

FP11-F

FP11F FLTG PNT PRTC
CKFPCCO

AH-F638C-MC
FICHE 1 OF 2

OCT 1981
COPYRIGHT © 79-81
MADE IN USA



FP11-F

FP11F FLTG PNT PRTC
CKFPCCO

AH-F638C-MC
FICHE 2 OF 2

OCT 1981
COPYRIGHT © 79-81
MADE IN USA



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

.REM 8

IDENTIFICATION

PRODUCT CODE: AC-F636C-MC
PRODUCT NAME: CKFPCCO FP11F FLTG PNT PRT C
PRODUCT DATE: APRIL, 1981
MAINTAINER: DIAGNOSTIC ENGINEERING
AUTHOR: ANTHONY VEZZA, DAN MILLEVILLE

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 APPEAR IN THIS DOCUMENT.

NO RESPONSIBILITY IS ASSUMED FOR THE USE OR RELIABILITY OF SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL OR ITS AFFILIATED COMPANIES.

COPYRIGHT (C) 1979, 1981 BY DIGITAL EQUIPMENT CORPORATION

HISTORY

NO CHANGES TO THE 11/34 FLOATING POINT DIAGNOSTIC PART 'A' WERE FOUND TO BE NEEDED TO ADAPT IT FOR USE ON THE 11/44.

THE FOLLOWING WAS ADDED TO THE 11/34 FLOATING POINT DIAGNOSTIC TO MAKE THE 'B' VERSION COVER THE 11/44:

1. TEST 22 - PROCESSOR LOOKS TO SEE IF APT IS CONTROLLING THE TEST, AND IF IT IS, CHECKS TO SEE IF THE USER HAS SELECTED THIS TEST BY CHECKING BIT 7 IN THE SWITCH REGISTER. IT HAS ALSO BEEN CHANGED SO THAT IF BIT 7 IS *ONE*, THE CODE WILL SELECT THE TEST.

THE FOLLOWING WAS ADDED TO THE 11/34 FLOATING POINT DIAGNOSTIC TO MAKE THE 'C' VERSION COVER THE 11/44:

1. TEST 76 - CHECKS THAT FP PROCESSOR DOESN'T ACCESS D-SPACE UNTIL CONDITIONS WARRANT.
2. TEST 77 TO 106 - CHECKS THAT SR1 MATCHES WHAT ACTUALLY HAPPENED TO THE REGISTER OF THE INSTRUCTION, AND THAT THE VALUE OF AUTO INCREMENT/DECREMENT WAS PROPER.

THE FOLLOWING WAS ADDED TO THE 'C' VERSION TO FURTHER INTENSIFY THE TEST:

1. TEST 77 - A BYTE TABLE OF EXPECTED DATA FOR SR1 CHECKS TO MAKE SURE THAT THE VALUE OF THE INCREMENT/DECREMENT IS PROPER FOR THAT INSTRUCTION.

ALL THREE PARTS WERE RE-RELEASED WITH A NEW SYSMAC THAT CHECKS BIT 0 OF THE CPU ERROR REGISTER (POWER MONITOR BIT). THE ADDITIONS WERE MADE IN THE SCOPE ROUTINE, EXECUTED AT THE BEGINNING OF EACH TEST, AND THE ERROR CALL ROUTINE. IF THE BIT BECOMES SET, AN ERROR IS CALLED FROM THE SCOPE ROUTINE. THE BIT IS CLEARED, AND THE TEST IS CONTINUED. IF THE BIT BECOMES SET IN THE MIDDLE OF A TEST, AND AN ERROR OCCURS FOR ANY REASON, THE ERROR ROUTINE WILL CALL *TWO* ERRORS, THE POWER MONITOR BIT ERROR FIRST, THEN THE ERROR ORIGINALLY CALLED. IN ADDITION, THE \$READ ROUTINE NOW CHECKS FOR A RANDOMLY INPUTED ^Q BEFORE A ^S IS TYPED. THIS BECAME NECESSARY WITH CERTAIN DATA CONNECTIONS OF SOME SYSTEMS.

41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85

86
87
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

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 INTERRUPTS TESTS
 - 8.7 ACT, APT AND XXDP COMPATIBILITY
9. PROGRAM DESCRIPTION
 - 9.1 CKFPCCO
10. LISTING
 - 10.1 CKFPCCO

137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193

1. ABSTRACT

THE THREE PROGRAMS:

CKFPAAO CKFPBAO CKFPCCO

ARE DESIGN TO DETECT AND REPORT LOGIC FAULTS IN THE PDP 11/44 FP11-F FLOATING POINT PROCESSOR. THE DESIGN IS AN ATTEMPT TO REACH ALL ROM STATES, TAKE ALL BRANCH MICRO TESTS (BUT'S) AND VERIFY ALL THE LOGIC. THEY CONSIST OF 161 (OCT) INDIVIDUAL TESTS SEQUENCED TO DETECT AND ATTEMPT TO IDENTIFY FAULTS WITH A MINIMUM HARDWARE OR SOFTWARE LEVEL. THE TESTS ARE PARTIONED INTO THREE STAND-ALONE PROGRAMS DESCRIBED BELOW.

NOTE THAT ERROR REPORTS IN THESE PROGRAMS ARE BASED UPON THE KNOWLEDGE THAT ALL PREVIOUS TESTS HAVE BEEN RUN AND IN MOST CASE THAT THERE IS ONLY A SINGLE POINT FAULT IN THE FP11-F. IF THE PROGRAMS OR TESTS ARE NOT RUN IN ORDER THEN ERROR MESSAGES MAY NOT BE ACCURATE.

A. CKFPAAO

CKFPAAO TESTS:

- 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)

B. CKFPBAO

CKFPBAO TESTS:

- ADDF, ADDD AND SUBD (ALL CONDITIONS NOT TESTED IN CKFPAAO)
- CMPD AND CMPF
- DIVD AND DIVF
- MULD AND MULF
- MODD AND MODF

C. CKFPCCO

CKFPCCO TESTS:

- STF AND STD (ALL MODES)
- STCFD AND STCDF
- CLRD AND CLRF
- NEGF AND NEGD

194 ABSF AND ABS
 195 TSTF AND TSTD
 196 NEGF, ABSF AND TSTF (ALL SOURCE MODES)
 197 NEGF, ABSF AND TSTF (ALL SOURCE MODES)
 198 LDFPS (ALL SOURCE MODES)
 199 LDCIF AND LDCLF
 200 LDCID AND LDCLD
 201 LDEXP
 202 STFPS (ALL DESTINATION MODES)
 203 STCFL AND STCFI
 204 STCDL AND STCDI
 205 STEXP
 206 STST
 207 I AND D SPACE TESTS (ALL MODES AND REGS 0 AND 7)
 208 AUTO INCREMENT/DECREMENT CHECK - SRI (ALL MODES AND REGS 1 AND 7)
 209

2. REQUIREMENTS

- 2.1 EQUIPMENT
 A PDP 11/44 WITH CONSOLE AND AN FP11-F FLOATING POINT PROCESSOR. NOTE THAT A SPECIAL INTERRUPTS TEST MODULE IS BEING DESIGNED FOR USE IN THE MANUFACTURING ENVIRONMENT. WHEN THIS DEVICE IS PRESENT THE PROGRAM CKFPBAO WILL MAKE USE OF IT TO TEST THE FPP INTERRUPT ON BUS REQUEST FUNCTIONS.
- 2.2 STORAGE
 ALL THREE PROGRAM REQUIRE A MEMORY SYSTEM OF AT LEAST 16K TO LOAD AND RUN.
- 2.3 PRELIMINARY PROGRAMS
 THESE THREE DIAGNOSTICS WILL ASSUME THAT THE PDP 11/44 CENTRAL PROCESSOR IS FAULTLESS, THEREFORE WHEN IN DOUBT RUN THE PDP 11/44 PROCESSOR DIAGNOSTICS BEFORE THESE FP11-F DIAGNOSTICS.

3. LOADING PROCEDURE

THE PROGRAMS WILL BE SUPPLIED ON THE USUAL 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

250

251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
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

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). OF RUNNING UNDER ACT, APT OR CHAIN THIS DOES NOT APPLY.
5. THE PROGRAM WILL LOOP AND AN END OF PASS AND ERROR SUMMARY 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>=...	4000	INHIBIT ITERATIONS
SW<10>=...	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>
SW<7>=1....	200	PRINT ERROR SUMMARY EVEN IF SW<13>=1. THIS APPLIES ONLY TO PROGRAM CKFPAA0.
SW<7>=1....	200	SELECT CORRECT INTERRUPT TEST IN PROGRAM CKFPBA0.

6. ERRORS

6.1 SUMMARIES

IN PROGRAM CKFPAA0 TESTS 1 AND 11 HAVE A SPECIAL ERROR SUMMARY FEATURE. THESE TWO TEST RUN MANY TEST PATTERNS THROUGH THE LOGIC. AFTER AN ERROR IS ENCOUNTERED, ONLY THE FIRST FIVE ERRORS ARE REPORTED (TYPED ON THE TTY). EVERY ERROR THOUGH IS LOGGED AND AN ERROR SUMMARY IS PRINTED WHEN THE TEST IS COMPLETE. NOTE THAT IS SW<13>=1 THIS SUMMARY WILL NOT BE TYPED UNLESS SW<7>=1. IN OTHER WORDS TO GET JUST AN ERROR SUMMARY FROM EITHER OF THESE TWO TESTS 1 AND 11 IN PROGRAM CKFPAA0 BOTH SWITCHES 13 AND 7 MUST - 1.

6. ERROR RECOVERY

308
309
310
311
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

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

NCNF

8. MISCELLANEOUS

8.1 EXECUTION TIMES

LESS THAN 10 SECONDS FOR EACH PROGRAM ON ANY PASS.

8.2 STACK POINTER

THE STACK POINTER IS INITIALIZED TO 1100 IN EACH OF THE THREE 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 AND THE TOTAL NUMBER OF ERRORS SINCE THE LAST END OF PASS MESSAGE..

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

IF THE USER DESIRES, A SOFTWARE SWITCH REGISTER CAN BE EXAMINED OR MODIFIED AT ANY TIME BY THE USER IF HE TYPES CNTRL/G WHILE THE PROGRAM IS RUNNING. THIS CNTRL/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 CNTRL/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

365
366
367
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

THE PROGRAM IS RUN AFTER LOADING ONLY IF THE
CONSOLE SWITCH REGISTER CONTAINS 177777.

8.6 INTERRUPTS TEST

IN PROGRAM CKFPBAO THERE IS A SPECIAL TEST FOR
CHECKING THE CORRECT FLOWS OF THE FPP. THIS TEST
CAN BE RUN ONLY IF A SPECIAL TEST MODULE IS IN THE
SYSTEM. THIS MODULE WILL PROBABLY ONLY BE USED IN
MANUFACTURING. IF THIS MODULE IS NOT IN THE SYSTEM
THIS TEST WILL AUTOMATICALLY BE DESELECTED. IF THIS
TEST MODULE IS ON THE SYSTEM AND SW<7>=1 THIS TEST
WILL BE RUN. IF SW<7> 0, THIS TEST WILL BE
DESELECTED.

8.7 ACT, APT AND XXDP COMPATIBILITY

THESE PROGRAMS ARE FULLY COMPATIBLE WITH:
APT
ACT
XXDP MONITOR AND CHAIN PROGRAMS.

9. PROGRAM DESCRIPTION

TEST 1 STF WITH ILLEGAL ACCUMULATOR TEST

THIS IS A TEST OF THE ST INSTRUCTION USING ILLEGAL
ACCUMULATOR 7, MODE 0.

TEST 2 FDST MODE 1, FLOATING MODE, TEST

THIS IS A TEST OF THE STF INSTRUCTION USING FDST
MODE 1.

TEST 3 FDST MODE 2 TEST

THIS IS A TEST OF BOTH STF AND STD WITH FDST MODE 2.

TEST 4 FDST MODE 2, WITH GR7, TEST

THIS IS A TEST OF STF WITH GR7 MODE 2 OR IMMEDIATE
MODE.

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

TEST 5 FDST MODE 4 TEST

THIS IS A TEST OF STD WITH FDST MODE 4.

TEST 6 FDST MODE 3 TEST

THIS IS A TEST OF FDST MODE 3 USING STD.

TEST 7 FDST MODE 5 TEST

THIS IS A TEST OF FDST MODE 5 USING STD.

TEST 10 FDST MODE 6, INDEX MODE, TEST

THIS IS A TEST OF FDST MODE 6, INDEX MODE, USING STD.

TEST 11 FDST MODE 7, INDEX DEFERRED MODE, TEST

THIS IS A TEST OF FDST MODE 7, INDEX DEFERRED MODE, USING STD.

TEST 12 STCFD TEST

THIS IS A TEST OF THE STCFD INSTRUCTION.

TEST 13 STCDF TEST

THIS IS A TEST OF THE STCDF INSTRUCTION.

TEST 14 STCFD WITH ILLEGAL ACCUMULATOR TEST

THIS TEST STCFD WITH ILLEGAL AC 6.

TEST 15 CLRD TEST

THIS IS A TEST OF THE CRLF AND CLRD INSTRUCTIONS.

TEST 16 CLRD WITH ILLEGAL ACCUMULATOR TEST

THIS IS A TEST OF CLRD WITH ILLEGAL AC7.

TEST 17 NEGF, ABSF AND TSTF SOURCE MODE 0 WITH ILLEGAL AC7, TEST

THIS IS A TEST OF THE SPECIAL DEST FLOWS USING THE

479 NEGD INST WITH MODE ZERO AND ILLEGAL AC7.
480
481 TEST 20 NEGF, ABSF AND TSTF SOURCE MODE 0 TEST
482 -----
483
484 THIS IS A TEST THE NEGF, ABSF AND TSTF SOURCE FLOWS.
485 THE NEGD INSTRUCTION IS USED TO TEST MODE 0
486
487 TEST 21 NEGF, ABSF AND TSTF SOURCE MODE 1 TEST
488 -----
489
490 THIS IS A TEST THE NEGF, ABSF AND TSTF SOURCE FLOWS.
491 THE NEGD INSTRUCTION IS USED TO TEST MODE 1
492
493 TEST 22 NEGF, ABSF AND TSTF SOURCE MODE 2 TEST
494 -----
495
496 THIS IS A TEST THE NEGF, ABSF AND TSTF SOURCE FLOWS.
497 THE ABSD INSTRUCTION IS USED TO TEST MODE 2
498
499 TEST 23 NEGF, ABSF AND TSTF SOURCE MODE 4 TEST
500 -----
501
502 THIS IS A TEST THE NEGF, ABSF AND TSTF SOURCE FLOWS.
503 THE ABSD INSTRUCTION IS USED TO TEST MODE 4
504
505 TEST 24 NEGF, ABSF AND TSTF SOURCE MODE 3 TEST
506 -----
507
508 THIS IS A TEST THE NEGF, ABSF AND TSTF SOURCE FLOWS.
509 THE ABSD INSTRUCTION IS USED TO TEST MODE 3
510
511 TEST 25 NEGF, ABSF AND TSTF SOURCE MODE 5 TEST
512 -----
513
514 THIS IS A TEST THE NEGF, ABSF AND TSTF SOURCE FLOWS.
515 THE NEGD INSTRUCTION IS USED TO TEST MODE 5
516
517 TEST 26 NEGF, ABSF AND TSTF SOURCE MODE 6 TEST
518 -----
519
520 THIS IS A TEST THE NEGF, ABSF AND TSTF SOURCE FLOWS.
521 THE ABSD INSTRUCTION IS USED TO TEST MODE 6
522
523 TEST 27 NEGF, ABSF AND TSTF SOURCE MODE 7 TEST
524 -----
525
526 THIS IS A TEST THE NEGF, ABSF AND TSTF SOURCE FLOWS.
527 THE ABSD INSTRUCTION IS USED TO TEST MODE 6
528
529 TEST 30 NEGF, ABSF AND TSTF SOURCE MODE 6, GR7, TEST
530 -----
531
532 THIS IS A TEST THE NEGF, ABSF AND TSTF SOURCE FLOWS.
533 THE NEGD INSTRUCTION IS USED TO TEST MODE 6
534
535 TEST 31 NEGF, ABSF AND TSTF SOURCE MODE 7, GR7, TEST

536 -----
537
538 THIS IS A TEST THE NEGF, ABSF AND TSTF SOURCE FLOWS.
539 THE ABSD INSTRUCTION IS USED TO TEST MODE 7
540
541 TEST 32 SPECIAL DEST, MODE 0, TEST
542 -----
543
544 THIS IS A TEST OF THE NEGF ABSF AND TSTF DESTINATION
545 FLOWS MODE 0 USING THE NEGD INSTR.
546
547 TEST 33 SPECIAL DEST, MODE 1, TEST
548 -----
549
550 THIS IS A TEST OF THE NEGF ABSF AND TSTF DESTINATION
551 FLOWS MODE 1 USING THE NEGD INSTR.
552
553 TEST 34 SPECIAL DEST, MODE 2, TEST
554 -----
555 THIS IS A TEST OF THE NEGF ABSF AND TSTF DESTINATION
556 FLOWS MODE 2 USING THE NEGD INSTR.
557
558 TEST 35 SPECIAL DEST, MODE 4, TEST
559 -----
560
561 THIS IS A TEST OF THE NEGF ABSF AND TSTF DESTINATION
562 FLOWS MODE 4 USING THE NEGD INSTR.
563
564 TEST 36 SPECIAL DEST, MODE 3, TEST
565 -----
566
567 THIS IS A TEST OF THE NEGF ABSF AND TSTF DESTINATION
568 FLOWS MODE 3 USING THE NEGD INSTR.
569
570 TEST 37 SPECIAL DEST, MODE 5, TEST
571 -----
572
573 THIS IS A TEST OF THE NEGF ABSF AND TSTF DESTINATION
574 FLOWS MODE 5 USING THE NEGD INSTR.
575
576 TEST 40 SPECIAL DEST, FLOATING MODE 2, TEST
577 -----
578
579 THIS IS A TEST OF THE NEGF ABSF AND TSTF DESTINATION
580 FLOWS MODE 2 USING THE NEGF INSTR.
581
582 TEST 41 SPECIAL DEST, MODE2, GR7 (IMMEDIATE), TEST
583 -----
584
585 THIS IS A TEST OF THE NEGF ABSF AND TSTF DESTINATION
586 FLOWS MODE 2(IMMEDIATE) USING THE NEGD INSTR.
587
588 TEST 42 SPECIAL DEST, MODE 6, TEST
589 -----
590
591 THIS IS A TEST OF THE NEGF ABSF AND TSTF DESTINATION
592 FLOWS MODE 6 USING THE NEGD INSTR.

593
594 TEST 43 SPECIAL DEST, MODE 7, TEST
595 -----
596
597 THIS IS A TEST OF THE NEGF ABSF AND TSTF DESTINATION
598 FLOWS MODE 7 USING THE NEGD INSTR.
599
600 TEST 44 NEGD, ABSD AND TSTD TEST
601 -----
602
603 THIS IS A TEST OF THE NEGD ABSD AND TSTD
604 INSTRUCTIONS.
605
606 TEST 45 SOURCE MODES, MODE 1 (FL=0), TEST
607 -----
608
609 THIS IS A TEST OF SOURCE MODE 1 USING THE LDFPS
610 INSTRUCTION.
611
612 TEST 46 SOURCE MODES, MODE 2 (FL=0), TEST
613 -----
614
615 THIS IS A TEST OF SOURCE MODE 2 USING THE LDFPS
616 INSTRUCTION.
617
618 TEST 47 SOURCE MODES, MODE 4 (FL=0), TEST
619 -----
620
621 THIS IS A TEST OF SOURCE MODE 4 USING THE LDFPS
622 INSTRUCTION.
623
624 TEST 50 SOURCE MODES, MODE 3 (FL=0), TEST
625 -----
626
627 THIS IS A TEST OF SOURCE MODE 3 USING THE LDFPS
628 INSTRUCTION.
629
630 TEST 51 SOURCE MODES, MODE 5 (FL=0), TEST
631 -----
632
633 THIS IS A TEST OF SOURCE MODE 5 USING THE LDFPS
634 INSTRUCTION.
635
636 TEST 52 SOURCE MODES, MODE 6 (FL=0), TEST
637 -----
638
639 THIS IS A TEST OF SOURCE MODE 6 USING THE LDFPS
640 INSTRUCTION.
641
642 TEST 53 SOURCE MODES, MODE 7 (FL=0), TEST
643 -----
644
645 THIS IS A TEST OF SOURCE MODE 7 USING THE LDFPS
646 INSTRUCTION.
647
648 TEST 54 SOURCE MODES, MODE 2 GR7 (FL=1), TEST
649 -----

```

650
651           THIS IS A TEST OF THE LDCLD WITH IMMEDIATE
652           ADDRESSING MODE
653
654 TEST 55     SOURCE MODES, MODE 2 (FL=1), TEST
655 -----
656           THIS IS A TEST OF THE LDCLD INSTRUCTION WITH MODE 2.
657
658 TEST 56     LDCIF AND LDCLF TEST
659 -----
660
661           THIS IS A TEST OF THE LDCIF AND THE LDCLF
662           INSTRUCTIONS.
663
664 TEST 57     LDCID AND LDCLD TEST
665 -----
666
667           THIS IS A TEST OF LDCID AND LDCLD
668
669 TEST 60     LDEXP TEST
670 -----
671
672           THIS IS A TEST OF THE LDEXP INST A SUBROUTINE IS
673           USED TO SET UP OPERANDS, EXECUTE THE LDEXP INST AND
674           CHECK THE RESULTS.
675
676 TEST 61     DESTINATION MODES, MODE 1 (FL=0), TEST
677 -----
678
679           THIS IS A TEST OF DESTINATION MODE 1 USING THE STFPS
680           INSTRUCTION
681
682 TEST 62     DESTINATION MODES, MODE 2 (FL=0), TEST
683 -----
684
685           THIS IS A TEST OF DESTINATION MODE 2 USING THE STFPS
686           INSTRUCTION
687
688 TEST 63     DESTINATION MODES, MODE 4 (FL=0), TEST
689 -----
690
691           THIS IS A TEST OF DESTINATION MODE 4 USING THE STFPS
692           INSTRUCTION
693
694 TEST 64     DESTINATION MODES, MODE 3 (FL=0), TEST
695 -----
696
697           THIS IS A TEST OF DESTINATION MODE 3 USING THE STFPS
698           INSTRUCTION
699
700 TEST 65     DESTINATION MODES, MODE 5 (FL=0), TEST
701 -----
702
703           THIS IS A TEST OF DESTINATION MODE 5 USING THE STFPS
704           INSTRUCTION
705
706 TEST 66     DESTINATION MODES, MODE 6 (FL=0), TEST

```


716 THIS IS A TEST OF DESTINATION MODE 7 USING THE STFPS
717 INSTRUCTION
718
719 TEST 70 DESTINATION MODES, MODE 2 (FL-1), TEST
720 -----
721
722 THIS IS A TEST OF DESTINATION MODE 2 USING STCOL
723 WITH REGISTER 0
724
725 TEST 71 DESTINATION MODES, MODE 4 (FL=1), TEST
726 -----
727
728 THIS IS A TEST OF DESTINATION MODE 4 USING STCDL
729 WITH REGISTER 0
730
731 TEST 72 STCDI AND STCDL TEST
732 -----
733
734 THIS IS A TEST OF THE STCDI AND STCDL INSTRUCTIONS.
735 NOTE THAT A SUBROUTINE, STCSUB, IS USED TO SET UP
736 THE OPERANDS, EXECUTE THE STC INSTRUCTION AND CHECK
737 THE RESULT.
738
739 TEST 73 STCFL AND STCFI TEST
740 -----
741
742 THIS IS A TEST OF STCFL AND STCFI. IT MAKES USE OF
743 THE SAME SUBROUTINE, STCSUB, WHICH WAS USED TO TEST
744 STCDL AND STCDI.
745
746 TEST 74 STEXP TEST
747 -----
748
749 THIS IS A TEST OF THE STEXP INSTRUCTION
750
751 TEST 75 STST TEST
752 -----
753
754 THIS IS A TEST OF THE STST INSTRUCTION. FIRST AN
755 ILLEGAL FPS OP CODE (INSTRUCTION) IS USED TO ENTER
756 AN ERROR CONDITION IN THE FEC AND FFA. THE STST IS
757 EXECUTED AND THE FEC AND FEA ARE CHECKED
758
759 TEST 76 D-SPACE NON-ACCESS TEST
760 -----
761
762 THIS IS A TEST THAT ENABLES D-SPACE, BUT MAKES IT
763 NON-RESIDENT, CAUSING A MEMORY MANAGEMENT TRAP
764 SHOULD IT BE ACCESSED DURING AN INSTRUCTION THAT
765 WILL NOT NORMALLY ACCESS D-SPACE.
766
767 TEST 77 AUTO INCREMENT/DECREMENT TEST
768 -----
769
770 THIS IS A TEST THAT ENABLES D-SPACE, BUT MAKES IT
771 NON-RESIDENT IN THE AREA OF THE TEST, FORCING A
772 MEMORY MANAGEMENT TRAP FOR EVERY FPP INSTRUCTION IN

773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790

THE TEST. SR1 IS THEN EXAMINED FOR PROPER CONTENTS.
SHOULD THE FPP INSTRUCTION FAIL TO ABORT, THE NEXT
INSTRUCTION IS AN IOT TRAP, AND CALLS AN ERROR TO
ANNOUNCE THE FPP INSTRUCTION'S FAILING TO CAUSE AN
ABORT, NOT ALLOWING PROPER EXAMINATION OF SR1.

10.

LISTING
-----8

000444
000003

MNUMBER=444
PROGNUM=3
.LIST ME
.NLIST MD
.NLIST MC
.NLIST CND
.NLIST BEY

```

1605 .MCALL .HEADER, .SWRHI, .EQUAT, .SETUP, .SCATCH, .SACT11
1606 .MCALL .SEOP, .SSAVE, .STYPOCT, .SCMTAG
1607 .MCALL .STYPDEC, .STRAP, .SPOWER, .SAPTHDR, .SAPTBL
1608 .MCALL .SAPTYPE, .SREAD
1609 .MCALL .EQUIV ;REMOVE FOR PDP-10
1610
.TITLE CKFPCCO FP11F FLTG PNT PRT C
;*COPYRIGHT (C) 1981
;*DIGITAL EQUIPMENT CORP.
;*MAYNARD, MASS. 01754
;*
;*
;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
;*PACKAGE (MAINDEC-11-DZQAC-C5), JAN, 1981.
;*
000001 $TN-1
1600C0 $SWR 160000 ;:HALT ON ERROR, LOOP ON TEST, INHIBIT ERROR TYP0UT

1611
1612
1613 000244 FPVECT=244
1614 000250 MMVECT=250
1615 177400 $SWR=177400
1616 00C200 $SWRMSK=200
1617 C00011 TAB=11
1618 000015 CRLF=15
1619
1620 .SBTTL BASIC DEFINITIONS
;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
001100 STACK= 1100
104000 ERROR=EMT
000004 SCOPE=IOT

;*MISCELLANEOUS DEFINITIONS
000011 HT= 11 ;:CODE FOR HORIZONTAL TAB
000012 LF= 12 ;:CODE FOR LINE FEED
000015 CR= 15 ;:CODE FOR CARRIAGE RETURN
000200 CRLF= 200 ;:CODE FOR CARRIAGE RETURN-LINE FEED
177776 PS= 177776 ;:PROCESSOR STATUS WORD
177776 PSW=PS
177774 STKLMT= 177774 ;:STACK LIMIT REGISTER
177772 PIRQ= 177772 ;:PROGRAM INTERRUPT REQUEST REGISTER
177570 DSWR= 177570 ;:HARDWARE SWITCH REGISTER
177570 DDISP= 177570 ;:HARDWARE DISPLAY REGISTER

;*GENERAL PURPOSE REGISTER DEFINITIONS
000000 R0= %0 ;:GENERAL REGISTER
000001 R1= %1 ;:GENERAL REGISTER
000002 R2= %2 ;:GENERAL REGISTER
000003 R3= %3 ;:GENERAL REGISTER
000004 R4= %4 ;:GENERAL REGISTER
000005 R5= %5 ;:GENERAL REGISTER
000006 R6= %6 ;:GENERAL REGISTER
000007 R7= %7 ;:GENERAL REGISTER
000006 SP= %6 ;:STACK POINTER
000007 PC= %7 ;:PROGRAM COUNTER

;*PRIORITY LEVEL DEFINITIONS
000000 PR0= 0 ;:PRIORITY LEVEL 0
000040 PR1= 40 ;:PRIORITY LEVEL 1
000100 PR2= 100 ;:PRIORITY LEVEL 2
000140 PR3= 140 ;:PRIORITY LEVEL 3

```



```
00020C PR4= 200 ::PRIORITY LEVEL 4
000240 PR5= 240 ::PRIORITY LEVEL 5
000300 PR6= 300 ::PRIORITY LEVEL 6
000340 PR7= 340 ::PRIORITY LEVEL 7
;*'SWITCH REGISTER' SWITCH DEFINITIONS
100000 SW15= 100000
040000 SW14= 40000
020000 SW13= 20000
010000 SW12= 10000
004000 SW11= 4000
002000 SW10= 2000
001000 SW09= 1000
000400 SW08= 400
000200 SW07= 200
000100 SW06= 100
000040 SW05= 40
000020 SW04= 20
000010 SW03= 10
000004 SW02= 4
000002 SW01= 2
000001 SW00= 1
001000 SW9=SW09
000400 SW8=SW08
000200 SW7=SW07
000100 SW6=SW06
000040 SW5=SW05
000020 SW4=SW04
000010 SW3=SW03
000004 SW2=SW02
000002 SW1=SW01
000001 SW0=SW00
;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
100000 BIT15= 100000
040000 BIT14= 40000
020000 BIT13= 20000
010000 BIT12= 10000
004000 BIT11= 4000
002000 BIT10= 2000
001000 BIT09= 1000
000400 BIT08= 400
000200 BIT07= 200
000100 BIT06= 100
000040 BIT05= 40
000020 BIT04= 20
000010 BIT03= 10
000004 BIT02= 4
000002 BIT01= 2
000001 BIT00= 1
001000 BIT9=BIT09
000400 BIT8=BIT08
000200 BIT7=BIT07
000100 BIT6=BIT06
000040 BIT5=BIT05
000020 BIT4=BIT04
000010 BIT3=BIT03
000004 BIT2=BIT02
000002 BIT1=BIT01
```

```
000001          BIT0=BIT00
;*BASIC "CPU" TRAP VECTOR ADDRESSES
000004  ERRVEC= 4          ;; TIME OUT AND OTHER ERRORS
000010  RESVEC= 10         ;; RESERVED AND ILLEGAL INSTRUCTIONS
000014  TBITVEC=14        ;; 'T' BIT
000014  TRIVEC= 14        ;; TRACE TRAP
000014  BPTVEC= 14        ;; BREAKPOINT TRAP (BPT)
000020  IOTVEC= 20        ;; INPUT/OUTPUT TRAP (IOT) **SCOPE**
000024  PWRVEC= 24        ;; POWER FAIL
000030  EMTVEC= 30        ;; EMULATOR TRAP (EMT) **ERROR**
000034  TRAPVEC=34        ;; "TRAP" TRAP
000060  TKVEC= 60         ;; TTY KEYBOARD VECTOR
000064  TPVEC= 64         ;; TTY PRINTER VECTOR
000240  PIRQVEC=240       ;; PROGRAM INTERRUPT REQUEST VECTOR

. SBTTL FPP REGISTER DEFINITIONS
1621          ACO          =%0
1622          AC1          =%1
1623          AC2          =%2
1624          AC3          =%3
1625          AC4          =%4
1626          AC5          =%5
1627          AC6          =%6
1628          AC7          =%7
1629          KIPDR0       =172300
1630          KIPDR1       =172302
1631          KIPDR2       =172304
1632          KIPDR3       =172306
1633          KIPDR4       =172310
1634          KIPDR7       =172316
1635          KIPAR0       =172340
1636          KIPAR1       =172342
1637          KIPAR2       =172344
1638          KIPAR3       =172346
1639          KIPAR4       =172350
1640          KIPAR7       =172356
1641          KDPDR0       =172320
1642          KDPDR1       =172322
1643          KDPDR2       =172324
1644          KDPDR3       =172326
1645          KDPDR4       =172330
1646          KDPDR7       =172336
1647          KDPAR0       =172360
1648          KDPAR1       =172362
1649          KDPAR2       =172364
1650          KDPAR3       =172366
1651          KDPAR4       =172370
1652          KDPAR7       =172376
1653          MMR0         =177572
1654          SR1          =177574
1655          MMR2         =177576
1656          MMR3         =172516
1657          DATA       =117760
1658          IOTRAP      =000020
1659
1660
1661
1662          . SBTTL TRAP CATCHER
1663          .=0
000000
```

;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS



000174 000174
000174 000000
000176 000000
000200 000137 006116

DISPREG: .WORD 0 ;;SOFTWARE DISPLAY REGISTER
SWREG: .WORD 0 ;;SOFTWARE SWITCH REGISTER
.SBTTL STARTING ADDRESS(ES)
JMP @*START ;;JUMP TO STARTING ADDRESS OF PROGRAM

1664

.SBTTL COMMON TAGS

*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
*USED IN THE PROGRAM.

Address	Value	Variable Name	Type	Value	Description
001100	001100	\$CMTAG:	.WORD	0	::START OF COMMON TAGS
001100	000000	\$TSTNM:	.BYTE	0	::CONTAINS THE TEST NUMBER
001102	000	\$ERFLG:	.BYTE	0	::CONTAINS ERROR FLAG
001103	000	\$ICNT:	.WORD	0	::CONTAINS SUBTEST ITERATION COUNT
001104	000000	\$LPADR:	.WORD	0	::CONTAINS SCOPE LOOP ADDRESS
001106	000000	\$LPERR:	.WORD	0	::CONTAINS SCOPE RETURN FOR ERRORS
001110	000000	\$ERTTL:	.WORD	0	::CONTAINS TOTAL ERRORS DETECTED
001112	000000	\$ITE'NB:	.BYTE	0	::CONTAINS ITEM CONTROL BYTE
001114	000	\$ERMAX:	.BYTE	1	::CONTAINS MAX. ERRORS PER TEST
001115	001	\$ERRPC:	.WORD	0	::CONTAINS PC OF LAST ERROR INSTRUCTION
001116	000000	\$GDADR:	.WORD	0	::CONTAINS ADDRESS OF 'GOOD' DATA
001120	000000	\$BDADR:	.WORD	0	::CONTAINS ADDRESS OF 'BAD' DATA
001122	000000	\$GDDAT:	.WORD	0	::CONTAINS 'GOOD' DATA
001124	000000	\$BDDAT:	.WORD	0	::CONTAINS 'BAD' DATA
0C1126	000000		.WORD	0	::RESERVED--NOT TO BE USED
001130	000000		.WORD	0	
001132	000000		.WORD	0	
001134	000	\$AUTOB:	.BYTE	0	::AUTOMATIC MODE INDICATOR
001135	000	\$INTAG:	.BYTE	0	::INTERRUPT MODE INDICATOR
001136	000000		.WORD	0	
001140	177570	SWR:	.WORD	DSWR	::ADDRESS OF SWITCH REGISTER
001142	177570	DISPLAY:	.WORD	DDISP	::ADDRESS OF DISPLAY REGISTER
001144	177560	\$TKS:			::TTY KBD STATUS
001146	177562	\$TKB:			::TTY KBD BUFFER
001150	177564	\$TPS:			::TTY PRINTER STATUS REG. ADDRESS
001152	177566	\$TPB:			::TTY PRINTER BUFFER REG. ADDRESS
001154	000	\$NULL:	.BYTE	0	::CONTAINS NULL CHARACTER FOR FILLS
001155	002	\$FILLS:	.BYTE	2	::CONTAINS # OF FILLER CHARACTERS REQUIRED
001156	012	\$FILLC:	.BYTE	12	::INSERT FILL CHARS. AFTER A 'LINE FEED'
001157	000	\$TPFLG:	.BYTE	0	::'TERMINAL AVAILABLE' FLAG (BIT<07>-0=YES)
001160	000000	\$REGAD:	.WORD	0	::CONTAINS THE ADDRESS FROM WHICH (\$REGO) WAS OBTAINED
	000024		.REPT	\$CM3	
001162	000000	\$REG0:	.WORD	0	::CONTAINS ((\$REGAD)+0)
001164	000000	\$REG1:	.WORD	0	::CONTAINS ((\$REGAD)+2)
001166	000000	\$REG2:	.WORD	0	::CONTAINS ((\$REGAD)+4)
001170	000000	\$REG3:	.WORD	0	::CONTAINS ((\$REGAD)+6)
001172	000000	\$REG4:	.WORD	0	::CONTAINS ((\$REGAD)+10)
001174	000000	\$REG5:	.WORD	0	::CONTAINS ((\$REGAD)+12)
001176	000000	\$REG6:	.WORD	0	::CONTAINS ((\$REGAD)+14)
001200	000000	\$REG7:	.WORD	0	::CONTAINS ((\$REGAD)+16)
001202	000000	\$REG10:	.WORD	0	::CONTAINS ((\$REGAD)+20)
001204	000000	\$REG11:	.WORD	0	::CONTAINS ((\$REGAD)+22)
001206	000000	\$REG12:	.WORD	0	::CONTAINS ((\$REGAD)+24)
001210	000000	\$REG13:	.WORD	0	::CONTAINS ((\$REGAD)+26)
001212	000000	\$REG14:	.WORD	0	::CONTAINS ((\$REGAD)+30)
001214	000000	\$REG15:	.WORD	0	::CONTAINS ((\$REGAD)+32)
001216	000000	\$REG16:	.WORD	0	::CONTAINS ((\$REGAD)+34)
001220	000000	\$REG17:	.WORD	0	::CONTAINS ((\$REGAD)+36)
001222	000000	\$REG20:	.WORD	0	::CONTAINS ((\$REGAD)+40)
001224	000000	\$REG21:	.WORD	0	::CONTAINS ((\$REGAD)+42)
001226	000000	\$REG22:	.WORD	0	::CONTAINS ((\$REGAD)+44)

```
001230 000000 $REG23: .WORD 0 ;;CONTAINS (($REGAD)+46)
001232 000024 .REPT 24
001234 000000 $TMP0: .WORD 0 ;;USER DEFINED
001236 000000 $TMP1: .WORD 0 ;;USER DEFINED
001240 000000 $TMP2: .WORD 0 ;;USER DEFINED
001242 000000 $TMP3: .WORD 0 ;;USER DEFINED
001244 000000 $TMP4: .WORD 0 ;;USER DEFINED
001246 000000 $TMP5: .WORD 0 ;;USER DEFINED
001250 000000 $TMP6: .WORD 0 ;;USER DEFINED
001252 000000 $TMP7: .WORD 0 ;;USER DEFINED
001254 000000 $TMP10: .WORD 0 ;;USER DEFINED
001256 000000 $TMP11: .WORD 0 ;;USER DEFINED
001260 000000 $TMP12: .WORD 0 ;;USER DEFINED
001262 000000 $TMP13: .WORD 0 ;;USER DEFINED
001264 000000 $TMP14: .WORD 0 ;;USER DEFINED
001266 000000 $TMP15: .WORD 0 ;;USER DEFINED
001270 000000 $TMP16: .WORD 0 ;;USER DEFINED
001272 000000 $TMP17: .WORD 0 ;;USER DEFINED
001274 000000 $TMP20: .WORD 0 ;;USER DEFINED
001276 000000 $TMP21: .WORD 0 ;;USER DEFINED
001300 000000 $TMP22: .WORD 0 ;;USER DEFINED
001302 000000 $TMP23: .WORD 0 ;;USER DEFINED
001304 000000 $TIMES: 0 ;;MAX. NUMBER OF ITERATIONS
001306 207 377 377 $ESCAPE: 0 ;;ESCAPE ON ERROR ADDRESS
001312 077 $BELL: .ASCII <207><377><377> ;;CODE FOR BELL
001313 015 $QUES: .ASCII /?/ ;;QUESTION MARK
001314 012 000 $CRLF: .ASCII <15> ;;CARRIAGE RETURN
$LF: .ASCII <12> ;;LINE FEED
*****
.SBTTL APT MAILBOX-ETABLE
*****
.EVEN
001316 $MAIL: ;;APT MAILBOX
001316 000000 $MSGTY: .WORD AMSGTY ;;MESSAGE TYPE CODE
001320 000000 $FATAL: .WORD AFATAL ;;FATAL ERROR NUMBER
001322 000000 $TESTN: .WORD ATESTN ;;TEST NUMBER
001324 000000 $PASS: .WORD APASS ;;PASS COUNT
001326 000000 $DEVCT: .WORD ADEVCT ;;DEVICE COUNT
001330 000000 $UNIT: .WORD AUNIT ;;I/O UNIT NUMBER
001332 000000 $MSGAD: .WORD AMSGAD ;;MESSAGE ADDRESS
001334 000000 $MSGLG: .WORD AMSGLG ;;MESSAGE LENGTH
001336 $ETABLE: ;;APT ENVIRONMENT TABLE
001336 000 $ENV: .BYTE AENV ;;ENVIRONMENT BYTE
001337 000 $ENVM: .BYTE AENVM ;;ENVIRONMENT MODE BITS
001340 000000 $SWREG: .WORD ASWREG ;;APT SWITCH REGISTER
001342 000000 $USWR: .WORD AUSWR ;;USER SWITCHES
001344 000000 $CPUOP: .WORD ACPUOP ;;CPU TYPE, OPTIONS
*
* BITS 15-11=CPU TYPE
* 11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
* 11/70=06,PDQ=07,Q=10
*
* BIT 10-REAL TIME CLOCK
* BIT 9=FLOATING POINT PROCESSOR
* BIT 8=MEMORY MANAGEMENT
001346 000 $MAMS1: .BYTE AMAMS1 ;;HIGH ADDRESS,M.S. BYTE
001347 000 $MTYP1: .BYTE AMTYP1 ;;MEM. TYPE,BLK#1
*
* MEM. TYPE BYTE -- (HIGH BYTE)
* 900 NSEC CORE 001
```

```

: *
: *
: *
001350 000000 $MADR1: .WORD AMADR1 ;;HIGH ADDRESS,BLK#1
: *
: *
: *
001352 000 $MAMS2: .BYTE AMAMS2 ;;HIGH ADDRESS,M.S. BYTE
001353 000 $MTYP2: .BYTE AMTYP2 ;;MEM.TYPE,BLK#2
001354 000000 $MADR2: .WORD AMADR2 ;;MEM.LAST ADDRESS,BLK#2
001356 000 $MAMS3: .BYTE AMAMS3 ;;HIGH ADDRESS,M.S.BYTE
001357 000 $MTYP3: .BYTE AMTYP3 ;;MEM.TYPE,BLK#3
001360 000000 $MADR3: .WORD AMADR3 ;;MEM.LAST ADDRESS,BLK#3
001362 000 $MAMS4: .BYTE AMAMS4 ;;HIGH ADDRESS,M.S.BYTE
001363 000 $MTYP4: .BYTE AMTYP4 ;;MEM.TYPE,BLK#4
001364 000000 $MADR4: .WORD AMADR4 ;;MEM.LAST ADDRESS,BLK#4
001366 000000 $VECT1: .WORD AVECT1 ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
001370 000000 $VECT2: .WORD AVECT2 ;;INTERRUPT VECTOR#2BUS PRIORITY#2
001372 000000 $BASE: .WORD ABASE ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
001374 000000 $DEVM: .WORD ADEVM ;;DEVICE MAP
001376 000000 $CDW1: .WORD ACDW1 ;;CONTROLLER DESCRIPTION WORD#1
001400 000000 $CDW2: .WORD ACDW2 ;;CONTROLLER DESCRIPTION WORD#2
001402 000000 $DDW0: .WORD ADDW0 ;;DEVICE DESCRIPTOR WORD#0
001404 000000 $DDW1: .WORD ADDW1 ;;DEVICE DESCRIPTOR WORD#1
001406 000000 $DDW2: .WORD ADDW2 ;;DEVICE DESCRIPTOR WORD#2
001410 000000 $DDW3: .WORD ADDW3 ;;DEVICE DESCRIPTOR WORD#3
001412 000000 $DDW4: .WORD ADDW4 ;;DEVICE DESCRIPTOR WORD#4
001414 000000 $DDW5: .WORD ADDW5 ;;DEVICE DESCRIPTOR WORD#5
001416 000000 $DDW6: .WORD ADDW6 ;;DEVICE DESCRIPTOR WORD#6
001420 000000 $DDW7: .WORD ADDW7 ;;DEVICE DESCRIPTOR WORD#7
001422 000000 $DDW8: .WORD ADDW8 ;;DEVICE DESCRIPTOR WORD#8
001424 000000 $DDW9: .WORD ADDW9 ;;DEVICE DESCRIPTOR WORD#9
001426 000000 $DDW10: .WORD ADDW10 ;;DEVICE DESCRIPTOR WORD#10
001430 000000 $DDW11: .WORD ADDW11 ;;DEVICE DESCRIPTOR WORD#11
001432 000000 $DDW12: .WORD ADDW12 ;;DEVICE DESCRIPTOR WORD#12
001434 000000 $DDW13: .WORD ADDW13 ;;DEVICE DESCRIPTOR WORD#13
001436 000000 $DDW14: .WORD ADDW14 ;;DEVICE DESCRIPTOR WORD#14
001440 000000 $DDW15: .WORD ADDW15 ;;DEVICE DESCRIPTOR WORD#15
001442 $ETEND:

```

```

300 NSEC BIPOLAR=00?
500 NSEC MOS-003
MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE

```


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

REPT	MNUMBER	EM	DH	DT	DF	ITEM
1668 001442	000444					
1670 001442	047662	066370	070212			: ITEM 1
001452	047736	066460	070232			: ITEM 2
001462	047766	066524	070254			: ITEM 3
001472	050016	066570	070276			: ITEM 4
001502	050077	066633	070320			: ITEM 5
001512	050110	066633	070346			: ITEM 6
001522	050253	066570	070276			: ITEM 7
001532	050275	066633	070320			: ITEM 10
001542	050310	066570	070276			: ITEM 11
001552	050340	066633	070320			: ITEM 12
001562	050352	066644	070346			: ITEM 13
001572	050352	066644	070346			: ITEM 14
001602	050420	066633	070320			: ITEM 15
001612	050432	066704	070360			: ITEM 16
001622	050456	066644	070346			: ITEM 17
001632	050502	066570	070360			: ITEM 20
001642	050520	066633	070320			: ITEM 21
001652	050520	066633	070320			: ITEM 22
001662	050532	066644	070346			: ITEM 23
001672	050557	066570	070360			: ITEM 24
001702	050574	066633	070320			: ITEM 25
001712	050606	066644	070346			: ITEM 26
001722	050633	066570	070360			: ITEM 27
001732	050650	066633	070320			: ITEM 30
001742	050662	066644	070346			: ITEM 31
001752	050706	066570	070360			: ITEM 32
001762	050724	066633	070320			: ITEM 33
001772	050736	066644	070346			: ITEM 34
002002	050763	066570	070360			: ITEM 35
002012	051000	066633	070320			: ITEM 36
002022	051012	066747	070402			: ITEM 37
002032	051072	066747	070402			: ITEM 40
002042	051072	067012	070446			: ITEM 41
002052	051012	066747	070402			: ITEM 42
002062	051072	066747	070402			: ITEM 43
002072	051012	066747	070402			: ITEM 44
002102	051012	066747	070402			: ITEM 45
002112	051072	066747	070402			: ITEM 46
002122	051012	066747	070402			: ITEM 47
002132	051012	066747	070402			: ITEM 50
002142	051551	066747	070402			: ITEM 51
002152	051610	066747	070402			: ITEM 52
002162	051610	067012	070446			: ITEM 53
002172	051551	066747	070402			: ITEM 54
002202	051551	066747	070402			: ITEM 55

002212	051610	066747	070402	.WORD	EM56,DH56,DT56,DF56	:ITEM 56
002222	051610	066747	070402	.WORD	EM57,DH57,DT57,DF57	:ITEM 57
002232	051551	066747	070402	.WORD	EM60,DH60,DT60,DF60	:ITEM 60
002242	051610	066747	070402	.WORD	EM61,DH61,DT61,DF61	:ITEM 61
002252	052305	03460	070360	.WORD	EM62,DH62,DT62,DF62	:ITEM 62
002262	052403	066524	070360	.WORD	EM63,DH63,DT63,DF63	:ITEM 63
002272	052420	066633	070320	.WORD	EM64,DH64,DT64,DF64	:ITEM 64
002302	052470	066460	070360	.WORD	EM65,DH65,DT65,DF65	:ITEM 65
002312	052514	066570	070276	.WORD	EM66,DH66,DT66,DF66	:ITEM 66
002322	052542	066460	070276	.WORD	EM67,DH67,DT67,DF67	:ITEM 67
002332	052624	066524	070276	.WORD	EM70,DH70,DT70,DF70	:ITEM 70
002342	052647	066633	070522	.WORD	EM71,DH71,DT71,DF71	:ITEM 71
002352	052664	066460	070276	.WORD	EM72,DH72,DT72,DF72	:ITEM 72
002362	052700	066633	070556	.WORD	EM73,DH73,DT73,DF73	:ITEM 73
002372	052714	066570	070276	.WORD	EM74,DH74,DT74,DF74	:ITEM 74
002402	052730	066460	070232	.WORD	EM75,DH75,DT75,DF75	:ITEM 75
002412	052744	066644	070346	.WORD	EM76,DH76,DT76,DF76	:ITEM 76
002422	053017	066633	070556	.WORD	EM77,DH77,DT77,DF77	:ITEM 77
002432	053032	066570	070276	.WORD	EM100,DH100,DT100,DF100	:ITEM 100
002442	053046	066460	070232	.WORD	EM101,DH101,DT101,DF101	:ITEM 101
002452	053062	066644	070346	.WORD	EM102,DH102,DT102,DF102	:ITEM 102
002462	053105	066633	070556	.WORD	EM103,DH103,DT103,DF103	:ITEM 103
002472	053120	066570	070276	.WORD	EM104,DH104,DT104,DF104	:ITEM 104
002502	053134	066460	070232	.WORD	EM105,DH105,DT105,DF105	:ITEM 105
002512	053150	066644	070346	.WORD	EM106,DH106,DT106,DF106	:ITEM 106
002522	053174	066644	070346	.WORD	EM107,DH107,DT107,DF107	:ITEM 107
002532	053210	066633	070556	.WORD	EM110,DH110,DT110,DF110	:ITEM 110
002542	053224	066570	070276	.WORD	EM111,DH111,DT111,DF111	:ITEM 111
002552	053240	066460	070232	.WORD	EM112,DH112,DT112,DF112	:ITEM 112
002562	053254	066644	070346	.WORD	EM113,DH113,DT113,DF113	:ITEM 113
002572	053300	066633	070556	.WORD	EM114,DH114,DT114,DF114	:ITEM 114
002602	053314	066570	070276	.WORD	EM115,DH115,DT115,DF115	:ITEM 115
002612	053330	066460	070232	.WORD	EM116,DH116,DT116,DF116	:ITEM 116
002622	053344	066644	070346	.WORD	EM117,DH117,DT117,DF117	:ITEM 117
002632	053367	066633	070556	.WORD	EM120,DH120,DT120,DF120	:ITEM 120
002642	053402	066570	070276	.WORD	EM121,DH121,DT121,DF121	:ITEM 121
002652	053416	066460	070232	.WORD	EM122,DH122,DT122,DF122	:ITEM 122
002662	053432	066644	070346	.WORD	EM123,DH123,DT123,DF123	:ITEM 123
002672	053456	066633	070556	.WORD	EM124,DH124,DT124,DF124	:ITEM 124
002702	053472	066570	070276	.WORD	EM125,DH125,DT125,DF125	:ITEM 125
002712	053506	066460	070232	.WORD	EM126,DH126,DT126,DF126	:ITEM 126
002722	053522	066644	070346	.WORD	EM127,DH127,DT127,DF127	:ITEM 127
002732	053546	066633	070556	.WORD	EM130,DH130,DT130,DF130	:ITEM 130
002742	053562	066460	070232	.WORD	EM131,DH131,DT131,DF131	:ITEM 131
002752	053576	066644	070346	.WORD	EM132,DH132,DT132,DF132	:ITEM 132
002762	053623	066633	070556	.WORD	EM133,DH133,DT133,DF133	:ITEM 133
002772	053636	066460	070232	.WORD	EM134,DH134,DT134,DF134	:ITEM 134
003002	053652	066633	070320	.WORD	EM135,DH135,DT135,DF135	:ITEM 135
003012	053722	066633	070320	.WORD	EM136,DH136,DT136,DF136	:ITEM 136
003022	053736	066460	070360	.WORD	EM137,DH137,DT137,DF137	:ITEM 137
003032	053752	066633	070320	.WORD	EM140,DH140,DT140,DF140	:ITEM 140
003042	053766	066570	070276	.WORD	EM141,DH141,DT141,DF141	:ITEM 141
003052	054024	066460	070276	.WORD	EM142,DH142,DT142,DF142	:ITEM 142
003062	054040	066533	070320	.WORD	EM143,DH143,DT143,DF143	:ITEM 143
003072	054054	066570	070276	.WORD	EM144,DH144,DT144,DF144	:ITEM 144
003102	054072	066460	070276	.WORD	EM145,DH145,DT145,DF145	:ITEM 145
003112	054106	066633	070320	.WORD	EM146,DH146,DT146,DF146	:ITEM 146

ERROR POINTER TABLE

003122	054122	066570	070276	.WORD	EM147, DH147, DT147, DF147	:ITEM 147
003132	054142	066460	070276	.WORD	EM150, DH150, DT150, DF150	:ITEM 150
003142	054156	066633	070320	.WORD	EM151, DH151, DT151, DF151	:ITEM 151
003152	054172	066570	070276	.WORD	EM152, DH152, DT152, DF152	:ITEM 152
003162	054212	066460	070276	.WORD	EM153, DH153, DT153, DF153	:ITEM 153
003172	054226	066633	070320	.WORD	EM154, DH154, DT154, DF154	:ITEM 154
003202	054242	066570	070276	.WORD	EM155, DH155, DT155, DF155	:ITEM 155
003212	054262	066460	070276	.WORD	EM156, DH156, DT156, DF156	:ITEM 156
003222	054276	066633	070320	.WORD	EM157, DH157, DT157, DF157	:ITEM 157
003232	054321	066570	070276	.WORD	EM160, DH160, DT160, DF160	:ITEM 160
003242	054367	066460	070276	.WORD	EM161, DH161, DT161, DF161	:ITEM 161
003252	054402	066633	070320	.WORD	EM162, DH162, DT162, DF162	:ITEM 162
003262	054426	066460	070276	.WORD	EM163, DH163, DT163, DF163	:ITEM 163
003272	054442	066704	070276	.WORD	EM164, DH164, DT164, DF164	:ITEM 164
003302	054472	066747	070402	.WORD	EM165, DH165, DT165, DF165	:ITEM 165
003312	054506	066747	070402	.WORD	EM166, DH166, DT166, DF166	:ITEM 166
003322	054522	066747	070402	.WORD	EM167, DH167, DT167, DF167	:ITEM 167
003332	054536	066747	070402	.WORD	EM170, DH170, DT170, DF170	:ITEM 170
003342	054552	066747	070402	.WORD	EM171, DH171, DT171, DF171	:ITEM 171
003352	054566	066747	070402	.WORD	EM172, DH172, DT172, DF172	:ITEM 172
003362	054602	067012	070446	.WORD	EM173, DH173, DT173, DF173	:ITEM 173
003372	054616	067012	070446	.WORD	EM174, DH174, DT174, DF174	:ITEM 174
003402	054632	067012	070446	.WORD	EM175, DH175, DT175, DF175	:ITEM 175
003412	054646	066460	070276	.WORD	EM176, DH176, DT176, DF176	:ITEM 176
003422	054670	067063	070512	.WORD	EM177, DH177, DT177, DF177	:ITEM 177
003432	054724	066747	070402	.WORD	EM200, DH200, DT200, DF200	:ITEM 200
003442	054777	066747	070402	.WORD	EM201, DH201, DT201, DF201	:ITEM 201
003452	055050	066747	070402	.WORD	EM202, DH202, DT202, DF202	:ITEM 202
003462	055122	066747	070402	.WORD	EM203, DH203, DT203, DF203	:ITEM 203
003472	055244	066747	070402	.WORD	EM204, DH204, DT204, DF204	:ITEM 204
003502	055317	066747	070402	.WORD	EM205, DH205, DT205, DF205	:ITEM 205
003512	055370	066747	070402	.WORD	EM206, DH206, DT206, DF206	:ITEM 206
003522	055442	066747	070402	.WORD	EM207, DH207, DT207, DF207	:ITEM 207
003532	055514	066747	070402	.WORD	EM210, DH210, DT210, DF210	:ITEM 210
003542	055566	066747	070402	.WORD	EM211, DH211, DT211, DF211	:ITEM 211
003552	055645	066747	070402	.WORD	EM212, DH212, DT212, DF212	:ITEM 212
003562	055716	066747	070402	.WORD	EM213, DH213, DT213, DF213	:ITEM 213
003572	056027	066747	070402	.WORD	EM214, DH214, DT214, DF214	:ITEM 214
003602	056114	066704	070276	.WORD	EM215, DH215, DT215, DF215	:ITEM 215
003612	056251	066633	070320	.WORD	EM216, DH216, DT216, DF216	:ITEM 216
003622	056264	066570	070276	.WORD	EM217, DH217, DT217, DF217	:ITEM 217
003632	056304	066460	070276	.WORD	EM220, DH220, DT220, DF220	:ITEM 220
003642	056320	066704	070276	.WORD	EM221, DH221, DT221, DF221	:ITEM 221
003652	056370	066633	070320	.WORD	EM222, DH222, DT222, DF222	:ITEM 222
003662	056404	066570	070276	.WORD	EM223, DH223, DT223, DF223	:ITEM 223
003672	056424	066460	070276	.WORD	EM224, DH224, DT224, DF224	:ITEM 224
003702	056440	066570	070276	.WORD	EM225, DH225, DT225, DF225	:ITEM 225
003712	056463	066460	070276	.WORD	EM226, DH226, DT226, DF226	:ITEM 226
003722	056474	067112	070346	.WORD	EM227, DH227, DT227, DF227	:ITEM 227
003732	056525	066570	070276	.WORD	EM230, DH230, DT230, DF230	:ITEM 230
003742	056550	066460	070276	.WORD	EM231, DH231, DT231, DF231	:ITEM 231
003752	056562	067112	070346	.WORD	EM232, DH232, DT232, DF232	:ITEM 232
003762	056574	066570	070276	.WORD	EM233, DH233, DT233, DF233	:ITEM 233
003772	056620	066460	070276	.WORD	EM234, DH234, DT234, DF234	:ITEM 234
004002	056632	067112	070346	.WORD	EM235, DH235, DT235, DF235	:ITEM 235
004012	056644	066570	070276	.WORD	EM236, DH236, DT236, DF236	:ITEM 236
004022	056671	066460	070276	.WORD	EM237, DH237, DT237, DF237	:ITEM 237

ERROR POINTER TABLE

004032	056702	067112	070346	.WORD	EM240,DH240,DT240,DF240	:ITEM 240
004042	056714	066570	070275	.WORD	EM241,DH241,DT241,DF241	:ITEM 241
004052	056741	066460	070276	.WORD	EM242,DH242,DT242,DF242	:ITEM 242
004062	056752	067112	070346	.WORD	EM243,DH243,DT243,DF243	:ITEM 243
004072	056764	066570	070276	.WORD	EM244,DH244,DT244,DF244	:ITEM 244
004102	057010	066460	070276	.WORD	EM245,DH245,DT245,DF245	:ITEM 245
004112	057022	066704	070276	.WORD	EM246,DH246,DT246,DF246	:ITEM 246
004122	057073	067112	070346	.WORD	EM247,DH247,DT247,DF247	:ITEM 247
004132	057104	066570	070276	.WORD	EM250,DH250,DT250,DF250	:ITEM 250
004142	057131	066460	070276	.WORD	EM251,DH251,DT251,DF251	:ITEM 251
004152	057022	066704	070276	.WORD	EM252,DH252,DT252,DF252	:ITEM 252
004162	057156	067112	070346	.WORD	EM253,DH253,DT253,DF253	:ITEM 253
004172	057022	066704	070276	.WORD	EM254,DH254,DT254,DF254	:ITEM 254
004202	057206	067112	070346	.WORD	EM255,DH255,DT255,DF255	:ITEM 255
004212	057232	066570	070276	.WORD	EM256,DH256,DT256,DF256	:ITEM 256
004222	057260	066460	070276	.WORD	EM257,DH257,DT257,DF257	:ITEM 257
004232	057272	066747	070402	.WORD	EM260,DH260,DT260,DF260	:ITEM 260
004242	057330	066747	070402	.WORD	EM261,DH261,DT261,DF261	:ITEM 261
004252	057342	066747	070402	.WORD	EM262,DH262,DT262,DF262	:ITEM 262
004262	057430	066747	070402	.WORD	EM263,DH263,DT263,DF263	:ITEM 263
004272	057455	066747	070402	.WORD	EM264,DH264,DT264,DF264	:ITEM 264
004302	057542	066747	070402	.WORD	EM265,DH265,DT265,DF265	:ITEM 265
004312	057613	066747	070402	.WORD	EM266,DH266,DT266,DF266	:ITEM 266
004322	057667	066747	070402	.WORD	EM267,DH267,DT267,DF267	:ITEM 267
004332	057754	066747	070402	.WORD	EM270,DH270,DT270,DF270	:ITEM 270
004342	060006	066747	070402	.WORD	EM271,DH271,DT271,DF271	:ITEM 271
004352	060045	066747	070402	.WORD	EM272,DH272,DT272,DF272	:ITEM 272
004362	060115	066747	070402	.WORD	EM273,DH273,DT273,DF273	:ITEM 273
004372	060152	066747	070402	.WORD	EM274,DH274,DT274,DF274	:ITEM 274
004402	060164	066747	070402	.WORD	EM275,DH275,DT275,DF275	:ITEM 275
004412	060247	066747	070402	.WORD	EM276,DH276,DT276,DF276	:ITEM 276
004422	060334	066747	070402	.WORD	EM277,DH277,DT277,DF277	:ITEM 277
004432	060401	066747	070402	.WORD	EM300,DH300,DT300,DF300	:ITEM 300
004442	060476	066747	070612	.WORD	EM301,DH301,DT301,DF301	:ITEM 301
004452	060523	066747	070612	.WORD	EM302,DH302,DT302,DF302	:ITEM 302
004462	060534	067012	070664	.WORD	EM303,DH303,DT303,DF303	:ITEM 303
004472	060546	066747	070612	.WORD	EM304,DH304,DT304,DF304	:ITEM 304
004502	060626	066747	070612	.WORD	EM305,DH305,DT305,DF305	:ITEM 305
004512	060722	066747	070612	.WORD	EM306,DH306,DT306,DF306	:ITEM 306
004522	061030	066747	070612	.WORD	EM307,DH307,DT307,DF307	:ITEM 307
004532	061100	066747	070612	.WORD	EM310,DH310,DT310,DF310	:ITEM 310
004542	061154	066747	070612	.WORD	EM311,DH311,DT311,DF311	:ITEM 311
004552	061224	066747	070612	.WORD	EM312,DH312,DT312,DF312	:ITEM 312
004562	061274	066747	070612	.WORD	EM313,DH313,DT313,DF313	:ITEM 313
004572	061344	066747	070612	.WORD	EM314,DH314,DT314,DF314	:ITEM 314
004602	061414	066747	070612	.WORD	EM315,DH315,DT315,DF315	:ITEM 315
004612	061464	066747	070612	.WORD	EM316,DH316,DT316,DF316	:ITEM 316
004622	061534	066747	070612	.WORD	EM317,DH317,DT317,DF317	:ITEM 317
004632	061604	066747	070612	.WORD	EM320,DH320,DT320,DF320	:ITEM 320
004642	061654	066747	070612	.WORD	EM321,DH321,DT321,DF321	:ITEM 321
004652	061724	066747	070736	.WORD	EM322,DH322,DT322,DF322	:ITEM 322
004662	061762	066747	070736	.WORD	EM323,DH323,DT323,DF323	:ITEM 323
004672	061774	067012	071002	.WORD	EM324,DH324,DT324,DF324	:ITEM 324
004702	062006	066747	070736	.WORD	EM325,DH325,DT325,DF325	:ITEM 325
004712	062006	066747	070736	.WORD	EM326,DH326,DT326,DF326	:ITEM 326
004722	062132	066747	070736	.WORD	EM327,DH327,DT327,DF327	:ITEM 327
004732	062202	066747	070736	.WORD	EM330,DH330,DT330,DF330	:ITEM 330

004742	062256	066747	070736	.WORD	EM331,DH331,DT331,DF331	:ITEM 331
004752	062326	066747	070736	.WORD	EM332,DH332,DT332,DF332	:ITEM 332
004762	061762	066747	070736	.WORD	EM333,DH333,DT333,DF333	:ITEM 333
004772	062424	066747	070736	.WORD	EM334,DH334,DT334,DF334	:ITEM 334
005002	062507	066747	070736	.WORD	EM335,DH335,DT335,DF335	:ITEM 335
005012	062556	066747	070736	.WORD	EM336,DH336,DT336,DF336	:ITEM 336
005022	062613	066747	070736	.WORD	EM337,DH337,DT337,DF337	:ITEM 337
005032	062667	066747	070736	.WORD	EM340,DH340,DT340,DF340	:ITEM 340
005042	062736	066747	070736	.WORD	EM341,DH341,DT341,DF341	:ITEM 341
005052	063006	066747	070736	.WORD	EM342,DH342,DT342,DF342	:ITEM 342
005062	063056	066747	070736	.WORD	EM343,DH343,DT343,DF343	:ITEM 343
005072	063133	066747	070736	.WORD	EM344,DH344,DT344,DF344	:ITEM 344
005102	063240	066747	070736	.WORD	EM345,DH345,DT345,DF345	:ITEM 345
005112	063310	066747	070736	.WORD	EM346,DH346,DT346,DF346	:ITEM 346
005122	063407	066747	070736	.WORD	EM347,DH347,DT347,DF347	:ITEM 347
005132	063433	066747	070736	.WORD	EM350,DH350,DT350,DF350	:ITEM 350
005142	063444	066644	070346	.WORD	EM351,DH351,DT351,DF351	:ITEM 351
005152	063546	066747	070736	.WORD	EM352,DH352,DT352,DF352	:ITEM 352
005162	063616	066747	070736	.WORD	EM353,DH353,DT353,DF353	:ITEM 353
005172	063666	066747	070736	.WORD	EM354,DH354,DT354,DF354	:ITEM 354
005202	063736	066747	070736	.WORD	EM355,DH355,DT355,DF355	:ITEM 355
005212	064006	066570	070232	.WORD	EM356,DH356,DT356,DF356	:ITEM 356
005222	064112	067132	070254	.WORD	EM357,DH357,DT357,DF357	:ITEM 357
005232	064146	066644	070346	.WORD	EM360,DH360,DT360,DF360	:ITEM 360
005242	064236	066460	070612	.WORD	EM361,DH361,DT361,DF361	:ITEM 361
005252	064254	067176	071046	.WORD	EM362,DH362,DT362,DF362	:ITEM 362
005262	064364	067220	071064	.WORD	EM363,DH363,DT363,DF363	:ITEM 363
005272	064432	067260	071106	.WORD	EM364,DH364,DT364,DF364	:ITEM 364
005302	064532	067327	070346	.WORD	EM365,DH365,DT365,DF365	:ITEM 365
005312	064615	067340	071120	.WORD	EM366,DH366,DT366,DF366	:ITEM 366
005322	064700	067372	071166	.WORD	EM367,DH367,DT367,DF367	:ITEM 367
005332	064763	067437	071212	.WORD	EM370,DH370,DT370,DF370	:ITEM 370
005342	000000	000000	000000	.WORD	EM371,DH371,DT371,DF371	:ITEM 371
005352	000000	000000	000000	.WORD	EM372,DH372,DT372,DF372	:ITEM 372
005362	000000	000000	000000	.WORD	EM373,DH373,DT373,DF373	:ITEM 373
005372	000000	000000	000000	.WORD	EM374,DH374,DT374,DF374	:ITEM 374
005402	000000	000000	000000	.WORD	EM375,DH375,DT375,DF375	:ITEM 375
005412	000000	000000	000000	.WORD	EM376,DH376,DT376,DF376	:ITEM 376
005422	000000	000000	000000	.WORD	EM377,DH377,DT377,DF377	:ITEM 377
005432	000000	000000	000000	.WORD	EM400,DH400,DT400,DF400	:ITEM 400
005442	065017	066570	070276	.WORD	EM401,DH401,DT401,DF401	:ITEM 401
005452	065041	066460	070276	.WORD	EM402,DH402,DT402,DF402	:ITEM 402
005462	065052	066644	070346	.WORD	EM403,DH403,DT403,DF403	:ITEM 403
005472	065202	067112	070346	.WORD	EM404,DH404,DT404,DF404	:ITEM 404
005502	065214	066570	070276	.WORD	EM405,DH405,DT405,DF405	:ITEM 405
005512	065230	066460	070276	.WORD	EM406,DH406,DT406,DF406	:ITEM 406
005522	065244	066644	070346	.WORD	EM407,DH407,DT407,DF407	:ITEM 407
005532	065314	067112	070346	.WORD	EM410,DH410,DT410,DF410	:ITEM 410
005542	065330	066570	070276	.WORD	EM411,DH411,DT411,DF411	:ITEM 411
005552	065354	066460	070276	.WORD	EM412,DH412,DT412,DF412	:ITEM 412
005562	065366	066644	070346	.WORD	EM413,DH413,DT413,DF413	:ITEM 413
005572	065436	067112	070346	.WORD	EM414,DH414,DT414,DF414	:ITEM 414
005602	065450	066570	070276	.WORD	EM415,DH415,DT415,DF415	:ITEM 415
005612	065475	066460	070276	.WORD	EM416,DH416,DT416,DF416	:ITEM 416
005622	065506	066644	070346	.WORD	EM417,DH417,DT417,DF417	:ITEM 417
005632	065552	067112	070346	.WORD	EM420,DH420,DT420,DF420	:ITEM 420
005642	065564	066570	070276	.WORD	EM421,DH421,DT421,DF421	:ITEM 421

005652	065611	066460	070276	.WORD	EM422,DH422,DT422,DF422	:ITEM 422
005662	065622	066644	070346	.WORD	EM423,DH423,DT423,DF423	:ITEM 423
005672	065634	067112	070346	.WORD	EM424,DH424,DT424,DF424	:ITEM 424
005702	065646	066570	070276	.WORD	EM425,DH425,DT425,DF425	:ITEM 425
005712	065672	066460	070276	.WORD	EM426,DH426,DT426,DF426	:ITEM 426
005722	065704	066644	070346	.WORD	EM427,DH427,DT427,DF427	:ITEM 427
005732	065756	067112	070346	.WORD	EM430,DH430,DT430,DF430	:ITEM 430
005742	065770	066644	070346	.WORD	EM431,DH431,DT431,DF431	:ITEM 431
005752	066016	066570	070276	.WORD	EM432,DH432,DT432,DF432	:ITEM 432
005762	066043	066460	070276	.WORD	EM433,DH433,DT433,DF433	:ITEM 433
005772	066054	066644	070346	.WORD	EM434,DH434,DT434,DF434	:ITEM 434
006002	066124	067112	070346	.WORD	EM435,DH435,DT435,DF435	:ITEM 435
006012	066136	066644	070346	.WORD	EM436,DH436,DT436,DF436	:ITEM 436
006022	066164	066570	070276	.WORD	EM437,DH437,DT437,DF437	:ITEM 437
006032	066206	066570	070276	.WORD	EM440,DH440,DT440,DF440	:ITEM 440
006042	066230	067502	071256	.WORD	EM441,DH441,DT441,DF441	:ITEM 441
006052	066230	067327	071274	.WORD	EM442,DH442,DT442,DF442	:ITEM 442
006062	066230	067327	071274	.WORD	EM443,DH443,DT443,DF443	:ITEM 443
006072	066355	066524	070276	.WORD	EM444,DH444,DT444,DF444	:ITEM 444

1671
1672
1673

.SBTTL ACT11 HOOKS

:HOOKS REQUIRED BY ACT11

\$SVPC=. ;SAVE PC

=46

\$ENDAD ;:1)SET LOC.46 TO ADDRESS OF \$ENDAD IN .\$EOP

-52

.WORD 0 ;:2)SET LOC.52 TO ZERO

-\$SVPC ;: RESTORE PC

.SBTTL APT PARAMETER BLOCK

:SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT

.\$X=. ;:SAVE CURRENT LOCATION

. 24 ;:SET POWER FAIL TO POINT TO START OF PROGRAM

200 ;:FOR APT START UP

-.44 ;:POINT TO APT INDIRECT ADDRESS PNTR.

\$APTHDR ;:POINT TO APT HEADER BLOCK

-. \$X ;:RESET LOCATION COUNTER

:SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC

:INTERFACE SPEC.

\$APTHD:

\$HIBTS: .WORD 0 ;:TWO HIGH BITS OF 18 BIT MAILBOX ADDR.

\$MBADR: .WORD \$MAIL ;:ADDRESS OF APT MAILBOX (BITS 0-15)

\$STMT: .WORD 10 ;:RUN TIME OF LONGEST TEST

\$PASTM: .WORD 40 ;:RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)

\$UNITM: .WORD 0 ;:ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT

.WORD \$ETEND-\$MAIL/2 ;:LENGTH MAILBOX-ETABLE(WORDS)

START:

.SBTTL INITIALIZE THE COMMON TAGS

:CLEAR THE COMMON TAGS (\$CMTAG) AREA

MOV # \$CMTAG,R6 ;:FIRST LOCATION TO BE CLEARED

CLR (R6)+ ;:CLEAR MEMORY LOCATION

000046 006102
000052 000046
000052 043422
000052 000052
000052 000000
1674 006102

000024 006102
000044 000024
000044 000200
000044 000044
000044 006102
000044 006102

006102
006102 000000
006104 001316
006106 000010
006110 000040
006112 000000
006114 000052

1675
1676
1677 006116

006116 012706 001100
006122 005026

```

006124 022706 001140      CMP      #SWR,R6 ;;DONE?
006130 001374              BNE      #-6      ;;LOOP BACK IF NO
006132 012706 001100      MOV      #STACK,SP ;;SETUP THE STACK POINTER
                                ;;INITIALIZE A FEW VECTORS
006136 012737 043502 000020      MOV      #SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
006144 012737 000340 000022      MOV      #340,@#IOTVEC+2 ;;LEVEL 7
006152 012737 044026 000030      MOV      #ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
006160 012737 000340 000032      MOV      #340,@#EMTVEC+2 ;;LEVEL 7
006166 012737 046210 000034      MOV      #TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
006174 012737 000340 000036      MOV      #340,@#TRAPVEC+2;LEVEL 7
006202 012737 046274 000024      MOV      #SPWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
006210 012737 000340 000026      MOV      #340,@#PWRVEC+2 ;;LEVEL 7
006216 013737 043236 043230      MOV      $ENDCT,$EOPCT ;;SETUP END-OF-PROGRAM COUNTER
006224 005037 001302              CLR      $TIMES      ;;INITIALIZE NUMBER OF ITERATIONS
006230 005037 001304              CLR      $ESCAPE     ;;CLEAR THE ESCAPE ON ERROR ADDRESS
006234 112737 000001 001115      MOV      #1,$ERMAX   ;;ALLOW ONE ERROR PER TEST
                                ;;INITIALIZE THE 'T-BIT' TRAP VECTOR. THEN LOAD LOCATION '$RTRN', IN
                                ;;THE 'END-OF-PASS' ($EOP) ROUTINE, WITH A 'RTI' OR 'RTT'.
006242 012737 043466 000014      MOV      #RTRN,@#TBITVEC ;;SET 'T' BIT VECTOR TO $RTRN
006250 012737 000340 000016      MOV      #340,@#TBITVEC+2 ;;LEVEL 7
006256 012737 042772 043466      MOV      #RTI,$RTRN    ;;SET $RTRN TO A RTI
006264 012737 006312 000010      MOV      #65,$@#RESVEC ;;TRY TO DO A RTT
006272 005046              CLR      -(SP)        ;;DUMMY PS
006274 012746 006302      MOV      #64$,-(SP)   ;;AND PC
006300 000006              RTT                ;;TRY THE RTT
006302 012737 000006 043466 64$:  MOV      #RTT,$RTRN   ;;RTT IS LEGAL--SET $RTRN TO A RTT
006310 000402              BR      66$          ;;RTT ILLEGAL--CLEAN OFF THE STACK
006312 062706 000010 65$:  ADD      #10,SP        ;;RTT ILLEGAL--CLEAN OFF THE STACK
006316 012737 000012 000010 66$:  MOV      #RESVEC+2,@#RESVEC ;;RESTORE TRAP CATCHER
006324 005037 043474              CLR      $TBIT       ;;CLEAR 'T' BIT SWITCH
006330 012737 006330 001106      MOV      #.,$LPADR    ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
006336 012737 006336 001110      MOV      #.,$LPERR    ;;SETUP THE ERROR LOOP ADDRESS
                                ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                ;;EQUAL TO A '-1', SETUP FOR A SOFTWARE SWITCH REGISTER.
006344 013746 000004      MOV      @#ERRVEC,-(SP) ;;SAVE ERROR VECTOR
006350 012737 006404 000004      MOV      #67$,@#ERRVEC ;;SET UP ERROR VECTOR
006356 012737 177570 001140      MOV      #DSWR,SWR    ;;SETUP FOR A HARDWARE SWICH REGISTER
006364 012737 177570 001142      MOV      #DDISP,DISPLAY ;;AND A HARDWARE DISPLAY REGISTER
006372 022777 177777 172540      CMP      #-1,@SWR     ;;TRY TO REFERENCE HARDWARE SWR
006400 001012              BNE      69$         ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                ;;AND THE HARDWARE SWR IS NOT -1
006402 000403              BR      68$         ;;BRANCH IF NO TIMEOUT
006404 012716 006412 67$:  MOV      #68$,(SP)   ;;SET UP FOR TRAP RETURN
006410 000002              RTI                ;;POINT TO SOFTWARE SWR
006412 012737 000176 001140 68$:  MOV      #SWREG,SWR   ;;POINT TO SOFTWARE SWR
006420 012737 000174 001142      MOV      #DISPREG,DISPLAY
006426 012637 000004 69$:  MOV      (SP)+,@#ERRVEC ;;RESTORE ERROR VECTOR
006432 005037 001324              CLR      $PASS       ;;CLEAR PASS COUNT
006436 132737 000200 001337      BITB    #APTSIZE,$ENVM ;;TEST USER SIZE UNDER APT
006444 001403              BEQ      70$        ;;YES,USE NON-APT SWITCH
006446 012737 001340 001140      MOV      #SSWREG,SWR  ;;NO,USE APT SWITCH REGISTER
006454
1678
                                .SBTTL TYPE PROGRAM NAME
                                ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
006454 005227 177777      INC      #-1          ;;FIRST TIME?
006460 001047              BNE      71$         ;;BRANCH IF NO
006462 022737 043422 000042      CMP      #SENDAD,@#42 ;;ACT-11?

```



```

006470 001443 BEQ 71$ ;;BRANCH IF YES
006472 104401 006540 TYPE ,72$ ;;TYPE ASCIZ STRING
.SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
006476 005737 000042 TST @#42 ;;ARE WE RUNNING UNDER XXDP/ACT?
006502 001012 BNE 73$ ;;BRANCH IF YES
006504 123727 001336 000001 CMPB $ENV,#1 ;;ARE WE RUNNING UNDER APT?
006512 001406 BEQ 73$ ;;BRANCH IF YES
006514 023727 001140 000176 CMP SWR,#SWREG ;;SOFTWARE SWITCH REG SELECTED?
006522 001005 BNE 74$ ;;BRANCH IF NO
006524 104405 GTSWR ;;GET SOFT-SWR SETTINGS
006526 000403 BR 74$
006530 112737 000001 001134 73$: MOVB #1,$AUTOB ;;SET AUTO-MODE INDICATOR
006536 000420 74$: BR 71$ ;;GET OVER THE ASCIZ
006600 ;;72$: .ASCIZ <CRLF>*CKFPCCO FP11F FLTG PNT PRT (*<CRLF>
;;71$:
1679
1680 006600 LOOP:
1681
1682
1683
1684
1690

```

```

1691          .SBTTL TEST # 1 - STF WITH ILLEGAL ACCUMULATOR TEST
              :*****
              :*TEST 1          STF WITH ILLEGAL ACCUMULATOR TEST
              :*
              :*THIS IS A TEST OF THE ST INSTRUCTION USING ILLEGAL ACCUMULATOR 7, MODE 0.
              :*
              :*****
1692          006600 000004          TST1:  SCOPE
1693          006602          0001:
1694          006602 104413          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
1695          006604 005000          CLR          RO          ;SET THE FPS.
1696          006606 170100          LDFPS        RO
1697          006610 012737 006646 000244          MOV          #000T,FPVECT ;SET UP FOR FP TRAPS.
1698          006616 012737 006624 001236          MOV          #1$, $TMP2
1699
1700          006624 174007          1$:          STF          AC0,AC7          ;THIS TEST INSTRUCTION SHOULD
1701
1702
1703          ;REPORT FAILURE OF USE OF ILLEGAL ACCUMULATOR 7 TO CAUSE AN FPP TRAP.
1704          006626          0002:
1705          006626 170200          STFPS        RO          ;GET FPS.
1706          006630 010037 001240          MOV          RO,$TMP3
1707          006634 170300          STST        RO          ;GET FEC.
1708          006636 010037 001242          MOV          RO,$TMP4
1709          006642 104001          3$:          ERROR        +1          ;STF WITH ILLEGAL ACCUMULATOR, MODE
1710
1711          006644 000434          BR          OODONE          ;0. DIDN'T TRAP. ST 765 TO ST 537.
1712
1713          ;TRAP TO 000T, HERE, WHEN THE EXPECTED ERROR OCCURS.
1714          006646 011600          000T:      MOV          (SP),RO          ;MAKE SURE THE ERROR OCCURRED
1715          006650 022700 006626          CMP          #0002,RO          ;AT THE CORRECT ADDRESS.
1716          006654 001402          BEQ          0003          ;BRANCH IF TRAP ADDRESS CORRECT.
1717          006656 000137 047240          JMP          FPSPUR ;IF INCORRECT GO REPORT SPURIOUS
1718
1719
1720          006662 170204          0003:      STFPS        R4          ;GET FPS.
1721          006664 170305          STST        R5          ;GET FEC.
1722          006666 010437 001240          MOV          R4,$TMP3          ;SAVE DATA IN CASE OF ERROR.
1723          006672 010537 001242          MOV          R5,$TMP4
1724          006676 012702 100000          MOV          #100000,R2          ;EXPECTED FPS
1725          006702 012703 000002          MOV          #2,R3          ;EXPECTED FEC
1726          006706 010237 001244          MOV          R2,$TMP5
1727          006712 010337 001246          MOV          R3,$TMP6
1728          006716 022626          CMP          (SP)+,(SP)+          ;RESET THE STACK.
1729
1730          006720 020204          CMP          R2,R4          ;WAS FPS CORRECT?
1731          006722 001402          BEQ          0004          ;BRANCH IF YES.
1732
1733          006724 104002          1$:          ERROR        +2          ;OTHERWISE REPORT FPS INCORRECTLY
1734          006726 000403          BR          OODONE          ;SET AFTER USE OF ILLEGAL ACC.
1735
1736          006730 020305          0004:      CMP          R3,R5          ;WAS THE FEC CORRECT?
1737          006732 001401          BEQ          OODONE          ;BRANCH IF CORRECT.
1738
1739          006734 104003          1$:          ERROR        +3          ;OTHERWISE REPORT INCORRECT FEC
              ;AFTER USE OF ILLEGAL ACC.
    
```

1740
1741 006736
006736 104412

000DONE:
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?).

1742
1743
1744
1750

```

1751          .SBTTL TEST # 2 - FDST MODE 1, FLOATING MODE, TEST
              :*****
              :*TEST 2          FDST MODE 1, FLOATING MODE, TEST
              :*
              :*THIS IS A TEST OF THE STF INSTRUCTION USING FDST MODE 1.
              :*
              :*****
006740 000004 TST2:  SCOPE
1752
1753 006742 PPP1:
006742 104413          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
1754
1755 006744 012700 177777          MOV          #-1,R0          ;SET UP A BACKGROUND PATTERN IN THE
1756 006750 012701 007100          MOV          #PPPBF0,R1       ;INPUT BUFFER.
1757 006754 012702 000014          MOV          #14,R2
1758 006760 010021          PPP2:  MOV          R0,(R1)+
1759 006762 077202          SCB          R2,PPP2
1760
1761 006764 012700 000200          MOV          #200,R0         ;SET FD MODE.
1762 006770 170100          LDFPS        R0
1763 006772 012700 007130          MOV          #PPPTP1,R0      ;PUT TEST DATA INTO ACO.
1764 006776 172410          LDD          (R0),ACO
1765
1766 007000 012700 007114          MOV          #PPPBF1,R0      ;FDST ADDRESS.
1767 007004 005002          CLR          R2             ;CLEAR THE FPS.
1768 007006 170102          LDFPS        R2
1769 007010 012737 007022 001236          MOV          #PPP3,$TMP2
1770 007016 010037 001240          MOV          R0,$TMP3
1771
1772 007022 174010          PPP3:  STF          ACO,(R0)    ;TEST INSTRUCTION.
1773
1774 007024 022700 007114          CMP          #PPPBF1,R0      ;WAS R0 MODIFIED DURING EXECUTION?
1775 007030 001404          BEQ          PPP4           ;BRANCH IF R0 NOT MODIFIED, CORRECT.
1776
1777 007032 010037 001242          MOV          R0,$TMP4        ;OTHERWISE REPORT ERROR, R0 MODIFIED.
1778 007036 104004          1$:  ERROR        +4
1779 007040 000456          BR          PPPDONE         ;GO TO NEXT TEST.
1780
1781 007042 012700 007114          PPP4:  MOV          #PPPBF1,R0 ;CHECK THE DATA IN THE OUTPUT BUFFER.
1782 007046 012701 007130          MOV          #PPPTP1,R1
1783 007052 022021          CMP          (R0)+,(R1)+
1784 007054 001031          BNE          PPP10         ;BRANCH IF INCORRECT.
1785 007056 022011          CMP          (R0)+,(R1)
1786 007060 001027          BNE          PPP10         ;BRANCH IF INCORRECT.
1787 007062 022720 177777          CMP          #-1,(R0)+      ;WAS FLOATING MODE USED?
1788 007066 001034          BNE          PPP15         ;BRANCH IF NOT.
1789 007070 022710 177777          CMP          #-1,(R0)
1790 007074 001031          BNE          PPP15
1791 007076 000437          BR          PPPDONE ;GO TO NEXT TEST.
1792
1793 007100 177777 177777 177777 PPPBF0: .WORD  -1,-1,-1,-1,-1,-1
1794
1795 007114 177777 177777 177777 PPPBF1: .WORD  -1,-1,-1,-1,-1,-1
1796
1797 007130 123456 023456          FPPTP1: .WORD  123456,23456
1798 007134 034567 045671          .WORD  34567,45671
1799
    
```



```
1800 ;REPORT DATA IN OUT PUT BUFFER INCORRECT.
1801 007140 012737 007130 001242 PPP10: MOV #PPPTP1,$TMP4
1802 007146 012737 007114 001240 MOV #PPPBF1,$TMP3
1803 007154 104005 1$: ERROR +5 ;BAD DATA.
1804 007156 000407 BR PPPDONE
1805
1806 ;REPORT FLOATING MODE NOT USED, BUT FD FAILED.
1807 007160 012737 007130 001242 PPP15: MOV #PPPTP1,$TMP4
1808 007166 012737 007114 001240 MOV #PPPBF1,$TMP3
1809 007174 104006 1$: ERROR +6 ;ST 707 TO 245 INTO 244 (BUT FD).
1810
1811 007176 PPPDONE: ;GO INITIALIZE THE FPS AND STACK; AND
007176 104412 RSETUP ;SEE IF THE USER HAS EXPRESSED
;THE DESIRE TO CHANGE THE SOFTWARE
;VIRTUAL CONSOLE SWITCH REGISTER (HAS
;THE USER TYPED CONTROL G?).

1812
1813
1814
1820
```

```

1821          .SBTTL TEST # 3 - FDST MODE 2 TEST
              :*****
              :*TEST 3          FDST MODE 2 TEST
              :*
              :*THIS IS A TEST OF BOTH STF AND STD WITH FDST MODE 2.
              :*
              :*****
007200 000004 TST3:  SCOPE
1822
1823          :FIRST TEST STF.
1824 007202 007202 104413 QQQ1:  LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
1825
1826 007204 012700 177777 MOV      #-1,R0      ;SET UP THE OUTPUT BUFFER.
1827 007210 012701 007342 MOV      #QQQBF0,R1
1828 007214 012702 000014 MOV      #14,R2
1829 007220 010021 QQQ2:  MCV      R0,(R1)+
1830 007222 077202 SOB      R2,QQQ2
1831
1832 007224 012700 000200 MOV      #200,R0      ;SET FD MODE.
1833 007230 170100 LDFPS   R0
1834 007232 012700 007372 MOV      #QQQTP1,R0   ;SETUP ACO.
1835 007236 172410 LDD     (R0),ACO
1836
1837 007240 012700 007356 MOV      #QQQBF1,R0   ;FDST ADDRESS.
1838 007244 005002 CLR     R2
1839 007246 170102 LDFPS   R2          ;SET FPS.
1840 007250 012737 007256 001236 MOV      #QQQ3,$TMP2
1841
1842 007256 174020 QQQ3:  STF     ACO,(R0)+ ;TEST INSTRUCTION.
1843
1844 007260 022700 007362 CMP     #QQQBF1+4,R0 ;WAS R0 INCREMENTED BY 4 PROPERLY?
1845
1846 007264 001407 BEQ     QQQ4          ;BRANCH IF R0 CORRECT.
1847 007266 010037 001242 MOV     R0,$TMP4     ;REPORT R0 INCORRECT AFTER FDST MODE 2.
1848 007272 012737 007362 001240 MOV     #QQQBF1+4,$TMP3
1849 007300 104007 1$:  ERROR +7 ;BAD CONSTANT USED OR DIDN'T GO 527 TO 642
1850 007302 000526 BR     QQQD ONE
1851 007304 012700 007356 QQQ4:  MOV     #QQQBF1,R0 ;WAS THE OUTPUT DATA CORRECT?
1852 007310 012701 007372 MOV     #QQQTP1,R1
1853 007314 022021 CMP     (R0)+,(R1)+
1854 007316 001031 BNE    QQQ10        ;BRANCH IF INCORRECT.
1855 007320 022021 CMP     (R0)+,(R1)+
1856 007322 001027 BNE    QQQ10        ;BRANCH IF INCORRECT.
1857 007324 022027 177777 CMP     (R0)+,#-1    ;SEE IF ANY OTHER DATA BUFFER WORDS WERE MODIFIED.
1858 007330 001024 BNE    QQQ10        ;BRANCH IF INCORRECT.
1859 007332 022027 177777 CMP     (R0)+,#-1
1860 007336 001021 BNE    QQQ10        ;BRANCH IF INCORRECT.
1861 007340 000430 BR     QQQ20
1862 007342 177777 177777 177777 QQQBF0: .WORD -1,-1,-1,-1,-1,-1
1863 007356 177777 177777 177777 QQQBF1: .WORD -1,-1,-1,-1,-1,-1
1864 007372 076543 QQQTP1: 76543
1865 007374 065432 65432
1866 007376 054321 54321
1867 007400 043210 43210
1868
1869 007402 012737 007372 001240 :REPORT OUTPUT DATA INCORRECT:
              QQQ10: MOV     #QQQTP1,$TMP3

```

```

1870 007410 012737 007356 001242      MOV      #QQQBF1,$TMP4
1871 007416 104010      1$:      ERROR      +10      ;BAD DATA
1872 007420 000457      BR        QQQDONE
1873
1874      ;NOW TEST STD MODE 2.
1875
1876 007422      QQQ20:
      007422 104413      LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.
1877 007424 012700 007342      MOV      #QQQBF0,R0      ;SET UP DEFAULT INPUT DATA BUFFER.
1878 007430 010001      MOV      R0,R1
1879 007432 012702 000014      MOV      #14,R2
1880 007436 010021      QQQ22:  MOV      R0,(R1)+
1881 007440 077202      SOB      R2,QQQ22
1882 007442 012700 000200      MOV      #200,R0      ;ENTER FLOATING DOUBLE MODE.
1883 007446 170100      LDFPS    R0
1884 007450 012700 007372      MOV      #QQQTP1,R0      ;LOAD ACO.
1885 007454 172410      LDD      (R0),ACO
1886 007456 012700 007356      MOV      #QQQBF1,R0      ;SET DESTINATION ADDRESS.
1887 007462 012737 007470 001236      MOV      #QQQ23,$TMP2
1888 007470 174020      QQQ23:  STD      ACO,(R0)+      ;TEST INSTRUCTION.
1889 007472 022700 007366      CMP      #QQQBF1+10,R0      ;WAS R0 INCREMENTED BY 10 CORRECTLY?
1890 007476 001407      BEQ      QQQ24      ;BRANCH IF CORRECT.
1891 007500 010037 001242      MOV      R0,$TMP4      ;REPORT R0 INCORRECTLY INCREMENTED.
1892 007504 012737 007366 001240      MOV      #QQQBF1+10,$TMP3
1893 007512 104011      1$:      ERROR      +11      ;DC NOT INCREM BY 10 BAD CONSTANT
1894 007514 000421      BR        QQQDONE
1895 007516 012700 007356      QQQ24:  MCV      #QQQBF1,R0      ;DID THE DATA REACH THE OUTPUT BUFFER CORRECTLY?
1896 007522 012701 007372      MOV      #QQQTP1,R1
1897 007526 012702 000004      MOV      #4,R2
1898 007532 022021      1$:      CMP      (R0)+,(R1)+
1899 007534 001002      BNE      QQQ25      ;BRANCH IF INCORRECT.
1900 007536 017203      SOB      R2,1$
1901 007540 000407      BR        QQQDONE
1902      ;REPORT DATA INCORRECT.
1903 007542 012737 007372 001240      QQQ25:  MOV      #QQQTP1,$TMP3
1904 007550 012737 007356 001242      MOV      #QQQBF1,$TMP4
1905 007556 104012      1$:      ERROR      +12      ;BAD DATA
1906 007560      QQQDONE:
      007560 104412      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?).
    
```

1907

1913

```
.SBTTL TEST # 4 - FDST MODE 2, WITH GR7, TEST
*****
*TEST 4      FDST MODE 2, WITH GR7, TEST
*
*THIS IS A TEST OF STF WITH GR7 MODE 2 OR IMMEDIATE MODE.
*
*****
TST4:  SCOPE
```

```
1914 007562 000004
1915 007564
      007564 104413
1916 007566 012700 007644
1917 007572 012701 007712
1918 007576 012702 000004
1919 007602 012021
1920 007604 077202
1921 007606 012700 000200
1922 007612 170100
1923 007614 012700 007722
1924 007620 172410
1925 007622 012737 007742 000004
1926 007630 012737 007642 001236
1927 007636 005001
1928 007640 005004
1929
1930
1931
1932
1933
1934
1935 007642 174027
1936 007644 005201
1937 007646 005201
1938 007650 005201
1939 007652 005201
1940 007654 012700 007732
1941 007660 012702 007644
1942 007664 012703 000004
1943 007670 022022
1944 007672 001051
1945 007674 077303
1946 007676 005704
1947 007700 001056
1948 007702 022701 000003
1949 007706 001053
1950 007710 000474
1951
1952 007712 005201
1953 007714 005201
1954 007716 005201
1955 007720 005201
1956
1957 007722 005204
1958 007724 005204
1959 007726 005204
1960 007730 005204
1961
```

```
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$:     MOV (R0)+,(R1)+
        SOB R2,1$
        MCV #200,R0        ;ENTER FLOATING DOUBLE MODE.
        LDFPS RC
        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.
```

```

1962 007732 005204 RRREXP: INC R4
1963 007734 005201 INC R1
1964 007736 005201 INC R1
1965 007740 005201 INC R1
1966 ;IF A FAILURE IN THE FDST FLOWS RESULTS IN AN ODD ADDRESS TRAP THROUGH
1967 ;4 TO HERE:
1968 007742 011602 RRR10: MOV (SP),R2 ;SEE IF THE TRAP WAS BECAUSE OF AN ODD ADDRESS.
1969 007744 032702 000001 BIT #1,R2 ;BRANCH IF YES.
1970 007750 001005 BNE RRR11 ;SEE IF THE TRAP OCCURRED AT THE TEST INSTRUCTION.
1971 007752 020227 007646 CMP R2,#RRR3+2 ;BRANCH IF YES.
1972 007756 001412 BEQ RRR12 ;OTHERWISE REPORT A SPURIOUS TRAP THROUGH VECTOR 4.
1973 007760 000137 047302 JMP CPSPUR ;REPORT A FAILURE IN THE FDST FLOWS RESULTED IN AN ODD ADDRESS TRAP.
1974 ;REPORT A FAILURE IN THE FDST FLOWS RESULTED IN AN ODD ADDRESS TRAP.
1975 007764 010237 001236 RRR11: MOV R2,$TMP2
1976 007770 012737 007646 001240 MOV #RRR3+2,$TMP3
1977 007776 022626 CMP (SP)+,(SP)+
1978 010000 104013 1$: ERROR +13 ;BAD CONSTANT #2 + PC ODD ADDR.
1979 010002 000437 BR RRRDONE
1980 010004 010237 001236 RRR12: MOV R2,$TMP2
1981 010010 022626 CMP (SP)+,(SP)+
1982 010012 104014 1$: ERROR +14 ;ODD ADDRESS TRAP
1983 010014 000432 BR RRRDONE ;WRONG MODE USED.
1984
1985 ;REPORT DATA INCORRECT:
1986 010016 012737 007644 001240 RRR25: MOV #RRR3,$TMP3
1987 010024 012737 007732 001242 MOV #RRREXP,$TMP4
1988 010032 104015 1$: ERROR +15 ;BAD DATA BUT GR7 FAIL
1989 010034 000422 BR RRRDONE
1990
1991 ;REPORT PC INCORRECT MODIFIED DURING THE EXECUTION OF FDST IMMEDIATE
1992 ;MODE. THE PC SHOULD HAVE BEEN INCREMENTED BY 2 BUT IT WASN'T.
1993 ;USE R1 AND R4 TO COMPUTE THE ACTUAL ACTION THAT WAS TAKEN ON THE PC.
1994 010036 012737 007646 001240 RRR15: MOV #RRR3+2,$TMP3
1995 010044 005704 TST R4 ;IS R4 CLEAR.
1996 010046 001404 BEQ 1$
1997 010050 012737 007644 001242 MOV #RRR3,$TMP4
1998 010056 000410 BR 2$
1999 010060 012702 007646 1$: MOV #RRR3+2,R2
2000 010064 062701 177775 ADD #-3,R1
2001 010070 006301 ASL R1
2002 010072 160102 SUB R1,R2
2003 010074 010237 001242 MOV R2,$TMP4
2004 010100 2$:
2005 010100 104016 3$: ERROR +16 ;BAD CONSTANT PC+
2006 010102 RRRDONE:
010102 104412 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?).
2007
    
```


2013

```
.SBTTL TEST # 5 - FDST MODE 4 TEST
:*****
:*TEST 5      FDST MODE 4 TEST
:*
:*THIS IS A TEST OF STD WITH FDST MODE 4.
:*
:*****
TST5:  SCOPE
```

```
010104 000004
2014
2015 010106
010106 104413
2016 010110 012700 177777
2017 010114 012701 010244
2018 010120 012702 000010
2019 010124 010021
2020 010126 077202
2021 010130 012700 000200
2022 010134 170100
2023 010136 012700 010264
2024 010142 172410
2025 010144 012737 010304 000004
2026 010152 012737 010164 001236
2027 010160 012700 010254
2028
2029 010164 174040
2030 010166 005201
2031 010170 020027 010244
2032 010174 001060
2033 010176 012700 010244
2034 010202 012701 010264
2035 010206 012702 000004
2036 010212 022021
2037 010214 001057
2038 010216 077203
2039 010220 012700 177777
2040 010224 012701 010254
2041 010230 012702 000004
2042 010234 020021
2043 010236 001056
2044 010240 077203
2045 010242 000463
2046
2047
2048 010244 177777
2049 010246 177777
2050 010250 177777
2051 010252 177777
2052 010254 177777
2053 010256 177777
2054 010260 177777
2055 010262 177777
2056
2057
2058 010264 147250
2059 010266 036147
2060 010270 025036
2061 010272 147250
```

```
SSS1:
      LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
      MOV            # -1,R0      ;SET UP THE OUTPUT BUFFER.
      MOV            #SSSBF0,R1
      MOV            #10,R2
1$:   MOV            R0,(R1)+
      SOB            R2,1$
      MCV            #200,R0      ;ENTER FLOATING DOUBLE MODE.
      LDFPS          R0
      MOV            #SSSTP1,R0   ;SET UP ACO.
      LDD            (R0),ACO
      MOV            #SSS10,ERRVECT ;SET UP FOR A TRAP TO 4.
      MOV            #SSS2,$TMP2
      MOV            #SSSA1,R0    ;SET UP THE DESTINATION ADDRESS.
SSS2:  STD            ACO,-(R0)   ;TEST INSTRUCTION.
      INC            R1
      CMP            R0,#SSSBF0   ;SEE IF R0 WAS DECREMENTED PROPERLY.
      BNE            SSS15       ;BRANCH IF R0 IS INCORRECT.
      MOV            #SSSBF0,R0   ;WAS THE OUTPUT DATA CORRECT?
      MOV            #SSSTP1,R1
1$:   MOV            #4,R2
      CMP            (R0)+,(R1)+
      BNE            SSS20       ;BRANCH IF INCORRECT.
      SOB            R2,1$
      MOV            # -1,R0      ;IS THE REST OF THE OUTPUT BUFFER CORRECT, -1?
      MOV            #SSSA1,R1
      MOV            #4,R2
2$:   CMP            R0,(R1)+
      BNE            SSS25       ;BRANCH IF INCORRECT.
      SOB            R2,2$
      BR             SSSDONE

;THIS IS THE OUTPUT DATA BUFFER.
SSSBF0: -1
      -1
      -1
      -1
SSSA1: -1
      -1
      -1
      -1

;THIS IS THE TEST DATA LOADED INTO ACO:
SSSTP1: 147250
      36147
      25036
      147250
```

```

2062 010274 177777      SSSTP2: -1
2063 010276 177777      -1
2064 010300 177777      -1
2065 010302 177777      -1
2066
2067
2068 010304 011600      ;IF AN ODD ADDRESS TRAP OCCURS COME HERE:
2069 010306 020027 010166 SSS10: MOV (SP),R0 ;SEE IF THE TRAP ACCURRED ON THE TEST INSTRUCTION.
2070 010312 001405      CMP R0,#SSS2+2
2071 010314 020027 010170 BEQ SSS11 ;BRANCH IF YES.
2072 010320 001402      CMP R0,#SSS2+4
2073 010322 000137 047302 BEQ SSS11 ;BRANCH IF YES.
2074 ;REPORT FAILURE IN FDST FLOWS RESULTED IN AN ODD ADDRESS.
2075 010326 010037 001236 SSS11: MOV R0,$TMP2 ;OTHERWISE GO REPORT A SPURIOUS TRAP THROUGH 4.
2076 010332 104017      2$: ERROR +17 ;FDST FORK X ODD AD RES.
2077 010334 000426      BR SSSDONE
2078
2079 ;REPORT R0 INCORRECTLY DECREMENTED.
2080 010336 010037 001242 SSS15: MOV R0,$TMP4
2081 010342 012737 010244 001240 MOV #SSSBF0,$TMP3
2082 010350 104020      1$: ERROR +20 ;R0 NOT DECRE PROP
2083 010352 000417      BR SSSDONE
2084
2085 ;REPORT OUTPUT DATA INCORRECT:
2086 010354 012737 010244 001240 SSS20: MOV #SSSBF0,$TMP3
2087 010362 012737 010264 001242 MOV #SSSTP1,$TMP4
2088 010370 104021      1$: ERROR +21 ;BAD DATA
2089 010372 000407      BR SSSDONE
2090 010374 012737 010254 001242 SSS25: MOV #SSSA1,$TMP4
2091 010402 012737 010274 001240 MOV #SSSTP2,$TMP3
2092 010410 104022      1$: ERROR +22 ;DATA BAD OUTSIDE TARGET AREA
2093 010412 010412      SSSDONE: 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?).
2094

```

2100
2101 010414 000004
2102 010416 104413
2103 010420 012701 010536
2104 010424 012700 177777
2105 010430 012702 000012
2106 010434 010021
2107 010436 077202
2108 010440 012737 010536 010552
2109 010446 012700 000200
2110 010452 170100
2111 010454 012700 010562
2112 010460 172410
2113 010462 012737 010572 000004
2114 010470 013737 010502 001236
2115 010476 012700 010552
2116
2117 010502 174030
2118
2119 010504 020027 010554
2120 010510 001046
2121 010512 012701 010536
2122 010516 012702 010562
2123 010522 012703 000004
2124 010526 022122
2125 010530 001045
2126 010532 077303
2127 010534 000452
2128
2129
2130 010536 177777
2131 010540 177777
2132 010542 177777
2133 010544 177777
2134 010546 177777
2135 010550 177777
2136 010552 010536
2137 010554 177777
2138 010556 177777
2139 010560 177777
2140 010562 101213
2141 010564 141516
2142 010566 071727
2143 010570 037475
2144
2145
2146 010572 011602
2147 010574 020227 010504
2148 010600 001405

```
.SBTTL TEST # 6 - FDST MODE 3 TEST
:*****
:*TEST 6      FDST MODE 3 TEST
:*
:*THIS IS A TEST OF FDST MODE 3 USING STD.
:*
:*****
TST6:  SCOPE

TTT1:
      LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
      MOV    #TTTBFO,R1 ;SET UP THE OUTPUT DATA BUFFER.
      MOV    #-1,R0
      MOV    #12,R2
1$:   MOV    R0,(R1)+
      SOB   R2,1$
      MCV   #TTTBFO,TTTA2
      MOV   #200,R0      ;ENTER DOUBLE FLOATING MODE.
      LDFPS R0
      MOV   #TTTTP1,R0   ;SET UP ACO.
      LDD   (R0),ACO
      MOV   #TTT10,ERRVECT ;SET UP FOR TRAPS TO 4.
      MOV   TTT2,$TMP2
      MOV   #TTTA2,R0    ;SET UP THE DESTINATION ADDRESS.

TTT2:  STD    ACO,@(R0)+ ;TEST INSTRUCTION.

      CMP   R0,#TTTA2+2 ;SEE IF R0 WAS INCREMENTED CORRECTLY.
      BNE  TTT15        ;BRANCH IF INCORRECT.
      MOV   #TTTBFO,R1 ;CHECK THE OUTPUT DATA BUFFER.
      MOV   #TTTTP1,R2
      MOV   #4,R3
*TTT3: CMP   (R1)+,(R2)+
      BNE  TTT20        ;BRANCH IF NOT CORRECT.
      SOB  R3,TTT3
      BR   TTTDONE

;THIS IS THE OUTPUT DATA BUFFER:
TTTBFO: -1
        -1
        -1
        -1
        -1
TTTA1:  -1
TTTA2:  TTTBFO
TTTA3:  -1
        -1
        -1

TTTTP1: 101213
        141516
        071727
        037475

;TRAP THROUGH VECTOR 4 TO HERE.
TTT10: MOV   (SP),R2
      CMP   R2,#TTT2+2 ;SEE IF THE TRAP ADDRESS IS THAT OF THE TEST INSTRUCTION.
      BEQ  TTT11        ;BRANCH IF YES.
```

```
2149 010602 020227 010506          CMP      R2,#TTT2+4
2150 010606 001402          BEQ      TTT11          ;BRANCH IF YES.
2151 010610 000137 047302          JMP      CPSPUR ;OTHERWISE GO REPORT A SPURIOUS TRAP TO 4.
2152
2153          ;REPORT A FAILURE IN THE FDST FLOWS RESULTED IN AN ODD ADDRESS TRAP.
2154 010614 010237 001236 TTT11:  MOV      R2,$TMP2
2155 010620 022626          CMP      (SP)+,(SP)+
2156 010622 104023 1$:      ERROR  +23          ;BET FDST X ODD ADR
2157 010624 000416          BR       TTTDONE
2158
2159          ;REPORT RO INCORRECT:
2160 010626 010037 001242 TTT15:  MOV      R0,$TMP4
2161 010632 012737 010554 001240  MOV      #TTTA2+2,$TMP3
2162 010640 104024 1$:      ERROR  +24          ;RO NOT INCREMENT PROPERLY
2163 010642 000407          BR       TTTDONE
2164
2165          ;REPORT INCORRECT OUTPUT DATA:
2166 010644 012737 010536 001240 TTT20:  MOV      #TTTBFO,$TMP3
2167 010652 012737 010562 001242  MOV      #TTTTP1,$TMP4
2168 010660 104025 1$:      ERROR  +25          ;BAD DATA
2169 010662          TTTDONE:
      010662 104412          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?).
```

2170

2176

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

2177 010664 000004
2178 010666 104413
2179 010670 012701 011006
2180 010674 012700 177777
2181 010700 012702 000012
2182 010704 010021
2183 010706 077202
2184 010710 012737 011006 011020
2185 010716 012700 000200
2186 010722 170100
2187 010724 012700 011032
2188 010730 172410
2189 010732 012737 011042 000004
2190 010740 C13737 010752 001236
2191 010746 012700 011022
2192 010752 174050
2193 010754 020027 011020
2194 010760 001046
2195 010762 012701 011006
2196 010766 012702 011032
2197 010772 012703 000004
2198 010776 022122
2199 011000 001045
2200 011002 077303
2201 011004 000452
2202
2203
2204 011006 177777
2205 011010 177777
2206 011012 177777
2207 011014 177777
2208 011016 177777
2209 011020 011006
2210 011022 177777
2211 011024 177777
2212 011026 177777
2213 011030 177777
2214 011032 020212
2215 011034 023242
2216 011036 020273
2217 011040 031323
2218
2219
2220 011042 011602
2221 011044 020227 010754
2222 011050 001405
2223 011052 020227 010756
2224 011056 001402

```
UUU1: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.  
MOV #UUUBF0,R1 ;SET UP THE OUTPUT DATA BUFFER.  
MOV #-1,R0  
MOV #12,R2  
1$: MOV R0,(R1)+  
SOB R2,1$  
MCV #UUUBF0,UUA1  
MOV #200,R0 ;ENTER DOUBLE FLOATING MODE.  
LDFPS R0  
MOV #UUUTP1,R0 ;SET UP ACO.  
LDD (R0),ACO  
MOV #UUU10,ERRVECT ;GET READY FOR ANY TRAPS TO 4.  
MOV UUU2,$TMP2  
UUU2: MOV #UUA2,R0 ;SET UP THE DESTINATION ADDRESS.  
STD ACO,@-(R0) ;TEST INSTRUCTION.  
CMP R0,#UUA2-2 ;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.
```



```
2225 011060 000137 047302      JMP      CPSPUR ;OTHERWISE REPORT A SPURIOUS TRAP TO 4.
2226      ;REPORT FAILURE OF FDST RESULTED IN AN ODD ADDRESS TRAP TO 4.
2227 011064 010237 001236      UUU11:  MOV      R2,$TMP2
2228 011070 022626              CMP      (SP)+,(SP)+
2229 011072 104026              1$:      ERROR   +26 ;BET FDST X ODD ADR
2230 011074 000416              BR       UUUDONE
2231
2232      ;REPORT R0 INCORRECT.
2233 011076 010037 001242      UJU15:  MOV      R0,$TMP4
2234 011102 012737 011024 001240  MOV      #UUUA2+2,$TMP3
2235 011110 104027              1$:      ERROR   +27 ;R0 NOT INCREMENT PROPERLY
2236 011112 000407              BR       UUUDONE
2237
2238      ;REPORT BAD DATA.
2239 011114 012737 011006 001242  UUU20:  MOV      #UUUBF0,$TMP4
2240 011122 012737 011032 001240  MOV      #UUUTP1,$TMP3
2241 011130 104030              1$:      ERROR   +30 ;BAD DATA
2242 011132              UUUDONE:
          011132 104412              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?).
```

2243

2249

2250 011134 000004
 2251 011136
 2252 011136 104413
 2253 011140 012700 000200
 2254 011144 170100
 2255 011146 012701 011256
 2256 011152 012700 177777
 2257 011156 012702 000004
 2258 011162 010021
 2259 011164 077202
 2260 011166 012767 011276 166610
 2261 011174 012700 011266
 2262 011200 172410
 2263 011202 012767 011220 170026
 2264 011210 012700 003355
 2265 011214 012701 000001
 2266 011220 174060 005701
 2267 011224 020027 003355
 2268 011230 001040
 2269 011232 012702 011256
 2270 011236 012703 011266
 2271 011242 012704 000004
 2272 011246 022223
 2273 011250 001037
 2274 011252 077403
 2275 011254 000444
 2276 011256 177777
 2277 011260 177777
 2278 011262 177777
 2279 011264 177777
 2280 011266 030313
 2281 011270 023334
 2282 011272 035363
 2283 011274 074041
 2284
 2285
 2286 011276 011602
 2287 011300 020227 011222
 2288 011304 001405
 2289 011306 020227 011224
 2290 011312 001402
 2291 011314 000167 035720
 2292
 2293 011320 010267 167712
 2294 011324 022626
 2295 011326 104031
 2296 011330 000416
 2297

```

.SBTTL TEST # 10 - FDST MODE 6, INDEX MODE, TEST
*****
*TEST 10      FDST MODE 6, INDEX MODE, TEST
*
*THIS IS A TEST OF FDST MODE 6, INDEX MODE, USING STD.
*
*****
TST10:  SCOPE
        .DSABL  AMA          ;DISABLE MODE 6 TO MODE 3 CONVERSIONS
VVV1:   LPERR              ;SET UP THE LOOP ON ERROR ADDRESS.
        MOV      #200,R0     ;ENTER DOUBLE FLOATING MODE.
        LDFPS   R0
        MOV      #VVVBF0,R1  ;SET UP THE OUT PUT DATA BUFFER.
        MOV      #-1,R0
        MOV      #4,R2
1$:     MCV      R0,(R1)+
        SOB     R2,1$
        MOV      #VVV10,ERRVECT ;SET UP VECTOR 4 INCASE OF ERROR.
        MOV      #VVVTP1,R0   ;SET UP ACO.
        LDD     (R0),ACO
        MOV      #VVV2,$TMP2
        MOV      #VVVBF0-5701,R0 ;SET UP THE DESTINATION ADDRESS.
        MOV      #1,R1
VVV2:   STD      ACO,5701(R0) ;TEST INSTRUCTION.
        CMP     R0,#VVVBF0-5701 ;SEE IF R0 WAS MODIFIED.
        BNE     VVV15        ;BRANCH IF INCORRECT.
        MOV     #VVVBF0,R2    ;WAS THE OUTPUT DATA CORRECT.
        MOV     #VVVTP1,R3
        MOV     #4,R4
1$:     CMP     (R2)+,(R3)+
        BNE     VVV20        ;BRANCH IF INCORRECT DATA.
        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     FPSPUR     ;OTHERWISE GO REPORT SPURIOUS TRAP TO 4.
;REPORT FAILURE OF FDST RESULTED IN AN ODD ADDRESS TRAP TO 4.
VVV11:  MOV     R2,$TMP2
        CMP     (SP)+,(SP)+
1$:     ERROR   +31        ;FDST FORK X ODD ADD
        BR     VVVDONE
    
```

```
2298 ;REPORT RO MODIFIED.
2299 011332 010067 167704 VVV15: MOV RO,$TMP4
2300 011336 012767 003355 167674 MOV #VVVBFO-5701,$TMP3
2301 011344 104032 1$: ERROR +32 ;RO MODIFIED.
2302 011346 000407 BR VVVDONE
2303
2304 ;REPORT INCORRECT DATA.
2305 011350 012767 011256 167662 VVV20: MOV #VVVBFO,$TMP3
2306 011356 012767 011266 167656 MOV #VVVTF1,$TMP4
2307 011364 104033 1$: ERROR +33 ;BAD DATA
2308 011366 VVVDONE: RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
011366 104412 ;SEE IF THE USER HAS EXPRESSED
;THE DESIRE TO CHANGE THE SOFTWARE
;VIRTUAL CONSOLE SWITCH REGISTER (HAS
;THE USER TYPED CONTROL G?).
;REENABLE MODE 6 TO MODE 3 CONVERSIONS
2309 .ENABL AMA
```

2315
 2316 011370 000004
 2317 011372 104413
 2318 011374 012700 000200
 2319 011400 170100
 2320 011402 012701 011520
 2321 011406 012700 177777
 2322 011412 012702 000004
 2323 011416 010021
 2324 011420 077202
 2325 011422 012737 011550 000004
 2326 011430 012700 011530
 2327 011434 172410
 2328 011436 012737 011462 001236
 2329 011444 012700 003637
 2330 011450 012701 000001
 2331 011454 012737 011520 011540
 2332 011462 174070 005701
 2333 011466 020027 003637
 2334 011472 001044
 2335 011474 012702 011520
 2336 011500 012703 011530
 2337 011504 012704 000004
 2338 011510 022223
 2339 011512 001043
 2340 011514 077403
 2341 011516 000450
 2342 011520 177777
 2343 011522 177777
 2344 011524 177777
 2345 011526 177777
 2346 011530 041424
 2347 011532 034445
 2348 011534 046475
 2349 011536 051525
 2350 011540 177777
 2351 011542 177777
 2352 011544 177777
 2353 011546 177777
 2354
 2355
 2356 011550 011602
 2357 011552 020227 011464
 2358 011556 001405
 2359 011560 020227 011466
 2360 011564 001402
 2361 011566 000137 047240
 2362
 2363 011572 010237 001236

```

.SBTTL TEST # 11 - FDST MODE 7, INDEX DEFERRED MODE, TEST
*****
*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
1$: MOV R0,(R1)+
    SOB R2,1$
    MOV #WWW10,ERRVECT ;SET UP FOR TRAPS TO 4.
    MOV #WWWTP1,R0      ;SET UP ACO.
    LDD (R0),ACO
    MOV #WWW2,$TMP2
    MOV #WWWBF1-5701,R0 ;SET UP THE DESTINATION ADDRESS.
    MOV #1,R1
    MOV #WWWBF0,WWWBF1
WWW2: STD ACO,@5701(R0) ;TEST INSTRUCTION.

    CMP R0,#WWWBF1-5701 ;IS R0 CORRECT?
    BNE WWW15           ;BRANCH IF INCORRECT.
    MOV #WWWBF0,R2      ;WAS THE DATA OUTPUT CORRECTLY?
    MOV #WWWTP1,R3
    MOV #4,R4
1$: CMP (R2)+,(R3)+
    BNE WWW20           ;BRANCH IF DATA IS INCORRECT.
    SOB R4,1$
    BR WWWDONE

WWWBF0: -1
        -1
        -1
        -1

WWWTP1: 41424
        34445
        46475
        051525

WWWBF1: -1
        -1
        -1
        -1

;TRAP THROUGH 4 TO HERE.
WWW10: MOV (SP),R2      ;SEE IF THE TRAP OCCURRED ON THE TEST INSTR.
    CMP R2,#WWW2+2
    BEQ WWW11          ;BRANCH IF YES.
    CMP R2,#WWW2+4
    BEQ WWW11          ;BRANCH IF YES.
    JMP FPSPUR ;OTHERWISE GO REPORT SPURIOUS TRAP TO 4.
;REPORT FAILURE OF FDST FORK RESULTED IN AN ODD ADDRESS TRAP TO 4.
WWW11: MOV R2,$TMP2
    
```

```

2364 011576 022626          CMP      (SP)+,(SP)+
2365 011600 104034          1$:    ERROR  +34          ;FDST FORK X ODD ADD
2366 011602 000416          BR      WWWDONE
2367
2368          ;REPORT RO MODIFIED.
2369 011604 010037 001242    WWW15:  MOV     R0,$TMP4
2370 011610 012737 003617 001240    MOV     #WWWBFO-5701,$TMP3
2371 011616 104035          1$:    ERROR  +35          ;RO MODIFIED!
2372 011620 000407          BR      WWWDONE
2373
2374          ;REPORT DATA INCORRECT
2375 011622 012737 011520 001240    WWW20:  MOV     #WWWBFO,$TMP3
2376 011630 012737 011530 001242    MOV     #WWWTP1,$TMP4
2377 011636 104036          1$:    ERROR  +36          ;BAD DATA
2378 011640          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?).
  
```

2374

2385
2386
2387
2388 011644 000004
2389 011646 004737 012202
2390 011652 000000
2391 011654 000000
2392 011656 000000
2393 011660 000000
2394 011662 000000
2395 011664 000000
2396 011666 000000
2397 011670 000000
2398 011672 000000
2399 011674 000000
2400 011676 177777
2401 011700 177777
2402 011702 047000
2403 011704 047004
2404 011706 177777
2405 011710 147004
2406 011712 104042
2407 011714 000401
2408 011716 104043
2409 011720
2410
2411 011720 104413 012202
2412 011722 004737
2413 011726 017203
2414 011730 142536
2415 011732 047506
2416 011734 172031
2417 011736 017203
2418 011740 142536
2419 011742 000000
2420 011744 000000
2421 011746 017203
2422 011750 142536
2423 011752 047506
2424 011754 172031
2425 011756 040000
2426 011760 040000
2427 011762 177777
2428 011764 177777
2429 011766 104044
2430 011770 000401
2431 011772 104040
2432 011774

```
.SBTTL TEST # 12 - STCFD TEST
:*****
:*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
              0
3$:            0          ;ERROR RES.
              0
              -1
              -1
4$:            47000      ;FPS BEFORE EXECUTION.
              47004      ;FPS AFTER EXECUTION.
              -1         ;FEC
              147004     ;ERROR FPS.
5$:            ERROR     +42      ;FDFL<---FDFLXST 767
              BR        6$
              ERROR     +43
6$:            ;BUT EZBT X ST560 TO 061 INTO 261
:
XXX2:
LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
JSR            PC,STCFDS
1$:            17203      ;AC
              142536
              47506
              172031
2$:            17203      ;RES
              142536
              0
              0
3$:            17203      ;ERROR RES.
              142536
              47506
              172031
4$:            40000      ;FPS BEFORE EXECUTION.
              40000      ;FPS AFTER EXECUTION.
              -1         ;FEC
              -1         ;ERROR FPS.
5$:            ERROR     +44      ;X11(1,0)<---0 X ST766
              BR        6$
              ERROR     +40
6$:
```

```

2433
2434 011774      :
      011774 104413 XXX3:
2435 011776 004737 012202 LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
2436 012002 050717 JSR          PC,STCFDS
2437 012004 027374 1$:          50717          ;AC
      012006 075767          27374
2438 012006 075767          75767
2439 012010 077071          77071
2440 012012 050717 2$:          50717          ;RES
      012014 027374          27374
2442 012016 000000          0
2443 012020 000000          0
2444 012022 000000 3$:          0          ;ERROR RES.
      012024 000000          0
2446 012026 000000          0
2447 012030 000000          0
2448 012032 047000 4$:          47000          ;FPS BEFORE EXECUTION.
      012034 047000          47000          ;FPS AFTER EXECUTION.
2450 012036 177777          -1          ;FEC
2451 012040 174002          174002          ;ERROR FPS.
2452 012042 104045 5$:          ERROR +45          ;BUT OPIC X ST251
      012044 000401          BR 6$
2454 012046 104046          ERROR +46          ;BUT EZBT X ST421
2455 012050 6$:
2456
2457 012050      :
      012050 104413 XXX4:
2458 012052 004737 012202 LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
2459 012056 020212 JSR          PC,STCFDS
2460 012060 032425 1$:          20212          ;AC
      012062 026272          32425
2461 012064 002123          26272
2462 012066 020212 2$:          02123          ;RES
      012070 032425          20212
2464 012072 000000          32425
2465 012074 000000          0
2466 012076 020212 3$:          0          ;ERROR RES.
      012100 032425          20212
2468 012102 100000          32425
2469 012104 000000          100000
2470 012106 040000          0
2471 012110 040000 4$:          40000          ;FPS BEFORE EXECUTION.
      012112 177777          40000          ;FPS AFTER EXECUTION.
2472 012114 177777          -1          ;FEC
2473 012116 104047          -1          ;ERROR FPS.
2474 012120 000401 5$:          ERROR +47          ;BUT FD IN ROUND X ST113
      012122 104040          BR 6$
2476 012124          ERROR +40
2477 012124 6$:
2478
2479
2480 012124      :
      012124 104413 XXX5:
2481 012126 004737 012202 LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
2482 012132 121314 JSR          PC,STCFDS
2483 012134 151617 1$:          121314          ;AC
      012136 101112          151617
2484 012140 131415          101112
2485 012142 121314          131415
2486 012144          121314          ;RES

```

2487 012144 151617
 2488 012146 000000
 2489 012150 000000
 2490 012152 021314
 2491 012154 151617
 2492 012156 000000
 2493 012160 000000
 2494 012162 040000
 2495 012164 040010
 2496 012166 177777
 2497 012170 177777
 2498 012172 104050
 2499 012174 000401
 2500 012176 104040
 2501 012200 000535
 2502
 2503
 2504
 2505
 2506
 2507
 2508
 2509
 2510
 2511
 2512
 2513
 2514
 2515
 2516
 2517
 2518
 2519
 2520
 2521
 2522
 2523
 2524
 2525
 2526
 2527
 2528
 2529
 2530
 2531
 2532
 2533
 2534
 2535
 2536
 2537
 2538 012202 012601
 2539 012204 012700 000200
 2540 012210 170100
 2541 012212 010100
 2542 012214 172410
 2543 012216 012700 177777

```

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$: ERROR +50 ;BUT ENBT X ST567 OR BAD SIGN ST460
    BR 6$
    ERROR +40
6$: BR XXXDONE
    
```

: 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 +X ;DATA ERROR.
: BR CONT
: ERR2: ERROR +X ;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
 : RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND STCFDS WILL
 : REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.

```

STCFDS: MOV (SP)+,R1 ;PICK UP THE POINTER TO THE OPERANDS.
        MCV #200,R0 ;ENTER DOUBLE FLOATING MODE.
        LDFPS R0
        MOV R1,R0 ;LOAD ACO.
        LDD (R0),ACO
        MOV #-1,R0 ;FILL THE OUTPUT BUFFER WITH -1'S.
    
```

```

2544 012222 012702 012464      MOV    #STCFT,R2
2545 012226 012703 000004      MOV    #4,R3
2546 012232 010022      1$:   MOV    R0,(R2)+
2547 012234 077302      SOB    R3,1$
2548 012236 016100 000030      MOV    30(R1),R0      ;LOAD THE FPS.
2549 012242 170100      LDFPS R0
2550 012244 012737 012256 001236      MOV    #2$,STMP2
2551 012252 012700 012464      MOV    #STCFT,R0      ;SET UP THE DESTINATION ADDRESS.
2552 012256 176010      2$:   STCFD  AC0,(R0)      ;TEST INSTRUCTION.
2553
2554 012260 170204      STFPS  R4      ;GET THE FPS.
2555 012262 170305      STST   R5      ;GET THE FEC.
2556 012264 010102      MOV    R1,R2      ;SAVE THE DATA IN CASE OF ERROR.
2557 012266 010237 001240      MOV    R2,STMP3
2558 012272 062702 000010      ADD    #10,R2
2559 012276 010237 001244      MOV    R2,STMP5
2560 012302 012737 012464 001242      MOV    #STCFT,STMP4
2561 012310 010437 001250      MOV    R4,STMP7
2562 012314 016137 000032 001252      MOV    32(R1),STMP10
2563
2564 012322 010102      MOV    R1,R2      ;CHECK THE RESULT.
2565 012324 062702 000010      ADD    #10,R2
2566 012330 012703 012464      MOV    #STCFT,R3
2567 012334 012700 000004      MOV    #4,R0
2568 012340 022223      3$:   CMP    (R2)+,(R3)+
2569 012342 001014      BNE    15$      ;BRANCH IF INCORRECT.
2570 012344 077003      SOB    R0,3$
2571
2572 012346 016102 000032      MOV    32(R1),R2
2573 012352 020204      CMP    R2,R4      ;IS THE FPS CORRECT?
2574 012354 001025      BNE    20$      ;BRANCH IF FPS INCORRECT.
2575 012356 005702      TST   R2      ;IF EXPECTED FPS IS NEGATIVE, THEN
2576 012360 100003      BPL   4$      ;GO AHEAD AND CHECK THE FEC.
2577 012362 026105 000036      CMP    36(R1),R5
2578 012366 001027      BNE    25$      ;BRANCH IF FEC IS INCORRECT.
2579 012370 000161 000046      4$:   JMP    46(R1)      ;RETURN.
2580
2581      ;RESULT INCORRECT:
2582 012374 010102      15$:  MOV    R1,R2      ;SEE IF ERROR WAS ANTICIPATED.
2583 012376 062702 000020      ADD    #20,R2
2584 012402 012703 012464      MOV    #STCFT,R3
2585 012406 012700 000004      MOV    #4,R0
2586 012412 022223      16$:  CMP    (R2)+,(R3)+
2587 012414 001003      BNE    17$      ;BRANCH IF NOT ANTICIPATED.
2588 012416 077003      SOB    R0,16$
2589 012420 000161 000040      JMP    40(R1)      ;IF ERROR WAS ANTICIPATED RETURN.
2590      ;OTHERWISE REPORT RESULT INCORRECT HERE.
2591 012424
2592 012424 104037      17$:
2593 012426 000760      18$:  ERROR  +37      ;DATA ERROR
2594      BR    4$
2595
2596 012430 020461 000034      ;FPS INCORRECT:
2597 012434 001002      20$:  CMP    R4,34(R1)      ;WAS THE ERROR ANTICIPATED.
2598 012436 000161 000044      BNE    21$      ;BRANCH IF NOT ANTICIPATED.
2599      JMP    44(R1)      ;IF IT WAS ANTICIPATED RETURN.
2600      ;THE FPS ERROR WAS NOT ANTICIPATED SO REPORT FPS INCORRECT HERE.

```

```
2601 012442
2602 012442 104040
2603 012444 000751
2604
2605
2606 012446 016137 000036 001256 ;REPORT FEC INCORRECT:
2607 012454 010537 001254 25$: MOV 36(R1),STMP12
2608 012460 104041 MOV R5,STMP11
2609 012462 000742 26$: ERROR +41 ;FEC X
2610 012464 177777 177777 BR 4$
2611 012474 STCFD: -1,-1,-1,-1
012474 104412 XXXDONE:
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?).
```

2612

2618

```
.SBTTL TEST # 13 - STCDF TEST  
:*****  
:TEST 13 STCDF TEST  
:*****  
:THIS IS A TEST OF THE STCDF INSTRUCTION.  
:*****  
TST13: SCOPE
```

012476 000004

```
2619  
2620  
2621 012500  
2622 012500 104413  
2623 012502 004737 013036  
2624 012506 000000  
2625 012510 000000  
2626 012512 000000  
2627 012514 000000  
2628 012516 000000  
2629 012520 000000  
2630 012522 177777  
2631 012524 177777  
2632 012526 000000  
2633 012530 000000  
2634 012532 000000  
2635 012534 000000  
2636 012536 047200  
2637 012540 047204  
2638 012542 177777  
2639 012544 177777  
2640 012546 104054  
2641 012550 000401  
2642 012552 104052  
2643  
2644 012554  
2645 012554 104413  
2646 012556 004737 013036  
2647 012562 067574  
2648 012564 073727  
2649 012566 170777  
2650 012570 067574  
2651 012572 067574  
2652 012574 073730  
2653 012576 177777  
2654 012600 177777  
2655 012602 067574  
2656 012604 073727  
2657 012606 177777  
2658 012610 177777  
2659 012612 040200  
2660 012614 040200  
2661 012616 177777  
2662 012620 177777  
2663 012622 104055  
2664 012624 000401  
2665 012626 104052
```

```
:AC=0  
YYY1:  
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.  
JSR PC,STCDF5  
1$: 0 ;AC  
0  
0  
0  
2$: 0 ;RES  
0  
-1  
-1  
3$: 0 ;ERROR RES.  
0  
0  
0  
4$: 47200 ;FPS BEFORE EXECUTION.  
47204 ;FPS AFTER EXECUTION.  
-1 ;FEC  
-1 ;ERROR FPS.  
5$: ERROR +54 ;FDFL<---FDFL X ST767  
BR 6$  
ERROR +52 ;FPS INCORRECT.  
6$:  
YYY2:  
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.  
JSR PC,STCDF5  
1$: 67574 ;AC0  
73727  
170777  
2$: 67574 ;RES  
73730  
-1  
-1  
3$: 67574 ;ERROR RES.  
73727  
-1  
-1  
4$: 40200 ;FPS BEFORE EXECUTION.  
40200 ;FPS AFTER EXECUTION.  
-1 ;FEC  
-1 ;ERROR FPS.  
5$: ERROR +55 ;EITHER ROUND FAILED OR WENT TO 766 X1(1,0)<---0 INTO 767  
BR 6$  
ERROR +52  
6$:
```

```
2666  
2667 012630  
012630 104413  
2668 012632 004737 013036  
2669 012636 077777  
2670 012640 177777  
2671 012642 100000  
2672 012644 000000  
2673 012646 000000  
2674 012650 000000  
2675 012652 177777  
2676 012654 177777  
2677 012656 077777  
2678 012660 177777  
2679 012662 177777  
2680 012664 177777  
2681 012666 040200  
2682 012670 040206  
2683 012672 177777  
2684 012674 040204  
2685 012676 104055  
2686 012700 000401  
2687 012702 104056  
2688 012704  
2689  
2690 012704  
012704 104413  
2691 012706 004737 013036  
2692 012712 077777  
2693 012714 177777  
2694 012716 100000  
2695 012720 000000  
2696 012722 000000  
2697 012724 000000  
2698 012726 177777  
2699 012730 177777  
2700 012732 077777  
2701 012734 177777  
2702 012736 177777  
2703 012740 177777  
2704 012742 040200  
2705 012744 040206  
2706 012746 177777  
2707 012750 140206  
2708 012752 104055  
2709 012754 000401  
2710 012756 104057  
2711 012760  
2712  
2713 012760  
012760 104413  
2714 012762 004737 013036  
2715 012766 177777  
2716 012770 177777  
2717 012772 100000  
2718 012774 000000  
2719 012776 100000
```

YYY3:
LPERR
JSR PC,STCDFS ;SET UP THE LOOF ON ERROR ADDRESS.
1\$: 77777 ;ACO
-1
100000
0
2\$: 0 ;RES
0
-1
-1
3\$: 77777 ;ERROR RES.
-1
-1
-1
4\$: 40200 ;FPS BEFORE EXECUTION.
40206 ;FPS AFTER EXECUTION.
-1 ;FEC
40204 ;ERROR FPS.
5\$: ERROR +55
BR 6\$
ERROR +56 ;BUT EZBT X ST421 TO 062 INTO 262
6\$:
YYY4:
LPERR
JSR PC,STCDFS ;SET UP THE LOOP ON ERROR ADDRESS.
1\$: 77777 ;ACO
-1
100000
0
2\$: 0 ;RES
0
-1
-1
3\$: 77777 ;ERROR RES.
-1
-1
-1
4\$: 40200 ;FPS BEFORE EXECUTION.
40206 ;FPS AFTER EXECUTION.
-1 ;FEC
140206 ;ERROR FPS.
5\$: ERROR +55
BR 6\$
ERHOR +57 ;BUT FIV ST262 TO 123 INTO 103
6\$:
YYY5:
LPERR
JSR PC,STCDFS ;SET UP THE LOOP ON ERROR ADDRESS.
1\$: 77777 ;ACO
-1
100000
0
2\$: 100000 ;RES

2720 013000 000000
 2721 013002 177777
 2722 013004 177777
 2723 013006 000000
 2724 013010 000000
 2725 013012 177777
 2726 013014 177777
 2727 013016 047200
 2728 013020 147216
 2729 013022 000010
 2730 013024 047206
 2731 013026 104060
 2732 013030 00040
 2733 013032 104061
 2734 013034 000535
 2735
 2736
 2737
 2738
 2739
 2740
 2741
 2742
 2743
 2744
 2745
 2746
 2747
 2748
 2749
 2750
 2751
 2752
 2753
 2754
 2755
 2756
 2757
 2758
 2759
 2760
 2761
 2762
 2763
 2764
 2765
 2766
 2767 013036 012601
 2768 013040 012700 000200
 2769 013044 170100
 2770 013046 010100
 2771 013050 172410
 2772 013052 012700 177777
 2773 013056 012702 013320
 2774 013062 012703 000004
 2775 013066 010022
 2776 013070 077302

```

0
-1
-1
3$: 0 ;ERROR RES.
0
-1
-1
4$: 47200 ;FPS BEFORE EXECUTION.
147216 ;FPS AFTER EXECUTION.
10 ;FEC
47206 ;ERROR FPS.
5$: ERROR +60 ;BUT FIV ST262 FAIL TO 103 INT 123
BR 6$
ERROR +61 ;BUT FLAG ST 147 X TO ST 361 INTO 365
6$: BR YYDONE
;THIS SUBROUTINE, STCFDS, IS USED TO SET UP THE OPERANDS, EXECUTE
;THE STCDF INSTRUCTION AND CHECK THE RESULTS. A CALL
;TO IT IS MADE THUS:
.....
JSR PC,STCFDS
ACARG: .WORD X,X,X,X ;AC OPERAND
RES: .WORD X,X,X,X ;EXPECTED RESULT
ERRES: .WORD X,X,X,X ;ERROR RESULT
FPSB: .WORD X ;FPS BEFORE EXECUTION
FPSA: .WORD X ;FPS AFTER EXECUTION
FEC: .WORD X ;EXPECTED FEC
ERFPS: .WORD X ;ERROR FPS.
ERR1: ERROR +X ;DATA ERROR.
BR CONT
ERR2: ERROR +X ;FPS ERROR.
CONT: ;RETURN ADDRESS
.....
;THE OPERANDS ARE SET UP (USING ACO AS THE ACCUMULATOR). THEN
;THE STCFD INSTRUCTION IS EXECUTED.
;THE RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
;COMPARED WITH FPSA IF THIS TOO IS CORRECT STCFDS RETURNS CONTROL
;TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD STCFDS
;COMPARE IT TO ERROR FPS. IF THIS MATCHES THEN STCFDS WILL RETURN
;TO THE ERROR CALL AT ERR2, OTHERWISE STCFDS ITSELF
;REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE
;STCFD IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
;ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
;THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN STCFDS
;WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR1. OTHERWISE THE
;RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND STCFDS WILL
;REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.
STCFDS: MOV (SP)+,R1 ;PICK UP THE POINTER TO THE OPERANDS.
MOV #200,R0 ;ENTER DOUBLE FLOATING MODE.
LDFPS R0
MOV R1,R0 ;LOAD ACO.
LDD (R0),ACO
MCM #-1,R0 ;FILL THE OUTPUT BUFFER WITH -1'S.
MOV #STCDT,R2
MOV #4,R3
1$: MOV R0,(R2)+
SOB R3,1$
    
```

```

2777 013072 016100 000030      MOV      30(R1),R0      ;LOAD THE FPS.
2778 013076 170100      LDFPS   R0
2779 013100 012737 013112 001236      MOV      #2$,STMP2
2780 013106 012700 013320      MOV      #STCDT,R0      ;SET UP THE DESTINATION ADDRESS.
2781 013112 176010      STCDF   ACO,(R0)      ;TEST INSTRUCTION.
2782
2783 013114 170204      STFPS   R4      ;GET THE FPS.
2784 013116 170305      STST   R5      ;GET THE FEC.
2785 013120 010102      MOV      R1,R2      ;SAVE THE DATA IN CASE OF ERROR.
2786 013122 010237 001240      MOV      R2,STMP3
2787 013126 062702 000010      ADD      #10,R2
2788 013132 010237 001244      MOV      R2,STMP5
2789 013136 012737 013320 001242      MOV      #STCDT,STMP4
2790 013144 010437 001250      MOV      R4,STMP7
2791 013150 016137 000032 001252      MOV      32(R1),STMP10
2792
2793 013156 010102      IACV   R1,R2      ;CHECK THE RESULT.
2794 013160 062702 000010      ADD      #10,R2
2795 013164 012703 013320      MOV      #STCDT,R3
2796 013170 012700 000004      MOV      #4,R0
2797 013174 022223      CMP     (R2)+,(R3)+      ;BRANCH IF INCORRECT.
2798 013176 001014      BNE     15$
2799 013200 077003      SOB     R0,3$
2800
2801 013202 016102 000032      MOV      32(R1),R2
2802 013206 020204      CMP     R2,R4      ;IS THE FPS CORRECT?
2803 013210 001025      BNE     20$      ;BRANCH IF FPS INCORRECT.
2804 013212 005702      TST    R2      ;IF EXPECTED FPS IS NEGATIVE, THEN
2805 013214 100003      BPL     4$      ;GO AHEAD AND CHECK THE FEC.
2806 013216 026105 000034      CMP     34(R1),R5
2807 013222 001027      BNE     25$      ;BRANCH IF FEC IS INCORRECT.
2808 013224 000161 000046      JMP     46(R1)      ;RETURN.
2809
2810      ;RESULT INCORRECT:
2811 013230 010102 15$:      MOV      R1,R2      ;SEE IF ERROR WAS ANTICIPATED.
2812 013232 062702 000020      ADD      #20,R2
2813 013236 012703 013320      MOV      #STCDT,R3
2814 013242 012700 000004      MOV      #4,R0
2815 013246 022223      CMP     (R2)+,(R3)+      ;BRANCH IF NOT ANTICIPATED.
2816 013250 001003      BNE     17$
2817 013252 077003      SOB     R0,16$
2818 013254 000161 000040      JMP     40(R1)      ;IF ERROR WAS ANTICIPATED RETURN.
2819      ;OTHERWISE REPORT RESULT INCORRECT HERE.
2820 013260 17$:
2821 013260 104051 18$:      ERROR   +51      ;DATA ERROR
2822 013262 000760      BR      4$
2823
2824      ;FPS INCORRECT:
2825 013264 020461 000034      CMP     R4,34(R1)      ;WAS THE ERROR ANTICIPATED.
2826 013270 001002      BNE     21$      ;BRANCH IF NOT ANTICIPATED.
2827 013272 000161 000044      JMP     44(R1)      ;IF IT WAS ANTICIPATED RETURN.
2828
2829      ;THE FPS ERROR WAS NOT ANTICIPATED SO REPORT FPS INCORRECT HERE.
2830 013276 21$:
2831 013276 104052 22$:      ERROR   +52      ;FPS X
2832 013300 000751      BR      4$
2833

```

```
2834  
2835 013302 016137 000036 001256 :REPORT FEC INCORRECT:  
2836 013310 010537 001254 25$: MOV 36(R1),STMP12  
2837 013314 104053 : MOV R5,STMP11  
2838 013316 000742 26$: ERROR +53 :FEC X  
2839 013320 177777 177777 177777 STCDT: -1,-1,-1,-1  
2840 013330 : YYYDONE:  
013330~ 104412 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?).
```

2846

```

.SBTTL TEST # 14 - STCFD WITH ILLEGAL ACCUMULATOR TEST
:*****
:TEST 14 STCFD WITH ILLEGAL ACCUMULATOR TEST
:
:THIS TEST STCFD WITH ILLEGAL AC 6.
:
:*****
    
```

```

2847 013332 000004
2848 013334
2849 013334 104413
2850 013336 012700 040000
2851 013342 170100
2852 013344 012737 013352 001236
2853 013352 176006
2854 013354 170204
2855 013356 170305
2856 013360 020427 140000
2857 013364 001004
2858 013366 022705 000002
2859 013372 001010
2860 013374 000415
2861
2862
2863 013376 010437 001242
2864 013402 012737 140000 001240
2865 013410 104062
2866 013412 000406
2867
2868
2869 013414 010537 001242
2870 013420 012737 000002 001240
2871 013426 104063
2872 013430
2873 013430 104412
    
```

```

TST14: SCOPE
ZZZ1:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
MOV #40000,R0 ;DISSABLE INTERRUPTS.
LDFPS R0
MOV #ZZZ2,$TMP2
ZZZ2: STCFD AC0,AC6 ;THIS TEST INSTRUCTION SHOULD CAUSE AN ERROR.

STFPS R4 ;GET FPS.
STST R5 ;GET FEC.
CMP R4,#140000 ;IS FPS CORRECT?
BNE ZZZ10 ;BRANCH IF INCORRECT FPS.
CMP #2,R5 ;IS FEC CORRECT?
BNE ZZZ15 ;BRANCH IF INCORRECT.
BR ZZZDONE

;REPORT FPS INCORRECT AFTER USE OF ILLEGAL ACCUMULATOR.
ZZZ10: MOV R4,$TMP4
MOV #140000,$TMP3
1$: ERROR +62 ;BUT FDSI ST767 X TO 567 INTO 577
BR ZZZDONE

;REPORT FEC INCORRECT AFTER USE OF ILLEGAL ACCUMULATOR.
ZZZ15: MOV R5,$TMP4
MOV #2,$TMP3
1$: ERROR +63 ;FEC<---2 ST577 X
ZZZDONE:
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?).
    
```

2879

2880 013432 000004
 2881 013434 104413
 2882 013436 012700 013622
 2883 013442 012701 013612
 2884 013446 012702 000004
 2885 013452 012021
 2886 013454 077202
 2887 013456 012700 013612
 2888 013462 012701 000213
 2889 013466 170101
 2890 013470 012737 013476 001236
 2891 013476 170410
 2892 013500 170205
 2893 013502 012702 000004
 2894 013506 012701 013612
 2895 013512 005721
 2896 013514 001010
 2897 013516 077203
 2898 013520 022705 000204
 2899 013524 001014
 2900 013526 020027 013612
 2901 013532 001020
 2902 013534 000442
 2903
 2904
 2905 013536 012737 013612 001240
 2906 013544 012737 013632 001242
 2907 013552 104064
 2908 013554 000432
 2909
 2910
 2911 013556 010437 001242
 2912 013562 012737 000204 001240
 2913 013570 104065
 2914 013572 000423
 2915
 2916
 2917 013574 010037 001242
 2918 013600 012737 013612 001240
 2919 013606 104066
 2920 013610 000414
 2921
 2922
 2923 013612 073475
 2924 013614 067707
 2925 013616 127347
 2926 013620 056770
 2927

```

.SBTTL TEST # 15 - CLRD TEST
:*****
:*TEST 15      CLRD TEST
:*
:*THIS IS A TEST OF THE CRLF AND CLRD INSTRUCTIONS.
:*
:*****
TST15: SCOPE
AAB1:
        LPERR                ;SET UP THE LOOP ON ERROR ADDRESS.
        MOV #AABTP1,R0        ;SET UP OUTPUT BUFFER
        MOV #AABBFO,R1
        MOV #4,R2
1$:     MOV (R0)+,(R1)+
        SOB R2,1$
        MOV #AABBFO,R0        ;SET UP DESTINATION OPERAND ADDRESS.
        MCV #213,R1          ;SET UP FPS.
        LDFPS R1
        MOV #2$,$TMP2
2$:     CLRD (R0)            ;TEST INSTRUCTION.

        STFPS R5             ;GET FPS.
        MOV #4,R2            ;SEE IF RESULT CLEAR, 0.
        MOV #AABBFO,R1
3$:     TST (R1)+
        BNE AAB2             ;BRANCH IF RESULT INCORRECT, NOT 0.
        SOB R2,3$
        CMP #204,R5          ;SEE IF FPS IS CORRECT.
        BNE AAB3             ;BRANCH IF INCORRECT.
        CMP R0,#AABBFO      ;SEE IF R0 IS CORRECT.
        BNE AAB4             ;BRANCH IF R0 IS INCORRECT.
        BR AABDONE

;RESULT NOT 0, REPORT ERROR.
AAB2:   MOV #AABBFO,$TMP3
        MOV #AABTP2,$TMP4
1$:     ERROR +64            ;BAD DATA - 0 X 11+ZERO ST770 X
        BR AABDONE

;REPORT FPS INCORRECT:
AAB3:   MOV R4,$TMP4
        MOV #204,$TMP3
1$:     ERROR +65            ;BAD FPS
        BR AABDONE

;REPORT R0 INCORRECT.
AAB4:   MOV R0,$TMP4
        MOV #AABBFO,$TMP3
1$:     ERROR +66
        BR AABDONE

;THIS IS THE TEST DATA BUFFER, OUTPUT DATA BUFFER.
AABBFO: 73475
        67707
        127347
        56770

;THIS IS THE DATA USED TO SET UP THE OUTPUT BUFFER.
    
```


2928 013622 073475
2929 013624 067707
2930 013626 127347
2931 013630 056770
2932
2933 013632 000000
2934 013634 000000
2935 013636 000000
2936 013640 000000
2937 013642
013642 104412

AABTP1: 73475
67707
127347
56770

:THIS IS THE EXPECTED DATA, RESULT:

AABTP2: 0
0
0
0

AABDUNE:
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?).

2938

```

2944 .SBTTL TEST # 16 - CLRD WITH ILLEGAL ACCUMULATOR TEST
      :*****
      :*TEST 16 CLRD WITH ILLEGAL ACCUMULATOR TEST
      :*
      :*THIS IS A TEST OF CLRD WITH ILLEGAL AC7.
      :*
      :*****
2945 013644 000004 TST16: SCOPE
      013646 104413 CCB1:
2946 013650 012700 040200 LPPER ;SET UP THE LOOP ON ERROR ADDRESS.
2947 013654 170100 MOV #40200,R0 ;SET UP THE FPS, NO INTERRUPTS AND FD 1.
2948 013656 012737 013664 001236 LDFPS R0
2949 013664 170407 CCB2: MOV #CCB2,$TMP2
      STFP ;TEST INSTRUCTION.
2950 STFP R4 ;GET FPS.
2951 013666 170204 STST R5 ;GET FEC.
2952 013670 170305 CMP R4,#140200 ;IS THE FPS CORRECT?
2953 013672 020427 140200 BNE CCB10 ;BRANCH IF FPS IS INCORRECT.
2954 013676 001004 CMP #2,R5 ;IS THE FEC CORRECT?
2955 013700 022705 000002 BNE CCB15 ;BRANCH IF FEC IS INCORRECT.
2956 013704 001010 BR CCBDONE
2957 013706 000415
2958
2959 ;REPORT INCORRECT FPS:
2960 013710 010437 001242 CCB10: MOV R4,$TMP4
2961 013714 012737 140200 001240 MOV #140200,$TMP3
2962 013722 104067 1$: ERROR +67 ;BUT FDST ST 700X TO 607 INTO 677
2963 013724 000406 BR CCBDONE
2964
2965 ;REPORT INCORRECT FEC:
2966 013726 010537 001242 CCB15: MOV R5,$TMP4
2967 013732 012737 000002 001240 MOV #2,$TMP3
2968 013740 104070 1$: ERROR +70 ;FEC<---2 ST 677 X
2969 013742 CCBDONE:
      013742 104412 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?).

2978
2979 ;TEST TITLE:NEGF, ABSF AND TSTF SOURCE MODE 0 WITH ILLEGAL AC7, TEST
    
```

2980

```
.SBTTL TEST # 17 - SEE ABOVE COMMENT FOR TEST TITLE
:*****
:*TEST 17 SEE ABOVE COMMENT FOR TEST TITLE
:*
:*THIS IS A TEST OF THE SPECIAL
:*DEST FLOWS USING THE NEGD INST
:*WITH MODE ZERO AND ILLEGAL
:*AC7.
:*
:*****
TST17: SCOPE
```

013744 000004

2981

2982 013746

013746 104413

2983 013750 012700 040200

2984 013754 170100

2985 013756 012737 013764 001236

2986

2987 013764 170707

2988

2989 013766 170204

2990 013770 170305

2991

2992 013772 022704 140200

2993 013776 001004

2994 014000 022705 000002

2995 014004 001010

2996 014006 000416

2997

2998

2999 014010 012737 140200 001240

3000 014016 010437 001242

3001 014022 104176

3002 014024 000407

3003

3004

3005 014026 012737 000002 001240

3006 014034 010537 001242

3007 014040 104377

3008 014042 000044

3009 014044

014044 104412

```
VVB1: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
MOV #40200,R0 ;SET UP THE FPS, FID 1 AND FD 1.
LDFPS R0
MCV #VVB2,$TMP2

VVB2: NEG D AC7 ;TEST INSTRUCTION.

STFPS R4 ;GET FPS.
STST R5 ;GET FEC.

CMP #140200,R4 ;IS FPS CORRECT?
BNE VVB10 ;BRANCH IF FPS IS INCORRECT.
CMP #2,R5 ;IS FEC CORRECT?
BNE VVB15 ;BRANCH IF FEC IS INCORRECT.
BR VVB DONE

;REPORT INCORRECT FPS:
VVB10: MOV #140200,$TMP3
MOV R4,$TMP4
1$: ERROR +176 ;FPS BAD
BR VVB DONE

;REPORT FEC INCORRECT:
VVB15: MOV #2,$TMP3
MOV R5,$TMP4
1$: ERROR +377 ;FEC BAD
.WORD 44

VVB DONE:
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?).
```

3010

3018

;TEST TITLE:NEG D, ABSF AND TSTF SOURCE MODE 0 TEST

3019

```
.SBTTL TEST # 20 - SEE ABOVE COMMENT FOR TEST TITLE
:*****
:*TEST 20 SEE ABOVE COMMENT FOR TEST TITLE
:*
:*THIS IS A TEST THE NEG, ABSF AND TSTF
:*SOURCE FLOWS. THE NEGD INSTRUCTION
:*IS USED TO TEST MODE 0
:*
:*****
TST20: SCOPE
```

```
3020 014046 000004
3021 014050
3021 014050 104413
3022 014052 012700 000200
3023 014056 170100
3024 014060 012700 014222
3025 014064 172410
3026 014066 005000
3027 014070 170100
3028 014072 012700 014232
3029 014076 172410
3030
3031 014100 012700 000201
3032 014104 170100
3033 014106 012737 014114 001236
3034
3035 014114 170700
3036
3037 014116 170205
3038 014120 012700 000200
3039 014124 170100
3040 014126 012700 014242
3041 014132 174010
3042
3043 014134 012701 000004
3044 014140 005720
3045 014142 001005
3046 014144 077103
3047 014146 022705 000204
3048 014152 001014
3049 014154 000442
3050
3051
3052 014156 012737 014232 001242
3053 014164 012737 014252 001240
3054 014172 012737 014242 001244
3055 014200 104071
3056 014202 000427
3057
3058
3059 014204 012737 000204 001240
3060 014212 010537 001242
3061 014216 104072
3062 014220 000420
3063
3064
3065 014222 101112

DDB1:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
MOV #200,R0 ;SET FD MODE.
LDFPS R0
MOV #DDBTP1,R0 ;SET UP ACO.
LDD (R0),ACO ;SET ACO - 0
CLR R0 ;CLEAR THE FPS.
LDFPS R0
MOV #DDBTP2,R0 ;LOAD ACO TO BE A FLOATING 0.
LDF (R0),ACO ;SET ACO=ZERO
;FLOAT
;SET FD MODE.
MOV #201,R0
LDFPS R0
MOV #DDB2,$TMP2

DDB2:
NEGD ACO ;TEST INSTRUCTION.
STFPS R5 ;GET FPS.
MOV #200,R0 ;SET FD MODE.
LDFPS R0
MOV #DDBBFO,R0 ;GET THE RESULT OUT OF ACO.
STD ACO,(R0) ;SEE IF THE RESULT IS CORRECT.
MOV #4,R1
1$: TST (R0)+
BNE DDB5 ;BRANCH IF THE RESULT IS INCORRECT.
SOB R1,1$
CMP #204,R5 ;IS THE FPS CORRECT?
BNE DDB6 ;BRANCH IF THE FPS IS INCORRECT.
BR DDBDONE

;RESULT INCORRECT, REPORT FAILURE:
DDB5: MOV #DDBTP2,$TMP4 ;EXPECT D0
MOV #DDBTP3,$TMP3 ;PREV FO IMPURE
MOV #DDBBFO,$TMP5 ;GOT
1$: ERROR +71
BR DDBDONE

;REPORT FPS INCORRECT:
DDB6: MOV #204,$TMP3
MOV R5,$TMP4
1$: ERROR +72
BR DDBDONE

;THESE ARE TEST DATA TABLES AND AN OUTPUT BUFFER.
DDBTP1: 101112
```

3066	014224	131415		131415
3067	014226	161710		161710
3068	014230	111213		111213
3069	014232	000000	DDBTP2:	0
3070	014234	000000		0
3071	014236	000000		0
3072	014240	000000		0
3073				
3074	014242	177777	DDBBF0:	-1
3075	014244	177777		-1
3076	014246	177777		-1
3077	014250	177777		-1
3078	014252	000000	DDBTP3:	0
3079	014254	000000		0
3080	014256	161710		161710
3081	014260	111213		111213
3082				
3083	014262		DDBDONE:	
	014262	104412		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?).

3084
3085

;TEST TITLE:NEGF, ABSF AND TSTF SOURCE MOD 1 TEST

3086

```
.SBTTL TEST # 21 - SEE ABOVE COMMENT FOR TEST TITLE
:*****
:*TEST 21 SEE ABOVE COMMENT FOR TEST TITLE
:*
:*THIS IS A TEST THE NEGF, ABSF AND TSTF
:*SOURCE FLOWS. THE NEGD INSTRUCTION
:*IS USED TO TEST MODE 1
:*
:*****
```

```
014264 000004
3087
3088 014266
014266 104413
3089 014270 012700 014376
3090 014274 012701 014426
3091 014300 012702 000004
3092 014304 012021
3093 014306 077202
3094 014310 012700 000200
3095 014314 170100
3096 014316 012700 014426
3097 014322 012737 014336 001236
3098 014330 012737 014436 000004
3099 014336 170700
3100
3101 014340 170205
3102 014342 012701 014426
3103 014346 012702 000004
3104 014352 005721
3105 014354 001046
3106 014356 077203
3107
3108 014360 020627 014426
3109 014364 001055
3110 014366 022705 000204
3111 014372 001061
3112 014374 000466
3113
3114
3115 014376 000177
3116 014400 167574
3117 014402 137271
3118 014404 107675
3119 014406 000000
3120 014410 000000
3121 014412 000000
3122 014414 000000
3123 014416 177777
3124 014420 177777
3125 014422 177777
3126 014424 177777
3127 014426 177777
3128 014430 177777
3129 014432 177777
3130 014434 177777
3131
3132
```

```
TST21: SCOPE
EEB1:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
MOV #EEBTP1,R0 ;SET UP THE DATA BUFFER.
MOV #EEBBF1,R1
MOV #4,R2
1$: MOV (R0)+,(R1)+
SOB R2,1$
MOV #200,R0 ;SET FD MODE.
LDFPS R0
MOV #EEBBF1,R0 ;SET UP THE OPERAND ADDRESS.
MOV #EEB2,$TMP2
MOV #EEB10,ERRVECT ;SET UP VECTOR 4 IN CASE OF ERROR.
EEB2: NEGD (R0) ;TEST INSTRUCTION.
STFPS R5 ;GET FPS.
MOV #EEBBF1,R1 ;SEE IF RESULT IS CORRECT.
MOV #4,R2
1$: TST (R1)+
BNE EEB15 ;BRANCH IF NOT CORRECT.
SOB R2,1$
CMP R0,#EEBBF1 ;IS R0 CORRECT?
BNE EEB20 ;BRANCH IF NOT CORRECT.
CMP #204,R5 ;IS THE FPS CORRECT?
BNE EEB25 ;BRANCH IF NOT CORRECT.
BR EEBDONE
;THESE ARE TEST DATA TABLES AND A BUFFER.
EEBTP1: 177
167574
137271
107675
EEBTP2: 0
0
0
0
EEBBF0: -1
-1
-1
-1
EEBBF1: -1
-1
-1
-1
;IF A TRAP TO 4 OCCURS COME HERE:
```

```

3133 014436 011602          EEB10: MOV      (SP),R2          ;SEE IF THE TRAP OCCURRED ON THE TEST INSTR.
3134 014440 020227 014340      CMP      R2,#EEB2+2
3135 014444 001405          BEQ      1$              ;BRANCH IF YES.
3136 014446 020227 014342      CMP      R2,#EEB2+4
3137 014452 001402          BEQ      1$              ;BRANCH IF YES.
3138 014454 000137 047302      JMP      CPSPUR ;OTHERWISE GO REPORT A SPURIOUS TRAP TO 4.
3139          ;REPORT A FAILURE IN THE FDST FLOWS RESULTED IN AN ODD ADDRESS TRAP TO 4.
3140 014460 022626          1$:  CMP      (SP)+,(SP)+      ;RESET THE STACK.
3141 014462 010237 001236      MOV      R2,$TMP2
3142 014466 104107          2$:  ERROR   +107            ;ODD ADRES
3143 014470 000430          BR       EEBDONE          ;BUT FDSTX IN ST 771
3144
3145          ;REPORT RESULT INCORRECT.
3146 014472 012737 014406 001242  EEB15: MOV      #EEBTP2,$TMP4
3147 014500 012737 014376 001240      MOV      #EEBTP1,$TMP3
3148 014506 012737 014426 001244      MOV      #EEBBF1,$TMP5
3149 014514 104073          1$:  ERROR   +73            ;BAD DATA X11*0 ST 312X
3150 014516 000415          BR       EEBDONE
3151
3152          ;RO INCORRECT:
3153 014520 012737 014426 001240  EEB20: MOV      #EEBBF1,$TMP3
3154 014526 010037 001242      MOV      R0,$TMP4
3155 014532 104074          1$:  ERROR   +74            ;RO BADX
3156 014534 000406          BR       EEBDONE
3157
3158          ;REPORT FPS INCORRECT:
3159 014536 010537 001240  EEB25: MOV      R5,$TMP3
3160 014542 012737 000204 001244      MOV      #204,$TMP5
3161 014550 104075          1$:  ERROR   +75            ;FPS X
3162
3163 014552          EEBDONE:
      014552 104412          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?).

3164
3165          ;TEST TITLE:NEGF, ABSF AND TSTF SOURCE MODE 2 TEST
    
```


3166
3167
3168 014554 000004
3169 014556 104413
3170 014560 012700 014666
3171 014564 012701 014716
3172 014570 012702 000004
3173 014574 012021
3174 014576 077202
3175 014600 012700 000200
3176 014604 170100
3177 014606 012700 014716
3178 014612 012737 014626 001236
3179 014620 012737 014726 000004
3180 014626 170620
3181
3182 014630 170205
3183 014632 012701 014716
3184 014636 012702 000004
3185 014642 005721
3186 014644 001046
3187 014646 077203
3188
3189 014650 020027 014726
3190 014654 001055
3191 014656 022705 000204
3192 014662 001061
3193 014664 000466
3194
3195
3196 014666 000177
3197 014670 167574
3198 014672 137271
3199 014674 107675
3200 014676 000000
3201 014700 000000
3202 014702 000000
3203 014704 000000
3204 014706 177777
3205 014710 177777
3206 014712 177777
3207 014714 177777
3208 014716 177777
3209 014720 177777
3210 014722 177777
3211 014724 177777
3212

```
.SBTTL TEST # 22 - SEE ABOVE COMMENT FOR TEST TITLE
:*****
:*TEST 22 SEE ABOVE COMMENT FOR TEST TITLE
:*
:*THIS IS A TEST THE NEGF, ABSF AND TSTF
:*SOURCE FLOWS. THE ABSD INSTRUCTION
:*IS USED TO TEST MODE 2
:*
:*****
TST22: SCOPE

FFB1:
      LPERR                                ;SET UP THE LOOP ON ERROR ADDRESS.
      MOV #FFBTP1,R0                       ;SET UP THE DATA BUFFER.
      MOV #FFBBF1,R1
      MOV #4,R2
1$:   MCV (R0)+,(R1)+
      SOB R2,1$
      MOV #200,R0                          ;SET FD.
      LDFPS R0
      MOV #FFBBF1,R0                       ;SET UP THE OPERAND ADDRESS.
      MOV #FFB2,$TMP2
      MOV #FFB10,ERRVECT ;SET UP VECTOR 4 IN CASE OF AN ERROR.

FFB2:  ABSD (R0)+                          ;TEST INSTRUCTION.

      STFPS R5                              ;GET FPS.
      MOV #FFBBF1,R1                       ;CHECK RESULT.
1$:   MOV #4,R2
      TST (R1)+
      BNE FFB15                             ;BRANCH IF INCORRECT.
      SOB R2,1$

      CMP R0,#FFBBF1+10                   ;IS R0 CORRECT?
      BNE FFB20                             ;BRANCH IF INCORRECT.
      CMP #204,R5                          ;IS THE FPS CORRECT?
      BNE FFB25                             ;BRANCH IF INCORRECT.
      BR FFBDONE

;THESE ARE TEST DATA TABLES AND DATA BUFFER.
FFBTP1: 177
        167574
        137271
        107675
FFBTP2: 0
        0
        0
        0
FFBBF0: -1
        -1
        -1
        -1
        -1
FFBBF1: -1
        -1
        -1
        -1
```

```

3213      :IF A TRAP TO 4 OCCURS COME HERE.
3214 014726 011602      FFB10: MOV (SP),R2      ;SEE IF THE TRAP OCCURRED ON THE TEST INSTRUCTION.
3215 014730 020227 014630      CMP R2,#FFB2+2
3216 014734 001405      BEQ 1$      ;BRANCH IF YES.
3217 014736 020227 014632      CMP R2,#FFB2+4
3218 014742 001402      BEQ 1$      ;BRANCH IF YES.
3219 014744 000137 047302      JMP CPSPUR ;OTHERWISE GO REPORT SPURIOUS TRAP TO 4.
3220      :REPORT AN FDST FLOW FAILURE RESULTED IN A TRAP TO 4.
3221 014750 022626      1$: CMP (SP)+,(SP)+
3222 014752 010237 001236      MOV R2,$TMP2
3223 014756 104076      2$: ERROR +76      ;ODD ADRES
3224 014760 000430      BR FFBDONE      ;BUT FDSTX IN ST 771
3225
3226      :REPORT RESULT INCORRECT:
3227 014762 012737 014676 001240      FFB15: MOV #FFBTP2,$TMP3
3228 014770 012737 014666 001242      MOV #FFBTP1,$TMP4
3229 014776 012737 014716 001244      MCV #FFBTF1,$TMP5
3230 015004 104077      1$: ERROR +77      ;BAD DATA X11*0 ST 312X
3231 015006 000415      BR FFBDONE
3232
3233      :REPORT R0 INCORRECT:
3234 015010 012737 014722 001240      FFB20: MOV #FFBTF1+4,$TMP3
3235 015016 010037 001242      MOV R0,$TMP4
3236 015022 104100      1$: ERROR +100      ;R0 BADX
3237 015024 000406      BR FFBDONE
3238
3239      :REPORT FPS INCORRECT:
3240 015026 010537 001240      FFB25: MOV R5,$TMP3
3241 015032 012737 000204 001244      MOV #204,$TMP5
3242 015040 104101      1$: ERROR +101      ;FPS X
3243
3244 015042      FFBDONE:
3244 015042 104412      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?).
3245      ;TEST TITLE:NEGF, ABSF AND TSTF SOURCE MODE 4 TEST
    
```

3246

```

.SBTTL TEST # 23 - SEE ABOVE COMMENT FOR TEST TITLE
*****
*TEST 23 SEE ABOVE COMMENT FOR TEST TITLE
*
*THIS IS A TEST THE NEGF, ABSF AND TSTF
*SOURCE FLOWS. THE ABSD INSTRUCTION
*IS USED TO TEST MODE 4
*
*****
    
```

```

015044 000004
3247
3248 015046
015046 104413
3249 015050 012700 015156
3250 015054 012701 015176
3251 015060 012702 000004
3252 015064 012021
3253 015066 077202
3254 015070 012700 000200
3255 015074 170100
3256 015076 012700 015206
3257 015102 012737 015116 001236
3258 015110 012737 015216 000004
3259
3260 015116 170640
3261
3262 015120 170205
3263 015122 012701 015176
3264 015126 012702 000004
3265 015132 005721
3266 015134 001046
3267 015136 077203
3268
3269 015140 020027 015176
3270 015144 001055
3271 015146 022705 000204
3272 015152 001061
3273 015154 000466
3274
3275
3276 015156 000177
3277 015160 117273
3278 015162 147576
3279 015164 177071
3280 015166 000000
3281 015170 000000
3282 015172 000000
3283 015174 000000
3284 015176 177777
3285 015200 177777
3286 015202 177777
3287 015204 177777
3288 015206 177777
3289 015210 177777
3290 015212 177777
3291 015214 177777
3292
    
```

```

TST23: SCOPE
GGB1:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
MOV #GGBTP1,R0 ;SET UP THE DATA BUFFER.
MOV #GGBBF0,R1
MOV #4,R2
1$: MCV (R0)+,(R1)+
SOB R2,1$
MOV #200,R0 ;SET FD.
LDFPS R0
MOV #GGBBF1,R0 ;SET UP THE OPERAND ADDRESS.
MOV #GGB2,$IMP2
MOV #GGB10,ERRVECT ;SET UP VECTOR 4 IN CASE OF AN ERROR.
GGB2: ABSD -(R0) ;TEST INSTRUCTION.
STFPS R5 ;GET FPS.
MOV #GGBBF0,R1 ;CHECK RESULT.
1$: MOV #4,R2
TST (R1)+
BNE GGB15 ;BRANCH IF INCORRECT.
SOB R2,1$
CMP R0,#GGBBF0 ;IS R0 CORRECT?
BNE GGB20 ;BRANCH IF INCORRECT.
CMP #204,R5 ;IS THE FPS CORRECT?
BNE GGB25 ;BRANCH IF INCORRECT.
BR GGBDONE
;THESE ARE TEST DATA TABLES AND DATA BUFFER.
GGBTP1: 177
117273
147576
177071
GGBTP2: 0
0
0
0
GGBBF0: -1
-1
-1
-1
GGBBF1: -1
-1
-1
-1
    
```

```

3293
3294 015216 011602
3295 015220 020227 015120
3296 015224 001405
3297 015226 020227 015122
3298 015232 001402
3299 015234 000137 047302
3300
3301 015240 022626
3302 015242 010237 001236
3303 015246 104102
3304 015250 000430
3305
3306
3307 015252 012737 015166 001240
3308 015260 012737 015156 001242
3309 015266 012737 015176 001244
3310 015274 104103
3311 015276 000415
3312
3313
3314 015300 012737 015176 001240
3315 015306 010037 001242
3316 015312 104104
3317 015314 000406
3318
3319
3320 015316 010537 001240
3321 015322 012737 000204 001244
3322 015330 104105
3323
3324 015332
      015332 104412
3325

;IF A TRAP TO 4 OCCURS COME HERE.
GGB10: MOV (SP),R2 ;SEE IF THE TRAP OCCURRED ON THE TEST INSTRUCTION.
      CMP R2,#GGB2+2
      BEQ 1$ ;BRANCH IF YES.
      CMP R2,#GGB2+4
      BEQ 1$ ;BRANCH IF YES.
      JMP CPSPUR ;OTHERWISE GO REPORT SPURIOUS TRAP TO 4.
;REPORT AN FDST FLOW FAILURE RESULTED IN A TRAP TO 4.
1$: CMP (SP)+,(SP)+
      MOV R2,$TMP2
2$: ERROR +102 ;ODD ADRES
      BR GGBDONE ;BUT FDSTX IN ST 771

;REPORT RESULT INCORRECT:
GGB15: MOV #GGBTP2,$TMP3
      MOV #GGBTP1,$TMP4
      MCV #GGBBF0,$TMP5
1$: ERROR +103 ;BAD DATA X11*0 ST 312X
      BR GGBDONE

;REPORT RO INCORRECT:
GGB20: MOV #GGBBF01,$TMP3
      MOV RO,$TMP4
1$: ERROR +104 ;RO BADX
      BR GGBDONE

;REPORT FPS INCORRECT:
GGB25: MOV R5,$TMP3
      MOV #204,$TMP5
1$: ERROR +105 ;FPS X

GGBDONE:
      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 TITLE:NEGF, ABSF AND TSTF SOURCE MODE 3 TEST
  
```

3326

```
.SBTTL TEST # 24 - SEE ABOVE COMMENT FOR TEST TITLE
:*****
:*TEST 24 SEE ABOVE COMMENT FOR TEST TITLE
:*
:*THIS IS A TEST THE NEGF, ABSF AND TSTF
:*SOURCE FLOWS. THE ABSD INSTRUCTION
:*IS USED TO TEST MODE 3
:*
:*****
```

```
015334 000004
3327
3328 015336
015336 104413
3329 015340 012700 015446
3330 015344 012701 015476
3331 015350 012702 000010
3332 015354 012021
3333 015356 077202
3334 015360 012700 000200
3335 015364 170100
3336 015366 012700 015506
3337 015372 012737 015406 001236
3338 015400 012737 015516 000004
3339
3340 015406 170630
3341
3342 015410 170205
3343 015412 012701 015476
3344 015416 012702 000004
3345 015422 005721
3346 015424 001052
3347 015426 077203
3348 015430 020027 015510
3349 015434 001061
3350 015436 022705 000204
3351 015442 001065
3352 015444 000472
3353
3354
3355 015446 000177
3356 015450 147576
3357 015452 177071
3358 015454 107576 015476 177777
3359 015466 000000 000000 000000
3360 015476 177777
3361 015500 177777
3362 015502 177777
3363 015504 177777
3364 015506 177777
3365 015510 177777
3366 015512 177777
3367 015514 177777
3368
3369
3370 015516 011602
3371 015520 020227 015410
3372 015524 001405
```

```
TST24: SCOPE
HHB1:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
MOV #HHBTP1,R0 ;SET UP THE DATA BUFFER.
MOV #HHBBF0,R1
MOV #10,R2
1$: MCV (R0)+,(R1)+
SOB R2,1$
MOV #200,R0 ;SET FD.
LDFPS R0
MCV #HHBBF1,R0 ;SET UP THE OPERAND ADDRESS.
MOV #HHB2,$IMP2
MOV #HHB10,ERRVECT ;SET UP VECTOR 4 IN CASE OF AN ERROR.
HHB2: ABSD @ (R0)+ ;TEST INSTRUCTION.
STFPS R5 ;GET FPS.
MOV #HHBBF0,R1 ;CHECK RESULT.
MOV #4,R2
1$: TST (R1)+
BNE HHB15 ;BRANCH IF INCORRECT.
SOB R2,1$
CMP R0,#HHBBF1+2 ;IS R0 CORRECT?
BNE HHB20 ;BRANCH IF INCORRECT.
MP #204,R5 ;IS THE FPS CORRECT?
BNE HHB25 ;BRANCH IF INCORRECT.
BR HHBDONE
;THESE ARE TEST DATA TABLES AND DATA BUFFER.
HHBTP1: 177
147576
177071
107576,HHBBF0,-1,-1,-1
HHBTP2: 0,0,0,0
HHBBF0: -1
-1
-1
-1
HHBBF1: -1
-1
-1
-1
;IF A TRAP TO 4 OCCURS COME HERE.
HHB10: MOV (SP),R2 ;SEE IF THE TRAP OCCURRED ON THE TEST INSTRUCTION.
CMP R2,#HHB2+2
BEQ 1$ ;BRANCH IF YES.
```

```

3373 015526 020227 015412          CMP      R2,#HHB2+4
3374 015532 001402          BEQ      1$          ;BRANCH IF YES.
3375 015534 000137 047302          JMP      CPSPUR ;OTHERWISE GO REPORT SPURIOUS TRAP TO 4.
3376          ;REPORT AN FDST FLOW FAILURE RESULTED IN A TRAP TO 4.
3377 015540 022626          1$:      CMP      (SP)+,(SP)+
3378 015542 010237 001236          MOV      R2,$TMP2
3379 015546 104106          2$:      ERROR   +106          ;ODD ADRES
3380 015550 000430          BR       HHBDONE          ;BUT FDSTX IN ST 771
3381
3382          ;REPORT RESULT INCORRECT:
3383 015552 012737 015466 001240      HHB15:  MOV      #HHBTP2,$TMP3
3384 015560 012737 015446 001242          MOV      #HHBTP1,$TMP4
3385 015566 012737 015476 001244          MOV      #HHBBF0,$TMP5
3386 015574 104110          1$:      ERROR   +110          ;BAD DATA X11*0 ST 3127
3387 015576 000415          BR       HHBDONE
3388
3389          ;REPORT R0 INCORRECT:
3390 015600 012737 015510 001240      HHB20:  MOV      #HHBBF1+2,$TMP3
3391 015606 010037 001242          MOV      R0,$TMP4
3392 015612 104111          1$:      ERROR   +111          ;R0 INCORRECT.
3393 015614 000406          BR       HHBDONE
3394          ;REPORT FPS INCORRECT:
3395 015616 010537 001240      HHB25:  MOV      R5,$TMP3
3396 015622 012737 000204 001244          MOV      #204,$TMP5
3397 015630 104112          1$:      ERROR   +112          ;FPSX
3398
3399          HHBDONE:
          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?).
3400          ;TEST TITLE:NEGF, ABSF AND TSTF SOURCE MODE 5 TEST
    
```

3401

```
.SBTTL TEST # 25 - SEE ABOVE COMMENT FOR TEST TITLE
:*****
:*TEST 25 SEE ABOVE COMMENT FOR TEST TITLE
:*
:*THIS IS A TEST THE NEGF, ABSF AND TSTF
:*SOURCE FLOWS. THE NEGD INSTRUCTION
:*IS USED TO TEST MODE 5
:*
:*****
TST25: SCOPE
```

```
015634 000004
3402
3403 015636
015636 104413
3404 015640 012700 015746
3405 015644 012701 015776
3406 015650 012702 000010
3407 015654 012021
3408 015656 077202
3409 015660 012700 000200
3410 015664 170100
3411 015666 012700 016010
3412 015672 012737 015706 001236
3413 015700 012737 016016 000004
3414
3415 015706 170750
3416
3417 015710 170205
3418 015712 012701 015776
3419 015716 012702 000004
3420 015722 005721
3421 015724 001052
3422 015726 077203
3423 015730 020027 016006
3424 015734 001061
3425 015736 022705 000204
3426 015742 001065
3427 015744 000472
3428
3429
3430 015746 000176
3431 015750 177074
3432 015752 127374
3433 015754 157677 015776 177777
3434 015766 000000
3435 015770 000000
3436 015772 000000
3437 015774 000000
3438 015776 177777
3439 016000 177777
3440 016002 177777
3441 016004 177777
3442 016006 177777
3443 016010 177777
3444 016012 177777
3445 016014 177777
3446
3447
```

```
IIB1:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
MOV #IIBTP1,R0 ;SET UP THE DATA BUFFER.
MOV #IIBBF0,R1
MOV #10,R2
1$: MCV (R0)+,(R1)+
SOB R2,1$
MOV #200,R0 ;SET FD.
LDFPS R0
MOV #IIBBF1+2,R0 ;SET UP THE OPERAND ADDRESS.
MOV #IIB2,$TMP2
MOV #IIB10,ERRVECT ;SET UP VECTOR 4 IN CASE OF AN ERROR.
IIB2: NEGD @-(R0) ;TEST INSTRUCTION.
STFPS R5 ;GET FPS.
MOV #IIBBF0,R1 ;CHECK RESULT.
MOV #4,R2
1$: TST (R1)+
BNE IIB15 ;BRANCH IF INCORRECT.
SOB R2,1$
CMP R0,#IIBBF1 ;IS R0 CORRECT?
BNE IIB20 ;BRANCH IF INCORRECT.
CMP #204,R5 ;IS THE FPS CORRECT?
BNE IIB25 ;BRANCH IF INCORRECT.
BR IIBDONE

;THESE ARE TEST DATA TABLES AND DATA BUFFER.
IIBTP1: 176
177074
127374
157677,IIBBF0,-1,-1,-1
IIBTP2: 0
0
0
0
IIBBF0: -1
-1
-1
-1
IIBBF1: -1
-1
-1
-1

;IF A TRAP TO 4 OCCURS COME HERE.
```


3478

```
.SBTTL TEST # 26 - NEGF, ABSF AND TSTF SRC MODE 6 TEST
:*****
:*TEST 26      NEGF, ABSF AND TSTF SRC MODE 6 TEST
:*
:*THIS IS A TEST THE NEGF, ABSF AND TSTF
:*SOURCE FLOWS. THE ABSD INSTRUCTION
:*IS USED TO TEST MODE 6
:*
:*****
```

```
016134 000004
3479
3480 016136
    016136 104413
3481 016140 012700 016250
3482 016144 012701 016272
3483 016150 012702 000004
3484 016154 012021
3485 016156 077202
3486 016160 012700 000200
3487 016164 170100
3488 016166 012700 016263
3489 016172 012767 016206 163036
3490 016200 012767 016312 161576
3491
3492 016206 170660 000007
3493
3494 016212 170205
3495 016214 012701 016272
3496 016220 012702 000004
3497 016224 005721
3498 016226 001047
3499 016230 077203
3500 016232 020027 016263
3501 016236 001043
3502 016240 022705 000204
3503 016244 001053
3504 016246 000467
3505
3506
3507 016250 000177
3508 016252 161524
3509 016254 131273
3510 016256 107174 000000
3511 016262 000000
3512 016264 000000
3513 016266 000000
3514 016270 000000
3515 016272 177777
3516 016274 177777
3517 016276 177777
3518 016300 177777
3519 016302 177777
3520 016304 177777
3521 016306 177777
3522 016310 177777
3523
3524
```

```
TST26: SCOPE
        .DSABL  AMA          ;DISABLE MODE 6 TO MODE 3 CONVERSIONS
JJB1:   LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
        MOV  #JJBTP1,R0    ;SET UP THE DATA BUFFER.
        MOV  #JJBFB0,R1
        MOV  #4,R2
1$:     MCV  (R0)+,(R1)+
        SOB  R2,1$
        MOV  #200,R0      ;SET FD.
        LDFPS R0
        MOV  #JJBFB0-7,R0 ;SET UP THE OPERAND ADDRESS.
        MOV  #JJB2,$TMP2
        MOV  #JJB10,ERRVECT ;SET UP VECTOR 4 IN CASE OF AN ERROR.
JJB2:   ABSD  7(R0)      ;TEST INSTRUCTION.
        STFPS R5         ;GET FPS.
        MOV  #JJBFB0,R1   ;CHECK RESULT.
        MOV  #4,R2
1$:     TST  (R1)+
        BNE  JJB15        ;BRANCH IF INCORRECT.
        SOB  R2,1$
        CMP  R0,#JJBFB0-7 ;IS R0 CORRECT?
        BNE  JJB15        ;BRANCH IF INCORRECT.
        CMP  #204,R5      ;IS THE FPS CORRECT?
        BNE  JJB20        ;BRANCH IF INCORRECT.
        BR   JJB DONE

;THESE ARE TEST DATA TABLES AND DATA BUFFER.
JJBTP1: 177
        161524
        131273
        107174,
JJBTP2: 0
        0
        0
        0
JJBFB0: -1
        -1
        -1
        -1
JJBFB1: -1
        -1
        -1
        -1

;IF A TRAP TO 4 OCCURS COME HERE.
```

```

3525 016312 011602          JJB10: MOV    (SP),R2      ;SEE IF THE TRAP OCCURRED ON THE TEST INSTRUCTION.
3526 016314 020227 016210      CMP    R2,#JJB2+2
3527 016320 001405          BEQ    1$              ;BRANCH IF YES.
3528 016322 020227 016212      CMP    R2,#JJB2+4
3529 016326 001402          BEQ    1$              ;BRANCH IF YES.
3530 016330 000167 030746      JMP    CPSPUR ;OTHERWISE GO REPORT SPURIOUS TRAP TO 4.
3531          ;REPORT AN FDST FLOW FAILURE RESULTED IN A TRAP TO 4.
3532 016334 022626          1$:  CMP    (SP)+,(SP)+
3533 016336 010267 162674      MOV    R2,$TMP2
3534 016342 104117          2$:  ERROR  +117        ;ODD ADRES
3535 016344 000430          BR     JJBDONE        ;BUT FDSTX IN ST 771
3536
3537          ;REPORT RESULT INCORRECT:
3538 016346 012767 016262 162664 JJB15: MOV    #JJBTP2,$TMP3
3539 016354 012767 016250 162660      MOV    #JJBTP1,$TMP4
3540 016362 012767 016272 162654      MOV    #JJBBO,$TMP5
3541 015370 104120          1$:  ERROR  +120        ;BAD DATA X11*0 ST 3127
3542 016372 000415          BR     JJBDONE
3543
3544          ;REPORT RO INCORRECT:
3545 016374 012767 016263 162636 JJB20: MOV    #JJBBO-7,$TMP3
3546 016402 010067 162634      MOV    R0,$TMP4
3547 016406 104124          1$:  ERROR  +124        ;RO BADX
3548 016410 000406          BR     JJBDONE
3549          ;REPORT FPS INCORRECT:
3550 016412 010567 162622          JJB25: MOV    R5,$TMP3
3551 016416 012767 000204 162620      MOV    #204,$TMP5
3552 016424 104122          1$:  ERROR  +122        ;FPSX
3553 016426          JJBDONE:
      016426 104412          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?).
                          ;REENABLE MODE 6 TO MODE 3 CONVERSIONS
3554          .ENABL  AMA
    
```

3555

```
.SBTTL TFST # 27 - NEGF, ABSF AND TSTF SRC MODE 7 TEST
*****
*TEST 2          NEGF, ABSF AND TSTF SRC MODE 7 TEST
*
*THIS IS , EST THE NEGF, ABSF AND TSTF
*SOURCE FLOWS. THE ABSD INSTRUCTION
*IS USED TO TEST MODE 6
*
*****
TST27: SCOPE
```

```
016430 000004
3556
3557 016432
016432 104413
3558 016434 012700 016544
3559 016440 012701 016574
3560 016444 012702 000010
3561 016450 012021
3562 016452 077202
3563 016454 012700 000200
3564 016460 170100
3565 016462 012700 016575
3566 016466 012737 016502 001236
3567 016474 012737 016614 000004
3568
3569 016502 170770 000007
3570
3571 016506 170205
3572 016510 012701 016574
3573 016514 012702 000004
3574 016520 005721
3575 016522 001052
3576 016524 077203
3577 016526 020027 016575
3578 016532 001061
3579 016534 022705 000204
3580 016540 001056
3581 016542 000472
3582
3583
3584 016544 000177
3585 016546 167574
3586 016550 137271
3587 016552 107675 016574 177777
3588 016564 000000
3589 016566 000000
3590 016570 000000
3591 016572 000000
3592 016574 177777
3593 016576 177777
3594 016600 177777
3595 016602 177777
3596 016604 177777
3597 016606 177777
3598 016610 177777
3599 016612 177777
3600
3601
```

```
KKB1:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
MOV #KKBTP1,R0 ;SET UP THE DATA BUFFER.
MOV #KKBBF0,R1
MOV #10,R2
1$: MOV (R0)+,(R1)+
SOB R2,1$
MOV #200,R0 ;SET FD.
LDFPS R0
MOV #KKBBF1-7,R0 ;SET UP THE OPERAND ADDRESS.
MOV #KKB2,$TMP2
MOV #KKB10,ERRVECT ;SET UP VECTOR 4 IN CASE OF AN ERROR.
KKB2: NEG D @7(R0) ;TEST INSTRUCTION.
STFPS R5 ;GET FPS.
MOV #KKBBF0,R1 ;CHECK RESULT.
MOV #4,R2
1$: TST (R1)+
BNE KKB15 ;BRANCH IF INCORRECT.
SOB R2,1$
CMP R0,#KKBBF1-7 ;IS R0 CORRECT?
BNE KKB20 ;BRANCH IF INCORRECT.
CMP #204,R5 ;IS THE FPS CORRECT?
BNE KKB20 ;BRANCH IF INCORRECT.
BR KKB DONE
;THESE ARE TEST DATA TABLES AND DATA BUFFER.
KKBTP1: 177
167574
137271
107675,KKB BF0,-1,-1,-1
KKBTP2: 0
0
0
0
KKB BF0: -1
-1
-1
-1
KKB BF1: -1
-1
-1
-1
;IF A TRAP TO 4 OCCURS COME HERE.
```

```

3602 016614 011602          KKB10: MOV      (SP),R2          ;SEE IF THE TRAP OCCURRED ON THE TEST INSTRUCTION.
3603 016616 020227 016504      CMP      R2,#KKB2+2
3604 016622 001405          BEQ      1$          ;BRANCH IF YES.
3605 016624 020227 016506      CMP      R2,#KKB2+4
3606 016630 001402          BEQ      1$          ;BRANCH IF YES.
3607 016632 000137 047302      JMP      CPSPUR ;OTHERWISE GO REPORT SPURIOUS TRAP TO 4.
3608          ;REPORT AN FDST FLOW FAILURE RESULTED IN A TRAP TO 4.
3609 016636 022626      1$:      CMP      (SP)+,(SP)+
3610 016640 010237 001236      MOV      R2,$TMP2
3611 016644 104123          2$:      ERROR   +123          ;ODD ADRES
3612 016646 000430          BR       KKBDONE        ;BUT FDSTX IN ST 771
3613
3614          ;REPORT RESULT INCORRECT:
3615 016650 012737 016564 001240 KKB15: MOV      #KKBTP2,$TMP3
3616 016656 012737 016544 001242      MOV      #KKBTP1,$TMP4
3617 016664 012737 016574 001244      MOV      #KKBBF0,$TMP5
3618 016672 104124          1$:      ERROR   +124          ;BAD DATA X11*0 ST 3127
3619 016674 000415          BR       KKBDONE
3620
3621          ;REPORT R0 INCORRECT:
3622 016676 012737 016575 001240 KKB20: MOV      #KKBBF1-7,$TMP3
3623 016704 010037 001242      MOV      R0,$TMP4
3624 016710 104125          1$:      ERROR   +125          ;R0 BADX
3625 016712 000406          BR       KKBDONE
3626          ;REPORT FPS INCORRECT:
3627 016714 010537 001240 KKB25: MOV      R5,$TMP3
3628 016720 012737 000204 001244      MOV      #204,$TMP5
3629 016726 104126          1$:      ERROR   +126          ;FPSX
3630
3631          KKBDONE:
          016730 104412      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?).
    
```

3632

```
.SBTTL TEST # 30 - NEGF, ABSF AND TSTF SRC MODE 6, GR7
*****
*TEST 30      NEGF, ABSF AND TSTF SRC MODE 6, GR7
*
*THIS IS A TEST THE NEGF, ABSF AND TSTF
*SOURCE FLOWS. THE NEGD INSTRUCTION
*IS USED TO TEST MODE 6
*
*****
```

```
3633 016732 000004
3634 016734
3635 016736 104413 017034
3636 016742 012701 017054
3637 016746 012702 000004
3638 016752 012021
3639 016754 077202
3640 016756 012700 00020C
3641 016762 170100
3642 016764 012767 017000 162244
3643 016772 012767 017074 161C04
3644
3645 017000 170767 000050
3646
3647 017004 170205
3648 017006 012701 017054
3649 017012 012702 000004
3650 017016 005721
3651 017020 001043
3652 017022 077203
3653 017024 022705 000204
3654 017030 001052
3655 017032 000457
3656
3657
3658 017034 000127
3659 017036 137475
3660 017040 147372
3661 017042 117057
3662 017044 000000
3663 017046 000000
3664 017050 000000
3665 017052 000000
3666 017054 177777
3667 017056 177777
3668 017060 177777
3669 017062 177777
3670 017064 177777
3671 017066 177777
3672 017070 177777
3673 017072 177777
3674
3675
3676 017074 011602 017002
3677 017076 020227
3678 017102 001405
```

```
TST30: SCOPE
        .DSABL  AMA          ;DISABLE MODE 6 TO MODE 3 CONVERSIONS
LLB1:   LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
        MOV  #LLBTP1,R0    ;SET UP THE DATA BUFFER.
        MOV  #LLBBF0,R1
        MOV  #4,R2
1$:     MGV  (R0)+,(R1)+
        SOB  R2,1$
        MOV  #200,R0      ;SET FD.
        LDFPS R0
        MOV  #LLB2,$TMP2
        MOV  #LLB10,ERRVECT ;SET UP VECTOR 4 IN CASE OF AN ERROR.
LLB2:   NEGD  LLBBF0      ;TEST INSTRUCTION.
        STFPS R5          ;GET FPS.
        MOV  #LLBBF0,R1    ;CHECK RESULT.
        MOV  #4,R2
1$:     TST  (R1)+
        BNE  LLB15        ;BRANCH IF INCORRECT.
        SOB  R2,1$
        CMP  #204,R5      ;IS THE FPS CORRECT?
        BNE  LLB25        ;BRANCH IF INCORRECT.
        BR   LLBDONE

;THESE ARE TEST DATA TABLES AND DATA BUFFER.
LLBTP1: 127
        137475
        147372
        117057
LLBTP2: 0
        0
        0
        0
LLBBF0: -1
        -1
        -1
        -1
        -1
LLBBF1: -1
        -1
        -1
        -1

;IF A TRAP TO 4 OCCURS COME HERE.
LLB10:  MOV  (SP),R2      ;SEE IF THE TRAP OCCURRED ON THE TEST INSTRUCTION.
        CMP  R2,#LLB2+2
        BEQ  1$         ;BRANCH IF YES.
```

```

3679 017104 020227 017004          CMP      R2,#LLB2+4
3680 017110 001402          BEQ      1$          ;BRANCH IF YES.
3681 017112 000167 030164          JMP      CPSPUR ;OTHERWISE GO REPORT SPURIOUS TRAP TO 4.
3682          ;REPORT AN FDST FLOW FAILURE RESULTED IN A TRAP TO 4.
3683 017116 022626          1$:      CMP      (SP)+,(SP)+
3684 017120 010267 162112          MOV      R2,$TMP2
3685 017124 104127          2$:      ERROR   +127          ;ODD ADRES
3686 017126 000421          BR       LLBDONE          ;BUT FDSTX IN ST 771
3687
3688          ;REPORT RESULT INCORRECT:
3689 017130 012767 017044 162102 LLB15:   MOV      #LLBTP2,$TMP3
3690 017136 012767 017034 162076          MOV      #LLBTP1,$TMP4
3691 017144 012767 017054 162072          MOV      #LLBBFO,$TMP5
3692 017152 104130          1$:      ERROR   +130          ;BAD DATA x11*0 ST 3127
3693 017154 000406          BR       LLBDONE
3694          ;REPORT FPS INCORRECT:
3695 017156 010567 162056          LLB25:   MOV      R5,$TMP3
3696 017162 012767 000204 162054          MOV      #204,$TMP5
3697 017170 104131          1$:      ERROR   +31          ;FPSX
3698
3699 017172          LLBDONE:
          017172 104412          RSE*UP          ;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?).
          ;REENABLE MODE 6 TO MODE 3 CONVERSIONS

3700          .ENABL  AMA
  
```

3701

```
.SBTTL TEST # 31 - NEGF, ABSF AND TSTF SRC MODE 7, GR7
:*****
:*TEST 31      NEGF, ABSF AND TSTF SRC MODE 7, GR7
:*
:*THIS IS A TEST THE NEGF, ABSF AND TSTF
:*SOURCE FLOWS. THE ABSD INSTRUCTION
:*IS USED TO TEST MODE 7
:*
:*****
TST31: SCOPE
```

```
017174 000004
3702
3703 017176
017176 104413
3704 017200 012700 017276
3705 017204 012701 017326
3706 017210 012702 000010
3707 017214 012021
3708 017216 077202
3709 017220 012700 000200
3710 017224 170100
3711 017226 012737 017242 001230
3712 017234 012737 017346 000004
3713
3714 017242 170677 000070
3715
3716 017246 170205
3717 017250 012701 017326
3718 017254 012702 000004
3719 017260 005721
3720 017262 001047
3721 017264 077203
3722 017266 022705 000204
3723 017272 001056
3724 017274 000463
3725
3726
3727 017276 000137
3728 017300 045607
3729 017302 101230
3730 017304 045607 017326 177777
3731 017316 000000
3732 017320 000000
3733 017322 000000
3734 017324 000000
3735 017326 177777
3736 017330 177777
3737 017332 177777
3738 017334 177777
3739 017336 177777
3740 017340 177777
3741 017342 177777
3742 017344 177777
3743
3744
3745 017346 011602
3746 017350 020227 017244
3747 017354 001405
```

```
MMB1:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
MOV #MMBTP1,R0 ;SET UP THE DATA BUFFER.
MOV #MMBBF0,R1
MOV #10,R2
1$: MOV (R0)+,(R1)+
SOB R2,1$
MOV #200,R0 ;SET FD.
LDFPS R0
MOV #MMB2,$TMP2
MOV #MMB10,ERRVECT ;SET UP VECTOR 4 IN CASE OF AN ERROR.
MMB2: ABSD @MMBBF1 ;TEST INSTRUCTION.
STFPS R5 ;GET FPS.
MOV #MMBBF0,R1 ;CHECK RESULT.
MOV #4,R2
1$: TST (R1)+
BNE MMB15 ;BRANCH IF INCORRECT.
SOB R2,1$
CMP #204,R5 ;IS THE FPS CORRECT?
BNE MMB25 ;BRANCH IF INCORRECT.
BR MMBDONE

:THESE ARE TEST DATA TABLES AND DATA BUFFER.
MMBTP1: 137
045607
101230
45607,MMBBF0,-1,-1,-1
MMBTP2: 0
0
0
0
MMBBFC: -1
-1
-1
-1
MMBBF1: -1
-1
-1
-1

:IF A TRAP TO 4 OCCURS COME HERE.
MMB10: MOV (SP),R2 ;SEE IF THE TRAP OCCURRED ON THE TEST INSTRUCTION.
CMP R2,#MMB2+2
BEQ 1$ ;BRANCH IF YES.
```


3775

```

.SBTTL TEST # 32 - SPECIAL DEST, MODE 0, TEST
*****
:TEST 32 SPECIAL DEST, MODE 0, TEST
:
:THIS IS A TEST OF THE NEGF ABSF AND TSTF DESTINATION FLOWS
:MODE 0 USING THE NEGD INSTR.
:
*****
    
```

```

017446 000004
3776
3777 017450
017450 104413
3778 017452 012700 000200
3779 017456 170100
3780 017460 012700 017546
3781 017464 172410
3782 017466 012737 017474 001236
3783
3784 017474 170700
3785
3786 017476 170205
3787 017500 012700 000200
3788 017504 170100
3789 017506 012700 017566
3790 017512 174010
3791 017514 012700 017566
3792 017520 012701 017556
3793 017524 012702 000004
3794 017530 022021
3795 017532 001021
3796 017534 077203
3797 017536 022705 000210
3798 017542 001033
3799 017544 000440
3800
3801
3802 017546 013572
3803 017550 046013
3804 017552 057246
3805 017554 013570
3806 017556 113572
3807 017560 046013
3808 017562 057246
3809 017564 013570
3810 017566 000000
3811 017570 000000
3812 017572 000000
3813 017574 000000
3814
3815
3816 017576 012737 017566 001241
3817 017604 012737 017556 001242
3818 017612 023737 017546 017566
3819 017620 001002
3820 017622 104135
3821 017624 000410
3822
    
```

```

TST32: SCOPE
NNB1:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
MOV #200,R0 ;SET FD.
LDFPS R0
MOV #NNBTP1,R0 ;SET UP ACO.
LDD (R0),ACO
MVC #NNB2,$TMP2
NNB2: NEGD ACO ;TEST INSTRUCTION.
STFPS R5 ;GET FPS.
MOV #200,R0 ;SET FD.
LDFPS R0
MOV #NNBBFC,R0 ;GET THE RESULT.
STD ACO,(R0)
MOV #NNBBFO,R0 ;IS THE RESULT CORRECT?
MOV #NNBTP2,R1
MOV #4,R2
1$: CMP (R0)+,(R1)+
BNE NNB10 ;BRANCH IF INCORRECT.
SOB R2,1$
CMP #210,R5 ;IS THE FPS CORRECT?
BNE NNB15 ;BRANCH IF INCORRECT.
BR NNB DONE
;THESE ARE DATA TABLES AND A DATA BUFFER.
NNBTP1: 013572
46013
57246
013570
NNBTP2: 113572
46013
57246
013570
NNBBFO: 0
0
0
0
;REPORT RESULT INCORRECT:
NNB10: MOV #NNBBFO,$TMP3
MOV #NNBTP2,$TMP4
CMP NNBTP1,NNBBFO
BNE NNB11
1$: ERROR +135 ;E10*200x ST 336
BR NNB DONE
    
```

```
3823 ;REPORT RESULT INCORRECT:
3824 017626 NNB11:
3825 017626 104136 1$: ERROR +136 ;BAD DATA NEGF
3826 017630 000406 BR NNBDONE
3827
3828 ;REPORT FPS INCORRECT:
3829 017632 010537 001242 NNB15: MOV R5,$TMP4
3830 017636 012737 000210 001240 MOV #210,$TMP3
3831 017644 104137 1$: ERROR +137 ;FPSX
3832
3833 017646 NNBDONE: ;GO INITIALIZE THE FPS AND STACK; AND
017646 104412 RSETUP ;SEE IF THE USER HAS EXPRESSED
;THE DESIRE TO CHANGE THE SOFTWARE
;VIRTUAL CONSOLE SWITCH REGISTER (HAS
;THE USER TYPED CONTROL G?).
```

3834

```
.SBTTL TEST # 33 - SPECIAL DEST, MODE 1, TEST
*****
*TEST 33 SPECIAL DEST, MODE 1, TEST
*
*THIS IS A TEST OF THE NEGF ABSF AND TSTF DESTINATION FLOWS
*MODE 1 USING THE NEGD INSTR.
*
*****
```

```
017650 000004
3835
3836 017652
017652 104413
3837 017654 012701 017764
3838 017660 012700 017774
3839 017664 012702 000004
3840 017670 012021
3841 017672 077202
3842 017674 012700 017764
3843 017700 042710 100000
3844 017704 012737 017720 001236
3845 017712 012701 000200
3846 017716 170101
3847
3848 017720 170710
3849 017722 170205
3850 017724 012701 017764
3851 017730 012702 017774
3852 017734 012703 000004
3853 017740 022122
3854 017742 001020
3855 017744 077303
3856 017746 022700 017764
3857 017752 001024
3858 017754 022705 000210
3859 017760 001030
3860 017762 000435
3861
3862
3863 017764 023245
3864 017766 026720
3865 017770 122324
3866 017772 052672
3867 017774 123245
3868 017776 026720
3869 020000 122324
3870 020002 052672
3871
3872
3873 020004 012737 017764 001240
3874 020012 012737 017774 001242
3875 020020 104140
3876 020022 000415
3877
3878
3879 020024 012737 017764 001240
3880 020032 010037 001242
3881 020036 104141
```

```
TST33: SCOPE
00B1:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
MOV #00BTP1,R1 ;SET UP THE DATA BUFFER.
MOV #00BTP2,R0
MOV #4,R2
1$: MOV (R0)+,(R1)+
SOB R2,1$
MOV #00BTP1,R0
BIC #100000,(R0) ;MAKE OPERAND POSITIVE.
MOV #00B2,$TMP2
MOV #200,R1 ;SET FD.
LDFPS R1
00B2: NEGD (R0) ;TEST INSTRUCTION.
STFPS R5 ;GET FPS.
MOV #00BTP1,R1 ;IS THE RESULT CORRECT.
MOV #00BTP2,R2
MOV #4,R3
1$: CMP (R1)+,(R2)+
BNE 00B10 ;BRANCH IF INCORRECT.
SOB R3,1$
CMP #00BTP1,R0 ;IS R0 CORRECT.
BNE 00B15 ;BRANCH IF INCORRECT.
CMP #210,R5 ;IS THE FPS CORRECT?
BNE 00B20 ;BRANCH IF INCORRECT.
BR 00BDONE
;THESE ARE DATA TABLES AND A DATA BUFFER.
00BTP1: 023245
26720
122324
52672
00BTP2: 123245
26720
122324
52672
;REPORT RESULT INCORRECT:
00B10: MOV #00BTP1,$TMP3
MOV #00BTP2,$TMP4
1$: ERROR +140 ;BAD DATA
BR 00BDONE
;REPORT R0 INCORRECT:
00B15: MOV #00BTP1,$TMP3
MOV R0,$TMP4
1$: ERROR +141 ;SPEC DESTX
```

3882 020040 000406 BR OOB DONE ;ROX
3883
3884 :REPORT FPS INCORRECT:
3885 020042 012737 000210 001240 OOB20: MOV #210,\$TMP3
3886 020050 010537 001242 MOV R5,\$TMP4
3887 020054 104142 1\$: ERROR +142
3888
3889 020056 OOB DONE:
020056 104412 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?).

3890

```
.SBTTL TEST # 34 - SPECIAL DEST, MODE 2, TEST
:*****
:*TEST 34 SPECIAL DEST, MODE 2, TEST
:*
:*THIS IS A TEST OF THE NEGF ABSF AND TSTF DESTINATION FLOWS
:*MODE 2 USING THE NEGD INSTR.
:*
:*****
```

```
3891 020060 000004
020062
020062 104413
3892
3893 020064 012701 020174
3894 020070 012700 020204
3895 020074 012702 000004
3896 020100 012021
3897 020102 077202
3898 020104 012700 020174
3899 020110 042710 100000
3900 020114 012737 020130 001236
3901 020122 012701 000200
3902 020126 170101
```

```
TST34: SCOPE
PPB1: .LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
MOV #PPBTP1,R1 ;SET UP THE DATA BUFFER.
MOV #PPBTP2,R0
MOV #4,R2
1$: MOV (R0)+,(R1)+
SOB R2,1$
MOV #PPBTP1,R0
BIC #'00000,(R0) ;MAKE OPERAND POSITIVE.
MOV #PPB2,$TMP2
MOV #200,R1 ;SET FD.
LDFPS R1
```

```
3903
3904 020130 170720
3905
3906 020132 170205
3907 020134 012701 020174
3908 020140 012702 020204
3909 020144 012703 000004
3910 020150 022122
3911 020152 001020
3912 020154 077303
3913 020156 022700 020204
3914 020162 001024
3915 020164 022705 000210
3916 020170 001030
3917 020172 000435
```

```
PPB2: NEGD (R0)+ ;TEST INSTRUCTION.
STFPS R5 ;GET FPS.
MOV #PPBTP1,R1 ;IS THE RESULT CORRECT.
MOV #PPBTP2,R2
MOV #4,R3
1$: CMP (R1)+,(R2)+
BNE PPB10 ;BRANCH IF INCORRECT.
SOB R3,1$
CMP #PPBTP1+10,R0 ;IS R0 CORRECT.
BNE PPB15 ;BRANCH IF INCORRECT.
CMP #210,R5 ;IS THE FPS CORRECT?
BNE PPB20 ;BRANCH IF INCORRECT.
BR PPBDONE
```

```
3918
3919
3920 020174 023245
3921 020176 026720
3922 020200 122324
3923 020202 052672
3924 020204 123245
3925 020206 026720
3926 020210 122324
3927 020212 052672
```

```
;THESE ARE DATA TABLES AND A DATA BUFFER.
PPBTP1: 023245
26720
122324
52672
PPBTP2: 123245
26720
122324
52672
```

```
3928
3929
3930 020214 012737 020174 001240
3931 020222 012737 020204 001242
3932 020230 104143
3933 020232 000415
```

```
;REPORT RESULT INCORRECT:
PPB10: MOV #PPBTP1,$TMP3
MOV #PPBTP2,$TMP4
1$: ERROR +143 ;BAD DATA
BR PPBDONE
```

```
3934
3935
3936 020234 012737 020204 001240
3937 020242 010037 001242
```

```
;REPORT R0 INCORRECT:
PPB15: MOV #PPBTP1+10,$TMP3
MOV R0,$TMP4
```


3947

```
.SBTTL TEST # 35 - SPECIAL DEST, MODE 4, TEST
:*****
:*TEST 35 SPECIAL DEST, MODE 4, TEST
:*
:*THIS IS A TEST OF THE NEGF ABSF AND TSTF DESTINATION FLOWS
:*MODE 4 USING THE NEGD INSTR.
:*
:*****
```

```
020270 000004
3948 020272
020272 104413
3949 020274 012701 020406
3950 020300 012700 020426
3951 020304 012702 000004
3952 020310 012021
3953 020312 077202
3954 020314 012700 020416
3955 020320 042760 100000 177770
3956 020326 012737 020342 001236
3957 020334 012701 000200
3958 020340 170101
3959
3960 020342 170740
3961
3962 020344 170205
3963 020346 012701 020406
3964 020352 012702 020426
3965 020356 012703 000004
3966 020362 022122
3967 020364 001024
3968 020366 077303
3969 020370 022700 020406
3970 020374 001030
3971 020376 022705 000210
3972 020402 001034
3973 020404 000441
3974
3975
3976 020406 023245
3977 020410 026720
3978 020412 122324
3979 020414 052672
3980 020416 177777 177777 177777
3981 020426 123245
3982 020430 026720
3983 020432 122324
3984 020434 052672
3985
3986
3987 020436 012737 020406 001240
3988 020444 012737 020426 001242
3989 020452 104146
3990 020454 000415
3991
3992
3993 020456 012737 020406 001240
3994 020464 010037 001242
```

```
TST35: SCOPE
QQB1:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
MOV #QQBTP1,R1 ;SET UP THE DATA BUFFER.
MOV #QQBTP2,R0
MOV #4,R2
1$: MOV (R0)+,(R1)+
SOB R2,1$
MOV #QQBTP1+10,R0
BIC #100000,-10(R0) ;MAKE OPERAND POSITIVE.
MOV #QQB2,$TMP2
MOV #200,R1 ;SET FD.
LDFPS R1
QQB2: NEGD -(R0) ;TEST INSTRUCTION.
STFPS R5 ;GET FPS.
MOV #QQBTP1,R1 ;IS THE RESULT CORRECT.
MOV #QQBTP2,R2
MOV #4,R3
1$: CMP (R1)+,(R2)+
BNE QQB10 ;BRANCH IF INCORRECT.
SOB R3,1$
CMP #QQBTP1,R0 ;IS R0 CORRECT.
BNE QQB15 ;BRANCH IF INCORRECT.
CMP #210,R5 ;IS THE FPS CORRECT?
BNE QQB20 ;BRANCH IF INCORRECT.
BR QQBDONE
;THESE ARE DATA TABLES AND A DATA BUFFER.
QQBTP1: 023245
26720
122324
52672
WORD -1,-1,-1,-1
QQBTP2: 123245
26720
122324
52672
;REPORT RESULT INCORRECT:
QQB10: MOV #QQBTP1,$TMP3
MOV #QQBTP2,$TMP4
1$: ERROR +146 ;BAD DATA
BR QQBDONE
;REPORT R0 INCORRECT:
QQB15: MOV #QQBTP1,$TMP3
MOV R0,$TMP4
```

```
3995 020470 104147          1$:      ERROR   +147          ;SPEC DESTX ROX
3996 020472 000406          BR      QOBDONE
3997
3998
3999
4000 020474 012737 000210 00*240 ;REPORT FPS INCORRECT:
4001 020502 010537 001242 00B20: MOV      #210,$TMP3
4002 020506 104150          MOV      R5,$TMP4
4003
4004 020510          1$:      ERROR   +150
      020510 104412          QOBDONE:
                                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?).
```

4005

4006

.SBTTL TEST # 36 - SPECIAL DEST, MODE 3, TEST

*TEST 36 SPECIAL DEST, MODE 3, TEST
*
*THIS IS A TEST OF THE NEGF ABSF AND TSTF DESTINATION FLOWS
*MODE 3 USING THE NEGD INSTR.
*

020512 000004
4007
4008 020514
020514 104413
4009 020516 012701 020634
4010 020522 01270C 020644
4011 020526 012702 000004
4012 020532 012021
4013 020534 077202
4014 020536 012700 020654
4015 020542 012710 020634
4016 020546 042737 10C000 020634
4017 020554 012737 020570 001236
4018 020562 012701 000200
4019 020566 170101
4020
4021 020570 170730
4022
4023 020572 170205
4024 020574 012701 020634
4025 020600 012702 020644
4026 020604 012703 000004
4027 020610 022122
4028 020612 001021
4029 020614 077303
4030 020616 022700 020656
4031 020622 001025
4032 020624 022705 000210
4033 020630 001031
4034 020632 000436
4035
4036
4037 020634 023245
4038 020636 026720
4039 020640 122324
4040 020642 052672
4041 020644 123245
4042 020646 026720
4043 020650 123324
4044 020652 052672
4045 020654 020634
4046
4047
4048 020656 012737 020634 001240
4049 020664 012737 020644 001242
4050 020672 104150
4051 020674 000415
4052
4053

TST36: SCOPE
RRB1: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
MOV #RRBTP1,R1 ;SET UP THE DATA BUFFER.
MOV #RRBTP2,R0
MOV #4,R2
1\$: MOV (R0)+,(R1)+
SCB R2,1\$
MOV #RRBTP3,R0
MOV #RRBTP1,(R0)
BIC #100000,RRBTP1 ;MAKE THE OPERAND POSITIVE.
MOV #RRB2,\$TMP2
MOV #200,R1 ;SET FD.
LDFPS R1
RRB2: NEGD @ (R0)+ ;TEST INSTRUCTION.
STFPS R5 ;GET FPS.
MOV #RRBTP1,R1 ;IS THE RESULT CORRECT.
MOV #RRBTP2,R2
MOV #4,R3
1\$: CMP (R1)+,(R2)+
BNE RRB10 ;BRANCH IF INCORRECT.
SOB R3,1\$
CMP #RRBTP3+2,R0 ;IS R0 CORRECT.
BNE RRB15 ;BRANCH IF INCORRECT.
CMP #210,R5 ;IS THE FPS CORRECT?
BNE RRB20 ;BRANCH IF INCORRECT.
BR RRB DONE
;THESE ARE DATA TABLES AND A DATA BUFFER.
RRBTP1: 023245
26720
122324
52672
RRBTP2: 123245
26720
123324
52672
RRBTP3: RRBTP1
;REPORT RESULT INCORRECT:
RRB10: MOV #RRBTP1,\$TMP3
MOV #RRBTP2,\$TMP4
1\$: ERROR +150 ;BAD DATA
BR RRB DONE
;REPORT R0 INCORRECT:

4066

```

.SBTTL TEST # 37 - SPECIAL DEST, MODE 5, TEST
*****
:TEST 37 SPECIAL DEST, MODE 5, TEST
:
:THIS IS A TEST OF THE NEGJ ABSF AND TSTF DESTINATION FLOWS
:MODE 5 USING THE NEGJ INSTR.
:
*****
    
```

```

4067 020732 000004
4067 020734
4067 020734 104413
4068 020736 012701 021056
4069 020742 012700 021066
4070 020746 012702 000004
4071 020752 012021
4072 020754 077202
4073 020756 012700 021100
4074 020762 012760 021056 177776
4075 020770 042737 100000 021056
4076 020776 012737 021012 001236
4077 021004 012701 000200
4078 021010 170101
4079
4080 021012 170750
4081
4082 021014 170205
4083 021016 012701 021056
4084 021022 012702 021066
4085 021026 012703 000004
4086 021032 022122
4087 021034 001021
4088 021036 077303
4089 021040 022700 021076
4090 021044 001025
4091 021046 022705 000210
4092 021052 001031
4093 021054 000436
4094
4095
4096 021056 023245
4097 021060 026720
4098 021062 122324
4099 021064 052672
4100 021066 123245
4101 021070 026270
4102 021072 122324
4103 021074 052672
4104 021076 021056
4105
4106
4107 021100 012737 021056 001240
4108 021106 012737 021066 001242
4109 021114 104154
4110 021116 000415
4111
4112
4113 021120 012737 021076 001240
    
```

```

TST37: SCOPE
SSB1:
    LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
    MOV #SSBTP1,R1 ;SET UP THE DATA BUFFER.
    MOV #SSBTP2,R0
    MOV #4,R2
1$: MOV (R0)+,(R1)+
    SOB R2,1$
    MCV #SSBTP3+2,R0
    MOV #SSBTP1,-2(R0)
    BIC #100000,SSBTP1 ;MAKE THE OPERAND POSITIVE.
    MOV #SSB2,$TMP2
    MOV #200,R1 ;SET FD.
    LDFPS R1
SSB2: NEGJ @-(R0) ;TEST INSTRUCTION.
    STFPS R5 ;GET FPS.
    MOV #SSBTP1,R1 ;IS THE RESULT CORRECT.
    MOV #SSBTP2,R2
    MOV #4,R3
1$: CMP (R1)+,(R2)+
    BNE SSB10 ;BRANCH IF INCORRECT.
    SOB R3,1$
    CMP #SSBTP3,R0 ;IS R0 CORRECT.
    BNE SSB15 ;BRANCH IF INCORRECT.
    CMP #210,R5 ;IS THE FPS CORRECT?
    BNE SSB20 ;BRANCH IF INCORRECT.
    BR SSBDONE
;THESE ARE DATA TABLES AND A DATA BUFFER.
SSBTP1: 023245
        26720
        122324
        52672
SSBTP2: 123245
        26270
        122324
        52672
SSBTP3: SSBTP1
;REPORT RESULT INCORRECT:
SSB10: MOV #SSBTP1,$TMP3
        MOV #SSBTP2,$TMP4
1$: ERROR +154 ;BAD DATA
    BR SSBDONE
;REPORT R0 INCORRECT:
SSB15: MOV #SSBTP3,$TMP3
    
```

```
4114 021126 010037 00'242
4115 021132 104155
4116 021134 000406
4117
4118
4119 021136 012737 000'10 00'240
4120 021144 010537 001242
4121 021150 104156
4122
4123 021152
      021152 104412

      1$:      MOV      R0,$TMP4
      ERROR   +155      ;SPEC DESTX ROX
      BR      SSBDONE

      ;REPORT FPS INCORRECT:
      SSB20:  MOV      #210,$TMP3
      MOV      R5,$TMP4
      1$:      ERROR   +156

      SSBDONE:
      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?).
```

4124

```

.SBTTL TEST # 40 - SPECIAL DEST, FLOATING MODE 2, TEST
*****
:TEST 40 SPECIAL DEST, FLOATING MODE 2, TEST
:
:THIS IS A TEST OF THE NEGF ABSF AND TSTF DESTINATION FLOWS
:MODE 2 USING THE NEGF INSTR.
:
*****
    
```

```

4125 021154 000004
4125 021156
4125 021156 104413
4126 021160 012701 021270
4127 021164 012700 021300
4128 021170 012702 000004
4129 021174 012021
4130 021176 077202
4131 021200 012700 021270
4132 021204 042710 100000
4133 021210 012737 021224 001236
4134 021216 012701 000000
4135 021222 170101
4136
4137 021224 170720
4138
4139 021226 170205
4140 021230 012701 021270
4141 021234 012702 021300
4142 021240 012703 000004
4143 021244 022122
4144 021246 001020
4145 021250 077303
4146 021252 022700 021274
4147 021256 001024
4148 021260 022705 000010
4149 021264 001030
4150 021266 000435
4151
4152
4153 021270 023245
4154 021272 026720
4155 021274 122324
4156 021276 052672
4157 021300 123245
4158 021302 026720
4159 021304 122324
4160 021306 052672
4161
4162
4163 021310 012737 021270 001240
4164 021316 012737 021300 001242
4165 021324 104150
4166 021326 000415
4167
4168
4169 021330 012737 021274 001240
4170 021336 010037 001242
4171 021342 104160
    
```

```

TST40: SCOPE
TTB1:
    LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
    MOV #TTBTP1,R1 ;SET UP THE DATA BUFFER.
    MOV #TTBTP2,R0
    MOV #4,R2
1$: MOV (R0)+,(R1)+
    SOB R2,1$
    MCV #TTBTP1,R0
    BIC #100000,(R0) ;MAKE OPERAND POSITIVE.
    MOV #TTB2,$TMP2
    MOV #000,R1 ;SET FD.
    LDFPS R1
TTB2: NEGF (R0)+ ;TEST INSTRUCTION.
    STFPS R5 ;GET FPS.
    MOV #TTBTP1,R1 ;IS THE RESULT CORRECT.
    MOV #TTBTP2,R2
    MOV #4,R3
1$: CMP (R1)+,(R2)+
    BNE TTB10 ;BRANCH IF INCORRECT.
    SOB R3,1$
    CMP #TTBTP1+4,R0 ;IS R0 CORRECT.
    BNE TTB15 ;BRANCH IF INCORRECT.
    CMP #010,R5 ;IS THE FPS CORRECT?
    BNE TTB20 ;BRANCH IF INCORRECT.
    BR TTBDONE

;THESE ARE DATA TABLES AND A DATA BUFFER.
TTBTP1: 023245
        26720
        122324
        52672
TTBTP2: 123245
        26720
        122324
        52672

;REPORT RESULT INCORRECT:
TTB10: MOV #TTBTP1,$TMP3
        MOV #TTBTP2,$TMP4
1$: ERROR +150 ;BAD DATA
    BR TTBDONE

;REPORT R0 INCORRECT:
TTB15: MOV #TTBTP1+4,$TMP3
        MOV R0,$TMP4
1$: ERROR +160 ;SPEC DESTX R0X
    
```

```
4172 021344 000406 BR TTBDONE
4173
4174 :REPORT FPS INCORRECT:
4175 021346 012737 000010 001240 TTBDONE: MOV #010,$TMP3
4176 021354 010537 001242 MOV R5,$TMP4
4177 021360 104161 1$: ERROR +161
4178
4179 021362 TTBDONE:
      021362 104412 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?).
```

4180

```
:TEST TITLE:SPECIAL DEST, MODE2, GR7 (IMMEDIATE)
```

4181

```
.SBTTL TEST # 41 - SEE ABOVE COMMENT FOR TEST TITLE
*****
:TEST 41 SEE ABOVE COMMENT FOR TEST TITLE
:
:THIS IS A TEST OF THE NEGJ ABSJ AND TSTJ DESTINATION FLOWS
:MODE 2(IMMEDIATE) USING THE NEGJ INSTR.
:
*****
```

```
TST41: SCOPE
UUB1:
    LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
    MOV #UUBTP2,R0
    MOV #UUBTP1,R1 ;SET UP THE DATA BUFFER.
    MOV #4,R2
1$: MOV (R0)+,(R1)+
    SOB R2,1$
    MCV #UUBTP1,R0
    BIC #100000,UUBTP1 ;MAKE THE OPERAND POSITIVE.
    MOV #UUB2,$TMP2
    MOV #200,R1 ;SET FD.
    LDFPS R1
    CLR R1
UUB2: NEGJ (R7)+ ;TEST INSTRUCTION.
UUBTP1: 5201,5201,5201,5201
;NOTE THAT AFTER EXECUTING THIS INSTRUCTION R1 SHOULD CONTAIN 3.
    STFPS R5 ;GET FPS.
    MOV #UUBTP1,R3 ;IS THE RESULT CORRECT.
    MOV #UUBTP2,R2
    MOV #4,R4
1$: CMP (R3)+,(R2)+
    BNE UUB10 ;BRANCH IF INCORRECT.
    SOB R4,1$
    CMP #3,R1 ;WAS R1 INCREMENTED CORRECTLY.
    BNE UUB15 ;BRANCH IF INCORRECT.
    CMP #210,R5 ;IS THE FPS CORRECT?
    BNE UUB20 ;BRANCH IF INCORRECT.
    BR UUBDONE
;THESE ARE DATA TABLE.
UUBTP2: 105201
    5201
    5201
    5201
;REPORT RESULT INCORRECT:
UUB10: MOV #UUBTP1,$TMP3
    MOV #UUBTP2,$TMP4
1$: ERROR +162 ;BAD DATA
    BR UUBDONE
;REPORT FPS INCORRECT:
UUB20: MCV #210,$TMP3
    MOV R5,$TMP4
1$: ERROR +163 ;FPS
    BR UUBDONE
```

```
4182 021364 000004
021366 104413
4183 021370 012700 021514
4184 021374 012701 021442
4185 021400 012702 000004
4186 021404 012021
4187 021406 077202
4188 021410 012700 021442
4189 021414 042737 100000 021442
4190 021422 012737 021440 001236
4191 021430 012701 000200
4192 021434 170101
4193 021436 005001
4194
4195 021440 170727
4196 021442 005201 005201 005201
4197
4198 021452 170205
4199 021454 012703 021442
4200 021460 012702 021514
4201 021464 012704 000004
4202 021470 022322
4203 021472 001014
4204 021474 077403
4205 021476 022701 000003
4206 021502 001027
4207 021504 022705 000210
4208 021510 001015
4209 021512 000436
4210
4211
4212 021514 105201
4213 021516 005201
4214 021520 005201
4215 021522 005201
4216
4217
4218 021524 012737 021442 001240
4219 021532 012737 021514 001242
4220 021540 104162
4221 021542 000422
4222
4223
4224 021544 012737 000210 001240
4225 021552 010537 001242
4226 021556 104163
4227 021560 000413
4228
```

```
4229 ;REPORT PC INCORRECTLY INCREMENTED DURING EXECUTION.
4230 021562 162701 000003 UUB5: SUB #3,R1
4231 021566 006301 ASL R1
4232 021570 012702 021444 MOV #UUBTP1+2,R2
4233 021574 010237 001240 MOV R2,$TMP3
4234 021600 160102 SUB R1,R2
4235 021602 010237 001242 MOV R2,$TMP4
4236 021606 104164 1S: ERROR +164 ;PC BAD CONSTAND B GR7X
4237
4238 021610 UUBDONE: ;GO INITIALIZE THE FPS AND STACK; AND
      021610 104412 RSETUP ;SEE IF THE USER HAS EXPRESSED
                          ;THE DESIRE TO CHANGE THE SOFTWARE
                          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                          ;THE USER TYPED CONTROL G?).
```


4239

021612 000004
 4240 021614
 4241 021614 104413
 4242 021616 012701 021740
 4243 021622 012700 021750
 4244 021626 012702 000004
 4245 021632 012021
 4246 021634 077202
 4247 021636 012700 014537
 4248 021642 042767 100000 000070
 4249 021650 012767 021666 157360
 4250 021656 012701 000200
 4251 021662 170101
 4252
 4253 021664 005001
 4254 021666 170760 005201
 4255 021672 170205
 4256 021674 005701
 4257 021676 001030
 4258 021700 012701 021740
 4259 021704 012702 021750
 4260 021710 012703 000004
 4261 021714 022122
 4262 021716 001030
 4263 021720 077303
 4264 021722 022700 014537
 4265 021726 001034
 4266 021730 022705 000210
 4267 021734 001040
 4268 021736 000445
 4269
 4270
 4271 021740 023245
 4272 021742 026720
 4273 021744 122324
 4274 021746 052672
 4275 021750 123245
 4276 021752 026720
 4277 021754 122324
 4278 021756 052672
 4279
 4280
 4281
 4282 021760 012767 021670 157254
 4283 021766 012767 021672 157244
 4284 021774 104215
 4285 021776 000425
 4286

```

.SBTTL TEST # 42 - SPECIAL DEST, MODE 6, TEST
:*****
:*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
.DSABL AMA ;DISABLE MODE 6 TO MODE 3 CONVERSIONS
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,$TMP2
MOV #200,R1 ;SET FD.
LDFPS R1
XXB2: CLR R1
NEG 5201(R0) ;TEST INSTRUCTION.
STFPS R5 ;GET FPS.
TST R1
BNE XXB25 ;WAS THE PC CORRECT AFTER EXECUTION?
MOV #XXBTP1,R1 ;IS THE RESULT CORRECT.
MOV #XXBTP2,R2
MOV #4,R3
1$: 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
;THESE ARE DATA TABLES AND A DATA BUFFER.
XXBTP1: 023245
26720
122324
52672
XXBTP2: 123245
26720
122324
52672
;REPORT PC INCORRECT AFTER EXECUTION.
XXB25: MOV #XXB2+2,$TMP4
MOV #XXB2+4,$TMP3
1$: ERROR +215 ;PC NOT INCREMENTED BY 2.
BR XXBDONE
    
```

```
4287 ;REPORT RESULT INCORRECT:
4288 022000 012767 021740 157232 XXB10: MOV #XXBTP1,$TMP3
4289 022006 012767 021750 157226 MOV #XXBTP2,$TMP4
4290 022014 104216 1$: ERROR +216 ;BAD DATA
4291 022016 000415 BR XXBDONE
4292
4293 ;REPORT R0 INCORRECT:
4294 022020 012767 014537 157212 XXB15: MOV #XXBTP1-5201,$TMP3
4295 022026 010067 157210 MOV R0,$TMP4
4296 022032 104217 1$: ERROR +217 ;SPEC DESTX R0X
4297 022034 000406 BR XXBDONE
4298
4299 ;REPORT FPS INCORRECT:
4300 4301 022036 012767 000210 157174 XXB20: MOV #210,$TMP3
4302 022044 010567 157172 MOV R5,$TMP4
4303 022050 104220 1$: ERROR +220
4304
4305 022052 XXBDONE: RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
022052 104412 ;SEE IF THE USER HAS EXPRESSED
;THE DESIRE TO CHANGE THE SOFTWARE
;VIRTUAL CONSOLE SWITCH REGISTER (HAS
;THE USER TYPED CONTROL G?).
;REENABLE MODE 6 TO MODE 3 CONVERSIONS

4306 .ENABL AMA
```

4307

.SBTTL TEST # 43 - SPECIAL DEST, MODE 7, TEST
 :*****
 :*TEST 43 SPECIAL DEST, MODE 7, TEST
 :*
 :*THIS IS A TEST OF THE NEGF ABSF AND TSTF DESTINATION FLOWS
 :*MODE 7 USING THE NEGD INSTR.
 :*
 :*****

022054 000004
 4308
 4309 022056
 022056 104413
 4310 022060 012701 022210
 4311 022064 012700 022220
 4312 022070 012702 000004
 4313 022074 012021
 4314 022076 077202
 4315 022100 012700 015027
 4316 022104 012760 022210 005201
 4317 022112 042737 100000 022210
 4318 022120 012737 022136 001236
 4319 022126 012701 000200
 4320 022132 170101
 4321
 4322 022134 005001
 4323 022136 170770 005201
 4324
 4325 022142 170205
 4326 022144 005701
 4327 022146 001031
 4328 022150 012701 022210
 4329 022154 012702 022220
 4330 022160 012703 000004
 4331 022164 022122
 4332 022166 001031
 4333 022170 077303
 4334 022172 022700 015027
 4335 022176 001035
 4336 022200 022705 000210
 4337 022204 001041
 4338 022206 000446
 4339
 4340
 4341 022210 023245
 4342 022212 026720
 4343 022214 122324
 4344 022216 052672
 4345 022220 123245
 4346 022222 026720
 4347 022224 123324
 4348 022226 052672
 4349 022230 022210
 4350
 4351
 4352 022232 013737 022140 001242
 4353 022240 013737 022142 001240
 4354 022246 104221

TST43: SCOPE
 YYB1:
 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
 MOV #YYBTP1,R1 ;SET UP THE DATA BUFFER.
 MOV #YYBTP2,R0
 MOV #4,R2
 1\$: MOV (R0)+,(R1)+
 SOB R2,1\$
 MOV #YYBTP3-5201,R0
 MOV #YYBTP1,5201(R0)
 BIC #100000,YYBTP1 ;MAKE THE OPERAND POSITIVE.
 MOV #YYB2,\$TMP2
 MOV #200,R1 ;SET FD.
 LDFPS R1
 YYB2: CLR R1
 NEGD @5201(R0) ;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,\$TMP4
 MOV YYB2+4,\$TMP3
 1\$: ERROR +221 ;PC NOT INCREMENTED BY 2.

```
4355 022250 000425 BR YYBDONE
4356
4357
4358 022252 012737 022210 001240 ;REPORT RESULT INCORRECT:
YYB10: MOV #YYBTP1,$TMP3
4359 022260 012737 022220 001242 MOV #YYBTP2,$TMP4
4360 022266 104222 1$: ERROR +222 ;BAD DATA
4361 022270 000415 BR YYBDONE
4362
4363
4364 022272 012737 015027 001240 ;REPORT RO INCORRECT:
YYB15: MOV #YYBTP3-5201,$TMP3
4365 022300 010037 001242 MOV R0,$TMP4
4366 022304 104223 1$: ERROR +223 ;SPEC DESTX ROX
4367 022306 000406 BR YYBDONE
4368
4369
4370 022310 012737 000210 001240 ;REPORT FPS INCORRECT:
YYB20: MOV #210,$TMP3
4371 022316 010537 001242 MCV R5,$TMP4
4372 022322 104224 1$: ERROR +224
4373
4374 022324 YYBDONE:
022324 104412 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?).
```

4380

```
.SBTTL TEST # 44 - NEGD, ABSD AND TSTD TEST
:*****
:TEST 44      NEGD, ABSD AND TSTD TEST
:
:THIS IS A TEST OF THE NEGD ABSD AND TSTD INSTRUCTIONS.
:
:*****
```

```
022326 000004
4381
4382 022330
      022330 104413
4383 022332 004737 023172
4384 022336 000000
4385 022340 016341
4386 022342 055772
4387 022344 021133
4388 022346 055447
4389 022350 116341
4390 022352 055772
4391 022354 021133
4392 022356 055447
4393 022360 016341
4394 022362 055772
4395 022364 021133
4396 022366 055447
4397 022370 000207
4398 022372 000210
4399 022374 000200
4400 022376 177777
4401 022400 104200
4402 022402 000401
4403 022404 104201
4404 022406
```

```
TST44: SCOPE
:TEST NEGD WITH POS NONZERO OPERAND
WWB1:
      LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
      JSR          PC,NATSUB
1$:      0           ;FLAG=NEGD.
2$:      16341      ;OPERAND.
          55772
          21133
          55447
3$:      116341    ;RESULT.
          55772
          21133
          55447
4$:      16341    ;ERROR RES.
          55772
          21133
          55447
5$:      207       ;FPS BEFORE EXECUTION.
          210       ;FPS AFTER EXECUTION.
          200       ;ERROR FPS.
          -1        ;FEC
6$:      ERROR    +200 ;E10<---E10*200X ST 336
          BR       7$
          ERROR    +201 ;BUT ENBT ST 336X WENT TO 053 INTO 453
7$:
```

```
4405
4406 022406
      022406 104413
4407 022410 004737 023172
4408 022414 000000
4409 022416 152525
4410 022420 053545
4411 022422 055565
4412 022424 057505
4413 022426 052525
4414 022430 053545
4415 022432 055565
4416 022434 057505
4417 022436 152525
4418 022440 053545
4419 022442 055565
4420 022444 057505
4421 022446 000217
4422 022450 000200
4423 022452 000210
4424 022454 177777
4425 022456 104200
4426 022460 000401
4427 022462 104202
```

```
:TEST NEGD WITH NEG OPERAND.
WWB2:
      LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
      JSR          PC,NATSUB
1$:      0           ;FLAG=NEGD.
2$:      152525     ;OPERAND.
          53545
          55565
          57505
3$:      52525     ;RESULT.
          53545
          55565
          57505
4$:      152525     ;ERROR RES.
          53545
          55565
          57505
5$:      217       ;FPS BEFORE EXECUTION.
          200       ;FPS AFTER EXECUTION.
          210       ;ERROR FPS.
          -1        ;FEC
6$:      ERROR    +200 ;E10<---E10*200X S336
          BR       7$
          ERROR    +202 ;BUT ENBT X ST336 TO 453 INTO 053
7$:
```

```

4428 022464
4429
4430 022464
      022464 104413
4431 022466 004737 023172
4432 022472 000001
4433 022474 060705
4434 022476 124735
4435 022500 060124
4436 022502 073560
4437 022504 060705
4438 022506 124735
4439 022510 060124
4440 022512 073560
4441 022514 160705
4442 022516 124735
4443 022520 060124
4444 022522 073560
4445 022524 000217
4446 022526 000200
4447 022530 000210
4448 022532 177777
4449 022534 104203
4450 022536 000401
4451 022540 104203
4452 022542
4453
4454 022542
      022542 104413
4455 022544 004737 023172
4456 022550 000001
4457 022552 154345
4458 022554 076567
4459 022556 032123
4460 022560 043234
4461 022562 054345
4462 022564 076567
4463 022566 032123
4464 022570 043234
4465 022572 154345
4466 022574 076567
4467 022576 032123
4468 022600 043234
4469 022602 000217
4470 022604 000200
4471 022606 177777
4472 022610 177777
4473 022612 104204
4474 022614 000401
4475 022616 104171
4476 022620
4477
4478 022620
      022620 104413
4479 022622 004737 023172
4480 022626 000002
4481 022630 012321

7$:
;TEST ABSD WITH POSITIVE OPERAND
WWB3:
      LPERR
      JSR      PC,NATSUB      ;SET UP THE LOOP ON ERROR ADDRESS.
1$:      1
      ;FLAG=ABSD.
2$:      60705
      ;OPERAND.
      124735
      60124
      73560
3$:      60705
      ;RESULT.
      124735
      60124
      73560
4$:      160705
      ;ERROR RES.
      124735
      60124
      73560
5$:      217
      ;FPS BEFORE EXECUTION.
      200
      ;FPS AFTER EXECUTION.
      210
      ;ERROR FPS.
      -1
      ;EITHER BUT OP1B
6$:      ERROR +203
      BR      7$
      ERROR +203
      ;BUT ST 055 TO 336 INTO 335
      ;OR BUT ENBT ST 335 TO 452 INTO 052
7$:
;TEST ABSD WITH NEG. OPERAND
WWB4:
      LPERR
      JSR      PC,NATSUB      ;SET UP THE LOOP ON ERROR ADDRESS.
1$:      1
      ;FLAG=ABSD.
2$:      154345
      ;OPERAND.
      76567
      32123
      43234
3$:      54345
      ;RESULT.
      76567
      32123
      43234
4$:      154345
      ;ERROR RES.
      76567
      32123
      43234
5$:      217
      ;FPS BEFORE EXECUTION.
      200
      ;FPS AFTER EXECUTION.
      -1
      ;ERROR FPS.
      -1
      ;E10*E10*200X ST 452
6$:      ERROR +204
      BR      7$
      ERROR +17:
7$:
;TEST WITH POSITIVE OP
WWB5:
      LPERR
      JSR      PC,NATSUB      ;SET UP THE LOOP ON ERROR ADDRESS.
1$:      2
      ;FLAG=TSTD.
2$:      12321
      ;OPERAND.
    
```

4482	022632	045654		45654	
4483	022634	070107		70107	
4484	022636	034543		34543	
4485	022640	012321	3\$:	12321	;RESULT.
4486	022642	045654		45654	
4487	022644	070107		70107	
4488	022646	034543		34543	
4489	022650	112321	4\$:	112321	;ERROR RES.
4490	022652	045654		45654	
4491	022654	070107		70107	
4492	022656	034543		34543	
4493	022660	000217	5\$:	217	;FPS BEFORE EXECUTION.
4494	022662	000200		200	;FPS AFTER EXECUTION.
4495	022664	000210		210	;ERROR FPS.
4496	022666	177777		-1	
4497	022670	104205	6\$:	ERROR +205	;BUT (OP1B) X ST044 TO 336 INTO 334
4498	022672	000401		BR 7\$	
4499	022674	104206		ERROR +206	;BUT ENBT ST 334 TO 453 INTO 053
4500	022676		7\$:		
4501			;TEST TSTD WITH NEG OP		
4502	022676		WWB6:		
	022676	104413		LPERR	;SET UP THE LOOP ON ERROR ADDRESS.
4503	022700	004737	023172	JSR PC,NATSUB	
4504	022704	000002	1\$:	2	;FLAG=TSTD.
4505	022706	123765	2\$:	123765	;OPERAND.
4506	022710	023407		23407	
4507	022712	034510		34510	
4508	022714	045621		45621	
4509	022716	123765	3\$:	123765	;RESULT.
4510	022720	023407		23407	
4511	022722	034510		34510	
4512	022724	045621		45621	
4513	022726	023765	4\$:	23765	;ERROR RES.
4514	022730	023407		23407	
4515	022732	034510		34510	
4516	022734	045621		45621	
4517	022736	000207	5\$:	207	;FPS BEFORE EXECUTION.
4518	022740	000210		210	;FPS AFTER EXECUTION.
4519	022742	000200		200	;ERROR FPS.
4520	022744	177777		-1	
4521	022746	104207	6\$:	ERROR +207	;BUT OPB1 ST 055 TO 335 INTO 334
4522	022750	000401		BR 7\$	
4523	022752	104210		ERROR +210	;BUT ENBT ST 334 TO 053 INTO 453
4524	022754		7\$:		
4525			;TEST TSTD 0 OP		
4526	022754		WWB7:		
	022754	104413		LPERR	;SET UP THE LOOP ON ERROR ADDRESS.
4527	022756	004737	023172	JSR PC,NATSUB	
4528	022762	000002	1\$:	2	;FLAG=TSTD.
4529	022764	000175	2\$:	175	;OPERAND.
4530	022766	176737		176737	
4531	022770	071727		71727	
4532	022772	037574		37574	
4533	022774	000175	3\$:	175	;RESULT.
4534	022776	176737		176737	
4535	023000	071727		71727	
4536	023002	037574		37574	

4592 023156 000014
 4593 023160 104211
 4594 023162 000401
 4595 023164 104214
 4596 023166
 4597 023166 000137 023606
 4598
 4599
 4600
 4601
 4602
 4603
 4604
 4605
 4606
 4607
 4608
 4609
 4610
 4611
 4612
 4613
 4614
 4615
 4616
 4617
 4618
 4619
 4620
 4621
 4622
 4623
 4624
 4625
 4626
 4627
 4628
 4629
 4630
 4631
 4632
 4633
 4634
 4635 023172 012601
 4636 023174 010102
 4637 023176 062702 000002
 4638 023202 012703 023574
 4639 023206 012704 000004
 4640 023212 012223
 4641 023214 077402
 4642 023216 016100 000032
 4643 023222 170100
 4644 023224 012700 023574
 4645 023230 011102
 4646 023232 006302
 4647 023234 006302
 4648 023236 012703 023252

```

14
6$:  ERROR +211      ;+
      BR      7$
      ERROR +214      ;BUT FIUV ST 257 TO 255 INTO 355
7$:  JMP      WWBDONE
    
```

: THIS SUBROUTINE, NATSUB, IS USED TO SET UP THE OPERANDS, EXECUTE
 : THE EITHER A TSTD, AN ABSD OR A NEGD 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 +X          ;DATA ERROR.
      BR      CONT
      ERR2:   ERROR +X          ;FPS ERROR.
      CONT:   .WORD  X           ;RETURN ADDRESS
    
```

: THE OPERAND IS SET UP IN NATBF1. THEN
 : THE EITHER THE TSTD, NEGD OR ABSD INSTRUCTION IS EXECUTED.
 : NATSUB USES THE FIRST OPERAND AS A FLAG TO DETERMINE WHICH INSTRUCTION
 : IS TO BE EXECUTED: 0 = NEGD, 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
 : COMPARE 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.

```

NATSUB: MOV      (SP)+,R1          ;GET A POINTER TO THE ARGUMENTS.
        MOV      R1,R2          ;COPY THE OPERAND.
        ADD      #2,R2
        MOV      #NATBF1,R3
        MOV      #4,R4
1$:     MOV      (R2)+,(R3)+
        SOB      R4,1$
        MOV      32(R1),R0       ;LOAD THE FPS.
        LDFPS   R0
        MOV      #NATBF1,R0     ;SET UP THE OPERAND ADDRESS.
        MOV      (R1),R2       ;GET THE FLAG TO DETERMINE WHICH
        ASL     R2              ;INSTRUCTION TO EXECUTE.
        ASL     R2              ;0 = NEGD, 1 = ABSD, 2 = TSTD
        MOV      #NATINS,R3
    
```

```

4649 023242 060203          ADD      R2,R3
4650 023244 010337 001236    MOV      R3,$TMP2
4651 023250 000113          JMP      (R3)          ;GO EXECUTE THE INSTRUCTION.
4652 023252 170710          NATINS: NEG D      (R0)
4653 023254 000403          BR       2$
4654 023256 170610          ABSD     (R0)
4655 023260 000401          BR       2$
4656 023262 170510          TSTD     (R0)
4657
4658 023264 170204          2$:     STFPS    R4          ;GET THE FPS.
4659 023266 170305          STST     R5          ;GET THE FEC.
4660 023270 010102          MOV      R1,R2
4661 023272 062702 000002          ADD      #2,R2
4662 023276 J10237 001240          MOV      R2,$TMP3
4663 023302 062702 000010          ADD      #10,R2
4664 023306 010237 001244          MOV      R2,$TMP5
4665 023312 012737 023574 001242          MCV      #NATBF1,$TMP4
4666 023320 010437 001250          MOV      R4,$TMP7
4667 023324 016137 000034 001252          MOV      34(R1),$TMP10
4668 023332 010100          MOV      R1,R0          ;WAS THE RESULT CORRECT?
4669 023334 062700 000012          ADD      #12,R0
4670 023340 012702 023574          MOV      #NATBF1,R2
4671 023344 012703 000004          MOV      #4,R3
4672 023350 022022          3$:     CMP      (R0)+,(R2)+
4673 023352 001014          BNE      10$          ;BRANCH IF INCORRECT.
4674 023354 077303          SOB      R3,3$
4675 023356 026104 000034          CMP      34(R1),R4          ;WAS THE FPS CORRECT?
4676 023362 001032          BNE      15$          ;BRANCH IF INCORRECT.
4677 023364 005761 000034          TST      34(R1)          ;IF THE EXPECTED FPS WAS NEGATIVE CHECK THE FEC.
4678 023370 100003          BPL      4$
4679 023372 026105 000040          CMP      40(R1),R5          ;WAS THE FEC CORRECT.
4680 023376 001037          BNE      20$          ;BRANCH IF INCORRECT.
4681 023400 000161 000050          4$:     JMP      50(R1)          ;RETURN.
4682
4683          ;THE RESULT WAS INCORRECT BUT WAS THIS FAILURE ANTICIPATED?
4684          ;SEE IF THE RESULT WAS ANTICIPATED:
4685 023404          10$:
4686 023404 011105          MOV      (R1),R5
4687 023406 006305          ASL      R5
4688 023410 006305          ASL      R5
4689 023412 062705 023524          ADD      #NATER1,R5
4690 023416 010100          MOV      R1,R0
4691 023420 062700 000022          ADD      #22,R0
4692 023424 012702 023574          MOV      #NATBF1,R2
4693 023430 012703 000004          MOV      #4,R3
4694 023434 022022          11$:     CMP      (R0)+,(R2)+
4695 023436 001003          BNE      12$          ;BRANCH IF NOT ANTICIPATED.
4696 023440 077303          SOB      R3,11$
4697
4698          ;THE ERROR WAS ANTICIPATED SO RETURN.
4699 023442 000161 000042          JMP      42(R1)
4700
4701          ;THE ERROR WAS NOT ANTICIPATED SO REPORT IT HERE.
4702 023446 000115          12$:     JMP      (R5)          ;GO TO THE PROPER ERROR CALL.
4703
4704          ;THE FPS WAS INCORRECT.
4705 023450 026105 000036          15$:     CMP      36(R1),R5          ;WAS THIS ERROR ANTICIPATED?
    
```

```

4706 023454 001002          BNE      16$          ;BRANCH IF NOT ANTICIPATED.
4707
4708          ;THE FPS ERROR WAS ANTICIPATED SO RETURN.
4709 023456 000161 000046    JMP      46(R1)
4710
4711          ;THE FPS FAILURE WAS NOT ANTICIPATED SO REPORT IT HERE.
4712 023462 011102    16$:  MOV      (R1),R2
4713 023464 006302      ASL      R2
4714 023466 006302      ASL      R2
4715 023470 062702 023542    ADD      #NATER2,R2
4716 023474 000112      JMP      (R2)          ;GO TO THE PROPER ERROR CALL.
4717
4718          ;REPORT THAT THE FEC WAS INCORRECT.
4719 023476 016137 000040 001256 20$:  MOV      40(R1),$TMP12
4720 023504 010537 001254      MOV      R5,$TMP11
4721 023510 011102      MOV      (R1),R2
4722 023512 006302      ASL      R2
4723 023514 006302      ASL      R2
4724 023516 062702 023556    ADD      #NATER3,R2
4725 023522 000112      JMP      (R2)          ;GO TO THE PROPER ERROR CALL.
4726
4727          ;THESE ARE THE ERROR CALLS FOR EACH INDIVIDUAL INSTRUCTION AND CONDITION.
4728 023524 104165    NATER1: ERROR +165          ;NEGD BAD DATA
4729 023526 000403      BR      NATRET
4730 023530 104166      ERROR +166          ;ABSD BAD DATA
4731 023532 000401      BR      NATRET
4732 023534 104167      ERROR +167          ;TSTD BAD DATA
4733 023536 000161 000050    NATRET: JMP      50(R1)
4734
4735          ;FPS INCORRECT:
4736 023542 104170    NATER2: ERROR +170          ;NEGD FPSX
4737 023544 000774      BR      NATRET
4738 023546 104171      ERROR +171          ;ABSD FPSX
4739 023550 000772      BR      NATRET
4740 023552 104172      ERROR +172          ;TSTD FPSX
4741 023554 000770      BR      NATRET
4742
4743          ;FEC INCORRECT:
4744 023556 104173    NATER3: ERROR +173          ;NEGD FECX
4745 023560 000766      BR      NATRET
4746 023562 104174      ERROR +174          ;ABSD FECX
4747 023564 000764      BR      NATRET
4748 023566 104175      ERROR +175          ;TSTD FECX
4749 023570 000762      BR      NATRET
4750
4751 023572 177777      .WORD   -1
4752 023574 177777 177777 177777 NATBF1: .WORD -1,-1,-1,-1,-1
4753
4754 023606          WWDONE:
         023606 104412      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?).
4755
4756
4763
    
```

```

4764          .SBTTL TEST # 45 - SOURCE MODES, MODE 1 (FL=0), TEST
               :*****
               :*TEST 45 SOURCE MODES, MODE 1 (FL=0), TEST
               :*
               :* THIS IS A TEST OF SOURCE MODE 1
               :* USING THE LDFPS INSTR
               :*****
023610 000004 TST45: SCOPE

4765
4766
4767 023612 AAC1:
023612 104413          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.

4768
4769 023614 012700 023672          MOV      #AACTP1,R0      ;SET UP TEST DATA IN BUFFER.
4770 023620 012710 147517          MOV      #147517,(R0)
4771 023624 012737 147517 001240          MCV      #147517,$TMP3      ;SAVE DATA IN CASE OF ERROR.
4772 023632 012737 023646 001236          MOV      #AAC2,$TMP2
4773 023640 012737 023732 000004          MOI      #AAC20,ERRVECT      ;SET UP FOR TRAPS TO 4.
4774 023646 170110 AAC2: LDFPS      (R0)          ;TEST INSTRUCTION.
4775
4776 023650 170205          STFPS   R5          ;GET FPS
4777
4778 023652 020027 023672          CMP      R0,#AACTP1        ;IS R0 CORRECT?
4779 023656 001007          BNE      AAC10            ;BR IF NOT.
4780 023660 022705 147517          CMP      #147517,R5        ;IS FPS CORRECT?
4781 023664 001013          BNE      AAC11            ;BR IF NOT.
4782 023666 000437          BR       AACDONE
4783
4784          ;TEST BUFFER AND DATA:
4785 023670 177777          -1
4786 023672 147517          AAC10: 147517
4787 023674 177777          -1
4788
4789          ;REPORT R0 INCORRECT.
4790 023676 012737 023672 001240 AAC10: MOV      #AACTP1,$TMP3
4791 023704 010037 001242          MOV      R0,$TMP4
4792 023710 104225          1$:      ERROR      +225      ;R0 BAD BUT FSRC FAILED
4793 023712 000425          BR       AACDONE
4794
4795          ;REPORT FPS INCORRECT.
4796 023714 012737 147517 001240 AAC11: MOV      #147517,$TMP3      ;REPORT FPS INCORRECT.
4797 023722 010537 001242          MOV      R5,$TMP4
4798 023726 104226          1$:      ERROR      +226
4799 023730 000416          BR       AACDONE
4800
4801          ;TRAP HERE THROUGH VECTOR FOUR. SEE IF THE TRAP WAS DURING
4802          ;EXECUTION OF THE FPS INSTRUCTION BEING TESTED. IF SO REPORT
4803          ;FAILURE. OTHERWISE GO TO THE SPURIOUS TRAP TO 4 HANDLING.
4804 023732 AAC20:
4805 023732 011602          MOV      (SP),R2
4806 023734 020227 023650          CMP      R2,#AAC2+2
4807 023740 001405          BEQ      1$
4808 023742 020227 023652          CMP      R2,#AAC2+4
4809 023746 001402          BEQ      1$
4810 023750 000137 047302          JMP      (PSPUR
4811 023754 022626          1$:      CMP      (SP)+,(SP)+
    
```

4812 023756 010237 001236
4813 023762 104227
4814 023764 000400
4815
4816 023766
023766 104412

2\$: MOV R2,\$TMP2
ERROR +227
BR AACDONE

AACDONE:
RSETUP

:ODD ADRES
:BUT FDSTX IN ST 771

: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?).

4817
4818

```

4819          .SBTTL TEST # 46 - SOURCE MODES, MODE 2 (FL=0), TEST
               :*****
               :*TEST 46          SOURCE MODES, MODE 2 (FL=0), TEST
               :*
               :* THIS IS A TEST OF SOURCE MODE 2
               :* USING THE LDFPS INSTR
               :*
               :*****
               :TST46: SCOPE

4820          023770 000004
4821          023772
4821          023772 104413
4822
4823          023774 012700 024052
4824          024000 012710 145212
4825          024004 012737 145212 001240
4826          024012 012737 024026 001236
4827          024020 012737 024112 000004
4828
4829          024026 170120
4830
4831          024030 170205
4832
4833          024032 020027 024054
4834          024036 001007
4835          024040 022705 145212
4836          024044 001013
4837          024046 000436
4838
4839
4840          :TEST BUFFER AND DATA:
4841          024050 177777
4842          024052 177777
4843          024054 177777
4844
4845
4846          :REPORT R0 INCORRECT.
4847          024056 012737 024054 001240 BBC10: MOV #BBC1P1+2,$TMP3
4848          024064 010037 001242      MOV R0,$TMP4
4849          024070 104230      1$: ERROR +230 ;R0 BAD BUT FSRC FAILED
4850          024072 000424      BR BBCDONE
4851
4852          :REPORT FPS INCORRECT.
4853          024074 012737 145212 001240 BBC11: MOV #145212,$TMP3 ;REPORT FPS INCORRECT.
4854          024102 010537 001242      MOV R5,$TMP4
4855          024106 104231      1$: ERROR +231
4856          024110 000415      BR BBCDONE
4857
4858          :TRAP HERE THROUGH VECTOR FOUR. SEE IF THE TRAP WAS DURING
4859          :EXECUTION OF THE FPS INSTRUCTION BEING TESTED. IF SO REPORT
4860          :FAILURE. OTHERWISE GO TO THE SPURIOUS TRAP TO 4 HANDLING.
4861          024112
4862          024112 011602
4863          024114 020227 024030
4864          024120 001405
4865          024122 020227 024032
4866          024126 001402
               BBC20: MOV (SP),R2
               CMP R2,#BBC2+2
               BEQ 1$
               CMP R2,#BBC2+4
               BEQ 1$
    
```

4867 024130 000137 047302
4868 024134 022626
4869 024136 010237 001236
4870 024142 104232
4871
4872
4873 024144
024144 104412

1\$: JMP CPSPUR
CMP (SP)+,(SP)+
MOV R2,\$TMP2
2\$: ERROR +232

:ODD ADRES
:BUT FDSTX IN ST 771

BBCDONE:
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?).

4874
4875

4876

```
.SBTTL TEST # 47 - SOURCE MODES, MODE 4 (FL=0), TEST
:*****
:TEST 47 SOURCE MODES, MODE 4 (FL=0), TEST
:
:* THIS IS A TEST OF SOURCE MODE 4
:* USING THE LDFPS INSTR
:
:*****
```

```
4877 024146 000004
4878 024150
024150 104413
4879
4880 024152 012700 024242
4881 024156 012760 105252 177776
4882 024164 012737 105252 001240
4883 024172 012737 024206 001236
4884 024200 012737 024306 000004
4885 024206 170140
4886 024210 170205
4887 024212 020C27 024240
4888 024216 001015
4889 024220 022705 105252
4890 024224 001021
4891 024226 000444
4892
4893 024230 177777 177777 177777
4894 024240 177777
4895 024242 177777 177777 177777
4896
4897 024252 012737 024240 001240
4898 024260 010037 001242
4899 024264 104233
4900 024266 000424
4901 024270 012737 105252 001240
4902 024276 010537 001242
4903 024302 104234
4904 024304 000415
4905 024306 011602
4906 024310 020227 024210
4907 024314 001405
4908 024316 020227 024212
4909 024322 001402
4910 024324 000137 047302
4911 024330 022626
4912 024332 010237 001236
4913 024336 104235
4914 024340
024340 104412
```

```
TST47: SCOPE
DDC1: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
MOV #DDCTP1+2,R0 ;SET UP THE TEST DATA BUFFER.
MOV #105252,-2(R0)
MOV #105252,$TMP3 ;SAVE DATA IN CASE OF ERROR.
MOV #DDC2,$TMP2
MOV #DDC20,ERRVEC
DDC2: LDFPS -(R0)
STFPS R5
CMP R0,#DDCTP1
BNE DDC10
CMP #105252,R5
BNE DDC11
BR DDCDONE
DDCTP1: -1,-1,-1,-1
-1
-1,-1,-1,-1
DDC10: MOV #DDCTP1,$TMP3
MOV R0,$TMP4
1$: ERROR +233 ;R0 BAD BUT FSRC FAILED
BR DDCDONE
DDC11: MOV #105252,$TMP3 ;REPORT FPS INCORRECT.
MOV R5,$TMP4
1$: ERROR +234
BR DDCDONE
DDC20: MOV (SP),R2
CMP R2,#DDC2+2
1$
BEQ 1$
CMP R2,#DDC2+4
1$
BEQ 1$
JMP CPSPUR
1$: CMP (SP)+,(SP)+
MOV R2,$TMP2
2$: ERROR +235 ;DDD ADRES
DDCDONE: 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?).
```


4915

```
.SBTTL TEST # 50 - SOURCE MODES, MODE 3 (FL=0), TEST
:*****
:*TEST 50 SOURCE MODES, MODE 3 (FL 0), TEST
:*
:* THIS IS A TEST OF SOURCE MODE 3
:* USING THE LDFPS INSTR
:*
:*****
```

```
4916 024342 000004
024344 104413
4917 024346 012700 024450
4918 024352 012710 024440
4919 024356 012737 103456 024440
4920 024364 012737 103456 001240
4921 024372 012737 024406 001236
4922 024400 012737 024516 000004
4923 024406 170130
4924 024410 170205
4925 024412 020027 024452
4926 024416 001021
4927 024420 022705 103456
4928 024424 001025
4929 024426 000450
4930
4931
4932
4933 024430 177777 177777 177777
4934 024440 177777
4935 024442 177777 177777 177777
4936 024450 024440 177777 177777
4937
4938
4939
4940 024462 012737 024452 001240
4941 024470 010037 001242
4942 024474 104236
4943 024476 000424
4944
4945
4946 024500 012737 103456 001240
4947 024506 010537 001242
4948 024512 104237
4949 024514 000415
4950
4951
4952
4953 024516 011602
4954 024520 020227 024410
4955 024524 001405
4956 024526 020227 024412
4957 024532 001402
4958 024534 000137 047302
4959 024540 022626
4960 024542 010237 001236
4961 024546 104240
4962 024550
```

```
TST50: SCOPE
EEC1: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
MOV #EECTP2,R0
MOV #EECTP1,(R0)
MOV #103456,EECTP1
MOV #103456,$TMP3
MOV #EEC2,$TMP2
MOV #EEC20,ERRVECT ;SET UP FOR TRAPS TO 4.
EEC2: LDFPS @ (R0)+ ;TEST INSTRUCTION.
STFPS R5 ;GET THE FPS.
CMP R0,#EECTP2+2 ;IS R0 CORRECT?
BNE EEC10 ;BR IF NOT.
CMP #103456,R5 ;IS THE FPS CORRECT?
BNE EEC11 ;BR IF NOT.
BR EECDONE

;TEST BUFFER AND DATA:
EECTP1: -1,-1,-1,-1
EECTP2: EECTP1,-1,-1,-1.

;REPORT R0 INCORRECT.
EEC10: MOV #EECTP2+2,$TMP3
MOV R0,$TMP4
1$: ERROR +236 ;R0 BAD BUT FSRC FAILED
BR EECDONE

;REPORT FPS INCORRECT.
EEC11: MOV #103456,$TMP3 ;REPORT FPS INCORRECT.
MOV R5,$TMP4
1$: ERROR +237
BR EECDONE

;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.
EEC20: MOV (SP),R2
CMP R2,#EEC2+2
BEQ 1$
CMP R2,#EEC2+4
BEQ 1$
JMP CPSPUR
1$: CMP (SP)+,(SP)+
MOV R2,$TMP2
2$: ERROR +240 ;DDD ADRES
EECDONE:
```

024550 104412

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?).

4963

```

.SBTTL TEST # 51 - SOURCE MODES, MODE 5 (FL=0), TEST
*****
*TEST 51 SOURCE MODES, MODE 5 (FL=0), TEST
*
* THIS IS A TEST OF SOURCE MODE 5
* USING THE LDFPS INSTR
*
*****
    
```

```

024552 000004
4964 024554
024554 104413
4965 024556 012700 024656
4966 024562 012760 024644 177776
4967 024570 012737 045412 024644
4968 024576 012737 045412 001240
4969 024604 012737 024554 001236
4970 024612 012737 024720 000004
4971 024620 170150
4972 024622 170205
4973 024624 020027 024654
4974 024630 001015
4975 024632 022705 045412
4976 024636 001021
4977 024640 000444
4978
4979
4980
4981 024642 177777
4982 024644 177777
4983 024646 177777 177777 177777
4984 024654 024644 177777 177777
4985
4986
4987
4988 024664 012737 024654 001240
4989 024672 010037 001242
4990 024676 104241
4991 024700 000424
4992
4993
4994 024702 012737 045412 001240
4995 024710 010537 001242
4996 024714 104242
4997 024716 000415
4998
4999
5000
5001 024720 011602
5002 024722 020227 024622
5003 024726 001405
5004 024730 020227 024624
5005 024734 001402
5006 024736 000137 047302
5007 024742 022626
5008 024744 010237 001236
5009 024750 104243
5010 024752
    
```

```

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,$TMP3 ;SAVE DATA IN CASE OF ERROR.
    MOV #FFC1,$TMP2
    MCV #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,$TMP3
    MOV R0,$TMP4
1$: ERROR +241 ;R0 BAD BUT FSRC FAILED
    BR FFCDONE

;REPORT FPS INCORRECT.
FFC11: MOV #45412,$TMP3 ;REPORT FPS INCORRECT.
    MOV R5,$TMP4
1$: ERROR +242
    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 C$SPUR
1$: CMP (SP)+,(SP)+
    MOV R2,$TMP2
2$: ERROR +243 ;ODD ADRES
    FFCDONE:
    
```

024752 104412

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?).

5011

```

.SBTTL TEST # 52 - SOURCE MODES, MODE 6 (FL=0), TEST
.....
*TEST 52 SOURCE MODES, MODE 6 (FL=0), TEST
*
* THIS IS A TEST OF SOURCE MODE 6
* USING THE LDFPS INSTR
*
.....
    
```

```

5012 024754 000004
5013 024756
5014 024756 104413
5015 024760 012700 017647
5016 024764 012767 046543 000056
5017 024772 012767 046543 154240
5018 025000 012767 025016 154230
5019 025006 005001
5020 025010 012767 025136 152766
5021 025016 170160 005201
5022 025022 170204
5023 025024 005701
5024 025026 001033
5025 025030 G20027 017647
5026 025034 001012
5027 025036 022704 046543
5028 025042 001016
5029 025044 000451
5030
5031
5032 025046 177777
5033 025050 177777 177777
5034 025060 177777
5035
5036
5037 025062 012767 017647 154150
5038 025070 010067 154146
5039 025074 104244
5040 025076 000434
5041
5042
5043 025100 012767 046543 154132
5044 025106 010467 154130
5045 025112 104245
5046 025114 000425
5047
5048
5049 025116 012767 025022 154114
5050 025124 012767 025020 154110
5051 025132 104246
5052 025134 000415
5053
5054
5055
5056 025136 011602
5057 025140 020227 025020
5058 025144 001405
    
```

```

TST52: SCOPE
.DSABLE AMA ;DISABLE MODE 6 TO MODE 3 CONVERSIONS
GGC1:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
MOV #GGCTP1-5201,R0 ;SET UP THE TEST DATA BUFFER.
MOV #46543,GGCTP1
MOV #46543,$TMP3 ;SAVE DATA IN CASE OF ERROR.
MOV #GGC2,$TMP2
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,$TMP3
MOV R0,$TMP4
1$: ERROR +244 ;R0 BAD BUT FSRC FAILED
BR GGCDONE

;REPORT FPS INCORRECT.
GGC11: MOV #46543,$TMP3 ;REPORT FPS INCORRECT.
MOV R4,$TMP4
1$: ERROR +245
BR GGCDONE

;REPORT PC INCORRECT AFTER INSTRUCTION.
GGC25: MOV #GGC2+4,$TMP3
MOV #GGC2+2,$TMP4
1$: ERROR +246 ;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
BEG 1$
    
```

5059 025146 020227 025022
5060 025152 001402
5061 025154 000167 022122
5062 025160 022626
5063 025162 010267 154050
5064 025166 104247
5065 025170
025170 104412

CMP R2,#GGC2+4
BEQ 18
JMP (PSPUR
18: CMP (SP)+,(SP)+
MOV R2,\$TMP2
28: ERROR +247
GGCDONE: RSETUP

:ODD ADRES

: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?).

5066

.ENABL AMA

:REENABLE MODE 6 TO MODE 7 CONVERSIONS

5067

```
.SBTTL TEST # 53 - SOURCE MODES, MODE 7 (FL=0), TEST
.....
:TEST 53 SOURCE MODES, MODE 7 (FL=0), TEST
:
: THIS IS A TEST OF SOURCE MODE 7
: USING THE LDFPS INSTR
:
:.....
```

```
5068 025172 000004
5068 025174
5068 025174 104413
5069 025176 012700 020103
5070 025202 012760 025274 005201
5071 025210 012737 004547 025274
5072 025216 012737 004547 001240
5073 025224 012737 025242 001236
5074 025232 005001
5075 025234 012737 025370 000004
5076 025242 170170 005201
5077 025246 170204
5078 025250 005701
5079 025252 001036
5080 025254 020027 020103
5081 025260 001015
5082 025262 022704 004547
5083 025266 001021
5084 025270 000454
5085
5086
5087
5088 025272 177777
5089 025274 177777 177777 177777
5090 025304 177777 177777 177777
5091
5092
5093 025314 012737 020103 001240
5094 025322 010037 001242
5095 025326 104250
5096 025330 000434
5097
5098
5099 025332 012737 004547 001240
5100 025340 010437 001242
5101 025344 104251
5102 025346 000425
5103
5104
5105 025350 012737 025246 001240
5106 025356 012737 025244 001242
5107 025364 104252
5108 025366 000415
5109
5110
5111
5112 025370 011602
5113 025372 020227 025244
5114 025376 001405

TST53: SCOPE
HHC1:
    LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
    MOV #HHC2P2-5201,R0 ;SET UP THE TEST DATA BUFFER.
    MOV #HHC2P1,5201(R0)
    MOV #4547,HHC2P1
    MOV #4547,$TMP3 ;SAVE DATA IN CASE OF ERROR.
    MOV #HHC2,$TMP2
    CLR R1
    MOV #HHC20,ERRVECT ;SET UP FOR TRAPS TO 4.
HHC2: LDFPS @520(R0) ;TEST INSTRUCTION.
    STFPS R4 ;GET THE FPS.
    TST R1 ;WAS PC CORRECT AFTER EXECUTION?
    BNE HHC25 ;BR IF NOT.
    CMP R0,#HHC2P2-5201 ;IS R0 CORRECT?
    BNE HHC10 ;BR IF NOT.
    CMP #4547,R4 ;IS THE FPS CORRECT?
    BNE HHC11 ;BR IF NOT.
    BR HHCDONE

;TEST BUFFER AND DATA:
-1
HHC2P1: .WORD -1,-1,-1,-1
HHC2P2: .WORD -1,-1,-1,-1

;REPORT R0 INCORRECT.
HHC10: MOV #HHC2P2-5201,$TMP3
    MOV R0,$TMP4
1$: ERROR +250 ;R0 BAD BUT FSRC FAILED
    BR HHCDONE

;REPORT FPS INCORRECT.
HHC11: MOV #4547,$TMP3 ;REPORT FPS INCORRECT.
    MOV R4,$TMP4
1$: ERROR +251
    BR HHCDONE

;REPORT PC INCORRECT AFTER INSTRUCTION.
HHC25: MOV #HHC2+4,$TMP3
    MOV #HHC2+2,$TMP4
1$: ERROR +252 ;PC X
    BR HHCDONE

;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.
HHC20: MOV (SP),R2
    CMP R2,#HHC2+2
    BEQ 1$
```

5115 025400 020227 025246
5116 025404 001402
5117 025406 000137 047302
5118 025412 022626
5119 025414 010237 001236
5120 025420 104253
5121 025422
025422 104412

CMP R2, #HHC2+4
BEQ 1\$
JMP CPSPUR
1\$: CMP (SP)+, (SP)+
MOV R2, \$TMP2
2\$: ERROR +253
HHC DONE: RSETUP

:DDD ADDRESS

: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?).

5122
5123
5130

.ENABL AMW

5131

```
.SBTTL TEST # 54 - SOURCE MODES, MODE 2 GR7 (FL=1), TEST
:*****
:TEST 54 SOURCE MODES, MODE 2 GR7 (FL=1), TEST
:
: THIS IS A TEST OF THE LDCLD WITH
: IMMEDIATE ADDRESSING MODE
:
:*****
```

```
025424 000004
5132
5133 025426
025426 104413
5134 025430 012737 025454 001236
5135 025436 012737 025526 000004
5136 025444 012700 000300
5137 025450 170100
5138 025452 005001
5139
5140 025454 177027
5141 025456 005201
5142 025460 005201
5143 025462 005201
5144 025464 005201
5145
5146 025466 020127 000003
5147 025472 001421
5148
5149
5150
5151 025474 012704 025460
5152 025500 162701 000003
5153 025504 006301
5154 025506 160104
5155 025510 010437 001242
5156 025514 012737 025460 001240
5157 025522 104254
5158 025524 000404
5159
5160
5161
5162 025526 011637 001236
5163 025532 022626
5164 025534 104255
5165
5166 025536
025536 104412
```

```
TS#54: SCOPE
IIC1:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
MOV #IIC2,$TMP2 ;SAVE DATA IN CASE OF ERROR.
MOV #IIC20,ERRVECT ;SET UP FOR TRAPS TO 4.
MOV #300,R0
LDFPS R0
CLR R1
IIC2: LDCLD (R7)+,AC0 ;TEST INSTRUCTION.
5201
5201
5201
5201
CMP R1,#3 ;WAS PC CORRECT AFTER EXECUTION?
BEQ IICDONE ;BR IF YES.
;REPORT PC INCORRECT AFTER INSTRUCTION.
IIC3: MOV #IIC2+4,R4
SUB #3,R1
ASL R1
SUB R1,R4
MOV R4,$TMP4
MOV #IIC2+4,$TMP3
1$: ERROR +254 ;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),$TMP2
CMP (SP)+,(SP)+
1$: ERROR +255 ;BAD CONSTANT ODD ADD
IICDONE:
RSETJP ;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?).
```

5167
5174

5175

```
.SBTTL TEST # 55 - SOURCE MODES, MODE 2 (FL=1), TEST
*****
*TEST 55 SOURCE MODES, MODE 2 (FL=1), TEST
*
* THIS IS A TEST OF THE LDCLD INSTR
* WITH MODE 2.
*****
```

```
025540 000004
5176
5177 025542
025542 104413
5178 025544 013737 025564 001236
5179 025552 012700 000300
5180 025556 170100
5181 025560 012700 025654
5182 025564 177020
5183
5184 025566 170204
5185 025570 012701 025664
5186 025574 012702 000200
5187 025600 170102
5188 025602 174011
5189 025604 020027 025660
5190 025610 001407
5191
5192 025612 010037 001242
5193 025616 012737 025660 001240
5194 025624 104256
5195 025626 000422
5196
5197 025630 022704 000300
5198 025634 001417
5199
5200
5201 025636 010437 001242
5202 025642 012737 000300 001240
5203 025650 104257
5204 025652 000410
5205
5206
5207
5208 025654 001234 067076 054321
5209 025664 177777 177777 177777
5210
5211 025674
025674 104412

TST55: SCOPE
TCC1:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
MOV TCC2,$TMP2 ;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,$TMP4
MOV #TCCBF0+4,$TMP3
1S: ERROR +256 ;BAD CONST
BR TCCDONE

TCC3: CMP #300,R4 ;IS THE FPS CORRECT?
BEQ TCCDONE
;REPORT FPS INCORRECT.
MOV R4,$TMP4
MOV #300,$TMP3
1S: ERROR +257 ;FPS X
BR TCCDONE

;TEST BUFFER AND DATA:
TCCBF0: .WORD 0'234,67076,54321,012345
TCCBF1: -1,-1,-1,-1

TCCDONE:
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?).

5212
5213
5220
```

5221
 5222
 5223
 5224
 5225
 5226 025700
 5227 025702 104413 027032
 5228
 5229 025706 000000 000000
 5230 025712 000000 000000
 5231 025716 177777 177777
 5232 025722 000000
 5233 025724 000004
 5234 025726 177777
 5235 025730 104260
 5236 025732 000401
 5237 025734 104261
 5238 025736
 5239
 5240
 5241 025736
 5242 025740 104413 027032
 5243
 5244 025744 000000 177777
 5245 025750 000000 000000
 5246 025754 004177 177400
 5247 025760 000000
 5248 025762 000004
 5249 025764 177777
 5250 025766 104262
 5251 025770 000401
 5252 025772 104261
 5253 025774
 5254
 5255
 5256 025774
 5257 025776 104413 027032
 5258 026002 000000 000000
 5259 026006 000000 000000
 5260 026012 177777 177777
 5261 026016 000100
 5262 026020 000104
 5263 026022 000004
 5264 026024 104260
 5265 026026 000401
 5266 026030 104263

```

.SBTTL TEST # 56 - LDCIF AND LDCLF TEST
*****
*TEST 56      LDCIF AND LDCLF TEST
*
* THIS IS A TEST OF THE LDCIF AND
* THE LDCLF INSTRUCTIONS.
*
*****
TST56: SCOPE

;ZERO  OPERAND FL=0

KKC1:
  LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
  JSR            PC,LDCFSUB ;GO EXECUTE INSTRUCTION.

1$: .WORD 0,0 ;FSRC OPERAND.
2$: .WORD 0,0 ;EXPECTED RESULT.
3$: .WORD -1,-1 ;ANTICIPATED ERRONEOUS RESULT.
4$: 0 ;FPS BEFORE EXECUTION.
   4 ;FPS AFTER EXECUTION.
   -1 ;ANTICIPATED ERRONEOUS FPS.
5$: ERROR +260 ;REPORT RESULT INCORRECT.
   BR 6$
   ERROR +261 ;REPORT FPS INCORRECT.

6$:
;ZERO  OPERAND FL=0

KKC2:
  LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
  JSR            PC,LDCFSUB ;GO EXECUTE THE INSTRUCTION.

1$: .WORD 0,-1 ;FSRC OPERAND.
2$: .WORD 0,0 ;EXPECTED RESULT.
3$: 4177,177400 ;ANTICIPATED ERRONEOUS RESULT.
4$: 0 ;FPS BEFORE EXECUTION.
   4 ;FPS AFTER EXECUTION.
   -1 ;ANTICIPATED ERRONEOUS FPS.
5$: ERROR +262 ;(BUT FL) ST
   BR 6$ ;277 TO 300
   ERROR +261 ;INTO 301

6$:
;ZERO  OPERAND FL-1

KKC3:
  LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
  JSR            PC,LDCFSUB ;GO EXECUTE THE INSTRUCTION.

1$: .WORD 0,0 ;FSRC OPERAND.
2$: .WORD 0,0 ;EXPECTED RESULT.
3$: .WORD -1,-1 ;ANTICIPATED ERRONEOUS RESULT.
4$: 100 ;FPS BEFORE EXECUTION.
   104 ;FPS AFTER EXECUTION.
   4 ;ANTICIPATED ERRONEOUS FPS.
5$: ERROR +260 ;REPORT RESULT INCORRECT.
   BR 6$
   ERROR +263 ;FL WAS CLR'ED
  
```

5267	026032			6\$:					
5268				:OPERAND	POSITIVE	FL=0			
5269	026032			KKC4:					
	026032	104413		LPERR				:SET UP THE LOOP ON ERROR ADDRESS.	
5270	026034	004737	027032	JSR	PC,LDCFSUB			:GO EXECUTE THE INSTRUCTION.	
5271	026040	040000	000000	1\$:	.WORD	40000,0		:FSRC OPERAND.	
5272	026044	043600	000000	2\$:	.WORD	43600,0		:EXPECTED RESULT.	
5273	026050	047600	000000	3\$:	.WORD	47600,0		:ANTICIPATED ERRONEOUS RESULT.	
5274	026054	000017		4\$:	17			:FPS BEFORE EXECUTION.	
5275	026056	000000			0			:FPS AFTER EXECUTION.	
5276	026060	177777			-1			:ANTICIPATED ERRONEOUS FPS.	
5277	026062	104264		5\$:	ERROR	+264	;ST 107	BAD	
5278	026064	000401			BR	6\$:CONSTANT 231 INSD	
5279	026066	104261			ERROR	+261		:215	
5280	026070			6\$:					
5281				:OPERAND=1,	FL=0				
5282	026070			KKC5:					
	026070	104413		LPERR				:SET UP THE LOOP ON ERROR ADDRESS.	
5283	026072	004737	027032	JSR	PC,LDCFSUB			:GO EXECUTE THE INSTRUCTION.	
5284	026076	000001	000000	1\$:	.WORD	1,0		:FSRC OPERAND.	
5285	026102	040200	000000	2\$:	.WORD	40200,0		:EXPECTED RESULT.	
5286	026106	044200	000000	3\$:	.WORD	44200,0		:ANTICIPATED ERRONEOUS RESULT.	
5287	026112	000017		4\$:	17			:FPS BEFORE EXECUTION.	
5288	026114	000000			0			:FPS AFTER EXECUTION.	
5289	026116	177777			-1			:ANTICIPATED ERRONEOUS FPS.	
5290	026120	104264		5\$:	ERROR	+264		:REPORT RESULT INCORRECT.	
5291	026122	000401			BR	6\$			
5292	026124	104261			ERROR	+261			
5293	026126			6\$:				:REPORT FPS INCORRECT.	
5294									
5295									
5296				:OPERAND-	PATTERN	FL=0			
5297	026126			KKC6:					
	026126	104413		LPERR				:SET UP THE LOOP ON ERROR ADDRESS.	
5298	026130	004737	027032	JSR	PC,LDCFSUB			:GO EXECUTE THE INSTRUCTION.	
5299	026134	000252	000000	1\$:	.WORD	252,0		:FSRC OPERAND.	
5300	026140	042052	000000	2\$:	.WORD	42052,0		:EXPECTED RESULT.	
5301	026144	046052	000000	3\$:	.WORD	46052,0		:ANTICIPATED ERRONEOUS RESULT.	
5302	026150	000000		4\$:	0			:FPS BEFORE EXECUTION.	
5303	026152	000000			0			:FPS AFTER EXECUTION.	
5304	026154	177777			-1			:ANTICIPATED ERRONEOUS FPS.	
5305	026156	104264		5\$:	ERROR	+264		:REPORT RESULT INCORRECT.	
5306	026160	000401			BR	6\$			
5307	026162	104261			ERROR	+261			
5308	026164			6\$:				:REPORT FPS INCORRECT.	
5309									
5310				:OPERAND--40000	FL=0				
5311	026164			KKC7:					
	026164	104413		LPERR				:SET UP THE LOOP ON ERROR ADDRESS.	
5312	026166	004737	027032	JSR	PC,LDCFSUB			:GO EXECUTE THE INSTRUCTION.	
5313	026172	140000	000000	1\$:	.WORD	-40000,0		:FSRC OPERAND.	
5314	026176	143600	000000	2\$:	.WORD	143600,0		:EXPECTED RESULT.	
5315	026202	043600	000000	3\$:	.WORD	43600,0		:ANTICIPATED ERRONEOUS RESULT.	
5316	026206	000007		4\$:	7			:FPS BEFORE EXECUTION.	
5317	026210	000010			10			:FPS AFTER EXECUTION.	
5318	026212	177777			-1			:ANTICIPATED ERRONEOUS FPS.	
5319	026214	104265		5\$:	ERROR	+265		:(SET SIGN) ST 146	

5320	026216	000401		BR	6\$			
5321	026220	104261		ERROR	+261			:REPORT FPS INCORRECT.
5322	026222			6\$:				
5323								
5324				:OPERAND=-1	FL=0			
5325	026222			KKC8:				
	026222	104413		LPERR				:SET UP THE LOOP ON ERROR ADDRESS.
5326	026224	004737	027032	JSR	PC,LDCFSUB			:GO EXECUTE THE INSTRUCTION.
5327	026230	177777	000000	1\$:	.WORD	-1,0		:FSRC OPERAND.
5328	026234	140200	000000	2\$:	.WORD	140200,0		:EXPECTED RESULT.
5329	026240	144000	000400	3\$:	.WORD	144000,400		:ANTICIPATED ERRONEOUS RESULT.
5330	026244	000000		4\$:	0			:FPS BEFORE EXECUTION.
5331	026246	000000			10			:FPS AFTER EXECUTION.
5332	026250	177777			-1			:ANTICIPATED ERRONEOUS FPS.
5333	026252	104266		5\$:	ERROR	+266		:ST 372 TO '52 INTO
5334	026254	000401		BR	6\$:112 (BUF XNBT)
5335	026256	104261		ERROR	+261			:REPORT FPS INCORRECT.
5336	026260			6\$:				
5337								
5338				:OPERAND PATTERN	FL=0			
5339	026260			KKC9:				
	026260	104413		LPERR				:SET UP THE LOOP ON ERROR ADDRESS.
5340	026262	004737	027032	JSR	PC,LDCFSUB			:GO EXECUTE THE INSTRUCTION.
5341	026266	125252	000000	1\$:	.WORD	125252,0		:FSRC OPERAND.
5342	026272	143652	126000	2\$:	.WORD	143652,126000		:EXPECTED RESULT.
5343	026276	043652	126000	3\$:	.WORD	43652,126000		:ANTICIPATED ERRONEOUS RESULT.
5344	026302	000007		4\$:	7			:FPS BEFORE EXECUTION.
5345	026304	000000			10			:FPS AFTER EXECUTION.
5346	026306	177777			-1			:ANTICIPATED ERRONEOUS FPS.
5347	026310	104265		5\$:	ERROR	+265		:REPORT RESULT INCORRECT.
5348	026312	000401		BR	6\$			
5349	026314	104261		ERROR	+261			:REPORT FPS INCORRECT.
5350	026316			6\$:				
5351								
5352				:OPERAND	POS	FL-1		
5353	026316			KKC10:				
	026316	104413		LPERR				:SET UP THE LOOP ON ERROR ADDRESS.
5354	026320	004737	027032	JSR	PC,LDCFSUB			:GO EXECUTE THE INSTRUCTION.
5355	026324	040000	000000	1\$:	.WORD	40000,0		:FSRC OPERAND.
5356	026330	047600	000000	2\$:	.WORD	47600,0		:EXPECTED RESULT.
5357	026334	043600	000000	3\$:	.WORD	43600,0		:ANTICIPATED ERRONEOUS RESULT.
5358	026340	000117		4\$:	117			:FPS BEFORE EXECUTION.
5359	026342	000100			100			:FPS AFTER EXECUTION.
5360	026344	177777			-1			:ANTICIPATED ERRONEOUS FPS.
5361	026346	104267		5\$:	ERROR	+267	:ST 107	CONSTANT
5362	026350	000401		BR	6\$:BAD 237 INST 217
5363	026352	104261		ERROR	+261			:REPORT FPS INCORRECT.
5364	026354			6\$:				
5365								
5366				:OPERAND 1	FL-1			
5367	026354			KKC11:				
	026354	104413		LPERR				:SET UP THE LOOP ON ERROR ADDRESS.
5368	026356	004737	027032	JSR	PC,LDCFSUB			:GO EXECUTE THE INSTRUCTION.
5369	026362	000000	000001	1\$:	.WORD	0,1		:FSRC OPERAND.
5370	026366	040200	000000	2\$:	.WORD	40200,0		:EXPECTED RESULT.
5371	026372	034200	000000	3\$:	.WORD	34200,0		:ANTICIPATED ERRONEOUS RESULT.
5372	026376	000100		4\$:	100			:FPS BEFORE EXECUTION.

```

5373 026400 000100          100          ;FPS AFTER EXECUTION.
5374 026402 177777          -1          ;ANTICIPATED ERRONEOUS FPS.
5375 026404 104267          5$: ERROR +267          ;REPORT RESULT INCORRECT.
5376 026406 000401          BR 6$
5377 026410 104261          ERROR +261          ;REPORT FPS INCORRECT.
5378 026412          6$:
5379
5380          ;OPERAND=          PATTERN FL=1
5381 026412          KKC12:
      026412 104413          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
5382 026414 004737 027032          JSR PC,LDCFSUB ;GO EXECUTE THE INSTRUCTION.
5383 026420 000000 000252          1$: .WORD 0,252 ;FSRC OPERAND.
5384 026424 042052 000000          2$: .WORD 42052,0 ;EXPECTED RESULT.
5385 026430 036052 000000          3$: .WORD 36052,0 ;ANTICIPATED ERRONEOUS RESULT.
5386 026434 000111          4$: 111 ;FPS BEFORE EXECUTION.
5387 026436 000100          100 ;FPS AFTER EXECUTION.
5388 026440 177777          -1 ;ANTICIPATED ERRONEOUS FPS.
5389 026442 104267          5$: ERROR +267          ;REPORT RESULT INCORRECT.
5390 026444 000401          BR 6$
5391 026446 104261          ERROR +261          ;REPORT FPS INCORRECT.
5392 026450          6$:
5393
5394          ;OPERAND=-40000,0          FL=1
5395 026450          KKC13:
      026450 104413          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
5396 026452 004737 027032          JSR PC,LDCFSUB ;GO EXECUTE THE INSTRUCTION.
5397 026456 140000 000000          1$: .WORD -40000,0 ;FSRC OPERAND.
5398 026462 147600 000000          2$: .WORD 147600,0 ;EXPECTED RESULT.
5399 026466 047600 000000          3$: .WORD 47600,0 ;ANTICIPATED ERRONEOUS RESULT.
5400 026472 000107          4$: 107 ;FPS BEFORE EXECUTION.
5401 026474 000110          110 ;FPS AFTER EXECUTION.
5402 026476 177777          -1 ;ANTICIPATED ERRONEOUS FPS.
5403 026500 104265          5$: ERROR +265          ;SET SIGN
5404 026502 000401          BR 6$
5405 026504 104261          ERROR +261          ;REPORT FPS INCORRECT.
5406 026506          6$:
5407
5408          ;OPERAND -1,-1          FL=1
5409 026506          KKC14:
      026506 104413          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
5410 026510 004737 027032          JSR PC,LDCFSUB ;GO EXECUTE THE INSTRUCTION.
5411 026514 177777 177777          1$: .WORD -1,-1 ;FSRC OPERAND.
5412 026520 140200 000000          2$: .WORD 140200,0 ;EXPECTED RESULT.
5413 026524 150000 000000          3$: .WORD 150000,0 ;ANTICIPATED ERRONEOUS RESULT.
5414 026530 000100          4$: 100 ;FPS BEFORE EXECUTION.
5415 026532 000110          110 ;FPS AFTER EXECUTION.
5416 026534 177777          -1 ;ANTICIPATED ERRONEOUS FPS.
5417 026536 104266          5$: ERROR +266          ;(BUT XNBT)
5418 026540 000401          BR 6$
5419 026542 104261          ERROR +261          ;REPORT FPS INCORRECT.
5420 026544          6$:
5421
5422          ;OPERAND=-PATTERN          FL=1,          ROUND MODE
5423 026544          KKC15:
      026544 104413          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
5424 026546 004737 027032          JSR PC,LDCFSUB ;GO EXECUTE THE INSTRUCTION.
5425 026552 125252 125252          1$: .WORD 125252,125252 ;FSRC OPERAND.
  
```

5426	026556	147652	125253	2\$:	.WORD	147652,125253	:	EXPECTED RESULT.
5427	026562	047652	125253	3\$:	.WORD	47652,125253	:	ANTICIPATED ERRONEOUS RESULT.
5428	026566	000105		4\$:	105		:	FPS BEFORE EXECUTION.
5429	026570	000110			110		:	FPS AFTER EXECUTION.
5430	026572	177777			-1		:	ANTICIPATED ERRONEOUS FPS.
5431	026574	104265		5\$:	ERROR	+265	:	REPORT RESULT INCORRECT.
5432	026576	000401			BR	6\$		
5433	026600	104261			ERROR	+261	:	REPORT FPS INCORRECT.
5434	026602			6\$:				
5435								
5436								
5437	026602							
	026602	104413						
5438	026604	004737	027032		LPERR		:	SET UP THE LOOP ON ERROR ADDRESS.
5439	026610	077777	177500		JSR	PC,LDCFSUB	:	GO EXECUTE THE INSTRUCTION.
5440	026614	047777	177777	1\$:	.WORD	77777,177500	:	FSRC OPERAND.
5441	026620	047777	177776	2\$:	.WORD	47777,177777	:	EXPECTED RESULT.
5442	026624	000117		3\$:	.WORD	47777,177776	:	ANTICIPATED ERRONEOUS RESULT.
5443	026626	000100		4\$:	117		:	FPS BEFORE EXECUTION.
5444	026630	177777			100		:	FPS AFTER EXECUTION.
5445	026632	104270			-1		:	ANTICIPATED ERRONEOUS FPS.
5446	026634	000401		5\$:	ERROR	+270	:	ST 631 INTO RND
5447	026636	104261			BR	6\$		
5448	026640				ERROR	+261	:	REPORT FPS INCORRECT.
5449				6\$:				
5450								
5451	026640							
	026640	104413						
5452	026642	004737	027032		LPERR		:	SET UP THE LOOP ON ERROR ADDRESS.
5453	026646	040000	000100		JSR	PC,LDCFSUB	:	GO EXECUTE THE INSTRUCTION.
5454	026652	047600	000001	1\$:	.WORD	40000,100	:	FSRC OPERAND.
5455	026656	047600	000000	2\$:	.WORD	47600,1	:	EXPECTED RESULT.
5456	026662	000102		3\$:	.WORD	47600,0	:	ANTICIPATED ERRONEOUS RESULT.
5457	026664	000100		4\$:	02		:	FPS BEFORE EXECUTION.
5458	026666	177777			100		:	FPS AFTER EXECUTION.
5459	026670	104270			-1		:	ANTICIPATED ERRONEOUS FPS.
5460	026672	000401		5\$:	ERROR	+270	:	REPORT RESULT INCORRECT.
5461	026674	104261			BR	6\$		
5462	026676				ERROR	+261	:	REPORT FPS INCORRECT.
5463				6\$:				
5464								
5465	026676							
	026676	104413						
5466	026700	004737	027032		LPERR		:	SET UP THE LOOP ON ERROR ADDRESS.
5467	026704	040000	000100		JSR	PC,LDCFSUB	:	GO EXECUTE THE INSTRUCTION.
5468	026710	047600	000000	1\$:	.WORD	40000,100	:	FSRC OPERAND.
5469	026714	047600	000001	2\$:	.WORD	47600,0	:	EXPECTED RESULT.
5470	026720	000157		3\$:	.WORD	47600,1	:	ANTICIPATED ERRONEOUS RESULT.
5471	026722	000140		4\$:	157		:	FPS BEFORE EXECUTION.
5472	026724	177777			140		:	FPS AFTER EXECUTION.
5473	026726	104271			-1		:	ANTICIPATED ERRONEOUS FPS.
5474	026730	000401		5\$:	ERROR	+271	:	ST 631 ... INTO TRNC
5475	026732	104261			BR	6\$		
5476	026734				ERROR	+261	:	REPORT FPS INCORRECT.
5477				6\$:				
5478	026734	104413						
	026734							

```

5479 026736 004737 027032          JSR      PC,LDCFSUB      ;GO EXECUTE THE INSTRUCTION.
5480 026742 100000 000000          1$:      .WORD    100000,0      ;FSRC OPERAND.
5481 026746 144000 000000          2$:      .WORD    144000,0      ;EXPECTED RESULT.
5482 026752 143600 000000          3$:      .WORD    143600,0      ;ANTICIPATED ERRONEOUS RESULT.
5483 026756 000007          4$:      7                      ;FPS BEFORE EXECUTION.
5484 026760 000010          ;FPS AFTER EXECUTION.
5485 026762 177777          ;ANTICIPATED ERRONEOUS FPS.
5486 026764 104272          5$:      ERROR    +272          ;ST 630 RH*R14+1
5487 026766 000401          BR        6$
5488 026770 104261          ERROR    +261          ;REPORT FPS INCORRECT.
5489 026772          6$:
5490
5491          ;OPERAND=100000,0      FL=1
5492 026772          KKC20:
      026772 104413          LPERR
5493 026774 004737 027032          JSR      PC,LDCFSUB      ;SET UP THE LOOP ON ERROR ADDRESS.
5494 027000 100000 000000          1$:      .WORD    100000,0      ;GO EXECUTE THE INSTRUCTION.
5495 027004 150000 000000          2$:      .WORD    150000,0      ;FSRC OPERAND.
5496 027010 147600 000000          3$:      .WORD    147600,0      ;EXPECTED RESULT.
5497 027014 000107          4$:      107                   ;ANTICIPATED ERRONEOUS RESULT.
5498 027016 000110          ;FPS BEFORE EXECUTION.
5499 027020 177777          ;FPS AFTER EXECUTION.
5500 027022 104272          5$:      ERROR    +272          ;ANTICIPATED ERRONEOUS FPS.
5501 027024 000401          BR        6$              ;REPORT RESULT INCORRECT.
5502 027026 104261          ERROR    +261          ;REPORT FPS INCORRECT.
5503 027030 000506          6$:      BR        KKCDONE
5504
5505          ;THIS SUBROUTINE, LDCFSUB, IS USED TO SET UP THE OPERANDS, EXECUTE
5506          ;THE LDCIF OR LDCLF INSTRUCTION AND CHECK THE RESULTS. A CALL
5507          ;TO IT IS MADE THUS:
5508          :
5509          :
5510          :          JSR      PC,LDCFSUB
5511          :          ACARG:  .WORD    X,X          ;AC OPERAND
5512          :          RES:     .WORD    X,X          ;EXPECTED RESULT
5513          :          ERRES:  .WORD    X,X          ;ERROR RESULT
5514          :          FPSB:   .WORD    X            ;FPS BEFORE EXECUTION
5515          :          FPSA:   .WORD    X            ;FPS AFTER EXECUTION
5516          :          ERFPS:  .WORD    X            ;ERROR FPS
5517          :          ERR1:   ERROR    +X          ;DATA ERROR
5518          :          BR        CONT
5519          :          ERR2:   ERROR    +X          ;FPS ERROR
5520          :          CONT:   ;RETURN ADDRESS
5521          :
5522          :THE OPERANDS ARE SET UP (USING ACO AS THE ACCUMULATOR). THEN
5523          :THE LDCIF OR LDCLF INSTRUCTION IS EXECUTED.
5524          :THE RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
5525          :COMPARED WITH FPSA IF THIS TOO IS CORRECT LDCFSUB RETURNS CONTROL
5526          :TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD LDCFSUB WILL
5527          :COMPARE IT TO ERROR FPS. IF THIS MATCHES THEN LDCFSUB WILL RETURN
5528          :TO THE ERROR CALL AT ERR2, OTHERWISE LDCFSUB ITSELF
5529          :REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE
5530          :LDCIF OR LDCLF IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
5531          :ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
5532          :THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN LDCFSUB
5533          :WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR1. OTHERWISE THE
5534          :RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND LDCFSUB
          :REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.
  
```



```

5535
5536 027032 012601          LDCFSUB:  MOV      (SP)+,R1      ;GET A POINTER TO THE ARGUMENTS.
5537 027034 016100 000014  MOV      14(R1),R0      ;SET THE FPS.
5538 027040 170100          LDFPS   R0
5539 027042 012737 027052 001236  MOV      #1,$TMP2
5540 027050 010100          MOV      R1,R0
5541
5542 027052 177010          1$:     LDCIF   (R0),AC0      ;TEST INSTRUCTION LDCIF OR LDCLF.
5543
5544 027054 170204          STFPS  R4              ;GET FPS.
5545 027056 012700 027236  MOV      #LDCT,R0      ;GET THE RESULT.
5546 027062 012702 000200  MOV      #200,R2
5547 027066 170102          LDFPS  R2
5548 027070 174010          STD    AC0,(R0)
5549
5550 027072 012702 027236  MOV      #LDCT,R2      ;SEE IF THE RESULT WAS CORRECT.
5551 027076 010237 001242  MCV     R2,$TMP4
5552 027102 010137 001240  MOV     R1,$TMP3
5553 027106 010103          MOV     R1,R3
5554 027110 062703 000004  ADD     #4,R3
5555 027114 010337 001244  MOV     R3,$TMP5
5556 027120 010437 001250  MOV     R4,$TMP7
5557 027124 016137 000016 001252  MOV     16(R1),$TMP10
5558 027132 010100          MOV     R1,R0
5559 027134 062700 000004  ADD     #4,R0
5560 027140 012703 000002  MOV     #2,R3
5561 027144 022022          2$:     CMP     (R0)+,(R2)+
5562 027146 001006          BNE    10$
5563 027150 077303          SOB    R3,2$
5564
5565 027152 026104 000016  CMP     16(R1),R4      ;SEE IF THE FPS WAS CORRECT.
5566 027156 001020          BNE    15$
5567 027160 000161 000030  3$:     JMP     30(R1)      ;BR IF INCORRECT.
5568
5569          ;RESULT IN CORRECT SO SEE IF THE FAILURE WAS ANTICIPATED.
5570 027164 012702 027236  10$:    MOV     #LDCT,R2
5571 027170 010100          MOV     R1,R0
5572 027172 062700 000010  ADD     #10,R0
5573 027176 012703 000002  MOV     #2,R3
5574 027202 022022          11$:    CMP     (R0)+,(R2)+
5575 027204 001003          BNE    13$
5576 027206 077303          SOB    R3,11$
5577 027210 000161 000022  JMP     22(R1)
5578
5579          ;THE FAILURE WAS NOT ANTICIPATED SO REPORT THE ERROR HERE.
5580 027214          13$:
5581
5582 027214 104260          14$:    ERROR  +260      ;BAD RES
5583 027216 000760          BR     3$
5584
5585
5586          ;THE FPS WAS INCORRECT SO SEE IF IT WAS ANTICIPATED.
5587 027220 026104 000020  15$:    CMP     20(R1),R4
5588 027224 001002          BNE    16$
5589 027226 000161 000026  JMP     26(R1)
5590
5591          ;FPS ERROR NOT ANTICIPATED SO REPORT IT HERE.
  
```

```
5592 027232          16$:  
5593 027232 104261  17$:  ERROR  +261  
5594 027234 000751          BR    3$      ;BAD FPS  
5595  
5596          ;DATA BUFFER:  
5597 027236 000000 000000 000000 LDCT: .WORD  C.0.0.0  
5598  
5599 027246          KKCDONE:  
      027246 104412          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?).
```

5605


```

027664 104413
5706 027666 004737 030050 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
5707 027672 077777 177777 JSR PC,LDCDSUB ;GO EXECUTE THE INSTRUCTION.
5708 027676 047777 177777 177000 1$: .WORD 77777,177777 ;FSRC OPERAND.
5709 027706 177777 177777 177000 2$: .WORD 47777,177777,177000,0 ;EXPECTED RESULT.
5710 027716 000317 3$: .WORD -1,-1,-1,-1 ;ANTICIPATED ERRONEOUS RESULT.
5711 027720 000300 4$: 317 ;FPS BEFORE EXECUTION.
5712 027722 177777 -1 ;FPS AFTER EXECUTION.
5713 027724 104273 5$: ERROR +273 ;ANTICIPATED ERRONEOUS FPS.
5714 027726 000401 BR 6$ ;REPORT RESULT INCORRECT.
5715 027730 104274 ERROR +274 ;REPORT FPS INCORRECT.
5716 027732 6$:
5717
5718 ;OPERAND -PATTERN FL-1 FD=1
5719
5720 027732 LLC9:
027732 104413 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
5721 027734 004737 030050 JSR PC,LDCDSUB ;GO EXECUTE THE INSTRUCTION.
5722 027740 177777 177526 1$: .WORD -1,-252 ;FSRC OPERAND.
5723 027744 142052 000000 000000 2$: .WORD 142052,0,0,0 ;EXPECTED RESULT.
5724 027754 136052 000000 000000 3$: .WORD 136052,0,0,0 ;ANTICIPATED ERRONEOUS RESULT.
5725 027764 000307 4$: 307 ;FPS BEFORE EXECUTION.
5726 027766 000310 310 ;FPS AFTER EXECUTION.
5727 027770 177777 -1 ;ANTICIPATED ERRONEOUS FPS.
5728 027772 104300 5$: ERROR +300 ;REPORT RESULT INCORRECT.
5729 027774 000401 BR 6$
5730 027776 104274 ERROR +274 ;REPORT FPS INCORRECT.
5731 030000 6$:
5732
5733 ;OPERAND=PATTERN FL=1 FD=1 FT=1
5734 030000 LLC10:
030000 104413 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
5735 030002 004737 030050 JSR PC,LDCDSUB ;GO EXECUTE THE INSTRUCTION.
5736 030006 012345 067012 1$: .WORD 12345,67012 ;FSRC OPERAND.
5737 030012 047247 025560 050000 2$: .WORD 47247,025560,050000,0 ;EXPECTED RESULT.
5738 030022 177777 177777 177777 3$: .WORD -1,-1,-1,-1 ;ANTICIPATED ERRONEOUS RESULT.
5739 030032 000352 4$: 352 ;FPS BEFORE EXECUTION.
5740 030034 000340 340 ;FPS AFTER EXECUTION.
5741 030036 177777 -1 ;ANTICIPATED ERRONEOUS FPS.
5742 030040 104273 5$: ERROR +273 ;REPORT RESULT INCORRECT.
5743 030042 000401 BR 6$
5744 030044 104274 ERROR +274 ;REPORT FPS INCORRECT.
5745 030046 000502 6$: BR LLCDONE
5746
5747 ;THIS SUBROUTINE, LDCDSUB, IS USED TO SET UP THE OPERANDS, EXECUTE
5748 ;THE LDCID OR LDCLD INSTRUCTION AND CHECK THE RESULTS. A CALL
5749 ;TO IT IS MADE THUS:
5750
5751 JSR PC,LDCDSUB
5752 ACARG: .WORD X,X ;AC OPERAND
5753 RES: .WORD X,X,X,X ;EXPECTED RESULT
5754 ERRES: .WORD X,X,X,X ;ERROR RESULT
5755 FPSB: .WORD X ;FPS BEFORE EXECUTION
5756 FPSA: .WORD X ;FPS AFTER EXECUTION
5757 ERFPS: .WORD X ;ERROR FPS.
5758 ERR1: ERROR +X ;DATA ERROR.
5759 BR CONT
  
```

```

5760          :          ERR2:  ERROR  +X          ;FPS ERROR.
5761          :          CONT:                    ;RETURN ADDRESS
5762          :
5763          :THE OPERANDS ARE SET UP (USING ACO AS THE ACCUMULATOR). THEN
5764          :THE LDCID OR LDCLD INSTRUCTION IS EXECUTED.
5765          :THE RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
5766          :COMPARED WITH FPSA IF THIS TOO IS CORRECT LDCDSUB RETURNS CONTROL
5767          :TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD LDCDSUB
5768          :COMPARE IT TO ERROR FPS. IF THIS MATCHES THEN LDCDSUB WILL RETURN
5769          :TO THE ERROR CALL AT ERR2, OTHERWISE LDCDSUB ITSELF
5770          :REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE
5771          :LDCID OR LDCLD IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
5772          :ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
5773          :THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN LDCDSUB
5774          :WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR1. OTHERWISE THE
5775          :RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND LDCDSUB WILL
5776          :REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.
5777
5778 030050 012601 LDCDSUB:      MOV      (SP)+,R1      ;GET A POINTER TO THE ARGUMENTS.
5779 030052 016100      MOV      24(R1),R0      ;SET THE FPS.
5780 030056 170100      LDFPS   R0
5781 030060 012737 030070 001236      MOV      #1$,STMP2
5782 030066 010100      MOV      R1,R0
5783 030070 177010      1$:      LDCID   (R0),ACO      ;TEST INSTRUCTION, LDCID OR LDCLD.
5784
5785 030072 170204      STFPS   R4      ;GET FPS.
5786 030074 012700 027236      MOV      #LDCT,R0      ;GET THE RESULT.
5787 030100 012702 000200      MOV      #200,R2
5788 030104 170102      LDFPS   R2
5789 030106 174010      STD      ACO,(R0)
5790
5791          :SEE IF THE RESULT IS CORRECT.
5792 030110 012702 027236      MOV      #LDCT,R2
5793 030114 010237 001242      MOV      R2,$TMP4
5794 030120 010137 001240      MOV      R1,$TMP3
5795 030124 010103      MOV      R1,R3
5796 030126 062703 000004      ADD      #4,R3
5797 030132 010337 001244      MOV      R3,$TMP5
5798 030136 010437 001250      MOV      R4,$TMP7
5799 030142 016137 000026 001252      MOV      26(R1),$TMP10
5800 030150 010100      MOV      R1,R0
5801 030152 062700 000004      ADD      #4,R0
5802 030156 012703 000002      MOV      #2,R3
5803 030162 022022      2$:      CMP      (R0)+,(R2)+
5804 030164 001006      BNE     10$      ;BR IF INCORRECT.
5805 030166 077303      SOB     R3,2$
5806
5807 030170 026104 000026      CMP      26(R1),R4      ;IS THE FPS CORRECT?
5808 030174 001020      BNE     15$      ;BR IF INCORRECT.
5809 030176 000161 000040      3$:      JMP      40(R1)      ;RETURN.
5810
5811          :THE RESULT WAS INCORRECT SO SEE IF THE ERROR WAS ANTICIPATED.
5812 030202 012702 027236      10$:     MOV      #LDCT,R2
5813 030206 010100      MOV      R1,R0
5814 030210 062700 000014      ADD      #14,R0
5815 030214 012703 000002      MOV      #2,R3
5816 030220 022022      11$:     CMP      (R0)+,(R2)+
  
```

5817 030222 001003
5818 030224 077303
5819 030226 000161 000032
5820 030232
5821
5822 030232 104273
5823 030234 000760
5824
5825
5826 030236 026104 000030
5827 030242 001002
5828 030244 000161 000036
5829
5830 030250
5831
5832 030250 104274
5833 030252 000751
5834
5835 030254
030254 104412

BNE 13\$
SOB R3,11\$
JMP 32(R1)
13\$:
:ERROR NOT ANTICIPATED SO REPORT RESULT INCORRECT HERE.
14\$: ERROR +273 ;BAD RES
BR 3\$
:THE FPS WAS INCORRECT. SEE IF FAILURE WAS ANTICIPATED.
15\$: CMP 30(R1),R4
BNE 16\$
JMP 36(R1)
:FPS ERROR WAS NOT ANTICIPATED SO REPORT FAILURE HERE.
16\$:
17\$: ERROR +274 ;BAD FPS
BR 3\$

LLCDONE:
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?).

5836
5845

5846

```
.SBTTL TEST # 60 - LDEXP TEST  
:*****  
:TEST 60 LDEXP TEST  
:  
:* THIS IS A TEST OF THE LDEXP INST  
:* A SUBROUTINE IS USED TO SET UP  
:* OPERANDS, EXECUTE THE LDEXP INST AND  
:* CHECK THE RESULTS.  
:  
:*****
```

5847 030256 000004

```
TST60: SCOPE  
:NON-ZERO RES. VALID EXPON-210 (EXCESS 200)-10
```

5848

5849 030260

```
MMC1: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.  
JSR PC,LDXSUB ;GO EXECUTE THE INSTRUCTION.
```

5850 030260 104413 031622
5851 030266 012345 067012 034567 1\$: .WORD 12345,67012,34567,012345 ;ACO OPERAND.
5852 030276 000010 2\$: .WORD 10 ;EXPONENT OPERAND.
5853 030300 042145 067012 034567 3\$: .WORD 42145,67012,34567,012345 ;EXPECTED RESULT.
5854 030310 002145 067012 034567 4\$: .WORD 2145,67012,34567,012345 ;ANTICIPATED ERRONEOUS RESULT.
5855 030320 047217 5\$: 47217 ;FPS BEFORE EXECUTION.
5856 030322 047200 47200 ;FPS AFTER EXECUTION.
5857 030324 147200 147200 ;ANTICIPATED ERRONEOUS FPS.
5858 030326 177777 -1 ;EXPECTED FEC.
5859 030330 104304 6\$: ERROR +304 ;E12+E12+200 BAD
5860 030332 000400 BR 7\$;ST 624
5861 030334 104305 7\$: ERROR +305 ;REPORT FPS INCORRECT.
 ;ST 625 INTO 304

5862

5863

5864 030336

```
:NON-ZERO RES NEG.
```

```
MMC2: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.  
JSR PC,LDXSUB ;EXPON=377
```

5865 030340 004737 031622
5866 030344 123456 070123 045670 1\$: .WORD 123456,70123,45670,123456 ;ACO OPERAND.
5867 030354 000177 2\$: .WORD 177 ;EXPONENT OPERAND.
5868 030356 177656 070123 045670 3\$: .WORD 177656,70123,45670,123456 ;EXPECTED RESULT.
5869 030366 137656 070123 045670 4\$: .WORD 137656,70123,45670,123456 ;ANTICIPATED ERRONEOUS RESULT.
5870 030376 047207 5\$: 47207 ;FPS BEFORE EXECUTION.
5871 030400 047210 47210 ;FPS AFTER EXECUTION.
5872 030402 147210 147210 ;ANTICIPATED ERRONEOUS FPS.
5873 030404 177777 -1 ;EXPECTED FEC.
5874 030406 104304 6\$: ERROR +304 ;REPORT RESULT INCORRECT.
5875 030410 000401 BR 7\$
5876 030412 104305 7\$: ERROR +305 ;REPORT FPS INCORRECT.

5878

5879

5880 030414

```
:NON-ZERO RES, EXP 256=(56)REAL
```

```
MMC3: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.  
JSR PC,LDXSUB ;GO EXECUTE THE INSTRUCTION.
```

5881 030414 104413 031622
5882 030416 004737 057645 043323 1\$: .WORD 73261,057645,43323,101760 ;ACO OPERAND.
5883 030422 073261 057645 043323 2\$: .WORD 56 ;EXPONENT OPERAND.
5884 030432 000056 057645 043323 3\$: .WORD 53461,057645,43323,101760 ;EXPECTED RESULT.
5885 030434 053461 177777 177777 4\$: .WORD -1,-1,-1,-1 ;ANTICIPATED ERRONEOUS RESULT.
5886 030444 177777 47200 5\$: 47200 ;FPS BEFORE EXECUTION.
5887 030454 047200 47200 ;FPS AFTER EXECUTION.
5888 030460 147200 147200 ;ANTICIPATED ERRONEOUS FPS.
5889 030462 177777 -1 ;EXPECTED FEC.


```
5944 ;EXP-0 (EXCESS 200)=-200 (OCT),POS FRAC, FIU=0
5945
5946 030704 MMC7:
      030704 104413 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
5947 030706 004737 031622 JSR PC,LDXSUB ;GO EXECUTE THE INSTRUCTION.
5948 030712 051525 035455 005675 1$: .WORD 51525,35455,5675,05152 ;ACO OPERAND.
5949 030722 177600 2$: .WORD -200 ;EXPONENT OPERAND.
5950 030724 000000 000000 000000 3$: .WORD 0,0,0,0 ;EXPECTED RESULT.
5951 030734 000125 035455 005675 4$: .WORD 00125,35455,5675,05152 ;ANTICIPATED ERRONEOUS RESULT.
5952 030744 045200 45200 ;FPS BEFORE EXECUTION.
5953 030746 045204 45204 ;FPS AFTER EXECUTION.
5954 030750 145204 145204 ;ANTICIPATED ERRONEOUS FPS.
5955 030752 177777 -1 ;EXPECTED FEC.
5956 030754 104311 6$: ERROR +311 ;(BUT FIU) ST 264 X ;REPORT RESULT INCORRECT.
5957 030756 000401 BR 7$
5958 030760 104302 ERROR +302 ;REPORT FPS INCORRECT.
5959 030762 7$:
5960
5961 ;EXP--1405 (EXCESS 200)=-1605 (OCT), FIU=1
5962 030762 MMC8:
      030762 104413 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
5963 030764 004737 031622 JSR PC,LDXSUB ;GO EXECUTE THE INSTRUCTION.
5964 030770 061626 062636 046566 1$: .WORD 61626,62636,46566,67606 ;ACO OPERAND.
5965 031000 176173 2$: .WORD -1605 ;EXPONENT OPERAND.
5966 031002 076626 062636 046566 3$: .WORD 76626,62636,46566,67606 ;EXPECTED RESULT.
5967 031012 000000 000000 000000 4$: .WORD 0,0,0,0 ;ANTICIPATED ERRONEOUS RESULT.
5968 031022 042200 5$: 42200 ;FPS BEFORE EXECUTION.
5969 031024 142200 142200 ;FPS AFTER EXECUTION.
5970 031026 042204 42204 ;ANTICIPATED ERRONEOUS FPS.
5971 031030 000012 12 ;EXPECTED FEC.
5972 031032 104312 6$: ERROR +312 ;(BUT EZBT) ST 544 TO 704 INTO 504
5973 031034 000401 BR 7$
5974 031036 104302 ERROR +302 ;REPORT FPS INCORRECT.
5975 031040 7$:
5976 ;EXP=-17416 (EXCESS 200)--17616 (OCT), FIU=0
5977 031040 MMC9:
      031040 104413 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
5978 031042 004737 031622 JSR PC,LDXSUB ;GO EXECUTE THE INSTRUCTION.
5979 031046 071727 037475 076777 1$: .WORD 71727,37475,76777,17273 ;ACO OPERAND.
5980 031056 160162 2$: .WORD -17616 ;EXPONENT OPERAND.
5981 031060 000000 000000 000000 3$: .WORD 0,0,0,0 ;EXPECTED RESULT.
5982 031070 074527 037475 076777 4$: .WORD 74527,37475,76777,17273 ;ANTICIPATED ERRONEOUS RESULT.
5983 031100 045200 5$: 45200 ;FPS BEFORE EXECUTION.
5984 031102 045204 45204 ;FPS AFTER EXECUTION.
5985 031104 145200 145200 ;ANTICIPATED ERRONEOUS FPS.
5986 031106 177777 -1 ;EXPECTED FEC.
5987 031110 104313 6$: ERROR +313 ;(BUT FIU) ST 504
5988 031112 000401 BR 7$
5989 031114 104302 ERROR +302 ;REPORT FPS INCORRECT.
5990 031116 7$:
5991
5992 ;EXP -1601 (EXCESS 200)=-2001 (OCT), FIU=1
5993 031116 MMC10:
      031116 104413 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
5994 031120 004737 031622 JSR PC,LDXSUB ;GO EXECUTE THE INSTRUCTION.
5995 031124 001020 030405 006070 1$: .WORD 01020,30405,06070,00102 ;ACO OPERAND.
5996 031134 177777 2$: .WORD -2001 ;EXPONENT OPERAND.
```

5997	031136	037620	030405	006070	3\$:	.WORD	37620,30405,06070,00102	:EXPECTED RESULT.
5998	031146	000000	000000	000000	4\$:	.WORD	0,0,0,0	:ANTICIPATED ERRONEOUS RESULT.
5999	031156	042200			5\$:	42200		:FPS BEFORE EXECUTION.
6000	031160	142200				142200		:FPS AFTER EXECUTION.
6001	031162	042204				42204		:ANTICIPATED ERRONEOUS FPS.
6002	031164	000012				12		:EXPECTED FEC.
6003	031166	104312			6\$:	ERROR	+312	: (BUT FIV) ST 504
6004	031170	000401				BR	7\$	
6005	031172	104302				ERROR	+302	:REPORT FPS INCORRECT.
6006	031174				7\$:			
6007								
6008								
6009	031174							:EXP-1206 (EXCESS 200)=1006 (OCT) FIV=1
								MMC11:
	031174	104413				LPERR		:SET UP THE LOOP ON ERROR ADDRESS.
6010	031176	0047	031622			JSR	PC,LDXSUB	:GO EXECUTE THE INSTRUCTION.
6011	031202	012131	014151	016171	1\$:	.WORD	12131,14151,16171,10111	:ACO OPERAND.
6012	031212	001006			2\$:	.WORD	1006	:EXPONENT OPERAND.
6013	031214	041531	014151	016171	3\$:	.WORD	41531,14151,16171,10111	:EXPECTED RESULT.
6014	031224	000000	000000	000000	4\$:	.WORD	0,0,0,0	:ANTICIPATED ERRONEOUS RESULT.
6015	031234	041200			5\$:	41200		:FPS BEFORE EXECUTION.
6016	031236	141202				141202		:FPS AFTER EXECUTION.
6017	031240	041204				41204		:ANTICIPATED ERRONEOUS FPS.
6018	031242	000010				10		:EXPECTED FEC.
6019	031244	104314			6\$:	ERROR	+314	: (BUT FIV) ST 104
6020	031246	000401				BR	7\$	
6021	031250	104302				ERROR	+302	:REPORT FPS INCORRECT.
6022	031252				7\$:			
6023								
6024								
6025	031252							:EXP-16315 (EXCESS 200)=16115 (OCT) FIV=0
								MMC12:
	031252	104413				LPERR		:SET UP THE LOOP ON ERROR ADDRESS.
6026	031254	004737	031622			JSR	PC,LDXSUB	:GO EXECUTE THE INSTRUCTION.
6027	031260	027262	025242	023222	1\$:	.WORD	27262,25242,23222,21202	:ACO OPERAND.
6028	031270	016115			2\$:	.WORD	16115	:EXPONENT OPERAND.
6029	031272	000000	000000	000000	3\$:	.WORD	0,0,0,0	:EXPECTED RESULT.
6030	031302	063262	025242	023222	4\$:	.WORD	63262,25242,23222,21202	:ANTICIPATED ERRONEOUS RESULT.
6031	031312	046200			5\$:	46200		:FPS BEFORE EXECUTION.
6032	031314	046206				46206		:FPS AFTER EXECUTION.
6033	031316	146202				146202		:ANTICIPATED ERRONEOUS FPS.
6034	031320	177777				-1		:EXPECTED FEC.
6035	031322	104315			6\$:	ERROR	+315	: (BUT FIV) ST 104
6036	031324	000401				BR	7\$	
6037	031326	104302				ERROR	+302	:REPORT FPS INCORRECT.
6038	031330				7\$:			
6039								
6040								
6041								
6042	031330							:EXP 11011 (EXCESS 200)=10611 (OCT) FIV=1
								MMC13:
	031330	104413				LPERR		:SET UP THE LOOP ON ERROR ADDRESS.
6043	031332	004737	031622			JSR	PC,LDXSUB	:GO EXECUTE THE INSTRUCTION.
6044	031336	030313	032333	034353	1\$:	.WORD	30313,32333,34353,36373	:ACO OPERAND.
6045	031346	010611			2\$:	.WORD	10611	:EXPONENT OPERAND.
6046	031350	002313	032333	034353	3\$:	.WORD	2313,32333,34353,36373	:EXPECTED RESULT.
6047	031360	000000	000000	000000	4\$:	.WORD	0,0,0,0	:ANTICIPATED ERRONEOUS RESULT.
6048	031370	041200			5\$:	41200		:FPS BEFORE EXECUTION.
6049	031372	141202				141202		:FPS AFTER EXECUTION.
6050	031374	041204				41204		:ANTICIPATED ERRONEOUS FPS.

6105 031616 104302
6106 031620
6107 031620 000540
6108
6109
6110
6111
6112
6113
6114
6115
6116
6117
6118
6119
6120
6121
6122
6123
6124
6125
6126
6127
6128
6129
6130
6131
6132
6133
6134
6135
6136
6137
6138
6139
6140
6141
6142 031622 012601
6143 031624 012700 000200
6144 031630 170100
6145 031632 010100
6146 031634 172410
6147 031636 012737 031660 001236
6148 031644 016100 000032
6149 031650 170100
6150 031652 010100
6151 031654 062700 000010
6152
6153 031660 176410
6154
6155 031662 170204
6156 031664 170305
6157 031666 012700 000200
6158 031672 170100
6159 031674 012700 032112
6160 031700 174010
6161 031702 010437 001250

```

7$:      ERROR  +302          ;REPORT FPS INCORRECT.
        BR      MMCDONE

;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  +X              ;DATA ERROR.
        BR      CONT
        ERR2:   ERROR  +X              ;FPS ERROR.
        CONT:   CONT                    ;RETURN ADDRESS

;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),ACO
        MOV     #1$, $TMP2
        MOV     32(R1),R0        ;SET UP THE FPS.
        LDFPS  R0
        MOV     R1,R0
        ADD     #10,R0

1$:      LDEXP  (R0),ACO          ;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,$TMP7
    
```

```

6162 031706 016137 000034 001252      MOV      34(R1), $TMP10
6163 031714 010537 001254      MOV      R5, $TMP11
6164 031720 016137 000040 001256      MOV      40(R1), $TMP12
6165 031726 010102      MOV      R1, R2
6166 031730 010237 001240      MOV      R2, $TMP3
6167 031734 062702 000010      ADD      #10, R2
6168 031740 011237 001242      MOV      (R2), $TMP4
6169 031744 062702 000002      ADD      #2, R2
6170 031750 010237 001244      MOV      R2, $TMP5
6171 031754 012737 032112 001246      MOV      #LDXT, $TMP6
6172 031762 012702 032112      MOV      #LDXT, R2      ;SEE IF THE RESULT WAS CORRECT.
6173 031766 010103      MOV      R1, R3
6174 031770 062703 000012      ADD      #12, R3
6175 031774 012700 000004      MOV      #4, R0
6176 032000 022223      2$:      CMP      (R2)+, (R3)+
6177 032002 001014      BNE      10$      ;BRANCH IF NOT CORRECT.
6178 032004 077003      SCB      R0, 2$
6179 032006 020461 000034      CMP      R4, 34(R1)      ;SEE IF THE FPS WAS CORRECT.
6180 032012 001026      BNE      15$      ;BRANCH IF NOT CORRECT.
6181 032014 005761 000034      TST      34(R1)
6182 032020 100003      BPL      3$
6183 032022 020561 000040      CMP      R5, 40(R1)      ;SEE IF THE FEC WAS CORRECT.
6184 032026 001027      BNE      20$      ;BRANCH IF NOT CORRECT.
6185
6186 032030 000161 000050      3$:      JMP      50(R1)      ;RETURN.
6187
6188      ;THE RESULT WAS INCORRECT SO SEE IF THE FAILURE WAS ANTICIPATED.
6189 032034 012702 032112      10$:     MOV      #LDXT, R2
6190 032040 010103      MOV      R1, R3
6191 032042 062703 000022      ADD      #22, R3
6192 032046 012700 000004      MOV      #4, R0
6193 032052 022223      11$:     CMP      (R2)+, (R3)+
6194 032054 001003      BNE      12$
6195 032056 077003      SOB      R0, 11$
6196 032060 000161 000042      JMP      42(R1)
6197
6198      ;THE ERROR WAS NOT ANTICIPATED SO REPORT IT HERE.
6199 032064      12$:
6200 032064 104301      13$:     ERROR   +301      ;BAD RES
6201 032066 000760      BR      3$
6202
6203      ;SEE IF THE FPS ERROR WAS ANTICIPATED.
6204 032070 026104 000036      15$:     CMP      36(R1), R4
6205 032074 001002      BNE      16$
6206 032076 000161 000046      JMP      46(R1)
6207 032102      16$:
6208      ;THE FPS WAS NOT ANTICIPATED SO REPORT IT HERE.
6209 032102 104302      17$:     ERROR   +302      ;BAD FPS
6210 032104 000751      BR      3$      ;BUT EZBTY8
6211      ;ST 063
6212
6213 032106      20$:
6214      ;REPORT FEC INCORRECT.
6215 032106 104303      21$:     ERROR   +303      ;BAD FEC
6216 032110 000747      BR      3$
6217
6218      ;DATA BUFFER:

```

6219 032112 000000 000000 000000 LDXT: .WORD 0,0,0,0

6220

6221 032122
032122 104412

MMCDONE:
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?).

6222
6223
6230

6231

```
.SBTTL TEST # 61 - DESTINATION MODES, MODE 1 (FL=0), TEST
:*****
:TEST 61 DESTINATION MODES, MODE 1 (FL 0), TEST
:
: THIS IS A TEST OF DESTINATION MODE 1 USING
: THE STFPS INSTRUCTION
:
:*****
TST61: SCOPE
```

```
032124 000004
6232
6233
6234 032126
032126 104413
6235 032130 012700 032226
6236 032134 012701 000006
6237 032140 012720 177777
6238 032144 077103
6239 032146 012700 102345
6240 032152 012737 032174 001236
6241 032160 012737 032326 000004
6242 032166 170100
6243 032170 012700 032232
6244
6245 032174 170210
6246 032176 020027 032232
6247 032202 001017
6248 032204 023727 032232 102345
6249 032212 001023
6250 032214 023727 032234 177777
6251 032222 001030
6252 032224 000453
6253
6254
6255 032226 177777 177777
6256 032232 177777 177777 177777
6257
6258
6259 032242 010037 001242
6260 032246 012737 032232 001240
6261 032254
032254 104377
032256 000001
6262
6263 032260 000435
6264
6265
6266 032262 012737 102345 001240
6267 032270 013737 032232 001242
6268 032276
032276 104377
032300 000002
6269
6270 032302 000424
6271
6272
6273
6274 032304 012737 177777 001240
```

```
NMC1:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
MOV #NMC1B0,R0 ;SET UP THE DATA BUFFER.
MOV #6,R1
1$: MOV #-1,(R0)+
SGB R1,1$
MOV #102345,R0
MOV #NMC2,$TMP2
MOV #NMC25,ERRVECT ;SET UP FOR TRAPS TO 4.
LDFPS R0 ;SET UP FPS.
MOV #NMC1B1,R0

NMC2: STFPS (R0) ;TEST INSTRUCTION.
CMP R0,#NMC1B1 ;IS R0 CORRECT?
BNE NMC10 ;BRANCH IF NOT CORRECT.
CMP NMC1B1,#102345 ;IS RESULT CORRECT?
BNE NMC15 ;BRANCH IF NOT CORRECT.
CMP NMC1B1+2,#-1 ;IS THE RESULT CORRECT?
BNE NMC20 ;BRANCH IF NOT CORRECT.
BR NMCDONE

;TEST DATA BUFFER:
NMC1B0: .WORD -1,-1
NMC1B1: .WORD -1,-1,-1,-1

;REPORT RC INCORRECT.
NMC10: MOV R0,$TMP4
MOV #NMC1B1,$TMP3
1$: ERROR +377
.WORD 1
;RO BAD (BUT
; FDST)X

;REPORT RESULT INCORRECT.
NMC15: MOV #102345,$TMP3
MOV NMC1B1,$TMP4
1$: ERROR +377
.WORD 2
;BAD DATA

;REPORT RESULT INCORRECT.
NMC20: MOV #-1,$TMP3
```

; ST 634

;BAD DATA


```

6275 032312 013737 032234 001242      MOV      NNC1B1+2,$TMP4
6276 032320      1$:      ERROR  +377
      032320 104377      .WORD  3
      032322 000003
6277
6278 032324 000413      BR       NNCDONE      ;(BUT GR7,FL)
6279
6280
6281
6282
6283      ;IF A TRAP TO VECTOR 4 OCCURS COME HERE TO SEE IF THE TRAP OCCURRED
6284 032326 011604      ;DURING EXECUTION OF THE FPP INSTRUCTION BEING TESTED, IF NOT GO
6285 032330 020427 032176      ;TO THE SPURIOUS TRAP TO 4 HANDLER.
6286 032334 001402      NNC25:  MOV      (SP),R4
6287 032336 000137 047302      CMP      R4,#NNC2+2
6288
6289 032342 011637 001236      1$:      MCV      (SP),$TMP2
6290 032346 022626      CMP      (SP)+,(SP)+
6291 032350      2$:      ERROR  +377
      032350 104377      .WORD  4
      032352 000004
6292
6293
6294 032354      NNCDONE:
      032354 104412      RSETUP
6295
6296

```

;ST 357 TO 416
 ;INTO 417

;(BUT FDST)+ 01634

;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?).

6297
 6298
 6299
 6300 032356 000004
 6301 032360 104413
 6302 032362 012700 032460
 6303 032366 012701 000006
 6304 032372 012720 177777
 6305 032376 077103
 6306 032400 012700 105412
 6307 032404 012737 032426 001236
 6308 032412 012737 032560 000004
 6309 032422 012700 032464
 6310
 6311 032426 170220
 6312 032430 020027 032466
 6313 032434 001017
 6314 032436 023727 032464 105412
 6315 032444 001023
 6316 032446 023727 032466 177777
 6317 032454 001030
 6318 032456 000453
 6319
 6320
 6321 032460 177777 177777
 6322 032464 177777 177777 177777
 6323
 6324
 6325 032474 010037 001242
 6326 032500 012737 032466 001240
 6327 032506
 032506 104377
 032510 000005
 6328
 6329 032512 000435
 6330
 6331
 6332 032514 012737 105412 001240
 6333 032522 013737 032464 001242
 6334 032530
 032530 104377
 032532 000006
 6335
 6336 032534 000424
 6337
 6338
 6339
 6340 032536 012737 177777 001240

```

.SBTTL TEST # 62 - DESTINATION MODES, MODE 2 (FL=0), TEST
*****
*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,$TMP2
        MOV #OOC25,ERRVCT ;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,$TMP4
        MOV #OOC1B1+2,$TMP3
    1$: ERROR +377
        .WORD 5
        BR OOCDONE ;R0 BAD (BUT
                    ; FDST)X

;REPORT RESULT INCORRECT.
OOC15: MOV #105412,$TMP3
        MOV OOC1B1,$TMP4
    1$: ERROR +377
        .WORD 6
        BR OOCDONE ;BAD DATA

;REPORT RESULT INCORRECT.
OOC20: MOV #-1,$TMP3
    
```

ST 634

```
6341 032544 013737 032466 001242      MOV      00C1B1+2,$TMP4
6342 032552      1$:      ERROR    +377
      032552 104377      .WORD    7
      032554 000007
6343                                     :(BUT GR7,FL)
6344 032556 000413      BR       00CDONE      ;ST 357 TO 416
6345                                     ;INTO 417
6346
6347      ;IF A TRAP TO VECTOR 4 OCCURS COME HERE TO SEE IF THE TRAP OCCURRED
6348      ;DURING EXECUTION OF THE FPP INSTRUCTION BEING TESTED, IF NOT GO
6349      ;TO THE SPURIOUS TRAP TO 4 HANDLER.
6350 032560 011604      00C25:  MOV      (SP),R4
6351 032562 020427 032430      CMP      R4,#00C2+2
6352 032566 001402      BEQ     1$
6353 032570 000137 047302      JMP     CPSPUR
6354
6355 032574 011637 001236      1$:     MCV     (SP),$TMP2
6356 032600 022626      CMP     (SP)+,(SP)+
6357 032602      2$:
      032602 104377      ERROR    +377
      032604 000010      .WORD    10
6358                                     :(BUT FDS1)+ $T634
6359
6360 032606      00CDONE:
      032606 104412      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?).
6361
6362
6363
```

6364

```
.SBTTL TEST # 63 - DESTINATION MODES, MODE 4 (FL=0), TEST
:*****
:TEST 63 DESTINATION MODES, MODE 4 (FL=0), TEST
:
:* THIS IS A TEST OF DESTINATION MODE 4 USING
:* THE STFPS INSTRUCTION
:
:*****
```

```
032610 000004
6365
6366 032612
032612 104413
6367 032614 012700 032712
6368 032620 012701 000006
6369 032624 012720 177777
6370 032630 077103
6371 032632 012700 105555
6372 032636 012737 032660 001236
6373 032644 012737 033012 000004
6374 032652 170100
6375 032654 012700 032720
6376
6377 032660 170240
6378 032662 020027 032716
6379 032666 001017
6380 032670 023727 032716 105555
6381 032676 001023
6382 032700 023727 032720 177777
6383 032706 001030
6384 032710 000453
6385
6386
6387 032712 177777 177777
6388 032716 177777 177777 177777
6389
6390
6391 032726 010037 001242
6392 032732 012737 032716 001240
6393 032740
032740 104377
032742 000011
6394
6395 032744 000435
6396
6397
6398 032746 012737 105555 001240
6399 032754 013737 032716 001242
6400 032762
032762 104377
032764 000012
6401
6402 032766 000424
6403
6404
6405
6406 032770 012737 177777 001240
6407 032776 013737 032720 001242
```

```
TST63: SCOPE
PPC1:
L PERR ;SET UP THE LOOP ON ERROR ADDRESS.
MOV #PPCTB0,R0 ;SET UP THE DATA BUFFER.
MOV #6,R1
1$: MOV #-1,(R0)+
SOB R1,1$
MCV #105555,R0
MOV #PPC2,$TMP2
MOV #PPC25,ERRVECT ;SET UP FOR TRAPS TO VECTOR 4.
LDFPS R0 ;SET UP FPS.
MOV #PPCTB1+2,R0
PPC2: STFPS -(R0) ;TEST INSTRUCTION.
CMP R0,#PPCTB1 ;IS R0 CORRECT?
BNE PPC10 ;BRANCH IF NOT CORRECT.
CMP PPCTB1,#105555 ;IS THE RESULT CORRECT?
BNE PPC15 ;BRANCH IF NOT CORRECT.
CMP PPCTB1+2,#-1 ;IS THE RESULT CORRECT?
BNE PPC20 ;BRANCH IF NOT CORRECT.
BR PPCDONE
:TEST DATA BUFFER:
PPCTB0: .WORD -1,-1
PPCTB1: .WORD -1,-1,-1,-1
:REPORT R0 INCORRECT.
PPC10: MOV R0,$TMP4
MOV #PPCTB1,$TMP3
1$: ERROR +377
.WORD 11
BR PPCDONE ;R0 BAD (BUT
; FDST)X
:REPORT RESULT INCORRECT.
PPC15: MOV #105555,$TMP3 ; ST 634
MOV PPCTB1,$TMP4
1$: ERROR +377
.WORD 12
BR PPCDONE ;BAD DATA
:REPORT RESULT INCORRECT.
PPC20: MOV #-1,$TMP3
MOV PPCTB1+2,$TMP4
```

```
6408 033004          1$:
      033004 104377      ERROR +377
      033006 000013      .WORD 13
6409
6410 033010 000413      BR      PPCDONE
6411
6412
6413
6414
6415
6416 033012 011604
6417 033014 020427 032662
6418 033020 001402
6419 033022 000137 047302
6420
6421 033026 011637 001236      1$:  MOV      (SP), $TMP2
6422 033032 022626      CMP      (SP)+, (SP)+
6423 033034
      033034 104377      2$:  ERROR +377
      033036 000014      .WORD 14
6424
6425
6426 033040          PPCDONE:
      033040 104412      RSETUP
6427
6428
6429
```

;(BUT GR7,FL)
;ST 357 TO 416
;INTO 417

;IF A TRAP TO VECTOR 4 OCCURS COME HERE TO SEE IF THE TRAP OCCURRED
;DURING EXECUTION OF THE FPP INSTRUCTION BEING TESTED, IF NOT GO
;TO THE SPURIOUS TRAP TO 4 HANDLER.

PPC25: MOV (SP),R4
CMP R4,#PPC2+2
BEQ 1\$
JMP CPSPUR

;(BUT FDST)+ ST634

;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?).

6430

```
.SBTTL TEST # 64 - DESTINATION MODES, MODE 3 (FL=0), TEST
:*****
:*TEST 64 DESTINATION MODES, MODE 3 (FL=0), TEST
:*
:* THIS IS A TEST OF DESTINATION MODE 3 USING
:* THE STFPS INSTRUCTION
:*
:*****
TST64: SCOPE
```

033042 000004

6431

6432 033044

033044 104413

6433 033046 012700 033150

6434 033052 012701 000010

6435 033056 012720 177777

6436 033062 077103

6437 033064 012700 106653

6438 033070 012737 033116 001236

6439 033076 012737 033254 000004

6440 033104 170100

6441 033106 012700 033164

6442 033112 012710 033154

6443

6444 033116 170230

6445 033120 020027 033166

6446 033124 001021

6447 033126 023727 033154 106653

6448 033134 001025

6449 033136 023727 033164 033154

6450 033144 001032

6451 033146 000455

6452

6453

6454 033150 177777 177777

6455 033154 177777 177777 177777

6456 033164 177777 177777

6457

6458

6459 033170 010037 001242

6460 033174 012737 033166 001240

6461 033202

033202 104377

033204 000015

6462

6463 033206 000435

6464

6465

6466 033210 012737 106653 001240

6467 033216 013737 033154 001242

6468 033224

033224 104377

033226 000016

6469

6470 033230 000424

6471

6472

6473

```
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,$TMP2
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 RO INCORRECT.
QQC10: MOV R0,$TMP4
MOV #QQCTB2+2,$TMP3
1$: ERROR +377
.WORD 15
;RO BAD (BUT
; FDST)X
BR QQCDONE

;REPORT RESULT INCORRECT.
QQC15: MOV #106653,$TMP3 ; ST 634
MOV QQCTB1,$TMP4
1$: ERROR +377
.WORD 16
;BAD DATA
BR QQCDONE

;REPORT RESULT INCORRECT.
```

```

6474 033232 012737 033164 001240 QQC20: MOV #QQCTB2,$TMP3 ;(BUT FDST)
6475 033240 013737 033156 001242 MOV QQCTB1+2,$TMP4
6476 033246 104377 1$: ERROR +377
033246 000017 .WORD 17
6477 033252 000413 BR QQCDONE
6478
6479
6480 ;IF A TRAP TO VECTOR 4 OCCURS COME HERE TO SEE IF THE TRAP OCCURRED
6481 ;DURING EXECUTION OF THE FPP INSTRUCTION BEING TESTED, IF NOT GO
6482 ;TO THE SPURIOUS TRAP TO 4 HANDLER.
6483 033254 011604 QQC25: MOV (SP),R4
6484 033256 020427 033120 CMP R4,#QQC2+2
6485 033262 001402 BEQ 1$
6486 033264 000137 047302 JMP CPSPUR
6487
6488 033270 011637 001236 1$: MCV (SP),$TMP2
6489 033274 022626 CMP (SP)+,(SP)+
6490 033276 104377 2$: ERROR +377
033300 000020 .WORD 20
6491 ;(BUT FDST)+ ST634
6492
6493 033302 QQCDONE:
033302 104412 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?).
6494
6495
6496
  
```

6497

```
.SBTTL TEST # 65 - DESTINATION MODES, MODE 5 (FL=0), TEST  
:*****  
:TEST 65 DESTINATION MODES, MODE 5 (FL=0), TEST  
:*****  
:* THIS IS A TEST OF DESTINATION MODE 5 USING  
:* THE STFPS INSTRUCTION  
:*****  
:TST65: SCOPE
```

0333J4 000004
6498
6499
6500 033306
6501 033310 104413 033414
6502 033314 012701 000006
6503 033320 012720 177777
6504 033324 077103
6505 033326 012700 004301
6506 033332 012737 033362 001236
6507 033340 012737 033520 000004
6508 033346 170100
6509 033350 012700 033432
6510 033354 012760 033420 177776
6511
6512 033362 170250
6513 033364 020027 033430
6514 033370 001021
6515 033372 023727 033420 004301
6516 033400 001025
6517 033402 023727 033430 033420
6518 033410 001032
6519 033412 000455
6520
6521
6522 033414 177777 177777
6523 033420 177777 177777 177777
6524 033430 177777 177777
6525
6526
6527 033434 010037 001242
6528 033440 012737 033430 001240
6529 033446
033446 104377
033450 000021
6530
6531 033452 000435
6532
6533
6534 033454 012737 004301 001240
6535 033462 013737 033420 001242
6536 033470
033470 104377
033472 000022
6537
6538 033474 000424
6539
6540

```
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,$TMP2  
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:  
RRCTB0: .WORD -1,-1  
RRCTB1: .WORD -1,-1,-1,-1  
RRCTB2: .WORD -1,-1  
  
:REPORT R0 INCORRECT.  
RRC10: MOV R0,$TMP4  
MOV #RRCTB2,$TMP3  
1$: ERROR +377  
.WORD 21  
  
:RO BAD (BUT  
: FDST)X  
  
:REPORT RESULT INCORRECT.  
RRC15: MOV #004301,$TMP3 ; ST 634  
MOV RRCTB1,$TMP4  
1$: ERROR +377  
.WORD 22  
  
:BAD DATA  
BR RRCDONE
```



```
6541 ;REPORT RESULT INCORRECT.
6542 033476 012737 033430 001240 RRC20: MOV #RRCB2,$TMP3 ;(BUT FDST)
6543 033504 013737 033422 001242 MOV RRCB1+2,$TMP4
6544 033512 104377 1$: ERROR +377
033512 000023 .WORD 23
6545 ;(BUT GR7,FL)
6546 033516 000413 BR RRCDONE ;ST 357 TO 416
6547 ;INTO 417
6548
6549 ;IF A TRAP TO VECTOR 4 OCCURS COME HERE TO SEE IF THE TRAP OCCURRED
6550 ;DURING EXECUTION OF THE FPP INSTRUCTION BEING TESTED, IF NOT GO
6551 ;TO THE SPURIOUS TRAP TO 4 HANDLER.
6552 033520 011604 RRC25: MOV (SP),R4
6553 033522 020427 033364 CMP R4,#RRC2+2
6554 033526 001402 BEQ 1$
6555 033530 000137 047302 JMP CPSPUR
6556
6557 033534 011637 001236 1$: MOV (SP),$TMP2
6558 033540 022626 CMP (SP)+,(SP)+
6559 033542 104377 2$: ERROR +377
033544 000024 .WORD 24
6560 ;(BUT FDST)+ ST634
6561
6562 033546 RRCDONE:
033546 104412 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?).
6563
6564
```

6565

```
.SBTTL TEST # 66 - DESTINATION MODES, MODE 6 (FL=0), TEST
:*****
:*TEST 66 DESTINATION MODES, MODE 6 (FL=0), TEST
:*
:* THIS IS A TEST OF DESTINATION MODE 6 USING
:* THE STFPS INSTRUCTION
:*
:*****
```

033550 000004
6566
6567
6568 033552
033552 1044*3
6569 033554 012700 033664
6570 033560 012701 000006
6571 033564 012720 177777
6572 033570 077103
6573 033572 012700 102514
6574 033576 012767 033622 145432
6575 033604 012767 033764 144172
6576 033612 170100
6577 033614 005001
6578 033616 012700 026467
6579
6580 033622 170260 005201
6581 033626 020127 000000
6582 033632 001070
6583 033634 020027 026467
6584 033640 001017
6585 033642 026727 000022 102514
6586 033650 001023
6587 033652 026727 000014 177777
6588 033660 001030
6589 033662 000456
6590
6591
6592 033664 177777 177777
6593 033670 177777 177777 177777
6594
6595
6596 033700 010067 145536
6597 033704 012767 026467 145326
6598 033712
033712 104377
033714 000025
6599
6600 033716 000440
6601
6602
6603 033720 012767 102534 145312
6604 033726 016767 177736 145306
6605 033734
033734 104377
033736 000026
6606
6607 033740 000427
6608

```
TST66: SCOPE  
.DSABL AMA ;DISABLE MODE 6 TO MODE 3 CONVERSIONS  
  
SSC1: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.  
MOV #SSCTB0,R0 ;SET UP THE DATA BUFFER.  
MOV #6,R1  
1$: MOV #-1,(R0)+  
SOB R1,1$  
MOV #102514,R0  
MOV #SSC2,$TMP2  
MOV #SSC25,ERRVECT ;SET UP FOR TRAPS TO VECTOR 4.  
LDFPS R0 ;SET UP FPS.  
CLR R1  
MOV #SSCTB1-5201,R0  
  
SSC2: STFPS 5201(R0) ;TEST INSTRUCTION.  
CMP R1,#0 ;WAS PC CORRECT AFTER EXECUTION?  
BNE SSC30 ;BRANCH IF NOT CORRECT.  
CMP R0,#SSCTB1-5201 ;IS R0 CORRECT?  
BNE SSC10 ;BRANCH IF NOT CORRECT.  
CMP SSCTB1,#102514 ;IS THE RESULT CORRECT?  
BNE SSC15 ;BRANCH IF NOT CORRECT.  
CMP SSCTB1+2,#-1 ;IS THE RESULT CORRECT?  
BNE SSC20 ;BRANCH IF NOT CORRECT.  
BR SSCDONE  
  
;TEST DATA BUFFER:  
SSCTB0: .WORD -1,-1  
SSCTB1: .WORD -1,-1,-1,-1  
  
;REPORT R0 INCORRECT.  
SSC10: MOV R0,$TMP4  
MOV #SSCTB1-5201,$TMP3  
1$: ERROR +377  
.WORD 25  
  
BR SSCDONE ;R0 BAD  
  
;REPORT RESULT INCORRECT.  
SSC15: MOV #102534,$TMP3  
MOV SSCTB1,$TMP4  
1$: ERROR +377  
.WORD 26  
  
BR SSCDONE ;BAD DATA
```

```

6609
6610
6611 033742 012767 177777 145270 :REPORT RESULT INCORRECT.
6612 033750 016767 177716 145264 SSC20: MOV #-1,$TMP3
6613 033756 104377 000027 1$: MOV SSC1B1+2,$TMP4
        033756 104377 ERROR +377
        033760 000027 .WORD 27
6614
6615 033762 000416 BR SSCDONE ;(BUT GR7,FL)
6616 ;ST 357 TO 416
6617 ;INTO 417
6618
6619 ;IF A TRAP TO VECTOR 4 OCCURS COME HERE TO SEE IF THE TRAP OCCURRED
6620 ;DURING EXECUTION OF THE FPP INSTRUCTION BEING TESTED, IF NOT GO
6621 ;TO THE SPURIOUS TRAP TO 4 HANDLER.
6621 033764 011604 SSC25: MOV (SP),R4
6622 033766 020427 033624 (MP R4,#SSC2+2
6623 033772 001402 BEQ 1$
6624 033774 000167 013302 JMP CPSPUR
6625
6626 034000 011667 145232 1$: MOV (SP),$TMP2
6627 034004 022626 (MP (SP)+,(SP)+
6628 034006 104377 2$: ERROR +377
        034010 000030 .WORD 30
6629 ;(BUT FDST)+ ST634
6630 034012 000402 BR SSCDONE
6631
6632 ;REPORT PC NOT INCREMENTED BY 2 DURING EXECUTION.
6633 034014 SSC30:
6634 034014 1$: ERROR +377
        034014 104377 .WORD 31
        034016 000031
6635 ;PC NOT
6636 ;INCREMENTED
6637 ;BY 2
6638
6639 034020 SSCDONE:
        034020 104412 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?).
        ;RESET MODE 6 TO MODE 3 CONVERSIONS
6640
6641 .ENABL AMA
    
```

```

6642
6643
6644 034024
6645 034026 012700 034144
6646 034032 012701 000U10
6647 034036 012720 177777
6648 034042 077103
6649 034044 012700 103747
6650 034050 012737 034102 001236
6651 034056 012737 034250 000004
6652 034064 170100
6653 034066 005001
6654 034070 012700 026757
6655 034074 012760 034150 005201
6656
6657 034102 170270 005201
6658 034106 022701 000000
6659 034112 001072
6660 034114 020027 026757
6661 034120 001021
6662 034122 023727 034150 103747
6663 034130 001025
6664 034132 023727 034152 177777
6665 034140 001032
6666 034142 000460
6667
6668
6669 034144 177777 177777
6670 034150 177777 177777 177777
6671 034160 177777 177777
6672
6673
6674 034164 010037 001242
6675 034170 012737 026757 001240
6676 034176
        034176 104377
        034200 000032
6677
6678 034202 000440
6679
6680
6681
6682 034204 012737 103747 001240
6683 034212 013737 034150 001242
6684 034220
        034220 104377
        034222 000033
6685
    
```

```

.SBTTL TEST # 67 - DESTINATION MODES, MODE 7 (FL=0), TEST
:*****
:*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.
MOV #TTCB0,R0                         ;SET UP THE DATA BUFFER.
MOV #10,R1
1$: MOV #-1,(R0)+
SOB R1,1$
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

;REPORT R0 INCORRECT.
TTC10: MOV R0,$TMP4
MOV #TTCB2-5201,$TMP3
1$: ERROR +377
.WORD 32 ;R0 BAD
BR TTCDONE

;REPORT RESULT INCORRECT.
TTC15: MOV #103747,$TMP3
MOV TTCB1,$TMP4
1$: ERROR +377
.WORD 33 ;BAD DATA
    
```

```

6686 034224 000427          BR      TTCDONE
6687
6688
6689
6690 034226 012737 177777 001240 :REPORT RESULT INCORRECT.
6691 034234 013737 034152 001242 TTC20: MOV      #-1,$TMP3
6692 034242          1S:      MOV      TTCTB1+2,$TMP4
        034242 104377          ERROR   +377
        034244 000034          .WORD   34
6693
6694 034246 000416          BR      TTCDONE
6695
6696
6697
6698
6699
6700 034250 011604
6701 034252 020427 034104
6702 034256 001402
6703 034260 000137 047302
6704 034264 011637 001236
6705 034270 022626
6706 034272
        034272 104377
        034274 000035
6707
6708 034276 000402          BR      TTCDONE
6709
6710
6711 034300
6712 034300
        034300 104377
        034302 000036
6713
6714
6715 034304
        034304 104412
TTCDONE: RSETUP

;IF A TRAP TO VECTOR 4 OCCURS COME HERE TO SEE IF THE TRAP OCCURRED
;DURING EXECUTION OF THE FPP INSTRUCTION BEING TESTED, IF NOT GO
;TO THE SPURIOUS TRAP TO 4 HANDLER.
TTC25: MCV      (SP),R4
        CMP      R4,#TTC2+2
        BEQ      1S
        JMP      CPSPUR
1S:     MOV      (SP),$TMP2
        CMP      (SP)+,(SP)+
2S:     ERROR   +377
        .WORD   35
; (BUT GR7,FL)
; ST 357 TO 416
; INTO 417
; (BUT FSDT)+ ST634

;REPORT PC NOT INCREMENTED BY 2 DURING EXECUTION.
TTC30:
1S:     ERROR   +377
        .WORD   36
; PC NOT
; INCREMENTED

;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?).

```

6723

```
.SBTTL TEST # 70 - DESTINATION MODES, MODE 2 (FL-1), TEST
*****
*TEST 70 DESTINATION MODES, MODE 2 (FL-1), TEST
*
* THIS IS A TEST OF DESTINATION MODE
* 2 USING STCOL WITH REGISTER 0
*****
```

```
6724 034306 000004
034310
6725 034310 104413
034312 012700 000300
6726 034316 170100
6727 034320 012700 034370
6728 034324 172410
6729 034326 012737 034340 001236
6730 034334 012700 034402
6731
6732 034340 175420
6733
6734 034342 020027 034406
6735 034346 001420
6736
6737
6738 034350 010037 001242
6739 034354 012737 034406 001240
6740 034362
034362 04377
034364 000037
6741
6742 034366 000410
6743
6744 034370 000000 000000 000000
6745 034400 177777
6746 034402 177777 177777 177777
6747
6748 034410
034410 104412
```

```
TST70: SCOPE
UUC1: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
MOV #300,RO ;SET UP FPS.
LDFPS RO
MOV #UUCTP1,RO ;SET UP THE ACO OPERAND.
LDD (RO),ACO
MOV #UUC2,$TMP2
MOV #UUCBFO,RO
UUC2: STCDL ACO,(RO)+ ;TEST INSTRUCTION.
CMP RO,#UUCBFO+4 ;IS RO CORRECT?
BEQ UUCDONE ;BRANCH IF CORRECT.
;REPORT RO INCORRECT.
UUC3: MOV RO,$TMP4
MOV #UUCBFO+4,$TMP3
1$: ERROR +377
.WORD 37 ;RO NOT INCR BY 4
BR UUCDONE
;TEST DATA BUFFER:
UUCTP1: .WORD 0,0,0,0
-1
UUCBFO: .WORD -1,-1,-1
UUCDONE: 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?).
```

6749

6756
 6757 034412 000004
 6758 034414
 6759 034414 104413
 6760 034416 012700 000300
 6761 034422 170100
 6762 034424 012700 034474
 6763 034430 172410
 6764 034432 012737 034444 001236
 6765 034440 012700 034512
 6766 034444 175440
 6767 034446 020027 034506
 6768 034452 001420
 6769 034454 010037 001242
 6770 034460 012737 034506 001240
 6771 034466
 6772 034466 104377
 6773 034470 000040
 6774 034472 000410
 6775 034474 000000 000000 000000
 6776 034504 177777
 6777 034506 177777 177777 177777
 6778 034514
 6779 034514 104412

```

.SBTTL TEST # 71 - DESTINATION MODES, MODE 4 (FL=1), TEST
*****
*TEST 71 DESTINATION MODES, MODE 4 (FL=1), TEST
*
* THIS IS A TEST OF DESTINATION MODE
* 4 USING STCDL WITH REGISTER 0
*****
TST71: SCOPE
VVC1:
    LPERR                ;SET UP THE LOOP ON ERROR ADDRESS.
    MOV #300,RO          ;SET UP FPS.
    LDFPS RO
    MOV #VVC1P1,RO      ;SET UP THE ACO OPERAND.
    LDD (RO),ACO
    MOV #VVC2,$TMP2
    MOV #VVCBFO+4,RO
VVC2: STCDL ACO,-(RO)    ;TEST INSTRUCTION.
    CMP RO,#VVCBFO      ;IS RO CORRECT?
    BEQ VVCDONE
    .REPORT RO INCORRECT.
VVC3: MOV RO,$TMP4
    MOV #VVCBFO,$TMP3
1$: ERROR +377
    .WORD 40
    ;RO NOT DECR BY 4
    BR VVCDONE
;TEST DATA BUFFER:
VVC1P1: .WORD 0,0,0,0
-1
VVCBFO: .WORD -1,-1,-1
VVCDONE:
    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?).
    
```

6783

6793

```
.SBTTL TEST # 72 - STCDI AND STCDL TEST
:*****
:TEST 72 STCDI AND STCDL TEST
:
:* THIS IS A TEST OF THE STCDI AND
:* STCDL INSTRUCTIONS. NOTE THAT A
:* SUBROUTINE, STCSUB, IS USED TO
:* SET UP THE OPERANDS, EXECUTE THE STC
:* INSTRUCTION AND CHECK THE RESULT.
:*****
TST72: SCOPE
```

```
034516 000004
6794
6795
6796 034520
034520 104413
6797 034522 004737 035666
6798 034526 020000 000000 000000
6799 034536 000000 000000
6800 034542 177777 177777
6801 034546 040300
6802 034550 040304
6803 034552 140304
6804 034554 177777
6805 034556 104322
6806 034560 000401
6807 034562 104325
6808 034564
6809
6810
6811
6812
6813 034564
034564 104413
6814 034566 004737 035666
6815 034572 040000 000000 000000
6816 034602 000000 000000
6817 034606 177777 177777
6818 034612 040313
6819 034614 040304
6820 034616 140304
6821 034620 177777
6822 034622 104322
6823 034624 000401
6824 034626 104326
6825 034630
6826
6827
6828 034630
034630 104413
6829 034632 004737 035666
6830 034636 047667 075757 157737
6831 034646 055675 173757
6832 034652 122102 004021
6833 034656 040717
6834 034660 040700
6835 034662 140705
```

```
;FIRST TEST STC WITH EXP 100 (EXCESS 200)
WWC1:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,STCSUB ;GO EXECUTE THE INSTRUCTION.
1$: .WORD 20000,0,0,0 ;ACO OPERAND.
2$: .WORD 0,0 ;EXPECTED RESULT.
3$: .WORD -1,-1 ;ERROR RES.
4$: 40300 ;FPS BEFORE EXECUTION.
40304 ;FPS AFTER EXECUTION.
140304 ;ANTICIPATED ERRONEOUS FPS.
-1 ;REPORT RESULT INCORRECT.
5$: ERROR +322 ;RESULT INCORP.
BR 6$
6$: ERROR +325 ;EITHER (BUT FLAG)
;ST 662
;OR CLEAR FLAG
;ST 774
```

```
;EXP=0 (OCT) FL=1 FIC=0
WWC2:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,STCSUB ;GO EXECUTE THE INSTRUCTION.
1$: .WORD 40000,0,0,0 ;AC ;ACO OPERAND.
2$: .WORD 0,0 ;EXPECTED RESULT.
3$: .WORD -1,-1 ;ANTICIPATED ERRONEOUS RESULT.
4$: 40313 ;FPS BEFORE EXECUTION.
40304 ;FPS AFTER EXECUTION.
140304 ;ANTICIPATED ERRONEOUS FPS.
-1 ;EXPECTED FEC.
5$: ERROR +322 ;REPORT RESULT INCORRECT.
BR 6$
6$: ERROR +326 ;REPORT FPS INCORRECT.
```

```
;EXP=37 (OCT) FL=1 FIC=1
WWC4:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,STCSUB ;GO EXECUTE THE INSTRUCTION.
1$: .WORD 47667,75757,157737,167773 ;ACO OPERAND.
2$: .WORD 55675,173757 ;EXPECTED RESULT.
3$: .WORD 122102,004021 ;ANTICIPATED ERRONEOUS RESULT.
4$: 40717 ;FPS BEFORE EXECUTION.
40700 ;FPS AFTER EXECUTION.
140705 ;ANTICIPATED ERRONEOUS FPS.
```


TEST # 72 - STCDI AND STCDL TEST

```

6836 034664 177777
6837 034666 104327
6838 034670 000401
6839 034672 104326
6840 034674
6841
6842
6843 034674
034674 104413
6844 034676 004737 035666
6845 034702 050000 000000 000000
6846 034712 000000 000000
6847 034716 177777 177777
6848 034722 040700
6849 034724 140705
6850 034726 040705
6851 034730 000006
6852 034732 104322
6853 034734 000401
6854 034736 104330
6855
6856 034740
6857
6858
6859 034740
034740 104413
6860 034742 004737 035666
6861 034746 050000 000000 000000
6862 034756 000000 000000
6863 034762 177777 177777
6864 034766 040312
6865 034770 040305
6866 034772 140305
6867 034774 177777
6868 034776 104322
6869 035000 000401
6870 035002 104331
6871 035004
6872
6873
6874 035004
035004 104413
6875 035006 004737 035666
6876 035012 046000 000001 000000
6877 035022 000200 000001
6878 035026 177777 177777
6879 035032 040700
6880 035034 040700
6881 035036 177777
6882 035040 177777
6883 035042 104322
6884 035044 000401
6885 035046 104323
6886 035050
6887
6888
6889 035050

```

-1 ;EXPECTED FEC.
5\$: ERROR +327 ;(BUT ENBT) ST 632
BR 6\$
ERROR +326 ;REPORT FPS INCORRECT.
6\$:
;EXP=40 (OCT) FL=1 FIC=1
WWC5:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,STCSUB ;GO EXECUTE THE INSTRUCTION.
1\$: .WORD 50000,0,0,0 ;ACO OPERAND.
2\$: .WORD 0,0 ;EXPECTED RESULT.
3\$: .WORD -1,-1 ;ANTICIPATED ERRONEOUS RESULT.
4\$: 40700 ;FPS BEFORE EXECUTION.
140705 ;FPS AFTER EXECUTION.
40705 ;ANTICIPATED ERRONEOUS FPS.
6 ;EXPECTED FEC.
5\$: ERROR +322 ;REPORT RESULT INCORRECT.
BR 6\$
ERROR +330 ;(BUT FIC) ST 004 ;REPORT FPS INCORRECT.
;TO 305 INTO
;315
6\$:
;EXP=40 (OCT) FL-1 FIC-0
WWC6:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,STCSUB ;GO EXECUTE THE INSTRUCTION.
1\$: .WORD 50000,0,0,0 ;ACO OPERAND.
2\$: .WORD 0,0 ;EXPECTED RESULT.
3\$: .WORD -1,-1 ;ANTICIPATED ERRONEOUS RESULT.
4\$: 40312 ;FPS BEFORE EXECUTION.
40305 ;FPS AFTER EXECUTION.
140305 ;ANTICIPATED ERRONEOUS FPS.
-1 ;EXPECTED FEC.
5\$: ERROR +322 ;REPORT RESULT INCORRECT.
BR 6\$
ERROR +331 ;(BUT FIC) ST 004 TO
;315 INTO 305
6\$:
;EXP=30 (OCT) FL=1 FIC-1
WWC7:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,STCSUB ;GO EXECUTE THE INSTRUCTION.
1\$: .WORD 46000,1,0,0 ;ACO OPERAND.
2\$: .WORD 200,1 ;EXPECTED RESULT.
3\$: .WORD -1,-1 ;ANTICIPATED ERRONEOUS RESULT.
4\$: 40700 ;FPS BEFORE EXECUTION.
40700 ;FPS AFTER EXECUTION.
-1 ;ANTICIPATED ERRONEOUS FPS.
-1 ;EXPECTED FEC.
5\$: ERROR +322 ;REPORT RESULT INCORRECT.
BR 6\$
ERROR +323 ;REPORT FPS INCORRECT.
6\$:
;EXP=27 (OCT) FL=1 FIC-1
WWC8:


```

6943 035262 104335      5$:  ERROR +335 ;(BUT ENBT) ST 632
6944 035264 000401      BR      6$
6945 035266 104336      6$:  ERROR +336 ;(SET FN) ST 473
6946 035270
6947
6948
6949 035270      ;EXP=37 (OCT), FL=1, FIC=1, AC NEG.
        WWC12:
        LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.
        JSR      PC,STCSUB ;GO EXECUTE THE INSTRUCTION.
6950 035272 104413 035666 000000 1$:  .WORD 147600,0,0,0 ;ACO OPERAND.
6951 035276 147600 000000 000000 2$:  .WORD 140000,0 ;EXPECTED RESULT.
6952 035306 140000 000000 3$:  .WORD 137777,0 ;ANTICIPATED ERRONEOUS RESULT.
6953 035312 137777 000000 4$:  40700 ;FPS BEFORE EXECUTION.
6954 035316 040700 ;FPS AFTER EXECUTION.
6955 035320 040710 -1 ;ANTICIPATED ERRONEOUS FPS.
6956 035322 177777 -1 ;EXPECTED FEC.
6957 035324 177777
6958 035326 104337      5$:  ERROR +337 ;(BUT COUT) ST 375
6959 035330 000401      BR      6$ ;ST 275 TO 074
6960 035332 104323      6$:  ERROR +323 ;INTO 274
6961 035334
6962
6963
6964 035334      ;EXP 37 (OCT), FL-1, FIC=1, AC NEG
        WWC13:
        LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.
        JSR      PC,STCSUB ;GO EXECUTE THE INSTRUCTION.
6965 035336 104413 035666 001000 1$:  .WORD 147600,0,1000,0 ;ACO OPERAND.
6966 035342 147600 000000 001000 2$:  .WORD 137777,177777 ;EXPECTED RESULT.
6967 035352 137777 177777 3$:  .WORD 140000,177777 ;ANTICIPATED ERRONEOUS RESULT.
6968 035356 140000 177777 4$:  40707 ;FPS BEFORE EXECUTION.
6969 035362 040707 ;FPS AFTER EXECUTION.
6970 035364 040710 -1 ;ANTICIPATED ERRONEOUS FPS.
6971 035366 177777 -1 ;EXPECTED FEC.
6972 035370 177777
6973 035372 104340      5$:  ERROR +340 ;(BUT COUT) ST 375
6974 035374 000401      BR      6$ ;TO 274 INTO 074
6975 035376 104323      6$:  ERROR +323 ;REPORT FPS INCORRECT.
6976 035400
6977
6978
6979 035400      ;EXP 41 (OCT), AC NEG, FL=1, FIC=1
        WWC14:
        LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.
        JSR      PC,STCSUB ;GO EXECUTE THE INSTRUCTION.
6980 035402 104413 035666 000000 1$:  .WORD 150200,0,0,0 ;ACO OPERAND.
6981 035406 150200 000000 000000 2$:  .WORD 0,0 ;EXPECTED RESULT.
6982 035416 000000 000000 3$:  .WORD -1,-1 ;ANTICIPATED ERRONEOUS RESULT.
6983 035422 177777 177777 4$:  40700 ;FPS BEFORE EXECUTION.
6984 035426 040700 ;FPS AFTER EXECUTION.
6985 035430 140705 -1 ;ANTICIPATED ERRONEOUS FPS.
6986 035432 177777 6 ;EXPECTED FEC.
6987 035434 000006 ;REPORT RESULT INCORRECT.
6988 035436 104322      5$:  ERROR +322
6989 035440 000401      BR      6$
6990 035442 104341      6$:  ERROR +341 ;(BUT EZBT) ST 377
6991 035444
6992
6993 03544      ;EXP-40 (OCT), AC NEG, FL=1, FIC=1
        WWC15:
        LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.
6994 035444 104413 035666 000000 1$:  JSR      PC,STCSUB ;GO EXECUTE THE INSTRUCTION.
6995 035452 150000 000001 000000 .WORD 150000,1,0,0 ;ACO OPERAND.
    
```

6996	035462	000000	000000		2\$:	.WORD	0,0		:EXPECTED RESULT.
6997	035466	100000	177600		3\$:	.WORD	100000,-200		:ANTICIPATED ERRONEOUS RESULT.
6998	035472	040700			4\$:	40700			:FPS BEFORE EXECUTION.
6999	035474	140705				140705			:FPS AFTER EXECUTION.
7000	035476	040700				40700			:ANTICIPATED ERRONEOUS FPS.
7001	035500	000006				6		:EXPECTED FEC.	
7002	035502	104342			5\$:	ERROR	+342		:(BUT COUT) ST 360
7003	035504	000401				BR	6\$:TO 654 INTO 454
7004	035506	104323				ERROR	+323		:REPORT FPS INCORRECT.
7005	035510				6\$:				
7006									
7007									
7008	035510								:EXP-40, AC NEGATIVE, FL=1, FIC=1
									WWC16:
						LPERR			:SET UP THE LOOP ON ERROR ADDRESS.
7009	035512	004737	035666			JSR	PC,STCSUB		:GO EXECUTE THE INSTRUCTION.
7010	035516	150001	000000	000000	1\$:	.WORD	150001,0,0,0		:ACO OPERAND.
7011	035526	000000	000000		2\$:	.WORD	0,0		:EXPECTED RESULT.
7012	035532	077400	000600		3\$:	.WORD	77400,0		:ANTICIPATED ERRONEOUS RESULT.
7013	035536	040700			4\$:	40700			:FPS BEFORE EXECUTION.
7014	035540	140705				140705			:FPS AFTER EXECUTION.
7015	035542	177777				-1			:ANTICIPATED ERRONEOUS FPS.
7016	035544	000006				6		:EXPECTED FEC.	
7017	035546	104343			5\$:	ERROR	+343		:REPORT RESULT INCORRECT.
7018	035550	000401				BR	6\$		
7019	035552	104323				ERROR	+323		:REPORT FPS INCORRECT.
7020	035554				6\$:				
7021									
7022									
7023									:EXP 40 (OCT), AC MOST NEG LONG INT, FL=1
7024									:FIC 1
7025	035554								WWC17:
						LPERR			:SET UP THE LOOP ON ERROR ADDRESS.
7026	035556	004737	035666			JSR	PC,STCSUB		:GO EXECUTE THE INSTRUCTION.
7027	035562	150000	000000	000000	1\$:	.WORD	150000,0,0,0		:ACO OPERAND.
7028	035572	100000	000000		2\$:	.WORD	100000,0		:EXPECTED RESULT.
7029	035576	000000	000000		3\$:	.WORD	0,0		:ANTICIPATED ERRONEOUS RESULT.
7030	035602	040700			4\$:	40700			:FPS BEFORE EXECUTION.
7031	035604	040710				40710			:FPS AFTER EXECUTION.
7032	035606	140705				140705			:ANTICIPATED ERRONEOUS FPS.
7033	035610	177777				-1		:EXPECTED FEC.	
7034	035612	104344			5\$:	ERROR	+344		:(BUT NBIT) ST 654
7035	035614	000401				BR	6\$:OR (BUT COUT) ST 454
7036	035616	104323				ERROR	+323		:REPORT FPS INCORRECT.
7037	035620				6\$:				
7038									
7039									:EXP=20, AC = MOST NEG INTEGER, FL=0, FIC=1
7040									
7041	035620								WWC18:
						LPERR			:SET UP THE LOOP ON ERROR ADDRESS.
7042	035622	004737	035666			JSR	PC,STCSUB		:GO EXECUTE THE INSTRUCTION.
7043	035626	144000	000001	000000	1\$:	.WORD	144000,1,0,0		:ACO OPERAND.
7044	035636	100000	177777		2\$:	.WORD	100000,-1		:EXPECTED RESULT.
7045	035642	100000	177400		3\$:	.WORD	100000,177400		:ANTICIPATED ERRONEOUS RESULT.
7046	035646	040600			4\$:	40600			:FPS BEFORE EXECUTION.
7047	035650	040610				40610			:FPS AFTER EXECUTION.
7048	035652	140605				140605			:ANTICIPATED ERRONEOUS FPS.
7049	035654	177777				-1		:EXPECTED FEC.	

7050 035656 104345
 7051 035660 000401
 7052 035662 104323
 7053
 7054 035664 000534
 7055
 7056
 7057
 7058
 7059
 7060
 7061
 7062
 7063
 7064
 7065
 7066
 7067
 7068
 7069
 7070
 7071
 7072
 7073
 7074
 7075
 7076
 7077
 7078
 7079
 7080
 7081
 7082
 7083
 7084
 7085
 7086
 7087
 7088 035666 01260*
 7089 035670 012700 000200
 7090 035674 170100
 7091 035676 010100
 7092 035700 172410
 7093 035702 012702 036146
 7094 035706 012700 000004
 7095 035712 012722 177777
 7096 035716 077003
 7097 035720 016100 000020
 7098 035724 170100
 7099 035726 012737 035740 001236
 7100 035734 012700 036146
 7101 035740 175410
 7102
 7103 035742 170204
 7104 035744 170305
 7105 035746 010102
 7106 035750 010237 001240

```

5$:  ERROR +345                ;(BUT FL) ST 633
      BR    6$                  ;TO 655 INTO 654
      ERROR +323                ;REPORT FPS INCORRECT.

6$:  BR    WWC DONE

;THIS SUBROUTINE, STCSUB, IS USED TO SET UP THE OPERANDS, EXECUTE
;THE STCDI OR STCDL INSTRUCTION AND CHECK THE RESULTS. A CALL
;TO IT IS MADE THUS:
;
;          JSR    PC,STCSUB
;          ACARG: .WORD  X,X,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.
;          FEC:   .WORD  X              ;EXPECTED FEC
;          ERR1:  ERROR +X              ;DATA ERROR.
;          BR    CONT                  ;
;          ERR2:  ERROR +X              ;FPS ERROR.
;          CONT:  CONT                  ;RETURN ADDRESS

;THE OPERANDS ARE SET UP (USING ACO AS THE ACCUMULATOR). THEN
;THE STCDI OR STCDL 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 STCSUB RETURNS CONTROL
;TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD STCSUB
;COMPARE IT TO ERROR FPS. IF THIS MATCHES THEN STCSUB WILL RETURN
;TO THE ERROR CALL AT ERR2, OTHERWISE STCSUB ITSELF
;REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE
;STCDI OR STCDL 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 STCSUB
;WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR1. OTHERWISE THE
;RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND STCSUB WILL
;REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.

STCSUB: MOV    (SP)+,R1          ;GET A POINTER TO THE ARGUMENTS.
        MOV    #200,R0          ;SET UP THE ACO OPERAND.
        LDFPS R0
        MOV    R1,R0
        LDD    (R0),ACO
        MOV    #STCIBF,R2      ;INITIALIZE THE OUT PUT BUFFER.
        MOV    #4,R0
1$:     MOV    #-1,(R2)+
        SOB   R0,1$
        MOV    20(R1),R0        ;SET THE FPS.
        LDFPS R0
        MOV    #2$,$TMP2
        MOV    #STCIBF,R0
2$:     STCDL  ACO,(R0)          ;TEST INSTRUCTION.

        STFPS R4                ;GET THE FPS.
        STST  R5                ;GET THE FEC.
        MOV    R1,R2
        MOV    R2,$TMP3
    
```

```

7107 035754 062702 000010      ADD      #10,R2
7108 035760 010237 001244      MOV      R2,$TMP5
7109 035764 012737 036146 001242  MOV      #STCIBF,$TMP4
7110 035772 010437 001250      MOV      R4,$TMP7
7111 035776 016137 000022 001252  MOV      22(R1),$TMP10
7112 036004 010102      MOV      R1,R2
7113 036006 062702 000010      ADD      #10,R2
7114 036012 012700 036146      MOV      #STCIBF,R0      ;SEE IF THE RESULT IS CORRECT.
7115 036016 012703 000002      MOV      #2,R3
7116 036022 022022      3$:      CMP      (R0)+,(R2)+
7117 036024 001014      BNE      15$
7118 036026 077303      SOB      R3,3$
7119 036030 016102 000022      MOV      22(R1),R2
7120 036034 020204      CMP      R2,R4      ;SEE IF THE FPS IS CORRECT.
7121 036036 001025      BNE      20$      ;BRANCH IF INCORRECT.
7122 036040 005702      ST      R2
7123 036042 100003      BFL      4$
7124 036044 026105 000026      CMP      26(R1),R5      ;SEE IF THE FEC IS CORRECT.
7125 036050 001027      BNE      25$      ;BRANCH IF INCORRECT.
7126
7127 036052 000161 000036      4$:      JMP      36(R1)      ;RETURN.
7128      ;DATA ERROR:
7129      ;SEE IF THE FAILURE WAS ANTICIPATED.
7130 036056 010102      15$:      MOV      R1,R2
7131 036060 062702 000014      ADD      #14,R2
7132 036064 012700 036146      MOV      #STCIBF,R0
7133 036070 012703 000002      MOV      #2,R3
7134 036074 022022      16$:      CMP      (R0)+,(R2)+
7135 036076 001003      BNE      17$
7136 036100 077303      SOB      R3,16$
7137 036102 000161 000030      JMP      30(R1)
7138 036106
7139      ;FAILURE WAS NOT ANTICIPATED SO REPORT INCORRECT RESULT HERE.
7140 036106 104322      18$:      ERROR   +322      ;DATA BAD
7141 036110 000760      BR      4$
7142
7143      ;FPS INCORRECT, SO SEE IF FAILURE WAS ANTICIPATED.
7144 036112 020461 000024      20$:      CMP      R4,24(R1)
7145 036116 001002      BNE      21$
7146 036120 000161 000034      JMP      34(R1)
7147 036124
7148      ;NOT ANTICIPATED SO REPORT BAD FPS HERE.
7149 036124 104323      21$:      ERROR   +323      ;FPS BAD
7150 036126 000751      BR      4$
7151
7152      ;REPORT INCORRECT FEC.
7153 036130 016137 000026 001256  25$:      MOV      26(R1),$TMP12
7154 036136 010537 001254      MOV      R5,$TMP11
7155 036142 104324      26$:      ERROR   +324
7156 036144 000742      BR      4$
7157
7158      ;DATA BUFFER:
7159 036146 177777 177777 177777  STCIBF: .WORD  -1,-1,-1,-1
7160
7161 036156      WWC DONE:
      036156 104412      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?).

7162
7163

7171

```

.SBTTL TEST # 73 - STCFL AND STCFI TEST
:*****
:*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
    
```

036160 000004

7172

7173

7174

7175 036162

```

;EXPONENT-37, PL=1
XXC1:
    
```

036162 104413

7176 036164 004737 035666

7177 036170 047777 177777 177777

7178 036200 077777 177600

7179 036204 077777 177777

7180 036210 040100

7181 036212 040100

7182 036214 177777

7183 036216 177777

7184 036220 104346

7185 036222 000401

7186 036224 104323

7187 036226

7188

7189 036226

036226 104412

```

LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
JSR            PC,STCSUB      ;GO EXECUTE THE INSTRUCTION.
1$: .WORD      47777,-1,-1,-1 ;ACO OPERAND.
2$: .WORD      77777,177600   ;EXPECTED RESULT.
3$: .WORD      77777,177777   ;ANTICIPATED ERRONEOUS RESULT.
4$: 40100          ;FPS BEFORE EXECUTION.
    40100          ;FPS AFTER EXECUTION.
    -1            ;ANTICIPATED ERRONEOUS FPS.
    -1            ;EXPECTED FEC.
5$: ERROR      +346          ;X11(1,0)+0 ST 773X
    BR         6$
    ERROR      +323          ;REPORT FPS INCORRECT.
6$:
    
```

```

XXCDONE:
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?).
    
```

7190

7191

7198

```
.SBTTL TEST # 74 - STEXP TEST  
:*****  
:*TEST 74 STEXP TEST  
:*  
:* THIS IS A TEST OF THE STEXP  
:* INSTRUCTION  
:*  
:*****
```

7199 036230 000004

TST74: SCOPE

7200

: EXP = 100 (EXCESS 200)

7201

YVC1:

036232 104413

LPERR

:SET UP THE LOOP ON ERROR ADDRESS.

7202 036234 004737 036520

JSR PC,STXSUB

7203 036240 020000 000000 000000

1\$: .WORD 20000,0,0,0

:AC

7204 036250 177700

2\$: -100

:EXP RES

7205 036252 052525

3\$: 52525

:ERROR EXP.

7206 036254 040000

4\$: 40000

:FPSB

7207 036256 040010

40010

:FPSA

7208 036260 040000

40000

:ERROR FPS

7209 036262 104347

5\$: ERROR +347

:BAD EXP

7210 036264 000401

BR 6\$

7211 036266 104352

6\$: ERROR +352

;(BUT ENBT) ST 376

7212 036270

7213

7214

: EXP - 200 (EXCESS 200)

7215 036270

YVC2:

036270 104413

LPERR

:SET UP THE LOOP ON ERROR ADDRESS.

7216 036272 004737 036520

JSR PC,STXSUB

:GO EXECUTE THE INSTRUCTION.

7217 036276 040000 000000 000000

1\$: .WORD 40000,0,0,0

:ACO OPERAND.

7218 036306 000000

2\$: 0

:EXPECTED EXPONENT RESULT.

7219 036310 052525

3\$: 52525

:ANTICIPATED ERRONEOUS RESULT.

7220 036312 040000

4\$: 40000

:FPS BEFORE EXECUTION.

7221 036314 040004

40004

:FPS AFTER EXECUTION.

7222 036316 040000

40000

:ANTICIPATED ERRONEOUS FPS.

7223 036320 104347

5\$: ERROR +347

:REPORT RESULT INCORRECT.

7224 036322 000401

BR 6\$

7225 036324 104353

6\$: ERROR +353

;(BUT EZBT) ST 071

:TO 072 INT 272

7226

7227 036326

6\$:

7228

: EXP 201 (EXCESS 200)

7229

YVC3:

7231 036326 104413

LPERR

:SET UP THE LOOP ON ERROR ADDRESS.

7232 036330 004737 036520

JSR PC,STXSUB

:GO EXECUTE THE INSTRUCTION.

7233 036334 040200 000000 000000

1\$: .WORD 40200,0,0,0

:ACO OPERAND.

7234 036344 000001

2\$: 1

:EXPECTED EXPONENT RESULT.

7235 036346 052525

3\$: 52525

:ANTICIPATED ERRONEOUS RESULT.

7236 036350 040000

4\$: 40000

:FPS BEFORE EXECUTION.

7237 036352 040000

40000

:FPS AFTER EXECUTION.

7238 036354 040004

40004

:ANTICIPATED ERRONEOUS FPS.

7239 036356 104347

5\$: ERROR +347

:REPORT RESULT INCORRECT.

7240 036360 000401

BR 6\$

7241 036362 104354

6\$: ERROR +354

;(BUT EZBT) ST 071

:TO 272 INTO 072

7242 036364

7243

```

7244 ; EXP - 375 (EXCESS 200)
7245
7246 036364 YYC4:
      036364 104413 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
7247 036366 004737 036520 JSR PC,STXSUB ;GO EXECUTE THE INSTRUCTION.
7248 036372 077200 000000 000000 1$: .WORD 77200,0,0,0 ;ACO OPERAND.
7249 036402 000175 2$: 175 ;EXPECTED EXPONENT RESULT.
7250 036404 052525 3$: 52525 ;ANTICIPATED ERRONEOUS RESULT.
7251 036406 040000 4$: 40000 ;FPS BEFORE EXECUTION.
7252 036410 040000 40000 ;FPS AFTER EXECUTION.
7253 036412 040010 40010 ;ANTICIPATED ERRONEOUS FPS.
7254 036414 104347 5$: ERROR +347 ;REPORT RESULT INCORRECT.
7255 036416 000401 BR 6$
7256 036420 104355 ERROR +355 ;(BUT ENBT) ST 376
7257 036422 6$: ;TO 471 INTO 071
7258
7259 ; EXP - 1 (EXCESS 200)
7260
7261 036422 YYC5:
      036422 104413 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
7262 036424 004737 036520 JSR PC,STXSUB ;GO EXECUTE THE INSTRUCTION.
7263 036430 000200 000000 000000 1$: .WORD 200,0,0,0 ;ACO OPERAND.
7264 036440 177601 2$: -177 ;EXPECTED EXPONENT RESULT.
7265 036442 052525 3$: 52525 ;ANTICIPATED ERRONEOUS RESULT.
7266 036444 040000 4$: 40000 ;FPS BEFORE EXECUTION.
7267 036446 040010 40010 ;FPS AFTER EXECUTION.
7268 036450 040000 40000 ;ANTICIPATED ERRONEOUS FPS.
7269 036452 104347 5$: ERROR +347 ;REPORT RESULT INCORRECT.
7270 036454 000401 BR 6$
7271 036456 104352 ERROR +352 ;REPORT FPS INCORRECT.
7272 036460 6$:
7273
7274 ; EXP = 156 (EXCESS 200)
7275
7276 036460 YYC6:
      036460 104413 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
7277 036462 004737 036520 JSR PC,STXSUB ;GO EXECUTE THE INSTRUCTION.
7278 036466 033400 000000 000000 1$: .WORD 33400,0,0,0 ;ACO OPERAND.
7279 036476 177756 2$: -22 ;EXPECTED EXPONENT RESULT.
7280 036500 052525 3$: 52525 ;ANTICIPATED ERRONEOUS RESULT.
7281 036502 047707 4$: 47707 ;FPS BEFORE EXECUTION.
7282 036504 047710 47710 ;FPS AFTER EXECUTION.
7283 036506 177777 -1 ;ANTICIPATED ERRONEOUS FPS.
7284 036510 104347 5$: ERROR +347 ;REPORT RESULT INCORRECT.
7285 036512 000401 BR 6$
7286 036514 104350 ERROR +350 ;REPORT FPS INCORRECT.
7287
7288 036516 000510 6$: BR YYCDONE
7289
7290 ;THIS SUBROUTINE, STXSUB, IS USED TO SET UP THE OPERANDS, EXECUTE
7291 ;THE STEXP INSTRUCTION AND CHECK THE RESULTS. A CALL
7292 ;TO IT IS MADE THUS:
7293 :
7294 : JSR PC,STXSUB
7295 : ACARG: .WORD X,X,X,X ;AC OPERAND
7296 : RES: .WORD X ;EXPECTED RESULT
7297 : ERRES: .WORD X ;ERROR RESULT

```


7355 036676 001002
7356 036700 000161 000026
7357
7358
7359 036704
7360 036704 104350
7361 036706 000764
7362
7363
7364 036710 022737 076543 036730
7365 036716 001760
7366 036720 104351
7367 036722 000756
7368
7369 036724 177777
7370 036726 177777 177777 177777
7371
7372 036740
036740 104412

BNE 6\$
JMP 26(R1)
:FPS ERROR WAS NOT ANTICIPATED SO REPORT ERROR HERE.
6\$:
7\$: ERROR +350 :FPS BAD
BR 4\$
:SEE IF MORE THAN ONE WORD WAS WRITTEN IN THE OUTPUT BUFFER.
10\$: CMP #76543,STXBF+2
BEQ 4\$
11\$: ERROR +351 :FDFL+0 ST 347X
BR 4\$
-1
STXBF: .WORD -1,-1,-1,-1,-1

YYCDONE:
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?).

7373

7384

```
.SBTTL TEST # 75 - STST TEST
*****
*TEST 75 STST TEST
*
* THIS IS A TEST OF THE STST
* INSTRUCTION. FIRST AN ILLEGAL FPS OP CODE
* (INSTRUCTION) IS USED TO ENTER AN
* ERROR CONDITION IN THE FEC AND
* FEA. THE STST IS EXECUTED AND
* THE FEC AND FEA ARE CHECKED
*****
```

TST75: SCOPE

7385	036742	000004							
7386	036744	104413							
7387	036746	012700	040000						
7388	036752	170100							
7389									
7390	036754	170003							
7391									
7392	036756	012700	037132						
7393	036762	012710	177777						
7394	036766	012760	177777	000002					
7395	036774	012737	037002	001236					
7396	037002	170310							
7397									
7398	037004	170204							
7399	037006	012700	037132						
7400	037012	011037	001240						
7401	037016	016037	000002	001242					
7402	037024	012737	000002	001244					
7403	037032	012737	036754	001246					
7404	037040	010437	001250						
7405	037044	012737	140000	001252					
7406									
7407	037052	022710	000002						
7408	037056	001010							
7409	037060	022760	036754	000002					
7410	037066	001006							
7411	037070	022704	140000						
7412	037074	001013							
7413	037076	000422							
7414									
7415									
7416	037100								
7417	037100	104356							
7418	037102	000420							
7419									
7420									
7421	037104	022760	177777	000002					
7422	037112	001402							
7423	037114	104357							
7424	037116	000412							
7425	037120								
7426	037120	104360							
7427	037122	000410							

```
ZZC1: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
MCV #40000,R0 ;SET FPS. FID=1.
LDFPS R0

ZZC2: .WORD 170003 ;ILLEGAL FPP
;OP CODE
MOV #ZZCBF,R0 ;SET UP THE OUTPUT BUFFER.
MOV #-1,(R0)
MOV #-1,2(R0)
MOV #ZZC3,$TMP2

ZZC3: STST (R0) ;GET FEC AND
;FEA
STFPS R4 ;GET FPS.
MOV #ZZCBF,R0
MOV (R0),$TMP3
MOV 2(R0),$TMP4
MOV #2,$TMP5
MOV #ZZC2,$TMP6
MOV R4,$TMP7
MOV #140000,$TMP10

CMP #2,(R0) ;SEE IF FEC IS CORRECT.
BNE ZZC5 ;BRANCH IF INCORRECT.
CMP #ZZC2,2(R0) ;SEE IF FEA, ADDRESS, IS CORRECT.
BNE ZZC10 ;BRANCH IF INCORRECT.
CMP #140000,R4 ;SEE IF FPS IS CORRECT.
BNE ZZC15 ;BRANCH IF INCORRECT.
BR ZZCDONE

;REPORT FEC INCORRECT
ZZC5:
1$: ERROR +356 ;STST BAD
BR ZZCDONE ;FECX

;REPORT FEA INCORRECT
ZZC10: CMP #-1,2(R0)
BEQ ZZC12
1$: ERROR +357 ;STST BAD FEA
BR ZZCDONE

ZZC12:
1$: ERROR +360 ;SET FD FL ST 636
BR ZZCDONE
```

```
7428  
7429 ;REPORT FPS INCORRECT  
7430 037124 ZZC15:  
7431 037124 104361 1$: ERROR +361 ;FPS X AFTER ST ST  
7432 037126 000406 BR ZZCDONE  
7433  
7434 ;DATA BUFFER:  
7435 037130 177777 -1  
7436 037132 177777 177777 177777 ZZCBF: .WORD -1,-1,-1,-1  
7437 037142 177777 -1  
7438  
7439 037144 ZZCDONE:  
037144 104412 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?).
```

```

7441 .SBTTL SETUP FOR TESTS 76 THROUGH 104
7442 037146 005037 177572 CLR MMRO ;MAKE SURE MEMORY MANAGEMENT IS OFF.
7443 037152 170127 040000 LDFPS #40000 ;LOAD FPS STATUS.
7444 037156 012700 077406 MOV #77406,R0 ;LOAD R0 WITH 77406
7445 037162 012701 077400 MOV #77400,R1 ;LOAD R1 WITH 77400
7446 037166 010037 172320 MOV R0,KDPDR0 ;MAKE KDPDR0 RESIDENT.
7447 037172 010137 172322 MOV R1,KDPDR1 ;MAKE :DPDR1 NON-RESIDENT.
7448 037176 010137 172324 MOV R1,KDPDR2 ;MAKE KDPDR2 NON-RESIDENT.
7449 037202 010037 172326 MOV R0,KDPDR3 ;MAKE KDPDR3 RESIDENT FOR ADDRESSES 60000-77756.
7450 037206 010037 172330 MOV R0,KDPDR4 ;MAKE KDPDR4 RESIDENT FOR ADDRESSES 77760-77776.
7451 037212 010037 172336 MOV R0,KDPDR7 ;MAKE KDPDR7 RESIDENT (I/O PAGE).
7452
7453 037216 005037 172360 CLR KDPAR0 ;MAP D-PAGE 0 FOR 0-4K.
7454 037222 012737 000200 172362 MOV #200,KDPAR1 ;MAP D-PAGE 1 FOR 4-8K.
7455 037230 012737 000400 172364 MOV #400,KDPAR2 ;MAP D-PAGE 2 FOR 8-12K.
7456 037236 012737 000600 172366 MOV #600,KDPAR3 ;MAP D-PAGE 3 FOR ACCESSING ADDRESSES 60000-77756.
7457 037244 012737 000600 172370 MOV #600,KDPAR4 ;MAP D-PAGE 4 FOR ACCESSING ADDRESSES 77760-77776.
7458 037252 012737 177600 172376 MOV #177600,KDPAR7 ;MAP D-PAGE 7 FOR I/O PAGE.
7459
7460 037260 010037 172300 MOV R0,KIPDR0 ;MAKE KIPDR0 RESIDENT.
7461 037264 010037 172302 MOV R0,KIPDR1 ;MAKE KIPDR1 RESIDENT.
7462 037270 010037 172304 MOV R0,KIPDR2 ;MAKE KIPDR2 RESIDENT.
7463 037274 010137 172306 MOV R1,KIPDR3 ;MAKE KIPDR3 NON-RESIDENT FOR USING ADDRESSES 60000-77756.
7464 037300 010137 172310 MOV R1,KIPDR4 ;MAKE KIPDR4 NON-RESIDENT FOR USING ADDRESSES 77760-77776.
7465 037304 010037 172316 MOV R0,KIPDR7 ;MAKE KIPDR7 RESIDENT (I/O PAGE).
7466
7467 037310 005037 172340 CLR KIPAR0 ;MAP I-PAGE 0 FOR 0-4K.
7468 037314 012737 000200 172342 MOV #200,KIPAR1 ;MAP I-PAGE 1 FOR 4-8K.
7469 037322 012737 000400 172344 MOV #400,KIPAR2 ;MAP I-PAGE 2 FOR 8-12K.
7470 037330 012737 000600 172346 MOV #600,KIPAR3 ;MAP I-PAGE 3 FOR ACCESSING ADDRESSES 60000-77756.
7471 037336 012737 000600 172350 MOV #600,KIPAR4 ;MAP I-PAGE 4 FOR ACCESSING ADDRESSES 77760-77776.
7472 037344 012737 177600 172356 MOV #177600,KIPAR7 ;MAP I-PAGE 7 FOR I/O PAGE.
7473
7474 037352 013737 000250 001262 MOV MMVECT,$TMP14 ;MOVE MM TRAP VECTOR TO $TMP14 FOR TEMP STORAGE.
7475 037360 012701 117760 MOV #DATA,R1 ;SET UP R1.
7476 037364 012702 117770 MOV #DATA+10,R2 ;SET UP R2.
7477 037370 012703 117772 MOV #DATA+12,R3 ;SET UP R3.
7478 037374 012737 040010 000250 MOV #TRAPV,MMVECT ;SET UP FOR FP TRAPS FOR THIS TEST.
7479 037402 012737 000340 000257 MOV #340,MMVECT+2
7480 037410 012737 000024 172516 MOV #24,MMR3 ;TURN ON 22-BIT KERNEL D-SPACE.
7481 037416 012737 117760 077770 MOV #DATA,77770 ;SET UP ADDRESS POINTER.
  
```

7495

.SBTTL TEST # 76 - ENABLE D-SPACE & SEE I-SPACE FORCED, MODE 1

*TEST 76 ENABLE D-SPACE & SEE I-SPACE FORCED, MODE 1
*
* THIS IS A TEST THAT WILL ENABLE D-SPACE AND MAKE IT RESIDENT SO THAT AN
* INSTRUCTION THAT ACCESSES I-SPACE WHEN IT NORMALLY SHOULDN'T WILL CAUSE A
* MEMORY MANAGEMENT TRAP/ABORT.
* ****ALL REFERENCES TO MICRO-FLOWS REFER TO *FP11-F-2 REV A* FLOWS****
* THE COMMENTS FOR EACH TEST LINE ARE WRITTEN SO YOU CAN GO TO THE MICRO FLOWS
* AND PINPOINT THE PROBLEM AREA. FROM THERE, HARDWARE ANALYSIS SHOULD BE EASIER.
* INSTRUCTION GROUPS ISOLATED BY BLANK LINES ARE TO BE EXECUTED TOGETHER
* DUE TO PROPER SETUP PURPOSES. I.E. THE LOCATION OR ADDRESS HAS TO BE
* INITIALIZED PROPERLY BEFORE THE INSTRUCTION CAN BE ACCOMPLISHED.
*

7496 037424 000004
7496 037426 104413
7497 037430 005237 177572
7498
7499 037434 170000
7500 037436 010100
7501 037440 170410
7502 037442 177010
7503 037444 172410
7504 037446 170310
7505 037450 005037 177572

TST76: SCOPE
ZZF1:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
INC MMRO ;TURN ON MEMORY MANAGEMENT.
;* THIS INSTRUCTION WILL TEST FOR A WORST-CASE HARDWARE PROBLEM.
CFCC ;* TEST INSTRUCTION WHICH SHOULD ALWAYS INVOKE D-SPACE.
MOV R1,R0 ;SETTING UP R0.
CLRF (R0) ;TESTING BLOCKS 27-K AND 27-R.
LDCIF (R0),ACO ;TESTING BLOCKS 28-F AND 28-P.
LDF (R0),ACO ;TESTING BLOCKS 4-J, 4-X, 4-Z AND 4-BB.
STST (R0) ;TESTING BLOCKS 33-E AND 33-P.
CLR MMRO ;TURN OFF MEMORY MANAGEMENT.

7506

```
.SBTTL TEST # 77 - ENABLE D-SPACE & SEE I-SPACE FORCED, MODE 2
:*****
:*TEST 77      ENABLE D-SPACE & SEE I-SPACE FORCED, MODE 2
:*
:* THIS IS A TEST THAT WILL ENABLE D-SPACE AND MAKE IT RESIDENT SO THAT AN
:* INSTRUCTION THAT ACCESSES I-SPACE WHEN IT NORMALLY SHOULDN'T WILL CAUSE A
:* MEMORY MANAGEMENT TRAP/ABORT.
:* *****ALL REFERENCES TO MICRO-FLOWS REFER TO *FP11-F-2 REV A* FLOWS*****
:* THE COMMENTS FOR EACH TEST LINE ARE WRITTEN SO YOU CAN GO TO THE MICRO FLOWS
:* AND PINPOINT THE PROBLEM AREA. FROM THERE, HARDWARE ANALYSIS SHOULD BE EASIER.
:* INSTRUCTION GROUPS ISOLATED BY BLANK LINES ARE TO BE EXECUTED TOGETHER
:* DUE TO PROPER SETUP PURPOSES. I.E. THE LOCATION OR ADDRESS HAS TO BE
:* INITIALIZED PROPERLY BEFORE THE INSTRUCTION CAN BE ACCOMPLISHED.
:*
:*****
```

```

037454 000004
7507 037456 005237 177572
7508 037462 010100
7509 037464 170520
7510 037466 170527 001000
7511
7512 037472 010100
7513 037474 170420
7514
7515
7516
7517
7518
7519 037476 170427 001000
7520
7521 037502 010100
7522 037504 177020
7523 037506 177027 001000
7524
7525 037512 010100
7526 037514 172420
7527 037516 172427 042572
7528
7529 037522 010100
7530 037524 170320
7531 037526 170327 001000
7532 037532 005037 177572
```

```
TST77: SCOPE
        INC      MMRO          ;TURN ON MEMORY MANAGEMENT.
        MOV      R1,R0        ;SETTING UP R0.
        TSTF    (R0)+        ;TESTING BLOCK 21-AA.
        TSTF    #1000
        MOV      R1,R0        ;CORRECTING R0.
        CLRF    (R0)+        ;TESTING BLOCKS 27-K AND 27-R.
        ;**NOTE** THE LOCATION AFTER THE CLRF, AND STST MODE 2 REG 7 INSTRUCTIONS
        ;**WILL** BE CHANGED ON SUBSEQUENT PASSES, BUT IS **NOT** INCORRECT. THE
        ;ACTUAL CONTENTS OF THOSE LOCATIONS IS IMMATERIAL, AS THIS TEST INSURES
        ;THAT THE INSTRUCTION DOES EXECUTE WITHOUT ACCESSING THAT LOCATION AS
        ;A D-SPACE ACCESS.
        CLRF    #1000
        MOV      R1,R0        ;CORRECTING R0.
        LDCIF   (R0)+,ACO     ;TESTING BLOCKS 28-F AND 28-P.
        LDCIF   #1000,ACO
        MOV      R1,R0        ;CORRECTING R0.B
        LDF     (R0)+,ACO     ;TESTING BLOCKS 4-NN, 4-X, 4-Z AND 4-BB.
        LDF     #1000,ACO
        MOV      R1,R0        ;CORRECTING R0.
        STST    (R0)+        ;TESTING BLOCKS 33-J AND 33-P.
        STST    #1000
        CLR     MMRC         ;TURN OFF MEMORY MANAGEMENT.
```

7533

```
.SBTTL TEST # 100 - ENABLE D-SPACE & SEE I-SPACE FORCED, MODE 3
*****
*TEST 100      ENABLE D-SPACE & SEE I-SPACE FORCED, MODE 3
*
* THIS IS A TEST THAT WILL ENABLE D-SPACE AND MAKE IT RESIDENT SO THAT AN
* INSTRUCTION THAT ACCESSES I-SPACE WHEN IT NORMALLY SHOULDN'T WILL CAUSE A
* MEMORY MANAGEMENT TRAP/ABORT.
* ****ALL REFERENCES TO MICRO-FLOWS REFER TO *FP11-F-2 REV A* FLOWS****
* THE COMMENTS FOR EACH TEST LINE ARE WRITTEN SO YOU CAN GO TO THE MICRO FLOWS
* AND PINPOINT THE PROBLEM AREA. FROM THERE, HARDWARE ANALYSIS SHOULD BE EASIER.
* INSTRUCTION GROUPS ISOLATED BY BLANK LINES ARE TO BE EXECUTED TOGETHER
* DUE TO PROPER SETUP PURPOSES. I.E. THE LOCATION OR ADDRESS HAS TO BE
* INITIALIZED PROPERLY BEFORE THE INSTRUCTION CAN BE ACCOMPLISHED.
*
```

```

037536 000004
7534 037540 005237 177572
7535 037544 010200
7536 037546 170530
7537 037550 170537 117760
7538
7539 037554 010200
7540 037556 170430
7541 037560 170437 117760
7542
7543 037564 010200
7544 037566 177030
7545 037570 177037 117760
7546
7547 037574 010200
7548 037576 172430
7549 037600 172437 117760
7550
7551 037604 010200
7552 037606 170330
7553 037610 170337 117760
7554 037614 005037 177572

```

```

TST100: SCOPE
          INC      MMRO          ;TURN ON MEMORY MANAGEMENT.
          MOV      R2,R0         ;SETTING UP R0.
          TSTF    @(R0)+        ;TESTING BLOCK 21-N.
          TSTF    DATA

          MOV      R2,R0         ;CORRECTING R0.
          CLRF    @(R0)+        ;TESTING BLOCKS 27-U, 27-T AND 27-R.
          CLRF    DATA

          MOV      R2,R0         ;CORRECTING R0.
          LDCIF   @(R0)+,ACO     ;TESTING BLOCKS 28-L, 28-N AND 28-P.
          LDCIF   DATA,ACO

          MOV      R2,R0         ;CORRECTING R0.
          LDF     @(R0)+,ACO     ;TESTING BLOCKS 4-R, 4-T, 4-X, 4-Z AND 4-BB.
          LDF     DATA,ACO

          MOV      R2,R0         ;CORRECTING R0.
          STST    @(R0)+        ;TESTING BLOCKS 33-L, 33-N AND 33-P.
          STST    DATA
          CLR      MMRO         ;TURN OFF MEMORY MANAGEMENT.

```

7555

```

.SBTTL TEST # 101 - ENABLE D-SPACE & SEE I-SPACE FORCED, MODE 4
:*****
:*TEST 101      ENABLE D-SPACE & SEE I-SPACE FORCED, MODE 4
:*
:* THIS IS A TEST THAT WILL ENABLE D-SPACE AND MAKE IT RESIDENT SO THAT AN
:* INSTRUCTION THAT ACCESSES I-SPACE WHEN IT NORMALLY SHOULDN'T WILL CAUSE A
:* MEMORY MANAGEMENT TRAP/ABORT.
:* ****ALL REFERENCES TO MICRO-FLOWS REFFR TO *FP11-F-2 REV A* FLOWS*****
:* THE COMMENTS FOR EACH TEST LINE ARE WRITTEN SO YOU CAN GO TO THE MICRO FLOWS
:* AND PINPOINT THE PROBLEM AREA. FROM THERE, HARDWARE ANALYSIS SHOULD BE EASIER.
:* INSTRUCTION GROUPS ISOLATED BY BLANK LINES ARE TO BE EXECUTED TOGETHER
:* DUE TO PROPER SETUP PURPOSES. I.E. THE LOCATION OR ADDRESS HAS TO BE
:* INITIALIZED PROPERLY BEFORE THE INSTRUCTION CAN BE ACCOMPLISHED.
:*
:*****

```

```

037620 000004
7556 037622 005237 177572
7557 037626 010200
7558 037630 172440
7559 037632 005037 177572

```

```

TST101: SCOPE
        INC      MMRO          ;TURN ON MEMORY MANAGEMENT.
        MOV      R2,R0        ;SETTING UP R0.
        LDF     -(R0),ACO     ;TESTING BLOCKS 4-J, 4-X, 4-Z AND 4-BB.
        CLR     MMRO         ;TURN OFF MEMORY MANAGEMENT.

```

7560

```
.SBTTL TEST # 102 - ENABLE D-SPACE & SEE I-SPACE FORCED, MODE 5
*****
*TEST 102      ENABLE D-SPACE & SEE I-SPACE FORCED, MODE 5
*
* THIS IS A TEST THAT WILL ENABLE D-SPACE AND MAKE IT RESIDENT SO THAT AN
* INSTRUCTION THAT ACCESSES I-SPACE WHEN IT NORMALLY SHOULDN'T WILL CAUSE A
* MEMORY MANAGEMENT TRAP/ABORT.
* ****ALL REFERENCES TO MICRO-FLOWS REFER TO *FP11-F-2 REV A* FLOWS*****
* THE COMMENTS FOR EACH TEST LINE ARE WRITTEN SO YOU CAN GO TO THE MICRO FLOWS
* AND PINPOINT THE PROBLEM AREA. FROM THERE, HARDWARE ANALYSIS SHOULD BE EASIER.
* INSTRUCTION GROUPS ISOLATED BY BLANK LINES ARE TO BE EXECUTED TOGETHER
* DUE TO PROPER SETUP PURPOSES. I.E. THE LOCATION OR ADDRESS HAS TO BE
* INITIALIZED PROPERLY BEFORE THE INSTRUCTION CAN BE ACCOMPLISHED.
*
```

```
037636 000004
7561 037640 005237 177572
7562 037644 010300
7563 037646 170550
7564
7565 037650 010300
7566 037652 170450
7567
7568 037654 010300
7569 037656 177050
7570
7571 037660 010300
7572 037662 172450
7573
7574 037664 010300
7575 037666 170350
7576 037670 005037 177572
```

```
*****
TST102: SCOPE
      INC      MMRO      ;TURN ON MEMORY MANAGEMENT.
      MOV      R3,R0     ;SETTING UP R0.
      TSTF     @-(R0)    ;TESTING BLOCK 21-U.
      MOV      R3,R0     ;CORRECTING R0.
      CLRF     @-(R0)    ;TESTING BLOCKS 27-X, 27-T AND 27-R.
      MOV      R3,R0     ;CORRECTING R0.
      LDCIF    @-(R0),A0 ;TESTING BLOCKS 28-S, 28-N AND 28-P.
      MOV      R3,R0     ;CORRECTING R0.
      LDF      @-(R0),A0 ;TESTING BLOCKS 4-U, 4-T, 4-X, 4-Z AND 4-BB.
      MOV      R3,R0     ;CORRECTING R0.
      STST     @-(R0)    ;TESTING BLOCKS 33-S, 33-N AND 33-P.
      CLR      MMRO     ;TURN OFF MEMORY MANAGEMENT.
```

7577

```

.SBTTL TEST # 103 - ENABLE D-SPACE & SEE I-SPACE FORCED, MODE 6
:*****
:*TEST 103      ENABLE D-SPACE & SEE I-SPACE FORCED, MODE 6
:*
:* THIS IS A TEST THAT WILL ENABLE D-SPACE AND MAKE IT RESIDENT SO THAT AN
:* INSTRUCTION THAT ACCESSES I-SPACE WHEN IT NORMALLY SHOULDN'T WILL CAUSE A
:* MEMORY MANAGEMENT TRAP/ABORT.
:* *****ALL REFERENCES TO MICRO-FLOWS REFER TO *FP11-F-2 REV A* FLOWS*****
:* THE COMMENTS FOR EACH TEST LINE ARE WRITTEN SO YOU CAN GO TO THE MICRO FLOWS
:* AND PINPOINT THE PROBLEM AREA. FROM THERE, HARDWARE ANALYSIS SHOULD BE EASIER.
:* INSTRUCTION GROUPS ISOLATED BY BLANK LINES ARE TO BE EXECUTED TOGETHER
:* DUE TO PROPER SETUP PURPOSES. I.E. THE LOCATION OR ADDRESS HAS TO BE
:* INITIALIZED PROPERLY BEFORE THE INSTRUCTION CAN BE ACCOMPLISHED.
:*
:*****

```

```

037674 000004
7578
7579 037676 005267 137670
7580 037702 170567 060052
7581 037706 170467 060046
7582 037712 177067 060042
7583 037716 172467 060036
7584 037722 170367 060032
7585 037726 005067 137640
7586

```

```

TST103: SCOPE
      .DSABL  AMA      ;DISABLE MODE 6 TO MODE 3 CONVERSION
      INC     MMRO     ;TURN ON MEMORY MANAGEMENT.
      TSTF    DATA    ;TESTING BLOCK 21-0.
      CLRF    DATA    ;TESTING BLOCKS 27-DD, 27-T AND 27-R.
      LDCIF   DATA,ACO ;TESTING BLOCKS 28-T, 28-N AND 28-P.
      LDF     DATA,ACO ;TESTING BLOCKS 4-DD, 4-T, 4-X, 4-Z AND 4-BB.
      STST    DATA    ;TESTING BLOCKS 33-T, 33-N AND 33-P.
      CLR     MMRO     ;TURN OFF MEMORY MANAGEMENT.
      .ENABL  AMA      ;REENABLE MODE 6 TO MODE 3 CONVERSION

```

7587

.SBTTL TEST # 104 - ENABLE D-SPACE & SEE I-SPACE FORCED, MODE 7

*TEST 104 ENABLE D-SPACE & SEE I-SPACE FORCED, MODE 7

*

* THIS IS A TEST THAT WILL ENAPLE D-SPACE AND MAKE IT RESIDENT SO THAT AN
* INSTRUCTION THAT ACCESSES I-SPACE WHEN IT NORMALLY SHOULDN'T WILL CAUSE A

* MEMORY MANAGEMENT TRAP/ABORT.

* ****ALL REFERENCES TO MICRO-FLOWS REFER TO *FP11-F-2 REV A* FLOWS****

* THE COMMENTS FOR EACH TEST LINE ARE WRITTEN SO YOU CAN GO TO THE MICRO FLOWS

* AND PINPOINT THE PROBLEM AREA. FROM THERE, HARDWARE ANALYSIS SHOULD BE EASIER.

* INSTRUCTION GROUPS ISOLATED BY BLANK LINES ARE TO BE EXECUTED TOGETHER

* DUE TO PROPER SETUP PURPOSES. I.E. THE LOCATION OR ADDRESS HAS TO BE

* INITIALIZED PROPERLY BEFORE THE INSTRUCTION CAN BE ACCOMPLISHED.

*

TEST 104: SCOPE

7588	037732	000004	
7588	037734	005237	177572
7589	037740	010200	
7590	037742	170470	000000
7591	037746	170477	060016
7592	037752	177070	000000
7593	037756	177077	060006
7594	037762	172470	000000
7595	037766	172477	057776
7596	037772	170370	000000
7597	037776	170377	057766
7598	040002	005037	177572
7599			
7600	040006	000431	

INC	MMRO	:TURN ON MEMORY MANAGEMENT.
MOV	R2,R0	:SETTING UP R0.
CLRF	@0(R0)	:TESTING BLOCKS 27-GG, 27-JJ, 27-T AND 27-R.
CLRF	@DATA+10	
LDCIF	@0(R0),ACC	:TESTING BLOCKS 28-W, 28-Z, 28-N AND 28-P.
LDCIF	@DATA+10,ACO	
LDF	@0(R0),ACO	:TESTING BLOCKS 4-GG, 4-JJ, 4-T, 4-X 4-Z AND 4-BB.
LDF	@DATA+10,ACO	
STST	@0(R0)	:TESTING BLOCKS 33-W, 33-Z, 33-N AN 33-P.
STST	@DATA+10	
CLR	MMRO	:TURN OFF MEMORY MANAGEMENT.
BR	ENDTST	:BRANCH TO ENI OF TEST ROUTINE.

7601					.SBTTL	TRAP HANDLER FOR UNEXPECTED MEMORY MANAGEMENT ABORTS	
7602	040010	042737	000001	177572	TRAPv: BIC	#1,MMR0	:TURN OFF MEMORY MANAGEMENT.
7603	040016	013737	177572	001240	MOV	MMR0,\$TMP3	:TRANSFER MMR0 TO \$TMP3 FOR ERROR PRINTING.
7604	040024	005237	001240		INC	\$TMP3	:REPLACE BIT CLEARED TURNING OFF MEMORY MANAGEMENT.
7605	040030	013737	177576	001236	MOV	MMR2,\$TMP2	:MOVE THE TRAP INSTRUCTION ADDRESS TO \$TMP13.
7606	040036	005037	177572		CLR	MMR0	:CLEAR ERROR BITS.
7607	040042	012637	001266		MOV	(SP)+,\$TMP16	:POP STACK AND SAVE 1ST CONTENTS.
7608	040046	012637	001270		MOV	(SP)+,\$TMP17	:POP STACK AGAIN AND SAVE 2ND CONTENTS.
7609	040052	104362			ERROR	+362	:FPP TRAP/ABORT ERROR CALL.
7610	040054	013746	001270		MOV	\$TMP17,-(SP)	:PUSH 2ND SAVED CONTENTS BACK ON STACK.
7611	040060	013746	001266		MOV	\$TMP16,-(SP)	:PUSH 1ST SAVED CONTENTS BACK ON STACK.
7612	040064	005237	177572		INC	MMR0	:TURN ON MEMORY MANAGEMENT.
7613	040070	000002			RTI		:RETURN FROM INTERRUPT.

7614	040072	005037	177572		ENDTST: CLR	MMRO	:TURN OFF MEMORY MANAGEMENT.
7615	040076	013737	001262	000250		MOV \$TMP14,MMVECT	:RESTORE MMVECT TO ITS ORIGINAL CONTENTS.
7616	040104				IDONE:		
	040104	104412			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?).
							:MAKE SURE MEMORY MANAGEMENT IS OFF.
7617	040106	005037	177572		CLR MMRO		:LOAD FLOATING POINT STATUS.
7618	040112	170127	040000		LDFPS #40000		:CLEAR THE TEMPORARY LOCATION.
7619	040116	005037	001272		CLR \$TMP20		:CLEAR UPPER BYTE - ALTERNATING BITS IN LOWER BYTE.
7620	040122	012737	000252	043014	MOV #252,STORE		:MOVE ALTERNATING BITS TO 2ND WORD.
7621	040130	012737	125252	043016	MOV #125252,STORE+2		:MOVE ALTERNATING BITS TO 3RD WORD.
7622	040136	012737	125252	043020	MOV #125252,STORE+4		:MOVE ALTERNATING BITS TO 4TH WORD.
7623	040144	012737	125252	043022	MOV #125252,STORE+6		
7624	040152	172437	043014		LDF STORE,AC0		:LOAD AC0.
7625	040156	172537	043014		LDF STORE,AC1		:LOAD AC1.
7626	040162	172637	043014		LDF STORE,AC2		:LOAD AC2.
7627	040166	172737	043014		LDF STORE,AC3		:LOAD AC3.
7628	040172	012700	043014		MOV #STORE,R0		:MOVE ADDRESS OF STORE TO R0.
7629	040176	012701	000030		MOV #30,R1		:MOVE LOOP COUNTER (CLEARING 30 WORDS) TO R1.
7630	040202	005020			1\$: CLR (R0)+		:CLEAR THE WORD.
7631	040204	077102			SOB R1,1\$:SUBTRACT 1 FROM R1 AND BRANCH IF NOT 0.
7632	040206	174037	043014		STF AC0,STORE		:STORE AC0.
7633	040212	174137	043024		STF AC1,STORE+10		:STORE AC1.
7634	040216	174237	043034		STF AC2,STORE+20		:STORE AC2.
7635	040222	174337	043044		STF AC3,STORE+30		:STORE AC3.
7636							
7637	040226	012737	077406	172320	MOV #77406,KDPDR0		:MAKE KDPDR0 RESIDENT.
7638	040234	012737	077406	172322	MOV #77406,KDPDR1		:MAKE KDPDR1 RESIDENT.
7639	040242	012737	077400	172324	MOV #77400,KDPDR2		:MAKE KDPDR2 NON-RESIDENT.
7640	040250	012737	077406	172326	MOV #77406,KDPDR3		:MAKE KDPDR3 RESIDENT.
7641	040256	012737	077406	172336	MOV #77406,KDPDR7		:MAKE KDPDR7 RESIDENT.
7642							
7643	040264	012737	077406	172300	MOV #77406,KIPDR0		:MAKE KIPDR0 RESIDENT.
7644	040272	012737	077406	172302	MOV #77406,KIPDR1		:MAKE KIPDR1 RESIDENT.
7645	040300	012737	077406	172304	MOV #77406,KIPDR2		:MAKE KIPDR2 RESIDENT.
7646	040306	012737	077406	172306	MOV #77406,KIPDR3		:MAKE KIPDR3 RESIDENT.
7647	040314	012737	077406	172316	MOV #77406,KIPDR7		:MAKE KIPDR7 RESIDENT.
7648							
7649	040322	005037	172360		CLR KDPAR0		:MAP D-PAGE 0 FOR 0-4k.
7650	040326	012737	000200	172362	MOV #200,KDPAR1		:MAP D-PAGE 1 FOR 4-8k.
7651	040334	012737	000400	172364	MOV #400,KDPAR2		:MAP D-PAGE 2 FOR 8-12k.
7652	040342	012737	000600	172366	MOV #600,KDPAR3		:MAP D-PAGE 3 FOR 12-16k.
7653	040350	012737	177600	172376	MOV #177600,KDPAR7		:MAP D-PAGE 7 FOR I/O PAGE.
7654							
7655	040356	005037	172340		CLR KIPAR0		:MAP I-PAGE 0 FOR 0-4k.
7656	040362	012737	000200	172342	MOV #200,KIPAR1		:MAP I-PAGE 1 FOR 4-8k.
7657	040370	012737	000400	172344	MOV #400,KIPAR2		:MAP I-PAGE 2 FOR 8-12k.
7658	040376	012737	000600	172346	MOV #600,KIPAR3		:MAP I-PAGE 3 FOR 12-16k.
7659	040404	012737	177600	172356	MOV #177600,KIPAR7		:MAP I-PAGE 7 FOR I/O PAGE.
7660							
7661	040412	012705	043074		MOV #BYTABL,R5		:SET UP BYTE TABLE POINTER R5.
7662	040416	013737	000250	001262	MOV MMVECT,\$TMP14		:TEMPORARILY STORE THE MMVECT VALUE.
7663	040424	012737	041636	000250	MOV #TRPV,MMVECT		:SET UP FOR FP TRAPS FOR THIS TEST.
7664	040432	012737	000340	000252	MOV #340,MMVECT+2		
7665	040440	013737	000020	040522	MOV IOTRAP,SAVIOI		:STORE THE IOTRAP VALUE IN SAVIOI


```

7665 040446 012737 000340 000022      MOV      #340,IOTRAP+2
7667
7668 040454 012737 000024 172516      MOV      #24,MMR3      ;TURN ON 22-BIT KERNEL D-SPACE.
7669 040462 012737 043000 043010      MOV      #NODAT,NODAT+10 ;SET UP ADDRESS POINTER.
7670 040470 012700 043000      MOV      #NODAT,R0      ;SET UP R0.
7671 040474 012702 043010      MOV      #NODAT+10,R2   ;SET UP R2.
7672 040500 012703 043012      MOV      #NODAT+12,R3   ;SET UP R3.
7673 040504 010037 043054      MOV      R0,STORE+40    ;STORE R0.
7674 040510 010237 043056      MOV      R2,STORE+42    ;STORE R2.
7675 040514 010337 043060      MOV      R3,STORE+44    ;STORE R3.
7676 040520 000401      BR       TST105         ;BRANCH OVER LOCATION SAVIOT
7677
7678 040522 000000      SAVIOT: .WORD 0        ;LOCATION TO SAVE THE SCOPE VECTOR ADDRESS

```

7691

```
.SBTTL TEST # 105 - AUTO INCREMENT/DECREMENT TEST, MODE 0
*****
*TEST 105      AUTO INCREMENT/DECREMENT TEST, MODE 0
*
* THIS TEST INSURES THAT AUTO INCREMENT/DECREMENT WORKS PROPERLY AND
* *ONLY* WHEN IT IS SUPPOSED TO. THIS IS DONE BY ENABLING 22-BIT KERNEL
* D-SPACE, BUT MAKING IT NON-RESIDENT, FORCING A MEMORY MANAGEMENT TRAP
* CONDITION. THIS ENABLES EXAMINING OF SR1 FOR PROPER CONTENTS.
* *****ALL REFERENCES TO MICRO-FLGWS REFER TO *FP11-F-2 REV A* FLOWS*****
* THE COMMENTS FOR EACH TEST LINE ARE WRITTEN SO YOU CAN GO TO THE MICRO
* FLOW AND PINPOINT THE PROBLEM AREA. FROM THERE, HARDWARE ANALYSIS SHOULD
* BE EASIER.
*
*****
```

```
7692 040524 000004
      040526
      040526 104413
7693
7694
7695
7696 040530 012737 041570 CGOC2G
7697 040536 005237 177572
7698 040542 170501
7699 040544 170401
7700 040546 177001
7701 040550 172401
7702 040552 170301
7703 040554 005037 177572
7704 040560 172437 043014
7705 040564 172537 043014
7706 040570 010046
7707 040572 010246
7708 040574 013737 040522 000020
```

```
TST105: SCOPE
1$:
      LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
      ;* THE FOLLOWING TESTS ARE FOR MODE 0 REG 1 (THESE SHOULD *NOT* ABORT).
      MOV #FALTRP,IOTRAP;SET UP FOR FAILURE OF TRAPS FOR THIS TEST.
      INC MMRO        ;TURN ON MEMORY MANAGEMENT.
      TSTF R1         ;FDST-NOTCLR PAGE 21.
      CLRF R1         ;FDST MODES PAGE 27.
      LDCIF R1,ACO    ;SOURCE MODES PAGE 28.
      LDF R1,ACO      ;FSRC MODES PAGE 4.
      STST R1        ;DEST MODES PAGE 33.
      CLR MMRO       ;TURN OFF MEMORY MANAGEMENT.
      LDF STORE,ACO  ;RESTORE ACO.
      LDF STORE,AC1  ;RESTORE AC1.
      MOV R0,-(SP)   ;SAVE R0 FOR NEXT TEST
      MOV R2,-(SP)   ;SAVE R2 FOR NEXT TEST
      MOV SAVIOT,IOTRAP ;RESTORE SCOPE TRAP VECTOR
```

7709

```
.SBTTL TEST # 106 - AUTO INCREMENT/DECREMENT TEST, MODE 1
:*****
:*TEST 106      AUTO INCREMENT/DECREMENT TEST, MODE 1
:*
:* THIS TEST INSURES THAT AUTO INCREMENT/DECREMENT WORKS PROPERLY AND
:* *ONLY* WHEN IT IS SUPPOSED TO. THIS IS DONE BY ENABLING 22-BIT KERNEL
:* D-SPACE, BUT MAKING IT NON-RESIDENT, FORCING A MEMORY MANAGEMENT TRAP
:* CONDITION. THIS ENABLES EXAMINING OF SR1 FOR PROPER CONTENTS.
:* *****ALL REFERENCES TO MICRO-FLOWS REFER TO *FP11-F-2 REV A* FLOWS*****
:* THE COMMENTS FOR EACH TEST LINE ARE WRITTEN SO YOU CAN GO TO THE MICRO
:* FLOW AND PINPOINT THE PROBLEM AREA. FROM THERE, HARDWARE ANALYSIS SHOULD
:* BE EASIER.
:*
:*****
```

```
040602 000004
7710 040604 0126C2
7711 040606 012600
7712 040610 012737 041570 000020
7713 040616 005237 177572
7714 040622 010001
7715 040624 010C04
7716 040626 17C511
7717 040630 C00004
7718 040632 170411
7719 040634 000004
7720 040636 177011
7721 040640 000004
7722 040642 172411
7723 040644 000004
7724 040646 170311
7725 040650 000004
7726 040652 005037 177572
7727 040656 010046
7728 040660 010246
7729 040662 013737 040520 000000
```

```
TST106: SCOPE
      MOV      (SP)+,R2      ;RESTORE R2 FOR THIS TEST
      MCV      (SP)+,R0      ;RESTORE R0 FOR THIS TEST
      MOV      #FALTRP,IOTRAP;SET UP FOR FAILURE OF TRAPS FOR THIS TEST.
      INC      MMR0          ;TURN ON MEMORY MANAGEMENT.
MODE1: MOV      R0,R1          ;SET UP R1.
      MOV      R0,R4          ;MOVE 'START' VALUE INTO R4.
      TSTF     (R1)           ;FDST-NOTCLR PAGE 21.
      IOT                      ;FORCE A TRAP.
      CLRF     (R1)           ;FDST MODES PAGE 27.
      IOT                      ;FORCE A TRAP.
      LDCIF    (R1),ACO       ;SOURCE MODES PAGE 28.
      IOT                      ;FORCE A TRAP.
      LDF      (R1),ACO       ;FSRC MODES PAGE 4.
      IOT                      ;FORCE A TRAP.
      STST     (R1)           ;DEST MODES PAGE 33.
      IOT                      ;FORCE A TRAP.
      CLR      MMR0          ;TURN OFF MEMORY MANAGEMENT.
      MOV      R0,-(SP)       ;SAVE R0 FOR NEXT TEST
      MOV      R2,-(SP)       ;SAVE R2 FOR NEXT TEST
      MCV      SAV,IOT,IOTRAP ;RESTORE SCOPE TRAP VECTOR
```

7730

```
.SBTTL TEST # 107 - AUTO INCREMENT/DECREMENT TEST, MODE 2
*****
*TEST 107      AUTO INCREMENT/DECREMENT TEST, MODE 2
*
* THIS TEST INSURES THAT AUTO INCREMENT/DECREMENT WORKS PROPERLY AND
* *ONLY* WHEN IT IS SUPPOSED TO. THIS IS DONE BY ENABLING 22-BIT KERNEL
* D-SPACE, BUT MAKING IT NON-RESIDENT, FORCING A MEMORY MANAGEMENT TRAP
* CONDITION. THIS ENABLES EXAMINING OF SR1 FOR PROPER CONTENTS.
* *****ALL REFERENCES TO MICRO-FLOWS REFER TO *FP11-F-2 REV A* FLOWS*****
* THE COMMENTS FOR EACH TEST LINE ARE WRITTEN SO YOU CAN GO TO THE MICRO
* FLOW AND PINPOINT THE PROBLEM AREA. FROM THERE, HARDWARE ANALYSIS SHOULD
* BE EASIER.
*
```

```
040670 000004
7731 040672 012737 041570 000020
7732 040700 012602
7733 040702 012600
7734 040704 005237 177572
7735 040710 170521
7736 040712 000004
7737
7738 040714 010001
7739 040716 170421
7740 040720 000004
7741
7742 040722 010001
7743 040724 177021
7744 040726 000004
7745
7746 040730 010001
7747 040732 172421
7748 040734 000004
7749
7750 040736 010001
7751 040740 170321
7752 040742 000004
7753 040744 005037 177572
7754 040750 013737 040522 000020
7755 040756 010046
7756 040760 010246
```

```
TST107: SCOPE
MOV #FALTRP,IOTRAP;SET UP FOR FAILURE OF TRAPS FOR THIS TEST.
MCMV (SP)+,R2 ;RESTORE R2 FOR THIS TEST
MOV (SP)+,RC ;RESTORE R0 FOR THIS TEST
INC MMRO ;TURN ON MEMORY MANAGEMENT.
TSTF (R1)+ ;FDST-NOTCLR PAGE 21.
LABEL1: IOT ;FORCE A TRAP.

MOV R0,R1 ;CORRECT R1.
CLRF (R1)+ ;FDST MODES PAGE 27.
IOT ;FORCE A TRAP.

MOV R0,R1 ;CORRECT R1.
LDCIF (R1)+,ACO ;SOURCE MODES PAGE 28.
IOT ;FORCE A TRAP.

MOV R0,R1 ;CORRECT R1.
LDF (R1)+,ACO ;FSRC MODES PAGE 4.
IOT ;FORCE A TRAP.

MOV R0,R1 ;CORRECT R1.
STST (R1)+ ;DEST MODES PAGE 33.
IOT ;FORCE A TRAP.
CLR MMRO ;TURN OFF MEMORY MANAGEMENT.
MOV SAVIOT,IOTRAP ;RESTORE SCOPE TRAP VECTOR
MOV R0,-(SP) ;SAVE R0 FOR NEXT TEST
MOV R2,-(SP) ;SAVE R2 FOR NEXT TEST
```

7757

```
.SBTTL TEST # 110 - AUTO INCREMENT/DECREMENT TEST, MODE 3
:*****
:*TEST 110 AUTO INCREMENT/DECREMENT TEST, MODE 3
:*
:* THIS TEST INSURES THAT AUTO INCREMENT/DECREMENT WORKS PROPERLY AND
:* *ONLY* WHEN IT IS SUPPOSED TO. THIS IS DONE BY ENABLING 22-BIT KERNEL
:* D-SPACE, BUT MAKING IT NON-RESIDENT, FORCING A MEMORY MANAGEMENT TRAP
:* CONDITION. THIS ENABLES EXAMINING OF SR1 FOR PROPER CONTENTS.
:*****ALL REFERENCES TO MICRO-FLOWS REFER TO *FP11-F-2 REV A* FLOWS*****
:* THE COMMENTS FOR EACH TEST LINE ARE WRITTEN SO YOU CAN GO TO THE MICRO
:* FLOW AND PINPOINT THE PROBLEM AREA. FROM THERE, HARDWARE ANALYSIS SHOULD
:* BE EASIER.
:*
```

```
*****
TST110: SCOPE
7758 040762 000004 041570 000020 MOV #FALTRP,IOTRAP;SET UP FOR FAILURE OF TRAPS FOR THIS TEST.
7759 040764 012737 MCV (SP)+,R2 ;RESTORE R2 FOR THIS TEST
7760 040774 012600 MOV (SP)+,R0 ;RESTORE R0 FOR THIS TEST
7761 040776 005237 177572 INC MMRO ;TURN ON MEMORY MANAGEMENT.
7762 041002 010201 MOV R2,R1 ;SET UP R1 FOR MODE 3.
7763 041004 010204 MOV R2,R4 ;MOVE 'START' VALUE INTO R4.
7764 041006 170531 TSTF @(R1)+ ;FDST-NOTCLR PAGE 21.
7765 041010 000004 IOT ;FORCE A TRAP.
7766 041012 170537 043000 TSTF NODAT
7767 041016 000004 IOT ;FORCE A TRAP.
7768
7769 041020 010201 MOV R2,R1 ;CORRECT R1.
7770 041022 170431 CLRF @(R1)+ ;FDST MODES PAGE 27.
7771 041024 000004 IOT ;FORCE A TRAP.
7772 041026 170437 043000 CLRF NODAT
7773 041032 000004 IOT ;FORCE A TRAP.
7774
7775 041034 010201 MOV R2,R1 ;CORRECT R1.
7776 041036 177031 LDCIF @(R1)+,ACO ;SOURCE MODES PAGE 28.
7777 041040 000004 IOT ;FORCE A TRAP.
7778 041042 177037 043000 LDCIF NODAT,ACO
7779 041046 000004 IOT ;FORCE A TRAP.
7780
7781 041050 010201 MOV R2,R1 ;CORRECT R1.
7782 041052 172431 LDF @(R1)+,ACO ;FSRC MODES PAGE 4.
7783 041054 000004 IOT ;FORCE A TRAP.
7784 041056 172437 043000 LDF NODAT,ACO
7785 041062 000004 IOT ;FORCE A TRAP.
7786
7787 041064 010201 MOV R2,R1 ;CORRECT R1.
7788 041066 170331 STST @(R1)+ ;DEST MODES PAGE 33.
7789 041070 000004 IOT ;FORCE A TRAP.
7790 041072 170337 043000 STST NODAT
7791 041076 000004 IOT ;FORCE A TRAP.
7792 041100 005037 177572 CLR MMRO ;TURN OFF MEMORY MANAGEMENT.
7793 041104 010046 MOV R0,-(SP) ;SAVE R0 FOR NEXT TEST
7794 041106 010246 MOV R2,-(SP) ;SAVE R2 FOR NEXT TEST
7795 041110 013737 040522 000020 MCV SAVIOT,IOTRAP ;RESTORE SCOPE TRAP VECTOR
```

7796

```
.SBTTL TEST # 111 - AUTO INCREMENT/DECREMENT TEST, MODE 4
:*****
:*TEST 111 AUTO INCREMENT/DECREMENT TEST, MODE 4
:*
:* THIS TEST INSURES THAT AUTO INCREMENT/DECREMENT WORKS PROPERLY AND
:* *ONLY* WHEN IT IS SUPPOSED TO. THIS IS DONE BY ENABLING 22-BIT KERNEL
:* D-SPACE, BUT MAKING IT NON-RESIDENT, FORCING A MEMORY MANAGEMENT TRAP
:* CONDITION. THIS ENABLES EXAMINING OF SR1 FOR PROPER CONTENTS.
:* *****ALL REFERENCES TO MICRO-FLOWS REFER TO *FP11-F-2 REV A* FLOWS*****
:* THE COMMENTS FOR EACH TEST LINE ARE WRITTEN SO YOU CAN GO TO THE MICRO
:* FLOW AND PINPOINT THE PROBLEM AREA. FROM THERE, HARDWARE ANALYSIS SHOULD
:* BE EASIER.
:*
:*****
```

```
TST111: SCOPE
7797 041116 000004          MOV #FALTRP,IOTRAP;SET UP FOR FAILURE OF TRAPS FOR THIS TEST.
7798 041120 012737 041570 000020      MOV (SP)+,R2 ;RESTORE R2 FOR THIS TEST
7799 041126 012602          MOV (SP)+,R0 ;RESTORE R0 FOR THIS TEST
7800 041130 012600          MOV MMR0 ;TURN ON MEMORY MANAGEMENT.
7801 041132 005237 177572      INC R2,R1 ;SET UP R1 FOR MODE 4.
7802 041136 010201          TSTF -(R1) ;FDST-NOTCLR PAGE 21.
7803 041140 170541          IOT ;FORCE A TRAP.
7804 041142 000004          MOV R2,R1 ;CORRECT R1.
7805 041144 010201          CLRF -(R1) ;FDST MODES PAGE 27.
7806 041146 170441          IOT ;FORCE A TRAP.
7807 041150 000004          MOV R2,R1 ;CORRECT R1.
7808 041152 010201          LDCIF -(R1),ACO ;SOURCE MODES PAGE 28.
7809 041154 177041          IOT ;FORCE A TRAP.
7810 041156 000004          MOV R2,R1 ;CORRECT R1.
7811 041160 010201          LDF -(R1),ACO ;FSRC MODES PAGE 4.
7812 041162 172441          IOT ;FORCE A TRAP.
7813 041164 000004          MOV R2,R1 ;CORRECT R1.
7814 041166 010201          STST -(R1) ;DEST MODES PAGE 33.
7815 041170 170341          IOT ;FORCE A TRAP.
7816 041172 000004          CLR MMR0 ;TURN OFF MEMORY MANAGEMENT.
7817 041174 005037 177572      MOV R0,-(SP) ;SAVE R0 FOR NEXT TEST
7818 041176 010046          MOV R2,-(SP) ;SAVE R2 FOR NEXT TEST
7819 041178 010246          MOV SAVIOT,IOTRAP ;RESTORE SCOPE TRAP VECTOR
7820 041180 013737 040522 000020
```

7824

```
.SBTTL TEST # 112 - AUTO INCREMENT/DECREMENT TEST, MODE 5
*****
*TEST 112 AUTO INCREMENT/DECREMENT TEST, MODE 5
*
* THIS TEST INSURES THAT AUTO INCREMENT/DECREMENT WORKS PROPERLY AND
* *ONLY* WHEN IT IS SUPPOSED TO. THIS IS DONE BY ENABLING 22-BIT KERNEL
* D-SPACE, BUT MAKING IT NON-RESIDENT, FORCING A MEMORY MANAGEMENT TRAP
* CONDITION. THIS ENABLES EXAMINING OF SR1 FOR PROPER CONTENTS.
* *****ALL REFERENCES TO MICRO-FLGWS REFER TO *FP11-F-2 REV A* FLOWS*****
* THE COMMENTS FOR EACH TEST LINE ARE WRITTEN SO YOU CAN GO TO THE MICRO
* FLOW AND PINPOINT THE PROBLEM AREA. FROM THERE, HARDWARE ANALYSIS SHOULD
* BE EASIER.
*
```

```
TST112: SCOPE
041212 000004
7825 041214 012737 041570 000020 MOV #FALTRP,IOTRAP;SET UP FOR FAILURE OF TRAPS FOR THIS TEST.
7826 041222 012602 MOV (SP)+,R2 ;RESTORE R2 FOR THIS TEST
7827 041224 012600 MOV (SP)+,R0 ;RESTORE R0 FOR THIS TEST
7828 041226 005237 177572 INC MMRO ;TURN ON MEMORY MANAGEMENT.
7829 041232 010301 MOV R3,R1 ;SET UP R1 FOR MODE 5.
7830 041234 010304 MOV R3,R4 ;MOVE 'START' VALUE INTO R4.
7831 041236 170551 TSTF @-(R1) ;FDST-NOTCLR PAGE 21.
7832 041240 000004 IOT ;FORCE A TRAP.
7833
7834 041242 010301 MOV R3,R1 ;CORRECT R1.
7835 041244 170451 CLRF @-(R1) ;FDST MODES PAGE 27.
7836 041246 000004 IOT ;FORCE A TRAP.
7837
7838 041250 010301 MOV R3,R1 ;CORRECT R1.
7839 041252 177051 LDCIF @-(R1),ACO ;SOURCE MODES PAGE 28.
7840 041254 000004 IOT ;FORCE A TRAP.
7841
7842 041256 010301 MOV R3,R1 ;CORRECT R1.
7843 041260 172451 LDF @-(R1),ACO ;FSRC MODES PAGE 4.
7844 041262 000004 IOT ;FORCE A TRAP.
7845
7846 041264 010301 MOV R3,R1 ;CORRECT R1.
7847 041266 170351 STST @-(R1) ;DEST MODES PAGE 33.
7848 041270 000004 IOT ;FORCE A TRAP.
7849 041272 005037 177572 CLR MMRO ;TURN OFF MEMORY MANAGEMENT.
7850 041276 013737 040522 000020 MOV SAVIO*,IOTRAP ;RESTORE SCOPE TRAP VECTOR
```

7851

.SBTT: TEST # 113 - AUTO INCREMENT/DECREMENT TEST, MODE 6

TEST 113 AUTO INCREMENT/DECREMENT TEST, MODE 6

* THIS TEST INSURES THAT AUTO INCREMENT/DECREMENT WORKS PROPERLY AND
* *ONLY* WHEN IT IS SUPPOSED TO. THIS IS DONE BY ENABLING 22-BIT KERNEL
* D-SPACE, BUT MAKING IT NON-RESIDENT, FORCING A MEMORY MANAGEMENT TRAP
* CONDITION. THIS ENABLES EXAMINING OF SR1 FOR PROPER CONTENTS.
* *****ALL REFERENCES TO MICRO-FLOWS REFER TO *FP11-F-2 REV A* FLOWS*****
* THE COMMENTS FOR EACH TEST LINE ARE WRITTEN SO YOU CAN GO TO THE MICRO
* FLOW AND PINPOINT THE PROBLEM AREA. FROM THERE, HARDWARE ANALYSIS SHOULD
* BE EASIER.
*

TST113: SCOPE

7852	041304	000004			.DSABL	AMA	:DISABLE MODE 6 TO MODE 3 CONVERSIONS
7853	041306	012767	041570	136504	MCV	#FALTRP,IOTRAP;	SET UP FOR FAILURE OF TRAPS FOR THIS TEST.
7854	041314	010301			MOV	R3,R1	:SET UP R1 FOR MODE 6.
7855	041316	010304			MOV	R3,R4	:MOVE 'START' VALUE INTO R4.
7856	041320	005267	136246		INC	MMRO	:TURN ON MEMORY MANAGEMENT.
7857	041324	170561	000000		TSTF	0(R1)	:FDST-NOTCLR PAGE 21.
7858	041330	000004			IOT		:FORCE A TRAP.
7859	041332	170567	001442		TSTF	NODAT	
7860	041336	000004			IOT		:FORCE A TRAP.
7861	041340	170461	000000		CLRF	0(R1)	:FDST MODES PAGE 27.
7862	041344	000004			IOT		:FORCE A TRAP.
7863	041346	170467	001426		CLRF	NODAT	
7864	041352	000004			IOT		:FORCE A TRAP.
7865	041354	177061	000000		LDCIF	0(R1),ACO	:SOURCE MODES PAGE 28.
7866	041360	000004			IOT		:FORCE A TRAP.
7867	041362	177067	001412		LDCIF	NODAT,ACO	
7868	041366	000004			IOT		:FORCE A TRAP.
7869	041370	172461	000000		LDF	0(R1),ACO	:FSRC MODES PAGE 4.
7870	041374	000004			IOT		:FORCE A TRAP.
7871	041376	172467	001376		LDF	NODAT,ACO	
7872	041402	000004			IOT		:FORCE A TRAP.
7873	041404	170361	000000		STST	0(R1)	:DEST MODES PAGE 33.
7874	041410	000004			IOT		:FORCE A TRAP.
7875	041412	170367	001362		STST	NODAT	
7876	041416	000004			IOT		:FORCE A TRAP.
7877	041420	005067	136146		CLR	MMRO	:TURN OFF MEMORY MANAGEMENT.
7878	041424	010246			MOV	R2,-(SP)	:SAVE R2 FOR NEXT TEST
7879	041426	016767	177070	136364	MOV	SAVIO*,IOTRAP	:RESTORE SCOPE TRAP VECTOR
7880					.ENABL	AMA	:REENABLE MODE 6 TO MODE 3 CONVERSIONS

7881

```
.SBTTL TEST # 114 - AUTO INCREMENT/DECREMENT TEST, MODE 7
*****
*TEST 114 AUTO INCREMENT/DECREMENT TEST, MODE 7
*
* THIS TEST INSURES THAT AUTO INCREMENT/DECREMENT WORKS PROPERLY AND
* *ONLY* WHEN IT IS SUPPOSED TO. THIS IS DONE BY ENABLING 22-BIT KERNEL
* D-SPACE, BUT MAKING IT NON-RESIDENT, FORCING A MEMORY MANAGEMENT TRAP
* CONDITION. THIS ENABLES EXAMINING OF SR1 FOR PROPER CONTENTS.
* *****ALL REFERENCES TO MICRO-FLOWS REFER TO *FP11-F-2 REV A* FLOWS*****
* THE COMMENTS FOR EACH TEST LINE ARE WRITTEN SO YOU CAN GO TO THE MICRO
* FLOW AND PINPOINT THE PROBLEM AREA. FROM THERE, HARDWARE ANALYSIS SHOULD
* BE EASIER.
*
*****
```

```
041434 000004
7882 041436 012737 041570 000020
7883 041444 012602
7884 041446 005237 177572
7885 041452 010201
7886 041454 010204
7887 041456 170571 000000
7888 041462 000004
7889 041464 170577 001320
7890 041470 000004
7891 041472 170471 000000
7892 041476 000004
7893 041500 170477 001304
7894 041504 000004
7895 041506 177071 000000
7896 041512 000004
7897 041514 177077 001270
7898 041520 000004
7899 041522 172471 000000
7900 041526 000004
7901 041530 172477 001254
7902 041534 000004
7903 041536 170371 000000
7904 041542 000004
7905 041544 170377 001240
7906 041550 000004
7907 041552 005037 177572
7908 041556 013737 040522 000020
7909 041564 000137 043160
```

```
TST114: SCOPE
MOV #FALTRP,IOTRAP;SET UP FOR FAILURE OF TRAPS FOR THIS TEST.
MCV (SP)+,R2 ;RESTORE R2 FOR THIS TEST
INC MMRO ;TURN ON MEMORY MANAGEMENT.
MOV R2,R1 ;SET UP R1 FOR MODE 7.
MOV R2,R4 ;MOVE 'START' VALUE TO R4.
TSTF @0(R1) ;FDST-NOTCLR PAGE 21.
IOT ;FORCE A TRAP.
TSTF @NODAT+10
IOT ;FORCE A TRAP.
CLRF @0(R1) ;FDST MODES PAGE 27.
IOT ;FORCE A TRAP.
CLRF @NODAT+10
IOT ;FORCE A TRAP.
LDCIF @0(R1),ACO ;SOURCE MODES PAGE 28.
IOT ;FORCE A TRAP.
LDCIF @NODAT+10,ACO
IOT ;FORCE A TRAP.
LDF @0(R1),ACO ;FSRC MODES PAGE 4.
IOT ;FORCE A TRAP.
LDF @NODAT+10,ACO
IOT ;FORCE A TRAP.
STST @0(R1) ;DEST MODES PAGE 33.
IOT ;FORCE A TRAP.
STST @NODAT+10
IOT ;FORCE A TRAP.
CLR MMPO ;TURN OFF MEMORY MANAGEMENT.
MOV SAVIOT,IOTRAP ;RESTORE SCOPE TRAP VECTOR
JMP ENDTES ;JUMP TO END TEST.
```

7910	041570	005057	177572		FALTRP: CLR	MMRO		;TURN OFF MEMORY MANAGEMENT.
7911	041574	011637	001260		MOV	(SP), \$TMP13		;MOVE NEXT INSTRUCTION ADDRESS TO \$TMP13.
7912								
7913								;THIS NEXT SECTION NOW CORRECTS THE CONTENTS OF \$TMP13 SO THAT IT POINTS
7914								;TO THE PREVIOUS FPP INSTRUCTION. IT DOES THIS BY SUBTRACTING 2 FROM THE
7915								;ADDRESS IN \$TMP13, REPLACING THE 170000 THAT THE BIC INSTRUCTION USES,
7916								;AND BIT CLEARING THE INSTRUCTION WITH 170000. IF THE INSTRUCTION THAT
7917								;\$TMP13 IS POINTING TO IS NOT AN FPP INSTRUCTION, THE 170000 WILL NOT
7918								;CLEAR, SATISFYING THE NEXT BRANCH. THE ADDRESS IS AGAIN CORRECTED,
7919								;AND THE TESTING PROCESS STARTS OVER. THIS CONTINUES UNTIL \$TMP13 IS
7920								;POINTING TO AN FPP INSTRUCTION, AND NORMALLY WILL NOT BE EXECUTED MORE
7921								;THAN THREE TIMES BEFORE FINDING THE INSTRUCTION.
7922	041600	162737	000002	001260	1\$:	SUB	#2, \$TMP13	;SUBTRACT 2 FROM \$TMP13.
7923	041606	012737	170000	041620		MOV	#170000, 2\$+4	;SET UP BIC DATA LOCATION.
7924	041614	047727	137440	170000	2\$:	BIC	@\$TMP13, #170000	;TEST TO SEE IF FPP INSTRUCTION.
7925	041622	001366				BNE	1\$;BRANCH BACK FOR ANOTHER TRY IF NOT.
7926	041624	012737	000364	001272		MOV	#364, \$TMP20	;MOVE FAILURE TO ABORT ERROR TO \$TMP20.
7927	041632	000137	042666			JMP	MULTER	;JUMP TO MULTIPLE ERROR HANDLER.

7928	041636	013737	177574	001236	TRPV:	MOV	SR1,\$TMP2	:MOVE SR1 TO \$TMP2 FOR TESTING.
7929	041644	013737	177576	001260		MOV	MMR2,\$TMP13	:TRANSFER ADDRESS OF INST. CAUSING TRAP TO \$TMP13.
7930	041652	005037	177572			CLR	MMR0	:TURN OFF MEMORY MANAGEMENT.
7931	041656	112737	000365	001272		MOVB	#365,\$TMP20	:MOVE 365, THE MODE 0 ERROR, TO LOWER BYTE IN ERROR POINTER.
7932	041664	022737	040622	001260		CMP	#MODE1,\$TMP13	:SEE IF INSTRUCTION CAUSING TRAP IS BEFORE MODE 1 (MODE 0).
7933	041672	002402				BLT	1\$:BRANCH AROUND MODE 0 ERROR JUMP IF NOT.
7934	041674	000137	042666			JMP	MULTER	:JUMP TO ERROR NEST.
7935	041700	017737	137354	001266	1\$:	MOV	@\$TMP13,\$TMP16	:MOVE INSTRUCTION CAUSING TRAP TO \$TMP16.
7936	041706	112737	000363	001272		MOVB	#363,\$TMP20	:MOVE 363, SR1 WRONG ERROR, TO LOWER BYTE IN ERROR POINTER.
7937	041714	005037	001240			CLR	\$TMP3	:CLEAR CALCULATED LOCATION.
7938	041720	012737	042112	001244		MOV	#65\$,\$TMP5	:MOVE NEXT CHECK ADDRESS TO \$TMP5.
7939	041726	022737	040712	001260		CMP	#LABEL1,\$TMP13	:SEE IF TRAP IS BEFORE MODE 2 REG 1 CLRF INST.
7940	041734	100053				BPL	61\$:BRANCH TO SR1=0 TEST IF SO.
7941	041736	012737	000060	041750		MOV	#60,200\$:SET UP BIC DATA POSITION.
7942	041744	043727	001266		2\$:	BIC	\$TMP16,(PC)+	:TEST TO SEE IF MODE 6 OR 7 INSTRUCTION.
7943	041750	000060			200\$:	.WORD	60	:LOCATION TO HOLD 60
7944	041752	001444				BEQ	61\$:BRANCH DIRECTLY TO BYTE TABLE TESTING IF SO.
7945								:THIS NEXT ROUTINE DETERMINES WHICH REGISTER WAS IN THE INSTRUCTION, AND
7946								:LOADS THE START AND END VALUES OF EITHER R1 OR R7 (PROGRAM COUNTER) INTO
7947								:\$TMP17 AND \$TMP3 RESPECTIVELY. THEY ARE THEN SUBTRACTED TO FIND THE
7948								:DIFFERENCE THAT ACTUALLY OCCURED. IF NO DIFFERENCE WAS FOUND, THE TEST
7949								:FOR ZERO IN SR1 IS ACCOMPLISHED. IF A DIFFERENCE IS FOUND, THE DIFFERENCE
7950								:IS SHIFTED LEFT 3 PLACES, THE TOP BYTE IS CLEARED, AND THE REGISTER
7951								:OF THE INSTRUCTION IS ADDED. \$TMP3 NOW CONTAINS WHAT SHOULD APPEAR
7952								:IN SR1, ACCORDING TO WHAT ACTUALLY HAPPENED TO THE REGISTER.
7953	041754	042737	177770	001266	4\$:	BIC	#177770,\$TMP16	:BIT CLEAR THE INSTRUCTION, LEAVING THE REG EXPOSED.
7954	041762	023727	001266	000007		CMP	\$TMP16,#7	:COMPARE REGISTER TO DETERMINE IF IT IS REG 7.
7955	041770	001405				BEQ	5\$:BRANCH TO THE REG 7 SETUP IF EQUAL TO REG 7.
7956	041772	019437	001270			MOV	R4,\$TMP17	:MOVE THE START VALUE TO \$TMP17.
7957	041776	010137	001240			MOV	R1,\$TMP3	:MOVE THE END VALUE TO \$TMP3.
7958	042002	000410				BR	6\$:BRANCH TO CONTINUE.
7959	042004	013737	001260	001270	5\$:	MOV	\$TMP13,\$TMP17	:MOVE THE START VALUE TO \$TMP17.
7960	042012	062737	000002	001270		ADD	#2,\$TMP17	:ADD 2 TO START VALUE FOR NORMAL INCREMENTING.
7961	042020	011637	001240			MOV	(SP),\$TMP3	:MOVE THE END VALUE TO \$TMP3.
7962	042024	163737	001270	001240	6\$:	SUB	\$TMP17,\$TMP3	:FIND THE DIFFERENCE THAT OCCURED.
7963	042032	001414				BEQ	61\$:BRANCH TO TEST FOR SR1=0 IF NO DIFFERENCE.
7964	042034	006337	001240			ASL	\$TMP3	:ARITHMETIC SHIFT LEFT \$TMP3 3
7965	042040	006337	001240			ASL	\$TMP3	:PLACES TO PUT DIFFERENCE FOUND
7966	042044	006337	001240			ASL	\$TMP3	:IN BITS 3 THROUGH 7.
7967	042050	042737	177400	001240		BIC	#177400,\$TMP3	:BIT CLEAR UPPER BYTE OF \$TMP3.
7968	042056	063737	001266	001240		ADD	\$TMP16,\$TMP3	:ADD THE REGISTER THAT WAS CHANGED, AND
7969	042064	111537	001276		61\$:	MOVB	(R5),\$TMP22	:MOVE EXPECTED DATA TO \$TMP22.
7970	042070	123725	001236			CMPB	\$TMP2,(R5)+	:COMPARE SR1 WITH TABLE DATA.
7971	042074	001004				BNE	62\$:BRANCH TO ERROR JUMP IF WRONG.
7972	042076	005305				DEC	R5	:CORRECT R5 BEFORE NEXT COMPARE.
7973	042100	123725	001240			CMPB	\$TMP3,(R5)+	:COMPARE CALCULATED WITH TABLE DATA.
7974	042104	001402				BEQ	65\$:BRANCH AROUND ERROR JUMP IF OK.
7975	042106	000137	042564		62\$:	JMP	7\$:JUMP TO ERROR REPORT IF INCORRECT.
7976	042112	132737	000001	001273	65\$:	BITB	#1,\$TMP20+1	:TEST TO SEE IF BIT 8 IS SET.
7977	042120	001402				BEQ	66\$:BRANCH AROUND AC SKIP JUMP IF NOT.
7978	042122	000137	042750			JMP	RETURN	:JUMP TO RETURN - AC TESTS ARE TO BE SKIPPED.
7979	042126	112737	000366	001272	66\$:	MOVB	#366,\$TMP20	:MOVE 366, AC LOAD ERROR, TO ERROR POINTER.
7980	042134	010037	043072			MOV	R0,STORE+56	:STORE R0 FOR USE LATER IN THIS ROUTINE.
7981	042140	005037	001236			CLR	\$TMP2	:MOVE A '0' IN 'AC CHANGED' LOCATION.
7982	042144	012737	042210	001244		MOV	#101\$,\$TMP5	:MOVE RETURN TO \$TMP5.
7983	042152	173437	043014			CMPF	STORE,ACO	:SEE IF ACO WAS CHANGED.
7984	042156	170000				CFCC		:COPY FPP CONDITION CODES TO CPU CODES.

7985	042160	001413				BEQ	101\$;BRANCH TO NEXT TEST IF OK.
7986	042162	174037	043062			STF	ACO,STORE+46		;STORE ACTUAL ACO FOR ERROR PRINTING.
7987	042166	012700	043014			MOV	#STORE,RO		;MOVE ADDRESS OF EXPECTED ACO TO RO.
7988						;THE NEXT TWO INSTRUCTIONS TRY TO RESTORE THE ACCUMULATOR AND CHECK THE ACCUMULATOR			
7989						;TO MAKE SURE IT WAS RESTORED PROPERLY FOR THE NEXT RUN THROUGH THIS TRAP HANDLER.			
7990						;IT IS *IMPORTANT* TO REALIZE THAT IF THE 'CMPF' FINDS A DIFFERENCE, THAT THE			
7991						;*FLOATING*POINT*STATUS* IS BEING CHANGED MISTAKENLY. AN ERROR IN THE MICROCODE			
7992						;HAS BEEN FOUND TO CAUSE THIS, SO CHECK THE REVISION OF THE ROM/PROM SET IN THE			
7993						;FPP YOU HAVE. IF YOU DO HAVE WHAT *SEEMS* TO BE THE LATEST REV, A NEW REV WILL			
7994						;BE COMING OUT TO CORRECT THIS PROBLEM. THIS SAME 'LDF/CMPF' SET OF RESTORE/			
7995						;CHECK INSTRUCTIONS IS ACCOMPLISHED FOR EACH ACCUMULATOR CHECK. IT IS ALSO			
7996						;IMPORTANT TO NOTE THAT IF AN ACCUMULATOR FAILS TO RESTORE PROPERLY, SUBSEQUENT			
7997						;PASSES THROUGH THE TRAP HANDLER WILL SKIP THE ACCUMULATOR CHECKS DUE TO THE			
7998						;BIT TEST #400 ABOVE. FOR EXAMPLE, IF ACO FAILS TO LOAD PROPERLY, AC1 THROUGH			
7999						;AC3 WILL STILL BE CHECKED. AS SOON AS ANOTHER FPP INSTRUCTION TRAPS IN THE			
8000						;MAIN TEST, ALL *FURTHER* ACO-AC3 CHECKS WILL BE SKIPPED.			
8001	042172	172437	043014			LDF	STORE,ACO		;RESTORE ACO.
8002	042176	173437	043014			CMPF	STORE,ACO		;SEE IF IT WAS RESTORED PROPERLY.
8003	042202	170000				CFCC			;COPY FPP CONDITION CODES TO CPU CODES.
8004	042204	001567				BEQ	7\$;BRANCH TO ERROR CALL IF OK.
8005	042206	000476				BR	113\$;BRANCH TO ERROR SETUP ROUTINE.
8006	042210	012737	000001	001236	101\$:	MOV	#1,\$TMP2		;PUT A '1' IN 'AC CHANGED' LOCATION.
8007	042216	012737	042262	001244		MOV	#102\$,\$TMP5		;MOVE RETURN TO \$TMP5.
8008	042224	173537	043024			CMPF	STORE+10,AC1		;SEE IF AC1 WAS CHANGED.
8009	042230	170000				CFCC			;COPY FPP CONDITION CODES TO CPU CODES.
8010	042232	001413				BEQ	102\$;BRANCH TO NEXT TEST IF OK.
8011	042234	174137	043062			STF	AC1,STORE+46		;STORE ACTUAL AC1 FOR ERROR PRINTING.
8012	042240	012700	043024			MOV	#STORE+10,RO		;MOVE ADDRESS OF EXPECTED AC1 TO RO.
8013	042244	172537	043024			LDF	STORE+10,AC1		;RESTORE AC1.
8014	042250	173537	043024			CMPF	STORE+10,AC1		;SEE IF IT WAS RESTORED PROPERLY.
8015	042254	170000				CFCC			;COPY FPP CONDITION CODES TO CPU CODES.
8016	042256	001542				BEQ	7\$;BRANCH TO ERROR CALL IF OK.
8017	042260	000451				BR	113\$;BRANCH TO ERROR SETUP ROUTINE.
8018	042262	012737	000002	001236	102\$:	MOV	#2,\$TMP2		;PUT A '2' IN 'AC CHANGED' LOCATION.
8019	042270	012737	042334	001244		MOV	#103\$,\$TMP5		;MOVE RETURN TO \$TMP5.
8020	042276	173637	043034			CMPF	STORE+20,AC2		;SEE IF AC2 WAS CHANGED.
8021	042302	170000				CFCC			;COPY FPP CONDITION CODES TO CPU CODES.
8022	042304	001413				BEQ	103\$;BRANCH TO NEXT TEST IF OK.
8023	042306	174237	043062			STF	AC2,STORE+46		;STORE ACTUAL AC2 FOR ERROR PRINTING.
8024	042312	012700	043034			MOV	#STORE+20,RO		;MOVE ADDRESS OF EXPECTED AC2 TO RO.
8025	042316	172637	043034			LDF	STORE+20,AC2		;RESTORE AC2.
8026	042322	173637	043034			CMPF	STORE+20,AC2		;SEE IF IT WAS RESTORED PROPERLY.
8027	042326	170000				CFCC			;COPY FPP CONDITION CODES TO CPU CODES.
8028	042330	001515				BEQ	7\$;BRANCH TO ERROR CALL IF OK.
8029	042332	000424				BR	113\$;BRANCH TO ERROR SETUP ROUTINE.
8030	042334	012737	000003	001236	103\$:	MOV	#3,\$TMP2		;PUT A '3' IN 'AC CHANGED' LOCATION.
8031	042342	012737	042414	001244		MOV	#100\$,\$TMP5		;MOVE RETURN TO \$TMP5.
8032	042350	173737	043044			CMPF	STORE+30,AC3		;SEE IF AC3 WAS CHANGED.
8033	042354	170000				CFCC			;COPY FPP CONDITION CODES TO CPU CODES.
8034	042356	001416				BEQ	100\$;BRANCH TO NEXT TEST IF OK.
8035	042360	174337	043062			STF	AC3,STORE+46		;STORE ACTUAL AC3 FOR ERROR PRINTING.
8036	042364	012700	043044			MOV	#STORE+30,RO		;MOVE ADDRESS OF EXPECTED AC3 TO RO.
8037	042370	172737	043044			LDF	STORE+30,AC3		;RESTORE AC3.
8038	042374	173737	043044			CMPF	STORE+30,AC3		;SEE IF IT WAS RESTORED PROPERLY.
8039	042400	170000				CFCC			;COPY FPP CONDITION CODES TO CPU CODES.
8040	042402	001470				BEQ	7\$;BRANCH TO ERROR CALL IF OK.
8041	042404	012737	000770	001272	113\$:	MOV	#770,\$TMP20		;MOVE 370 FOR AC LOAD FAILURE, & SET BIT 8 OF ERROR POINTER.

8042	042412	000464				BR	7\$;BRANCH TO ERROR CALL.
8043	042414	005037	001236		100\$:	CLR	\$TMP2		;CLEAR 'REGISTER CHANGED' LOCATION.
8044	042420	112737	000367	001272		MOVB	#367,\$TMP20		;MOVE 367, GENERAL REGISTER CHANGED ERROR, TO POINTER.
8045	042426	012737	042462	001244		MOV	#120,\$TMP5		;MOVE RETURN TO \$TMP5.
8046	042434	023700	043054			CMP	STORE+40,R0		;SEE IF R0 WAS CHANGED.
8047	042440	001410				BEQ	120\$;BRANCH TO NEXT TEST IF OK.
8048	042442	010037	001246			MOV	R0,\$TMP6		;MOVE ACTUAL R0 TO LOCATION FOR ERROR PRINTING.
8049	042446	013737	043054	001240		MOV	STORE+40,\$TMP3		;MOVE EXPECTED TO LOCATION FOR ERROR PRINTING.
8050	042454	013700	043054			MOV	STORE+40,R0		;RESTORE R0.
8051	042460	000441				BR	7\$;BRANCH TO ERROR CALL.
8052	042462	012737	000002	001236	120\$:	MOV	#2,\$TMP2		;PUT A '2' IN 'REGISTER CHANGED' LOCATION.
8053	042470	012737	042524	001244		MOV	#130,\$TMP5		;MOVE RETURN TO \$TMP5.
8054	042476	023702	043056			CMP	STORE+42,R2		;SEE IF R2 WAS CHANGED.
8055	042502	001410				BEQ	130\$;BRANCH TO NEXT TEST IF OK.
8056	042504	010237	001246			MOV	R2,\$TMP6		;MOVE ACTUAL R2 TO LOCATION FOR ERROR PRINTING.
8057	042510	013737	043056	001240		MOV	STORE+42,\$TMP3		;MOVE EXPECTED TO LOCATION FOR ERROR PRINTING.
8058	042516	013702	043056			MCV	STORE+42,R2		;RESTORE R2.
8059	042522	000420				BR	7\$;BRANCH TO ERROR CALL.
8060	042524	012737	000003	001260	130\$:	MOV	#3,\$TMP13		;PUT A '3' IN 'REGISTER CHANGED' LOCATION.
8061	042532	012737	042750	001244		MOV	#RETURN,\$TMP5		;MOVE RETURN TO \$TMP5.
8062	042540	023703	043060			CMP	STORE+44,R3		;SEE IF R3 WAS CHANGED.
8063	042544	001501				BEQ	RETURN		;BRANCH TO RETURN IF OK.
8064	042546	010337	001246			MOV	R3,\$TMP6		;MOVE ACTUAL R3 TO LOCATION FOR ERROR PRINTING.
8065	042552	013737	043060	001240		MOV	STORE+44,\$TMP3		;MOVE EXPECTED TO LOCATION FOR ERROR PRINTING.
8066	042560	013703	043060			MOV	STORE+44,R3		;RESTORE R3.
8067	042564	116537	177777	043156	7\$:	MOVB	-1(R5),EXPTD		;MOVE DATA FROM TABLE FOR POSSIBLE USE ;DPM001
8068	042572	122737	000370	001272		CMPB	#370,\$TMP20		;TEST TO SEE IF AC INFO NEEDS TO BE STORED.
8069	042600	001404				BEQ	71\$;BRANCH TO INFO STORE ROUTINE IF SO.
8070	042602	122737	000366	001272		CMPB	#366,\$TMP20		;TEST TO SEE IF AC INFO NEEDS TO BE STORED.
8071	042610	001026				BNE	MULTER		;SKIP AC INFO ROUTINE IF NOT.
8072	042612	012037	001240		7\$:	MOV	(R0)+,\$TMP3		;MOVE 1ST WORD OF ACTUAL AC DATA TO \$TMP3.
8073	042616	012037	001242			MOV	(R0)+,\$TMP4		;MOVE 2ND WORD OF ACTUAL AC DATA TO \$TMP4.
8074	042622	012037	001246			MOV	(R0)+,\$TMP6		;MOVE 3RD WORD OF ACTUAL AC DATA TO \$TMP6.
8075	042626	012037	001250			MOV	(R0)+,\$TMP7		;MOVE 4TH WORD OF ACTUAL AC DATA TO \$TMP7.
8076	042632	013700	043072			MOV	STORE+56,R0		;RESTORE R0 TO WHAT IT HAD AT BEGINNING OF TRAP.
8077	042636	013737	043062	001252		MOV	STORE+46,\$TMP10		;MOVE 1ST WORD OF EXPECTED AC DATA TO \$TMP10.
8078	042644	013737	043064	001254		MOV	STORE+50,\$TMP11		;MOVE 2ND WORD OF EXPECTED AC DATA TO \$TMP11.
8079	042652	013737	043066	001256		MCV	STORE+52,\$TMP12		;MOVE 3RD WORD OF EXPECTED AC DATA TO \$TMP12.
8080	042660	013737	043070	001274		MOV	STORE+54,\$TMP21		;MOVE 4TH WORD OF EXPECTED AC DATA TO \$TMP21.
8081	042666	012637	001266		MULTER:	MOV	(SP)+,\$TMP16		;SAVE 1ST CONTENTS OF STACK AND POP IT ONCE.
8082	042672	012637	001270			MOV	(SP)+,\$TMP17		;SAVE 2ND CONTENTS OF STACK AND POP IT AGAIN.
8083	042676	142737	000377	042712		BICB	#377,74\$;CLEAR OUT LAST ERROR OFFSET FROM ERROR INSTRUCTION.
8084	042704	153737	001272	042712		BISB	\$TMP20,74\$;PUT ERROR NUMBER TO BE ACCOMPLISHED IN ERROR INSTRUCTION.
8085									;THIS ERROR IS DEFINED BY THE CONTENTS OF THE LOWER BYTE OF LOCATION [1272]
8086	042712	104000			74\$:	ERROR	+0		
8087	042714	013746	001270			MOV	\$TMP17,-(SP)		;PUSH 2ND CONTENTS BACK ON THE STACK.
8088	042720	013746	001266			MOV	\$TMP16,-(SP)		;PUSH 1ST CONTENTS BACK ON THE STACK.
8089	042724	022737	000364	001272		CMP	#364,\$TMP20		;SEE IF RETURN ROUTINE IS TO BE SKIPPED.
8090	042732	001417				BEQ	RTI		;BRANCH TO RTI IF SO.
8091	042734	022737	000365	001272		CMP	#365,\$TMP20		;SEE IF RETURN ROUTINE IS TO BE SKIPPED.
8092	042742	001413				BEQ	RTI		;BRANCH TO RTI IF SO.
8093	042744	000177	136274			JMP	@\$TMP5		;JUMP TO CONTINUE CHECKING.
8094	042750	022776	000004	000000	RETURN:	CMP	#4,@0(SP)		;SEE IF INSTRUCTION IS THE IOT.
8095	042756	001403				BEQ	9\$;BRANCH IF THE IOT HAS BEEN FOUND.
8096	042760	062716	000002			ADD	#2,(SP)		;CORRECT PC RETURN.
8097	042764	000771				BR	RETURN		;BRANCH BACK FOR ANOTHER TRY.
8098	042766	062716	000002		9\$:	ADD	#2,(SP)		;CORRECT PC RETURN TO POINT AFTER IOT FOUND.

8099 042772 005237 177572
 8100 042776 000002
 8101
 8102 043000
 8103
 8104
 8105
 8106
 8107
 8108
 8109
 8110
 8111
 8112
 8113
 8114
 8115
 8116
 8117
 8118
 8119 043014
 8120
 8121
 8122
 8123 043074 000 000 000
 8124 043124 341 000 361
 8125 043156 000000
 8126 043160 005037 177572
 8127 043164 013737 001262 000250
 8128 043172 013737 040522 000020
 8129 043200
 043200 104412

RTI: INC MMRC ;TURN ON MEMORY MANAGEMENT, AND
 RTI ;RETURN FROM INTERRUPT.
 NODAT: .BLKW 6 ;LOCATION IN NON-RES. D-SPACE USED TO FORCE A TRAP.
 ;THE 'STORE' LOCATION BELOW IS PARTITIONED TO RESERVE *4* WORDS FOR EACH FP
 ;ACCUMULATOR, EVEN THOUGH ONLY 2 ARE REQUIRED FOR STORING A FLOATING NUMBER.
 ;THIS IS BECAUSE *IF* THE FPS IS CHANGED BY A PROBLEM IN THE FPP, SO THAT A
 ;*DOUBLE* IS STORED, *4* WORDS RESERVED WILL GUARANTEE THAT THE NEXT DATA BLOCK
 ;WILL NOT BE DISTURBED. PARTITIONING IS AS FOLLOWS:
 : WORD(S) USE
 :-----
 : 1 - 4 STORE AC0
 : 5 -10 STORE AC1
 : 11-14 STORE AC2
 : 15-20 STORE AC3
 : 21 STORE R0
 : 22 STORE R2
 : 23 STORE R3
 : 24-27 STORE ACTUAL AC
 : 30 STORE ACTUAL R0 SO R0 CAN BE USED IN AC ERROR CALLS
 ;STORAGE LOCATIONS FOR THE FLOATING ACCUMULATORS & DATA.
 STORE: .BLKW 30 ;THE FOLLOWING BYTE TABLE WILL BE USED TO CHECK THE VALUES OF SR1 AND THE CALCULATED
 ;VALUES. SR1 MAY TRACK THE ACTIVE REGISTER PROPERLY, BUT IF THE *VALUE* OF THE
 ;INCREMENT/DECREMENT IS WRONG, AN ERROR STILL EXISTS.
 RYTABL: .BYTE 0,0,0,0,0,0,41,21,41,41,0,0,0,27,0,27,0,27,0,27,0,341,361,341
 .BYTE 341,0,361,361,361,361,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
 EXPCTD: .WORD 0 ;LOCATION TO HOLD DATA FROM BYTE TABLE ;DPM001
 ENDIES: CLR MMRO ;TURN OFF MEMORY MANAGEMENT.
 MOV \$TMP14,MMVECT ;RESTORE MMVECT CONTENTS.
 MOV SAVIOT,IOTRAP ;RESTORE IOTRAP CONTENTS.
 DIDONE: 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?).

8130
 8131 043202
 8132
 8133
 8134
 8135

TST115:
 .SBTTL END OF PASS ROUTINE
 ;*****
 ;*INCREMENT THE PASS NUMBER (\$PASS)
 ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
 ;*IF SW12-1 INHIBIT TRACE TRAP
 ;*IF THERES A MONITOR GO TO IT
 ;*IF THERE ISN'T JUMP TO LOOP
 \$EOP:
 SCOPE
 CLR \$TSTNM ;:ZERO THE TEST NUMBER
 CLR \$TIMES ;:ZERO THE NUMBER OF ITERATIONS
 INC \$PASS ;:INCREMENT THE PASS NUMBER
 BIC #10000,\$PASS ;:DON'T ALLOW A NEG. NUMBER
 DEC (PC)+ ;:LOOP?
 \$EOPCT: .WORD 1
 BGT \$DOAGN ;:YES

043202
 043202 000004
 043204 005037 001102
 043210 005037 001302
 043214 005237 001324
 043220 042737 100000 001324
 043226 005327
 043230 000001
 043232 003077

END OF PASS ROUTINE

043234	012737				MOV	(PC)+,a(PC)+	::RESTORE COUNTER	
043236	000001				\$ENDCT:	.WORD 1		
043240	043230				\$EOPCT			
043242	104401	043250			TYPE	.65\$::TYPE ASCIZ STRING	
043246	000407				BR	64\$::GET OVER THE ASCIZ	
				::65\$:	.ASCIZ	<12><15>/END PASS #/		
043266				64\$:				
043266	013746	001324			MOV	\$PASS,-(SP)	::SAVE \$PASS FOR TYPEOUT	
							::TYPE PASS NUMBER IN OCTAL	
							::GO TYPE--OCTAL ASCII	
043272	104403				TYPOS			
043274	006				.BYTE	6	::TYPE 6 DIGITS	
043275	000				.BYTE	0	::SUPPRESS LEADING ZEROS	
043276	005737	001112			TST	\$ERTTL	::SEE IF ANY ERRORS TO REPORT	:DPM001
043302	001430				BEQ	9000\$::BRANCH IF NOT	:DPM001
043304	104401	043312			TYPE	.67\$::TYPE ASCIZ STRING	
043310	000421				BR	66\$::GET OVER THE ASCIZ	
				::67\$:	.ASCIZ	/ TOTAL ERRORS SINCE LAST REPORT /		
043354				66\$:				
043354	013746	001112			MOV	\$ERTTL,-(SP)	::SAVE \$ERTTL FOR TYPEOUT	
							::TOTAL NUMBER OF ERRORS IN OCTAL	
							::GO TYPE--OCTAL ASCII	
043360	104403				TYPOS			
043362	006				.BYTE	6	::TYPE 6 DIGITS	
043363	000				.BYTE	0	::SUPPRESS LEADING ZEROS	
043364	104401	001313			9000\$:	.SCLRF	::TYPE CARRIAGE RETURN, LINE FEED	
043370	005037	001112			CLR	\$ERTTL	::CLEAR ERROR TOTAL	
043374	013700	000042			\$GET42:	MOV @#42,R0	::GET MONITOR ADDRESS	
043400	001414				BEQ	\$DOAGN	::BRANCH IF NO MONITOR	
043402	005046				CLR	-(SP)	::INSURE THE 'T' BIT IS CLEAR	
043404	012746	043412			MOV	#\$CLR.T,-(SP)	::SETUP FOR AN RTI OR RTT	
043410	000426				BR	\$RTRN	::GO DO AN RTI OR RTT TO LOAD THE PSW	
							::WITH A CLEARED 'T' BIT	
043412					\$CLR.T:			
043412	013700	000042			MOV	@#42,R0	::INSURE R0 CONTAINS THE MONITORS	
043416	001405				BEQ	\$DOAGN	::RETURN ADDRESS	
043420	000005				RESET		::CLEAR THE WORLD	
043422	004710				\$ENDAD:	JSR PC,(R0)	::GO TO MONITOR	
043424	000240				NOP		::SAVE ROOM	
043426	000240				NOP		::FOR	
043430	000240				NOP		::ACT11	
043432					\$DOAGN:			
043432	104400				TRAP		::PUSH OLD PSW AND PC ON STACK	
043434	042716	000020			BIC	#20,(SP)	::CLEAR THE 'T' BIT	
043440	032777	010000	135472		BIT	#BIT12,@SWR	::RUN WITH TRACE TRAP?	
043446	001005				BNE	1\$::BR IF NO	
043450	005137	043474			COM	\$TBIT	::IS IT TIME FOR TRACE TRAP	
043454	100402				BMI	1\$::BR IF NO	
043456	052716	000020			BIS	#20,(SP)	::SET TRACE TRAP	
043462	012746	043470			1\$:	MOV	#\$LOOP,-(SP)	::JUMP TO START OF TEST
043466	000002				\$RTRN:	RTI	::RETURN--THIS IS CHANGED TO	
							::AN 'RTT' IF 'RTT' IS A LEGAL	
							::INSTRUCTION	
043470					\$LOOP:			
043470	000137				JMP	a(PC)+	::RETURN	
043472	006600				\$RTNAD:	.WORD LOOP		
043474	000000				\$TBIT:	.WORD 0	::'T' BIT STATE INDICATOR	
043476	377	377	000		\$ENULL:	.BYTE -1,-1,0	::NULL CHARACTER STRING	
						.EVEN		

8136
8137

.SBTTL SCOPE HANDLER ROUTINE

```

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

```

043502          $SCOPE:
043502 104406          CKSWR          ;;TEST FOR CHANGE IN SOF*-SWR
043504 032777 040000 135426 1$: BIT #BIT14,@SWR          ;;LOOP ON PRESENT TEST?
043512 001134          BNE $OVER          ;;YES IF SW14=1
          ;#####START OF CODE FOR THE XOR TESTER#####
043514 000416 $XTSTR: BR 6$          ;;IF RUNNING ON THE 'XOR' TESTER CHANGE
          ;;THIS INSTRUCTION TO A 'NOP' (NOP 240)
043516 013746 000004          MOV ERRVEC,-(SP)          ;;SAVE THE CONTENTS OF THE ERROR VECTOR
043522 012737 043542 000004          MOV #5$,ERRVEC          ;;SET FOR TIMEOUT
043530 005737 177060          TST 177060          ;;TIME OUT ON XOR?
043534 012637 000004          MOV (SP)+,ERRVEC          ;;RESTORE THE ERROR VECTOR
043540 000503          BR $SVLAD          ;;GO TO THE NEXT TEST
043542 022626 5$: CMP (SP)+,(SP)+          ;;CLEAR THE STACK AFTER A TIME OUT
043544 012637 000004          MOV (SP)+,ERRVEC          ;;RESTORE THE ERROR VECTOR
043550 000443          BR 7$          ;;LOOP ON THE PRESENT TEST
043552          6$;#####END OF CODE FOR THE XOR TESTER#####
043552 032777 000400 135360          BIT #BIT08,@SWR          ;;LOOP ON SPEC. TEST?
043560 001404          BEQ 2$          ;;BR IF NO
043562 127737 135352 001102          CMPB @SWR,$TSTNM          ;;ON THE RIGHT TEST? SWR<7:0>
043570 001505          BEQ $OVER          ;;BR IF YES
043572 013737 177766 044020 2$: MOV 177766,CPSAVE          ;;MOVE CPU ERR REG VALUE TO LOC FOR TST ;DPM001
043600 032737 000000 044020          BIT #BIT00,CPSAVE          ;;SEE IF THE POWER MONITOR BIT IS ON ;DPM001
043606 001411          BEQ 2000$          ;;BRANCH TO CONTINUE ROUTINE IF CLEAR ;DPM001
043610 042737 000001 177766          BIC #BIT00,177766          ;;CLEAR THE BIT FOUND TO BE SET ;DPM001
043616 012737 177777 001320          MOV #-1,$FATAL          ;;MOVE -1 TO $FATAL INDICATING PWR MON ER ;DPM001
043624 104177          ERROR +177          ;;CALL SPECIAL POWER FAIL BIT ERROR CALL ;DPM001
043626 105037 001103          CLRB $ERFLG          ;;CLEAR ERROR FLAG FOR CHECK BELOW ;DPM001
043632 105737 001103 2000$: TSTB $ERFLG          ;;HAS AN ERROR OCCURRED?
043636 001421          BEQ 3$          ;;BR IF NO
043640 123737 001115 001103          CMPB $ERMAX,$ERFLG          ;;MAX. ERRORS FOR THIS TEST OCCURRED?
043646 101015          BHI 7$          ;;BR IF NO
043650 032777 001000 135262          BIT #BIT09,@SWR          ;;LOOP ON ERROR?
043656 001404          BEQ 4$          ;;BR IF NO
043660 013737 001110 001106 7$: MOV $LPERR,$LPADR          ;;SET LOOP ADDRESS TO LAST SCOPE
043666 000446          BR $OVER
043670 105037 001103 4$: CLRB $ERFLG          ;;ZERO THE ERROR FLAG
043674 005037 001302          CLR $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
043700 000415          BR 1$          ;;ESCAPE TO THE NEXT TEST
043702 032777 004000 135230 3$: BIT #BIT11,@SWR          ;;INHIBIT ITERATIONS?
043710 001011          BNE 1$          ;;BR IF YES
043712 005737 001324          TST $PASS          ;;IF FIRST PASS OF PROGRAM
043716 001406          BEQ 1$          ;;INHIBIT ITERATIONS
    
```



```

043720 005237 001104          INC      $ICNT          ;; INCREMENT ITERATION COUNT
043724 023737 001302 001104  CMP      $TIMES,$ICNT  ;; CHECK THE NUMBER OF ITERATIONS MADE
043732 002024          BGE      $OVER        ;; BR IF MORE ITERATION REQUIRED
043734 012737 000001 001104 1$:      MOV      #1,$ICNT     ;; REINITIALIZE THE ITERATION COUNTER
043742 013737 044022 001302          MOV      $MXCNT,$TIMES ;; SET NUMBER OF ITERATIONS TO DO
043750 105237 001102          $SVLAD: INCB     $STSTNM  ;; COUNT TEST NUMBERS
043754 113737 001102 001322          MOV      $STSTNM,$STSTNM ;; SET TEST NUMBER IN APT MAILBOX
043762 011637 001106          MOV      (SP),$LPADR   ;; SAVE SCOPE LOOP ADDRESS
043766 011637 001110          MOV      (SP),$LPERR  ;; SAVE ERROR LOOP ADDRESS
043772 005037 001304          CLR      $ESCAPE     ;; CLEAR THE ESCAPE FROM ERROR ADDRESS
043776 112737 000001 001115          MOV      #1,$ERMAX   ;; ONLY ALLOW ONE(1) ERROR ON NEXT TEST
044004 013777 001102 135130 $OVER:  MOV      $STSTNM,@DISPLAY ;; DISPLAY TEST NUMBER
044012 013716 001106          MOV      $LPADR,(SP)  ;; FUDGE RETURN ADDRESS
044016 000002          RTI              ;; FIXES PS
044020 000000          CPSAVE: .WORD     0   ;; LOCATION TO SAVE CPU ERR REG CONTENTS ;DPM001
044022 000001          $MXCNT: 1          ;; MAX. NUMBER OF ITERATIONS

```

8138
8139

.SBTTL ERROR HANDLER ROUTINE

```

*****
*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
*AND GO TO ERTYPE ON ERROR
*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
*SW15=1      HALT ON ERROR
*SW13=1      INHIBIT ERROR TYPEOUTS
*SW10=1      BELL ON ERROR
*SW09=1      LOOP ON ERROR
*CALL
*          ERROR  N          ;; ERROR=EMT AND N=ERROR ITEM NUMBER

```

```

044024 000000          IBSAVE: .WORD     0   ;; LOC'N TO HOLD $ERRPC DURING DUAL ERR ;DPM001
044026          $ERROR:
044026 104406          7$:      CKSWR          ;; TEST FOR CHANGE IN SOFT-SWR
044030 105237 001103          INCB     $ERFLG      ;; SET THE ERROR FLAG
044034 001775          BEQ      7$          ;; DON'T LET THE FLAG GO TO ZERO
044036 013777 001102 135076          MOV      $STSTNM,@DISPLAY ;; DISPLAY TEST NUMBER AND ERROR FLAG
044044 032777 002000 135066          BIT      #BIT10,@SWR  ;; BELL ON ERROR?
044052 001402          BEQ      1$          ;; NO - SKIP
044054 104401 001306          TYPE     $BELL     ;; RING BELL
044060 005237 001112          1$:      INC      $ERTTL  ;; COUNT THE NUMBER OF ERRORS
044064 011637 001116          MOV      (SP),$ERRPC  ;; GET ADDRESS OF ERROR INSTRUCTION
044070 162737 000002 001116          SUB      #2,$ERRPC
044076 117737 135014 001114          MOV      @ $ERRPC,$ITEMB ;; STRIP AND SAVE THE ERROR ITEM CODE
044104 122737 000177 001114          CMPB    #177,$ITEMB  ;; SEE IF PWR MON ERROR CALL ;DPM001
044112 001421          BEQ      1000$     ;; BRANCH IF SO TO CALL THE ERROR ;DPM001
044114 013737 177766 044020          MOV      177766,CPSAVE ;; MOVE CPU ERR REG TO CPSAVE FOR TEST ;DPM001
044122 032737 000001 044020          BIT      #BIT00,CPSAVE ;; SEE IF POWER MONITOR BIT IS SET ;DPM001
044130 001412          BEQ      1000$     ;; BRANCH IF OK ;DPM001
044132 042737 000001 177766          BIC      #BIT00,177766 ;; CLEAR THE BIT FOUND SET ;DPM001
044140 013737 001116 044024          MOV      $ERRPC,IBSAVE ;; SAVE $ERROR ;DPM001
044146 104177          ERROR  +177      ;; CALL POWER MONITOR BIT ERROR ;DPM001
044150 013737 044024 001116          MOV      IBSAVE,$ERRPC ;; RESTORE $ERROR ;DPM001
044156          1000$:
044156 032777 020000 134754          BIT      #BIT13,@SWR  ;; SKIP TYPEOUT IF SET
044164 001004          BNE     20$          ;; SKIP TYPEOUTS
044166 004737 046460          JSR     PC,ERTYPE   ;; GO TO USER ERROR ROUTINE

```

```

044172 104401 001313          TYPE      , $CRLF
044176          20$:
044176 122737 000001 001336  CMPB     #APTENV,$ENV      ;;RUNNING IN APT MODE
044204 001023          BNE      2$              ;;NO,SKIP APT ERROR REPORT
044206 005037 044250          CLR      21$             ;;CLEAR ANY PREVIOUS NUMBER FROM 21$      ;DPM001
044212 113737 001114 044250  MOVB     $ITEMB,21$       ;;SET ITEM NUMBER AS ERROR NUMBER
044220 122737 000377 001114  CMPB     #377,$ITEMB     ;;SEE IF ITEM # IS OVER 400              ;DPM001
044226 001006          BNE      900$           ;;BRANCH IF NOT                          ;DPM001
044230 012737 000400 044250  MOV      #400,21$        ;;MOVE BASE OF 400 TO 21$                ;DPM001
044236 067737 134654 044250  ADD      @SERRPC,21$     ;;ADD NUMBER OVER 400 TO 21$            ;DPM001
044244 004737 045312          900$:   JSR      PC,$ATY4       ;;REPORT FATAL ERRCR TO APT
044250          21$:   .BYTE    0
044251          .BYTE    0
044252 000777          22$:   BR      22$              ;;APT ERROR LOOP
044254 005737 044024          2$:   TST     IBSAVE         ;;SEE IF POWER FAIL ERROR CALL          ;DPM001
044260 001005          BNE      3$              ;;BRANCH IF NOT - HALT NOT ALLOWED     ;DPM001
044262 005777 134652          TST     @SWR            ;;HALT ON ERROR
044266 100002          BPL      3$              ;;SKIP IF CONTINUE
044270 000000          HALT
044272 104406          CKSWR
044274 032777 001000 134636  3$:   BIT     #BIT09,@SWR   ;;TEST FOR CHANGE IN SOFT-SWR
044302 001405          BEQ     4$              ;;LOOP ON ERROR SWITCH SET?
044304 005737 044024          TST     IBSAVE         ;;BR IF NO
044310 001002          BNE      4$              ;;SEE IF ERRGR IS PWR MONITOR BIT ERROR ;DPM001
044312 013716 001110          MOV     $LPERR,(SP)     ;;BRANCH IF SO - DON'T FUDGE RETURN     ;DPM001
044316 005737 001304          4$:   TST     $ESCAPE       ;;FUDGE RETURN FOR LOOPING
044322 001405          BEQ     5$              ;;CHECK FOR AN ESCAPE ADDRESS
044324 005737 044024          TST     IBSAVE         ;;BR IF NONE
044330 001002          BNE      5$              ;;SEE IF ERROR IS PWR MONITOR BIT ERROR ;DPM001
044332 013716 001304          MOV     $ESCAPE,(SP)  ;;BRANCH IF SO - DON'T FUDGE RETURN     ;DPM001
044336          5$:   CMP     #SENDAD,42    ;;FUDGE RETURN ADDRESS FOR ESCAPE
044336 022737 043422 000042  CMP      #SENDAD,42     ;;ACT-11 AUTO-ACCEPT?
044344 001001          BNE      6$              ;;BRANCH IF NO
044346 000000          HALT
044350          6$:   BIT     #BIT09,@SWR   ;;YES
044350 032777 001000 134562  BIT      #BIT09,@SWR
044356 001013          BNE      ERM10
044360 011637 001162          MOV     (SP),$REG0     ;;SEE IF ERROR #377
044364 062737 177776 001162  ADD      #-2,$REG0
044372 122777 000377 134562  CMPB     #377,@$REG0
044400 001002          BNE      ERM10
044402 062716 000002          ADD      #2,(SP)
044406 000002          ERM10: RTI
  
```

8140
8141

```

.SBTTL SAVE AND RESTORE R0-R5 ROUTINES
*****
;SAVE R0-R5
;CALL:
;* SAVREG
;UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
;*
;*TOP---(+16)
;* +2---(+18)
;* +4---R5
;* +6---R4
;* +8---R3
;+10---R2
  
```

```

    044410
    044410 010046
    044412 010146
    044414 010246
    044416 010346
    044420 010446
    044422 010546
    044424 016646 000022
    044430 016646 000022
    044434 016646 000022
    044440 016646 000022
    044444 000002
    
```

```

; *+12---R1
; *+14---R0
$SAVREG:
MOV R0,-(SP) ;;PUSH R0 ON STACK
MOV R1,-(SP) ;;PUSH R1 ON STACK
MOV R2,-(SP) ;;PUSH R2 ON STACK
MOV R3,-(SP) ;;PUSH R3 ON STACK
MOV R4,-(SP) ;;PUSH R4 ON STACK
MOV R5,-(SP) ;;PUSH R5 ON STACK
MOV 22(SP),-(SP) ;;SAVE PS OF MAIN FLOW
MOV 22(SP),-(SP) ;;SAVE PC OF MAIN FLOW
MOV 22(SP),-(SP) ;;SAVE PS OF CALL
MOV 22(SP),-(SP) ;;SAVE PC OF CALL
RTI
; *RESTORE R0-R5
; *CALL:
; * RESREG
$RESREG:
MOV (SP)+,22(SP) ;;RESTORE PC OF CALL
MOV (SP)+,22(SP) ;;RESTORE PS OF CALL
MOV (SP)+,22(SP) ;;RESTORE PC OF MAIN FLOW
MOV (SP)+,22(SP) ;;RESTORE PS OF MAIN FLOW
MOV (SP)+,R5 ;;POP STACK INTO R5
MOV (SP)+,R4 ;;POP STACK INTO R4
MOV (SP)+,R3 ;;POP STACK INTO R3
MOV (SP)+,R2 ;;POP STACK INTO R2
MOV (SP)+,R1 ;;POP STACK INTO R1
MOV (SP)+,R0 ;;POP STACK INTO R0
RTI
    
```

```

044446
044446 012666 000022
044452 012666 000022
044456 012666 000022
044462 012666 000022
044466 012605
044470 012604
044472 012603
044474 012602
044476 012601
044500 012600
044502 000002
    
```

```

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

8142
 8143

.SBTTL TYPE ROUTINE

```

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

```

044504 000000
044506 105737 001157
044512 100002
044514 000000
044516 000432
044520 010046
044522 017600 000002
044526 122737 000001 001336
044534 001011
044536 132737 000100 001337
044544 001405
    
```

```

EOASCII: .WORD 0 ;;LOC TO HOLD ADRS OF TERMINATOR BYTE ;DPM001
$TYPE: TSTB $TPFLG ;;IS THERE A TERMINAL?
BPL 1$ ;;BR IF YES
HALT ;;HALT HERE IF NO TERMINAL
BR 3$ ;;LEAVE
1$: MOV R0,-(SP) ;;SAVE R0
MOV @2(SP),R0 ;;GET ADDRESS OF ASCIZ STRING
CMPB #APTENV,$ENV ;;RUNNING IN APT MODE
BNE 62$ ;;NO,GO CHECK FOR APT CONSOLE
BITB #APTPOOL,$ENVM ;;SPOOL MESSAGE TO APT
BEQ 62$ ;;NO,GO CHECK FOR CONSOLE
    
```

```

044546 010037 044556      MOV      R0,61$      ;;SETUP MESSAGE ADDRESS FOR APT
044552 004737 045302      JSR      PC,$ATY3   ;;SPOOL MESSAGE TO APT
044556 000000                .WORD      0        ;;MESSAGE ADDRESS
044560 132737 000040 001337 61$:  BITB     #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
044566 001005                BNE      60$        ;;YES,SKIP TYPE OUT
044570 112046                MOVB     (R0)+,-(SP)  ;;PUSH CHARACTER TO BE TYPED ONTO STACK
044572 001007                BNE      4$         ;;BR I` IT ISN'T THE TERMINATOR
044574 005726                TST     (SP)+       ;;IF TERMINATOR POP IT OFF THE STACK
044576 010037 044504      MOV      R0,EOASCI  ;;SAVE ADRS OF TERMINATOR - POSSIBLY USED;DPM001
044602 012600                MOV      (SP)+,R0   ;;RESTORE R0
044604 062716 000002      3$:     ADD     #2,(SP)  ;;ADJUST RETURN PC
044610 000002                RTI                     ;;RETURN
044612 122716 000011      4$:     CMPB     #HT,(SP)  ;;BRANCH IF <HT>
044616 001430                BEQ      8$         ;;BRANCH IF NOT <CRLF>
044620 122716 000200      CMPB     #CRLF,(SP)
044624 001006                BNE      5$         ;;POP <CR><LF> EQUIV
044626 005726                TST     (SP)+       ;;TYPE A CR AND LF
044630 104401                TYPE
044632 001313                $CRLF
044634 105037 045042      CLRB     $CHARCNT   ;;CLEAR CHARACTER COUNT
044640 000753                BR      2$         ;;GET NEXT CHARACTER
044642 004737 044724      5$:     JSR      PC,$TYPEC  ;;GO TYPE THIS CHARACTER
044646 123726 001156      6$:     CMPB     $FILLC,(SP)+  ;;IS IT TIME FOR FILLER CHARS.?
044652 001346                BNE      2$         ;;IF NO GO GET NEXT CHAR.
044654 013746 001154      MOV      $NULL,-(SP)  ;;GET # OF FILLER CHARS. NEEDED
;;AND THE NULL CHAR.
044660 105366 000001      7$:     DECB     1(SP)   ;;DOES A NULL NEED TO BE TYPED?
044664 002770                BLT     6$         ;;BR IF NO--GO POP THE NULL OFF OF STACK
044666 004737 044724      JSR      PC,$TYPEC  ;;GO TYPE A NULL
044672 105337 045042      DECB     $CHARCNT   ;;DO NOT COUNT AS A COUNT
044676 000770                BR      7$         ;;LOOP
  
```

;HORIZONTAL TAB PROCESSOR

```

044700 112716 000040      8$:     MOVB     #' ,(SP)  ;;REPLACE TAB WITH SPACE
044704 004737 044724      9$:     JSR      PC,$TYPEC  ;;TYPE A SPACE
044710 132737 000007 045042  BITB     #7,$CHARCNT  ;;BRANCH IF NOT AT
044716 001372                BNE      9$         ;;TAB STOP
044720 005726                TST     (SP)+       ;;POP SPACE OFF STACK
044722 000722                BR      2$         ;;GET NEXT CHARACTER
044724                $TYPEC:
044724 105777 134214      TSTB     @TKS        ;;CHAR IN KYBD BUFFER?      :MJD001
044730 100022                BPL     10$         ;;BR IF NOT      :MJD001
044732 017746 134210      MOV      @TKB,-(SP)  ;;GET CHAR      :MJD001
044736 042716 177600      BIC     #177600,(SP)  ;;STRIP EXTRANEIOUS BITS  :MJD001
044742 122716 000023      CMPB     #$XOFF,(SP)  ;;WAS CHAR XOFF      :MJD001
044746 001012                BNE     102$        ;;BR IF NOT      :MJD001
044750                101$:
044750 105777 134170      TSTB     @TKS        ;;WAIT FOR CHAR      :MJD001
044754 100375                BPL     101$        ;;BR IF NOT      :MJD001
044756 117716 134164      MOVB     @TKB,(SP)   ;;GET CHAR      :MJD001
044762 042716 177600      BIC     #177600,(SP)  ;;STRIP IT      :MJD001
044766 122716 000021      CMPB     #$XON,(SP)  ;;WAS IT XON?      :MJD001
044772 001366                BNE     101$        ;;BR IF NOT      :MJD001
044774                102$:
044774 005726                TST     (SP)+       ;;FIX STACK      :MJD001
044776                10$:
  
```

TYPE ROUTINE

```

044776 105777 134146          TSTB    @STPS          ;;WAIT UNTIL PRINTER IS READY
045002 100375                BPL     10$
045004 116677 000002 134140    MOVB    2(SP),@STPB    ;;LOAD CHAR TO BE TYPED INTO DATA REG.
045012 122766 000015 000002    CMPB    #CR,2(SP)     ;;IS CHARACTER A CARRIAGE RETURN?
045020 001003                BNE     1$            ;;BRANCH IF NO
045022 105037 045042          CLRB    $CHARCNT     ;;YES--CLEAR CHARACTER COUNT
045026 000406                BR      $TYPEX       ;;EXIT
045030 122766 000012 000002 1$:  CMPB    #LF,2(SP)     ;;IS CHARACTER A LINE FEED?
045036 001402                BEQ     $TYPEX       ;;BRANCH IF YES
045040 105227                INCB    (PC)+        ;;COUNT THE CHARACTER
045042 000000                $CHARCNT: .WORD    0 ;;CHARACTER COUNT STORAGE
045044 000207                $TYPEX: RTS        PC
    
```

8144
8145

```

.SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
*****
*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
*OCTAL (ASCII) NUMBER AND TYPE IT.
*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
*CALL:
*      MOV     NUM,-(SP)      ;;NUMBER TO BE TYPED
*      TYPOS   ;;CALL FOR TYPEOUT
*      .BYTE  N              ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
*      .BYTE  M              ;;M=1 OR 0
*                               ;;1=TYPE LEADING ZEROS
*                               ;;0=SUPPRESS LEADING ZEROS
*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
*$TYPOS OR $TYPOC
*CALL:
*      MOV     NUM,-(SP)      ;;NUMBER TO BE TYPED
*      TYPON   ;;CALL FOR TYPEOUT
*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
*CALL:
*      MOV     NUM,-(SP)      ;;NUMBER TO BE TYPED
*      TYPOC   ;;CALL FOR TYPEOUT
045046 017646 000000          $TYPOS: MOV    @ (SP),-(SP)    ;;PICKUP THE MODE
045052 116637 000001 045271    MOVB    1(SP), $OFILL    ;;LOAD ZERO FILL SWITCH
045060 112637 045273          MOVB    (SP)+, $OMODE+1  ;;NUMBER OF DIGITS TO TYPE
045064 062716 000002          ADD     #2,(SP)        ;;ADJUST RETURN ADDRESS
045070 000406                BR      $TYPON
045072 112737 000001 045271    $TYPOC: MOVB   #1,$OFILL  ;;SET THE ZERO FILL SWITCH
045100 112737 000006 045273    MOVB    #6,$OMODE+1    ;;SET FOR SIX(6) DIGITS
045106 112737 000005 045270    $TYPON: MOVB   #5,$OCNT  ;;SET THE ITERATION COUNT
045114 010346                MOV     R3,-(SP)      ;;SAVE R3
045116 010446                MOV     R4,-(SP)      ;;SAVE R4
045120 010546                MOV     R5,-(SP)      ;;SAVE R5
045122 113704 045273          MOVB    $OMODE+1,R4    ;;GET THE NUMBER OF DIGITS TO TYPE
045126 005404                NEG     R4
045130 062704 000006          ADD     #6,R4         ;;SUBTRACT IT FOR MAX. ALLOWED
045134 110437 045272          MOVB    R4,$OMODE     ;;SAVE IT FOR USE
045140 113704 045271          MOVB    $OFILL,R4     ;;GET THE ZERO FILL SWITCH
045144 016605 000012          MOV     12(SP),R5    ;;PICKUP THE INPUT NUMBER
045150 005003                CLR     R3            ;;CLEAR THE OUTPUT WORD
045152 006105                1$:  ROL     R5         ;;ROTATE MSB INTO 'C'
045154 000404                BR      3$           ;;GO DO MSB
    
```

8146
8147

```

045156 006105      2$:  ROL    R5          ;;FORM THIS DIGIT
045160 006105      ROL    R5
045162 006105      ROL    R5
045164 010503      MOV    R5,R3
045166 006103      3$:  ROL    R3          ;;GET LSB OF THIS DIGIT
045170 105337 045272  DECB  $CMODE        ;;TYPE THIS DIGIT?
045174 100016      BPL    7$          ;;BR IF NO
045176 042703 177770  BIC    #177770,R3  ;;GET RID OF JUNK
045202 001002      BNE    4$          ;;TEST FOR 0
045204 005704      TST   R4          ;;SUPPRESS THIS 0?
045206 001403      BEQ   5$          ;;BR IF YES
045210 005204      4$:  INC   R4          ;;DON'T SUPPRESS ANYMORE 0'S
045212 052703 000060  BIS   #'0,R3      ;;MAKE THIS DIGIT ASCII
045216 052703 000040  5$:  BIS   #' ,R3    ;;MAKE ASCII IF NOT ALREADY
045222 110337 045266  MOVB  R3,8$       ;;SAVE FOR TYPING
045226 104401 045266  TYPE  ,8$        ;;GO TYPE THIS DIGIT
045232 105337 045270  7$:  DECB  $OCNT     ;;COUNT BY 1
045236 003347      BGT   2$          ;;BR IF MORE TO DO
045240 002402      BLT   6$          ;;BR IF DONE
045242 005204      INC   R4          ;;INSURE LAST DIGIT ISN'T A BLANK
045244 000744      BR    2$          ;;GO DO THE LAST DIGIT
045246 012605      6$:  MOV   (SP)+,R5   ;;RESTORE R5
045250 012604      MOV   (SP)+,R4   ;;RESTORE R4
045252 012603      MOV   (SP)+,R3   ;;RESTORE R3
045254 016666 000002 000004  MOV   2(SP),4(SP) ;;SET THE STACK FOR RETURNING
045262 012616      MOV   (SP)+,(SP)
045264 000002      RTI          ;;RETURN
045266      000      8$:  .BYTE  0          ;;STORAGE FOR ASCII DIGIT
045267      000      .BYTE  0          ;;TERMINATOR FOR TYPE ROUTINE
045270      000      $OCNT: .BYTE  0     ;;OCTAL DIGIT COUNTER
045271      000      $OFILL: .BYTE  0   ;;ZERO FILL SWITCH
045272 000000      $OMODE: .WORD  0    ;;NUMBER OF DIGITS TO TYPE

.SBTTL  APT COMMUNICATIONS ROUTINE
*****
045274 112737 000001 045540  $ATY1: MOVB  #1,$FFLG  ;;TO REPORT FATAL ERROR
045302 112737 000001 045536  $ATY3: MOVB  #1,$MFLG  ;;TO TYPE A MESSAGE
045310 000403      BR    $ATYC
045312 112737 000001 045540  $ATY4: MOVB  #1,$FFLG  ;;TO ONLY REPORT FATAL ERROR
045320      $ATYC:
045320 010046      MOV   R0,-(SP)    ;;PUSH R0 ON STACK
045322 010146      MOV   R1,-(SP)    ;;PUSH R1 ON STACK
045324 105737 045536      TSTB  $MFLG      ;;SHOULD TYPE A MESSAGE?
045330 001450      BEQ   5$          ;;IF NOT: BR
045332 122737 000001 001336  CMPB  #APTENV,$ENV ;;OPERATING UNDER APT?
045340 001031      BNE   3$          ;;IF NOT: BR
045342 132737 000100 001337  BITB  #APTPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
045350 001425      BEQ   3$          ;;IF NOT: BR
045352 017600 000004      MOV   @4(SP),R0   ;;GET MESSAGE ADDR.
045356 062766 000002 000004  ADD   #2,4(SP)     ;;BUMP RETURN ADDR.
045364 005737 001316      1$:  TST   $MSGTYPE   ;;SEE IF DONE W/ LAST XMISSION?
045370 001375      BNE   1$          ;;IF NOT: WAIT
045372 010037 001332      MOV   R0,$MSGAD   ;;PUT ADDR IN MAILBOX
045376 105720      2$:  TSTB  (R0)+     ;;FIND END OF MESSAGE
045400 001376      BNE   2$
045402 163700 001332      SUB   $MSGAD,R0   ;;SUB START OF MESSAGE
045406 006200      ASR   R0          ;;GET MESSAGE LNTH IN WORDS
    
```

```

045410 010037 001334      MOV      R0,$MSG LGT      ;;PUT LENGTH IN MAILBOX
045414 012737 000004 001316  MOV      #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
045422 000413          BR          5$
045424 017637 000004 045450 3$:  MOV      @4(SP),4$      ;;PUT MSG ADDR IN JSR LINKAGE
045432 062766 000002 000004  ADD      #2,4(SP)      ;;BUMP RETURN ADDRESS
045440 013746 177776      MOV      177776,-(SP)   ;;PUSH 177776 ON STACK
045444 004737 044506      JSR      PC,$TYPE      ;;CALL TYPE MACRO
045450 000000          .WORD    0
045452          5$:
045452 105737 045540      10$:  TSTB     $FFLG          ;;SHOULD REPORT FATAL ERROR?
045456 001416          BEQ      12$          ;;IF NOT: BR
045460 005737 001336      TST      $ENV          ;;RUNNING UNDER APT?
045464 001413          BEQ      12$          ;;IF NOT: BR
045466 005737 001316      11$:  TST      $MSGTYPE     ;;FINISHED LAST MESSAGE?
045472 001375          BNE      11$          ;;IF NOT: WAIT
045474 017637 000004 001320  MOV      @4(SP),$FATAL  ;;GET ERROR #
045502 062766 000002 000004  ADD      #2,4(SP)      ;;BUMP RETURN ADDR.
045510 005237 001316      INC      $MSGTYPE     ;;TELL APT TO TAKE ERROR
045514 105037 045540      12$:  CLRB     $FFLG          ;;CLEAR FATAL FLAG
045520 105037 045537      CLRB     $LFLG        ;;CLEAR LOG FLAG
045524 105037 045536      CLRB     $MFLG        ;;CLEAR MESSAGE FLAG
045530 012601          MOV      (SP)+,R1      ;;POP STACK INTO R1
045532 012600          MOV      (SP)+,R0      ;;POP STACK INTO R0
045534 000207          RTS      PC          ;;RETURN
045536      000          $MFLG: .BYTE    0      ;;MESSG. FLAG
045537      000          $LFLG: .BYTE    0      ;;LOG FLAG
045540      000          $FFLG: .BYTE    0      ;;FATAL FLAG
          .EVEN
          000200  APTSIZE=200
          000001  APTENV=001
          000100  APTSPool=100
          000040  APTCSUP=040
  
```

8148
8149

```

.SBTTL TTY INPUT ROUTINE
;*****
.ENABL LSB
;*****
;SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
;ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
;SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
;WHEN OPERATING IN TTY FLAG MODE.
  
```

```

045542 022737 000176 001140  $CKSWR: CMP      #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
045550 001074          BNE      15$          ;;BRANCH IF NO
045552 105777 133366      TSTB     @TKS          ;;CHAR THERE?
045556 100071          BPL      15$          ;;IF NO, DON'T WAIT AROUND
045560 117746 133362      MOV      @TKB,-(SP)    ;;SAVE THE CHAR
045564 042716 177600      BIC      #^C177,(SP)  ;;STRIP-OFF THE ASCII
045570 022726 000007      CMP      #7,(SP)+     ;;IS IT A CONTROL G?
045574 001062          BNE      15$          ;;NO, RETURN TO USER
045576 123727 001134 000001  CMP      $AUTOB,#1     ;;ARE WE RUNNING IN AUTO-MODE?
045604 001456          BEQ      15$          ;;BRANCH IF YES
045606 104401 046161      TYPE     ,$CNTLG      ;;ECHO THE CONTROL-G (^G)
045612 104401 046166      $GTSWR: TYPE     ,$MSWR  ;;TYPE CURRENT CONTENTS
045616 013746 000176      MOV      SWREG,-(SP)  ;;SAVE SWREG FOR TYPEOUT
045622 104402          TYPOC          ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
045624 104401 046177      TYPE     ,$MNEW      ;;PROMPT FOR NEW SWR
045630 005046      19$:  CLR      -(SP)      ;;CLEAR COUNTER
  
```

```

045632 005046          CLR      -(SP)          ;;THE NEW SWR
045634 105777 133304  7$:  TSTB    @STKS          ;;CHAR THERE?
045640 100375          BPL      7$             ;;IF NOT TRY AGAIN
045642 117746 133300  MOVB    @STKB, -(SP)     ;;PICK UP CHAR
045646 042716 177600  BIC    #^C177, (SP)    ;;MAKE IT 7-BIT ASCII
045652 021627 000025  9$:  CMP     (SP), #25     ;;IS IT A CONTROL-U?
045656 001005          BNE     10$            ;;BRANCH IF NOT
045660 104401 046154  TYPE    , $CNTLU      ;;YES, ECHO CONTROL-U (^U)
045664 062706 000006  20$:  ADD     #6, SP      ;;IGNORE PREVIOUS INPUT
045670 000757          BR      19$            ;;LET'S TRY IT AGAIN
045672 021627 000015  10$:  CMP     (SP), #15     ;;IS IT A <CR>?
045676 001022          BNE     16$            ;;BRANCH IF NO
045700 005766 000004  TST     4(SP)          ;;YES, IS IT THE FIRST CHAR?
045704 001403          BEQ     11$            ;;BRANCH IF YES
045706 016677 000002 133224 MOV     2(SP), @SWR     ;;SAVE NEW SWR
045714 062706 000006  11$:  ADD     #6, SP      ;;CLEAR UP STACK
045720 104401 001313  14$:  TYPE    , $CRLF      ;;ECHO <CR> AND <LF>
045724 123727 001135 000001 CMPB    $INTAG, #1     ;;RE-ENABLE TTY KBD INTERRUPTS?
045732 001003          BNE     15$            ;;BRANCH IF NOT
045734 012777 000100 133202 MOV     #100, @STKS    ;;RE-ENABLE TTY KBD INTERRUPTS
045742 000002          RTI                    ;;RETURN
045744 004737 044724  15$:  JSR     PC, $TYPEC     ;;ECHO CHAR
045750 021627 000060  16$:  CMP     (SP), #60     ;;CHAR < 0?
045754 002420          BLT     18$            ;;BRANCH IF YES
045756 021627 000067  CMP     (SP), #67     ;;CHAR > 7?
045762 003015          BGT     18$            ;;BRANCH IF YES
045764 042726 000060  BIC    #60, (SP)+     ;;STRIP-OFF ASCII
045770 005766 000002  TST     2(SP)          ;;IS THIS THE FIRST CHAR
045774 001403          BEQ     17$            ;;BRANCH IF YES
045776 006316          ASL     (SP)          ;;NO, SHIFT PRESENT
046000 006316          ASL     (SP)          ;;CHAR OVER TO MAKE
046002 006316          ASL     (SP)          ;;ROOM FOR NEW ONE.
046004 005266 000002  17$:  INC     2(SP)          ;;KEEP COUNT OF CHAR
046010 056616 177776  BIS    -2(SP), (SP)   ;;SET IN NEW CHAR
046014 000707          BR      7$            ;;GET THE NEXT ONE
046016 104401 001312  18$:  TYPE    , $QUES      ;;TYPE ?<CR><LF>
046022 000720          BR      20$          ;;SIMULATE CONTROL-U

```

```

.DSABL  LSB
*****
;THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
;CALL:
;* RDCHR          ;;INPUT A SINGLE CHARACTER FROM THE TTY
;* RETURN HERE   ;;CHARACTER IS ON THE STACK
;*              ;;WITH PARITY BIT STRIPPED OFF

```

```

046024 011646          $RDCHR: MOV     (SP), -(SP)    ;;PUSH DOWN THE PC
046026 016666 000004 000002 MOV     4(SP), 2(SP)   ;;SAVE THE PS
046034 105777 133104  1$:  TSTB    @STKS          ;;WAIT FOR
046040 100375          BPL     1$             ;;A CHARACTER
046042 117766 133100 000004 MOVB    @STKB, 4(SP)   ;;READ THE TTY
046050 042766 177600 000004 BIC    #^C<177>, 4(SP) ;;GET RID OF JUNK IF ANY
046056 026627 000004 000023 CMP     4(SP), #23     ;;IS IT A CONTROL-S?
046064 001013          BNE     3$            ;;BRANCH IF NO
046066 105777 133052  2$:  TSTB    @STKS          ;;WAIT FOR A CHARACTER
046072 100375          BPL     2$            ;;LOOP UNTIL ITS THERE
046074 117746 133046  MOVB    @STKB, -(SP)   ;;GET CHARACTER
046100 042716 177600  BIC    #^C177, (SP)   ;;MAKE IT 7-BIT ASCII

```


TTY INPUT ROUTINE

046104	022627	000021		
046110	001366			
046112	000750			
046114	026627	000004	000021	3\$:
046122	001744			
046124	026627	000004	000140	
046132	002407			
046134	026627	000004	000175	
046142	003003			
046144	042766	000040	000004	
046152	000002			4\$:
046154	136	125	015	\$CNTLU: .ASCIZ /^U/<15><12>
046161	136	107	015	\$CNTLG: .ASCIZ /^G/<15><12>
046166	015	012	123	\$MSWR: .ASCIZ <15><12>/SWR = /
046177	040	040	116	\$MNEW: .ASCIZ / NEW = /

```

CMP      (SP)+,#21      ;; IS IT A CONTROL-Q?
BNE      2$             ;; IF NOT DISCARD IT
BR       1$             ;; YES, RESUME
CMP      4(SP),#$XON    ;; IS IT A RANDOM XON?
BEQ      1$             ;; BRANCH IF YES
CMP      4(SP),#140     ;; IS IT UPPER CASE?
BLT      4$             ;; BRANCH IF YES
CMP      4(SP),#175     ;; IS IT A SPECIAL CHAR?
BGT      4$             ;; BRANCH IF YES
BIC      #40,4(SP)      ;; MAKE IT UPPER CASE
RTI      ;; GO BACK TO USER
$CNTLU:  .ASCIZ /^U/<15><12> ;; CONTROL 'U'
$CNTLG:  .ASCIZ /^G/<15><12> ;; CONTROL 'G'
$MSWR:   .ASCIZ <15><12>/SWR = /
$MNEW:   .ASCIZ / NEW = /
    
```

:RAN001
:RAN001

8150
8151

```

.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.
    
```

046210	010046			
046212	016600	000002		
046216	005740			
046220	111000			
046222	006300			
046224	016000	046244		
046230	000200			
046232	011646			
046234	016666	000004	000002	
046242	000002			

```

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

;; THIS IS USE TO HANDLE THE 'GETPRI' MACRO
$TRAP2: MOV      (SP),-(SP)    ;; MOVE THE PC DOWN
        MOV      4(SP),2(SP)   ;; MOVE THE PSW DOWN
        RTI      ;; RESTORE THE PSW
    
```

```

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

046244	046232			
046246	044506			
046250	045072			
046252	045046			
046254	045106			
046256	045612			
046260	045542			
046262	046024			
046264	044410			
046266	044446			
8152	046270	047364		
8153	046272	047356		
8154		000030		
8155				
8156				

```

ROUTINE
-----
$TRPAD: .WORD      $TRAP2
        $TYPE      ;; CALL=TYPE      TRAP+1(104401)  TTY TYPEOUT ROUTINE
        $TYPOC     ;; CALL=TYPOC     TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
        $TYPOS     ;; CALL=TYPOS     TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
        $TYPON     ;; CALL=TYPON     TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
        $GTSWR     ;; CALL=GTSWR     TRAP+5(104405)  GET SOFT-SWR SETTING
        $CKSWR     ;; CALL=CKSWR     TRAP+6(104406)  TEST FOR CHANGE IN SOFT-SWR
        $RDCHR     ;; CALL=RDCHR     TRAP+7(104407)  TTY TYPEIN CHARACTER ROUTINE
        $SAVREG    ;; CALL=SAVREG    TRAP+10(104410) SAVE R0-R5 ROUTINE
        $RESREG    ;; CALL=RESREG    TRAP+11(104411) RESTORE R0-R5 ROUTINE
        .RSET      ;; CALL=RSETUP    TRAP+12(104412) ROUTINE TO INITIALIZE AT END OF EACH TEST
        .LPER      ;; CALL=LPER      TRAP+13(104413) ROUTINE TO SET UP LOOP ON ERROR ADDRESS
$TERM-.-$TRPAD
    
```

```

.SBTTL POWER DOWN AND UP ROUTINES
*****
    
```

046274	012737	046452	000024	
046302	012737	000340	000026	

```

:POWER DOWN ROUTINE
$PWRDN: MOV      #SILLUP,@#PWRVEC ;; SET FOR FAST UP
        MOV      #340,@#PWRVEC+2 ;; PRIO:7
    
```

```

046310 010046      MOV      R0,-(SP)      ;;PUSH R0 ON STACK
046312 010146      MOV      R1,-(SP)      ;;PUSH R1 ON STACK
046314 010246      MOV      R2,-(SP)      ;;PUSH R2 ON STACK
046316 010346      MOV      R3,-(SP)      ;;PUSH R3 ON STACK
046320 010446      MOV      R4,-(SP)      ;;PUSH R4 ON STACK
046322 010546      MOV      R5,-(SP)      ;;PUSH R5 ON STACK
046324 017746      MOV      @SWR,-(SP)    ;;PUSH @SWR ON STACK
046330 010637      MOV      SP,$SAVR6    ;;SAVE SP
046334 012737      MOV      #SPWRUP,@#PWRVEC ;;SET UP VECTOR
046342 000000      HALT
046344 000776      BR      -2           ;;HANG UP
*****
:POWER UP ROUTINE
046346 012737      046452 000024 $PWRUP: MOV      #SILLUP,@#PWRVEC ;;SET FOR FAST DOWN
046354 013706      046456      MOV      $SAVR6,SP      ;;GET SP
046360 005037      046456      CLR      $SAVR6        ;;WAIT LOOP FOR THE TTY
046364 005237      046456      1$: INC     $SAVR6      ;;WAIT FOR THE INC
046370 001375      BNE     1$           ;;OF WORD
046372 012677      132542      MOV      (SP)+,@SWR    ;;POP STACK INTO @SWR
046376 012605      MOV      (SP)+,R5     ;;POP STACK INTO R5
046400 012604      MOV      (SP)+,R4     ;;POP STACK INTO R4
046402 012603      MOV      (SP)+,R3     ;;POP STACK INTO R3
046404 012602      MOV      (SP)+,R2     ;;POP STACK INTO R2
046406 012601      MOV      (SP)+,R1     ;;POP STACK INTO R1
046410 012600      MOV      (SP)+,R0     ;;POP STACK INTO R0
046412 012737      046274 000024      MOV      #SPWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
046420 012737      000340 000026      MOV      #340,@#PWRVEC+2 ;;PRIO:7
046426 104401      TYPE     ;;REPORT THE POWER FAILURE
046430 047434      $PWRMG: .WORD  POWERM   ;;POWER FAIL MESSAGE POINTER
046432 012716      MOV      (PC)+,(SP)   ;;RESTART AT START
046434 006116      $PWRAD: .WORD  START   ;;RESTART ADDRESS
046436 042766      000020 000002      BIC      #20,2(SP)    ;;CLEAR 'T' BIT
046444 005037      043474      CLR      $TBIT        ;;CLEAR THE 'T' BIT FLAG
046450 000002      RTI
046452 000000      $ILLUP: HALT        ;;THE POWER UP SEQUENCE WAS STARTED
046454 000776      BR      -2           ;; BEFORE THE POWER DOWN WAS COMPLETE
046456 000000      $SAVR6: 0           ;;PUT THE SP HERE

```

8157
8158
8159
8160

8161
8162
8163
8164
8165
8166

```

8167 046460 104401
8168 046462 001313
8169 046464 113737 001102 001232
8170 046472 042737 177400 001232
8171 046500 013737 001116 001234
8172 046506 010046
8173 046510 117700 132520
8174 046514 042700 177400
8175 046520 001006

```

```

.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
        MOVB  $STNM,$TMP0
        BIC   #177400,$TMP0
        MOV   $ERRPC,$TMP1      ;GET PC OF CALL
        MOV   R0,-(SP)          ;SAVE R0
        MOVB  @TMP1,R0          ;GET THE ITEM NUMBER.
        BIC   #177400,R0
        BNE  21$

```

```

8176 046522 013746 001116      MOV      $ERRPC,-(SP)      ;MOVE ERROR PC TO STACK FOR PRINTING
8177 046526 104402              TYPOC                    ;GO TYPE THE ERROR PC
8178 046530 104401 001313      TYPE      , $CRLF        ;TYPE A <CRLF>
8179 046534 000561              BR       ERT5            ;BRANCH TO EXIT
8180 046536 022700 000377      21$:    CMP      #377,R0
8181 046542 001004              BNE     20$
8182 046544 017600 000004      MOV      @4(SP),R0
8183 046550 062700 000400      ADD     #400,R0
8184 046554 010037 047104      20$:    MOV      R0,ERRNUM      ;SAVE THE ERROR # FOR POSSIBLE USE      ;DPM001
8185 046560 005300              DEC     R0                ;OTHERWISE MAKE R0 AN
8186 046562 006300              ASL     R0                ;INDEX FOR THE TABLE.
8187 046564 006300              ASL     R0
8188 046566 006300              ASL     R0
8189 046570 062700 001442      ADD     #$ERRTB,R0
8190 046574 012037 046604      22$:    MOV      (R0)+,2$
8191 046600 001406              BEQ     3$
8192 046602 104401              TYPE
8193 046604 000000      2$:    .WORD    0
8194 046606 004737 047106      JSR     PC,PASCIZ        ;GO CHECK FOR AND PRINT ANY ADDTL ASCIZ ;DPM001
8195 046612 104401 001313      TYPE      , $CRLF
8196 046616 012037 046626      3$:    MOV      (R0)+,4$
8197 046622 001406              BEQ     5$
8198 046624 104401              TYPE
8199 046626 000000      4$:    .WORD    0
8200 046630 004737 047106      JSR     PC,PASCIZ        ;GO CHECK FOR AND PRINT ANY ADDTL ASCIZ ;DPM001
8201 046634 104401              TYPE
8202 046636 001313      .WORD    $CRLF
8203 046640 010146      5$:    MOV      R1,-(SP)        ;SAVE R1,R2 AND R3
8204 046642 010246      MOV      R2,-(SP)
8205 046644 010346      MOV      R3,-(SP)
8206 046646 012001      MOV      (R0)+,R1        ;GET THE ADDRESS OF THE DATA TABLE.
8207 046650 001506      BEQ     ERT4            ;RETURN IF NO DATA.
8208 046652 011000      MOV      (R0),R0        ;GET A POINTER TO THE DATA FORMAT TABLE.
8209 046654 105710      ERT1:   TSTB     (R0)        ;FORMAT ZERO?
8210 046656 001003      BNE     7$
8211 046660 013146      MOV      @ (R1)+,-(SP)   ;FORMAT ZERO SO TYPE
8212 046662 104402              TYPOC                    ;AN OCTAL NUMBER.
8213 046664 000473      BR       ERT2
8214 046666 122710 000002      7$:    CMPB     #2,(R0)        ;FORMAT TWO?
8215 046672 001010      BNE     9$
8216 046674 013102      MOV      @ (R1)+,R2      ;FORMAT TWO SO TYPE TWO
8217 046676 012246      MOV      (R2)+,-(SP)    ;OCTAL NUMBERS.
8218 046700 104402              TYPOC
8219 046702 104401              TYPE
8220 046704 047500      .WORD    SPACE
8221 046706 011246      MOV      (R2)+,-(SP)
8222 046710 104402              TYPOC
8223 046712 000460      BR       ERT2
8224 046714 122710 000003      9$:    CMPB     #3,(R0)        ;FORMAT THREE?
8225 046720 001011      BNE     10$
8226 046722 013102      MOV      @ (R1)+,R2      ;FORMAT THREE SO TYPE 4 OCTAL NUMBERS.
8227 046724 012703 000004      MOV      #4,R3          ;LOOP COUNTER
8228 046730 012246      90$:   MOV      (R2)+,-(SP)    ;MOVE DATA TO THE STACK
8229 046732 104402              TYPOC                    ;TYPE AN OCTAL NUMBER
8230 046734 104401              TYPE                      ;TYPE A SPACE
8231 046736 047500      .WORD    SPACE
8232 046740 077305      SOB     R3,90$          ;SUBTRACT 1 AND BRANCH IF NOT DONE YET ;DPM001
  
```

```

8233 046742 000444
8234 046744 122710 000004      10$: BR      ERT2      :EXIT
8235 046750 001004      CMPB    #4,(R0)    :FORMAT FOUR?
8236 046752 013146      BNE     11$
8237 046754 104403      MOV     @ (R1)+,-(SP) :FORMAR FOUR SO TYPE
8238 046756      016      TYPOS   :AN OCTAL NUMBER
8239 046757      000      .BYTE  16      :SUPPRESSING LEADING ZEROES.
8240 046760 000435      .BYTE  0
8241 046762 122710 000005      11$: BR      ERT2
8242 046766 001005      CMPB    #5,(R0)    :FORMAT FIVE?
8243 046770 012137 046776      BNE     13$
8244 046774 104401      MOV     (R1)+,12$   :FORMAT FIVE SO TYPE AN
8245 046776 000000      TYPE   :ASCIZ STRING.
8246 047000 000427      * 12$: .WORD  0
8247 047002 122710 000011      13$: BR      ERT3
8248 047006 001005      CMPB    #11,(R0)   :FORMAT ELEVEN?
8249 047010 013137 047016      BNE     15$
8250 047014 104401      MOV     @ (R1)+,14$ :FORMAT ELEVEN SO PICK
8251 047016 000000      TYPE   :A POINTER TO AN ASCIZ
8252 047020 000417      14$: .WORD  0      :STRING.
8253 047022 122710 000012      15$: BR      ERT3
8254 047026 001011      CMPB    #12,(R0)   :FORMAT TWELVE?
8255 047030 013102      BNE     17$
8256 047032 012703 000006      MOV     @ (R1)+,R2  :FORMAT TWELVE SO TYPE
8257 047036 012246      MOV     #6,R3      :TYPE SIX OCTAL NUMBERS
8258 047040 104402      MOV     (R2)+,-(SP)
8259 047042 104401      TYPOC
8260 047044 047500      TYPE
8261 047046 077305      .WORD  SPACE
8262 047050 000401      SOB    R3,16$
8263 047052 000000      BR     ERT2
8264 047054 104401      17$: HALT
8265 047056 047503      ERT2: TYPE
8266 047060 005200      .WORD  $TAB
8267 047062 005711      ERT3: INC    R0
8268 047064 001273      TST   (R1)
8269 047066 104401      BNE   ERT1
8270 047070 001313      ERT4: TYPE
8271 047072 012603      .WORD  $CRLF
8272 047074 012602      MOV   (SP)+,R3
8273 047076 012601      MOV   (SP)+,R2
8274 047100 012600      MOV   (SP)+,R1
8275 047102 000207      ERT5: MOV   (SP)+,R0
8276      RTS   PC
8277 047104 000000      . ERRNUM: .WORD  0
:LOCATION TO HOLD CURRENT ERROR NUMBER ;DPM001

```

8278					.SBTTL	SUBROUTINE TO LOOK FOR ANY ADDITIONAL MSGS TO PRINT	
8279	047106	010446			PASCIZ: MOV	R4,-(SP)	:SAVE R4
8280	047110	013704	044504		1\$: MOV	EOASCII,R4	:MOVE END OF ASCII ADDRESS TO R4
8281	047114	122714	00C377		CMPB	#377,(R4)	:SEE IF ADD'L MSGS NEED TO BE PRINTED
8282	047120	001013			BNE	4\$:BRANCH IF NOT
8283	047122	005204			INC	R4	:INCREMENT TO NEXT BYTE
8284	047124	032704	000001		BIT	#BIT00,R4	:SEE IF ODD ADDRESS
8285	047130	001401			BEQ	2\$:BRANCH IF NOT
8286	047132	005204			INC	R4	:POINT TO WORD ADDRESS
8287	047134	012437	047144		2\$: MOV	(R4)+,3\$:MOVE ADDRESS TO LOCATION FOR TYPING
8288	047140	001763			BEQ	1\$:GO CHK FOR MORE MSG ADRS'S IF NO MORE
8289	047142	104401			TYPE		:TYPE THE MESSAGE, ADDRESS IN NEXT WORD
8290	047144	000000			3\$: .WORD	0	:LOCATION FOR MESSAGE ADDRESS
8291	047146	000772			BR	2\$:BRANCH BACK TO LOAD OTHER MESSAGES
8292	047150	122714	000376		4\$: CMPB	#376,(R4)	:SEE IF SPECIFIC MSGS NEED PRINTING
8293	047154	001027			BNE	11\$:BRANCH TO EXIT IF NOT
8294	047156	005204			INC	R4	:INCREMENT TO NEXT BYTE
8295	047160	032704	000001		BIT	#BIT00,R4	:SEE IF ODD ADDRESS
8296	047164	001401			BEQ	5\$:BRANCH IF NOT
8297	047166	005204			INC	R4	:POINT TO WORD ADDRESS
8298	047170	005046			5\$: CLR	-(SP)	:CLEAR COUNTER LOCATION ON STACK
8299	047172	005216			6\$: INC	(SP)	:INCREMENT COUNTER
8300	047174	023724	047104		CMP	ERRNUM,(R4)+	:SEE IF ITEM NUMBER MATCHES ERROR
8301	047200	001374			BNE	6\$:BRANCH IF NOT TO SEARCH AGAIN
8302	047202	005724			7\$: TST	(R4)+	:SEE IF AT TERMINATOR YET
8303	047204	001376			BNE	7\$:BRANCH BACK UNTIL IT IS
8304	047206	005744			TST	-(R4)	:PUT R4 AT THE TERMINATOR
8305	047210	005724			8\$: TST	(R4)+	:POINT R4 TO NEXT ADDRESS
8306	047212	005316			DEC	(SP)	:SEE IF COUNTER DOWN TO ZERO YET
8307	047214	001375			BNE	8\$:BRANCH BACK IF NOT TO ADVANCE R4
8308	047216	005726			9\$: TST	(SP)+	:POP COUNTER OFF STACK
8309	047220	011437	047230		MOV	(R4),10\$:MOVE ADDRESS TO LOCATION TO TYPE
8310	047224	001403			BEQ	11\$:BRANCH IF NO EXTRA MESSAGE TO PRINT
8311	047226	104401			TYPE		:TYPE THE ASCII MESSAGE
8312	047230	000000			10\$: .WORD	0	:LOCATION FOR ASCII MESSAGE ADDRESS
8313	047232	000726			BR	1\$:GO BACK TO MAKE SURE NO MORE MSGS
8314	047234	012604			11\$: MOV	(SP)+,R4	:RESTORE R4
8315	047236	000207			RTS	PC	:EXIT

```

8316 .SBTTL FPP SPURIOUS TRAP TO 244 HANDLER
8317 :*****
8318 :*****
8319 :*THIS ROUTINE HANDLES UNEXPECTED TRAPS TO THE FPP TRAP VECTOR AT 244.
8320 :*THE LAST FPP INSTRUCTION EXECUTED AND ITS ADDRESS HAS BEEN RECORDED
8321 :*THESE ALONG WITH THE FEC, FPS AND PC OF TRAP ARE REPORTED.
8322 047240 042737 000001 177572 FPSPUR: BIC #BIT00,MMR0 ;MAKE SURE MEMORY MANAGEMENT IS OFF ;DPM001
8323 047246 011637 001236 MOV (SP),$TMP2 ;SAVE PC OF TRAP.
8324 047252 022626 CMP (SP)+,(SP)+ ;RESTORE SP.
8325 047254 170200 STFPS R0 ;GET FPS
8326 047256 010037 001240 MOV R0,$TMP3 ;MOVE IT TO $TMP3
8327 047262 170300 STST R0 ;GET FEC
8328 047264 010037 001242 MOV R0,$TMP4 ;MOVE IT TO $TMP4
8329 047270 104377 ERROR +377 ;CALL ERROR
8330 047272 000041 .WORD 41 ;ERROR #441
8331 047274 104412 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?).
8332 047276 000137 043202 JMP $EOP ;JUMP TO $EOP
  
```

8333
8334

.SBTTL CPU SPURIOUS TRAP TO 4 HANDLER

*THIS ROUTINE REPORTS UNEXPECTED CPU TRAPS TO VECTOR 4.

8335
8336

8337 047302 042737 000001 177572
8338 047310 011637 001236
8339 047314 022626
8340 047316 104377
8341 047320 000042
8342 047322 104412

CPSPUR: BIC #BIT00,MMRO ;MAKE SURE MEMORY MANAGEMENT IS OFF ;DPM001
MOV (SP),\$TMP2 ;SAVE PC OF TRAP.
CMP (SP)+,(SP)+ ;CLEAN STACK
ERROR +377 ;CALL ERROR
.WORD 42 ;ERROR #442
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 \$EOP ;JUMP TO \$EOP

8343 047324 000137 043202

8344
8345

.SBTTL CPU SPURIOUS TRAP TO 10 HANDLER

::*****
::*****

::*THIS ROUTINE REPORTS UNEXPECTED CPU TRAPS TO VECTOR 10.*

8346
8347

8348 047330 042737 000001 177572
8349 047336 011637 001236
8350 047342 022626
8351 047344 104377
8352 047346 000043
8353 047350 104412

CPTWO: BIC #BIT00,MMRO ;MAKE SURE MEMORY MANAGEMENT IS OFF -DPM001
MOV (SP),\$TMP2 ;SAVE PC OF TRAP.
CMP (SP)+,(SP)+ ;CLEAN STACK
1\$: ERROR +377 ;CALL ERROR
.WORD 43 ;ERROR #443
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?).
;JUMP TO \$EOP

8354 047352 000137 043202

JMP \$EOP

8355
8356

.SBTTL SET LOOP ON ERROR ADDRESS ROUTINE

::*****
::*****

8357
8358 047356 011637 001110
8359 047362 000002

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

```

8360                                     .SBTTL FLAG RESET AND CONSOLE TEST ROUTINE
8361                                     :*****
8362                                     :*****
8363                                     :*THIS ROUTINE WILL BE CALLED AT THE END OF EACH TEST TO
8364                                     :*RESET THE STACK, CLEAR THE FPS AND SEE IF THE USER HAS TYPED
8365                                     :* CONTROL G ON THE TERMINAL. IF THE USER HAS TYPED CONTROL G AND
8366                                     :*THERE IS NO PHYSICAL CONSOLE SWITCH REGISTER THEN THE CONTENTS
8367                                     :*OF THE SOFTWARE SWITCH REGISTER WILL BE TYPED IN OCTAL ON THE
8368                                     :*TELETYPE AND THE USER CAN MODIFY IT.
8369 047364 023727 001140 177570 .RSET:  CMP     SWR,#177570      ;SEE IF THERE IS A PHYSICAL CONSOLE SWITCH REGISTER.
8370 047372 001001                BNE     1$              ;BRANCH IF NOT
8371 047374 104406                CKSWR                    ;OTHERWISE TYPE THE CONTENTS OF THE PROGRAM VIRTUAL
8372                                     ;SWITCH REGISTER AND GIVE THE USER CHANCE TO MODIFY IT
8373 047376 012737 047240 000244 1$:  MOV     #FPSPUR,FPVECT
8374 047404 012737 047302 000004      MUV     #CPSPUR,ERRVECT
8375 047412 012737 047330 000010      MOV     #CPTWO,10
8376 047420 011600                MOV     (SP),R0        ;SAVE RETURN ADDRESS.
8377 047422 012706 001100          MOV     #STACK,SP     ;RESET THE STACK POINTER.
8378 047426 005004                CLR     R4             ;CLEAR THE FPS.
8379 047430 170104                LDFPS  R4
8380 047432 000110                JMP     (R0)          ;RETURN.

```

SPECIAL MESSAGES

8381					.SBTTL	SPECIAL MESSAGES
8382	047434	200	120	117	POWERM: .ASCIZ	<CRLF>'POWER FAILURE. PROGRAM RESTARTING.'
8383	047500	040	040	000	SPACE: .ASCIZ	' '
8384	047503	011	000		\$TAB: .ASCIZ	<TAB>
8385	047505	107	117	124	MS1: .ASCIZ	'GOT RESULT:'<TAB><TAB>
8386	047523	105	130	120	MS2: .ASCIZ	'EXPECTED RESULT:'<TAB>
8387	047545	101	103	040	MS3: .ASCIZ	'AC OPERAND:'<TAB><TAB>
8388	047563	123	117	125	MS4: .ASCIZ	'SOURCE OPERAND: '<TAB>
8389	047545				MS10=MS3	
8390	047605	105	130	120	MS11: .ASCIZ	'EXPONENT OPERAND:'<TAB>
8391	047630	114	117	101	MS20: .ASCIZ	'LOADED:'<TAB><TAB>
8392	047642	124	122	111	MS21: .ASCIZ	'TRIED TO LCAD:'<TAB>

8393						.SBTTL	ERROR MESSAGES
8413	047662	123	000		EM1:	.ASCIZ	:S:
8414	047664	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8415	047666	047676	047707	047726		.WORD	EM1B,EM1C,EM1D,0
8416	047676	124	106	040	EM1B:	.ASCIZ	:TF A,AC7:
8417	047707	040	104	111	EM1C:	.ASCIZ	:DID NOT TRAP.:
8418	047726	040	106	111	EM1D:	.ASCIZ	:FID=0.:
8419	047736	123	000		EM2:	.ASCIZ	:S:
8420	047740	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8421	047742	047676	047752	047726		.WORD	EM1B,EM2B,EM1D,0
8422	047752	056	040	106	EM2B:	.ASCIZ	:.FPS BAD.:
8423	047766	123	000		EM3:	.ASCIZ	:S:
8424	047770	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8425	047772	047676	050002	047726		.WORD	EM1B,EM3B,EM1D,0
8426	050002	056	040	106	EM3B:	.ASCIZ	:.FEC BAD.:
8427	050016	123	000		EM4:	.ASCIZ	:S:
8428	050020	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8429	050022	050034	050045	050057		.WORD	EM4A,EM4B,EM4C,EM4D,0
8430	050034	124	106	040	EM4A:	.ASCIZ	:TF A,(R):
8431	050045	056	040	122	EM4B:	.ASCIZ	:.RO BAD.:
8432	050057	056	040	106	EM4C:	.ASCIZ	:.FDST:
8433	050066	040	106	101	EM4D:	.ASCIZ	:FAILED.:
8434	050077	123	000		EM5:	.ASCIZ	:S:
8435	050101	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8436	050102	050034	050066	000000		.WORD	EM4A,EM4D,0
8437	050110	123	000		EM6:	.ASCIZ	:S:
8438	050112	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8439	050114	050034	050057	050066		.WORD	EM4A,EM4C,EM4D,EM6C
8440	050124				EM6C:		
	050124	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
	050126	050167	050164	050176		.WORD	CRBUT,IFD,PRST,N7,N0,N7,WENTTO,N2,N4,N5,INSTOF,N2,N4,N4,0
8441	050164	106	104	000	IFD:	.ASCIZ	:FD:
8442	050167	200	050	102	CRBUT:	.ASCIZ	:<CRLF>'(BUT :
8443	050176	051	040	123	PRST:	.ASCIZ	:) ST :
8444	050204	040	127	105	WENTTO:	.ASCIZ	:WENT TO :
8445	050216	040	111	116	INSTOF:	.ASCIZ	:INSTEAD OF
8446	050233	060	000		N0:	.ASCIZ	:0:
8447	050235	061	000		N1:	.ASCIZ	:1:
8448	050237	062	000		N2:	.ASCIZ	:2:
8449	050241	063	000		N3:	.ASCIZ	:3:
8450	050243	064	000		N4:	.ASCIZ	:4:
8451	050245	065	000		N5:	.ASCIZ	:5:
8452	050247	066	000		N6:	.ASCIZ	:6:
8453	050251	067	000		N7:	.ASCIZ	:7:
8454	050253	123	000		EM7:	.ASCIZ	:S:
8455	050255	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8456	050256	050034	050272	050045		.WORD	EM4A,EM7B,EM4B,EM4C,EM4D,0
8457	050272	053	056	000	EM7B:	.ASCIZ	:+:
8458	050275	123	000		EM10:	.ASCIZ	:S:

ERROR MESSAGES

8459	050277		377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
8460	050300	050034	050272	050066			.EVEN		
8461	050310	123	000		EM11:	.WORD	EM4A,EM7B,EM4D,0		
8462	050312	377				.ASCIZ	'S'		
						.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN			
8463	050314	050326	050045	050057		.WORD	EM11B,EM4B,EM4C,EM4D,0		
8464	050326	124	104	040	EM11B:	.ASCIZ	'TD A,(R)+'		
8465	050340	123	000		EM12:	.ASCIZ	'S'		
8466	050342	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN			
8467	050344	050326	050066	000000		.WORD	EM11B,EM4D,0		
8468	050352	123	000		EM13:	.ASCIZ	'S'		
8469	050354	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN			
8470	050356	050364	050374	000000		.WORD	EM13A,EM13B,0		
8471	050364	124	104	040	EM13A:	.ASCIZ	'TD A,#N'		
8472	050374	040	124	122	EM13B:	.ASCIZ	'TRAP TO 4 IN FDST.:'		
8473		050352			EM14=EM13				
8474	050420	123	000		EM15:	.ASCIZ	'S'		
8475	050422	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN			
8476	050424	050364	050066	000000		.WORD	EM13A,EM4D,0		
8477	050432	120	103	040	EM16:	.ASCIZ	'PC BAD AFTER S'		
8478	050451	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN			
8479	050452	050364	000000			.WORD	EM13A,0		
8480	050456	123	000		EM17:	.ASCIZ	'S'		
8481	050460	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN			
8482	050462	050470	050374	000000		.WORD	EM17B,EM13B,0		
8483	050470	124	104	040	EM17B:	.ASCIZ	'TD A,-(R)'		
8484	050502	123	000		EM20:	.ASCIZ	'S'		
8485	050504	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN			
8486	050506	050470	050045	050057		.WORD	EM17B,EM4B,EM4C,EM4D,0		
8487	050520	123	000		EM21:	.ASCIZ	'S'		
8488	050522	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN			
8489	050524	050470	050066	000000		.WORD	EM17B,EM4D,0		
8490		050520			EM22=EM21				
8491	050532	123	000		EM23:	.ASCIZ	'S'		
8492	050534	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN			
8493	050536	050544	050374	000000		.WORD	EM23B,EM13B,0		
8494	050544	124	104	040	EM23B:	.ASCIZ	'TD A,@(R)+'		
8495	050557	123	000		EM24:	.ASCIZ	'S'		
8496	050561	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN			
8497	050562	050544	050045	050057		.WORD	EM23B,EM4B,EM4C,EM4D,0		
8498	050574	123	000		EM25:	.ASCIZ	'S'		
8499	050576	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN			
8500	050600	050544	050066	000000		.WORD	EM23B,EM4D,0		
8501	050606	123	000		EM26:	.ASCIZ	'S'		
8502	050610	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN			

ERROR MESSAGES

Address	Offset	PC	MACRO	Code	Label	Instruction	Comment
8503	050612	050620	050374	000000		.WORD EM26B,EM13B,0	
8504	050620	124	104	040	EM26B:	.ASCIZ 'TD A,@-(R)'	
8505	050633	123	000		EM27:	.ASCIZ 'S'	
8506	050635	377				.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN	
8507	050636	050620	050045	050057		.WORD EM26B,EM4B,EM4C,EM4D,0	
8508	050650	123	000		EM30:	.ASCIZ 'S'	
8509	050652	377				.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN	
8510	050654	050620	050066	000000		.WORD EM26B,EM4D,0	
8511	050662	123	000		EM31:	.ASCIZ 'S'	
8512	050664	377				.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN	
8513	050666	050674	050374	000000		.WORD EM31B,EM13B,0	
8514	050674	124	104	040	EM31B:	.ASCIZ 'TD A,N(R)'	
8515	050706	123	000		EM32:	.ASCIZ 'S'	
8516	050710	377				.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DCWN	
						.EVEN	
8517	050712	050674	050045	050057		.WORD EM31B,EM4B,EM4C,EM4D,0	
8518	050724	123	000		EM33:	.ASCIZ 'S'	
8519	050726	377				.BYTE 377 ;FLAGS 'ERTYPE' 0 PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN	
8520	050730	050674	050066	000000		.WORD EM31B,EM4D,0	
8521	050736	123	000		EM34:	.ASCIZ 'S'	
8522	050740	377				.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN	
8523	050742	050750	050374	000000		.WORD EM34B,EM13B,0	
8524	050750	124	104	040	EM34B:	.ASCIZ 'TD A,@N(R)'	
8525	050763	123	000		EM35:	.ASCIZ 'S'	
8526	050765	377				.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN	
8527	050766	050750	050045	050057		.WORD EM34B,EM4B,EM4C,EM4D,0	
8528	051000	123	000		EM36:	.ASCIZ 'S'	
8529	051002	377				.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN	
8530	051004	050750	050066	000000		.WORD EM34B,EM4D,0	
8531	051012	123	124	103	EM37:	.ASCIZ 'STCFD A,(R) FAILED.:'	
8532	051036	376				.BYTE 376 ;FLAGS 'ERTYPE' TO PRINT ERR # SPECIFIC ASCIZ MSGS, ADRS 3 LINES DOW	
						.EVEN	
8533	051040	000037	000042	000044		.WORD 37,42,44,45,47,50,0	
8534	051056	000000	051132	051245		.WORD 0,EM42B,EM44B,EM45B,EM47B,EM50B	
8535	051072	123	124	103	EM40:	.ASCIZ 'STCFD A,(R)'	
8536	051106	376				.BYTE 376 ;FLAGS 'ERTYPE' TO PRINT ERR # SPECIFIC ASCIZ MSGS, ADRS 3 LINES DOW	
						.EVEN	
8537	051110	000040	000041	000043		.WORD 40,41,43,46,0	
8538	051122	047752	050002	051166		.WORD EM2B,EM3B,EM43B,EM46B	
8539		051072			EM41=EM40		
8540		051012			EM42=EM37		
8541	051132	200	111	116	EM42B:	.ASCIZ '<CRLF>:INVERT FDFL ST 767 FAILED.:'	
8542		051072			EM43=EM40		
8543	051166	056	040	106	EM43B:	.ASCIZ ': FPS BAD.:'	
8544	051201	377				.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN	
	051202	050167	051240	050176		.WORD CRBUT,IEZBT,PRST,N5,N6,N0,WENTTO,N0,N6,N1,INSTOF,N2,N6,N1,0	
8545	051240	105	132	102	IEZBT:	.ASCIZ 'IEZBT:'	
8546		051012			EM44=EM37		
8547	05 245	200	114	117	EM44B:	.ASCIZ '<CRLF>.LOW ORDER BITS OF X11 DID NOT GET 0 ST ~.6.'	

Line	Job	Unit	Page	Code	Msg	Text
8548		051012			EM45=EM37	
8549	051322	200	050	102	EM45B: .ASCIZ	<CRLF>:(BUT OP1C) ST 251 FAILED.:
8550		051072			EM46=EM40	
8551	051355	056	040	106	EM46B: .ASCIZ	! . FPS BAD.:
8552	051370	377			.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
		.EVEN			.WORD	CRBUT, IEZBT, PRST, N4, N2, N1, WENTTO, N2, N6, N2, INSTOF, N0, N6, N2, 0
8553	051372	050167	051240	050176	EM47=EM37	
8554	051430	040	000		EM47B: .ASCIZ	!
8555	051432	377			.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
		.EVEN			.WORD	CRBUT, IFD, PRST, N1, N1, N3, WENTTO, N4, N1, N5, INSTOF, N4, N1, N4, 0
8556	051434	050167	050164	050176	EM50=EM37	
8557	051472	040	123	111	EM50B: .ASCIZ	! SIGN BAD.:
8558	051505	377			.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
		.EVEN			.WORD	CRBUT, IENBT, PRST, N5, N6, N7, WENTTO, N0, N6, N0, INSTOF, N4, N6, N0, 0
8559	051506	050167	051544	050176	IENBT: .ASCIZ	!ENBT!
8560	051544	105	116	102	EM51: .ASCIZ	!STCDF A, (R)!
8561	051551	123	124	103	.BYTE	376 ;FLAGS 'ERTYPE' TO PRINT ERR # SPECIFIC ASCIZ MSGS, ADRS 3 LINES DOW
		.EVEN			.WORD	51, 54, 55, 60, 0
8562	051566	000051	000054	000055	.WORD	EM4D, EM54B, EM55B, EM60B
8563	051600	050066	051652	051716	EM52: .ASCIZ	!STD A, (R)!
8564	051610	123	124	104	.BYTE	376 ;FLAGS 'ERTYPE' TO PRINT ERR # SPECIFIC ASCIZ MSGS, ADRS 3 LINES DOW
		.EVEN			.WORD	52, 53, 56, 57, 61, 0
8566	051624	000052	000053	000056	.WORD	EM2B, EM3B, EM56B, EM57B, EM61B
8567	051640	047752	050002	052007	.WORD	
8568		051610			EM53=EM52	
8569		051551			EM54=EM51	
8570	051652	040	106	101	EM54B: .ASCIZ	! FAILED. !<CRLF>! INVERT FDFL ST 767 FAILED. '
8571		051551			EM55-EM51	
8572	051716	200	122	117	EM55B: .ASCIZ	<CRLF>! ROUND ERROR, OR,
8573	051737	377			.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
		.EVEN			.WORD	CRBUT, IBKOUT, PRST, N4, N0, N0, WENTTO, N7, N6, N6, INSTOF, N7, N6, N7, 0
8574	051740	050167	051776	050176	IBKOUT: .ASCIZ	!BREAKOUT!
8575	051776	102	122	105	EM56=EM52	
8576	052007	051610			EM56B: .ASCIZ	! . FPS BAD.:
8577	052022	056	040	106	.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
		.EVEN			.WORD	CRBUT, IEZBT, PRST, N4, N2, N1, WENTTO, N0, N6, N2, INSTOF, N2, N6, N2, 0
8578	052024	050167	051240	050176	EM57=EM52	
8579	052062	051610			EM57B: .ASCIZ	! FPS BAD. FIV=0.!
8580	052103	040	106	120	.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
		.EVEN			.WORD	CRBUT, IFIV, PRST, N2, N6, N2, WENTTO, N1, N2, N3, INSTOF, N1, N0, N3, 0
8581	052104	050167	052142	050176	IFIV: .ASCIZ	!FIV!
8582	052142	106	111	126	EM60=EM51	
8583	052146	051551			EM60B: .ASCIZ	! FAILED. FIV=1.!
8584	052166	040	106	101	.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
		.EVEN			.WORD	CRBUT, IFIV, PRST, N2, N6, N2, WENTTO, N1, N0, N3, INSTOF, N1, N2, N3, 0
8585	052170	050167	052142	050176	EM61=EM52	
8586	052226	051610			EM61B: .ASCIZ	! . FPS BAD.
8587	052241	056	040	106	.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
		.EVEN				

ERROR MESSAGES

8588	052242	050167	052300	050176		.WORD	CRBUT,IFLAG,PRST,N1,N4,N7,WENTTO,N3,N6,N1,INSTOF,N3,N6,N5,0
8589	052300	106	114	101	IFLAG:	.ASCIZ	:FLAG:
8590	052305	123	124	103	EM62:	.ASCIZ	:STCF:
8591	052312	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
8591	052314	052324	047752	052334		.EVEN	
8592	052314	104	040	101	EM62A:	.WORD	EM62A,EM2B,EM62B,0
8593	052324	040	000		EM62B:	.ASCIZ	:D A,AC6:
8594	052336	377				.ASCIZ	: :
8595	052340	050167	052376	050176		.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
8596	052376	106	104	123	IFDST:	.EVEN	
8597	052403	123	124	103	EM63:	.WORD	CRBUT,IFDST,PRST,N7,N6,N7,WENTTO,N5,N6,N7,INSTOF,N5,N7,N7,0
8598	052410	377				.ASCIZ	:FDST:
8599	052412	052324	050002	000000		.ASCIZ	:STCF:
8600	052420	103	114	122	EM64:	.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
8601	052463	377				.EVEN	
8601	052464	050066	000000			.WORD	EM62A,EM3B,0
8605	052470	103	114	122	EM65:	.ASCIZ	:CLRD (R) FAILED:<CRLF>:ZERO X11 AT ST 770:
8606	052513	000				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
8607	052514	103	114	122	EM66:	.WORD	EM4D,0
8608	052520	377				.ASCIZ	:CLRD (R). FPS BAD.:
8609	052522	052534	050045	050057		.BYTE	0
8610	052534	104	040	050	EM66B:	.ASCIZ	:CLR:
8611	052542	103	114	122	EM67:	.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
8612	052542	377				.EVEN	
8613	052566	050167	052376	050176		.WORD	EM66B,EM4B,EM4C,EM4D,0
8614	052624	103	114	122	EM70:	.ASCIZ	:CLRD AC7. FPS BAD.:
8615	052647	116	105	107	EM71:	.ASCIZ	:CLRD (R). FEC BAD.:
8616	052656	377				.ASCIZ	:NEGF A:
8617	052660	050066	000000			.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
8618	052664	116	105	107	EM72:	.EVEN	
8619	052674	047752	000000			.WORD	EM4D,0
8620	052700	116	105	107	EM73:	.ASCIZ	:NEGF A:
8621	052704	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
8622	052706	052534	050066	000000		.EVEN	
8623	052714	116	105	107	EM74:	.WORD	EM66B,EM4D,0
8624	052720	377				.ASCIZ	:NEG:
8625	052722	052534	050045	000000		.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
8626	052730	116	105	107	EM75:	.EVEN	
8627	052734	377				.WORD	EM66B,EM4B,0
8628	052736	052534	047752	000000		.ASCIZ	:NEG:
8629	052744	101	102	123	EM76:	.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
8630	052750	377				.EVEN	
						.WORD	EM66B,EM2B,0
						.ASCIZ	:ABS:
						.BYTE	377 ;FLAGS 'FRYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN

8674	053262	053270	052767	000000		.WORD	EM113B,EM76B,0
8675	053270	104	040	100	EM113B:	.ASCIZ	:D @-(R):
8676	053300	116	105	107	EM114:	.ASCIZ	:NEG:
8677	053304	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8678	053306	053270	050066	000000		.WORD	EM113B,EM4D,0
8679	053314	116	105	107	EM115:	.ASCIZ	:NEG:
8680	053320	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8681	053322	053270	050045	000000		.WORD	EM113B,EM4B,0
8682	053330	116	105	107	EM116:	.ASCIZ	:NEG:
8683	053334	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8684	053336	053270	047752	000000		.WORD	EM113B,EM2B,0
8685	053344	101	102	123	EM117:	.ASCIZ	:ABS:
8686	053350	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8687	053352	053360	052767	000000		.WORD	EM117B,EM76B,0
8688	053360	104	040	116	EM117B:	.ASCIZ	:D N(R):
8689	053367	101	102	123	EM120:	.ASCIZ	:ABS:
8690	053373	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8691	053374	053360	050066	000000		.WORD	EM117B,EM4D,0
8692	053402	101	102	123	EM121:	.ASCIZ	:ABS:
8693	053406	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8694	053410	053360	050045	000000		.WORD	EM117B,EM4B,0
8695	053416	101	102	123	EM122:	.ASCIZ	:ABS:
8696	053422	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8697	053424	053360	047752	000000		.WORD	EM117B,EM2B,0
8698	053432	116	105	107	EM123:	.ASCIZ	:NEG:
8699	053436	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8700	053440	053446	052767	000000		.WORD	EM123B,EM76B,0
8701	053446	104	040	100	EM123B:	.ASCIZ	:D @N(R):
8702	053456	116	105	107	EM124:	.ASCIZ	:NEG:
8703	053462	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8704	053464	053446	050066	000000		.WORD	EM123B,EM4D,0
8705	053472	116	105	107	EM125:	.ASCIZ	:NEG:
8706	053476	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8707	053500	053446	050045	000000		.WORD	EM123B,EM4B,0
8708	053506	116	105	107	EM126:	.ASCIZ	:NEG:
8709	053512	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8710	053514	053446	047752	000000		.WORD	EM123B,EM2B,0
8711	053522	116	105	107	EM127:	.ASCIZ	:NEG:
8712	053526	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8713	053530	053536	052767	000000		.WORD	EM127B,EM76B,0
8714	053536	104	040	116	EM127B:	.ASCIZ	:D N(R?):
8715	053546	116	105	107	EM130:	.ASCIZ	:NEG:
8716	053552	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8717	053554	053536	050066	000000		.WORD	EM127B,EM4D,0

8718	053562	116	105	107	EM131:	.ASCIZ	NEG!			
8719	053566	377				.BYTE	377		;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN				
8720	053570	053536	047752	000000		.WORD	EM127B,EM2B,0			
8721	053576	101	102	123	EM132:	.ASCIZ	!ABS!			
8722	053602	377				.BYTE	377		;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN				
8723	053604	053612	052767	000000		.WORD	EM132B,EM76B,0			
8724	053612	104	040	100	EM132B:	.ASCIZ	!D @N(R7)!			
8725	053623	101	102	123	EM133:	.ASCIZ	!ABS!			
8726	053627	377				.BYTE	377		;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN				
8727	053630	053612	050066	000000		.WORD	EM132B,EM4D,0			
8728	053636	101	102	123	EM134:	.ASCIZ	!ABS!			
8729	053642	377				.BYTE	377		;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN				
8730	053644	053612	047752	000000		.WORD	EM132B,EM2B,0			
8731	053652	116	105	107	EM135:	.ASCIZ	!NEGD A FAILED.!<CRLF>!XOR SIGN BIT 336!			
8732	053715	377				.BYTE	377		;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN				
8733	053716	050066	000000			.WORD	EM4D,0			
8734	053722	116	105	107	EM136:	.ASCIZ	!NEGD A!			
8735	053731	377				.BYTE	377		;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN				
8736	053732	050066	000000			.WORD	EM4D,0			
8737	053736	116	105	107	EM137:	.ASCIZ	!NEGD A!			
8738	053745	377				.BYTE	377		;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN				
8739	053746	047752	000000			.WORD	EM2B,0			
8740	053752	116	105	107	EM140:	.ASCIZ	!NEG!			
8741	053756	377				.BYTE	377		;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN				
8742	053760	052534	050066	000000		.WORD	EM66B,EM4D,0			
8743	053766	116	105	107	EM141:	.ASCIZ	!NEG!			
8744	053772	377				.BYTE	377		;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN				
8745	053774	052534	050045	054006		.WORD	EM66B,EM4B,EM141C,EM4D,0			
8746	054006	040	123	120	EM141C:	.ASCIZ	!SPECIAL DEST!			
8747	054024	116	105	107	EM142:	.ASCIZ	!NEG!			
8748	054030	377				.BYTE	377		;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN				
8749	054032	052534	047752	000000		.WORD	EM66B,EM2B,0			
8750	054040	116	105	107	EM143:	.ASCIZ	!NEG!			
8751	054044	377				.BYTE	377		;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN				
8752	054046	052760	050066	000000		.WORD	EM76A,EM4D,0			
8753	054054	116	105	107	EM144:	.ASCIZ	!NEG!			
8754	054060	377				.BYTE	377		;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN				
8755	054062	052760	050045	054006		.WORD	EM76A,EM4B,EM141C,0			
8756	054072	116	105	107	EM145:	.ASCIZ	!NEG!			
8757	054076	377				.BYTE	377		;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN				
8758	054100	052760	047752	000000		.WORD	EM76A,EM2B,0			
8759	054106	116	105	107	EM146:	.ASCIZ	!NEG!			
8760	054112	377				.BYTE	377		;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN				

ERROR MESSAGES

8761	054114	053076	050066	000000		.WORD	EM102B,EM4D,0
8762	054122	116	105	107	EM147:	.ASCIZ	:NEG!
8763	054126	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8764	054130	053076	050045	054006		.WORD	EM102B,EM4B,EM141C,EM4D,0
8765	054142	116	105	107	EM150:	.ASCIZ	:NEG!
8766	054146	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8767	054150	053076	047752	000000		.WORD	EM102B,EM2B,0
8768	054156	116	105	107	EM151:	.ASCIZ	:NEG!
8769	054162	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8770	054164	053164	050066	000000		.WORD	EM106B,EM4D,0
8771	054172	116	105	107	EM152:	.ASCIZ	:NEG!
8772	054176	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8773	054200	053164	050045	054006		.WORD	EM106B,EM4B,EM141C,EM4D,0
8774	054212	116	105	107	EM153:	.ASCIZ	:NEG!
8775	054216	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8776	054220	053164	047752	000000		.WORD	EM106B,EM2B,0
8777	054226	116	105	107	EM154:	.ASCIZ	:NEG!
8778	054232	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8779	054234	053270	050066	000000		.WORD	EM113B,EM4D,0
8780	054242	116	105	107	EM155:	.ASCIZ	:NEG!
8781	054246	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8782	054250	053270	050045	054006		.WORD	EM113B,EM4B,EM141C,EM4D,0
8783	054262	116	105	107	EM156:	.ASCIZ	:NEG!
8784	054266	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8785	054270	053270	047752	000000		.WORD	EM113B,EM2B,0
8786	054276	116	105	107	EM157:	.ASCIZ	:NEG!
8787	054302	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8788	054304	054312	050066	000000		.WORD	EM157B,EM4D,0
8789	054312	106	040	050	EM157B:	.ASCIZ	:F (R)+!
8790	054321	116	105	107	EM160:	.ASCIZ	:NEG!
8791	054325	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8792	054326	054312	050045	054342		.WORD	EM157B,EM4B,EM160B,EM141C,EM4D,0
8793	054342	056	040	102	EM160B:	.ASCIZ	!. BAD CONSTAN ^T USED.!
8794	054367	116	105	107	EM161:	.ASCIZ	:NEG!
8795	054373	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8796	054374	054312	047752	000000		.WORD	EM157B,EM2B,0
8797	054402	116	105	107	EM162:	.ASCIZ	:NEG!
8798	054406	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8799	054410	054416	050066	000000		.WORD	EM162B,EM4D,0
8800	054416	104	040	050	EM162B:	.ASCIZ	:D (R7)+!
8801	054426	116	105	107	EM163:	.ASCIZ	:NEG!
8802	054432	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8803	054434	054416	047752	000000		.WORD	EM162B,EM2B,0
8804	054442	120	103	040	EM164:	.ASCIZ	:PC BAD AFTER NEG!

ERROR MESSAGES

8805	054463	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
8806	054464	054416	054342	000000		.EVEN		
8807	054472	116	105	107	EM165:	.WORD	EM162B,EM160B,0	
8808	054476	377				.ASCIZ	!NEG!	
						.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
8809	054500	052534	050066	000000		.WORD	EM66B,EM4D,0	
8810	054506	101	102	123	EM166:	.ASCIZ	!ABS!	
8811	054512	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
8812	054514	052534	050066	000000		.WORD	EM66B,EM4D,0	
8813	054522	124	123	124	EM167:	.ASCIZ	!TST!	
8814	054526	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
8815	054530	052534	050066	000000		.WORD	EM66B,EM4D,0	
8816	054536	116	105	107	EM170:	.ASCIZ	!NEG!	
8817	054542	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
8818	054544	052534	047752	000000		.WORD	EM66B,EM2B,0	
8819	054552	101	102	123	EM171:	.ASCIZ	!ABS!	
8820	054556	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
8821	054560	052534	047752	000000		.WORD	EM66B,EM2B,0	
8822	054566	124	123	124	EM172:	.ASCIZ	!TST!	
8823	054572	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
8824	054574	052534	047752	000000		.WORD	EM66B,EM2B,0	
8825	054602	116	105	107	EM173:	.ASCIZ	!NEG!	
8826	054606	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
8827	054610	052534	050002	000000		.WORD	EM66B,EM3B,0	
8828	054616	101	102	123	EM174:	.ASCIZ	!ABS!	
8829	054622	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
8830	054624	052534	050002	000000		.WORD	EM66B,EM3B,0	
8831	054632	124	123	124	EM175:	.ASCIZ	!TST!	
8832	054636	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
8833	054640	052534	050002	000000		.WORD	EM66B,EM3B,0	
8834	054646	116	105	107	EM176:	.ASCIZ	!NEG!	
8835	054652	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
8836	054654	054662	047752	000000		.WORD	EM176B,EM2B,0	
8837	054662	106	040	101	EM176B:	.ASCIZ	!F AC7!	
8838	054670	120	117	127	EM177:	.ASCIZ	!POWER MONITOR BIT FOUND SET!	
8839	054724	116	105	107	EM200:	.ASCIZ	!NEG!	
8840	054730	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
8841	054732	052534	050066	054742		.WORD	EM66B,EM4D,EM200C,0	
8842	054742	200	130	117	EM200C:	.ASCIZ	<CRLF>!XOR SIGN BIT FAILED ST 336.!	
8843	054777	116	105	107	EM201:	.ASCIZ	!NEG!	
8844	055003	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
8845	055004	052534	047752			.WORD	EM66B,EM2B	
8846	055010	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
	055012	050167	051544	050176		.WORD	CRBUT, IENBT, PRST, N3, N3, N6, WENTTO, NO, N5, N3, INSTOF, N4, N5, N3, 0	

ERROR MESSAGES

8847	055050	116	105	107	EM202:	.ASCIZ	!NEG!	
8848	055054	377				.BYTE	377	; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
8849	055056	052534	047752			.WORD	EM66B,EM2B	
8850	055062	377				.BYTE	377	; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
	055064	050167	051544	050176		.WORD	CRBUT,IENBT,PRST,N3,N3,N6,WENTTO,N4,N5,N3,INSTOF,NO,N5,N3,0	
8851	055122	101	102	123	EM203:	.ASCIZ	!ABS!	
8852	055126	377				.BYTE	377	; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
8853	055130	052534	050066			.WORD	EM66B,EM4D	
8854	055134	377				.BYTE	377	; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
	055136	050167	055237	050176		.WORD	CRBUT,IOP1B,PRST,NO,N5,N5,WENTTO,N3,N3,N6,INSTOF,N3,N3,N5,OR	
8855	055174	050167	051544	050176		.WORD	CRBUT,IENBT,PRST,N3,N3,N5,WENTTO,N4,N5,N2,INSTOF,NO,N5,N2,0	
8856	055232	054	040	117	OR:	.ASCIZ	!OR!	
8857	055237	117	120	061	IOP1B:	.ASCIZ	!OP1B!	
8858	055244	101	102	123	EM204:	.ASCIZ	!ABS!	
8859	055250	377				.BYTE	377	; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
8860	055252	052534	050066	055262		.WORD	EM66B,EM4D,EM204C,0	
8861	055262	200	130	117	EM204C:	.ASCIZ	<CRLF>!XOR SIGN BIT FAILED ST 452.!	
8862	055317	124	123	124	EM205:	.ASCIZ	!TST!	
8863	055323	377				.BYTE	377	; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
8864	055324	052534	050066			.WORD	EM66B,EM4D	
8865	055330	377				.BYTE	377	; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
	055332	050167	055237	050176		.WORD	CRBUT,IOP1B,PRST,NO,N5,N5,WENTTO,N3,N3,N6,INSTOF,N3,N3,N4,0	
8866	055370	124	123	124	EM206:	.ASCIZ	!TST!	
8867	055374	377				.BYTE	377	; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
8868	055376	052534	047752			.WORD	EM66B,EM2B	
8869	055402	377				.BYTE	377	; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
	055404	050167	051544	050176		.WORD	CRBUT,IENBT,PRST,N3,N3,N4,WENTTO,N4,N5,N3,INSTOF,NO,N5,N3,0	
8870	055442	124	123	124	EM207:	.ASCIZ	!TST!	
8871	055446	377				.BYTE	377	; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
8872	055450	052534	050066			.WORD	EM66B,EM4D	
8873	055454	377				.BYTE	377	; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
	055456	050167	055237	050176		.WORD	CRBUT,IOP1B,PRST,NO,N5,N7,WENTTO,N3,N3,N5,INSTOF,N3,N3,N4,0	
8874	055514	124	123	124	EM210:	.ASCIZ	!TST!	
8875	055520	377				.BYTE	377	; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
8876	055522	052534	050066			.WORD	EM66B,EM4D	
8877	055526	377				.BYTE	377	; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
	055530	050167	051544	050176		.WORD	CRBUT,IENBT,PRST,N3,N3,N4,WENTTO,NO,N5,N3,INSTOF,N4,N5,N3,0	
8878	055566	124	123	124	EM211:	.ASCIZ	!TST!	
8879	055572	377				.BYTE	377	; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
8880	055574	052534	050066	050167		.WORD	EM66B,EM4D,CRBUT,IOP1B,PRST,N2,N5,N5,WENTTO,N3132,INSTOF,N3,N1,NO,0	
8881	055632	063	061	061	N3132:	.ASCIZ	!311 OR 312!	
8882	055645	124	123	124	EM212:	.ASCIZ	!TST!	
8883	055651	377				.BYTE	377	; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN

8884	055652	052534	047752			.EVEN	
8885	055656	377				.WORD	EM66B,EM2B
						.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
	055660	050167	051544	050176		.EVEN	
8886	055716	124	123	124	EM213:	.WORD	CRBUT, IENBT, PRST, N3, N1, NO, WENTTO, N4, NO, N2, INSTOF, NO, NO, N2, 0
8887	055722	377				.ASCIZ	!TST!
						.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
	055724	052534	047752	055774		.EVEN	
8888	055734	377				.WORD	EM66B,EM2B,EM213C,EM213D
						.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
	055736	050167	056022	050176		.EVEN	
8890	055774	040	106	111	EM213C:	.WORD	CRBUT, IFIUUV, PRST, N2, N5, N7, WENTTO, N3, N5, N5, INSTOF, N2, N5, N5, 0
8891	056004	054	040	117	EM213D:	.ASCIZ	! FIUV=0!
8892	056022	106	111	125	IFIUUV:	.ASCIZ	! OPERAND=-0.!
8893	056027	124	123	124	EM214:	.ASCIZ	! FIUV!
8894	056033	377				.ASCIZ	!TST!
						.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
	056034	052534	047752	056104		.EVEN	
8895	056044	377				.WORD	EM66B,EM2B,EM214B,EM213D
						.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
	056046	050167	056022	050176		.EVEN	
8897	056104	040	106	111	EM214B:	.WORD	CRBUT, IFIUUV, PRST, N2, N5, N7, WENTTO, N2, N5, N5, INSTOF, N3, N5, N5, 0
8903	056114	120	000		EM215:	.ASCIZ	! FIUV=1!
8904	056116	377				.ASCIZ	!P!
						.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
	056120	056163	056151	056200		.EVEN	
8905	056140	067	064	066	N746T2:	.WORD	PCBAD, NEGDNR, BADCON, N746T2, BUTIN, EM141C, EM4D, 0
8906	056151	116	105	107	NEGDNR:	.ASCIZ	!746 746.!
8907	056163	103	040	102	PCBAD:	.ASCIZ	!NEGD N(R)!
8908	056200	056	040	102	BADCON:	.ASCIZ	!C BAD AFTER !
8909	056225	056	000		PERIOD:	.ASCIZ	! . BAD CONSTANT USED !
8910	056227	200	117	122	BUTIN:	.ASCIZ	! .!
8911	056251	116	105	107	EM216:	.ASCIZ	<CRLF>!OR (BUT FDST) !N.
8912	056255	377				.ASCIZ	!NEG!
						.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
	056256	053360	050066	000000		.EVEN	
8914	056264	116	105	107	EM217:	.WORD	EM117B,EM4D,0
8915	056270	377				.ASCIZ	!NEG!
						.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
	056272	053360	050045	054006		.EVEN	
8917	056304	116	105	107	EM220:	.WORD	EM117B,EM4B,EM141C,EM4D,0
8918	056310	377				.ASCIZ	!NEG!
8919						.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
	056312	053360	047752	000000		.EVEN	
8920	056320	120	000		EM221:	.WORD	EM117B,EM2B,0
8921	056322	377				.ASCIZ	!P!
						.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
	056324	056163	056355	056200		.EVEN	
8923	056344	067	064	067	N747T2:	.WORD	PCBAD, NEGDAR, BADCON, N747T2, BUTIN, EM141C, EM4D, 0
8924	056355	116	105	107	NEGDAR:	.ASCIZ	!747 747.!
8925	056370	116	105	107	EM222:	.ASCIZ	!NEGD @N(R)!
8926	056374	377				.ASCIZ	!NEG!
						.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
	056376	053446	050066	000000		.EVEN	
8928	056404	116	105	107	EM223:	.WORD	EM123B,EM4D,0
8929	056410	377				.ASCIZ	!NEG!
8930						.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN

8931	056412	053446	050045	054006		.EVEN	
8932	056424	116	105	107	EM224:	.WORD	EM123B,EM4B,EM141C,EM4D,0
8933	056430	377				.ASCIZ	:NEG!
						.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8934	056432	053446	047752	000000		.WORD	EM123B,EM2B,0
8935	056440	114	000		EM225:	.ASCIZ	:L!
8936	056442	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8937	056444	056452	050045	000000		.WORD	EM225B,EM4B,0
8938	056452	104	106	120	EM225B:	.ASCIZ	:DFPS (R)!
8939	056463	114	000		EM226:	.ASCIZ	:L!
8940	056465	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8941	056466	056452	047752	000000		.WORD	EM225B,EM2B,0
8942	056474	114	000		EM227:	.ASCIZ	:L!
8943	056476	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8944	056500	056452	056506	000000		.WORD	EM225B,EM227B,0
8945	056506	040	124	122	EM227B:	.ASCIZ	' TRAPPED TO 4.!
8946	056525	114	000		EM230:	.ASCIZ	:L!
8947	056527	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8948	056530	056536	050045	000000		.WORD	EM230B,EM4B,0
8949	056536	104	106	120	FM230B:	.ASCIZ	:DFPS (R)+!
8950	056550	114	000		EM231:	.ASCIZ	:L!
8951	056552	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8952	056554	056536	047752	000000		.WORD	EM230B,EM2B,0
8953	056562	114	000		EM232:	.ASCIZ	:L!
8954	056564	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8955	056566	056536	056506	000000		.WORD	EM230B,EM227B,0
8956	056574	114	000		EM233:	.ASCIZ	:L!
8957	056576	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8958	056600	056606	050045	000000		.WORD	EM233B,EM4B,0
8959	056606	104	106	120	EM233B:	.ASCIZ	:DFPS -(R)!
8960	056620	114	000		EM234:	.ASCIZ	:L!
8961	056622	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8962	056624	056606	047752	000000		.WORD	EM233B,EM2B,0
8963	056632	114	000		EM235:	.ASCIZ	:L!
8964	056634	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8965	056636	056606	056506	000000		.WORD	EM233B,EM227B,0
8966	056644	114	000		EM236:	.ASCIZ	:L!
8967	056646	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8968	056650	056656	050045	000000		.WORD	EM236B,EM4B,0
8969	056656	104	106	120	EM236B:	.ASCIZ	:DFPS @ (R)+!
8970	056671	114	000		EM237:	.ASCIZ	:L!
8971	056673	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8972	056674	056656	047752	000000		.WORD	EM236B,EM2B,0
8973	056702	114	000		EM240:	.ASCIZ	:L!
8974	056704	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN

ERROR MESSAGES

Line	Address	Offset	Value	Code	Label	Details
8975	056706	056656	056506	000000		.EVEN
8976	056714	114	000		EM241:	.WORD EM236B,EM227B,0
8977	056716	377				.ASCIZ !L! .BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
8978	056720	056726	050045	000000		.EVEN
8979	056726	104	106	120	EM241B:	.WORD EM241B,EM4B,0
8980	056741	114	000		EM242:	.ASCIZ !DFPS @-(R)!
8981	056743	377				.ASCIZ !L! .BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
8982	056744	056726	047752	000000		.EVEN
8983	056752	114	000		EM243:	.WORD EM241B,EM2B,0
8984	056754	377				.ASCIZ !L! .BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
8985	056756	056726	056506	000000		.EVEN
8986	056764	114	000		EM244:	.WORD EM241B,EM227B,0
8987	056766	377				.ASCIZ !L! .BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
8988	056770	056776	050045	000000		.EVEN
8989	056776	104	106	120	EM244B:	.WORD EM244B,EM4B,0
8990	057010	114	000		EM245:	.ASCIZ !DFPS N(R)!
8991	057012	377				.ASCIZ !L! .BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
8992	057014	056776	047752	000000		.EVEN
8993	057022	120	103	040	FM246:	.WORD EM244B,EM2B,0
8994	057040	376				.ASCIZ !PC BAD AFTER ! .BYTE 376 ;FLAGS 'ERTYPE' TO PRINT ERR # SPECIFIC ASCIZ MSGS, ADRS 3 LINES DOW
8995	057042	000246	000252	000254		.EVEN
8996	057052	057060	057142	057170		.WORD 246,252,254,0
8997	057060	114	104	106	EM246B:	.WORD EM246B,EM252B,EM254B
8998	057073	114	000		EM247:	.ASCIZ !LDFPS N(R)!
8999	057075	377				.ASCIZ !L! .BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9000	057076	056776	056506	000000		.EVEN
9001	057104	114	000		EM250:	.WORD EM244B,EM227B,0
9002	057106	377				.ASCIZ !L! .BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9003	057110	057116	050045	000000		.EVEN
9004	057116	104	106	120	EM250B:	.WORD EM250B,EM4B,0
9005	057131	114	000		EM251:	.ASCIZ !DFPS @N(R)!
9006	057133	377				.ASCIZ !L! .BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9007	057134	057116	047752	000000		.EVEN
9008		057022			EM252=EM246	.WORD EM250B,EM2B,0
9009	057142	114	104	106	EM252B:	.ASCIZ !LDFPS @N(R)!
9010	057156	114	000		EM253:	.ASCIZ !L! .BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9011	057160	377				.EVEN
9012	057162	057116	056506	000000		.WORD EM250B,EM227B,0
9013		057022			EM254=EM246	.ASCIZ !LDFPS @N(R)!
9014	057170	114	104	103	EM254B:	.ASCIZ !LDCLD (R7)+,A,
9015	057206	114	104	103	EM255:	.ASCIZ !LDCLD (R7)+,A,
9016	057224	377				.ASCIZ !L! .BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9017	057226	056506	000000			.EVEN
9018	057232	114	000		EM256:	.WORD EM227B,0
9019	057234	377				.ASCIZ !L! .BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN

ERROR MESSAGES

Line	Address	Offset	Value	Label	Code	Message
9020	057236	057244	050045	000000		.EVEN
9021	057244	104	103	114	EM256B:	.WORD EM256B,EM4B,0
9022	057260	114	000		EM257:	.ASCIZ ;DCLD (R)+,A'
9023	057262	377				.ASCIZ ;L' .BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9024	057264	057244	047752	000000		.EVEN
9025	057272	114	000		EM260:	.WORD EM256B,EM2B,0
9026	057274	377				.ASCIZ ;L' .BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9027	057276	057304	050066	000000		.EVEN
9028	057304	104	103	111	EM260B:	.WORD EM260B,EM4D,0
9029	057330	114	000		EM261:	.ASCIZ ;DCIF OR LDCLF (R),A'
9030	057332	377				.ASCIZ ;L' .BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9031	057334	057304	047752	000000		.EVEN
9032	057342	114	000		EM262:	.WORD EM260B,EM2B,0
9033	057344	377				.ASCIZ ;L' .BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9034	057346	057415	050066			.EVEN
9035	057352	377				.WORD EM262B,EM4D .BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9036	057412	050167	057412	050176		.EVEN
9037	057415	106	114	000	IFL:	.WORD CRBUT,IFL,PRST,N2,N7,N7,WENTTO,N3,NO,NO,INSTOF,N3,NO,N1,0
9038	057430	104	103	111	FM262B:	.ASCIZ ;IFL'
9039	057432	114	000		EM263:	.ASCIZ ;DCIF (R),A'
		377				.ASCIZ ;L' .BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9040	057434	057442	047752	000000		.EVEN
9041	057442	104	103	114	EM263B:	.WORD EM263B,EM2B,0
9042	057455	114	000		EM264:	.ASCIZ ;DCLF (R),A'
9043	057457	377				.ASCIZ ;L' .BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9044	057460	057415	050066	057470		.EVEN
9045	057470	200	125	123	EM264C:	.WORD EM262B,EM4D,EM264C,0
9046	057542	114	000		EM265:	.ASCIZ <CRLF>;USED CONSTANT 237 INSTEAD OF 217 ST 107.;
9047	057544	377				.ASCIZ ;L' .BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9048	057546	057304	050066	057556		.EVEN
9049	057556	200	123	105	EM265C:	.WORD EM260B,EM4D,EM265C,0
9050	057613	114	000		EM266:	.ASCIZ <CRLF>'SET SIGN BIT FAILED ST 146.;
9051	057615	377				.ASCIZ ;L' .BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9052	057616	057304	050066			.EVEN
9053	057622	377				.WORD EM260B,EM4D .BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9054	057624	050167	057662	050176		.EVEN
9055	057662	130	116	102	IXNBT:	.WORD CRBUT,IXNBT,PRST,N3,N7,N2,WENTTO,N1,N5,N2,INSTOF,N1,N1,N2,0
9056	057667	114	000		EM267:	.ASCIZ ;IXNBT'
		377				.ASCIZ ;L' .BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9057	057672	057442	050066	057702		.EVEN
9058	057702	200	125	123	EM267C:	.WORD EM263B,EM4D,EM267C,0
9059	057754	114	000		EM270:	.ASCIZ <CRLF>;USED CONSTANT 217 INSTEAD OF 237 ST 107.'
9060	057756	377				.ASCIZ ;L' .BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9061	057760	057442	050066	057770		.EVEN
						.WORD EM263B,EM4D,EM270C,0

ERROR MESSAGES

9062	057770	040	122	117	EM270C:	.ASCIZ	' ROUND ERROR.
9063	060006	114	000		EM271:	.ASCIZ	'L'
9064	060010	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9065	060012	057442	050066	060022		.WORD	EM263B,EM4D,EM271C,0
9066	060022	040	124	122	EM271C:	.ASCIZ	' TRUNCATION ERROR.'
9067	060045	114	000		EM272:	.ASCIZ	'L'
9068	060047	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9069	060050	057304	050066	060060		.WORD	EM260B,EM4D,EM272C,0
9070	060060	200	122	061	EM272C:	.ASCIZ	<CRLF>'R14 NOT INCREMENTED ST 630.'
9071	060115	114	000		EM273:	.ASCIZ	'L'
9072	060117	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9073	060120	060126	050066	000000		.WORD	EM273B,EM4D,0
9074	060126	104	103	111	EM273B:	.ASCIZ	'DCID OR LDCLD (R),A'
9075	060152	114	000		EM274:	.ASCIZ	'L'
9076	060154	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9077	060156	060126	047752	000000		.WORD	EM273B,EM2B,0
9078	060164	114	000		EM275:	.ASCIZ	'L'
9079	060166	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9080	060170	060234	050066			.WORD	EM275B,EM4D
9081	060174	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
	060176	050167	057412	050176		.WORD	CRBUT,IFL,PRST,N2,N7,N7,WENTTO,N3,NO,NO,INSTOF,N3,NO,N1,0
9082	060234	104	103	111	EM275B:	.ASCIZ	'DCID (R),A'
9083	060247	114	000		EM276:	.ASCIZ	'L'
9084	060251	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9085	060252	060234	050066	060262		.WORD	EM275B,EM4D,EM276C,0
9086	060262	200	125	123	EM276C:	.ASCIZ	<CRLF>'USED CONSTANT 237 INSTEAD OF 217 ST 107.'
9087	060334	114	000		EM277:	.ASCIZ	'L'
9088	060336	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9089	060340	060234	050066	060350		.WORD	EM275B,EM4D,EM277C,0
9090	060350	200	123	105	EM277C:	.ASCIZ	<CRLF>'SET SIGN FAILED ST 146.'
9091	060401	114	104	103	EM300:	.ASCIZ	'LDCLD (R),A'
9092	060415	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9093	060416	050066	060424	000000		.WORD	EM4D,EM300C,0
9094	060424	200	125	123	EM300C:	.ASCIZ	<CRLF>'USED CONSTANT 217 INSTEAD OF 237 ST 107.'
9095	060476	114	000		EM301:	.ASCIZ	'L'
9096	060500	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9097	060502	060510	050066	000000		.WORD	EM301B,EM4D,0
9098	060510	104	105	130	EM301B:	.ASCIZ	'DEXP (R),A'
9099	060523	114	000		EM302:	.ASCIZ	'L'
9100	060525	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9101	060526	060510	047752	000000		.WORD	EM301B,EM2B,0
9102	060534	114	000		EM303:	.ASCIZ	'L'
9103	060536	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9104	060540	060510	050002	000000		.WORD	EM301B,EM3B,0
9105	060546	114	000		EM304:	.ASCIZ	'L'

9106	060550		377			.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9107	060552	060510	050066	060562		.EVEN		
9108	060562	200	105	130	EM304C:	.WORD	EM301B,EM4D,EM304C,0	
9109	060626	114	000		EM305:	.ASCIZ	<CRLF>.EXCESS 200 CALCULATION ST 624 BAD.!	
9110	060630	377				.ASCIZ	!L!	
						.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9111	060632	060510	047752	060642		.WORD	EM301B,EM2B,EM305C,0	
9112	060642	200	050	102	EM305C:	.ASCIZ	<CRLF>! (BUT ENBT,EZBT,XNBT) ST 625 DID NOT GO TO 304.!	
9113	060722	114	000		EM306:	.ASCIZ	!L!	
9114	060724	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9115	060726	060510	047752			.WORD	EM301B,EM2B	
9116	060732	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
	060734	050167	051240	050176		.WORD	CRBUT,IEZBT,PRST,N5,N4,N4,WENTTO,N5,N0,N4,INSTOF,N7,N0,N4,OR	
9117	060772	050167	051240	050176		.WORD	CRBUT,IEZBT,PRST,N7,N0,N4,WENTTO,N2,N6,N4,INSTOF,N0,N6,N4,0	
9118	061030	114	000		EM307:	.ASCIZ	!L!	
9119	061032	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9120	061034	060510	050066			.WORD	EM301B,EM4D	
9121	061040	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
	061042	050167	051240	050176		.WORD	CRBUT,IEZBT,PRST,N7,N0,N4,WENTTO,N0,N6,N4,INSTOF,N2,N6,N4,0	
9122	061100	114	000		FM310:	.ASCIZ	!L!	
9123	061102	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9124	061104	060510	047752			.WORD	EM301B,EM2B	
9125	061110	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
	061112	050167	061150	050176		.WORD	CRBUT,IFIU,PRST,N2,N6,N4,WENTTO,N1,N1,N5,INSTOF,N1,N5,N5,0	
9126	061150	106	111	125	IFIU:	.ASCIZ	!IFIU!	
9127	061154	114	000		EM311:	.ASCIZ	!L!	
9128	061156	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9129	061160	060510	050066			.WORD	EM301B,EM4D	
9130	061164	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
	061166	050167	061150	050176		.WORD	CRBUT,IFIU,PRST,N2,N6,N4,WENTTO,N1,N5,N5,INSTOF,N1,N1,N5,0	
9131	061224	114	000		EM312:	.ASCIZ	!L!	
9132	061226	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9133	061230	060510	050066			.WORD	EM301B,EM4D	
9134	061234	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
	061236	050167	051240	050176		.WORD	CRBUT,IEZBT,PRST,N5,N4,N4,WENTTO,N7,N0,N4,INSTOF,N5,N0,N4,0	
9135	061274	114	000		EM313:	.ASCIZ	!L!	
9136	061276	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9137	061300	060510	050066			.WORD	EM301B,EM4D	
9138	061304	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
	061306	050167	061150	050176		.WORD	CRBUT,IFIU,PRST,N5,N0,N4,WENTTO,N1,N5,N5,INSTOF,N1,N1,N5,0	
9139	061344	114	000		EM314:	.ASCIZ	!L!	
9140	061346	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9141	061350	060510	050066			.WORD	EM301B,EM4D	

ERROR MESSAGES

9142	61354	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
	061356	050167	052142	050176		.EVEN		
9143	061414	114	000		EM315:	.WORD	CRBUT,IFIV,PRST,N1,N0,N4,WENTTO,N1,N1,N6,INSTOF,N1,N3,N6,0	
9144	061416	377				.ASCIZ	'L'	
						.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9145	061420	060510	050066			.WORD	EM301B,EM4D	
9146	061424	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
	061426	050167	052142	050176		.WORD	CRBUT,IFIV,PRST,N1,N0,N4,WENTTO,N1,N3,N6,INSTOF,N1,N1,N6,0	
9147	061464	114	000		EM316:	.ASCIZ	'L'	
9148	061466	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9149	061470	060510	050066			.WORD	EM301B,EM4D	
9150	061474	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
	061476	050167	052142	050176		.WORD	CRBUT,IFIV,PRST,N1,N4,N4,WENTTO,N1,N1,N6,INSTOF,N1,N3,N6,0	
9151	061534	114	000		EM317:	.ASCIZ	'L'	
9152	061536	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9153	061540	060510	050066			.WORD	EM301B,EM4D	
9154	061544	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
	061546	050167	052142	050176		.WORD	CRBUT,IFIV,PRST,N1,N4,N4,WENTTO,N1,N3,N6,INSTOF,N1,N1,N6,0	
9155	061604	114	000		FM320:	.ASCIZ	'L'	
9156	061606	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9157	061610	060510	050066			.WORD	EM301B,EM4D	
9158	061614	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
	061616	050167	052142	050176		.WORD	CRBUT,IFIV,PRST,N3,N4,N4,WENTTO,N1,N1,N6,INSTOF,N1,N3,N6,0	
9159	061654	114	000		EM321:	.ASCIZ	'L'	
9160	061656	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9161	061660	060510	050066			.WORD	EM301B,EM4D	
9162	061664	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
	061666	050167	052142	050176		.WORD	CRBUT,IFIV,PRST,N3,N4,N4,WENTTO,N1,N3,N6,INSTOF,N1,N1,N6,0	
9163	061724	123	000		EM322:	.ASCIZ	'S'	
9164	061726	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9165	061730	061736	050066	000000		.WORD	EM322B,EM4D,0	
9166	061736	124	103	104	EM322B:	.ASCIZ	'TCDI OR STCDL (R),A'	
9167	061762	123	000		EM323:	.ASCIZ	'S'	
9168	061764	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9169	061766	061736	047752	000000		.WORD	EM322B,EM2B,0	
9170	061774	123	000		EM324:	.ASCIZ	'S'	
9171	061776	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9172	062000	061736	050002	000000		.WORD	EM322B,EM3B,0	
9173	062006	123	000		EM325:	.ASCIZ	'S'	
9174	062010	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9175	062012	062022	047752	062035		.WORD	EM325B,EM2B,EM325C,0	
9176	062022	124	103	104	EM325B:	.ASCIZ	'TCDL (R),A'	
9177	062035	200	103	114	EM325C:	.ASCIZ	<CRLF>;CLEAR FLAG ST 774 FAILED, OR	

Line	Address	Offset	Value	Label	Text
9178	062073	377			.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
	062074	050167	052300	050176	.EVEN .WORD CRBUT,IFLAG,PRST,N6,N6,N2,WENTTO,N3,N6,N5,INSTOF,N3,N6,N1,0
9179	062132	062006	000	EM326=EM325	.ASCIZ :S!
9180	062132	123		EM327:	.ASCIZ :S!
9181	062134	377			.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
	062136	062022	050066		.EVEN .WORD EM325B,EM4D
9182	062136	062022	050066		.WORD EM325B,EM4D
9183	062142	377			.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
	062144	050167	051544	050176	.EVEN .WORD CRBUT,IENBT,PRST,N6,N3,N2,WENTTO,N4,N7,N3,INSTOF,N0,N7,N3,0
9184	062202	123	000	EM330:	.ASCIZ :S!
9185	062204	377			.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
	062206	062022	047752		.EVEN .WORD EM325B,EM2B
9186	062206	062022	047752		.WORD EM325B,EM2B
9187	062212	377			.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
	062214	050167	062252	050176	.EVEN .WORD CRBUT,IFIC,PRST,N0,N0,N4,WENTTO,N3,N0,N5,INSTOF,N3,N1,N5,0
9188	062252	106	111	103 IFIC:	.ASCIZ :FIC'
9189	062256	123	000	EM331:	.ASCIZ :S!
9190	062260	377			.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
	062262	062022	047752		.EVEN .WORD EM325B,EM2B
9191	062262	062022	047752		.WORD EM325B,EM2B
9192	062266	377			.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
	062270	050167	062252	050176	.EVEN .WORD CRBUT,IFIC,PRST,N0,N0,N4,WENTTO,N3,N1,N5,INSTOF,N3,N0,N5,0
9193	062326	123	000	EM332:	.ASCIZ :S!
9194	062330	377			.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
	062332	062342	050066	062355	.EVEN .WORD EM332B,EM4D,EM332C,0
9195	062332	062342	050066	062355	.WORD EM332B,EM4D,EM332C,0
9196	062342	124	103	104 EM332B:	.ASCIZ :TCDI (R),A!
9197	062355	200	125	12- EM332C:	.ASCIZ <CRLF>!USED CONSTANT 37 INSTEAD OF 17 ST 66.!
9198	061762			EM333=EM323	
9199	062424	123	000	EM334:	.ASCIZ :S!
9200	062426	377			.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
	062430	062342	047752	062440	.EVEN .WORD EM332B,EM2B,EM334C,0
9201	062430	062342	047752	062440	.WORD EM332B,EM2B,EM334C,0
9202	062440	200	125	123 EM334C:	.ASCIZ <CRLF>!USED CONSTANT 37 INSTEAD OF 17 ST 66.!
9203	062507	123	000	EM335:	.ASCIZ :S!
9204	062511	377			.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
	062512	062342	050066		.EVEN .WORD EM332B,EM4D
9205	062512	062342	050066		.WORD EM332B,EM4D
9206	062516	377			.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
	062520	050167	051544	050176	.EVEN .WORD CRBUT,IENBT,PRST,N6,N3,N2,WENTTO,N0,N7,N3,INSTOF,N4,N7,N3,0
9207	062556	123	000	EM336:	.ASCIZ :S!
9208	062560	377			.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
	062562	062342	047752	062574	.EVEN .WORD EM332B,EM2B,EM336C,EM4D,0
9209	062562	062342	047752	062574	.WORD EM332B,EM2B,EM336C,EM4D,0
9210	062574	200	123	105 EM336C:	.ASCIZ <CRLF>!SET FN ST 473!
9211	062613	123	000	EM337:	.ASCIZ :S!
9212	062615	377			.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
	062616	062022	050066		.EVEN .WORD EM325B,EM4D
9213	062616	062022	050066		.WORD EM325B,EM4D
9214	062622	377			.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
	062624	050167	062662	050176	.EVEN .WORD CRBUT,ICOUT,PRST,N2,N7,N5,WENTTO,N0,N7,N4,INSTOF,N2,N7,N4,0

Line	Address	Code	Text	Macro	ICOUT	Word	Byte	Even	Word	Text
9215	062662	103	117	125	ICOUT:	.ASCIZ	:COUT:			
9216	062667	123	000		EM340:	.ASCIZ	:S!			
9217	062671	377				.BYTE	377			;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9218	062672	062022	050066			.EVEN				
9219	062676	377				.WORD	EM325B,EM4D			
9220	062700	050167	062662	050176		.BYTE	377			;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9221	062736	123	000		EM341:	.ASCIZ	:S!			CRBUT, ICOUT, PRST, N2, N7, N5, WENTTO, N2, N7, N4, INSTOF, N0, N7, N4, 0
9222	062742	062022	047752			.BYTE	377			;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9223	062746	377				.EVEN				
9224	062750	050167	051240	050176		.WORD	EM325B,EM2B			
9225	063006	123	000		EM342:	.ASCIZ	:S!			CRBUT, IEZBT, PRST, N3, N7, N7, WENTTO, N6, N3, N3, INSTOF, N4, N3, N3, 0
9226	063010	377				.BYTE	377			;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9227	063012	062022	050066			.EVEN				
9228	063020	050167	062662	050176		.WORD	EM325B,EM4D			
9229	063056	123	000		EM343:	.ASCIZ	:S!			CRBUT, ICOUT, PRST, N3, N6, N0, WENTTO, N6, N5, N4, INSTOF, N4, N5, N4, 0
9230	063062	062022	050066			.BYTE	377			;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9231	063066	377				.EVEN				
9232	063070	050167	063126	050176		.WORD	EM325B,EM4D			
9233	063126	116	102	111	INBIT:	.ASCIZ	:INBIT!			CRBUT, INBIT, PRST, N6, N5, N4, WENTTO, N5, N3, N1, INSTOF, N4, N3, N1, 0
9234	063133	123	000		EM344:	.ASCIZ	:S!			
9235	063135	377				.BYTE	377			;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9236	063136	062022	050066			.EVEN				
9237	063142	377				.WORD	EM325B,EM4D			
9238	063144	050167	062662	050176		.ASCIZ	:S!			CRBUT, ICOUT, PRST, N3, N6, N0, WENTTO, N4, N5, N4, INSTOF, N6, N5, N4, OR
9239	063202	050167	063126	050176		.WORD	EM325B,EM4D			CRBUT, INBIT, N6, N5, N4, WENTTO, N4, N3, N1, INSTOF, N5, N3, N1, 0
9240	063240	123	000		EM345:	.ASCIZ	:S!			
9241	063242	377				.BYTE	377			;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9242	063244	062342	050066			.EVEN				
9243	063250	377				.WORD	EM332B,EM4D			
9244	063252	050167	057412	050176		.ASCIZ	:STCFL (R),A!			CRBUT, IFL, PRST, N6, N3, N3, WENTTO, N6, N5, N5, INSTOF, N6, N5, N4, 0
9245	063310	123	124	103	EM346:	.ASCIZ	:S!			
9246	063324	377				.BYTE	377			;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9247	063326	050066	063334	000000		.EVEN				
9248	063334	200	132	105	EM346C:	.ASCIZ	<CRLF>:ZERO LOW ORDER PART OF X11 FAILED ST 773.!			
9249	063407	123	000		EM347:	.ASCIZ	:S!			
9250	063411	377				.BYTE	377			;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9251	063412	063420	050066	000000		.EVEN				
9252	063420	124	105	130	EM347B:	.ASCIZ	:TEXP A, (R)!			EM347B,EM4D,0
9253	063433	123	000		EM350:	.ASCIZ	:S!			
9254	063435	377				.BYTE	377			;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN

ERROR MESSAGES

Address	Offset	PC	SP	OP	Label	Instruction	Comment
9252	063436	063420	047752	000000		.EVEN	
9253	063444	115	117	122	EM351:	.WORD	EM347B,EM2B,0
9254	063467	127	122	111		.ASCII	!MORE THAN ONE WORD
9255	063540	377				.ASCIZ	!WRITTEN BY STEXP A,(R).!<CRLF>!ZERO FDFL ST 347!
						.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9256	063542	050066	000000			.EVEN	
9257	063546	123	000		EM352:	.WORD	EM4D,0
9258	063550	377				.ASCIZ	!S!
						.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9259	063552	063420	047752			.EVEN	
9260	063556	377				.WORD	EM347B,EM2B
						.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
	063560	050167	051544	050176		.EVEN	
9261	063616	123	000		EM353:	.WORD	CRBUT, IENB*, PRST, N3, N7, N6, WENTTO, NO, N7, N1, INSTOF, N4, N7, N1, 0
9262	063620	377				.ASCIZ	!S!
						.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9263	063622	063420	047752			.EVEN	
9264	063626	377				.WORD	EM347B,EM2B
						.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
	063630	050167	051240	050176		.EVEN	
9265	063666	123	000		EM354:	.WORD	CRBUT, IEZBT, PRST, NO, N7, N1, WENTTO, NO, N7, N2, INSTOF, N2, N7, N2, 0
9266	063670	377				.ASCIZ	!S!
						.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9267	063672	063420	047752			.EVEN	
9268	063676	377				.WORD	EM347B,EM2B
						.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
	063700	050167	051240	050176		.EVEN	
9269	063736	123	000		EM355:	.WORD	CRBUT, IEZBT, PRST, NO, N7, N1, WENTTO, N2, N7, N2, INSTOF, NO, N7, N2, 0
9270	063740	377				.ASCIZ	!S!
						.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9271	063742	063420	047752			.EVEN	
9272	063746	377				.WORD	EM347B,EM2B
						.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
	063750	050167	051544	050176		.EVEN	
9273	064006	123	124	123	EM356:	.WORD	CRBUT, IENBT, PRST, N3, N7, N6, WENTTO, N4, N7, N1, INSTOF, NO, N7, N1, 0
9274	064035	377				.ASCIZ	!STST (R) GOT BAD FEC.!<CRLF>
						.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9275	064036	064042	000000			.EVEN	
9276	064042	101	106	124	EM356B:	.WORD	EM356B,0
9277	064112	123	124	123	EM357:	.ASCIZ	!AFTER EXECUTING AN ILLEGAL FPP OP CODE.!
9278	064141	377				.ASCIZ	!STST (R) GOT BAD FEA.!<CRLF>
						.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9279	064142	064042	000000			.EVEN	
9280	064146	117	116	114	EM360:	.WORD	EM356B,0
9281	064211	123	105	124		.ASCII	!ONLY ONE WORD WRITTEN BY STST (R).!
9282	064231	377				.ASCIZ	!SET FDFL ST 636!
						.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9283	064232	050066	000000			.EVEN	
9284	064236	123	124	123	EM361:	.WORD	EM4D,0
9285	064247	377				.ASCIZ	!STST (R)!
						.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9286	064250	047752	000000			.EVEN	
9287	064254	116	117	116	EM362:	.WORD	EM2B,0
9288	064322	111	115	120		.ASCII	!NON-RESIDENT MEMORY MANAGEMENT TRAP -
9289	064364	104	111	106	EM363:	.ASCII	!IMPROPER D-SPACE ACCESS ATTEMPTED.
9290	064432	106	120	120	EM364:	.ASCII	!DIFFERENCE BETWEEN SR1 AND CALCULATED!
						.ASCII	!FPP INSTRUCTION FAILED TO ABORT, NOT

9291	064477	101	114	114		.ASCIZ	:ALLOWNG EXAMINATION OF SR1:
9292	064532	115	117	104	EM365:	.ASCIZ	:MODE 0 INSTRUCTION ABORTED WHEN IT SHOULD NOT HAVE:
9293	064615	106	120	120	EM366:	.ASCIZ	:FPP ACCUMULATOR WAS CHANGED IN THE EXPECTED ABORT.:
9294	064700	107	105	116	EM367:	.ASCIZ	:GENERAL REGISTER WAS CHANGED IN THE EXPECTED ABORT.:
9295	064763	106	120	120	EM370:	.ASCIZ	:FPP UNABLE TO RESTORE AN AC:
9296		000000			EM371=0		
9297		000000			EM372=0		
9298		000000			EM373=0		
9299		000000			EM374=0		
9300		000000			EM375=0		
9301		000000			EM376=0		
9302		000000			EM377=0		
9303		000000			EM400=0		
9304	065017	123	000		EM401:	.ASCIZ	:S:
9305	065021	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
						.WORD	EM401B,EM4B,0
9306	065022	065030	050045	000000	EM401B:	.ASCIZ	:TFPS (R):
9307	065030	124	106	120	EM402:	.ASCIZ	:S:
9308	065041	123	000			.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9309	065043	377				.EVEN	
						.WORD	EM401B,EM4D,0
9310	065044	065030	050066	000000	EM403:	.ASCIZ	:M:
9311	065052	115	000			.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9312	065054	377				.EVEN	
						.WORD	EM403B,EM403C,PERIOD
9313	065056	065132	065171	056225		.WORD	CRBUT,IGR7FL,PRST,N3,N5,N7,WENTTO,N4,N1,N6,INSTOF,N4,N1,N7,0
9314	065064	050167	065122	050176	IGR7FL:	.ASCIZ	:GR7,-FL:
9315	065122	107	122	067	EM403B:	.ASCIZ	:ORE THAN ONE WORD WRITTEN BY S:
9316	065132	117	122	105	EM403C:	.ASCIZ	:TFPS (R):
9317	065171	124	106	120	EM404:	.ASCIZ	:S:
9318	065202	123	000			.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9319	065204	377				.EVEN	
						.WORD	EM401B,EM227B,0
9320	065206	065030	056506	000000	EM405:	.ASCIZ	:S:
9321	065214	123	000			.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9322	065216	377				.EVEN	
						.WORD	EM403C,EM7B,EM4B,0
9323	065220	065171	050272	050045	EM406:	.ASCIZ	:S:
9324	065230	123	000			.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9325	065232	377				.EVEN	
						.WORD	EM403C,EM7B,EM4D,0
9326	065234	065171	050272	050066	EM407:	.ASCIZ	:M:
9327	065244	115	000			.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9328	065246	377				.EVEN	
						.WORD	EM403B,EM403C,EM7B
9329	065250	065132	065171	050272		.WORD	CRBUT,IGR7FL,PRST,N3,N5,N7,WENTTO,N4,N1,N6,INSTOF,N4,N1,N7,0
9330	065256	050167	065122	050176	EM410:	.ASCIZ	:S:
9331	065314	123	000			.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9332	065316	377				.EVEN	
						.WORD	EM403C,EM7B,EM227B,0
9333	065320	065171	050272	056506	EM411:	.ASCIZ	:S:
9334	065330	123	000			.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9335	065332	377				.EVEN	
						.WORD	EM411B,EM4B,0
9336	065334	065342	050045	000000	EM411B:	.ASCIZ	:TFPS -(R):
9337	065342	124	106	120	EM412:	.ASCIZ	:S:
9338	065354	123	000			.ASCIZ	:S:

ERROR MESSAGES

9339	065356	377				.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9340	065360	065342	050066	000000		.EVEN
9341	065366	115	000		EM413:	.WORD EM411B,EM4D,0
9342	065370	377				.ASCIZ 'M'
						.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN
9343	065372	065132	065342	056225		.WORD EM403B,EM411B,PERIOD
9344	065400	050167	065122	050176		.WORD CRBUT,IGR7FL,PRST,N3,N5,N7,WENTTO,N4,N1,N6,INSTOF,N4,N1,N7,0
9345	065436	123	000		EM414:	.ASCIZ 'S'
9346	065440	377				.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN
9347	065442	065342	056506	000000		.WORD EM411B,EM227B,0
9348	065450	123	000		EM415:	.ASCIZ 'S'
9349	065452	377				.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN
9350	065454	065462	050045	000000		.WORD EM415B,EM4B,0
9351	065462	124	106	120	EM415B:	.ASCIZ 'TFPS @(R)+'
9352	065475	123	000		EM416:	.ASCIZ 'S'
9353	065477	377				.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN
9354	065500	065462	050066	000000		.WORD EM415B,EM4D,0
9355	065506	123	000		EM417:	.ASCIZ 'S'
9356	065510	377				.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN
9357	065512	065462	065520	000000		.WORD EM415B,EM417B,0
9358	065520	040	104	111	EM417B:	.ASCIZ 'DID NOT DEFER THE WRITE.'
9359	065552	123	000		EM420:	.ASCIZ 'S'
9360	065554	377				.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN
9361	065556	065462	056506	000000		.WORD EM415B,EM227B,0
9362	065564	123	000		EM421:	.ASCIZ 'S'
9363	065566	377				.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN
9364	065570	065576	050045	000000		.WORD EM421B,EM4B,0
9365	065576	124	106	120	EM421B:	.ASCIZ 'TFPS @-(R)'
9366	065611	123	000		EM422:	.ASCIZ 'S'
9367	065613	377				.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN
9368	065614	065576	050066	000000		.WORD EM421B,EM4D,0
9369	065622	123	000		EM423:	.ASCIZ 'S'
9370	065624	377				.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN
9371	065626	065576	065520	000000		.WORD EM421B,EM417B,0
9372	065634	123	000		EM424:	.ASCIZ 'S'
9373	065636	377				.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN
9374	065640	065576	056506	000000		.WORD EM421B,EM227B,0
9375	065646	123	000		EM425:	.ASCIZ 'S'
9376	065650	377				.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN
9377	065652	065660	050045	000000		.WORD EM425B,EM4B,0
9378	065660	124	106	120	EM425B:	.ASCIZ 'TFPS N(R)'
9379	065672	123	000		EM426:	.ASCIZ 'S'
9380	065674	377				.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN
9381	065676	065660	050066	000000		.WORD EM425B,EM4D,0
9382	065704	115	000		EM427:	.ASCIZ 'M'

ERROR MESSAGES

9383	065706	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9384	065710	065132	065660	056225		.EVEN		
9385	065716	377				.WORD	EM403B,EM425B,PERIOD	
						.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
	065720	050167	065122	050176		.EVEN		
9386	065756	123	000		EM430:	.WORD	CRBUT,IGR7FL,PRST,N3,N5,N7,WENTTO,N4,N1,N6,INSTOF,N4,N1,N7,0	
9387	065760	377				.ASCIZ	!S!	
						.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9388	065762	065660	056506	000000		.WORD	EM425B,EM227B,0	
9389	065770	120	103	040	EM431:	.ASCIZ	!PC BAD AFTER S!	
9390	066007	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9391	066010	065660	054342	000000		.WORD	EM425B,EM160B,0	
9392	066016	123	000		EM432:	.ASCIZ	!S!	
9393	066020	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9394	066022	066030	050045	000000		.WORD	EM432B,EM4B,0	
9395	066030	124	106	120	EM432B:	.ASCIZ	!TFPS @N(R)!	
9396	066043	123	000		EM433:	.ASCIZ	!S!	
9397	066045	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9398	066046	066030	050066	000000		.WORD	EM432B,EM4D,0	
9399	066054	115	000		EM434:	.ASCIZ	!M!	
9400	066056	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9401	066060	065132	066030	056225		.WORD	EM403B,EM432B,PERIOD	
9402	066066	050167	065122	050176		.WORD	CRBUT,IGR7FL,PRST,N3,N5,N7,WENTTO,N4,N1,N6,INSTOF,N4,N1,N7,0	
9403	066124	123	000		EM435:	.ASCIZ	!S!	
9404	066126	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9405	066130	066030	056506	000000		.WORD	EM432B,EM227B,0	
9406	066136	120	103	040	EM436:	.ASCIZ	!PC BAD AFTER S!	
9407	066155	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9408	066156	066030	054342	000000		.WORD	EM432B,EM160B,0	
9409	066164	123	124	103	EM437:	.ASCIZ	!STCDL A,(R)+!	
9410	066201	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9411	066202	050045	000000			.WORD	EM4B,0	
9412	066206	123	124	103	EM440:	.ASCIZ	!STCDL A,-(R)!	
9413	066223	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9414	066224	050045	000000			.WORD	EM4B,0	
9415	066230	125	116	105	EM441:	.ASCIZ	!UNEXPECTED !	
9416	066244	376				.BYTE	376	;FLAGS 'ERTYPE' TO PRINT ERR # SPECIFIC ASCIZ MSGS, ADRS 3 LINES DOW
						.EVEN		
9417	066246	000441	000442	000443		.WORD	441,442,443,0	
9418	066256	066264	066333	066333		.WORD	EM441B,EM442B,EM442B	
9419	066264	106	120	120	EM441B:	.ASCIZ	!FPP !	
9420	066271	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9421	066272	066276	000000			.WORD	TRAPTO,0	
9422	066276	124	122	101	TRAPTO:	.ASCIZ	!TRAP TO !	
9423	066307	376				.BYTE	376	;FLAGS 'ERTYPE' TO PRINT ERR # SPECIFIC ASCIZ MSGS, ADRS 3 LINES DOW
						.EVEN		
9424	066310	000441	000442	000443		.WORD	441,442,443,0	

C
F

ERROR MESSAGES

9425	066320	066326	066346	066351		.WORD	N244,N04,N10
9426	066326	062	064	064	N244:	.ASCIZ	'244.'
9427		066230			EM442=EM441		
9428	066333	103	120	125	EM442B:	.ASCIZ	'CPU '
9429	066340	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9430	066342	066276	000000			.WORD	TRAPTO,0
9431	066346	064	056	000	N04:	.ASCIZ	'4.'
9432		066230			EM443=FM441		
9433	066351	061	060	056	N10:	.ASCIZ	'10.'
9434	066355	116	105	107	EM444:	.ASCIZ	'NEG'
9435	066361	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9436	066362	054662	050002	000000		.WORD	EM176B,EM3B,0

DATA TABLE HEADERS					DATA TABLE HEADERS	
9437					.SBTTL	DATA TABLE HEADERS
9438	066370	040	000		.ASCIZ	;
9439	066372	377		DH1:	.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
					.EVEN	
9440	066374	066404	066436	066445	.WORD	DH1B,DH1C,DH1D,0
9441	066404	040	124	105	DH1B:	.ASCIZ ; TEST.<TAB>!PC OF CALL.<TAB>'PC OF ;
9442	066436	105	122	122	DH1C:	.ASCIZ ;ERROR.;
9443	066445	011	106	120	DH1D:	.ASCIZ <TAB>!FPS.<TAB>!FEC.;
9444	066460	040	000		DH2:	.ASCIZ
9445	066462	377			.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
					.EVEN	
9446	066464	066404	066436	066474	.WORD	DH1B,DH1C,DH2B,0
9447	066474	011	107	117	DH2B:	.ASCIZ <TAB>!GOT FPS.<TAB>!EXPECTED FPS.;
9448	066524	040	000		DH3:	.ASCIZ
9449	066526	377			.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
					.EVEN	
9450	066530	066404	066436	066540	.WORD	DH1B,DH1C,DH3B,0
9451	066540	011	107	117	DH3B:	.ASCIZ <TAB>!GOT FEC.<TAB>!EXPECTED FEC.;
9452	066570	040	000		DH4:	.ASCIZ
9453	066572	377			.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
					.EVEN	
9454	066574	066404	066436	066604	.WORD	DH1B,DH1C,DH4B,0
9455	066604	011	107	117	DH4B:	.ASCIZ <TAB> GOT RO. <TAB>!EXPECTED RO.;
9456	066633	040	000		DH5:	.ASCIZ
9457	066635	377			.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
					.EVEN	
9458	066636	066404	066436	000000	.WORD	DH1B,DH1C,0
9459		066633			DH6=DH5	
9460		066570			DH7=DH4	
9461		066633			DH10=DH5	
9462		066570			DH11=DH4	
9463		066633			DH12=DH5	
9464	066644	040	040	124	DH13:	.ASCIZ ; TEST.<TAB>!PC OF CALL.<TAB>'PC OF TRAP.;
9465		066644			DH14=DH13	
9466		066633			DH15=DH5	
9467	066704	040	000		DH16:	.ASCIZ ; ;
9468	066706	377			.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
					.EVEN	
9469	066710	066404	066436	066720	.WORD	DH1B,DH1C,DH16B,0
9470	066720	011	107	117	DH16B:	.ASCIZ <TAB>'GOT PC. <TAB>!EXPECTED PC.;
9471		066644			DH17=DH13	
9472		066570			DH20=DH4	
9473		066633			DH21=DH5	
9474		066633			DH22=DH5	
9475		066644			DH23=DH13	
9476		066570			DH24=DH4	
9477		066633			DH25=DH5	
9478		066644			DH26=DH13	
9479		066570			DH27=DH4	
9480		066633			DH30=DH5	
9481		066644			DH31=DH13	
9482		066570			DH32=DH4	
9483		066633			DH33=DH5	
9484		066644			DH34=DH13	
9485		066570			DH35=DH4	
9486		066633			DH36=DH5	
9487	066747	040	000		DH37:	.ASCIZ ; ;

DATA TABLE HEADERS

9488	066751	377				.BYTE	377	:FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9489	066752	066404	066436	066762		.EVEN		
9490	066762	011	107	117	DH37B:	.WORD	DH1B,DH1C,DH37B,0	
9491		066747				.ASCIZ	<TAB> GOT FPS.!	<TAB>!EXPECTED FPS.!
9492	067012	040	000		DH40=DH37			
9493	067014	377			DH41:	.ASCIZ	! !	
						.BYTE	377	:FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9494	067016	066404	066436	067026		.WORD	DH1B,DH1C,DH41B,0	
9495	067026	011	106	120	DH41B:	.ASCIZ	<TAB>'FPS.!	<TAB>!GOT FEC. EXPECTED FEC.!
9496		066747			DH42=DH37			
9497		066747			DH43=DH37			
9498		066747			DH44=DH37			
9499		066747			DH45=DH37			
9500		066747			DH46=DH37			
9501		066747			DH47=DH37			
9502		066747			DH50=DH37			
9503		066747			DH51=DH37			
9504		066747			DH52=DH37			
9505		067012			DH53=DH41			
9506		066747			DH54=DH37			
9507		066747			DH55=DH37			
9508		066747			DH56=DH37			
9509		066747			DH57=DH37			
9510		066747			DH60=DH37			
9511		066747			DH61=DH37			
9512		066460			DH62=DH2			
9513		066524			DH63=DH3			
9514		066633			DH64=DH5			
9515		066460			DH65=DH2			
9516		066570			DH66=DH4			
9517		066460			DH67=DH2			
9518		066524			DH70=DH3			
9519		066460			DH176=DH2			
9520	067063	124	105	123	DH177:	.ASCIZ	.TESTNO	ERR PC CPUERR!
9521		066633			DH71=DH5			
9522		066460			DH72=DH2			
9523		066644			DH107=DH13			
9524		066633			DH73=DH5			
9525		066570			DH74=DH4			
9526		066460			DH75=DH2			
9527		066644			DH76=DH107			
9528		066633			DH77=DH5			
9529		066570			DH100=DH4			
9530		066460			DH101=DH2			
9531		066644			DH102=DH107			
9532		066633			DH103=DH5			
9533		066570			DH104=DH4			
9534		066460			DH105=DH2			
9535		066644			DH106=DH107			
9536		066633			DH110=DH5			
9537		066570			DH111=DH4			
9538		066460			DH112=DH2			
9539		066644			DH113=DH107			
9540		066633			DH114=DH5			
9541		066570			DH115=DH4			
9542		066460			DH116=DH2			

9543	066644	DH117=DH107
9544	066633	DH120=DH5
9545	066570	DH121=DH4
9546	066460	DH122=DH2
9547	066644	DH123=DH107
9548	066633	DH124=DH5
9549	066570	DH125=DH4
9550	066460	DH126=DH2
9551	066644	DH127=DH107
9552	066633	DH130=DH5
9553	066460	DH131=DH2
9554	066644	DH132=DH107
9555	066633	DH133=DH5
9556	066460	DH134=DH2
9557	066633	DH135=DH5
9558	066633	DH136=DH5
9559	066460	DH137=DH2
9560	066633	DH140=DH5
9561	066570	DH141=DH4
9562	066460	DH142=DH2
9563	066633	DH143=DH5
9564	066570	DH144=DH4
9565	066460	DH145=DH2
9566	066633	DH146=DH5
9567	066570	DH147=DH4
9568	066460	DH150=DH2
9569	066633	DH151=DH5
9570	066570	DH152=DH4
9571	066460	DH153=DH2
9572	066633	DH154=DH5
9573	066570	DH155=DH4
9574	066460	DH156=DH2
9575	066633	DH157=DH5
9576	066570	DH160=DH4
9577	066460	DH161=DH2
9578	066633	DH162=DH5
9579	066460	DH163=DH2
9580	066704	DH164=DH16
9581	066747	DH165=DH37
9582	066747	DH166=DH37
9583	066747	DH167=DH37
9584	066747	DH170=DH37
9585	066747	DH171=DH37
9586	066747	DH172=DH37
9587	067012	DH173=DH41
9588	067012	DH174=DH41
9589	067012	DH175=DH41
9590	066747	DH200=DH37
9591	066747	DH201=DH37
9592	066747	DH202=DH37
9593	066747	DH203=DH37
9594	066747	DH204=DH37
9595	066747	DH205=DH37
9596	066747	DH206=DH37
9597	066747	DH207=DH37
9598	066747	DH210=DH37
9599	066747	DH211=DH37

9600		066747				DH212=DH37
9601		066747				DH213=DH37
9602		066747				DH214=DH37
9603		066704				DH215=DH16
9604		066633				DH216=DH5
9605		066570				DH217=DH4
9606		066460				DH220=DH2
9607		066704				DH221=DH16
9608		066633				DH222=DH5
9609		066570				DH223=DH4
9610		066460				DH224=DH2
9611		066570				DH225=DH4
9612		066460				DH226=DH2
9613	067112	040	000			DH227: .ASCIZ ' ' ;
9614	067114	377				.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN
						.WORD DH1B,DH227B,0
9615	067116	066404	067124	000000		DH227B: .ASCIZ .TRAP.!
9616	067124	124	122	101		
9617		066570				DH230=DH4
9618		066460				DH231=DH2
9619		067112				DH232=DH227
9620		066570				DH233=DH4
9621		066460				DH234=DH2
9622		067112				DH235=DH227
9623		066570				DH236=DH4
9624		066460				DH237=DH2
9625		067112				DH240=DH227
9626		066570				DH241=DH4
9627		066460				DH242=DH2
9628		067112				DH243=DH227
9629		066570				DH244=DH4
9630		066460				DH245=DH2
9631		066704				DH246=DH16
9632		067112				DH247=DH227
9633		066570				DH250=DH4
9634		066460				DH251=DH2
9635		066704				DH252=DH16
9636		067112				DH253=DH227
9637		066704				DH254=DH16
9638		067112				DH255=DH227
9639		066570				DH256=DH4
9640		066460				DH257=DH2
9641		066747				DH260=DH37
9642		066747				DH261=DH37
9643		066747				DH262=DH37
9644		066747				DH263=DH37
9645		066747				DH264=DH37
9646		066747				DH265=DH37
9647		066747				DH266=DH37
9648		066747				DH267=DH37
9649		066747				DH270=DH37
9650		066747				DH271=DH37
9651		066747				DH272=DH37
9652		066747				DH273=DH37
9653		066747				DH274=DH37
9654		066747				DH275=DH37
9655		066747				DH276=DH37

9656	066747				DH277=DH37
9657	066747				DH300=DH37
9658	066747				DH301=DH37
9659	066747				DH302=DH37
9660	067012				DH303=DH41
9661	066747				DH304=DH37
9662	066747				DH305=DH37
9663	066747				DH306=DH37
9664	066747				DH307=DH37
9665	066747				DH310=DH37
9666	066747				DH311=DH37
9667	066747				DH312=DH37
9668	066747				DH313=DH37
9669	066747				DH314=DH37
9670	066747				DH315=DH37
9671	066747				DH316=DH37
9672	066747				DH317=DH37
9673	066747				DH320=DH37
9674	066747				DH321=DH37
9675	066747				DH322=DH37
9676	066747				DH323=DH37
9677	067012				DH324=DH41
9678	066747				DH325=DH37
9679	066747				DH326=DH37
9680	066747				DH327=DH37
9681	066747				DH330=DH37
9682	066747				DH331=DH37
9683	066747				DH332=DH37
9684	066747				DH333=DH37
9685	066747				DH334=DH37
9686	066747				DH335=DH37
9687	066747				DH336=DH37
9688	066747				DH337=DH37
9689	066747				DH340=DH37
9690	066747				DH341=DH37
9691	066747				DH342=DH37
9692	066747				DH343=DH37
9693	066747				DH344=DH37
9694	066747				DH345=DH37
9695	066747				DH346=DH37
9696	066747				DH347=DH37
9697	066747				DH350=DH37
9698	066644				DH351=DH13
9699	066747				DH352=DH37
9700	066747				DH353=DH37
9701	066747				DH354=DH37
9702	066747				DH355=DH37
9703	066570				DH356=DH11
9704	067132	040	000		DH357: .ASCIZ ; ;
9705	067134	377			.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
					.EVEN
9706	067136	066404	066436	067146	.WORD DH1B,DH1C,DH357B,0
9707	067146	011	107	117	DH357B: .ASCIZ <TAB> .GOT FEA. '<TAB>':EXPECTED FEA.
9708		066644			DH360=DH13
9709		066460			DH361=DH2
9710	067176	040	000		DH362: .ASCIZ ; ;
9711	067200	377			.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN

9763		066460				DH426=DH2
9764		066644				DH427=DH13
9765		067112				DH430=DH227
9766		066644				DH431=DH13
9767		066570				DH432=DH4
9768		066460				DH433=DH2
9769		066644				DH434=DH13
9770		067112				DH435=DH227
9771		066644				DH436=DH13
9772		066570				DH437=DH4
9773		066570				DH440=DH4
9774	067502	040		000		DH441: .ASCIZ
9775	067504	377				.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN
9776	067506	066404	066436	067516		.WORD DH1B,DH1C,DH441B,0
9777	067516	011	106	105		DH441B: .ASCIZ
9778		067327				<TAB>'FEC.:'
9779		067327				DH442=DH365
9780		066524				DH443=DH442
						DH444=DH3

9838		067533			DF176=DF2	
9839	067635	004	000	000	DF177: .BYTE	4,0,0
9840	067640	004	000	005	DF71: .BYTE	4,0,5,0,5,5,3,5,5,3,5,5,3
9841		067533			DF72=DF2	
9842		067555			DF107=DF6	
9843		067640			DF73=DF71	
9844		067533			DF74=DF2	
9845		067533			DF75=DF2	
9846		067555			DF76=DF6	
9847		067640			DF77=DF71	
9848		067533			DF100=DF2	
9849		067533			DF101=DF2	
9850		067555			DF102=DF6	
9851		067640			DF103=DF71	
9852		067533			DF104=DF2	
9853		067533			DF105=DF2	
9854		067555			DF106=DF6	
9855		067640			DF110=DF71	
9856		067533			DF111=DF2	
9857		067533			DF112=DF2	
9858		067555			DF113=DF6	
9859		067640			DF114=DF71	
9860		067533			DF115=DF2	
9861		067533			DF116=DF2	
9862		067555			DF117=DF6	
9863		067640			DF120=DF71	
9864		067533			DF121=DF2	
9865		067533			DF122=DF2	
9866		067555			DF123=DF6	
9867		067640			DF124=DF71	
9868		067533			DF125=DF2	
9869		067533			DF126=DF2	
9870		067555			DF127=DF6	
9871		067640			DF130=DF71	
9872		067533			DF131=DF2	
9873		067555			DF132=DF6	
9874		067640			DF133=DF71	
9875		067533			DF134=DF2	
9876		067561			DF135=DF12	
9877		067561			DF136=DF12	
9878		067533			DF137=DF2	
9879		067561			DF140=DF12	
9880		067533			DF141=DF2	
9881		067533			DF142=DF2	
9882		067561			DF143=DF12	
9883		067533			DF144=DF2	
9884		067533			DF145=DF2	
9885		067561			DF146=DF12	
9886		067533			DF147=DF2	
9887		067533			DF150=DF2	
9888		067561			DF151=DF12	
9889		067533			DF152=DF2	
9890		067533			DF153=DF2	
9891		067561			DF154=DF12	
9892		067533			DF155=DF2	
9893		067533			DF156=DF2	
9894		067561			DF157=DF12	

9895	067533			DF160=DF2	
9896	067533			DF161=DF2	
9897	067561			DF162=DF12	
9898	067533			DF163=DF2	
9899	067533			DF164=DF2	
9900	067533			DF215=DF2	
9901	067561			DF216=DF12	
9902	067533			DF217=DF2	
9903	067533			DF220=DF2	
9904	067533			DF221=DF2	
9905	067561			DF222=DF12	
9906	067533			DF223=DF2	
9907	067533			DF224=DF2	
9908	067573			DF165=DF37	
9909	067573			DF166=DF37	
9910	067573			DF167=DF37	
9911	067573			DF170=DF37	
9912	067573			DF171=DF37	
9913	067573			DF172=DF37	
9914	067614			DF173=DF41	
9915	067614			DF174=DF41	
9916	067614			DF175=DF41	
9917	067573			DF200=DF37	
9918	067573			DF201=DF37	
9919	067573			DF202=DF37	
9920	067573			DF203=DF37	
9921	067573			DF204=DF37	
9922	067573			DF205=DF37	
9923	067573			DF206=DF37	
9924	067573			DF207=DF37	
9925	067573			DF210=DF37	
9926	067573			DF211=DF37	
9927	067573			DF212=DF37	
9928	067573			DF213=DF37	
9929	067573			DF214=DF37	
9930	067655	004	000	005	DF225: .BYTE 4,0,5,0,5,0,5,0
9931	067655				DF226=DF225
9932	067665	004	000	005	DF227: .BYTE 4,0,5,0
9933	067655				DF230=DF225
9934	067655				DF231=DF225
9935	067665				DF232=DF227
9936	067655				DF233=DF225
9937	067655				DF234=DF225
9938	067665				DF235=DF227
9939	067655				DF236=DF225
9940	067655				DF237=DF225
9941	067665				DF240=DF227
9942	067655				DF241=DF225
9943	067655				DF242=DF225
9944	067665				DF243=DF227
9945	067655				DF244=DF225
9946	067655				DF245=DF225
9947	067655				DF246=DF225
9948	067665				DF247=DF227
9949	067655				DF250=DF225
9950	067655				DF251=DF225
9951	067655				DF252=DF225

9952	067665			DF253=DF227	
9953	067655			DF254=DF225	
9954	067665			DF255=DF227	
9955	067655			DF256=DF225	
9956	067655			DF257=DF225	
9957	067671	000	005	DF260: .BYTE	4,0,5,0,5,0,5,0,5,5,2,5,5,2,5,5,2
9958	067671			DF261=DF260	
9959	067671			DF262=DF260	
9960	067671			DF263=DF260	
9961	067671			DF264=DF260	
9962	067671			DF265=DF260	
9963	067671			DF266=DF260	
9964	067671			DF267=DF260	
9965	067671			DF270=DF260	
9966	067671			DF271=DF260	
9967	067671			DF272=DF260	
9968	067712	000	005	DF273: .BYTE	4,0,5,0,5,0,5,0,5,5,2,5,5,3,5,5,3
9969	067712			DF274=DF273	
9970	067712			DF275=DF273	
9971	067712			DF276=DF273	
9972	067712			DF277=DF273	
9973	067712			DF300=DF273	
9974	067733	000	005	DF301: .BYTE	4,0,5,0,5,0,5,0,5,5,3,5,5,0,5,5,3,5,5,3
9975	067733			DF302=DF301	
9976	067757	000	005	DF303: .BYTE	4,0,5,0,5,0,0,0,5,5,3,5,5,0,5,5,3,5,5,3
9977	067733			DF304=DF301	
9978	067733			DF305=DF301	
9979	067733			DF306=DF301	
9980	067733			DF307=DF301	
9981	067733			DF310=DF301	
9982	067733			DF311=DF301	
9983	067733			DF312=DF301	
9984	067733			DF313=DF301	
9985	067733			DF314=DF301	
9986	067733			DF315=DF301	
9987	067733			DF316=DF301	
9988	067733			DF317=DF301	
9989	067733			DF320=DF301	
9990	067733			DF321=DF301	
9991	070003	000	005	DF322: .BYTE	4,0,5,0,5,0,5,0,5,5,3,5,5,2,5,5,2
9992	070003			DF323=DF322	
9993	070024	000	005	DF324: .BYTE	4,0,5,0,5,0,0,0,5,5,3,5,5,2,5,5,2
9994	070003			DF325=DF322	
9995	070003			DF326=DF322	
9996	070003			DF327=DF322	
9997	070003			DF330=DF322	
9998	070003			DF331=DF322	
9999	070003			DF332=DF322	
10000	070003			DF333=DF322	
10001	070003			DF334=DF322	
10002	070003			DF335=DF322	
10003	070003			DF336=DF322	
10004	070003			DF337=DF322	
10005	070003			DF340=DF322	
10006	070003			DF341=DF322	
10007	070003			DF342=DF322	
10008	070003			DF343=DF322	

10066	067665			35=DF227	
10067	067665			436=DF227	
10068	067655			DF437=DF225	
10069	067655			DF440=DF225	
10070	070203	004	000	DF441: .BYTE	4,0,5,0,5,0
10071	070203			DF442=DF441	
10072	070203			DF443=DF441	
10073	067533			DF444=DF2	
10074				.EVEN	

10132		070276			DT66=DT4	
10133		070276			DT67=DT4	
10134		070276			DT70=DT4	
10135		070276			DT176=DT4	
10136	070512	001232	001116	044020	DT177: .WORD	\$TMP0,\$ERRPC,CPSAVE,0
10137	070522	001232	001234	047503	DT71: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,\$CRLF,MS3,\$TMP3,\$CRLF,MS1
10138	070544	001244	001313	047523	.WORD	\$TMP5,\$CRLF,MS2,\$TMP4,0
10139		070276			DT72=DT4	
10140		070346			DT107=DT6	
10141	070556	001232	001234	047503	DT73: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,\$CRLF,MS4,\$TMP4
10142	070574	001313	047505	001244	.WORD	\$CRLF,MS1,\$TMP5,\$CRLF,MS2,\$TMP3,0
10143		070276			DT74=DT4	
10144		070232			DT75=DT2	
10145		070346			DT76=DT6	
10146		070556			DT77=DT73	
10147		070276			DT100=DT4	
10148		070232			DT101=DT2	
10149		070346			DT102=DT6	
10150		070556			DT103=DT73	
10151		070276			DT104=DT4	
10152		070232			DT105=DT2	
10153		070346			DT106=DT6	
10154		070556			DT110=DT73	
10155		070276			DT111=DT4	
10156		070232			DT112=DT2	
10157		070346			DT113=DT6	
10158		070556			DT114=DT73	
10159		070276			DT115=DT4	
10160		070232			DT116=DT2	
10161		070346			DT117=DT6	
10162		070556			DT120=DT73	
10163		070276			DT121=DT4	
10164		070232			DT122=DT2	
10165		070346			DT123=DT6	
10166		070556			DT124=DT73	
10167		070276			DT125=DT4	
10168		070232			DT126=DT2	
10169		070346			DT127=DT6	
10170		070556			DT130=DT73	
10171		070232			DT131=DT2	
10172		070346			DT132=DT6	
10173		070556			DT133=DT73	
10174		070232			DT134=DT2	
10175		070320			DT135=DT5	
10176		070320			DT136=DT5	
10177		070360			DT137=DT16	
10178		070320			DT140=DT5	
10179		070276			DT141=DT4	
10180		070276			DT142=DT4	
10181		070320			DT143=DT5	
10182		070276			DT144=DT4	
10183		070276			DT145=DT4	
10184		070320			DT146=DT5	
10185		070276			DT147=DT4	
10186		070276			DT150=DT4	
10187		070320			DT151=DT5	
10188		070276			DT152=DT4	

10189	070276	DT153=DT4
10190	070320	DT154=DT5
10191	070276	DT155=DT4
10192	070276	DT156=DT4
10193	070320	DT157=DT5
10194	070276	DT160=DT4
10195	070276	DT161=DT4
10196	070320	DT162=DT5
10197	070276	DT163=DT4
10198	070276	DT164=DT4
10199	070276	DT215=DT4
10200	070320	DT216=DT5
10201	070276	DT217=DT4
10202	070276	DT220=DT4
10203	070276	DT221=DT4
10204	070320	DT222=DT5
10205	070276	DT223=DT4
10206	070276	DT224=DT4
10207	070402	DT165=DT37
10208	070402	DT166=DT37
10209	070402	DT167=DT37
10210	070402	DT170=DT37
10211	070402	DT171=DT37
10212	070402	DT172=DT37
10213	070446	DT173=DT41
10214	070446	DT174=DT41
10215	070446	DT175=DT41
10216	070402	DT200=DT37
10217	070402	DT201=DT37
10218	070402	DT202=DT37
10219	070402	DT203=DT37
10220	070402	DT204=DT37
10221	070402	DT205=DT37
10222	070402	DT206=DT37
10223	070402	DT207=DT37
10224	070402	DT210=DT37
10225	070402	DT211=DT37
10226	070402	DT212=DT37
10227	070402	DT213=DT37
10228	070402	DT214=DT37
10229	070276	DT225=DT4
10230	070276	DT226=DT4
10231	070346	DT227=DT6
10232	070276	DT230=DT4
10233	070276	DT231=DT4
10234	070346	DT232=DT6
10235	070276	DT233=DT4
10236	070276	DT234=DT4
10237	070346	DT235=DT6
10238	070276	DT236=DT4
10239	070276	DT237=DT4
10240	070346	DT240=DT6
10241	070276	DT241=DT4
10242	070276	DT242=DT4
10243	070346	DT243=DT6
10244	070276	DT244=DT4
10245	070276	DT245=DT4

10246		070276			DT246=DT4	
10247		070346			DT247=DT6	
10248		070276			DT250=DT4	
10249		070276			DT251=DT4	
10250		070276			DT252=DT4	
10251		070346			DT253=DT6	
10252		070276			DT254=DT4	
10253		070346			DT255=DT6	
10254		070276			DT256=DT4	
10255		070276			DT257=DT4	
10256		070402			DT260=DT37	
10257		070402			DT261=DT37	
10258		070402			DT262=DT37	
10259		070402			DT263=DT37	
10260		070402			DT264=DT37	
10261		070402			DT265=DT37	
10262		070402			DT266=DT37	
10263		070402			DT267=DT37	
10264		070402			DT270=DT37	
10265		070402			DT271=DT37	
10266		070402			DT272=DT37	
10267		070402			DT273=DT37	
10268		070402			DT274=DT37	
10269		070402			DT275=DT37	
10270		070402			DT276=DT37	
10271		070402			DT277=DT37	
10272		070402			DT300=DT37	
10273	070612	001232	001234	047503	DT301: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,\$TAB,\$TMP7,\$TAB,\$TMP10
10274	070632	001313	047545	001240	.WORD	\$CRLF,MS10,\$TMP3,\$CRLF,MS11,\$TMP4
10275	070646	001313	047505	001246	.WORD	\$CRLF,MS1,\$TMP6,\$CRLF,MS2,\$TMP5,0
10276		070612			DT302=DT301	
10277	070664	001232	001234	047503	DT303: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,\$TAB,\$TMP7,\$TMP11,\$TMP12
10278	070704	001313	047545	001240	.WORD	\$CRLF,MS10,\$TMP3,\$CRLF,MS11,\$TMP4
10279	070720	001313	047505	001246	.WORD	\$CRLF,MS1,\$TMP6,\$CRLF,MS2,\$TMP5,0
10280		070612			DT304=DT301	
10281		070612			DT305=DT301	
10282		070612			DT306=DT301	
10283		070612			DT307=DT301	
10284		070612			DT310=DT301	
10285		070612			DT311=DT301	
10286		070612			DT312=DT301	
10287		070612			DT313=DT301	
10288		070612			DT314=DT301	
10289		070612			DT315=DT301	
10290		070612			DT316=DT301	
10291		070612			DT317=DT301	
10292		070612			DT320=DT301	
10293		070612			DT321=DT301	
10294	070736	001232	001234	047503	DT322: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,\$TAB,\$TMP7,\$TAB,\$TMP10
10295	070756	001313	047545	001240	.WORD	\$CRLF,MS10,\$TMP3,\$CRLF,MS1,\$TMP4,\$CRLF,MS2,\$TMP5,0
10296		070736			DT323=DT322	
10297	071002	001232	001234	047503	DT324: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,\$TAB,\$TMP7,\$TMP11,\$TMP12
10298	071022	001313	047545	001240	.WORD	\$CRLF,MS10,\$TMP3,\$CRLF,MS1,\$TMP4,\$CRLF,MS2,\$TMP5,0
10299		070736			DT325=DT322	
10300		070736			DT326=DT322	
10301		070736			DT327=DT322	
10302		070736			DT330=DT322	

10303		070736			DT331=DT322	
10304		070736			DT332=DT322	
10305		070736			DT333=DT322	
10306		070736			DT334=DT322	
10307		070736			DT335=DT322	
10308		070736			DT336=DT322	
10309		070736			DT337=DT322	
10310		070736			DT340=DT322	
10311		070736			DT341=DT322	
10312		070736			DT342=DT322	
10313		070736			DT343=DT322	
10314		070736			DT344=DT322	
10315		070736			DT345=DT322	
10316		070736			DT346=DT322	
10317		070736			DT347=DT322	
10318		070736			DT350=DT322	
10319		070346			DT351=DT6	
10320		070736			DT352=DT322	
10321		070736			DT353=DT322	
10322		070736			DT354=DT322	
10323		070736			DT355=DT322	
10324		070232			DT356=DT2	
10325		070254			DT357=DT3	
10326		070346			DT360=DT6	
10327		070612			DT361=DT302	
10328	071046	001232	001234	047503	DT362: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,\$TAB,\$TMP3,0
10329	071064	001232	001234	047503	DT363: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP13,\$TAB,\$TMP2,\$TMP3,EXPCTD,0
10330	071106	001232	001234	047503	DT364: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP13,0
10331		070346			DT365=DT6	
10332	071120	001232	001234	047503	DT366: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP13,\$TAB,\$TMP2
10333	071134	001313	047523	001240	.WORD	\$CRLF,MS2,\$TMP3,\$TMP4,\$TMP6,\$TMP7
10334	071150	001313	047505	001252	.WORD	\$CRLF,MS1,\$TMP10,\$TMP11,\$TMP12,\$TMP21,0
10335	071166	001232	001234	047503	DT367: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,\$TMP13,\$TAB,\$TMP6,\$TAB,\$TMP3,0
10336	071212	001232	001234	047503	DT370: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,\$TMP13
10337	071224	001313	047642	001240	.WORD	\$CRLF,MS21,\$TMP3,\$TMP4,\$TMP6,\$TMP7
10338	071240	001313	047630	001252	.WORD	\$CRLF,MS20,\$TMP10,\$TMP11,\$TMP12,\$TMP21,0
10339		000000			DT371=0	
10340		000000			DT372=0	
10341		000000			DT373=0	
10342		000000			DT374=0	
10343		000000			DT375=0	
10344		000000			DT376=0	
10345		000000			DT377=0	
10346		000000			DT400=0	
10347		070276			DT401=DT4	
10348		070276			DT402=DT4	
10349		070346			DT403=DT6	
10350		070346			DT404=DT6	
10351		070276			DT405=DT4	
10352		070276			DT406=DT4	
10353		070346			DT407=DT6	
10354		070346			DT410=DT6	
10355		070276			DT411=DT4	
10356		070276			DT412 DT4	
10357		070346			DT413=DT6	
10358		070346			DT414=DT6	
10359		070276			DT415=DT4	

10360		070276				DT416=DT4	
10361		070346				DT417=DT6	
10362		070346				DT420=DT6	
10363		070276				DT421=DT4	
10364		070276				DT422=DT4	
10365		070346				DT423=DT6	
10366		070346				DT424=DT6	
10367		070276				DT425=DT4	
10368		070276				DT426=DT4	
10369		070346				DT427=DT6	
10370		070346				DT430=DT6	
10371		070346				DT431=DT6	
10372		070276				DT432=DT4	
10373		070276				DT433=DT4	
10374		070346				DT434=DT6	
10375		070346				DT435=DT6	
10376		070346				DT436=DT6	
10377		070276				DT437=DT4	
10378		070276				DT440=DT4	
10379	071256	001232	001234	047503		DT441: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,\$TAB,\$TMP3,0
10380	071274	001232	001234	047503		DT442: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,0
10381		071274				DT443=DT442	
10382		070276				DT444=DT4	
10383		000001				.END	

AABDF0 013612
AABDON 013642
AABTP1 013622
AABTP2 013632
AAB1 013434
AAB2 013536
AAB3 013556
AAB4 013574
AACDON 023766
AACTP1 023672
AAC1 023612
AAC10 023676
AAC11 023714
AAC2 023646
AAC20 023732
ABASE = 000000
ACDW1 = 000000
ACDW2 = 000000
ACPUOP = 000000
ACO = %000000
AC1 = %000001
AC2 = %000002
AC3 = %000003
AC4 = %000004
AC5 = %000005
AC6 = %000006
AC7 = %000007
ADDW0 = 000000
ADDW1 = 000000
ADDW10 = 000000
ADDW11 = 000000
ADDW12 = 000000
ADDW13 = 000000
ADDW14 = 000000
ADDW15 = 000000
ADDW2 = 000000
ADDW3 = 000000
ADDW4 = 000000
ADDW5 = 000000
ADDW6 = 000000
ADDW7 = 000000
ADDW8 = 000000
ADDW9 = 000000
ADEVCT = 000000
ADEVM = 000000
AENV = 000000
AENVM = 000000
AFATAL = 000000
AMADR1 = 000000
AMADR2 = 000000
AMADR3 = 000000
AMADR4 = 000000
AMAMS1 = 000000
AMAMS2 = 000000
AMAMS3 = 000000
AMAMS4 = 000000
AMSGAD = 000000

AMSGLG = 000000
AMSGTY = 000000
AMTYP1 = 000000
AMTYP2 = 000000
AMTYP3 = 000000
AMTYP4 = 000000
APASS = 000000
APRIOR = 000000
APTCSU = 000040
APTENV = 000001
APTSIZ = 000200
APTSP0 = 000100
ASWREG = 000000
ATESTN = 000000
AUNIT = 000000
AUSWR = 000000
AVECT1 = 000000
AVECT2 = 000000
BADCON 056200
BBCDON 024144
BBC1 023772
BBC10 024056
BBC11 024074
BBC2 024026
BBC20 024112
BIT0 = 000001
BIT00 = 000001
BIT01 = 000002
BIT02 = 000004
BIT03 = 000010
BIT04 = 000020
BIT05 = 000040
BIT06 = 000100
BIT07 = 000200
BIT08 = 000400
BIT09 = 001000
BIT1 = 000002
BIT10 = 002000
BIT11 = 004000
BIT12 = 010000
BIT13 = 020000
BIT14 = 040000
BIT15 = 100000
BIT2 = 000004
BIT3 = 000010
BIT4 = 000020
BIT5 = 000040
BIT6 = 000100
BIT7 = 000200
BIT8 = 000400
BIT9 = 001000
BPTVEC = 000014
BUTIN 056227
BYTABL 043074
CCBDON 013742
CCB1 013646

CCB10 013710
CCB15 013726
CCB2 013664
CKSWR = 104406
CNT = 000445
CPSAVE 044020
CPSPUR 047302
CPTWO 047330
CR = 000015
CRBUT 050167
CRLF = 000200
DATA = 117760
DDBBF0 014242
DDBDON 014262
DDBTP1 014222
DDBTP2 014232
DDBTP3 014252
DDB1 014050
DDB2 014114
DDB5 014156
DDB6 014204
DDCDON 024340
DDCTP1 024240
DDC1 024150
DDC10 024252
DDC11 024270
DDC2 024206
DDC20 024306
DDISP = 177570
DF1 067524
DF10 = 067543
DF100 = 067533
DF101 = 067533
DF102 = 067555
DF103 = 067640
DF104 = 067533
DF105 = 067533
DF106 = 067555
DF107 = 067555
DF11 = 067533
DF110 = 067640
DF111 = 067533
DF112 = 067533
DF113 = 067555
DF114 = 067640
DF115 = 067533
DF116 = 067533
DF117 = 067555
DF12 067561
DF120 = 067640
DF121 = 067533
DF122 = 067533
DF123 = 067555
DF124 = 067640
DF125 = 067533
DF126 = 067533
DF127 = 067555

DF13 = 067555
DF130 = 067640
DF131 = 067533
DF132 = 067555
DF133 = 067640
DF134 = 067533
DF135 = 067561
DF136 = 067561
DF137 = 067533
DF14 = 067555
DF140 = 067561
DF141 = 067533
DF142 = 067533
DF143 = 067561
DF144 = 067533
DF145 = 067533
DF146 = 067561
DF147 = 067533
DF15 = 067561
DF150 = 067533
DF151 = 067561
DF152 = 067533
DF153 = 067533
DF154 = 067561
DF155 = 067533
DF156 = 067533
DF157 = 067561
DF16 = 067533
DF160 = 067533
DF161 = 067533
DF162 = 067561
DF163 = 067533
DF164 = 067533
DF165 = 067573
DF166 = 067573
DF167 = 067573
DF17 = 067555
DF170 = 067573
DF171 = 067573
DF172 = 067573
DF173 = 067614
DF174 = 067614
DF175 = 067614
DF176 = 067533
DF177 067635
DF2 067533
DF20 = 067533
DF200 = 067573
DF201 = 067573
DF202 = 067573
DF203 = 067573
DF204 = 067573
DF205 = 067573
DF206 = 067573
DF207 = 067573
DF21 = 067561
DF210 = 067573

DF211 = 067573
DF212 = 067573
DF213 = 067573
DF214 = 067573
DF215 = 067533
DF216 = 067561
DF217 = 067533
DF22 = 067561
DF220 = 067533
DF221 = 067533
DF222 = 067561
DF223 = 067533
DF224 = 067533
DF225 = 067655
DF226 = 067655
DF227 = 067665
DF23 = 067555
DF230 = 067655
DF231 = 067655
DF232 = 067665
DF233 = 067655
DF234 = 067655
DF235 = 067665
DF236 = 067655
DF237 = 067655
DF24 = 067533
DF240 = 067665
DF241 = 067655
DF242 = 067655
DF243 = 067665
DF244 = 067655
DF245 = 067655
DF246 = 067655
DF247 = 067665
DF25 = 067561
DF250 = 067655
DF251 = 067655
DF252 = 067655
DF253 = 067665
DF254 = 067655
DF255 = 067665
DF256 = 067655
DF257 = 067655
DF26 = 067555
DF260 067671
DF261 = 067671
DF262 = 067671
DF263 = 067671
DF264 = 067671
DF265 = 067671
DF266 = 067671
DF267 = 067671
DF27 = 067533
DF270 = 067671
DF271 = 067671
DF272 = 067671
DF273 067712

DF274 = 067712	DF356 = 067655	DF44 = 067573	DH117 = 066644	DH28 = 066474
DF275 = 067712	DF357 = 067655	DF440 = 067655	DH12 = 066633	DH20 = 066570
DF276 = 067712	DF36 = 067561	DF441 = 070203	DH120 = 066633	DH200 = 066747
DF277 = 067712	DF360 = 067665	DF442 = 070203	DH121 = 066570	DH201 = 066747
DF3 = 067533	DF361 = 070066	DF443 = 070203	DH122 = 066460	DH202 = 066747
DF30 = 067561	DF362 = 070112	DF444 = 067533	DH123 = 066644	DH203 = 066747
DF300 = 067712	DF363 = 070120	DF45 = 067573	DH124 = 066633	DH204 = 066747
DF301 = 067733	DF364 = 067555	DF46 = 067573	DH125 = 066570	DH205 = 066747
DF302 = 067733	DF365 = 067555	DF47 = 067573	DH126 = 066460	DH206 = 066747
DF303 = 067757	DF366 = 070130	DF5 = 067543	DH127 = 066644	DH207 = 066747
DF304 = 067733	DF367 = 070152	DF50 = 067573	DH13 = 066644	DH21 = 066633
DF305 = 067733	DF37 = 067573	DF51 = 067573	DH130 = 066633	DH210 = 066747
DF306 = 067733	DF370 = 070162	DF52 = 067573	DH131 = 066460	DH211 = 066747
DF307 = 067733	DF371 = 000000	DF53 = 067573	DH132 = 066644	DH212 = 066747
DF31 = 067555	DF372 = 000000	DF54 = 067573	DH133 = 066633	DH213 = 066747
DF310 = 067733	DF373 = 000000	DF55 = 067573	DH134 = 066460	DH214 = 066747
DF311 = 067733	DF374 = 000000	DF56 = 067573	DH135 = 066633	DH215 = 066704
DF312 = 067733	DF375 = 000000	DF57 = 067573	DH136 = 066633	DH216 = 066633
DF313 = 067733	DF376 = 000000	DF6 = 067555	DH137 = 066460	DH217 = 066570
DF314 = 067733	DF377 = 000000	DF60 = 067573	DH14 = 066644	DH22 = 066633
DF315 = 067733	DF4 = 067533	DF61 = 067573	DH140 = 066633	DH220 = 066460
DF316 = 067733	DF40 = 067573	DF62 = 067533	DH141 = 066570	DH221 = 066704
DF317 = 067733	DF400 = 000000	DF63 = 067533	DH142 = 066460	DH222 = 066633
DF32 = 067533	DF401 = 067655	DF64 = 067543	DH143 = 066633	DH223 = 066570
DF320 = 067733	DF402 = 067655	DF65 = 067533	DH144 = 066570	DH224 = 066460
DF321 = 067733	DF403 = 067665	DF66 = 067533	DH145 = 066460	DH225 = 066570
DF322 = 070003	DF404 = 067665	DF67 = 067533	DH146 = 066633	DH226 = 066460
DF323 = 070003	DF405 = 067655	DF7 = 067533	DH147 = 066570	DH227 = 067112
DF324 = 070024	DF406 = 067655	DF70 = 067533	DH15 = 066633	DH227B = 067124
DF325 = 070003	DF407 = 067665	DF71 = 067640	DH150 = 066460	DH23 = 066644
DF326 = 070003	DF41 = 067614	DF72 = 067533	DH151 = 066633	DH230 = 066570
DF327 = 070003	DF410 = 067665	DF73 = 067640	DH152 = 066570	DH231 = 066460
DF33 = 067561	DF411 = 067655	DF74 = 067533	DH153 = 066460	DH232 = 067112
DF330 = 070003	DF412 = 067655	DF75 = 067533	DH154 = 066633	DH233 = 066570
DF331 = 070003	DF413 = 067665	DF76 = 067555	DH155 = 066570	DH234 = 066460
DF332 = 070003	DF414 = 067665	DF77 = 067640	DH156 = 066460	DH235 = 067112
DF333 = 070003	DF415 = 067655	DH1 = 066370	DH157 = 066633	DH236 = 066570
DF334 = 070003	DF416 = 067655	DH1B = 066404	DH16 = 066704	DH237 = 066460
DF335 = 070003	DF417 = 067665	DH1C = 066436	DH16B = 066720	DH24 = 066570
DF336 = 070003	DF42 = 067573	DH1D = 066445	DH160 = 066570	DH240 = 067112
DF337 = 070003	DF420 = 067665	DH1E = 066633	DH161 = 066460	DH241 = 066570
DF34 = 067555	DF421 = 067655	DH1F = 066570	DH162 = 066633	DH242 = 066460
DF340 = 070003	DF422 = 067655	DH1G = 066460	DH163 = 066460	DH243 = 067112
DF341 = 070003	DF423 = 067665	DH1H = 066644	DH164 = 066704	DH244 = 066570
DF342 = 070003	DF424 = 067665	DH1I = 066633	DH165 = 066747	DH245 = 066460
DF343 = 070003	DF425 = 067655	DH1J = 066570	DH166 = 066747	DH246 = 066704
DF344 = 070003	DF426 = 067655	DH1K = 066460	DH167 = 066747	DH247 = 067112
DF345 = 070003	DF427 = 067665	DH1L = 066644	DH17 = 066644	DH25 = 066633
DF346 = 070003	DF43 = 067573	DH1M = 066644	DH170 = 066747	DH250 = 066570
DF347 = 070045	DF430 = 067665	DH1N = 066570	DH171 = 066747	DH251 = 066460
DF35 = 067533	DF431 = 067665	DH1O = 066633	DH172 = 066747	DH252 = 066704
DF350 = 070045	DF432 = 067655	DH1P = 066570	DH173 = 067012	DH253 = 067112
DF351 = 067665	DF433 = 067655	DH1Q = 066460	DH174 = 067012	DH254 = 066704
DF352 = 070045	DF434 = 067665	DH1R = 066644	DH175 = 067012	DH255 = 067112
DF353 = 070045	DF435 = 067665	DH1S = 066633	DH176 = 066460	DH256 = 066570
DF354 = 070045	DF436 = 067665	DH1T = 066570	DH177 = 067063	DH257 = 066460
DF355 = 070045	DF437 = 067655	DH1U = 066460	DH2 = 066460	DH26 = 066644

SYMBOL TABLE

DH260 = 066747	DH341 = 066747	DH413 = 066644	DH75 = 066460	DT153 = 070276
DH261 = 066747	DH342 = 066747	DH414 = 067112	DH76 = 066644	DT154 = 070320
DH262 = 066747	DH343 = 066747	DH415 = 066570	DH77 = 066633	DT155 = 070276
DH263 = 066747	DH344 = 066747	DH416 = 066460	DIDONE 043200	DT156 = 070276
DH264 = 066747	DH345 = 066747	DH417 = 066644	DISPLA 001142	DT157 = 070320
DH265 = 066747	DH346 = 066747	DH42 = 066747	DISPRE 000174	DT16 = 070360
DH266 = 066747	DH347 = 066747	DH420 = 067112	DSWR = 177570	DT160 = 070276
DH267 = 066747	DH35 = 066570	DH421 = 066570	DT1 = 070212	DT161 = 070276
DH27 = 066570	DH350 = 066747	DH422 = 066460	DT10 = 070320	DT162 = 070320
DH270 = 066747	DH351 = 066644	DH423 = 066644	DT100 = 070276	DT163 = 070276
DH271 = 066747	DH352 = 066747	DH424 = 067112	DT101 = 070232	DT164 = 070276
DH272 = 066747	DH353 = 066747	DH425 = 066570	DT102 = 070346	DT165 = 070402
DH273 = 066747	DH354 = 066747	DH426 = 066460	DT103 = 070556	DT166 = 070402
DH274 = 066747	DH355 = 066747	DH427 = 066644	DT104 = 070276	DT167 = 070402
DH275 = 066747	DH356 = 066570	DH43 = 066747	DT105 = 070232	DT17 = 070346
DH276 = 066747	DH357 067132	DH430 = 067112	DT106 = 070346	DT170 = 070402
DH277 = 066747	DH357B 067146	DH431 = 066644	DT107 = 070346	DT171 = 070402
DH3 = 066524	DH36 = 066633	DH432 = 066570	DT11 = 070276	DT172 = 070402
DH3B = 066540	DH360 = 066644	DH433 = 066460	DT110 = 070556	DT173 = 070446
DH30 = 066633	DH361 = 066460	DH434 = 066644	DT111 = 070276	DT174 = 070446
DH300 = 066747	DH362 067176	DH435 = 067112	DT112 = 070232	DT175 = 070446
DH301 = 066747	DH362B 067212	DH436 = 066644	DT113 = 070346	DT176 = 070276
DH302 = 066747	DH363 067220	DH437 = 066570	DT114 = 070556	DT177 = 070512
DH303 = 067012	DH363B 067234	DH44 = 066747	DT115 = 070276	DT2 = 070232
DH304 = 066747	DH364 067260	DH440 = 066570	DT116 = 070232	DT20 = 070360
DH305 = 066747	DH364B 067272	DH441 067502	DT117 = 070346	DT200 = 070402
DH306 = 066747	DH365 067327	DH441B 067516	DT12 = 070320	DT201 = 070402
DH307 = 066747	DH366 067340	DH442 = 067327	DT120 = 070556	DT202 = 070402
DH31 = 066644	DH366B 067354	DH443 = 067327	DT121 = 070276	DT203 = 070402
DH310 = 066747	DH367 067372	DH444 = 066524	DT122 = 070232	DT204 = 070402
DH311 = 066747	DH367B 067406	DH45 = 066747	DT123 = 070346	DT205 = 070402
DH312 = 066747	DH37 066747	DH46 = 066747	DT124 = 070556	DT206 = 070402
DH313 = 066747	DH37B 066762	DH47 = 066747	DT125 = 070276	DT207 = 070402
DH314 = 066747	DH370 067437	DH5 = 066633	DT126 = 070232	DT21 = 070320
DH315 = 066747	DH371 = 000000	DH50 = 066747	DT127 = 070346	DT210 = 070402
DH316 = 066747	DH372 = 000000	DH51 = 066747	DT13 = 070346	DT211 = 070402
DH317 = 066747	DH373 = 000000	DH52 = 066747	DT130 = 070556	DT212 = 070402
DH32 = 066570	DH374 = 000000	DH53 = 067012	DT131 = 070232	DT213 = 070402
DH320 = 066747	DH375 = 000000	DH54 = 066747	DT132 = 070346	DT214 = 070402
DH321 = 066747	DH376 = 000000	DH55 = 066747	DT133 = 070556	DT215 = 070276
DH322 = 066747	DH377 = 000000	DH56 = 066747	DT134 = 070232	DT216 = 070320
DH323 = 066747	DH4 066570	DH57 = 066747	DT135 = 070320	DT217 = 070276
DH324 = 067012	DH4B C66604	DH6 = 066633	DT136 = 070320	DT22 = 070320
DH325 = 066747	DH40 = 066747	DH60 = 066747	DT137 = 070360	DT220 = 070276
DH326 = 066747	DH400 = 000000	DH61 = 066747	DT14 = 070346	DT221 = 070276
DH327 = 066747	DH401 = 066570	DH62 = 066460	DT140 = 070320	DT222 = 070320
DH33 = 066633	DH402 = 066460	DH63 = 066524	DT141 = 070276	DT223 = 070276
DH330 = 066747	DH403 = 066644	DH64 = 066633	DT142 = 070276	DT224 = 070276
DH331 = 066747	DH404 = 067112	DH65 = 066460	DT143 = 070320	DT225 = 070276
DH332 = 066747	DH405 = 066570	DH66 = 066570	DT144 = 070276	DT226 = 070276
DH333 = 066747	DH406 = 066460	DH67 = 066460	DT145 = 070276	DT227 = 070346
DH334 = 066747	DH407 = 066644	DH7 = 066570	DT146 = 070320	DT23 = 070346
DH335 = 066747	DH41 067012	DH70 = 066524	DT147 = 070276	DT230 = 070276
DH336 = 066747	DH41B 067026	DH71 = 066633	DT15 = 070320	DT231 = 070276
DH337 = 066747	DH410 = 067112	DH72 = 066460	DT150 = 070276	DT232 = 070346
DH34 = 066644	DH411 = 066570	DH73 = 066633	DT151 = 070320	DT233 = 070276
DH340 = 066747	DH412 = 066460	DH74 = 066570	DT152 = 070276	DT234 = 070276

DT235 = 070346	DT317 = 070612	DT400 = 000000	DT63 = 070360	EM115 = 053314
DT236 = 070276	DT32 = 070360	DT401 = 070276	DT64 = 070320	EM116 = 053330
DT237 = 070276	DT320 = 070612	DT402 = 070276	DT65 = 070360	EM117 = 053344
DT24 = 070360	DT321 = 070612	DT403 = 070346	DT66 = 070276	EM117B = 053360
DT240 = 070346	DT322 = 070736	DT404 = 070346	DT67 = 070276	EM12 = 050340
DT241 = 070276	DT323 = 070736	DT405 = 070276	DT7 = 070276	EM120 = 053367
DT242 = 070276	DT324 = 071002	DT406 = 070276	DT70 = 070276	EM121 = 053402
DT243 = 070346	DT325 = 070736	DT407 = 070346	DT71 = 070522	EM122 = 053416
DT244 = 070276	DT326 = 070736	DT41 = 070446	DT72 = 070276	EM123 = 053432
DT245 = 070276	DT327 = 070736	DT410 = 070346	DT73 = 070556	EM123B = 053446
DT246 = 070276	DT33 = 070320	DT411 = 070276	DT74 = 070276	EM124 = 053456
DT247 = 070346	DT330 = 070736	DT412 = 070276	DT75 = 070232	EM125 = 053472
DT25 = 070320	DT331 = 070736	DT413 = 070346	DT76 = 070346	EM126 = 053506
DT250 = 070276	DT332 = 070736	DT414 = 070346	DT77 = 070556	EM127 = 053522
DT251 = 070276	DT333 = 070736	DT415 = 070276	EEBBF0 = 014416	EM127B = 053536
DT252 = 070276	DT334 = 070736	DT416 = 070276	EEBBF1 = 014426	EM13 = 050352
DT253 = 070346	DT335 = 070736	DT417 = 070346	EEBDON = 014552	EM13A = 050364
DT254 = 070276	DT336 = 070736	DT42 = 070402	EEBTP1 = 014376	EM13B = 050374
DT255 = 070346	DT337 = 070736	DT420 = 070346	EEBTP2 = 014406	EM130 = 053546
DT256 = 070276	DT34 = 070346	DT421 = 070276	EEB1 = 014266	EM131 = 053562
DT257 = 070276	DT340 = 070736	DT422 = 070276	EEB10 = 014436	EM132 = 053576
DT26 = 070346	DT341 = 070736	DT423 = 070346	EEB15 = 014472	EM132B = 053612
DT260 = 070402	DT342 = 070736	DT424 = 070346	EEB2 = 014336	EM133 = 053623
DT261 = 070402	DT343 = 070736	DT425 = 070276	EEB20 = 014520	EM134 = 053636
DT262 = 070402	DT344 = 070736	DT426 = 070276	EEB25 = 014536	EM135 = 053652
DT263 = 070402	DT345 = 070736	DT427 = 070346	EECDON = 024550	EM136 = 053722
DT264 = 070402	DT346 = 070736	DT43 = 070402	EECTP1 = 024440	EM137 = 053736
DT265 = 070402	DT347 = 070736	DT430 = 070346	EECTP2 = 024450	EM14 = 050352
DT266 = 070402	DT35 = 070360	DT431 = 070346	EEC1 = 024344	EM140 = 053752
DT267 = 070402	DT350 = 070736	DT432 = 070276	EEC10 = 024462	EM141 = 053766
DT27 = 070360	DT351 = 070346	DT433 = 070276	EEC11 = 024500	EM141C = 054006
DT270 = 070402	DT352 = 070736	DT434 = 070346	EEC2 = 024406	EM142 = 054024
DT271 = 070402	DT353 = 070736	DT435 = 070346	EEC20 = 024516	EM143 = 054040
DT272 = 070402	DT354 = 070736	DT436 = 070346	EMTVEC = 000030	EM144 = 054054
DT273 = 070402	DT355 = 070736	DT437 = 070276	EM1 = 047662	EM145 = 054072
DT274 = 070402	DT356 = 070232	DT44 = 070402	EM1B = 047676	EM146 = 054106
DT275 = 070402	DT357 = 070254	DT440 = 070276	EM1C = 047707	EM147 = 054122
DT276 = 070402	DT36 = 070320	DT441 = 071256	EM1D = 047726	EM15 = 050420
DT277 = 070402	DT360 = 070346	DT442 = 071274	EM10 = 050275	EM150 = 054142
DT3 = 070254	DT361 = 070612	DT443 = 071274	EM100 = 053032	EM151 = 054156
DT30 = 070320	DT362 = 071046	DT444 = 070276	EM101 = 053046	EM152 = 054172
DT300 = 070402	DT363 = 071064	DT45 = 070402	EM102 = 053062	EM153 = 054212
DT301 = 070612	DT364 = 071106	DT46 = 070402	EM102B = 053076	EM154 = 054226
DT302 = 070612	DT365 = 070346	DT47 = 070402	EM103 = 053105	EM155 = 054242
DT303 = 070664	DT366 = 071120	DT5 = 070320	EM104 = 053120	EM156 = 054262
DT304 = 070612	DT367 = 071166	DT50 = 070402	EM105 = 053134	EM157 = 054276
DT305 = 070612	DT37 = 070402	DT51 = 070402	EM106 = 053150	EM157B = 054312
DT306 = 070612	DT370 = 071212	DT52 = 070402	EM106B = 053164	EM16 = 050432
DT307 = 070612	DT371 = 000000	DT53 = 070446	EM107 = 053174	EM160 = 054321
DT31 = 070346	DT372 = 000000	DT54 = 070402	EM11 = 050310	EM160B = 054342
DT310 = 070612	DT373 = 000000	DT55 = 070402	EM11B = 050326	EM161 = 054367
DT311 = 070612	DT374 = 000000	DT56 = 070402	EM110 = 053210	EM162 = 054402
DT312 = 070612	DT375 = 000000	DT57 = 070402	EM111 = 053224	EM162B = 054416
DT313 = 070612	DT376 = 000000	DT6 = 070346	EM112 = 053240	EM163 = 054426
DT314 = 070612	DT377 = 000000	DT60 = 070402	EM113 = 053254	EM164 = 054442
DT315 = 070612	DT4 = 070276	DT61 = 070402	EM113B = 053270	EM165 = 054472
DT316 = 070612	DT40 = 070402	DT62 = 070360	EM114 = 053300	EM166 = 054506

EM167	054522	EM235	056632	EM276	060247	EM341	062736	EM411B	065342
EM17	050456	EM236	056644	EM276C	060262	EM342	063006	EM412	065354
EM17B	050470	EM236B	056656	EM277	060334	EM343	063056	EM413	065366
EM170	054536	EM237	056671	EM277C	060350	EM344	063133	EM414	065436
EM171	054552	EM24	050557	EM3	047766	EM345	063240	EM415	065450
EM172	054566	EM240	056702	EM3B	050002	EM346	063310	EM415B	065462
EM173	054602	EM241	056714	EM30	050650	EM346C	063334	EM416	065475
EM174	054616	EM241B	056726	EM300	060401	EM347	063407	EM417	065506
EM175	054632	EM242	056741	EM300C	060424	EM347B	063420	EM417B	065520
EM176	054646	EM243	056752	EM301	060476	EM35	050763	EM42 =	051012
EM176B	054662	EM244	056764	EM301B	060510	EM350	063433	EM42B	051132
EM177	054670	EM244B	056776	EM302	060523	EM351	063444	EM420	065552
EM2	047736	EM245	057010	EM303	060534	EM352	063546	EM421	065564
EM2B	047752	EM246	057022	EM304	060546	EM353	063616	EM421B	065576
EM20	050502	EM246B	057060	EM304C	060562	EM354	063666	EM422	065611
EM200	054724	EM247	057073	EM305	060626	EM355	063736	EM423	065622
EM200C	054742	EM25	050574	EM305C	060642	EM356	064006	EM424	065634
EM201	054777	EM250	057104	EM306	060722	EM356B	064042	EM425	065646
EM202	055050	EM250B	057116	EM307	061030	EM357	064112	EM425B	065660
EM203	055122	EM251	057131	EM31	050662	EM36	051000	EM426	065672
EM204	055244	EM252 =	057022	EM31B	050674	EM360	064146	EM427	065704
EM204C	055262	EM252B	057142	EM310	061100	EM361	064236	EM43 =	051072
EM205	055317	EM253	057156	EM311	061154	EM362	064254	EM43B	051166
EM206	055370	EM254 =	057022	EM312	061224	EM363	064364	EM430	065756
EM207	055442	EM254B	057170	EM313	061274	EM364	064432	EM431	065770
EM21	050520	EM255	057206	EM314	061344	EM365	064532	EM432	066016
EM210	055514	EM256	057232	EM315	061414	EM366	064615	EM432B	066030
EM211	055566	EM256B	057244	EM316	061464	EM367	064700	EM433	066043
EM212	055645	EM257	057260	EM317	061534	EM37	051012	EM434	066054
EM213	055716	EM26	050606	EM32	050706	EM370	064763	EM435	066124
EM213C	055774	EM26B	050620	EM320	061604	EM371 =	000000	EM436	066136
EM213D	056004	EM260	057272	EM321	061654	EM372 =	000000	EM437	066164
EM214	056027	EM260B	057304	EM322	061724	EM373 =	000000	EM44 =	051012
EM214B	056104	EM261	057330	EM322B	061736	EM374 =	000000	EM44B	051245
EM215	056114	EM262	057342	EM323	061762	EM375 =	000000	EM440	066206
EM216	056251	EM262B	057415	EM324	061774	EM376 =	000000	EM441	066230
EM217	056264	EM263	057430	EM325	062006	EM377 =	000000	EM441B	066264
EM22 =	050520	EM263B	057442	EM325B	062022	EM4	050016	EM442 =	066230
EM220	056304	EM264	057455	EM325C	062035	EM4A	050034	EM442B	066333
EM221	056320	EM264C	057470	EM326 =	062006	EM4B	050045	EM443 =	066230
EM222	056370	EM265	057542	EM327	062132	EM4C	050057	EM444	066355
EM223	056404	EM265C	057556	EM33	050724	EM4D	050066	EM45 =	051012
EM224	056424	EM266	057613	EM330	062202	EM40	051072	EM45B	051322
EM225	056440	EM267	057667	EM331	062256	EM400 =	000000	EM46 =	051072
EM225B	056452	EM267C	057702	EM332	062326	EM401	065017	EM46B	051355
EM226	056463	EM27	050633	EM332B	062342	EM401B	065030	EM47 =	051012
EM227	056474	EM270	057754	EM332C	062355	EM402	065041	EM47B	051430
EM227B	056506	EM270L	057770	EM333 =	061762	EM403	065052	EM5	050077
EM23	050532	EM271	060006	EM334	062424	EM403B	065132	EM50 =	051012
EM23B	050544	EM271C	060022	EM334C	062440	EM403C	065171	EM50B	051472
EM230	056525	EM272	060045	EM335	062507	EM404	065202	EM51	051551
EM230B	056536	EM272C	060060	EM336	062556	EM405	065214	EM52	051610
EM231	056550	EM273	060115	EM336C	062574	EM406	065230	EM53 =	051610
EM232	056562	EM273B	060126	EM337	062613	EM407	065244	EM54 =	051551
EM233	056574	EM274	060152	EM34	050736	EM41 =	051072	EM54B	051652
EM233B	056606	EM275	060164	EM34B	050750	EM410	065314	EM55 =	051551
EM234	056620	EM275B	060234	EM340	062667	EM411	065330	EM55B	051716

SYMBOL TABLE

EM56 = 051610	FFCDON 024752	IFD 050164	KIPAR3= 172346	LLB25 017156
EM56B 052007	FFCTP1 024644	IFDST 052376	KIPAR4= 172350	LLCDON 030254
EM57 = 051610	FFCTP2 024654	IFIC 062252	KIPAR7= 172356	LLC1 027252
EM57B 052062	FFC1 024554	IFIU 061150	KIPDR0= 172300	LLC10 030000
EM6 050110	FFC10 024664	IFIUV 056022	KIPDR1= 172302	LLC2 027320
EM6C 050124	FFC11 024702	IFIV 052142	KIPDR2= 172304	LLC3 027366
EM60 = 051551	FFC2 024620	IFL 057412	KIPDR3= 172306	LLC4 027434
EM60B 052146	FFC20 024720	IFLAG 052300	KIPDR4= 172310	LLC5 027502
EM61 = 051610	FPSPUR 047240	IGR7FL 065122	KIPDR7= 172316	LLC6 027550
EM61B 052226	FPVECT= 000244	IIBBF0 015776	KKBBF0 016574	LLC7 027616
EM62 052305	GGBBF0 015176	IIBBF1 016006	KKBBF1 016604	LLC8 027664
EM62A 052324	GGBBF1 015206	IIBDON 016132	KKBDON 016730	LLC9 027732
EM62B 052334	GGBDON 015332	IIBTP1 015746	KKBTP1 016544	LOOP 006600
EM63 052403	GGBTP1 015156	IIBTP2 015766	KKBTP2 016564	LPERR = 104413
EM64 052420	GGBTP2 015166	IIB1 015636	KKB1 016432	MMBBF0 017326
EM65 052470	GGB1 015046	IIB10 016016	KKB10 016614	MMBBF1 017336
EM66 052514	GGB10 015216	IIB15 016052	KKB15 016650	MMBDON 017444
EM66B 052534	GGB15 015252	IIB2 015706	KKB2 016502	MMBTP1 017276
EM67 052542	GGB2 015116	IIB20 016100	KKB20 016676	MMBTP2 017316
EM7 050253	GGB20 015300	IIB25 016116	KKB25 016714	MMB1 017176
EM7B 050272	GGB25 015316	IICDON 025536	KKCDON 027246	MMB10 017346
EM70 052624	GGCDON 025170	IIC1 025426	KKC1 025700	MMB15 017402
EM71 052647	GGCTP1 025050	IIC2 025454	KKC10 026316	MMB2 017242
EM72 052664	GGC1 024756	IIC20 025526	KKC11 026354	MMB25 017430
EM73 052700	GGC10 025062	IIC3 025474	KKC12 026412	MMCDON 032122
EM74 052714	GGC11 025100	INBIT 063126	KKC13 026450	MMC1 030260
EM75 052730	GGC2 025016	INSTOF 050216	KKC14 026506	MMC10 031116
EM76 052744	GGC20 025136	IOP1B 055237	KKC15 026544	MMC11 031174
EM76A 052760	GGC25 025116	IOTRAP= 000020	KKC16 026602	MMC12 031252
EM76B 052767	GTSWR = 104405	IOTVEC= 000020	KKC17 026640	MMC13 031330
EM77 053017	HHBBF0 015476	IXNBT 057662	KKC18 026676	MMC14 031406
ENDTES 043160	HHBBF1 015506	JJBBF0 016272	KKC19 026734	MMC15 031464
ENDTST 040072	HHBDON 015632	JJBBF1 016302	KKC2 025736	MMC16 031542
EQASCI 044504	HHBTP1 015446	JJBDON 016426	KKC20 026772	MMC2 030336
ERM10 044406	HHBTP2 015466	JJBTP1 016250	KKC3 025774	MMC3 030414
ERRNUM 047104	HHB1 015336	JJBTP2 016262	KKC4 026032	MMC4 030472
ERROR = 104000	HHB10 015516	JJB1 016136	KKC5 026070	MMC5 030550
ERRVEC= 000004	HHB15 015552	JJB10 016312	KKC6 026126	MMC6 030626
ERTYPE 046460	HHB2 015406	JJB15 016346	KKC7 026164	MMC7 030704
ERT1 046654	HHB20 015600	JJB2 016206	KKC8 026222	MMC8 030762
ERT2 047054	HHB25 015616	JJB20 016374	KKC9 026260	MMC9 031040
ERT3 047060	HHCDON 025422	JJB25 016412	LABEL1 040712	MMR0 = 177572
ERT4 047066	HHCTP1 025274	KDPAR0= 172360	LDCDSU 030050	MMR2 = 177576
ERT5 047100	HHCTP2 025304	KDPAR1= 172362	LDCFSU 027032	MMR3 = 172516
EXPCTD 043156	HHC1 025174	KDPAR2= 172364	LDCT 027236	MMVECT= 000250
FALTRP 041570	HHC10 025314	KDPAR3= 172366	LDXSUB 031622	MNUMBE= 000444
FFBBF0 014706	HHC11 025332	KDPAR4= 172370	LDXT 032112	MODE1 040622
FFBBF1 014716	HHC2 025242	KDPAR7= 172376	LF = 000012	MS1 047505
FFBDON 015042	HHC20 025370	KDPDR0= 172320	LLBBF0 017054	MS10 = 047545
FFBTP1 014666	HHC25 025350	KDPDR1= 172322	LLBBF1 017064	MS11 047605
FFBTP2 014676	HT = 000011	KDPDR2= 172324	LLBDON 017172	MS2 047523
FFB1 014556	IBKOUT 051776	KDPDR3= 172326	LLBTP1 017034	MS20 047630
FFB10 014726	IBSAVE 044024	KDPDR4= 172330	LLBTP2 017044	MS21 047642
FFB15 014762	ICOUT 062662	KDPDR7= 172336	LLB1 016734	MS3 047545
FFB2 014626	IDONE 040104	KIPAR0= 172340	LLB10 017074	MS4 047563
FFB20 015010	IENBT 051544	KIPAR1= 172342	LLB15 017130	MULTER 042666
FFB25 015026	IEZBT 051240	KIPAR2= 172344	LLB2 017000	NATBF1 023574

NATER1	023524	OOC25	032560	QQB1	020272	RRR10	007742	STORE	043014
NATER2	023542	OOODON	006736	QQB10	020436	RRR11	007764	STXBF	036726
NATER3	023556	OOOT	006646	QQB15	020456	RRR12	010004	STXSUB	036520
NATINS	023252	O001	006602	QQB2	020342	RRR15	010036	SWR	001140
NATRET	023536	O002	006626	QQB20	020474	RRR2	007642	SWREG	000176
NATSUB	023172	O003	006662	QQCDON	033302	RRR25	010016	SW0	= 000001
NEGDA	056355	O004	006730	QQCTB0	033150	RRR3	007644	SW00	= 000001
NEGDNR	056151	OR	055232	QQCTB1	033154	RRR4	007670	SW01	= 000002
NNBBFO	017566	PASCIZ	047106	QQCTB2	033164	RSETUP=	104412	SW02	= 000004
NNBDON	017646	PCBAD	056163	QQC1	033044	RTI	042772	SW03	= 000010
NNBTP1	017546	PERIOD	056225	QQC10	033170	R6	=%000006	SW04	= 000020
NNBTP2	017556	PIRQ =	177772	QQC15	033210	R7	=%000007	SW05	= 000040
NNB1	017450	PIRQVE=	000240	QQC2	033116	SAVIOT	040522	SW06	= 000100
NNB10	017576	POWERM	047434	QQC20	033232	SAVREG=	104410	SW07	= 000200
NNB11	017626	PPBDON	020266	QQC25	033254	SCOPE =	000004	SW08	= 000400
NNB15	017632	PPBTP1	020174	QQQBF0	007342	SPACE	047500	SW09	= 001000
NNB2	017474	PPBTP2	020204	QQQBF1	007356	SR1 =	177574	SW1	= 000002
NNCDON	032354	PPB1	020062	QQQDON	007560	SSBDON	021152	SW10	= 002000
NNCTB0	032226	PPB10	020214	QQQTP1	007372	SSBTP1	021056	SW11	= 004000
NNCTB1	032232	PPB15	020234	QQQ1	007202	SSBTP2	021066	SW12	= 010000
NNC1	032126	PPB2	020130	QQQ10	007402	SSBTP3	021076	SW13	= 020000
NNC10	032242	PPB20	020252	QQQ2	007220	SSB1	020734	SW14	= 040000
NNC15	032262	PPCDON	033040	QQQ20	007422	SSB10	021100	SW15	= 100000
NNC2	032174	PPCTB0	032712	QQQ22	007436	SSB15	021120	SW2	= 000004
NNC20	032304	PPCTB1	032716	QQQ23	007470	SSB2	021012	SW3	= 000010
NNC25	032326	PPC1	032612	QQQ24	007516	SSB20	021136	SW4	= 000020
NODAT	043000	PPC10	032726	QQQ25	007542	SSCDON	034020	SW5	= 000040
NO	050233	PPC15	032746	QQQ3	007256	SSCTB0	033664	SW6	= 000100
NO4	066346	PPC2	032660	QQQ4	007304	SSCTB1	033670	SW7	= 000200
N1	050235	PPC20	032770	RDCHR =	104407	SSC1	033552	SW8	= 000400
N10	066351	PPC25	033012	RESREG=	104411	SSC10	033700	SW9	= 001000
N2	050237	PPPBF0	007100	RESVEC=	000010	SSC15	033720	TAB	= 000011
N244	066326	PPPBF1	007114	RETURN	042750	SSC2	033622	TBITVE=	000014
N3	050241	PPPDON	007176	RRBDON	020730	SSC20	033742	TCCBF0	025654
N3132	055632	PPPTP1	007130	RRBTP1	020634	SSC25	033764	TCCBF1	025664
N4	050243	PPP1	006742	RRBTP2	020644	SSC30	034014	TCCDON	025674
N5	050245	PPP10	007140	RRBTP3	020654	SSSA1	010254	TCC1	025542
N6	050247	PPP15	007160	RRB1	020514	SSSBF0	010244	TCC2	025564
N7	050251	PPP2	006760	RRB10	020656	SSSDON	010412	TCC3	025630
N746T2	056140	PPP3	007022	RRB15	020676	SSSTP1	010264	TKVEC =	000060
N747T2	056344	PPP4	007042	RRB2	020570	SSSTP2	010274	TPVEC =	000064
O0BDON	020056	PROGNUM=	000003	RRB20	020714	SSS1	010106	TRAPTO	066276
O0BTP1	017764	PRST	050176	RRCDON	033546	SSS10	010304	TRAPV	040010
O0BTP2	017774	PRO =	000000	RRCTB0	033414	SSS11	010326	TRAPVE=	000034
O0B1	017652	PR1 =	000040	RRCTB1	033420	SSS15	010336	TRPV	041636
O0B10	020004	PR2 =	000100	RRCTB2	033430	SSS2	010164	TRTVEC=	000014
O0B15	020024	PR3 =	000140	RRC1	033306	SSS20	010354	TST1	006600
O0B2	017720	PR4 =	000200	RRC10	033434	SSS25	010374	TST10	011134
O0B20	020042	PR5 =	000240	RRC15	033454	STACK =	001100	TST100	037536
O0CDON	032606	PR6 =	000300	RRC2	033362	START	006116	TST101	037620
O0CTB0	032460	PR7 =	000340	RRC20	033476	STCDF5	013036	TST102	037636
O0CTB1	032464	PS =	177776	RRC25	033520	STCDT	013320	TST103	037674
O0C1	032360	PSW =	177776	RRRDON	010102	STCFDS	012202	TST104	037732
O0C10	032474	PWRVEC=	000024	RRREXP	007732	STCFT	012464	TST105	040524
O0C15	032514	QQBDON	020510	RRRTP1	007712	STCIBF	036146	TST106	040602
O0C2	032426	QQBTP1	020406	RRRTP2	007722	STCSUB	035666	TST107	040670
O0C20	032536	QQBTP2	020426	RRR1	007564	STKLMT=	177774	TST11	011370

TST110	040762	TST7	010664	UUC2	034340	WWC2	034564	ZZCBF	037132
TST111	041116	TST70	034306	UUC3	034350	WWC4	034630	ZZCDON	037144
TST112	041212	TST71	034412	UUUA1	011020	WWC5	034674	ZZC1	036744
TST113	041304	TST72	034516	UUUA2	011022	WWC6	034740	ZZC10	037104
TST114	041434	TST73	036160	UUUA3	011024	WWC7	035004	ZZC12	037120
TST115	043202	TST74	036230	UUUBFO	011006	WWC8	035050	ZZC15	037124
TST12	011642	TST75	036742	UUUDON	011132	WWC9	035114	ZZC2	036754
TST13	012476	TST76	037424	UUUTP1	011032	WWWBF0	011520	ZZC3	037002
TST14	013332	TST77	037454	UUU1	010666	WWWBF1	011540	ZZC5	037100
TST15	013432	TTBDON	021362	UUU10	011042	WWWDON	011640	ZZF1	037426
TST16	013644	TTBTP1	021270	UUU11	011064	WWWTP1	011530	ZZZDON	013430
TST17	013744	TTBTP2	021300	UUU15	011076	WWW1	011372	ZZZ1	013334
TST2	006740	TTB1	021156	UUU2	010752	WWW10	011550	ZZZ10	013376
TST20	014046	TTB10	021310	UUU20	011114	WWW11	011572	ZZZ15	013414
TST21	014264	TTB15	021330	UUU3	010776	WWW15	011604	ZZZ2	013352
TST22	014554	TTB2	021224	VVBDON	014044	WWW2	011462	\$APTHD	006102
TST23	015044	TTB20	021346	VVB1	013746	WWW20	011622	\$ATYC	045320
TST24	015334	TTCDON	034304	VVB10	014010	XXBDON	022052	\$ATY1	045274
TST25	015634	TTCTB0	034144	VVB15	014026	XXBTP1	021740	\$ATY3	045302
TST26	016134	TTCTB1	034150	VVB2	013764	XXBTP2	021750	\$ATY4	045312
TST27	016430	TTCTB2	034160	VVCBFO	034506	XXB1	021614	\$AUTOB	001134
TST3	007200	TTC1	034024	VVCDON	034514	XXB10	022000	\$BASE	001372
TST30	016732	TTC10	034164	VVCTP1	034474	XXB15	022020	\$BDADR	001122
TST31	017174	TTC15	034204	VVC1	034414	XXB2	021666	\$BDDAT	001126
TST32	017446	TTC2	034102	VVC2	034444	XXB20	022036	\$BELL	001306
TST33	017650	TTC20	034226	VVC3	034454	XXB25	021760	\$CDW1	001376
TST34	020060	TTC25	034250	VVVBFO	011256	XXCDON	036226	\$CDW2	001400
TST35	020270	TTC30	034300	VVVDON	011366	XXC1	036162	\$CHARC	045042
TST36	020512	TTTA1	010550	VVVTP1	011266	XXXDON	012474	\$CKSWR	045542
TST37	020732	TTTA2	010552	VVV1	011136	XXX1	011644	\$CLR.T	043412
TST4	007562	TTTA3	010554	VVV10	011276	XXX2	011720	\$CMTAG	001100
TST40	021154	TTTBFO	010536	VVV11	011320	XXX3	011774	\$CM1 =	000024
TST41	021364	TTTDON	010662	VVV15	011332	XXX4	012050	\$CM2 =	000050
TST42	021612	TTTTP1	010562	VVV2	011220	XXX5	012124	\$CM3 =	000024
TST43	022054	TTT1	010416	VVV20	011350	YBBDON	022324	\$CM4 =	000024
TST44	022326	TTT10	010572	WENTTO	050204	YBTP1	022210	\$CNTLG	046161
TST45	023610	TTT11	010614	WWBDON	023606	YBTP2	022220	\$CNTLU	046154
TST46	023770	TTT15	010626	WWB1	022330	YBTP3	022230	\$CPUOP	001344
TST47	024146	TTT2	010502	WWB2	022406	YB1	022056	\$CRLF	001313
TST5	010104	TTT20	010644	WWB3	022464	YB10	022252	\$DDW0	001402
TST50	024342	TTT3	010526	WWB4	022542	YB15	022272	\$DDW1	001404
TST51	024552	TYPE =	104401	WWB5	022620	YB2	022136	\$DDW10	001426
TST52	024754	TYPOC =	104402	WWB6	022676	YB20	022310	\$DDW11	001430
TST53	025172	TYPON =	104404	WWB7	022754	YB25	022232	\$DDW12	001432
TST54	025424	TYPOS =	104403	WWB8	023032	YYCDON	036740	\$DDW13	001434
TST55	025540	UUBDON	021610	WWB9	023110	YYC1	036232	\$DDW14	001436
TST56	025676	UUBTP1	021442	WWCDON	036156	YYC2	036270	\$DDW15	001440
TST57	027250	UUBTP2	021514	WWC1	034520	YYC3	036326	\$DDW2	001406
TST6	010414	UUB1	021366	WWC10	035160	YYC4	036364	\$DDW3	001410
TST60	030256	UUB10	021524	WWC11	035224	YYC5	036422	\$DDW4	001412
TST61	032124	UUB15	021562	WWC12	035270	YYC6	036460	\$DDW5	001414
TST62	032356	UUB2	021440	WWC13	035334	YYYDON	013330	\$DDW6	001416
TST63	032610	UUB20	021544	WWC14	035400	YYY1	012500	\$DDW7	001420
TST64	033042	UUCBFO	034402	WWC15	035444	YYY2	012554	\$DDW8	001422
TST65	033304	UUCDON	034410	WWC16	035510	YYY3	012630	\$DDW9	001424
TST66	033550	UUCTP1	034370	WWC17	035554	YYY4	012704	\$DEVCT	001326
TST67	034022	UUC1	034310	WWC18	035620	YYY5	012760	\$DEVM	001374

SYMBOL TABLE

\$DOAGN	043432	\$LOOP	043470	\$PWRMG	046430	\$SETUP=	000137	\$TMP5	001244
\$ENDAD	043422	\$LPADR	001106	\$PWUP	046346	\$STUP =	177777	\$TMP6	001246
\$ENDCT	043236	\$LPERR	001110	\$QUES	001312	\$SVLAD	043750	\$TMP7	001250
\$ENULL	043476	\$MADR1	001350	\$RDCHR	046024	\$SVPC =	006102	\$TN =	000115
\$ENV	001336	\$MADR2	001354	\$RDSZ =	000001	\$SWR =	177400	\$TPB	001152
\$ENVM	001337	\$MADR3	001360	\$REGAD	001160	\$SWREG	001340	\$TPFLG	001157
\$EOP	043202	\$MADR4	001364	\$REG0	001162	\$SWRMK =	000000	\$TPS	001150
\$EOPCT	043230	\$MAIL	001316	\$REG1	001164	\$SWRMS=	000200	\$TRAP	046210
\$ERFLG	001103	\$MAMS1	001346	\$REG10	001202	\$TAB	047503	\$TRAP2	046232
\$ERMAX	001115	\$MAMS2	001352	\$REG11	001204	\$TBIT	043474	\$TRP =	000014
\$ERROR	044026	\$MAMS3	001356	\$REG12	001206	\$TERM =	000030	\$TRPAD	046244
\$ERRPC	001116	\$MAMS4	001362	\$REG13	001210	\$TESTN	001322	\$TSTM	006106
\$ERRTB	001442	\$MBADR	006104	\$REG14	001212	\$TIMES	001302	\$TSTM	001102
\$ERTTL	001112	\$MFLG	045536	\$REG15	001214	\$TKB	001146	\$TYPE	044506
\$ESCAP	001304	\$MNEW	046177	\$REG16	001216	\$TKS	001144	\$TYPEC	044724
\$ETABL	001336	\$MSGAD	001332	\$REG17	001220	\$TMP0	001232	\$TYPEX	045044
\$ETEND	001442	\$MSGLG	001334	\$REG2	001166	\$TMP1	001234	\$TYPOC	045072
\$FATAL	001320	\$MSGTY	001316	\$REG20	001222	\$TMP10	001252	\$TYPON	045106
\$FFLG	045540	\$MSWR	046166	\$REG21	001224	\$TMP11	001254	\$TYPOS	045046
\$FILLC	001156	\$SMTYP1	001347	\$REG22	001226	\$TMP12	001256	\$UNIT	001330
\$FILLS	001155	\$SMTYP2	001353	\$REG23	001230	\$TMP13	001260	\$UNITM	006112
\$GDADR	001120	\$SMTYP3	001357	\$REG3	001170	\$TMP14	001262	\$USWR	001342
\$GDDAT	001124	\$SMTYP4	001363	\$REG4	001172	\$TMP15	001264	\$VECT1	001366
\$GET42	043374	\$SMXCNT	044022	\$REG5	001174	\$TMP16	001266	\$VECT2	001370
\$GTSWR	045612	\$NULL	001154	\$REG6	001176	\$TMP17	001270	\$XOFF =	000023
\$HD =	000003	\$NWTST=	000001	\$REG7	001200	\$TMP2	001236	\$XON =	000021
\$HISTS	006102	\$SOCNT	045270	\$RESRE	044446	\$TMP20	001272	\$XTSTR	043514
\$ICNT	001104	\$SOMODE	045272	\$RTNAD	043472	\$TMP21	001274	\$SGET4=	000001
\$ILLUP	046452	\$SOVER	044004	\$RTRN	043466	\$TMP22	001276	\$OFILL	045271
\$INTAG	001135	\$SPASS	001324	\$SAVRE	044410	\$TMP23	001300	.LPER	047356
\$ITEMB	001114	\$SPASTM	006110	\$SAVR6	046456	\$TMP3	001240	.RSET	047364
\$LF	001314	\$PWAD	046434	\$SCOPE	043502	\$TMP4	001242	.SX =	006102
\$LFLG	045537	\$PWADN	046274						

. ABS. 071306 000
000000 001
ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 60456 WORDS (237 PAGES)
DYNAMIC MEMORY: 20034 WORDS (77 PAGES)
ELAPSED TIME: 00:50:11
CKFPCC.BIN,CKFPCC/CR/-SP/NL:TOC=CKFPCC.MLB/ML,CKFPCC.P11