ERROR RES.
2196 006062 073727      73727
2197 006064 177777      -1
2198 006066 177777      -1
2199 006070 040200      4$: 40200          ;FPS BEFORE EXECUTION.
2200 006072 040200      40200          ;FPS AFTER EXECUTION.
2201 006074 177777      -1
2202 006076 177777      -1
2203 006100 104001      5$: ERROR 1          ;ERROR FPS.
2204 006102 000401      BR      6$          ;EITHER ROUND FAILED OR WENT TO 766 X1(1,0)<---0 INTO 76
2205 006104 104001      ERROR 1
2206 006106
2207
2208 006106
2209 006106 104414      YYY3: LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
2210 006110 004767 000200 JSR      PC,STCDF5
2211 006114 077777      1$: 77777          ;ACO
2212 006116 177777      -1
2213 006120 100000      100000
2214 006122 000000      0
2215 006124 000000      2$: 0             ;RES
2216 006126 000000      0
2217 006130 177777      -1
2218 006132 177777      -1
2219 006134 077777      3$: 77777          ;ERROR RES.
2220 006136 177777      -1
2221 006140 177777      -1
2222 006142 177777      -1
2223 006144 040200      4$: 40200          ;FPS BEFORE EXECUTION.
2224 006146 040206      40206          ;FPS AFTER EXECUTION.
2225 006150 177777      -1
2226 006152 040204      40204          ;FEC
2227 006154 104001      5$: ERROR 1          ;ERROR FPS.
2228 006156 000401      BR      6$
2229 006160 104001      ERROR 1          ;BUT EZBT X ST421 TO 062 INTO 262
2230 006162
2231
2232 006162
2233 006162 104414      YYY4: LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
2234 006164 004767 000124 JSR      PC,STCDF5

```

```
2235 006170 077777 1$: 77777 ;ACO
2236 006172 177777 -1
2237 006174 100000 100000
2238 006176 000000 0
2239 006200 000000 2$: 0 ;RES
2240 006202 000000 0
2241 006204 177777 -1
2242 006206 177777 -1
2243 006210 077777 3$: 77777 ;ERROR RES.
2244 006212 177777 -1
2245 006214 177777 -1
2246 006216 177777 -1
2247 006220 040200 4$: 40200 ;FPS BEFORE EXECUTION.
2248 006222 040206 40206 ;FPS AFTER EXECUTION.
2249 006224 177777 -1 ;FEC
2250 006226 140206 140206 ;ERROR FPS.
2251 006230 104001 5$: ERROR 1
2252 006232 000401 BR 6$
2253 006234 104001 ERROR 1 ;BUT FIV ST262 TO 123 INTO 103
2254 006236
2255
2256 006236
2257 006236 104414 YYY5: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
2258 006240 004767 000050 JSR PC,STCFDS
2259 006244 177777 1$: 177777 ;ACO
2260 006246 177777 -1
2261 006250 100000 100000
2262 006252 000000 0
2263 006254 100000 2$: 100000 ;RES
2264 006256 000000 0
2265 006260 177777 -1
2266 006262 177777 -1
2267 006264 000000 3$: 0 ;ERROR RES.
2268 006266 000000 0
2269 006270 177777 -1
2270 006272 177777 -1
2271 006274 047200 4$: 47200 ;FPS BEFORE EXECUTION.
2272 006276 147216 147216 ;FPS AFTER EXECUTION.
2273 006300 000010 10 ;FEC
2274 006302 047206 47206 ;ERROR FPS.
2275 006304 104001 5$: ERROR 1 ;BUT FIV ST262 FAIL TO 103 INT 123
2276 006306 000401 BR 6$
2277 006310 104001 ERROR 1 ;BUT FLAG ST 147 X TO ST 361 INTO 365
2278 006312 000535 6$: BR YYYDONE
2279 ;THIS SUBROUTINE, STCFDS, IS USED TO SET UP THE OPERANDS, EXECUTE
2280 ;THE STCDF INSTRUCTION AND CHECK THE RESULTS. A CALL
2281 ;TO IT IS MADE THUS:
2282 :
2283 : JSR PC,@STCFDS
2284 : ACARG: .WORD X,X,X,X ;AC OPERAND
2285 : RES: .WORD X,X,X,X ;EXPECTED RESULT
2286 : ERRES: .WORD X,X,X,X ;ERROR RESULT
2287 : FPSB: .WORD X ;FPS BEFORE EXECUTION
2288 : FPSA: .WORD X ;FPS AFTER EXECUTION
2289 : FEC: .WORD X ;EXPECTED FEC
2290 : ERFPS: .WORD X ;ERROR FPS.
```

```

2291          :          ERR1:  ERROR  1          ;DATA ERROR.
2292          :          BR      CONT          ;
2293          :          ERR2:  ERROR  1          ;FPS ERROR.
2294          :          CONT:                    ;RETURN ADDRESS
2295          :
2296          :THE OPERANDS ARE SET UP (USING ACO AS THE ACCUMULATOR). THEN
2297          :THE STCDF INSTRUCTION IS EXECUTED.
2298          :THE RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
2299          :COMPARED WITH FPSA IF THIS TOO IS CORRECT STCFDS RETURNS CONTROL
2300          :TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD STCFDS
2301          :COMPARE IT TO ERROR FPS. IF THIS MATCHES THEN STCFDS WILL RETURN
2302          :TO THE ERROR CALL AT ERR2, OTHERWISE STCFDS ITSELF
2303          :REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE
2304          :STCDF IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
2305          :ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
2306          :THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN STCFDS
2307          :WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR1. OTHERWISE THE
2308          :RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND STCFDS WILL
2309          :REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.
2310
2311 006314 012601          STCFDS: MOV      (SP)+,R1          ;PICK UP THE POINTER TO THE OPERANDS.
2312 006316 012700 000200      MOV      #200,R0          ;ENTER DOUBLE FLOATING MODE.
2313 006322 170100          LDFPS   R0
2314 006324 010100          MOV      R1,R0          ;LOAD ACO.
2315 006326 172410          LDD     (R0),ACO
2316 006330 012700 177777      MOV      #-1,R0          ;FILL THE OUTPUT BUFFER WITH -1'S.
2317 006334 012702 006576      MOV      #STCDT,R2
2318 006340 012703 000004      MOV      #4,R3
2319 006344 010022          1$:  MOV      R0,(R2)+
2320 006346 077302          SOB     R3,1$
2321 006350 016100 000030      MOV      30(R1),R0          ;LOAD THE FPS.
2322 006354 170100          LDFPS   R0
2323 006356 012737 006370 001236      MOV      #2$,@#STMP2
2324 006364 012700 006576          MOV      #STCDT,R0          ;SET UP THE DESTINATION ADDRESS.
2325 006370 176010          2$:  STCDF   ACO,(R0)          ;TEST INSTRUCTION.
2326
2327 006372 170204          STFPS   R4          ;GET THE FPS.
2328 006374 170305          STST   R5          ;GET THE FEC.
2329 006376 010102          MOV      R1,R2          ;SAVE THE DATA IN CASE OF ERROR.
2330 006400 010237 001240      MOV      R2,@#STMP3
2331 006404 062702 000010      ADD     #10,R2
2332 006410 010237 001244          MOV      R2,@#STMP5
2333 006414 012737 006576 001242      MOV      #STCDT,@#STMP4
2334 006422 010437 001250          MOV      R4,@#STMP7
2335 006426 016137 000032 001252      MOV      32(R1),@#STMP10
2336
2337 006434 010102          MOV      R1,R2          ;CHECK THE RESULT.
2338 006436 062702 000010      ADD     #10,R2
2339 006442 012703 006576          MOV      #STCDT,R3
2340 006446 012700 000004          MOV      #4,R0
2341 006452 022223          3$:  CMP     (R2)+,(R3)+
2342 006454 001014          BNE    15$          ;BRANCH IF INCORRECT.
2343 006456 077003          SOB     R0,3$
2344
2345 006460 016102 000032          MOV      32(R1),R2
2346 006464 020204          CMP     R2,R4          ;IS THE FPS CORRECT?
  
```

```
2347 006466 001025          BNE      20$          ;BRANCH IF FPS INCORRECT.
2348 006470 005702          TST      R2          ;IF EXPECTED FPS IS NEGATIVE, THEN
2349 006472 100003          BPL      4$          ;GO AHEAD AND CHECK THE FEC.
2350 006474 026105 000034    CMP      34(R1),R5
2351 006500 001027          BNE      25$          ;BRANCH IF FEC IS INCORRECT.
2352 006502 000161 000046    4$:     JMP      46(R1)    ;RETURN.
2353
2354          ;RESULT INCORRECT:
2355 006506 010102          15$:     MOV      R1,R2          ;SEE IF ERROR WAS ANTICIPATED.
2356 006510 062702 000020    ADD      #20,R2
2357 006514 012703 006576    MOV      #STCDT,R3
2358 006520 012700 000004    MOV      #4,R0
2359 006524 022223          16$:     CMP      (R2)+,(R3)+
2360 006526 001003          BNE      17$          ;BRANCH IF NOT ANTICIPATED.
2361 006530 077003          SOB      R0,16$
2362 006532 000161 000040    JMP      40(R1)    ;IF ERROR WAS ANTICIPATED RETURN.
2363          ;OTHERWISE REPORT RESULT INCORRECT HERE.
2364 006536 104001          17$:
2365 006536 000760          18$:     ERROR   1          ;DATA ERROR
2366 006540
2367
2368          ;FPS INCORRECT:
2369 006542 020461 000034    20$:     CMP      R4,34(R1)          ;WAS THE ERROR ANTICIPATED.
2370 006546 001002          BNE      21$          ;BRANCH IF NOT ANTICIPATED.
2371 006550 000161 000044    JMP      44(R1)    ;IF IT WAS ANTICIPATED RETURN.
2372
2373          ;THE FPS ERROR WAS NOT ANTICIPATED SO REPORT FPS INCORRECT HERE.
2374 006554
2375 006554 104001          21$:
2376 006556 000751          22$:     ERROR   1          ;FPS X
2377
2378          ;REPORT FEC INCORRECT:
2379 006560 016137 000036 001256    25$:     MOV      36(R1),@#$TMP12
2380 006566 010537 001254    MOV      R5,@#$TMP11
2381 006572 104001          26$:     ERROR   1          ;FEC X
2382 006574 000742          BR       4$
2383 006576 177777 177777    STCDT:  -1,-1,-1,-1
2384 006604 177777
2385 006606
2386 006606 104413          YYYDONE:
2387          RSETUP          ;GO INITIALIZE THE FPS AND STACK; AND
2388          ;SEE IF THE USER HAS EXPRESSED
2389          ;THE DESIRE TO CHANGE THE SOFTWARE
2390          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
2391          ;THE USER TYPED CONTROL G?).
2392          ;*****
2393          ;*TEST 14          STCDF WITH ILLEGAL ACCUMULATOR TEST
2394          ;*
2395          ;*THIS TEST STCDF WITH ILLEGAL AC 6.
2396          ;*
2397          ;*****
2397 006610 000004          TST14:  SCOPE
2398
2399          ZZZ1:
2400 006612 104414          LPERR
2401 006614 012700 040000    MOV      #40000,R0          ;SET UP THE LOOP ON ERROR ADDRESS.
2402 006620 170100          LDFPS   R0          ;DISSABLE INTERRUPTS.
```

```

CJKDDA KEF11-A DIAG PART 2          MACY11 30A(1052) 20-JUN-79 11:39 PAGE 47
CJKDDA.P11 20-JUN-79 11:22          T14      STCFD WITH ILLEGAL ACCUMULATOR TEST
                                           I 4
                                           SEQ 0047

2403 006622 012737 006630 001236      MOV      #ZZZZ,@#$TMP2
2404 006630 176006                ZZZ2:   STCFD      ACO,AC6          ;THIS TEST INSTRUCTION SHOULD CAUSE AN ERROR.
2405
2406 006632 170204                STFPS    R4              ;GET FPS.
2407 006634 170305                STST     R5              ;GET FEC.
2408 006636 020427 140000          CMP      R4,#140000      ;IS FPS CORRECT?
2409 006642 001004                BNE      ZZZ10           ;BRANCH IF INCORRECT FPS.
2410 006644 022705 000002          CMP      #2,R5          ;IS FEC CORRECT?
2411 006650 001010                BNE      ZZZ15           ;BRANCH IF INCORRECT.
2412 006652 000415                BR       ZZZDONE
2413
2414                                ;REPORT FPS INCORRECT AFTER USE OF ILLEGAL ACCUMULATOR.
2415 006654 010437 001242          ZZZ10:  MOV      R4,@#$TMP4
2416 006660 012737 140000 001240  MOV      #140000,@#$TMP3
2417 006666 104001                1$:     ERROR      1          ;BUT FDST ST767 X TO 567 INTO 577
2418 006670 000406                BR       ZZZDONE
2419
2420                                ;REPORT FEC INCORRECT AFTER USE OF ILLEGAL ACCUMULATOR.
2421 006672 010537 001242          ZZZ15:  MOV      R5,@#$TMP4
2422 006676 012737 000002 001240  MOV      #2,@#$TMP3
2423 006704 104001                1$:     ERROR      1          ;FEC<---2 ST577 X
2424 006706
2425 006706 104413                ZZZDONE: RSETUP          ;GO INITIALIZE THE FPS AND STACK; AND
2426                                ;SEE IF THE USER HAS EXPRESSED
2427                                ;THE DESIRE TO CHANGE THE SOFTWARE
2428                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
2429                                ;THE USER TYPED CONTROL G?).
2430
2431                                ;*****
2432                                ;*TEST 15      CLRD TEST
2433                                ;*
2434                                ;*THIS IS A TEST OF THE CRLF AND CLRD INSTRUCTIONS.
2435                                ;*
2436                                ;*****
2437 006710 000004          TST15:  SCOPE
2438 006712          AAB1:
2439 006712 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
2440 006714 012700 007100      MOV      #AABTP1,R0      ;SET UP OUTPUT BUFFER
2441 006720 012701 007070      MOV      #AABBF0,R1
2442 006724 012702 000004      MOV      #4,R2
2443 006730 012021          1$:     MOV      (R0)+,(R1)+
2444 006732 077202          SOB      R2,1$
2445 006734 012700 007070      MOV      #AABBF0,R0      ;SET UP DESTINATION OPERAND ADDRESS.
2446 006740 012701 000213      MOV      #213,R1         ;SET UP FPS.
2447 006744 170101          LDFPS     R1
2448 006746 012737 006754 001236  MOV      #2$,@#$TMP2
2449 006754 170410          2$:     CLRD      (R0)          ;TEST INSTRUCTION.
2450
2451 006756 170205          STFPS     R5              ;GET FPS.
2452 006760 012702 000004      MOV      #4,R2              ;SEE IF RESULT CLEAR, 0.
2453 006764 012701 007070      MOV      #AABBF0,R1
2454 006770 005721          3$:     TST      (R1)+
2455 006772 001010          BNE      AAB2              ;BRANCH IF RESULT INCORRECT, NOT 0.
2456 006774 077203          SOB      R2,3$
2457 006776 022705 000204      CMP      #204,R5          ;SEE IF FPS IS CORRECT.
2458 007002 001014          BNE      AAB3              ;BRANCH IF INCORRECT.

```



```
2459 007004 020027 007070          CMP    R0,#AABBF0          ;SEE IF R0 IS CORRECT.
2460 007010 001020          BNE    AAB4                ;BRANCH IF R0 IS INCORRECT.
2461 007012 000442          BR     AABDONE
2462
2463          ;RESULT NOT 0, REPORT ERROR.
2464 007014 012737 007070 001240 AAB2:  MOV    #AABBF0,@#STMP3
2465 007022 012737 007110 001242      MOV    #AABTP2,@#STMP4
2466 007030 104001          1$:   ERROR 1                ;BAD DATA = 0 X 11+ZERO ST770 X
2467 007032 000432          BR     AABDONE
2468
2469          ;REPORT FPS INCORRECT:
2470 007034 010437 001242          AAB3:  MOV    R4,@#STMP4
2471 007040 012737 000204 001240      MOV    #204,@#STMP3
2472 007046 104001          1$:   ERROR 1                ;BAD FPS
2473 007050 000423          BR     AABDONE
2474
2475          ;REPORT R0 INCORRECT.
2476 007052 010037 001242          AAB4:  MOV    R0,@#STMP4
2477 007056 012737 007070 001240      MOV    #AABBF0,@#STMP3
2478 007064 104001          1$:   ERROR 1
2479 007066 000414          BR     AABDONE
2480
2481          ;THIS IS THE TEST DATA BUFFER, OUTPUT DATA BUFFER.
2482 007070 073475          AABBF0: 73475
2483 007072 067707          67707
2484 007074 127347          127347
2485 007076 056770          56770
2486          ;THIS IS THE DATA USED TO SET UP THE OUTPUT BUFFER.
2487 007100 073475          AABTP1: 73475
2488 007102 067707          67707
2489 007104 127347          127347
2490 007106 056770          56770
2491          ;THIS IS THE EXPECTED DATA, RESULT:
2492 007110 000000          AABTP2: 0
2493 007112 000000          0
2494 007114 000000          0
2495 007116 000000          0
2496 007120          AABDONE:
2497 007120 104413          RSETUP                ;GO INITIALIZE THE FPS AND STACK; AND
2498                                     ;SEE IF THE USER HAS EXPRESSED
2499                                     ;THE DESIRE TO CHANGE THE SOFTWARE
2500                                     ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
2501                                     ;THE USER TYPED CONTROL G?).
2502
2503          ;*****
2504          ;*TEST 16          CLRD WITH ILLEGAL ACCUMULATOR TEST
2505          ;*
2506          ;*THIS IS A TEST OF CLRD WITH ILLEGAL AC7.
2507          ;*
2508          ;*****
2509 007122 000004          TST16: SCOPE
2510 007124          CCB1:
2511 007124 104414          LPERR                ;SET UP THE LOOP ON ERROR ADDRESS.
2512 007126 012700 040200      MOV    #40200,R0      ;SET UP THE FPS, NO INTERRUPTS AND FD=1.
2513 007132 170100          R0
2514 007134 012737 007142 001236      MOV    #CCB2,@#STMP2
```

K 4

```
2515 007142 170407          CCB2:  CLRD  AC7          ;TEST INSTRUCTION.
2516
2517 007144 170204          STFPS  R4          ;GET FPS.
2518 007146 170305          STST   R5          ;GET FEC.
2519 007150 020427 140200    CMP    R4,#140200    ;IS THE FPS CORRECT?
2520 007154 001004          BNE    CCB10        ;BRANCH IF FPS IS INCORRECT.
2521 007156 022705 000002    CMP    #2,R5        ;IS THE FEC CORRECT?
2522 007162 001010          BNE    CCB15        ;BRANCH IF FEC IS INCORRECT.
2523 007164 000415          BR     CCBDONE
2524
2525          ;REPORT INCORRECT FPS:
2526 007166 010437 001242    CCB10: MOV    R4,@#STMP4
2527 007172 012737 140200 001240  MOV    #140200,@#STMP3
2528 007200 104001          1$:   ERROR 1          ;BUT FDST ST 700X TO 607 INTO 677
2529 007202 000406          BR     CCBDONE
2530
2531          ;REPORT INCORRECT FEC:
2532 007204 010537 001242    CCB15: MOV    R5,@#STMP4
2533 007210 012737 000002 001240  MOV    #2,@#STMP3
2534 007216 104001          1$:   ERROR 1          ;FEC<---2 ST 677 X
2535 007220          CCBDONE:
2536 007220 104413          RSETUP          ;GO INITIALIZE THE FPS AND STACK; AND
2537          ;SEE IF THE USER HAS EXPRESSED
2538          ;THE DESIRE TO CHANGE THE SOFTWARE
2539          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
2540          ;THE USER TYPED CONTROL G?).
2541
2542          ;*****
2543          ;*TEST 17      NEGF, ABSF AND TSTF SOURCE MODE 0 WITH ILLEGAL AC7, TEST
2544          ;*
2545          ;*THIS IS A TEST OF THE SPECIAL
2546          ;*DEST FLOWS USING THE NEGD INST
2547          ;*WITH MODE ZERO AND ILLEGAL
2548          ;*AC7.
2549          ;*
2550          ;*****
2551 007222 000004          TST17: SCOPE
2552
2553          VVB1:
2554 007224 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
2555 007226 012700 040200    MOV    #40200,R0    ;SET UP THE FPS, FID-1 AND FD 1.
2556 007232 170100          LDFPS  R0
2557 007234 012737 007242 001236  MOV    #VVB2,@#STMP2
2558
2559 007242 170707          VVB2:  NEGD  AC7          ;TEST INSTRUCTION.
2560
2561 007244 170204          STFPS  R4          ;GET FPS.
2562 007246 170305          STST   R5          ;GET FEC.
2563
2564 007250 022704 140200    CMP    #140200,R4    ;IS FPS CORRECT?
2565 007254 001004          BNE    VVB10        ;BRANCH IF FPS IS INCORRECT.
2566 007256 022705 000002    CMP    #2,R5        ;IS FEC CORRECT?
2567 007262 001010          BNE    VVB15        ;BRANCH IF FEC IS INCORRECT.
2568 007264 000415          BR     VVBDONE
2569
2570          ;REPORT INCORRECT FPS:
```

```

2571 007266 012737 140200 001240 VVB10: MOV #140200,@#STMP3
2572 007274 010437 001242 MOV R4,@#STMP4
2573 007300 104002 1$: ERROR 2 ;FPS BAD
2574 007302 000406 BR VVB DONE
2575
2576 ;REPORT FEC INCORRECT:
2577 007304 012737 000002 001240 VVB15: MOV #2,@#STMP3
2578 007312 010537 001242 MOV R5,@#STMP4
2579 007316 104002 1$: ERROR 2 ;FEC BAD
2580
2581 007320 VVB DONE:
2582 007320 104413 RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
2583 ;SEE IF THE USER HAS EXPRESSED
2584 ;THE DESIRE TO CHANGE THE SOFTWARE
2585 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
2586 ;THE USER TYPED CONTROL G?).
2587
2588 ;*****
2589 ;*TEST 20 NEGF, ABSF AND TSTF SOURCE MODE 0 TEST
2590 ;*
2591 ;*THIS IS A TEST THE NEGF, ABSF AND TSTF
2592 ;*SOURCE FLOWS. THE NEGD INSTRUCTION
2593 ;*IS USED TO TEST MODE 0
2594 ;*
2595 ;*****
2596 007322 000004 TST20: SCOPE
2597
2598 007324 DDB1:
2599 007324 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
2600 007326 012700 000200 MOV #200,R0 ;SET FD MODE.
2601 007332 170100 LDFPS R0
2602 007334 012700 007476 MOV #DDBTP1,R0 ;SET UP ACO.
2603 007340 172410 LDD (R0),AC0 ;SET ACO = 0
2604 007342 005000 CLR R0 ;CLEAR THE FPS.
2605 007344 170100 LDFPS R0
2606 007346 012700 007506 MOV #DDBTP2,R0 ;LOAD ACO TO BE A FLOATING 0.
2607 007352 172410 LDF (R0),AC0 ;SET ACO=ZERO
2608 ;FLOAT
2609 007354 012700 000201 MOV #201,R0 ;SET FD MODE.
2610 007360 170100 LDFPS R0
2611 007362 012737 007370 001236 MOV #DDB2,@#STMP2
2612
2613 007370 170700 DDB2: NEGD ACO ;TEST INSTRUCTION.
2614
2615 007372 170205 STFPS R5 ;GET FPS.
2616 007374 012700 000200 MOV #200,R0 ;SET FD MODE.
2617 007400 170100 LDFPS R0
2618 007402 012700 007516 MOV #DDBBF0,R0 ;GET THE RESULT OUT OF ACO.
2619 007406 174010 STD ACO,(R0)
2620 ;SEE IF THE RESULT IS CORRECT.
2621 007410 012701 000004 1$: MOV #4,R1
2622 007414 005720 TST (R0)+
2623 007416 001005 BNE DDB5 ;BRANCH IF THE RESULT IS INCORRECT.
2624 007420 077103 SOB R1,1$
2625 007422 022705 000204 CMP #204,R5 ;IS THE FPS CORRECT?
2626 007426 001014 BNE DDB6 ;BRANCH IF THE FPS IS INCORRECT.
    
```

```

2627 007430 000442 BR DDBDONE
2628
2629 ;RESULT INCORRECT, REPORT FAILURE:
2630 007432 012737 007506 001242 DDB5: MOV #DDBTP2,@#STMP4 ;EXPECT D0
2631 007440 012737 007526 001240 MOV #DDBTP3,@#STMP3 ;PREV FO IMPURE
2632 007446 012737 007516 001244 MOV #DDBBF0,@#STMP5 ;GOT
2633 007454 104002 1$: ERROR 2
2634 007456 000427 BR DDBDONE
2635
2636 ;REPORT FPS INCORRECT:
2637 007460 012737 000204 001240 DDB6: MOV #204,@#STMP3
2638 007466 010537 001242 MOV R5,@#STMP4
2639 007472 104002 1$: ERROR 2
2640 007474 000420 BR DDBDONE
2641
2642 ;THESE ARE TEST DATA TABLES AND AN OUTPUT BUFFER.
2643 007476 101112 DDBTP1: 101112
2644 007500 131415 131415
2645 007502 161710 161710
2646 007504 111213 111213
2647 007506 000000 DDBTP2: 0
2648 007510 000000 0
2649 007512 000000 0
2650 007514 000000 0
2651
2652 007516 177777 DDBBF0: -1
2653 007520 177777 -1
2654 007522 177777 -1
2655 007524 177777 -1
2656 007526 000000 DDBTP3: 0
2657 007530 000000 0
2658 007532 161710 161710
2659 007534 111213 111213
2660
2661 007536 DDBDONE:
2662 007536 104413 RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
2663 ;SEE IF THE USER HAS EXPRESSED
2664 ;THE DESIRE TO CHANGE THE SOFTWARE
2665 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
2666 ;THE USER TYPED CONTROL G?).
2667
2668 ;:*****
2669 ;*TEST 21 NEGF, ABSF AND TSTF SOURCE MODE 1 TEST
2670 ;*
2671 ;*THIS IS A TEST THE NEGF, ABSF AND TSTF
2672 ;*SOURCE FLOWS. THE NEGD INSTRUCTION
2673 ;*IS USED TO TEST MODE 1
2674 ;*
2675 ;:*****
2676 007540 000004 TST21: SCOPE
2677
2678 007542 EEB1:
2679 007542 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
2680 007544 012700 007652 MOV #EEBTP1,R0 ;SET UP THE DATA BUFFER.
2681 007550 012701 007702 MOV #EEBBF1,R1
2682 007554 012702 000004 MOV #4,R2
    
```

```
2683 007560 012021      1$:  MOV      (R0)+,(R1)+
2684 007562 077202      SOB      R2,1$
2685 007564 012700 000200  MOV      #200,R0      ;SET FD MODE.
2686 007570 170100      LDFPS   R0
2687 007572 012700 007702  MOV      #EEBBF1,R0    ;SET UP THE OPERAND ADDRESS.
2688 007576 012737 007612 001236  MOV      #EEB2,@#STMP2
2689 007604 012737 007712 000004  MOV      #EEB10,@#ERRVECT ;SET UP VECTOR 4 IN CASE OF ERROR.
2690 007612 170710      EEB2:  NEG D      (R0)      ;TEST INSTRUCTION.
2691
2692 007614 170205      STFPS   R5      ;GET FPS.
2693 007616 012701 007702  MOV      #EEBBF1,R1    ;SEE IF RESULT IS CORRECT.
2694 007622 012702 000004  MOV      #4,R2
2695 007626 005721      1$:  TST      (R1)+
2696 007630 001046      BNE     EEB15      ;BRANCH IF NOT CORRECT.
2697 007632 077203      SOB     R2,1$
2698
2699 007634 020027 007702  CMP     R0,#EEBBF1    ;IS R0 CORRECT?
2700 007640 001055      BNE     EEB20      ;BRANCH IF NOT CORRECT.
2701 007642 022705 000204  CMP     #204,R5      ;IS THE FPS CORRECT?
2702 007646 001061      BNE     EEB25      ;BRANCH IF NOT CORRECT.
2703 007650 000466      BR      EEBDONE
2704
2705      ;THESE ARE TEST DATA TABLES AND A BUFFER.
2706 007652 000177      EEBTP1: 177
2707 007654 167574      167574
2708 007656 137271      137271
2709 007660 107675      107675
2710 007662 000000      EEBTP2: 0
2711 007664 000000      0
2712 007666 000000      0
2713 007670 000000      0
2714 007672 177777      EEBBF0: -1
2715 007674 177777      -1
2716 007676 177777      -1
2717 007700 177777      -1
2718 007702 177777      EEBBF1: -1
2719 007704 177777      -1
2720 007706 177777      -1
2721 007710 177777      -1
2722
2723      ;IF A TRAP TO 4 OCCURS COME HERE:
2724 007712 011602      EEB10: MOV      (SP),R2      ;SEE IF THE TRAP OCCURRED ON THE TEST INSTR.
2725 007714 020227 007614  CMP     R2,#EEB2+2
2726 007720 001405      BEQ     1$      ;BRANCH IF YES.
2727 007722 020227 007616  CMP     R2,#EEB2+4
2728 007726 001402      BEQ     1$      ;BRANCH IF YES.
2729 007730 000137 036172  JMP     @#CPSPUR    ;OTHERWISE GO REPORT A SPURIOUS TRAP TO 4.
2730      ;REPORT A FAILURE IN THE FDST FLOWS RESULTED IN AN ODD ADDRESS TRAP TO 4.
2731 007734 022626      1$:  CMP     (SP)+,(SP)+      ;RESET THE STACK.
2732 007736 010237 001236  MOV     R2,@#STMP2
2733 007742 104002      2$:  ERROR   2      ;ODD ADRES
2734 007744 000430      BR      EEBDONE      ;BUT FDSTX IN ST 771
2735
2736      ;REPORT RESULT INCORRECT.
2737 007746 012737 007662 001242  EEB15: MOV     #EEBTP2,@#STMP4
2738 007754 012737 007652 001240  MOV     #EEBTP1,@#STMP3
```

```
2739 007762 012737 007702 001244      MOV    #EEBF1,@#STMP5
2740 007770 104002      1$:    ERROR    2          ;BAD DATA X11*0 ST 312X
2741 007772 000415      BR     EEBDONE
2742
2743      ;RO INCORRECT:
2744 007774 012737 007702 001240  EEB20: MOV    #EEBF1,@#STMP3
2745 010002 010037 001242      MOV    R0,@#STMP4
2746 010006 104002      1$:    ERROR    2          ;RO BADX
2747 010010 000406      BR     EEBDONE
2748
2749      ;REPORT FPS INCORRECT:
2750 010012 010537 001240  EEB25: MOV    R5,@#STMP3
2751 010016 012737 000204 001244  MOV    #204,@#STMP5
2752 010024 104002      1$:    ERROR    2          ;FPS X
2753
2754 010026      EEBDONE:
2755 010026 104413      RSETUP          ;GO INITIALIZE THE FPS AND STACK; AND
2756                      ;SEE IF THE USER HAS EXPRESSED
2757                      ;THE DESIRE TO CHANGE THE SOFTWARE
2758                      ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
2759                      ;THE USER TYPED CONTROL G?).
2760
2761      ;*****
2762      ;*TEST 22      NEGF, ABSF AND TSTF SOURCE MODE 2 TEST
2763      ;*
2764      ;*THIS IS A TEST THE NEGF, ABSF AND TSTF
2765      ;*SOURCE FLOWS. THE ABSD INSTRUCTION
2766      ;*IS USED TO TEST MODE 2
2767      ;*
2768      ;*****
2769 010030 000004  TS122: SCOPE
2770
2771 010032      FFB1:
2772 010032 104414      LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
2773 010034 012700 010142  MOV    #FFB1P1,R0      ;SET UP THE DATA BUFFER.
2774 010040 012701 010172  MOV    #FFBF1,R1
2775 010044 012702 000004  MOV    #4,R2
2776 010050 012021      1$:    MOV    (R0)+,(R1)+
2777 010052 077202      SOB    R2,1$
2778 010054 012700 000200  MOV    #200,R0        ;SET FD.
2779 010060 170100      LDFPS  R0
2780 010062 012700 010172  MOV    #FFBF1,R0      ;SET UP THE OPERAND ADDRESS.
2781 010066 012737 010102 001236  MOV    #FFB2,@#STMP2
2782 010074 012737 010202 000004  MOV    #FFB10,@#ERRVECT ;SET UP VECTOR 4 IN CASE OF AN ERROR.
2783
2784 010102 170620      FFB2:  ABSD    (R0)+    ;TEST INSTRUCTION.
2785
2786 010104 170205      STFPS  R5            ;GET FPS.
2787 010106 012701 010172  MOV    #FFBF1,R1      ;CHECK RESULT.
2788 010112 012702 000004  MOV    #4,R2
2789 010116 005721      1$.    TST    (R1)+
2790 010120 001046      BNE   FFB15          ;BRANCH IF INCORRECT.
2791 010122 077203      SOB    R2,1$
2792
2793 010124 020027 010202      CMP    R0,#FFBF1+10  ;IS R0 CORRECT?
2794 010130 001055      BNE   FFB20          ;BRANCH IF INCORRECT.
```

```
2795 010132 022705 000204      CMP      #204,R5      ;IS THE FPS CORRECT?
2796 010136 001061              BNE      FFB25       ;BRANCH IF INCORRECT.
2797 010140 000466              BR       FFBDONE
2798
2799      ;THESE ARE TEST DATA TABLES AND DATA BUFFER.
2800 010142 000177      FFBTP1: 177
2801 010144 167574              167574
2802 010146 137271              137271
2803 010150 107675              107675
2804 010152 000000      FFBTP2: 0
2805 010154 000000              0
2806 010156 000000              0
2807 010160 000000              0
2808 010162 177777      FFBFBF0: -1
2809 010164 177777              -1
2810 010166 177777              -1
2811 010170 177777              -1
2812 010172 177777      FFBFBF1: -1
2813 010174 177777              -1
2814 010176 177777              -1
2815 010200 177777              -1
2816
2817      ;IF A TRAP TO 4 OCCURS COME HERE.
2818 010202 011602      FFB10: MOV      (SP),R2      ;SEE IF THE TRAP OCCURRED ON THE TEST INSTRUCTION.
2819 010204 020227 010104      CMP      R2,#FFB2+2
2820 010210 001405              BEQ      1$           ;BRANCH IF YES.
2821 010212 020227 010106      CMP      R2,#FFB2+4
2822 010216 001402              BEQ      1$           ;BRANCH IF YES.
2823 010220 000137 036172      JMP      @WCPSPUR     ;OTHERWISE GO REPORT SPURIOUS TRAP TO 4.
2824      ;REPORT AN FDST FLOW FAILURE RESULTED IN A TRAP TO 4.
2825 010224 022626      1$:  CMP      (SP)+,(SP)+
2826 010226 010237 001236      MOV      R2,@$TMP2
2827 010232 104002      2$:  ERROR   2           ;ODD ADRES
2828 010234 000430              BR       FFBDONE     ;BUT FDSTX IN ST 771
2829
2830      ;REPORT RESULT INCORRECT:
2831 010236 012737 010152 001240      FFB15: MOV      #FFBTP2,@$TMP3
2832 010244 012737 010142 001242      MOV      #FFBTP1,@$TMP4
2833 010252 012737 010172 001244      MOV      #FFBFBF1,@$TMP5
2834 010260 104002      1$:  ERROR   2           ;BAD DATA X11*0 ST 312X
2835 010262 000415              BR       FFBDONE
2836
2837      ;REPORT RO INCORRECT:
2838 010264 012737 010176 001240      FFB20: MOV      #FFBFBF1+4,@$TMP3
2839 010272 010037 001242      MOV      R0,@$TMP4
2840 010276 104002      1$:  ERROR   2           ;RO BADX
2841 010300 000406              BR       FFBDONE
2842
2843      ;REPORT FPS INCORRECT:
2844 010302 010537 001240      FFB25: MOV      R5,@$TMP3
2845 010306 012737 000204 001244      MOV      #204,@$TMP5
2846 010314 104002      1$:  ERROR   2           ;FPS X
2847
2848 010316      FFBDONE:
2849 010316 104413      RSETUP
2850      ;GO INITIALIZE THE FPS AND STACK; AND
      ;SEF IF THE USER HAS EXPRESSED
```

```
2851                                     ;THE DESIRE TO CHANGE THE SOFTWARE
2852                                     ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
2853                                     ;THE USER TYPED CONTROL G?).
2854
2855 :*****
2856 :*TEST 23      NEGF, ABSF AND TSTF SOURCE MODE 4 TEST
2857 :*
2858 :*THIS IS A TEST THE NEGF, ABSF AND TSTF
2859 :*SOURCE FLOWS. THE ABSD INSTRUCTION
2860 :*IS USED TO TEST MODE 4
2861 :*
2862 :*****
2862 010320 000004 TST23: SCOPE
2863
2864 010322 GGB1:
2865 010322 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
2866 010324 012700 010432 MOV #GGBTP1,R0 ;SET UP THE DATA BUFFER.
2867 010330 012701 010452 MOV #GGBBF0,R1
2868 010334 012702 000004 MOV #4,R2
2869 010340 012021 1$: MOV (R0)+,(R1)+
2870 010342 077202 SOB R2,1$
2871 010344 012700 000200 MOV #200,R0 ;SET FD.
2872 010350 170100 LDFPS R0
2873 010352 012700 010462 MOV #GGBBF1,R0 ;SET UP THE OPERAND ADDRESS.
2874 010356 012737 010372 001236 MOV #GGB2,@#STMP2
2875 010364 012737 010472 000004 MOV #GGB10,@#ERRVECT ;SET UP VECTOR 4 IN CASE OF AN ERROR.
2876
2877 010372 170640 GGB2: ABSD -(R0) ;TEST INSTRUCTION.
2878
2879 010374 170205 STFPS R5 ;GET FPS.
2880 010376 012701 010452 MOV #GGBBF0,R1 ;CHECK RESULT.
2881 010402 012702 000004 MOV #4,R2
2882 010406 005721 1$: TST (R1)+
2883 010410 001046 BNE GGB15 ;BRANCH IF INCORRECT.
2884 010412 077203 SOB R2,1$
2885
2886 010414 020027 010452 CMP R0,#GGBBF0 ;IS R0 CORRECT?
2887 010420 001055 BNE GGB20 ;BRANCH IF INCORRECT.
2888 010422 022705 000204 CMP #204,R5 ;IS THE FPS CORRECT?
2889 010426 001061 BNE GGB25 ;BRANCH IF INCORRECT.
2890 010430 000466 BR GGBDONE
2891
2892 :THESE ARE TEST DATA TABLES AND DATA BUFFER.
2893 010432 000177 GGBTP1: 177
2894 010434 117273 117273
2895 010436 147576 147576
2896 010440 177071 177071
2897 010442 000000 GGBTP2: 0
2898 010444 000000 0
2899 010446 000000 0
2900 010450 000000 0
2901 010452 177777 GGBBF0: -1
2902 010454 177777 -1
2903 010456 177777 -1
2904 010460 177777 -1
2905 010462 177777 GGBBF1: -1
2906 010464 177777 -1
```



```
2907 010466 177777 -1
2908 010470 177777 -1
2909
2910 ;IF A TRAP TO 4 OCCURS COME HERE.
2911 010472 011602 GGB10: MOV (SP),R2 ;SEE IF THE TRAP OCCURRED ON THE TEST INSTRUCTION.
2912 010474 020227 010374 CMP R2,#GGB2+2
2913 010500 001405 BEQ 1$ ;BRANCH IF YES.
2914 010502 020227 010376 CMP R2,#GGB2+4
2915 010506 001402 BEQ 1$ ;BRANCH IF YES.
2916 010510 000137 036172 JMP @#CPSPUR ;OTHERWISE GO REPORT SPURIOUS TRAP TO 4.
2917 ;REPORT AN FDST FLOW FAILURE RESULTED IN A TRAP TO 4.
2918 010514 022626 1$: CMP (SP)+,(SP)+
2919 010516 010237 001236 MOV R2,@#STMP2
2920 010522 104002 2$: ERROR 2 ;ODD ADRES
2921 010524 000430 BR GGBDONE ;BUT FDSTX IN ST 771
2922
2923 ;REPORT RESULT INCORRECT:
2924 010526 012737 010442 001240 GGB15: MOV #GGBTP2,@#STMP3
2925 010534 012737 010432 001242 MOV #GGBTP1,@#STMP4
2926 010542 012737 010452 001244 MOV #GGBBF0,@#STMP5
2927 010550 104002 1$: ERROR 2 ;BAD DATA X11*0 ST 312X
2928 010552 000415 BR GGBDONE
2929
2930 ;REPORT R0 INCORRECT:
2931 010554 012737 010452 001240 GGB20: MOV #GGBBF01,@#STMP3
2932 010562 010037 001242 MOV R0,@#STMP4
2933 010566 104002 1$: ERROR 2 ;R0 BADX
2934 010570 000406 BR GGBDONE
2935
2936 ;REPORT FPS INCORRECT:
2937 010572 010537 001240 GGB25: MOV R5,@#STMP3
2938 010576 012737 000204 001244 MOV #204,@#STMP5
2939 010604 104002 1$: ERROR 2 ;FPS X
2940
2941 GGBDONE:
2942 010606 104413 RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
2943 ;SEE IF THE USER HAS EXPRESSED
2944 ;THE DESIRE TO CHANGE THE SOFTWARE
2945 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
2946 ;THE USER TYPED CONTROL G?).
2947 ;:*****
2948 ;*TEST 24 NEGF, ABSF AND TSTF SOURCE MODE 3 TEST
2949 ;*
2950 ;*THIS IS A TEST THE NEGF, ABSF AND TSTF
2951 ;*SOURCE FLOWS. THE ABSD INSTRUCTION
2952 ;*IS USED TO TEST MODE 3
2953 ;*
2954 ;:*****
2955 010610 000004 TST24: SCOPE
2956
2957 HHB1:
2958 010612 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
2959 010614 012700 010722 MOV #HHBTP1,R0 ;SET UP THE DATA BUFFER.
2960 010620 012701 010752 MOV #HHBBF0,R1
2961 010624 012702 000010 MOV #10,R2
2962 010630 012021 1$: MOV (R0)+,(R1)+
```

```
2963 010632 077202          SOB      R2,1$
2964 010634 012700 000200    MOV      #200,R0          ;SET FD.
2965 010640 170100          LDFPS   R0
2966 010642 012700 010762    MOV      #HHBF1,R0       ;SET UP THE OPERAND ADDRESS.
2967 010646 012737 010662 001236  MOV      #HHB2,@#STMP2
2968 010654 012737 010772 000004  MOV      #HHB10,@#ERRVECT ;SET UP VECTOR 4 IN CASE OF AN ERROR.
2969
2970 010662 170630          -HHB2:  ABSD   @(R0)+      ;TEST INSTRUCTION.
2971
2972 010664 170205          STFPS   R5                ;GET FPS.
2973 010666 012701 010752    MOV      #HHBF0,R1       ;CHECK RESULT.
2974 010672 012702 000004    MOV      #4,R2
2975 010676 005721          1$:     TST      (R1)+
2976 010700 001052          BNE     HHB15             ;BRANCH IF INCORRECT.
2977 010702 077203          SOB     R2,1$
2978 010704 020027 010764    CMP     R0,#HHBF1+2      ;IS R0 CORRECT?
2979 010710 001061          BNE     HHB20             ;BRANCH IF INCORRECT.
2980 010712 022705 000204    CMP     #204,R5          ;IS THE FPS CORRECT?
2981 010716 001065          BNE     HHB25             ;BRANCH IF INCORRECT.
2982 010720 000472          BR      HHBDONE
2983
2984          ;THESE ARE TEST DATA TABLES AND DATA BUFFER.
2985 010722 000177          HHBTP1: 177
2986 010724 147576          147576
2987 010726 177071          177071
2988 010730 107576 010752 177777  107576,HHBF0,-1,-1,-1
2989 010736 177777          177777
2990 010742 000000 000000 000000 HHBTP2: 0,0,0,0
2991 010750 000900
2992 010752 177777          HHBF0:  -1
2993 010754 177777          -1
2994 010756 177777          -1
2995 010760 177777          -1
2996 010762 177777          HHBF1:  -1
2997 010764 177777          -1
2998 010766 177777          -1
2999 010770 177777          -1
3000
3001          ;IF A TRAP TO 4 OCCURS COME HERE.
3002 010772 011602          HHB10:  MOV      (SP),R2      ;SEE IF THE TRAP OCCURRED ON THE TEST INSTRUCTION.
3003 010774 020227 010664    CMP     R2,#HHB2+2
3004 011000 001405          BEQ     1$                ;BRANCH IF YES.
3005 011002 020227 010666    CMP     R2,#HHB2+4
3006 011006 001402          BEQ     1$                ;BRANCH IF YES.
3007 011010 000137 036172    JMP     @#CPSPUR          ;OTHERWISE GO REPORT SPURIOUS TRAP TO 4.
3008          ;REPORT AN FDST FLOW FAILURE RESULTED IN A TRAP TO 4.
3009 011014 022626          1$:     CMP     (SP)+,(SP)+
3010 011016 010237 001236    MOV     R2,@#STMP2
3011 011022 104002          2$:     ERROR   2          ;ODD ADRES
3012 011024 000430          BR      HHBDONE          ;BUT FDSTX IN ST 771
3013
3014          ;REPORT RESULT INCORRECT:
3015 011026 012737 010742 001240  HHB15:  MOV      #HHBTP2,@#STMP3
3016 011034 012737 010722 001242    MOV     #HHBTP1,@#STMP4
3017 011042 012737 010752 001244    MOV     #HHBF0,@#STMP5
3018 011050 104002          1$:     ERROR   2          ;BAD DATA X11*0 ST 3127
```

```
3019 011052 000415 BR HMBDONE
3020
3021 ;REPORT R0 INCORRECT:
3022 011054 012737 010764 001240 HMB20: MOV #HMBBF1+2 @#STMP3
3023 011062 010037 001242 MOV R0,@#STMP4
3024 011066 104002 1$: ERROR 2 ;R0 INCORRECT.
3025 011070 000406 BR HMBDONE
3026 ;REPORT FPS INCORRECT:
3027 011072 010537 001240 HMB25: MOV R5,@#STMP3
3028 011076 012737 000204 001244 MOV #204,@#STMP5
3029 011104 104002 1$: ERROR 2 ;FPSX
3030
3031 011106 HMBDONE:
3032 011106 104413 RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
3033 ;SEE IF THE USER HAS EXPRESSED
3034 ;THE DESIRE TO CHANGE THE SOFTWARE
3035 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
3036 ;THE USER TYPED CONTROL G?).
3037 ;*****
3038 ;*TEST 25 NEGF, ABSF AND TSTF SOURCE MODE 5 TEST
3039 ;*
3040 ;*THIS IS A TEST THE NEGF, ABSF AND TSTF
3041 ;*SOURCE FLOWS. THE NEGD INSTRUCTION
3042 ;*IS USED TO TEST MODE 5
3043 ;*
3044 ;*****
3045 011110 000004 TST25: SCOPE
3046
3047 011112 IIB1:
3048 011112 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
3049 011114 012700 011222 MOV #IIBTP1,R0 ;SET UP THE DATA BUFFER.
3050 011120 012701 011252 MOV #IIBBF0,R1
3051 011124 012702 000010 MOV #10,R2
3052 011130 012021 1$: MOV (R0)+,(R1)+
3053 011132 077202 SOB R2,1$
3054 011134 012700 000200 MOV #200,R0 ;SET FD.
3055 011140 170100 LDFPS R0
3056 011142 012700 011264 MOV #IIBBF1+2,R0 ;SET UP THE OPERAND ADDRESS.
3057 011146 012737 011162 001236 MOV #IIB2,@#STMP2
3058 011154 012737 011272 000004 MOV #IIB10,@#ERRVECT ;SET UP VECTOR 4 IN CASE OF AN ERROR.
3059
3060 011162 170750 IIB2: NEGD @-(R0) ;TEST INSTRUCTION.
3061
3062 011164 170205 STFPS R5 ;GET FPS.
3063 011166 012701 011252 MOV #IIBBF0,R1 ;CHECK RESULT.
3064 011172 012702 000004 MOV #4,R2
3065 011176 005721 1$: TST (R1)+
3066 011200 001052 BNE IIB15 ;BRANCH IF INCORRECT.
3067 011202 077203 SOB R2,1$
3068 011204 020027 011262 CMP R0,#IIBBF1 ;IS R0 CORRECT?
3069 011210 001061 BNE IIB20 ;BRANCH IF INCORRECT.
3070 011212 022705 000204 CMP #204,R5 ;IS THE FPS CORRECT?
3071 011216 001065 BNE IIB25 ;BRANCH IF INCORRECT.
3072 011220 000472 BR IIBDONE
3073
3074 ;THESE ARE TEST DATA TABLES AND DATA BUFFER.
```

```
3075 011222 000176 IIBTP1: 176
3076 011224 177074 177074
3077 011226 127374 127374
3078 011230 157677 011252 177777 157677,IIBBF0,-1,-1,-1
3079 011236 177777 177777
3080 011242 000000 IIBTP2: 0
3081 011244 000000 0
3082 011246 000000 0
3083 011250 000000 0
3084 011252 177777 IIBBF0. -1
3085 011254 177777 -1
3086 011256 177777 -1
3087 011260 177777 -1
3088 011262 177777 IIBBF1: -1
3089 011264 177777 -1
3090 011266 177777 -1
3091 011270 177777 -1
3092
3093 ;IF A TRAP TO 4 OCCURS COME HERE.
3094 011272 011602 IIB10: MOV (SP),R2 ;SEE IF THE TRAP OCCURRED ON THE TEST INSTRUCTION.
3095 011274 020227 011164 CMP R2,#IIB2+2
3096 011300 001405 BEQ 1$ ;BRANCH IF YES.
3097 011302 020227 011166 CMP R2,#IIB2+4
3098 011306 001402 BEQ 1$ ;BRANCH IF YES.
3099 011310 000137 036172 JMP @#CPSPUR ;OTHERWISE GO REPORT SPURIOUS TRAP TO 4.
3100 ;REPORT AN FDST FLOW FAILURE RESULTED IN A TRAP TO 4.
3101 011314 022626 1$: CMP (SP)+,(SP)+
3102 011316 010237 001236 MOV R2,@#STMP2
3103 011322 104002 2$: ERROR 2 ;ODD ADRES
3104 011324 000430 BR IIBDONE ;BUT FDS'X IN ST 771
3105
3106 ;REPORT RESULT INCORRECT:
3107 011326 012737 011242 001240 IIB15: MOV #IIBTP2,@#STMP3
3108 011334 012737 011222 001242 MOV #IIBTP1,@#STMP4
3109 011342 012737 011252 001244 MOV #IIBBF0,@#STMP5
3110 011350 104002 1$: ERROR 2 ;BAD DATA X11*0 ST 3127
3111 011352 000415 BR IIBDONE
3112
3113 ;REPORT R0 INCORRECT:
3114 011354 012737 011262 001240 IIB20: MOV #IIBBF1,@#STMP3
3115 011362 010037 001242 MOV R0,@#STMP4
3116 011366 104002 1$: ERROR 2 ;R0 BADX
3117 011370 000406 BR IIBDONE
3118 ;REPORT FPS INCORRECT:
3119 011372 010537 001240 IIB25: MOV R5,@#STMP3
3120 011376 012737 000204 001244 MOV #204,@#STMP5
3121 011404 104002 1$: ERROR 2 ;FPSX
3122
3123 011406 IIBDONE:
3124 011406 104413 RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
3125 ;SEE IF THE USER HAS EXPRESSED
3126 ;THE DESIRE TO CHANGE THE SOFTWARE
3127 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
3128 ;THE USER TYPED CONTROL G?).
3129
3130 ;:*****
```

```
3131 ;*TEST 26 NEGF, ABSF AND TSTF SOURCE MODE 6 TEST
3132 ;*
3133 ;*THIS IS A TEST THE NEGF, ABSF AND TSTF
3134 ;*SOURCE FLOWS. THE ABSD INSTRUCTION
3135 ;*IS USED TO TEST MODE 6
3136 ;*
3137 ;*****
3138 011410 000004 TST26: SCOPE
3139
3140 011412 JJB1:
3141 011412 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
3142 011414 012700 011524 MOV #JJBTP1,R0 ;SET UP THE DATA BUFFER.
3143 011420 012701 011546 MOV #JJBFF0,R1
3144 011424 012702 000004 MOV #4,R2
3145 011430 012021 1$: MOV (R0)+,(R1)+
3146 011432 077202 SOB R2,1$
3147 011434 012700 000200 MOV #200,R0 ;SET FD.
3148 011440 170100 LDFPS R0
3149 011442 012700 011537 MOV #JJBFF0-7,R0 ;SET UP THE OPERAND ADDRESS.
3150 011446 012737 011462 001236 MOV #JJB2,@#STMP2
3151 011454 012737 011566 000004 MOV #JJB10,@#ERRVECT ;SET UP VECTOR 4 IN CASE OF AN ERROR.
3152
3153 011462 170660 000007 JJB2: ABSD 7(R0) ;TEST INSTRUCTION.
3154
3155 011466 170205 STFPS R5 ;GET FPS.
3156 011470 012701 011546 MOV #JJBFF0,R1 ;CHECK RESULT.
3157 011474 012702 000004 MOV #4,R2
3158 011500 005721 1$: TST (R1)+
3159 011502 001047 BNE JJB15 ;BRANCH IF INCORRECT.
3160 011504 077203 SOB R2,1$
3161 011506 020027 011537 CMP R0,#JJBFF0-7 ;IS R0 CORRECT?
3162 011512 001043 BNE JJB15 ;BRANCH IF INCORRECT.
3163 011514 022705 000204 CMP #204,R5 ;IS THE FPS CORRECT?
3164 011520 001053 BNE JJB20 ;BRANCH IF INCORRECT.
3165 011522 000467 BR JJBDONE
3166
3167 ;THESE ARE TEST DATA TABLES AND DATA BUFFER.
3168 011524 000177 JJBTP1: 177
3169 011526 161524 161524
3170 011530 131273 131273
3171 011532 107174 000000 107174,
3172 011536 000000 JJBTP2: 0
3173 011540 000000 0
3174 011542 000000 0
3175 011544 000000 0
3176 011546 177777 JJBFF0: -1
3177 011550 177777 -1
3178 011552 177777 -1
3179 011554 177777 -1
3180 011556 177777 JJBFF1: -1
3181 011560 177777 -1
3182 011562 177777 -1
3183 011564 177777 -1
3184
3185 ;IF A TRAP TO 4 OCCURS COME HERE.
3186 011566 011602 JJB10: MOV (SP),R2 ;SEE IF THE TRAP OCCURRED ON THE TEST INSTRUCTION.
```

```

3187 011570 020227 011464      CMP      R2,#JJB2+2
3188 011574 001405      BEQ      1$ ;BRANCH IF YES.
3189 011576 020227 011466      CMP      R2,#JJB2+4
3190 011602 001402      BEQ      1$ ;BRANCH IF YES.
3191 011604 000137 036172      JMP      @#CPSPUR ;OTHERWISE GO REPORT SPURIOUS TRAP TO 4.
3192 ;REPORT AN FDST FLOW FAILURE RESULTED IN A TRAP TO 4.
3193 011610 022626      1$:      CMP      (SP)+,(SP)+
3194 011612 010237 001236      MOV      R2,@#STMP2
3195 011616 104002      2$:      ERROR   2 ;ODD ADRES
3196 011620 000430      BR       JJB DONE ;BUT FDSTX IN ST 771
3197
3198 ;REPORT RESULT INCORRECT:
3199 011622 012737 011536 001240 JJB15:   MOV      #JJBTP2,@#STMP3
3200 011630 012737 011524 001242      MOV      #JJBTP1,@#STMP4
3201 011636 012737 011546 001244      MOV      #JJBFB0,@#STMP5
3202 011644 104002      1$:      ERROR   2 ;BAD DATA X11*0 ST 3127
3203 011646 000415      BR       JJB DONE
3204
3205 ;REPORT R0 INCORRECT:
3206 011650 012737 011537 001240 JJB20:   MOV      #JJBFB0-7,@#STMP3
3207 011656 010037 001242      MOV      R0,@#STMP4
3208 011662 104002      1$:      ERROR   2 ;R0 BADX
3209 011664 000406      BR       JJB DONE
3210 ;REPORT FPS INCORRECT:
3211 011666 010537 001240 JJB25:   MOV      R5,@#STMP3
3212 011672 012737 000204 001244      MOV      #204,@#STMP5
3213 011700 104002      1$:      ERROR   2 ;FPSX
3214 011702      JJB DONE:
3215 011702 104413      RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
3216 ;SEE IF THE USER HAS EXPRESSED
3217 ;THE DESIRE TO CHANGE THE SOFTWARE
3218 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
3219 ;THE USER TYPED CONTROL G?).
3220 ;*****
3221 ;*TEST 27 NEGFL, ABSF AND TSTF SOURCE MODE 7 TEST
3222 ;*
3223 ;*THIS IS A TEST THE NEGFL, ABSF AND TSTF
3224 ;*SOURCE FLOWS. THE ABSD INSTRUCTION
3225 ;*IS USED TO TEST MODE 6
3226 ;*
3227 ;*****
3228 011704 000004      TST27:  SCOPE
3229
3230 KKB1:
3231 011706      LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
3232 011710 104414      MOV      #KKBTP1,R0 ;SET UP THE DATA BUFFER.
3233 011714 012700 012020      MOV      #KKBFB0,R1
3234 011720 012701 012050      MOV      #10,R2
3235 011724 012702 000010      1$:      MOV      (R0)+,(R1)+
3236 011726 012021      SOB     R2,1$
3237 011726 077202      MOV      #200,R0 ;SET FD.
3238 011730 012700 000200      LDFPS   R0
3239 011734 170100      MOV      #KKBFB1-7,R0 ;SET UP THE OPERAND ADDRESS.
3240 011736 012700 012051      MOV      #KKB2,@#STMP2
3241 011742 012737 011756 001236      MOV      #KKB10,@#ERRVECT ;SET UP VECTOR 4 IN CASE OF AN ERROR.
3242 011750 012737 012070 000004
    
```

```
3243 011756 170770 000007      KKB2:  NEGD  @7(R0)           ;TEST INSTRUCTION.
3244
3245 011762 170205              STFPS  R5           ;GET FPS.
3246 011764 012701 012050      MOV    #KKBBF0,R1   ;CHECK RESULT.
3247 011770 012702 000004      MOV    #4,R2
3248 011774 005721              1$:   TST    (R1)+
3249 011776 001052              BNE    KKB15        ;BRANCH IF INCORRECT.
3250 012000 077203              SOB    R2,1$
3251 012002 020027 012051      CMP    R0,#KKBBF1-7 ;IS R0 CORRECT?
3252 012006 001061              BNE    KKB20        ;BRANCH IF INCORRECT.
3253 012010 022705 000204      CMP    #204,R5     ;IS THE FPS CORRECT?
3254 012014 001056              BNE    KKB20        ;BRANCH IF INCORRECT.
3255 012016 000472              BR     KKBDONE
3256
3257                               ;THESE ARE TEST DATA TABLES AND DATA BUFFER.
3258 012020 000177      KKBTP1: 177
3259 012022 167574      167574
3260 012024 137271      137271
3261 012026 107675 012050 177777 107675,KKBBF0,-1,-1,-1
3262 012034 177777 177777
3263 012040 000000      KKBTP2: 0
3264 012042 000000      0
3265 012044 000000      0
3266 012046 000000      0
3267 012050 177777      KKBBF0: -1
3268 012052 177777      -1
3269 012054 177777      -1
3270 012056 177777      -1
3271 012060 177777      KKBBF1: -1
3272 012062 177777      -1
3273 012064 177777      -1
3274 012066 177777      -1
3275
3276                               ;IF A TRAP TO 4 OCCURS COME HERE.
3277 012070 011602      KKB10: MOV    (SP),R2   ;SEE IF THE TRAP OCCURRED ON THE TEST INSTRUCTION.
3278 012072 020227 011760      CMP    R2,#KKB2+2
3279 012076 001405      BEQ    1$           ;BRANCH IF YES.
3280 012100 020227 011762      CMP    R2,#KKB2+4
3281 012104 001402      BEQ    1$           ;BRANCH IF YES.
3282 012106 000137 036172      JMP    @#CPSPUR    ;OTHERWISE GO REPORT SPURIOUS TRAP TO 4.
3283                               ;REPORT AN FDST FLOW FAILURE RESULTED IN A TRAP TO 4.
3284 012112 022626      1$:   CMP    (SP)+,(SP)+
3285 012114 010237 001236      MOV    R2,@#TMP2
3286 012120 104002      2$:   ERROR  2           ;ODD ADRES
3287 012122 000430      BR     KKBDONE     ;BUT FDSTX IN ST 771
3288
3289                               ;REPORT RESULT INCORRECT:
3290 012124 012737 012040 001240  KKB15: MOV    #KKBTP2,@#TMP3
3291 012132 012737 012020 001242      MOV    #KKBTP1,@#TMP4
3292 012140 012737 012050 001244      MOV    #KKBBF0,@#TMP5
3293 012146 104002      1$:   ERROR  2           ;BAD DATA X11*0 ST 3127
3294 012150 000415      BR     KKBDONE
3295
3296                               ;REPORT R0 INCORRECT:
3297 012152 012737 012051 001240  KKB20: MOV    #KKBBF1-7,@#TMP3
3298 012160 010037 001242      MOV    R0,@#TMP4
```

```

3299 012164 104002          1$:      ERROR      2          ;RO BADX
3300 012166 000406          BR          KKB DONE
3301          ;REPORT FPS INCORRECT:
3302 012170 010537 001240   KKB25:  MOV          R5,@#STMP3
3303 012174 012737 000204 001244   MOV          #204,@#STMP5
3304 012202 104002          1$:      ERROR      2          ;FPSX
3305
3306 012204
3307 012204 104413          KKB DONE:
3308          RSETUP
3309          ;GO INITIALIZE THE FPS AND STACK; AND
3310          ;SEE IF THE USER HAS EXPRESSED
3311          ;THE DESIRE TO CHANGE THE SOFTWARE
3312          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
3313          ;THE USER TYPED CONTROL G?).
3314          ;*****
3315          ;*TEST 30          NEGF, ABSF AND TSTF SOURCE MODE 6, GR7, TEST
3316          ;*
3317          ;*THIS IS A TEST THE NEGF, ABSF AND TSTF
3318          ;*SOURCE FLOWS. THE NEGD INSTRUCTION
3319          ;*IS USED TO TEST MODE 6
3320          ;*
3321          ;*****
3322 012206 000004          TST30:  SCOPE
3323 012210          LLB1:
3324 012210 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
3325 012212 012700 012310   MOV          #LLBTP1,R0          ;SET UP THE DATA BUFFER.
3326 012216 012701 012330   MOV          #LLBBF0,R1
3327 012222 012702 000004   MOV          #4,R2
3328 012226 012021          1$:      MOV          (R0)+,(R1)+
3329 012230 077202          SOB          R2,1$
3330 012232 012700 000200   MOV          #200,R0          ;SET FD.
3331 012236 170100          LDFPS       R0
3332 012240 012737 012254 001236   MOV          #LLB2,@#STMP2
3333 012246 012737 012350 000004   MOV          #LLB10,@#ERRVECT ;SET UP VECTOR 4 IN CASE OF AN ERROR.
3334
3335 012254 170767 000050          LLB2:  NEGD      LLBBF0          ;TEST INSTRUCTION.
3336
3337 012260 170205          STFPS       R5          ;GET FPS.
3338 012262 012701 012330   MOV          #LLBBF0,R1          ;CHECK RESULT.
3339 012266 012702 000004   MOV          #4,R2
3340 012272 005721          1$:      TST          (R1)+
3341 012274 001043          BNE         LLB15          ;BRANCH IF INCORRECT.
3342 012276 077203          SOB          R2,1$
3343 012300 022705 000204   CMP          #204,R5          ;IS THE FPS CORRECT?
3344 012304 001052          BNE         LLB25          ;BRANCH IF INCORRECT.
3345 012306 000457          BR          LLBDONE
3346
3347          ;THESE ARE TEST DATA TABLES AND DATA BUFFER.
3348          LLBTP1: 127
3349          137475
3350          147372
3351          117057
3352          LLBTP2: 0
3353          0
3354          0
3355          0
3356          LLBBF0: -1
  
```



```
3355 012332 177777 -1
3356 012334 177777 -1
3357 012336 177777 -1
3358 012340 177777 LLBBF1: -1
3359 012342 177777 -1
3360 012344 177777 -1
3361 012346 177777 -1
3362
3363 ;IF A TRAP TO 4 OCCURS COME HERE.
3364 012350 011602 LLB10: MOV (SP),R2 ;SEE IF THE TRAP OCCURRED ON THE TEST INSTRUCTION.
3365 012352 020227 012256 CMP R2,#LLB2+2
3366 012356 001405 BEQ 1$ ;BRANCH IF YES.
3367 012360 020227 012260 CMP R2,#LLB2+4
3368 012364 001402 BEQ 1$ ;BRANCH IF YES.
3369 012366 000137 036172 JMP @#CPSPUR ;OTHERWISE GO REPORT SPURIOUS TRAP TO 4.
3370 ;REPORT AN FDST FLOW FAILURE RESULTED IN A TRAP TO 4.
3371 012372 022626 1$: CMP (SP)+,(SP)+
3372 012374 010237 001236 MOV R2,@#STMP2
3373 012400 104002 2$: ERROR 2 ;ODD ADRES
3374 012402 000421 BR LLBDONE ;BUT FDSTX IN ST 771
3375
3376 ;REPORT RESULT INCORRECT:
3377 012404 012737 012320 001240 LLB15: MOV #LLBTP2,@#STMP3
3378 012412 012737 012310 001242 MOV #LLBTP1,@#STMP4
3379 012420 012737 012330 001244 MOV #LLBBF0,@#STMP5
3380 012426 104002 1$: ERROR 2 ;BAD DATA X11*0 ST 3127
3381 012430 000406 BR LLBDONE
3382 ;REPORT FPS INCORRECT:
3383 012432 010537 001240 LLB25: MOV R5,@#STMP3
3384 012436 012737 000204 001244 MOV #204,@#STMP5
3385 012444 104002 1$: ERROR 2 ;FPSX
3386
3387 012446 LLBDONE:
3388 012446 104413 RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
3389 ;SEE IF THE USER HAS EXPRESSED
3390 ;THE DESIRE TO CHANGE THE SOFTWARE
3391 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
3392 ;THE USER TYPED CONTROL G?).
3393
3394 ;*****
3395 ;*TEST 31 NEG, ABSF AND TSTF SOURCE MODE 7, GR7, TEST
3396 ;*
3397 ;*THIS IS A TEST THE NEG, ABSF AND TSTF
3398 ;*SOURCE FLOWS. THE ABSD INSTRUCTION
3399 ;*IS USED TO TEST MODE 7
3400 ;*
3401 ;*****
3401 012450 000004 TST31: SCOPE
3402
3403 MMB1:
3404 012452 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
3405 012454 104414 MOV #MMBTP1,R0 ;SET UP THE DATA BUFFER.
3406 012454 012700 012552 MOV #MMBBF0,R1
3407 012460 012701 012602 MOV #10,R2
3408 012464 012702 000010 1$: MOV (R0)+,(R1)+
3409 012472 077202 SOB R2,1$
3410 012474 012700 000200 MOV #200,R0 ;SET FD.
```

```

3411 012500 170100          LDFPS  R0
3412 012502 012737 012516 001236  MOV   #MMB2,@#STMP2
3413 012510 012737 012622 000004  MOV   #MMB10,@#ERRVECT ;SET UP VECTOR 4 IN CASE OF AN ERROR.
3414
3415 012516 170677 000070  MMB2:  ABSD  @MMBBF1          ;TEST INSTRUCTION.
3416
3417 012522 170205          STFPS  R5          ;GET FPS.
3418 012524 012701 012602  MOV   #MMBBF0,R1      ;CHECK RESULT.
3419 012530 012702 000004  MOV   #4,R2
3420 012534 005721          1$:   TST   (R1)+
3421 012536 001047          BNE   MMB15          ;BRANCH IF INCORRECT.
3422 012540 077203          SOB   R2,1$
3423 012542 022705 000204  CMP   #204,R5        ;IS THE FPS CORRECT?
3424 012546 001056          BNE   MMB25          ;BRANCH IF INCORRECT.
3425 012550 000463          BR    MMBDONE
3426
3427          ;THESE ARE TEST DATA TABLES AND DATA BUFFER.
3428 012552 000137  MMBTP1: 137
3429 012554 045607          045607
3430 012556 101230          101230
3431 012560 045607 012602 177777  45607,MMBBF0,-1,-1,-1
3432 012566 177777 177777
3433 012572 000000  MMBTP2: 0
3434 012574 000000          0
3435 012576 000000          0
3436 012600 000000          0
3437 012602 177777  MMBBF0: -1
3438 012604 177777          -1
3439 012606 177777          -1
3440 012610 177777          -1
3441 012612 177777  MMBBF1: -1
3442 012614 177777          -1
3443 012616 177777          -1
3444 012620 177777          -1
3445
3446          ;IF A TRAP TO 4 OCCURS COME HERE.
3447 012622 011602  MMB10: MOV   (SP),R2          ;SEE IF THE TRAP OCCURRED ON THE TEST INSTRUCTION.
3448 012624 020227 012520  CMP   R2,#MMB2+2
3449 012630 001405          BEQ   1$          ;BRANCH IF YES.
3450 012632 020227 012522  CMP   R2,#MMB2+4
3451 012636 001402          BEQ   1$          ;BRANCH IF YES.
3452 012640 000137 036172  JMP   @#CPSPUR      ;OTHERWISE GO REPORT SPURIOUS TRAP TO 4.
3453          ;REPORT AN FDST FLOW FAILURE RESULTED IN A TRAP TO 4.
3454 012644 022626 1$:   CMP   (SP)+,(SP)+
3455 012646 010237 001236  MOV   R2,@#STMP2
3456 012652 104002 2$:   ERROR 2          ;ODD ADRES
3457 012654 000421          BR    MMBDONE      ;BUT FDSTX IN ST 771
3458
3459          ;REPORT RESULT INCORRECT:
3460 012656 012737 012572 001240  MMB15: MOV   #MMBTP2,@#STMP3
3461 012664 012737 012552 001242  MOV   #MMBTP1,@#STMP4
3462 012672 012737 012602 001244  MOV   #MMBBF0,@#STMP5
3463 012700 104002 1$:   ERROR 2          ;BAD DATA X11*0 ST 3127
3464 012702 000406          BR    MMBDONE
3465          ;REPORT FPS INCORRECT:
3466 012704 010537 001240  MMB25: MOV   R5,@#STMP3

```

```
3467 012710 012737 000204 001244      MOV    #204, @#STMP5
3468 012716 104002      1$:    ERROR 2          ;FPSX
3469
3470 012720      MMBDONE.
3471 012720 104413      RSETUP          ;GO INITIALIZE THE FPS AND STACK; AND
3472                                     ;SEE IF THE USER HAS EXPRESSED
3473                                     ;THE DESIRE TO CHANGE THE SOFTWARE
3474                                     ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
3475                                     ;THE USER TYPED CONTROL G?).
3476
3477 :*****
3478 :*TEST 32      SPECIAL DEST, MODE 0, TEST
3479 :*
3480 :*THIS IS A TEST OF THE NEGF ABSF AND TSTF DESTINATION FLOWS
3481 :*MODE 0 USING THE NEGD INSTR.
3482 :*
3483 :*****
3483 012722 000004      TST32: SCOPE
3484
3485 012724      NNB1:
3486 012724 104414      LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
3487 012726 012700 000200      MOV    #200,R0      ;SET FD.
3488 012732 170100      LDFPS  R0
3489 012734 012700 013022      MOV    #NNBTP1,R0   ;SET UP ACO.
3490 012740 172410      LDD    (R0),ACO
3491 012742 012737 012750 001236      MOV    #NNB2, @#STMP2
3492
3493 012750 170700      NNB2:  NEGD    ACO          ;TEST INSTRUCTION.
3494
3495 012752 170205      STFPS  R5          ;GET FPS.
3496 012754 012700 000200      MOV    #200,R0      ;SET FD.
3497 012760 170100      LDFPS  R0
3498 012762 012700 013042      MOV    #NNBBF0,R0   ;GET THE RESULT.
3499 012766 174010      STD    ACO,(R0)
3500 012770 012700 013042      MOV    #NNBBF0,R0   ;IS THE RESULT CORRECT?
3501 012774 012701 013032      MOV    #NNBTP2,R1
3502 013000 012702 000004      MOV    #4,R2
3503 013004 022021      1$:    CMP    (R0)+,(R1)+
3504 013006 001021      BNE   NNB10        ;BRANCH IF INCORRECT.
3505 013010 077203      SOB   R2,1$
3506 013012 022705 000210      CMP    #210,R5      ;IS THE FPS CORRECT?
3507 013016 001033      BNE   NNB15        ;BRANCH IF INCORRECT.
3508 013020 000440      BR    NNBDONE
3509
3510      ;THESE ARE DATA TABLES AND A DATA BUFFER.
3511 013022 013572      NNBTP1: 013572
3512 013024 046013      46013
3513 013026 057246      7246
3514 013030 013570      13570
3515 013032 113572      NNBTP2: 113572
3516 013034 046013      46013
3517 013036 057246      57246
3518 013040 013570      013570
3519 013042 000000      NNBBF0: 0
3520 013044 000000      0
3521 013046 000000      0
3522 013050 000000      0
```

```

3523
3524
3525 013052 012737 013042 001240 ;REPORT RESULT INCORRECT:
NNB10: MOV #NNBBF0,@#STMP3
3526 013060 012737 013032 001242 MOV #NNBTP2,@#STMP4
3527 013066 023737 013022 013042 CMP @NNBTP1,@NNBBF0
3528 013074 001002 BNE NNB11
3529 013076 104001 1$: ERROR 1 ;E10*200X ST 336
3530 013100 000410 BR NNB DONE
3531
3532 ;REPORT RESULT INCORRECT:
3533 013102 NNB11:
3534 013102 104001 1$: ERROR 1 ;BAD DATA NEGF
3535 013104 000406 BR NNB DONE
3536
3537 ;REPORT FPS INCORRECT:
3538 013106 010537 001242 NNB15: MOV R5,@#STMP4
3539 013112 012737 000210 001240 MOV #210,@#STMP3
3540 013120 104001 1$: ERROR 1 ;FPSX
3541
3542 013122 NNB DONE:
3543 013122 104413 RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
3544 ;SEE IF THE USER HAS EXPRESSED
3545 ;THE DESIRE TO CHANGE THE SOFTWARE
3546 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
3547 ;THE USER TYPED CONTROL G?).
3548
3549 :*****
3550 :*TEST 33 SPECIAL DEST, MODE 1, TEST
3551 :*
3552 :*THIS IS A TEST OF THE NEGF ABSF AND TSTF DESTINATION FLOWS
3553 :*MODE 1 USING THE NEGD INSTR.
3554 :*
3555 :*****
3556 TST33: SCOPE
3557 013124 000004
3558 013126 OOB1:
3559 013126 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
3560 013130 012701 013240 MOV #OOBTP1,R1 ;SET UP THE DATA BUFFER.
3561 013134 012700 013250 MOV #OOBTP2,R0
3562 013140 012702 000004 MOV #4,R2
3563 013144 012021 1$: MOV (R0)+,(R1)+
3564 013146 077202 SOB R2,1$
3565 013150 012700 013240 MOV #OOBTP1,R0
3566 013154 042710 100000 BIC #100000,(R0) ;MAKE OPERAND POSITIVE.
3567 013160 012737 013174 001236 MOV #OOB2,@#STMP2
3568 013166 012701 000200 MOV #200,R1 ;SET FD.
3569 013172 170101 LDFPS R1
3570 013174 170710 OOB2: NEGD (R0) ;TEST INSTRUCTION.
3571 013176 170205 STFPS R5 ;GET FPS.
3572 013200 012701 013240 MOV #OOBTP1,R1 ;IS THE RESULT CORRECT.
3573 013204 012702 013250 MOV #OOBTP2,R2
3574 013210 012703 000004 MOV #4,R3
3575 013214 022122 1$: CMP (R1)+,(R2)+
3576 013216 001020 BNE OOB10 ;BRANCH IF INCORRECT.
3577 013220 077303 SOB R3,1$
3578 013222 022700 013240 CMP #OOBTP1,R0 ;IS R0 CORRECT.

```

```

3579 013226 001024          BNE      00B15          ;BRANCH IF INCORRECT.
3580 013230 022705 000210   CMP      #210,R5       ;IS THE FPS CORRECT?
3581 013234 001030          BNE      00B20          ;BRANCH IF INCORRECT.
3582 013236 000435          BR       00BDONE
3583
3584          ;THESE ARE DATA TABLES AND A DATA BUFFER.
3585 013240 023245   00BTP1: 023245
3586 013242 026720          26720
3587 013244 122324          122324
3588 013246 052672          52672
3589 013250 123245   00BTP2: 123245
3590 013252 026720          26720
3591 013254 122324          122324
3592 013256 052672          52672
3593
3594          ;REPORT RESULT INCORRECT:
3595 013260 012737 013240 001240 00B10: MOV      #00BTP1,@#STMP3
3596 013266 012737 013250 001242 MOV      #00BTP2,@#STMP4
3597 013274 104002          1$:      ERROR      2          ;BAD DATA
3598 013276 000415          BR       00BDONE
3599
3600          ;REPORT R0 INCORRECT:
3601 013300 012737 013240 001240 00B15: MOV      #00BTP1,@#STMP3
3602 013306 010037 001242          MOV      R0,@#STMP4
3603 013312 104002          1$:      ERROR      2          ;SPEC DESTX
3604 013314 000406          BR       00BDONE          ;ROX
3605
3606          ;REPORT FPS INCORRECT:
3607 013316 012737 000210 001240 00B20: MOV      #210,@#STMP3
3608 013324 010537 001242          MOV      R5,@#STMP4
3609 013330 104002          1$:      ERROR      2
3610
3611 013332          00BDONE:
3612 013332 104413          RSETUP          ;GO INITIALIZE THE FPS AND STACK; AND
3613          ;SEE IF THE USER HAS EXPRESSED
3614          ;THE DESIRE TO CHANGE THE SOFTWARE
3615          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
3616          ;THE USER TYPED CONTROL G?).
3617
3618          ;*****
3619          ;*TEST 34          SPECIAL DEST, MODE 2, TEST
3620          ;*
3621          ;*THIS IS A TEST OF THE NEGF ABSF AND TSTF DESTINATION FLOWS
3622          ;*MODE 2 USING THE NEGD INSTR.
3623          ;*
3624          ;*****
3624 013334 000004          TST34: SCOPE
3625 013336          PPB1:
3626 013336 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
3627
3628 013340 012701 013450          MOV      #PPBTP1,R1          ;SET UP THE DATA BUFFER.
3629 013344 012700 013460          MOV      #PPBTP2,R0
3630 013350 012702 000004          MOV      #4,R2
3631 013354 012021          1$:      MOV      (R0)+,(R1)+
3632 013356 077202          SOB      R2,1$
3633 013360 012700 013450          MOV      #PPBTP1,R0
3634 013364 042710 100000          BIC      #100000,(R0)      ;MAKE OPERAND POSITIVE.

```

```
3635 013370 012737 013404 001236      MOV      #PPB2,@#STMP2
3636 013376 012701 000200      MOV      #200,R1          ;SET FD.
3637 013402 170101      LDFPS   R1
3638
3639 013404 170720      PPB2:   NEGD   (R0)+      ;TEST INSTRUCTION.
3640
3641 013406 170205      STFPS   R5              ;GET FPS.
3642 013410 012701 013450      MOV      #PPBTP1,R1      ;IS THE RESULT CORRECT.
3643 013414 012702 013460      MOV      #PPBTP2,R2
3644 013420 012703 000004      MOV      #4,R3
3645 013424 022122      1$:    CMP     (R1)+,(R2)+
3646 013426 001020      BNE     PPB10          ;BRANCH IF INCORRECT.
3647 013430 077303      SOB     R3,1$
3648 013432 022700 013460      CMP     #PPBTP1+10,R0   ;IS R0 CORRECT.
3649 013436 001024      BNE     PPB15          ;BRANCH IF INCORRECT.
3650 013440 022705 000210      CMP     #210,R5        ;IS THE FPS CORRECT?
3651 013444 001030      BNE     PPB20          ;BRANCH IF INCORRECT.
3652 013446 000435      BR      PPBDONE
```

```
3653
3654      ;THESE ARE DATA TABLES AND A DATA BUFFER.
3655 013450 023245      PPBTP1: 023245
3656 013452 026720      26720
3657 013454 122324      122324
3658 013456 052672      52672
3659 013460 123245      PPBTP2: 123245
3660 013462 026720      26720
3661 013464 122324      122324
3662 013466 052672      52672
```

```
3663
3664      ;REPORT RESULT INCORRECT:
3665 013470 012737 013450 001240      PPB10:  MOV     #PPBTP1,@#STMP3
3666 013476 012737 013460 001242      MOV     #PPBTP2,@#STMP4
3667 013504 104001      1$:    ERROR   1          ;BAD DATA
3668 013506 000415      BR      PPBDONE
```

```
3669
3670      ;REPORT R0 INCORRECT:
3671 013510 012737 013460 001240      PPB15:  MOV     #PPBTP1+10,@#STMP3
3672 013516 010037 001242      MOV     R0,@#STMP4
3673 013522 104001      1$:    ERROR   1          ;SPEC DESTX ROX
3674 013524 000406      BR      PPBDONE
```

```
3675
3676      ;REPORT FPS INCORRECT:
3677 013526 012737 000210 001240      PPB20:  MOV     #210,@#STMP3
3678 013534 010537 001242      MOV     R5,@#STMP4
3679 013540 104001      1$:    ERROR   1
```

```
3680
3681 013542      PPBDONE:
3682 013542 104413      RSETUP      ;GO INITIALIZE THE FPS AND STACK; AND
3683      ;SEE IF THE USER HAS EXPRESSED
3684      ;THE DESIRE TO CHANGE THE SOFTWARE
3685      ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
3686      ;THE USER TYPED CONTROL G?).
```

```
3687      ;*****
3688      ;*TEST 35      SPECIAL DEST, MODE 4, TEST
3689      ;*
3690      ;*THIS IS A TEST OF THE NEGF ABSF AND TSTF DESTINATION FLOWS
```

```

3691 ;*MODE 4 USING THE NEGD INSTR.
3692 ;*
3693 ;*****
3694 013544 000004 TS135: SCOPE
3695 013546 QQB1:
3696 013546 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
3697 013550 012701 013662 MOV #QQBTP1,R1 ;SET UP THE DATA BUFFER.
3698 013554 012700 013702 MOV #QQBTP2,R0
3699 013560 012702 000004 MOV #4,R2
3700 013564 012021 1$: MOV (R0)+,(R1)+
3701 013566 077202 SOB R2,1$
3702 013570 012700 013672 MOV #QQBTP1+10,R0
3703 013574 042760 100000 177770 BIC #100000,-10(R0) ;MAKE OPERAND POSITIVE.
3704 013602 012737 013616 001236 MOV #QQB2,@#STMP2
3705 013610 012701 000200 MOV #200,R1 ;SET FD.
3706 013614 170101 LDFPS R1
3707
3708 013616 170740 QQB2: NEGD -(R0) ;TEST INSTRUCTION.
3709
3710 013620 170205 STFPS R5 ;GET FPS.
3711 013622 012701 013662 MOV #QQBTP1,R1 ;IS THE RESULT CORRECT.
3712 013626 012702 013702 MOV #QQBTP2,R2
3713 013632 012703 000004 MOV #4,R3
3714 013636 022122 1$: CMP (R1)+,(R2)+
3715 013640 001024 BNE QQB10 ;BRANCH IF INCORRECT.
3716 013642 077303 SOB R3,1$
3717 013644 022700 013662 CMP #QQBTP1,R0 ;IS R0 CORRECT.
3718 013650 001030 BNE QQB15 ;BRANCH IF INCORRECT.
3719 013652 022705 000210 CMP #210,R5 ;IS THE FPS CORRECT?
3720 013656 001034 BNE QQB20 ;BRANCH IF INCORRECT.
3721 013660 000441 BR QQBDONE
3722
3723 ;THESE ARE DATA TABLES AND A DATA BUFFER.
3724 013662 023245 QQBTP1: 023245
3725 013664 026720 26720
3726 013666 122324 122324
3727 013670 052672 52672
3728 013672 177777 177777 177777 .WORD -1,-1,-1,-1
3729 013700 177777
3730 013702 123245 QQBTP2: 123245
3731 013704 026720 26720
3732 013706 122324 122324
3733 013710 052672 52672
3734
3735 ;REPORT RESULT INCORRECT:
3736 013712 012737 013662 001240 QQB10: MOV #QQBTP1,@#STMP3
3737 013720 012737 013702 001242 MOV #QQBTP2,@#STMP4
3738 013726 104001 1$: ERROR 1 ;BAD DATA
3739 013730 000415 BR QQBDONE
3740
3741 ;REPORT R0 INCORRECT:
3742 013732 012737 013662 001240 QQB15: MOV #QQBTP1,@#STMP3
3743 013740 010037 001242 MOV R0,@#STMP4
3744 013744 104001 1$: ERROR 1 ;SPEC DESTX ROX
3745 013746 000406 BR QQBDONE
3746

```

```
3747
3748
3749 013750 012737 000210 001240 :REPORT FPS INCORRECT:
3750 013756 010537 001242 QQB20: MOV #210,@#TMP3
3751 013762 104001 1$: MOV R5,@#TMP4
3752
3753 013764 QQBDONE:
3754 013764 104413 RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
3755 ;SEE IF THE USER HAS EXPRESSED
3756 ;THE DESIRE TO CHANGE THE SOFTWARE
3757 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
3758 ;THE USER TYPED CONTROL G?).
3759
3760
3761 :*****
3762 :*TEST 36 SPECIAL DEST, MODE 3, TEST
3763 :*
3764 :*THIS IS A TEST OF THE NEGF ABSF AND TSTF DESTINATION FLOWS
3765 :*MODE 3 USING THE NEGD INSTR.
3766 :*
3767 013766 000004 :*****
3768 TST36: SCOPE
3769
3770 013770 RRB1:
3771 013772 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
3772 013776 012701 014110 MOV #RRBTP1,R1 ;SET UP THE DATA BUFFER.
3773 014002 012700 014120 MOV #RRBTP2,R0
3774 014006 012021 MOV #4,R2
3775 014010 077202 1$: MOV (R0)+,(R1)+
3776 014012 012700 014130 SOB R2,1$
3777 014016 012710 014110 MOV #RRBTP3,R0
3778 014022 042737 100000 014110 BIC #100000,@#RRBTP1 ;MAKE THE OPERAND POSITIVE.
3779 014030 012737 014044 001236 MOV #RRB2,@#TMP2
3780 014036 012701 000200 MOV #200,R1 ;SET FD.
3781 014042 170101 LDFPS R1
3782
3783 014044 170730 RRB2: NEGD @(R0)+ ;TEST INSTRUCTION.
3784
3785 014046 170205 STFPS R5 ;GET FPS.
3786 014050 012701 014110 MOV #RRBTP1,R1 ;IS THE RESULT CORRECT.
3787 014054 012702 014120 MOV #RRBTP2,R2
3788 014060 012703 000004 MOV #4,R3
3789 014064 022122 1$: CMP (R1)+,(R2)+
3790 014066 001021 BNE RRB10 ;BRANCH IF INCORRECT.
3791 014070 077303 SOB R3,1$
3792 014072 022700 014132 CMP #RRBTP3+2,R0 ;IS R0 CORRECT.
3793 014076 001025 BNE RRB15 ;BRANCH IF INCORRECT.
3794 014100 022705 000210 CMP #210,R5 ;IS THE FPS CORRECT?
3795 014104 001031 BNE RRB20 ;BRANCH IF INCORRECT.
3796 014106 000436 BR RRBDONE
3797
3798
3799 014110 023245 ;THESE ARE DATA TABLES AND A DATA BUFFER.
3800 014112 026720 RRBTP1: 023245
3801 014114 122324 26720
3802 014116 052672 122324
52672
```



```
3803 014120 123245
3804 014122 026720
3805 014124 123324
3806 014126 052672
3807 014130 014110
3808
3809
3810 014132 012737 014110 001240 ;REPORT RESULT INCORRECT:
3811 014140 012737 014120 001242 RRB10: MOV #RRBTP1,@#STMP3
3812 014146 104001 1$: ERROR 1 ;BAD DATA
3813 014150 000415 BR RRBTP1
3814
3815
3816 014152 012737 014132 001240 ;REPORT R0 INCORRECT:
3817 014160 010037 001242 RRB15: MOV #RRBTP3+2,@#STMP3
3818 014164 104001 1$: ERROR 1 ;SPEC DESTX R0X
3819 014166 000406 BR RRBTP1
3820
3821
3822 014170 012737 000210 001240 ;REPORT FPS INCORRECT:
3823 014176 010537 001242 RRB20: MOV #210,@#STMP3
3824 014202 104001 1$: ERROR 1
3825
3826 014204
3827 014204 104413 RRBTP1 RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
3828 ;SEE IF THE USER HAS EXPRESSED
3829 ;THE DESIRE TO CHANGE THE SOFTWARE
3830 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
3831 ;THE USER TYPED CONTROL G?).
3832
3833
3834 ;*****
3835 ;*TEST 37 SPECIAL DEST, MODE 5, TEST
3836 ;*
3837 ;*THIS IS A TEST OF THE NEGF ABSF AND TSTF DESTINATION FLOWS
3838 ;*MODE 5 USING THE NEGD INSTR.
3839 ;*
3840 ;*****
3841 014206 000004 TST37: SCOPE
3842 014210 SSB1:
3843 014210 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
3844 014212 012701 014332 MOV #SSBTP1,R1 ;SET UP THE DATA BUFFER.
3845 014216 012700 014342 MOV #SSBTP2,R0
3846 014222 012702 000004 MOV #4,R2
3847 014226 012021 1$: MOV (R0)+,(R1)+
3848 014232 012700 014354 SOB R2,1$
3849 014236 012760 014332 177776 MOV #SSBTP3+2,R0
3850 014244 042737 100000 014332 BIC #100000,@#SSBTP1 ;MAKE THE OPERAND POSITIVE.
3851 014252 012737 014266 001236 MOV #SSB2,@#STMP2
3852 014260 012701 000200 MOV #200,R1 ;SET FD.
3853 014264 170101 LDFPS R1
3854
3855 014266 170750 SSB2: NEGD @-(R0) ;TEST INSTRUCTION.
3856
3857 014270 170205 STFPS R5 ;GET FPS.
3858 014272 012701 014332 MOV #SSBTP1,R1 ;IS THE RESULT CORRECT.
```

```

3859 014276 012702 014342      MOV    #SSBTP2,R2
3860 014302 012703 000004      MOV    #4,R3
3861 014306 022122      1$:  CMP    (R1)+,(R2)+
3862 014310 001021      BNE    SSB10          ;BRANCH IF INCORRECT.
3863 014312 077303      SOB    R3,1$
3864 014314 022700 014352      CMP    #SSBTP3,R0    ;IS R0 CORRECT.
3865 014320 001025      BNE    SSB15          ;BRANCH IF INCORRECT.
3866 014322 022705 000210      CMP    #210,R5       ;IS THE FPS CORRECT?
3867 014326 001031      BNE    SSB20          ;BRANCH IF INCORRECT.
3868 014330 000436      BR     SSBDONE
3869
3870      ;THESE ARE DATA TABLES AND A DATA BUFFER.
3871 014332 023245      SSBTP1: 023245
3872 014334 026720      26720
3873 014336 122324      122324
3874 014340 052672      52672
3875 014342 123245      SSBTP2: 123245
3876 014344 026270      26270
3877 014346 122324      122324
3878 014350 052672      52672
3879 014352 014332      SSBTP3: SSBTP1
3880
3881      ;REPORT RESULT INCORRECT:
3882 014354 012737 014332 001240      SSB10: MOV    #SSBTP1,@#$TMP3
3883 014362 012737 014342 001242      MOV    #SSBTP2,@#$TMP4
3884 014370 104001      1$:  ERROR 1          ;BAD DATA
3885 014372 000415      BR     SSBDONE
3886
3887      ;REPORT R0 INCORRECT:
3888 014374 012737 014352 001240      SSB15: MOV    #SSBTP3,@#$TMP3
3889 014402 010037 001242      MOV    R0,@#$TMP4
3890 014406 104001      1$:  ERROR 1          ;SPEC DESTX ROX
3891 014410 000406      BR     SSBDONE
3892
3893      ;REPORT FPS INCORRECT:
3894 014412 012737 000210 001240      SSB20: MOV    #210,@#$TMP3
3895 014420 010537 001242      MOV    R5,@#$TMP4
3896 014424 104001      1$:  ERROR 1
3897
3898      SSBDONE:
3899 014426 104413      RSETUP          ;GO INITIALIZE THE FPS AND STACK; AND
3900      ;SEE IF THE USER HAS EXPRESSED
3901      ;THE DESIRE TO CHANGE THE SOFTWARE
3902      ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
3903      ;THE USER TYPED CONTROL G?).
3904      ;*****
3905      ;*TEST 40      SPECIAL DEST, FLOATING MODE 2, TEST
3906      ;*
3907      ;*THIS IS A TEST OF THE NEGF ABSF AND TSTF DESTINATION FLOWS
3908      ;*MODE 2 USING THE NEGF INSTR.
3909      ;*
3910      ;*****
3911 014430 000004      TST40: SCOPE
3912 014432      TTBT1:
3913 014432 104414      LPERR
3914 014434 012701 014544      MOV    #TTBT1,R1    ;SET UP THE LOOP ON ERROR ADDRESS.
                        ;SET UP THE DATA BUFFER.

```

```

3915 014440 012700 014554      MOV      #TTBTP2,R0
3916 014444 012702 000004      MOV      #4,R2
3917 014450 012021      1$:     MOV      (R0)+,(R1)+
3918 014452 077202      SOB      R2,1$
3919 014454 012700 014544      MOV      #TTBTP1,R0
3920 014460 042710 100000      BIC      #100000,(R0)      ;MAKE OPERAND POSITIVE.
3921 014464 012737 014500 001236      MOV      #TTB2,@#$TMP2
3922 014472 012701 000000      MOV      #000,R1      ;SET FD.
3923 014476 170101      LDFPS   R1
3924
3925 014500 170720      TTB2:   NEGf      (R0)+      ;TEST INSTRUCTION.
3926
3927 014502 170205      STFPS   R5      ;GET FPS.
3928 014504 012701 014544      MOV      #TTBTP1,R1      ;IS THE RESULT CORRECT.
3929 014510 012702 014554      MOV      #TTBTP2,R2
3930 014514 012703 000004      MOV      #4,R3
3931 014520 022122      1$:     CMP      (R1)+,(R2)+
3932 014522 001020      BNE     TTB10      ;BRANCH IF INCORRECT.
3933 014524 077303      SOB      R3,1$
3934 014526 022700 014550      CMP      #TTBTP1+4,R0      ;IS R0 CORRECT.
3935 014532 001024      BNE     TTB15      ;BRANCH IF INCORRECT.
3936 014534 022705 000010      CMP      #010,R5      ;IS THE FPS CORRECT?
3937 014540 001030      BNE     TTB20      ;BRANCH IF INCORRECT.
3938 014542 000435      BR      TTBDONE
3939
3940      ;THESE ARE DATA TABLES AND A DATA BUFFER.
3941 014544 023245      TTBTP1: 023245
3942 014546 026720      26720
3943 014550 122324      122324
3944 014552 052672      52672
3945 014554 123245      TTBTP2: 123245
3946 014556 026720      26720
3947 014560 122324      122324
3948 014562 052672      52672
3949
3950      ;REPORT RESULT INCORRECT:
3951 014564 012737 014544 001240      TTB10:  MOV      #TTBTP1,@#$TMP3
3952 014572 012737 014554 001242      MOV      #TTBTP2,@#$TMP4
3953 014600 104001      1$:     ERROR   1      ;BAD DATA
3954 014602 000415      BR      TTBDONE
3955
3956      ;REPORT R0 INCORRECT:
3957 014604 012737 014550 001240      TTB15:  MOV      #TTBTP1+4,@#$TMP3
3958 014612 010037 001242      MOV      R0,@#$TMP4
3959 014616 104001      1$:     ERROR   1      ;SPEC DESTX ROX
3960 014620 000406      BR      TTBDONE
3961
3962      ;REPORT FPS INCORRECT:
3963 014622 012737 000010 001240      TTB20:  MOV      #010,@#$TMP3
3964 014630 010537 001242      MOV      R5,@#$TMP4
3965 014634 104001      1$:     ERROR   1
3966
3967 014636      TTBDONE:
3968 014636 104413      RSETUP      ;GO INITIALIZE THE FPS AND STACK; AND
3969      ;SEE IF THE USER HAS EXPRESSED
3970      ;THE DESIRE TO CHANGE THE SOFTWARE
    
```

```
3971 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
3972 ;THE USER TYPED CONTROL G?).
3973 ;*****
3974 ;*TEST 41 SPECIAL DEST, MODE2, GR7 (IMMEDIATE), TEST
3975 ;*
3976 ;*THIS IS A TEST OF THE NFGF ABSF AND TSTF DESTINATION FLOWS
3977 ;*MODE 2(IMMEDIATE) USING THE NEGD INSTR.
3978 ;*
3979 ;*****
3980 014640 000004 TST41: SCOPE
3981 014642 UUB1:
3982 014642 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
3983 014644 012700 014770 MOV #UUBTP2,R0
3984 014650 012701 014716 MOV #UUBTP1,R1 ;SET UP THE DATA BUFFER.
3985 014654 012702 000004 MOV #4,R2
3986 014660 012021 1$: MOV (R0)+,(R1)+
3987 014662 077202 SOB R2,1$
3988 014664 012700 014716 MOV #UUBTP1,R0
3989 014670 042737 100000 014716 BIC #100000,@#UUBTP1 ;MAKE THF OPERAND POSITIVE.
3990 014676 012737 014714 001236 MOV #UUB2,@#STMP2
3991 014704 012701 000200 MOV #200,R1 ;SET FD.
3992 014710 170101 LDFPS R1
3993 014712 005001 CLR R1
3994
3995 014714 170727 UUB2: NEGD (R7)+ ;TEST INSTRUCTION.
3996 014716 005201 005201 005201 UUBTP1: 5201,5201,5201,5201
3997 014724 005201
3998
3999 ;NOTE THAT AFTER EXECUTING THIS INSTRUCTION R1 SHOULD CONTAIN 3.
4000 014726 170205 STFPS R5 ;GET FPS.
4001 014730 012703 014716 MOV #UUBTP1,R3 ;IS THE RESULT CORRECT.
4002 014734 012702 014770 MOV #UUBTP2,R2
4003 014740 012704 000004 MOV #4,R4
4004 014744 022322 1$: CMP (R3)+,(R2)+
4005 014746 001014 BNE UUB10 ;BRANCH IF INCORRECT.
4006 014750 077403 SOB R4,1$
4007 014752 022701 000003 CMP #3,R1 ;WAS R1 INCREMENTED CORRECTLY.
4008 014756 001027 BNE UUB15 ;BRANCH IF INCORRECT.
4009 014760 022705 000210 CMP #210,R5 ;IS THE FPS CORRECT?
4010 014764 001015 BNE UUB20 ;BRANCH IF INCORRECT.
4011 014766 000436 BR UUBDONE
4012
4013 ;THESE ARE DATA TABLE.
4014 UUBTP2: 105201
4015 5201
4016 5201
4017 5201
4018
4019 ;REPORT RESULT INCORRECT:
4020 015000 012737 014716 001240 UUB10: MOV #UUBTP1,@#STMP3
4021 015006 012737 014770 001242 MOV #UUBTP2,@#STMP4
4022 015014 104001 1$: ERROR 1 ;BAD DATA
4023 015016 000422 BR UUBDONE
4024
4025 ;REPORT FPS INCORRECT:
4026 015020 012737 000210 001240 UUB20: MOV #210,@#STMP3
MOV R5,@#STMP4
```

4027 015032 104001
4028 015034 000413
4029
4030
4031 015036 162701 000003
4032 015042 006301
4033 015044 012702 014720
4034 015050 010237 001240
4035 015054 160102
4036 015056 010237 001242
4037 015062 104001
4038
4039 015064
4040 015064 104413
4041
4042
4043
4044
4045
4046
4047
4048
4049
4050
4051
4052 015066 000004
4053 015070
4054 015070 104414
4055 015072 012701 015214
4056 015076 012700 015224
4057 015102 012702 000004
4058 015106 012021
4059 015110 077202
4060 015112 012700 010013
4061 015116 042737 100000 015214
4062 015124 012737 015142 001236
4063 015132 012701 000200
4064 015136 170101
4065
4066 015140 005001
4067 015142 170760 005201
4068
4069 015146 170205
4070 015150 005701
4071 015152 001030
4072 015154 012701 015214
4073 015160 012702 015224
4074 015164 012703 000004
4075 015170 022122
4076 015172 001030
4077 015174 077303
4078 015176 022700 010013
4079 015202 001034
4080 015204 022705 000210
4081 015210 001040
4082 015212 000445

1\$: ERROR 1 ;FPS
BR UUBDONE
;REPORT PC INCORRECTLY INCREMENTED DURING EXECUTION.
UUB15: SUB #3,R1
ASL R1
MOV #UUBTP1+2,R2
MOV R2,@#STMP3
SUB R1,R2
MOV R2,@#STMP4
1\$: ERROR 1 ;PC BAD CONSTAND B GR7X
UUBDONE: 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 42 SPECIAL DEST, MODE 6, TEST
;*
;*THIS IS A TEST OF THE NEGF ABSF AND TSTF DESTINATION FLOWS
;*MODE 6 USING THE NEGD INSTR.
;*
;*****
TST42: SCOPE
XXB1: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
MOV #XXBTP1,R1 ;SET UP THE DATA BUFFER.
MOV #XXBTP2,R0
MOV #4,R2
1\$: MOV (R0)+,(R1)+
SOB R2,1\$
MOV #XXBTP1-5201,R0
BIC #100000,@#XXBTP1;MAKE OPERAND POSITIVE.
MOV #XXB2,@#STMP2
MOV #200,R1 ;SET FD.
LDFPS R1
XXB2: CLR R1
NEGD 5201(R0) ;TEST INSTRUCTION.
STFPS R5 ;GET FPS.
TST R1
BNE XXB25 ;WAS THE PC CORRECT AFTER EXECUTION?
MOV #XXBTP1,R1 ;IS THE RESULT CORRECT.
MOV #XXBTP2,R2
MOV #4,R3
1\$: CMP (R1)+,(R2)+
BNE XXB10 ;BRANCH IF INCORRECT.
SOB R3,1\$
CMP #XXBTP1-5201,R0 ;IS R0 CORRECT.
BNE XXB15 ;BRANCH IF INCORRECT.
CMP #210,R5 ;IS THE FPS CORRECT?
BNE XXB20 ;BRANCH IF INCORRECT.
BR XXBDONE

```
4083
4084 ;THESE ARE DATA TABLES AND A DATA BUFFER.
4085 015214 023245 XXBTP1: 023245
4086 015216 026720          26720
4087 015220 122324          122324
4088 015222 052672          52672
4089 015224 123245 XXBTP2: 123245
4090 015226 026720          26720
4091 015230 122324          122324
4092 015232 052672          52672
4093
4094
4095 ;REPORT PC INCORRECT AFTER EXECUTION.
4096 015234 012737 015144 001242 XXB25: MOV #XXB2+2,@#$TMP4
4097 015242 012737 015146 001240      MOV #XXB2+4,@#$TMP3
4098 015250 104001      1$: ERROR 1 ;PC NOT INCREMENTED BY 2.
4099 015252 000425      BR XXBDONE
4100
4101 ;REPORT RESULT INCORRECT:
4102 015254 012737 015214 001240 XXB10: MOV #XXBTP1,@#$TMP3
4103 015262 012737 015224 001242      MOV #XXBTP2,@#$TMP4
4104 015270 104001      1$: ERROR 1 ;BAD DATA
4105 015272 000415      BR XXBDONE
4106
4107 ;REPORT R0 INCORRECT:
4108 015274 012737 010013 001240 XXB15: MOV #XXBTP1-5201,@#$TMP3
4109 015302 010037 001242      MOV R0,@#$TMP4
4110 015306 104001      1$: ERROR 1 ;SPEC DESTX ROX
4111 015310 000406      BR XXBDONE
4112
4113
4114 ;REPORT FPS INCORRECT:
4115 015312 012737 000210 001240 XXB20: MOV #210,@#$TMP3
4116 015320 010537 001242      MOV R5,@#$TMP4
4117 015324 104001      1$: ERROR 1
4118
4119 015326 XXBDONE:
4120 015326 104413      RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
4121 ;SEE IF THE USER HAS EXPRESSED
4122 ;THE DESIRE TO CHANGE THE SOFTWARE
4123 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
4124 ;THE USER TYPED CONTROL G?).
4125
4126 ;*****
4127 ;*TEST 43 SPECIAL DEST, MODE 7, TEST
4128 ;*
4129 ;*THIS IS A TEST OF THE NEGF ABSF AND TSTF DESTINATION FLOWS
4130 ;*MODE 7 USING THE NEGD INSTR.
4131 ;*
4132 ;*****
4133 015330 000004 TST43: SCOPE
4134
4135 015332 YYB1:
4136 015332 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
4137 015334 012701 015464 MOV #YYBTP1,R1 ;SET UP THE DATA BUFFER.
4138 015340 012700 015474 MOV #YYBTP2,R0
```

```

4139 015344 012702 000004
4140 015350 012021
4141 015352 077202
4142 015354 012700 010303
4143 015360 012760 015464 005201
4144 015366 042737 100000 015464
4145 015374 012737 015412 001236
4146 015402 012701 000200
4147 015406 170101
4148
4149 015410 005001
4150 015412 170770 005201
4151
4152 015416 170205
4153 015420 005701
4154 015422 001031
4155 015424 012701 015464
4156 015430 012702 015474
4157 015434 012703 000004
4158 015440 022122
4159 015442 001031
4160 015444 077303
4161 015446 022700 010303
4162 015452 001035
4163 015454 022705 000210
4164 015460 001041
4165 015462 000446
4166
4167
4168 015464 023245
4169 015466 026720
4170 015470 122324
4171 015472 052672
4172 015474 123245
4173 015476 026720
4174 015500 123324
4175 015502 052672
4176 015504 015464
4177
4178
4179 015506 016737 177702 001242
4180 015514 016737 177676 001240
4181 015522 104002
4182 015524 000425
4183
4184
4185 015526 012737 015464 001240
4186 015534 012737 015474 001242
4187 015542 104002
4188 015544 000415
4189
4190
4191 015546 012737 010303 001240
4192 015554 010037 001242
4193 015560 104002
4194 015562 000406

      1$: MOV #4,R2
      MOV (R0)+,(R1)+
      SOB R2,1$
      MOV #YYBTP3-5201,R0
      MOV #YYBTP1,5201(R0)
      BIC #100000,@#YYBTP1 ;MAKE THE OPERAND POSITIVE.
      MOV #YYB2,@#STMP2
      MOV #200,R1 ;SET FD.
      LDFPS R1

YYB2: CLR R1
      NEG R1 ;TEST INSTRUCTION.
      STFPS R5 ;GET FPS.
      TST R1 ;WAS THE PC CORRECT AFTER EXECUTION?
      BNE YYB25
      MOV #YYBTP1,R1 ;IS THE RESULT CORRECT.
      MOV #YYBTP2,R2
      MOV #4,R3
      1$: CMP (R1)+,(R2)+
      BNE YYB10 ;BRANCH IF INCORRECT.
      SOB R3,1$
      CMP #YYBTP3-5201,R0 ;IS R0 CORRECT.
      BNE YYB15 ;BRANCH IF INCORRECT.
      CMP #210,R5 ;IS THE FPS CORRECT?
      BNE YYB20 ;BRANCH IF INCORRECT.
      BR YYBDONE

;THESE ARE DATA TABLES AND A DATA BUFFER.
YYBTP1: 023245
      26720
      122324
      52672
YYBTP2: 123245
      26720
      123324
      52672
YYBTP3: YYBTP1

;REPORT PC INCORRECT AFTER EXECUTION.
YYB25: MOV YYB2+2,@#STMP4
      MOV YYB2+4,@#STMP3
      1$: ERROR 2 ;PC NOT INCREMENTED BY 2.
      BR YYBDONE

;REPORT RESULT INCORRECT:
YYB10: MOV #YYBTP1,@#STMP3
      MOV #YYBTP2,@#STMP4
      1$: ERROR 2 ;BAD DATA
      BR YYBDONE

;REPORT R0 INCORRECT:
YYB15: MOV #YYBTP3-5201,@#STMP3
      MOV R0,@#STMP4
      1$: ERROR 2 ;SPEC DESTX ROX
      BR YYBDONE

```

```
4195
4196
4197 015564 012737 000210 001240 ;REPORT FPS INCORRECT:
4198 015572 010537 001242 YYB20: MOV #210,2#STMP3
4199 015576 104002 1$: ERROR 2
4200
4201 015600
4202 015600 104413 YBBDONE: RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
4203 ;SEE IF THE USER HAS EXPRESSED
4204 ;THE DESIRE TO CHANGE THE SOFTWARE
4205 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
4206 ;THE USER TYPED CONTROL G?).
4207
4208 ;*****
4209 ;*TEST 44 NEG, ABSD AND TSTD TEST
4210 ;*
4211 ;*THIS IS A TEST OF THE NEG, ABSD AND TSTD INSTRUCTIONS.
4212 ;*
4213 015602 000004 ;*****
4214 TST44: SCOPE
4215 015604 104414 ;TEST NEG, ABSD AND TSTD INSTRUCTIONS.
4216 015604 104414 ;TEST NEG, ABSD AND TSTD INSTRUCTIONS.
4217 015606 004767 000634 ;*****
4218 015612 000000 ;*****
4219 015614 016341 ;*****
4220 015616 055772 ;*****
4221 015620 021133 ;*****
4222 015622 055447 ;*****
4223 015624 116341 ;*****
4224 015626 055772 ;*****
4225 015630 021133 ;*****
4226 015632 055447 ;*****
4227 015634 016341 ;*****
4228 015636 055772 ;*****
4229 015640 021133 ;*****
4230 015642 055447 ;*****
4231 015644 000207 ;*****
4232 015646 000210 ;*****
4233 015650 000200 ;*****
4234 015652 177777 ;*****
4235 015654 104002 ;*****
4236 015656 000401 ;*****
4237 015660 104002 ;*****
4238 015662 ;*****
4239 ;*****
4240 015662 ;*****
4241 015662 104414 000556 ;*****
4242 015664 004767 ;*****
4243 015670 000000 ;*****
4244 015672 152525 ;*****
4245 015674 053545 ;*****
4246 015676 055565 ;*****
4247 015700 057505 ;*****
4248 015702 052525 ;*****
4249 015704 053545 ;*****
4250 015706 055565 ;*****
```



```

4251 015710 057505
4252 015712 152525
4253 015714 053545
4254 015716 055565
4255 015720 057505
4256 015722 000217
4257 015724 000200
4258 015726 000210
4259 015730 177777
4260 015732 104002
4261 015734 000401
4262 015736 104002
4263 015740
4264
4265 015740
4266 015740 104414
4267 015742 004767 000500
4268 015746 000001
4269 015750 060705
4270 015752 124735
4271 015754 060124
4272 015756 073560
4273 015760 060705
4274 015762 124735
4275 015764 060124
4276 015766 073560
4277 015770 160705
4278 015772 124735
4279 015774 060124
4280 015776 073560
4281 016000 000217
4282 016002 000200
4283 016004 000210
4284 016006 177777
4285 016010 104002
4286 016012 000401
4287 016014 104002
4288 016016
4289
4290 016016
4291 016016 104414
4292 016020 004767 000422
4293 016024 000001
4294 016026 154345
4295 016030 076567
4296 016032 032123
4297 016034 043234
4298 016036 054345
4299 016040 076567
4300 016042 032123
4301 016044 043234
4302 016046 154345
4303 016050 076567
4304 016052 032123
4305 016054 043234
4306 016056 000217

      57505
48:    152525      ;ERROR RES.
      53545
      55565
      57505
58:    217        ;FPS BEFORE EXECUTION.
      200        ;FPS AFTER EXECUTION.
      210        ;ERROR FPS.
      -1         ;FEC
68:    ERROR     2   ;E10<---E10*200X S336
      BR        78
      ERROR     2
78:
;TEST ABSD WITH POSITIVE OPERAND
WMB3:
      LPERR
      JSR      PC,NATSUB      ;SET UP THE LOOP ON ERROR ADDRESS.
18:    1
28:    60705
      124735
      60124
38:    73560
      60705      ;RESULT.
      124735
      60124
      73560
48:    160705
      124735
      60124
58:    73560
      217        ;FPS BEFORE EXECUTION.
      200        ;FPS AFTER EXECUTION.
      210        ;ERROR FPS.
      -1         ;EITHER BUT OP18
68:    ERROR     2   ;BUT ST 055 TO 336 INTO 335
      BR        78
      ERROR     2
78:
;TEST ABSD WITH NEG. OPERAND
WMB4:
      LPERR
      JSR      PC,NATSUB      ;SET UP THE LOOP ON ERROR ADDRESS.
18:    1
28:    154345
      76567
      32123
38:    43234
      54345      ;RESULT.
      76567
48:    154345
      76567
      32123
58:    43234
      217        ;FPS BEFORE EXECUTION.

```

```

4307 016060 000200          200          ;FPS AFTER EXECUTION.
4308 016062 177777          -1          ;ERROR FPS.
4309 016064 177777          -1
4310 016066 104002        6$:      ERROR    2          ;E10*E10*200X ST 452
4311 016070 000401          BR          7$
4312 016072 104002          ERROR    2
4313 016074
4314
4315 016074
4316 016074 104414          WWS5:      LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
4317 016076 004767 000344      JSR          PC,NATSUB
4318 016102 000002        1$:      2          ;FLAG-TSTD.
4319 016104 012321        2$:      12321         ;OPERAND.
4320 016106 045654          45654
4321 016110 070107          70107
4322 016112 034543          34543
4323 016114 012321        3$:      12321         ;RESULT.
4324 016116 045654          45654
4325 016120 070107          70107
4326 016122 034543          34543
4327 016124 112321        4$:      112321         ;ERROR RES.
4328 016126 045654          45654
4329 016130 070107          70107
4330 016132 034543          34543
4331 016134 000217        5$:      217          ;FPS BEFORE EXECUTION.
4332 016136 000200          200         ;FPS AFTER EXECUTION.
4333 016140 000210          210         ;ERROR FPS.
4334 016142 177777          -1
4335 016144 104002        6$:      ERROR    2          ;BUT (OP1B) X ST044 TO 336 INTO 334
4336 016146 000401          BR          7$
4337 016150 104002          ERROR    2          ;BUT ENBT ST 334 TO 453 INTO 053
4338 016152
4339
4340 016152
4341 016152 104414          WWS6:      LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
4342 016154 004767 000266      JSR          PC,NATSUB
4343 016160 000002        1$:      2          ;FLAG=TSTD.
4344 016162 123765        2$:      123765         ;OPERAND.
4345 016164 023407          23407
4346 016166 034510          34510
4347 016170 045621          45621
4348 016172 123765        3$:      123765         ;RESULT.
4349 016174 023407          23407
4350 016176 034510          34510
4351 016200 045621          45621
4352 016202 023765        4$:      23765         ;ERROR RES.
4353 016204 023407          23407
4354 016206 034510          34510
4355 016210 045621          45621
4356 016212 000207        5$:      207          ;FPS BEFORE EXECUTION.
4357 016214 000210          210         ;FPS AFTER EXECUTION.
4358 016216 000200          200         ;ERROR FPS.
4359 016220 177777          -1
4360 016222 104002        6$:      ERROR    2          ;BU* OPB1 ST 055 TO 335 INTO 334
4361 016224 000401          BR          7$
4362 016226 104002          ERROR    2          ;BUT ENBT ST 334 TO 053 INTO 453

```

```
4363 016230      7$:  
4364      ;TEST TSTD 0 OP  
4365 016230      WWB7:  
4366 016230 104414      LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.  
4367 016232 004767 000210      JSR      PC,NATSUB  
4368 016236 000002      1$:      2      ;FLAG=TSTD.  
4369 016240 000175      2$:      175      ;OPERAND.  
4370 016242 176737      176737  
4371 016244 071727      71727  
4372 016246 037574      37574  
4373 016250 000175      3$:      175      ;RESULT.  
4374 016252 176737      176737  
4375 016254 071727      71727  
4376 016256 037574      37574  
4377 016260 000000      4$:      0      ;ERROR RES.  
4378 016262 000000      0  
4379 016264 000000      0  
4380 016266 000000      0  
4381 016270 000200      5$:      200      ;FPS BEFORE EXECUTION.  
4382 016272 000204      204      ;FPS AFTER EXECUTION.  
4383 016274 000214      214      ;ERROR FPS.  
4384 016276 177777      -1  
4385 016300 104002      6$:      ERROR 2      ;BUT OP1B ST 255 TO 311 OR 312 INTO 310  
4386 016302 000401      BR 7$  
4387 016304 104002      ERROR 2      ;BUT ENBT ST 310 TO 402 INTO 002  
4388 016306      7$:  
4389      ;TEST TSTD -0 OP FIUV=0  
4390 016306      WWB8:  
4391 016306 104414      LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.  
4392 016310 004767 000132      JSR      PC,NATSUB  
4393 016314 000002      1$:      2      ;FLAG=TSTD.  
4394 016316 100123      2$:      100123      ;OPERAND.  
4395 016320 021012      21012  
4396 016322 034565      34565  
4397 016324 043210      43210  
4398 016326 100123      3$:      100123      ;RESULT.  
4399 016330 021012      21012  
4400 016332 034565      34565  
4401 016334 043210      43210  
4402 016336 000000      4$:      0      ;ERROR RES.  
4403 016340 000000      0  
4404 016342 000000      0  
4405 016344 000000      0  
4406 016346 040203      5$:      40203      ;FPS BEFORE EXECUTION.  
4407 016350 040214      040214      ;FPS AFTER EXECUTION.  
4408 016352 140214      140214      ;ERROR FPS.  
4409 016354 177777      -1  
4410 016356 104002      6$:      ERROR 2      ;+  
4411 016360 000401      BR 7$  
4412 016362 104002      ERROR 2      ;BUT FIUV ST 257 TO 355 INTO 255  
4413 016364      7$:  
4414      ;TEST TSTD -0 OP FIUV=1  
4415 016364      WWB9:  
4416 016364 104414      LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.  
4417 016366 004767 000054      JSR      PC,NATSUB  
4418 016372 000002      1$:      2      ;FLAG=TSTD.
```

```

4419 016374 100137      2$: 100137      ;OPERAND.
4420 016376 024613      24613
4421 016400 057024      57024
4422 016402 060137      60137
4423 016404 100137      3$: 100137      ;RESULT.
4424 016406 024613      24613
4425 016410 057024      57024
4426 016412 060137      60137
4427 016414 000000      4$: 0           ;ERROR RES.
4428 016416 000000      0
4429 016420 000000      0
4430 016422 000000      0
4431 016424 044200      5$: 44200       ;FPS BEFORE EXECUTION.
4432 016426 144214      144214       ;FPS AFTER EXECUTION.
4433 016430 044214      044214       ;ERROR FPS.
4434 016432 000014      14
4435 016434 104002      6$: ERROR 2       ;+
4436 016436 000401      BR 7$
4437 016440 104002      ERROR 2
4438 016442
4439 016442 000167 000414 7$:          ;BUT FIUV ST 257 TO 255 INTO 355
4440          JMP      WWBDONE
  
```

:THIS SUBROUTINE, NATSUB, IS USED TO SET UP THE OPERANDS, EXECUTE
 :THE EITHER A TSTD, AN ABSD OR A NEG, INSTRUCTION AND CHECK THE RESULTS. A CALL
 :TO IT IS MADE THUS:

```

:
:      JSR      PC, NATSUB
:      FLAG:   .WORD  X           ;INSTRUCTION TYPE FLAG.
:      ACARG:  .WORD  X,X,X,X     ;OPERAND
:      RES:    .WORD  X,X,X,X     ;EXPECTED RESULT
:      ERRES:  .WORD  X,X,X,X     ;ERROR RESULT
:      FPSB:   .WORD  X           ;FPS BEFORE EXECUTION
:      FPSA:   .WORD  X           ;FPS AFTER EXECUTION
:      FEC:    .WORD  X           ;EXPECTED FEC
:      ERFPS:  .WORD  X           ;ERROR FPS.
:      ERR1:   ERROR 2           ;DATA ERROR.
:              BR      CONT
:      ERR2:   ERROR 2           ;FPS ERROR.
:      CONT:   ;RETURN ADDRESS
  
```

:THE OPERAND IS SET UP IN NATBF1. THEN
 :THE EITHER THE TSTD, NEG, OR ABSD INSTRUCTION IS EXECUTED.
 :NATSUB USES THE FIRST OPERAND AS A FLAG TO DETERMINE WHICH INSTRUCTION
 :IS TO BE EXECUTED: 0 = NEG, 1 = ABSD, 2 = TSTD.
 :THE RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
 :COMPARED WITH FPSA. IF THIS TOO IS CORRECT NATSUB RETURNS CONTROL
 :TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD NATSUB
 :COMPARES IT TO ERROR FPS. IF THIS MATCHES THEN NATSUB WILL RETURN
 :TO THE ERROR CALL AT ERR2, OTHERWISE NATSUB ITSELF
 :REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE
 :INSTRUCTION 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 NATSUB
 :WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR1. OTHERWISE THE
 :RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND NATSUB WILL
 :REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.

```

4475
4476
4477 016446 012601          NATSUB: MOV      (SP)+,R1          ;GET A POINTER TO THE ARGUMENTS.
4478 016450 010102          MOV      R1,R2          ;COPY THE OPERAND.
4479 016452 062702 000002   ADD      #2,R2
4480 016456 012703 017050   MOV      #NATBF1,R3
4481 016462 012704 000004   MOV      #4,R4
4482 016466 012223          1$:  MOV      (R2)+,(R3)+
4483 016470 077402          SOB      R4,1$
4484 016472 016100 000032   MOV      32(R1),R0      ;LOAD THE FPS.
4485 016476 170100          LDFPS   R0
4486 016500 012700 017050   MOV      #NATBF1,R0      ;SET UP THE OPERAND ADDRESS.
4487 016504 011102          MOV      (R1),R2        ;GET THE FLAG TO DETERMINE WHICH
4488 016506 006302          ASL      R2              ;INSTRUCTION TO EXECUTE.
4489 016510 006302          ASL      R2              ;0 = NEG, 1 = ABSD, 2 = TSTD
4490 016512 012703 016526   MOV      #NATINS,R3
4491 016516 060203          ADD      R2,R3
4492 016520 010337 001236   MOV      R3,@#STMP2
4493 016524 000113          JMP      (R3)            ;GO EXECUTE THE INSTRUCTION.
4494 016526 170710          NATINS: NEG, (R0)
4495 016530 000403          BR      2$
4496 016532 170610          ABSD   (R0)
4497 016534 000401          BR      2$
4498 016536 170510          TSTD   (R0)
4499
4500 016540 170204          2$:  STFPS   R4              ;GET THE FPS.
4501 016542 170305          STST   R5              ;GET THE FEC.
4502 016544 010102          MOV      R1,R2
4503 016546 062702 000002   ADD      #2,R2
4504 016552 010237 001240   MOV      R2,@#STMP3
4505 016556 062702 000010   ADD      #10,R2
4506 016562 010237 001244   MOV      R2,@#STMP5
4507 016566 012737 017050 001242   MOV      #NATBF1,@#STMP4
4508 016574 010437 001250   MOV      R4,@#STMP7
4509 016600 016137 000034 001252   MOV      34(R1),@#STMP10
4510 016606 010100          MOV      R1,R0          ;WAS THE RESULT CORRECT?
4511 016610 062700 000012   ADD      #12,R0
4512 016614 012702 017050   MOV      #NATBF1,R2
4513 016620 012703 000004          3$:  MOV      #4,R3
4514 016624 022022          CMP      (R0)+,(R2)+
4515 016626 001014          BNE     10$             ;BRANCH IF INCORRECT.
4516 016630 077303          SOB     R3,3$
4517 016632 026104 000034   CMP     34(R1),R4      ;WAS THE FPS CORRECT?
4518 016636 001032          BNE     15$             ;BRANCH IF INCORRECT.
4519 016640 005761 000034   TST    34(R1)          ;IF THE EXPECTED FPS WAS NEGATIVE CHECK THE FEC.
4520 016644 100003          BPL     4$
4521 016646 026105 000040   CMP     40(R1),R5      ;WAS THE FEC CORRECT.
4522 016652 001037          BNE     20$             ;BRANCH IF INCORRECT.
4523 016654 000161 000050          4$:  JMP     50(R1)          ;RETURN.
4524
4525          ;THE RESULT WAS INCORRECT BUT WAS THIS FAILURE ANTICIPATED?
4526          ;SEE IF THE RESULT WAS ANTICIPATED:
4527 016660          10$:
4528 016660 011105          MOV     (R1),R5
4529 016662 006305          ASL    R5
4530 016664 006305          ASL    R5
  
```

```
4531 016666 062705 017000      ADD    #NATER1,R5
4532 016672 010100      MOV    R1,R0
4533 016674 062700 000022      ADD    #22,R0
4534 016700 012702 017050      MOV    #NATBF1,R2
4535 016704 012703 000004      MOV    #4,R3
4536 016710 022022      11$:  CMP    (R0)+,(R2)+
4537 016712 001003      BNE   12$          ;BRANCH IF NOT ANTICIPATED.
4538 016714 077303      SOB   R3,11$
4539
4540      ;THE ERROR WAS ANTICIPATED SO RETURN.
4541 016716 000161 000042      JMP   42(R1)
4542
4543      ;THE ERROR WAS NOT ANTICIPATED SO REPORT IT HERE.
4544 016722 000115      12$:  JMP   (R5)          ;GO TO THE PROPER ERROR CALL.
4545
4546      ;THE FPS WAS INCORRECT.
4547 016724 026105 000036      15$:  CMP    36(R1),R5      ;WAS THIS ERROR ANTICIPATED?
4548 016730 001002      BNE   16$          ;BRANCH IF NOT ANTICIPATED.
4549
4550      ;THE FPS ERROR WAS ANTICIPATED SO RETURN.
4551 016732 000161 000046      JMP   46(R1)
4552
4553      ;THE FPS FAILURE WAS NOT ANTICIPATED SO REPORT IT HERE.
4554 016736 011102      16$:  MOV    (R1),R2
4555 016740 006302      ASL   R2
4556 016742 006302      ASL   R2
4557 016744 062702 017016      ADD   #NATER2,R2
4558 016750 000112      JMP   (R2)          ;GO TO THE PROPER ERROR CALL.
4559
4560      ;REPORT THAT THE FEC WAS INCORRECT.
4561 016752 016137 000040 001256 20$:  MOV    40(R1),@#$TMP^2
4562 016760 010537 001254      MOV    R5,@#$TMP11
4563 016764 011102      MOV    (R1),R2
4564 016766 006302      ASL   R2
4565 016770 006302      ASL   R2
4566 016772 062702 017032      ADD   #NATER3,R2
4567 016776 000112      JMP   (R2)          ;GO TO THE PROPER ERROR CALL.
4568
4569      ;THESE ARE THE ERROR CALLS FOR EACH INDIVIDUAL INSTRUCTION AND CONDITION.
4570 017000 104002      NATER1: ERROR 2          ;NEGD BAD DATA
4571 017002 000403      BR    NATRET
4572 017004 104002      ERROR 2          ;ABSD BAD DATA
4573 017006 000401      BR    NATRET
4574 017010 104002      ERROR 2          ;TSTD BAD DATA
4575 017012 000161 000050      NATRET: JMP 50(R1)
4576
4577      ;FPS INCORRECT:
4578 017016 104002      NATER2: ERROR 2          ;NEGD FPSX
4579 017020 000774      BR    NATRET
4580 017022 104002      ERROR 2          ;ABSD FPSX
4581 017024 000772      BR    NATRET
4582 017026 104002      ERROR 2          ;TSTD FPSX
4583 017030 000770      BR    NATRET
4584
4585      ;FEC INCORRECT:
4586 017032 104002      NATER3: ERROR 2          ;NEGD FECX
```

```

4587 017034 000766 BR NATRET
4588 017036 104002 ERROR 2 ;ABSD FEEX
4589 017040 000764 BR NATRET
4590 017042 104002 ERROR 2 ;TSTD FEEX
4591 017044 000762 BR NATRET
4592
4593 017046 177777 .WORD -1
4594 017050 177777 177777 NATBF1: .WORD -1,-1,-1,-1,-1
4595 017056 177777 177777
4596
4597 017062 WWBDONE:
4598 017062 104413 RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
4599 ;SEE IF THE USER HAS EXPRESSED
4600 ;THE DESIRE TO CHANGE THE SOFTWARE
4601 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
4602 ;THE USER TYPED CONTROL G?).
4603
4604
4605
4606
4607 ;*****
4608 ;*TEST 45 SOURCE MODES, MODE 1 (FL=0), TEST
4609 ;*
4610 ;* THIS IS A TEST OF SOURCE MODE 1
4611 ;* USING THE LDFPS INSTR
4612 ;*
4613 017064 000004 ;*****
4614 TST45: SCOPE
4615
4616 017066 AAC1:
4617 017066 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
4618
4619 017070 012700 017146 MOV #AACTP1,R0 ;SET UP TEST DATA IN BUFFER.
4620 017074 012710 147517 MOV #147517,(R0)
4621 017100 012737 147517 001240 MOV #147517,@#STMP3 ;SAVE DATA IN CASE OF ERROR.
4622 017106 012737 017122 001236 MOV #AAC2,@#STMP2
4623 017114 012737 017206 000004 MOV #AAC20,@#ERRVECT ;SET UP FOP TRAPS TO 4.
4624 017122 170*10 AAC2: LDFPS (R0) ;TEST INSTRUCTION.
4625
4626 017124 170205 STFPS R5 ;GET FPS
4627
4628 017126 020027 017146 CMP R0,#AACTP1 ;IS R0 CORRECT?
4629 017132 001007 BNE AAC10 ;BR IF NOT.
4630 017134 022705 147517 CMP #147517,R5 ;IS FPS CORRECT?
4631 017140 001013 BNE AAC11 ;BR IF NOT.
4632 017142 000437 BR AACDONE
4633
4634 ;TEST BUFFER AND DATA:
4635 017144 177777 -1
4636 017146 147517 AACTP1: 147517
4637 017150 177777 -1
4638
4639 ;REPORT R0 INCORRECT.
4640 017152 012737 017146 001240 AAC10: MOV #AACTP1,@#STMP3
4641 017160 010037 001242 MOV R0,@#STMP4
4642 017164 104001 1$: ERROR 1 ;R0 BAD BUT FSRC FAILED

```

```
4643 017166 000425 BR AACDONE
4644
4645 :REPORT FPS INCORRECT.
4646 017170 012737 147517 001240 AAC11: MOV #147517,@#STMP3 ;REPORT FPS INCORRECT.
4647 017176 010537 001242 AAC11: MOV R5,@#STMP4
4648 017202 104001 1$: ERROR 1
4649 017204 000416 BR AACDONE
4650
4651 :TRAP HERE THROUGH VECTOR FOUR. SEE IF THE TRAP WAS DURING
4652 :EXECUTION OF THE FPS INSTRUCTION BEING TESTED. IF SO REPORT
4653 :FAILURE. OTHERWISE GO TO THE SPURIOUS TRAP TO 4 HANDLING.
4654 017206 AAC20:
4655 017206 011602 MOV (SP),R2
4656 017210 020227 017124 CMP R2,#AAC2+2
4657 017214 001405 BEQ 1$
4658 017216 020227 017126 CMP R2,#AAC2+4
4659 017222 001402 BEQ 1$
4660 017224 000137 036172 JMP @#CPSPUR
4661 017230 022626 1$: CMP (SP)+,(SP)+
4662 017232 010237 001236 MOV R2,@#STMP2
4663 017236 104001 2$: ERROR 1 ;ODD ADRES
4664 017240 000400 BR AACDONE ;BUT FDSTX IN ST 771
4665
4666 017242 AACDONE:
4667 017242 104413 RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
4668 ;SEE IF THE USER HAS EXPRESSED
4669 ;THE DESIRE TO CHANGE THE SOFTWARE
4670 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
4671 ;THE USER TYPED CONTROL G?).
4672
4673
4674
4675 :*****
4676 :*TEST 46 SOURCE MODES, MODE 2 (FL=0), TEST
4677 :*
4678 :* THIS IS A TEST OF SOURCE MODE 2
4679 :* USING THE LDFPS INSTR
4680 :*
4681 :*****
4682 TST46: SCOPE
4683
4684 017246 BBC1:
4685 017246 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
4686 017250 012700 017326 MOV #BBC1P1,R0 ;SET UP TEST DATA IN BUFFER.
4687 017254 012710 145212 MOV #145212,(R0)
4688 017260 012737 145212 001240 MOV #145212,@#STMP3 ;SAVE DATA IN CASE OF ERROR.
4689 017266 012737 017302 001236 MOV #BBC2,@#STMP2
4690 017274 012737 017366 000004 MOV #BBC20,@#ERRVECT ;SET UP FOR TRAPS TO 4.
4691
4692 017302 170120 BBC2: LDFPS (R0)+ ;TEST INSTRUCTION.
4693
4694 017304 170205 STFPS R5 ;GET FPS
4695
4696 017306 020027 017330 CMP R0,#BBC1P1+2 ;IS R0 CORRECT?
4697 017312 001007 BNE BBC10 ;BR IF NOT.
4698 017314 022705 145212 CMP #145212,R5 ;IS THE FPS CORRECT?
```



```
4699 017320 001013          BNE   BBC11          ;BR IF NOT.
4700 017322 000436          BR    BBCDONE
4701
4702
4703          ;TEST BUFFER AND DATA:
4704 017324 177777          -1
4705 017326 177777          BBCTP1: .WORD  -1
4706 017330 177777          -1
4707
4708
4709          ;REPORT R0 INCORRECT.
4710 017332 012737 017330 001240 BBC10: MOV   #BBCTP1+2,@#STMP3
4711 017340 010037 001242          MOV   R0,@#STMP4
4712 017344 104001          1$:   ERROR 1          ;R0 BAD BUT FSRC FAILED
4713 017346 000424          BR    BBCDONE
4714
4715          ;REPORT FPS INCORRECT.
4716 017350 012737 145212 001240 BBC11: MOV   #145212,@#STMP3 ;REPORT FPS INCORRECT.
4717 017356 010537 001242          MOV   R5,@#STMP4
4718 017362 104001          1$:   ERROR 1
4719 017364 000415          BR    BBCDONE
4720
4721          ;TRAP HERE THROUGH VECTOR FOUR. SEE IF THE TRAP WAS DURING
4722          ;EXECUTION OF THE FPS INSTRUCTION BEING TESTED. IF SO REPORT
4723          ;FAILURE. OTHERWISE GO TO THE SPURIOUS TRAP TO 4 HANDLING.
4724 017366          BBC20:
4725 017366 011602          MOV   (SP),R2
4726 017370 020227 017304          CMP   R2,#BBC2+2
4727 017374 001405          BEQ  1$
4728 017376 020227 017306          CMP   R2,#BBC2+4
4729 017402 001402          BEQ  1$
4730 017404 000137 036172          JMP  @#CPSPUR
4731 017410 022626          1$:   CMP   (SP)+,(SP)+
4732 017412 010237 001236          MOV   R2,@#STMP2
4733 017416 104001          2$:   ERROR 1          ;ODD ADRES
4734          ;BUT FDSTX IN ST 771
4735
4736 017420          BBCDONE:
4737 017420 104413          RSETUP          ;GO INITIALIZE THE FPS AND STACK; AND
4738          ;SEE IF THE USER HAS EXPRESSED
4739          ;THE DESIRE TO CHANGE THE SOFTWARE
4740          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
4741          ;THE USER TYPED CONTROL G?).
4742
4743
4744          ;*****
4745          ;*TEST 47          SOURCE MODES, MODE 4 (FL=0), TEST
4746          ;*
4747          ;* THIS IS A TEST OF SOURCE MODE 4
4748          ;* USING THE LDFPS INSTR
4749          ;*
4750          ;*****
4751 017422 000004          TST47: SCOPE
4752
4753 017424          DDC1:
4754 017424 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
```

```
4755
4756 017426 012700 017516          MOV      #DDCTP1+2,R0      ;SET UP THE TEST DATA BUFFER.
4757 017432 012760 105252 177776    MOV      #105252,-2(R0)
4758 017440 012737 105252 001240    MOV      #105252,@#STMP3 ;SAVE DATA IN CASE OF ERROR.
4759 017446 012737 017462 001236    MOV      #DDC2,@#STMP2
4760 017454 012737 017562 000004    MOV      #DDC20,@#ERRVEC
4761 017462 170140          DDC2:   LDFPS  -(R0)
4762 017464 170205          STFPS  R5
4763 017466 020027 017514          CMP      R0,#DDCTP1
4764 017472 001015          BNE     DDC10
4765 017474 022705 105252          CMP      #105252,R5
4766 017500 001021          BNE     DDC11
4767 017502 000444          BR      DDCDONE
4768
4769 017504 177777 177777 177777    -1,-1,-1,-1
4770 017512 177777
4771 017514 177777
4772 017516 177777 177777 177777    DDC10:  -1
4773 017524 177777          -1,-1,-1,-1
4774
4775 017526 012737 017514 001240    DDC10:  MOV      #DDCTP1,@#STMP3
4776 017534 010037 001242          MOV      R0,@#STMP4
4777 017540 104001          1$:    ERROR  1          ;R0 BAD BUT FSRC FAILED
4778 017542 000424          BR      DDCDONE
4779 017544 012737 105252 001240    DDC11:  MOV      #105252,@#STMP3 ;REPORT FPS INCORRECT.
4780 017552 010537 001242          MOV      R5,@#STMP4
4781 017556 104001          1$:    ERROR  1
4782 017560 000415          BR      DDCDONE
4783 017562 011602          DDC20:  MOV      (SP),R2
4784 017564 020227 017464          CMP      R2,#DDC2+2
4785 017570 001405          BEQ     1$
4786 017572 020227 017466          CMP      R2,#DDC2+4
4787 017576 001402          BEQ     1$
4788 017600 000137 036172          JMP      @#CPSPUR
4789 017604 022626          1$:    CMP      (SP)+,(SP)+
4790 017606 010237 001236          MOV      R2,@#STMP2
4791 017612 104001          2$:    ERROR  1          ;DDD ADRES
4792 017614
4793 017614 104413          DDCDONE: RSETUP          ;GO INITIALIZE THE FPS AND STACK; AND
4794
4795
4796
4797
4798
4799
4800
4801
4802
4803
4804
4805 017616 000004          ;*****
4806 017620          ;*TEST 50 SOURCE MODES, MODE 3 (FL=0), TEST
4807 017620 104414          ;*
4808 017622 012700 017724          ;* THIS IS A TEST OF SOURCE MODE 3
4809 017626 012710 017714          ;* USING THE LDFPS INSTR
4810 017632 012767 103456 000054          ;*
          ;*****
          TST50: SCOPE
          EEC1:
          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
          MOV      #EECTP2,R0
          MOV      #EECTP1,(R0)
          MOV      #103456,EECTP1
```

```

4811 017640 012737 103456 001240      MOV      #103456,@#$TMP3
4812 017646 012737 017662 001236      MOV      #EEC2,@#$TMP2
4813 017654 012737 017772 000004      MOV      #EEC20,@#ERRVECT ;SET UP FOR TRAPS TO 4.
4814 017662 170130          EEC2:  LDFPS  @(R0)+          ;TEST INSTRUCTION.
4815 017664 170205          STFPS   R5              ;GET THE FPS.
4816 017666 020027 017726      CMP      R0,#EECTP2+2    ;IS R0 CORRECT?
4817 017672 001021          BNE     EEC10            ;BR IF NOT.
4818 017674 022705 103456      CMP      #103456,R5      ;IS THE FPS CORRECT?
4819 017700 001025          BNE     EEC11            ;BR IF NOT.
4820 017702 000450          BR      EECDONE
4821
4822
4823          ;TEST BUFFER AND DATA:
4824 017704 177777 177777 177777      -1,-1,-1,-1
4825 017712 177777
4826 017714 177777      EECTP1: -1
4827 017716 177777 177777 177777      -1,-1,-1
4828 017724 017714 177777 177777      EECTP2: EECTP1,-1,-1,-1,
4829 017732 177777 000000
4830
4831
4832          ;REPORT R0 INCORRECT.
4833 017736 012737 017726 001240      EEC10:  MOV      #EECTP2+2,@#$TMP3
4834 017744 010037 001242          MOV      R0,@#$TMP4
4835 017750 104001          1$:     ERROR   1          ;R0 BAD BUT FSRC FAILED
4836 017752 000424          BR      EECDONE
4837
4838          ;REPORT FPS INCORRECT.
4839 017754 012737 103456 001240      EEC11:  MOV      #103456,@#$TMP3 ;REPORT FPS INCORRECT.
4840 017762 010537 001242          MOV      R5,@#$TMP4
4841 017766 104001          1$:     ERROR   1
4842 017770 000415          BR      EECDONE
4843          ;TRAP HERE THROUGH VECTOR FOUR. SEE IF THE TRAP WAS DURING
4844          ;EXECUTION OF THE FPS INSTRUCTION BEING TESTED. IF SO REPORT
4845          ;FAILURE. OTHERWISE GO TO THE SPURIOUS TRAP TO 4 HANDLING.
4846 017772 011602          EEC20:  MOV      (SP),R2
4847 017774 020227 017664          CMP      R2,#EEC2+2
4848 020000 001405          BEQ     1$
4849 020002 020227 017666          CMP      R2,#EEC2+4
4850 020006 001402          BEQ     1$
4851 020010 000137 036172          JMP     @#CPSPUR
4852 020014 022626          1$:     CMP      (SP)+,(SP)+
4853 020016 010237 001236          MOV      R2,@#$TMP2
4854 020022 104001          2$:     ERROR   1          ;DDD ADRES
4855 020024          EECDONE:
4856 020024 104413          RSETUP          ;GO INITIALIZE THE FPS AND STACK; AND
4857          ;SEE IF THE USER HAS EXPRESSED
4858          ;THE DESIRE TO CHANGE THE SOFTWARE
4859          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
4860          ;THE USER TYPED CONTROL G?).
4861          ;*****
4862          ;*TEST 51          SOURCE MODES, MODE 5 (FL=0), TEST
4863          ;*
4864          ;* THIS IS A TEST OF SOURCE MODE 5
4865          ;* USING THE LDFPS INSTR
4866          ;*

```

```

4867
4868 020026 000004
4869 020030
4870 020030 104414
4871 020032 012700 020132
4872 020036 012760 020120 177776
4873 020044 012737 045412 020120
4874 020052 012737 045412 001240
4875 020060 012737 020030 001236
4876 020066 012737 020174 000004
4877 020074 170150
4878 020076 170205
4879 020100 020027 020130
4880 020104 001015
4881 020106 022705 045412
4882 020112 001021
4883 020114 000444
4884
4885
4886
4887 020116 177777
4888 020120 177777
4889 020122 177777 177777 177777
4890 020130 020120 177777 177777
4891 020136 177777
4892
4893
4894
4895 020140 012737 020130 001240
4896 020146 010037 001242
4897 020152 104001
4898 020154 000424
4899
4900
4901 020156 012737 045412 001240
4902 020164 010537 001242
4903 020170 104001
4904 020172 000415
4905
4906
4907
4908 020174 011602
4909 020176 020227 020076
4910 020202 001405
4911 020204 020227 020100
4912 020210 001402
4913 020212 000137 036172
4914 020216 022626
4915 020220 010237 001236
4916 020224 104001
4917 020226
4918 020226 104413
4919
4920
4921
4922

:*****
TST51: SCOPE
FFC1:
      LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
      MOV #FFCTP2+2,R0 ;SET UP THE TEST DATA BUFFER.
      MOV #FFCTP1,-2(R0)
      MOV #45412,@#FFCTP1
      MOV #45412,@#STMP3 ;SAVE DATA IN CASE OF ERROR.
      MOV #FFC1,@#STMP2
      MOV #FFC20,@#ERRVECT ;SET UP FOR TRAPS TO 4.
FFC2: LDFPS @-(R0) ;TEST INSTRUCTION.
      STFPS R5 ;GET THE FPS.
      CMP R0,#FFCTP2 ;IS R0 CORRECT?
      BNE FFC10 ;BR IF NOT.
      CMP #45412,R5 ;IS THE FPS CORRECT?
      BNE FFC11 ;BR IF NOT.
      BR FFCDONE

;TEST BUFFER AND DATA:
      -1
FFCTP1: -1
      -1,-1,-1
FFCTP2: FFCTP1,-1,-1,-1

;REPORT R0 INCORRECT.
FFC10: MOV #FFCTP2,@#STMP3
      MOV R0,@#STMP4
1$: ERROR 1 ;R0 BAD BUT FSRC FAILED
      BR FFCDONE

;REPORT FPS INCORRECT.
FFC11: MOV #45412,@#STMP3 ;REPORT FPS INCORRECT.
      MOV R5,@#STMP4
1$: ERROR 1
      BR FFCDONE

;TRAP HERE THROUGH VECTOR FOUR. SEE IF THE TRAP WAS DURING
;EXECUTION OF THE FPS INSTRUCTION BEING TESTED. IF SO REPORT
;FAILURE. OTHERWISE GO TO THE SPURIOUS TRAP TO 4 HANDLING.
FFC20: MOV (SP),R2
      CMP R2,#FFC2+2
      BEQ 1$
      CMP R2,#FFC2+4
      BEQ 1$
      JMP @#CPSPUR
1$: CMP (SP)+,(SP)+
      MOV R2,@#STMP2
2$: ERROR 1 ;ODD ADRES
FFCDONE: 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?).

```

```
4923
4924
4925
4926
4927
4928
4929
4930 020230 000004
4931 020232
4932 020232 104414
4933 020234 012700 013123
4934 020240 012737 046543 020324
4935 020246 012737 046543 001240
4936 020254 012737 020272 001236
4937 020262 005001
4938 020264 012737 020412 000004
4939 020272 170160 005201
4940 020276 170204
4941 020300 005701
4942 020302 001033
4943 020304 020027 013123
4944 020310 001012
4945 020312 022704 046543
4946 020316 001016
4947 020320 000451
4948
4949
4950
4951 020322 177777
4952 020324 177777 177777 177777
4953 020332 177777
4954 020334 177777
4955
4956
4957 020336 012737 013123 001240
4958 020344 010037 001242
4959 020350 104001
4960 020352 000434
4961
4962
4963 020354 012737 046543 001240
4964 020362 010437 001242
4965 020366 104001
4966 020370 000425
4967
4968
4969 020372 012737 020276 001240
4970 020400 012737 020274 001242
4971 020406 104001
4972 020410 000415
4973
4974
4975
4976 020412 011602
4977 020414 020227 020274
.978 020420 001405

*****
*TEST 52 SOURCE MODES, MODE 6 (FL=0), TEST
*
* THIS IS A TEST OF SOURCE MODE 6
* USING THE LDFPS INSTR
*
*****
TST52: SCOPE
GGC1:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
MOV #GGCTP1-5201,R0 ;SET UP THE TEST DATA BUFFER.
MOV #46543,@#GGCTP1
MOV #46543,@#STMP3 ;SAVE DATA IN CASE OF ERROR.
MOV #GGC2,@#STMP2
CLR R1
MOV #GGC20,@#ERRVECT ;SET UP FOR TRAPS TO 4.
GGC2: LDFPS 5201(R0) ;TEST INSTRUCTION.
STFPS R4 ;GET THE FPS.
TST R1 ;WAS PC CORRECT AFTER EXECUTION?
BNE GGC25 ;BR IF NOT.
CMP R0,#GGCTP1-5201 ;IS R0 CORRECT?
BNE GGC10 ;BR IF NOT.
CMP #46543,R4 ;IS THE FPS CORRECT?
BNE GGC11 ;BR IF NOT.
BR GGCDONE

;TEST BUFFER AND DATA:
-1
GGCTP1: -1,-1,-1,-1
-1

;REPORT R0 INCORRECT.
GGC10: MOV #GGCTP1-5201,@#STMP3
MOV R0,@#STMP4
1$: ERROR 1 ;R0 BAD BUT FSRC FAILED
BR GGCDONE

;REPORT FPS INCORRECT.
GGC11: MOV #46543,@#STMP3 ;REPORT FPS INCORRECT.
MOV R4,@#STMP4
1$: ERROR 1
BR GGCDONE

;REPORT PC INCORRECT AFTER INSTRUCTION.
GGC25: MOV #GGC2+4,@#STMP3
MOV #GGC2+2,@#STMP4
1$: ERROR 1 ;PC X
BR GGCDONE

;TRAP HERE THROUGH VECTOR FOUR. SEE IF THE TRAP WAS DURING
;EXECUTION OF THE FPS INSTRUCTION BEING TESTED. IF SO REPORT
;FAILURE. OTHERWISE GO TO THE SPURIOUS TRAP TO 4 HANDLING.
GGC20: MOV (SP),R2
CMP R2,#GGC2+2
BEQ 1$
```

```
4979 020422 020227 020276          CMP      R2,#GGC2+4
4980 020426 001402                    BEQ      1$
4981 020430 000137 036172          JMP      @#CPSPUR
4982 020434 022626          1$:    CMP      (SP)+,(SP)+
4983 020436 010237 001236          MOV      R2,@#STMP2
4984 020442 104001          2$:    ERROR  1          ;ODD ADRES
4985 020444
4986 020444 104413          GGCDONE: RSETUP          ;GO INITIALIZE THE FPS AND STACK; AND
4987
4988
4989
4990
4991
4992
4993
4994
4995
4996
4997
4998 020446 000004          ;*****
4999 020450          ;*TEST 53          SOURCE MODES, MODE 7 (FL=0), TEST
5000 020450 104414          ;*
5001 020452 012700 013357          ;* THIS IS A TEST OF SOURCE MODE 7
5002 020456 012760 020550 005201          ;* USING THE LDFPS INSTR
5003 020464 012737 004547 020550          ;*
5004 020472 012737 004547 001240          ;*
5005 020500 012737 020516 001236          ;*
5006 020506 005001          ;*****
5007 020510 012737 020644 000004          TST53: SCOPE
5008 020516 170170 005201          HHC1:
5009 020522 170204          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
5010 020524 005701          MOV      #HHCTP2-5201,R0 ;SET UP THE TEST DATA BUFFER.
5011 020526 001036          MOV      #HHCTP1,5201(R0)
5012 020530 020027 013357          MOV      #4547,@#HHCTP1
5013 020534 001015          MOV      #4547,@#STMP3 ;SAVE DATA IN CASE OF ERROR.
5014 020536 022704 004547          MOV      #HHC2,@#STMP2
5015 020542 001021          CLR      R1
5016 020544 000454          MOV      #HHC20,@#ERRVECT ;SET UP FOR TRAPS TO 4.
5017
5018
5019          HHC2: LDFPS @5201(R0) ;TEST INSTRUCTION.
5020 020546 177777          STFPS R4 ;GET THE FPS.
5021 020550 177777 177777 177777          TST R1 ;WAS PC CORRECT AFTER EXECUTION?
5022 020556 177777          BNE HHC25 ;BR IF NOT.
5023 020560 177777 177777 177777          CMP R0,#HHCTP2-5201 ;IS R0 CORRECT?
5024 020566 177777          BNE HHC10 ;BR IF NOT.
5025
5026          CMP #4547,R4 ;IS THE FPS CORRECT?
5027 020570 012737 013357 001240          BNE HHC11 ;BR IF NOT.
5028 020576 010037 001242          BR HHCDONE
5029 020602 104001          ;TEST BUFFER AND DATA:
5030 020604 000434          -1
5031
5032          HHC10: .WORD -1,-1,-1,-1
5033 020606 012737 004547 001240          HHC10: MOV #HHCTP2-5201,@#STMP3
5034 020614 010437 001242          MOV R0,@#STMP4
          1$: ERROR 1 ;R0 BAD BUT FSRC FAILED
          BR HHCDONE
          ;REPORT FPS INCORRECT.
          HHC11: MOV #4547,@#STMP3 ;REPORT FPS INCORRECT.
          MOV R4,@#STMP4
```

```

5035 020620 104001          1$:  ERROR 1
5036 020622 000425          BR    HHC DONE
5037
5038 ;REPORT PC INCORRECT AFTER INSTRUCTION.
5039 020624 012737 020522 001240 HHC25: MOV  #HHC2+4,@#STMP3
5040 020632 012737 020520 001242      MOV  #HHC2+2,@#STMP4
5041 020640 104001          1$:  ERROR 1 ;PC X
5042 020642 000415          BR    HHC DONE
5043 ;TRAP HERE THROUGH VECTOR FOUR. SEE IF THE TRAP WAS DURING
5044 ;EXECUTION OF THE FPS INSTRUCTION BEING TESTED. IF SO REPORT
5045 ;FAILURE. OTHERWISE GO TO THE SPURIOUS TRAP TO 4 HANDLING.
5046 020644 011602          HHC20: MOV  (SP),R2
5047 020646 020227 020520      CMP   R2,#HHC2+2
5048 020652 001405          BEQ   1$
5049 020654 020227 020522      CMP   R2,#HHC2+4
5050 020660 001402          BEQ   1$
5051 020662 000137 036172      JMP   @#CPSPUR
5052 020666 022626          1$:  CMP   (SP)+,(SP)+
5053 020670 010237 001236      MOV   R2,@#STMP2
5054 020674 104001          2$:  ERROR 1 ;DDD ADDRESS
5055 020676
5056 020676 104413          HHC DONE: RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
5057 ;SEE IF THE USFR HAS EXPRESSED
5058 ;THE DESIRE TO CHANGE THE SOFTWARE
5059 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
5060 ;THE USER TYPED CONTROL G?).
5061
5062
5063
5064
5065 :*****
5066 :*TEST 54 SOURCE MODES, MODE 2 GR7 (FL=1), TEST
5067 :*
5068 :* THIS IS A TEST OF THE LDCLD WITH
5069 :* IMMEDIATE ADDRESSING MODE
5070 :*****
5071 020700 000004          TST54: SCOPE
5072
5073 IIC1:
5074 020702 104414          LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
5075 020704 012737 020730 001236      MOV   #IIC2,@#STMP2 ;SAVE DATA IN CASE OF ERROR.
5076 020712 012737 021002 000004      MOV   #IIC20,@#ERRVECT ;SET UP FOR TRAPS TO 4.
5077 020720 012700 000300      MOV   #300,R0
5078 020724 170100          LDFPS R0
5079 020726 005001          CLR   R1
5080
5081 IIC2: LDCLD (R7)+,ACO ;TEST INSTRUCTION.
5082 020732 005201          5201
5083 020734 005201          5201
5084 020736 005201          5201
5085 020740 005201          5201
5086
5087 020742 020127 000003      CMP   R1,#3 ;WAS PC CORRECT AFTER EXECUTION?
5088 020746 001421          BEQ   IIC DONE ;BR IF YES.
5089
5090
    
```

5091
5092 020750 012704 020734
5093 020754 162701 000003
5094 020760 006301
5095 020762 160104
5096 020764 010437 001242
5097 020770 012737 020734 001240
5098 020776 104001
5099 021000 000404
5100
5101
5102
5103 021002 011637 001236
5104 021006 022626
5105 021010 104001
5106
5107 021012
5108 021012 104413
5109
5110
5111
5112
5113
5114
5115
5116
5117
5118
5119
5120
5121
5122 021014 000004
5123
5124 021016
5125 021016 104414
5126 021020 016737 000014 001236
5127 021026 012700 000300
5128 021032 170100
5129 021034 012700 021130
5130 021040 177020
5131
5132 021042 170204
5133 021044 012701 021140
5134 021050 012702 000200
5135 021054 170102
5136 021056 174011
5137 021060 020027 021134
5138 021064 001407
5139
5140 021066 010037 001242
5141 021072 012737 021134 001240
5142 021100 104001
5143 021102 000422
5144
5145 021104 022704 000300
5146 021110 001417

```

;REPORT PC INCORRECT AFTER INSTRUCTION.
IIC3:  MOV    #IIC2+4,R4
      SUB    #3,R1
      ASL    R1
      SUB    R1,R4
      MOV    R4,@#STMP4
      MOV    #IIC2+4,@#STMP3
1$:    ERROR  1          ;BAD CONSTANT
      BR    IICDONE

;TRAP HERE THROUGH VECTOR FOUR. SEE IF THE TRAP WAS DURING
;EXECUTION OF THE FPS INSTRUCTION BEING TESTED. IF SO REPORT
;FAILURE. OTHERWISE GO TO THE SPURIOUS TRAP TO 4 HANDLING.
IIC20: MOV    (SP),@#STMP2
      CMP    (SP)+,(SP)+
1$:    ERROR  1          ;BAD CONSTANT ODD ADD

IICDONE:
      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 55          SOURCE MODES, MODE 2 (FL=1), TEST
;*
;* THIS IS A TEST OF THE LDCLD INSTR
;* WITH MODE 2.
;*
;*****
TST55: SCOPE

TCC1:  LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
      MOV    TCC2,@#STMP2 ;SAVE DATA IN CASE OF ERROR.
      MOV    #300,R0
      LDFPS  R0
      MOV    #TCCBF0,R0 ;SET UP THE TEST DATA BUFFER.
TCC2:  LDCLD    (R0)+,ACO ;TEST INSTRUCTION.

      STFPS  R4          ;GET THE FPS.
      MOV    #TCCBF1,R1 ;GET THE RESULT.
      MOV    #200,R2
      LDFPS  R2
      STD   ACO,(R1)
      CMP    R0,#TCCBF0+4 ;IS R0 CORRECT?
      BEQ   TCC3

;REPORT R0 INCORRECT.
      MOV    R0,@#STMP4
      MOV    #TCCBF0+4,@#STMP3
1$:    ERROR  1          ;BAD CONST
      BR    ICCDONE

TCC3:  CMP    #300,R4 ;IS THE FPS CORRECT?
      BEQ   TCCDONE

```



```
5147  
5148 ;REPORT FPS INCORRECT.  
5149 021112 010437 001242 MOV R4,@#STMP4  
5150 021116 012737 000300 001240 MOV #300,@#STMP3  
5151 021124 104001 1$: ERROR 1 ;FPS X  
5152 021126 000410 BR TCCDONE  
5153  
5154  
5155 ;TEST BUFFER AND DATA:  
5156 021130 001234 067076 054321 TCCBF0: .WORD 01234,67076,54321,012345  
5157 021136 012345  
5158 021140 177777 177777 177777 TCCBF1: -1,-1,-1,-1  
5159 021146 177777  
5160  
5161 021150 TCCDONE:  
5162 021150 104413 RSETUP ;GO INITIALIZE THE FPS AND STACK; AND  
5163 ;SEE IF THE USER HAS EXPRESSED  
5164 ;THE DESIRE TO CHANGE THE SOFTWARE  
5165 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS  
5166 ;THE USER TYPED CONTROL G?).  
5167  
5168  
5169  
5170  
5171 :*****  
5172 :*TEST 56 LDCIF AND LDCLF TEST  
5173 :*  
5174 :* THIS IS A TEST OF THE LDCIF AND  
5175 :* THE LDCLF INSTRUCTIONS.  
5176 :*****  
5177 021152 000004 TST56: SCOPE  
5178  
5179  
5180 ;ZERO OPERAND FL=0  
5181  
5182 021154 KKC1:  
5183 021154 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.  
5184 021156 004737 022306 JSR PC,@#LDCFSUB ;GO EXECUTE INSTRUCTION.  
5185  
5186 021162 000000 000000 1$: .WORD 0,0 ;FSRC OPERAND.  
5187 021166 000000 000000 2$: .WORD 0,0 ;EXPECTED RESULT.  
5188 021172 177777 177777 3$: .WORD -1,-1 ;ANTICIPATED ERRONEOUS RESULT.  
5189 021176 000000 4$: 0 ;FPS BEFORE EXECUTION.  
5190 021200 000004 4 4 ;FPS AFTER EXECUTION.  
5191 021202 177777 -1 ;ANTICIPATED ERRONEOUS FPS.  
5192 021204 104001 5$: ERROR 1 ;REPORT RESULT INCORRECT.  
5193 021206 000401 BR 6$  
5194 021210 104001 ERROR 1 ;REPORT FPS INCORRECT.  
5195 021212 6$:  
5196 ;ZERO OPERAND FL=0  
5197  
5198 021212 KKC2:  
5199 021212 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.  
5200 021214 004737 022306 JSR PC,@#LDCFSUB ;GO EXECUTE THE INSTRUCTION.  
5201  
5202 021220 000000 177777 1$: .WORD 0,-1 ;FSRC OPERAND.
```

5203	021224	000000	000000	2\$:	.WORD	0,0		:EXPECTED RESULT.
5204	021230	004177	177400	3\$:	4177,177400			:ANTICIPATED ERRONEOUS RESULT.
5205	021234	000000		4\$:	0			:FPS BEFORE EXECUTION.
5206	021236	000004			4			:FPS AFTER EXECUTION.
5207	021240	177777			-1			:ANTICIPATED ERRONEOUS FPS.
5208	021242	104001		5\$:	ERROR	1		:(BUT FL) ST
5209	021244	000401			BR	6\$:277 TO 300
5210	021246	104001			ERROR	1		:INTO 301
5211	021250			6\$:				
5212				;ZERO	OPERAND	FL=1		
5213								
5214	021250			KKC3:				
5215	021250	104414			LPERR			:SET UP THE LOOP ON ERROR ADDRESS.
5216	021252	004737	022306		JSR	PC,@#LDCFSUB		:GO EXECUTE THE INSTRUCTION.
5217								
5218	021256	000000	000000	1\$:	.WORD	0,0		:FSRC OPERAND.
5219	021262	000000	000000	2\$:	.WORD	0,0		:EXPECTED RESULT.
5220	021266	177777	177777	3\$:	.WORD	-1,-1		:ANTICIPATED ERRONEOUS RESULT.
5221	021272	000100		4\$:	100			:FPS BEFORE EXECUTION.
5222	021274	000104			104			:FPS AFTER EXECUTION.
5223	021276	000004			4			:ANTICIPATED ERRONEOUS FPS.
5224	021300	104001		5\$:	ERROR	1		:REPORT RESULT INCORRECT.
5225	021302	000401			BR	6\$		
5226	021304	104001			ERROR	1		:FL WAS CLR'ED
5227	021306			6\$:				
5228				;OPERAND	POSITIVE	FL=0		
5229	021306			KKC4:				
5230	021306	104414			LPERR			:SET UP THE LOOP ON ERROR ADDRESS.
5231	021310	004737	022306		JSR	PC,@#LDCFSUB		:GO EXECUTE THE INSTRUCTION.
5232	021314	040000	000000	1\$:	.WORD	40000,0		:FSRC OPERAND.
5233	021320	043600	000000	2\$:	.WORD	43600,0		:EXPECTED RESULT.
5234	021324	047600	000000	3\$:	.WORD	47600,0		:ANTICIPATED ERRONEOUS RESULT.
5235	021330	000017		4\$:	17			:FPS BEFORE EXECUTION.
5236	021332	000000			0			:FPS AFTER EXECUTION.
5237	021334	177777			-1			:ANTICIPATED ERRONEOUS FPS.
5238	021336	104001		5\$:	ERROR	1		:ST 107 BAD
5239	021340	000401			BR	6\$:CONSTANT 231 INSD
5240	021342	104001			ERROR	1		:215
5241	021344			6\$:				
5242				;OPERAND=1,	FL=0			
5243	021344			KKC5:				
5244	021344	104414			LPERR			:SET UP THE LOOP ON ERROR ADDRESS.
5245	021346	004737	022306		JSR	PC,@#LDCFSUB		:GO EXECUTE THE INSTRUCTION.
5246	021352	000001	000000	1\$:	.WORD	1,0		:FSRC OPERAND.
5247	021356	040200	000000	2\$:	.WORD	40200,0		:EXPECTED RESULT.
5248	021362	044200	000000	3\$:	.WORD	44200,0		:ANTICIPATED ERRONEOUS RESULT.
5249	021366	000017		4\$:	17			:FPS BEFORE EXECUTION.
5250	021370	000000			0			:FPS AFTER EXECUTION.
5251	021372	177777			-1			:ANTICIPATED ERRONEOUS FPS.
5252	021374	104001		5\$:	ERROR	1		:REPORT RESULT INCORRECT.
5253	021376	000401			BR	6\$		
5254	021400	104001			ERROR	1		:REPORT FPS INCORRECT.
5255	021402			6\$:				
5256								
5257								
5258				;OPERAND-	PATTERN	FL=0		

5259 021402
5260 021402 104414
5261 021404 004737 022306
5262 021410 000252 000000
5263 021414 042052 000000
5264 021420 046052 000000
5265 021424 000000
5266 021426 000000
5267 021430 177777
5268 021432 104000
5269 021434 000401
5270 021436 104001
5271 021440
5272
5273
5274 021440
5275 021440 104414
5276 021442 004737 022306
5277 021446 140000 000000
5278 021452 143600 000000
5279 021456 043600 000000
5280 021462 000007
5281 021464 000010
5282 021466 177777
5283 021470 104001
5284 021472 000401
5285 021474 104001
5286 021476
5287
5288
5289 021476
5290 021476 104414
5291 021500 004737 022306
5292 021504 177777 000000
5293 021510 140200 000000
5294 021514 144000 000400
5295 021520 000000
5296 021522 000010
5297 021524 177777
5298 021526 104001
5299 021530 000401
5300 021532 104001
5301 021534
5302
5303
5304 021534
5305 021534 104414
5306 021536 004737 022306
5307 021542 125252 000000
5308 021546 143652 126000
5309 021552 043652 126000
5310 021556 000007
5311 021560 000010
5312 021562 177777
5313 021564 104001
5314 021566 000401

KKC6:

LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,@#LDCFSUB ;GO EXECUTE THE INSTRUCTION.
1\$: .WORD 252,0 ;FSRC OPERAND.
2\$: .WORD 42052,0 ;EXPECTED RESULT.
3\$: .WORD 46052,0 ;ANTICIPATED ERRONEOUS RESULT.
4\$: 0 ;FPS BEFORE EXECUTION.
0 ;FPS AFTER EXECUTION.
-1 ;ANTICIPATED ERRONEOUS FPS.
5\$: ERROR ;REPORT RESULT INCORRECT.
BR 6\$
ERROR 1 ;REPORT FPS INCORRECT.

;OPERAND=-40000 FL=0

KKC7:

LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,@#LDCFSUB ;GO EXECUTE THE INSTRUCTION.
1\$: .WORD -40000,0 ;FSRC OPERAND.
2\$: .WORD 143600,0 ;EXPECTED RESULT.
3\$: .WORD 43600,0 ;ANTICIPATED ERRONEOUS RESULT.
4\$: 7 ;FPS BEFORE EXECUTION.
10 ;FPS AFTER EXECUTION.
-1 ;ANTICIPATED ERRONEOUS FPS.
5\$: ERROR 1 ;(SET SIGN) ST 146
BR 6\$
ERROR 1 ;REPORT FPS INCORRECT.

;OPERAND=-1 FL=0

KKC8:

LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,@#LDCFSUB ;GO EXECUTE THE INSTRUCTION.
1\$: .WORD -1,0 ;FSRC OPERAND.
2\$: .WORD 140200,0 ;EXPECTED RESULT.
3\$: .WORD 144000,400 ;ANTICIPATED ERRONEOUS RESULT.
4\$: 0 ;FPS BEFORE EXECUTION.
10 ;FPS AFTER EXECUTION.
-1 ;ANTICIPATED ERRONEOUS FPS.
5\$: ERROR 1 ;ST 372 TO 152 INTO
BR 6\$;112 (BUF XNBT)
ERROR 1 ;REPORT FPS INCORRECT.

;OPERAND=PATTERN FL=0

KKC9:

LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,@#LDCFSUB ;GO EXECUTE THE INSTRUCTION.
1\$: .WORD 125252,0 ;FSRC OPERAND.
2\$: .WORD 143652,126000 ;EXPECTED RESULT.
3\$: .WORD 43652,126000 ;ANTICIPATED ERRONEOUS RESULT.
4\$: 7 ;FPS BEFORE EXECUTION.
10 ;FPS AFTER EXECUTION.
-1 ;ANTICIPATED ERRONEOUS FPS.
5\$: ERROR 1 ;REPORT RESULT INCORRECT.
BR 6\$

```
5315 021570 104001          ERROR 1          ,REPORT FPS INCORRECT.
5316 021572          6$:
5317
5318          ;OPERAND      POS      FL-1
5319          KKC10:
5320 021572 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
5321 021574 004737 022306  JSR      PC,@#LDCFSUB ;GO EXECUTE THE INSTRUCTION.
5322 021600 040000 000000  1$:      .WORD 40000,0 ;FSRC OPERAND.
5323 021604 047600 000000  2$:      .WORD 47600,0 ;EXPECTED RESULT.
5324 021610 043600 000000  3$:      .WORD 43600,0 ;ANTICIPATED ERRONEOUS RESULT.
5325 021614 000117          4$:      117          ;FPS BEFORE EXECUTION.
5326 021616 000100          ;FPS AFTER EXECUTION.
5327 021620 177777          ;ANTICIPATED ERRONEOUS FPS.
5328 021622 104001          5$:      ERROR 1          ;ST 107 CONSTANT
5329 021624 000401          BR      6$          ;BAD 237 INST 217
5330 021626 104001          ERROR 1          ;REPORT FPS INCORRECT.
5331 021630          6$:
5332
5333          ;OPERAND-1    FL=1
5334 021630          KKC11:
5335 021630 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
5336 021632 004737 022306  JSR      PC,@#LDCFSUB ;GO EXECUTE THE INSTRUCTION.
5337 021636 000000 000001  1$:      .WORD 0,1 ;FSRC OPERAND.
5338 021642 040200 000000  2$:      .WORD 40200,0 ;EXPECTED RESULT.
5339 021646 034200 000000  3$:      .WORD 34200,0 ;ANTICIPATED ERRONEOUS RESULT.
5340 021652 000100          4$:      100          ;FPS BEFORE EXECUTION.
5341 021654 000100          ;FPS AFTER EXECUTION.
5342 021656 177777          ;ANTICIPATED ERRONEOUS FPS.
5343 021660 104001          5$:      ERROR 1          ;REPORT RESULT INCORRECT.
5344 021662 000401          BR      6$          ;
5345 021664 104001          ERROR 1          ;REPORT FPS INCORRECT.
5346 021666          6$:
5347
5348          ;OPERAND=      PATTERN FL=1
5349 021666          KKC12:
5350 021666 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
5351 021670 004737 022306  JSR      PC,@#LDCFSUB ;GO EXECUTE THE INSTRUCTION.
5352 021674 000000 000252  1$:      .WORD 0,252 ;FSRC OPERAND.
5353 021700 042052 000000  2$:      .WORD 42052,0 ;EXPECTED RESULT.
5354 021704 036052 000000  3$:      .WORD 36052,0 ;ANTICIPATED ERRONEOUS RESULT.
5355 021710 000111          4$:      111          ;FPS BEFORE EXECUTION.
5356 021712 000100          ;FPS AFTER EXECUTION.
5357 021714 177777          ;ANTICIPATED ERRONEOUS FPS.
5358 021716 104001          5$:      ERROR 1          ;REPORT RESULT INCORRECT.
5359 021720 000401          BR      6$          ;
5360 021722 104001          ERROR 1          ;REPORT FPS INCORRECT.
5361 021724          6$:
5362
5363          ;OPERAND=-40000,0  FL=1
5364 021724          KKC13:
5365 021724 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
5366 021726 004737 022306  JSR      PC,@#LDCFSUB ;GO EXECUTE THE INSTRUCTION.
5367 021732 140000 000000  1$:      .WORD -40000,0 ;FSRC OPERAND.
5368 021736 147600 000000  2$:      .WORD 147600,0 ;EXPECTED RESULT.
5369 021742 047600 000000  3$:      .WORD 47600,0 ;ANTICIPATED ERRONEOUS RESULT.
5370 021746 000107          4$:      107          ;FPS BEFORE EXECUTION.
```

5371	021750	0	10		110				;FPS AFTER EXECUTION.
5372	021752	177777			-1				;ANTICIPATED ERRONEOUS FPS.
5373	021754	104001			5\$:	ERROR	1		;SET SIGN
5374	021756	000401				BR	6\$		
5375	021760	104001				ERROR	1		;REPORT FPS INCORRECT.
5376	021762				6\$:				
5377									
5378									
5379	021762								;OPERAND=-1,-1 FL-1
5380	021762	104414			KKC14:				
5381	021764	004737	022306			LPERR			;SET UP THE LOOP ON ERROR ADDRESS.
5382	021770	177777	177777			JSR	PC,@#LDCFSUB		;GO EXECUTE THE INSTRUCTION.
5383	021774	140200	000000		1\$:	.WORD	-1,-1		;FSRC OPERAND.
5384	022000	150000	000000		2\$:	.WORD	140200,0		;EXPECTED RESULT.
5385	022004	000100			3\$:	.WORD	150000,0		;ANTICIPATED ERRONEOUS RESULT.
5386	022006	000110			4\$:	100			;FPS BEFORE EXECUTION.
5387	022010	177777				110			;FPS AFTER EXECUTION.
5388	022012	104001			5\$:	-1			;ANTICIPATED ERRONEOUS FPS.
5389	022014	000401				ERROR	1		; (BUT XNBT)
5390	022016	104001				BR	6\$		
5391	022020					ERROR	1		;REPORT FPS INCORRECT.
5392					6\$:				
5393									
5394	022020								;OPERAND=-PATTERN FL=1, ROUND MODE
5395	022020	104414			KKC15:				
5396	022022	004737	022306			LPERR			;SET UP THE LOOP ON ERROR ADDRESS.
5397	022026	125252	125252			JSR	PC,@#LDCFSUB		;GO EXECUTE THE INSTRUCTION.
5398	022032	147652	125253		1\$:	.WORD	125252,125252		;FSRC OPERAND.
5399	022036	047652	125253		2\$:	.WORD	147652,125253		;EXPECTED RESULT.
5400	022042	000105			3\$:	.WORD	47652,125253		;ANTICIPATED ERRONEOUS RESULT.
5401	022044	000110			4\$:	105			;FPS BEFORE EXECUTION.
5402	022046	177777				110			;FPS AFTER EXECUTION.
5403	022050	104001			5\$:	-1			;ANTICIPATED ERRONEOUS FPS.
5404	022052	000401				ERROR	1		;REPORT RESULT INCORRECT.
5405	022054	104001				BR	6\$		
5406	022056					ERROR	1		;REPORT FPS INCORRECT.
5407					6\$:				
5408									
5409	022056								;OPERAND=77777,177500 FL=1, ROUND MODE
5410	022056	104414			KKC16:				
5411	022060	004737	022306			LPERR			;SET UP THE LOOP ON ERROR ADDRESS.
5412	022064	077777	177500			JSR	PC,@#LDCFSUB		;GO EXECUTE THE INSTRUCTION.
5413	022070	047777	177777		1\$:	.WORD	77777,177500		;FSRC OPERAND.
5414	022074	047777	177776		2\$:	.WORD	47777,177777		;EXPECTED RESULT.
5415	022100	000117			3\$:	.WORD	47777,177776		;ANTICIPATED ERRONEOUS RESULT.
5416	022102	000100			4\$:	117			;FPS BEFORE EXECUTION.
5417	022104	177777				100			;FPS AFTER EXECUTION.
5418	022106	104001			5\$:	-1			;ANTICIPATED ERRONEOUS FPS.
5419	022110	000401				ERROR	1		;ST 631 INTO RND
5420	022112	104001				BR	6\$		
5421	022114					ERROR	1		;REPORT FPS INCORRECT.
5422					6\$:				
5423									
5424	022114								;OPERAND=40000,000100 FL=1, ROUND MODE
5425	022114	104414			KKC17:				
5426	022116	004737	022306			LPERR			;SET UP THE LOOP ON ERROR ADDRESS.
						JSR	PC,@#LDCFSUB		;GO EXECUTE THE INSTRUCTION.

```
5427 022122 040000 000100 1$: .WORD 40000,100 ;FSRC OPERAND.
5428 022126 047600 000001 2$: .WORD 47600,1 ;EXPECTED RESULT.
5429 022132 047600 000000 3$: .WORD 47600,0 ;ANTICIPATED ERRONEOUS RESULT.
5430 022136 000102 4$: 102 ;FPS BEFORE EXECUTION.
5431 022140 000100 100 ;FPS AFTER EXECUTION.
5432 022142 177777 -1 ;ANTICIPATED ERRONEOUS FPS.
5433 022144 104001 5$: ERROR 1 ;REPORT RESULT INCORRECT.
5434 022146 000401 BR 6$
5435 022150 104001 ERROR 1 ;REPORT FPS INCORRECT.
5436 022152 6$:
5437
5438 ;OPERAND=40000,000100 FL=1, TRUNC MODE
5439 022152 KKC18:
5440 022152 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
5441 022154 004737 022306 JSR PC,@#LDCFSUB ;GO EXECUTE THE INSTRUCTION.
5442 022160 040000 000100 1$: .WORD 40000,100 ;FSRC OPERAND.
5443 022164 047600 000000 2$: .WORD 47600,0 ;EXPECTED RESULT.
5444 022170 047600 000001 3$: .WORD 47600,1 ;ANTICIPATED ERRONEOUS RESULT.
5445 022174 000157 4$: 157 ;FPS BEFORE EXECUTION.
5446 022176 000140 140 ;FPS AFTER EXECUTION.
5447 022200 177777 -1 ;ANTICIPATED ERRONEOUS FPS.
5448 022202 104001 5$: ERROR 1 ;ST 631 ... INTO TRNC
5449 022204 000401 BR 6$
5450 022206 104001 ERROR 1 ;REPORT FPS INCORRECT.
5451 022210 6$:
5452 ;OPERAND=100000,0 (MOST NEG #) FL=0
5453 022210 KKC19:
5454 022210 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
5455 022212 004737 022306 JSR PC,@#LDCFSUB ;GO EXECUTE THE INSTRUCTION.
5456 022216 100000 000000 1$: .WORD 100000,0 ;FSRC OPERAND.
5457 022222 144000 000000 2$: .WORD 144000,0 ;EXPECTED RESULT.
5458 022226 143600 000000 3$: .WORD 143600,0 ;ANTICIPATED ERRONEOUS RESULT.
5459 022232 000007 4$: 7 ;FPS BEFORE EXECUTION.
5460 022234 000010 10 ;FPS AFTER EXECUTION.
5461 022236 177777 -1 ;ANTICIPATED ERRONEOUS FPS.
5462 022240 104001 5$: ERROR 1 ;ST 630 RH*R14+1
5463 022242 000401 BR 6$
5464 022244 104001 ERROR 1 ;REPORT FPS INCORRECT.
5465 022246 6$:
5466
5467 ;OPERAND=100000,0 FL=1
5468 022246 KKC20:
5469 022246 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
5470 022250 004737 022306 JSR PC,@#LDCFSUB ;GO EXECUTE THE INSTRUCTION.
5471 022254 100000 000000 1$: .WORD 100000,0 ;FSRC OPERAND.
5472 022260 150000 000000 2$: .WORD 150000,0 ;EXPECTED RESULT.
5473 022264 147600 000000 3$: .WORD 147600,0 ;ANTICIPATED ERRONEOUS RESULT.
5474 022270 000107 107 ;FPS BEFORE EXECUTION.
5475 022272 000110 110 ;FPS AFTER EXECUTION.
5476 022274 177777 -1 ;ANTICIPATED ERRONEOUS FPS.
5477 022276 104001 5$: ERROR 1 ;REPORT RESULT INCORRECT.
5478 022300 000401 BR 6$
5479 022302 104001 ERROR 1 ;REPORT FPS INCORRECT.
5480 022304 000506 6$: BR KKCDONE
5481
5482 ;THIS SUBROUTINE, LDCFSUB, IS USED TO SET UP THE OPERANDS, EXECUTE
```

5483
 5484
 5485
 5486
 5487
 5488
 5489
 5490
 5491
 5492
 5493
 5494
 5495
 5496
 5497
 5498
 5499
 5500
 5501
 5502
 5503
 5504
 5505
 5506
 5507
 5508
 5509
 5510
 5511
 5512
 5513
 5514
 5515
 5516
 5517
 5518
 5519
 5520
 5521
 5522
 5523
 5524
 5525
 5526
 5527
 5528
 5529
 5530
 5531
 5532
 5533
 5534
 5535
 5536
 5537
 5538

:THE LDCIF OR LDCLF INSTRUCTION AND CHECK THE RESULTS. A CALL
 :TO IT IS MADE THUS:

```

JSR    PC,@#LDCFSUB
ACARG: .WORD  X,X           ;AC 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
ERFPS: .WORD  X             ;ERROR FPS
ERR1:  ERROR  1;DATA ERROR
        BR     CONT
ERR2:  ERROR  1;FPS ERROR
CONT:  ;RETURN ADDRESS
  
```

:THE OPERANDS ARE SET UP (USING ACO AS THE ACCUMULATOR). THEN
 :THE LDCIF OR LDCLF INSTRUCTION IS EXECUTED.
 :THE RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
 :COMPARED WITH FPSA IF THIS TOO IS CORRECT LDCFSUB RETURNS CONTROL
 :TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD LDCFSUB WILL
 :COMPARE IT TO ERROR FPS. IF THIS MATCHES THEN LDCFSUB WILL RETURN
 :TO THE ERROR CALL AT ERR2, OTHERWISE LDCFSUB ITSELF
 :REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE
 :LDCIF OR LDCLF 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 LDCFSUB
 :WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR1. OTHERWISE THE
 :RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND LDCFSUB
 :REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.

```

LDCFSUB:  MOV     (SP)+,R1      ;GET A POINTER TO THE ARGUMENTS.
          MOV     14(R1),R0    ;SET THE FPS.
          LDFPS  R0
          MOV     #1$,@#STMP2
          MOV     R1,R0
1$:       LDCIF  (R0),ACO      ;TEST INSTRUCTION LDCIF OR LDCLF.
          STFPS  R4           ;GET FPS.
          MOV     #LDCT,R0     ;GET THE RESULT.
          MOV     #200,R2
          LDFPS  R2
          STD    ACO,(R0)
          MOV     #LDCT,R2     ;SEE IF THE RESULT WAS CORRECT.
          MOV     R2,@#STMP4
          MOV     R1,@#STMP3
          MOV     R1,R3
          ADD    #4,R3
          MOV     R3,@#STMP5
          MOV     R4,@#STMP7
          MOV     16(R1),@#STMP10
          MOV     R1,R0
          ADD    #4,R0
          MOV     #2,R3
2$:       CMP    (R0)+,(R2)+
  
```

```

022306 012601
022310 016100 000014
022314 170100
022316 012737 022326 001236
022324 010100
022326 177010
022330 170204
022332 012700 022512
022336 012702 000200
022342 170102
022344 174010
022346 012702 022512
022352 010237 001242
022356 010137 001240
022362 010103
022364 062703 000004
022370 010337 001244
022374 010437 001250
022400 016137 000016 001252
022406 010100
022410 062700 000004
022414 012703 000002
022420 022022
  
```

```
5539 022422 001006          BNE    10$          ;BR IF INCORRECT.
5540 022424 077303          SOB    R3,2$
5541
5542 022426 026104 000016          CMP    16(R1),R4          ;SEE IF THE FPS WAS CORRECT.
5543 022432 001020          BNE    15$          ;BR IF INCORRECT.
5544 022434 000161 000030 3$:    JMP    30(R1)          ;RETURN.
5545
5546          ;RESULT IN CORRECT SO SEE IF THE FAILURE WAS ANTICIPATED.
5547 022440 012702 022512 10$:   MOV    #LDCT,R2
5548 022444 010100          MOV    R1,R0
5549 022446 062700 000010          ADD    #10,R0
5550 022452 012703 000002          MOV    #2,R3
5551 022456 022022 11$:   CMP    (R0)+,(R2)+
5552 022460 001003          BNE    13$
5553 022462 077303          SOB    R3,11$
5554 022464 000161 000022          JMP    22(R1)
5555
5556          ;THE FAILURE WAS NOT ANTICIPATED SO REPORT THE ERROR HERE.
5557 022470 13$:
5558
5559 022470 104001 14$:   ERROR  1          ;BAD RES
5560 022472 000760          BR     3$
5561
5562
5563          ;THE FPS WAS INCORRECT SO SEE IF IT WAS ANTICIPATED.
5564 022474 026104 000020 15$:   CMP    20(R1),R4
5565 022500 001002          BNE    16$
5566 022502 000161 000026          JMP    26(R1)
5567
5568          ;FPS ERROR NOT ANTICIPATED SO REPORT IT HERE.
5569 022506 16$:
5570 022506 104001 17$:   ERROR  1          ;BAD FPS
5571 022510 000751          BR     3$
5572
5573          ;DATA BUFFER:
5574 022512 000000 000000 000000 LDCT:  .WORD  0,0,0,0
5575 022520 000000
5576
5577 022522 KKCDONE:
5578 022522 104413          RSETUP          ;GO INITIALIZE THE FPS AND STACK; AND
5579          ;SEE IF THE USER HAS EXPRESSED
5580          ;THE DESIRE TO CHANGE THE SOFTWARE
5581          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
5582          ;THE USER TYPED CONTROL G?).
5583
5584
5585          ;*****
5586          ;*TEST 57          LDCID AND LDCLD TEST
5587          ;*
5588          ;* THIS IS A TEST OF LDCID AND LDCLD
5589          ;*
5590          ;*****
5591 022524 000004 TST57:  SCOPE
5592          ;OPERAND=0          FL=0,          FD=1
5593 022526
5594 022526 104414          LLC1:          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
```


B 9

```

5651 022742 000217          4$: 217          ;FPS BEFORE EXECUTION.
5652 022744 000200          : 200          ;FPS AFTER EXECUTION.
5653 022746 177777          : -1          ;ANTICIPATED ERRONEOUS FPS.
5654 022750 104001          5$: ERROR 1      ;ST 107 BAD CONST
5655 022752 000401          BR 6$
5656 022754 104001          ERROR 1        ;REPORT FPS INCORRECT.
5657 022756
5658
5659          ;OPERAND=-40000 FL=0   FD=1
5660 022756          LLC5:
5661 022756 104414          LPERR
5662 022760 004737 023324  JSR PC,@#LDCDSUB ;SET UP THE LOOP ON ERROR ADDRESS.
5663 022764 140000 000000  1$: .WORD -40000,0 ;GO EXECUTE THE INSTRUCTION.
5664 022770 143600 000000 000000 2$: .WORD 143600,0,0,0 ;FSRC OPERAND.
5665 022776 000000          : .WORD 143600,0,0,0 ;EXPECTED RESULT.
5666 023000 043600 000000 000000 3$: .WORD 43600,0,0,0 ;ANTICIPATED ERRONEOUS RESULT.
5667 023006 000000
5668 023010 000200          4$: 200          ;FPS BEFORE EXECUTION.
5669 023012 000210          : 210          ;FPS AFTER EXECUTION.
5670 023014 177777          : -1          ;ANTICIPATED ERRONEOUS FPS.
5671 023016 104001          5$: ERROR 1      ;(SET SIGN) ST 176
5672 023020 000401          BR 6$
5673 023022 104001          ERROR 1        ;REPORT FPS INCORRECT.
5674 023024
5675
5676          ;OPERAND=40000,0     FL=1   FD=1
5677 023024          LLC6:
5678 023024 104414          LPERR
5679 023026 004737 023324  JSR PC,@#LDCDSUB ;SET UP THE LOOP ON ERROR ADDRESS.
5680 023032 040000 000000  1$: .WORD 40000,0 ;GO EXECUTE THE INSTRUCTION.
5681 023036 047600 000000 000000 2$: .WORD 47600,0,0,0 ;FSRC OPERAND.
5682 023044 000000          : .WORD 47600,0,0,0 ;EXPECTED RESULT.
5683 023046 043600 000000 000000 3$: .WORD 43600,0,0,0 ;ANTICIPATED ERRONEOUS RESULT.
5684 023054 000000
5685 023056 000317          : 317          ;FPS BEFORE EXECUTION.
5686 023060 000300          : 300          ;FPS AFTER EXECUTION.
5687 023062 177777          : -1          ;ANTICIPATED ERRONEOUS FPS.
5688 023064 104001          5$: ERROR 1      ;ST 107 BAD CONS
5689 023066 000401          BR 6$
5690 023070 104274          ERROR 274     ;REPORT FPS INCORRECT.
5691 023072
5692
5693          ;OPERAND 0,1     FL 1   FD=1
5694 023072          LLC7:
5695 023072 104414          LPERR
5696 023074 004737 023324  JSR PC,@#LDCDSUB ;SET UP THE LOOP ON ERROR ADDRESS.
5697 023100 000000 000001  1$: .WORD 0,1 ;GO EXECUTE THE INSTRUCTION.
5698 023104 040200 000000 000000 2$: .WORD 40200,0,0,0 ;FSRC OPERAND.
5699 023112 000000          : .WORD 40200,0,0,0 ;EXPECTED RESULT.
5700 023114 034200 000000 000000 3$: .WORD 34200,0,0,0 ;ANTICIPATED ERRONEOUS RESULT.
5701 023122 000000
5702 023124 000300          4$: 300          ;FPS BEFORE EXECUTION.
5703 023126 000300          : 300          ;FPS AFTER EXECUTION.
5704 023130 177777          : -1          ;ANTICIPATED ERRONEOUS FPS.
5705 023132 104001          5$: ERROR 1      ;REPORT FPS INCORRECT.
5706 023134 000401          BR 6$

```

```

5707 023136 104001          ERROR 1          ;REPORT FPS INCORRECT.
5708 023140          6$:
5709
5710          ;OPERAND=77777,177777 FL=1 FD-1
5711 023140          LLC8:
5712 023140 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
5713 023142 004737 023324  JSR PC,LDCDSUB ;GO EXECUTE THE INSTRUCTION.
5714 023146 077777 177777 1$: .WORD 77777,177777 ;FSRC OPERAND.
5715 023152 047777 177777 177000 2$: .WORD 47777,177777,177000,0 ;EXPECTED RESULT.
5716 023160 000000
5717 023162 177777 177777 177777 3$: .WORD -1,-1,-1,-1 ;ANTICIPATED ERRONEOUS RESULT.
5718 023170 177777
5719 023172 000317 4$: 317 ;FPS BEFORE EXECUTION.
5720 023174 000300 300 ;FPS AFTER EXECUTION.
5721 023176 177777 -1 ;ANTICIPATED ERRONEOUS FPS.
5722 023200 104001 5$: ERROR 1 ;REPORT RESULT INCORRECT.
5723 023202 000401 BR 6$
5724 023204 104001 ERROR 1 ;REPORT FPS INCORRECT.
5725 023206 6$:
5726
5727          ;OPERAND--PATTERN FL=1 FD 1
5728
5729 023206          LLC9:
5730 023206 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
5731 023210 004767 000110  JSR PC,LDCDSUB ;GO EXECUTE THE INSTRUCTION.
5732 023214 177777 177526 1$: .WORD -1,-252 ;FSRC OPERAND.
5733 023220 142052 000000 000000 2$: .WORD 142052,0,0,0 ;EXPECTED RESULT.
5734 023226 000000
5735 023230 136052 000000 000000 3$: .WORD 136052,0,0,0 ;ANTICIPATED ERRONEOUS RESULT.
5736 023236 000000
5737 023240 000307 4$: 307 ;FPS BEFORE EXECUTION.
5738 023242 000310 310 ;FPS AFTER EXECUTION.
5739 023244 177777 -1 ;ANTICIPATED ERRONEOUS FPS.
5740 023246 104001 5$: ERROR 1 ;REPORT RESULT INCORRECT.
5741 023250 000401 BR 6$
5742 023252 104001 ERROR 1 ;REPORT FPS INCORRECT.
5743 023254 6$:
5744
5745          ;OPERAND-PATTERN FL=1 FD=1 FT=1
5746 023254          LLC10:
5747 023254 104414          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
5748 023256 004767 000042  JSR PC,LDCDSUB ;GO EXECUTE THE INSTRUCTION.
5749 023262 012345 067012 1$: .WORD 12345,67012 ;FSRC OPERAND.
5750 023266 047247 025560 050000 2$: .WORD 47247,025560,050000,0 ;EXPECTED RESULT.
5751 023274 000000
5752 023276 177777 177777 177777 3$: .WORD -1,-1,-1,-1 ;ANTICIPATED ERRONEOUS RESULT.
5753 023304 177777
5754 023306 000352 4$: 352 ;FPS BEFORE EXECUTION.
5755 023310 000340 340 ;FPS AFTER EXECUTION.
5756 023312 177777 -1 ;ANTICIPATED ERRONEOUS FPS.
5757 023314 104001 5$: ERROR 1 ;REPORT RESULT INCORRECT.
5758 023316 000401 BR 6$
5759 023320 104001 ERROR 1 ;REPORT FPS INCORRECT.
5760 023322 000502 6$: BR LLCDONE
5761
5762          ;THIS SUBROUTINE, LDCDSUB, IS USED TO SET UP THE OPERANDS, EXECUTE
    
```

5763
 5764
 5765
 5766
 5767
 5768
 5769
 5770
 5771
 5772
 5773
 5774
 5775
 5776
 5777
 5778
 5779
 5780
 5781
 5782
 5783
 5784
 5785
 5786
 5787
 5788
 5789
 5790
 5791
 5792
 5793
 5794
 5795
 5796
 5797
 5798
 5799
 5800
 5801
 5802
 5803
 5804
 5805
 5806
 5807
 5808
 5809
 5810
 5811
 5812
 5813
 5814
 5815
 5816
 5817
 5818

:THE LDCID OR LDCLD INSTRUCTION AND CHECK THE RESULTS. A CALL
 :TO IT IS MADE THUS:

```

JSR      PC,@#LDCDSUB
ACARG:   .WORD  X,X           ;AC OPERAND
RES:     .WORD  X,X,X,X      ;EXPECTED RESULT
ERRES:   .WORD  X,X,X,X      ;ERROR RESULT
FPSB:    .WORD  X           ;FPS BEFORE EXECUTION
FPSA:    .WORD  X           ;FPS AFTER EXECUTION
ERFPS:   .WORD  X           ;ERROR FPS.
ERR1:    ERROR  1;DATA ERROR.
          BR     CONT
ERR2:    ERROR  1;FPS ERROR.
CONT:                                         ;RETURN ADDRESS
  
```

:THE OPERANDS ARE SET UP (USING ACO AS THE ACCUMULATOR). THEN
 :THE LDCID OR LDCLD INSTRUCTION IS EXECUTED.
 :THE RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
 :COMPARED WITH FPSA IF THIS TOO IS CORRECT LDCDSUB RETURNS CONTROL
 :TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD LDCDSUB
 :COMPARE IT TO ERROR FPS. IF THIS MATCHES THEN LDCDSUB WILL RETURN
 :TO THE ERROR CALL AT ERR2, OTHERWISE LDCDSUB ITSELF
 :REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE
 :LDCID OR LDCLD 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 LDCDSUB
 :WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR1. OTHERWISE THE
 :RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND LDCDSUB WILL
 :REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.

```

LDCDSUB:  MOV      (SP)+,R1      ;GET A POINTER TO THE ARGUMENTS.
          MOV      24(R1),R0     ;SET THE FPS.
          LDFPS   R0
          MOV      #1$,@#STMP2
          MOV      R1,R0
1$:       LDCID   (R0),ACO       ;TEST INSTRUCTION, LDCID OR LDCLD.

          STFPS   R4             ;GET FPS.
          MOV      #LDCT,R0      ;GET THE RESULT.
          MOV      #200,R2
          LDFPS   R2
          STD     ACO,(R0)

;SEE IF THE RESULT IS CORRECT.
          MOV      #LDCT,R2
          MOV      R2,@#STMP4
          MOV      R1,@#STMP3
          MOV      R1,R3
          ADD     #4,R3
          MOV      R3,@#STMP5
          MOV      R4,@#STMP7
          MOV      26(R1),@#STMP10
          MOV      R1,R0
          ADD     #4,R0
          MOV      #2,R3
2$:       CMP     (R0)+,(R2)+
  
```

```

023324 012601
023326 016100 000024
023332 170100
023334 012737 023344 001236
023342 010100
023344 177010

023346 170204
023350 012700 022512
023354 012702 000200
023360 170102
023362 174010

023364 012702 022512
023370 010237 001242
023374 010137 001240
023400 010103
023402 062703 000004
023406 010337 001244
023412 010437 001250
023416 016137 000026 001252
023424 010100
023426 062700 000004
023432 012703 000002
023436 022022
  
```

```
5819 023440 001006      BNE      10$          ;BR IF INCORRECT.
5820 023442 077303      SOB      R3,2$
5821
5822 023444 026104 000026      CMP      26(R1),R4          ;IS THE FPS CORRECT?
5823 023450 001020      BNE      15$          ;BR IF INCORRECT.
5824 023452 000161 000040      3$:     JMP      40(R1)      ;RETURN.
5825
5826      ;THE RESULT WAS INCORRECT SO SEE IF THE ERROR WAS ANTICIPATED.
5827 023456 012702 022512      10$:     MOV      #LDCT,R2
5828 023462 010100      MOV      R1,R0
5829 023464 062700 000014      ADD      #14,R0
5830 023470 012703 000002      MOV      #2,R3
5831 023474 022022      11$:     CMP      (R0)+,(R2)+
5832 023476 001003      BNE      13$
5833 023500 077303      SOB      R3,11$
5834 023502 000161 000032      JMP      32(R1)
5835 023506
5836      ;ERROR NOT ANTICIPATED SO REPORT RESULT INCORRECT HERE.
5837 023506 104001      14$:     ERROR    1          ;BAD RES
5838 023510 000760      BR      3$
5839
5840      ;THE FPS WAS INCORRECT. SEE IF FAILURE WAS ANTICIPATED.
5841 023512 026104 000030      15$:     CMP      30(R1),R4
5842 023516 001002      BNE      16$
5843 023520 000161 000036      JMP      36(R1)
5844
5845      ;FPS ERROR WAS NOT ANTICIPATED SO REPORT FAILURE HERE.
5846      16$:
5847 023524 104001      17$:     ERROR    1          ;BAD FPS
5848 023526 000751      BR      3$
5849
5850      LLCDONE:
5851 023530 104413      RSETUP          ;GO INITIALIZE THE FPS AND STACK; AND
5852                                     ;SEE IF THE USER HAS EXPRESSED
5853                                     ;THE DESIRE TO CHANGE THE SOFTWARE
5854                                     ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
5855                                     ;THE USER TYPED CONTROL G?).
5856
5857
5858      ;*****
5859      ;*TEST 60      LDEXP TEST
5860      ;*
5861      ;* THIS IS A TEST OF THE LDEXP INST
5862      ;* A SUBROUTINE IS USED TO SET UP
5863      ;* OPERANDS, EXECUTE THE LDEXP INST AND
5864      ;* CHECK THE RESULTS.
5865      ;*
5866      ;*****
5867 023532 000004      TST60:  SCOPE
5868
5869      ; NON-ZERO RES. VALID EXPON=210 (EXCESS 200)=10
5870      ;MMCI:
5871 023534 104414      LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
5872 023536 004767 001334      JSR      PC,LDXSUB      ;GO EXECUTE THE INSTRUCTION.
5873 023542 012345 067012 034567 1$:     .WORD    12345,67012,34567,012345      ;ACO OPERAND.
5874 023550 012345
```

```

5875 023552 000010
5876 023554 042145 067012 034567 2$: .WORD 10 ;EXPONENT OPERAND.
5877 023562 012345 3$: .WORD 42145,67012,34567,012345 ;EXPECTED RESULT.
5878 023564 002145 067012 034567 4$: .WORD 2145,67012,34567,012345 ;ANTICIPATED ERRONEOUS RESULT.
5879 023572 012345
5880 023574 047217 5$: 47217 ;FPS BEFORE EXECUTION.
5881 023576 047200 47200 ;FPS AFTER EXECUTION.
5882 023600 147200 147200 ;ANTICIPATED ERRONEOUS FPS.
5883 023602 177777 -1 ;EXPECTED FEC.
5884 023604 104001 6$: ERROR 1 ;E12+E12+200 BAD
5885 023606 000400 BR 7$ ;ST 624
5886 023610 104001 7$: ERROR 1 ;REPORT FPS INCORRECT.
5887
5888 ;NON-ZERO RES NEG. ;ST 625 INTO 304
5889 023612 ;NON-ZERO
5890 023612 104414 MMC2:
5891 023614 004737 025076 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
5892 023620 123456 070123 045670 1$: JSR PC,@#LDXSUB ;EXPON=377
5893 023626 123456 .WORD 123456,70123,45670,123456 ;ACO OPERAND.
5894 023630 000177 2$: .WORD 177 ;EXPONENT OPERAND.
5895 023632 177656 070123 045670 3$: .WORD 177656,70123,45670,123456 ;EXPECTED RESULT.
5896 023640 123456
5897 023642 137656 070123 045670 4$: .WORD 137656,70123,45670,123456 ;ANTICIPATED ERRONEOUS RESULT.
5898 023650 123456
5899 023652 047207 5$: 47207 ;FPS BEFORE EXECUTION.
5900 023654 047210 47210 ;FPS AFTER EXECUTION.
5901 023656 147210 147210 ;ANTICIPATED ERRONEOUS FPS.
5902 023660 177777 -1 ;EXPECTED FEC.
5903 023662 104001 6$: ERROR 1 ;REPORT RESULT INCORRECT.
5904 023664 000401 BR 7$
5905 023666 104001 7$: ERROR 1 ;REPORT FPS INCORRECT.
5906 023670
5907
5908 ;NON-ZERO RES, EXP=256=(56)REAL
5909 023670 MMC3:
5910 023670 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
5911 023672 004737 025076 JSR PC,@#LDXSUB ;GO EXECUTE THE INSTRUCTION.
5912 023676 073261 057645 043323 1$: .WORD 73261,057645,43323,101760 ;ACO OPERAND.
5913 023704 101760
5914 023706 000056 2$: .WORD 56 ;EXPONENT OPERAND.
5915 023710 053461 057645 043323 3$: .WORD 53461,057645,43323,101760 ;EXPECTED RESULT.
5916 023716 101760
5917 023720 177777 177777 177777 4$: .WORD -1,-1,-1,-1 ;ANTICIPATED ERRONEOUS RESULT.
5918 023726 177777
5919 023730 047200 5$: 47200 ;FPS BEFORE EXECUTION.
5920 023732 047200 47200 ;FPS AFTER EXECUTION.
5921 023734 147200 147200 ;ANTICIPATED ERRONEOUS FPS.
5922 023736 177777 -1 ;EXPECTED FEC.
5923 023740 104001 6$: ERROR 1 ;REPORT RESULT INCORRECT.
5924 023742 000401 BR 7$
5925 023744 104001 7$: ERROR 1 ;REPORT FPS INCORRECT.
5926 023746
5927
5928 ;EXP=27 (EXCESS 200)=-151 (OCT)
5929 023746 MMC4:
5930 023746 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
    
```

```

5931 023750 004737 025076 JSR PC,@#LDXSUB ;GO EXECUTE THE INSTRUCTION.
5932 023754 012223 024252 062720 1$: .WORD 12223,24252,62720,21222 ;ACO OPERAND.
5933 023762 021222
5934 023764 177627 2$: .WORD -151 ;EXPONENT OPERAND.
5935 023766 005623 024252 062720 3$: .WORD 5623,24252,62720,21222 ;EXPECTED RESULT.
5936 023774 021222
5937 023776 177777 177777 4$: .WORD -1,-1,-1,-1 ;ANTICIPATED ERRONEOUS RESULT.
5938 024004 177777 5$: 47200 ;FPS BEFORE EXECUTION.
5939 024006 047200 47200 ;FPS AFTER EXECUTION.
5940 024010 047200 147200 ;ANTICIPATED ERRONEOUS FPS.
5941 024012 147200 -1 ;EXPECTED FEC.
5942 024014 177777 6$: ERROR 1 ;REPORT RESULT INCORRECT.
5943 024016 104001 BR 7$
5944 024020 000401 ERROR 1 ;(BUT EZBT) ST 544 TO 504 INTO 704 0 (BUT EXBT) ST 704 1
5945 024022 104001
5946 024024 7$:
5947
5948 ;EXP=0 (EXCESS 200)--200 (OCT), POSITIVE FRAC
5949 ; FIV=1
5950 024024 MMC5:
5951 024024 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
5952 024026 004737 025076 JSR PC,@#LDXSUB ;GO EXECUTE THE INSTRUCTION.
5953 024032 030131 032334 035363 1$: .WORD 30131,32334,35363,73031 ;ACO OPERAND.
5954 024040 073031
5955 024042 177600 2$: .WORD -200 ;EXPONENT OPERAND.
5956 024044 000131 032334 035363 3$: .WORD 00131,32334,35363,73031 ;EXPECTED RESULT.
5957 024052 073031
5958 024054 000000 000000 000000 4$: .WORD 0,0,0,0 ;ANTICIPATED ERRONEOUS RESULT.
5959 024062 000000 5$: 42200 ;FPS BEFORE EXECUTION.
5960 024064 042200 142204 ;FPS AFTER EXECUTION.
5961 024066 142204 42202 ;ANTICIPATED ERRONEOUS FPS.
5962 024070 042202 12 ;EXPECTED FEC.
5963 024072 000012 6$: ERROR 1 ;(BUT EXBT) ST 704 TO 64 INST 264
5964 024074 104001 BR 7$
5965 024076 000401 ERROR 1 ;(BUT FIU) ST 264 X
5966 024100 104001 7$:
5967 024102
5968
5969 ;EXP=0 (EXCESS 200)--200 (OCT), NEG FRACT,FIU=1
5970 024102 MMC6:
5971 024102 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
5972 024104 004737 025076 JSR PC,@#LDXSUB ;GO EXECUTE THE INSTRUCTION.
5973 024110 140414 024344 045464 1$: .WORD 140414,24344,45464,74045 ;ACO OPERAND.
5974 024116 074045
5975 024120 177600 2$: .WORD -200 ;EXPONENT OPERAND.
5976 024122 100014 024344 045464 3$: .WORD 100014,24344,45464,74045 ;-0 ;EXPECTED RESULT.
5977 024130 074045
5978 024132 000000 000000 000000 4$: .WORD 0,0,0,0 ;ANTICIPATED ERRONEOUS RESULT.
5979 024140 000000 5$: 42200 ;FPS BEFORE EXECUTION.
5980 024142 042200 142214 ;FPS AFTER EXECUTION.
5981 024144 142214 42214 ;ANTICIPATED ERRONEOUS FPS.
5982 024146 042214 12 ;EXPECTED FEC.
5983 024150 000012 6$: ERROR 1 ;REPORT RESULT INCORRECT.
5984 024152 104001 BR 7$
5985 024154 000401 ERROR 1 ;REPORT FPS INCORRECT.
5986 024156 104001
    
```

```
5987 024160 7$:  
5988  
5989 ;EXP=0 (EXCESS 200)=-200 (OCT),POS FRAC, FIU=0  
5990  
5991 024160 MMC7:  
5992 024160 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.  
5993 024162 004737 025076 JSR PC,@#LDXSUB ;GO EXECUTE THE INSTRUCTION.  
5994 024166 051525 035455 005675 1$: .WORD 51525,35455,5675,05152 ;ACO OPERAND.  
5995 024174 005152  
5996 024176 177600 2$: .WORD -200 ;EXPONENT OPERAND.  
5997 024200 000000 000000 000000 3$: .WORD 0,0,0,0 ;EXPECTED RESULT.  
5998 024206 000000  
5999 024210 000125 035455 005675 4$: .WORD 00125,35455,5675,05152 ;ANTICIPATED ERRONEOUS RESULT.  
6000 024216 005152  
6001 024220 045200 45200 ;FPS BEFORE EXECUTION.  
6002 024222 045204 45204 ;FPS AFTER EXECUTION.  
6003 024224 145204 145204 ;ANTICIPATED ERRONEOUS FPS.  
6004 024226 177777 -1 ;EXPECTED FEC.  
6005 024230 104001 6$: ERROR 1 ;(BUT FIU) ST 264 X ;REPORT RESULT INCORRECT  
6006 024232 000401 BR 7$  
6007 024234 104001 ERROR 1 ;REPORT FPS INCORRECT.  
6008 024236 7$:  
6009  
6010 ;EXP=-1405 (EXCESS 200)=-1605 (OCT), FIU=1  
6011 024236 MMC8:  
6012 024236 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.  
6013 024240 004737 025076 JSR PC,@#LDXSUB ;GO EXECUTE THE INSTRUCTION.  
6014 024244 061626 062636 046566 1$: .WORD 61626,62636,46566,67606 ;ACO OPERAND.  
6015 024252 067606  
6016 024254 176173 2$: .WORD -1605 ;EXPONENT OPERAND.  
6017 024256 076626 062636 046566 3$: .WORD 76626,62636,46566,67606 ;EXPECTED RESULT.  
6018 024264 067606  
6019 024266 000000 000000 000000 4$: .WORD 0,0,0,0 ;ANTICIPATED ERRONEOUS RESULT.  
6020 024274 000000  
6021 024276 042200 5$: 42200 ;FPS BEFORE EXECUTION.  
6022 024300 142200 142200 ;FPS AFTER EXECUTION.  
6023 024302 042204 42204 ;ANTICIPATED ERRONEOUS FPS.  
6024 024304 000012 12 ;EXPECTED FEC.  
6025 024306 104001 6$: ERROR 1 ;(BUT EZBT) ST 544 TO 704 INTO 504  
6026 024310 000401 BR 7$  
6027 024312 104001 ERROR 1 ;REPORT FPS INCORRECT.  
6028 024314 7$:  
6029 ;EXP=-17416 (EXCESS 200)=-17616 (OCT), FIU=0  
6030 024314 MMC9:  
6031 024314 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.  
6032 024316 004737 025076 JSR PC,@#LDXSUB ;GO EXECUTE THE INSTRUCTION.  
6033 024322 071727 037475 076777 1$: .WORD 71727,37475,76777,17273 ;ACO OPERAND.  
6034 024330 017273  
6035 024332 160162 2$: .WORD -17616 ;EXPONENT OPERAND.  
6036 024334 000000 000000 000000 3$: .WORD 0,0,0,0 ;EXPECTED RESULT.  
6037 024342 000000  
6038 024344 074527 037475 076777 4$: .WORD 74527,37475,76777,17273 ;ANTICIPATED ERRONEOUS RESULT.  
6039 024352 017273  
6040 024354 045200 5$: 45200 ;FPS BEFORE EXECUTION.  
6041 024356 045204 45204 ;FPS AFTER EXECUTION.  
6042 024360 145200 145200 ;ANTICIPATED ERRONEOUS FPS.
```



```
6043 024362 177777 -1 ;EXPECTED FEC.
6044 024364 104001 6$: ERROR 1 ;(BUT FIU) ST 504
6045 024366 000401 BR 7$
6046 024370 104001 ERROR 1 ;REPORT FPS INCORRECT.
6047 024372 7$:
6048
6049 ;EXP=-1601 (EXCESS 200)=-2001 (OCT), FIU=1
6050 024372 MMC10:
6051 024372 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
6052 024374 004737 025076 JSR PC,@#LDXSUB ;GO EXECUTE THE INSTRUCTION.
6053 024400 001020 030405 006070 1$: .WORD 01020,30405,06070,00102 ;ACO OPERAND.
6054 024406 000102
6055 024410 175777 2$: .WORD -2001 ;EXPONENT OPERAND.
6056 024412 037620 030405 006070 3$: .WORD 37620,30405,06070,00102 ;EXPECTED RESULT.
6057 024420 000102
6058 024422 000000 000000 000000 4$: .WORD 0,0,0,0 ;ANTICIPATED ERRONEOUS RESULT.
6059 024430 000000
6060 024432 042200 5$: 42200 ;FPS BEFORE EXECUTION.
6061 024434 142200 142200 ;FPS AFTER EXECUTION.
6062 024436 042204 42204 ;ANTICIPATED ERRONEOUS FPS.
6063 024440 000012 12 ;EXPECTED FEC.
6064 024442 104001 6$: ERROR 1 ;(BUT FIU) ST 504
6065 024444 000401 BR 7$
6066 024446 104001 ERROR 1 ;REPORT FPS INCORRECT.
6067 024450 7$:
6068
6069 ;EXP=1206 (EXCESS 200)=1006 (OCT) FIV =1
6070 024450 MMC11:
6071 024450 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
6072 024452 004737 025076 JSR PC,@#LDXSUB ;GO EXECUTE THE INSTRUCTION.
6073 024456 012131 014151 016171 1$: .WORD 12131,14151,16171,10111 ;ACO OPERAND.
6074 024464 010111
6075 024466 001006 2$: .WORD 1006 ;EXPONENT OPERAND.
6076 024470 041531 014151 016171 3$: .WORD 41531,14151,16171,10111 ;EXPECTED RESULT.
6077 024476 010111
6078 024500 000000 000000 000000 4$: .WORD 0,0,0,0 ;ANTICIPATED ERRONEOUS RESULT.
6079 024506 000000
6080 024510 041200 5$: 41200 ;FPS BEFORE EXECUTION.
6081 024512 141202 141202 ;FPS AFTER EXECUTION.
6082 024514 041204 41204 ;ANTICIPATED ERRONEOUS FPS.
6083 024516 000010 10 ;EXPECTED FEC.
6084 024520 104001 6$: ERROR 1 ;(BUT FIV) ST 104
6085 024522 000401 BR 7$
6086 024524 104001 ERROR 1 ;REPORT FPS INCORRECT.
6087 024526 7$:
6088
6089 ;EXP-16315 (EXCESS 200)=16115 (OCT) FIV=0
6090 024526 MMC12:
6091 024526 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
6092 024530 004737 025076 JSR PC,@#LDXSUB ;GO EXECUTE THE INSTRUCTION.
6093 024534 027262 025242 023222 1$: .WORD 27262,25242,23222,21202 ;ACO OPERAND.
6094 024542 021202
6095 024544 016115 2$: .WORD 16115 ;EXPONENT OPERAND.
6096 024546 000000 000000 000000 3$: .WORD 0,0,0,0 ;EXPECTED RESULT.
6097 024554 000000
6098 024556 063262 025242 023222 4$: .WORD 63262,25242,23222,21202 ;ANTICIPATED ERRONEOUS RESULT.
```

6099 024564 021202
6100 024566 046200 5\$: 46200 ;FPS BEFORE EXECUTION.
6101 024570 046206 46206 ;FPS AFTER EXECUTION.
6102 024572 146202 146202 ;ANTICIPATED ERRONEOUS FPS.
6103 024574 177777 -1 ;EXPECTED FEC.
6104 024576 104001 6\$: ERROR 1 ;(BUT FIV) ST 104
6105 024600 000401 BR 7\$
6106 024602 104001 ERROR 1 ;REPORT FPS INCORRECT.
6107 024604 7\$:
6108
6109 ;EXP=11011 (EXCESS 200)=10611 (OCT) FIV=1
6110
6111 024604 MMC13:
6112 024604 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
6113 024606 004737 025076 JSR PC, @#LDXSUB ;GO EXECUTE THE INSTRUCTION.
6114 024612 030313 032333 034353 1\$: .WORD 30313,32333,34353,36373 ;ACO OPERAND.
6115 024620 036373
6116 024622 010611 2\$: .WORD 10611 ;EXPONENT OPERAND.
6117 024624 002313 032333 034353 3\$: .WORD 2313,32333,34353,36373 ;EXPECTED RESULT.
6118 024632 036373
6119 024634 000000 000000 000000 4\$: .WORD 0,0,0,0 ;ANTICIPATED ERRONEOUS RESULT.
6120 024642 000000
6121 024644 041200 5\$: 41200 ;FPS BEFORE EXECUTION.
6122 024646 141202 141202 ;FPS AFTER EXECUTION.
6123 024650 041204 41204 ;ANTICIPATED ERRONEOUS FPS.
6124 024652 000010 10 ;EXPECTED FEC.
6125 024654 104001 6\$: ERROR 1 ;(BUT FIV) ST 144
6126 024656 000401 BR 7\$
6127 024660 104001 ERROR 1 ;REPORT FPS INCORRECT.
6128 024662 7\$:
6129
6130 ;EXP=17123 (EXCESS 200)=16723 (OCT) FIV=0
6131
6132 024662 MMC14:
6133 024662 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
6134 024664 004737 025076 JSR PC, @#LDXSUB ;GO EXECUTE THE INSTRUCTION.
6135 024670 040414 042434 044454 1\$: .WORD 40414,42434,44454,46474 ;ACO OPERAND.
6136 024676 046474
6137 024700 016723 2\$: .WORD 16723 ;EXPONENT OPERAND.
6138 024702 000000 000000 000000 3\$: .WORD 0,0,0,0 ;EXPECTED RESULT.
6139 024710 000000
6140 024712 024614 042434 044454 4\$: .WORD 24614,42434,44454,46474 ;ANTICIPATED ERRONEOUS RESULT.
6141 024720 046474
6142 024722 046200 5\$: 46200 ;FPS BEFORE EXECUTION.
6143 024724 046206 46206 ;FPS AFTER EXECUTION.
6144 024726 146202 146202 ;ANTICIPATED ERRONEOUS FPS.
6145 024730 177777 -1 ;EXPECTED FEC.
6146 024732 104001 6\$: ERROR 1 ;(BUT FIV) ST 144
6147 024734 000401 BR 7\$
6148 024736 104001 ERROR 1 ;REPORT FPS INCORRECT.
6149 024740 7\$:
6150
6151 ;EXP= 254 (OCT)= 454 (EXCESS 200) FIV=1
6152
6153 024740 MMC15:
6154 024740 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.

```

6155 024742 004737 025076 JSR PC,@#LDXSUB ;GO EXECUTE THE INSTRUCTION.
6156 024746 050515 052535 054555 1$: .WORD 50515,52535,54555,56575 ;ACO OPERAND.
6157 024754 056575
6158 024756 000254 2$: .WORD 254 ;EXPONENT OPERAND.
6159 024760 013115 052535 054555 3$: .WORD 13115,52535,54555,56575 ;EXPECTED RESULT.
6160 024766 056575
6161 024770 000000 000000 000000 4$: .WORD 0,0,0,0 ;ANTICIPATED ERRONEOUS RESULT.
6162 024776 000000
6163 025000 041200 5$: 41200 ;FPS BEFORE EXECUTION.
6164 025002 141202 141202 ;FPS AFTER EXECUTION.
6165 025004 041204 41204 ;ANTICIPATED ERRONEOUS FPS.
6166 025006 000010 10 ;EXPECTED FEC.
6167 025010 104001 6$: ERROR 1 ;(BUT FIV) ST344
6168 025012 000401 BR 7$
6169 025014 104001 ERROR 1 ;REPORT FPS INCORRECT.
6170 025016 7$:
6171
6172 ;EXP= 313 (OCT)= 513(EXCESS 200) FIV=0
6173
6174 025016 MMC16:
6175 025016 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
6176 025020 004737 025076 JSR PC,@#LDXSUB ;GO EXECUTE THE INSTRUCTION.
6177 025024 060616 062636 064656 1$: .WORD 60616,62636,64656,66676 ;ACO OPERAND.
6178 025032 066676
6179 025034 000313 2$: .WORD 313 ;EXPONENT OPERAND.
6180 025036 000000 000000 000000 3$: .WORD 0,0,0,0 ;EXPECTED RESULT.
6181 025044 000000
6182 025046 022616 062636 064656 4$: .WORD 22616,62636,64656,66676 ;ANTICIPATED ERRONEOUS RESULT.
6183 025054 066676
6184 025056 046200 5$: 46200 ;FPS BEFORE EXECUTION.
6185 025060 046206 46206 ;FPS AFTER EXECUTION.
6186 025062 146202 146202 ;ANTICIPATED ERRONEOUS FPS.
6187 025064 177777 -1 ;EXPECTED FEC.
6188 025066 104001 6$: ERROR 1 ;(BUT FIV) ST 344
6189 025070 000401 BR 7$
6190 025072 104001 ERROR 1 ;REPORT FPS INCORRECT.
6191 025074 7$:
6192 025074 000540 BR MMCDONE
6193
6194
6195
6196
6197
6198
6199
6200
6201
6202
6203
6204
6205
6206
6207
6208
6209
6210
    
```

;THIS SUBROUTINE, LDXSUB, IS USED TO SET UP THE OPERANDS, EXECUTE
 ;THE LDEXP INSTRUCTION AND CHECK THE RESULTS. A CALL
 ;TO IT IS MADE THUS:

```

:
: JSR PC,@#LDXSUB
: ACARG: .WORD X,X,X,X ;AC OPERAND
: EXP: .WORD X ;EXPONENT
: RES: .WORD X,X,X,X ;EXPECTED RESULT
: ERRES: .WORD X,X,X,X ;ERROR RESULT
: FPSB: .WORD X ;FPS BEFORE EXECUTION
: FPSA: .WORD X ;FPS AFTER EXECUTION
: ERFPS: .WORD X ;ERROR FPS.
: FEC: .WORD X ;EXPECTED FEC
: ERR1: ERROR 1;DATA ERROR.
: BR CONT
: ERR2: ERROR 1;FPS ERROR.
: CONT: ;RETURN ADDRESS
    
```

```

6211
6212
6213
6214
6215
6216
6217
6218
6219
6220
6221
6222
6223
6224
6225
6226
6227 025076 012601
6228 025100 012700 000200
6229 025104 170100
6230 025106 010100
6231 025110 172410
6232 025112 012737 025134 001236
6233 025120 016100 000032
6234 025124 170100
6235 025126 010100
6236 025130 062700 000010
6237
6238 025134 176410
6239
6240 025136 170204
6241 025140 170305
6242 025142 012700 000200
6243 025146 170100
6244 025150 012700 025366
6245 025154 174010
6246 025156 010437 001250
6247 025162 016137 000034 001252
6248 025170 010537 001254
6249 025174 016137 000040 001256
6250 025202 010102
6251 025204 010237 001240
6252 025210 062702 000010
6253 025214 011237 001242
6254 025220 062702 000002
6255 025224 010237 001244
6256 025230 012737 025366 001246
6257 025236 012702 025366
6258 025242 010103
6259 025244 062703 000012
6260 025250 012700 000004
6261 025254 022223
6262 025256 001014
6263 025260 077003
6264 025262 020461 000034
6265 025266 001026
6266 025270 005761 000034

```

```

;THE OPERANDS ARE SET UP (USING ACO AS THE ACCUMULATOR). THEN
;THE LDEXP INSTRUCTION IS EXECUTED.
;THE RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
;COMPARED WITH FPSA IF THIS TOO IS CORRECT LDXSUB RETURNS CONTROL
;TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD LDXSUB
;COMPARE IT TO ERROR FPS. IF THIS MATCHES THEN LDXSUB WILL RETURN
;TO THE ERROR CALL AT ERR2, OTHERWISE LDXSUB ITSELF
;REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE
;LDEXP 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 LDXSUB
;WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR1. OTHERWISE THE
;RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND LDXSUB WILL
;REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.

LDXSUB: MOV (SP)+,R1 ;GET A POINTER TO THE ARGUMENTS.
MOV #200,R0 ;LOAD THE ACO OPERAND.
LDFPS R0
MOV R1,R0
LDD (R0),ACU
MOV #1$,@#STMP2
MOV 32(R1),R0 ;SET UP THE FPS.
LDFPS R0
MOV R1,R0
ADD #10,R0

1$: LDEXP (R0),ACU ;TEST INSTRUCTION.

STFPS R4 ;GET THE FPS.
STST R5 ;GET THE FEC.
MOV #200,R0 ;GET THE RESULT.
LDFPS R0
MOV #LDXT,R0
STD ACO,(R0)
MOV R4,@#STMP7
MOV 34(R1),@#STMP10
MOV R5,@#STMP11
MOV 40(R1),@#STMP12
MOV R1,R2
MOV R2,@#STMP3
ADD #10,R2
MOV (R2),@#STMP4
ADD #2,R2
MOV R2,@#STMP5
MOV #LDXT,@#STMP6
MOV #LDXT,R2 ;SEE IF THE RESULT WAS CORRECT.
MOV R1,R3
ADD #12,R3
MOV #4,R0

2$: CMP (R2)+,(R3)+ ;BRANCH IF NOT CORRECT.
BNE 10$
SOB R0,2$
CMP R4,34(R1) ;SEE IF THE FPS WAS CORRECT.
BNE 15$ ;BRANCH IF NOT CORRECT.
TST 34(R1)

```

```
6267 025274 100003          BPL      3$
6268 025276 020561 000040    CMP      R5,40(R1)      ;SEE IF THE FEC WAS CORRECT.
6269 025302 001027          BNE      20$            ;BRANCH IF NOT CORRECT.
6270
6271 025304 000161 000050    3$:     JMP      50(R1)      ;RETURN.
6272
6273          ;THE RESULT WAS INCORRECT SO SEE IF THE FAILURE WAS ANTICIPATED.
6274 025310 012702 025366    10$:     MOV      #LDXT,R2
6275 025314 010103          MOV      R1,R3
6276 025316 062703 000022    ADD      #22,R3
6277 025322 012700 000004    MOV      #4,R0
6278 025326 022223    11$:     CMP      (R2)+,(R3)+
6279 025330 001003          BNE      12$
6280 025332 077003          SOB      R0,11$
6281 025334 000161 000042    JMP      42(R1)
6282
6283          ;THE ERROR WAS NOT ANTICIPATED SO REPORT IT HERE.
6284 025340    12$:
6285 025340 104001    13$:     ERROR   1          ;BAD RES
6286 025342 000760    BR      3$
6287
6288          ;SEE IF THE FPS ERROR WAS ANTICIPATED.
6289 025344 026104 000036    15$:     CMP      36(R1),R4
6290 025350 001002          BNE      16$
6291 025352 000161 000046    JMP      46(R1)
6292
6293          ;THE FPS WAS NOT ANTICIPATED SO REPORT IT HERE.
6294 025356 104001    17$:     ERROR   1          ;BAD FPS
6295 025360 000751    BR      3$            ;BUT EZBTY8
6296                                ;ST 063
6297
6298 025362    20$:
6299          ;REPORT FEC INCORRECT.
6300 025362 104001    21$:     ERROR   1          ;BAD FEC
6301 025364 000747    BR      3$
6302
6303          ;DATA BUFFER:
6304 025366 000000 000000 000000 LDXT:    .WORD  0,0,0,0
6305 025374 000000
6306
6307 025376    MMCDONE:
6308 025376 104413    RSETUP          ;GO INITIALIZE THE FPS AND STACK; AND
6309                                ;SEE IF THE USER HAS EXPRESSED
6310                                ;THE DESIRE TO CHANGE THE SOFTWARE
6311                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
6312                                ;THE USER TYPED CONTROL G?).
6313
6314
6315
6316          ;*****
6317          ;*TEST 61      DESTINATION MODES, MODE 1 (FL=0), TEST
6318          ;*
6319          ;* THIS IS A TEST OF DESTINATION MODE 1 USING
6320          ;* THE STFPS INSTRUCTION
6321          ;*
6322          ;*****
```

```
6323 025400 000004 TST61: SCOPE
6324
6325
6326 025402 NNC1:
6327 025402 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
6328 025404 012700 025502 MOV #NNCTB0,R0 ;SET UP THE DATA BUFFER.
6329 025410 012701 000006 MOV #6,R1
6330 025414 012720 177777 1$: MOV #-1,(R0)+
6331 025420 077103 SOB R1,1$
6332 025422 012700 102345 MOV #102345,R0
6333 025426 012737 025450 001236 MOV #NNC2,@#STMP2
6334 025434 012737 025574 000004 MOV #NNC25,@#ERRVECT ;SET UP FOR TRAPS TO 4.
6335 025442 170100 LDFPS R0 ;SET UP FPS.
6336 025444 012700 025506 MOV #NNCTB1,R0
6337
6338 025450 170210 NNC2: STFPS (R0) ;TEST INSTRUCTION.
6339 025452 020027 025506 CMP R0,#NNCTB1 ;IS R0 CORRECT?
6340 025456 001017 BNE NNC10 ;BRANCH IF NOT CORRECT.
6341 025460 023727 J25506 102345 CMP @#NNCTB1,#102345 ;IS RESULT CORRECT?
6342 025466 001022 BNE NNC15 ;BRANCH IF NOT CORRECT.
6343 025470 023727 025510 177777 CMP @#NNCTB1+2,#-1 ;IS THE RESULT CORRECT?
6344 025476 001026 BNE NNC20 ;BRANCH IF NOT CORRECT.
6345 025500 000447 BR NNCDONE
6346
6347 ;TEST DATA BUFFER:
6348 025502 177777 177777 NNCTB0: .WORD -1,-1
6349 025506 177777 177777 177777 NNCTB1: .WORD -1,-1,-1,-1
6350 025514 177777
6351
6352 ;REPORT RO INCORRECT.
6353 025516 010037 001242 NNC10: MOV R0,@#STMP4
6354 025522 012737 025506 001240 MOV #NNCTB1,@#STMP3
6355 025530 104001 1$: ERROR 1 ;RO BAD (BUT
6356 025532 000432 BR NNCDONE ; FDST)X
6357
6358 ;REPORT RESULT INCORRECT.
6359 025534 012737 102345 001240 NNC15: MOV #102345,@#STMP3 ; ST 634
6360 025542 013737 025506 001242 MOV @#NNCTB1,@#STMP4
6361 025550 104001 1$: ERROR 1 ;BAD DATA
6362 025552 000422 BR NNCDONE
6363
6364 ;REPORT RESULT INCORRECT.
6365 025554 012737 177777 001240 NNC20: MOV #-1,@#STMP3
6366 025562 013737 025510 001242 MOV @#NNCTB1+2,@#STMP4
6367 025570 104001 1$: ERROR 1 ;(BUT GR7,FL)
6368 025572 000412 BR NNCDONE ;ST 357 TO 416
6369 ;INTO 417
6370
6371 ;IF A TRAP TO VECTOR 4 OCCURS COME HERE TO SEE IF THE TRAP OCCURRED
6372 ;DURING EXECUTION OF THE FPP INSTRUCTION BEING TESTED, IF NOT GO
6373 ;TO THE SPURIOUS TRAP TO 4 HANDLER.
6374 NNC25: MOV (SP),R4
6375 025574 011604 CMP R4,#NNC2+2
6376 025576 020427 025452 BEQ 1$
6377 025602 001402 JMP @#CPSPUR
6378 025604 000137 036172
```

6379
6380 025610 011637 001236
6381 025614 022626
6382 025616 104001
6383
6384 025620
6385 025620 104413
6386
6387
6388
6389
6390
6391
6392
6393
6394
6395
6396
6397
6398
6399 025622 000004
6400
6401
6402 025624
6403 025624 104414
6404 025626 012700 025724
6405 025632 012701 000006
6406 025636 012720 177777
6407 025642 077103
6408 025644 012700 105412
6409 025650 012737 025672 001236
6410 025656 012737 026016 000004
6411 025664 170100
6412 025666 012700 025730
6413
6414 025672 170220
6415 025674 020027 025732
6416 025700 001017
6417 025702 023727 025730 105412
6418 025710 001022
6419 025712 023727 025732 177777
6420 025720 001026
6421 025722 000447
6422
6423
6424 025724 177777 177777
6425 025730 177777 177777 177777
6426 025736 177777
6427
6428
6429 025740 010037 001242
6430 025744 012737 025732 001240
6431 025752 104001
6432 025754 000432
6433
6434

1\$: MOV (SP),@#STMP2
CMP (SP)+,(SP)+
2\$: ERROR 1 ;(BUT FDST)+ ST634
NINCDONE:
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 62 DESTINATION MODES, MODE 2 (FL 0), TEST
:*****
:THIS IS A TEST OF DESTINATION MODE 2 USING
:THE STFPS INSTRUCTION
:*****
TST62: SCOPE

OOC1: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
MOV #OOC1B0,R0 ;SET UP THE DATA BUFFER.
MOV #6,R1
1\$: MOV #-1,(R0)+
SOB R1,1\$
MOV #105412,R0
MOV #OOC2,@#STMP2
MOV #OOC25,@#ERRVECT ;SET UP FOR TRAPS TO VECTOR 4.
LDFPS R0 ;SET UP FPS.
MOV #OOC1B1,R0
OOC2: STFPS (R0)+ ;TEST INSTRUCTION.
CMP R0,#OOC1B1+2 ;IS R0 CORRECT?
BNE OOC10 ;BRANCH IF NOT CORRECT.
CMP @#OOC1B1,#105412 ;IS THE RESULT CORRECT?
BNE OOC15 ;BRANCH IF NOT CORRECT.
CMP @#OOC1B1+2,#-1 ;IS THE RESULT CORRECT?
BNE OOC20 ;BRANCH IF NOT CORRECT.
BR OOCDONE

;TEST DATA BUFFER:
OOC1B0: .WORD -1,-1
OOC1B1: .WORD -1,-1,-1,-1

;REPORT R0 INCORRECT.
OOC10: MOV R0,@#STMP4
MOV #OOC1B1+2,@#STMP3
1\$: FRROR 1 ;R0 BAD (BUT
BR OOCDONE ; FDST)X

;REPORT RESULT INCORRECT.

```

6435 025756 012737 105412 001240 00C15: MOV #105412,@#STMP3 ; ST 634
6436 025764 013737 025730 001242 MOV @#OCTB1,@#STMP4
6437 025772 104001 1$: ERROR 1 ;BAD DATA
6438 025774 000422 BR 00CDONE
6439
6440
6441 ;REPORT RESULT INCORRECT.
6442 025776 012737 177777 001240 00C20: MOV #-1,@#STMP3
6443 026004 013737 025732 001242 MOV @#OCTB1+2,@#STMP4
6444 026012 104001 1$: ERROR 1 ;(BUT GR7,FL)
6445 026014 000412 BR 00CDONE ;ST 357 TO 416
6446 ;INTO 417
6447
6448 ;IF A TRAP TO VECTOR 4 OCCURS COME HERE TO SEE IF THE TRAP OCCURRED
6449 ;DURING EXECUTION OF THE FPP INSTRUCTION BEING TESTED, IF NOT GO
6450 ;TO THE SPURIOUS TRAP TO 4 HANDLER.
6451 026016 011604 00C25: MOV (SP),R4
6452 026020 020427 025674 CMP R4,#00C2+2
6453 026024 001402 BEQ 1$
6454 026026 000137 036172 JMP @#(PSPUR)
6455
6456 026032 011637 001236 1$: MOV (SP),@#STMP2
6457 026036 022626 CMP (SP)+,(SP)+
6458 026040 104001 2$: ERROR 1 ;(BUT FDST)+ ST634
6459
6460 026042 00CDONE:
6461 026042 104413 RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
6462 ;SEE IF THE USER HAS EXPRESSED
6463 ;THE DESIRE TO CHANGE THE SOFTWARE
6464 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
6465 ;THE USER TYPED CONTROL G?).
6466
6467
6468
6469
6470 ;*****
6471 ;*TEST 63 DESTINATION MODES, MODE 4 (FL=0), TEST
6472 ;*
6473 ;* THIS IS A TEST OF DESTINATION MODE 4 USING
6474 ;* THE STFPS INSTRUCTION
6475 ;*
6476 ;*****
6476 026044 000004 TST63: SCOPE
6477
6478 026046 PPC1:
6479 026046 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
6480 026050 012700 026146 MOV #PPCTB0,R0 ;SET UP THE DATA BUFFER.
6481 026054 012701 000006 MOV #6,R1
6482 026060 012720 177777 1$: MOV #-1,(R0)+
6483 026064 077103 SOB R1,1$
6484 026066 012700 105555 MOV #105555,R0
6485 026072 012737 026114 001236 MOV #PPC2,@#STMP2
6486 026100 012737 026240 000004 MOV #PPC25,@#ERRVECT ;SET UP FOR TRAPS TO VECTOR 4.
6487 026106 170100 LDFPS R0 ;SET UP FPS.
6488 026110 012700 026154 MOV #PPCTB1+2,R0
6489
6490 026114 170240 PPC2: STFPS -(R0) ;TEST INSTRUCTION.

```



```

6491 026116 020027 026152      CMP      RO,#PPCTB1      ;IS RO CORRECT?
6492 026122 001017              BNE      PPC10           ;BRANCH IF NOT CORRECT.
6493 026124 023727 026152 105555  CMP      @PPCTB1,#105555 ;IS THE RESULT CORRECT?
6494 026132 001022              BNE      PPC15           ;BRANCH IF NOT CORRECT.
6495 026134 023727 026154 177777  CMP      @PPCTB1+2,#-1   ;IS THE RESULT CORRECT?
6496 026142 001026              BNE      PPC20           ;BRANCH IF NOT CORRECT.
6497 026144 000447              BR       PPCDONE
6498
6499
6500 026146 177777 177777      ;TEST DATA BUFFER:
6501 026152 177777 177777 177777  PPC10: .WORD  -1,-1
6502 026160 177777      PPC11: .WORD  -1,-1,-1,-1
6503
6504
6505 026162 010037 001242      ;REPORT RO INCORRECT.
6506 026166 012737 026152 001240  PPC10: MOV      RO,@STMP4
6507 026174 104001              MOV      #PPCTB1,@STMP3
6508 026176 000432      1$:  ERROR      1          ;RO BAD (BUT
6509                                     BR       PPCDONE          ;FDST)X
6510
6511 026200 012737 105555 001240  ;REPORT RESULT INCORRECT.
6512 026206 013737 026152 001242  PPC15: MOV      #105555,@STMP3
6513 026214 104001              MOV      @PPCTB1,@STMP4
6514 026216 000422      1$:  ERROR      1          ;BAD DATA
6515
6516
6517
6518 026220 012737 177777 001240  ;REPORT RESULT INCORRECT.
6519 026226 013737 026154 001242  PPC20: MOV      #-1,@STMP3
6520 026234 104001              MOV      @PPCTB1+2,@STMP4
6521 026236 000412      1$:  ERROR      1          ;(BUT GR7,FL)
6522                                     BR       PPCDONE          ;ST 357 TO 416
6523                                     ;INTO 417
6524
6525
6526
6527 026240 011604              ;IF A TRAP TO VECTOR 4 OCCURS COME HERE TO SEE IF THE TRAP OCCURRED
6528 026242 020427 026116      ;DURING EXECUTION OF THE FPP INSTRUCTION BEING TESTED, IF NOT GO
6529 026246 001402              ;TO THE SPURIOUS TRAP TO 4 HANDLER.
6530 026250 000137 036172      PPC25: MOV      (SP),R4
6531                                     CMP      R4,#PPC2+2
6532                                     BEQ      1$
6533                                     JMP      @CPSPUR
6534
6535
6536 026254 011637 001236      1$:  MOV      (SP),@STMP2
6537 026260 022626              CMP      (SP)+,(SP)+
6538 026262 104001      2$:  ERROR      1          ;(BUT FDST)+ ST634
6539
6540
6541
6542
6543
6544
6545
6546
PPCDONE:
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 TYFED CONTROL G?).
;*****
;*TEST 64 DESTINATION MODES, MODE 3 (FL=0), TEST
    
```

6547
6548
6549
6550
6551
6552 026266 000004
6553
6554 026270
6555 026270 104414
6556 026272 012700 026374
6557 026276 012701 000010
6558 026302 012720 177777
6559 026306 077103
6560 026310 012700 106653
6561 026314 012737 026342 001236
6562 026322 012737 026472 000004
6563 026330 170100
6564 026332 012700 026410
6565 026336 012710 026400
6566
6567 026342 170230
6568 026344 020027 026412
6569 026350 001021
6570 026352 023727 026400 106653
6571 026360 001024
6572 026362 023727 026410 026400
6573 026370 001030
6574 026372 000451
6575
6576
6577 026374 177777 177777
6578 026400 177777 177777 177777
6579 026406 177777
6580 026410 177777 177777
6581
6582
6583 026414 010037 001242
6584 026420 012737 026412 001240
6585 026426 104001
6586 026430 000432
6587
6588
6589 026432 012737 106653 001240
6590 026440 013737 026400 001242
6591 026446 104001
6592 026450 000422
6593
6594
6595
6596 026452 012737 026410 001240
6597 026460 013737 026402 001242
6598 026466 104001
6599 026470 000412
6600
6601
6602

```

: *
: * THIS IS A TEST OF DESTINATION MODE 3 USING
: * THE STFPS INSTRUCTION
: *
: *****
TST64: SCOPE

QQC1:
LPERR                                ;SET UP THE LOOP ON ERROR ADDRESS.
MOV #QQCTB0,R0                       ;SET UP THE DATA BUFFER.
MOV #10,R1
1$: MOV #-1,(R0)+
SOB R1,1$
MOV #106653,R0
MOV #QQC2,@#STMP2
MOV #QQC25,@#ERRVECT ;SET UP FOR TRAPS TO VECTOR 4.
LDFPS R0                               ;SET UP FPS.
MOV #QQCTB2,R0
MOV #QQCTB1,(R0)

QQC2: STFPS @ (R0)+ ;TEST INSTRUCTION.
CMP R0,#QQCTB2+2 ;IS R0 CORRECT?
BNE QQC10 ;BRANCH IF NOT CORRECT.
CMP @#QQCTB1,#106653 ;IS THE RESULT CORRECT?
BNE QQC15 ;BRANCH IF NOT CORRECT.
CMP @#QQCTB2,#QQCTB1 ;IS THE RESULT CORRECT?
BNE QQC20 ;BRANCH IF NOT CORRECT.
BR QQCDONE

;TEST DATA BUFFER:
QQCTB0: .WORD -1,-1
QQCTB1: .WORD -1,-1,-1,-1
QQCTB2: .WORD -1,-1

;REPORT R0 INCORRECT.
QQC10: MOV R0,@#STMP4
MOV #QQCTB2+2,@#STMP3
1$: ERROR 1 ;R0 BAD (BUT
BR QQCDONE ; FDST)X

;REPORT RESULT INCORRECT.
QQC15: MOV #106653,@#STMP3 ; ST 634
MOV @#QQCTB1,@#STMP4
1$: ERROR 1 ;BAD DATA
BR QQCDONE

;REPORT RESULT INCORRECT.
QQC20: MOV #QQCTB2,@#STMP3 ;(BUT FDST)
MOV @#QQCTB1+2,@#STMP4
1$: ERROR 1
BR QQCDONE

;IF A TRAP TO VECTOR 4 OCCURS COME HERE TO SEE IF THE TRAP OCCURRED
```

6603
6604
6605 026472 011604
6606 026474 020427 026344
6607 026500 001402
6608 026502 000137 036172
6609
6610 026506 011637 001236
6611 026512 022626
6612 026514 104001
6613
6614 026516
6615 026516 104413
6616
6617
6618
6619
6620
6621
6622
6623
6624
6625
6626
6627
6628
6629
6630 026520 000004
6631
6632
6633 026522
6634 026522 104414
6635 026524 012700 026630
6636 026530 012701 000006
6637 026534 012720 177777
6638 026540 077103
6639 026542 012700 004301
6640 026546 012737 026576 001236
6641 026554 012737 026726 000004
6642 026562 170100
6643 026564 012700 026646
6644 026570 012760 026634 177776
6645
6646 026576 170250
6647 026600 020027 026644
6648 026604 001021
6649 026606 023727 026634 004301
6650 026614 001024
6651 026616 023727 026644 026634
6652 026624 001030
6653 026626 000451
6654
6655
6656 026630 177777 177777
6657 026634 177777 177777 177777
6658 026642 177777

:DURING EXECUTION OF THE FPP INSTRUCTION BEING TESTED, IF NOT GO
:TO THE SPURIOUS TRAP TO 4 HANDLER.
QQC25: MOV (SP),R4
CMP R4,#QQC2+2
BEQ 1\$
JMP @#CPSPUR
1\$: MOV (SP),@#STMP2
CMP (SP)+,(SP)+
2\$: ERROR 1 ;(BUT FDST)+ ST634
QQCDONE: 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 65 DESTINATION MODES, MODE 5 (FL=0), TEST
:*
:* THIS IS A TEST OF DESTINATION MODE 5 USING
:* THE STFPS INSTRUCTION
:*
:*****
TST65: SCOPE

RRC1: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
MOV #RRCTB0,R0 ;SET UP THE DATA BUFFER.
MOV #6,R1
1\$: MOV #-1,(R0)+
SOB R1,1\$
MOV #004301,R0
MOV #RRC2,@#STMP2
MOV #RRC25,@#ERRVECT ;SET UP FOR TRAPS TO VECTOR 4.
LDFPS R0 ;SET UP FPS.
MOV #RRCTB2+2,R0
MOV #RRCTB1,-2(R0)
RRC2: STFPS @-(R0) ;TEST INSTRUCTION.
CMP R0,#RRCTB2 ;IS R0 CORRECT?
BNE RRC10 ;BRANCH IF NOT CORRECT.
CMP @#RRCTB1,#004301 ;IS THE RESULT CORRECT?
BNE RRC15 ;BRANCH IF NOT CORRECT.
CMP @#RRCTB2,#RRCTB1 ;IS THE RESULT CORRECT?
BNE RRC20 ;BRANCH IF NOT CORRECT.
BR RRCDONE

:TEST DATA BUFFER:
RRC10: .WORD -1,-1
RRC15: .WORD -1,-1,-1,-1

CJKDDA REF 11-A DIAG PART 2
CJKDDA.P11 20-JUN-79 11:22

MACY11 30A(1052) 20-JUN-79 11:39 PAGE 123
T65 DESTINATION MODES, MODE 5 (FL-0), TEST

SEQ 0123

```

6659 026644 177777 177777      RRCTB2: .WORD  -1,-1
6660
6661      :REPORT RO INCORRECT.
6662 026650 010037 001242      RRC10: MOV    RO,@#STMP4
6663 026654 012737 026644 001240  MOV    #RRCTB2,@#STMP3
6664 026662 104001      1$:    ERROR  1      :RO BAD (BUT
6665 026664 000432      BR     RRCDONE      ;    FDST)X
6666
6667      :REPORT RESULT INCORRECT.
6668 026666 012737 004301 001240  RRC15: MOV    #004301,@#STMP3      :    ST 634
6669 026674 013737 026634 001242  MOV    @#RRCTB1,@#STMP4
6670 026702 104001      1$:    ERROR  1      :BAD DATA
6671 026704 000422      BR     RRCDONE
6672
6673
6674      :REPORT RESULT INCORRECT.
6675 026706 012737 026644 001240  RRC20: MOV    #RRCTB2,@#STMP3      :BUT FDST)
6676 026714 013737 026636 001242  MOV    @#RRCTB1+2,@#STMP4
6677 026722 104001      1$:    ERROR  1      : (BUT GR7,FL)
6678 026724 000412      BR     RRCDONE      ;ST 357 TO 416
6679      ;INTO 417
6680
6681      :IF A TRAP TO VECTOR 4 OCCURS COME HERE TO SEE IF THE TRAP OCCURRED
6682      :DURING EXECUTION OF THE FPP INSTRUCTION BEING TESTED, IF NOT GO
6683      :TO THE SPURIOUS TRAP TO 4 HANDLER.
6684 026726 011604      RRC25: MOV    (SP),R4
6685 026730 020427 026600      CMP    R4,#RRC2+?
6686 026734 001402      BEQ   1$
6687 026736 000137 036172      JMP   @#CPSPUR
6688
6689 026742 011637 001236      1$:    MOV    (SP),@#STMP2
6690 026746 022626      CMP    (SP)+,(SP)+
6691 026750 104001      2$:    ERROR  1      ;(BUT FDST)+ ST634
6692
6693      RRCDONE:
6694 026752 104413      RSETUP      ;GO INITIALIZE THE FPS AND STACK; AND
6695      ;SEE IF THE USER HAS EXPRESSED
6696      ;THE DESIRE TO CHANGE THE SOFTWARE
6697      ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
6698      ;THE USER TYPED CONTROL G?).
6699
6700
6701      :*****
6702      :*TEST 66      DESTINATION MODES, MODE 6 (FL-0), TEST
6703      :*
6704      :* THIS IS A TEST OF DESTINATION MODE 6 USING
6705      :* THE STFPS INSTRUCTION
6706      :*
6707      :*****
6708 026754 000004      TST66: SCOPE
6709
6710
6711      SSC1:
6712 026756 104414      LPERR
6713 026760 012700 027070      MOV    #SSCTB0,RO      ;SET UP THE LOOP ON ERROR ADDRESS.
6714 026764 012701 000006      MOV    #6,R1          ;SET UP THE DATA BUFFER.

```

```

6715 026770 012720 177777 1$: MOV #-1,(R0)+
6716 026774 077103 SOB R1,1$
6717 026776 012700 102514 MOV #102514,R0
6718 027002 012737 027026 001236 MOV #SSC2,@#STMP2
6719 027010 012737 027162 000004 MOV #SSC25,@#ERRVECT ;SET UP FOR TRAPS TO VECTOR 4.
6720 027016 170100 LDFPS R0 ;SET UP FPS.
6721 027020 005001 CLR R1
6722 027022 012700 021673 MOV #SSCTB1-5201,R0
6723
6724 027026 170260 005201 SSC2: STFPS 5201(R0) ;TEST INSTRUCTION.
6725 027032 020127 000000 CMP R1,#0 ;WAS PC CORRECT AFTER EXECUTION?
6726 027036 001064 BNE SSC30 ;BRANCH IF NOT CORRECT.
6727 027040 020027 021673 CMP R0,#SSCTB1-5201 ;IS R0 CORRECT?
6728 027044 001017 BNE SSC10 ;BRANCH IF NOT CORRECT.
6729 027046 023727 027074 102514 CMP @#SSCTB1,#102514 ;IS THE RESULT CORRECT?
6730 027054 001022 BNE SSC15 ;BRANCH IF NOT CORRECT.
6731 027056 023727 027076 177777 CMP @#SSCTB1+2,#-1 ;IS THE RESULT CORRECT?
6732 027064 001026 BNE SSC20 ;BRANCH IF NOT CORRECT.
6733 027066 000451 BR SSCDONE
6734
6735 ;TEST DATA BUFFER:
6736 027070 177777 177777 SSCTB0: .WORD -1,-1
6737 027074 177777 177777 177777 SSCTB1: .WORD -1,-1,-1,-1
6738 027102 177777
6739
6740 ;REPORT RO INCORRECT.
6741 027104 010037 001242 SSC10: MOV R0,@#STMP4
6742 027110 012737 021673 001240 MOV #SSCTB1-5201,@#STMP3
6743 027116 104001 1$: ERROR 1 ;RO BAD
6744 027120 000434 BR SSCDONE
6745
6746 ;REPORT RESULT INCORRECT.
6747 027122 012737 102534 001240 SSC15: MOV #102534,@#STMP3
6748 027130 013737 027074 001242 MOV @#SSCTB1,@#STMP4
6749 027136 104001 1$: ERROR 1 ;BAD DATA
6750 027140 000424 BR SSCDONE
6751
6752 ;REPORT RESULT INCORRECT.
6753 027142 012737 177777 001240 SSC20: MOV #-1,@#STMP3
6754 027150 013737 027076 001242 MOV @#SSCTB1+2,@#STMP4
6755 027156 104001 1$: ERROR 1 ;(BUT GR7,FL)
6756 027160 000414 BR SSCDONE ;ST 357 TO 416
6757 ;INTO 417
6758
6759 ;IF A TRAP TO VECTOR 4 OCCURS COME HERE TO SEE IF THE TRAP OCCURRED
6760 ;DURING EXECUTION OF THE FPP INSTRUCTION BEING TESTED, IF NOT GO
6761 ;TO THE SPURIOUS TRAP TO 4 HANDLER.
6762
6763 027162 011604 SSC25: MOV (SP),R4
6764 027164 020427 027030 CMP R4,#SSC2+?
6765 027170 001402 BEQ 1$
6766 027172 000137 036172 JMP @#CPSPUR
6767
6768 027176 011637 001236 1$: MOV (SP),@#STMP2
6769 027202 022626 CMP (SP)+,(SP)+
6770 027204 104001 2$: ERROR 1 ;(BUT FDST)+ ST634

```

6771 027206 000401

BR SSCDONE

6772

6773

6774 027210

6775 027210 104001

6776

6777

6778

6779 027212

6780 027212 104413

6781

6782

6783

6784

6785

6786

6787

6788

6789

6790

6791

6792

6793

6794 027214 000004

6795

6796 027216

6797 027216 104414

6798 027220 012700 027336

6799 027224 012701 000010

6800 027230 012720 177777

6801 027234 077103

6802 027236 012700 103747

6803 027242 012737 027274 001236

6804 027250 012737 027434 000004

6805 027256 170100

6806 027260 005001

6807 027262 012700 022151

6808 027266 012760 027342 005201

6809

6810 027274 170270 005201

6811 027300 022701 000000

6812 027304 001066

6813 027306 020027 022151

6814 027312 001021

6815 027314 023727 027342 103747

6816 027322 001024

6817 027324 023727 027344 177777

6818 027332 001030

6819 027334 000453

6820

6821

6822 027336 177777 177777

6823 027342 177777 177777 177777

6824 027350 177777

6825 027352 177777 177777

6826

:REPORT PC NOT INCREMENTED BY 2 DURING EXECUTION.

SSC30:

1\$: ERROR 1

:PC NOT
:INCREMENTED
:BY 2

SSCDONE:

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 67 DESTINATION MODES, MODE 7 (FL=0), TEST

:*

:* THIS IS A TEST OF DESTINATION MODE 7 USING

:* THE STFPS INSTRUCTION

:*

TST67: SCOPE

TTC1:

LPERR

:SET UP THE LOOP ON ERROR ADDRESS.
:SET UP THE DATA BUFFER.

MOV #TTCB0,R0

MOV #10,R1

1\$: MOV #-1,(R0)+

SQB R1,1\$

MOV #103747,R0

MOV #TTC2,@\$TMP2

MOV #TTC25,@\$ERRVECT ;SET UP FOR TRAPS TO VECTOR 4.

LDFPS R0

:SET UP FPS.

CLR R1

MOV #TTCB2-5201,R0

MOV #TTCB1,5201(R0)

TTC2: STFPS @5201(R0)

:TEST INSTRUCTION.

CMP #0,R1

:WAS PC CORRECT AFTER EXECUTION?

BNE TTC30

:BRANCH IF NOT CORRECT.

CMP R0,#TTCB2-5201

:IS R0 CORRECT?

BNE TTC10

:BRANCH IF NOT CORRECT.

CMP @#TTCB1,#103747

:IS THE RESULT CORRECT?

BNE TTC15

:BRANCH IF NOT CORRECT.

CMP @#TTCB1+2,#-1

:IS THE RESULT CORRECT?

BNE TTC20

:BRANCH IF NOT CORRECT.

BR TTCDONE

:TEST DATA BUFFER:

TTCB0: .WORD -1,-1

TTCB1: .WORD -1,-1,-1,-1

TTCB2: .WORD -1,-1

```

6827      :REPORT R0 INCORRECT.
6828 027356 010037 001242      TTC10: MOV R0,@#STMP4
6829 027362 012737 022151 001240      MOV #TTCB2-5201,@#STMP3
6830 027370 104001      1$: ERROR 1 ;R0 BAD
6831 027372 000434      BR TTCDONE
6832
6833
6834      :REPORT RESULT INCORRECT.
6835 027374 012737 103747 001240      TTC15: MOV #103747,@#STMP3
6836 027402 013737 027342 001242      MOV @#TTCB1,@#STMP4
6837 027410 104001      1$: ERROR 1 ;BAD DATA
6838 027412 000424      BR TTCDONE
6839
6840      :REPORT RESULT INCORRECT.
6841
6842 027414 012737 177777 001240      TTC20: MOV #-1,@#STMP3
6843 027422 013737 027344 001242      MOV @#TTCB1+2,@#STMP4
6844 027430 104001      1$: ERROR 1 ;(BUT GR7,FL)
6845 027432 000414      BR TTCDONE ;ST 357 TO 416
6846 ;INTO 417
6847
6848      :IF A TRAP TO VECTOR 4 OCCURS COME HERE TO SEE IF THE TRAP OCCURRED
6849      :DURING EXECUTION OF THE FPP INSTRUCTION BEING TESTED, IF NOT GO
6850      :TO THE SPURIOUS TRAP TO 4 HANDLER.
6851 027434 011604      TTC25: MOV (SP),R4
6852 027436 020427 027276      CMP R4,#TTC2+2
6853 027442 001402      BEQ 1$
6854 027444 000137 036172      JMP @#CPSPUR
6855 027450 011637 001236      1$: MOV (SP),@#STMP2
6856 027454 022626      CMP (SP)+,(SP)+
6857 027456 104001      2$: ERROR 1 ;(BUT FSDT)+ ST634
6858 027460 000401      BR TTCDONE
6859
6860      :REPORT PC NOT INCREMENTED BY 2 DURING EXECUTION.
6861 027462      TTC30:
6862 027462 104001      1$: ERROR 1 ;PC NOT
6863 ;INCREMENTED
6864 027464      TTCDONE:
6865 027464 104413      RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
6866 ;SEE IF THE USER HAS EXPRESSED
6867 ;THE DESIRE TO CHANGE THE SOFTWARE
6868 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
6869 ;THE USER TYPED CONTROL G?).
6870
6871      ;*****
6872      ;*TEST 70 DESTINATION MODES, MODE 2 (FL=1), TEST
6873      ;*
6874      ;* THIS IS A TEST OF DESTINATION MODE
6875      ;* 2 USING STCOL WITH REGISTER 0
6876      ;*
6877      ;*****
6878 027466 000004      TST70: SCOPE
6879 027470      UUC1:
6880 027470 104414      LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
6881 027472 012700 000300      MOV #300,R0 ;SET UP FPS.
6882 027476 170100      LDFPS R0

```

```
6883 027500 012700 027546      MOV    #UUCTP1,R0      ;SET UP THE ACO OPERAND.
6884 027504 172410              LDD    (R0),ACO
6885 027506 012737 027520 001236  MOV    #UUC2,@#STMP2
6886 027514 012700 027560      MOV    #UUCBFO,R0
6887
6888 027520 175420              UUC2:  STCDL  ACO,(R0)+  ;TEST INSTRUCTION.
6889
6890 027522 020027 027564      CMP    RO,#UUCBFO+4   ;IS RO CORRECT?
6891 027526 001417              BEQ    UUCDONE        ;BRANCH IF CORRECT.
6892
6893                          ;REPORT RO INCORRECT.
6894 027530 010037 001242      UUC3:  MOV    RO,@#STMP4
6895 027534 012737 027564 001240  MOV    #UUCBFO+4,@#STMP3
6896 027542 104001              1$:    ERROR 1          ;RO NOT INCR BY 4
6897 027544 000410              BR     UUCDONE
6898
6899 027546 000000 000000 000000  ;TEST DATA BUFFER:
6900 027554 000000              UUCTP1: .WORD 0,0,0,0
6901 027556 177777              -1
6902 027560 177777 177777 177777  UUCBFO: .WORD -1,-1,-1
6903
6904 027566              UUCDONE:
6905 027566 104413              RSETUP                ;GO INITIALIZE THE FPS AND STACK; AND
6906                                     ;SEE IF THE USER HAS EXPRESSED
6907                                     ;THE DESIRE TO CHANGE THE SOFTWARE
6908                                     ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
6909                                     ;THE USER TYPED CONTROL G?).
6910
6911                          ;*****
6912                          ;*TEST 71      DESTINATION MODES, MODE 4 (FL=1), TEST
6913                          ;*
6914                          ;* THIS IS A TEST OF DESTINATION MODE
6915                          ;* 4 USING STCDL WITH REGISTER 0
6916                          ;*
6917                          ;*****
6918 027570 000004      TST71: SCOPE
6919
6920 027572              VVC1:
6921 027572 104414              LPERR                ;SET UP THE LOOP ON ERROR ADDRESS.
6922 027574 012700 000300      MOV    #300,R0      ;SET UP FPS.
6923 027600 170100              LDFPS  R0
6924 027602 012700 027650      MOV    #VVC1P1,R0   ;SET UP THE ACO OPERAND.
6925 027606 172410              LDD    (R0),ACO
6926 027610 012737 027622 001236  MOV    #VVC2,@#STMP2
6927 027616 012700 027666      MOV    #VVCBFO+4,R0
6928
6929 027622 175440              VVC2:  STCDL  ACO,-(R0)  ;TEST INSTRUCTION.
6930
6931 027624 020027 027662      CMP    RO,#VVCBFO   ;IS RO CORRECT?
6932 027630 001417              BEQ    VVCDONE
6933
6934                          ;REPORT RO INCORRECT.
6935 027632 010037 001242      VVC3:  MOV    RO,@#STMP4
6936 027636 012737 027662 001240  MOV    #VVCBFO,@#STMP3
6937 027644 104001              1$:    ERROR 1          ;RO NOT DECR BY 4
6938 027646 000410              BR     VVCDONE
```



```
6939 ;TEST DATA BUFFER:
6940 027650 000000 000000 000000 VVCTP1: .WORD 0,0,0,0
6941 027656 000000
6942 027660 177777 -1
6943 027662 177777 177777 177777 VVCBF0: .WORD -1,-1,-1
6944
6945 027670 VVCDONE:
6946 027670 104413 RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
6947 ;SEE IF THE USER HAS EXPRESSED
6948 ;THE DESIRE TO CHANGE THE SOFTWARE
6949 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
6950 ;THE USER TYPED CONTROL G?).
6951
6952 ;*****
6953 ;*TEST 72 STCDI AND STCDL TEST
6954 ;*
6955 ;* THIS IS A TEST OF THE STCDI AND
6956 ;* STCDL INSTRUCTIONS. NOTE THAT A
6957 ;* SUBROUTINE, STCSUB, IS USED TO
6958 ;* SET UP THE OPERANDS, EXECUTE THE STC
6959 ;* INSTRUCTION AND CHECK THE RESULT.
6960 ;*
6961 ;*****
6962 027672 000004 TST72: SCOPE
6963
6964 ;FIRST TEST STC WITH EXP=100 (EXCESS 200)
6965 027674 WWC1:
6966 027674 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
6967 027676 004737 031042 JSR PC,@STCSUB ;GO EXECUTE THE INSTRUCTION.
6968 027702 020000 000000 000000 1$: .WORD 2000,0,0,0 ;ACO OPERAND.
6969 027710 000000
6970 027712 000000 000000 2$: .WORD 0,0 ;EXPECTED RESULT.
6971 027716 177777 177777 3$: .WORD -1,-1 ;ERROR RES.
6972 027722 040300 4$: 40300 ;FPS BEFORE EXECUTION.
6973 027724 040304 40304 ;FPS AFTER EXECUTION.
6974 027726 140304 140304 ;ANTICIPATED ERRONEOUS FPS.
6975 027730 177777 -1 ;REPORT RESULT INCORRECT.
6976 027732 104001 5$: ERROR 1 ;RESULT INCORP.
6977 027734 000401 BR 6$
6978 027736 104001 ERROR 1 ;EITHER (BUT FLAG)
6979 027740 6$: ;ST 662
6980 ;OR CLEAR FLAG
6981 ;ST 774
6982
6983 ;EXP=0 (OCT) FL=1 FIC=0
6984 027740 WWC2:
6985 027740 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
6986 027742 004737 031042 JSR PC,@STCSUB ;GO EXECUTE THE INSTRUCTION.
6987 027746 040000 000000 000000 1$: .WORD 4000,0,0,0 ;AC ;ACO OPERAND.
6988 027754 000000
6989 027756 000000 000000 2$: .WORD 0,0 ;EXPECTED RESULT.
6990 027762 177777 177777 3$: .WORD -1,-1 ;ANTICIPATED ERRONEOUS RESULT.
6991 027766 040313 4$: 40313 ;FPS BEFORE EXECUTION.
6992 027770 040304 40304 ;FPS AFTER EXECUTION.
6993 027772 140304 140304 ;ANTICIPATED ERRONEOUS FPS.
6994 027774 177777 -1 ;EXPECTED FEC.
```

```

6995 027776 104001      5$:  ERROR 1      ;REPORT RESULT INCORRECT.
6996 030000 000401      BR      6$
6997 030002 104001      ERROR 1      ;REPORT FPS INCORRECT.
6998 030004
6999
7000      ;EXP=37 (OCT)  FL=1  FIC=1
7001 030004      WWC4:
7002 030004 104414      LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.
7003 030006 004737 031042  JSR      PC,@#STCSUB ;GO EXECUTE THE INSTRUCTION.
7004 030012 047667 075757 157737 1$:  .WORD 47667,75757,157737,167773 ;ACO OPERAND.
7005 030020 167773
7006 030022 055675 173757 2$:  .WORD 55675,173757 ;EXPECTED RESULT.
7007 030026 122102 004021 3$:  .WORD 122102,004021 ;ANTICIPATED ERRONEOUS RESULT.
7008 030032 040717 4$:  40717 ;FPS BEFORE EXECUTION.
7009 030034 040700 ;FPS AFTER EXECUTION.
7010 030036 140705 ;ANTICIPATED ERRONEOUS FPS.
7011 030040 177777 -1 ;EXPECTED FEC.
7012 030042 104001 5$:  ERROR 1 ;(BUT ENBT) ST 632
7013 030044 000401 BR      6$
7014 030046 104001 ERROR 1 ;REPORT FPS INCORRECT.
7015 030050
7016
7017      ;EXP=40 (OCT)  FL=1  FIC=1
7018 030050      WWC5:
7019 030050 104414      LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.
7020 030052 004737 031042  JSR      PC,@#STCSUB ;GO EXECUTE THE INSTRUCTION.
7021 030056 050000 000000 000000 1$:  .WORD 50000,0,0,0 ;ACO OPERAND.
7022 030064 000000
7023 030066 000000 000000 2$:  .WORD 0,0 ;EXPECTED RESULT.
7024 030072 177777 177777 3$:  .WORD -1,-1 ;ANTICIPATED ERRONEOUS RESULT.
7025 030076 040700 4$:  40700 ;FPS BEFORE EXECUTION.
7026 030100 140705 ;FPS AFTER EXECUTION.
7027 030102 040705 40705 ;ANTICIPATED ERRONEOUS FPS.
7028 030104 000006 6 ;EXPECTED FEC.
7029 030106 104001 5$:  ERROR 1 ;REPORT RESULT INCORRECT.
7030 030110 000401 BR      6$
7031 030112 104001 ERROR 1 ;(BUT FIC) ST 004 ;REPORT FPS INCORRECT.
7032 ;TO 305 INTO
7033 030114 6$:  ;315
7034
7035      ;EXP=40 (OCT)  FL=1  FIC=0
7036 030114      WWC6:
7037 030114 104414      LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.
7038 030116 004737 031042  JSR      PC,@#STCSUB ;GO EXECUTE THE INSTRUCTION.
7039 030122 050000 000000 000000 1$:  .WORD 50000,0,0,0 ;ACO OPERAND.
7040 030130 000000
7041 030132 000000 000000 2$:  .WORD 0,0 ;EXPECTED RESULT.
7042 030136 177777 177777 3$:  .WORD -1,-1 ;ANTICIPATED ERRONEOUS RESULT.
7043 030142 040312 4$:  40312 ;FPS BEFORE EXECUTION.
7044 030144 040305 40305 ;FPS AFTER EXECUTION.
7045 030146 140305 140305 ;ANTICIPATED ERRONEOUS FPS.
7046 030150 177777 -1 ;EXPECTED FEC.
7047 030152 104001 5$:  ERROR 1 ;REPORT RESULT INCORRECT.
7048 030154 000401 BR      6$
7049 030156 104001 ERROR 1 ;(BUT FIC) ST 004 TO
7050 030160 6$:  ;315 INTO 305

```

```

7051
7052           ;EXP=30 (OCT)  FL=1  FIC=1
7053 030160   WWC7:
7054 030160   104414           LPERR
7055 030162   004737 031042     JSR      PC,@#STCSUB  ;SET UP THE LOOP ON ERROR ADDRESS.
7056 030166   046000 000001 000000 1$:  .WORD  46000,1,0,0  ;GO EXECUTE THE INSTRUCTION.
7057 030174   000000           ;ACO OPERAND.
7058 030176   000200 000001     2$:  .WORD  200,1  ;EXPECTED RESULT.
7059 030202   177777 177777     3$:  .WORD  -1,-1  ;ANTICIPATED ERRONEOUS RESULT.
7060 030206   040700           4$:  40700  ;FPS BEFORE EXECUTION.
7061 030210   040700           40700  ;FPS AFTER EXECUTION.
7062 030212   177777           -1      ;ANTICIPATED ERRONEOUS FPS.
7063 030214   177777           -1      ;EXPECTED FEC.
7064 030216   104001     5$:  ERROR  1  ;REPORT RESULT INCORRECT.
7065 030220   000401     BR      6$
7066 030222   104001     ERROR  1  ;REPORT FPS INCORRECT.
7067 030224
7068
7069           ;EXP=27 (OCT)  FL=1  FIC=1
7070 030224   WWC8:
7071 030224   104414           LPERR
7072 030226   004737 031042     JSR      PC,@#STCSUB  ;SET UP THE LOOP ON ERROR ADDRESS.
7073 030232   045600 000001 000000 1$:  .WORD  45600,1,0,0  ;GO EXECUTE THE INSTRUCTION.
7074 030240   000000           ;ACO OPERAND.
7075 030242   000100 000000     2$:  .WORD  100,0  ;EXPECTED RESULT.
7076 030246   177777 177777     3$:  .WORD  -1,-1  ;ANTICIPATED ERRONEOUS RESULT.
7077 030252   040700           4$:  40700  ;FPS BEFORE EXECUTION.
7078 030254   040700           40700  ;FPS AFTER EXECUTION.
7079 030256   177777           -1      ;ANTICIPATED ERRONEOUS FPS.
7080 030260   177777           -1      ;EXPECTED FEC.
7081 030262   104001     5$:  ERROR  1  ;REPORT RESULT INCORRECT.
7082 030264   000401     BR      6$
7083 030266   104001     ERROR  1  ;REPORT FPS INCORRECT.
7084 030270
7085
7086           ;EXP=17 (OCT)  FL=0  FIC=1
7087 030270   WWC9:
7088 030270   104414           LPERR
7089 030272   004737 031042     JSR      PC,@#STCSUB  ;SET UP THE LOOP ON ERROR ADDRESS.
7090 030276   043600 000000 000000 1$:  .WORD  43600,0,0,0  ;GO EXECUTE THE INSTRUCTION.
7091 030304   000000           ;ACO OPERAND.
7092 030306   040000 177777     2$:  .WORD  40000,-1  ;EXPECTED RESULT.
7093 030312   000000 177777     3$:  .WORD  0,-1  ;ANTICIPATED ERRONEOUS RESULT.
7094 030316   040600           4$:  40600  ;FPS BEFORE EXECUTION.
7095 030320   040600           40600  ;FPS AFTER EXECUTION.
7096 030322   140604           140604  ;ANTICIPATED ERRONEOUS FPS.
7097 030324   177777           -1      ;EXPECTED FEC.
7098 030326   104001     5$:  ERROR  1  ;BAD CONSTANT ST 066
7099 030330   000401     BR      6$
7100 030332   104001     ERROR  1  ;REPORT FPS INCORRECT.
7101 030334
7102
7103           ;EXP=20 (OCT)  FL=0  FIC=1
7104 030334   WWC10:
7105 030334   104414           LPERR
7106 030336   004737 031042     JSR      PC,@#STCSUB  ;SET UP THE LOOP ON ERROR ADDRESS.
  
```

7107	030342	044000	000000	000000	1\$:	.WORD	44000,0,0,0	:ACO OPERAND.
7108	030350	000000						
7109	030352	000000	177777		2\$:	.WORD	0,-1	:EXPECTED RESULT.
7110	030356	177777	177777		3\$:	.WORD	-1,-1	:ANTICIPATED ERRONEOUS RESULT.
7111	030362	040600			4\$:		40600	:FPS BEFORE EXECUTION.
7112	030364	140605					140605	:FPS AFTER EXECUTION.
7113	030366	040600					40600	:ANTICIPATED ERRONEOUS FPS.
7114	030370	000006					6	:EXPECTED FEC.
7115	030372	104001			5\$:	ERROR	1	:REPORT RESULT INCORRECT.
7116	030374	000401				BR	6\$	
7117	030376	104001				ERROR	1	:BAD CONSTANT ST 066
7118	030400				6\$:			
7119								
7120								:EXP=10 (OCT), AC NEGATIVE, FL=0, FIC=1
7121	030400							WWC11:
7122	030400	104414				LPERR		:SET UP THE LOOP ON ERROR ADDRESS.
7123	030402	004737	031042			JSR	PC, @STCSUB	:GO EXECUTE THE INSTRUCTION.
7124	030406	142000	000000	000000	1\$:	.WORD	142000,0,0,0	:ACO OPERAND.
7125	030414	000000						
7126	030416	177600	177777		2\$:	.WORD	177600,-1	:EXPECTED RESULT.
7127	030422	000200	000000		3\$:	.WORD	200,0	:ANTICIPATED ERRONEOUS RESULT
7128	030426	040600			4\$:		40600	:FPS BEFORE EXECUTION.
7129	030430	040610					40610	:FPS AFTER EXECUTION.
7130	030432	040600					40600	:ANTICIPATED ERRONEOUS FPS.
7131	030434	177777					-1	:EXPECTED FEC.
7132	030436	104001			5\$:	ERROR	1	:(BUT ENBT) ST 632
7133	030440	000401				BR	6\$	
7134	030442	104001				ERROR	1	:(SET FN) ST 473
7135	030444				6\$:			
7136								
7137								:EXP 37 (OCT), FL-1, FIC=1, AC NEG.
7138	030444							WWC12:
7139	030444	104414				LPERR		:SET UP THE LOOP ON ERROR ADDRESS.
7140	030446	004737	031042			JSR	PC, @STCSUB	:GO EXECUTE THE INSTRUCTION.
7141	030452	147600	000000	000000	1\$:	.WORD	147600,0,0,0	:ACO OPERAND.
7142	030460	000000						
7143	030462	140000	000000		2\$:	.WORD	140000,0	:EXPECTED RESULT.
7144	030466	137777	000000		3\$:	.WORD	137777,0	:ANTICIPATED ERRONEOUS RESULT.
7145	030472	040700			4\$:		40700	:FPS BEFORE EXECUTION.
7146	030474	040710					40710	:FPS AFTER EXECUTION.
7147	030476	177777					-1	:ANTICIPATED ERRONEOUS FPS.
7148	030500	177777					-1	:EXPECTED FEC.
7149	030502	104001			5\$:	ERROR	1	:(BUT COUT) ST 375
7150	030504	000401				BR	6\$:ST 275 TO 074
7151	030506	104001				ERROR	1	:INTO 274
7152	030510				6\$:			
7153								
7154								:EXP-37 (OCT), FL=1, FIC=1, AC NEG
7155	030510							WWC13:
7156	030510	104414				LPERR		:SET UP THE LOOP ON ERROR ADDRESS.
7157	030512	004737	031042			JSR	PC, @STCSUB	:GO EXECUTE THE INSTRUCTION.
7158	030516	147600	000000	001000	1\$:	.WORD	147600,0,1000,0	:ACO OPERAND.
7159	030524	000000						
7160	030526	137777	177777		2\$:	.WORD	137777,177777	:EXPECTED RESULT.
7161	030532	140000	177777		3\$:	.WORD	140000,177777	:ANTICIPATED ERRONEOUS RESULT.
7162	030536	040707			4\$:		40707	:FPS BEFORE EXECUTION.

7163	030540	040710				40710			:FPS AFTER EXECUTION.
7164	030542	177777				-1			:ANTICIPATED ERRONEOUS FPS.
7165	030544	177777				-1			:EXPECTED FEC.
7166	030546	104001			5\$:	ERROR	1		:(BUT COUT) ST 375
7167	030550	000401				BR	6\$:TO 274 INTO 074
7168	030552	104001				ERROR	1		:REPORT FPS INCORRECT.
7169	030554				6\$:				
7170									
7171									
7172	030554								:EXP=41 (OCT), AC NEG, FL=1, FIC=1
7173	030554	104414				LPERR			WVC14: :SET UP THE LOOP ON ERROR ADDRESS.
7174	030556	004737	031042			JSR	PC, @STCSUB		:GO EXECUTE THE INSTRUCTION.
7175	030562	150200	000000	000000	1\$:	.WORD	150200,0,0,0		:ACO OPERAND.
7176	030570	000000							
7177	030572	000000	000000		2\$:	.WORD	0,0		:EXPECTED RESULT.
7178	030576	177777	177777		3\$:	.WORD	-1,-1		:ANTICIPATED ERRONEOUS RESULT.
7179	030602	040700			4\$:	40700			:FPS BEFORE EXECUTION.
7180	030604	140705				140705			:FPS AFTER EXECUTION.
7181	030606	177777				-1			:ANTICIPATED ERRONEOUS FPS.
7182	030610	000006				6			:EXPECTED FEC.
7183	030612	104001			5\$:	ERROR	1		:REPORT RESULT INCORRECT.
7184	030614	000401				BR	6\$		
7185	030616	104001				ERROR	1		:(BUT EZBT) ST 377
7186	030620				6\$:				
7187									
7188	030620								:EXP 40 (OCT), AC NEG, FL=1, FIC=1
7189	030620	104414				LPERR			WVC15: :SET UP THE LOOP ON ERROR ADDRESS.
7190	030622	004737	031042			JSR	PC, @STCSUB		:GO EXECUTE THE INSTRUCTION.
7191	030626	150000	000001	000000	1\$:	.WORD	150000,1,0,0		:ALO OPERAND.
7192	030634	000000							
7193	030636	000000	000000		2\$:	.WORD	0,0		:EXPECTED RESULT.
7194	030642	100000	177600		3\$:	.WORD	100000,-200		:ANTICIPATED ERRONEOUS RESULT.
7195	030646	040700			4\$:	40700			:FPS BEFORE EXECUTION.
7196	030650	140705				140705			:FPS AFTER EXECUTION.
7197	030652	040700				40700			:ANTICIPATED ERRONEOUS FPS.
7198	030654	000006				6			:EXPECTED FEC.
7199	030656	104001			5\$:	ERROR	1		:(BUT COUT) ST 360
7200	030660	000401				BR	6\$:TO 654 INTO 454
7201	030662	104001				ERROR	1		:REPORT FPS INCORRECT.
7202	030664				6\$:				
7203									
7204									
7205	030664								:EXP=40, AC NEGATIVE, FL=1, FIC=1
7206	030664	104414				LPERR			WVC16: :SET UP THE LOOP ON ERROR ADDRESS.
7207	030666	004737	031042			JSR	PC, @STCSUB		:GO EXECUTE THE INSTRUCTION.
7208	030672	150001	000000	000000	1\$:	.WORD	150001,0,0,0		:ACO OPERAND.
7209	030700	000000							
7210	030702	000000	000000		2\$:	.WORD	0,0		:EXPECTED RESULT.
7211	030706	077400	000000		3\$:	.WORD	77400,0		:ANTICIPATED ERRONEOUS RESULT.
7212	030712	040700			4\$:	40700			:FPS BEFORE EXECUTION.
7213	030714	140705				140705			:FPS AFTER EXECUTION.
7214	030716	177777				-1			:ANTICIPATED ERRONEOUS FPS.
7215	030720	000006				6			:EXPECTED FEC.
7216	030722	104001			5\$:	ERROR	1		:REPORT RESULT INCORRECT.
7217	030724	000401				BR	6\$		
7218	030726	104001				ERROR	1		:REPORT FPS INCORRECT.

```

7219 030730      6$:
7220
7221
7222      :EXP 40      (OCT), AC MOST NEG LONG INT, FL 1
7223      :FIC=1
7224 030730      WWC17:
7225 030730      104414      LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.
7226 030732      004737      031042      JSR      PC,@STCSUB ;GO EXECUTE THE INSTRUCTION.
7227 030736      150000      000000      000000 1$:      .WORD      150000,0,0,0 ;ACO OPERAND.
7228 030744      000000
7229 030746      100000      000000      2$:      .WORD      100000,0 ;EXPECTED RESULT.
7230 030752      000000      000000      3$:      .WORD      0,0 ;ANTICIPATED ERRONEOUS RESULT.
7231 030756      040700      4$:      40700 ;FPS BEFORE EXECUTION.
7232 030760      040710      40710 ;FPS AFTER EXECUTION.
7233 030762      140705      140705 ;ANTICIPATED ERRONEOUS FPS.
7234 030764      177777      -1 ;EXPECTED FEC.
7235 030766      104001      5$:      ERROR      1 ;(BUT NBIT) ST 654
7236 030770      000401      BR      6$ ;OR (BUT COUT) ST 454
7237 030772      104001      ERROR      1 ;REPORT FPS INCORRECT.
7238 030774      6$:
7239
7240      :EXP-20, AC - MOST NEG INTEGER, FL-0, FIC 1
7241
7242 030774      WWC18:
7243 030774      104414      LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.
7244 030776      004737      031042      JSR      PC,@STCSUB ;GO EXECUTE THE INSTRUCTION.
7245 031002      144000      000001      000000 1$:      .WORD      144000,1,0,0 ;ACO OPERAND.
7246 031010      000000
7247 031012      100000      177777      2$:      .WORD      100000,-1 ;EXPECTED RESULT.
7248 031016      100000      177400      3$:      .WORD      100000,177400 ;ANTICIPATED ERRONEOUS RESULT.
7249 031022      040600      4$:      40600 ;FPS BEFORE EXECUTION.
7250 031024      040610      40610 ;FPS AFTER EXECUTION.
7251 031026      140605      140605 ;ANTICIPATED ERRONEOUS FPS.
7252 031030      177777      -1 ;EXPECTED FEC.
7253 031032      104001      5$:      ERROR      1 ;(BUT FL) ST 633
7254 031034      000401      BR      6$ ;TO 655 INTO 654
7255 031036      104001      ERROR      1 ;REPORT FPS INCORRECT.
7256
7257 031040      000534      6$:      BR      WWC18
7258
7259      ;THIS SUBROUTINE, STCSUB, IS USED TO SET UP THE OPERANDS, EXECUTE
7260      ;THE STCDI OR STCDL INSTRUCTION AND CHECK THE RESULTS. A CALL
7261      ;TO IT IS MADE THUS:
7262      :
7263      :
7264      :      JSR      PC,@STCSUB
7265      :      ACARG:  .WORD      X,X,X,X ;AC OPERAND
7266      :      RES:      .WORD      X,X ;EXPECTED RESULT
7267      :      ERRES:  .WORD      X,X ;ERROR RESULT
7268      :      FPSB:   .WORD      X ;FPS BEFORE EXECUTION
7269      :      FPSA:   .WORD      X ;FPS AFTER EXECUTION
7270      :      ERFPS:  .WORD      X ;ERROR FPS.
7271      :      FEC:    .WORD      X ;EXPECTED FEC
7272      :      ERR1:  ERROR      1;DATA ERROR.
7273      :      BR      CONT
7274      :      ERR2:  ERROR      1;FPS ERROR.
7275      :      CONT:  ;RETURN ADDRESS

```

```

7275
7276      ;THE OPERANDS ARE SET UP (USING ACO AS THE ACCUMULATOR). THEN
7277      ;THE STCDI OR STCDL INSTRUCTION IS EXECUTED.
7278      ;THE RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
7279      ;COMPARED WITH FPSA IF THIS TOO IS CORRECT STCSUB RETURNS CONTROL
7280      ;TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD STCSUB
7281      ;COMPARE IT TO ERROR FPS. IF THIS MATCHES THEN STCSUB WILL RETURN
7282      ;TO THE ERROR CALL AT ERR2, OTHERWISE STCSUB ITSELF
7283      ;REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE
7284      ;STCDI OR STCDL IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
7285      ;ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
7286      ;THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN STCSUB
7287      ;WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR1. OTHERWISE THE
7288      ;RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND STCSUB WILL
7289      ;REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.
7290
7291 031042 012601      STCSUB: MOV      (SP)+,R1      ;GET A POINTER TO THE ARGUMENTS.
7292 031044 012700 000200      MOV      #200,R0      ;SET UP THE ACO OPERAND.
7293 031050 170100      LDFPS   R0
7294 031052 010100      MOV      R1,R0
7295 031054 172410      LDD     (R0),ACO
7296 031056 012702 031322      MOV      #STCIBF,R2      ;INITIALIZE THE OUT PUT BUFFER.
7297 031062 012700 000004      MOV      #4,R0
7298 031066 012722 177777      1$:     MOV      #-1,(R2)+
7299 031072 077003      SOB     R0,1$
7300 031074 016100 000020      MOV      20(R1),R0      ;SET THE FPS.
7301 031100 170100      LDFPS   R0
7302 031102 012737 031114 001236      MOV      #2$,@#STMP2
7303 031110 012700 031322      MOV      #STCIBF,R0
7304 031114 175410      2$:     STCDL   ACO,(R0)      ;TEST INSTRUCTION.
7305
7306 031116 170204      STFPS   R4      ;GET THE FPS.
7307 031120 170305      STST   R5      ;GET THE FEC.
7308 031122 010102      MOV      R1,R2
7309 031124 010237 001240      MOV      R2,@#STMP3
7310 031130 062702 000010      ADL     #10,R2
7311 031134 010237 001244      MOV      R2,@#STMP5
7312 031140 012737 031322 001242      MOV      #STCIBF,@#STMP4
7313 031146 010437 001250      MOV      R4,@#STMP7
7314 031152 016137 000022 001252      MOV      22(R1),@#STMP10
7315 031160 010102      MOV      R1,R2
7316 031162 062702 000010      ADD     #10,R2
7317 031166 012700 031322      MOV      #STCIBF,R0      ;SEE IF THE RESULT IS CORRECT.
7318 031172 012703 000002      MOV      #2,R3
7319 031176 022022      3$:     CMP      (R0)+,(R2)+
7320 031200 001014      BNE     15$
7321 031202 077303      SOB     R3,3$
7322 031204 016102 000022      MOV      22(R1),R2
7323 031210 020204      CMP     R2,R4      ;SEE IF THE FPS IS CORRECT.
7324 031212 001025      BNE     20$      ;BRANCH IF INCORRECT.
7325 031214 005702      TST    R2
7326 031216 100003      BPL    4$
7327 031220 026105 000026      CMP     26(R1),R5      ;SEE IF THE FEC IS CORRECT.
7328 031224 001027      BNE     25$      ;BRANCH IF INCORRECT.
7329
7330 031226 000161 000036      4$:     JMP     36(R1)      ;RETURN.
  
```

7331
7332
7333 031232 010102
7334 031234 062702 000014
7335 031240 012700 031322
7336 031244 012703 000002
7337 031250 022022
7338 031252 001003
7339 031254 077303
7340 031256 000161 000030
7341 031262
7342
7343 031262 104001
7344 031264 000760
7345
7346
7347 031266 020461 000024
7348 031272 001002
7349 031274 000161 000034
7350 031300
7351
7352 031300 104001
7353 031302 000751
7354
7355
7356 031304 016137 000026 001256
7357 031312 010537 001254
7358 031316 104001
7359 031320 000742
7360
7361
7362 031322 177777 177777 177777
7363 031330 177777
7364
7365 031332
7366 031332 104413
7367
7368
7369
7370
7371
7372
7373
7374
7375
7376
7377
7378
7379
7380
7381 031334 000004
7382
7383
7384
7385 031336
7386 031336 104414

```
:DATA ERROR:  
:SEE IF THE FAILURE WAS ANTICIPATED.  
15$:  MOV R1,R2  
      ADD #14,R2  
      MOV #STCIBF,R0  
      MOV #2,R3  
16$:  CMP (R0)+,(R2)+  
      BNE 17$  
      SOB R3,16$  
      JMP 30(R1)  
17$:  
:FAILURE WAS NOT ANTICIPATED SO REPORT INCORRECT RESULT HERE.  
18$:  ERROR 1 ;DATA BAD  
      BR 4$  
:FPS INCORRECT, SO SEE IF FAILURE WAS ANTICIPATED.  
20$:  CMP R4,24(R1)  
      BNE 21$  
      JMP 34(R1)  
21$:  
:NOT ANTICIPATED SO REPORT BAD FPS HERE.  
22$:  ERROR 1 ;FPS BAD  
      BR 4$  
:REPORT INCORRECT FEC.  
25$:  MOV 26(R1),@#$TMP12  
      MOV R5,@#$TMP11  
26$:  ERROR 1  
      BR 4$  
:DATA BUFFER:  
STCIBF: .WORD -1,-1,-1,-1  
WWCDONE:  
      RSETUP ;GO INITIALIZE THE FPS AND STACK; AND  
              ;SFE IF THE USER HAS EXPRESSED  
              ;THE DESIRE TO CHANGE THE SOFTWARE  
              ;VIRTUAL CONSOLE SWITCH REGISTER (HAS  
              ;THE USER TYPED CONTROL G?).  
:*****  
:TEST 73 STCFL AND STCFI TEST  
:*****  
: THIS IS A TEST OF STCFL AND STCFI. IT  
: MAKES USE OF THE SAME SUBROUTINE, STCSUB,  
: WHICH WAS USED TO TEST STCDL AND STCDI.  
:*****  
TST73: SCOPE  
:EXPONENT=37, FL=1  
XXC1:  
      LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
```



```

7387 031340 004737 031042 JSR PC,@STCSUB ;GO EXECUTE THE INSTRUCTION.
7388 031344 047777 177777 177777 1$: .WORD 47777,-1,-1,-1 ;ACO OPERAND.
7389 031352 177777
7390 031354 077777 177600 2$: .WORD 77777,177600 ;EXPECTED RESULT.
7391 031360 077777 177777 3$: .WORD 77777,177777 ;ANTICIPATED ERRONEOUS RESULT.
7392 031364 040100 4$: 40100 ;FPS BEFORE EXECUTION.
7393 031366 040100 40100 ;FPS AFTER EXECUTION.
7394 031370 177777 -1 ;ANTICIPATED ERRONEOUS FPS.
7395 031372 177777 -1 ;EXPECTED FEC.
7396 031374 104001 5$: ERROR 1 ;X11(1,0)+0 ST 773X
7397 031376 000401 BR 6$
7398 031400 104001 ERROR 1 ;REPORT FPS INCORRECT.
7399 031402
7400
7401 031402
7402 031402 104413 XXCDONE:
7403 RSETUP
7404
7405
7406
7407
7408
7409
7410
7411
7412
7413
7414
7415
7416 031404 000004
7417
7418
7419 031406
7420 031406 104414
7421 031410 004737 031674 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
7422 031414 020000 000000 000000 1$: JSR PC,@STXSUB
7423 031422 000000 .WORD 20000,0,0,0 ;AC
7424 031424 177700 2$: -100 ;EXP RES
7425 031426 052525 3$: 52525 ;ERROR EXP.
7426 031430 040000 4$: 40000 ;FPSB
7427 031432 040010 40010 ;FPSA
7428 031434 040000 40000 ;ERROR FPS
7429 031436 104001 5$: ERROR 1 ;BAD EXP
7430 031440 000401 BR 6$
7431 031442 104001 ERROR 1 ;+(BUT ENBT) ST 376
7432 031444
7433
7434
7435 031444
7436 031444 104414
7437 031446 004737 031674 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
7438 031452 040000 000000 000000 1$: JSR PC,@STXSUB ;GO EXECUTE THE INSTRUCTION.
7439 031460 000000 .WORD 40000,0,0,0 ;ACO OPERAND.
7440 031462 000000 2$: 0 ;EXPECTED EXPONENT RESULT.
7441 031464 052525 3$: 52525 ;ANTICIPATED ERRONEOUS RESULT.
7442 031466 040000 4$: 40000 ;FPS BEFORE EXECUTION.

```

```

*****
*TEST 74 STEXP TEST
*
* THIS IS A TEST OF THE STEXP
* INSTRUCTION
*
*****
TST74: SCOPE

```

```

; EXP = 100 (EXCESS 200)
YYC1:

```

```

; EXP = 200 (EXCESS 200)
YYC2:

```

```
7443 031470 040004          40004          ;FPS AFTER EXECUTION.
7444 031472 040000          40000          ;ANTICIPATED ERRONEOUS FPS.
7445 031474 104001          5$: ERROR      1          ;REPORT RESULT INCORRECT.
7446 031476 000401          BR              6$
7447 031500 104001          ERROR          1          ;(BUT EZBT) ST 071
7448                                     ;TO 072 INT 272
7449 031502          6$:
7450
7451          ; EXP = 201 (EXCESS 200)
7452
7453 031502          YYC3:
7454 031502 104414          LPERR
7455 031504 004737 031674          JSR          PC,@#STXSUB ;SET UP THE LOOP ON ERROR ADDRESS.
7456 031510 040200 000000 000000 1$: .WORD      40200.0,0,0 ;GO EXECUTE THE INSTRUCTION.
7457 031516 000000          ;ACO OPERAND.
7458 031520 000001          2$: 1          ;EXPECTED EXPONENT RESULT.
7459 031522 052525          3$: 52525      ;ANTICIPATED ERRONEOUS RESULT.
7460 031524 040000          4$: 40000      ;FPS BEFORE EXECUTION.
7461 031526 040000          ;FPS AFTER EXECUTION.
7462 031530 040004          ;ANTICIPATED ERRONEOUS FPS.
7463 031532 104001          5$: ERROR      1          ;REPORT RESULT INCORRECT.
7464 031534 000401          BR              6$
7465 031536 104001          ERROR          1          ;(BUT EZBT) ST 071
7466 031540          6$:          ;TO 272 INTO 072
7467
7468          ; EXP = 375 (EXCESS 200)
7469
7470 031540          YYC4:
7471 031540 104414          LPERR
7472 031542 004737 031674          JSR          PC,@#STXSUB ;SET UP THE LOOP ON ERROR ADDRESS.
7473 031546 077200 000000 000000 1$: .WORD      77200.0,0,0 ;GO EXECUTE THE INSTRUCTION.
7474 031554 000000          ;ACO OPERAND.
7475 031556 000175          2$: 175        ;EXPECTED EXPONENT RESULT.
7476 031560 052525          3$: 52525      ;ANTICIPATED ERRONEOUS RESULT.
7477 031562 040000          4$: 40000      ;FPS BEFORE EXECUTION.
7478 031564 040000          ;FPS AFTER EXECUTION.
7479 031566 040010          ;ANTICIPATED ERRONEOUS FPS.
7480 031570 104001          5$: ERROR      1          ;REPORT RESULT INCORRECT.
7481 031572 000401          BR              6$
7482 031574 104001          ERROR          1          ;(BUT ENBT) ST 376
7483 031576          6$:          ;TO 471 INTO 071
7484
7485          ; EXP = 1 (EXCESS 200)
7486
7487 031576          YYC5:
7488 031576 104414          LPERR
7489 031600 004737 031674          JSR          PC,@#STXSUB ;SET UP THE LOOP ON ERROR ADDRESS.
7490 031604 000700 000000 000000 1$: .WORD      200.0,0,0 ;GO EXECUTE THE INSTRUCTION.
7491 031612 000000          ;ACO OPERAND.
7492 031614 177601          2$: -177       ;EXPECTED EXPONENT RESULT.
7493 031616 052525          3$: 52525      ;ANTICIPATED ERRONEOUS RESULT.
7494 031620 040000          4$: 40000      ;FPS BEFORE EXECUTION.
7495 031622 040010          ;FPS AFTER EXECUTION.
7496 031624 040000          ;ANTICIPATED ERRONEOUS FPS.
7497 031626 104001          5$: ERROR      1          ;REPORT RESULT INCORRECT.
7498 031630 000401          BR              6$
```

```
7499 031632 104001          ERROR 1          ;REPORT FPS INCORRECT.
7500 031634          6$:
7501
7502          ; EXP = 156 (EXCESS 200)
7503
7504 031634          YYC6:
7505 031634 104414          LPERP          ;SET UP THE LOOP ON ERROR ADDRESS.
7506 031636 004737 031674 000000 000000 1$: JSR PC,@#STXSUB ;GO EXECUTE THE INSTRUCTION.
7507 031642 033400          .WORD 33400,0,0,0 ;ACO OPERAND.
7508 031650 000000
7509 031652 177756          2$: -22          ;EXPECTED EXPONENT RESULT.
7510 031654 052525          3$: 52525          ;ANTICIPATED ERRONEOUS RESULT.
7511 031656 047707          4$: 47707          ;FPS BEFORE EXECUTION.
7512 031660 047710          47710          ;FPS AFTER EXECUTION.
7513 031662 177777          -1          ;ANTICIPATED ERRONEOUS FPS.
7514 031664 104001          5$: ERROR 1          ;REPORT RESULT INCORRECT.
7515 031666 000401          BR 6$
7516 031670 104001          ERROR 1          ;REPORT FPS INCORRECT.
7517
7518 031672 000510          6$: BR YYCDONE
7519
7520          ;THIS SUBROUTINE, STXSUB, IS USED TO SET UP THE OPERANDS, EXECUTE
7521          ;THE STEXP INSTRUCTION AND CHECK THE RESULTS. A CALL
7522          ;TO IT IS MADE THUS:
7523          :
7524          :
7525          : JSR PC,@#STXSUB
7526          : ACARG: .WORD X,X,X,X          ;AC OPERAND
7527          : RES: .WORD X          ;EXPECTED RESULT
7528          : ERRES: .WORD X          ;ERROR RESULT
7529          : FPSB: .WORD X          ;FPS BEFORE EXECUTION
7530          : FPSA: .WORD X          ;FPS AFTER EXECUTION
7531          : ERFPS: .WORD X          ;ERROR FPS.
7532          : ERR1: ERROR 1;DATA ERROR.
7533          : BR CONT
7534          : ERR2: ERROR 1;FPS ERROR.
7535          : CONT:          ;RETURN ADDRESS
7536          :
7537          ;THE OPERANDS ARE SET UP (USING ACO AS THE ACCUMULATOR). THEN
7538          ;THE STEXP INSTRUCTION IS EXECUTED.
7539          ;THE RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
7540          ;COMPARED WITH FPSA IF THIS TOO IS CORRECT STXSUB RETURNS CONTROL
7541          ;TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD STXSUB
7542          ;COMPARE IT TO ERROR FPS. IF THIS MATCHES THEN STXSUB WILL RETURN
7543          ;TO THE ERROR CALL AT ERR2, OTHERWISE STXSUB ITSELF
7544          ;REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE
7545          ;STEXP IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
7546          ;ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
7547          ;THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN STXSUB
7548          ;WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR1. OTHERWISE THE
7549          ;RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND STXSUB WILL
7550          ;REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.
7551 031674 012601          STXSUB: MOV (SP)+,R1          ;GET A POINTER TO THE ARGUMENTS.
7552 031676 010102          MOV R1,R2
7553 031700 010237 001240          MOV R2,@#TMP3
7554 031704 062702 000010          ADD #10,R2
```

```
7555 031710 012237 001244      MOV      (R2)+, @#STMP5
7556 031714 012737 031762 001236  MOV      #1$, @#STMP2
7557 031722 012737 123456 032102  MOV      #123456, @#STXBF
7558 031730 012737 076543 032104  MOV      #76543, @#STXBF+2
7559 031736 012700 000200      MOV      #200, R0
7560 031742 170100      LDFPS   R0
7561 031744 010100      MOV      R1, R0      ;SET UP THE ACO OPERAND.
7562 031746 172410      LDD      (R0), ACO
7563 031750 016100 000016      MOV      16(R1), R0  ;SET THE FPS.
7564 031754 170100      LDFPS   R0
7565 031756 012700 032102      MOV      #STXBF, R0
7566 031762 175010      1$:     STEXP      ACO, (R0)      ;TEST INSTRUCTION.
7567 031764 170204      STFPS   R4          ;GET FPS.
7568 031766 010437 001250      MOV      R4, @#STMP7
7569 031772 016137 000016 001252  MOV      16(R1), @#STMP10
7570 032000 013737 032102 001242  MOV      @#STXBF, @#STMP4
7571 032006 026137 000010 032102  CMP      10(R1), @#STXBF ;WAS RESULT CORRECT?
7572 032014 001411      BEQ      5$          ;BRANCH IF CORRECT.
7573 032016 026137 000012 032102  CMP      12(R1), @#STXBF ;OTHERWISE SEE IF THE FAILURE WAS ANTICIPATED.
7574 032024 001002      BNE      2$
7575 032026 000161 000022      JMP      22(R1)
7576
7577      ;IF NOT ANTICIPATED REPORT ERROR HERE.
7578 032032      2$:
7579 032032 104001      3$:     ERROR      1          ;EXP BAD
7580 032034 000161 000030      4$:     JMP      30(R1)
7581
7582 032040 020461 000016      5$:     CMP      R4, 16(R1)      ;SEE IF THE FPS IS CORRECT.
7583 032044 001407      BEQ      10$         ;BRANCH IF CORRECT.
7584 032046 020461 000020      CMP      R4, 20(R1)      ;SEE IF THE FAILURE WAS ANTICIPATED.
7585 032052 001002      BNE      6$
7586 032054 000161 000026      JMP      26(R1)
7587
7588      ;FPS ERROR WAS NOT ANTICIPATED SO REPORT ERROR HERE.
7589 032060      6$:
7590 032060 104001      7$:     ERROR      1          ;FPS BAD
7591 032062 000764      BR      4$
7592
7593      ;SEE IF MORE THAN ONE WORD WAS WRITTEN IN THE OUTPUT BUFFER.
7594 032064 022737 076543 032104 10$:    CMP      #76543, @#STXBF+2
7595 032072 001760      BEQ      4$
7596 032074 104001      11$:   ERROR      1          ;FDFL+0 ST 347X
7597 032076 000756      BR      4$
7598
7599 032100 177777      -1
7600 032102 177777 177777 177777 STXBF:  .WORD  -1,-1,-1,-1,-1
7601 032110 177777 177777
7602
7603 032114      YYCDONE:
7604 032114 104413      RSETUP      ;GO INITIALIZE THE FPS AND STACK; AND
7605      ;SEE IF THE USER HAS EXPRESSED
7606      ;THE DESIRE TO CHANGE THE SOFTWARE
7607      ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
7608      ;THE USER TYPED CONTROL G?).
7609
7610      ;:*****
```

```
7611      ;*TEST 75      STST TEST
7612      ;*
7613      ;* THIS IS A TEST OF THE STST
7614      ;* INSTRUCTION. FIRST AN ILLEGAL FPS OP CODE
7615      ;* (INSTRUCTION) IS USED TO ENTER AN
7616      ;* ERROR CONDITION IN THE FEC AND
7617      ;* FEA. THE STST IS EXECUTED AND
7618      ;* THE FEC AND FEA ARE CHECKED
7619      ;*
7620      ;*****
7621 032116 000004 TST75: SCOPE
7622
7623 032120 ZC1:
7624 032120 104414 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
7625 032122 012700 040000 MOV #40000,R0 ;SET FPS. FID=1.
7626 032126 170100 LDFPS R0
7627
7628 032130 170003 ZC2: .WORD 170003 ;ILLEGAL FPP
7629 ;OP CODE
7630 032132 012700 032306 MOV #ZC1,R0 ;SET UP THE OUTPUT BUFFER.
7631 032136 012710 177777 MOV #-1,(R0)
7632 032142 012760 177777 000002 MOV #-1,2(R0)
7633 032150 012737 032156 001236 ZC3: MOV #ZC3,@#STMP2
7634 032156 170310 STST (R0) ;GET FEC AND
7635 ;FEA
7636 032160 170204 STFPS R4 ;GET FPS.
7637 032162 012700 032306 MOV #ZC1,R0
7638 032166 011037 001240 MOV (R0),@#STMP3
7639 032172 016037 000002 001242 MOV 2(R0),@#STMP4
7640 032200 012737 000002 001244 MOV #2,@#STMP5
7641 032206 012737 032130 001246 MOV #ZC2,@#STMP6
7642 032214 010437 001250 MOV R4,@#STMP7
7643 032220 012737 140000 001252 MOV #140000,@#STMP10
7644
7645 032226 022710 000002 CMP #2,(R0) ;SEE IF FEC IS CORRECT.
7646 032232 001010 BNE ZC5 ;BRANCH IF INCORRECT.
7647 032234 022760 032130 000002 CMP #ZC2,2(R0) ;SEE IF FEA, ADDRESS, IS CORRECT.
7648 032242 001006 BNE ZC10 ;BRANCH IF INCORRECT.
7649 032244 022704 140000 CMP #140000,R4 ;SEE IF FPS IS CORRECT.
7650 032250 001013 BNE ZC15 ;BRANCH IF INCORRECT.
7651 032252 000422 BR ZCDONE
7652
7653 ;REPORT FEC INCORRECT
7654 032254 ZC5:
7655 032254 104001 1$: ERROR 1 ;STST BAD
7656 032256 000420 BR ZCDONE ;FEAX
7657
7658 ;REPORT FEA INCORRECT
7659 032260 022760 177777 000002 ZC10: CMP #-1,2(R0)
7660 032266 001402 BEQ ZC12
7661 032270 104001 1$: ERROR 1 ;STST BAD FEA
7662 032272 000412 BR ZCDONE
7663 032274 ZC12:
7664 032274 104001 1$: ERROR 1 ;SET FD FL ST 636
7665 032276 000410 BR ZCDONE
7666
```

```

7667 ;REPORT FPS INCORRECT
7668 032300 ZC15:
7669 032300 104001 1$: ERROR 1 ;FPS X AFTER ST ST
7670 032302 000406 BR ZZCDONE
7671
7672 ;DATA BUFFER:
7673 032304 177777 -1
7674 032306 177777 177777 177777 ZZCBF: .WORD -1,-1,-1,-1
7675 032314 177777
7676 032316 177777 -1
7677
7678 032320 012706 001100 ZZCDONE:MOV #STACK,SP ;SET UP STACK POINTER
7679 032324 104413 RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
7680 ;SEE IF THE USER HAS EXPRESSED
7681 ;THE DESIRE TO CHANGE THE SOFTWARE
7682 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
7683 ;THE USER TYPED CONTROL G?).
7684
7685
7686
7687
7688
7689
7690
7691
7692
7693
7694
7695
7696 032326 000004
7697
7698 032330 012746 144724 AAD1: MOV #144724, -(SP) ;PUT FRACTION ON STACK
7699 032334 012746 040600 MOV #40600, -(SP) ;PUT EXPONENT ON STACK
7700 032340 005046 CLR -(SP) ;PUT SUBTRAHEND FRACTION ON STACK
7701 032342 012746 040600 MOV #40600, -(SP) ;PUT SUBTRAHEND EXPONENT ON STACK
7702 032346 172466 000004 LDF 4(SP), ACO ;LOAD FP ACCUMULATORS
7703 032352 173026 SUBF (SP)+, ACO ;DO SUBTRACTION
7704 032354 174037 032404 STF ACO, @#AADBFB ;GET AND STORE ANSWER
7705 032360 022737 036711 032404 CMP #36711, @#AADBFB ;IS EXPONENT CORRECT
7706 032366 001401 BEQ 1$ ;IF YES GO CHECK FRACTION
7707 032370 104002 ERROR 2 ;BAD EXPONENT FROM SUBTRACTION
7708 032372 022737 152000 032406 1$: CMP #152000, @#AADBFB+2 ;IS FRACTION CORRECT
7709 032400 001403 BEQ AADDONE ;IF YES GO TO END OF TEST
7710 032402 104002 ERROR 2 ;FRACTION INCORRECT
7711
7712 032404 000000 AADBFB: .WORD 0
7713 032406 000000 .WORD 0
7714
7715 032410 012706 001100 AADDONE: MOV #STACK, SP ;RESTORE STACK POINTER
7716 032414 104413 RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
7717 ;SEE IF THE USER HAS EXPRESSED
7718 ;THE DESIRE TO CHANGE THE SOFTWARE
7719 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
7720 ;THE USER TYPED CONTROL G?).
7721
7722
    
```

```

:*****
:*TEST 76 SPECIAL CASE TEST
:*THIS TEST IS DERIVED FROM THE FORTRAN 4 BENCH MARKS
:*IT WAS FOUND THAT THIS CODE FAILED EVEN THOUGH THE
:*FP UNIT HAD PASSED THE DIAGNOSTICS. THE HARDWARE WAS
:*MODIFIED TO CURRECT THE ERROR BUT SINCE A TEST DEFICIENCY
:*WAS INDICATED THIS TEST WAS ADDED.
:*ALL THE TEST DOES IS PUT A MIXED NUMBER IN THE FPAC
:*AND SUBTRACTS A WHOLE NUMBER FROM IT.
:*****
TST76: SCOPE
    
```

```
7723
7724
7725
7726
7727
7728
7729
7730
7731 032416 000004
7732
7733 032420 032737 000001 001336 ZD1: BIT #1, @#SENV ;ARE WE ON APT
7734 032426 001403 BEQ ZD2 ;IF NO DO THIS TEST ALWAYS
7735 032430 005737 001324 TST @#SPASS ;IF YES THEN CHECK PASS COUNTER
7736 032434 001122 BNE EXITER ;AND ONLY DO IT ON FIRST PASS
7737 032436 005001 ZD2: CLR R1 ;INIALIZE A COUPLE OF COUNTERS
7738 032440 005000 CLR R0
7739 032442 172627 040400 LDF #2, AC2 ;MAKE SURE FP ACCUMALATOR NON-ZERO
7740 032446 016767 145412 146556 MOV TPVEC, $TMP0 ;SAVE INTERRUPT VECTOR
7741 032454 016767 145406 146552 MOV TPVEC+2, $TMP1 ;SAVE INTERRUPT PRIORITY
7742 032462 012767 032534 145374 MOV #3$, TPVEC ;SET UP VECTOR FOR THIS TEST
7743 032470 005067 145372 CLR TPVEC+2 ;SET NEW PRIORITY TO 0
7744 032474 005067 145276 CLR PS ;SET PROCESSOR PRIORITY TO 0
7745 032500 105077 146446 CLR @STPB ;SEND A CHARACTER
7746 032504 105777 146440 1$: TSTB @STPS ;WAIT FOR DONE ON THIS CHARACTER
7747 032510 100375 BPL 1$
7748 032512 105077 146434 CLRB @STPB ;SEND A SECOND CHARACTER
7749 032516 052777 000100 146424 BIS #BIT6, @STPS ;SET INTERRUPT ENABLE
7750 032524 005200 2$: INC R0 ;INCREMENT COUNT
7751 032526 001376 BNE 2$ ;IF NO INTERRUPT BEFORE 0 ERROR
7752 032530 000005 RESET ;CLEAR ALL BITS IN SLU
7753 032532 104003 ERROR 3 ;NO INTERRUPT OCCURRED
7754 032534 166700 000112 3$: SUB Y, R0 ;SUBTRACT TIME FOR FP INSTRUCTION
7755 032540 010067 000110 MOV R0, Z ;SAVE PRE LOOP COUNT
7756 032544 012767 032624 145312 MOV #7$, TPVEC ;SET UP VECTOR FOR INTERRUPT FROM FP
7757 032552 005100 4$: COM R0 ;MAKE COUNT NEGATIVE
7758 032554 005077 146370 CLR @STPS ;DON'T ALLOW INTERRUPTS ON FIRST CHARACTER
7759 032560 105077 146366 CLRB @STPB ;SEND FIRST CHARACTER
7760 032564 105777 146360 5$: TSTB @STPS ;WAIT FOR READY
7761 032570 100375 BPL 5$
7762 032572 105077 146354 CLRB @STPB ;SEND SECOND CHARACTER
7763 032576 052777 000100 146344 BIS #BIT6, @STPS ;SET INTERRUPT ENABLE
7764 032604 005200 6$: INC R0 ;DO PRE LOOP
7765 032606 001376 BNE 6$
7766 032610 171227 040400 10$: MULF #2, AC2 ;DO FP INSTRUCTION
7767 032614 171227 040400 MULF #2, AC2 ;JUST INCASE INTERRUPT HAPPENS TOO LATE
7768 032620 000005 RESET ;CLEAR ALL STATUS BITS IN SLU
7769 032622 104003 ERROR 3 ;NO INTERRUPT OCCURRED SO ERROR
7770 032624 022716 032610 7$: CMP #10$, (SP) ;IS PC ON STACK POINTING TO FP INSTRUCTION
7771 032630 001412 BEQ ZDDONE ;IF YES WE ARE DONE
7772 032632 062706 000004 ADD #4, SP ;UPDATE STACK POINTER
7773 032636 062767 000002 000010 ADD #2, Z ;ADD A LITTLE TIME TO PRE LOOP
7774 032644 016700 000004 MOV Z, R0 ;PUT NEW COUNT IN LOOP COUNTER
7775 032650 000740 BR 4$ ;DO IT ALL AGAIN
7776
7777 032652 000027 Y: .WORD 2,
7778 032654 000000 Z: .WORD 0
```

```
7779
7780 032656 ZZDDONE:
7781 032656 042777 000100 146264 BIC #BIT6, @STPS ;CLEAR INTERRUPT ENABLE BIT
7782 032664 016767 146342 145172 MOV $TMP0, TPVEC ;RESTORE INTERRUPT VECTOR
7783 032672 016767 146336 145166 MOV $TMP1, TPVEC+2 ;RESTORE INTERRUPT PRIORITY
7784 032700 104413 RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
7785 ;SEE IF THE USER HAS EXPRESSED
7786 ;THE DESIRE TO CHANGE THE SOFTWARE
7787 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
7788 ;THE USER TYPED CONTROL G?).
7789
7790 032702 012737 032724 000004 EXITER: MOV #1$, @#4 ;SET UP FOR TIMEOUT OF NO MULTI-TESTER
7791 032710 012737 000340 000006 MOV #340, @#6 ;
7792 032716 012737 000001 164000 MOV #1, @#164000 ;SET 'STOP' ON MULTI-TESTER
7793 032724 012737 000006 000004 1$: MOV #6, @#4 ;RESTORE TRAP CATCHER
7794 032732 005037 000006 CLR @#6
7795
7796 032736 TST100:
7797
7798
7799
7800 .SBTTL END OF PASS ROUTINE
7801
7802 ;*****
7803 ;*INCREMENT THE PASS NUMBER ($PASS)
7804 ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
7805 ;*TYPE 'END PASS #XXXXX' (WHERE XXXXX IS A DECIMAL NUMBER)
7806 ;*IF SW12=1 INHIBIT TRACE TRAP
7807 ;*IF THERES A MONITOR GO TO IT
7808 ;*IF THERE ISN'T JUMP TO LOOP
7809
7810 $EOP:
7811 032736 000004 SCOPE
7812 032740 005067 146136 CLR $TSTNM ;:ZERO THE TEST NUMBER
7813 032744 005067 146332 CLR $TIMES ;:ZERO THE NUMBER OF ITERATIONS
7814 032750 005267 146350 INC $PASS ;:INCREMENT THE PASS NUMBER
7815 032754 042767 100000 146342 BIC #100000, $PASS ;:DON'T ALLOW A NEG. NUMBER
7816 032762 005327 DEC (PC)+ ;:LOOP?
7817 032764 000001 $EOPCT: .WORD 1
7818 032766 003031 BGT $DOAGN ;:YES
7819 032770 012737 MOV (PC)+, @ (PC)+ ;:RESTORE COUNTER
7820 032772 000001 $ENDCT: .WORD 1
7821 032774 032764 $EOPCT
7822 032776 104401 033121 TYPE , $ENDMG ;:TYPE 'END PASS #'
7823 033002 016746 146316 MOV $PASS, -(SP) ;:SAVE $PASS FOR TYPEOUT
7824 033006 104405 TYPDS ;:GO TYPE--DECIMAL ASCII WITH SIGN
7825 033010 104401 033116 TYPE , $ENULL ;:TYPE A NULL CHARACTER
7826 033014 013700 000042 $GET42: MOV @#42, R0 ;:GET MONITOR ADDRESS
7827 033020 001414 BEQ $DOAGN ;:BRANCH IF NO MONITOR
7828 033022 005046 CLR -(SP) ;:INSURE THE 'T' BIT IS CLEAR
7829 033024 012746 033032 MOV # $CLR.T, -(SP) ;:SETUP FOR AN RTI OR RTT
7830 033030 000426 BR $RTRN ;:GO DO AN RTI OR RTT TO LOAD THE PSW
7831 ;:WITH A CLEARED 'T' BIT
7832 033032 $CLR.T:
7833 033032 013700 000042 MOV @#42, R0 ;:INSURE R0 CONTAINS THE MONITORS
7834 033036 001405 BEQ $DOAGN ;:RETURN ADDRESS
```



```
7835 033040 000005          RESET          ;;CLEAR THE WORLD
7836 033042 004710          $ENDAD: JSR    PC,(R0)      ;;GO TO MONITOR
7837 033044 000240          NOP              ;;SAVE ROOM
7838 033046 000240          NOP              ;;FOR
7839 033050 000240          NOP              ;;ACT11
7840 033052          $DOAGN:
7841 033052 104400          TRAP           ;;PUSH OLD PSW AND PC ON STACK
7842 033054 042716 000020    BIC    #20,(SP)      ;;CLEAR THE 'T' BIT
7843 033060 032777 010000 146052 BIT    #BIT12,@SWR    ;;RUN WITH TRACE TRAP?
7844 033066 001005          BNE    1$          ;;BR IF NO
7845 033070 005167 000020    COM    $TBIT       ;;IS IT TIME FOR TRACE TRAP
7846 033074 100402          BMI    1$          ;;BR IF NO
7847 033076 052716 000020    BIS    #20,(SP)    ;;SET TRACE TRAP
7848 033102 012746 033110    1$:  MOV    # $LOOP,-(SP) ;;JUMP TO START OF TEST
7849 033106 000002          $RTRN: RTI          ;;RETURN--THIS IS CHANGED TO
7850                                     ;;AN 'RTT' IF 'RTT' IS A LEGAL
7851                                     ;;INSTRUCTION
7852 033110          $LOOP:
7853 033110 000137          JMP    @(PC)+       ;;RETURN
7854 033112 002056          $RTNAD: .WORD  LOOP
7855 033114 000000          $TBIT:  .WORD  0
7856 033116      377      377      000    $ENULL: .BYTE  -1,-1,0 ;;'T' BIT STATE INDICATOR
7857 033121      015    042412 042116    $ENDMG: .ASCIZ <15><12>/END PASS #/
7858 033126 050040 051501 020123
7859 033134 000043
7860
7861          .SBTTL SCOPE HANDLER ROUTINE
7862
7863          ;*****
7864          ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
7865          ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
7866          ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
7867          ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
7868          ;*SW14=1      LOOP ON TEST
7869          ;*SW11=1      INHIBIT ITERATIONS
7870          ;*SW09=1     LOOP ON ERROR
7871          ;*SW08=1     LOOP ON TEST IN SWR<7:0>
7872          ;*CALL
7873          ;*      SCOPE          ;;SCOPE=IOT
7874
7875          $SCOPE:
7876 033136 104407          CKSWR
7877 033140 032777 040000 145772 1$:  BIT    #BIT14,@SWR    ;;TEST FOR CHANGE IN SOFT-SWR
7878 033146 001114          BNE    $OVER       ;;LOOP ON PRESENT TEST?
7879                                     ;;YES IF SW14=1
7880 033150 000416          ;*****START OF CODE FOR THE XOR TESTER*****
7881          $XTSTR: BR    6$          ;;IF RUNNING ON THE 'XOR' TESTER CHANGE
7882          MOV    @ERRVEC,-(SP)     ;;THIS INSTRUCTION TO A 'NOP' (NOP=240)
7883          MOV    #5$,@ERRVEC      ;;SAVE THE CONTENTS OF THE ERROR VECTOR
7884          TST    @177060          ;;SET FOR TIMEOUT
7885          MOV    (SP)+,@ERRVEC    ;;TIME OUT ON XOR?
7886          BR    $SVLAD           ;;RESTORE THE ERROR VECTOR
7887          CMP    (SP)+,(SP)+      ;;GO TO THE NEXT TEST
7888          MOV    (SP)+,@ERRVEC    ;;CLEAR THE STACK AFTER A TIME OUT
7889          BR    7$              ;;RESTORE THE ERROR VECTOR
7890          6$:;*****END OF CODE FOR THE XOR TESTER*****
7890 033206 000423          7$:  BR    7$          ;;LOOP ON THE PRESENT TEST
```

```

7891 033206 032777 000400 145724 BIT #BIT08,@SWR ;;LOOP ON SPEC. TEST?
7892 033214 001404 BEQ 2$ ;;BR IF NO
7893 033216 127767 145716 145656 (MPB @SWR,$STNM ;;ON THE RIGHT TEST? SWR<7:0>
7894 033224 001465 BEQ $OVER ;;BR IF YES
7895 033226 105767 145651 2$: TSTB $ERFLG ;;HAS AN ERROR OCCURRED?
7896 033232 001421 BEQ 3$ ;;BR IF NO
7897 033234 126767 145655 145641 (MPB $ERMAX,$ERFLG ;;MAX. ERRORS FOR THIS TEST OCCURRED?
7898 033242 101015 BHI 3$ ;;BR IF NO
7899 033244 032777 001000 145666 BIT #BIT09,@SWR ;;LOOP ON ERROR?
7900 033252 001404 BEQ 4$ ;;BR IF NO
7901 033254 016767 145630 145624 7$: MOV $LPERR,$LPADR ;;SFT LOOP ADDRESS TO LAST SCOPE
7902 033262 000446 BR $OVER
7903 033264 105067 145613 4$: CLRB $ERFLG ;;ZERO THE ERROR FLAG
7904 033270 005067 146006 CLR $TIMES ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
7905 033274 000415 BR 1$ ;;ESCAPE TO THE NEXT TEST
7906 033276 032777 004000 145634 3$: BIT #BIT11,@SWR ;;INHIBIT ITERATIONS?
7907 033304 001011 BNE 1$ ;;BR IF YES
7908 033306 005767 146012 TST $PASS ;;IF FIRST PASS OF PROGRAM
7909 033312 001406 BEQ 1$ ;; INHIBIT ITERATIONS
7910 033314 005267 145564 INC $ICNT ;;INCREMENT ITERATION COUNT
7911 033320 026767 145756 145556 (MP $TIMES,$ICNT ;;CHECK THE NUMBER OF ITERATIONS MADE
7912 033326 002024 BGE $OVER ;;BR IF MORE ITERATION REQUIRED
7913 033330 012767 000001 145546 1$: MOV #1,$ICNT ;;REINITIALIZE THE ITERATION COUNTER
7914 033336 016767 000052 145736 MOV $MXCNT,$TIMES ;;SET NUMBER OF ITERATIONS TO DO
7915 033344 105267 145532 $SVLAD: INCB $STNM ;;COUNT TEST NUMBERS
7916 033350 116767 145526 145744 MOVB $STNM,$TESTN ;;SET TEST NUMBER IN APT MAILBOX
7917 033356 011667 145524 MOV (SP),$LPADR ;;SAVE SCOPE LOOP ADDRESS
7918 033362 011667 145522 MOV (SP),$LPERR ;;SAVE ERROR LOOP ADDRESS
7919 033366 005067 145712 CLR $ESCAPE ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
7920 033372 112767 000001 145515 MOVB #1,$ERMAX ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
7921 033400 016777 145476 145534 $OVER: MOV $STNM,@DISPLAY ;;DISPLAY TEST NUMBER
7922 033406 016716 145474 MOV $LPADR,(SP) ;;FUDGE RETURN ADDRESS
7923 033412 000002 RTI ;;FIXES PS
7924 033414 000001 $MXCNT: 1 ;;MAX. NUMBER OF ITERATIONS
7925
7926 .SBTTL ERROR HANDLER ROUTINE
7927
7928 ;*****
7929 ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
7930 ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
7931 ;*AND GO TO ERTYPE ON ERROR
7932 ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
7933 ;*SW15=1 HALT ON ERROR
7934 ;*SW13=1 INHIBIT ERROR TYPEOUTS
7935 ;*SW10 1 BELL ON ERROR
7936 ;*SW09-1 LOOP ON ERROR
7937 ;*CALL
7938 ;* ERROR N ;;ERROR EMT AND N-ERROR ITEM NUMBER
7939
7940 $ERROR:
7941 033416 104407 CKSWR ;;TEST FOR CHANGE IN SOFT-SWR
7942 033420 105267 145457 7$: INCB $ERFLG ;;SET THE ERROR FLAG
7943 033424 001775 BEQ 7$ ;;DON'T LET THE FLAG GO TO ZERO
7944 033426 016777 145450 145506 MOV $STNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
7945 033434 032777 002000 145476 BIT #BIT10,@SWR ;;BELL ON ERROR?
7946 033442 001402 BEQ 1$ ;;NO - SKIP
    
```

```
7947 033444 104401 001306          TYPE      ,SBELL          ;;RING BELL
7948 033450 005267 145436          1$:      INC      $ERTTL          ;;COUNT THE NUMBER OF ERRORS
7949 033454 011667 145436          MOV      (SP), $ERRPC          ;;GET ADDRESS OF ERROR INSTRUCTION
7950 033460 162767 000002 145430      SUB      #2, $ERRPC
7951 033466 117767 145424 145420      MOV      @ $ERRPC, $ITEMB          ;;STRIP AND SAVE THE ERROR ITEM CODE
7952 033474 032777 020000 145436      BIT      #BIT13, @SWR          ;;SKIP TYPEOUT IF SET
7953 033502 001004          BNE      20$          ;;SKIP TYPEOUTS
7954 033504 004767 002314          JSR      PC, ERTYPE          ;;GO TO USER ERROR ROUTINE
7955 033510 104401 001313          TYPE      , $CRLF
7956 033514          20$:
7957 033514 122767 000001 145614      CMP      #APTENV, $ENV          ;;RUNNING IN APT MODE
7958 033522 001007          BNE      2$          ;;NO, SKIP APT ERROR REPORT
7959 033524 116767 145364 000004      MOV      $ITEMB, 21$          ;;SET ITEM NUMBER AS ERROR NUMBER
7960 033532 004767 001126          JSR      PC, $ATY4          ;;REPORT FATAL ERROR TO APT
7961 033536          000          21$:      .BYTE      0
7962 033537          000          .BYTE      0
7963 033540 000777          22$:      BR      22$          ;;APT ERROR LOOP
7964 033542 005777 145372          2$:      TST      @SWR          ;;HALT ON ERROR
7965 033546 100002          BPL      3$          ;;SKIP IF CONTINUE
7966 033550 000000          HALT          ;;HALT ON ERROR!
7967 033552 104407          CKSWR          ;;TEST FOR CHANGE IN SOFT-SWR
7968 033554 032777 001000 145356      3$:      BIT      #BIT09, @SWR          ;;LOOP ON ERROR SWITCH SET?
7969 033562 001402          BEQ      4$          ;;BR IF NO
7970 033564 016716 145320          MOV      $LPERR, (SP)          ;;FUDGE RETURN FOR LOOPING
7971 033570 005767 145510          4$:      TST      $ESCAPE          ;;CHECK FOR AN ESCAPE ADDRESS
7972 033574 001402          BEQ      5$          ;;BR IF NONE
7973 033576 016716 145502          MOV      $ESCAPE, (SP)          ;;FUDGE RETURN ADDRESS FOR ESCAPE
7974 033602          5$:
7975 033602 022737 033042 000042      CMP      # $ENDAD, @#42          ;;ACT-11 AUTO-ACCEPT?
7976 033610 001001          BNE      6$          ;;BRANCH IF NO
7977 033612 000000          HALT          ;;YES
7978 033614          6$:
7979 033614 000002          RTI          ;;RETURN
7980
7981          .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
7982
7983          ;;*****
7984          ;;*SAVE R0-R5
7985          ;;*CALL:
7986          ;;*      SAVREG
7987          ;;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
7988          ;;*
7989          ;;*TOP---(+16)
7990          ;;* +2---(+18)
7991          ;;* +4---R5
7992          ;;* +6---R4
7993          ;;* +8---R3
7994          ;;*+10---R2
7995          ;;*+12---R1
7996          ;;*+14---R0
7997
7998          $SAVREG:
7999          033616 010046          MOV      R0, -(SP)          ;;PUSH R0 ON STACK
8000          033620 010146          MOV      R1, -(SP)          ;;PUSH R1 ON STACK
8001          033622 010246          MOV      R2, -(SP)          ;;PUSH R2 ON STACK
8002          033624 010346          MOV      R3, -(SP)          ;;PUSH R3 ON STACK
```

```
8003 033626 010446          MOV      R4,-(SP)      ;;PUSH R4 ON STACK
8004 033630 010546          MOV      R5,-(SP)      ;;PUSH R5 ON STACK
8005 033632 016646 000022    MOV      22(SP),-(SP)  ;;SAVE PS OF MAIN FLOW
8006 033636 016646 000022    MOV      22(SP),-(SP)  ;;SAVE PC OF MAIN FLOW
8007 033642 016646 000022    MOV      22(SP),-(SP)  ;;SAVE PS OF CALL
8008 033646 016646 000022    MOV      22(SP),-(SP)  ;;SAVE PC OF CALL
8009 033652 000002          RTI
8010
8011          ;*RESTORE R0-R5
8012          ;*CALL:
8013          ;*
8014          ;* RESREG
8014 033654          $RESREG:
8015 033654 012666 000022    MOV      (SP)+,22(SP)  ;;RESTORE PC OF CALL
8016 033660 012666 000022    MOV      (SP)+,22(SP)  ;;RESTORE PS OF CALL
8017 033664 012666 000022    MOV      (SP)+,22(SP)  ;;RESTORE PC OF MAIN FLOW
8018 033670 012666 000022    MOV      (SP)+,22(SP)  ;;RESTORE PS OF MAIN FLOW
8019 033674 012605          MOV      (SP)+,R5      ;;POP STACK INTO R5
8020 033676 012604          MOV      (SP)+,R4      ;;POP STACK INTO R4
8021 033700 012603          MOV      (SP)+,R3      ;;POP STACK INTO R3
8022 033702 012602          MOV      (SP)+,R2      ;;POP STACK INTO R2
8023 033704 012601          MOV      (SP)+,R1      ;;POP STACK INTO R1
8024 033706 012600          MOV      (SP)+,R0      ;;POP STACK INTO R0
8025 033710 000002          RTI
8026
8027          .SBTTL TYPE ROUTINE
8028
8029          ;*****
8030          ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
8031          ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
8032          ;*NOTE1:          $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
8033          ;*NOTE2:          $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
8034          ;*NOTE3:          $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
8035          ;*
8036          ;*CALL:
8037          ;*1) USING A TRAP INSTRUCTION
8038          ;*          TYPE          ,MESADR          ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
8039          ;*OR
8040          ;*          TYPE
8041          ;*          MESADR
8042          ;*
8043          ;*
8044 033712 105767 145241    $TYPE:  TSTB          $TPFLG          ;;IS THERE A TERMINAL?
8045 033716 100002          BPL          1$          ;;BR IF YES
8046 033720 000000          HALT          ;;HALT HERE IF NO TERMINAL
8047 033722 000430          BR          3$          ;;LEAVE
8048 033724 010046          1$:  MOV      R0,-(SP)      ;;SAVE R0
8049 033726 017600 000002    MOV      @2(SP),R0      ;;GET ADDRESS OF ASCIZ STRING
8050 033732 122767 000001 145376    CMPB     #APTENV,$ENV      ;;RUNNING IN APT MODE
8051 033740 001011          BNE          62$          ;;NO,GO CHECK FOR APT CONSOLE
8052 033742 132767 000100 145367    BITB     #APTSPOOL,$ENVM  ;;SPOOL MESSAGE TO APT
8053 033750 001405          BEQ          62$          ;;NO,GO CHECK FOR CONSOLE
8054 033752 010067 000004    MOV      R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
8055 033756 004767 000672    JSR      PC,$ATY3       ;;SPOOL MESSAGE TO APT
8056 033762 000000          61$:  .WORD          0          ;;MESSAGE ADDRESS
8057 033764 132767 000040 145345    62$:  BITB     #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
8058 033772 001003          BNE          60$          ;;YES,SKIP TYPE OUT
```

```
8059 033774 112046      2$:   MOVB   (RO)+,-(SP)   ;;PUSH CHARACTER TO BE TYPED ONTO STACK
8060 033776 001005      BNE    4$              ;;BR IF IT ISN'T THE TERMINATOR
8061 034000 005726      TST    (SP)+          ;;IF TERMINATOR POP IT OFF THE STACK
8062 034002 012600      60$:  MCV    (SP)+,R0    ;;RESTORE R0
8063 034004 062716 000002  3$:   ADD    #2,(SP)     ;;ADJUST RETURN PC
8064 034010 000002      RTI                    ;;RETURN
8065 034012 122716 000011  4$:   CMPB   #HT,(SP)    ;;BRANCH IF <HT>
8066 034016 001430      BEQ    8$              ;;BRANCH IF NOT <CRLF>
8067 034020 122716 000200      CMPB   #CRLF,(SP)    ;;BRANCH IF NOT <CRLF>
8068 034024 001006      BNE    5$              ;;BRANCH IF NOT <CRLF>
8069 034026 005726      TST    (SP)+          ;;POP <CR><LF> EQUIV
8070 034030 104401      TYPE                    ;;TYPE A CR AND LF
8071 034032 001313      $CRLF
8072 034034 105067 000130      CLRB   $CHARCNT      ;;CLEAR CHARACTER COUNT
8073 034040 000755      BR     2$              ;;GET NEXT CHARACTER
8074 034042 004767 000056  5$:   JSR    PC,$TYPEPC   ;;GO TYPE THIS CHARACTER
8075 034046 126726 145104  6$:   CMPB   $FILLC,(SP)+ ;;IS IT TIME FOR FILLER CHARS.?
8076 034052 001350      BNE    2$              ;;IF NO GO GET NEXT CHAR.
8077 034054 016746 145074      MOV    $NULL,-(SP)   ;;GET # OF FILLER CHARS. NEEDED
8078                                ;;AND THE NULL CHAR.
8079 034060 105366 000001  7$:   DECB   1(SP)        ;;DOES A NULL NEED TO BE TYPED?
8080 034064 002770      BLT    6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
8081 034066 004767 000032      JSR    PC,$TYPEPC   ;;GO TYPE A NULL
8082 034072 105367 000072      DECB   $CHARCNT     ;;DO NOT COUNT AS A COUNT
8083 034076 000770      BR     7$            ;;LOOP
```

;HORIZONTAL TAB PROCESSOR

```
8084
8085
8086
8087 034100 112716 000040  8$:   MOVB   #' ,(SP)    ;;REPLACE TAB WITH SPACE
8088 034104 004767 000014  9$:   JSR    PC,$TYPEPC   ;;TYPE A SPACE
8089 034110 132767 000007 000052  BITB   #7,$CHARCNT   ;;BRANCH IF NOT AT
8090 034116 001372      BNE    9$              ;;TAB STOP
8091 034120 005726      TST    (SP)+          ;;POP SPACE OFF STACK
8092 034122 000724      BR     2$              ;;GET NEXT CHARACTER
8093 034124 105777 145020  $TYPEPC: TSTB   @ $TPS     ;;WAIT UNTIL PRINTER IS READY
8094 034130 100375      BPL    $TYPEPC
8095 034132 116677 000002 145012  MOVB   2(SP),@$TPB   ;;LOAD CHAR TO BE TYPED INTO DATA REG.
8096 034140 122766 000015 000002  CMPB   #CR,2(SP)     ;;IS CHARACTER A CARRIAGE RETURN?
8097 034146 001003      BNE    1$              ;;BRANCH IF NO
8098 034150 105067 000014      CLRB   $CHARCNT     ;;YES--CLEAR CHARACTER COUNT
8099 034154 000406      BR     $TYPEPC
8100 034156 122766 000012 000002  1$:   CMPB   #LF,2(SP)    ;;IS CHARACTER A LINE FEED?
8101 034164 001402      BEQ    $TYPEPC      ;;BRANCH IF YES
8102 034166 105227      INCB   (PC)+        ;;COUNT THE CHARACTER
8103 034170 000000      $CHARCNT: .WORD 0   ;;CHARACTER COUNT STORAGE
8104 034172 000207      $TYPEPC: RTS    PC
```

.SBTTL BINARY TO OCTAL (ASCII) AND TYPE

```
8105
8106
8107
8108
8109 *****
8110 ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
8111 ;*OCTAL (ASCII) NUMBER AND TYPE IT.
8112 ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
8113 ;*CALL:
8114 ;*   MOV    NUM,-(SP)   ;;NUMBER TO BE TYPED
```

```

8115          : *      TYPOS          ::CALL FOR TYPEOUT
8116          : *      .BYTE      N          ::N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
8117          : *      .BYTE      M          ::M=1 OR 0
8118          : *                                     ::1=TYPE LEADING ZEROS
8119          : *                                     ::0=SUPPRESS LEADING ZEROS
8120          : *
8121          : *$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
8122          : *$TYPOS OR $TYPOC
8123          : *CALL:
8124          : *      MOV      NUM,-(SP)      ::NUMBER TO BE TYPED
8125          : *      TYPON                    ::CALL FOR TYPEOUT
8126          : *
8127          : *$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
8128          : *CALL:
8129          : *      MOV      NUM,-(SP)      ::NUMBER TO BE TYPED
8130          : *      TYPOC                    ::CALL FOR TYPEOUT
8131
8132 034174 017646 000000          $TYPOS: MOV      @(SP),-(SP)      ::PICKUP THE MODE
8133 034200 116667 000001 000211 MOVVB    1(SP), $OFILL      ::LOAD ZERO FILL SWITCH
8134 034206 112667 000207          MOVVB    (SP)+, $OMODE+1    ::NUMBER OF DIGITS TO TYPE
8135 034212 062716 000002          ADD      #2,(SP)          ::ADJUST RETURN ADDRESS
8136 034216 000406          BR      $TYPON
8137 034220 112767 000001 000171 $TYPOC: MOVVB    #1, $OFILL      ::SET THE ZERO FILL SWITCH
8138 034226 112767 000006 000165 MOVVB    #6, $OMODE+1    ::SET FOR SIX(6) DIGITS
8139 034234 112767 000005 000154 $TYPON: MOVVB    #5, $OCNT      ::SET THE ITERATION COUNT
8140 034242 010346          MOV      R3,-(SP)        ::SAVE R3
8141 034244 010446          MOV      R4,-(SP)        ::SAVE R4
8142 034246 010546          MOV      R5,-(SP)        ::SAVE R5
8143 034250 116704 000145          MOVVB    $OMODE+1,R4    ::GET THE NUMBER OF DIGITS TO TYPE
8144 034254 005404          NEG      R4
8145 034256 062704 000006          ADD      #6,R4          ::SUBTRACT IT FOR MAX. ALLOWED
8146 034262 110467 000132          MOVVB    R4, $OMODE      ::SAVE IT FOR USE
8147 034266 116704 000125          MOVVB    $OFILL,R4      ::GET THE ZERO FILL SWITCH
8148 034272 016605 000012          MOV      12(SP),R5      ::PICKUP THE INPUT NUMBER
8149 034276 005003          CLR      R3            ::CLEAR THE OUTPUT WORD
8150 034300 006105          1$: ROL      R5          ::ROTATE MSB INTO 'C'
8151 034302 000404          BR      3$            ::GO DO MSB
8152 034304 006105          2$: ROL      R5          ::FORM THIS DIGIT
8153 034306 006105          ROL      R5
8154 034310 006105          ROL      R5
8155 034312 010503          MOV      R5,R3
8156 034314 006103          3$: ROL      R3          ::GET LSB OF THIS DIGIT
8157 034316 105367 000076          DECB    $OMODE          ::TYPE THIS DIGIT?
8158 034322 100016          BPL     7$            ::BR IF NO
8159 034324 042703 177770          BIC     #177770,R3     ::GET RID OF JUNK
8160 034330 001002          BNE     4$            ::TEST FOR 0
8161 034332 005704          TST     R4            ::SUPPRESS THIS 0?
8162 034334 001403          BEQ     5$            ::BR IF YES
8163 034336 005204          4$: INC      R4          ::DON'T SUPPRESS ANYMORE 0'S
8164 034340 052703 000060          BIS     #'0,R3        ::MAKE THIS DIGIT ASCII
8165 034344 052703 000040          5$: BIS     #' ,R3      ::MAKE ASCII IF NOT ALREADY
8166 034350 110367 000040          MOVVB    R3,8$        ::SAVE FOR TYPING
8167 034354 104401 034414          TYPE    ,8$          ::GO TYPE THIS DIGIT
8168 034360 105367 000032          7$: DECB    $OCNT      ::COUNT BY 1
8169 034364 003347          BGT     2$            ::BR IF MORE TO DO
8170 034366 002402          BLT     6$            ::BR IF DONE
  
```

```

8171 034370 005204          INC      R4          ;;INSURE LAST DIGIT ISN'T A BLANK
8172 034372 000744          BR       2$          ;;GO DO THE LAST DIGIT
8173 034374 012605          6$:     MOV      (SP)+,R5      ;;RESTORE R5
8174 034376 012604          MOV      (SP)+,R4      ;;RESTORE R4
8175 034400 012603          MOV      (SP)+,R3      ;;RESTORE R3
8176 034402 016666 000002 000004 MOV      2(SP),4(SP)    ;;SET THE STACK FOR RETURNING
8177 034410 012616          MOV      (SP)+,(SP)
8178 034412 000002          RTI                    ;;RETURN
8179 034414      000          8$:     .BYTE   0          ;;STORAGE FOR ASCII DIGIT
8180 034415      000          .BYTE   0          ;;TERMINATOR FOR TYPE ROUTINE
8181 034416      000          $OCNT: .BYTE   0          ;;OCTAL DIGIT COUNTER
8182 034417      000          $OFILL: .BYTE   0        ;;ZERO FILL SWITCH
8183 034420 000000          $OMODE: .WORD   0        ;;NUMBER OF DIGITS TO TYPE
8184
8185          .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
8186
8187          ;;*****
8188          ;;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
8189          ;;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT.  DEPENDING ON WHETHER THE
8190          ;;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
8191          ;;*BEFORE THE FIRST DIGIT OF THE NUMBER.  LEADING ZEROS WILL ALWAYS BE
8192          ;;*REPLACED WITH SPACES.
8193          ;;*CALL:
8194          ;;*      MOV      NUM,-(SP)      ;;PUT THE BINARY NUMBER ON THE STACK
8195          ;;*      TYPDS                    ;;GO TO THE ROUTINE
8196
8197          $TYPDS:
8198          MOV      R0,-(SP)      ;;PUSH R0 ON STACK
8199          MOV      R1,-(SP)      ;;PUSH R1 ON STACK
8200          MOV      R2,-(SP)      ;;PUSH R2 ON STACK
8201          MOV      R3,-(SP)      ;;PUSH R3 ON STACK
8202          MOV      R5,-(SP)      ;;PUSH R5 ON STACK
8203          MOV      #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
8204          MOV      20(SP),R5      ;;GET THE INPUT NUMBER
8205          BPL      1$          ;;BR IF INPUT IS POS.
8206          NEG      R5          ;;MAKE THE BINARY NUMBER POS.
8207          MOVVB   #'-,1(SP)      ;;MAKE THE ASCII NUMBER NEG.
8208          CLR      R0          ;;ZERO THE CONSTANTS INDEX
8209          MOV      #$DBLK,R3      ;;SETUP THE OUTPUT POINTER
8210          MOVVB   #' ,(R3)+      ;;SET THE FIRST CHARACTER TO A BLANK
8211          CLR      R2          ;;CLEAR THE BCD NUMBER
8212          MOV      $DTBL(R0),R1  ;;GET THE CONSTANT
8213          SUB      R1,R5          ;;FORM THIS BCD DIGIT
8214          BLT      4$          ;;BR IF DONE
8215          INC      R2          ;;INCREASE THE BCD DIGIT BY 1
8216          BR      3$
8217          ADD      R1,R5          ;;ADD BACK THE CONSTANT
8218          TST      R2          ;;CHECK IF BCD DIGIT=0
8219          BNE      5$          ;;FALL THROUGH IF 0
8220          TSTB   (SP)          ;;STILL DOING LEADING 0'S?
8221          BMI      7$          ;;BR IF YES
8222          ASLB   (SP)          ;;MSD?
8223          BCC      6$          ;;BR IF NO
8224          MOVVB   1(SP),-1(R3)    ;;YES--SET THE SIGN
8225          BIS      #'0,R2        ;;MAKE THE BCD DIGIT ASCII
8226          BIS      #' ,R2        ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT

```

CJKDDA KEF11-A DIAG PART 2
CJKDDA.P11 20-JUN-79 11:22

MACY11 30A(1052) 20-JUN-79 11:39 PAGE 151
CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

SEQ 0151

```

8227 034542 110223      MOVB   R2,(R3)+      ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
8228 034544 005720      TST    (R0)+        ;;JUST INCREMENTING
8229 034546 020027 000010  CMP    RO,#10       ;;CHECK THE TABLE INDEX
8230 034552 002746      BLT    2$           ;;GO DO THE NEXT DIGIT
8231 034554 003002      BGT    8$           ;;GO TO EXIT
8232 034556 010502      MOV    R5,R2        ;;GET THE LSD
8233 034560 000764      BR     6$           ;;GO CHANGE TO ASCII
8234 034562 105726      8$:  TSTB   (SP)+      ;;WAS THE LSD THE FIRST NON-ZERO?
8235 034564 100003      BPL    9$           ;;BR IF NO
8236 034566 116663 177777 177776  MOVB   -1(SP),-2(R3) ;;YES--SET THE SIGN FOR TYPING
8237 034574 105013      9$:  CLRB   (R3)       ;;SET THE TERMINATOR
8238 034576 012605      MOV    (SP)+,R5     ;;POP STACK INTO R5
8239 034600 012603      MOV    (SP)+,R3     ;;POP STACK INTO R3
8240 034602 012602      MOV    (SP)+,R2     ;;POP STACK INTO R2
8241 034604 012601      MOV    (SP)+,R1     ;;POP STACK INTO R1
8242 034606 012600      MOV    (SP)+,R0     ;;POP STACK INTO R0
8243 034610 104401 034636      TYPE   $DBLK        ;;NOW TYPE THE NUMBER
8244 034614 016666 000002 000004  MOV    2(SP),4(SP)  ;;ADJUST THE STACK
8245 034622 012616      MOV    (SP)+,(SP)
8246 034624 000002      RTI                    ;;RETURN TO USER
8247 034626 023420      $DTBL: 10000.
8248 034630 001750      1000.
8249 034632 000144      100.
8250 034634 000012      10.
8251 034636 000004      $DBLK: .BLKW 4

```

8252
8253 .SBTTL APT COMMUNICATIONS ROUTINE
8254

```

8255 *****
8256 034646 112767 000001 000236 $ATY1: MOVB   #1,$FFLG      ;;TO REPORT FATAL ERROR
8257 034654 112767 000001 000226 $ATY3: MOVB   #1,$MFLG      ;;TO TYPE A MESSAGE
8258 034662 000403      BR     $ATYC
8259 034664 112767 000001 000220 $ATY4: MOVB   #1,$FFLG      ;;TO ONLY REPORT FATAL ERROR
8260 034672      $ATYC:
8261 034672 010046      MOV    RO,-(SP)     ;;PUSH RO ON STACK
8262 034674 010146      MOV    R1,-(SP)     ;;PUSH R1 ON STACK
8263 034676 105767 000206      TSTB   $MFLG        ;;SHOULD TYPE A MESSAGE?
8264 034702 001450      BEQ    5$           ;;IF NOT: BR
8265 034704 122767 000001 144424  CMPB   #APTENV,$ENV  ;;OPERATING UNDER APT?
8266 034712 001031      BNE    3$           ;;IF NOT: BR
8267 034714 132767 000100 144415  BITB   #APTPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
8268 034722 001425      BEQ    3$           ;;IF NOT: BR
8269 034724 017600 000004      MOV    @4(SP),R0    ;;GET MESSAGE ADDR.
8270 034730 062766 000002 000004  ADD    #2,4(SP)     ;;BUMP RETURN ADDR.
8271 034736 005767 144354      1$:  TST    $MSGTYPE    ;;SEE IF DONE W/ LAST XMISSION?
8272 034742 001375      BNE    1$           ;;IF NOT: WAIT
8273 034744 010067 144362      MOV    RO,$MSGAD    ;;PUT ADDR IN MAILBOX
8274 034750 105720      2$:  TSTB   (R0)+      ;;FIND END OF MESSAGE
8275 034752 001376      BNE    2$
8276 034754 166700 144352      SUB    $MSGAD,R0    ;;SUB START OF MESSAGE
8277 034760 006200      ASR    RO           ;;GET MESSAGE LNTH IN WORDS
8278 034762 010067 144346      MOV    RO,$MSGGLT   ;;PUT LENGTH IN MAILBOX
8279 034766 012767 000004 144322  MOV    #4,$MSGTYPE  ;;TELL APT TO TAKE MSG.
8280 034774 000413      BR     5$
8281 034776 017667 000004 000016  3$:  MOV    @4(SP),4$    ;;PUT MSG ADDR IN JSR LINKAGE
8282 035004 062766 000002 000004  ADD    #2,4(SP)     ;;BUMP RETURN ADDRESS

```



```
8283 035012 016746 142760      MOV      177776,-(SP)      ;;PUSH 177776 ON STACK
8284 035016 004767 176670      JSR      PC,$TYPE        ;;CALL TYPE MACRO
8285 035022 000000      4$:      .WORD          0
8286 035024      5$:
8287 035024 105767 000062      10$:     TSTB     $FFLG      ;;SHOULD REPORT FATAL ERROR?
8288 035030 001416      BEQ      12$             ;;IF NOT: BR
8289 035032 005767 144300      TST     $ENV            ;;RUNNING UNDER APT?
8290 035036 001413      BEQ      12$             ;;IF NOT: BR
8291 035040 005767 144252      11$:     TST     $MSGTYPE     ;;FINISHED LAST MESSAGE?
8292 035044 001375      BNE      11$             ;;IF NOT: WAIT
8293 035046 017667 000004 144244     MOV      @4(SP),$FATAL    ;;GET ERROR #
8294 035054 062766 000002 000004     ADD      #2,4(SP)        ;;BUMP RETURN ADDR.
8295 035062 005267 144230      INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
8296 035066 105067 000020      12$:     CLRB     $FFLG      ;;CLEAR FATAL FLAG
8297 035072 105067 000013      CLRB     $LFLG          ;;CLEAR LOG FLAG
8298 035076 105067 000006      CLRB     $MFLG          ;;CLEAR MESSAGE FLAG
8299 035102 012601      MOV      (SP)+,R1        ;;POP STACK INTO R1
8300 035104 012600      MOV      (SP)+,R0        ;;POP STACK INTO R0
8301 035106 000207      RTS      PC              ;;RETURN
8302 035110      000      $MFLG: .BYTE      0      ;;MESSG. FLAG
8303 035111      000      $LFLG: .BYTE      0      ;;LOG FLAG
8304 035112      000      $FFLG: .BYTE      0      ;;FATAL FLAG
8305      035114      .EVEN
8306      000200      APTSIZE=200
8307      000001      APTENV=001
8308      000100      APTSPool=100
8309      000040      APTCSUP=040
8310
8311      .SBTTL  TTY INPUT ROUTINE
8312
8313      ;:*****
8314      .ENABL  LSB
8315
8316      ;:*****
8317      ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
8318      ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
8319      ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
8320      ;*WHEN OPERATING IN TTY FLAG MODE.
8321 035114 022767 000176 144016 $CKSWR: CMP      #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
8322 035122 001074      BNE      15$             ;;BRANCH IF NO
8323 035124 105777 144014      TSTB     @$TKS           ;;CHAR THERE?
8324 035130 100071      BPL      15$             ;;IF NO, DON'T WAIT AROUND
8325 035132 117746 144010      MOV      @$TKB,-(SP)     ;;SAVE THE CHAR
8326 035136 042716 177600      BIC      #^C177,(SP)    ;;STRIP-OFF THE ASCII
8327 035142 022726 000007      CMP      #7,(SP)+       ;;IS IT A CONTROL G?
8328 035146 001062      BNE      15$             ;;NO, RETURN TO USER
8329 035150 126727 143760 000001      CMPB     $AUTOB,#1      ;;ARE WE RUNNING IN AUTO-MODE?
8330 035156 001456      BEQ      15$             ;;BRANCH IF YES
8331
8332 035160 104401 035523      TYPE     , $CNTLG        ;;ECHO THE CONTROL-G (^G)
8333 035164 104401 035530      $GTSWR: TYPE     , $MSWR      ;;TYPE CURRENT CONTENTS
8334 035170 016746 143002      MOV      SWREG,-(SP)    ;;SAVE SWREG FOR TYPEOUT
8335 035174 104402      TYPOC   , $MNEW         ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
8336 035176 104401 035541      TYPE     , $MNEW         ;;PROMPT FOR NEW SWR
8337 035202 005046      19$:     CLR      -(SP)        ;;CLEAR COUNTER
8338 035204 005046      CLR      -(SP)        ;;THE NEW SWR
```

```
8339 035206 105777 143732      7$:  TSTB  @STKS      ;;CHAR THERE?
8340 035212 100375              BPL  7$             ;;IF NOT TRY AGAIN
8341
8342 035214 117746 143726      MOVB  @STKB,-(SP)    ;;PICK UP CHAR
8343 035220 042716 177600      BIC  #'C177,(SP)   ;;MAKE IT 7-BIT ASCII
8344
8345
8346
8347 035224 021627 000025      9$:  CMP    (SP),#25  ;;IS IT A CONTROL-U?
8348 035230 001005              BNE  10$           ;;BRANCH IF NOT
8349 035232 104401 035516      TYPE  ,$CNTLU      ;;YES, ECHO CONTROL-U (^U)
8350 035236 062706 000006      20$: ADD    #6,SP     ;;IGNORE PREVIOUS INPUT
8351 035242 000757              BR   19$           ;;LET'S TRY IT AGAIN
8352
8353
8354 035244 021627 000015      10$: CMP    (SP),#15  ;;IS IT A <CR>?
8355 035250 001022              BNE  16$           ;;BRANCH IF NO
8356 035252 005766 000004      TST   4(SP)        ;;YES, IS IT THE FIRST CHAR?
8357 035256 001403              BEQ  11$           ;;BRANCH IF YES
8358 035260 016677 000002 143652  MOV    2(SP),@SWR   ;;SAVE NEW SWR
8359 035266 062706 000006      11$: ADD    #6,SP     ;;CLEAR UP STACK
8360 035272 104401 001313      14$: TYPE  ,$CRLF    ;;ECHO <CR> AND <LF>
8361 035276 126727 143633 000001  CMPB  $INTAG,#1    ;;RE-ENABLE TTY KBD INTERRUPTS?
8362 035304 001003              BNE  15$           ;;BRANCH IF NOT
8363 035306 012777 000100 143630  MOV    #100,@STKS  ;;RE-ENABLE TTY KBD INTERRUPTS
8364 035314 000002              RTI                    ;;RETURN
8365 035316 004767 176602      16$: JSR    PC,$TYPEC ;;ECHO CHAR
8366 035322 021627 000060      CMP    (SP),#60    ;;CHAR < 0?
8367 035326 002420              BLT  18$           ;;BRANCH IF YES
8368 035330 021627 000067      CMP    (SP),#67    ;;CHAR > 7?
8369 035334 003015              BGT  18$           ;;BRANCH IF YES
8370 035336 042726 000060      BIC    #60,(SP)+   ;;STRIP-OFF ASCII
8371 035342 005766 000002      TST   2(SP)        ;;IS THIS THE FIRST CHAR
8372 035346 001403              BEQ  17$           ;;BRANCH IF YES
8373 035350 006316              ASL  (SP)          ;;NO, SHIFT PRESENT
8374 035352 006316              ASL  (SP)          ;;CHAR OVER TO MAKE
8375 035354 006316              ASL  (SP)          ;;ROOM FOR NEW ONE.
8376 035356 005266 000002      17$: INC    2(SP)    ;;KEEP COUNT OF CHAR
8377 035362 056616 177776      BIS    -2(SP),(SP) ;;SET IN NEW CHAR
8378 035366 000707              BR   7$            ;;GET THE NEXT ONE
8379 035370 104401 001312      18$: TYPE  ,$QUES    ;;TYPE ?<CR><LF>
8380 035374 000720              BR   20$           ;;SIMULATE CONTROL-U
8381 .DSABL  LSB
8382
8383
8384 ;;*****
8385 ;;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
8386 ;;*CALL:
8387 ;;*      RDCHR      ;;INPUT A SINGLE CHARACTER FROM THE TTY
8388 ;;*      RETURN HERE ;;CHARACTER IS ON THE STACK
8389 ;;*              ;;WITH PARITY BIT STRIPPED OFF
8390 ;;*
8391 ;;*
8392 035376 011646 000004 000002 $RDCHR: MOV    (SP),-(SP) ;;PUSH DOWN THE PC
8393 035400 016666 000004 000002  MOV    4(SP),2(SP) ;;SAVE THE PS
8394 035406 105777 143532      1$:  TSTB  @STKS      ;;WAIT FOR
```

```
8395 035412 100375          BPL      1$          ;;A CHARACTER
8396 035414 117766 143526 000004  MOVB    @TKB,4(SP)  ;;READ THE TTY
8397 035422 042766 177600 000004  BIC     #^C<177>,4(SP) ;;GET RID OF JUNK IF ANY
8398 035430 026627 000004 000023  CMP     4(SP),#23   ;;IS IT A CONTROL-S?
8399 035436 001013          BNE     3$          ;;BRANCH IF NO
8400 035440 105777 143500      2$:  TSTB    @TKS          ;;WAIT FOR A CHARACTER
8401 035444 100375          BPL     2$          ;;LOOP UNTIL ITS THERE
8402 035446 117746 143474  MOVB    @TKB,-(SP)  ;;GET CHARACTER
8403 035452 042716 177600  BIC     #^C177,(SP) ;;MAKE IT 7-BIT ASCII
8404 035456 022627 000021  CMP     (SP)+,#21   ;;IS IT A CONTROL-Q?
8405 035462 001366          BNE     2$          ;;IF NOT DISCARD IT
8406 035464 000750          BR      1$          ;;YES, RESUME
8407 035466 026627 000004 000140  3$:  CMP     4(SP),#140 ;;IS IT UPPER CASE?
8408 035474 002407          BLT     4$          ;;BRANCH IF YES
8409 035476 026627 000004 000175  CMP     4(SP),#175  ;;IS IT A SPECIAL CHAR?
8410 035504 003003          BGT     4$          ;;BRANCH IF YES
8411 035506 042766 000040 000004  BIC     #40,4(SP)   ;;MAKE IT UPPER CASE
8412 035514 000002          RTI                    ;;GO BACK TO USER
8413 035516 052536 005015      000  $CNTLU: .ASCIZ /^U/<15><12> ;;CONTROL 'U'
8414 035523      136 006507 000012  $CNTLG: .ASCIZ /^G/<15><12> ;;CONTROL 'G'
8415 035530 005015 053523 020122  $MSWR:  .ASCIZ <15><12>/SWR = /
8416 035536 020075      000
8417 035541      040 047040 053505  $MNEW:  .ASCIZ / NEW = /
8418 035546 036440 000040
```

.SBTTL TRAP DECODER

```
*****
*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE 'TRAP' INSTRUCTION
*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
*GO TO THAT ROUTINE.
```

```
8428 035552 010046          $TRAP: MOV     R0,-(SP)  ;;SAVE R0
8429 035554 016600 000002  MOV     2(SP),R0    ;;GET TRAP ADDRESS
8430 035560 005740          TST     -(R0)       ;;BACKUP BY 2
8431 035562 111000          MOVB    (R0),R0     ;;GET RIGHT BYTE OF TRAP
8432 035564 006300          ASL     R0          ;;POSITION FOR INDEXING
8433 035566 016000 035606  MOV     $TRPAD(R0),R0 ;;INDEX TO TABLE
8434 035572 000200          RTS     R0         ;;GO TO ROUTINE
```

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

```
8439 035574 011646          $TRAP2: MOV    (SP),-(SP) ;;MOVE THE PC DOWN
8440 035576 016666 000004 000002  MOV     4(SP),2(SP)  ;;MOVE THE PSW DOWN
8441 035604 000002          RTI                    ;;RESTORE THE PSW
```

.SBTTL TRAP TABLE

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

```
8448      : ROUTINE
8449      : -----
8450 035606 035574  $TRPAD: .WORD  $TRAP2
```

Address	Code	Label	Value	Value	Value	Value	Description
8451	035610	033712					\$TYPE ::CALL=TYPE TRAP+1(104401) TTY TYPEOUT ROUTINE
8452	035612	034220					\$TYPOC ::CALL=TYPOC TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
8453	035614	034174					\$TYPOS ::CALL=TYPOS TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
8454	035616	034234					\$TYPON ::CALL=TYPON TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
8455	035620	034422					\$TYPDS ::CALL=TYPDS TRAP+5(104405) TYPE DECIMAL NUMBER (WITH SIGN)
8456							
8457	035622	035164					\$GTSWR ::CALL=GTSWR TRAP+6(104406) GET SOFT-SWR SETTING
8458							
8459	035624	035114					\$CKSWR ::CALL=CKSWR TRAP+7(104407) TEST FOR CHANGE IN SOFT-SWR
8460	035626	035376					\$RDCHR ::CALL=RDCHR TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
8461	035630	033616					\$SAVREG ::CALL=SAVREG TRAP+11(104411) SAVE R0-R5 ROUTINE
8462	035632	033654					\$RESREG ::CALL=RESREG TRAP+12(104412) RESTORE R0-R5 ROUTINE
8463	035634	036234					.RSET ::CALL=RSETUP TRAP+13(104413) ROUTINE TO INITIALIZE AT FND OF EACH TES
8464	035636	036226					.LPER ::CALL=LPER TRAP+14(104414) ROUTINE TO SET UP LOOP ON ERROR ADDRESS
8465		000032					\$TERM=-.\$TRPAD
8466							
8467							.SBTTL POWER DOWN AND UP ROUTINES
8468							
8469							::*****
8470							:POWER DOWN ROUTINE
8471	035640	012737	036016	000024			\$PWRDN: MOV #ILLUP,@PWRVEC ::SET FOR FAST UP
8472	035646	012737	000340	000026			MOV #340,@PWRVEC+2 ::PRIO:7
8473	035654	010046					MOV R0,-(SP) ::PUSH R0 ON STACK
8474	035656	010146					MOV R1,-(SP) ::PUSH R1 ON STACK
8475	035660	010246					MOV R2,-(SP) ::PUSH R2 ON STACK
8476	035662	010346					MOV R3,-(SP) ::PUSH R3 ON STACK
8477	035664	010446					MOV R4,-(SP) ::PUSH R4 ON STACK
8478	035666	010546					MOV R5,-(SP) ::PUSH R5 ON STACK
8479	035670	017746	143244				MOV @SWR,-(SP) ::PUSH @SWR ON STACK
8480	035674	010667	000122				MOV SP,\$SAVR6 ::SAVE SP
8481	035700	012737	035712	000024			MOV #PWRUP,@PWRVEC ::SET UP VECTOR
8482	035706	000000					HALT
8483	035710	000776					BR -2 ::HANG UP
8484							
8485							::*****
8486							:POWER UP ROUTINE
8487	035712	012737	036016	000024			\$PWRUP: MOV #ILLUP,@PWRVEC ::SET FOR FAST DOWN
8488	035720	016706	000076				MOV \$SAVR6,SP ::GET SP
8489	035724	005067	000072				CLR \$SAVR6 ::WAIT LOOP FOR THE TTY
8490	035730	005267	000066				1\$: INC \$SAVR6 ::WAIT FOR THE INC
8491	035734	001375					BNE 1\$::OF WORD
8492	035736	012677	143176				MOV (SP)+,@SWR ::POP STACK INTO @SWR
8493	035742	012605					MOV (SP)+,R5 ::POP STACK INTO R5
8494	035744	012604					MOV (SP)+,R4 ::POP STACK INTO R4
8495	035746	012603					MOV (SP)+,R3 ::POP STACK INTO R3
8496	035750	012602					MOV (SP)+,R2 ::POP STACK INTO R2
8497	035752	012601					MOV (SP)+,R1 ::POP STACK INTO R1
8498	035754	012600					MOV (SP)+,R0 ::POP STACK INTO R0
8499	035756	012737	035640	000024			MOV #PWRDN,@PWRVEC ::SET UP THE POWER DOWN VECTOR
8500	035764	012737	000340	000026			MOV #340,@PWRVEC+2 ::PRIO:7
8501	035772	104401					TYPE ::REPORT THE POWER FAILURE
8502	035774	036304					\$PWARMG: .WORD POWERM ::POWER FAIL MESSAGE POINTER
8503	035776	012716					MOV (PC)+,(SP) ::RESTART AT START
8504	036000	001370					\$PWRAD: .WORD START ::RESTART ADDRESS
8505	036002	042766	000020	000002			BIC #20,2(SP) ::CLEAR 'T' BIT
8506	036010	005067	175100				CLR \$TBIT ::CLEAR THE 'T' BIT FLAG

8507 036014 000002
8508 036016 000000
8509 036020 000776
8510 036022 000000

RTI
\$ILLUP: HALT ;:THE POWER UP SEQUENCE WAS STARTED
BR ;: BEFORE THE POWER DOWN WAS COMPLETE
\$SAVR6: 0 ;:PUT THE SP HERE

8511
8512
8513
8514
8515
8516
8517
8518
8519
8520
8521
8522 036024 104401
8523 036026 001313
8524 036030 113737 001102 001232
8525 036036 042737 177400 001232
8526 036044 013737 001116 001234
8527 036052 010046
8528
8529 036054 113700 001114
8530 036060 042700 177400
8531 036064 001007
8532 036066 104401 036355
8533 036072 013746 001116
8534 036076 104402
8535 036100 000137 036134
8536
8537 036104 005300
8538
8539 036106 006300
8540 036110 062700 001346
8541 036114 011037 036124
8542 036120 001405
8543 036122 104401
8544 036124 000000
8545 036126 104401
8546 036130 001313
8547 036132 000755
8548 036134 012600
8549 036136 000207

.SBTTL ERROR TYPE OUT ROUTINE
:*****
:*****
:*THIS ROUTINE IS CALLED TO TYPE AN ERROR MESSAGE WHICH IS INCLUDED
:*IN THE ERROR MESSAGE DATA TABLE. IT IS CALLED BY THE \$ERROR ROUTINE
:*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
MOV @#\$STSTM,@#\$TMP0
BIC #177400,@#\$TMP0
MOV @#\$ERRPC,@#\$TMP1 ;GET PC OF CALL
MOV RO,-(SP) ;SAVE RO

MOV @#\$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 PO
ADD # \$ERRTB,RO
MOV (RO),@#2\$;PICK UP ADDRESS OF ERROR MESSAGE
BEQ ERT5 ;
TYPE
2\$: .WORD 0
TYPE
.WORD \$CRLF
BR PCTYP
ERT5: MOV (SP)+,RO ;RESTORE RO.
RTS PC ;AND RETURN.

8550
8551
8552
8553
8554
8555
8556
8557
8558
8559
8560
8561 036140 011637 001236
8562 036144 022626

.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),@#\$TMP? ;SAVE PC OF TRAP.
CMP (SP)+,(SP)+ ;RESTORE SP.

8563 036146 170200
8564 036150 010037 001240
8565 036154 170300
8566 036156 010037 001242
8567 036162 104001
8568 036164 104413
8569
8570
8571
8572
8573 036166 000137 032736
8574
8575
8576
8577
8578
8579
8580
8581 036172 011637 001236
8582 036176 022626
8583 036200 104000
8584 036202 104413
8585
8586
8587
8588
8589 036204 000137 032736
8590
8591
8592
8593
8594
8595
8596
8597 036210 011637 001236
8598 036214 022626
8599 036216 104002
8600 036220 104413
8601
8602
8603
8604
8605 036222 000137 032736
8606
8607
8608
8609
8610
8611
8612
8613
8614
8615 036226 011637 001110
8616 036232 000002
8617
8618

```
STFPS R0 ;GET FPS
MOV R0,@#STMP3
STST R0 ;GET FEC
MOV R0,@#STMP4
1$: ERROR 1
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?).
JMP @#SEOP

.SBTTL CPU SPURIOUS TRAP TO 4 HANDLER
:*****
:*****
:THIS ROUTINE REPORTS UNEXPECTED CPU TRAPS TO VECTOR 4.
:*
CPSPUR: MOV (SP),@#STMP2 ;SAVE PC OF TRAP.
CMP (SP)+,(SP)+
1$: ERROR 0
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?).
JMP @#SEOP

.SBTTL CPU SPURIOUS TRAP TO 10 HANDLER
:*****
:*****
:THIS ROUTINE REPORTS UNEXPECTED CPU TRAPS TO VECTOR 10.
:*
CPTWO: MOV (SP),@#STMP2 ;SAVE PC OF TRAP.
CMP (SP)+,(SP)+
1$: ERROR 2
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?).
JMP @#SEOP

.SBTTL SET LOOP ON ERROR ADDRESS ROUTINE
:*****
:*****
:*****
:*****
:*****
.LPER: MOV (SP),@#SLPERR
RTI

.SBTTL FLAG RESET AND CONSOLE TEST ROUTINE
```

```
8619  
8620  
8621  
8622  
8623  
8624  
8625  
8626  
8627  
8628 036234 023727 001140 177570  
8629  
8630 036242 001001  
8631 036244 104407  
8632  
8633  
8634  
8635 036246 012737 036140 J00244 1$: MOV #FPSPUR,@#FPVECT  
8636 036254 012737 036172 000004 MOV #CPSPUR,@#ERRVECT  
8637 036262 012737 036210 000010 MOV #CPTWO,@#10  
8638 036270 011600 MOV (SP),R0  
8639 036272 012706 001100 MOV #STACK,SP  
8640 036276 005004 CLR R4  
8641 036300 170104 LDFPS R4  
8642 036302 000110 JMP (R0)  
8643  
8644  
8645
```

```
*****  
*****  
*THIS ROUTINE WILL BE CALLED AT THE END OF EACH TEST TO  
*RESET THE STACK, CLEAR THE FPS AND SEE IF THE USER HAS TYPED  
* CONTROL G ON THE TERMINAL. IF THE USER HAS TYPED CONTROL G AND  
*THERE IS NO PHYSICAL CONSOLE SWITCH REGISTER THEN THE CONTENTS  
*OF THE SOFTWARE SWITCH REGISTER WILL BE TYPED IN OCTAL ON THE  
*TELETYPE AND THE USER CAN MODIFY IT.  
*
```

```
:SEE IF THERE IS A PHYSICAL  
:CONSOLE SWITCH REGISTER.  
:BRANCH IF NO.  
:OTHERWISE TYPE THE CONTENTS  
:OF THE PROGRAM VIRTUAL SWITCH REGISTER  
:AND GIVE THE USER A CHANCE TO  
:MODIFY IT.
```

.NLIST BEX

;THESE ARE SPECIAL MESSAGES:

```
036304 050200 053517 051105 POWERM: .ASCIZ <CRLF>'POWER FAILURE. PROGRAM RESTARTING.'  
036350 020040 000 SPACE: .ASCIZ ' '  
036353 011 000 $TAB: .ASCIZ <TAB>  
036355 106 047514 052101 EMSG: .ASCIZ /FLOATING POINT ERROR, STOPPED AT PC= /  
036423 120 047522 040502 EM1: .ASCIZ /PROBABLY BAD FP1 CHIP./  
036452 051120 041117 041101 EM2: .ASCIZ /PROBABLY BAD HYBRID FP CHIP./  
036507 116 020117 047111 EM3: .ASCIZ /NO INTERRUPT FROM SLU IN ALLOTTED TIME /  
000001 .END
```

AABBF0	007070	AMADR3=	000000	BIT5	=	000040	EEC2	017662	HHBBF1	010762	
AABDON	007120	AMADR4=	000000	BIT6	=	000100	EEC20	017772	HHBDON	011106	
AABTP1	007100	AMAMS1=	000000	BIT7	=	000200	EMSG	036355	HHBTP1	010722	
AABTP2	007110	AMAMS2=	000000	BIT8	=	000400	EMTVEC=	000030	HHBTP2	010742	
AAB1	006712	AMAMS3=	000000	BIT9	=	001000	EM1	036423	HHB1	010612	
AAB2	007014	AMAMS4=	000000	BPTVEC=	000014		EM2	036452	HHB10	010772	
AAB3	007034	AMSGAD=	000000	CCBDON	007220		EM3	036507	HHB15	011026	
AAB4	007052	AMSGLG=	000000	CCB1	007124		ERRVEC=	000004	HHB2	010662	
AACDON	017242	AMSGTY=	000000	CCB10	007166		ERTYPE	036024	HHB20	011054	
AACTP1	017146	AMTYP1=	000000	CCB15	007204		ERT5	036134	HHB25	011072	
AAC1	017066	AMTYP2=	000000	CCB2	007142		EXITER	032702	HHCDON	020676	
AAC10	017152	AMTYP3=	000000	CKSWR	=	104407	FFBBF0	010162	HHCTP1	020550	
AAC11	017170	AMTYP4=	000000	CPSPUR	036172		FFBBF1	010172	HHCTP2	020560	
AAC2	017122	APASS	=	CPTWO	036210		FFBDON	010316	HHC1	020450	
AAC20	017206	APRIOR=	000000	CR	=	000015	FFBTP1	010142	HHC10	020570	
AADBF	032404	APTCSU=	000040	CRLF	=	000200	FFBTP2	010152	HHC11	020606	
AADDON	032410	APTENV=	000001	DDBBF0	007516		FFB1	010032	HHC2	020516	
AAD1	032330	APTSIZ=	000200	DDBDON	007536		FFB10	010202	HHC20	020644	
ABASE	=	APTSPO=	000100	DDBTP1	007476		FFB15	010236	HHC25	020624	
ACDW1	=	ASWREG=	000000	DDBTP2	007506		FFB2	010102	HT	=	000011
ACDW2	=	ATESTN=	000000	DDBTP3	007526		FFB20	010264	IIBBF0	011252	
ACPUOP=	000000	AUNIT	=	DDB1	007324		FFB25	010302	IIBBF1	011262	
ACO	=	AUSWR	=	DDB2	007370		FFCDON	020226	IIBDON	011406	
AC1	=	AVECT1=	000000	DDB5	007432		FFCTP1	020120	IIBTP1	011222	
AC2	=	AVECT2=	000000	DDB6	007460		FFCTP2	020130	IIBTP2	011242	
AC3	=	BBCDON	017420	DDCDON	017614		FFC1	020030	IIB1	011112	
AC4	=	BBCTP1	017326	DDCTP1	017514		FFC10	020140	IIB10	011272	
AC5	=	BBC1	017246	DDC1	017424		FFC11	020156	IIB15	011326	
AC6	=	BBC10	017332	DDC10	017526		FFC2	020074	IIB2	011162	
AC7	=	BBC11	017350	DDC11	017544		FFC20	020174	IIB20	011354	
ADDWO	=	BBC2	017302	DDC2	017462		FPSPUR	036140	IIB25	011372	
ADDW1	=	BBC20	017366	DDC20	017562		FPVECT=	000244	IICDON	021012	
ADDW10=	000000	BIT0	=	DDISP	=	177570	GGBBF0	010452	IIC1	020702	
ADDW11=	000000	BIT00	=	DISPLA	001142		GGBBF1	010462	IIC2	020730	
ADDW12=	000000	BIT01	=	DISPRE	000174		GGBDON	010606	IIC20	021002	
ADDW13=	000000	BIT02	=	DSWR	=	177570	GGBTP1	010432	IIC3	020750	
ADDW14=	000000	BIT03	=	EEBBF0	007672		GGBTP2	010442	IOTVEC=	000020	
ADDW15=	000000	BIT04	=	EEBBF1	007702		GGB1	010322	JJBBF0	011546	
ADDW2	=	BIT05	=	EEBDON	010026		GGB10	010472	JJBBF1	011556	
ADDW3	=	BIT06	=	EEBTP1	007652		GGB15	010526	JJBDON	011702	
ADDW4	=	BIT07	=	EEBTP2	007662		GGB2	010372	JJBTP1	011524	
ADDW5	=	BIT08	=	EEB1	007542		GGB20	010554	JJBTP2	011536	
ADDW6	=	BIT09	=	EEB10	007712		GGB25	010572	JJB1	011412	
ADDW7	=	BIT1	=	EEB15	007746		GGCDON	020444	JJB10	011566	
ADDW8	=	BIT10	=	EEB2	007612		GGCTP1	020324	JJB15	011622	
ADDW9	=	BIT11	=	EEB20	007774		GGC1	020232	JJB2	011462	
ADEVCT=	000000	BIT12	=	EEB25	010012		GGC10	020336	JJB20	011650	
ADEVVM	=	BIT13	=	EECDON	020024		GGC11	020354	JJB25	011666	
AENV	=	BIT14	=	EECTP1	017714		GGC2	020272	KKBBF0	012050	
AENVM	=	BIT15	=	EECTP2	017724		GGC20	020412	KKBBF1	012060	
AFATAL=	000000	BIT2	=	EEC1	017620		GGC25	020372	KKBDON	012204	
AMADR1=	000000	BIT3	=	EEC10	017736		GTSWR	=	104406	KKBTP1	012020
AMADR2=	000000	BIT4	=	EEC11	017754		HHBBF0	010752	KKBTP2	012040	

KKB1	011706	LLC9	023206	NNC15	025534	PPP10	002416	RRBTP1	014110
KKB10	012070	LOOP	002056	NNC2	025450	PPP15	002436	RRBTP2	014120
KKB15	012124	LPERR =	104414	NNC20	025554	PPP2	002236	RRBTP3	014130
KKB2	011756	MMBBF0	012602	NNC25	025574	PPP3	002300	RRB1	013770
KKB20	012152	MMBBF1	012612	OOBDON	013332	PPP4	002320	RRB10	014132
KKB25	012170	MMBDON	012720	OOBTP1	013240	PR0GNU=	000003	RRB15	014152
KKCDON	022522	MMBTP1	012552	OOBTP2	013250	PR0	= 000000	RRB2	014044
KKC1	021154	MMBTP2	012572	OOB1	013126	PR1	= 0C0040	RRB20	014170
KKC10	021572	MMB1	012452	OOB10	013260	PR2	= 000100	RRC DON	026752
KKC11	021630	MMB10	012622	OOB15	013300	PR3	= 000140	RRC TBO	026630
KKC12	021666	MMB15	012656	OOB2	013174	PR4	= 000200	RRC TBI	026634
KKC13	021724	MMB2	012516	OOB20	013316	PR5	= 000240	RRC TBI2	026644
KKC14	021762	MMB25	012704	OOCDON	026042	PR6	= 000300	RRC1	026522
KKC15	022020	MMCDON	025376	OOCTB0	025724	PR7	= 000340	RRC10	026650
KKC16	022056	MMC1	023534	OOCTB1	025730	PS	= 177776	RRC15	026666
KKC17	022114	MMC10	024372	OOC1	025624	PSW	= 177776	RRC2	026576
KKC18	022152	MMC11	024450	OOC10	025740	PWRVEC=	000024	RRC20	026706
KKC19	022210	MMC12	024526	OOC15	025756	QQBDON	013764	RRC25	026726
KKC2	021212	MMC13	024604	OOC2	025672	QQBTP1	013662	RRRDON	003360
KKC20	022246	MMC14	024662	OOC20	025776	QQBTP2	013702	RRREXP	003210
KKC3	021250	MMC15	024740	OOC25	026016	QQB1	013546	RRRTP1	003170
KKC4	021306	MMC16	025016	OOODON	002214	QQB10	013712	RRRTP2	003200
KKC5	021344	MMC2	023612	OOOT	002124	QQB15	013732	RRR1	003042
KKC6	021402	MMC3	023670	OOO1	002060	QQB2	013616	RRR10	003220
KKC7	021440	MMC4	023746	OOO2	002104	QQB20	013750	RRR11	003242
KKC8	021476	MMC5	024024	OOO3	002140	QQCDON	026516	RRR12	003262
KKC9	021534	MMC6	024102	OOO4	002206	QQCTB0	026374	RRR15	003314
LDCDSU	023324	MMC7	024160	PCTYP	036066	QQCTB1	026400	RRR2	003120
LDCFSU	022306	MMC8	024236	PIRQ =	177772	QQCTB2	026410	RRR25	003274
LDCT	022512	MMC9	024314	PIRQVE=	000240	QQC1	026270	RRR3	003122
LDXSUB	025076	MNUMBE=	000443	POWERM	036304	QQC10	026414	RRR4	003146
LDXT	025366	MULT	036104	PPBDON	013542	QQC15	026432	RSETUP=	104413
LF =	000012	NATBF1	017050	PPBTP1	013450	QQC2	026342	R6	=%000006
LLBBF0	012330	NATER1	017000	PPBTP2	013460	QQC20	026452	R7	=%000007
LLBBF1	012340	NATER2	017016	PPB1	013336	QQC25	026472	SAVREG=	104411
LLBDON	012446	NATER3	017032	PPB10	013470	QQQBF0	002620	SPACE	036350
LLBTP1	012310	NATINS	016526	PPB15	013510	QQQBF1	002634	SSBDON	014426
LLBTP2	012320	NATRET	017012	PPB2	013404	QQQDON	003036	SSBTP1	014332
LLB1	012210	NATSUB	016446	PPB20	013526	QQQTP1	002650	SSBTP2	014342
LLB10	012350	NNBBF0	013042	PPCDON	026264	QQQ1	002460	SSBTP3	014352
LLB15	012404	NNBDON	013122	PPCTB0	026146	QQQ10	002660	SSB1	014210
LLB2	012254	NNBTP1	013022	PPCTB1	026152	QQQ2	002476	SSB10	014354
LLB25	012432	NNBTP2	013032	PPC1	026046	QQQ20	002700	SSB15	014374
LLCDON	023530	NNB1	012724	PPC10	026162	QQQ22	002714	SSB2	014266
LLC1	022526	NNB10	013052	PPC15	026200	QQQ23	002746	SSB20	014412
LLC10	023254	NNB11	013102	PPC2	026114	QQQ24	002774	SSCDON	027212
LLC2	022574	NNB15	013106	PPC20	026220	QQQ25	003020	SSCTB0	027070
LLC3	022642	NNB2	012750	PPC25	026240	QQQ3	002534	SSCTB1	027074
LLC4	022710	NNCDON	025620	PPCBF0	002356	QQQ4	002562	SSC1	026756
LLC5	022756	NNCTB0	025502	PPCBF1	002372	RDCHR =	104410	SSC10	027104
LLC6	023024	NNCTB1	025506	PPPDON	002454	RESREG=	104412	SSC15	027122
LLC7	023072	NNC1	025402	PPPTP1	002406	RESVEC=	000010	SSC2	027026
LLC8	023140	NNC10	025516	PPP1	002220	RRBDON	014204	SSC20	027142

SSC25	027162	TAB	= 000011	TST53	020446	TTT20	004122	VVV20	004626
SSC30	027210	TBITVE	= 000014	TST54	020700	TTT3	004004	WWBDON	017062
SSSA1	003532	TCCBFC	021130	TST55	021014	TYPDS	= 104405	WWB1	015604
SSSBFO	003522	TCCBF1	021140	TST56	021152	TYPE	= 104401	WWB2	015662
SSSDON	003670	TCCDON	021150	TST57	022524	TYPOC	= 104402	WWB3	015740
SSSTP1	003542	TCC1	021016	TST6	003672	TYPON	= 104404	WWB4	016016
SSSTP2	003552	TCC2	021040	TST60	023532	TYPOS	= 104403	WWB5	016074
SSS1	003364	TCC3	021104	TST61	025400	UUBDON	015064	WWB6	016152
SSS10	003562	TKVEC	= 000060	TST62	025622	UUBTP1	014716	WWB7	016230
SSS11	003604	TPVEC	= 000064	TST63	026044	UUBTP2	014770	WWB8	016306
SSS15	003614	TRAPVE	= 000034	TST64	026266	UUB1	014642	WWB9	016364
SSS2	003442	TRTVEC	= 000014	TST65	026520	UUB10	015000	WWCDON	031332
SSS20	003632	TST1	002056	TST66	026754	UUB15	015036	WWC1	027674
SSS25	003652	TST10	004412	TST67	027214	UUB2	014714	WWC10	030334
STACK	= 001100	TST100	032736	TST7	004142	UUB20	015020	WWC11	030400
START	001370	TST11	004646	TST70	027466	UUCBFO	027560	WWC12	030444
STCDF5	006314	TST12	005120	TST71	027570	UUCDON	027566	WWC13	030510
STCDT	006576	TST13	005754	TST72	027672	UUCTP1	027546	WWC14	030554
STCFDS	005460	TST14	006610	TST73	031334	UUC1	027470	WWC15	030620
STCFT	005742	TST15	006710	TST74	031404	UUC2	027520	WWC16	030664
STCIBF	031322	TST16	007122	TST75	032116	UUC3	027530	WWC17	030730
STCSUB	031042	TST17	007222	TST76	032326	UUUA1	004276	WWC18	030774
STKLMT	= 177774	TST2	002216	TST77	032416	UUUA2	004300	WWC2	027740
STXBF	032102	TST20	007322	TTBDON	014636	UUUA3	004302	WWC4	030004
STXSUB	031674	TST21	007540	TTBTP1	014544	UUUBFO	004264	WWC5	030050
SWR	001140	TST22	010030	TTBTP2	014554	UUUDON	004410	WWC6	030114
SWREG	000176	TST23	010320	TTB1	014432	UUUTP1	004310	WWC7	030160
SW0	= 000001	TST24	010610	TTB10	014564	UUU1	004144	WWC8	030224
SW00	= 000001	TST25	011110	TTB15	014604	UUU10	004320	WWC9	030270
SW01	= 000002	TST26	011410	TTB2	014500	UUU11	004342	WWWBF0	004776
SW02	= 000004	TST27	011704	TTB20	014622	UUU15	004354	WWWBF1	005016
SW03	= 000010	TST3	002456	TTCDON	027464	UUU2	004230	WWWDON	005116
SW04	= 000020	TST30	012206	TTCTB0	027336	UUU20	004372	WWWTP1	005006
SW05	= 000040	TST31	012450	TTCTB1	027342	UUU3	004254	WWW1	004650
SW06	= 000100	TST32	012722	TTCTB2	027352	VVBDON	007320	WWW10	005026
SW07	= 000200	TST33	013124	TTC1	027216	VVB1	007224	WWW11	005050
SW08	= 000400	TST34	013334	TTC10	027356	VVB10	007266	WWW15	005062
SW09	= 001000	TST35	013544	TTC15	027374	VVB15	007304	WWW2	004740
SW1	= 000002	TST36	013766	TTC2	027274	VVB2	007242	WWW20	005100
SW10	= 002000	TST37	014206	TTC20	027414	VVCBFO	027662	XXBDON	015326
SW11	= 004000	TST4	003040	TTC25	027434	VVCDON	027670	XXBTP1	015214
SW12	= 010000	TST40	014430	TTC30	027462	VVCTP1	027650	XXBTP2	015224
SW13	= 020000	TST41	014640	TTA1	004026	VVC1	027572	XXB1	015070
SW14	= 040000	TST42	015066	TTA2	004030	VVC2	027622	XXB10	015254
SW15	= 100000	TST43	015330	TTA3	004032	VVC3	027632	XXB15	015274
SW2	= 000004	TST44	015602	TTBFO	004014	VVBF0	004534	XXB2	015142
SW3	= 000010	TST45	017064	TTTTON	004140	VVVDON	004644	XXB20	015312
SW4	= 000020	TST46	017244	TTTTP1	004040	VVTP1	004544	XXB25	015234
SW5	= 000040	TST47	017422	TTT1	003674	VVV1	004414	XXCDON	031402
SW6	= 000100	TST5	003362	TTT10	004050	VVV10	004554	XXC1	031336
SW7	= 000200	TST50	017616	TTT11	004072	VVV11	004576	XXXDON	005752
SW8	= 000400	TST51	020026	TTT15	004104	VVV15	004610	XXX1	005122
SW9	= 001000	TST52	020230	TTT2	003760	VVV2	004476	XXX2	005176

XXX3	005252	ZZZ2	006630	\$FFLG	035112	\$REG10	001202	\$TMP13	001260
XXX4	005326	\$APTHD	001354	\$FILLC	001156	\$REG11	001204	\$TMP14	001262
XXX5	005402	\$ATYC	034672	\$FILLS	001155	\$REG12	001206	\$TMP15	001264
Y	032652	\$ATY1	034646	\$GDADR	001120	\$REG13	001210	\$TMP16	001266
YYBDON	015600	\$ATY3	034654	\$GDDAT	001124	\$REG14	001212	\$TMP17	001270
YYBTP1	015464	\$ATY4	034664	\$GET42	033014	\$REG15	001214	\$TMP2	001236
YYBTP2	015474	\$AUTOB	001134	\$GTSWR	035164	\$REG16	001216	\$TMP20	001272
YYBTP3	015504	\$BDADR	001122	\$HD =	000003	\$REG17	001220	\$TMP21	001274
YYB1	015332	\$BDDAT	001126	\$HIBTS	001354	\$REG2	001166	\$TMP22	001276
YYB10	015526	\$BELL	001306	\$ICNT	001104	\$REG20	001222	\$TMP23	001300
YYB15	015546	\$CHARC	034170	\$ILLUP	036016	\$REG21	001224	\$TMP3	001240
YYB2	015412	\$CKSWR	035114	\$INTAG	001135	\$REG22	001226	\$TMP4	001242
YYB20	015564	\$CLR.T	033032	\$ITEMB	001114	\$REG23	001230	\$TMP5	001244
YYB25	015506	\$CMTAG	001100	\$LF	001314	\$REG3	001170	\$TMP6	001246
YYCDON	032114	\$CM1 =	000024	\$LFLG	035111	\$REG4	001172	\$TMP7	001250
YYC1	031406	\$CM2 =	000050	\$LOOP	033110	\$REG5	001174	\$TN =	000100
YYC2	031444	\$CM3 =	000024	\$LPADR	001106	\$REG6	001176	\$TPB	001152
YYC3	031502	\$CM4 =	000024	\$LPERR	001110	\$REG7	001200	\$TPFLG	001157
YYC4	031540	\$CNTLG	035523	\$MAIL	001316	\$RESRE	033654	\$TPS	001150
YYC5	031576	\$CNTLU	035516	\$MBADR	001356	\$RTNAD	033112	\$TRAP	035552
YYC6	031634	\$CPUOP	001344	\$MFLG	035110	\$RTRN	033106	\$TRAP2	035574
YYYDON	006606	\$CRLF	001313	\$MNEW	035541	\$SAVRE	033616	\$TRP =	000015
YYY1	005756	\$DBLK	034636	\$MSGAD	001332	\$SAVR6	036022	\$TRPAD	035606
YYY2	006032	\$DEVCT	001326	\$MSGLG	001334	\$SCOPE	033136	\$TSTM	001360
YYY3	006106	\$DOAGN	033052	\$MSGTY	001316	\$SETUP=	000137	\$TSTNM	001102
YYY4	006162	\$DTBL	034626	\$MSWR	035530	\$STUP =	177777	\$TYPDS	034422
YYY5	006236	\$ENDAD	033042	\$MXCNT	033414	\$SVLAD	033344	\$TYPE	033712
Z	032654	\$ENDCT	032772	\$NULL	001154	\$SVPC =	001354	\$TYPEC	034124
ZZCBF	032306	\$ENDMG	033121	\$NWTST=	000001	\$SWR =	177400	\$TYPEX	034172
ZZCDON	032320	\$ENULL	033116	\$OCNT	034416	\$SWREG	001340	\$TYPOC	034220
ZZC1	032120	\$ENV	001336	\$OMODE	034420	\$SWRMK=	000000	\$TYPON	034234
ZZC10	032260	\$ENVM	001337	\$OVER	033400	\$SWRMS=	000200	\$TYPOS	034174
ZZC12	032274	\$EOP	032736	\$PASS	001324	\$TAB	036353	\$UNIT	001330
ZZC15	032300	\$EOPCT	032764	\$PASTM	001362	\$TBIT	033114	\$UNITM	001364
ZZC2	032130	\$ERFLG	001103	\$PWRAD	036000	\$TERM =	000032	\$USWR	001342
ZZC3	032156	\$ERMAX	001115	\$PWRDN	035640	\$TESTN	001322	\$XTSTR	033150
ZZC5	032254	\$ERROR	033416	\$PWRMG	035774	\$TIMES	001302	\$GET4=	000001
ZZDDON	032656	\$ERRPC	001116	\$PWRUP	035712	\$TKB	001146	\$OFILL	034417
ZZD1	032420	\$ERRTB	001346	\$QUES	001312	\$TKS	001144	. =	036557
ZZD2	032436	\$ERTTL	001112	\$RDCHR	035376	\$TMP0	001232	.LPER	036226
ZZZDON	006706	\$ESCAP	001304	\$RDSZ =	000001	\$TMP1	001234	.RSET	036234
ZZZ1	006612	\$ETABL	001336	\$REGAD	001160	\$TMP10	001252	.\$X =	001354
ZZZ10	006654	\$ETEND	001346	\$REGO	001162	\$TMP11	001254		
ZZZ15	006672	\$FATAL	001320	\$REG1	001164	\$TMP12	001256		

. ABS. 036557 000

ERRORS DETECTED: 0

MULE:CJKDDA,MULE:CJKDDA,SEQ/SOL/NL:TOC=CJKDDA.P11
RUN-TIME: 98 70 3 SECONDS
RUN-TIME RATIO: 378/172=2.1
CORE USED: 27K (53 PAGES)