

.REM 8

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

IDENTIFICATION

PRODUCT CODE: AC-F636D-MC
PRODUCT NAME: CKFPCDO FP11F FLTG PNT PRT C
PRODUCT DATE: OCTOBER, 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, 1982 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.

THE FOLLOWING WAS ADDED TO THE 'D' VERSION: (LABELED BY ';DPM002' IN COMMENTS)

IN TEST 74 (TESTING 'STEXP'), IN EACH LOCATION WHERE THE CODE GOES TO CHECK THE 'STEXP' INSTRUCTION IN THE SUBROUTINE, IT NOW DOES THAT SECTION 3 TIMES IN ORDER TO CHECK THE ADDITIONAL MODES 2 AND 4 (AUTO INCREMENT/DECREMENT). A TABLE WAS ADDED CONSISTING OF THREE 3-WORD DATA GROUPS. THE FIRST AND LAST GROUPS PROVIDE THE DATA TO LOAD AND CHECK THE PROPER PRE AND POST VALUES RESPECTIVELY OF R0 USED IN EXECUTING THE INSTRUCTION. THE SECOND GROUP IS THE OP CODES OF THE THREE MODES OF THE INSTRUCTION. THE 'STEXP' OP CODE IS PLACED IN THE FLOW JUST BEFORE EXECUTION.

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

IN TESTS 76 THROUGH 114, LOOP ON ERROR ADDRESS LOAD TO \$LPERR WAS ADDED JUST BEFORE EACH TEST INSTRUCTION.

IN THE END-OF-PASS SECTION, THE EOP MESSAGE IS PRINTED EVERY 1000 PASSES BECAUSE EXECUTION TIME PER PASS IS SO SHORT.

IN THE \$TYPCS ROUTINE, IT NO LONGER PRINTS SPACE CHARACTERS AHEAD OF NON-ZERO DIGITS.

106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155

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 CKFPCDO
- 10. LISTING
 - 10.1 CKFPCDO

157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213

1. ABSTRACT

THE THREE PROGRAMS:

CKFPACO CKFPBBO CKFPCDO

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. CKFPACO

CKFPACO 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. CKFPBBO

CKFPBBO TESTS:

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

C. CKFPCDO

CKFPCDO TESTS:

SIF AND STD (ALL MODES)
STCFD AND STCDF
CLR D AND CLR F
NEGF AND NEG D

214 ABSF AND ABS
 215 TSTF AND TSTD
 216 NEGF, ABSF AND TSTF (ALL SOURCE MODES)
 217 NEGF, ABSF AND TSTF (ALL SOURCE MODES)
 218 LDFPS (ALL SOURCE MODES)
 219 LDCIF AND LDCLF
 220 LDCID AND LDCLD
 221 LDEXP
 222 STFPS (ALL DESTINATION MODES)
 223 STCFL AND STCFI
 224 STCDL AND STCDI
 225 STEXP
 226 STST
 227 I AND D SPACE TESTS (ALL MODES AND REGS 0 AND 7)
 228 AUTO INCREMENT/DECREMENT CHECK - SR1 (ALL MODES AND REGS 1 AND 7)
 229
 230

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 CKFPBBO 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

214
 215
 216
 217
 218
 219
 220
 221
 222
 223
 224
 225
 226
 227
 228
 229
 230
 231
 232
 233
 234
 235
 236
 237
 238
 239
 240
 241
 242
 243
 244
 245
 246
 247
 248
 249
 250
 251
 252
 253
 254
 255
 256
 257
 258
 259
 260
 261
 262
 263
 264
 265
 266
 267
 268
 269
 270

271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327

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...	10000	HALT ON ERROR
SW<14>=1...	4000	LOOP ON CURRENT TEST
SW<13>=1...	2000	INHIBIT ERROR TYPE OUTS
SW<12>=1...	1000	INHIBIT T-BIT TRAPPING
SW<11>=1...	400	INHIBIT ITERATIONS
SW<10>=1...	200	RING TTY BELL ON ERROR
SW<9>=1....	100	LOOP ON ERROR
SW<8>=1....	40	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 CKFPACO.
SW<7>=1....	200	SELECT CORRECT INTERRUPT TEST IN PROGRAM CKFPBBO.

6. ERRORS

6.1 SUMMARIES

IN PROGRAM CKFPACO 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 CKFPACO BOTH SWITCHES 13 AND 7 MUST = 1.

6. ERROR RECOVERY

328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384

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

NONE

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

385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441

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

8.6 INTERRUPTS TEST

IN PROGRAM CKFPBBO 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.

442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498

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

```

499          NEG D INST WITH MODE ZERO AND ILLEGAL AC7.
500
501          TEST 20          NEG F, ABSF AND TSTF SOURCE MODE 0 TEST
502          -----
503
504          THIS IS A TEST THE NEG F, ABSF AND TSTF SOURCE FLOWS.
505          THE NEG D INSTRUCTION IS USED TO TEST MODE 0
506
507          TEST 21          NEG F, ABSF AND TSTF SOURCE MODE 1 TEST
508          -----
509
510          THIS IS A TEST THE NEG F, ABSF AND TSTF SOURCE FLOWS.
511          THE NEG D INSTRUCTION IS USED TO TEST MODE 1
512
513          TEST 22          NEG F, ABSF AND TSTF SOURCE MODE 2 TEST
514          -----
515
516          THIS IS A TEST THE NEG F, ABSF AND TSTF SOURCE FLOWS.
517          THE ABS D INSTRUCTION IS USED TO TEST MODE 2
518
519          TEST 23          NEG F, ABSF AND TSTF SOURCE MODE 4 TEST
520          -----
521
522          THIS IS A TEST THE NEG F, ABSF AND TSTF SOURCE FLOWS.
523          THE ABS D INSTRUCTION IS USED TO TEST MODE 4
524
525          TEST 24          NEG F, ABSF AND TSTF SOURCE MODE 3 TEST
526          -----
527
528          THIS IS A TEST THE NEG F, ABSF AND TSTF SOURCE FLOWS.
529          THE ABS D INSTRUCTION IS USED TO TEST MODE 3
530
531          TEST 25          NEG F, ABSF AND TSTF SOURCE MODE 5 TEST
532          -----
533
534          THIS IS A TEST THE NEG F, ABSF AND TSTF SOURCE FLOWS.
535          THE NEG D INSTRUCTION IS USED TO TEST MODE 5
536
537          TEST 26          NEG F, ABSF AND TSTF SOURCE MODE 6 TEST
538          -----
539
540          THIS IS A TEST THE NEG F, ABSF AND TSTF SOURCE FLOWS.
541          THE ABS D INSTRUCTION IS USED TO TEST MODE 6
542
543          TEST 27          NEG F, ABSF AND TSTF SOURCE MODE 7 TEST
544          -----
545
546          THIS IS A TEST THE NEG F, ABSF AND TSTF SOURCE FLOWS.
547          THE ABS D INSTRUCTION IS USED TO TEST MODE 6
548
549          TEST 30          NEG F, ABSF AND TSTF SOURCE MODE 6, GR7, TEST
550          -----
551
552          THIS IS A TEST THE NEG F, ABSF AND TSTF SOURCE FLOWS.
553          THE NEG D INSTRUCTION IS USED TO TEST MODE 6
554
555          TEST 31          NEG F, ABSF AND TSTF SOURCE MODE 7, GR7, TEST

```

556 -----
557
558 THIS IS A TEST THE NEGF, ABSF AND TSTF SOURCE FLOWS.
559 THE ABSD INSTRUCTION IS USED TO TEST MODE 7
560
561 TEST 32 SPECIAL DEST, MODE 0, TEST
562 -----
563
564 THIS IS A TEST OF THE NEGF ABSF AND TSTF DESTINATION
565 FLOWS MODE 0 USING THE NEGD INSTR.
566
567 TEST 33 SPECIAL DEST, MODE 1, TEST
568 -----
569
570 THIS IS A TEST OF THE NEGF ABSF AND TSTF DESTINATION
571 FLOWS MODE 1 USING THE NEGD INSTR.
572
573 TEST 34 SPECIAL DEST, MODE 2, TEST
574 -----
575 THIS IS A TEST OF THE NEGF ABSF AND TSTF DESTINATION
576 FLOWS MODE 2 USING THE NEGD INSTR.
577
578 TEST 35 SPECIAL DEST, MODE 4, TEST
579 -----
580
581 THIS IS A TEST OF THE NEGF ABSF AND TSTF DESTINATION
582 FLOWS MODE 4 USING THE NEGD INSTR.
583
584 TEST 36 SPECIAL DEST, MODE 3, TEST
585 -----
586
587 THIS IS A TEST OF THE NEGF ABSF AND TSTF DESTINATION
588 FLOWS MODE 3 USING THE NEGD INSTR.
589
590 TEST 37 SPECIAL DEST, MODE 5, TEST
591 -----
592
593 THIS IS A TEST OF THE NEGF ABSF AND TSTF DESTINATION
594 FLOWS MODE 5 USING THE NEGD INSTR.
595
596 TEST 40 SPECIAL DEST, FLOATING MODE 2, TEST
597 -----
598
599 THIS IS A TEST OF THE NEGF ABSF AND TSTF DESTINATION
600 FLOWS MODE 2 USING THE NEGF INSTR.
601
602 TEST 41 SPECIAL DEST, MODE2, GR7 (IMMEDIATE), TEST
603 -----
604
605 THIS IS A TEST OF THE NEGF ABSF AND TSTF DESTINATION
606 FLOWS MODE 2(IMMEDIATE) USING THE NEGD INSTR.
607
608 TEST 42 SPECIAL DEST, MODE 6, TEST
609 -----
610
611 THIS IS A TEST OF THE NEGF ABSF AND TSTF DESTINATION
612 FLOWS MODE 6 USING THE NEGD INSTR.

```

613
614
615 TEST 43 SPECIAL DEST, MODE 7, TEST
616 -----
617 THIS IS A TEST OF THE NEGF ABSF AND TSTF DESTINATION
618 FLOWS MODE 7 USING THE NEGD INSTR.
619
620 TEST 44 NEGD, ABSD AND TSTD TEST
621 -----
622 THIS IS A TEST OF THE NEGD ABSD AND TSTD
623 INSTRUCTIONS.
624
625 TEST 45 SOURCE MODES, MODE 1 (FL=0), TEST
626 -----
627 THIS IS A TEST OF SOURCE MODE 1 USING THE LDFPS
628 INSTRUCTION.
629
630 TEST 46 SOURCE MODES, MODE 2 (FL=0), TEST
631 -----
632 THIS IS A TEST OF SOURCE MODE 2 USING THE LDFPS
633 INSTRUCTION.
634
635 TEST 47 SOURCE MODES, MODE 4 (FL=0), TEST
636 -----
637 THIS IS A TEST OF SOURCE MODE 4 USING THE LDFPS
638 INSTRUCTION.
639
640 TEST 50 SOURCE MODES, MODE 3 (FL=0), TEST
641 -----
642 THIS IS A TEST OF SOURCE MODE 3 USING THE LDFPS
643 INSTRUCTION.
644
645 TEST 51 SOURCE MODES, MODE 5 (FL=0), TEST
646 -----
647 THIS IS A TEST OF SOURCE MODE 5 USING THE LDFPS
648 INSTRUCTION.
649
650 TEST 52 SOURCE MODES, MODE 6 (FL=0), TEST
651 -----
652 THIS IS A TEST OF SOURCE MODE 6 USING THE LDFPS
653 INSTRUCTION.
654
655 TEST 53 SOURCE MODES, MODE 7 (FL=0), TEST
656 -----
657 THIS IS A TEST OF SOURCE MODE 7 USING THE LDFPS
658 INSTRUCTION.
659
660 TEST 54 SOURCE MODES, MODE 2 GR7 (FL=1), TEST
661 -----
662
663
664
665
666
667
668
669

```

670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726

THIS IS A TEST OF THE LDCLD WITH IMMEDIATE ADDRESSING MODE

TEST 55 SOURCE MODES, MODE 2 (FL=1), TEST

THIS IS A TEST OF THE LDCLD INSTRUCTION WITH MODE 2.

TEST 56 LDCIF AND LDCLF TEST

THIS IS A TEST OF THE LDCIF AND THE LDCLF INSTRUCTIONS.

TEST 57 LDCID AND LDCLD TEST

THIS IS A TEST OF LDCID AND LDCLD

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.

TEST 61 DESTINATION MODES, MODE 1 (FL=0), TEST

THIS IS A TEST OF DESTINATION MODE 1 USING THE STFPS INSTRUCTION

TEST 62 DESTINATION MODES, MODE 2 (FL=0), TEST

THIS IS A TEST OF DESTINATION MODE 2 USING THE STFPS INSTRUCTION

TEST 63 DESTINATION MODES, MODE 4 (FL=0), TEST

THIS IS A TEST OF DESTINATION MODE 4 USING THE STFPS INSTRUCTION

TEST 64 DESTINATION MODES, MODE 3 (FL=0), TEST

THIS IS A TEST OF DESTINATION MODE 3 USING THE STFPS INSTRUCTION

TEST 65 DESTINATION MODES, MODE 5 (FL=0), TEST

THIS IS A TEST OF DESTINATION MODE 5 USING THE STFPS INSTRUCTION

TEST 66 DESTINATION MODES, MODE 6 (FL=0), TEST

736 THIS IS A TEST OF DESTINATION MODE 7 USING THE STFPS
 737 INSTRUCTION
 738
 739 TEST 70 DESTINATION MODES, MODE 2 (FL=1), TEST
 740 -----
 741
 742 THIS IS A TEST OF DESTINATION MODE 2 USING STCOL
 743 WITH REGISTER 0
 744
 745 TEST 71 DESTINATION MODES, MODE 4 (FL=1), TEST
 746 -----
 747
 748 THIS IS A TEST OF DESTINATION MODE 4 USING STCDL
 749 WITH REGISTER 0
 750
 751 TEST 72 STCDI AND STCDL TEST
 752 -----
 753
 754 THIS IS A TEST OF THE STCDI AND STCDL INSTRUCTIONS.
 755 NOTE THAT A SUBROUTINE, STCSUB, IS USED TO SET UP
 756 THE OPERANDS, EXECUTE THE STC INSTRUCTION AND CHECK
 757 THE RESULT.
 758
 759 TEST 73 STCFL AND STCFI TEST
 760 -----
 761
 762 THIS IS A TEST OF STCFL AND STCFI. IT MAKES USE OF
 763 THE SAME SUBROUTINE, STCSUB, WHICH WAS USED TO TEST
 764 STCDL AND STCDI.
 765
 766 TEST 74 STEXP TEST
 767 -----
 768
 769 THIS IS A TEST OF THE STEXP INSTRUCTION
 770
 771 TEST 75 STST TEST
 772 -----
 773
 774 THIS IS A TEST OF THE STST INSTRUCTION. FIRST AN
 775 ILLEGAL FPS OP CODE (INSTRUCTION) IS USED TO ENTER
 776 AN ERROR CONDITION IN THE FEC AND FEA. THE STST IS
 777 EXECUTED AND THE FEC AND FEA ARE CHECKED
 778
 779 TEST 76 D-SPACE NON-ACCESS TEST
 780 -----
 781
 782 THIS IS A TEST THAT ENABLES D-SPACE, BUT MAKES IT
 783 NON-RESIDENT, CAUSING A MEMORY MANAGEMENT TRAP
 784 SHOULD IT BE ACCESSED DURING AN INSTRUCTION THAT
 785 WILL NOT NORMALLY ACCESS D-SPACE.
 786
 787 TEST 77 AUTO INCREMENT/DECREMENT TEST
 788 -----
 789
 790 THIS IS A TEST THAT ENABLES D-SPACE, BUT MAKES IT
 791 NON-RESIDENT IN THE AREA OF THE TEST, FORCING A
 792 MEMORY MANAGEMENT TRAP FOR EVERY FPP INSTRUCTION IN

793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810

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 BEX

```

1974          .MCALL .HEADER, .SWRHI, .EQUAT, .SETUP, .SCATCH, .SACT11
1975          .MCALL .SSAVE, .SCMTAG
1976          .MCALL .STYPDEC, .STRAP, .SPOWER, .SAPTHDR, .SAPTBL5
1977          .MCALL .SAPTYPE, .SREAD
1978          .MCALL .EQUIV                                ;REMGVE FOR PDP-10
1979
          .TITLE CKFPCDO 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
          160000 $SWR=160000      ;;HALT ON ERROR, LOOP ON TEST, INHIBIT ERROR TYP0UT

1980
1981
1982          000244      FPVECT=244
1983          000250      MMVECT=250
1984          177400      $SWR=177400
1985          000200      $SWRMSK=200
1986          000011      TAB=11
1987          000015      CRLF=15
1988
1989          .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

```

```
000200 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
ERRVEC= 4      ;;TIME OUT AND OTHER ERRORS
RESVEC= 10     ;;RESERVED AND ILLEGAL INSTRUCTIONS
TBITVEC=14    ;;'T' BIT
TRTVEC= 14    ;;TRACE TRAP
BPTVEC= 14    ;;BREAKPOINT TRAP (BPT)
IOTVEC= 20    ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
PWRVEC= 24    ;;POWER FAIL
EMTVEC= 30    ;;EMULATOR TRAP (EMT) **ERROR**
TRAPVEC=34    ;;'TRAP' TRAP
TKVEC= 60     ;;TTY KEYBOARD VECTOR
TPVEC= 64     ;;TTY PRINTER VECTOR
PIRQVEC=240   ;;PROGRAM INTERRUPT REQUEST VECTOR

1990          .SBTTL FPP REGISTER DEFINITIONS
1991          AC0      =%0
1992          AC1      =%1
1993          AC2      =%2
1994          AC3      =%3
1995          AC4      =%4
1996          AC5      =%5
1997          AC6      =%6
1998          AC7      =%7
1999          KIPDR0   =172300
2000          KIPDR1   =172302
2001          KIPDR2   =172304
2002          KIPDR3   =172306
2003          KIPDR4   =172310
2004          KIPDR7   =172316
2005          KIPAR0   =172340
2006          KIPAR1   =172342
2007          KIPAR2   =172344
2008          KIPAR3   =172346
2009          KIPAR4   =172350
2010          KIPAR7   =172356
2011          KDPDR0   =172320
2012          KDPDR1   =172322
2013          KDPDR2   =172324
2014          KDPDR3   =172326
2015          KDPDR4   =172330
2016          KDPDR7   =172336
2017          KDPAR0   =172360
2018          KDPAR1   =172362
2019          KDPAR2   =172364
2020          KDPAR3   =172366
2021          KDPAR4   =172370
2022          KDPAR7   =172376
2023          MMR0     =177572
2024          SR1      =177574
2025          MMR2     =177576
2026          MMR3     =172516
2027          DATA    =117760
2028          IOTRAP   =000020
2029
2031          .SBTTL TRAP CATCHER
2032          .=0
000000
    
```

TRAP CATCHER

;*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

2033

```
.SBTTL COMMON TAGS
:*****
:*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
:*USED IN THE PROGRAM.
.=1100

001100 001100
001100 000000
001102 000
001103 000
001104 000000
001106 000000
001110 000000
001112 000000
001114 000
001115 001
001116 000000
001120 000000
001122 000000
001124 000000
001126 000000
001130 000000
001132 000000
001134 000
001135 000
001136 000000
001140 177570
001142 177570
001144 177560
001146 177562
001150 177564
001152 177566
001154 000
001155 002
001156 012
001157 000
001160 000000

001162 000024
001164 000000
001166 000000
001170 000000
001172 000000
001174 000000
001176 000000
001200 000000
001202 000000
001204 000000
001206 000000
001210 000000
001212 000000
001214 000000
001216 000000
001220 000000
001222 000000
001224 000000
001226 000000

.SBTTL COMMON TAGS
:*****
:*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
:*USED IN THE PROGRAM.
.=1100

$CMTAG: .WORD 0 ;;START OF COMMON TAGS
$TSTNM: .BYTE 0 ;;CONTAINS THE TEST NUMBER
$ERFLG: .BYTE 0 ;;CONTAINS ERROR FLAG
$ICNT: .WORD 0 ;;CONTAINS SUBTEST ITERATION COUNT
$LPADR: .WORD 0 ;;CONTAINS SCOPE LOOP ADDRESS
$LPERR: .WORD 0 ;;CONTAINS SCOPE RETURN FOR ERRORS
$ERTTL: .WORD 0 ;;CONTAINS TOTAL ERRORS DETECTED
$ITEMB: .BYTE 0 ;;CONTAINS ITEM CONTROL BYTE
$ERMAX: .BYTE 1 ;;CONTAINS MAX. ERRORS PER TEST
$ERRPC: .WORD 0 ;;CONTAINS PC OF LAST ERROR INSTRUCTION
$GDADR: .WORD 0 ;;CONTAINS ADDRESS OF 'GOOD' DATA
$BDADR: .WORD 0 ;;CONTAINS ADDRESS OF 'BAD' DATA
$GDDAT: .WORD 0 ;;CONTAINS 'GOOD' DATA
$BDDAT: .WORD 0 ;;CONTAINS 'BAD' DATA
        .WORD 0 ;;RESERVED--NOT TO BE USED
        .WORD 0
$AUTOB: .BYTE 0 ;;AUTOMATIC MODE INDICATOR
$INTAG: .BYTE 0 ;;INTERRUPT MODE INDICATOR
        .WORD 0
SWR: .WORD DSWR ;;ADDRESS OF SWITCH REGISTER
DISPLAY: .WORD DDISP ;;ADDRESS OF DISPLAY REGISTER
$TKS: 177560 ;;TTY KBD STATUS
$TKB: 177562 ;;TTY KBD BUFFER
$TPS: 177564 ;;TTY PRINTER STATUS REG. ADDRESS
$TPB: 177566 ;;TTY PRINTER BUFFER REG. ADDRESS
$NULL: .BYTE 0 ;;CONTAINS NULL CHARACTER FOR FILLS
$FILLS: .BYTE 2 ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
$FILLC: .BYTE 12 ;;INSERT FILL CHARS. AFTER A 'LINE FEED'
$TPFLG: .BYTE 0 ;;'TERMINAL AVAILABLE' FLAG (BIT<07>=0=YES)
$REGAD: .WORD 0 ;;CONTAINS THE ADDRESS FROM
        ;;WHICH ($REGO) WAS OBTAINED

$REPT $CM3
$REG0: .WORD 0 ;;CONTAINS (($REGAD)+0)
$REG1: .WORD 0 ;;CONTAINS (($REGAD)+2)
$REG2: .WORD 0 ;;CONTAINS (($REGAD)+4)
$REG3: .WORD 0 ;;CONTAINS (($REGAD)+6)
$REG4: .WORD 0 ;;CONTAINS (($REGAD)+10)
$REG5: .WORD 0 ;;CONTAINS (($REGAD)+12)
$REG6: .WORD 0 ;;CONTAINS (($REGAD)+14)
$REG7: .WORD 0 ;;CONTAINS (($REGAD)+16)
$REG10: .WORD 0 ;;CONTAINS (($REGAD)+20)
$REG11: .WORD 0 ;;CONTAINS (($REGAD)+22)
$REG12: .WORD 0 ;;CONTAINS (($REGAD)+24)
$REG13: .WORD 0 ;;CONTAINS (($REGAD)+26)
$REG14: .WORD 0 ;;CONTAINS (($REGAD)+30)
$REG15: .WORD 0 ;;CONTAINS (($REGAD)+32)
$REG16: .WORD 0 ;;CONTAINS (($REGAD)+34)
$REG17: .WORD 0 ;;CONTAINS (($REGAD)+36)
$REG20: .WORD 0 ;;CONTAINS (($REGAD)+40)
$REG21: .WORD 0 ;;CONTAINS (($REGAD)+42)
$REG22: .WORD 0 ;;CONTAINS (($REGAD)+44)
```

COMMON TAGS

```

001230 000000 $REG23: .WORD 0 ;;CONTAINS (($REGAD)+46)
      000024 .REPT 24
001232 000000 $TMP0: .WORD 0 ;;USER DEFINED
001234 000000 $TMP1: .WORD 0 ;;USER DEFINED
001236 000000 $TMP2: .WORD 0 ;;USER DEFINED
001240 000000 $TMP3: .WORD 0 ;;USER DEFINED
001242 000000 $TMP4: .WORD 0 ;;USER DEFINED
001244 000000 $TMP5: .WORD 0 ;;USER DEFINED
001246 000000 $TMP6: .WORD 0 ;;USER DEFINED
001250 000000 $TMP7: .WORD 0 ;;USER DEFINED
001252 000000 $TMP10: .WORD 0 ;;USER DEFINED
001254 000000 $TMP11: .WORD 0 ;;USER DEFINED
001256 000000 $TMP12: .WORD 0 ;;USER DEFINED
001260 000000 $TMP13: .WORD 0 ;;USER DEFINED
001262 000000 $TMP14: .WORD 0 ;;USER DEFINED
001264 000000 $TMP15: .WORD 0 ;;USER DEFINED
001266 000000 $TMP16: .WORD 0 ;;USER DEFINED
001270 000000 $TMP17: .WORD 0 ;;USER DEFINED
001272 000000 $TMP20: .WORD 0 ;;USER DEFINED
001274 000000 $TMP21: .WORD 0 ;;USER DEFINED
001276 000000 $TMP22: .WORD 0 ;;USER DEFINED
001300 000000 $TMP23: .WORD 0 ;;USER DEFINED
001302 000000 $TIMES: 0 ;;MAX. NUMBER OF ITERATIONS
001304 000000 $ESCAPE: 0 ;;ESCAPE ON ERROR ADDRESS
001306 207 377 377 $BELL: .ASCII <207><377><377> ;;CODE FOR BELL
001312 077 $QUES: .ASCII /?/ ;;QUESTION MARK
001313 015 $CRLF: .ASCII <15> ;;CARRIAGE RETURN
001314 012 000 $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 $SUNIT: .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 $$WREG: .WORD ASWREG ;;APT SWITCH REGISTER
001342 000000 $USWR: .WORD AUSWR ;;USER SWITCHES
001344 000000 $CPUOP: .WORD ACPUOP ;;CPU TYPE OPTIONS
*
*
* 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

```

```

300 NSEC BIPOLAR=002
500 NSEC MOS=003
;*
;*
001350 000000 $MADR1: .WORD AMADR1 ;;HIGH ADDRESS,BLK#1
;*
MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF 'TYPE' ABOVE
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 $DEVW: .WORD ADEVW ;;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 SETEND:

```


.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:

2037	001442	000444					
2039	001442	052346	071204	073122	.REPT	MNUMBER	
	001452	052422	071274	073142	.WORD	EM1,DH1,DT1,DF1	:ITEM 1
	001462	052452	071340	073164	.WORD	EM2,DH2,DT2,DF2	:ITEM 2
	001472	052502	071404	073206	.WORD	EM3,DH3,DT3,DF3	:ITEM 3
	001502	052563	071447	073230	.WORD	EM4,DH4,DT4,DF4	:ITEM 4
	001512	052574	071447	073256	.WORD	EM5,DH5,DT5,DF5	:ITEM 5
	001522	052737	071404	073206	.WORD	EM6,DH6,DT6,DF6	:ITEM 6
	001532	052761	071447	073230	.WORD	EM7,DH7,DT7,DF7	:ITEM 7
	001542	052774	071404	073206	.WORD	EM10,DH10,DT10,DF10	:ITEM 10
	001552	053024	071447	073230	.WORD	EM11,DH11,DT11,DF11	:ITEM 11
	001562	053036	071460	073256	.WORD	EM12,DH12,DT12,DF12	:ITEM 12
	001572	053036	071460	073256	.WORD	EM13,DH13,DT13,DF13	:ITEM 13
	001602	053104	071447	073230	.WORD	EM14,DH14,DT14,DF14	:ITEM 14
	001612	053116	071520	073270	.WORD	EM15,DH15,DT15,DF15	:ITEM 15
	001622	053142	071460	073256	.WORD	EM16,DH16,DT16,DF16	:ITEM 16
	001632	053166	071404	073270	.WORD	EM17,DH17,DT17,DF17	:ITEM 17
	001642	053204	071447	073230	.WORD	EM20,DH20,DT20,DF20	:ITEM 20
	001652	053204	071447	073230	.WORD	EM21,DH21,DT21,DF21	:ITEM 21
	001662	053216	071460	073256	.WORD	EM22,DH22,DT22,DF22	:ITEM 22
	001672	053243	071404	073270	.WORD	EM23,DH23,DT23,DF23	:ITEM 23
	001702	053260	071447	073230	.WORD	EM24,DH24,DT24,DF24	:ITEM 24
	001712	053272	071460	073256	.WORD	EM25,DH25,DT25,DF25	:ITEM 25
	001722	053317	071404	073270	.WORD	EM26,DH26,DT26,DF26	:ITEM 26
	001732	053334	071447	073230	.WORD	EM27,DH27,DT27,DF27	:ITEM 27
	001742	053346	071460	073256	.WORD	EM30,DH30,DT30,DF30	:ITEM 30
	001752	053372	071404	073270	.WORD	EM31,DH31,DT31,DF31	:ITEM 31
	001762	053410	071447	073230	.WORD	EM32,DH32,DT32,DF32	:ITEM 32
	001772	053422	071460	073256	.WORD	EM33,DH33,DT33,DF33	:ITEM 33
	002002	053447	071404	073270	.WORD	EM34,DH34,DT34,DF34	:ITEM 34
	002012	053464	071447	073230	.WORD	EM35,DH35,DT35,DF35	:ITEM 35
	002022	053476	071563	073312	.WORD	EM36,DH36,DT36,DF36	:ITEM 36
	002032	053556	071563	073312	.WORD	EM37,DH37,DT37,DF37	:ITEM 37
	002042	053556	071626	073356	.WORD	EM40,DH40,DT40,DF40	:ITEM 40
	002052	053476	071563	073312	.WORD	EM41,DH41,DT41,DF41	:ITEM 41
	002062	053556	071563	073312	.WORD	EM42,DH42,DT42,DF42	:ITEM 42
	002072	053476	071563	073312	.WORD	EM43,DH43,DT43,DF43	:ITEM 43
	002102	053476	071563	073312	.WORD	EM44,DH44,DT44,DF44	:ITEM 44
	002112	053556	071563	073312	.WORD	EM45,DH45,DT45,DF45	:ITEM 45
	002122	053476	071563	073312	.WORD	EM46,DH46,DT46,DF46	:ITEM 46
	002132	053476	071563	073312	.WORD	EM47,DH47,DT47,DF47	:ITEM 47
	002142	054235	071563	073312	.WORD	EM50,DH50,DT50,DF50	:ITEM 50
	002152	054274	071563	073312	.WORD	EM51,DH51,DT51,DF51	:ITEM 51
	002162	054274	071626	073356	.WORD	EM52,DH52,DT52,DF52	:ITEM 52
	002172	054235	071563	073312	.WORD	EM53,DH53,DT53,DF53	:ITEM 53
	002202	054235	071563	073312	.WORD	EM54,DH54,DT54,DF54	:ITEM 54
					.WORD	EM55,DH55,DT55,DF55	:ITEM 55

ERROR POINTER TABLE

002212	054274	071563	073312	.WORD	EM56,DH56,DT56,DF56	:ITEM 56
002222	054274	071563	073312	.WORD	EM57,DH57,DT57,DF57	:ITEM 57
002232	054235	071563	073312	.WORD	EM60,DH60,DT60,DF60	:ITEM 60
002242	054274	071563	073312	.WORD	EM61,DH61,DT61,DF61	:ITEM 61
002252	054771	071274	073270	.WORD	EM62,DH62,DT62,DF62	:ITEM 62
002262	055067	071340	073270	.WORD	EM63,DH63,DT63,DF63	:ITEM 63
002272	055104	071447	073230	.WORD	EM64,DH64,DT64,DF64	:ITEM 64
002302	055154	071274	073270	.WORD	EM65,DH65,DT65,DF65	:ITEM 65
002312	055200	071404	073206	.WORD	EM66,DH66,DT66,DF66	:ITEM 66
002322	055226	071274	073206	.WORD	EM67,DH67,DT67,DF67	:ITEM 67
002332	055310	071340	073206	.WORD	EM70,DH70,DT70,DF70	:ITEM 70
002342	055333	071447	073432	.WORD	EM71,DH71,DT71,DF71	:ITEM 71
002352	055350	071274	073206	.WORD	EM72,DH72,DT72,DF72	:ITEM 72
2362	055364	071447	073466	.WORD	EM73,DH73,DT73,DF73	:ITEM 73
002372	055400	071404	073206	.WORD	EM74,DH74,DT74,DF74	:ITEM 74
002402	055414	071274	073142	.WORD	EM75,DH75,DT75,DF75	:ITEM 75
002412	055430	071460	073256	.WORD	EM76,DH76,DT76,DF76	:ITEM 76
002422	055503	071447	073466	.WORD	EM77,DH77,DT77,DF77	:ITEM 77
002432	055516	071404	073206	.WORD	EM100,DH100,DT100,DF100	:ITEM 100
002442	055532	071274	073142	.WORD	EM101,DH101,DT101,DF101	:ITEM 101
002452	055546	071460	073256	.WORD	EM102,DH102,DT102,DF102	:ITEM 102
002462	055571	071447	073466	.WORD	EM103,DH103,DT103,DF103	:ITEM 103
002472	055604	071404	073206	.WORD	EM104,DH104,DT104,DF104	:ITEM 104
002502	055620	071274	073142	.WORD	EM105,DH105,DT105,DF105	:ITEM 105
002512	055634	071460	073256	.WORD	EM106,DH106,DT106,DF106	:ITEM 106
002522	055660	071460	073256	.WORD	EM107,DH107,DT107,DF107	:ITEM 107
002532	055674	071447	073466	.WORD	EM110,DH110,DT110,DF110	:ITEM 110
002542	055710	071404	073206	.WORD	EM111,DH111,DT111,DF111	:ITEM 111
002552	055724	071274	073142	.WORD	EM112,DH112,DT112,DF112	:ITEM 112
002562	055740	071460	073256	.WORD	EM113,DH113,DT113,DF113	:ITEM 113
002572	055764	071447	073466	.WORD	EM114,DH114,DT114,DF114	:ITEM 114
002602	056000	071404	073206	.WORD	EM115,DH115,DT115,DF115	:ITEM 115
002612	056014	071274	073142	.WORD	EM116,DH116,DT116,DF116	:ITEM 116
002622	056030	071460	073256	.WORD	EM117,DH117,DT117,DF117	:ITEM 117
002632	056053	071447	073466	.WORD	EM120,DH120,DT120,DF120	:ITEM 120
002642	056066	071404	073206	.WORD	EM121,DH121,DT121,DF121	:ITEM 121
002652	056102	071274	073142	.WORD	EM122,DH122,DT122,DF122	:ITEM 122
002662	056116	071460	073256	.WORD	EM123,DH123,DT123,DF123	:ITEM 123
002672	056142	071447	073466	.WORD	EM124,DH124,DT124,DF124	:ITEM 124
002702	056156	071404	073206	.WORD	EM125,DH125,DT125,DF125	:ITEM 125
002712	056172	071274	073142	.WORD	EM126,DH126,DT126,DF126	:ITEM 126
002722	056206	071460	073256	.WORD	EM127,DH127,DT127,DF127	:ITEM 127
002732	056232	071447	073466	.WORD	EM130,DH130,DT130,DF130	:ITEM 130
002742	056246	071274	073142	.WORD	EM131,DH131,DT131,DF131	:ITEM 131
002752	056262	071460	073256	.WORD	EM132,DH132,DT132,DF132	:ITEM 132
002762	056307	071447	073466	.WORD	EM133,DH133,DT133,DF133	:ITEM 133
002772	056322	071274	073142	.WORD	EM134,DH134,DT134,DF134	:ITEM 134
003002	056336	071447	073230	.WORD	EM135,DH135,DT135,DF135	:ITEM 135
003012	056406	071447	073230	.WORD	EM136,DH136,DT136,DF136	:ITEM 136
003022	056422	071274	073270	.WORD	EM137,DH137,DT137,DF137	:ITEM 137
003032	056436	071447	073230	.WORD	EM140,DH140,DT140,DF140	:ITEM 140
003042	056452	071404	073206	.WORD	EM141,DH141,DT141,DF141	:ITEM 141
003052	056510	071274	073206	.WORD	EM142,DH142,DT142,DF142	:ITEM 142
003062	056524	071447	073230	.WORD	EM143,DH143,DT143,DF143	:ITEM 143
003072	056540	071404	073206	.WORD	EM144,DH144,DT144,DF144	:ITEM 144
003102	056556	071274	073206	.WORD	EM145,DH145,DT145,DF145	:ITEM 145
003112	056572	071447	073230	.WORD	EM146,DH146,DT146,DF146	:ITEM 146

003122	056606	071404	073206	.WORD	EM147, DH147, DT147, DF147	:ITEM 147
003132	056626	071274	073206	.WORD	EM150, DH150, DT150, DF150	:ITEM 150
003142	056642	071447	073230	.WORD	EM151, DH151, DT151, DF151	:ITEM 151
003152	056656	071404	073206	.WORD	EM152, DH152, DT152, DF152	:ITEM 152
003162	056676	071274	073206	.WORD	EM153, DH153, DT153, DF153	:ITEM 153
003172	056712	071447	073230	.WORD	EM154, DH154, DT154, DF154	:ITEM 154
003202	056726	071404	073206	.WORD	EM155, DH155, DT155, DF155	:ITEM 155
003212	056746	071274	073206	.WORD	EM156, DH156, DT156, DF156	:ITEM 156
003222	056762	071447	073230	.WORD	EM157, DH157, DT157, DF157	:ITEM 157
003232	057005	071404	073206	.WORD	EM160, DH160, DT160, DF160	:ITEM 160
003242	057053	071274	073206	.WORD	EM161, DH161, DT161, DF161	:ITEM 161
003252	057066	071447	073230	.WORD	EM162, DH162, DT162, DF162	:ITEM 162
003262	057112	071274	073206	.WORD	EM163, DH163, DT163, DF163	:ITEM 163
003272	057126	071520	073206	.WORD	EM164, DH164, DT164, DF164	:ITEM 164
003302	057156	071563	073312	.WORD	EM165, DH165, DT165, DF165	:ITEM 165
003312	057172	071563	073312	.WORD	EM166, DH166, DT166, DF166	:ITEM 166
003322	057206	071563	073312	.WORD	EM167, DH167, DT167, DF167	:ITEM 167
003332	057222	071563	073312	.WORD	EM170, DH170, DT170, DF170	:ITEM 170
003342	057236	071563	073312	.WORD	EM171, DH171, DT171, DF171	:ITEM 171
003352	057252	071563	073312	.WORD	EM172, DH172, DT172, DF172	:ITEM 172
003362	057266	071626	073356	.WORD	EM173, DH173, DT173, DF173	:ITEM 173
003372	057302	071626	073356	.WORD	EM174, DH174, DT174, DF174	:ITEM 174
003402	057316	071626	073356	.WORD	EM175, DH175, DT175, DF175	:ITEM 175
003412	057332	071274	073206	.WORD	EM176, DH176, DT176, DF176	:ITEM 176
003422	057354	071677	073422	.WORD	EM177, DH177, DT177, DF177	:ITEM 177
003432	057410	071563	073312	.WORD	EM200, DH200, DT200, DF200	:ITEM 200
003442	057463	071563	073312	.WORD	EM201, DH201, DT201, DF201	:ITEM 201
003452	057534	071563	073312	.WORD	EM202, DH202, DT202, DF202	:ITEM 202
003462	057606	071563	073312	.WORD	EM203, DH203, DT203, DF203	:ITEM 203
003472	057730	071563	073312	.WORD	EM204, DH204, DT204, DF204	:ITEM 204
003502	060003	071563	073312	.WORD	EM205, DH205, DT205, DF205	:ITEM 205
003512	060054	071563	073312	.WORD	EM206, DH206, DT206, DF206	:ITEM 206
003522	060126	071563	073312	.WORD	EM207, DH207, DT207, DF207	:ITEM 207
003532	060200	071563	073312	.WORD	EM210, DH210, DT210, DF210	:ITEM 210
003542	060252	071563	073312	.WORD	EM211, DH211, DT211, DF211	:ITEM 211
003552	060331	071563	073312	.WORD	EM212, DH212, DT212, DF212	:ITEM 212
003562	060402	071563	073312	.WORD	EM213, DH213, DT213, DF213	:ITEM 213
003572	060513	071563	073312	.WORD	EM214, DH214, DT214, DF214	:ITEM 214
003602	060600	071520	073206	.WORD	EM215, DH215, DT215, DF215	:ITEM 215
003612	060735	071447	073230	.WORD	EM216, DH216, DT216, DF216	:ITEM 216
003622	060750	071404	073206	.WORD	EM217, DH217, DT217, DF217	:ITEM 217
003632	060770	071274	073206	.WORD	EM220, DH220, DT220, DF220	:ITEM 220
003642	061004	071520	073206	.WORD	EM221, DH221, DT221, DF221	:ITEM 221
003652	061054	071447	073230	.WORD	EM222, DH222, DT222, DF222	:ITEM 222
003662	061070	071404	073206	.WORD	EM223, DH223, DT223, DF223	:ITEM 223
003672	061110	071274	073206	.WORD	EM224, DH224, DT224, DF224	:ITEM 224
003702	061124	071404	073206	.WORD	EM225, DH225, DT225, DF225	:ITEM 225
003712	061147	071274	073206	.WORD	EM226, DH226, DT226, DF226	:ITEM 226
003722	061160	071726	073256	.WORD	EM227, DH227, DT227, DF227	:ITEM 227
003732	061211	071404	073206	.WORD	EM230, DH230, DT230, DF230	:ITEM 230
003742	061234	071274	073206	.WORD	EM231, DH231, DT231, DF231	:ITEM 231
003752	061246	071726	073256	.WORD	EM232, DH232, DT232, DF232	:ITEM 232
003762	061260	071404	073206	.WORD	EM233, DH233, DT233, DF233	:ITEM 233
003772	061304	071274	073206	.WORD	EM234, DH234, DT234, DF234	:ITEM 234
004002	061316	071726	073256	.WORD	EM235, DH235, DT235, DF235	:ITEM 235
004012	061330	071404	073206	.WORD	EM236, DH236, DT236, DF236	:ITEM 236
004022	061355	071274	073206	.WORD	EM237, DH237, DT237, DF237	:ITEM 237

004032	061366	071726	073256	.WORD	EM240,DH240,DT240,DF240	:ITEM 240
004042	061400	071404	073206	.WORD	EM241,DH241,DT241,DF241	:ITEM 241
004052	061425	071274	073206	.WORD	EM242,DH242,DT242,DF242	:ITEM 242
004062	061436	071726	073256	.WORD	EM243,DH243,DT243,DF243	:ITEM 243
004072	061450	071404	073206	.WORD	EM244,DH244,DT244,DF244	:ITEM 244
004102	061474	071274	073206	.WORD	EM245,DH245,DT245,DF245	:ITEM 245
004112	061506	071520	073206	.WORD	EM246,DH246,DT246,DF246	:ITEM 246
004122	061557	071726	073256	.WORD	EM247,DH247,DT247,DF247	:ITEM 247
004132	061570	071404	073206	.WORD	EM250,DH250,DT250,DF250	:ITEM 250
004142	061615	071274	073206	.WORD	EM251,DH251,DT251,DF251	:ITEM 251
004152	061506	071520	073206	.WORD	EM252,DH252,DT252,DF252	:ITEM 252
004162	061642	071726	073256	.WORD	EM253,DH253,DT253,DF253	:ITEM 253
004172	061506	071520	073206	.WORD	EM254,DH254,DT254,DF254	:ITEM 254
004202	061672	071726	073256	.WORD	EM255,DH255,DT255,DF255	:ITEM 255
004212	061716	071404	073206	.WORD	EM256,DH256,DT256,DF256	:ITEM 256
004222	061744	071274	073206	.WORD	EM257,DH257,DT257,DF257	:ITEM 257
004232	061756	071563	073312	.WORD	EM260,DH260,DT260,DF260	:ITEM 260
004242	062014	071563	073312	.WORD	EM261,DH261,DT261,DF261	:ITEM 261
004252	062026	071563	073312	.WORD	EM262,DH262,DT262,DF262	:ITEM 262
004262	062114	071563	073312	.WORD	EM263,DH263,DT263,DF263	:ITEM 263
004272	062141	071563	073312	.WORD	EM264,DH264,DT264,DF264	:ITEM 264
004302	062226	071563	073312	.WORD	EM265,DH265,DT265,DF265	:ITEM 265
004312	062277	071563	073312	.WORD	EM266,DH266,DT266,DF266	:ITEM 266
004322	062353	071563	073312	.WORD	EM267,DH267,DT267,DF267	:ITEM 267
004332	062440	071563	073312	.WORD	EM270,DH270,DT270,DF270	:ITEM 270
004342	062472	071563	073312	.WORD	EM271,DH271,DT271,DF271	:ITEM 271
004352	062531	071563	073312	.WORD	EM272,DH272,DT272,DF272	:ITEM 272
004362	062601	071563	073312	.WORD	EM273,DH273,DT273,DF273	:ITEM 273
004372	062636	071563	073312	.WORD	EM274,DH274,DT274,DF274	:ITEM 274
004402	062650	071563	073312	.WORD	EM275,DH275,DT275,DF275	:ITEM 275
004412	062733	071563	073312	.WORD	EM276,DH276,DT276,DF276	:ITEM 276
004422	063020	071563	073312	.WORD	EM277,DH277,DT277,DF277	:ITEM 277
004432	063065	071563	073312	.WORD	EM300,DH300,DT300,DF300	:ITEM 300
004442	063162	071563	073522	.WORD	EM301,DH301,DT301,DF301	:ITEM 301
004452	063207	071563	073522	.WORD	EM302,DH302,DT302,DF302	:ITEM 302
004462	063220	071626	073574	.WORD	EM303,DH303,DT303,DF303	:ITEM 303
004472	063232	071563	073522	.WORD	EM304,DH304,DT304,DF304	:ITEM 304
004502	063312	071563	073522	.WORD	EM305,DH305,DT305,DF305	:ITEM 305
004512	063406	071563	073522	.WORD	EM306,DH306,DT306,DF306	:ITEM 306
004522	063514	071563	073522	.WORD	EM307,DH307,DT307,DF307	:ITEM 307
004532	063564	071563	073522	.WORD	EM310,DH310,DT310,DF310	:ITEM 310
004542	063640	071563	073522	.WORD	EM311,DH311,DT311,DF311	:ITEM 311
004552	063710	071563	073522	.WORD	EM312,DH312,DT312,DF312	:ITEM 312
004562	063760	071563	073522	.WORD	EM313,DH313,DT313,DF313	:ITEM 313
004572	064030	071563	073522	.WORD	EM314,DH314,DT314,DF314	:ITEM 314
004602	064100	071563	073522	.WORD	EM315,DH315,DT315,DF315	:ITEM 315
004612	064150	071563	073522	.WORD	EM316,DH316,DT316,DF316	:ITEM 316
004622	064220	071563	073522	.WORD	EM317,DH317,DT317,DF317	:ITEM 317
004632	064270	071563	073522	.WORD	EM320,DH320,DT320,DF320	:ITEM 320
004642	064340	071563	073522	.WORD	EM321,DH321,DT321,DF321	:ITEM 321
004652	064410	071563	073646	.WORD	EM322,DH322,DT322,DF322	:ITEM 322
004662	064446	071563	073646	.WORD	EM323,DH323,DT323,DF323	:ITEM 323
004672	064460	071626	073712	.WORD	EM324,DH324,DT324,DF324	:ITEM 324
004702	064472	071563	073646	.WORD	EM325,DH325,DT325,DF325	:ITEM 325
004712	064472	071563	073646	.WORD	EM326,DH326,DT326,DF326	:ITEM 326
004722	064616	071563	073646	.WORD	EM327,DH327,DT327,DF327	:ITEM 327
004732	064666	071563	073646	.WORD	EM330,DH330,DT330,DF330	:ITEM 330

004742	064742	071563	073646	.WORD	EM331,DH331,DT331,DF331	:ITEM 331
004752	065012	071563	073646	.WORD	EM332,DH332,DT332,DF332	:ITEM 332
004762	064446	071563	073646	.WORD	EM333,DH333,DT333,DF333	:ITEM 333
004772	065110	071563	073646	.WORD	EM334,DH334,DT334,DF334	:ITEM 334
005002	065173	071563	073646	.WORD	EM335,DH335,DT335,DF335	:ITEM 335
005012	065242	071563	073646	.WORD	EM336,DH336,DT336,DF336	:ITEM 336
005022	065277	071563	073646	.WORD	EM337,DH337,DT337,DF337	:ITEM 337
005032	065353	071563	073646	.WORD	EM340,DH340,DT340,DF340	:ITEM 340
005042	065422	071563	073646	.WORD	EM341,DH341,DT341,DF341	:ITEM 341
005052	065472	071563	073646	.WORD	EM342,DH342,DT342,DF342	:ITEM 342
005062	065542	071563	073646	.WORD	EM343,DH343,DT343,DF343	:ITEM 343
005072	065617	071563	073646	.WORD	EM344,DH344,DT344,DF344	:ITEM 344
005102	065724	071563	073646	.WORD	EM345,DH345,DT345,DF345	:ITEM 345
005112	065774	071563	073646	.WORD	EM346,DH346,DT346,DF346	:ITEM 346
005122	066073	071563	073646	.WORD	EM347,DH347,DT347,DF347	:ITEM 347
005132	066117	071563	073646	.WORD	EM350,DH350,DT350,DF350	:ITEM 350
005142	066130	071460	073256	.WORD	EM351,DH351,DT351,DF351	:ITEM 351
005152	066232	071563	073646	.WORD	EM352,DH352,DT352,DF352	:ITEM 352
005162	066302	071563	073646	.WORD	EM353,DH353,DT353,DF353	:ITEM 353
005172	066352	071563	073646	.WORD	EM354,DH354,DT354,DF354	:ITEM 354
005202	066422	071563	073646	.WORD	EM355,DH355,DT355,DF355	:ITEM 355
005212	066472	071404	073142	.WORD	EM356,DH356,DT356,DF356	:ITEM 356
005222	066576	071746	073164	.WORD	EM357,DH357,DT357,DF357	:ITEM 357
005232	066632	071460	073256	.WORD	EM360,DH360,DT360,DF360	:ITEM 360
005242	066722	071274	073522	.WORD	EM361,DH361,DT361,DF361	:ITEM 361
005252	066740	072012	073756	.WORD	EM362,DH362,DT362,DF362	:ITEM 362
005262	067050	072034	073774	.WORD	EM363,DH363,DT363,DF363	:ITEM 363
005272	067116	072074	074016	.WORD	EM364,DH364,DT364,DF364	:ITEM 364
005302	067216	072143	073256	.WORD	EM365,DH365,DT365,DF365	:ITEM 365
005312	067301	072154	074030	.WORD	EM366,DH366,DT366,DF366	:ITEM 366
005322	067364	072206	074076	.WORD	EM367,DH367,DT367,DF367	:ITEM 367
005332	067447	072253	074122	.WORD	EM370,DH370,DT370,DF370	:ITEM 370
005342	067503	072316	074166	.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	067633	071404	073206	.WORD	EM401,DH401,DT401,DF401	:ITEM 401
005452	067655	071274	073206	.WORD	EM402,DH402,DT402,DF402	:ITEM 402
005462	067666	071460	073256	.WORD	EM403,DH403,DT403,DF403	:ITEM 403
005472	070016	071726	073256	.WORD	EM404,DH404,DT404,DF404	:ITEM 404
005502	070030	071404	073206	.WORD	EM405,DH405,DT405,DF405	:ITEM 405
005512	070044	071274	073206	.WORD	EM406,DH406,DT406,DF406	:ITEM 406
005522	070060	071460	073256	.WORD	EM407,DH407,DT407,DF407	:ITEM 407
005532	070130	071726	073256	.WORD	EM410,DH410,DT410,DF410	:ITEM 410
005542	070144	071404	073206	.WORD	EM411,DH411,DT411,DF411	:ITEM 411
005552	070170	071274	073206	.WORD	EM412,DH412,DT412,DF412	:ITEM 412
005562	070202	071460	073256	.WORD	EM413,DH413,DT413,DF413	:ITEM 413
005572	070252	071726	073256	.WORD	EM414,DH414,DT414,DF414	:ITEM 414
005602	070264	071404	073206	.WORD	EM415,DH415,DT415,DF415	:ITEM 415
005612	070311	071274	073206	.WORD	EM416,DH416,DT416,DF416	:ITEM 416
005622	070322	071460	073256	.WORD	EM417,DH417,DT417,DF417	:ITEM 417
005632	070366	071726	073256	.WORD	EM420,DH420,DT420,DF420	:ITEM 420
005642	070400	071404	073206	.WORD	EM421,DH421,DT421,DF421	:ITEM 421

005652	070425	071274	073206	.WORD	EM422,DH422,DT422,DF422	:ITEM 422
005662	070436	071460	073256	.WORD	EM423,DH423,DT423,DF423	:ITEM 423
005672	070450	071726	073256	.WORD	EM424,DH424,DT424,DF424	:ITEM 424
005702	070462	071404	073206	.WORD	EM425,DH425,DT425,DF425	:ITEM 425
005712	070506	071274	073206	.WORD	EM426,DH426,DT426,DF426	:ITEM 426
005722	070520	071460	073256	.WORD	EM427,DH427,DT427,DF427	:ITEM 427
005732	070572	071726	073256	.WORD	EM430,DH430,DT430,DF430	:ITEM 430
005742	070604	071460	073256	.WORD	EM431,DH431,DT431,DF431	:ITEM 431
005752	070632	071404	073206	.WORD	EM432,DH432,DT432,DF432	:ITEM 432
005762	070657	071274	073206	.WORD	EM433,DH433,DT433,DF433	:ITEM 433
005772	070670	071460	073256	.WORD	EM434,DH434,DT434,DF434	:ITEM 434
006002	070740	071726	073256	.WORD	EM435,DH435,DT435,DF435	:ITEM 435
006012	070752	071460	073256	.WORD	EM436,DH436,DT436,DF436	:ITEM 436
006022	071000	071404	073206	.WORD	EM437,DH437,DT437,DF437	:ITEM 437
006032	071022	071404	073206	.WORD	EM440,DH440,DT440,DF440	:ITEM 440
006042	071044	072412	074210	.WORD	EM441,DH441,DT441,DF441	:ITEM 441
006052	071044	072143	074226	.WORD	EM442,DH442,DT442,DF442	:ITEM 442
006062	071044	072143	074226	.WORD	EM443,DH443,DT443,DF443	:ITEM 443
006072	071171	071340	073206	.WORD	EM444,DH444,DT444,DF444	:ITEM 444

2040
 2041
 2042

.SBTTL ACT11 HOOKS
 :*****
 :HOOKS REQUIRED BY ACT11

	006102			\$SVPC=.	:SAVE PC
	000046	046104		=46	
	000052			\$ENDAD	::1)SET LOC.46 TO ADDRESS OF \$ENDAD IN .SEOP
	000052	000000		=52	
	006102			.WORD 0	::2)SET LOC.52 TO ZERO
				=\$SVPC	:: RESTORE PC

2043

.SBTTL APT PARAMETER BLOCK
 :*****
 :SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
 :*****

	006102			.\$X=.	::SAVE CURRENT LOCATION
	000024	000200		=24	::SET POWER FAIL TO POINT TO START OF PROGRAM
	000044	006102		200	::FOR APT START UP
	006102	006102		=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.

	006102			\$APTHD:	
	006102	000000		\$HIBTS: .WORD 0	::TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
	006104	001316		\$MBADR: .WORD \$MAIL	::ADDRESS OF APT MAILBOX (BITS 0-15)
	006106	000010		\$STM: .WORD 10	::RUN TIM OF LONGEST TEST
	006110	000040		\$PASTM: .WORD 40	::RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
	006112	000000		\$UNITM: .WORD 0	::ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
	006114	000052		.WORD	SETEND-\$MAIL/2 ::LENGTH MAILBOX-ETABLE(WORDS)

2044
 2045
 2046

006116

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

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 046170 000020  MOV      #SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
006144 012737 000340 000022  MOV      #340,@#IOTVEC+2 ;;LEVEL 7
006152 012737 046514 000030  MOV      #ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
006160 012737 000340 000032  MOV      #340,@#EMTVEC+2 ;;LEVEL 7
006166 012737 050704 000034  MOV      #STRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
006174 012737 000340 000036  MOV      #340,@#TRAPVEC+2;LEVEL 7
006202 012737 050766 000024  MOV      #SPWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
006210 012737 000340 000026  MOV      #340,@#PWRVEC+2 ;;LEVEL 7
006216 013737 045464 045452  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 046150 000014  MOV      #RTRN,@#TBITVEC ;;SET 'T' BIT VECTOR TO RTRN
006250 012737 000340 000016  MOV      #340,@#TBITVEC+2 ;;LEVEL 7
006256 012737 045210 046150  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$,-(SP)  ;;AND PC
006300 000006              RTT                ;;TRY THE RTT
006302 012737 000006 046150 64$: MOV      #RTT,$RTRN  ;;RTT IS LEGAL--SET RTRN TO A RTT
006310 000402              BR                ;;
006312 062706 000010 65$: ADD      #10,SP   ;;RTT ILLEGAL--CLEAN OFF THE STACK
006316 012737 000012 000010 66$: MOV      #RESVEC+2,@#RESVEC ;;RESTORE TRAP CATCHER
006324 005037 046156      CLR      $TBIT      ;;CLEAR 'T' BIT SWITCH
006330 012737 006330 001106  MOV      #.,$LPADR   ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
006336 012737 006336 001110  MOV      #.,$LPERR   ;;SETUP THE ERROR LOOP ADDRESS
                                ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                ;;EQUAL TO A '-1', SETUP FOR A SOFTWARE SWITCH REGISTER.
006344 013746 000004      MOV      @#ERRVEC,-(SP) ;;AVE ERROR VECTOR
006350 012737 006404 000004  MOV      #67$,@#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 001C12              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                ;;
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      $PASS     ;;CLEAR PASS COUNT
006436 132737 000200 001337  BITB     #APTSIZE,$EVM ;;TEST USER SIZE UNDER APT
006444 001403              BEQ      70$      ;;YES,USE NON-APT SWITCH
006446 012737 001340 001140  MOV      #SSWREG,$X  ;;NO,USE APT SWITCH REGISTER
006454
2047
                                .SBTTL TYPE PROGRAM NAME
                                ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
006454 005227 177777      INC      #-1        ;;FIRST TIME?
006460 001047              BNE      71$        ;;BRANCH IF NO
006462 022737 046104 000042  CMP      #SENDAD,@#42 ;;ACT-11?

```

```

006470 001443
006472 104401 006540
006476 005737 000042
006502 001012
006504 123727 001336 000001
006512 001406
006514 023727 001140 000176
006522 001005
006524 104405
006526 000403
006530 112737 000001 001134
006536
006536 000420
006600
2048 006600 104401 006606
006604 000434
006676
2049 006676 104401 006704
006702 000426
006760
2050 006760 005037 046164
2051 006764

```

```

      BEQ 71$          ;;BRANCH IF YES
      TYPE 72$        ;;TYPE ASCIZ STRING
      .SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
      TST 74$         ;;ARE WE RUNNING UNDER XXDP/ACT?
      BNE 73$         ;;BRANCH IF YES
      CMPB $ENV,#1    ;;ARE WE RUNNING UNDER APT?
      BEQ 73$         ;;BRANCH IF YES
      CMP SWR,#SWREG ;;SOFTWARE SWITCH REG SELECTED?
      BNE 74$         ;;BRANCH IF NO
      GTSWR           ;;GET SOFT-SWR SETTINGS
      BR 74$
      MOVB #1,$AUTOB ;;SET AUTO-MODE INDICATOR
      BR 71$          ;;GET OVER THE ASCIZ
      .ASCIZ <CRLF>*CKFPCDO FP11F FLTG PNT PRT C*<CRLF>
      TYPE 76$        ;;TYPE ASCIZ STRING
      BR 75$         ;;GET OVER THE ASCIZ
      .ASCIZ !EOP MESSAGE WILL PRINT EVERY 1000 PASSES (15 SECONDS)!<CRLF>
      TYPE 78$        ;;TYPE ASCIZ STRING
      BR 77$         ;;GET OVER THE ASCIZ
      .ASCIZ !HIT ANY KEY TO ENABLE/DISABLE EOP MESSAGES!<CRLF>
      CLR EPENDS      ;CLR EOP FLAG INITIALLY ENABLING EOP'S ;DPM002
      LOOP:

```


2057

```
.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.
:*
:*****
```

```
2058 006764 000004
2059 006766 012737 006774 001110
2060 006774 005000
2061 006776 170100
2062 007000 012737 007036 000244
2063 007006 012737 007014 001236
2064
2065 007014 174007
2066
2067
2068
2069
2070 007016 170200
2071 007020 010037 001240
2072 007024 170300
2073 007026 010037 001242
2074 007032 104001
2075
2076 007034 000434
2077
2078
2079 007036 011600
2080 007040 022700 007016
2081 007044 001402
2082 007046 000137 051732
2083
2084
2085 007052 170204
2086 007054 170305
2087 007056 010437 001240
2088 007062 010537 001242
2089 007066 012702 100000
2090 007072 012703 000002
2091 007076 010237 001244
2092 007102 010337 001246
2093 007106 022626
2094
2095 007110 020204
2096 007112 001402
2097
2098 007114 104002
2099 007116 000403
2100
2101 007120 020305
2102 007122 001401
2103
2104 007124 104003
2105
2106 007126

TST1: SCOPE
1000$: MOV #1000$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
        CLR R0 ;SET THE FPS.
        LDFPS R0
        MOV #200$, FPVECT ;SET UP FOR FP TRAPS.
        MOV #1$, $TMP2
1$: STF ACO, AC7 ;THIS TEST INSTRUCTION SHOULD
        ;CAUSE A TRAP.
        ;REPORT FAILURE OF USE OF ILLEGAL ACCUMULATOR 7 TO CAUSE AN FPP TRAP.
210$: STFPS R0 ;GET FPS.
        MOV R0, $TMP3
        STST R0 ;GET FEC.
        MOV R0, $TMP4
        ERROR +1 ;STF WITH ILLEGAL ACCUMULATOR, MODE
        ;0, DIDN'T TRAP. ST 765 TO ST 537.
        BR 240$
;TRAP TO 200$, HERE, WHEN THE EXPECTED ERROR OCCURS.
200$: MOV (SP), R0 ;MAKE SURE THE ERROR OCCURRED
        CMP #210$, R0 ;AT THE CORRECT ADDRESS.
        BEQ 220$ ;BRANCH IF TRAP ADDRESS CORRECT.
        JMP FPSPUR ;IF INCORRECT GO REPORT SPURIOUS
        ;FP TRAP.
220$: STFPS R4 ;GET FPS.
        STST R5 ;GET FEC.
        MOV R4, $TMP3 ;SAVE DATA INCASE OF ERROR.
        MOV R5, $TMP4
        MOV #100000, R2 ;EXPECTED FPS
        MOV #2, R3 ;EXPECTED FEC
        MOV R2, $TMP5
        MOV R3, $TMP6
        CMP (SP)+, (SP)+ ;RESET THE STACK.
        CMP R2, R4 ;WAS FPS CORRECT?
        BEQ 230$ ;BRANCH IF YES.
        ;OTHERWISE REPORT FPS INCORRECTLY
        ;SET AFTER USE OF ILLEGAL ACC.
        ERROR +2
        BR 240$
230$: CMP R3, R5 ;WAS THE FEC CORRECT?
        BEQ 240$ ;BRANCH IF CORRECT.
        ;OTHERWISE REPORT INCORRECT FEC
        ;AFTER USE OF ILLEGAL ACC.
        ERROR +3
240$:
```

007126 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?).

2112

```
.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.
:*
:*****
```

```

2113 007130 000004          TST2:  SCOPE
2113 007132 012737 007140 001110  MOV    #1000$,SLPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
2114 007140 012700 177777 1000$:  MOV    #-1,R0        ;SET UP A BACKGROUND PATTERN IN THE
2115 007144 012701 007274      MOV    #200$,R1      ;INPUT BUFFER.
2116 007150 012702 000014      MOV    #14,R2
2117 007154 010021 210$:   MOV    R0,(R1)+
2118 007156 077202      SOB    R2,210$
2119
2120 007160 012700 000200      MOV    #200,R0      ;SET FD MODE.
2121 007164 170100      LDFPS  R0
2122 007166 012700 007324      MOV    #220$,R0     ;PUT TEST DATA INTO ACO.
2123 007172 172410      LDD    (R0),ACO
2124
2125 007174 012700 007310      MOV    #230$,R0     ;FDST ADDRESS.
2126 007200 005002      CLR    R2           ;CLEAR THE FPS.
2127 007202 170102      LDFPS  R2
2128 007204 012737 007216 001236  MOV    #240$,STMP2
2129 007212 010037 001240      MOV    R0,STMP3
2130
2131 007216 174010      240$:  STF    ACO,(R0)    ;TEST INSTRUCTION.
2132
2133 007220 022700 007310      CMP    #230$,R0     ;WAS R0 MODIFIED DURING EXECUTION?
2134 007224 001404      BEQ    245$        ;BRANCH IF R0 NOT MODIFIED, CORRECT.
2135
2136 007226 010037 001242      MOV    R0,STMP4     ;OTHERWISE REPORT ERROR, R0 MODIFIED.
2137 007232 104004      ERROR  +4
2138 007234 000456      BR     250$        ;GO TO NEXT TEST.
2139
2140 007236 012700 007310      245$:  MOV    #230$,R0     ;CHECK THE DATA IN THE OUTPUT BUFFER.
2141 007242 012701 007324      MOV    #220$,R1
2142 007246 022021      CMP    (R0)+,(R1)+
2143 007250 001031      BNE    260$        ;BRANCH IF INCORRECT.
2144 007252 022011      CMP    (R0)+,(R1)
2145 007254 001027      BNE    260$        ;BRANCH IF INCORRECT.
2146 007256 022720 177777      CMP    #-1,(R0)+   ;WAS FLOATING MODE USED?
2147 007262 001034      BNE    270$        ;BRANCH IF NOT.
2148 007264 022710 177777      CMP    #-1,(R0)
2149 007270 001031      BNE    270$
2150 007272 000437      BR     250$        ;GO TO NEXT TEST.
2151
2152 007274 177777 177777 177777 200$:  .WORD  -1,-1,-1,-1,-1,-1
2153
2154 007310 177777 177777 177777 230$:  .WORD  -1,-1,-1,-1,-1,-1
2155
2156 007324 123456 023456      220$:  .WORD  123456,23456
2157 007330 034567 045671      .WORD  34567,45671
2158
2159      ;REPORT DATA IN OUT PUT BUFFER INCORRECT.
2160 007334 012737 007324 001242 260$:  MOV    #220$,STMP4
2161 007342 012737 007310 001240      MOV    #230$,STMP3
```

```
2162 007350 104005          ERROR +5          ;BAD DATA.
2163 007352 000407          BR      250$
2164
2165          ;REPORT FLOATING MODE NOT USED, BUT FD FAILED.
2166 007354 012737 007324 001242 270$: MOV #220$,STMP4
2167 007362 C'2737 007310 001240  MOV #230$,STMP3
2168 007370 104006          ERROR +6          ;ST 707 TO 245 INTO 244 (BUT FD).
2169
2170          250$: 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?).
```

2176

```
.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.
:*
:*****
```

```
TST3: SCOPE
:FIRST TEST STF.
2177 007374 000004
2178 007376 012737 007404 001110  MOV #1000$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
2179 007404 012700 177777 1000$: MOV #-1, R0 ;SET UP THE OUTPUT BUFFER.
2180 007410 012701 007542  MOV #200$, R1
2181 007414 012702 000014  MOV #14, R2
2182 007420 010021 210$: MOV R0, (R1)+
2183 007422 077202  SOB R2, 210$
2184
2185 007424 012700 000200  MOV #200, R0 ;SET FD MODE.
2186 007430 170100  LDFPS R0
2187 007432 012700 007572  MOV #220$, R0 ;SETUP ACO.
2188 007436 172410  LDD (R0), ACO
2189
2190 007440 012700 007556  MOV #230$, R0 ;FDST ADDRESS.
2191 007444 005002  CLR R2
2192 007446 170102  LDFPS R2 ;SET FPS.
2193 007450 012737 007456 001236  MOV #240$, $TMP2
2194
2195 007456 174020 240$: STF ACO, (R0)+ ;TEST INSTRUCTION.
2196
2197 007460 022700 007562  CMP #230$+4, R0 ;WAS R0 INCREMENTED BY 4 PROPERLY?
2198
2199 007464 001407  BEQ 250$ ;BRANCH IF R0 CORRECT.
2200 007466 010037 001242  MOV R0, $TMP4 ;REPORT R0 INCORRECT AFTER FDST MODE 2.
2201 007472 012737 007562 001240  MOV #230$+4, $TMP3
2202 007500 104007  ERROR +7 ;BAD CONSTANT USED OR DIDN'T GO 527 TO 642
2203 007502 000530  BR 260$
2204 007504 012700 007556 250$: MOV #230$, R0 ;WAS THE OUTPUT DATA CORRECT?
2205 007510 012701 007572  MOV #220$, R1
2206 007514 022021  CMP (R0)+, (R1)+
2207 007516 001031  BNE 270$ ;BRANCH IF INCORRECT.
2208 007520 022021  CMP (R0)+, (R1)+
2209 007522 001027  BNE 270$ ;BRANCH IF INCORRECT.
2210 007524 022027 177777  CMP (R0)+, #-1 ;SEE IF ANY OTHER DATA BUFFER WORDS WERE MODIFIED.
2211 007530 001024  BNE 270$ ;BRANCH IF INCORRECT.
2212 007532 022027 177777  CMP (R0)+, #-1
2213 007536 001021  BNE 270$ ;BRANCH IF INCORRECT.
2214 007540 000433  BR 280$
2215 007542 177777 177777 200$: .WORD -1, -1, -1, -1, -1, -1
2216 007556 177777 177777 230$: .WORD -1, -1, -1, -1, -1, -1
2217 007572 076543 220$: 76543
2218 007574 065432 65432
2219 007576 054321 54321
2220 007600 043210 43210
2221
2222 007602 012737 007572 001240 ;REPORT OUTPUT DATA INCORRECT:
2223 007610 012737 007556 001242 270$: MOV #220$, $TMP3
2224 007616 104010  ERROR +10 ;BAD DATA
2225 007620 000461  BR 260$
```



```

2265 .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.
:
.....
2266 007766 000004 007776 001110 TST4: SCOPE
2267 007770 012737 010054 001110 MOV #200$,SLPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
2268 010002 012701 010122 200$: MOV #210$,R0 ;SET UP THE DATA BUFFER FOLLOWING THE TEST INSTRUCTION.
2269 010006 012702 000004 MOV #220$,R1
2270 010012 012021 1$: MOV #4,R2
2271 010014 077202 SOB (R0)+,(R1)+
2272 010016 012700 000200 MOV R2,1$ ;ENTER FLOATING DOUBLE MODE.
2273 010022 170100 LDFPS R0
2274 010024 012700 010132 MOV #230$,R0 ;SET UP ACO.
2275 010030 172410 LDD (R0),ACO
2276 010032 012737 010152 000004 MOV #240$,ERRVECT ;SET UP FOR AN ODD ADDRESS.
2277 010040 012737 010052 001236 MOV #250$,STMP2
2278 010046 005001 CLR R1
2279 010050 005004 CLR R4
2280 ;THIS IS THE TEST INSTRUCTION. IT SHOULD MODIFY THE FIRST LOCATION
2281 ;AFTER IT TO BE AN INCREMENT R4, INC R4, INSTRUCTION INSTEAD
2282 ;OF AN INCREMENT R1 INSTRUCTION. THE INCREMENT R4 SHOULD NOT BE
2283 ;EXECUTED SINCE THE PC SHOULD BE INCREMENTED BY TWO DURING IMMEDIATE
2284 ;MODE ADDRESSING. THUS AFTER THE EXECUTION OF THE NEXT 5 INSTRUCTIONS
2285 ;R1 SHOULD CONTAIN 3 AND R4 SHOULD CONTAIN 0.
2286 010052 174027 250$: STD ACO,(R7)+ ;TEST INSTRUCTION.
2287 010054 005201 210$: INC R1 ;THE STD INSTRUCTION SHOULD CHANGE THIS TO INC R4.
2288 010056 005201 INC R1
2289 010060 005201 INC R1
2290 010062 005201 INC R1
2291 010064 012700 0101+2 MOV #260$,R0 ;SEE IF THE DATA WAS OUTPUT CORRECTLY.
2292 010070 012702 010054 MOV #210$,R2
2293 010074 012703 000004 MOV #4,R3
2294 010100 022022 280$: CMP (R0)+,(R2)+
2295 010102 001051 BNE 270$ ;BRANCH IF INCORRECT.
2296 010104 077303 SOB R3,280$
2297 010106 005704 TST R4 ;MAKE SURE R4 IS 0.
2298 010110 001056 BNE 290$ ;BRANCH IF R4 IS INCORRECT.
2299 010112 022701 000003 CMP #3,R1 ;SEE IF R1 IS CORRECT.
2300 010116 001053 BNE 290$ ;BRANCH IF R1 IS INCORRECT.
2301 010120 000474 BR 300$
2302 ;THESE ARE TEST DATA PATTERNS USED TO SET UP THE OUTPUT BUFFER AT 210$.
2303 010122 005201 220$: INC R1
2304 010124 005201 INC R1
2305 010126 005201 INC R1
2306 010130 005201 INC R1
2307 ;THIS IS THE DATA PUT IN ACO BEFORE EXECUTION OF THE STD.
2308 010132 005204 230$: INC R4
2309 010134 005204 INC R4
2310 010136 005204 INC R4
2311 010140 005204 INC R4
2312 ;THIS IS THE EXPECTED DATA AT 210$ AFTER EXECUTION OF THE STD.
2313 010142 005204 260$: INC R4
2314 010144 005201 INC R1
    
```

```

2315 010146 005201          INC      R1
2316 010150 005201          INC      R1
2317          ;IF A FAILURE IN THE FDST FLOWS RESULTS IN AN ODD ADDRESS TRAP THROUGH
2318          ;4 TO HERE:
2319 010152 011602          240$: MOV      (SP),R2          ;SEE IF THE TRAP WAS BECAUSE OF AN ODD ADDRESS.
2320 010154 032702 000001    BIT      #1,R2
2321 010160 001005          BNE     310$          ;BRANCH IF YES.
2322 010162 020227 010056    CMP      R2,#210$+2    ;SEE IF THE TRAP OCCURRED AT THE TEST INSTRUCTION.
2323 010166 001412          BEQ     320$          ;BRANCH IF YES.
2324 010170 000137 051774    JMP     CPSPUR        ;OTHERWISE REPORT A SPURIOUS TRAP THROUGH VECTOR 4.
2325          ;REPORT A FAILURE IN THE FDST FLOWS RESULTED IN AN ODD ADDRESS TRAP.
2326 010174 010237 001236    310$: MOV      R2,$TMP2
2327 010200 012737 010056 001240  MOV      #210$+2,$TMP3
2328 010206 022626          CMP      (SP)+,(SP)+
2329 010210 104013          ERROR   +13          ;BAD CONSTANT #2 + PC ODD ADDR.
2330 010212 000437          BR      300$
2331 010214 010237 001236    320$: MOV      R2,$TMP2
2332 010220 022626          CMP      (SP)+,(SP)+
2333 010222 104014          ERROR   +14          ;ODD ADDRESS TRAP
2334 010224 000432          BR      300$          ;WRONG MODE USED.
2335
2336          ;REPORT DATA INCORRECT:
2337 010226 012737 010054 001240  270$: MOV      #210$,$TMP3
2338 010234 012737 010142 001242  MOV      #260$,$TMP4
2339 010242 104015          ERROR   +15          ;BAD DATA BUT GR7 FAIL
2340 010244 000422          BR      300$
2341
2342          ;REPORT PC INCORRECT MODIFIED DURING THE EXECUTION OF FDST IMMEDIATE
2343          ;MODE. THE PC SHOULD HAVE BEEN INCREMENTED BY 2 BUT IT WASN'T.
2344          ;USE R1 AND R4 TO COMPUTE THE ACTUAL ACTION THAT WAS TAKEN ON THE PC.
2345 010246 012737 010056 001240  290$: MOV      #210$+2,$TMP3
2346 010254 005704          TST     R4          ;IS R4 CLEAR.
2347 010256 001404          BEQ     100$
2348 010260 012737 010054 001242  MOV      #210$,$TMP4
2349 010266 000410          BR      110$
2350 010270 012702 010056          100$: MOV      #210$+2,R2
2351 010274 062701 177775    ADD     #-3,R1
2352 010300 006301          ASL     R1
2353 010302 160102          SUB     R1,R2
2354 010304 010237 001242    MOV      R2,$TMP4
2355 010310 104016          110$: ERROR   +16          ;BAD CONSTANT PC+
2356 010312          300$: 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?).
    
```


2362

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

2363 010314 000004
2364 010316 012737 010324 001110
2365 010324 012700 177777
2366 010330 012701 010460
2367 010334 012702 000010
2368 010340 010021
2369 010342 077202
2370 010344 012700 000200
2371 010350 170100
2372 010352 012700 010500
2373 010356 172410
2374 010360 012737 010520 000004
2375 010366 012737 010400 001236
2376 010374 012700 010470
2377 010400 174040
2378 010402 005201
2379 010404 020027 010460
2380 010410 001060
2381 010412 012700 010460
2382 010416 012701 010500
2383 010422 012702 000004
2384 010426 022021
2385 010430 001057
2386 010432 077203
2387 010434 012700 177777
2388 010440 012701 010470
2389 010444 012702 000004
2390 010450 020021
2391 010452 001056
2392 010454 077203
2393 010456 000463
2394
2395
2396 010460 177777
2397 010462 177777
2398 010464 177777
2399 010466 177777
2400 010470 177777
2401 010472 177777
2402 010474 177777
2403 010476 177777
2404
2405
2406 010500 147250
2407 010502 036147
2408 010504 025036
2409 010506 147250
2410 010510 177777
2411 010512 177777

TST5: SCOPE
200\$: MOV #200\$,SLPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
MOV #-1,R0 ;SET UP THE OUTPUT BUFFER.
MOV #210\$,R1
MOV #10,R2
1\$: MOV R0,(R1)+
SOB R2,1\$
MOV #200,R0 ;ENTER FLOATING DOUBLE MODE.
LDFPS R0
MOV #220\$,R0 ;SET UP ACO.
LDD (R0),ACO
MOV #240\$,ERRVECT ;SET UP FOR A TRAP TO 4.
MOV #230\$,STMP2
MOV #250\$,R0 ;SET UP THE DESTINATION ADDRESS.
230\$: STD ACO,-(R0) ;TEST INSTRUCTION.
INC R1
CMP R0,#210\$;SEE IF R0 WAS DECREMENTED PROPERLY.
BNE 260\$;BRANCH IF R0 IS INCORRECT.
MOV #210\$,R0 ;WAS THE OUTPUT DATA CORRECT?
MOV #220\$,R1
MOV #4,R2
110\$: CMP (R0)+,(R1)+
BNE 270\$;BRANCH IF INCORRECT.
SOB R2,110\$
MOV #-1,R0 ;IS THE REST OF THE OUTPUT BUFFER CORRECT. -1?
MOV #250\$,R1
MOV #4,R2
120\$: CMP R0,(R1)+
BNE 280\$;BRANCH IF INCORRECT.
SOB R2,120\$
BR 290\$
:THIS IS THE OUTPUT DATA BUFFER.
210\$: -1
-1
-1
-1
250\$: -1
-1
-1
-1
2404
:THIS IS THE TEST DATA LOADED INTO ACO:
220\$: 147250
36147
25036
147250
300\$: -1
-1

```

2412 010514 177777 -1
2413 010516 177777 -1
2414
2415
2416 010520 011600 010402 :IF AN ODD ADDRESS TRAP OCCURS COME HERE:
2417 010522 020027 010402 240$: MOV (SP),R0 ;SEE IF THE TRAP ACCURRED ON THE TEST INSTRUCTION.
2418 010526 001405 BEQ 310$ ;BRANCH IF YES.
2419 010530 020027 010404 CMP R0,#230$+4
2420 010534 001402 BEQ 310$ ;BRANCH IF YES.
2421 010536 000137 051774 JMP CPSPUR ;OTHERWISE GO REPORT A SPURIOUS TRAP THROUGH 4.
2422 :REPORT FAILURE IN FDST FLOWS RESULTED IN AN ODD ADDRESS.
2423 010542 010037 001236 310$: MOV R0,$TMP2
2424 010546 104017 ERROR +17 ;FDST FORK X ODD AD RES.
2425 010550 000426 BR 290$
2426
2427 :REPORT R0 INCORRECTLY DECREMENTED.
2428 010552 010037 001242 260$: MOV R0,$TMP4
2429 010556 012737 010460 001240 MOV #210$,$TMP3
2430 010564 104020 ERROR +20 ;R0 NOT DECRE PROP
2431 010566 000417 BR 290$
2432
2433 :REPORT OUTPUT DATA INCORRECT:
2434 010570 012737 010460 001240 270$: MOV #210$,$TMP3
2435 010576 012737 010500 001242 MOV #220$,$TMP4
2436 010604 104021 ERROR +21 ;BAD DATA
2437 010606 000407 BR 290$
2438 010610 012737 010470 001242 280$: MOV #250$,$TMP4
2439 010616 012737 010510 001240 MOV #300$,$TMP3
2440 010624 104022 ERROR +22 ;DATA BAD OUTSIDE TARGET AREA
2441 010626 104412 290$: 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?).
  
```

2447

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

2448	010630	000004			TST6:	SCOPE			
2449	010632	012737	010640	001110		MOV	#200\$, \$LPERR	;SET UP THE LOOP ON ERROR ADDRESS.	;DPM002
2450	010640	012701	010756		200\$:	MOV	#210\$, R1	;SET UP THE OUTPUT DATA BUFFER.	
2451	010644	012700	177777			MOV	#-1, R0		
2452	010650	012702	000012			MOV	#12, R2		
2453	010654	010021			1\$:	MOV	R0, (R1)+		
2454	010656	077202				SOB	R2, 1\$		
2455	010660	012737	010756	010772		MOV	#210\$, 220\$		
2456	010666	012700	000200			MOV	#200, R0	;ENTER DOUBLE FLOATING MODE.	
2457	010672	170i00				LDFPS	R0		
2458	010674	012700	011002			MOV	#230\$, R0	;SET UP ACO.	
2459	010700	172410				LDD	(R0), ACO		
2460	010702	012737	011012	000004		MOV	#240\$, ERRVECT	;SET UP FOR TRAPS TO 4.	
2461	010710	013737	010722	001236		MOV	250\$, STMP2		
2462	010716	012700	010772			MOV	#220\$, R0	;SET UP THE DESTINATION ADDRESS.	
2463	010722	174030			250\$:	STD	ACO, a(R0)+	;TEST INSTRUCTION.	
2464									
2465	010724	020027	010774			CMP	R0, #220\$+2	;SEE IF R0 WAS INCREMENTED CORRECTLY.	
2466	010730	001046				BNE	260\$;BRANCH IF INCORRECT.	
2467	010732	012701	010756			MOV	#210\$, R1	;CHECK THE OUTPUT DATA BUFFER.	
2468	010736	012702	011002			MOV	#230\$, R2		
2469	010742	012703	000004			MOV	#4, R3		
2470	010746	022122			270\$:	CMP	(R1)+, (R2)+		
2471	010750	001045				BNE	280\$;BRANCH IF NOT CORRECT.	
2472	010752	077303				SOB	R3, 270\$		
2473	010754	000452				BR	300\$		
2474									
2475									
2476	010756	177777							
2477	010760	177777							
2478	010762	177777							
2479	010764	177777							
2480	010766	177777							
2481	010770	177777							
2482	010772	010756			220\$:	210\$			
2483	010774	177777				-1			
2484	010776	177777				-1			
2485	011000	177777				-1			
2486	011002	101213			230\$:	101213			
2487	011004	141516				141516			
2488	011006	071727				71727			
2489	011010	037475				37475			
2490									
2491									
2492	011012	011602							
2493	011014	020227	010724		240\$:	MOV	(SP), R2	;SEE IF THE TRAP ADDRESS IS THAT OF THE TEST INSTRUCTION.	
2494	011020	001405				CMP	R2, #250\$+2		
2495	011022	020227	010726			BEQ	290\$;BRANCH IF YES.	
2496	011026	001402				CMP	R2, #250\$+4		
						BEQ	290\$;BRANCH IF YES.	

:THIS IS THHE OUTPUT DATA BUFFER:

```
2497 011030 000137 051774          JMP      CPSPUR ;OTHERWISE GO REPORT A SPURIOUS TRAP TO 4.
2498
2499          ;REPORT A FAILURE IN THE FDST FLOWS RESULTED IN AN ODD ADDRESS TRAP.
2500 011034 010237 001236          290$:   MOV      R2,$TMP2
2501 011040 022626                    CMP      (SP)+,(SP)+
2502 011042 104023                    ERROR   +23          ;BET FDST X ODD ADR
2503 011044 000416                    BR       300$
2504
2505          ;REPORT RO INCORRECT:
2506 011046 010037 001242          260$:   MOV      R0,$TMP4
2507 011052 012737 010774 001240    MOV      #220$+2,$TMP3
2508 011060 104024                    ERROR   +24          ;RO NOT INCREMENT PROPERLY
2509 011062 000407                    BR       300$
2510
2511          ;REPORT INCORRECT OUTPUT DATA:
2512 011064 012737 010756 001240    280$:   MOV      #210$,$TMP3
2513 011072 012737 011002 001242    MOV      #230$,$TMP4
2514 011100 104025                    ERROR   +25          ;BAD DATA
2515 011102
          300$:   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?).
          011102 104412
```

2521

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

2522	011104	000004							
2523	011106	012737	011114	001110					
2524	011114	012701	011232						
2525	011120	012700	177777						
2526	011124	012702	000012						
2527	011130	010021							
2528	011132	077202							
2529	011134	012737	011232	011244					
2530	011142	012700	000200						
2531	011146	170100							
2532	011150	012700	011256						
2533	011154	172410							
2534	011156	012737	011266	000004					
2535	011164	013737	011176	001236					
2536	011172	012700	011246						
2537	011176	174050							
2538	011200	020027	011244						
2539	011204	010146							
2540	011206	012701	011232						
2541	011212	012702	011256						
2542	011216	012703	000004						
2543	011222	022122							
2544	011224	001045							
2545	011226	077303							
2546	011230	000452							

2547									
2548	011232	177777							
2549	011234	177777							
2550	011236	177777							
2551	011240	177777							
2552	011242	177777							
2553	011244	011232							
2554	011246	177777							
2555	011250	177777							
2556	011252	177777							
2557	011254	177777							
2558	011256	020212							
2559	011260	023242							
2560	011262	026273							
2561	011264	031323							

2562									
2563									
2564	011266	011602							
2565	011270	020227	011200						
2566	011274	001405							
2567	011276	020227	011202						
2568	011302	001402							
2569	011304	000137	051774						
2570									

:THIS IS THE OUTPUT DATA BUFFER

:IF A TRAP TO 4 OCCURS COME HERE.

240\$:	MOV	(SP),R2							
	CMP	R2,#250\$+2							
	BEQ	300\$							
	CMP	R2,#250\$+4							
	BEQ	300\$							
	JMP	CPSPUR							

2571	011310	010237	001236	
2572	011314	022626		
2573	011316	104026		
2574	011320	000416		
2575				
2576				
2577	011322	010037	001242	
2578	011326	012737	011250	001240
2579	011334	104027		
2580	011336	000407		
2581				
2582				
2583	011340	012737	011232	001242
2584	011346	012737	011256	001240
2585	011354	104030		
2586	011356	104412		

```

300$:  MOV    R2,$TMP2
        CMP    (SP)+,(SP)+
        ERROR  +26
        BR     290$
;BET FDST X ODD ADR

;REPORT RO INCORRECT.
270$:  MOV    R0,$TMP4
        MOV    #260$+2,$TMP3
        ERROR  +27
        BR     290$
;RO NOT INCREMENT PROPERLY

;REPORT BAD DATA.
310$:  MOV    #210$,$TMP4
        MOV    #230$,$TMP3
        ERROR  +30
;BAD DATA

290$:  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?).
  
```

2592

```
.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.
:*
:*****
```

011360 000004

2593
 2594 011362 012767 011370 167520
 2595 011370 012700 000200
 2596 011374 170100
 2597 011376 012701 011506
 2598 011402 012700 177777
 2599 011406 012702 000004
 2600 011412 010021
 2601 011414 077202
 2602 011416 012767 011526 166360
 2603 011424 012700 011516
 2604 011430 172410
 2605 011432 012767 011450 167576
 2606 011440 012700 003605
 2607 011444 012701 000001
 2608 011450 174060 005701
 2609
 2610 011454 020027 003605
 2611 011460 001040
 2612 011462 012702 011506
 2613 011466 012703 011516
 2614 011472 012704 000004
 2615 011476 022223
 2616 011500 001037
 2617 011502 077403
 2618 011504 000444
 2619 011506 177777
 2620 011510 177777
 2621 011512 177777
 2622 011514 177777
 2623 011516 030313
 2624 011520 023334
 2625 011522 035363
 2626 011524 074041
 2627
 2628
 2629 011526 011602
 2630 011530 020227 011452
 2631 011534 001405
 2632 011536 020227 011454
 2633 011542 001402
 2634 011544 000167 040162
 2635
 2636 011550 010267 167462
 2637 011554 022626
 2638 011556 104031
 2639 011560 000416
 2640
 2641

```
TST10: SCOPE
.DSABL AMA ;DISABLE MODE 6 TO MODE 3 CONVERSIONS
MOV #200$,SLPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
200$: MOV #200,R0 ;ENTER DOUBLE FLOATING MODE.
LDFPS R0
MOV #210$,R1 ;SET UP THE OUT PUT DATA BUFFER.
MOV #-1,R0
MOV #4,R2
1$: MOV R0,(R1)+
SOB R2,1$
MOV #220$,ERRVECT ;SET UP VECTOR 4 INCASE OF EPROR.
MOV #230$,R0 ;SET UP AC0.
LDD (R0),AC0
MOV #240$,STMP2
MOV #210$-5701,R0 ;SET UP THE DESTINATION ADDRESS.
240$: MOV #1,R1
STD AC0,5701(R0) ;TEST INSTRUCTION.
CMP R0,#210$-5701 ;SEE IF R0 WAS MODIFIED.
BNE 250$ ;BRANCH IF INCORRECT.
MOV #210$,R2 ;WAS THE OUTPUT DATA CORRECT.
MOV #230$,R3
MOV #4,R4
2$: CMP (R2)+,(R3)+
BNE 260$ ;BRANCH IF INCORRECT DATA.
SOB R4,2$
BR 270$
210$: -1
-1
-1
-1
230$: 30313
23334
35363
74041
;COME HERE AFTER A TRAP THROUGH VECTOR 4.
220$: MOV (SP),R2 ;SEE IF THE TRAP OCCURRED ON THE TEST INSTR.
CMP R2,#240$+2
BEQ 280$ ;BRANCH IF YES.
CMP R2,#240$+4
BEQ 280$ ;BRANCH IF YES.
JMP FPSPUR ;OTHERWISE GO REPORT SPURIOUS TRAP TO 4.
;REPORT FAILURE OF FDST RESULTED IN AN ODD ADDRESS TRAP TO 4.
280$: MOV R2,STMP2
CMP (SP)+,(SP)+
ERROR +31 ;FDST FORK X ODD ADD
BR 270$
;REPORT R0 MODIFIED.
```

```
2642 011562 010067 167454 250$: MOV R0,$TMP4
2643 011566 012767 003605 167444 MOV #210$-5701,$TMP3
2644 011574 104032 ERROR +32 ;RO MODIFIED.
2645 011576 000407 BR 270$
2646
2647 ;REPORT INCORRECT DATA.
2648 011600 012767 011506 167432 260$: MOV #210$,$TMP3
2649 011606 012767 011516 167426 MOV #230$,$TMP4
2650 011614 104033 ERROR +33 ;BAD DATA
2651 011616 270$: RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
011616 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

2652 .ENABL AMA
```


2658

```
.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
2659 011620 000004 011630 001110  MOV      #200$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
2660 011630 012700 000200 200$:  MOV      #200, R0 ;ENTER DOUBLE FLOATING MODE.
2661 011634 170100  LDFPS   R0
2662 011636 012701 011754  MOV      #210$, R1 ;SET UP THE OUTPUT DATA BUFFER.
2663 011642 012700 177777  MOV      #-1, R0
2664 011646 012702 000004  MOV      #4, R2
2665 011652 010021 100$:  MOV      R0, (R1)+
2666 011654 077202  SOB     R2, 100$
2667 011656 012737 012004 000004  MOV      #220$, ERRVECT ;SET UP FOR TRAPS TO 4.
2668 011664 012700 011764  MOV      #230$, R0 ;SET UP ACO.
2669 011670 172410  LDD     (R0), ACO
2670 011672 012737 011716 001236  MOV      #240$, $TMP2
2671 011700 012700 004073  MOV      #250$-5701, R0 ;SET UP THE DESTINATION ADDRESS.
2672 011704 012701 000001  MOV      #1, R1
2673 011710 012737 011754 011774  MOV      #210$, 250$
2674 011716 174070 005701 240$:  STD     ACO, @5701(R0) ;TEST INSTRUCTION.
2675
2676 011722 020027 004073  CMP     R0, #250$-5701 ;IS R0 CORRECT?
2677 011726 001044  BNE    260$ ;BRANCH IF INCORRECT.
2678 011730 012702 011754  MOV     #210$, R2 ;WAS THE DATA OUTPUT CORRECTLY?
2679 011734 012703 011764  MOV     #230$, R3
2680 011740 012704 000004  MOV     #4, R4
2681 011744 022223 110$:  CMP     (R2)+, (R3)+ ;BRANCH IF DATA IS INCORRECT.
2682 011746 001043  BNE    270$
2683 011750 077403  SOB    R4, 110$
2684 011752 000450  BR     R4, 280$
2685 011754 177777 210$:  -1
2686 011756 177777  -1
2687 011760 177777  -1
2688 011762 177777  -1
2689 011764 041424 230$:  41424
2690 011766 034445  34445
2691 011770 046475  46475
2692 011772 051525  051525
2693 011774 177777 250$:  -1
2694 011776 177777  -1
2695 012000 177777  -1
2696 012002 177777  -1
2697
2698 ;TRAP THROUGH 4 TO HERE.
2699 012004 011602 220$:  MOV     (SP), R2 ;SEE IF THE TRAP OCCURRED ON THE TEST INSTR.
2700 012006 020227 011720  CMP     R2, #240$+2
2701 012012 001405  BEQ    290$ ;BRANCH IF YES.
2702 012014 020227 011722  CMP     R2, #240$+4
2703 012020 001402  BEQ    290$ ;BRANCH IF YES.
2704 012022 000137 051732  JMP     FPSPUR ;OTHERWISE GO REPORT SPURIOUS TRAP TO 4.
2705 ;REPORT FAILURE OF FDST FORK RESULTED IN AN ODD ADDRESS TRAP TO 4.
2706 012026 010237 001236 290$:  MOV     R2, $TMP2
2707 012032 022626  CMP     (SP)+, (SP)+
```


2727

```

.SBTTL TEST # 12 - STCFD TEST
:*****
:*TEST 12      STCFD TEST
:*
:*THIS IS A TEST OF THE STCFD INSTRUCTION.
:*
:*****
    
```

```

012076 000004
2728
2729 012100 012737 012106 001110      MOV      #200$, $LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.      ;DPM002
2730 012106 004737 012462      JSR      PC, 1000$
2731 012112 000000      1$:      0      ;AC
2732 012114 000000      0
2733 012116 000000      0
2734 012120 000000      0
2735 012122 000000      2$:      0      ;RES
2736 012124 000000      0
2737 012126 000000      0
2738 012130 000000      0
2739 012132 000000      3$:      0      ;ERROR RES.
2740 012134 000000      0
2741 012136 177777      -1
2742 012140 177777      -1
2743 012142 047000      4$:      47000      ;FPS BEFORE EXECUTION.
2744 012144 047004      47004      ;FPS AFTER EXECUTION.
2745 012146 177777      -1      ;FEC
2746 012150 147004      147004      ;ERROR FPS.
2747 012152 104042      5$:      ERROR      +42      ;FDFL<---FDFLXST 767
2748 012154 000401      BR      6$
2749 012156 104043      ERROR      +43      ;BUT EZBT X ST560 TO 061 INTO 261
2750 012160
2751
2752 012160 012737 012166 001110      MOV      #210$, $LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.      ;DPM002
2753 012166 004737 012462      JSR      PC, 1000$
2754 012172 017203      11$:     17203      ;AC
2755 012174 142536      142536
2756 012176 047506      47506
2757 012200 172031      172031
2758 012202 017203      12$:     17203      ;RES
2759 012204 142536      142536
2760 012206 000000      0
2761 012210 000000      0
2762 012212 017203      13$:     17203      ;ERROR RES.
2763 012214 142536      142536
2764 012216 047506      47506
2765 012220 172031      172031
2766 012222 040000      14$:     40000      ;FPS BEFORE EXECUTION.
2767 012224 040000      040000      ;FPS AFTER EXECUTION.
2768 012226 177777      -1      ;FEC
2769 012230 177777      -1      ;ERROR FPS.
2770 012232 104044      15$:     ERROR      +44      ;X11(1,0)<---0 X ST766
2771 012234 000401      BR      16$
2772 012236 104040      ERROR      +40
2773 012240
2774
2775 012240 012737 012246 001110      MOV      #220$, $LPERR      ;SE1 UP THE LOOP ON ERRJR ADDRESS.      ;DPM002
2776 012246 004737 012462      JSR      PC, 1000$
    
```

```

2777 012252 050717          21$: 50717          ;AC
2778 012254 027374          27374
2779 012256 075767          75767
2780 012260 077071          77071
2781 012262 050717          22$: 50717          ;RES
2782 012264 027374          27374
2783 012266 000000          0
2784 012270 000000          0
2785 012272 000000          23$: 0          ;ERROR RES.
2786 012274 000000          0
2787 012276 000000          0
2788 012300 000000          0
2789 012302 047000          24$: 47000          ;FPS BEFORE EXECUTION.
2790 012304 047000          47000          ;FPS AFTER EXECUTION.
2791 012306 177777          -1          ;FEC
2792 012310 174002          174002          ;ERROR FPS.
2793 012312 104045          25$: ERROR +45          ;BUT OPIC X ST251
2794 012314 000401          BR 26$
2795 012316 104046          ERROR +46          ;BUT EZBT X ST421
2796 012320
2797
2798 012320 012737 012326 001110 MOV #230$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
2799 012326 004737 012462 230$: JSR PC,1000$
2800 012332 020212 31$: 20212          ;AC
2801 012334 032425          32425
2802 012336 026272          26272
2803 012340 002123          02123
2804 012342 020212          32$: 20212          ;RES
2805 012344 032425          32425
2806 012346 000000          0
2807 012350 000000          0
2808 012352 020212          33$: 20212          ;ERROR RES.
2809 012354 032425          32425
2810 012356 100000          100000
2811 012360 000000          0
2812 012362 040000          34$: 40000          ;FPS BEFORE EXECUTION.
2813 012364 040000          40000          ;FPS AFTER EXECUTION.
2814 012366 177777          -1          ;FEC
2815 012370 177777          -1          ;ERROR FPS.
2816 012372 104047          35$: ERROR +47          ;BUT FD IN ROUND X ST113
2817 012374 000401          BR 36$
2818 012376 104040          ERROR 40
2819 012400
2820
2821 012400 012737 012406 001110 MOV #2,0$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
2822 012406 004737 012462 240$: JSR PC,1000$
2823 012412 121314 41$: 121314          ;AC
2824 012414 151617          151617
2825 012416 101112          101112
2826 012420 131415          131415
2827 012422 121314          42$: 121314          ;RES
2828 012424 151617          151617
2829 012426 000000          0
2830 012430 000000          0
2831 012432 021314          43$: 21314          ;ERROR RES.
2832 012434 151617          151617
2833 012436 000000          0

```

2834 012440 000000
2835 012442 040000
2836 012444 040010
2837 012446 177777
2838 012450 177777
2839 012452 104050
2840 012454 000401
2841 012456 104040
2842 012460 000535

44\$: 0
40000
40010
-1
-1
45\$: ERROR +50
BR 46\$
ERROR +40
46\$: BR 250\$

:FPS BEFORE EXECUTION.
:FPS AFTER EXECUTION.
:FEC
:ERROR FPS.
;BUT ENBT X ST567 OR BAD SIGN ST460

```

2843 ;THIS SUBROUTINE, 1000$, IS USED TO SET UP THE OPERANDS, EXECUTE
2844 ;THE STCFD INSTRUCTION AND CHECK THE RESULTS. A CALL
2845 ;TO IT IS MADE THUS:
2846
2847 JSR PC,1000$
2848 ACARG: .WORD X,X,X,X ;AC OPERAND
2849 RES: .WORD X,X,X,X ;EXPECTED RESULT
2850 ERRES: .WORD X,X,X,X ;ERROR RESULT
2851 FPSB: .WORD X ;FPS BEFORE EXECUTION
2852 FPSA: .WORD X ;FPS AFTER EXECUTION
2853 FLC: .WORD X ;EXPECTED FEC
2854 ERFPS: .WORD X ;ERROR FPS.
2855 ERR1: ERROR +X ;DATA ERROR.
2856 BR CONT
2857 ERR2: ERROR +X ;FPS ERROR.
2858 CONT: ;RETURN ADDRESS
2859
2860 ;THE OPERANDS ARE SET UP (USING ACO AS THE ACCUMULATOR). THEN
2861 ;THE STCFD INSTRUCTION IS EXECUTED.
2862 ;THE RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
2863 ;COMPARED WITH FPSA IF THIS TOO IS CORRECT 1000$ RETURNS CONTROL
2864 ;TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD 1000$
2865 ;COMPARE IT TO ERROR FPS. IF THIS MATCHES THEN 1000$ WILL RETURN
2866 ;TO THE ERROR CALL AT ERR2, OTHERWISE 1000$ ITSELF
2867 ;REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE
2868 ;STCFD IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
2869 ;ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
2870 ;THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN 1000$
2871 ;WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR1. OTHERWISE THE
2872 ;RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND 1000$ WILL
2873 ;REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.
2874
2875 1000$: MOV (SP)+,R1 ;PICK UP THE POINTER TO THE OPERANDS.
2876 MOV #200,R0 ;ENTER DOUBLE FLOATING MODE.
2877 LDFPS R0
2878 MOV R1,R0 ;LOAD ACO.
2879 LDD (R0),ACO
2880 MOV #-1,R0 ;FILL THE OUTPUT BUFFER WITH -1'S.
2881 MOV #245$,R2
2882 MOV #4,R3
2883 51$: MOV R0,(R2)+
2884 SOB R3,51$
2885 MOV 30(R1),R0 ;LOAD THE FPS.
2886 LDFPS R0
2887 MOV #52$,STMP2
2888 MOV #245$,R0 ;SET UP THE DESTINATION ADDRESS.
2889 52$: STCFD ACO,(R0) ;TEST INSTRUCTION.
2890
2891 STFPS R4 ;GET THE FPS.
2892 STST R5 ;GET THE FEC.
2893 MOV R1,R2 ;SAVE THE DATA IN CASE Ce ERROR.
2894 MOV R2,STMP3
2895 ADD #10,R2
2896 MOV R2,STMP5
2897 MOV #245$,STMP4
2898 MOV R4,STMP7
2899 MOV 32(R1),STMP10

```


2954

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

```

012756 000004
2955
2956 012760 012737 012766 001110
2957 012766 004737 013342
2958 012772 000000
2959 012774 000000
2960 012776 000000
2961 013000 000000
2962 013002 000000
2963 013004 000000
2964 013006 177777
2965 013010 177777
2966 013012 000000
2967 013014 000000
2968 013016 000000
2969 013020 000000
2970 013022 047200
2971 013024 047204
2972 013026 177777
2973 013030 177777
2974 013032 104054
2975 013034 000401
2976 013036 104052
2977 013040
2978
2979 013040 012737 013046 001110
2980 013046 004737 013342
2981 013052 067574
2982 013054 073727
2983 013056 170777
2984 013060 067574
2985 013062 067574
2986 013064 073730
2987 013066 177777
2988 013070 177777
2989 013072 067574
2990 013074 073727
2991 013076 177777
2992 013100 177777
2993 013102 040200
2994 013104 040200
2995 013106 177777
2996 013110 177777
2997 013112 104055
2998 013114 000401
2999 013116 104052
3000 013120
3001
3002 013120 012737 013126 001110
3003 013126 004737 013342

TST13: SCOPE
:AC=0
200$: MOV #200$,SLPERR ;SET UP THE LOOP ON ERFOR ADDRESS. ;DPM002
      JSR PC,1000$
1$: 0 ;AC
   0
   0
2$: 0 ;RES
   0
   -1
3$: 0 ;ERROR RES.
   0
   0
4$: 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$:
;
210$: MOV #210$,SLPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
      JSR PC,1000$
11$: 67574 ;ACO
     73727
     170777
     67574
12$: 67574 ;RES
     73730
     -1
     -1
13$: 67574 ;ERROR RES.
     73727
     -1
     -1
14$: 40200 ;FPS BEFORE EXECUTION.
     40200 ;FPS AFTER EXECUTION.
     -1 ;FEC
     -1 ;ERROR FPS.
15$: ERROR +55 ;EITHER ROUND FAILED OR WENT TO 766 X1(1,0)<---0 INTO 767
     BR 16$
     ERROR +52
16$:
;
220$: MOV #220$,SLPERR ;SET UP THE LUOP ON ERROR ADDRESS. ;DPM002
      JSR PC,1000$
```


3004	013132	077777			21\$:	77777			:ACO	
3005	013134	177777				-1				
3006	013136	100000				100000				
3007	013140	000000				0				
3008	013142	000000			22\$:	0			:RES	
3009	013144	000000				0				
3010	013146	177777				-1				
3011	013150	177777				-1				
3012	013152	077777			23\$:	77777			:ERROR RES.	
3013	013154	177777				-1				
3014	013156	177777				-1				
3015	013160	177777				-1				
3016	013162	040200			24\$:	40200			:FPS BEFORE EXECUTION.	
3017	013164	040206				40206			:FPS AFTER EXECUTION.	
3018	013166	177777				-1			:FEC	
3019	013170	040204				40204			:ERROR FPS.	
3020	013172	104055			25\$:	ERROR	+55			
3021	013174	000401				BR	26\$			
3022	013176	104056				ERROR	+56		:BUT EZBT X ST421 TO 062 INTO 262	
3023	013200				26\$:					
3024					:					
3025	013200	012737	013206	001110		MOV	#230\$, \$LPERR		:SET UP THE LOOP ON ERROR ADDRESS.	:DPM002
3026	013206	004737	013342		230\$:	JSR	PC,1000\$			
3027	013212	077777			31\$:	77777			:ACO	
3028	013214	177777				-1				
3029	013216	100000				100000				
3030	013220	000000				0				
3031	013222	000000			32\$:	0			:RES	
3032	013224	000000				0				
3033	013226	177777				-1				
3034	013230	177777				-1				
3035	013232	077777			33\$:	77777			:ERROR RES.	
3036	013234	177777				-1				
3037	013236	177777				-1				
3038	013240	177777				-1				
3039	013242	040200			34\$:	40200			:FPS BEFORE EXECUTION.	
3040	013244	040206				40206			:FPS AFTER EXECUTION.	
3041	013246	177777				-1			:FEC	
3042	013250	140206				140206			:ERROR FPS.	
3043	013252	104055			35\$:	ERROR	+55			
3044	013254	000401				BR	36\$			
3045	013256	104057				ERROR	+57		:BUT FIV ST262 TO 123 INTO 103	
3046	013260				36\$:					
3047					:					
3048	013260	012737	013266	001110		MOV	#240\$, \$LPERR		:SET UP THE LOOP ON ERROR ADDRESS.	:DPM002
3049	013266	004737	013342		240\$:	JSR	PC,1000\$			
3050	013272	177777			41\$:	177777			:ACO	
3051	013274	177777				-1				
3052	013276	100000				100000				
3053	013300	000000				0				
3054	013302	100000			42\$:	100000			:RES	
3055	013304	000000				0				
3056	013306	177777				-1				
3057	013310	177777				-1				
3058	013312	000000			43\$:	0			:ERROR RES.	
3059	013314	000000				0				
3060	013316	177777				-1				

3061	013320	177777		-1	
3062	013322	047200	44\$:	47200	:FPS BEFORE EXECUTION.
3063	013324	147216		147216	:FPS AFTER EXECUTION.
3064	013326	000010		10	:FEC
3065	013330	047206		47206	:ERROR FPS.
3066	013332	104060	45\$:	ERROR	:BUT FIV ST262 FAIL TO 103 INT 123
3067	013334	000401		BR	
3068	013336	104061		ERROR	
3069	013340	000535	46\$:	BR	:BUT FLAG ST 147 X TO ST 361 INTO 365
				250\$	

3070
3071
3072
3073
3074
3075
3076
3077
3078
3079
3080
3081
3082
3083
3084
3085
3086
3087
3088
3089
3090
3091
3092
3093
3094
3095
3096
3097
3098
3099
3100
3101
3102 013342 012601
3103 013344 012700 000200
3104 013350 170100
3105 013352 010100
3106 013354 172410
3107 013356 012700 177777
3108 013362 012702 013624
3109 013366 012703 000004
3110 013372 010022
3111 013374 077302
3112 013376 016100 000030
3113 013402 170100
3114 013404 012737 013416 001236
3115 013412 012700 013624
3116 013416 176010
3117
3118 013420 170204
3119 013422 170305
3120 013424 010102
3121 013426 010237 001240
3122 013432 062702 000010
3123 013436 010237 001244
3124 013442 012737 013624 001242
3125 013450 010437 001250
3126 013454 016137 000032 001252

: THIS SUBROUTINE, 1000\$, IS USED TO SET UP THE OPERANDS, EXECUTE
: THE STCDF INSTRUCTION AND CHECK THE RESULTS. A CALL
: TO IT IS MADE THUS:

```

      JSR      PC,1000$
      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:   .WORD  X              ;RETURN ADDRESS
    
```

: THE OPERANDS ARE SET UP (USING ACO AS THE ACCUMULATOR). THEN
: THE STCDF 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
: STCDF 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.

```

1000$: MOV      (SP)+,R1      ;PICK UP THE POINTER TO THE OPERANDS.
      MOV      #200,R0      ;ENTER DOUBLE FLOATING MODE.
      LDFPS   R0
      MOV      R1,R0        ;LOAD ACO.
      LDD     (R0),ACO
      MOV      #-1,R0       ;FILL THE OUTPUT BUFFER WITH -1'S.
      MOV      #260$,R2
      MOV      #4,R3
51$:  MOV      R0,(R2)+
      SOB     R3,51$
      MOV      30(R1),R0    ;LOAD THE FPS.
      LDFPS   R0
      MOV      #52$,STMP2
      MOV      #260$,R0
52$:  STCDF   ACO,(R0)      ;SET UP THE DESTINATION ADDRESS.
      ;TEST INSTRUCTION.

      STFPS   R4            ;GET THE FPS.
      STST   R5            ;GET THE FEC.
      MOV     R1,R2        ;SAVE THE DATA IN CASE OF ERROR.
      MOV     R2,STMP3
      ADD    #10,R2
      MOV     R2,STMP5
      MOV     #260$,STMP4
      MOV     R4,STMP7
      MOV     32(R1),STMP10
    
```

```

3127
3128 013462 010102          MOV      R1,R2          ;CHECK THE RESULT.
3129 013464 062702 000010  ADD      #10,R2
3130 013470 012703 013624  MOV      #260$,R3
3131 013474 012700 000004  MOV      #4,R0
3132 013500 022223          53$:    CMP      (R2)+,(R3)+
3133 013502 001014          BNE      65$            ;BRANCH IF INCORRECT.
3134 013504 077003          SOB      R0,53$
3135
3136 013506 016102 000032  MOV      32(R1),R2
3137 013512 020204          CMP      R2,R4          ;IS THE FPS CORRECT?
3138 013514 001025          BNE      70$            ;BRANCH IF FPS INCORRECT.
3139 013516 005702          TST      R2             ;IF EXPECTED FPS IS NEGATIVE, THEN
3140 013520 100003          BPL      54$            ;GO AHEAD AND CHECK THE FEC.
3141 013522 026105 000034  CMP      34(R1),R5
3142 013526 001027          BNE      75$            ;BRANCH IF FEC IS INCORRECT.
3143 013530 000161 000046  54$:    JMP      46(R1)         ;RETURN.
3144
3145          ;RESULT INCORRECT:
3146 013534 010102 65$:    MOV      R1,R2          ;SEE IF ERROR WAS ANTICIPATED.
3147 013536 062702 000020  ADD      #20,R2
3148 013542 012703 013624  MOV      #260$,R3
3149 013546 012700 000004  MOV      #4,R0
3150 013552 022223          66$:    CMP      (R2)+,(R3)+
3151 013554 001003          BNE      67$            ;BRANCH IF NOT ANTICIPATED.
3152 013556 077003          SOB      R0,66$
3153 013560 000161 000040  67$:    JMP      40(R1)         ;IF ERROR WAS ANTICIPATED RETURN.
3154          ;OTHERWISE REPORT RESULT INCORRECT HERE.
3155 013564          68$:    ERROR   +51          ;DATA ERROR
3156 013564 104051          BR       54$
3157 013566 000760
3158
3159          ;FPS INCORRECT:
3160 013570 020461 000034  70$:    CMP      R4,34(R1)      ;WAS THE ERROR ANTICIPATED.
3161 013574 001002          BNE      71$            ;BRANCH IF NOT ANTICIPATED.
3162 013576 000161 000044  71$:    JMP      44(R1)         ;IF IT WAS ANTICIPATED RETURN.
3163
3164          ;THE FPS ERROR WAS NOT ANTICIPATED SO REPORT FPS INCORRECT HERE.
3165 013602          72$:    ERROR   +52          ;FPS X
3166 013602 104052          BR       54$
3167 013604 000751
3168
3169          ;REPORT FEC INCORRECT:
3170 013606 016137 000036 001256  75$:    MOV      36(R1),$TMP12
3171 013614 010537 001254          MOV      R5,$TMP11
3172 013620 104053          76$:    ERROR   +53          ;FEC X
3173 013622 000742          BR       54$
3174 013624 177777 177777 260$:   .WORD   -1,-1,-1,-1
3175 013634          250$:   RSETUP
3176 013634 104412

```

3181

```
.SBTTL TEST # 14 - STCFD WITH ILLEGAL ACCUMULATOR TEST
:*****
:*TEST 14 STCFD WITH ILLEGAL ACCUMULATOR TEST
:*
:*THIS TEST STCFD WITH ILLEGAL AC 6.
:*
:*****
TST14: SCOPE
```

```
013636 000004
3182
3183 013640 012737 013646 001110 200$: MOV #200$,SLPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
3184 013646 012700 040000 MOV #40000,R0 ;DISSABLE INTERRUPTS.
3185 013652 170100 LDFPS R0
3186 013654 012737 013662 001236 210$: MOV #210$,STMP2 ;THIS TEST INSTRUCTION SHOULD CAUSE AN ERROR.
3187 013662 176006 STCFD AC0,AC6
3188 STFPS R4 ;GET FPS.
3189 013664 170204 STST R5 ;GET FEC.
3190 013666 170305 CMP R4,#140000 ;IS FPS CORRECT?
3191 013670 020427 140000 BNE 220$ ;BRANCH IF INCORRECT FPS.
3192 013674 001004 000002 CMP #2,R5 ;IS FEC CORRECT?
3193 013676 022705 BNE 230$ ;BRANCH IF INCORRECT.
3194 013702 001010 BR 240$
3195 013704 000415
3196
3197 ;REPORT FPS INCORRECT AFTER USE OF ILLEGAL ACCUMULATOR.
3198 013706 010437 001242 001240 220$: MOV R4,$TMP4
3199 013712 012737 140000 001240 MOV #140000,$TMP3
3200 013720 104062 ERROR +62 ;BUT FDST ST767 X TO 567 INTO 577
3201 013722 000406 BR 240$
3202
3203 ;REPORT FEC INCORRECT AFTER USE OF ILLEGAL ACCUMULATOR.
3204 013724 010537 001242 001240 230$: MOV R5,$TMP4
3205 013730 012737 000002 001240 MOV #2,$TMP3
3206 013736 104063 ERROR +63 ;FEC<---2 ST577 X
3207 013740 104412 240$: 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?).
```

3213

.SBTTL TEST # 15 - CLRD TEST
:*****
:TEST 15 CLRD TEST
:*****
:THIS IS A TEST OF THE CRLF AND CLRD INSTRUCTIONS.
:*****

```

3214 013742 000004
3214 013744 012737 013752 001110 1$T15: SCOPE
3215 013752 012700 014136 200$: MOV #200$,SLPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
3216 013756 012701 014126 200$: MOV #210$,R0 ;SET UP OUTPUT BUFFER
3217 013762 012702 000004 200$: MOV #220$,R1
3218 013766 012021 1$: MOV #4,R2
3219 013770 077202 1$: MOV (R0)+,(R1)+
3220 013772 012700 014126 SOB R2,1$
3221 013776 012701 000213 2$: MOV #220$,R0 ;SET UP DESTINATION OPERAND ADDRESS.
3222 014002 170101 2$: MOV #213,R1 ;SET UP FPS.
3223 014004 012737 014012 001236 2$: LDFPS R1
3224 014012 170410 2$: MOV #2$,STMP2
3225 2$: CLRD (R0) ;TEST INSTRUCTION.
3226 014014 170205 STFPS R5 ;GET FPS.
3227 014016 012702 000004 MOV #4,R2 ;SEE IF RESULT CLEAR, 0.
3228 014022 012701 014126 3$: MOV #220$,R1
3229 014026 005721 3$: TST (R1)+
3230 014030 001010 BNE 230$ ;BRANCH IF RESULT INCORRECT, NOT 0.
3231 014032 077203 SOB R2,3$
3232 014034 022705 000204 CMP #204,R5 ;SEE IF FPS IS CORRECT.
3233 014040 001014 BNE 240$ ;BRANCH IF INCORRECT.
3234 014042 020027 014126 CMP R0,#220$ ;SEE IF R0 IS CORRECT.
3235 014046 001020 BNE 250$ ;BRANCH IF R0 IS INCORRECT.
3236 014050 000442 BR 260$
3237
3238 ;RESULT NOT 0, REPORT ERROR.
3239 014052 012737 014126 001240 230$: MOV #220$,STMP3
3240 014060 012737 014146 001242 MOV #270$,STMP4
3241 014066 104064 ERROR +64 ;BAD DATA = 0 X 11+ZERO ST770 X
3242 014070 000432 BR 260$
3243
3244 ;REPORT FPS INCORRECT:
3245 014072 010437 001242 001240 240$: MOV R4,STMP4
3246 014076 012737 000204 MOV #204,STMP3
3247 014104 104065 ERROR +65 ;BAD FPS
3248 014106 000423 BR 260$
3249
3250 ;REPORT R0 INCORRECT.
3251 014110 010037 001242 001240 250$: MOV R0,STMP4
3252 014114 012737 014126 MOV #220$,STMP3
3253 014122 104066 ERROR +66
3254 014124 000414 BR 260$
3255
3256 ;THIS IS THE TEST DATA BUFFER, OUTPUT DATA BUFFER.
3257 014126 073475 220$: 73475
3258 014130 067707 67707
3259 014132 127347 127347
3260 014134 056770 56770
3261 ;THIS IS THE DATA USED TO SET UP THE OUTPUT BUFFER.
3262 014136 073475 210$: 73475

```

TEST # 15 - CLRD TEST

3263	014140	067707
3264	014142	127347
3265	014144	056770
3266		
3267	014146	000000
3268	014150	000000
3269	014152	000000
3270	014154	000000
3271	014156	
	014156	104412

67707
127347
56770
:THIS IS THE EXPECTED DATA, RESULT:
270\$: 0
0
0
0
260\$: 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?).

3277

```
.SBTTL TEST # 16 - CLRD WITH ILLEGAL ACCUMULATOR TEST
:*****
:*TEST 16 CLRD WITH ILLEGAL ACCUMULATOR TEST
:*
:*THIS IS A TEST OF CLRD WITH ILLEGAL AC7.
:*
:*****
```

```
3278 014160 000004 014170 001110 TST16: SCOPE
3279 014170 012737 040200 200$: MOV #200$,SLPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
3280 014174 170100 LDFPS R0 ;SET UP THE FPS, NO INTERRUPTS AND FD-1.
3281 014176 012737 014204 001236 210$: MOV #210$,STMP2
3282 014204 170407 CLRD AC7 ;TEST INSTRUCTION.
3283
3284 014206 170204 STFPS R4 ;GET FPS.
3285 014210 170305 STST R5 ;GET FEC.
3286 014212 020427 140200 CMP R4,#140200 ;IS THE FPS CORRECT?
3287 014216 001004 BNE 220$ ;BRANCH IF FPS IS INCORRECT.
3288 014220 022705 000002 CMP #2,R5 ;IS THE FEC CORRECT?
3289 014224 001010 BNE 230$ ;BRANCH IF FEC IS INCORRECT.
3290 014226 000415 BR 240$
3291
3292 ;REPORT INCORRECT FPS:
3293 014230 010437 001242 220$: MOV R4,STMP4
3294 014234 012737 140200 001240 MOV #140200,STMP3
3295 014242 104067 ERROR +67 ;BUT FDST ST 700X TO 607 INTO 677
3296 014244 000406 BR 240$
3297
3298 ;REPORT INCORRECT FEC:
3299 014246 010537 001242 230$: MOV R5,STMP4
3300 014252 012737 000002 001240 MOV #2,STMP3
3301 014260 104070 ERROR +70 ;FEC<---2 ST 677 X
3302 014262 104412 240$: 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?).
```


3311

```
.SBTTL TEST # 17 - NEGF, ABSF & TSTF SRC MD 0 WITH ILLGL AC7
:*****
:*TEST 17      NEGF, ABSF & TSTF SRC MD 0 WITH ILLGL AC7
:*
:*THIS IS A TEST OF THE SPECIAL
:*DEST FLOWS USING THE NEGD INST
:*WITH MODE ZERO AND ILLEGAL
:*AC7.
:*****
```

```
TST17: SCOPE
3312 014264 000004          MOV      #200$,SLPERR      ;SET UP THE LOOP ON ERROR ADDRESS.      ;DPM002
3313 014274 012737 014274 001110 200$: MOV      #40200,R0      ;SET UP THE FPS, FID=1 AND FD=1.
3314 014300 170100          LDFPS   R0
3315 014302 012737 014310 001236 200$: MOV      #210$,STMP2
3316
3317 014310 170707          210$: NEGD   AC7          ;TEST INSTRUCTION.
3318
3319 014312 170204          STFPS   R4          ;GET FPS.
3320 014314 170305          STST    R5          ;GET FEC.
3321
3322 014316 022704 140200          CMP      #140200,R4          ;IS FPS CORRECT?
3323 014322 001004          BNE     220$          ;BRANCH IF FPS IS INCORRECT.
3324 014324 022705 000002          CMP      #2,R5          ;IS FEC CORRECT?
3325 014330 001010          BNE     230$          ;BRANCH IF FEC IS INCORRECT.
3326 014332 000416          BR      240$
3327
3328          :REPORT INCORRECT FPS:
3329 014334 012737 140200 001240 220$: MOV      #140200,STMP3
3330 014342 010437 001242          MOV      R4,STMP4
3331 014346 104176          ERROR   +176          ;FPS BAD
3332 014350 000407          BR      240$
3333
3334          :REPORT FEC INCORRECT:
3335 014352 012737 000002 001240 230$: MOV      #2,STMP3
3336 014360 010537 001242          MOV      R5,STMP4
3337 014364 104377          ERROR   +377          ;FEC BAD
3338 014366 000044          .WORD   44
3339 014370          240$: 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?).
          104412
```

3347

```
.SBTTL TEST # 20 - NEGF, ABSF & TSTF SRC MODE 0 TEST
:*****
:*TEST 20      NEGF, ABSF & TSTF SRC MODE 0 TEST
:*
:*THIS IS A TEST THE NEGF, ABSF AND TSTF
:*SOURCE FLOWS. THE NEGD INSTRUCTION
:*IS USED TO TEST MODE 0
:*
:*****
```

```
TST20:  JCOPE
3348 014372 000004      MOV      #200$, $LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.      ;DPM002
3349 014402 012737 014402 001110 200$:  MOV      #200, R0          ;SET FD MODE.
3350 014406 170100      LDFPS   R0
3351 014410 012700 014552      MOV      #210$, R0          ;SET UP ACO.
3352 014414 172410      LDD     (R0), ACO          ;SET ACO = 0
3353 014416 005000      CLR     R0                 ;CLEAR THE FPS.
3354 014420 170100      LDFPS   R0
3355 014422 012700 014562      MOV      #220$, R0          ;LOAD ACO TO BE A FLOATING 0.
3356 014426 172410      LDF     (R0), ACO          ;SET ACO=ZERO
3357                                     ;FLOAT
3358 014430 012700 000201      MOV      #201, R0          ;SET FD MODE.
3359 014434 170100      LDFPS   R0
3360 014436 012737 014444 001236 230$:  MOV      #230$, $TMP2
3361                                     ;TEST INSTRUCTION.
3362 014444 170700      NEGD    ACO
3363                                     ;GET FPS.
3364 014446 170205      STFPS   R5                 ;SET FD MODE.
3365 014450 012700 000200      MOV      #200, R0
3366 014454 170100      LDFPS   R0
3367 014456 012700 014572      MOV      #240$, R0          ;GET THE RESULT OUT OF ACO.
3368 014462 174010      STD     ACO, (R0)
3369                                     ;SEE IF THE RESULT IS CORRECT.
3370 014464 012701 000004 1$:      MOV      #4, R1
3371 014470 005720      TST     (R0)+
3372 014472 001005      BNE     250$                ;BRANCH IF THE RESULT IS INCORRECT.
3373 014474 077103      SOB     R1, 1$
3374 014476 022705 000204      CMP     #204, R5            ;IS THE FPS CORRECT?
3375 014502 001014      BNE     260$                ;BRANCH IF THE FPS IS INCORRECT.
3376 014504 000442      BR      270$
3377                                     ;RESULT INCORRECT, REPORT FAILURE:
3378 250$:  MOV      #220$, $TMP4      ;EXPECT DO
3379 014506 012737 014562 001242      MOV      #280$, $TMP3      ;PREV FO IMPURE
3380 014514 012737 014602 001240      MOV      #240$, $TMP5      ;GOT
3381 014522 012737 014572 001244      ERROR   +71
3382 014530 104071      BR      270$
3383 014532 000427
3384
3385                                     ;REPORT FPS INCORRECT:
3386 014534 012737 000204 001240 260$:  MOV      #204, $TMP3
3387 014542 010537 001242      MOV      R5, $TMP4
3388 014546 104072      ERROR   +72
3389 014550 000420      BR      270$
3390
3391                                     ;THESE ARE TEST DATA TABLES AND AN OUTPUT BUFFER.
3392 014552 101112 210$:  101112
3393 014554 131415      131415
3394 014556 161710      161710
```

3395	014560	111213		111213
3396	014562	000000	220\$:	0
3397	014564	000000		0
3398	014566	000000		0
3399	014570	000000		0
3400				
3401	014572	177777	240\$:	-1
3402	014574	177777		-1
3403	014576	177777		-1
3404	014600	177777		-1
3405	014602	000000	280\$:	0
3406	014604	000000		0
3407	014606	161710		161710
3408	014610	111213		111213
3409				
3410	014612		270\$:	
	014612	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?).

3411

.SBTTL TEST # 21 - NEGF, ABSF & TSTF SRC MODE 1
:*****
:*TEST 21 NEGF, ABSF & TSTF SRC MODE 1
:*
:*THIS IS A TEST THE NEGF, ABSF AND TSTF
:*SOURCE FLOWS. THE NEGD INSTRUCTION
:*IS USED TO TEST MODE 1
:*
:*****

```
TST21: SCOPE
3412 014614 000004
3412 014616 012737 014624 001110 MOV #200$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
3413 014624 012700 014732 200$: MOV #210$, R0 ;SET UP THE DATA BUFFER.
3414 014630 012701 014762 MOV #220$, R1
3415 014634 012702 000004 MOV #4, R2
3416 014640 012021 1$: MOV (R0)+, (R1)+
3417 014642 077202 SOB R2, 1$
3418 014644 012700 000200 MOV #200, R0 ;SET FD MODE.
3419 014650 170100 LDFPS R0
3420 014652 012700 014762 MOV #220$, R0 ;SET UP THE OPERAND ADDRESS.
3421 014656 012737 014672 001236 MOV #230$, $TMP2
3422 014664 012737 014772 000004 MOV #240$, $ERRVECT ;SET UP VECTOR 4 IN CASE OF ERROR.
3423 014672 170710 230$: NEGD (R0) ;TEST INSTRUCTION.
3424
3425 014674 170205 STFPS R5 ;GET FPS.
3426 014676 012701 014762 MOV #220$, R1 ;SEE IF RESULT IS CORRECT.
3427 014702 012702 000004 MOV #4, R2
3428 014706 005721 2$: TST (R1)+
3429 014710 001046 BNE 250$ ;BRANCH IF NOT CORRECT.
3430 014712 077203 SOB R2, 2$
3431
3432 014714 020027 014762 CMP R0, #220$ ;IS R0 CORRECT?
3433 014720 001055 BNE 260$ ;BRANCH IF NOT CORRECT.
3434 014722 022705 000204 CMP #204, R5 ;IS THE FPS CORRECT?
3435 014726 001061 BNE 270$ ;BRANCH IF NOT CORRECT.
3436 014730 000466 BR 280$
3437
; THESE ARE TEST DATA TABLES AND A BUFFER.
3438
3439 014732 000177 210$: 177
3440 014734 167574 167574
3441 014736 137271 137271
3442 014740 107675 107675
3443 014742 000000 214$: 0
3444 014744 000000 0
3445 014746 000000 0
3446 014750 000000 0
3447 014752 177777 -1
3448 014754 177777 -1
3449 014756 177777 -1
3450 014760 177777 -1
3451 014762 177777 220$: -1
3452 014764 177777 -1
3453 014766 177777 -1
3454 014770 177777 -1
3455
; IF A TRAP TO 4 OCCURS COME HERE:
3456
3457 014772 011602 240$: MOV (SP), R2 ;SEE IF THE TRAP OCCURRED ON THE TEST INSTR.
3458 014774 020227 014674 CMP R2, #230$+2
```

```

3459 015000 001405          BEQ      3$          ;BRANCH IF YES.
3460 015002 020227 014676   CMP      R2,#230$+4
3461 015006 001402          BEQ      3$          ;BRANCH IF YES.
3462 015010 000137 051774   JMP      CPSPUR ;OTHERWISE GO REPORT A SPURIOUS TRAP TO 4.
3463          ;REPORT A FAILURE IN THE FDST FLOWS RESULTED IN AN ODD ADDRESS TRAP TO 4.
3464 015014 022626          3$: CMP      (SP)+,(SP)+ ;RESET THE STACK.
3465 015016 010237 001236   MOV      R2,$TMP2
3466 015022 104107          ERROR   +107        ;ODD ADRES
3467 015024 000430          BR       280$        ;BUT FDSTX IN ST 771
3468
3469          ;REPORT RESULT INCORRECT.
3470 015026 012737 014742 001242 250$: MOV      #214$,$TMP4
3471 015034 012737 014732 001240   MOV      #210$,$TMP3
3472 015042 012737 014762 001244   MOV      #220$,$TMP5
3473 015050 104073          ERROR   +73
3474 015052 000415          BR       280$
3475
3476          ;RO INCORRECT:
3477 015054 012737 014762 001240 260$: MOV      #220$,$TMP3
3478 015062 010037 001242          MOV      R0,$TMP4
3479 015066 104074          ERROR   +74          ;RO BADX
3480 015070 000406          BR       280$
3481
3482          ;REPORT FPS INCORRECT:
3483 015072 010537 001240 270$: MOV      R5,$TMP3
3484 015076 012737 000204 001244   MOV      #204,$TMP5
3485 015104 104075          ERROR   +75          ;FPS X
3486
3487 015106          280$: RSETUP
      015106 104412          ;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?).
    
```

3488

```

.SBTTL TEST # 22 - NEGF, ABSF & TSTF SRC MODE 2 TEST
*****
*TEST 22      NEGF, ABSF & TSTF SRC MODE 2 TEST
*
*THIS IS A TEST THE NEGF, ABSF AND TSTF
*SOURCE FLOWS. THE ABSD INSTRUCTION
*IS USED TO TEST MODE 2
*
*****
    
```

```

TST22: SCOPE
3489 015110 000004 015120 001110 200$: MOV #200$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
3490 015112 012737 012700 015226 MOV #210$, R0 ;SET UP THE DATA BUFFER.
3491 015124 012701 015256 MOV #220$, R1
3492 015130 012702 000004 MOV #4, R2
3493 015134 012021 1$: MOV (R0)+, (R1)+
3494 015136 077202 SOB R2, 1$
3495 015140 012700 000200 MOV #200, R0 ;SET FD.
3496 015144 170100 LDFPS R0
3497 015146 012700 015256 MOV #220$, R0 ;SET UP THE OPERAND ADDRESS.
3498 015152 012737 015166 001236 MOV #230$, $TMP2
3499 015160 012737 015266 000004 MOV #240$, $ERRVECT ;SET UP VECTOR 4 IN CASE OF AN ERROR.
3500
3501 015166 170620 230$: ABSD (R0)+ ;TEST INSTRUCTION.
3502
3503 015170 170205 STFPS R5 ;GET FPS.
3504 015172 012701 015256 MOV #220$, R1 ;CHECK RESULT.
3505 015176 012702 000004 MOV #4, R2
3506 015202 005721 2$: TST (R1)+
3507 015204 001046 BNE 250$ ;BRANCH IF INCORRECT.
3508 015206 077203 SOB R2, 2$
3509
3510 015210 020027 015266 CMP R0, #220$+10 ;IS R0 CORRECT?
3511 015214 001055 BNE 260$ ;BRANCH IF INCORRECT.
3512 015216 022705 000204 CMP #204, R5 ;IS THE FPS CORRECT?
3513 015222 001061 BNE 270$ ;BRANCH IF INCORRECT.
3514 015224 000466 BR 280$
3515
3516 ;THESE ARE TEST DATA TABLES AND DATA BUFFER.
3517 015226 000177 210$: 177
3518 015230 167574 167574
3519 015232 137271 137271
3520 015234 107675 107675
3521 015236 000000 290$: 0
3522 015240 000000 0
3523 015242 000000 0
3524 015244 000000 0
3525 015246 177777 -1
3526 015250 177777 -1
3527 015252 177777 -1
3528 015254 177777 -1
3529 015256 177777 220$: -1
3530 015260 177777 -1
3531 015262 177777 -1
3532 015264 177777 -1
3533
3534 ;IF A TRAP TO 4 OCCURS COME HERE.
3535 015266 011602 240$: MOV (SP), R2 ;SEE IF THE TRAP OCCURRED ON THE TEST INSTRUCTION.
    
```

```

3536 015270 020227 015170      CMP      R2,#230$+2
3537 015274 001405      BEQ      3$          ;BRANCH IF YES.
3538 015276 020227 015172      CMP      R2,#230$+4
3539 015302 001402      BEQ      3$          ;BRANCH IF YES.
3540 015304 000137 051774      JMP      CPSPUR ;OTHERWISE GO REPORT SPURIOUS TRAP TO 4.
3541          ;REPORT AN FDST FLOW FAILURE RESULTED IN A TRAP TO 4.
3542 015310 022626      3$:      CMP      (SP)+,(SP)+
3543 015312 010237 001236      MOV      R2,$TMP2
3544 015316 104076      ERROR   +76        ;ODD ADRES
3545 015320 000430      BR       280$      ;BUT FDSTX IN ST 771
3546
3547          ;REPORT RESULT INCORRECT:
3548 015322 012737 015236 001240      250$:   MOV      #290$,$TMP3
3549 015330 012737 015226 001242      MOV      #210$,$TMP4
3550 015336 012737 015256 001244      MOV      #220$,$TMP5
3551 015344 104077      ERROR   +77        ;BAD DATA X11*0 ST 312X
3552 015346 000415      BR       280$
3553
3554          ;REPORT RO INCORRECT:
3555 015350 012737 015262 001240      260$:   MOV      #220$+4,$TMP3
3556 015356 010037 001242      MOV      R0,$TMP4
3557 015362 104100      ERROR   +100       ;RO BADX
3558 015364 000406      BR       280$
3559
3560          ;REPORT FPS INCORRECT:
3561 015366 010537 001240      270$:   MOV      R5,$TMP3
3562 015372 012737 000204 001244      MOV      #204,$TMP5
3563 015400 104101      ERROR   +101       ;FPS X
3564
3565 015402      280$:   RSETUP          ;GO INITIALIZE THE FPS AND STACK; AND
          015402 104412          ;SEE IF THE USER HAS EXPRESSED
          ;THE DESIRE TO CHANGE THE SOFTWARE
          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
          ;THE USER TYPED CONTROL G?).
    
```

3566

```
.SBTTL TEST # 23 - NEGF, ABSF & TSTF SRC MODE 4 TEST
:*****
:*TEST 23      NEGF, ABSF & TSTF SRC MODE 4 TEST
:*
:*THIS IS A TEST THE NEGF, ABSF AND TSTF
:*SOURCE FLOWS. THE ABSD INSTRUCTION
:*IS USED TO TEST MODE 4
:*
:*****
```

```
TST23: SCOPE
3567 015404 000004      015414 001110      MOV      #200$, $LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.      ,DPM002
3568 015406 012737      015414 015522      MOV      #210$, R0          ;SET UP THE DATA BUFFER.
3569 015414 012700      015522      MOV      #220$, R1
3570 015420 012701      015542      MOV      #4, R2
3571 015424 012702      000004      MOV      (R0)+, (R1)+
3572 015430 012021      SOB      R2, 1$
3573 015432 077202      MOV      #200, R0          ;SET FD.
3574 015434 012700      000200      LDFPS   R0
3575 015440 170100      MOV      #230$, R0        ;SET UP THE OPERAND ADDRESS.
3576 015442 012700      015552      MOV      #240$, $TMP2
3577 015444 012737      015462 001236      MOV      #250$, $ERRVECT   ;SET UP VECTOR 4 IN CASE OF AN ERROR.
3578 015454 012737      015562 000004
3579 015462 170640      240$:   ABSD      -(R0)          ;TEST INSTRUCTION.
3580
3581 015464 170205      STFPS   R5                ;GET FPS.
3582 015466 012701      015542      MOV      #220$, R1        ;CHECK RESULT.
3583 015472 012702      000004      MOV      #4, R2
3584 015476 005721      2$:     TST      (R1)+
3585 015500 001046      BNE     260$              ;BRANCH IF INCORRECT.
3586 015502 077203      SOB     R2, 2$
3587
3588 015504 020027      015542      CMP     R0, #220$        ;IS R0 CORRECT?
3589 015510 001055      BNE     270$              ;BRANCH IF INCORRECT.
3590 015512 022705      000204      CMP     #204, R5         ;IS THE FPS CORRECT?
3591 015516 001061      BNE     280$              ;BRANCH IF INCORRECT.
3592 015520 000466      BR     290$
3593
3594      ;THESE ARE TEST DATA TABLES AND DATA BUFFER.
3595 015522 000177      210$:   177
3596 015524 117273      117273
3597 015526 147576      147576
3598 015530 177071      177071
3599 015532 000000      300$:   0
3600 015534 000000      0
3601 015536 000000      0
3602 015540 000000      0
3603 015542 177777      220$:   -1
3604 015544 177777      -1
3605 015546 177777      -1
3606 015550 177777      -1
3607 015552 177777      230$:   -1
3608 015554 177777      -1
3609 015556 177777      -1
3610 015560 177777      -1
3611
3612      ;IF A TRAP TO 4 OCCURS COME HERE.
3613 015562 011602      250$:   MOV      (SP), R2      ;SEE IF THE TRAP OCCURRED ON THE TEST INSTRUCTION.
```


3644

```
.SBTTL TEST # 24 - NEGF, ABSF & TSTF SRC MODE 3 TEST
:*****
:*TEST 24      NEGF, ABSF & TSTF SRC MODE 3 TEST
:*
:*THIS IS A TEST THE NEGF, ABSF AND TSTF
:*SOURCE FLOWS. THE ABSD INSTRUCTION
:*IS USED TO TEST MODE 3
:*
:*****
```

```
015700 000004
3645 015702 012737 015710 001110
3646 015710 012700 016016
3647 015714 012701 016046
3648 015720 012702 000010
3649 015724 012021
3650 015726 077202
3651 015730 012700 000200
3652 015734 170100
3653 015736 012700 016056
3654 015742 012737 015756 001236
3655 015750 012737 016066 000004
3656
3657 015756 170630
3658
3659 015760 170205
3660 015762 012701 016046
3661 015766 012702 000004
3662 015772 005721
3663 015774 001052
3664 015776 077203
3665 016000 020027 016060
3666 016004 001061
3667 016006 022705 000204
3668 016012 001065
3669 016014 000472
3670
3671
3672 016016 000177
3673 016020 147576
3674 016022 177071
3675 016024 107576 016046 177777
3676 016036 000000 000000 000000
3677 016046 177777
3678 016050 177777
3679 016052 177777
3680 016054 177777
3681 016056 177777
3682 016060 177777
3683 016062 177777
3684 016064 177777
3685
3686
3687 016066 011602
3688 016070 020227 015760
3689 016074 001405
3690 016076 020227 015762
3691 016102 001402
```

```
TST24: SCOPE
MOV #200$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
200$: MOV #210$, R0 ;SET UP THE DATA BUFFER.
MOV #220$, R1
MOV #10, R2
1$: MOV (R0)+, (R1)+
SOB R2, 1$
MOV #200, R0 ;SET FD.
LDFPS R0
MOV #230$, R0 ;SET UP THE OPERAND ADDRESS.
MOV #240$, $TMP2
MOV #250$, ERRVECT ;SET UP VECTOR 4 IN CASE OF AN ERROR.
240$: ABSD @ (R0)+ ;TEST INSTRUCTION.
STFPS R5 ;GET FPS.
MOV #220$, R1 ;CHECK RESULT.
2$: MOV #4, R2
TST (R1)+
BNE 260$ ;BRANCH IF INCORRECT.
SOB R2, 2$
CMP R0, #230$+2 ;IS R0 CORRECT?
BNE 270$ ;BRANCH IF INCORRECT.
CMP #204, R5 ;IS THE FPS CORRECT?
BNE 280$ ;BRANCH IF INCORRECT.
BR 290$

:THESE ARE TEST DATA TABLES AND DATA BUFFER.
210$: 177
147576
177071
107576, 220$, -1, -1, -1
300$: 0, 0, 0, 0
220$: -1
-1
-1
-1
230$: -1
-1
-1
-1

:IF A TRAP TO 4 OCCURS COME HERE.
250$: MOV (SP), R2 ;SEE IF THE TRAP OCCURRED ON THE TEST INSTRUCTION.
CMP R2, #240$+2
BEQ 3$ ;BRANCH IF YES.
CMP R2, #240$+4
BEQ 3$ ;BRANCH IF YES.
```

```

3692 016104 000137 051774
3693
3694 016110 022626
3695 016112 010237 001236
3696 016116 104106
3697 016120 000430
3698
3699
3700 016122 012737 016036 001240
3701 016130 012737 016016 001242
3702 016136 012737 016046 001244
3703 016144 104110
3704 016146 000415
3705
3706
3707 016150 012737 016060 001240
3708 016156 010037 001242
3709 016162 104111
3710 016164 000406
3711
3712 016166 010537 001240
3713 016172 012737 000204 001244
3714 016200 104112
3715
3716 016202
    016202 104412

;REPORT AN FDST FLOW FAILURE RESULTED IN A TRAP TO 4.
38$: JMP CPSPUR ; OTHERWISE GO REPORT SPURIOUS TRAP TO 4.
CMP (SP)+, (SP)+
MOV R2, $TMP2
ERROR +106 ; ODD ADRES
BR 290$ ; BUT FDSTX IN ST 771

;REPORT RESULT INCORRECT:
260$: MOV #300$, $TMP3
MOV #210$, $TMP4
MOV #220$, $TMP5
ERROR +110 ; BAD DATA X11=0 ST 3127
BR 290$

;REPORT RO INCORRECT:
270$: MOV #230$+2, $TMP3
MOV RO, $TMP4
ERROR +111 ; RO INCORRECT.
BR 290$

;REPORT FPS INCORRECT:
280$: MOV R5, $TMP3
MOV #204, $TMP5
ERROR +112 ; FPSX

290$: 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?).
    
```

3717

```

.SBTTL TEST # 25 - NEGF, ABSF & TSTF SRC MODE 5 TEST
*****
*TEST 25      NEGF, ABSF & TSTF SRC MODE 5 TEST
*
*THIS IS A TEST THE NEGF, ABSF AND TSTF
*SOURCE FLOWS. THE NEGD INSTRUCTION
*IS USED TO TEST MODE 5
*
*****
    
```

```

TST25: SCOPE
3718 016204 000004          016214 001110      MOV    #200$, $LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.      ;DPM002
3719 016206 012737          016322      MOV    #210$, R0          ;SET UP THE DATA BUFFER.
3720 016220 012701          016352      MOV    #220$, R1
3721 016224 012702          000010      MOV    #10, R2
3722 016230 012021          1$:      MOV    (R0)+, (R1)+
3723 016232 077202          SOB    R2, 1$
3724 016234 012700          000200      MOV    #200, R0          ;SET FD.
3725 016240 170100          LDFPS  R0
3726 016242 012700          016364      MOV    #230$+2, R0       ;SET UP THE OPERAND ADDRESS.
3727 016246 012737          016262      MOV    #240$, $TMP2
3728 016254 012737          016372      MOV    #250$, $ERRVECT  ;SET UP VECTOR 4 IN CASE OF AN ERROR.
3729
3730 016262 170750          240$:  NEGD  @-(R0)        ;TEST INSTRUCTION.
3731
3732 016264 170205          STFPS  R5                ;GET FPS.
3733 016266 012701          016352      MOV    #220$, R1          ;CHECK RESULT.
3734 016272 012702          000004      MOV    #4, R2
3735 016276 005721          2$:      TST    (R1)+
3736 016300 001052          BNE    260$              ;BRANCH IF INCORRECT.
3737 016302 077203          SOB    R2, 2$
3738 016304 020027          016362      CMP    R0, #230$        ;IS R0 CORRECT?
3739 016310 001061          BNE    270$              ;BRANCH IF INCORRECT.
3740 016312 022705          000204      CMP    #204, R5         ;IS THE FPS CORRECT?
3741 016316 001065          BNE    280$              ;BRANCH IF INCORRECT.
3742 016320 000472          BR     290$
3743
3744          ;THESE ARE TEST DATA TABLES AND DATA BUFFER.
3745 016322 000176          210$:  176
3746 016324 177074          177074
3747 016326 127374          127374
3748 016330 157677          016352  177777      157677, 220$, -1, -1, -1
3749 016342 000000          300$:  0
3750 016344 000000          0
3751 016346 000000          0
3752 016350 000000          0
3753 016352 177777          220$:  -1
3754 016354 177777          -1
3755 016356 177777          -1
3756 016360 177777          -1
3757 016362 177777          230$:  -1
3758 016364 177777          -1
3759 016366 177777          -1
3760 016370 177777          -1
3761
3762          ;IF A TRAP TO 4 OCCURS COME HERE.
3763 016372 011602          250$:  MOV    (SP), R2        ;SEE IF THE TRAP OCCURRED ON THE TEST INSTRUCTION.
3764 016374 020227          016264      CMP    R2, #240$+2
    
```

```

3765 016400 001405          BEQ      3$          ;BRANCH IF YES.
3766 016402 020227 016266  CMP      R2,#240$+4
3767 016406 001402          BEQ      3$          ;BRANCH IF YES.
3768 016410 000137 051774  JMP      CPSPUR ;OTHERWISE GO REPORT SPURIOUS TRAP TO 4.
3769          :REPORT AN FDST FLOW FAILURE RESULTED IN A TRAP TO 4.
3770 016414 022626          3$: CMP      (SP)+,(SP)+
3771 016416 010237 001236  MOV      R2,$TMP2
3772 016422 104113          ERROR   +113       ;ODD ADRES
3773 016424 000430          BR       290$      ;BUT FDSTX IN ST 771
3774
3775          :REPORT RESULT INCORRECT:
3776 016426 012737 016342 001240 260$: MOV      #300$,$TMP3
3777 016434 012737 016322 001242  MOV      #210$,$TMP4
3778 016442 012737 016352 001244  MOV      #220$,$TMP5
3779 016450 104114          ERROR   +114       ;BAD DATA X11*0 ST 3127
3780 016452 000415          BR       290$
3781
3782          :REPORT R0 INCORRECT:
3783 016454 012737 016362 001240 270$: MOV      #230$,$TMP3
3784 016462 010037 001242  MOV      R0,$TMP4
3785 016466 104115          ERROR   +115       ;R0 BADX
3786 016470 000406          BR       290$
3787          :REPORT FPS INCORRECT:
3788 016472 010537 001240 280$: MOV      R5,$TMP3
3789 016476 012737 000204 001244  MOV      #204,$TMP5
3790 016504 104116          ERROR   +116       ;FPSX
3791
3792 016506          290$: RSETUP          ;GO INITIALIZE THE FPS AND STACK; AND
          016506 104412          ;SEE IF THE USER HAS EXPRESSED
          ;THE DESIRE TO CHANGE THE SOFTWARE
          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
          ;THE USER TYPED CONTROL G?).
  
```

3793

```

.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 5
:
:*****
  
```

3794	016510	000004			TST26: SCOPE		
3795	016512	012767	016520	162370	.DSABL	AMA	:DISABLE MODE 6 TO MODE 3 CONVERSIONS
3796	016520	012700	016630		MOV	#200\$, \$LPERR	:SET UP THE LOOP ON ERROR ADDRESS.
3797	016524	012701	016652		200\$: MOV	#210\$, R0	:SET UP THE DATA BUFFER.
3798	016530	012702	000004		MOV	#220\$, R1	
3799	016534	012021			1\$: MOV	#4, R2	
3800	016536	077202			MOV	(R0)+, (R1)+	
3801	016540	012700	000200		SOB	R2, 1\$	
3802	016544	170100			MOV	#200, R0	:SET FD.
3803	016546	012700	016643		LDFPS	R0	
3804	016552	012767	016566	162456	MOV	#220\$-7, R0	:SET UP THE OPERAND ADDRESS.
3805	016560	012767	016672	161216	MOV	#230\$, \$TMP2	
3806					MOV	#240\$, ERRVECT	:SET UP VECTOR 4 IN CASE OF AN ERROR.
3807	016546	170660	000007		230\$: ABSD	7(R0)	:TEST INSTRUCTION.
3808							
3809	016572	170205			STFPS	R5	:GET FPS.
3810	016574	012701	016652		MOV	#220\$, R1	:CHECK RESULT.
3811	016600	012702	000004		MOV	#4, R2	
3812	016604	005721			2\$: TST	(R1)+	
3813	016606	001047			BNE	250\$:BRANCH IF INCORRECT.
3814	016610	077203			SOB	R2, 2\$	
3815	016612	020027	016643		CMP	R0, #220\$-7	:IS R0 CORRECT?
3816	016616	001056			BNE	260\$:BRANCH IF INCORRECT.
3817	016620	022705	000204		CMP	#204, R5	:IS THE FPS CORRECT?
3818	016624	001062			BNE	270\$:BRANCH IF INCORRECT.
3819	016626	000467			BR	280\$	
3820							
3821							
3822	016630	000177			:THESE ARE TEST DATA TABLES AND DATA BUFFER.		
3823	016632	161524			210\$:	177	
3824	016634	131273				161524	
3825	016636	107174	000000			131273	
3826	016642	000000			290\$:	107174	
3827	016644	000000				0	
3828	016646	000000				0	
3829	016650	000000				0	
3830	016652	177777			220\$:	-1	
3831	016654	177777				-1	
3832	016656	177777				-1	
3833	016660	177777				-1	
3834	016662	177777				-1	
3835	016664	177777				-1	
3836	016666	177777				-1	
3837	016670	177777				-1	
3838							
3839							
3840	016672	011602			:IF A TRAP TO 4 OCCURS COME HERE.		
					240\$: MOV	(SP), R2	:SEE IF THE TRAP OCCURRED ON THE TEST INSTRUCTION.

```

3841 016674 020227 016570          CMP      R2,#230$+2
3842 016700 001405          BEQ      3$          ;BRANCH IF YES.
3843 016702 020227 016572          CMP      R2,#230$+4
3844 016706 001402          BEQ      3$          ;BRANCH IF YES.
3845 016710 000167 033060          JMP      CPSPUR ;OTHERWISE GO REPORT SPURIOUS TRAP TO 4.
3846          ;REPORT AN FDST FLOW FAILURE RESULTED IN A TRAP TO 4.
3847 016714 022626          3$: CMP      (SP)+,(SP)+
3848 016716 010267 162314          MOV      R2,$TMP2
3849 016722 104117          ERROR   +117      ;ODD ADRES
3850 016724 000422          BR       270$     ;BUT FDSTX IN ST 771
3851
3852          ;REPORT RESULT INCORRECT:
3853 016726 012767 016642 162304 250$: MOV      #290$,$TMP3
3854 016734 012767 016630 162300          MOV      #210$,$TMP4
3855 016742 012767 016652 162274          MOV      #220$,$TMP5
3856 016750 104120          ERROR   +120      ;BAD DATA X11+0 ST 3127
3857 016752 000407          BR       270$
3858
3859          ;REPORT RO INCORRECT:
3860 016754 012767 016643 162256 260$: MOV      #220$-7,$TMP3
3861 016762 010067 162254          MOV      R0,$TMP4
3862 016766 104124          ERROR   +124      ;RO BADX
3863 016770 000400          BR       270$
3864          ;REPORT FPS INCORRECT:
3865 016772 010567 162242 270$: MOV      R5,$TMP3
3866 016776 012767 000204 162240          MOV      #204,$TMP5
3867 017004 104122          ERROR   +122      ;FPSX
3868 017006          280$: 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
3869          .ENABL  AMA

```

3870

```

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

```

TST27: SCOPE
3871 017010 000004          017020 001110      MOV    #200$, $LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.      ;DPM002
3872 017012 012737          017130      MOV    #210$, R0          ;SET UP THE DATA BUFFER.
3873 017024 012701          017160      MOV    #220$, R1
3874 017030 012702          000010      MOV    #10, R2
3875 017034 012021          1$:      MOV    (R0)+, (R1)+
3876 017036 077202          SOB    R2, 1$
3877 017040 012700          000200      MOV    #200, R0          ;SET FD.
3878 017044 170100          LDFPS  R0
3879 017046 012700          017161      MOV    #230$-7, R0       ;SET UP THE OPERAND ADDRESS.
3880 017052 012737          017066      MOV    #240$, $IMP2
3881 017060 012737          017200      MOV    #250$, ERRVECT    ;SET UP VECTOR 4 IN CASE OF AN ERROR.
3882
3883 017066 170770          000007      240$:  NEGD    @7(R0)      ;TEST INSTRUCTION.
3884
3885 017072 170205          STFPS  R5                ;GET FPS.
3886 017074 012701          017160      MOV    #220$, R1          ;CHECK RESULT.
3887 017100 012702          000004      MOV    #4, R2
3888 017104 005721          2$:      TST    (R1)+
3889 017106 001052          BNE    260$              ;BRANCH IF INCORRECT.
3890 017110 077203          SOB    R2, 2$
3891 017112 020027          017161      CMP    R0, #230$-7       ;IS R0 CORRECT?
3892 017116 001061          BNE    270$              ;BRANCH IF INCORRECT.
3893 017120 022705          000204      CMP    #204, R5          ;IS THE FPS CORRECT?
3894 017124 001065          BNE    280$              ;BRANCH IF INCORRECT.
3895 017126 000472          BR     290$
3896
3897          ;THESE ARE TEST DATA TABLES AND DATA BUFFER.
3898 017130 000177          210$:  177
3899 017132 167574          167574
3900 017134 137271          137271
3901 017136 107675          017160 177777      107675, 220$, -1, -1, -1
3902 017150 000000          300$:  0
3903 017152 000000          0
3904 017154 000000          0
3905 017156 000000          0
3906 017160 177777          220$:  -1
3907 017162 177777          -1
3908 017164 177777          -1
3909 017166 177777          -1
3910 017170 177777          230$:  -1
3911 017172 177777          -1
3912 017174 177777          -1
3913 017176 177777          -1
3914
3915          ;IF A TRAP TO 4 OCCURS COME HERE.
3916 017200 011602          250$:  MOV    (SP), R2        ;SEE IF THE TRAP OCCURRED ON THE TEST INSTRUCTION.
3917 017202 020227          017070      CMP    R2, #240$+2
    
```



```

3918 017206 001405          BEQ      3$          ;BRANCH IF YES.
3919 017210 020227 017072  CMP      R2,#240$+4
3920 017214 001402          BEQ      3$          ;BRANCH IF YES.
3921 017216 000137 051774  JMP      CPSPUR ;OTHERWISE GO REPORT SPURIOUS TRAP TO 4.
3922          ;REPORT AN FDST FLOW FAILURE RESULTED IN A TRAP TO 4.
3923 017222 022626          3$: CMP      (SP)+,(SP)+
3924 017224 010237 001236  MOV      R2,$TMP2
3925 017230 104123          ERROR   +123        ;ODD ADRES
3926 017232 000430          BR       290$       ;BUT FDSTX IN ST 771
3927
3928          ;REPORT RESULT INCORRECT:
3929 017234 012737 017150 001240 260$: MOV      #300$,$TMP3
3930 017242 012737 017130 001242  MOV      #210$,$TMP4
3931 017250 012737 017160 001244  MOV      #220$,$TMP5
3932 017256 104124          ERROR   +124        ;BAD DATA X11*0 ST 3127
3933 017260 000415          BR       290$
3934
3935          ;REPORT RO INCORRECT:
3936 017262 012737 017161 001240 270$: MOV      #230$-7,$TMP3
3937 017270 010037 001242  MOV      R0,$TMP4
3938 017274 104125          ERROR   +125        ;RO BADX
3939 017276 000406          BR       290$
3940          ;REPORT FPS INCORRECT:
3941 017300 010537 001240 280$: MOV      R5,$TMP3
3942 017304 012737 000204 001244  MOV      #204,$TMP5
3943 017312 104126          ERROR   +126        ;FPSX
3944
3945 017314          290$: RSETUP          ;GO INITIALIZE THE FPS AND STACK; AND
          104412          ;SEE IF THE USER HAS EXPRESSED
          ;THE DESIRE TO CHANGE THE SOFTWARE
          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
          ;THE USER TYPED CONTROL G?).
    
```

3946

```
.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
:*
:*****
```

```
017316 000004
3947          SCOPE
3948 017320 012767 017326 161562   .DSABL  AMA          ;DISABLE MODE 6 TO MODE 3 CONVERSIONS
3949 017326 012700 017424          MOV    #200$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS.      :DPM002
3950 017332 012701 017444 200$:  MOV    #210$, R0          ;SET UP THE DATA BUFFER.
3951 017336 012702 000004          MOV    #220$, R1
3952 017342 012021          MOV    #4, R2
3953 017344 077202 1$:      MOV    (R0)+, (R1)+
3954 017346 012700 000200          SOB    R2, 1$
3955 017352 170100          MOV    #200, R0          ;SET FD.
3956 017354 012767 017370 161654   LDFPS  R0
3957 017362 012767 017464 160414   MOV    #230$, $TMP2
3958          MOV    #240$, $ERRVECT ;SET UP VECTOR 4 IN CASE OF AN ERROR.
3959 017370 170767 000050 230$:  NEGD  220$          ;TEST INSTRUCTION.
3960          STFPS  R5          ;GET FPS.
3961 017374 170205          MOV    #220$, R1          ;CHECK RESULT.
3962 017376 012701 017444          MOV    #4, R2
3963 017402 012702 000004 2$:      TST    (R1)+
3964 017406 005721          BNE   250$          ;BRANCH IF INCORRECT.
3965 017410 001043          SOB    R2, 2$
3966 017412 077203 000204   CMP    #204, R5          ;IS THE FPS CORRECT?
3967 017414 022705          BNE   260$          ;BRANCH IF INCORRECT.
3968 017420 001052          BR    270$
3969 017422 000457
3970
3971          ;THESE ARE TEST DATA TABLES AND DATA BUFFER.
3972 017424 000127 210$:  127
3973 017426 137475          137475
3974 017430 147372          147372
3975 017432 117057          117057
3976 017434 000000 280$:  0
3977 017436 000000          0
3978 017440 000000          0
3979 017442 000000          0
3980 017444 177777 220$:  -1
3981 017446 177777          -1
3982 017450 177777          -1
3983 017452 177777          -1
3984 017454 177777          -1
3985 017456 177777          -1
3986 017460 177777          -1
3987 017462 177777          -1
3988
3989          ;IF A TRAP TO 4 OCCURS COME HERE.
3990 017464 011602 240$:  MOV    (SP), R2          ;SEE IF THE TRAP OCCURRED ON THE TEST INSTRUCTION.
3991 017466 020227 017372          CMP    R2, #230$+2
3992 017472 001405          BEQ   3$              ;BRANCH IF YES.
3993 017474 020227 017374          CMP    R2, #230$+4
```

```

3994 017500 001402          BEQ      3$          ;BRANCH IF YES.
3995 017502 000167 032266   JMP      CPSPUR    ;OTHERWISE GO REPORT SPURIOUS TRAP TO 4.
3996          :REPORT AN FDST FLOW FAILURE RESULTED IN A TRAP TO 4.
3997 017506 022626          3$:      CMP      (SP)+,(SP)+
3998 017510 010267 161522   MOV      R2,$TMP2
3999 017514 104127          ERROR   +127      ;ODD ADRES
4000 017516 000421          BR       270$     ;BUT FDSTX IN ST 771
4001
4002          :REPORT RESULT INCORRECT:
4003 017520 012767 017434 161512 250$:    MOV      #280,$TMP3
4004 017526 012767 017424 161506   MOV      #210,$TMP4
4005 017534 012767 017444 161502   MOV      #220,$TMP5
4006 017542 104130          ERROR   +130      ;BAD DATA X11*0 ST 3127
4007 017544 000406          BR       270$
4008          :REPORT FPS INCORRECT:
4009 017546 010567 161466 161464 260$:    MOV      R5,$TMP3
4010 017552 012767 000204          MOV      #204,$TMP5
4011 017560 104131          ERROR   +131      ;FPSX
4012
4013          270$:    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
4014          .ENABL  AMA
  
```

4015

.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
 :
 :*****

4016	017564	000004							
4017	017566	012737	017574	001110					
4018	017574	012700	017672						
4019	017600	012701	017722						
4020	017604	012702	000010						
4021	017610	012021							
4022	017612	077202							
4023	017614	012700	000200						
4024	017620	170100							
4025	017622	012737	017636	001236					
4026	017630	012737	017742	000004					
4027	017636	170677	000070						
4028									
4029	017642	170205							
4030	017644	012701	017722						
4031	017650	012702	000004						
4032	017654	005721							
4033	017656	001047							
4034	017660	077203							
4035	017662	022705	000204						
4036	017666	001056							
4037	017670	000463							
4038									
4039									
4040	017672	000137							
4041	017674	045607							
4042	017676	101230							
4043	017700	045607	017722	177777					
4044	017712	000000							
4045	017714	000000							
4046	017716	000000							
4047	017720	000000							
4048	017722	177777							
4049	017724	177777							
4050	017726	177777							
4051	017730	177777							
4052	017732	177777							
4053	017734	177777							
4054	017736	177777							
4055	017740	177777							
4056									
4057									
4058	017742	011602							
4059	017744	020227	017640						
4060	017750	001405							
4061	017752	020227	017642						
4062	017756	001402							

```

TST31: SCOPE
200$: MOV #200$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
      MOV #210$, R0 ;SET UP THE DATA BUFFER.
      MOV #220$, R1
      MOV #10, R2
1$: MOV (R0)+, (R1)+
     SOB R2, 1$
     MOV #200, R0 ;SET FD.
     LDFPS R0
     MOV #230$, $TMP2
     MOV #240$, ERRVECT ;SET UP VECTOR 4 IN CASE OF AN ERROR.
230$: ABSD @250$ ;TEST INSTRUCTION.
      STFPS R5 ;GET FPS.
      MOV #220$, R1 ;CHECK RESULT.
      MOV #4, R2
2$: TST (R1)+
     BNE 260$ ;BRANCH IF INCORRECT.
     SOB R2, 2$
     CMP #204, R5 ;IS THE FPS CORRECT?
     BNE 270$ ;BRANCH IF INCORRECT.
     BR 280$

;THESE ARE TEST DATA TABLES AND DATA BUFFER.
210$: 137
      045607
      101230
      45607, 220$, -1, -1, -1
290$: 0
      0
      0
      0
220$: -1
      -1
      -1
      -1
250$: -1
      -1
      -1
      -1

;IF A TRAP TO 4 OCCURS COME HERE.
240$: MOV (SP), R2 ;SEE IF THE TRAP OCCURRED ON THE TEST INSTRUCTION.
      CMP R2, #230$+2
      BEQ 3$ ;BRANCH IF YES.
      CMP R2, #230$+4
      BEQ 3$ ;BRANCH IF YES.
    
```

```

4063 017760 000137 051774          JMP      CPSPUR ;OTHERWISE GO REPORT SPURIOUS TRAP TO 4.
4064          :REPORT AN FDST FLOW FAILURE RESULTED IN A TRAP TO 4.
4065 017764 022626          $$$:    CMP      (SP)+,(SP)+
4066 017766 010237 001236          MOV      R2,$TMP2
4067 017772 104132          ERROR   +132          ;ODD ADRES
4068 017774 000421          BR      280$          ;BUT FDSTX IN ST 771
4069
4070          :REPORT RESULT INCORRECT:
4071 017776 012737 017712 001240 260$:  MOV      #290$,$TMP3
4072 020004 012737 017672 001242      MOV      #210$,$TMP4
4073 020012 012737 017722 001244      MOV      #220$,$TMP5
4074 020020 104133          ERROR   +133          ;BAD DATA X11+0 ST 3127
4075 020022 000406          BR      280$
4076          :REPORT FPS INCORRECT:
4077 020024 010537 001240 001244 270$:  MOV      R5,$TMP3
4078 020030 012737 000204          MOV      #204,$TMP5
4079 020036 104134          ERROR   +134          ;FPSX
4080
4081 020040          280$:  RSETUP          ;GO INITIALIZE THE FPS AND STACK; AND
      020040 104412          ;SEE IF THE USER HAS EXPRESSED
          ;THE DESIRE TO CHANGE THE SOFTWARE
          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
          ;THE USER TYPED CONTROL G?).
    
```

```

4088      .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.
          :*
          :*****
          TST32: SCOPE
4089 020042 000004      MOV      #200$, $LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.      ;DPM002
4090 020052 012737 020052 001110      MOV      #200$, RO      ;SET FD.
          LDFPS      RO
4091 020056 170100      MOV      #210$, RO      ;SET UP ACO.
4092 020060 012700 020146      LDD      (RO), ACO
4093 020064 172410      MOV      #220$, $TMP2
4094 020066 012737 020074 001236      NEGD      ACO      ;TEST INSTRUCTION.
4095
4096 020074 170700      STFPS     R5      ;GET FPS.
4097
4098 020076 170205      MOV      #200$, RO      ;SET FD.
4099 020100 012700 000200      LDFPS     RO
          MOV      #230$, RO      ;GET THE RESULT.
4100 020104 170100      STD      ACO, (RO)
          MOV      #230$, RO      ;IS THE RESULT CORRECT?
4101 020106 012700 020166      MOV      #240$, R1
          MOV      #4, R2
4102 020112 174010      1$:      CMP      (R0)+, (R1)+      ;BRANCH IF INCORRECT.
          BNE     250$
          SOB     R2, 1$
4103 020114 012700 020166      CMP      #210$, R5      ;IS THE FPS CORRECT?
4104 020120 012701 020156      BNE     260$      ;BRANCH IF INCORRECT.
4105 020124 012702 000004      BR      270$
4106 020130 022021
4107 020132 001021
4108 020134 077203
4109 020136 022705 000210
4110 020142 001033
4111 020144 000440
4112
4113      :THESE ARE DATA TABLES AND A DATA BUFFER.
4114 020146 013572      210$:      013572
4115 020150 046013      46013
4116 020152 057246      57246
4117 020154 013570      013570
4118 020156 113572      240$:      113572
4119 020160 046013      46013
4120 020162 057246      57246
4121 020164 013570      013570
4122 020166 000000      230$:      0
4123 020170 000000      0
4124 020172 000000      0
4125 020174 000000      0
4126
4127      :REPORT RESULT INCORRECT:
4128 020176 012737 020166 001240      250$:      MOV      #230$, $TMP3
4129 020204 012737 020156 001242      MOV      #240$, $TMP4
4130 020212 023737 020146 020166      CMP      210$, 230$
4131 020220 001002      BNE     280$
4132 020222 104135      ERROR   +135      ;E10*200X ST 336
4133 020224 000410      BR      270$
4134
4135      :REPORT RESULT INCORRECT:
4136 020226 104136      280$:      ERROR   +136      ;BAD DATA NEGF
    
```

4137 020230 000406

BR 270\$

4138

4139

4140 020232 010537 001242

:REPORT FPS INCORRECT:

4141 020236 012737 000210 001240

260\$: MOV R5,\$TMP4

4142 020244 104137

MOV #210,\$TMP3

4143

ERROR +137

:FPSX

4144 020246

270\$:

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?).

020246 104412

```

4145          .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.
               :*
               :*****
4146 020250 000004          TST33: SCOPE
4146 020252 012737 020260 001110  MOV    #200$, $LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.
4147 020260 012701 020370          MOV    #210$, R1          ;SET UP THE DATA BUFFER.
4148 020264 012700 020400          MOV    #220$, R0
4149 020270 012702 000004          MOV    #4, R2
4150 020274 012021          1$:   MOV    (R0)+, (R1)+
4151 020276 077202          SOB    R2, 1$
4152 020300 012700 020370          MOV    #210$, R0
4153 020304 042710 100000          BIC    #100000, (R0)    ;MAKE OPERAND POSITIVE.
4154 020310 012737 020324 001236  MOV    #230$, $TMP2
4155 020316 012701 000200          MOV    #200, R1        ;SET FD.
4156 020322 170101          LDFPS  R1
4157
4158 020324 170710          230$: NEGD   (R0)          ;TEST INSTRUCTION.
4159 020326 170205          STFPS  R5              ;GET FPS.
4160 020330 012701 020370          MOV    #210$, R1
4161 020334 012702 020400          MOV    #220$, R2
4162 020340 012703 000004          MOV    #4, R3
4163 020344 022122          2$:   CMP    (R1)+, (R2)+
4164 020346 001020          BNE    240$           ;BRANCH IF INCORRECT.
4165 020350 077303          SOB    R3, 2$
4166 020352 022700 020370          CMP    #210$, R0
4167 020356 001024          BNE    250$           ;IS R0 CORRECT.
4168 020360 022705 000210          CMP    #210, R5
4169 020364 001030          BNE    260$           ;IS THE FPS CORRECT?
4170 020366 000435          BR     270$           ;BRANCH IF INCORRECT.
4171
4172          ;THESE ARE DATA TABLES AND A DATA BUFFER.
4173 020370 023245          210$: 023245
4174 020372 026720          26720
4175 020374 122324          122324
4176 020376 052672          52672
4177 020400 123245          220$: 123245
4178 020402 026720          26720
4179 020404 122324          122324
4180 020406 052672          52672
4181
4182          ;REPORT RESULT INCORRECT:
4183 020410 012737 020370 001240  240$: MOV    #210$, $TMP3
4184 020416 012737 020400 001242  MOV    #220$, $TMP4
4185 020424 104140          ERROR  +140          ;BAD DATA
4186 020426 000415          BR     270$
4187
4188          ;REPORT RO INCORRECT:
4189 020430 012737 020370 001240  250$: MOV    #210$, $TMP3
4190 020436 010037 001242          MOV    R0, $TMP4
4191 020442 104141          ERROR  +141          ;SPEC DESTX
4192 020444 000406          BR     270$          ;ROX
4193
    
```


4194
4195 020446 012737 000210 001240 :REPORT FPS INCORRECT:
4196 020454 010537 001242 260\$: MOV #210,\$TMP3
4197 020460 104142 MOV R5,\$TMP4
4198 ERROR +142
4199 020462 270\$:
020462 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?).

4200

```
.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.
*
*****
```

```
TST34: SCOPE
4201 020464 000074 020474 001110 MOV #200$,SLPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
4202 020474 012701 020604 200$: MOV #210$,R1 ;SET UP THE DATA BUFFER.
4203 020500 012700 020614 MOV #220$,R0
4204 020504 012702 000004 MOV #4,R2
4205 020510 012021 1$: MOV (R0)+,(R1)+
4206 020512 077202 SOB R2,1$
4207 020514 012700 020604 MOV #210$,R0
4208 020520 042710 100000 BIC #100000,(R0) ;MAKE OPERAND POSITIVE.
4209 020524 012737 020540 001236 MOV #230$,STMP2
4210 020532 012701 000200 MOV #200,R1 ;SET FD.
4211 020536 170101 LDFPS R1
4212
4213 020540 170720 230$: NEGD (R0)+ ;TEST INSTRUCTION.
4214
4215 020542 170205 STFPS R5 ;GET FPS.
4216 020544 012701 020604 MOV #210$,R1 ;IS THE RESULT CORRECT.
4217 020550 012702 020614 MOV #220$,R2
4218 020554 012703 000004 MOV #4,R3
4219 020560 022122 2$: CMP (R1)+,(R2)+
4220 020562 001020 BNE 240$ ;BRANCH IF INCORRECT.
4221 020564 077303 SOB R3,2$
4222 020566 022700 020614 CMP #210$+10,R0 ;IS R0 CORRECT.
4223 020572 001024 BNE 250$ ;BRANCH IF INCORRECT.
4224 020574 022705 000210 CMP #210,R5 ;IS THE FPS CORRECT?
4225 020600 001030 BNE 260$ ;BRANCH IF INCORRECT.
4226 020602 000435 BR 270$
4227
;THESE ARE DATA TABLES AND A DATA BUFFER.
4228 210$: 023245
4229 020604 023245 26720
4230 020606 026720 122324
4231 020610 122324 52672
4232 020612 052672 220$: 123245
4233 020614 123245 26720
4234 020616 026720 122324
4235 020620 122324 52672
4236 070622 052672
4237
;REPORT RESULT INCORRECT:
4238 240$: MOV #210$,STMP3
4239 020624 012737 020604 001240 MOV #220$,STMP4
4240 020632 012737 020614 001242 ERROR +143 ;BAD DATA
4241 020640 104143 BR 270$
4242 020642 000415
4243
;REPORT R0 INCORRECT:
4244 250$: MOV #210$+10,STMP3
4245 020644 012737 020614 001240 MOV R0,STMP4
4246 020652 010037 001242 ERROR +144 ;SPEC DESTX ROX
4247 020655 104144 BR 270$
4248 020660 000406
```

```
4249  
4250  
4251 020662 012737 000210 001240 :REPORT FPS INCORRECT:  
4252 020670 010537 001242 260$: MOV #210,$TMP3  
4253 020674 104145          MOV R5,$TMP4  
4254          ERROR +145  
4255          270$: 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?).
```

4256

.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.
 :*
 :*****

```

4257 020700 000004
4258 020702 012737 020710 001110
4259 020710 012701 021022
4260 020714 012700 021042
4261 020720 012702 000004
4262 020724 012021
4263 020726 077202
4264 020730 012700 021032
4265 020734 042760 100000 177770
4266 020742 012737 020756 001236
4267 020750 012701 000200
4268 020754 170101
4269 020756 170740
4270
4271 020760 170205
4272 020762 012701 021022
4273 020766 012702 021042
4274 020772 012703 000004
4275 020776 022122
4276 021000 001024
4277 021002 077303
4278 021004 022700 021022
4279 021010 001030
4280 021012 022705 000210
4281 021016 001034
4282 021020 000441
4283
4284
4285 021022 023245
4286 021024 026720
4287 021026 122324
4288 021030 052672
4289 021032 177777 177777 177777
4290 021042 123245
4291 021044 026720
4292 021046 122324
4293 021050 052672
4294
4295
4296 021052 012737 021022 001240
4297 021060 012737 021042 001242
4298 021066 104146
4299 021070 000415
4300
4301
4302 021072 012737 021022 001240
4303 021100 010037 001242
4304 021104 104147

TST35: SCOPE
200$: MOV #200$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
      MOV #210$, R1 ;SET UP THE DATA BUFFER.
      MOV #220$, R0
      MOV #4, R2
1$: MOV (R0)+, (R1)+
   SOB R2, 1$
   MOV #210$+10, R0
   BIC #100000, -10(R0) ;MAKE OPERAND POSITIVE.
   MOV #230$, $TMP2
   MOV #200, R1 ;SET FD.
   LDFPS R1

230$: NEGD -(R0) ;TEST INSTRUCTION.

      STFPS R5 ;GET FPS.
      MOV #210$, R1 ;IS THE RESULT CORRECT.
      MOV #220$, R2
      MOV #4, R3
2$: CMP (R1)+, (R2)+
   BNE 240$ ;BRANCH IF INCORRECT.
   SOB R3, 2$
   CMP #210$, R0 ;IS R0 CORRECT.
   BNE 250$ ;BRANCH IF INCORRECT.
   CMP #210, R5 ;IS THE FPS CORRECT?
   BNE 260$ ;BRANCH IF INCORRECT.
   BR 270$

;THESE ARE DATA TABLES AND A DATA BUFFER.
210$: 023245
      26720
      122324
      52672
      .WORD -1, -1, -1, -1
220$: 123245
      26720
      122324
      52672

;REPORT RESULT INCORRECT:
240$: MOV #210$, $TMP3
      MOV #220$, $TMP4
      ERROR +146 ;BAD DATA
      BR 270$

;REPORT R0 INCORRECT:
250$: MOV #210$, $TMP3
      MOV R0, $TMP4
      ERROR +147 ;SPEC DESTX ROX
    
```

4305 021106 000406 BR 270\$
4306
4307 :REPORT FPS INCORRECT:
4308 021110 012737 000210 001240 260\$: MOV #210,\$TMP3
4309 021116 010537 001242 MOV R5,\$TMP4
4310 021122 104150 ERROR +150
4311
4312 021124 270\$: RSETUP
021124 104412

;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?).

4313

```
.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.
:*****
```

```
TST36: SCOPE
4314 021126 000004 021136 001110 200$: MOV #200$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
4315 021130 012737 021254 200$: MOV #210$, R1 ;SET UP THE DATA BUFFER.
4316 021136 012701 021264 MOV #220$, R0
4317 021142 012700 000004 MOV #4, R2
4318 021146 012702 1$: MOV (R0)+, (R1)+
4319 021152 012021 SOB R2, 1$
4320 021154 077202 MOV #230$, R0
4321 021156 012700 021274 MOV #210$, (R0)
4322 021162 012710 021254 BIC #100000, 210$ ;MAKE THE OPERAND POSITIVE.
4323 021166 042737 100000 021254 MOV #240$, $TMP2
4324 021174 012737 021210 001236 MOV #200, R1 ;SET FD.
4325 021202 012701 000200 LDFPS R1
4326 021206 170101
4327 021210 170730 240$: NEGD @ (R0)+ ;TEST INSTRUCTION.
4328
4329 021212 170205 STFPS R5 ;GET FPS.
4330 021214 012701 021254 MOV #210$, R1 ;IS THE RESULT CORRECT.
4331 021220 012702 021264 MOV #220$, R2
4332 021224 012703 000004 MOV #4, R3
4333 021230 022122 2$: CMP (R1)+, (R2)+
4334 021232 001021 BNE 250$ ;BRANCH IF INCORRECT.
4335 021234 077303 SOB R3, 2$
4336 021236 022700 021276 CMP #230$+2, R0 ;IS R0 CORRECT.
4337 021242 001025 BNE 260$ ;BRANCH IF INCORRECT.
4338 021244 022705 000210 CMP #210, R5 ;IS THE FPS CORRECT?
4339 021250 001031 BNE 270$ ;BRANCH IF INCORRECT.
4340 021252 000436 BR 280$
4341
4342 ;THESE ARE DATA TABLES AND A DATA BUFFER.
4343 021254 023245 210$: 023245
4344 021256 026720 26720
4345 021260 122324 122324
4346 021262 052672 52672
4347 021264 123245 220$: 123245
4348 021266 026720 26720
4349 021270 123324 123324
4350 021272 052672 52672
4351 021274 021254 230$: 210$
4352
4353 ;REPORT RESULT INCORRECT:
4354 021276 012737 021254 001240 250$: MOV #210$, $TMP3
4355 021304 012737 021264 001242 MOV #220$, $TMP4
4356 021312 104150 ERROR +150 ;BAD DATA
4357 021314 000415 BR 280$
4358
4359 ;REPORT R0 INCORRECT:
4360 021316 012737 021276 001240 260$: MOV #230$+2, $TMP3
4361 021324 010037 001242 MOV R0, $TMP4
```

4362 021330 104152
4363 021332 000406
4364
4365
4366 021334 012737 000210 001240
4367 021342 010537 001242
4368 021346 104153
4369
4370 021350
021350 104412

ERROR +152 ;SPEC DESTX ROX
BR 280\$
:REPORT FPS INCORRECT:
270\$: MOV #210,\$TMP3
MOV R5,\$TMP4
ERROR +153
280\$: 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?).

4371

```

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

```

TST37: SCOPE
4372 021352 000004 021362 001110 MOV #200$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
4373 021354 012737 021502 200$: MOV #210$, R1 ;SET UP THE DATA BUFFER.
4374 021362 012701 021512 MOV #220$, R0
4375 021372 012702 000004 MOV #4, R2
4376 021376 012021 1$: MOV (R0)+, (R1)+
4377 021400 077202 SOB R2, 1$
4378 021402 012700 021524 MOV #230$+2, R0
4379 021406 012760 021502 177776 MOV #210$, -2(R0)
4380 021414 042737 100000 021502 BIC #100000, 210$ ;MAKE THE OPERAND POSITIVE.
4381 021422 012737 021436 001236 MOV #240$, $TMP2
4382 021430 012701 000200 MOV #200, R1 ;SET FD.
4383 021434 170101 LDFPS R1
4384
4385 021436 170750 240$: NEGD @-(R0) ;TEST INSTRUCTION.
4386
4387 021440 170205 STFPS R5 ;GET FPS.
4388 021442 012701 021502 MOV #210$, R1 ;IS THE RESULT CORRECT.
4389 021446 012702 021512 MOV #220$, R2
4390 021452 012703 000004 MOV #4, R3
4391 021456 022122 2$: CMP (R1)+, (R2)+ ;BRANCH IF INCORRECT.
4392 021460 001021 BNE 250$
4393 021462 077303 SOB R3, 2$
4394 021464 022700 021522 CMP #230$, R0 ;IS R0 CORRECT.
4395 021470 001025 BNE 260$ ;BRANCH IF INCORRECT.
4396 021472 022705 000210 CMP #210, R5 ;IS THE FPS CORRECT?
4397 021476 001031 BNE 270$ ;BRANCH IF INCORRECT.
4398 021500 000436 BR 280$
4399
4400 :THESE ARE DATA TABLES AND A DATA BUFFER.
4401 021502 023245 210$: 023245
4402 021504 026720 26720
4403 021506 122324 122324
4404 021510 052672 52672
4405 021512 123245 220$: 123245
4406 021514 026270 26270
4407 021516 122324 122324
4408 021520 052672 52672
4409 021522 021502 230$: 210$
4410
4411 :REPORT RESULT INCORRECT:
4412 021524 012737 021502 001240 250$: MOV #210$, $TMP3
4413 021532 012737 021512 001242 MOV #220$, $TMP4
4414 021540 104154 ERROR +154 ;BAD DATA
4415 021542 000415 BR 280$
4416
4417 :REPORT R0 INCORRECT:
4418 021544 012737 021522 001240 260$: MOV #230$, $TMP3
4419 021552 010037 001242 MOV R0, $TMP4
    
```


4420 021556 104155
4421 021560 000406

ERROR +155
BR 280\$

:SPEC DESTX ROX

4422
4423
4424 021562 012737 000210 001240
4425 021570 010537 001242
4426 021574 104156

:REPORT FPS INCORRECT:
270\$: MOV #210,\$TMP3
MOV R5,\$TMP4
ERROR +156

4427
4428 021576 104412
021576

280\$: 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?).

4429

```

.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.
:*
:*****
  
```

```

TST40: SCOPE
4430 021600 000004 021610 001110 MOV #200$,SLPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
4431 021602 012737 021720 200$: MOV #210$,R1 ;SET UP THE DATA BUFFER.
4432 021610 012701 021720 MOV #220$,R0
4433 021614 012700 021730 MOV #4,R2
4434 021620 012702 000004 1$: MOV (R0)+,(R1)+
4435 021624 012021 SOB R2,1$
4436 021626 077202 MOV #210$,R0
4437 021630 012700 021720 BIC #100000,(R0) ;MAKE OPERAND POSITIVE.
4438 021634 042710 100000 001236 MOV #230$,STMP2
4439 021640 012737 021654 MOV #000,R1 ;SET FD.
4440 021646 012701 000000 LDFPS R1
4441 021652 170101
4442 021654 170720 230$: NEGF (R0)+ ;TEST INSTRUCTION.
4443 STFPS R5 ;GET FPS.
4444 021656 170205 MOV #210$,R1 ;IS THE RESULT CORRECT.
4445 021660 012701 021720 MOV #220$,R2
4446 021664 012702 021730 MOV #4,R3
4447 021670 012703 000004 2$: CMP (R1)+,(R2)+
4448 021674 022122 BNE 240$ ;BRANCH IF INCORRECT.
4449 021676 001020 SOB R3,2$
4450 021700 077303 CMP #210$+4,R0 ;IS R0 CORRECT.
4451 021702 022700 021724 BNE 250$ ;BRANCH IF INCORRECT.
4452 021706 001024 000010 CMP #010,R5 ;IS THE FPS CORRECT?
4453 021710 022705 BNE 260$ ;BRANCH IF INCORRECT.
4454 021714 001030 BR 270$
4455 021716 000435
4456 ;THESE ARE DATA TABLES AND A DATA BUFFER.
4457 210$: 023245
4458 021720 023245 26720
4459 021722 026720 122324
4460 021724 122324 52672
4461 021726 052672 220$: 123245
4462 021730 123245 26720
4463 021732 026720 122324
4464 021734 122324 52672
4465 021736 052672
4466 ;REPORT RESULT INCORRECT:
4467 240$: MOV #210$,STMP3
4468 021740 012737 021720 001240 MOV #220$,STMP4
4469 021746 012737 021730 001242 ERROR +150 ;BAD DATA
4470 021754 104150 BR 270$
4471 021756 000415
4472 ;REPORT R0 INCORRECT:
4473 250$: MOV #210$+4,STMP3
4474 021760 012737 021724 001240 MOV R0,STMP4
4475 021766 010037 001242 ERROR +160 ;SPEC DESTX ROX
4476 021772 104160 BR 270$
4477 021774 000406
  
```

4478
4479
4480 021776 012737 000010 001240 :REPORT FPS INCORRECT:
4481 022004 010537 001242 260\$: MOV #010,\$TMP3
4482 022010 104161 MOV R5,\$TMP4
4483 ERROR +161
4484 022012 270\$: RSETUP
022012 104412

: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?).

4485

```
.SBTTL TEST # 41 - SPECIAL DEST, MODE2, GR7
*****
*TEST 41      SPECIAL DEST, MODE2, GR7
*
*THIS IS A TEST OF THE NEG, ABS, AND TST DESTINATION FLOWS
*MODE 2(IMMEDIATE) USING THE NEG, INSTR.
*
*****
```

```
TST41: SCOPE
4486 022014 000004 022024 001110      MOV    #200$, $LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.      ;DPM002
4487 022016 012737 022150      MOV    #210$, R0
4488 022030 012701 022076      MOV    #220$, R1      ;SET UP THE DATA BUFFER.
4489 022034 012702 000004      MOV    #4, R2
4490 022040 012021      MOV    (R0)+, (R1)+
4491 022042 077202      SOB    R2, 1$
4492 022044 012700 022076      MOV    #220$, R0
4493 022050 042737 100000 022076      BIC    #100000, 220$    ;MAKE THE OPERAND POSITIVE.
4494 022056 012737 022074 001236      MOV    #230$, $TMP2
4495 022064 012701 000200      MOV    #200$, R1      ;SET FD.
4496 022070 170101      LDFPS R1
4497 022072 005001      CLR   R1
4498
4499 022074 170727      230$:  NEG    (R7)+      ;TEST INSTRUCTION.
4500 022076 005201 005201 005201 220$:  5201, 5201, 5201, 5201
4501      ;NOTE THAT AFTER EXECUTING THIS INSTRUCTION R1 SHOULD CONTAIN 3.
4502 022106 170205      STFPS R5      ;GET FPS.
4503 022110 012703 022076      MOV    #220$, R3      ;IS THE RESULT CORRECT.
4504 022114 012702 022150      MOV    #210$, R2
4505 022120 012704 000004      MOV    #4, R4
4506 022124 022322      2$:   CMP    (R3)+, (R2)+
4507 022126 001014      BNE   240$      ;BRANCH IF INCORRECT.
4508 022130 077403      SOB   R4, 2$
4509 022132 022701 000003      CMP   #3, R1      ;WAS R1 INCREMENTED CORRECTLY.
4510 022136 001027      BNE   250$      ;BRANCH IF INCORRECT.
4511 022140 022705 000210      CMP   #210, R5    ;IS THE FPS CORRECT?
4512 022144 001015      BNE   260$      ;BRANCH IF INCORRECT.
4513 022146 000436      BR    270$
4514
4515      ;THESE ARE DATA TABLE.
4516 022150 105201      210$:  105201
4517 022152 005201      5201
4518 022154 005201      5201
4519 022156 005201      5201
4520
4521      ;REPORT RESULT INCORRECT:
4522 022160 012737 022076 001240 240$:  MOV    #220$, $TMP3
4523 022166 012737 022150 001242      MOV    #210$, $TMP4
4524 022174 104162      ERROR +162      ;BAD DATA
4525 022176 000422      BR    270$
4526
4527      ;REPORT FPS INCORRECT:
4528 022200 012737 000210 001240 260$:  MOV    #210, $TMP3
4529 022206 010537 001242      MOV    R5, $TMP4
4530 022212 104163      ERROR +163      ;FPS
4531 022214 000413      BR    270$
4532
4533      ;REPORT PC INCORRECTLY INCREMENTED DURING EXECUTION.
```

4534 022216 162701 000003
4535 022222 006301
4536 022224 012702 022100
4537 022230 010237 001240
4538 022234 160102
4539 022236 010237 001242
4540 022242 104164
4541
4542 022244
022244 104412

250\$: SUB #3,R1
ASL R1
MOV #220\$+2,R2
MOV R2,\$TMP3
SUB R1,R2
MOV R2,\$TMP4
ERROR +164

;PC BAD CONSTAND B GR7X

270\$: 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?).

4543

.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 NEGJ INSTR.
 :*
 :*****

022246	000004				TST42: SCOPE		
4544					.DSABL	AMA	:DISABLE MODE 6 TO MODE 3 CONVERSIONS
4545	022250	012767	022256	156632	MOV	#200\$, \$LPERR	:SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
4546	022256	012701	022400		200\$: MOV	#210\$, R1	:SET UP THE DATA BUFFER.
4547	022262	012700	022410		MOV	#220\$, R0	
4548	022266	012702	000004		MOV	#4, R2	
4549	022272	012021			1\$: MOV	(R0)+, (R1)+	
4550	022274	077202			SOB	R2, 1\$	
4551	022276	012700	015177		MOV	#210\$-5201, R0	
4552	022302	042767	100000	000070	BIC	#100000, 210\$:MAKE OPERAND POSITIVE.
4553	022310	012767	022326	156720	MOV	#230\$, \$TMP2	
4554	022316	012701	000200		MOV	#200, R1	:SET FD.
4555	022322	170101			LDFPS	R1	
4556							
4557	022324	005001			230\$: CLR	R1	
4558	022326	170760	005201		NEGJ	5201(R0)	:TEST INSTRUCTION.
4559	022332	170205			STFPS	R5	:GET FPS.
4560	022334	005701			TST	R1	
4561	022336	001030			BNE	240\$:WAS THE PC CORRECT AFTER EXECUTION?
4562	022340	012701	022400		MOV	#210\$, R1	:IS THE RESULT CORRECT.
4563	022344	012702	022410		MOV	#220\$, R2	
4564	022350	012703	000004		MOV	#4, R3	
4565	022354	022122			2\$: CMP	(R1)+, (R2)+	
4566	022356	001030			BNE	250\$:BRANCH IF INCORRECT.
4567	022360	077303			SOB	R3, 2\$	
4568	022362	022700	015177		CMP	#210\$-5201, R0	:IS R0 CORRECT.
4569	022366	001034			BNE	260\$:BRANCH IF INCORRECT.
4570	022370	022705	000210		CMP	#210, R5	:IS THE FPS CORRECT?
4571	022374	001040			BNE	270\$:BRANCH IF INCORRECT.
4572	022376	000445			BR	280\$	
4573							
4574							
4575	022400	023245			210\$:	023245	
4576	022402	026720				26720	
4577	022404	122324				122324	
4578	022406	052672				52672	
4579	022410	123245			220\$:	123245	
4580	022412	026720				26720	
4581	022414	122324				122324	
4582	022416	052672				52672	
4583							
4584							
4585							
4586	022420	012767	022330	156614	:REPORT PC INCORRECT AFTER EXECUTION.		
4587	022426	012767	022332	156604	240\$: MOV	#230\$+2, \$TMP4	
4588	022434	104215			MOV	#230\$+4, \$TMP3	
4589	022436	000425			ERROR	+215	:PC NOT INCREMENTED BY 2.
4590					BR	280\$	
4591							

:REPORT RESULT INCORRECT:

```

4592 022440 012767 022400 156572 250$: MOV #210$,STMP3
4593 022446 012767 022410 156566 MOV #220$,STMP4
4594 022454 104216 ERROR +216 ;BAD DATA
4595 022456 000415 BR 280$
4596
4597 ;REPORT R0 INCORRECT:
4598 022460 012767 015177 156552 260$: MOV #210$-5201,$STMP3
4599 022466 010067 156550 MOV R0,$STMP4
4600 022472 104217 ERROR +217 ;SPEC DESTX R0X
4601 022474 000406 BR 280$
4602
4603
4604 ;REPORT FPS INCORRECT:
4605 022476 012767 000210 156534 270$: MOV #210,$STMP3
4606 022504 010567 156532 MOV R5,$STMP4
4607 022510 104220 ERROR +220
4608
4609 022512 280$: RSETJP ;GO INITIALIZE THE FPS AND STACK; AND
022512 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

4610 .ENABL AMA
    
```

4611

.....
 :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.
 :.....

4612	022514	000004							
4613	022516	012737	022524	001110					
4614	022524	012701	022654						
4615	022530	012700	022664						
4616	022534	012702	000004						
4617	022540	012021							
4618	022542	077202							
4619	022544	012700	015473						
4620	022550	012760	022654	005201					
4621	022556	042737	100000	022654					
4622	022564	012737	022602	001236					
4623	022572	012701	000200						
4624	022576	170101							
4625	022600	005001							
4626	022602	170770	005201						
4627									
4628	022606	170205							
4629	022610	005701							
4630	022612	001031							
4631	022614	012701	022654						
4632	022620	012702	022664						
4633	022624	012703	000004						
4634	022630	022122							
4635	022632	001031							
4636	022634	077303							
4637	022636	022700	015473						
4638	022642	001035							
4639	022644	022705	000210						
4640	022650	001041							
4641	022652	000446							
4642									
4643									
4644	022654	023245							
4645	022656	026720							
4646	022660	122324							
4647	022662	052672							
4648	022664	123245							
4649	022666	026720							
4650	022670	123324							
4651	022672	052672							
4652	022674	022654							
4653									
4654									
4655	022676	013737	022604	001242					
4656	022704	013737	022606	001240					
4657	022712	104221							
4658	022714	000425							
4659									

```

TST43: SCOPE
200$: MOV #200$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPH002
      MOV #210$, R1 ;SET UP THE DATA BUFFER.
      MOV #220$, R0
      MOV #4, R2
1$: MOV (R0)+, (R1)+
     SOB R2, 1$
     MOV #230$-5201, R0
     MOV #210$, 5201(R0)
     BIC #100000, 210$ ;MAKE THE OPERAND POSITIVE.
     MOV #240$, $TMP2
     MOV #200, R1 ;SET FD.
     LDFPS R1

240$: CLR R1
     NEGD @5201(R0) ;TEST INSTRUCTION.

     STFPS R5 ;GET FPS.
     TST R1 ;WAS THE PC CORRECT AFTER EXECUTION?
     BNE 250$
     MOV #210$, R1 ;IS THE RESULT CORRECT.
     MOV #220$, R2
     MOV #4, R3
2$: CMP (R1)+, (R2)+
     BNE 260$ ;BRANCH IF INCORRECT.
     SOB R3, 2$
     CMP #230$-5201, R0 ;IS R0 CORRECT.
     BNE 270$ ;BRANCH IF INCORRECT.
     CMP #210, R5 ;IS THE FPS CORRECT?
     BNE 280$ ;BRANCH IF INCORRECT.
     BR 290$

;THESE ARE DATA TABLES AND A DATA BUFFER.
210$: 023245
      26720
      122324
      52672
220$: 123245
      26720
      123324
      52672
230$: 210$

;REPORT PC INCORRECT AFTER EXECUTION.
250$: MOV 240$+2, $TMP4
      MOV 240$+4, $TMP3
      ERROR +221 ;PC NOT INCREMENTED BY 2.
      BR 290$
    
```



```
4660
4661 022716 012737 022654 001240 ;REPORT RESULT INCORRECT:
4662 022724 012737 022664 001242 260$: MOV #210$, $TMP3
4663 022732 104222 ERROR +222 ;BAD DATA
4664 022734 000415 BR 290$
4665
4666 ;REPORT R0 INCORRECT:
4667 022736 012737 015473 001240 270$: MOV #230$-5201, $TMP3
4668 022744 010037 001242 MOV R0, $TMP4
4669 022750 104223 ERROR +223 ;SPEC DESTX ROX
4670 022752 000406 BR 290$
4671
4672 ;REPORT FPS INCORRECT:
4673 022754 012737 000210 001240 280$: MOV #210, $TMP3
4674 022762 010537 001242 MOV R5, $TMP4
4675 022766 104224 ERROR +224
4676
4677 022770 290$: RSETUP
022770 104412 ;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?).
```

4683

```
.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.
*
*****
```

```
022772 000004
4684
4685 022774 012737 023002 001110
4686 023002 004737 023702
4687 023006 000000
4688 023010 016341
4689 023012 055772
4690 023014 021133
4691 023016 055447
4692 023020 116341
4693 023022 055772
4694 023024 021133
4695 023026 055447
4696 023030 016341
4697 023032 055772
4698 023034 021133
4699 023036 055447
4700 023040 000207
4701 023042 000210
4702 023044 000200
4703 023046 177777
4704 023050 104200
4705 023052 000401
4706 023054 104201
4707 023056
```

```
TST44: SCOPE
:TEST NEGD WITH POS NONZERO OPERAND
MOV #200$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
200$: JSR PC,1000$
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$:
```

```
4708
4709 023056 012737 023064 001110
4710 023064 004737 023702
4711 023070 000000
4712 023072 152525
4713 023074 053545
4714 023076 055565
4715 023100 057505
4716 023102 052525
4717 023104 053545
4718 023106 055565
4719 023110 057505
4720 023112 152525
4721 023114 053545
4722 023116 055565
4723 023120 057505
4724 023122 000217
4725 023124 000200
4726 023126 000210
4727 023130 177777
4728 023132 104200
4729 023134 000401
4730 023136 104202
4731 023140
4732
```

```
:TEST NEGD WITH NEG OPERAND.
MOV #210$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
210$: JSR PC,1000$
11$: 0 ;FLAG=NEGD.
12$: 152525 ;OPERAND.
53545
55565
57505
13$: 52525 ;RESULT.
53545
55565
57505
14$: 152525 ;ERROR RES.
53545
55565
57505
15$: 217 ;FPS BEFORE EXECUTION.
200 ;FPS AFTER EXECUTION.
210 ;ERROR FPS.
-1 ;FEC
16$: ERROR +200 ;E10<---E10*200X S336
BR 17$
ERROR +201 ;BUT ENBT X ST336 TO 453 INTO 053
17$:
```

```
:TEST ABSD WITH POSITIVE OPERAND
```

4733	023140	012737	023146	001110	MOV	#220\$, \$LPERR	;SET UP THE LOOP ON ERROR ADDRESS.	;DPM002
4734	023146	004737	023702	220\$:	JSR	PC,1000\$		
4735	023152	000001		21\$:	1		;FLAG=ABSD.	
4736	023154	060705		22\$:	60705		;OPERAND.	
4737	023156	124735			124735			
4738	023160	060124			60124			
4739	023162	073560			73560			
4740	023164	060705		23\$:	60705		;RESULT.	
4741	023166	124735			124735			
4742	023170	060124			60124			
4743	023172	073560			73560			
4744	023174	160705		24\$:	160705		;ERROR RES.	
4745	023176	124735			124735			
4746	023200	060124			60124			
4747	023202	073560			73560			
4748	023204	000217		25\$:	217		;FPS BEFORE EXECUTION.	
4749	023206	000200			200		;FPS AFTER EXECUTION.	
4750	023210	000210			210		;ERROR FPS.	
4751	023212	177777			-1		;EITHER BUT OP1B	
4752	023214	104203		26\$:	ERROR	+203	;BUT ST 055 TO 336 INTO 335	
4753	023216	000401			BR	27\$		
4754	023220	104203			ERROR	+203	;OR BUT ENBT ST 335 TO 452 INTO 052	
4755	023222			27\$:				
4756					;TEST	ABSD WITH NEG. OPERAND		
4757	023222	012737	023230	001110	MOV	#230\$, \$LPERR	;SET UP THE LOOP ON ERROR ADDRESS.	;DPM002
4758	023230	004737	023702	230\$:	JSR	PC,1000\$		
4759	023234	000001		31\$:	1		;FLAG=ABSD.	
4760	023236	154345		32\$:	154345		;OPERAND.	
4761	023240	076567			76567			
4762	023242	032123			32123			
4763	023244	043234			43234			
4764	023246	054345		33\$:	54345		;RESULT.	
4765	023250	076567			76567			
4766	023252	032123			32123			
4767	023254	043234			43234			
4768	023256	154345		34\$:	154345		;ERROR RES.	
4769	023260	076567			76567			
4770	023262	032123			32123			
4771	023264	043234			43234			
4772	023266	000217		35\$:	217		;FPS BEFORE EXECUTION.	
4773	023270	000200			200		;FPS AFTER EXECUTION.	
4774	023272	177777			-1		;ERROR FPS.	
4775	023274	177777			-1			
4776	023276	104204		36\$:	ERROR	+204	;E10*E10*200X ST 452	
4777	023300	000401			BR	37\$		
4778	023302	104171			ERROR	+171		
4779	023304			37\$:				
4780					;TEST	WITH POSITIVE OP		
4781	023304	012737	023312	001110	MOV	#240\$, \$LPERR	;SET UP THE LOOP ON ERROR ADDRESS.	;DPM002
4782	023312	004737	023702	240\$:	JSR	PC,1000\$		
4783	023316	000002		41\$:	2		;FLAG=TSTD.	
4784	023320	012321		42\$:	12321		;OPERAND.	
4785	023322	045654			45654			
4786	023324	070107			70107			
4787	023326	034543			34543			
4788	023330	012321		43\$:	12321		;RESULT.	
4789	023332	045654			45654			

4790	023334	070107				70107		
4791	023336	034543				34543		
4792	023340	112321			44\$:	112321		:ERROR RES.
4793	023342	045654				45654		
4794	023344	070107				70107		
4795	023346	034543				34543		
4796	023350	000217			45\$:	217		:FPS BEFORE EXECUTION.
4797	023352	000200				200		:FPS AFTER EXECUTION.
4798	023354	000210				210		:ERROR FPS.
4799	023356	177777				-1		
4800	023360	104205			46\$:	ERROR +205		:BUT (OP1B) X ST044 TO 336 INTO 334
4801	023362	000401				BR 47\$		
4802	023364	104206				ERROR +206		:BUT ENBT ST 334 TO 453 INTO 053
4803	023366				47\$:			
4804						:TEST TSTD WITH NEG OP		
4805	023366	012737	023374	001110		MOV #250\$, \$LPERR		:SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
4806	023374	004737	023702		250\$:	JSR PC,1000\$		
4807	023400	000002			51\$:	2		:FLAG=TSTD.
4808	023402	123765			52\$:	123765		:OPERAND.
4809	023404	023407				23407		
4810	023406	034510				34510		
4811	023410	045621				45621		
4812	023412	123765			53\$:	123765		:RESULT.
4813	023414	023407				23407		
4814	023416	034510				34510		
4815	023420	045621				45621		
4816	023422	023765			54\$:	23765		:ERROR RES.
4817	023424	023407				23407		
4818	023426	034510				34510		
4819	023430	045621				45621		
4820	023432	000207			55\$:	207		:FPS BEFORE EXECUTION.
4821	023434	000210				210		:FPS AFTER EXECUTION.
4822	023436	000200				200		:ERROR FPS.
4823	023440	177777				-1		
4824	023442	104207			56\$:	ERROR +207		:BUT OPB1 ST 055 TO 335 INTO 334
4825	023444	000401				BR 57\$		
4826	023446	104210				ERROR +210		:BUT ENBT ST 334 TO 053 INTO 453
4827	023450				57\$:			
4828						:TEST TSTD 0 OP		
4829	023450	012737	023456	001110		MOV #260\$, \$LPERR		:SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
4830	023456	004737	023702		260\$:	JSR PC,1000\$		
4831	023462	000002			61\$:	2		:FLAG=TSTD.
4832	023464	000175			62\$:	175		:OPERAND.
4833	023466	176737				176737		
4834	023470	071727				71727		
4835	023472	037574				37574		
4836	023474	000175			63\$:	175		:RESULT.
4837	023476	176737				176737		
4838	023500	071727				71727		
4839	023502	037574				37574		
4840	023504	000000			64\$:	0		:ERROR RES.
4841	023506	000000				0		
4842	023510	000000				0		
4843	023512	000000				0		
4844	023514	000200			65\$:	200		:FPS BEFORE EXECUTION.
4845	023516	000204				204		:FPS AFTER EXECUTION.
4846	023520	000214				214		:ERROR FPS.

```

4847 023522 177777          -1
4848 023524 104211          66$:  ERROR +211          ;BUT OP1B ST 255 TO 311 OR 312 INTO 310
4849 023526 000401          BR 67$
4850 023530 104212          ERROR +212          ;BUT ENBT ST 310 TO 402 INTO 002
4851 023532
4852
4853 023532 012737 023540 001110 67$:  TSTD -0 OP FIUV=0
4854 023540 004737 023702          MOV #270$, $LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.          ;DPM002
4855 023544 000002          270$: JSR PC,1000$
4856 023546 100123          71$: 2          ;FLAG=TSTD.
4857 023550 021012          72$: 100123          ;OPERAND.
4858 023552 034565          21012
4859 023554 043210          34565
4860 023556 100123          73$: 100123          ;RESULT.
4861 023560 021012          21012
4862 023562 034565          34565
4863 023564 043210          43210
4864 023566 000000          74$: 0          ;ERROR RES.
4865 023570 000000          0
4866 023572 000000          0
4867 023574 000000          0
4868 023576 040203          75$: 40203          ;FPS BEFORE EXECUTION.
4869 023600 040214          040214          ;FPS AFTER EXECUTION.
4870 023602 140214          140214          ;ERROR FPS.
4871 023604 177777          -1
4872 023606 104211          76$:  ERROR +211          ;+
4873 023610 000401          BR 77$
4874 023612 104213          ERROR +213          ;BUT FIUV ST 257 TO 355 INTO 255
4875 023614
4876
4877 023614 012737 023622 001110 77$:  TSTD -0 OP FIUV=1
4878 023622 004737 023702          MOV #280$, $LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.          ;DPM002
4879 023626 000002          280$: JSR PC,1000$
4880 023630 100137          81$: 2          ;FLAG=TSTD.
4881 023632 024613          82$: 100137          ;OPERAND.
4882 023634 057024          24613
4883 023636 060137          57024
4884 023640 100137          83$: 100137          ;RESULT.
4885 023642 024613          24613
4886 023644 057024          57024
4887 023646 060137          60137
4888 023650 000000          84$: 0          ;ERROR RES.
4889 023652 000000          0
4890 023654 000000          0
4891 023656 000000          0
4892 023660 044200          85$: 44200          ;FPS BEFORE EXECUTION.
4893 023662 144214          144214          ;FPS AFTER EXECUTION.
4894 023664 044214          044214          ;ERROR FPS.
4895 023666 000014          14
4896 023670 104211          86$:  ERROR +211          ;+
4897 023672 000401          BR 87$
4898 023674 104214          ERROR +214          ;BUT FIUV ST 257 TO 255 INTO 355
4899 023676
4900 023676 000137 024316          87$:  JMP 290$

```

```

4901 :THIS SUBROUTINE, 1000$, IS USED TO SET UP THE OPERANDS, EXECUTE
4902 :THE EITHER A TSTD, AN ABSD OR A NEGD INSTRUCTION AND CHECK THE RESULTS. A CALL
4903 :TO IT IS MADE THUS:
4904 :
4905 JSR PC,1000$
4906 FLAG: .WORD X ;INSTRUCTION TYPE FLAG.
4907 ACARG: .WORD X,X,X,X ;OPERAND
4908 RES: .WORD X,X,X,X ;EXPECTED RESULT
4909 ERRES: .WORD X,X,X,X ;ERROR RESULT
4910 FPSB: .WORD X ;FPS BEFORE EXECUTION
4911 FPSA: .WORD X ;FPS AFTER EXECUTION
4912 FEC: .WORD X ;EXPECTED FEC
4913 ERFPS: .WORD X ;ERROR FPS.
4914 ERR1: ERROR +X ;DATA ERROR.
4915 BR CONT
4916 ERR2: ERROR +X ;FPS ERROR.
4917 CONT: ;RETURN ADDRESS
    
```

```

4918 :THE OPERAND IS SET UP IN MATBF1. THEN
4919 :THE EITHER THE TSTD, NEGD OR ABSD INSTRUCTION IS EXECUTED.
4920 :1000$ USES THE FIRST OPERAND AS A FLAG TO DETERMINE WHICH INSTRUCTION
4921 :IS TO BE EXECUTED: 0 = NEGD, 1 = ABSD, 2 = TSTD.
4922 :THE RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
4923 :COMPARED WITH FPSA. IF THIS IS CORRECT 1000$ RETURNS CONTROL
4924 :TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD 1000$
4925 :COMPARE IT TO ERROR FPS. IF THIS MATCHES THEN 1000$ WILL RETURN
4926 :TO THE ERROR CALL AT ERR2, OTHERWISE 1000$ ITSELF
4927 :REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE
4928 :INSTRUCTION IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
4929 :ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
4930 :THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN 1000$
4931 :WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR1. OTHERWISE THE
4932 :RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND 1000$ WILL
4933 :REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.
    
```

```

4936
4937 023702 012601 1000$: MOV (SP)+,R1 ;GET A POINTER TO THE ARGUMENTS
4938 023704 010102 MOV R1,R2 ;COPY THE OPERAND.
4939 023706 062702 000G02 ADD #2,R2
4940 023712 012703 024304 MOV #1200$,R3
4941 023716 012704 000004 MOV #4,R4
4942 023722 012223 91$: MOV (R2)+,(R3)+
4943 023724 077402 SOB R4,91$
4944 023726 016100 000032 MOV 32(R1),R0 ;LOAD THE FPS.
4945 023732 170100 LDFPS R0
4946 023734 012700 024304 MOV #1200$,R0 ;SET UP THE OPERAND ADDRESS.
4947 023740 011102 MOV (R1),R2 ;GET THE FLAG TO DETERMINE WHICH
4948 023742 006302 ASL R2 ;INSTRUCTION TO EXECUTE.
4949 023744 006302 ASL R2 ;0 = NEGD, 1 = ABSD, 2 = TSTD
4950 023746 012703 023762 MOV #1210$,R3
4951 023752 060203 ADD R2,R3
4952 023754 010337 001236 MOV R3,$TMP2
4953 023760 000113 JMP (R3) ;GO EXECUTE THE INSTRUCTION.
4954 023762 170710 1210$: NEGD (R0)
4955 023764 000403 BR 92$
4956 023766 170610 ABSD (R0)
4957 023770 000401 BR 92$
    
```

4958	023772	170510		TSTD	(R0)	
4959						
4960	023774	170204		92\$: STFPS	R4	:GET THE FPS.
4961	023776	170305		STST	R5	:GET THE FEC.
4962	024000	010102		MOV	R1,R2	
4963	024002	062702	000002	ADD	#2,R2	
4964	024006	010237	001240	MOV	R2,\$TMP3	
4965	024012	062702	000010	ADD	#10,R2	
4966	024016	010237	001244	MOV	R2,\$TMP5	
4967	024022	012737	024304	001242	MOV	#1200,\$TMP4
4968	024030	010437	001250	MOV	R4,\$TMP7	
4969	024034	016137	000034	001252	MOV	34(R1),\$TMP10
4970	024042	010100		MOV	R1,R0	:WAS THE RESULT CORRECT?
4971	024044	062700	000012	ADD	#12,R0	
4972	024050	012702	024304	MOV	#1200\$,R2	
4973	024054	012703	000004	MOV	#4,R3	
4974	024060	022022		93\$: CMP	(R0)+,(R2)+	
4975	024062	001014		BNE	100\$:BRANCH IF INCORRECT.
4976	024064	077303		SOB	R3,93\$	
4977	024066	026104	000034	CMP	34(R1),R4	:WAS THE FPS CORRECT?
4978	024072	001032		BNE	105\$:BRANCH IF INCORRECT.
4979	024074	005761	000034	TST	34(R1)	:IF THE EXPECTED FPS WAS NEGATIVE CHECK THE FEC.
4980	024100	100003		BPL	94\$	
4981	024102	026105	000040	CMP	40(R1),R5	:WAS THE FEC CORRECT.
4982	024106	001037		BNE	110\$:BRANCH IF INCORRECT.
4983	024110	000161	000050	94\$: JMP	50(R1)	:RETURN.
4984						
4985						:THE RESULT WAS INCORRECT BUT WAS THIS FAILURE ANTICIPATED?
4986						:SEE IF THE RESULT WAS ANTICIPATED:
4987	024114			100\$: MOV	(R1),R5	
4988	024114	011105		ASL	R5	
4989	024116	006305		ASL	R5	
4990	024120	006305		ADD	#1220\$,R5	
4991	024122	062705	024234	MOV	R1,R0	
4992	024126	010100		ADD	#22,R0	
4993	024130	062700	000022	MOV	#1200\$,R2	
4994	024134	012702	024304	MOV	#4,R3	
4995	024140	012703	000004	101\$: CMP	(R0)+,(R2)+	
4996	024144	022022		BNE	102\$:BRANCH IF NOT ANTICIPATED.
4997	024146	001003		SOB	R3,101\$	
4998	024150	077303				
4999						:THE ERROR WAS ANTICIPATED SO RETURN.
5000				JMP	42(R1)	
5001	024152	000161	000042			:THE ERROR WAS NOT ANTICIPATED SO REPORT IT HERE.
5002				102\$: JMP	(R5)	:GO TO THE PROPER ERROR CALL.
5003						
5004	024156	000115				:THE FPS WAS INCORRECT.
5005				105\$: CMP	36(R1),R5	:WAS THIS ERROR ANTICIPATED?
5006				BNE	106\$:BRANCH IF NOT ANTICIPATED.
5007	024160	026105	000036			
5008	024164	001002				:THE FPS ERROR WAS ANTICIPATED SO RETURN.
5009				JMP	46(R1)	
5010						:THE FPS FAILURE WAS NOT ANTICIPATED SO REPORT IT HERE.
5011	024166	000161	000046	106\$: MOV	(R1),R2	
5012						
5013						
5014	024172	011102				

5063

```

.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
:
:*****
    
```

```

5064 024320 000004
5064 024322 012737 024330 001110
5065 024330 012700 024406
5066 024334 012710 147517
5067 024340 012737 147517 001240
5068 024346 012737 024362 001236
5069 024354 012737 024446 000004
5070 024362 170110
5071
5072 024364 170205
5073
5074 024366 020027 024406
5075 024372 001007
5076 024374 022705 147517
5077 024400 001013
5078 024402 000436
5079
5080
5081 024404 177777
5082 024406 147517
5083 024410 177777
5084
5085
5086 024412 012737 024406 001240
5087 024420 010037 001242
5088 024424 104225
5089 024426 000424
5090
5091
5092 024430 012737 147517 001240
5093 024436 010537 001242
5094 024442 104226
5095 024444 000415
5096
5097
5098
5099
5100 024446
5101 024446 011602
5102 024450 020227 024364
5103 024454 001405
5104 024456 020227 024366
5105 024462 001402
5106 024464 000137 051774
5107 024470 022626
5108 024472 010237 001236
5109 024476 104227
5110
5111 024500

TST45: SCOPE
MOV #200$,SLPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
200$: MOV #210$,R0 ;SET UP TEST DATA IN BUFFER.
MOV #147517,(R0)
MOV #147517,$TMP3 ;SAVE DATA IN CASE OF ERROR.
MOV #220$, $TMP2
MOV #230$,ERRVECT ;SET UP FOR TRAPS TO 4.
220$: LDFPS (R0) ;TEST INSTRUCTION.

STFPS R5 ;GET FPS

CMP R0,#210$ ;IS R0 CORRECT?
BNE 240$ ;BR IF NOT.
CMP #147517,R5 ;IS FPS CORRECT?
BNE 250$ ;BR IF NOT.
BR 260$

;TEST BUFFER AND DATA:
-1
210$: 147517
-1

;REPORT R0 INCORRECT.
240$: MOV #210$, $TMP3
MOV R0,$TMP4
ERROR +225 ;R0 BAD BUT FSRC FAILED
BR 260$

;REPORT FPS INCORRECT.
250$: MOV #147517,$TMP3 ;REPORT FPS INCORRECT.
MOV R5,$TMP4
ERROR +226
BR 260$

;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.
230$: MOV (SP),R2
CMP R2,#220$+2
BEQ 1$
CMP R2,#220$+4
BEQ 1$
JMP CPSPUR
1$: CMP (SP)+,(SP)+
MOV R2,$TMP2
ERROR +227 ;ODD ADRES

260$:
    
```

024500 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?).

5112

```
.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
:
:*****
```

```
5113 024502 000004
5113 024504 012737 024512 001110
5114 024512 012700 024570
5115 024516 012710 145212
5116 024522 012737 145212 001240
5117 024530 012737 024544 001236
5118 024536 012737 024630 000004
5119
5120 024544 170120
5121
5122 024546 170205
5123
5124 024550 020027 024572
5125 024554 001007
5126 024556 022705 145212
5127 024562 001013
5128 024564 000436
5129
5130
5131
5132 024566 177777
5133 024570 177777
5134 024572 177777
5135
5136
5137
5138 024574 012737 024572 001240
5139 024602 010037 001242
5140 024606 104230
5141 024610 000424
5142
5143
5144 024612 012737 145212 001240
5145 024620 010537 001242
5146 024624 104231
5147 024626 000415
5148
5149
5150
5151
5152 024630
5153 024630 011602
5154 024632 020227 024546
5155 024636 001405
5156 024640 020227 024550
5157 024644 001402
5158 024646 000137 051774
5159 024652 022626
5160 024654 010237 001236

TST46: SCOPE
200$: MOV #200$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
MOV #210$, R0 ;SET UP TEST DATA IN BUFFER.
MOV #145212, (R0)
MOV #145212, $TMP3 ;SAVE DATA IN CASE OF ERROR.
MOV #220$, $TMP2
MOV #230$, $ERRVECT ;SET UP FOR TRAPS TO 4.
220$: LDFPS (R0)+ ;TEST INSTRUCTION.
STFPS R5 ;GET FPS
CMP R0, #210$+2 ;IS R0 CORRECT?
BNE 240$ ;BR IF NOT.
CMP #145212, R5 ;IS THE FPS CORRECT?
BNE 250$ ;BR IF NOT.
BR 260$

;TEST BUFFER AND DATA:
210$: .WORD -1
-1

;REPORT R0 INCORRECT.
240$: MOV #210$+2, $TMP3
MOV R0, $TMP4
ERROR +230 ;R0 BAD BUT FSRC FAILED
BR 260$

;REPORT FPS INCORRECT.
250$: MOV #145212, $TMP3 ;REPORT FPS INCORRECT.
MOV R5, $TMP4
ERROR +231
BR 260$

;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.
230$: MOV (SP), R2
CMP R2, #220$+2
BEQ 1$
CMP R2, #220$+4
BEQ 1$
JMP CPSPUR
1$: CMP (SP)+, (SP)+
MOV R2, $TMP2
```

5161 024660 104232
5162
5163 024662 104412
024662 104412

ERROR +232
260\$: 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?).

5164

```

.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
*****
    
```

```

5165 024664 000004 024674 001110 TST47: SCOPE
5166 024666 012737 024674 001110 MOV #200$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
5167 024700 012760 105252 177776 200$: MOV #210$+2, R0 ;SET UP THE TEST DATA BUFFER.
5168 024706 012737 105252 001240 MOV #105252, -2(R0)
5169 024714 012737 024730 001236 MOV #105252, $TMP3 ;SAVE DATA IN CASE OF ERROR.
5170 024722 012737 025030 000004 MOV #220$, $TMP2
5171 024730 170140 220$: MOV #230$, ERRVEC
5172 024732 170205 LDFPS -(R0)
5173 024734 020027 024762 STFPS R5
5174 024740 001015 CMP R0, #210$
5175 024742 022705 105252 BNE 240$
5176 024746 001021 CMP #105252, R5
5177 024750 000444 BNE 250$
5178 BR 260$
5179 024752 177777 177777 177777 210$: -1, -1, -1, -1
5180 024762 177777 -1
5181 024764 177777 177777 177777 -1, -1, -1, -1
5182
5183 024774 012737 024762 001240 240$: MOV #210$, $TMP3
5184 025002 010037 001242 MOV R0, $TMP4
5185 025006 104233 ERROR +233 ;R0 BAD BUT FSRC FAILED
5186 025010 000424 BR 260$
5187 025012 012737 105252 001240 250$: MOV #105252, $TMP3 ;REPORT FPS INCORRECT.
5188 025020 010537 001242 MOV R5, $TMP4
5189 025024 104234 ERROR +234
5190 025026 000415 BR 260$
5191 025030 011602 230$: MOV (SP), R2
5192 025032 020227 024732 CMP R2, #220$+2
5193 025036 001405 BEQ 1$
5194 025040 020227 024734 CMP R2, #220$+4
5195 025044 001402 BEQ 1$
5196 025046 000137 051774 JMP CPSPUR
5197 025052 022626 1$: CMP (SP)+, (SP)+
5198 025054 010237 001236 MOV R2, $TMP2
5199 025060 104235 ERROR +235 ;DDD ADRES
5200 025062 104412 260$: 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?).
    
```

5201

```
.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  
:*****
```

```
025064 000004  
5202 025066 012737 025074 001110  
5203 025074 012700 025176  
5204 025100 012710 025166  
5205 025104 012737 103456 025166  
5206 025112 012737 103456 001240  
5207 025120 012737 025134 001236  
5208 025126 012737 025244 000004  
5209 025134 170130  
5210 025136 170205  
5211 025140 020027 025200  
5212 025144 001021  
5213 025146 022705 103456  
5214 025152 001025  
5215 025154 000450  
5216  
5217  
5218  
5219 025156 177777 177777 177777  
5220 025166 177777  
5221 025170 177777 177777 177777  
5222 025176 025166 177777 177777  
5223  
5224  
5225  
5226 025210 012737 025200 001240  
5227 025216 010037 001242  
5228 025222 104236  
5229 025224 000424  
5230  
5231  
5232 025226 012737 103456 001240  
5233 025234 010537 001242  
5234 025240 104237  
5235 025242 C00415  
5236  
5237  
5238  
5239 025244 011602  
5240 025246 020227 025136  
5241 025252 001405  
5242 025254 020227 025140  
5243 025260 001402  
5244 025262 000137 051774  
5245 025266 022626  
5246 025270 010237 001236  
5247 025274 104240  
5248 025276 104412
```

```
TST50: SCOPE  
MOV #200$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002  
200$: MOV #210$, R0  
MOV #220$, (R0)  
MOV #103456, 220$  
MOV #103456, $TMP3  
MOV #230$, $TMP2  
MOV #240$, ERRVECT ;SET UP FOR TRAPS TO 4.  
230$: LDFPS @(R0)+ ;TEST INSTRUCTION.  
STFPS R5 ;GET THE FPS.  
CMP R0, #210$+2 ;IS R0 CORRECT?  
BNE 250$ ;BR IF NOT.  
CMP #103456, R5 ;IS THE FPS CORRECT?  
BNE 260$ ;BR IF NOT.  
BR 270$  
:TEST BUFFER AND DATA:  
220$: -1, -1, -1, -1  
-1  
210$: -1, -1, -1  
220$, -1, -1, -1,  
:REPORT R0 INCORRECT.  
250$: MOV #210$+2, $TMP3  
MOV R0, $TMP4  
ERROR +236 ;R0 BAD BUT FSRC FAILED  
BR 270$  
:REPORT FPS INCORRECT.  
260$: MOV #103456, $TMP3 ;REPORT FPS INCORRECT.  
MOV R5, $TMP4  
ERROR +237  
BR 270$  
-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.  
240$: MOV (SP), R2  
CMP R2, #230$+2  
BEQ 1$  
CMP R2, #230$+4  
BEQ 1$  
JMP C PSPUR  
1$: CMP (SP)+, (SP)+  
MOV R2, $TMP2  
ERROR +240 ;DDD ADRES  
270$: 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?).

5249

```
.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
*****
```

```
TST51: SCOPE
5250 025300 000004          200$: MOV #200$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
5251 025302 012737 025310 001110 MOV #210$+2, R0 ;SET UP THE TEST DATA BUFFER.
5252 025310 012700 025410 MOV #220$, -2(R0)
5253 025314 012760 025376 177776 MOV #45412, 220$
5254 025322 012737 045412 025376 MOV #45412, $TMP3 ;SAVE DATA IN CASE OF ERROR.
5255 025330 012737 045412 001240 MOV #230$, $TMP2
5256 025336 012737 025352 001236 MOV #240$, ERRVECT ;SET UP FOR TRAPS TO 4.
5257 025344 012737 025452 000004 230$: LDFPS @-(R0) ;TEST INSTRUCTION.
5258 025352 170150 STFPS R5 ;GET THE FPS.
5259 025354 170205 CMP R0, #210$ ;IS R0 CORRECT?
5260 025356 020027 025406 BNE 250$ ;BR IF NOT.
5261 025362 001015 CMP #45412, R5 ;IS THE FPS CORRECT?
5262 025364 022705 045412 BNE 260$ ;BR IF NOT.
5263 025370 001021 BR 270$
5264 025372 000444
5265
5266 ;TEST BUFFER AND DATA:
5267 025374 177777 -1
5268 025376 177777 220$: -1
5269 025400 177777 177777 177777 -1, -1, -1
5270 025406 025376 177777 177777 210$: 220$, -1, -1, -1
5271
5272
5273 ;REPORT RO INCORRECT.
5274 025416 012737 025406 001240 250$: MOV #210$, $TMP3
5275 025424 010037 001242 MOV R0, $TMP4
5276 025430 104241 ERROR +241 ;RO BAD BUT FSRC FAILED
5277 025432 000424 BR 270$
5278
5279 ;REPORT FPS INCORRECT.
5280 025434 012737 045412 001240 260$: MOV #45412, $TMP3 ;REPORT FPS INCORRECT.
5281 025442 010537 001242 MOV R5, $TMP4
5282 025446 104242 ERROR +242
5283 025450 000415 BR 270$
5284
5285 ;TRAP HERE THROUGH VECTOR FOUR. SEE IF THE TRAP WAS DURING
5286 ;EXECUTION OF THE FPS INSTRUCTION BEING TESTED. IF SO REPORT
5287 ;FAILURE. OTHERWISE GO TO THE SPURIOUS TRAP TO 4 HANDLING.
5288 025452 011602 240$: MOV (SP), R2
5289 025454 020227 025354 CMP R2, #230$+2
5290 025460 001405 BEQ 1$
5291 025462 020227 025356 CMP R2, #230$+4
5292 025466 001402 BEQ 1$
5293 025470 000137 051774 JMP CPSPUR
5294 025474 022626 1$: CMP (SP)+, (SP)+
5295 025476 010237 001236 MOV R2, $TMP2
5296 025502 104243 ERROR +243 ;ODD ADRES
5297 025504
5298 025504 104412 270$: 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?).

5297

```

.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
*
*****
    
```

```

025506 000004
5298
5299 025510 012767 025515 153372
5300 025516 012700 020405
5301 025522 012767 046543 000056
5302 025530 012767 046543 153502
5303 025536 012767 025554 153472
5304 025544 005001
5305 025546 012767 025674 152230
5306 025554 170160 005201
5307 025560 170204
5308 025562 005701
5309 025564 001033
5310 025566 020027 020405
5311 025572 001012
5312 025574 022704 046543
5313 025600 001016
5314 025602 000451
5315
5316
5317
5318 025604 177777
5319 025606 177777 177777 177777
5320 025616 177777
5321
5322
5323 025620 012767 020405 153412
5324 025626 010067 153410
5325 025632 104244
5326 025634 000434
5327
5328
5329 025636 012767 046543 153374
5330 025644 010467 153372
5331 025650 104245
5332 025652 000425
5333
5334
5335 025654 012767 025560 153356
5336 025662 012767 025556 153352
5337 025670 104246
5338 025672 000415
5339
5340
5341
5342 025674 011602
5343 025676 020227 025556
5344 025702 001405
5345 025704 020227 025560

    SCOPE
    DSABLE AMA ;DISABLE MODE 6 TO MODE 3 CONVERSIONS
    MOV #200$,SLPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
    200$: MOV #210$-5201,R0 ;SET UP THE TEST DATA BUFFER.
    MOV #46543,210$
    MOV #46543,$TMP3 ;SAVE DATA IN CASE OF ERROR.
    MOV #220$,$TMP2
    CLR R1
    MOV #230$,ERRVECT ;SET UP FOR TRAPS TO 4.
    220$: LDFPS 5201(R0) ;TEST INSTRUCTION.
    STFPS R4 ;GET THE FPS.
    TST R1 ;WAS PC CORRECT AFTER EXECUTION?
    BNE 240$ ;BR IF NOT.
    CMP R0,#210$-5201 ;IS R0 CORRECT?
    BNE 250$ ;BR IF NOT.
    CMP #46543,R4 ;IS THE FPS CORRECT?
    BNE 260$ ;BR IF NOT.
    BR 270$

    ;TEST BUFFER AND DATA:
    -1
    210$: -1,-1,-1,-1
    -1

    ;REPORT R0 INCORRECT.
    250$: MOV #210$-5201,$TMP3
    MOV R0,$TMP4
    ERROR +244 ;R0 BAD BUT FSRC FAILED
    BR 270$

    ;REPORT FPS INCORRECT.
    260$: MOV #46543,$TMP3 ;REPORT FPS INCORRECT.
    MOV R4,$TMP4
    ERROR +245
    BR 270$

    ;REPORT PC INCORRECT AFTER INSTRUCTION.
    240$: MOV #220$+4,$TMP3
    MOV #220$+2,$TMP4
    ERROR +246 ;PC X
    BR 270$

    ;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 ~ HANDLING.
    230$: MOV (SP),R2
    CMP R2,#220$+2
    BEQ 1$
    CMP R2,#220$+4
    
```

5346 025710 001402
5347 025712 000167 024056
5348 025716 022626
5349 025720 010267 153312
5350 025724 104247
5351 025726 104412

1\$: BEQ 1\$
JMP CPSPUR
CMP (SP)+,(SP)+
MOV R2,\$TMP2
ERROR +247

;ODD ADRES

270\$:

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?).

5352

.ENABL AMA

;REENABLE MODE 6 TO MODE 7 COMVERSIONS

5353

```
.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
*****
```

```
TST53: SCOPE
5354 025730 000004 025740 001110 MOV #200$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
5355 025732 012737 020645 200$: MOV #210$-5201, R0 ;SET UP THE TEST DATA BUFFER.
5356 025744 012760 026036 005201 MOV #220$, 5201(R0)
5357 025752 012737 004547 026036 MOV #4547, 220$
5358 025760 012737 004547 001240 MOV #4547, $TMP3 ;SAVE DATA IN CASE OF ERROR.
5359 025766 012737 026004 001236 MOV #230$, $TMP2
5360 025774 005001 CLR R1
5361 025776 012737 026132 000004 MOV #240$, ERRVCT ;SET UP FOR TRAPS TO 4.
5362 026004 170170 005201 230$: LDFPS @5201(R0) ;TEST INSTRUCTION.
5363 026010 170204 STFPS R4 ;GET THE FPS.
5364 026012 005701 TST R1 ;WAS PC CORRECT AFTER EXECUTION?
5365 026014 001036 BNE 250$ ;BR IF NOT.
5366 026016 020027 020645 CMP R0, #210$-5201 ;IS R0 CORRECT?
5367 026022 001015 BNE 260$ ;BR IF NOT.
5368 026024 022704 004547 CMP #4547, R4 ;IS THE FPS CORRECT?
5369 026030 001021 BNE 270$ ;BR IF NOT.
5370 026032 000454 BR 280$
5371
5372
5373 ;TEST BUFFER AND DATA:
5374 026034 177777 -1
5375 026036 177777 177777 177777 220$: .WORD -1, -1, -1, -1
5376 026046 177777 177777 177777 210$: .WORD -1, -1, -1, -1
5377
5378 ;REPORT R0 INCORRECT.
5379 026056 012737 020645 001240 260$: MOV #210$-5201, $TMP3
5380 026064 010037 001242 MOV R0, $TMP4
5381 026070 104250 ERROR +250 ;R0 BAD BUT FSRC FAILED
5382 026072 000434 BR 280$
5383
5384 ;REPORT FPS INCORRECT.
5385 026074 012737 004547 001240 270$: MOV #4547, $TMP3 ;REPORT FPS INCORRECT.
5386 026102 010437 001242 MOV R4, $TMP4
5387 026106 104251 ERROR +251
5388 026110 000425 BR 280$
5389
5390 ;REPORT PC INCORRECT AFTER INSTRUCTION.
5391 026112 012737 026010 001240 250$: MOV #230$+4, $TMP3
5392 026120 012737 026006 001242 MOV #230$+2, $TMP4
5393 026126 104252 ERROR +252 ;PC X
5394 026130 000415 BR 280$
5395
5396 ;TRAP HERE THROUGH VECTOR FOUR. SEE IF THE TRAP WAS DURING
5397 ;EXECUTION OF THE FPS INSTRUCTION BEING TESTED. IF SO REPORT
5398 ;FAILURE. OTHERWISE GO TO THE SPURIOUS TRAP TO 4 HANDLING.
5398 026132 011602 240$: MOV (SP), R2
5399 026134 020227 026006 CMP R2, #230$+2
5400 026140 001405 BEQ 1$
5401 026142 020227 026010 CMP R2, #230$+4
```

5402 026146 001402
5403 026150 000137 051774
5404 026154 022626
5405 026156 010237 001236
5406 026162 104253
5407 026164
026164 104412

BEQ 1\$
JMP CPSPUR
1\$: CMP (SP)+,(SP)+
MOV R2,\$TMP2
ERROR +253

;DDD ADDRESS

280\$: 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?).

5408

.ENABL AMA

5415

```
.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
:*
:*****
```

```
TST54: SCOPE
5416 026166 000004          026176 001110      MOV #200$,SLPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
5417 026176 012737          026222 001236      200$: MOV #210$,STMP2 ;SAVE DATA IN CASE OF ERROR.
5418 026204 012737          026274 000004      MOV #220$,ERRVECT ;SET UP FOR TRAPS TO 4.
5419 026212 012700          000300      MOV #300,R0
5420 026216 170100          LDFPS R0
5421 026220 005001          CLR R1
5422
5423 026222 177027          210$: LDCLD (R7)+,ACO ;TEST INSTRUCTION.
5424 026224 005201          5201
5425 026226 005201          5201
5426 026230 005201          5201
5427 026232 005201          5201
5428
5429 026234 020127          000003      CMP R1,#3 ;WAS PC CORRECT AFTER EXECUTION?
5430 026240 001421          BEQ 230$ ;BR IF YES.
5431
5432
5433 ;REPORT PC INCORRECT AFTER INSTRUCTION.
5434 026242 012704          026242 000003      MOV #210$+4,R4
5435 026246 162701          SUB #3,R1
5436 026252 006301          ASL R1
5437 026254 160104          SUB R1,R4
5438 026256 010437          001242      MOV R4,STMP4
5439 026262 012737          026226 001240      MOV #210$+4,STMP3
5440 026270 104254          ERROR +254 ;BAD CONSTANT
5441 026272 000404          BR 230$
5442 ;TRAP HERE THROUGH VECTOR FOUR. SEE IF THE TRAP WAS DURING
5443 ;EXECUTION OF THE FPS INSTRUCTION BEING TESTED. IF SO REPORT
5444 ;FAILURE. OTHERWISE GO TO THE SPURIOUS TRAP TO 4 HANDLING.
5445 026274 011637          001236      220$: MOV (SP),STMP2
5446 026300 022626          CMP (SP)+,(SP)+
5447 026302 104255          ERROR +255 ;BAD CONSTANT ODD ADD
5448
5449 026304          230$: RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
026304 104412          ;SEE IF THE USER HAS EXPRESSED
;THE DESIRE TO CHANGE THE SOFTWARE
;VIRTUAL CONSOLE SWITCH REGISTER (HAS
;THE USER TYPED CONTROL G?).
```

5456

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.
 *

5457	026306	000004				TST55: SCOPE		
5458	026310	012737	026344	001110		MOV #200\$, \$LPERR	:SET UP THE LOOP ON ERROR ADDRESS.	:DPM002
5459	026316	013737	026344	001236	200\$:	MOV 210\$, \$TMP2	:SAVE DATA IN CASE OF ERROR.	
5460	026324	012700	000300			MOV #300, R0		
5461	026330	170100				LDFPS R0		
5462	026332	012700	026426		210\$:	MOV #220\$, R0	:SET UP THE TEST DATA BUFFER.	
5463	026336	177020				LDCLD (R0)+, ACO	:TEST INSTRUCTION.	
5464	026340	170204				STFPS R4	:GET THE FPS.	
5465	026342	012701	026436			MOV #230\$, R1	:GET THE RESULT.	
5466	026346	012702	000200			MOV #200, R2		
5467	026352	170102				LDFPS R2		
5468	026354	174011				STD ACO, (R1)		
5469	026356	020027	026432			CMR R0, #220\$+4	:IS R0 CORRECT?	
5470	026362	001407				BEQ 240\$		
5471						:REPORT R0 INCORRECT.		
5472	026364	010037	001242			MOV R0, \$TMP4		
5473	026370	012737	026432	001240		MOV #220\$+4, \$TMP3		
5474	026376	104256				ERROR +256	:BAD CONST	
5475	026400	000422				BR 250\$		
5476								
5477	026402	022704	000300		240\$:	CMR #300, R4	:IS THE FPS CORRECT?	
5478	026406	001417				BEQ 250\$		
5479								
5480						:REPORT FPS INCORRECT.		
5481	026410	010437	001242			MOV R4, \$TMP4		
5482	026414	012737	000300	001240		MOV #300, \$TMP3		
5483	026422	104257				ERROR +257	:FPS X	
5484	026424	000410				BR 250\$		
5485								
5486						:TEST BUFFER AND DATA:		
5487	026426	001234	067076	054321	220\$:	.WORD 01234, 67076, 54321, 012345		
5488	026436	177777	177777	177777	230\$:	-1, -1, -1, -1		
5489								
5490	026446				250\$:	RSETUP	:GO INITIALIZE THE FPS AND STACK; AND	
	026446	104412					:SEE IF THE USER HAS EXPRESSED	
							:THE DESIRE TO CHANGE THE SOFTWARE	
							:VIRTUAL CONSOLE SWITCH REGISTER (HAS	
							:THE USER TYPED CONTROL G?).	

5497

```

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

5498	026450	000004							
5499	026452	012737	026460	001110	:ZERO	SCOPE OPERAND FL=0			
5500	026460	004737	027724		200\$:	MOV #200\$, \$LPERR JSR PC, 1000\$:SET UP THE LOOP ON ERROR ADDRESS. :GO EXECUTE INSTRUCTION.		:DPM002
5501					1\$:	.WORD 0,0	:FSRC OPERAND.		
5502	026464	000000	000000		2\$:	.WORD 0,0	:EXPECTED RESULT.		
5503	026470	000000	000000		3\$:	.WORD -1,-1	:ANTICIPATED ERRONEOUS RESULT.		
5504	026474	177777	177777		4\$:	0	:FPS BEFORE EXECUTION.		
5505	026500	000000				4	:FPS AFTER EXECUTION.		
5506	026502	000004				-1	:ANTICIPATED ERRONEOUS FPS.		
5507	026504	177777			5\$:	ERROR +260	:REPORT RESULT INCORRECT.		
5508	026506	104260				BR 6\$			
5509	026510	000401				ERROR +261	:REPORT FPS INCORRECT.		
5510	026512	104261			6\$:				
5511	026514				:ZERO	OPERAND FL=0			
5512									
5513									
5514	026514	012737	026522	001110		MOV #210\$, \$LPERR	:SET UP THE LOOP ON ERROR ADDRESS.		:DPM002
5515	026522	004737	027724		210\$:	JSR PC, 1000\$:GO EXECUTE THE INSTRUCTION.		
5516									
5517	026526	000000	177777		11\$:	.WORD 0,-1	:FSRC OPERAND.		
5518	026532	000000	000000		12\$:	.WORD 0,0	:EXPECTED RESULT.		
5519	026536	004177	177400		13\$:	4177,177400	:ANTICIPATED ERRONEOUS RESULT.		
5520	026542	000000			14\$:	0	:FPS BEFORE EXECUTION.		
5521	026544	000004				4	:FPS AFTER EXECUTION.		
5522	026546	177777				-1	:ANTICIPATED ERRONEOUS FPS.		
5523	026550	104262			15\$:	ERROR +262	:(BUT FL) ST		
5524	026552	000401				BR 16\$:277 TO 300		
5525	026554	104261				ERROR +261	:INTO 301		
5526	026556				16\$:				
5527					:ZERO	OPERAND FL=1			
5528									
5529	026556	012737	026564	001110		MOV #220\$, \$LPERR	:SET UP THE LOOP ON ERROR ADDRESS.		:DPM002
5530	026564	004737	027724		220\$:	JSR PC, 1000\$:GO EXECUTE THE INSTRUCTION.		
5531	026570	000000	000000		21\$:	.WORD 0,0	:FSRC OPERAND.		
5532	026574	000000	000000		22\$:	.WORD 0,0	:EXPECTED RESULT.		
5533	026600	177777	177777		23\$:	.WORD -1,-1	:ANTICIPATED ERRONEOUS RESULT.		
5534	026604	000100			24\$:	100	:FPS BEFORE EXECUTION.		
5535	026606	000104				104	:FPS AFTER EXECUTION.		
5536	026610	000004				4	:ANTICIPATED ERRONEOUS FPS.		
5537	026612	104260			25\$:	ERROR +260	:REPORT RESULT INCORRECT.		
5538	026614	000401				BR 26\$			
5539	026616	104263				ERROR +263	:FL WAS CLR'ED		
5540	026620				26\$:				
5541					:OPERAND	POSITIVE	FL=0		
5542	026620	012737	026626	001110		MOV #230\$, \$LPERR	:SET UP THE LOOP ON ERROR ADDRESS.		:DPM002
5543	026626	004737	027724		230\$:	JSR PC, 1000\$:GO EXECUTE THE INSTRUCTION.		
5544	026632	040000	000000		31\$:	.WORD 40000,0	:FSRC OPERAND.		
5545	026636	043600	000000		32\$:	.WORD 43600,0	:EXPECTED RESULT.		

5546	026642	047600	000000		33\$:	.WORD	47600,0		:ANTICIPATED ERRONEOUS RESULT.	
5547	026646	000017			34\$:	17			:FPS BEFORE EXECUTION.	
5548	026650	000000				0			:FPS AFTER EXECUTION.	
5549	026652	177777				-1			:ANTICIPATED ERRONEOUS FPS.	
5550	026654	104264			35\$:	ERROR	+264	:ST 107	BAD	
5551	026656	000401				BR	36\$:CONSTANT 231 !NSD	
5552	026660	104261				ERROR	+261		:215	
5553	026662				36\$:					
5554					:OPERAND=1,		FL=0			
5555	026662	012737	026670	001110		MOV	#240\$, \$LPERR		:SET UP THE LOOP ON ERROR ADDRESS.	:DPM002
5556	026670	004737	027724		240\$:	JSR	PC,1000\$:GO EXECUTE THE INSTRUCTION.	
5557	026674	000001	000000		41\$:	.WORD	1,0		:FSRC OPERAND.	
5558	026700	040200	000000		42\$:	.WORD	40200,0		:EXPECTED RESULT.	
5559	026704	044200	000000		43\$:	.WORD	44200,0		:ANTICIPATED ERRONEOUS RESULT.	
5560	026710	000017			44\$:	17			:FPS BEFORE EXECUTION.	
5561	026712	000000				0			:FPS AFTER EXECUTION.	
5562	026714	177777				-1			:ANTICIPATED ERRONEOUS FPS.	
5563	026716	104264			45\$:	ERROR	+264		:REPORT RESULT INCORRECT.	
5564	026720	000401				BR	46\$			
5565	026722	104261				ERROR	+261		:REPORT FPS INCORRECT.	
5566	026724				46\$:					
5567										
5568										
5569					:OPERAND=		PATTERN FL=0			
5570	026724	012737	026732	001110		MOV	#250\$, \$LPERR		:SET UP THE LOOP ON ERROR ADDRESS.	:DPM002
5571	026732	004737	027724		250\$:	JSR	PC,1000\$:GO EXECUTE THE INSTRUCTION.	
5572	026736	000252	000000		51\$:	.WORD	252,0		:FSRC OPERAND.	
5573	026742	042052	000000		52\$:	.WORD	42052,0		:EXPECTED RESULT.	
5574	026746	046052	000000		53\$:	.WORD	46052,0		:ANTICIPATED ERRONEOUS RESULT.	
5575	026752	000000			54\$:	0			:FPS BEFORE EXECUTION.	
5576	026754	000000				0			:FPS AFTER EXECUTION.	
5577	026756	177777				-1			:ANTICIPATED ERRONEOUS FPS.	
5578	026760	104264			55\$:	ERROR	+264		:REPORT RESULT INCORRECT.	
5579	026762	000401				BR	56\$			
5580	026764	104261				ERROR	+261		:REPORT FPS INCORRECT.	
5581	026766				56\$:					
5582										
5583					:OPERAND=-40000		FL=0			
5584	026766	012737	026774	001110		MOV	#260\$, \$LPERR		:SET UP THE LOOP ON ERROR ADDRESS.	:DPM002
5585	026774	004737	027724		260\$:	JSR	PC,1000\$:GO EXECUTE THE INSTRUCTION.	
5586	027000	140000	000000		61\$:	.WORD	-40000,0		:FSRC OPERAND.	
5587	027004	143600	000000		62\$:	.WORD	143600,0		:EXPECTED RESULT.	
5588	027010	043600	000000		63\$:	.WORD	43600,0		:ANTICIPATED ERRONEOUS RESULT.	
5589	027014	000007			64\$:	7			:FPS BEFORE EXECUTION.	
5590	027016	000010				10			:FPS AFTER EXECUTION.	
5591	027020	177777				-1			:ANTICIPATED ERRONEOUS FPS.	
5592	027022	104265			65\$:	ERROR	+265		:(SET SIGN) ST 146	
5593	027024	000401				BR	66\$			
5594	027026	104261				ERROR	+261		:REPORT FPS INCORRECT.	
5595	027030				66\$:					
5596										
5597					:OPERAND=-1		FL=0			
5598	027030	012737	027036	001110		MOV	#270\$, \$LPERR		:SET UP THE LOOP ON ERROR ADDRESS.	:DPM002
5599	027036	004737	027724		270\$:	JSR	PC,1000\$:GO EXECUTE THE INSTRUCTION.	
5600	027042	177777	000000		71\$:	.WORD	-1,0		:FSRC OPERAND.	
5601	027046	140200	000000		72\$:	.WORD	140200,0		:EXPECTED RESULT.	
5602	027052	144000	000400		73\$:	.WORD	144000,400		:ANTICIPATED ERRONEOUS RESULT.	

5603	027056	000000			74\$:	0				:FPS BEFORE EXECUTION.	
5604	027060	000010				10				:FPS AFTER EXECUTION.	
5605	027062	177777				-1				:ANTICIPATED ERRONEOUS FPS.	
5606	027064	104266			75\$:	ERROR	+266			:ST 372 TO 152 INTO	
5607	027066	000612				BR	6\$:112 (BUF XMBT)	
5608	027070	104261				ERROR	+261			:REPORT FPS INCORRECT.	
5609	027072				76\$:						
5610											
5611											
5612	027072	012737	027100	001110		:OPERAND=	PATTERN	FL=0			
5613	027100	004737	027724		280\$:	MOV	#280\$, \$LPERR			:SET UP THE LOOP ON ERROR ADDRESS.	:DPM002
5614	027104	125252	000000		81\$:	JSR	PC, 1000\$:GO EXECUTE THE INSTRUCTION.	
5615	027110	143652	126000		82\$:	.WORD	125252, 0			:FSRC OPERAND.	
5616	027114	043652	126000		83\$:	.WORD	143652, 126000			:EXPECTED RESULT.	
5617	027120	000007			84\$:	.WORD	43652, 126000			:ANTICIPATED ERRONEOUS RESULT.	
5618	027122	000010				7				:FPS BEFORE EXECUTION.	
5619	027124	177777				10				:FPS AFTER EXECUTION.	
5620	027126	104265			85\$:	-1				:ANTICIPATED ERRONEOUS FPS.	
5621	027130	000401				ERROR	+265			:REPORT RESULT INCORRECT.	
5622	027132	104261				BR	86\$				
5623	027134				86\$:	ERROR	+261			:REPORT FPS INCORRECT.	
5624											
5625											
5626	027134	012737	027142	001110		:OPERAND	POS	FL-1			
5627	027142	004737	027724		290\$:	MOV	#290\$, \$LPERR			:SET UP THE LOOP ON ERROR ADDRESS.	:DPM002
5628	027146	040000	000000		91\$:	JSR	PC, 1000\$:GO EXECUTE THE INSTRUCTION.	
5629	027152	047600	000000		92\$:	.WORD	40000, 0			:FSRC OPERAND.	
5630	027156	043600	000000		93\$:	.WORD	47600, 0			:EXPECTED RESULT.	
5631	027162	000117			94\$:	.WORD	43600, 0			:ANTICIPATED ERRONEOUS RESULT.	
5632	027164	000100				117				:FPS BEFORE EXECUTION.	
5633	027166	177777				100				:FPS AFTER EXECUTION.	
5634	027170	104267			95\$:	-1				:ANTICIPATED ERRONEOUS FPS.	
5635	027172	000401				ERROR	+267			:CONSTANT	
5636	027174	104261				BR	96\$:BAD 237 INST 217	
5637	027176				96\$:	ERROR	+261			:REPORT FPS INCORRECT.	
5638											
5639											
5640	027176	012737	027204	001110		:OPERAND=	1	FL=1			
5641	027204	004737	027724		300\$:	MOV	#300\$, \$LPERR			:SET UP THE LOOP ON ERROR ADDRESS.	:DPM002
5642	027210	000000	000001		101\$:	JSR	PC, 1000\$:GO EXECUTE THE INSTRUCTION.	
5643	027214	040200	000000		102\$:	.WORD	0, 1			:FSRC OPERAND.	
5644	027220	034200	000000		103\$:	.WORD	40200, 0			:EXPECTED RESULT.	
5645	027224	000100			104\$:	.WORD	34200, 0			:ANTICIPATED ERRONEOUS RESULT.	
5646	027226	000100				100				:FPS BEFORE EXECUTION.	
5647	027230	177777				100				:FPS AFTER EXECUTION.	
5648	027232	104267			105\$:	-1				:ANTICIPATED ERRONEOUS FPS.	
5649	027234	000401				ERROR	+267			:REPORT RESULT INCORRECT.	
5650	027236	104261				BR	106\$				
5651	027240				106\$:	ERROR	+261			:REPORT FPS INCORRECT.	
5652											
5653											
5654	027240	012737	027246	001110		:OPERAND=	PATTERN	FL=1			
5655	027246	004737	027724		310\$:	MOV	#310\$, \$LPERR			:SET UP THE LOOP ON ERROR ADDRESS.	:DPM002
5656	027252	000000	000252		111\$:	JSR	PC, 1000\$:GO EXECUTE THE INSTRUCTION.	
5657	027256	042052	000000		112\$:	.WORD	0, 252			:FSRC OPERAND.	
5658	027262	036052	000000		113\$:	.WORD	42052, 0			:EXPECTED RESULT.	
5659	027266	000111			114\$:	.WORD	36052, 0			:ANTICIPATED ERRONEOUS RESULT.	
						111				:FPS BEFORE EXECUTION.	

```

5660 027270 000100          100          ;FPS AFTER EXECUTION.
5661 027272 177777          -1          ;ANTICIPATED ERRONEOUS FPS.
5662 027274 104267      115$:  ERROR    +267          ;REPORT RESULT INCORRECT.
5663 027276 000401          BR           116$
5664 027300 104261          ERROR    +267          ;REPORT FPS INCORRECT.
5665 027302      116$:
5666
5667      ;OPERAND=-40000,0    FL=1
5668 027302 012737 027310 001110  MOV     #320$, $LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.      ;DPM002
5669 027310 004737 027724      320$:  JSR     PC,1000$          ;GO EXECUTE THE INSTRUCTION.
5670 027314 140000 000000      121$:  .WORD  -40000,0          ;FSRC OPERAND.
5671 027320 147600 000000      122$:  .WORD  147600,0          ;EXPECTED RESULT.
5672 027324 047600 000000      123$:  .WORD  47600,0          ;ANTICIPATED ERRONEOUS RESULT
5673 027330 000107          124$:  107          ;FPS BEFORE EXECUTION.
5674 027332 000110          110          ;FPS AFTER EXECUTION.
5675 027334 177777          -1          ;ANTICIPATED ERRONEOUS FPS.
5676 027336 104265      125$:  ERROR    +265      ;SET SIGN
5677 027340 000401          BR           126$
5678 027342 104261          ERROR    +261          ;REPORT FPS INCORRECT.
5679 027344      126$:
5680
5681      ;OPERAND=-1,-1    FL=1
5682 027344 012737 027352 001110  MOV     #330$, $LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.      ;DPM002
5683 027352 004737 027724      330$:  JSR     PC,1000$          ;GO EXECUTE THE INSTRUCTION.
5684 027356 177777          131$:  .WORD  -1,-1          ;FSRC OPERAND.
5685 027362 140200 000000      132$:  .WORD  140200,0          ;EXPECTED RESULT.
5686 027366 150000 000000      133$:  .WORD  150000,0          ;ANTICIPATED ERRONEOUS RESULT.
5687 027372 000100          134$:  100          ;FPS BEFORE EXECUTION.
5688 027374 000110          110          ;FPS AFTER EXECUTION.
5689 027376 177777          -1          ;ANTICIPATED ERRONEOUS FPS.
5690 027400 104266      135$:  ERROR    +266          ;(BUT XMBT)
5691 027402 000401          BR           136$
5692 027404 104261          ERROR    +261          ;REPORT FPS INCORRECT.
5693 027406      136$:
5694
5695      ;OPERAND=-PATTERN    FL=1,    ROUND MODE
5696 027406 012737 027414 001110  MOV     #340$, $LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.      ;DPM002
5697 027414 004737 027724      340$:  JSR     PC,1000$          ;GO EXECUTE THE INSTRUCTION.
5698 027420 125252 125252          141$:  .WORD  125252,125252          ;FSRC OPERAND.
5699 027424 147652 125253          142$:  .WORD  147652,125253          ;EXPECTED RESULT.
5700 027430 047652 125253          143$:  .WORD  47652,125253          ;ANTICIPATED ERRONEOUS RESULT.
5701 027434 000105          144$:  105          ;FPS BEFORE EXECUTION.
5702 027436 000110          110          ;FPS AFTER EXECUTION.
5703 027440 177777          -1          ;ANTICIPATED ERRONEOUS FPS.
5704 027442 104265      145$:  ERROR    +265          ;REPORT RESULT INCORRECT.
5705 027444 000401          BR           146$
5706 027446 104261          ERROR    +261          ;REPORT FPS INCORRECT.
5707 027450      146$:
5708
5709      ;OPERAND=77777,177500    FL=1,    ROUND MODE
5710 027450 012737 027456 001110  MOV     #350$, $LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.      ;DPM002
5711 027456 004737 027724      350$:  JSR     PC,1000$          ;GO EXECUTE THE INSTRUCTION.
5712 027462 077777 177500          151$:  .WORD  77777,177500          ;FSRC OPERAND.
5713 027466 047777 177777          152$:  .WORD  47777,177777          ;EXPECTED RESULT.
5714 027472 047777 177776          153$:  .WORD  47777,177776          ;ANTICIPATED ERRONEOUS RESULT.
5715 027476 000117          154$:  117          ;FPS BEFORE EXECUTION.
5716 027500 000100          100          ;FPS AFTER EXECUTION.
    
```

```

5717 027502 177777          -1          ;ANTICIPATED ERRONEOUS FPS.
5718 027504 104270          155$:  ERROR  +270          ;ST 631 INTO RND
5719 027506 000401          BR        156$
5720 027510 104261          ERROR  +261          ;REPORT FPS INCORRECT.
5721 027512          156$:
5722
5723          ;OPERAND=40000,000100 FL=1, ROUND MODE
5724 027512 012737 027520 001110  MOV      #360$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
5725 027520 004737 027724 360$:  JSR      PC,1000$ ;GO EXECUTE THE INSTRUCTION.
5726 027524 040000 000100 161$:  .WORD   40000,100 ;FSRC OPERAND.
5727 027530 047600 000001 162$:  .WORD   47600,1 ;EXPECTED RESULT.
5728 027534 047600 000000 163$:  .WORD   47600,0 ;ANTICIPATED ERRONEOUS RESULT.
5729 027540 000102 164$:  102 ;FPS BEFORE EXECUTION.
5730 027542 000100 100 ;FPS AFTER EXECUTION.
5731 027544 177777          -1          ;ANTICIPATED ERRONEOUS FPS.
5732 027546 104270          165$:  ERROR  +270          ;REPORT RESULT INCORRECT.
5733 027550 000401          BR        166$
5734 027552 104261          ERROR  +261          ;REPORT FPS INCORRECT.
5735 027554          166$:
5736
5737          ;OPERAND=40000,000100 FL=1, TRUNC MODE
5738 027554 012737 027562 001110  MOV      #370$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
5739 027562 004737 027724 370$:  JSR      PC,1000$ ;GO EXECUTE THE INSTRUCTION.
5740 027566 040000 000100 171$:  .WORD   40000,100 ;FSRC OPERAND.
5741 027572 047600 000000 172$:  .WORD   47600,0 ;EXPECTED RESULT.
5742 027576 047600 000001 173$:  .WORD   47600,1 ;ANTICIPATED ERRONEOUS RESULT.
5743 027602 000157 174$:  157 ;FPS BEFORE EXECUTION.
5744 027604 000140 140 ;FPS AFTER EXECUTION.
5745 027606 177777          -1          ;ANTICIPATED ERRONEOUS FPS.
5746 027610 104271          175$:  ERROR  +271          ;ST 631 ... INTO TRNC
5747 027612 000401          BR        176$
5748 027614 104261          ERROR  +261          ;REPORT FPS INCORRECT.
5749 027616          176$:
5750          ;OPERAND=100000,0 (MOST NEG #) FL=0
5751 027616 012737 027624 001110  MOV      #380$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
5752 027624 004737 027724 380$:  JSR      PC,1000$ ;GO EXECUTE THE INSTRUCTION.
5753 027630 100000 000000 181$:  .WORD   100000,0 ;FSRC OPERAND.
5754 027634 144000 000000 182$:  .WORD   144000,0 ;EXPECTED RESULT.
5755 027640 143600 000000 183$:  .WORD   143600,0 ;ANTICIPATED ERRONEOUS RESULT.
5756 027644 000007 184$:  7 ;FPS BEFORE EXECUTION.
5757 027646 000010 10 ;FPS AFTER EXECUTION.
5758 027650 177777          -1          ;ANTICIPATED ERRONEOUS FPS.
5759 027652 104272          185$:  ERROR  +272          ;ST 630 RH*R14+1
5760 027654 000401          BR        186$
5761 027656 104261          ERROR  +261          ;REPORT FPS INCORRECT.
5762 027660          186$:
5763
5764          ;OPERAND=100000,0 FL=1
5765 027660 012737 027666 001110  MOV      #390$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
5766 027666 004737 027724 390$:  JSR      PC,1000$ ;GO EXECUTE THE INSTRUCTION.
5767 027672 100000 000000 191$:  .WORD   100000,0 ;FSRC OPERAND.
5768 027676 150000 000000 192$:  .WORD   150000,0 ;EXPECTED RESULT.
5769 027702 147600 000000 193$:  .WORD   147600,0 ;ANTICIPATED ERRONEOUS RESULT.
5770 027706 000107 194$:  107 ;FPS BEFORE EXECUTION.
5771 027710 000110 110 ;FPS AFTER EXECUTION.
5772 027712 177777          -1          ;ANTICIPATED ERRONEOUS FPS.
5773 027714 104272          195$:  ERROR  +272          ;REPORT RESULT INCORRECT.
    
```

5774 027716 000401
5775 027720 104261
5776 027722 000506

BR 196\$
ERROR +261
196\$: BR 400\$

;REPORT FPS INCORRECT.

TEST # 56 - LDCIF AND LDCLF TEST

5777
5778
5779
5780
5781
5782
5783
5784
5785
5786
5787
5788
5789
5790
5791
5792
5793
5794
5795
5796
5797
5798
5799
5800
5801
5802
5803
5804
5805
5806
5807
5808
5809
5810
5811
5812
5813
5814
5815
5816
5817
5818
5819
5820
5821
5822
5823
5824
5825
5826
5827
5828
5829
5830
5831
5832
5833

027724 012601
027726 016100 000014
027732 170100
027734 012737 026464 001236
027742 010100
027744 177010
027746 170204
027750 012700 030130
027754 012702 000200
027760 170102
027762 174010
027764 012702 030130
027770 010237 001242
027774 010137 001240
030000 010103
030002 062703 000004
030006 010337 001244
030012 010437 001250
030016 016137 000016 00 252
030024 010100
030026 062700 000004
030032 012703 000002
030036 022022

```
:THIS SUBROUTINE, 1000$, IS USED TO SET UP THE OPERANDS, EXECUTE  
:THE LDCIF OR LDCLF INSTRUCTION AND CHECK THE RESULTS. A CALL  
:TO IT IS MADE THUS:  
:  
:       JSR      PC,1000$  
:       ACARG: .WORD X,X           ;AC OPERAND  
:       RES:   .WORD X,X           ;EXPECTED RESULT  
:       ERRES: .WORD X,X           ;ERROR RESULT  
:       FPSB:  .WORD X             ;FPS BEFORE EXECUTION  
:       FPSA:  .WORD X             ;FPS AFTER EXECUTION  
:       ERFPS: .WORD X             ;ERROR FPS  
:       ERR1:  ERROR +X            ;DATA ERROR  
:       BR     BR   CONT           ;  
:       ERR2:  ERROR +X            ;FPS ERROR  
:       CONT:  .WORD +X            ;RETURN ADDRESS  
:  
:THE OPERANDS ARE SET UP (USING ACO AS THE ACCUMULATOR). THEN  
:THE LDCIF OR LDCLF INSTRUCTION IS EXECUTED.  
:THE RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS  
:COMPARED WITH FPSA IF THIS TOO IS CORRECT 1000$ RETURNS CONTROL  
:TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD 1000$ WILL  
:COMPARE IT TO ERROR FPS. IF THIS MATCHES THEN 1000$ WILL RETURN  
:TO THE ERROR CALL AT ERR2, OTHERWISE 1000$ ITSELF  
:REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE  
:LDCIF OR LDCLF IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE  
:ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN  
:THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN 1000$  
:WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR1. OTHERWISE THE  
:RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND 1000$  
:REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.
```

```
1000$: MOV      (SP)+,R1           ;GET A POINTER TO THE ARGUMENTS.  
        MOV      14(R1),R0       ;SET THE FPS.  
        LDFPS   R0  
        MOV      #1$,STMP2  
        MOV      R1,R0  
  
1001$: LDCIF   (R0),ACO           ;TEST INSTRUCTION LDCIF OR LDCLF.  
  
        STFPS   R4               ;GET FPS.  
        MOV      #1200$,R0       ;GET THE RESULT.  
        MOV      #200,R2  
        LDFPS   R2  
        STD     ACO,(R0)  
  
        MOV      #1200$,R2       ;SEE IF THE RESULT WAS CORRECT.  
        MOV      R2,STMP4  
        MOV      R1,STMP3  
        MOV      R1,R3  
        ADD     #4,R3  
        MOV      R3,STMP5  
        MOV      R4,STMP7  
        MOV      16(R1),STMP10  
        MOV      R1,R0  
        ADD     #4,R0  
        MOV      #2,R3  
1002$: CMP      (R0)+,(R2)+
```

```

5834 030040 001006          BNE      1010$          ;BR IF INCORRECT.
5835 030042 077303          SOB      R3,1002$
5836
5837 030044 026104 000016          CMP      16(R1),R4          ;SEE IF THE FPS WAS CORRECT.
5838 030050 001020          BNE      1015$          ;BR IF INCORRECT.
5839 030052 000161 000030 1003$: JMP      30(R1)          ;RETURN.
5840
5841          ;RESULT IN CORRECT SO SEE IF THE FAILURE WAS ANTICIPATED.
5842 030056 012702 030130 1010$: MOV      #1200$,R2
5843 030062 010100          MOV      R1,R0
5844 030064 062700 000010          ADD      #10,R0
5845 030070 012703 000002 1011$: MOV      #2,R3
5846 030074 022022          CMP      (R0)+,(R2)+
5847 030076 001003          BNE      1013$
5848 030100 077303          SOB      R3,1011$
5849 030102 000161 000022          JMP      22(R1)
5850
5851          ;THE FAILURE WAS NOT ANTICIPATED SO REPORT THE ERROR HERE.
5852 030106 1013$:
5853
5854 030106 104260 1014$: ERROR  +260          ;BAD RES
5855 030110 000760          BR       1003$
5856
5857          ;THE FPS WAS INCORRECT SO SEE IF IT WAS ANTICIPATED.
5858 030112 026104 000020 1015$: CMP      20(R1),R4
5859 030116 001002          BNE      1016$
5861 030120 000161 000026          JMP      26(R1)
5862
5863          ;FPS ERROR NOT ANTICIPATED SO REPORT IT HERE.
5864 030124 1016$:
5865 030124 104261 1017$: ERROR  +261          ;BAD FPS
5866 030126 000751          BR       1003$
5867
5868          ;DATA BUFFER:
5869 030130 000000 000000 000000 1200$: .WORD  0,0,0,0
5870
5871 030140 104412 400$:          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?).
    
```

5877

```

.SBTTL TEST # 57 - LDCID AND LDCLD TEST
*****
*TEST 57      LDCID AND LDCLD TEST
*
* THIS IS A TEST OF LDCID AND LDCLD
*
*****
    
```

5878	030142	000004				TST57: SCOPE			
5878	030144	012737	030152	001110		:OPERAND=0	FL=0, FD=1		
5880	030152	004737	031012		200\$:	MOV	#200\$, \$LPERR	:SET UP THE LOOP ON ERROR ADDRESS.	:DPM002
5881	030156	000000	000000		1\$:	JSR	PC,1000\$:GO EXECUTE THE INSTRUCTION.	
5882	030162	000000	000000	000000	2\$:	.WORD	0,0	:FSRC OPERAND.	
5883	030172	177777	177777	177777	3\$:	.WORD	0,0,0,0	:EXPECTED RESULT.	
5884	030202	000213			4\$:	.WORD	-1,-1,-1,-1	:ANTICIPATED ERRONEOUS RESULT.	
5885	030204	000204				213		:FPS BEFORE EXECUTION.	
5886	030206	177777				204		:FPS AFTER EXECUTION.	
5887	030210	104273			5\$:	ERROR	+273	:ANTICIPATED ERRONEOUS FPS.	
5888	030212	000401				BR	6\$:REPORT RESULT INCORRECT.	
5889	030214	104274				ERROR	+274	:REPORT FPS INCORRECT.	
5890	030216				6\$:				
5891						:OPERAND=0	FL=0, FD=1		
5892	030216	012737	030224	001110		MOV	#210\$, \$LPERR	:SET UP THE LOOP ON ERROR ADDRESS.	:DPM002
5893	030224	004737	031012		210\$:	JSR	PC,1000\$:GO EXECUTE THE INSTRUCTION.	
5894	030230	000000	177777		11\$:	.WORD	0,-1	:FSRC OPERAND.	
5895	030234	000000	000000	000000	12\$:	.WORD	0,0,0,0	:EXPECTED RESULT.	
5896	030244	004177	177400	000000	13\$:	.WORD	4177,177400,0,0	:ANTICIPATED ERRONEOUS RESULT.	
5897	030254	000200			14\$:	200		:FPS BEFORE EXECUTION.	
5898	030256	000204				204		:FPS AFTER EXECUTION.	
5899	030260	177777				-1		:ANTICIPATED ERRONEOUS FPS.	
5900	030262	104275			15\$:	ERROR	+275	:(BUT FL)\$+277	
5901	030264	000401				BR	16\$:TO 300 INTO 301	
5902	030266	104274				ERROR	+274	:REPORT FPS INCORRECT.	
5903	030270				16\$:				
5904						:OPERAND=0	FL=1, FD=1		
5905						MOV	#220\$, \$LPERR	:SET UP THE LOOP ON ERROR ADDRESS.	:DPM002
5906	030270	012737	030276	001110	220\$:	JSR	PC,1000\$:GO EXECUTE THE INSTRUCTION.	
5907	030276	004737	031012		21\$:	.WORD	0,0	:FSRC OPERAND.	
5908	030302	000000	000000		22\$:	.WORD	0,0,0,0	:EXPECTED RESULT.	
5909	030306	000000	000000	000000	23\$:	.WORD	-1,-1,-1,-1	:ANTICIPATED ERRONEOUS RESULT.	
5910	030316	177777	177777	177777	24\$:	211		:FPS BEFORE EXECUTION.	
5911	030326	000211				204		:FPS AFTER EXECUTION.	
5912	030330	000204				-1		:ANTICIPATED ERRONEOUS FPS.	
5913	030332	177777			25\$:	ERROR	+273	:REPORT RESULT INCORRECT.	
5914	030334	104273				BR	26\$		
5915	030336	000401				ERROR	+274	:REPORT FPS INCORRECT.	
5916	030340	104274			26\$:				
5917	030342					:OPERAND=40000	FL=0, FD=1		
5918						MOV	#230\$, \$LPERR	:SET UP THE LOOP ON ERROR ADDRESS.	:DPM002
5919						JSR	PC,1000\$:GO EXECUTE THE INSTRUCTION.	
5920	030342	012737	030350	001110	230\$:	.WORD	40000,0	:FSRC OPERAND.	
5921	030350	004737	031012		31\$:	.WORD	43600,0,0,0	:EXPECTED RESULT.	
5922	030354	040000	000000		32\$:	.WORD	47600,0,0,0	:ANTICIPATED ERRONEOUS RESULT.	
5923	030360	043600	000000	000000	33\$:	.WORD		:FPS BEFORE EXECUTION.	
5924	030370	047600	000000	000000	34\$:	217		:FPS AFTER EXECUTION.	
5925	030400	000217				200			
5926	030402	000200							

6017
 6018
 6019
 6020
 6021
 6022
 6023
 6024
 6025
 6026
 6027
 6028
 6029
 6030
 6031
 6032
 6033
 6034
 6035
 6036
 6037
 6038
 6039
 6040
 6041
 6042
 6043
 6044
 6045
 6046
 6047
 6048
 6049
 6050
 6051
 6052
 6053
 6054
 6055
 6056
 6057
 6058
 6059
 6060
 6061
 6062
 6063
 6064
 6065
 6066
 6067
 6068
 6069
 6070
 6071
 6072
 6073

031012 012601
 031014 016100 000024
 031020 170100
 031022 012737 031032 001236
 031030 010100
 031032 177010
 031034 170204
 031036 012700 031216
 031042 012702 000200
 031046 170102
 031050 174010
 031052 012702 031216
 031056 010237 001242
 031062 010137 001240
 031066 010103
 031070 062703 000004
 031074 010337 001244
 031100 010437 001250
 031104 016137 000026 001252
 031112 010100
 031114 062700 000004
 031120 012703 000002
 031124 022022

:THIS SUBROUTINE, 1000\$, IS USED TO SET UP THE OPERANDS, EXECUTE
 :THE LDCID OR LDCLD INSTRUCTION AND CHECK THE RESULTS. A CALL
 :TO IT IS MADE THUS:

```

      JSR      PC,1000$
      ACARG:  .WORD  X,X           ;AC OPERAND
      RES:    .WORD  X,X,X,X       ;EXPECTED RESULT
      ERRES:  .WORD  X,X,X,X       ;ERROR RESULT
      FPSB:   .WORD  X             ;FPS BEFORE EXECUTION
      FPSA:   .WORD  X             ;FPS AFTER EXECUTION
      ERFPS:  .WORD  X             ;ERROR FPS.
      ERR1:   ERROR  +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 LDCID OR LDCLD INSTRUCTION IS EXECUTED.
 :THE RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
 :COMPARED WITH FPSA IF THIS TOO IS CORRECT 1000\$ RETURNS CONTROL
 :TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD 1000\$
 :COMPARE IT TO ERROR FPS. IF THIS MATCHES THEN 1000\$ WILL RETURN
 :TO THE ERROR CALL AT ERR2, OTHERWISE 1000\$ ITSELF
 :REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE
 :LDCID OR LDCLD IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
 :ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
 :THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN 1000\$
 :WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR1. OTHERWISE THE
 :RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND 1000\$ WILL
 :REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.

```

1000$: MOV      (SP)+,R1           ;GET A POINTER TO THE ARGUMENTS.
      MOV      24(R1),R0         ;SET THE FPS.
      LDFPS   R0
      MOV      #101$,STMP2
      MOV      R1,R0
101$:  LDCID   (R0),ACO          ;TEST INSTRUCTION, LDCID OR LDCLD.

      STFPS   R4                 ;GET FPS.
      MOV      #1200$,R0         ;GET THE RESULT.
      MOV      #200,R2
      LDFPS   R2
      STD     ACO,(R0)

;SEE IF THE RESULT IS CORRECT.
      MOV      #1200$,R2
      MOV      R2,STMP4
      MOV      R1,STMP3
      MOV      R1,R3
      ADD     #4,R3
      MOV      R3,STMP5
      MOV      R4,STMP7
      MOV      26(R1),STMP10
      MOV      R1,R0
      ADD     #4,R0
      MOV      #2,R3
102$:  CMP     (R0)+,(R2)+
  
```

```

6074 031126 001006          BNE 110$          ;BR IF INCORRECT.
6075 031130 077303          SOB R3,102$
6076
6077 031132 026104 000026          CMP 26(R1),R4          ;IS THE FPS CORRECT?
6078 031136 001020          BNE 115$          ;BR IF INCORRECT.
6079 031140 000161 000040 103$: JMP 40(R1)          ;RETURN.
6080
6081          ;THE RESULT WAS INCORRECT SO SEE IF THE ERROR WAS ANTICIPATED.
6082 031144 012702 031216 110$: MOV #1200$,R2
6083 031150 010100          MOV R1,R0
6084 031152 062700 000014          ADD #14,R0
6085 031156 012703 000002          MOV #2,R3
6086 031162 022022 111$: CMP (R0)+,(R2)+
6087 031164 001003          BNE 113$
6088 031166 077303          SOB R3,111$
6089 031170 000161 000032          JMP 32(R1)
6090 031174
6091          ;ERROR NOT ANTICIPATED SO REPORT RESULT INCORRECT HERE.
6092 031174 104273 114$: ERROR +273          ;BAD RES
6093 031176 000760          BR 103$
6094
6095          ;THE FPS WAS INCORRECT. SEE IF FAILURE WAS ANTICIPATED.
6096 031200 026104 000030 115$: CMP 30(R1),R4
6097 031204 001002          BNE 116$
6098 031206 000161 000036          JMP 36(R1)
6099          ;FPS ERROR WAS NOT ANTICIPATED SO REPORT FAILURE HERE.
6100 031212 116$:
6101
6102 031212 104274 117$: ERROP +274          ;BAD FPS
6103 031214 000751          BR 103$
6104
6105
6106          ;DATA BUFFER:
6107 031216 000000 000000 000000 1200$: .WORD 0,0,0,0
6108
6109 031226 104412 300$: 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?).
    
```

6118

```
.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.
:*
:*****
```

```
031230 000004
6119
6120 031232 012737 031240 001110
6121 031240 004737 032674
6122 031244 012345 067012 034567
6123 031254 000010
6124 031256 042145 067012 034567
6125 031266 002145 067012 034567
6126 031276 047217
6127 031300 047200
6128 031302 147200
6129 031304 177777
6130 031306 104304
6131 031310 000400
6132 031312 104305
6133
6134
6135 031314 012737 031322 001110
6136 031322 004737 032674
6137 031326 123456 070123 045670
6138 031336 000177
6139 031340 177656 070123 045670
6140 031350 137656 070123 045670
6141 031360 047207
6142 031362 047210
6143 031364 147210
6144 031366 177777
6145 031370 104304
6146 031372 000401
6147 031374 104305
6148 031376
6149
6150
6151 031376 012737 031404 001110
6152 031404 004737 032674
6153 031410 073261 057645 043323
6154 031420 000056
6155 031422 053461 057645 043323
6156 031432 177777 177777 177777
6157 031442 047200
6158 031444 047200
6159 031446 147200
6160 031450 177777
6161 031452 104301
6162 031454 000401
6163 031456 104305
6164 031460

TST60: SCOPE
:NON-ZERO RES. VALID EXPON=210 (EXCESS 200)=10
      MOV #200$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
200$: JSR PC, 1000$ ;GO EXECUTE THE INSTRUCTION.
1$: .WORD 12345, 67012, 34567, 012345 ;ACO OPERAND.
2$: .WORD 10 ;EXPONENT OPERAND.
3$: .WORD 42145, 67012, 34567, 012345 ;EXPECTED RESULT.
4$: .WORD 2145, 67012, 34567, 012345 ;ANTICIPATED ERRONEOUS RESULT.
5$: 47217 ;FPS BEFORE EXECUTION.
    47200 ;FPS AFTER EXECUTION.
    147200 ;ANTICIPATED ERRONEOUS FPS.
    -1 ;EXPECTED FEC.
6$: ERROR +304 ;E12+E12+200 BAD
    BR 7$ ;ST 624
7$: ERROR +305 ;REPORT FPS INCORRECT.
    ;ST 625 INTO 304

:NON-ZERO RES. NEG.
      MOV #210$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
210$: JSR PC, 1000$ ;EXPON=377
11$: .WORD 123456, 70123, 45670, 123456 ;ACO OPERAND.
12$: .WORD 177 ;EXPONENT OPERAND.
13$: .WORD 177656, 70123, 45670, 123456 ;EXPECTED RESULT.
14$: .WORD 137656, 70123, 45670, 123456 ;ANTICIPATED ERRONEOUS RESULT.
15$: 47207 ;FPS BEFORE EXECUTION.
    47210 ;FPS AFTER EXECUTION.
    147210 ;ANTICIPATED ERRONEOUS FPS.
    -1 ;EXPECTED FEC.
16$: ERROR +304 ;REPORT RESULT INCORRECT.
    BR 17$ ;ST 624
17$: ERROR +305 ;REPORT FPS INCORRECT.

:NON-ZERO RES. EXP=256=(56)REAL
      MOV #220$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
220$: JSR PC, 1000$ ;GO EXECUTE THE INSTRUCTION.
21$: .WORD 73261, 057645, 43323, 101760 ;ACO OPERAND.
22$: .WORD 56 ;EXPONENT OPERAND.
23$: .WORD 53461, 057645, 43323, 101760 ;EXPECTED RESULT.
24$: .WORD -1, -1, -1, -1 ;ANTICIPATED ERRONEOUS RESULT.
25$: 47200 ;FPS BEFORE EXECUTION.
    47200 ;FPS AFTER EXECUTION.
    147200 ;ANTICIPATED ERRONEOUS FPS.
    -1 ;EXPECTED FEC.
26$: ERROR +301 ;REPORT RESULT INCORRECT.
    BR 27$ ;ST 624
27$: ERROR +305 ;REPORT FPS INCORRECT.
```

```
6165
6166
6167 031460 012737 031466 001110 ;EXP=27 (EXCESS 200)=-151 (OCT)
        MOV #230$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
6168 031466 004737 032674 230$: JSR PC, 1000$ ;GO EXECUTE THE INSTRUCTION.
6169 031472 012223 024252 062720 31$: .WORD 12223, 24252, 62720, 21222 ;ACO OPERAND.
6170 031502 177627 32$: .WORD -151 ;EXPONENT OPERAND.
6171 031504 005623 024252 062720 33$: .WORD 5623, 24252, 62720, 21222 ;EXPECTED RESULT.
6172 031514 177777 177777 177777 34$: .WORD -1, -1, -1 ;ANTICIPATED ERRONEOUS RESULT.
6173 031524 047200 47200 ;FPS BEFORE EXECUTION.
6174 031526 047200 47200 ;FPS AFTER EXECUTION.
6175 031530 147200 147200 ;ANTICIPATED ERRONEOUS FPS.
6176 031532 177777 -1 ;EXPECTED FEC.
6177 031534 104301 36$: ERROR +301 ;REPORT RESULT INCORRECT.
6178 031536 000401 BR 37$
6179 031540 104306 ERROR +306 ;(BUT EZBT) ST 544 TO 504 INTO 704 0 (BUT EXBT) ST 704 INTO
6180 031542 37$:
6181
6182 ;EXP=0 (EXCESS 200)=-200 (OCT), POSITIVE FRAC
6183 ; FIV=1
6184 031542 012737 031550 001110 MOV #240$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
6185 031550 004737 032674 240$: JSR PC, 1000$ ;GO EXECUTE THE INSTRUCTION.
6186 031554 030131 032334 035363 41$: .WORD 30131, 32334, 35363, 73031 ;ACO OPERAND.
6187 031564 177600 42$: .WORD -200 ;EXPONENT OPERAND.
6188 031566 000131 032334 035363 43$: .WORD 00131, 32334, 35363, 73031 ;EXPECTED RESULT.
6189 031576 000000 000000 000000 44$: .WORD 0, 0, 0, 0 ;ANTICIPATED ERRONEOUS RESULT.
6190 031606 042200 45$: 42200 ;FPS BEFORE EXECUTION.
6191 031610 142204 142204 ;FPS AFTER EXECUTION.
6192 031612 042202 42202 ;ANTICIPATED ERRONEOUS FPS.
6193 031614 000012 12 ;EXPECTED FEC.
6194 031616 104307 46$: ERROR +307 ;(BUT EXBT) ST 704 TO 64 INST 264
6195 031620 000401 BR 47$
6196 031622 104310 ERROR +310 ;(BUT FIU) ST 264 X
6197 031624 47$:
6198
6199 ;EXP=0 (EXCESS 200)=-200 (OCT), NEG FRACT, FIU=1
6200 031624 012737 031632 001110 MOV #250$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
6201 031632 004737 032674 250$: JSR PC, 1000$ ;GO EXECUTE THE INSTRUCTION.
6202 031636 140414 024344 045464 51$: .WORD 140414, 24344, 45464, 74045 ;ACO OPERAND.
6203 031646 177600 52$: .WORD -200 ;EXPONENT OPERAND.
6204 031650 100014 024344 045464 53$: .WORD 100014, 24344, 45464, 74045 ;-0 ;EXPECTED RESULT.
6205 031660 000000 000000 000000 54$: .WORD 0, 0, 0, 0 ;ANTICIPATED ERRONEOUS RESULT.
6206 031670 042200 55$: 42200 ;FPS BEFORE EXECUTION.
6207 031672 142214 142214 ;FPS AFTER EXECUTION.
6208 031674 042214 42214 ;ANTICIPATED ERRONEOUS FPS.
6209 031676 000012 12 ;EXPECTED FEC.
6210 031700 104307 56$: ERROR +307 ;REPORT RESULT INCORRECT.
6211 031702 000401 BR 57$
6212 031704 104310 ERROR +310 ;REPORT FPS INCORRECT.
6213 031706 57$:
6214
6215 ;EXP=0 (EXCESS 200)=-200 (OCT), POS FRAC, FIU=0
6216
6217 031706 012737 031714 001110 MOV #260$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
6218 031714 004737 032674 260$: JSR PC, 1000$ ;GO EXECUTE THE INSTRUCTION.
6219 031720 051525 035455 005675 61$: .WORD 51525, 35455, 5675, 05152 ;ACO OPERAND.
6220 031730 177600 62$: .WORD -200 ;EXPONENT OPERAND.
6221 031732 000000 000000 000000 63$: .WORD 0, 0, 0, 0 ;EXPECTED RESULT.
```

```
6222 031742 000125 035455 005675 64$: .WORD 00125,35455,5675,05152 ;ANTICIPATED ERRONEOUS RESULT.
6223 031752 045200 45200 ;FPS BEFORE EXECUTION.
6224 031754 045204 45204 ;FPS AFTER EXECUTION.
6225 031756 145204 145204 ;ANTICIPATED ERRONEOUS FPS.
6226 031760 177777 -1 ;EXPECTED FEC.
6227 031762 104311 66$: ERROR +311 ;(BUT FIU) ST 264 X ;REPORT RESULT INCORRECT.
6228 031764 000401 BR 67$
6229 031766 104302 ERROR +302 ;REPORT FPS INCORRECT.
6230 031770 67$:
6231
6232 ;EXP=-1405 (EXCESS 200)=-1605 (OCT), FIU=1
6233 031770 012737 031776 001110 MOV #270$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
6234 031776 004737 032674 270$: JSR PC,1000$ ;GO EXECUTE THE INSTRUCTION.
6235 032002 061626 062636 046566 71$: .WORD 61626,62636,46566,67606 ;ACO OPERAND.
6236 032012 176173 72$: .WORD -1605 ;EXPONENT OPERAND.
6237 032014 076626 062636 046566 73$: .WORD 76626,62636,46566,67606 ;EXPECTED RESULT.
6238 032024 000000 000000 000000 74$: .WORD 0,0,0,0 ;ANTICIPATED ERRONEOUS RESULT.
6239 032034 042200 42200 ;FPS BEFORE EXECUTION.
6240 032036 142200 142200 ;FPS AFTER EXECUTION.
6241 032040 042204 42204 ;ANTICIPATED ERRONEOUS FPS.
6242 032042 000012 12 ;EXPECTED FEC.
6243 032044 104312 76$: ERROR +312 ;(BUT EZBT) ST 544 TO 704 INTO 504
6244 032046 000401 BR 77$
6245 032050 104302 ERROR +302 ;REPORT FPS INCORRECT.
6246 032052 77$:
6247 ;EXP=-17416 (EXCESS 200)=-17616 (OCT), FIU=0
6248 032052 012737 032060 001110 MOV #280$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
6249 032060 004737 032674 280$: JSR PC,1000$ ;GO EXECUTE THE INSTRUCTION.
6250 032064 071727 037475 076777 81$: .WORD 71727,37475,76777,17273 ;ACO OPERAND.
6251 032074 160162 82$: .WORD -17616 ;EXPONENT OPERAND.
6252 032076 000000 000000 000000 83$: .WORD 0,0,0,0 ;EXPECTED RESULT.
6253 032106 074527 037475 076777 84$: .WORD 74527,37475,76777,17273 ;ANTICIPATED ERRONEOUS RESULT.
6254 032116 045200 45200 ;FPS BEFORE EXECUTION.
6255 032120 045204 45204 ;FPS AFTER EXECUTION.
6256 032122 145200 145200 ;ANTICIPATED ERRONEOUS FPS.
6257 032124 177777 -1 ;EXPECTED FEC.
6258 032126 104313 86$: ERROR +313 ;(BUT FIU) ST 504
6259 032130 000401 BR 87$
6260 032132 104302 ERROR +302 ;REPORT FPS INCORRECT.
6261 032134 87$:
6262 ;EXP=-1601 (EXCESS 200)=-2001 (OCT), FIU=1
6263
6264 032134 012737 032142 001110 MOV #290$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
6265 032142 004737 032674 290$: JSR PC,1000$ ;GO EXECUTE THE INSTRUCTION.
6266 032146 001020 030405 006070 91$: .WORD 01020,30405,06070,00102 ;ACO OPERAND.
6267 032156 175777 92$: .WORD -2001 ;EXPONENT OPERAND.
6268 032160 037620 030405 006070 93$: .WORD 37620,30405,06070,00102 ;EXPECTED RESULT.
6269 032170 000000 000000 000000 94$: .WORD 0,0,0,0 ;ANTICIPATED ERRONEOUS RESULT.
6270 032200 042200 42200 ;FPS BEFORE EXECUTION.
6271 032202 142200 142200 ;FPS AFTER EXECUTION.
6272 032204 042204 42204 ;ANTICIPATED ERRONEOUS FPS.
6273 032206 000012 12 ;EXPECTED FEC.
6274 032210 104312 96$: ERROR +312 ;(BUT FIU) ST 504
6275 032212 000401 BR 97$
6276 032214 104302 ERROR +302 ;REPORT FPS INCORRECT.
6277 032216 97$:
6278
```

```
6279 :EXP=1206 (EXCESS 200)=1006 (OCT) FIV=1
6280 032216 012737 032224 001110 MOV #300$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
6281 032224 004737 032674 JSR PC,1000$ ;GO EXECUTE THE INSTRUCTION.
6282 032230 012131 014151 016171 101$: .WORD 12131,14151,16171,10111 ;ACO OPERAND.
6283 032240 001006 102$: .WORD 1006 ;EXPONENT OPERAND.
6284 032242 041531 014151 016171 103$: .WORD 41531,14151,16171,10111 ;EXPECTED RESULT.
6285 032252 000000 000000 000000 104$: .WORD 0,0,0,0 ;ANTICIPATED ERRONEOUS RESULT.
6286 032262 041200 41200 ;FPS BEFORE EXECUTION.
6287 032264 141202 141202 ;FPS AFTER EXECUTION.
6288 032266 041204 41204 ;ANTICIPATED ERRONEOUS FPS.
6289 032270 000010 10 ;EXPECTED FEC.
6290 032272 104314 106$: ERROR +314 ;(BUT FIV) ST 104
6291 032274 000401 BR 107$
6292 032276 104302 ERROR +302 ;REPORT FPS INCORRECT.
6293 032300 107$:
6294
6295 :EXP=16315 (EXCESS 200)=16115 (OCT) FIV=0
6296 032300 012737 032306 001110 MOV #310$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
6297 032306 004737 032674 JSR PC,1000$ ;GO EXECUTE THE INSTRUCTION.
6298 032312 027262 025242 023222 111$: .WORD 27262,25242,23222,21202 ;ACO OPERAND.
6299 032322 016115 112$: .WORD 16115 ;EXPONENT OPERAND.
6300 032324 000000 000000 000000 113$: .WORD 0,0,0,0 ;EXPECTED RESULT.
6301 032334 063262 025242 023222 114$: .WORD 63262,25242,23222,21202 ;ANTICIPATED ERRONEOUS RESULT.
6302 032344 046200 46200 ;FPS BEFORE EXECUTION.
6303 032346 046206 46206 ;FPS AFTER EXECUTION.
6304 032350 146202 146202 ;ANTICIPATED ERRONEOUS FPS.
6305 032352 177777 -1 ;EXPECTED FEC.
6306 032354 104315 116$: ERROR +315 ;(BUT FIV) ST 104
6307 032356 000401 BR 117$
6308 032360 104302 ERROR +302 ;REPORT FPS INCORRECT.
6309 032362 117$:
6310
6311 :EXP=11011 (EXCESS 200)=10611 (OCT) FIV=1
6312
6313 032362 012737 032370 001110 MOV #320$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
6314 032370 004737 032674 JSR PC,1000$ ;GO EXECUTE THE INSTRUCTION.
6315 032374 030313 032333 034353 121$: .WORD 30313,32333,34353,36373 ;ACO OPERAND.
6316 032404 010611 122$: .WORD 10611 ;EXPONENT OPERAND.
6317 032406 002313 032333 034353 123$: .WORD 2313,32333,34353,36373 ;EXPECTED RESULT.
6318 032416 000000 000000 000000 124$: .WORD 0,0,0,0 ;ANTICIPATED ERRONEOUS RESULT.
6319 032426 041200 41200 ;FPS BEFORE EXECUTION.
6320 032430 141202 141202 ;FPS AFTER EXECUTION.
6321 032432 041204 41204 ;ANTICIPATED ERRONEOUS FPS.
6322 032434 000010 10 ;EXPECTED FEC.
6323 032436 104316 126$: ERROR +316 ;(BUT FIV) ST 144
6324 032440 000401 BR 127$
6325 032442 104302 ERROR +302 ;REPORT FPS INCORRECT.
6326 032444 127$:
6327
6328 :EXP=17123 (EXCESS 200)=16723 (OCT) FIV=0
6329
6330 032444 012737 032452 001110 MOV #330$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
6331 032452 004737 032674 JSR PC,1000$ ;GO EXECUTE THE INSTRUCTION.
6332 032456 040414 042434 044454 131$: .WORD 40414,42434,44454,46474 ;ACO OPERAND.
6333 032466 016723 132$: .WORD 16723 ;EXPONENT OPERAND.
6334 032470 000000 000000 000000 133$: .WORD 0,0,0,0 ;EXPECTED RESULT.
6335 032500 024614 042434 044454 134$: .WORD 24614,42434,44454,46474 ;ANTICIPATED ERRONEOUS RESULT.
```



```

6336 032510 046200          135$: 46200              ;FPS BEFORE EXECUTION.
6337 032512 046206          46206              ;FPS AFTER EXECUTION.
6338 032514 146202          146202             ;ANTICIPATED ERRONEOUS FPS.
6339 032516 177777          -1                 ;EXPECTED FEC.
6340 032520 104317          136$: ERROR +317      ;(BUT FIV) ST 144
        032522 000401          BR 137$
6342 032524 104302          ERROR +302        ;REPORT FPS INCORRECT.
6343 032526
6344
6345          ;EXP= 254 (OCT)=    454 (EXCESS 200)      FIV=1
6346
6347 032526 012737 032534 00110    MOV #340$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS.      ;DPM002
6348 032534 004737 032674          340$: JSR PC,1000$ ;GO EXECUTE THE INSTRUCTION.
6349 032540 050515 052535 054555    141$: .WORD 50515,52535,54555,56575 ;ACO OPERAND.
6350 032550 000254          142$: .WORD 254 ;EXPONENT OPERAND.
6351 032552 013115 052535 054555    143$: .WORD 13115,52535,54555,56575 ;EXPECTED RESULT.
6352 032562 000000 000000 000000    144$: .WORD 0,0,0,0 ;ANTICIPATED ERRONEOUS RESULT.
6353 032572 041200          145$: 41200 ;FPS BEFORE EXECUTION.
6354 032574 141202          141202             ;FPS AFTER EXECUTION.
6355 032576 041204          41204             ;ANTICIPATED ERRONEOUS FPS.
6356 032600 000010          10                 ;EXPECTED FEC.
6357 032602 104320          146$: ERROR +320      ;(BUT FIV) ST344
        032604 000401          BR 147$
6359 032606 104302          ERROR +302        ;REPORT FPS INCORRECT.
6360 032610
6361
6362          ;EXP= 313 (OCT)=    513 (EXCESS 200) FIV=0
6363
6364 032610 012737 032616 001110    MOV #350$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS.      ;DPM002
6365 032616 004737 032674          350$: JSR PC,1000$ ;GO EXECUTE THE INSTRUCTION.
6366 032622 060616 062636 064656    151$: .WORD 60616,62636,64656,66676 ;ACO OPERAND.
6367 032632 000313          152$: .WORD 313 ;EXPONENT OPERAND.
6368 032634 000000 000000 000000    153$: .WORD 0,0,0,0 ;EXPECTED RESULT.
6369 032644 022616 062636 064656    154$: .WORD 22616,62636,64656,66676 ;ANTICIPATED ERRONEOUS RESULT.
6370 032654 046200          155$: 46200 ;FPS BEFORE EXECUTION.
6371 032656 046206          46206              ;FPS AFTER EXECUTION.
6372 032660 146202          146202             ;ANTICIPATED ERRONEOUS FPS.
6373 032662 177777          -1                 ;EXPECTED FEC.
6374 032664 104321          156$: ERROR +321      ;(BUT FIV) ST 344
        032666 000401          BR 157$
6376 032670 104302          ERROR +302        ;REPORT FPS INCORRECT.
6377 032672
6378 032672 000540          BR 360$
    
```

6379
6380
6381
6382
6383
6384
6385
6386
6387
6388
6389
6390
6391
6392
6393
6394
6395
6396
6397
6398
6399
6400
6401
6402
6403
6404
6405
6406
6407
6408
6409
6410
6411
6412
6413
6414
6415
6416
6417
6418
6419
6420
6421
6422
6423
6424
6425
6426
6427
6428
6429
6430
6431
6432
6433
6434
6435

032674 012601
032676 012700 000200
032702 170100
032704 010100
032706 172410
032710 012737 032732 001236
032716 016100 000032
032722 170100
032724 010100
032726 062700 000010
032732 176410
032734 170204
032736 170305
032740 012700 000200
032744 170100
032746 012700 033164
032752 174010
032754 010437 001250
032760 016137 000034 001252
032766 010537 001254
032772 016137 000040 001256
033000 010102

:THIS SUBROUTINE, 1000\$, IS USED TO SET UP THE OPERANDS, EXECUTE
:THE LDEXP INSTRUCTION AND CHECK THE RESULTS. A CALL
:TO IT IS MADE THUS:

```

JSR      PC,1000$
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 1000\$ RETURNS CONTROL
:TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD 1000\$
:COMPARE IT TO ERROR FPS. IF THIS MATCHES THEN 1000\$ WILL RETURN
:TO THE ERROR CALL AT ERR2, OTHERWISE 1000\$ 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 1000\$
:WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR1. OTHERWISE THE
:RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND 1000\$ WILL
:REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.

```

1000$:  MOV      (SP)+,R1      :GET A POINTER TO THE ARGUMENTS.
        MOV      #200,R0     :LOAD THE ACO OPERAND.
        LDFPS   R0
        MOV      R1,R0
        LDD     (R0),ACO
        MOV      #161$,STMP2
        MOV      32(R1),R0   :SET UP THE FPS.
        LDFPS   R0
        MOV      R1,R0
        ADD     #10,R0

161$:   LDEXP   (R0),ACO     :TEST INSTRUCTION.

        STFPS   R4          :GET THE FPS.
        STST    R5          :GET THE FEC.
        MOV      #200,R0     :GET THE RESULT.
        LDFPS   R0
        MOV      #1200$,R0
        STD     ACO,(R0)
        MOV      R4,$TMP7
        MOV      34(R1),$TMP10
        MOV      R5,$TMP11
        MOV      40(R1),$TMP12
        MOV      R1,R2
    
```

```

6436 033002 010237 001240      MOV      R2,$TMP3
6437 033006 062702 000010      ADD      #10,R2
6438 033012 011237 001242      MOV      (R2),$TMP4
6439 033016 062702 000002      ADD      #2,R2
6440 033022 010237 001244      MOV      R2,$TMP5
6441 033026 012737 033164      MOV      #1200,$TMP6
6442 033034 012702 033164      MOV      #1200$,R2      ;SEE IF THE RESULT WAS CORRECT.
6443 033040 010103      MOV      R1,R3
6444 033042 062703 000012      ADD      #12,R3
6445 033046 012700 00C004      MOV      #4,R0
6446 033052 022223      162$:    CMP      (R2)+,(R3)+
6447 033054 001014      BNE      170$           ;BRANCH IF NOT CORRECT.
6448 033056 077003      SOB      R0,162$
6449 033060 020461 000034      CMP      R4,34(R1)     ;SEE IF THE FPS WAS CORRECT.
6450 033064 001026      BNE      175$           ;BRANCH IF NOT CORRECT.
6451 033066 005761 000034      TST      34(R1)
6452 033072 100003      BPL      163$
6453 033074 020561 000040      CMP      R5,40(R1)     ;SEE IF THE FEC WAS CORRECT.
6454 033100 001027      BNE      180$           ;BRANCH IF NOT CORRECT.
6455
6456 033102 000161 000050      163$:    JMP      50(R1)         ;RETURN.
6457
6458      ;THE RESULT WAS INCORRECT SO SEE IF THE FAILURE WAS ANTICIPATED.
6459 033106 012702 033164      170$:    MOV      #1200$,R2
6460 033112 010103      MOV      R1,R3
6461 033114 062703 000022      ADD      #22,R3
6462 033120 012700 000004      MOV      #4,R0
6463 033124 022223      171$:    CMP      (R2)+,(R3)+
6464 033126 001003      BNE      172$
6465 033130 077003      SOB      R0,171$
6466 033132 000161 000042      JMP      42(R1)
6467
6468      ;THE ERROR WAS NOT ANTICIPATED SO REPORT IT HERE.
6469 033136      172$:
6470 033136 104301      173$:    ERROR  +301           ;BAD RES
6471 033140 000760      BR      163$
6472
6473      ;SEE IF THE FPS ERROR WAS ANTICIPATED.
6474 033142 026104 000036      175$:    CMP      36(R1),R4
6475 033146 001002      BNE      176$
6476 033150 000161 000046      JMP      46(R1)
6477 033154      176$:
6478      ;THE FPS WAS NOT ANTICIPATED SO REPORT IT HERE.
6479 033154 104302      177$:    ERROR  +302           ;BAD FPS
6480 033156 000751      BR      163$           ;BUT EZBTY8
6481                                     ;ST 063
6482
6483 033160      180$:
6484      ;REPORT FEC INCORRECT.
6485 033160 104303      181$:    ERROR  +303           ;BAD FEC
6486 033162 000747      BR      163$
6487
6488      ;DATA BUFFER:
6489 033164 000000 000000 000000 000000 1200$: .WORD 0,0,0,0
6490
6491 033174      360$:
        033174 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?).

```

6498 .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
6499 033176 000004 033206 001110 MOV #200$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
6500 033200 012737 033304 200$: MOV #210$, R0 ;SET UP THE DATA BUFFER.
6501 033212 012701 000006 MOV #6, R1
6502 033216 012720 177777 1$: MOV #-1, (R0)+
6503 033222 077103 SOB R1, 1$
6504 033224 012700 102345 MOV #102345, R0
6505 033230 012737 033252 001236 MOV #220$, $TMP2
6506 033236 012737 033404 000004 MOV #230$, ERRVECT ;SET UP FOR TRAPS TO 4.
6507 033244 170100 LDFPS R0 ;SET UP FPS.
6508 033246 012700 033310 MOV #240$, R0
6509
6510 033252 170210 220$: STFPS (R0) ;TEST INSTRUCTION.
6511 033254 020027 033310 CMP R0, #240$ ;IS R0 CORRECT?
6512 033260 001017 BNE 250$ ;BRANCH IF NOT CORRECT.
6513 033262 023727 033310 102345 CMP 240$, #102345 ;IS RESULT CORRECT?
6514 033270 001023 BNE 260$ ;BRANCH IF NOT CORRECT.
6515 033272 023727 033312 177777 CMP 240$+2, #-1 ;IS THE RESULT CORRECT?
6516 033300 001030 BNE 270$ ;BRANCH IF NOT CORRECT.
6517 033302 000453 BR 280$
6518
6519 :TEST DATA BUFFER:
6520 033304 177777 177777 210$: .WORD -1, -1
6521 033310 177777 177777 177777 240$: .WORD -1, -1, -1, -1
6522
6523 :REPORT R0 INCORRECT.
6524 033320 010037 001242 250$: MOV R0, $TMP4
6525 033324 012737 033310 001240 MOV #240$, $TMP3
6526 033332 104377 ERROR +377
6527 033334 000001 .WORD 1 ;RO BAD (BUT
6528 033336 000435 BR 280$ ; FDST)X
6529
6530 :REPORT RESULT INCORRECT.
6531 033340 012737 102345 001240 260$: MOV #102345, $TMP3 ; ST 634
6532 033346 013737 033310 001242 MOV 240$, $TMP4
6533 033354 104377 ERROR +377
6534 033356 000002 .WORD 2 ;BAD DATA
6535
6536 033360 000424 BR 280$
6537
6538 :REPORT RESULT INCORRECT.
6539 033362 012737 177777 001240 270$: MOV #-1, $TMP3
6540 033370 013737 033312 001242 MOV 240$+2, $TMP4
6541 033376 104377 ERROR +377
6542 033400 000003 .WORD 3 ;(BUT GR7, FL)
6543 033402 000413 BR 280$ ;ST 357 TO 416
    
```

```
6544  
6545  
6546  
6547  
6548  
6549 033404 011604  
6550 033406 020427 033254  
6551 033412 001402  
6552 033414 000137 051774  
6553  
6554 033420 011637 001236  
6555 033424 022626  
6556 033426 104377  
033430 000004  
6557  
6558  
6559 033432  
033432 104412
```

: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.

```
230$: MOV (SP),R4  
CMP R4,#220$+2  
BEQ 2$  
JMP CPSPUR
```

```
2$: MOV (SP),$TMP2  
CMP (SP)+,(SP)+  
ERROR +377  
.WORD 4
```

;(BUT FDST)+ ST634

```
280$: 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?).

```

6560          .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
              :*
              :*****
6561 033434 000004          TST62: SCOPE
6562 033436 012737 033444 001110      MOV      #200$, $LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.      ;DPM002
6563 033444 012700 033542              200$:   MOV      #210$, R0      ;SET UP THE DATA BUFFER.
6564 033450 012701 000006              MOV      #6, R1
6565 033454 012720 177777              1$:    MOV      #-1, (R0)+
6566 033460 077103              SOB      R1, 1$
6567 033462 012700 105412              MOV      #105412, R0
6568 033466 012737 033510 001236      MOV      #220$, $TMP2
6569 033474 012737 033642 000004      MOV      #230$, ERRVECT ;SET UP FOR TRAPS TO VECTOR 4.
6570 033502 170100              LDFPS   R0      ;SET UP FPS.
6571 033504 012700 033546              MOV      #240$, R0
6572 033510 170220              220$:  STFPS   (R0)+      ;TEST INSTRUCTION.
6573 033512 020027 033550              CMP      R0, #240$+2      ;IS R0 CORRECT?
6574 033516 001017              BNE      250$      ;BRANCH IF NOT CORRECT.
6575 033520 023727 033546 105412      CMP      240$, #105412 ;IS THE RESULT CORRECT?
6576 033526 001023              BNE      260$      ;BRANCH IF NOT CORRECT.
6577 033530 023727 033550 177777      CMP      240$+2, #-1 ;IS THE RESULT CORRECT?
6578 033536 001030              BNE      270$      ;BRANCH IF NOT CORRECT.
6579 033540 000453              BR       280$
6580
6581          ;TEST DATA BUFFER:
6582 033542 177777 177777 177777      210$:   .WORD   -1, -1
6583 033546 177777 177777 177777      240$:   .WORD   -1, -1, -1, -1
6584
6585          ;REPORT R0 INCORRECT.
6586 033556 010037 001242              250$:   MOV      R0, $TMP4
6587 033562 012737 033550 001240      MOV      #240$+2, $TMP3
6588 033570 104377              ERROR   +377
6589 033572 000005              .WORD   5
6590
6591          ;RO BAD (BUT
6592          ;      FDST)X
6593 033576 012737 105412 001240      :REPORT RESULT INCORRECT.
6594 033604 013737 033546 001242      260$:   MOV      #105412, $TMP3
6595 033612 104377 000006              MOV      240$, $TMP4
6596 033614 000006              ERROR   +377
6597          ;BAD DATA
6598          .WORD   6
6599
6600          ;REPORT RESULT INCORRECT.
6601 033620 012737 177777 001240      270$:   MOV      #-1, $TMP3
6602 033626 013737 033550 001242      MOV      240$+2, $TMP4
6603 033634 104377              ERROR   +377
6604 033636 000007              .WORD   7
6605          ;(BUT GR7, FL)
6605 033640 000413              BR       280$
6605          ;ST 357 TO 416
    
```

6606
 6607
 6608
 6609
 6610
 6611 033642 011604
 6612 033644 020427 033512
 6613 033650 001402
 6614 033652 000137 051774
 6615
 6616 033656 011637 001236
 6617 033662 022626
 6618 033664 104377
 033666 000010
 6619
 6620
 6621 033670
 033670 104412

: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.
 230\$: MOV (SP),R4
 CMP R4,#220\$+2
 BEQ 2\$
 JMP CPSPUR
 2\$: MOV (SP),STMP2
 CMP (SP)+,(SP)+
 ERROR +377
 .WORD 10
 : (BUT FDST)+ ST634
 280\$: 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?).

6622

```

.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
:
*****
    
```

```

6623 033672 000004
6624 033702 012700 033702 001110
6625 033706 012701 034000
6626 033712 012720 000006 177777
6627 033716 077103
6628 033720 012700 105555
6629 033724 012737 033746 001236
6630 033732 012737 034100 000004
6631 033740 170100
6632 033742 012700 034006
6633
6634 033746 170240
6635 033750 020027 034004
6636 033754 001017
6637 033756 023727 034004 105555
6638 033764 001023
6639 033766 023727 034006 177777
6640 033774 001030
6641 033776 000453
6642
6643
6644 034000 177777 177777
6645 034004 177777 177777 177777
6646
6647
6648 034014 010037 001242
6649 034020 012737 034004 001240
6650 034026 104377
034030 000011
6651
6652 034032 000435
6653
6654
6655 034034 012737 105555 001240
6656 034042 013737 034004 001242
6657 034050 104377
034052 000012
6658
6659 034054 000424
6660
6661
6662
6663 034056 012737 177777 001240
6664 034064 013737 034006 001242
6665 034072 104377
034074 000013
6666
6667 034076 000413
    
```

```

TST63: SCOPE
MOV #200$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
200$: MOV #210$, R0 ;SET UP THE DATA BUFFER.
MOV #6, R1
1$: MOV #-1, (R0)+
SOB R1, 1$
MOV #105555, R0
MOV #220$, $TMP2
MOV #230$, $ERRVECT ;SET UP FOR TRAPS TO VECTOR 4.
LDFPS R0 ;SET UP FPS.
MOV #240$+2, R0
220$: STFPS -(R0) ;TEST INSTRUCTION.
CMP R0, #240$ ;IS R0 CORRECT?
BNE 250$ ;BRANCH IF NOT CORRECT.
CMP 240$, #105555 ;IS THE RESULT CORRECT?
BNE 260$ ;BRANCH IF NOT CORRECT.
CMP 240$+2, #-1 ;IS THE RESULT CORRECT?
BNE 270$ ;BRANCH IF NOT CORRECT.
BR 280$

:TEST DATA BUFFER:
210$: .WORD -1, -1
240$: .WORD -1, -1, -1, -1

:REPORT R0 INCORRECT.
250$: MOV R0, $TMP4
MOV #240$, $TMP3
ERROR +377
.WORD 11
;RO BAD (BUT
; FDST)X

:REPORT RESULT INCORRECT.
260$: MOV #105555, $TMP3 ; ST 634
MOV 240$, $TMP4
ERROR +377
.WORD 12
;BAD DATA

:REPORT RESULT INCORRECT.
270$: MOV #-1, $TMP3
MOV 240$+2, $TMP4
ERROR +377
.WORD 13
;(BUT GR7, FL)
;ST 357 TO 416
    
```

6668
6669
6670
6671
6672
6673 034100 011604
6674 034102 020427 033750
6675 034106 001402
6676 034110 000137 051774
6677
6678 034114 011637 001236
6679 034120 022626
6680 034122 104377
034124 000014
6681
6682 034126
034126 104412

: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.

230\$: MOV (SP),R4
CMP R4,#220\$+2
BEQ 2\$
JMP (PSPUR

2\$: MOV (SP),\$TMP2
CMP (SP)+,(SP)+
ERROR +377
.WORD 14

;(BUT FDST)+ ST634

280\$: 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?).

6683

```

.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
:
:*****
    
```

```

6684 034130 000004
6685 034132 012737 034140 001110
6686 034140 012700 034242
6687 034144 012701 000010
6688 034150 012720 177777
6689 034154 077103
6690 034156 012700 106653
6691 034162 012737 034210 001236
6692 034177 012737 034346 000004
6693 034176 170100
6694 034200 012700 034256
6695 034204 012710 034246
6696 034210 170230
6697 034212 020027 034260
6698 034216 001021
6699 034220 023727 034246 106653
6700 034226 001025
6701 034230 023727 034256 034246
6702 034236 001032
6703 034240 000455
6704
6705
6706 034242 177777 177777
6707 034246 177777 177777 177777
6708 034256 177777 177777
6709
6710
6711 034262 010037 001242
6712 034266 012737 034260 001240
6713 034274 104377
6714 034276 000015
6715 034300 000435
6716
6717
6718 034302 012737 106653 001240
6719 034310 013737 034246 001242
6720 034316 104377
6721 034320 000016
6722 034322 000424
6723
6724
6725
6726 034324 012737 034256 001240
6727 034332 013737 034250 001242
6728 034340 104377
6729 034342 000017

TST64: SCOPE
200$: MOV #200$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
      MOV #210$, R0 ;SET UP THE DATA BUFFER.
      MOV #10, R1
1$: MOV #-1, (R0)+
     SOB R1, 1$
     MOV #106653, R0
     MOV #220$, $TMP2
     MOV #230$, $ERRVECT ;SET UP FOR TRAPS TO VECTOR 4.
     LDFPS R0 ;SET UP FPS.
     MOV #240$, R0
     MOV #250$, (R0)

220$: STFPS @ (R0)+ ;TEST INSTRUCTION.
      CMP R0, #240$+2 ;IS R0 CORRECT?
      BNE 260$ ;BRANCH IF NOT CORRECT.
      CMP 250$, #106653 ;IS THE RESULT CORRECT?
      BNE 270$ ;BRANCH IF NOT CORRECT.
      CMP 240$, #250$ ;IS THE RESULT CORRECT?
      BNE 280$ ;BRANCH IF NOT CORRECT.
      BR 290$

:TEST DATA BUFFER:
210$: .WORD -1, -1
250$: .WORD -1, -1, -1, -1
240$: .WORD -1, -1

:REPORT R0 INCORRECT.
260$: MOV R0, $TMP4
      MOV #240$+2, $TMP3
      ERROR +377
      .WORD 15
;RC BAD (BUT
: FDST)X

:REPORT RESULT INCORRECT.
270$: MOV #106653, $TMP3 ; ST 634
      MOV 250$, $TMP4
      ERROR +377
      .WORD 16
;BAD DATA

:REPORT RESULT INCORRECT.
280$: MOV #240$, $TMP3 ; (BUT FDST)
      MOV 250$+2, $TMP4
      ERROR +377
      .WORD 17
    
```

6729 034344 000413

BR 290\$

6730

6731

6732

6733

6734

6735 034346 011604 034212

6736 034350 020427 034212

6737 034354 001402 051774

6738 034356 000137 051774

6739 034362 011637 001236

6740 034366 022626 001236

6741 034370 104377 000020

6742 034372 000020 000020

6743 034374 104412 000020

6744 034374 104412 000020

: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.

230\$: MOV (SP),R4
 CMP R4,#220\$+2
 BEQ 2\$
 JMP CPSPUR

2\$: MOV (SP),STMP2
 CMP (SP)+,(SP)+
 ERROR +377
 .WORD 20

;(BUT FDST)+ ST634

290\$: 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?).

6745

```

.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
*
*****
    
```

```

6746 034376 000004
6746 034400 012737 034406 001110
6747 034406 012700 034512
6748 034412 012701 000006
6749 034416 012720 177777
6750 034422 077103
6751 034424 012700 004301
6752 034430 012737 034460 001236
6753 034436 012737 034616 000004
6754 034444 170100
6755 034446 012700 034530
6756 034452 012760 034516 177776
6757
6758 034460 170250
6759 034462 020027 034526
6760 034466 001021
6761 034470 023727 034516 004301
6762 034476 001025
6763 034500 023727 034526 034516
6764 034506 001032
6765 034510 000455
6766
6767
6768 034512 177777 177777
6769 034516 177777 177777 177777
6770 034526 177777 177777
6771
6772
6773 034532 010037 001242
6774 034536 012737 034526 001240
6775 034544 104377
    034546 000021
6776
6777 034550 000435
6778
6779
6780 034552 012737 004301 001240
6781 034560 013737 034516 001242
6782 034566 104377
    034570 000022
6783
6784 034572 000424
6785
6786
6787
6788 034574 012737 034526 001240
6789 034602 013737 034520 001242
6790 034610 104377
    034612 000023

TST65: SCOPE
MOV #200$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
200$: MOV #210$, R0 ;SET UP THE DATA BUFFER.
MOV #6, R1
1$: MOV #-1, (R0)+
SOS R1, 1$
MOV #004301, R0
MOV #220$, $TMP2
MOV #230$, $ERRVECT ;SET UP FOR TRAPS TO VECTOR 4.
LDFPS R0 ;SET UP FPS.
MOV #240$+2, R0
MOV #250$, -2(R0)

220$: STFPS @-(R0) ;TEST INSTRUCTION.
CMP R0, #240$ ;IS R0 CORRECT?
BNE 260$ ;BRANCH IF NOT CORRECT.
CMP 250$, #004301 ;IS THE RESULT CORRECT?
BNE 270$ ;BRANCH IF NOT CORRECT.
CMP 240$, #250$ ;IS THE RESULT CORRECT?
BNE 280$ ;BRANCH IF NOT CORRECT.
BR 290$

;TEST DATA BUFFER:
210$: .WORD -1, -1
250$: .WORD -1, -1, -1, -1
240$: .WORD -1, -1

;REPORT RO INCORRECT.
260$: MOV R0, $TMP4
MOV #240$, $TMP3
ERROR +377
.WORD 21

;RC BAD (BUT
; FDST)X

;REPORT RESULT INCORRECT.
270$: MOV #004301, $TMP3 ; ST 634
MOV 250$, $TMP4
ERROR +377
.WORD 22

;BAD DATA
BR 290$

;REPORT RESULT INCORRECT.
280$: MOV #240$, $TMP3 ;BUT FDST)
MOV 250$+2, $TMP4
ERROR +377
.WORD 23
    
```


6808

```
.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
:*****
```

```

6809 034646 000004
6810 034650 012767 034656 144232
6811 034656 012700 034766
6812 034662 012701 000096
6813 034666 012720 177777
6814 034672 077103
6815 034674 012700 102514
6816 034700 012767 034724 144330
6817 034706 012767 035066 143070
6818 034714 170100
6819 034716 005001
6820 034720 012700 027571
6821
6822 034724 170260 005201
6823 034730 020127 000000
6824 034734 001070
6825 034736 020027 027571
6826 034742 001017
6827 034744 026727 000022 102514
6828 034752 001023
6829 034754 026727 000014 177777
6830 034762 001030
6831 034764 000456
6832
6833
6834 034766 177777 177777
6835 034772 177777 177777 177777
6836
6837
6838 035002 010067 144234
6839 035006 012767 027571 144224
6840 035014 104377
035016 000025
6841
6842 035020 000440
6843
6844
6845 035022 012767 102534 144210
6846 035030 016767 177736 144204
6847 035036 104377
035040 000026
6848
6849 035042 000427
6850
6851
6852
6853 035044 012767 177777 144166
6854 035052 016767 177716 144162

TST66: SCOPE
.DSABL AMA ;DISABLE MODE 6 TO MODE 3 CONVERSIONS
MOV #200$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
200$: MOV #210$, R0 ;SET UP THE DATA BUFFER.
MOV #6, R1
1$: MOV #-1, (R0)+
SOB R1, 1$
MOV #102514, R0
MOV #220$, $TMP2
MOV #230$, $ERRVECT ;SET UP FOR TRAPS TO VECTOR 4.
LDFPS R0 ;SET UP FPS.
CLR R1
MOV #240$-5201, R0
220$: STFPS 5201(R0) ;TEST INSTRUCTION.
CMP R1, #0 ;WAS PC CORRECT AFTER EXECUTION?
BNE 250$ ;BRANCH IF NOT CORRECT.
CMP R0, #240$-5201 ;IS R0 CORRECT?
BNE 260$ ;BRANCH IF NOT CORRECT.
CMP 240$, #102514 ;IS THE RESULT CORRECT?
BNE 270$ ;BRANCH IF NOT CORRECT.
CMP 240$+2, #-1 ;IS THE RESULT CORRECT?
BNE 280$ ;BRANCH IF NOT CORRECT.
BR 290$

;TEST DATA BUFFER:
210$: .WORD -1,-1
240$: .WORD -1,-1,-1,-1

;REPORT R0 INCORRECT.
260$: MOV R0, $TMP4
MOV #240$-5201, $TMP3
ERROR +377
.WORD 25
;RO BAD
BR 290$

;REPORT RESULT INCORRECT.
270$: MOV #102534, $TMP3
MOV 240$, $TMP4
ERROR +377
.WORD 26
;BAD DATA
BR 290$

;REPORT RESULT INCORRECT.
280$: MOV #-1, $TMP3
MOV 240$+2, $TMP4
```

```

6855 035060 104377          ERROR +377
        035062 000027          .WORD 27
6856
6857 035064 000416          BR      290$                ;(BUT GR7,FL)
6858
6859
6860
6861
6862
6863 035066 011604          ;IF A TRAP TO VECTOR 4 OCCURS COME HERE TO SEE IF THE TRAP OCCURRED
6864 035070 020427 034726  ;DURING EXECUTION OF THE FPP INSTRUCTION BEING TESTED, IF NOT GO
6865 035074 001402          ;TO THE SPURIOUS TRAP TO 4 HANDLER.
6866 035076 000167 014672  230$:  MOV      (SP),R4
        035076 000167 014672  ;CMP      R4,#220$+2
6867
6868 035102 011667 144130  2$:   MOV      (SP),$TMP2
6869 035106 022626          ;CMP      (SP)+,(SP)+
6870 035110 104377          ERROR +377
        035112 000030          .WORD 30
6871
6872 035114 000402          BR      290$                ;(BUT FDST)+ ST634
6873
6874
6875 035116
6876 035116 104377          ;REPORT PC NOT INCREMENTED BY 2 DURING EXECUTION.
        035120 000031          250$:  ERROR +377
        035120 000031          .WORD 31
6877
6878 035122
        035122 104412          290$:  RSETUP                ;PC NOT INCREMENTED BY 2
6879
        .ENABL AMA                ;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
    
```


6880

```
.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
6881 035124 000004 035134 001110 MOV #200$,SLPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
6882 035126 012737 035252 MOV #210$,RO ;SET UP THE DATA BUFFER.
6883 035134 012700 035252 MOV #10,R1
6884 035140 012701 000010
6885 035144 012720 177777 1$: MOV #-1,(RO)+
6886 035150 077103 SOB R1,1$
6887 035152 012700 103747 MOV #103747,RO
6888 035156 012737 035210 001236 MOV #220$,STMP2
6889 035164 012737 035356 000004 MOV #230$,ERRVECT ;SET UP FOR TRAPS TO VECTOR 4.
6890 035172 170100 LDFPS R0 ;SET UP FPS.
6891 035174 005001 CLR R1
6892 035176 012700 030065 MOV #240$-5201,RO
6893 035202 012760 035256 005201 MOV #250$,5201(RO)
6894 035210 170270 005201 220$: STFPS @5201(RO) ;TEST INSTRUCTION.
6895 035214 022701 000000 CMP #0,R1 ;WAS PC CORRECT AFTER EXECUTION?
6896 035220 001072 BNE 260$ ;BRANCH IF NOT CORRECT.
6897 035222 020027 030065 CMP RO,#240$-5201 ;IS RO CORRECT?
6898 035226 001021 BNE 270$ ;BRANCH IF NOT CORRECT.
6899 035230 023727 035256 103747 CMP 250$,#103747 ;IS THE RESULT CORRECT?
6900 035236 001025 BNE 280$ ;BRANCH IF NOT CORRECT.
6901 035240 023727 035260 177777 CMP 250$+2,#-1 ;IS THE RESULT CORRECT?
6902 035246 001032 BNE 290$ ;BRANCH IF NOT CORRECT.
6903 035250 000460 BR 300$
6904
6905 ;TEST DATA BUFFER:
6906 035252 177777 177777 210$: .WORD -1,-1
6907 035256 177777 177777 177777 250$: .WORD -1,-1,-1,-1
6908 035266 177777 177777 240$: .WORD -1,-1
6909
6910 ;REPORT RO INCORRECT.
6911 035272 010037 001242 001240 270$: MOV RO,STMP4
6912 035276 012737 030065 MOV #240$-5201,STMP3
6913 035304 104377 ERROR +377
6914 035306 000032 .WORD 32 ;RO BAD
6915 035310 000440 BR 300$
6916
6917
6918 ;REPORT RESULT INCORRECT.
6919 035312 012737 103747 001240 280$: MOV #103747,STMP3
6920 035320 013737 035256 001242 MOV 250$,STMP4
6921 035326 104377 ERROR +377
6922 035330 000033 .WORD 33 ;BAD DATA
6923 035332 000427 BR 300$
6924
6925
6926 ;REPORT RESULT INCORRECT.
```

```

6927 035334 012737 177777 001240 290$: MOV #1,$TMP3
6928 035342 013737 035260 001242 MOV 250$+2,$TMP4
6929 035350 104377 ERROR +377
      035352 000034 .WORD 34
6930
6931 035354 000416 BR 300$ ;(BUT GR7,FL)
6932 ;ST 357 TO 416
6933 ;INTO 417
6934
6935 ;IF A TRAP TO VECTOR 4 OCCURS COME HERE TO SEE IF THE TRAP OCCURRED
6936 ;DURING EXECUTION OF THE FPP INSTRUCTION BEING TESTED, IF NOT GO
6937 035356 011604 ;TO THE SPURIOUS TRAP TO 4 HANDLER.
6938 035360 020427 035212 230$: MOV (SP),R4
6939 035364 001402 CMP R4,#220$+2
6940 035366 000137 051774 BEQ 2$
6941 035372 011637 001236 JMP CPSPUR
6942 035376 022626 2$: MOV (SP),$TMP2
6943 035400 104377 CMP (SP)+,(SP)+
      035402 000035 ERROR +377
      .WORD 35 ;(BUT FSDT)+ ST634
6944
6945 035404 000402 BR 300$
6946
6947 ;REPORT PC NOT INCREMENTED BY 2 DURING EXECUTION.
6948 035406 260$: ERROR +377
      035406 104377 .WORD 36
      035410 000036 ;PC NOT
      ;INCREMENTED
6949
6950
6951 035412 300$: RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
      035412 104412 ;SEE IF THE USER HAS EXPRESSED
      ;THE DESIRE TO CHANGE THE SOFTWARE
      ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
      ;THE USER TYPED CONTROL G?).
    
```

6958

```

.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
:*
:*****
    
```

```

6959 035414 000004 035424 001110 TST70: SCOPE
6960 035416 012737 000300      MOV #200$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
6961 035424 012700      MOV #300$, R0 ;SET UP FPS.
6962 035430 170100      LDFPS R0
6963 035432 012700 035502      MOV #210$, R0 ;SET UP THE ACO OPERAND.
6964 035436 172410      LDD (R0), ACO
6965 035440 012737 035452 001236      MOV #220$, $TMP2
6966 035446 012700 035514      MOV #230$, R0
6967 035452 175420      220$: STCDL ACO, (R0)+ ;TEST INSTRUCTION.
6968
6969 035454 020027 035520      CMP R0, #230$+4 ;IS R0 CORRECT?
6970 035460 001420      BEQ 240$ ;BRANCH IF CORRECT.
6971
6972 ;REPORT R0 INCORRECT.
6973 035462 010037 001242      MOV R0, $TMP4
6974 035466 012737 035520 001240      MOV #230$+4, $TMP3
6975 035474 104377      ERROR +377
        035476 000037      .WORD 37
        ;RO NOT INCR BY 4
6976
6977 035500 000410      BR 240$
6978 ;TEST DATA BUFFER:
6979 035502 000000 000000 000000 210$: .WORD 0,0,0,0
6980 035512 177777      -1
6981 035514 177777 177777 177777 230$: .WORD -1,-1,-1
6982
6983 035522      240$: RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
        035522 104412 ;SEE IF THE USER HAS EXPRESSED
        ;THE DESIRE TO CHANGE THE SOFTWARE
        ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
        ;THE USER TYPED CONTROL G?).
    
```

6990

```
.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
6991 035524 000004 035534 001110 MOV #200$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
6992 035534 012700 000300 200$: MOV #300, R0 ;SET UP FPS.
6993 035540 170100 LDFPS R0
6994 035542 012700 035612 MOV #210$, R0 ;SET UP THE ACO OPERAND.
6995 035546 172410 LDD (R0), ACO
6996 035550 012737 035562 001236 MOV #220$, $TMP2
6997 035556 012700 035630 MOV #230$+4, R0
6998
6999 035562 175440 220$: STCDL ACO, -(R0) ;TEST INSTRUCTION.
7000
7001 035564 020027 035624 CMP R0, #230$ ;IS R0 CORRECT?
7002 035570 001420 BEQ 240$
7003
7004 ;REPORT R0 INCORRECT.
7005 035572 010037 001242 MOV R0, $TMP4
7006 035576 012737 035624 001240 MOV #230$, $TMP3
7007 035604 104377 ERROR +377
035606 000040 .WORD 40 ;R0 NOT DECR BY 4
7008
7009 035610 000410 BR 240$
7010 ;TEST DATA BUFFER:
7011 035612 000000 000000 000000 210$: .WORD 0,0,0,0
7012 035622 177777 -1
7013 035624 177777 177777 177777 230$: .WORD -1,-1,-1
7014
7015 035632 240$: RSETUP ;GO INITIALIZE THE FPS AND STACK; AND
035632 104412 ;SEE IF THE USER HAS EXPRESSED
;THE DESIRE TO CHANGE THE SOFTWARE
;VIRTUAL CONSOLE SWITCH REGISTER (HAS
;THE USER TYPED CONTROL G?).
```

7025

```

.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.
:
*****
    
```

```

035634 000004
7026
7027 035636 012737 035644 001110
7028 035644 004737 037110
7029 035650 020000 000000 000000
7030 035660 000000 000000
7031 035664 177777 177777
7032 035670 040300
7033 035672 040304
7034 035674 40304
7035 035676 177777
7036 035700 104322
7037 035702 000401
7038 035704 104325
7039 035706
7040
7041
7042
7043
7044 035706 012737 035714 001110
7045 035714 004737 037110
7046 035720 040000 000000 000000
7047 035730 000000 000000
7048 035734 177777 177777
7049 035740 040313
7050 035742 040304
7051 035744 140304
7052 035746 177777
7053 035750 104322
7054 035752 000401
7055 035754 104326
7056 035756
7057
7058
7059 035756 012737 035764 001110
7060 035764 004737 037110
7061 035770 047667 075757 157737
7062 036000 055675 173757
7063 036004 122102 004021
7064 036010 040717
7065 036012 040700
7066 036014 140705
7067 036016 177777
7068 036020 104327
7069 036022 000401
7070 036024 104326

TST72: SCOPE
:FIRST TEST STC WITH EXP=100 (EXCESS 200)
:MOV #200$, $LPERR :SET UP THE LOOP ON ERROR ADDRESS. :DPM002
200$: 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
:MOV #210$, $LPERR :SET UP THE LOOP ON ERROR ADDRESS. :DPM002
210$: JSR PC, STCSUB :GO EXECUTE THE INSTRUCTION.
11$: .WORD 40000,0,0,0 :AC :ACO OPERAND.
12$: .WORD 0,0 :EXPECTED RESULT.
13$: .WORD -1,-1 :ANTICIPATED ERRONEOUS RESULT.
14$: 40313 :FPS BEFORE EXECUTION.
40304 :FPS AFTER EXECUTION.
140304 :ANTICIPATED ERRONEOUS FPS.
-1 :EXPECTED FEC.
15$: ERROR +322 :REPORT RESULT INCORRECT.
BR 16$
16$: ERROR +326 :REPORT FPS INCORRECT.

:EXP=37 (OCT) FL=1 FIC=1
:MOV #220$, $LPERR :SET UP THE LOOP ON ERROR ADDRESS. :DPM002
220$: JSR PC, STCSUB :GO EXECUTE THE INSTRUCTION.
21$: .WORD 47667,75757,157737,167773 :ACO OPERAND.
22$: .WORD 55675,173757 :EXPECTED RESULT.
23$: .WORD 122102,004021 :ANTICIPATED ERRONEOUS RESULT.
24$: 40717 :FPS BEFORE EXECUTION.
40700 :FPS AFTER EXECUTION.
140705 :ANTICIPATED ERRONEOUS FPS.
-1 :EXPECTED FEC.
25$: ERROR +327 ;(BUT ENBT) ST 632
BR 26$
ERROR +326 :REPORT FPS INCORRECT.
    
```

7071	036026				26\$:				
7072									
7073						:EXP=40 (OCT)	FL=1 FIC=1		
7074	036026	012737	036034	001110		MOV	#230\$, \$LPERR	:SET UP THE LOOP ON ERROR ADDRESS.	:DPM002
7075	036034	004737	037110		230\$:	JSR	PC, STCSUB	:GO EXECUTE THE INSTRUCTION.	
7076	036040	050000	000000	000000	31\$:	.WORD	50000,0,0,0	:ACO OPERAND.	
7077	036050	000000	000000		32\$:	.WORD	0,0	:EXPECTED RESULT.	
7078	036054	177777	177777		33\$:	.WORD	-1,-1	:ANTICIPATED ERRONEOUS RESULT.	
7079	036060	040700			34\$:	40700		:FPS BEFORE EXECUTION.	
7080	036062	140705				140705		:FPS AFTER EXECUTION.	
7081	036064	040705				40705		:ANTICIPATED ERRONEOUS FPS.	
7082	036066	000006				6	:EXPECTED FEC.		
7083	036070	104322			35\$:	ERROR	+322	:REPORT RESULT INCORRECT.	
7084	036072	000401				BR	36\$		
7085	036074	104330				ERROR	+330	:(BUT FIC) ST 004	:REPORT FPS INCORRECT.
7086								:TO 305 INTO	
7087	036076				36\$:			:315	
7088									
7089						:EXP=40 (OCT)	FL=1 FIC=0		
7090	036076	012737	036104	001110		MOV	#240\$, \$LPERR	:SET UP THE LOOP ON ERROR ADDRESS.	:DPM002
7091	036104	004737	037110		240\$:	JSR	PC, STCSUB	:GO EXECUTE THE INSTRUCTION.	
7092	036110	050000	000000	000000	41\$:	.WORD	50000,0,0,0	:ACO OPERAND.	
7093	036120	000000	000000		42\$:	.WORD	0,0	:EXPECTED RESULT.	
7094	036124	177777	177777		43\$:	.WORD	-1,-1	:ANTICIPATED ERRONEOUS RESULT.	
7095	036130	040312			44\$:	40312		:FPS BEFORE EXECUTION.	
7096	036132	040305				40305		:FPS AFTER EXECUTION.	
7097	036134	140305				140305		:ANTICIPATED ERRONEOUS FPS.	
7098	036136	177777				-1	:EXPECTED FEC.		
7099	036140	104322			45\$:	ERROR	+322	:REPORT RESULT INCORRECT.	
7100	036142	000401				BR	46\$		
7101	036144	104331				ERROR	+331	:(BUT FIC) ST 004 TO	
7102	036146				46\$:			:315 INTO 305	
7103									
7104						:EXP=30 (OCT)	FL=1 FIC=1		
7105	036146	012737	036154	001110		MOV	#250\$, \$LPERR	:SET UP THE LOOP ON ERROR ADDRESS.	:DPM002
7106	036154	004737	037110		250\$:	JSR	PC, STCSUB	:GO EXECUTE THE INSTRUCTION.	
7107	036160	046000	000001	000000	51\$:	.WORD	46000,1,0,0	:ACO OPERAND.	
7108	036170	000200	000001		52\$:	.WORD	200,1	:EXPECTED RESULT.	
7109	036174	177777	177777		53\$:	.WORD	-1,-1	:ANTICIPATED ERRONEOUS RESULT.	
7110	036200	040700			54\$:	40700		:FPS BEFORE EXECUTION.	
7111	036202	040700				40700		:FPS AFTER EXECUTION.	
7112	036204	177777				-1	:ANTICIPATED ERRONEOUS FPS.		
7113	036206	177777				-1	:EXPECTED FEC.		
7114	036210	104322			55\$:	ERROR	+322	:REPORT RESULT INCORRECT.	
7115	036212	000401				BR	56\$		
7116	036214	104323				ERROR	+323	:REPORT FPS INCORRECT.	
7117	036216				56\$:				
7118									
7119						:EXP=27 (OCT)	FL=1 FIC=1		
7120	036216	012737	036224	001110		MOV	#260\$, \$LPERR	:SET UP THE LOOP ON ERROR ADDRESS.	:DPM002
7121	036224	004737	037110		260\$:	JSR	PC, STCSUB	:GO EXECUTE THE INSTRUCTION.	
7122	036230	045600	000001	000000	61\$:	.WORD	45600,1,0,0	:ACO OPERAND.	
7123	036240	000100	000000		62\$:	.WORD	100,0	:EXPECTED RESULT.	
7124	036244	177777	177777		63\$:	.WORD	-1,-1	:ANTICIPATED ERRONEOUS RESULT.	
7125	036250	040707			64\$:	40707		:FPS BEFORE EXECUTION.	
7126	036252	040700				40700		:FPS AFTER EXECUTION.	
7127	036254	177777				-1	:ANTICIPATED ERRONEOUS FPS.		

```

7128 036256 177777          -1          :EXPECTED FEC.
7129 036260 104322          65$:  ERROR  +322          :REPORT RESULT INCORRECT.
7130 036262 000401          BR          66$
7131 036264 104323          ERROR  +323          :REPORT FPS INCORRECT.
7132 036266
7133
7134          :EXP=17 (OCT)  FL=0  FIC=1
7135 036266 012737 036274 001110  MOV #270$, $LPERR :SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
7136 036274 004737 037110 270$: JSR PC, STCSUB :GO EXECUTE THE INSTRUCTION.
7137 036300 043600 000000 000000 71$: .WORD 43600,0,0,0 :ACO OPERAND.
7138 036310 040000 177777 72$: .WORD 40000,-1 :EXPECTED RESULT.
7139 036314 000000 177777 73$: .WORD 0,-1 :ANTICIPATED ERRONEOUS RESULT.
7140 036320 040600 74$: 40600 :FPS BEFORE EXECUTION.
7141 036322 040600 40600 :FPS AFTER EXECUTION.
7142 036324 140604 140604 :ANTICIPATED ERRONEOUS FPS.
7143 036326 177777 -1 :EXPECTED FEC.
7144 036330 104332 75$: ERROR +332 :BAD CONSTANT ST 066
7145 036332 000401 BR 76$
7146 036334 104333 ERROR +333 :REPORT FPS INCORRECT.
7147 036336
7148
7149          :EXP=20 (OCT)  FL=0  FIC=1
7150 036336 012737 036344 001110  MOV #280$, $LPERR :SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
7151 036344 004737 037110 280$: JSR PC, STCSUB :GO EXECUTE THE INSTRUCTION.
7152 036350 044000 000000 000000 81$: .WORD 44000,0,0,0 :ACO OPERAND.
7153 036360 000000 177777 82$: .WORD 0,-1 :EXPECTED RESULT.
7154 036364 177777 177777 83$: .WORD -1,-1 :ANTICIPATED ERRONEOUS RESULT.
7155 036370 040600 84$: 40600 :FPS BEFORE EXECUTION.
7156 036372 140605 140605 :FPS AFTER EXECUTION.
7157 036374 040600 40600 :ANTICIPATED ERRONEOUS FPS.
7158 036376 000006 6 :EXPECTED FEC.
7159 036400 104322 85$: ERROR +322 :REPORT RESULT INCORRECT.
7160 036402 000401 BR 86$
7161 036404 104334 ERROR +334 :BAD CONSTANT ST 066
7162 036406
7163
7164          :EXP=10 (OCT), AC NEGATIVE, FL=0, FIC=1
7165 036406 012737 036414 001110  MOV #290$, $LPERR :SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
7166 036414 004737 037110 290$: JSR PC, STCSUB :GO EXECUTE THE INSTRUCTION.
7167 036420 142000 000000 000000 91$: .WORD 142000,0,0,0 :ACO OPERAND.
7168 036430 177600 177777 92$: .WORD 177600,-1 :EXPECTED RESULT.
7169 036434 000200 000000 93$: .WORD 200,0 :ANTICIPATED ERRONEOUS RESULT.
7170 036440 040600 94$: 40600 :FPS BEFORE EXECUTION.
7171 036442 040610 40610 :FPS AFTER EXECUTION.
7172 036444 040600 40600 :ANTICIPATED ERRONEOUS FPS.
7173 036446 177777 -1 :EXPECTED FEC.
7174 036450 104335 95$: ERROR +335 ;(BUT ENBT) ST 632
7175 036452 000401 BR 96$
7176 036454 104336 ERROR +336 ;(SET FN) ST 473
7177 036456
7178
7179          :EXP=37 (OCT), FL=1, FIC=1, AC NEG.
7180 036456 012737 036464 001110  MOV #300$, $LPERR :SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
7181 036464 004737 037110 300$: JSR PC, STCSUB :GO EXECUTE THE INSTRUCTION.
7182 036470 147600 000000 000000 101$: .WORD 147600,0,0,0 :ACO OPERAND.
7183 036500 140000 000000 102$: .WORD 140000,0 :EXPECTED RESULT.
7184 036504 137777 000000 103$: .WORD 137777,0 :ANTICIPATED ERRONEOUS RESULT.
    
```

```

7185 036510 040700 104$: 40700 ;FPS BEFORE EXECUTION.
7186 036512 040710 ;FPS AFTER EXECUTION.
7187 036514 177777 -1 ;ANTICIPATED ERRONEOUS FPS.
7188 036516 177777 -1 ;EXPECTED FEC.
7189 036520 104337 105$: ERROR +337 ;(BUT COUT) ST 375
7190 036522 000401 BR 106$ ;ST 275 TO 074
7191 036524 104323 ERROR +323 ;INTO 274
7192 036526
7193
7194 ;EXP=37 (OCT), FL=1, FIC=1, AC NEG
7195 036526 012737 036534 001110 MOV #310$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
7196 036534 004737 037110 310$: JSR PC, STCSUB ;GO EXECUTE THE INSTRUCTION.
7197 036540 147600 000000 001000 111$: .WORD 147600,0,1000,0 ;ACO OPERAND.
7198 036550 137777 177777 112$: .WORD 137777,177777 ;EXPECTED RESULT.
7199 036554 140000 177777 113$: .WORD 140000,177777 ;ANTICIPATED ERRONEOUS RESULT.
7200 036560 040707 114$: 40707 ;FPS BEFORE EXECUTION.
7201 036562 040710 ;FPS AFTER EXECUTION.
7202 036564 177777 -1 ;ANTICIPATED ERRONEOUS FPS.
7203 036566 177777 -1 ;EXPECTED FEC.
7204 036570 104340 115$: ERROR +340 ;(BUT COUT) ST 375
7205 036572 000401 BR 116$ ;TO 274 INTO 074
7206 036574 104323 ERROR +323 ;REPORT FPS INCORRECT.
7207 036576
7208
7209 ;EXP=41 (OCT), AC NEG, FL=1, FIC=1
7210 036576 012737 036604 001110 MOV #320$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
7211 036604 004737 037110 320$: JSR PC, STCSUB ;GO EXECUTE THE INSTRUCTION.
7212 036610 150200 000000 000000 121$: .WORD 150200,0,0,0 ;ACO OPERAND.
7213 036620 000000 000000 122$: .WORD 0,0 ;EXPECTED RESULT.
7214 036624 177777 177777 123$: .WORD -1,-1 ;ANTICIPATED ERRONEOUS RESULT.
7215 036630 040700 124$: 40700 ;FPS BEFORE EXECUTION.
7216 036632 140705 ;FPS AFTER EXECUTION.
7217 036634 177777 -1 ;ANTICIPATED ERRONEOUS FPS.
7218 036636 000006 6 ;EXPECTED FEC.
7219 036640 104322 125$: ERROR +322 ;REPORT RESULT INCORRECT.
7220 036642 000401 BR 126$
7221 036644 104341 ERROR +341 ;(BUT EZBT) ST 377
7222 036646
7223 ;EXP=40 (OCT), AC NEG, FL=1, FIC=1
7224 036646 012737 036654 001110 MOV #330$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
7225 036654 004737 037110 330$: JSR PC, STCSUB ;GO EXECUTE THE INSTRUCTION.
7226 036660 150000 000001 000000 131$: .WORD 150000,1,0,0 ;ACO OPERAND.
7227 036670 000000 000000 132$: .WORD 0,0 ;EXPECTED RESULT.
7228 036674 100000 177600 133$: .WORD 100000,-200 ;ANTICIPATED ERRONEOUS RESULT.
7229 036700 040700 134$: 40700 ;FPS BEFORE EXECUTION.
7230 036702 140705 ;FPS AFTER EXECUTION.
7231 036704 040700 ;ANTICIPATED ERRONEOUS FPS.
7232 036706 000006 6 ;EXPECTED FEC.
7233 036710 104342 135$: ERROR +342 ;(BUT COUT) ST 360
7234 036712 000401 BR 136$ ;TO 654 INTO 454
7235 036714 104323 ERROR +323 ;REPORT FPS INCORRECT.
7236 036716
7237
7238 ;EXP=40, AC NEGATIVE, FL=1, FIC=1
7239 036716 012737 036724 001110 MOV #340$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
7240 036724 004737 037110 340$: JSR PC, STCSUB ;GO EXECUTE THE INSTRUCTION.
7241 036730 150001 000000 000000 141$: .WORD 150001,0,0,0 ;ACO OPERAND.
    
```


7242	036740	000000	000000		142\$:	.WORD	0,0		:EXPECTED RESULT.
7243	036744	077400	000000		143\$:	.WORD	77400,0		:ANTICIPATED ERRONEOUS RESULT.
7244	036750	040700			144\$:	40700			:FPS BEFORE EXECUTION.
7245	036752	140705				140705			:FPS AFTER EXECUTION.
7246	036754	177777				-1			:ANTICIPATED ERRONEOUS FPS.
7247	036756	000006				6		:EXPECTED FEC.	
7248	036760	104343			145\$:	ERROR	+343		:REPORT RESULT INCORRECT.
7249	036762	000401				BR	146\$		
7250	036764	104323				ERROR	+323		:REPORT FPS INCORRECT.
7251	036766				146\$:				
7252									
7253					:EXP 40	(OCT), AC MOST NEG LONG INT, FL=1			
7254					:FIC=1				
7255	036766	012737	036774	001110		MOV	#350\$, \$LPERR		:SET UP THE LOOP ON ERROR ADDRESS.
7256	036774	004737	037110		350\$:	JSR	PC, STCSUB		:GO EXECUTE THE INSTRUCTION.
7257	037000	150000	000000	000000	151\$:	.WORD	150000,0,0,0		:ACO OPERAND.
7258	037010	100000	000000		152\$:	.WORD	100000,0		:EXPECTED RESULT.
7259	037014	000000	000000		153\$:	.WORD	0,0		:ANTICIPATED ERRONEOUS RESULT.
7260	037020	040700			154\$:	40700			:FPS BEFORE EXECUTION.
7261	037022	040710				40710			:FPS AFTER EXECUTION.
7262	037024	140705				140705			:ANTICIPATED ERRONEOUS FPS.
7263	037026	177777				-1		:EXPECTED FEC.	
7264	037030	104344			155\$:	ERROR	+344		: (BUT NBIT) ST 654
7265	037032	000401				BR	156\$:OR (BUT COUT) ST 454
7266	037034	104323				ERROR	+323		:REPORT FPS INCORRECT.
7267	037036				156\$:				
7268									
7269					:EXP=20,	AC = MOST NEG INTEGER, FL=0, FIC=1			
7270									
7271	037036	012737	037044	001110		MOV	#360\$, \$LPERR		:SET UP THE LOOP ON ERROR ADDRESS.
7272	037044	004737	037110		360\$:	JSR	PC, STCSUB		:GO EXECUTE THE INSTRUCTION.
7273	037050	144000	000001	000000	161\$:	.WORD	144000,1,0,0		:ACO OPERAND.
7274	037060	100000	177777		162\$:	.WORD	100000,-1		:EXPECTED RESULT.
7275	037064	100000	177400		163\$:	.WORD	100000,177400		:ANTICIPATED ERRONEOUS RESULT.
7276	037070	040600			164\$:	40600			:FPS BEFORE EXECUTION.
7277	037072	040610				40610			:FPS AFTER EXECUTION.
7278	037074	140605				140605			:ANTICIPATED ERRONEOUS FPS.
7279	037076	177777				-1		:EXPECTED FEC.	
7280	037100	104345			165\$:	ERROR	+345		: (BUT FL) ST 633
7281	037102	000401				BR	166\$:TO 655 INTO 654
7282	037104	104323				ERROR	+323		:REPORT FPS INCORRECT.
7283									
7284	037106	000534			166\$:	BR	WWCDONE		

7285
7286
7287
7288
7289
7290
7291
7292
7293
7294
7295
7296
7297
7298
7299
7300
7301
7302
7303
7304
7305
7306
7307
7308
7309
7310
7311
7312
7313
7314
7315
7316
7317 037110 012601
7318 037112 012700 000200
7319 037116 170100
7320 037120 010100
7321 037122 172410
7322 037124 012702 037370
7323 037130 012700 000004
7324 037134 012722 177777
7325 037140 077003
7326 037142 016100 000020
7327 037146 170100
7328 037150 012737 037162 001236
7329 037156 012700 037370
7330 037162 175410
7331
7332 037164 170204
7333 037166 170305
7334 037170 010102
7335 037172 010237 001240
7336 037176 062702 000010
7337 037202 010237 001244
7338 037206 012737 037370 001242
7339 037214 010437 001250
7340 037220 016137 000022 001252
7341 037226 010102

: 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      #1200$,R2    ;INITIALIZE THE OUT PUT BUFFER.
        MOV      #4,R0
1$:     MOV      #-1,(R2)+
        SOB     R0,1$
        MOV      20(R1),R0    ;SET THE FPS.
        LDFPS   R0
        MOV      #2$,STMP2
        MOV      #1200$,R0
2$:     STCDL   ACO,(R0)      ;TEST INSTRUCTION.

        STFPS   R4           ;GET THE FPS.
        STST   R5           ;GET THE FEC.
        MOV      R1,R2
        MOV      R2,STMP3
        ADD     #10,R2
        MOV      R2,STMP5
        MOV      #1200$,STMP4
        MOV      R4,STMP7
        MOV      22(R1),STMP10
        MOV      R1,R2
    
```

```

7342 037230 062702 000010      ADD      #10,R2
7343 037234 012700 037370      MOV      #1200$,R0      ;SEE IF THE RESULT IS CORRECT.
7344 037240 012703 000002      MOV      #2,R3
7345 037244 022022      3$:    CMP      (R0)+,(R2)+
7346 037246 001014      BNE     15$
7347 037250 077303      SOB     R3,3$
7348 037252 016102 000022      MOV      22(R1),R2
7349 037256 020204      CMP      R2,R4      ;SEE IF THE FPS IS CORRECT.
7350 037260 001025      BNE     20$      ;BRANCH IF INCORRECT.
7351 037262 005702      TST     R2
7352 037264 100003      BPL     4$
7353 037266 026105 000026      CMP      26(R1),R5      ;SEE IF THE FEC IS CORRECT.
7354 037272 001027      BNE     25$      ;BRANCH IF INCORRECT.
7355
7356 037274 000161 000036      4$:    JMP      36(R1)      ;RETURN.
7357      ;DATA ERROR.
7358      ;SEE IF THE FAILURE WAS ANTICIPATED.
7359 037300 010102      15$:   MOV      R1,R2
7360 037302 062702 000014      ADD     #14,R2
7361 037306 012700 037370      MOV     #1200$,R0
7362 037312 012703 000002      MOV     #2,R3
7363 037316 022022      16$:   CMP     (R0)+,(R2)+
7364 037320 001003      BNE     17$
7365 037322 077303      SOB     R3,16$
7366 037324 000161 000030      JMP     30(R1)
7367 037330      17$:
7368      ;FAILURE WAS NOT ANTICIPATED SO REPORT INCORRECT RESULT HERE.
7369 037330 104322      18$:   ERROR   +322      ;DATA BAD
7370 037332 000760      BR      4$
7371
7372      ;FPS INCORRECT, SO SEE IF FAILURE WAS ANTICIPATED.
7373 037334 020461 000024      20$:   CMP     R4,24(R1)
7374 037340 001002      BNE     21$
7375 037342 000161 000034      JMP     34(R1)
7376 037346      21$:
7377      ;NOT ANTICIPATED SO REPORT BAD FPS HERE.
7378 037346 104323      22$:   ERROR   +323      ;FPS BAD
7379 037350 000751      BR      4$
7380
7381      ;REPORT INCORRECT FEC.
7382 037352 016137 000026 001256      25$:   MOV     26(R1),$TMP12
7383 037360 010537 001254      MOV     R5,$TMP11
7384 037364 104324      26$:   ERROR   +324
7385 037366 000742      BR      4$
7386
7387      ;DATA BUFFER:
7388 037370 177777 177777 177777      1200$: .WORD  -1,-1,-1,-1
7389
7390 037400      WWCDONE:
       037400 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?).
    
```

7398

```
.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.  
:*****
```

```
037402 000004  
7399  
7400 037404 012737 037412 001110  
7401 037412 004737 037110  
7402 037416 047777 177777 177777  
7403 037426 077777 177600  
7404 037432 077777 177777  
7405 037436 040100  
7406 037440 040100  
7407 037442 177777  
7408 037444 177777  
7409 037446 104346  
7410 037450 000401  
7411 037452 104323  
7412 037454 104412
```

```
TST73: SCOPE  
:EXPONENT=37, FL=1  
MOV #200$,SLPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002  
JSR PC,STCSUB ;GO EXECUTE THE INSTRUCTION.  
1$: .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$  
6$: ERROR +323 ;REPORT FPS INCORRECT.  
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?).
```

7419

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

```
037456 000004
7420
7421 037460 012737 037472 001110
7422 037466 012703 040372
7423 037472 004737 040106
7424 037476 020000 000000 000000
7425 037506 177700
7426 037510 052525
7427 037512 040000
7428 037514 040010
7429 037516 040000
7430 037520 104347
7431 037522 000401
7432 037524 104352
7433 037526 005723
7434 037530 022703 040400
7435 037534 001356
7436
7437
7438 037536 012737 037550 001110
7439 037544 012703 040372
7440 037550 004737 040106
7441 037554 040000 000000 000000
7442 037564 000000
7443 037566 052525
7444 037570 040000
7445 037572 040004
7446 037574 040000
7447 037576 104347
7448 037600 000401
7449 037602 104353
7450
7451 037604 005723
7452 037606 022703 040400
7453 037612 001356
7454
7455
7456
7457
7458 037614 012737 037626 001110
7459 037622 012703 040372
7460 037626 004737 040106
7461 037632 040200 000000 000000
7462 037642 000001
7463 037644 052525
7464 037646 040000
7465 037650 040000
7466 037652 040004
7467 037654 104347

TST74: SCOPE
: EXP = 100 (EXCESS 200)
MOV #200$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
MOV #1200$, R3 ;SETUP DATA TABLE POINTER ;DPM002
200$: JSR PC, 1000$
1$: .WORD 2000,0,0,0 ;AC
2$: -100 ;EXP RES
3$: 52525 ;ERROR EXP.
4$: 40000 ;FPSB
40010 ;FPSA
40000 ;ERROR FPS
5$: ERROR +347 ;BAD EXP
BR 6$
ERROR +352 ;+(BUT ENBT) ST 376
6$: TST (R3)+ ;ADVANCE POINTER ;DPM002
CMP #1200$+6,R3 ;HAVE WE TESTED ALL 3 STATES? ;DPM002
BNE 200$ ;BRANCH IF NOT ;DPM002

: EXP = 200 (EXCESS 200)
MOV #210$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
MOV #1200$, R3 ;SETUP DATA TABLE POINTER ;DPM002
210$: JSR PC, 1000$ ;GO EXECUTE THE INSTRUCTION.
11$: .WORD 4000,0,0,0 ;ACO OPERAND.
12$: 0 ;EXPECTED EXPONENT RESULT.
13$: 52525 ;ANTICIPATED ERRONEOUS RESULT.
14$: 40000 ;FPS BEFORE EXECUTION.
40004 ;FPS AFTER EXECUTION.
40000 ;ANTICIPATED ERRONEOUS FPS.
15$: ERROR +347 ;REPORT RESULT INCORRECT.
BR 16$
ERROR +353 ;(BUT EZBT) ST 071
16$: TST (R3)+ ;TO 072 INT 272
CMP #1200$+6,R3 ;ADVANCE POINTER ;DPM002
BNE 210$ ;HAVE WE TESTED ALL 3 STATES? ;DPM002
;BRANCH IF NOT ;DPM002

: EXP = 201 (EXCESS 200)
MOV #220$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
MOV #1200$, R3 ;SETUP DATA TABLE POINTER ;DPM002
220$: JSR PC, 1000$ ;GO EXECUTE THE INSTRUCTION.
21$: .WORD 40200,0,0,0 ;ACO OPERAND.
22$: 1 ;EXPECTED EXPONENT RESULT.
23$: 52525 ;ANTICIPATED ERRONEOUS RESULT.
24$: 40000 ;FPS BEFORE EXECUTION.
40000 ;FPS AFTER EXECUTION.
40004 ;ANTICIPATED ERRONEOUS FPS.
25$: ERROR +347 ;REPORT RESULT INCORRECT.
```

7468	037656	000401				BR	26\$			
7469	037660	104354				ERROR	+354			
7470	037662	005723			26\$:	TST	(R3)+		;(BUT EZBT) ST 071 TO 272 INTO 072	
7471	037664	022703	040400			CMP	#1200\$+6,R3		:ADVANCE POINTER	:DPM002
7472	037670	001356				BNE	220\$:HAVE WE TESTED ALL 3 STATES?	:DPM002
7473									:BRANCH IF NOT	:DPM002
7474										
7475										
7476										
7477	037672	012737	037704	001110		MOV	#230\$, \$LPERR		:SET UP THE LOOP ON ERROR ADDRESS.	:DPM002
7478	037700	012703	040372			MOV	#1200\$, R3		:SETUP DATA TABLE POINTER	:DPM002
7479	037704	004737	040106		230\$:	JSR	PC, 1000\$:GO EXECUTE THE INSTRUCTION.	
7480	037710	077200	000000	000000	31\$:	.WORD	77200,0,0,0		:ACO OPERAND.	
7481	037720	000175			32\$:	175			:EXPECTED EXPONENT RESULT.	
7482	037722	052525			33\$:	52525			:ANTICIPATED ERRONEOUS RESULT.	
7483	037724	040000			34\$:	40000			:FPS BEFORE EXECUTION.	
7484	037726	040000				40000			:FPS AFTER EXECUTION.	
7485	037730	040010				40010			:ANTICIPATED ERRONEOUS FPS.	
7486	037732	104347			35\$:	ERROR	+347		:REPORT RESULT INCORRECT.	
7487	037734	000401				BR	36\$			
7488	037736	104355				ERROR	+355		;(BUT ENBT) ST 376 TO 471 INTO 071	
7489	037740	005723			36\$:	TST	(R3)+		:ADVANCE POINTER	:DPM002
7490	037742	022703	040400			CMP	#1200\$+6,R3		:HAVE WE TESTED ALL 3 STATES?	:DPM002
7491	037746	001356				BNE	230\$:BRANCH IF NOT	:DPM002
7492										
7493										
7494										
7495										
7496	037750	012737	037762	001110		MOV	#240\$, \$LPERR		:SET UP THE LOOP ON ERROR ADDRESS.	:DPM002
7497	037756	012703	040372			MOV	#1200\$, R3		:SETUP DATA TABLE POINTER	:DPM002
7498	037762	004737	040106		240\$:	JSR	PC, 1000\$:GO EXECUTE THE INSTRUCTION.	
7499	037766	000200	000000	000000	41\$:	.WORD	200,0,0,0		:ACO OPERAND.	
7500	037776	177601			42\$:	-177			:EXPECTED EXPONENT RESULT.	
7501	040000	052525			43\$:	52525			:ANTICIPATED ERRONEOUS RESULT.	
7502	040002	040000			44\$:	40000			:FPS BEFORE EXECUTION.	
7503	040004	040010				40010			:FPS AFTER EXECUTION.	
7504	040006	040000				40000			:ANTICIPATED ERRONEOUS FPS.	
7505	040010	104347			45\$:	ERROR	+347		:REPORT RESULT INCORRECT.	
7506	040012	000401				BR	46\$			
7507	040014	104352				ERROR	+352		:REPORT FPS INCORRECT.	
7508	040016	005723			46\$:	TST	(R3)+		:ADVANCE POINTER	:DPM002
7509	040020	022703	040400			CMP	#1200\$+6,R3		:HAVE WE TESTED ALL 3 STATES?	:DPM002
7510	040024	001356				BNE	240\$:BRANCH IF NOT	:DPM002
7511										
7512										
7513										
7514										
7515	040026	012737	040040	001110		MOV	#250\$, \$LPERR		:SET UP THE LOOP ON ERROR ADDRESS.	:DPM002
7516	040034	012703	040372			MOV	#1200\$, R3		:SETUP DATA TABLE POINTER	:DPM002
7517	040040	004737	040106		250\$:	JSR	PC, 1000\$:GO EXECUTE THE INSTRUCTION.	
7518	040044	033400	000000	000000	51\$:	.WORD	33400,0,0,0		:ACO OPERAND.	
7519	040054	177756			52\$:	-22			:EXPECTED EXPONENT RESULT.	
7520	040056	052525			53\$:	52525			:ANTICIPATED ERRONEOUS RESULT.	
7521	040060	047707			54\$:	47707			:FPS BEFORE EXECUTION.	
7522	040062	047710				47710			:FPS AFTER EXECUTION.	
7523	040064	177777				-1			:ANTICIPATED ERRONEOUS FPS.	
7524	040066	104347			55\$:	ERROR	+347		:REPORT RESULT INCORRECT.	

7525 040070 000401
7526 040072 104350
7527
7528 040074 005723
7529 040076 022703 040400
7530 040102 001356
7531 040104 000543

56\$: BR 56\$
ERROR +350
TST (R3)+
CMP #1200\$+6,R3
BNE 250\$
BR 260\$

:REPORT FPS INCORRECT.
:ADVANCE POINTER
:HAVE WE TESTED ALL 3 STATES?
:BRANCH IF NOT

:DPM002
:DPM002
:DPM002

7532
7533
7534
7535
7536
7537
7538
7539
7540
7541
7542
7543
7544
7545
7546
7547
7548
7549
7550
7551
7552
7553
7554
7555
7556
7557
7558
7559
7560
7561
7562
7563
7564
7565
7566
7567
7568
7569
7570
7571
7572
7573
7574
7575
7576
7577
7578
7579
7580
7581
7582
7583
7584
7585
7586
7587
7588

: THIS SUBROUTINE, 1000\$, IS USED TO SET UP THE OPERANDS, EXECUTE
: THE STEXP INSTRUCTION AND CHECK THE RESULTS. A CALL
: TO IT IS MADE THUS:
:
: JSR PC,1000\$
: ACARG: .WORD X,X,X,X ;AC OPERAND
: RES: .WORD X ;EXPECTED RESULT
: ERRES: .WORD X ;ERROR RESULT
: FPSB: .WORD X ;FPS BEFORE EXECUTION
: FPSA: .WORD X ;FPS AFTER EXECUTION
: 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 STEXP 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 1000\$ RETURNS CONTROL
: TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD 1000\$
: COMPARE IT TO ERROR FPS. IF THIS MATCHES THEN 1000\$ WILL RETURN
: TO THE ERROR CALL AT ERR2, OTHERWISE 1000\$ ITSELF
: REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE
: STEXP 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 1000\$
: WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR1. OTHERWISE THE
: RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND 1000\$ WILL
: REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.

1000\$: MOV (SP)+,R1 ;GET A POINTER TO THE ARGUMENTS.
MOV R1,R2
MOV R2,\$TMP3
ADD #10,R2
MOV (R2)+,\$TMP5
MOV #61,\$TMP2
MOV #123456,1210\$
MOV #76543,1210\$+2
MOV #200,R0
LDFPS R0
MOV R1,R0 ;SET UP THE ACO OPERAND.
LDD (R0),ACO
MOV 16(R1),R0 ;SET THE FPS.
LDFPS R0
MOV (R3),R0 ;SETUP R0 FROM THE TABLE ;DPM002
MOV 6(R3),61\$;MOVE TEST INSTRUCTION TO ITS POSITION ;DPM002
61\$: .WORD 0 ;LOCATION FOR TEST INSTRUCTION. ;DPM002
STFPS R4 ;GET FPS.
CMP 14(R3),R0 ;SEE IF R0 WAS PROPER ;DPM002
BEQ 101\$;BRANCH IF OK ;DPM002
MOV 61\$,\$TMP13 ;MOVE OP CODE OF FAILING INST TO \$TMP13 ;DPM002
MOV 14(R3),\$TMP3 ;MOVE EXPECTED DATA TO \$TMP3 ;DPM002
MOV R0,\$TMP4 ;MOVE RECEIVED DATA TO \$TMP4 ;DPM002
ERROR +371 ;STEXP AUTO INCREMENTED/DECREMENTED ;DPM002
;RO INCORRECTLY ;DPM002

::*****


```

7589
7590
7591 040234 010437 001250
7592 040240 016137 000016 001252
7593 040246 013737 040360 001242
7594 040254 026137 000010 040360
7595 040262 001411
7596 040264 026137 000012 040360
7597 040272 001002
7598 040274 000161 000022
7599
7600
7601 040300
7602 040300 104347
7603 040302 000161 000030
7604
7605 040306 020461 000016
7606 040312 001407
7607 040314 020461 000020
7608 040320 001002
7609 040322 000161 000026
7610
7611
7612 040326
7613 040326 104350
7614 040330 000764
7615
7616
7617 040332 022737 076543 040362
7618 040340 001760
7619 040342 104351
7620 040344 000756
7621
7622 040346 177777 177777 177777
7623 040360 177777 177777 177777
7624 040372 040360 040360 040362
7625 040400 175010
7626 040402 175020
7627 040404 175040
7628 040406 040360 040362 040360
7629
7630 040414
      040414 104412

```

```

: IF THIS FAILURE OCCURED, IS M7095 ECO #10 IN?
:.....
101$: MOV R4,$TMP7
      MOV 16(R1),$TMP10
      MOV 1210$,$TMP4
      CMP 10(R1),1210$ ;WAS RESULT CORRECT?
      BEQ 65$ ;BRANCH IF CORRECT.
      CMP 12(R1),1210$ ;OTHERWISE SEE IF THE FAILURE WAS ANTICIPATED.
      BNE 62$
      JMP 22(R1)

: IF NOT ANTICIPATED REPORT ERROR HERE.
62$:
63$: ERROR +347 ;EXP BAD
64$: JMP 30(R1)

65$: CMP R4,16(R1) ;SEE IF THE FPS IS CORRECT.
      BEQ 70$ ;BRANCH IF CORRECT.
      CMP R4,20(R1) ;SEE IF THE FAILURE WAS ANTICIPATED.
      BNE 66$
      JMP 26(R1)

: FPS ERROR WAS NOT ANTICIPATED SO REPORT ERROR HERE.
66$:
67$: ERROR +350 ;FPS BAD
      BR 64$

: SEE IF MORE THAN ONE WORD WAS WRITTEN IN THE OUTPUT BUFFER.
70$: CMP #76543,1210$+2
      BEQ 64$
71$: ERROR +351 ;DFDL+0 ST 347X
      BR 64$

1210$: .WORD -1,-1,-1,-1,-1
1200$: .WORD -1,-1,-1,-1,-1
      .WORD 1210$,1210$,1210$+2 ;DATA FOR RO LOAD ;DPM002
      STEXP ACO,(R0) ;TEST INSTRUCTION (NOT EXECUTED HERE) ;DPM002
      STEXP ACO,(R0)+ ;TEST INSTRUCTION (NOT EXECUTED HERE) ;DPM002
      STEXP ACO,-(R0) ;TEST INSTRUCTION (NOT EXECUTED HERE) ;DPM002
      .WORD 1210$,1210$+2,1210$ ;EXPECTED RO END VALUE ;DPM002

260$: 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?).

```

7641

```

.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
7642 040416 000004 040426 001110 200$: MOV #200$, $LPERR ;SET UP THE LOOP ON ERROR ADDRESS. ;DPM002
7643 040420 012737 040000 200$: MOV #40000, R0 ;SET FPS. FID=1.
7644 040426 012700 040000 LDFPS R0
7645 040432 170100 210$: .WORD 170003 ;ILLEGAL FPP
7646 040434 170003 ;OP CODE
7647 040436 012700 040612 MOV #220$, R0 ;SET UP THE OUTPUT BUFFER.
7648 040442 012710 177777 MOV #-1, (R0)
7649 040446 012760 000002 MOV #-1, 2(R0)
7650 040454 012737 040462 001236 230$: MOV #230$, $TMP2
7651 040462 170310 230$: STST (R0) ;GET FEC AND
7652 ;FEA
7653 040464 170204 STFPS R4 ;GET FPS.
7654 040466 012700 040612 MOV #220$, R0
7655 040472 011037 001240 MOV (R0), $TMP3
7656 040476 016037 000002 001242 MOV 2(R0), $TMP4
7657 040504 012737 000002 001244 MOV #2, $TMP5
7658 040512 012737 040434 001246 MOV #210$, $TMP6
7659 040520 010437 001250 MOV R4, $TMP7
7660 040524 012737 140000 001252 MOV #140000, $TMP10
7661
7662 040532 022710 000002 CMP #2, (R0) ;SEE IF FEC IS CORRECT.
7663 040536 001010 BNE 240$ ;BRANCH IF INCORRECT.
7664 040540 022760 040434 000002 CMP #210$, 2(R0) ;SEE IF FEA, ADDRESS, IS CORRECT.
7665 040546 001006 BNE 250$ ;BRANCH IF INCORRECT.
7666 040550 022704 170000 CMP #140000, R4 ;SEE IF FPS IS CORRECT.
7667 040554 001013 BNE 260$ ;BRANCH IF INCORRECT.
7668 040556 000422 BR 270$
7669
7670 ;REPORT FEC INCORRECT
7671 040560 104356 240$: ERROR +356 ;STST BAD
7672 040562 000420 BR 270$ ;FECX
7673
7674 ;REPORT FEA INCORRECT
7675 040564 022760 177777 000002 250$: CMP #-1, 2(R0)
7676 040572 001402 BEO 280$
7677 040574 104357 ERROR +357 ;STST BAD FEA
7678 040576 000412 BR 270$
7679 040600 104360 280$: ERROR +360 ;SET FD FL ST 636
7680 040602 000410 BR 270$
7681
7682 ;REPORT FPS INCORRECT
7683 040604 104361 260$: ERROR +361 ;FPS X AFTER ST ST
7684 040606 000406 BR 270$
7685

```

```
7686 ;DATA BUFFER:  
7687 040610 177777 -1  
7688 040612 177777 177777 177777 220S: .WORD -1,-1,-1,-1  
7689 040622 177777 -1  
7690  
7691 040624 270S:
```

040624 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?).

76 THROUGH 104

:MAKE SURE MEMORY MANAGEMENT IS OFF.

:LOAD FPS STATUS.

:LOAD R0 WITH 77406

:LOAD R1 WITH 77400

:MAKE KDPDR0 RESIDENT.

:MAKE KDPDR1 NON-RESIDENT.

:MAKE KDPDR2 NON-RESIDENT.

:MAKE KDPDR3 RESIDENT FOR ADDRESSES 60000-77756.

:MAKE KDPDR4 RESIDENT FOR ADDRESSES 77760-77776.

:MAKE KDPDR7 RESIDENT (I/O PAGE).

:MAP D-PAGE 0 FOR 0-4K.

:MAP D-PAGE 1 FOR 4-8K.

:MAP D-PAGE 2 FOR 8-12K.

:MAP D-PAGE 3 FOR ACCESSING ADDRESSES 60000-77756.

:MAP D-PAGE 4 FOR ACCESSING ADDRESSES 77760-77776.

:MAP D-PAGE 7 FOR I/O PAGE.

:MAKE KIPDR0 RESIDENT.

:MAKE KIPDR1 RESIDENT.

:MAKE KIPDR2 RESIDENT.

:MAKE KIPDR3 NON-RESIDENT FOR USING ADDRESSES 60000-77756.

:MAKE KIPDR4 NON-RESIDENT FOR USING ADDRESSES 77760-77776.

:MAKE KIPDR7 RESIDENT (I/O PAGE).

:MAP I-PAGE 0 FOR 0-4K.

:MAP I-PAGE 1 FOR 4-8K.

:MAP I-PAGE 2 FOR 8-12K.

:MAP I-PAGE 3 FOR ACCESSING ADDRESSES 60000-77756.

:MAP I-PAGE 4 FOR ACCESSING ADDRESSES 77760-77776.

:MAP I-PAGE 7 FOR I/O PAGE.

:MOVE MM TRAP VECTOR TO \$TMP14 FOR TEMP STORAGE.

:SET UP R1.

:SET UP R2.

:SET UP R3.

:SET UP FOR FP TRAPS FOR THIS TEST.

:TURN ON 22-BIT KERNEL D-SPACE.

:SET UP ADDRESS POINTER.

```

7692
7693 040626 005037 177572
7694 040632 170127 040000
7695 040636 012700 077406
7696 040642 012701 077400
7697 040646 010037 172320
7698 040652 010137 172322
7699 040656 010137 172324
7700 040662 010037 172326
7701 040666 010037 172330
7702 040672 010037 172336
7703
7704 040676 005037 172360
7705 040702 012737 000200 172362
7706 040710 012737 000400 172364
7707 040716 012737 000600 172366
7708 040724 012737 000600 172370
7709 040732 012737 177600 172376
7710
7711 040740 010037 172300
7712 040744 010037 172302
7713 040750 010037 172304
7714 040754 010137 172306
7715 040760 010137 172310
7716 040764 010037 172316
7717
7718 040770 005037 172340
7719 040774 012737 000200 172342
7720 041002 012737 000400 172344
7721 041010 012737 000600 172346
7722 041016 012737 000600 172350
7723 041024 012737 177600 172356
7724
7725 041032 013737 000250 001262
7726 041040 012701 117760
7727 041044 012702 117770
7728 041050 012703 117772
7729 041054 012737 041714 000250
7730 041062 012737 000340 000252
7731 041070 012737 000024 172516
7732 041076 012737 117760 077770
    
```

```

.SBTTL SETUP FOR TESTS
CLR MMR0
LDFPS #40000
MOV #77406,R0
MOV #77400,R1
MOV R0,KDPDR0
MOV R1,KDPDR1
MOV R1,KDPDR2
MOV R0,KDPDR3
MOV R0,KDPDR4
MOV R0,KDPDR7
CLR KDPAR0
MOV #200,KDPAR1
MOV #400,KDPAR2
MOV #600,KDPAR3
MOV #600,KDPAR4
MOV #177600,KDPAR7
MOV R0,KIPDR0
MOV R0,KIPDR1
MOV R0,KIPDR2
MOV R1,KIPDR3
MOV R1,KIPDR4
MOV R0,KIPDR7
CLR KIPAR0
MOV #200,KIPAR1
MOV #400,KIPAR2
MOV #600,KIPAR3
MOV #600,KIPAR4
MOV #177600,KIPAR7
MOV MMVECT,$TMP14
MOV #DATA,R1
MOV #DATA+10,R2
MOV #DATA+12,R3
MOV #TRAPV,MMVECT
MOV #340,MMVECT+2
MOV #24,MMR3
MOV #DATA,77770
    
```

```

: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?).
76 THROUGH 104
:MAKE SURE MEMORY MANAGEMENT IS OFF.
:LOAD FPS STATUS.
:LOAD R0 WITH 77406
:LOAD R1 WITH 77400
:MAKE KDPDR0 RESIDENT.
:MAKE KDPDR1 NON-RESIDENT.
:MAKE KDPDR2 NON-RESIDENT.
:MAKE KDPDR3 RESIDENT FOR ADDRESSES 60000-77756.
:MAKE KDPDR4 RESIDENT FOR ADDRESSES 77760-77776.
:MAKE KDPDR7 RESIDENT (I/O PAGE).
:MAP D-PAGE 0 FOR 0-4K.
:MAP D-PAGE 1 FOR 4-8K.
:MAP D-PAGE 2 FOR 8-12K.
:MAP D-PAGE 3 FOR ACCESSING ADDRESSES 60000-77756.
:MAP D-PAGE 4 FOR ACCESSING ADDRESSES 77760-77776.
:MAP D-PAGE 7 FOR I/O PAGE.
:MAKE KIPDR0 RESIDENT.
:MAKE KIPDR1 RESIDENT.
:MAKE KIPDR2 RESIDENT.
:MAKE KIPDR3 NON-RESIDENT FOR USING ADDRESSES 60000-77756.
:MAKE KIPDR4 NON-RESIDENT FOR USING ADDRESSES 77760-77776.
:MAKE KIPDR7 RESIDENT (I/O PAGE).
:MAP I-PAGE 0 FOR 0-4K.
:MAP I-PAGE 1 FOR 4-8K.
:MAP I-PAGE 2 FOR 8-12K.
:MAP I-PAGE 3 FOR ACCESSING ADDRESSES 60000-77756.
:MAP I-PAGE 4 FOR ACCESSING ADDRESSES 77760-77776.
:MAP I-PAGE 7 FOR I/O PAGE.
:MOVE MM TRAP VECTOR TO $TMP14 FOR TEMP STORAGE.
:SET UP R1.
:SET UP R2.
:SET UP R3.
:SET UP FOR FP TRAPS FOR THIS TEST.
:TURN ON 22-BIT KERNEL D-SPACE.
:SET UP ADDRESS POINTER.
    
```

7746

.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.
 :*

 TST76: SCOPE

7747	041104	000004			MOV	#200\$,\$LPERR	;SET UP THE LOOP ON ERROR ADDRESS.	;DPM002
7748	041106	012737	041114	001110			;* THIS INSTRUCTION WILL TEST FOR A WORST-CASE HARDWARE PROBLEM.	
7749	041114	170000			CFCC		;* TEST INSTRUCTION WHICH SHOULD ALWAYS INVOKE D-SPACE.	
7750	041116	010100			MOV	R1,R0	;SETTING UP R0.	
7751	041120	004737	042430		JSR	PC,SETERL	;GO SET ERROR LOOP TO ADRS OF NEXT INST	;DPM002
7752	041124	170410			CLRF	(R0)	;TESTING BLOCKS 27-K AND 27-R.	
7753	041126	004737	042430		JSR	PC,SETERL	;GO SET ERROR LOOP TO ADRS OF NEXT INST	;DPM002
7754	041132	177010			LDCIF	(R0),ACO	;TESTING BLOCKS 28-F AND 28-P.	
7755	041134	004737	042430		JSR	PC,SETERL	;GO SET ERROR LOOP TO ADRS OF NEXT INST	;DPM002
7756	041140	172410			LDF	(R0),ACO	;TESTING BLOCKS 4-J, 4-X, 4-Z AND 4-BB.	
7757	041142	004737	042430		JSR	PC,SETERL	;GO SET ERROR LOOP TO ADRS OF NEXT INST	;DPM002
7758	041146	170310			STST	(R0)	;TESTING BLOCKS 33-E AND 33-P.	
7759	041150	004737	042430		JSR	PC,SETERL	;GO SET ERROR LOOP TO ADRS OF NEXT INST	;DPM002
7760	041154	005037	177572		CLR	MMRO	;TURN OFF MEMORY MANAGEMENT.	

7761

.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.

*

7762	041160	000004		TST77: SCOPE		
	041162	010100		MOV R1,R0		;SETTING UP R0.
7763	041164	004737	04 1430	JSR PC,SETERL		;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
7764	041170	170520		TSTF (R0)+		;TESTING BLOCK 21-AA.
7765	041172	004737	042430	JSR PC,SETERL		;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
7766	041176	170527	001000	TSTF #1000		;TESTING BLOCK 21-AA.
7767						
7768	041202	010100		MOV R1,R0		;CORRECTING R0.
7769	041204	004737	042430	JSR PC,SETERL		;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
7770	041210	170420		CLRF (R0)+		;TESTING BLOCKS 27-K AND 27-R.
7771						;**NOTE**: THE LOCATION AFTER THE CLRF, AND STST MODE 2 REG 7 INSTRUCTIONS
7772						;**WILL* BE CHANGED ON SUBSEQUENT PASSES, BUT IS **NOT** INCORRECT. THE
7773						;ACTUAL CONTENTS OF THOSE LOCATIONS IS IMMATERIAL, AS THIS TEST INSURES
7774						;THAT THE INSTRUCTION DOES EXECUTE WITHOUT ACCESSING THAT LOCATION AS
7775						;A D-SPACE ACCESS.
7776	041212	170427	001000	CLRF #1000		;TESTING BLOCKS 27-K AND 27-R.
7777						
7778	041216	010100		MOV R1,R0		;CORRECTING R0.
7779	041220	004737	042430	JSR PC,SETERL		;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
7780	041224	177020		LDCIF (R0)+,ACO		;TESTING BLOCKS 28-F AND 28-P.
7781	041226	004737	042430	JSR PC,SETERL		;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
7782	041232	177027	001000	LDCIF #1000,ACO		;TESTING BLOCKS 28-F AND 28-P.
7783						
7784	041236	010100		MOV R1,R0		;CORRECTING R0.
7785	041240	004737	042430	JSR PC,SETERL		;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
7786	041244	172420		LDF (R0)+,ACO		;TESTING BLOCKS 4-NN, 4-X, 4-Z AND 4-BB.
7787	041246	004737	042430	JSR PC,SETERL		;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
7788	041252	172427	042572	LDF #1000,ACO		;TESTING BLOCKS 4-NN, 4-X, 4-Z AND 4-BB.
7789						
7790	041256	010100		MOV R1,R0		;CORRECTING R0.
7791	041260	004737	042430	JSR PC,SETERL		;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
7792	041264	170320		STST (R0)+		;TESTING BLOCKS 33-J AND 33-P.
7793	041266	004737	042430	JSR PC,SETERL		;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
7794	041272	170327	001000	STST #1000		;TESTING BLOCKS 33-J AND 33-P.
7795	041276	005037	177572	CLR MMRO		;TURN OFF MEMORY MANAGEMENT.

7796

.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.

*

TST100: SCOPE

7797	041302	000004		MOV	R2,R0	;SETTING UP R0.
7798	041304	010200		JSR	PC,SETERL	;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
7799	041306	004737	042430	TSTF	@(R0)+	;TESTING BLOCK 21-N.
7800	041312	170530		JSR	PC,SETERL	;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
7801	041314	004737	042430	TSTF	DATA	;TESTING BLOCK 21-N.
7802	041320	170537	117760			
7803	041324	010200		MOV	R2,R0	;CORRECTING R0.
7804	041326	004737	042430	JSR	PC,SETERL	;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
7805	041332	170430		CLRF	@(R0)+	;TESTING BLOCKS 27-U, 27-T AND 27-R.
7806	041334	004737	042430	JSR	PC,SETERL	;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
7807	041340	170437	117760	CLRF	DATA	;TESTING BLOCKS 27-U, 27-T AND 27-R.
7808						
7809	041344	010200		MOV	R2,R0	;CORRECTING R0.
7810	041346	004737	042430	JSR	PC,SETERL	;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
7811	041352	177030		LDCIF	@(R0)+,ACO	;TESTING BLOCKS 28-L, 28-N AND 28-P.
7812	041354	004737	042430	JSR	PC,SETERL	;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
7813	041360	177037	117760	LDCIF	DATA,ACO	;TESTING BLOCKS 28-L, 28-N AND 28-P.
7814						
7815	041364	010200		MOV	R2,R0	;CORRECTING R0.
7816	041366	004737	042430	JSR	PC,SETERL	;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
7817	041372	172430		LDF	@(R0)+,ACO	;TESTING BLOCKS 4-R, 4-T, 4-X, 4-Z AND 4-BB.
7818	041374	004737	042430	JSR	PC,SETERL	;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
7819	041400	172437	117760	LDF	DATA,ACO	;TESTING BLOCKS 4-R, 4-T, 4-X, 4-Z AND 4-BB.
7820						
7821	041404	010200		MOV	R2,R0	;CORRECTING R0.
7822	041406	004737	042430	JSR	PC,SETERL	;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
7823	041412	170330		STST	@(R0)+	;TESTING BLOCKS 33-L, 33-N AND 33-P.
7824	041414	004737	042430	JSR	PC,SETERL	;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
7825	041420	170337	117760	STST	DATA	;TESTING BLOCKS 33-L, 33-N AND 33-P.
7826	041424	005037	177572	CLR	MMRO	;TURN OFF MEMORY MANAGEMENT.

7827

```
.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 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.
:*
:*****
```

```
041430 000004
7828 041432 010200
7829 041434 004737 042430
7830 041440 172440
7831 041442 005037 177572
```

```
TST101: SCOPE
MOV      R2,R0      ;SETTING UP R0.
JSR      PC,SETERL  ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
LDF      -(R0),ACO  ;TESTING BLOCKS 4-J, 4-X, 4-Z AND 4-BB.
CLR      MMRO       ;TURN OFF MEMORY MANAGEMENT.
```


7832

```
.SBYTL 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.  
:  
:*****
```

```
041446 000004  
7833 041450 010300  
7834 041452 004737 042430  
7835 041456 170550  
7836  
7837 041460 010300  
7838 041462 004737 042430  
7839 041466 170450  
7840  
7841 041470 010300  
7842 041472 004737 042430  
7843 041476 177050  
7844  
7845 041500 010300  
7846 041502 004737 042430  
7847 041506 172450  
7848  
7849 041510 010300  
7850 041512 004737 042430  
7851 041516 170350  
7852 041520 005037 177572
```

```
TST102: SCOPE  
MOV R3,R0 ;SETTING UP R0.  
JSR PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002  
TSTF @-(R0) ;TESTING BLOCK 21-U.  
  
MOV R3,R0 ;CORRECTING R0.  
JSR PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002  
CLRF @-(R0) ;TESTING BLOCKS 27-X, 27-T AND 27-R.  
  
MOV R3,R0 ;CORRECTING R0.  
JSR PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002  
LDCIF @-(R0),ACO ;TESTING BLOCKS 28-S, 28-N AND 28-P.  
  
MOV R3,R0 ;CORRECTING R0.  
JSR PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002  
LDF @-(R0),ACO ;TESTING BLOCKS 4-U, 4-T, 4-X, 4-Z AND 4-BB.  
  
MOV R3,R0 ;CORRECTING R0.  
JSR PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002  
STST @-(R0) ;TESTING BLOCKS 33-S, 33-N ADN 33-P.  
CLR MMRO ;TURN OFF MEMORY MANAGEMENT.
```

7853

```
.SBTTL TEST # 103 - ENABLE D-SPACE & SEE I-SPACE FORCED, MODE 6  
:*****  
:*TEST 103      NABLE 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.  
:*  
:*****
```

```
041524 000004  
7854  
7855 041526 004767 000676  
7856 041532 170567 056222  
7857 041536 004767 000666  
7858 041542 170467 056212  
7859 041546 004767 000656  
7860 041552 177067 056202  
7861 041556 004767 000646  
7862 041562 172467 056172  
7863 041566 004767 000636  
7864 041572 170367 056162  
7865 041576 005067 135770  
7866
```

```
TST103: SCOPE  
      .DSABL  AMA      ;DISABLE MODE 6 TO MODE 3 CONVERSION  
      JSR    PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002  
      TSTF   DATA     ;TESTING BLOCK 21-0.  
      JSR    PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002  
      CLRF   DATA     ;TESTING BLOCKS 27-DD, 27-T AND 27-R.  
      JSR    PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002  
      LDCIF  DATA,ACO  ;TESTING BLOCKS 28-T, 28-N ADN 28-P.  
      JSR    PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002  
      LDF    DATA,ACO  ;TESTING BLOCKS 4-DD, 4-T, 4-X, 4-Z AND 4-BB.  
      JSR    PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002  
      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
```

7867

.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 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 GRUPS 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.

* *

041602 000004
 7868 041604 010200
 7869 041606 004737 042430
 7870 041612 170470 000000
 7871 041616 004737 042430
 7872 041622 170477 056142
 7873 041626 004737 042430
 7874 041632 177070 000000
 7875 041636 004737 042430
 7876 041642 177077 056122
 7877 041646 004737 042430
 7878 041652 172470 000000
 7879 041656 004737 042430
 7880 041662 172477 056102
 7881 041666 004737 042430
 7882 041672 170370 000000
 7883 041676 004737 042430
 7884 041702 170377 056062
 7885 041706 005037 177572
 7886
 7887 041712 000431

TST104: SCOPE

```

MOV R2,R0 ;SETTING UP R0.
JSR PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
CLRF @0(R0) ;TESTING BLOCKS 27-GG, 27-JJ, 27-T AND 27-R.
JSR PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
CLRF @DATA+10 ;TESTING BLOCKS 27-GG, 27-JJ, 27-T AND 27-R.
JSR PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
LDCIF @0(R0),ACO ;TESTING BLOCKS 28-W, 28-Z, 28-N AND 28-P.
JSR PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
LV IF @DATA+10,ACO ;TESTING BLOCKS 28-W, 28-Z, 28-N AND 28-P.
JSR PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
LDF @0(R0),ACO ;TESTING BLOCKS 4-GG, 4-JJ, 4-T, 4-X 4-Z AND 4-BB.
JSR PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
LDF @DATA+10,ACO ;TESTING BLOCKS 4-GG, 4-JJ, 4-T, 4-X 4-Z AND 4-BB.
JSR PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
STST @0(R0) ;TESTING BLOCKS 33-W, 33-Z, 33-N AN 33-P.
JSR PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
STST @DATA+10 ;TESTING BLOCKS 33-W, 33-Z, 33-N AN 33-P.
CLR MMRO ;TURN OFF MEMORY MANAGEMENT.

BR ENDTST ;BRANCH TO END OF TEST ROUTINE.
    
```

7888					.SBTTL	TRAP HANDLER FOR UNEXPECTED MEMORY MANAGEMENT ABORTS	
7889	041714	042737	000001	177572	TRAPV: BIC	#1,MMRO	:TURN OFF MEMORY MANAGEMENT.
7890	041722	013737	177572	001240	MOV	MMRO,\$TMP3	:TRANSFER MMRO TO \$TMP3 FOR ERROR PRINTING.
7891	041730	005237	001240		INC	\$TMP3	:REPLACE BIT CLEARED TURNING OFF MEMORY MANAGEMENT.
7892	041734	013737	177576	001236	MOV	MMR2,\$TMP2	:MOVE THE TRAP INSTRUCTION ADDRESS TO \$TMP13.
7893	041742	005037	177572		CLR	MMRO	:CLEAR ERROR BITS.
7894	041746	012637	001266		MOV	(SP)+,\$TMP16	:POP STACK AND SAVE 1ST CONTENTS.
7895	041752	012637	001270		MOV	(SP)+,\$TMP17	:POP STACK AGAIN AND SAVE 2ND CONTENTS.
7896	041756	104362			ERROR	+362	:FPP TRAP/ABORT ERROR CALL.
7897	041760	013746	001270		MOV	\$TMP17,-(SP)	:PUSH 2ND SAVED CONTENTS BACK ON STACK.
7898	041764	013746	001266		MOV	\$TMP16,-(SP)	:PUSH 1ST SAVED CONTENTS BACK ON STACK.
7899	041770	005237	177572		INC	MMRO	:TURN ON MEMORY MANAGEMENT.
7900	041774	000002			RTI		:RETURN FROM INTERRUPT.

7901	041776	005037	177572		ENDTST: CLR	MMRO	:TURN OFF MEMORY MANAGEMENT.
7902	042002	013737	001262	000250	MOV	\$TMP14,MMVECT	:RESTORE MMVECT TO ITS ORIGINAL CONTENTS.
7903	042010				IDONE:		
	042010	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. :LOAD FLOATING POINT STATUS. :CLEAR THE TEMPORARY LOCATION. :CLEAR UPPER BYTE - ALTERNATING BITS IN LOWER BYTE. :MOVE ALTERNATING BITS TO 2ND WORD. :MOVE ALTERNATING BITS TO 3RD WORD. :MOVE ALTERNATING BITS TO 4TH WORD. :LOAD AC0. :LOAD AC1. :LOAD AC2. :LOAD AC3. :MOVE ADDRESS OF STORE TO R0. :MOVE LOOP COUNTER (CLEARING 30 WORDS) TO R1. :CLEAR THE WORD. :SUBTRACT 1 FROM R1 AND BRANCH IF NOT 0. :STORE AC0. :STORE AC1. :STORE AC2. :STORE AC3.
7904	042012	005037	177572		CLR	MMRO	
7905	042016	170127	040000		LDFPS	#40000	
7906	042022	005037	001272		CLR	\$TMP20	
7907	042026	012737	000252	045232	MOV	#252,STORE	
7908	042034	012737	125252	045234	MOV	#125252,STORE+2	
7909	042042	012737	125252	045236	MOV	#125252,STORE+4	
7910	042050	012737	125252	045240	MOV	#125252,STORE+6	
7911	042056	172437	045232		LDF	STORE,AC0	
7912	042062	172537	045232		LDF	STORE,AC1	
7913	042066	172637	045232		LDF	STORE,AC2	
7914	042072	172737	045232		LDF	STORE,AC3	
7915	042076	012700	045232		MOV	#STORE,R0	
7916	042102	012701	000030		MOV	#30,R1	
7917	042106	005020			CLR	(R0)+	
7918	042110	077102			SOB	R1,1\$	
7919	042112	174037	045232		STF	AC0,STORE	
7920	042116	174137	045242		STF	AC1,STORE+10	
7921	042122	174237	045252		STF	AC2,STORE+20	
7922	042126	174337	045262		STF	AC3,STORE+30	
7923							
7924	042132	012737	077406	172320	MOV	#77406,KDPDR0	:MAKE KDPDR0 RESIDENT.
7925	042140	012737	077406	172322	MOV	#77406,KDPDR1	:MAKE KDPDR1 RESIDENT.
7926	042146	012737	077400	172324	MOV	#77400,KDPDR2	:MAKE KDPDR2 NON-RESIDENT.
7927	042154	012737	077406	172326	MOV	#77406,KDPDR3	:MAKE KDPDR3 RESIDENT.
7928	042162	012737	077406	172336	MOV	#77406,KDPDR7	:MAKE KDPDR7 RESIDENT.
7929							
7930	042170	012737	077406	172300	MOV	#77406,KIPDR0	:MAKE KIPDR0 RESIDENT.
7931	042176	012737	077406	172302	MOV	#77406,KIPDR1	:MAKE KIPDR1 RESIDENT.
7932	042204	012737	077406	172304	MOV	#77406,KIPDR2	:MAKE KIPDR2 RESIDENT.
7933	042212	012737	077406	172306	MOV	#77406,KIPDR3	:MAKE KIPDR3 RESIDENT.
7934	042220	012737	077406	172316	MOV	#77406,KIPDR7	:MAKE KIPDR7 RESIDENT.
7935							
7936	042226	005037	172360		CLR	KDPAR0	:MAP D-PAGE 0 FOR 0-4K.
7937	042232	012737	000200	172362	MOV	#200,KDPAR1	:MAP D-PAGE 1 FOR 4-8K.
7938	042240	012737	000400	172364	MOV	#400,KDPAR2	:MAP D-PAGE 2 FOR 8-12K.
7939	042246	012737	000600	172366	MOV	#600,KDPAR3	:MAP D-PAGE 3 FOR 12-16K.
7940	042254	012737	177600	172376	MOV	#177600,KDPAR7	:MAP D-PAGE 7 FOR I/O PAGE.
7941							
7942	042262	005037	172340		CLR	KIPAR0	:MAP I-PAGE 0 FOR 0-4K.
7943	042266	012737	000200	172342	MOV	#200,KIPAR1	:MAP I-PAGE 1 FOR 4-8K.
7944	042274	012737	000400	172344	MOV	#400,KIPAR2	:MAP I-PAGE 2 FOR 8-12K.
7945	042302	012737	000600	172346	MOV	#600,KIPAR3	:MAP I-PAGE 3 FOR 12-16K.
7946	042310	012737	177600	172356	MOV	#177600,KIPAR7	:MAP I-PAGE 7 FOR I/O PAGE.
7947							
7948	042316	012705	045312		MOV	#BYTABL,R5	:SET UP BYTE TABLE POINTER R5.
7949	042322	013737	000250	001262	MOV	MMVECT,\$TMP14	:TEMPORARILY STORE THE MMVECT VALUE.
7950	042330	012737	044054	000250	MOV	#TRPV,MMVECT1	:SET UP FOR FP TRAPS FOR THIS TEST.
7951	042336	012737	000340	000252	MOV	#340,MMVECT+2	
7952	042344	013737	000020	042426	MOV	IOTRAP,SAVIOT	:STORE THE IOTRAP VALUE IN SAVIOT

```

7953 042352 012737 000340 000022      MOV      #340,10TRAP+2
7954
7955 042360 012737 000024 172516      MOV      #24,MMR3      ;TURN ON 22-BIT KERNEL D-SPACE.
7956 042366 012737 045216 045226      MOV      #NODAT,NODAT+10 ;SET UP ADDRESS POINTER.
7957 042374 012700 045216      MOV      #NODAT,R0      ;SET UP R0.
7958 042400 012702 045226      MOV      #NODAT+10,R2   ;SET UP R2.
7959 042404 012703 045230      MOV      #NODAT+12,R3   ;SET UP R3.
7960 042410 010037 045272      MOV      R0,STORE+40    ;STORE R0.
7961 042414 010237 045274      MOV      R2,STORE+42    ;STORE R2.
7962 042420 010337 045276      MOV      R3,STORE+44    ;STORE R3.
7963 042424 000410      BR       TST105         ;BRANCH OVER LOCATION SAVIOT AND SETERL SUBROUTINE
7964
7965 042426 000000      SAVIOT: .WORD 0         ;LOCATION TO SAVE THE SCOPE VECTOR ADDRESS
7966
7967 042430 005037 177572      SETERL: CLR      MMRO      ;TURN OFF MEMORY MANAGEMENT      ;DPM002
7968 042434 011637 001110      MOV      (SP),$LPERR    ;MOVE RETURN ADDRESS TO $LPERR   ;DPM002
7969 042440 005237 177572      INC      MMRO          ;TURN MEMORY MANAGEMENT BACK ON  ;DPM002
7970 042444 000207      RTS       PC           ;RETJRN                          ;DPM002
  
```

7983

```
.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-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.
:*
:*****
TST105: SCOPE
```

042446 000004

```
7984
7985
7986
7987 042450 012737 044006 000020
7988 042456 004737 042430
7989 042462 170501
7990 042464 004737 042430
7991 042470 170401
7992 042472 004737 042430
7993 042476 177001
7994 042500 004737 042430
7995 042504 172401
7996 042506 004737 042430
7997 042512 170301
7998 042514 005037 177572
7999 042520 172437 045232
8000 042524 172537 045232
8001 042530 010046
8002 042532 010246
8003 042534 013737 042426 000020
```

; * 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.
JSR PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
TSTF R1 ;FDST-NOTCLR PAGE 21.
JSR PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
CLR R1 ;FDST MODES PAGE 27.
JSR PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
LDCIF R1,ACO ;SOURCE MODES PAGE 28.
JSR PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
LDF R1,ACO ;FSRC MODES PAGE 4.
JSR PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
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
```

8004

```
.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.
:*
:*****
```

```
042542 000004
8005 042544 012602
8006 042546 012600
8007 042550 012737 044006 000020
8008 042556 010001
8009 042560 010004
8010 042562 004737 042430
8011 042566 170511
8012 042570 000004
8013 042572 004737 042430
8014 042576 170411
8015 042600 000004
8016 042602 004737 042430
8017 042606 177011
8018 042610 000004
8019 042612 004737 042430
8020 042616 172411
8021 042620 000004
8022 042622 004737 042430
8023 042626 170311
8024 042630 000004
8025 042632 005037 177572
8026 042636 010046
8027 042640 010246
8028 042642 013737 042426 000020
```

```
TST106: SCOPE
MOV (SP)+,R2 ;RESTORE R2 FOR THIS TEST
MOV (SP)+,R0 ;RESTORE R0 FOR THIS TEST
MOV #FALTRP,IOTRAP;SET UP FOR FAILURE OF TRAPS FOR THIS TEST.
MODE1: MOV R0,R1 ;SET UP R1.
MOV R0,R4 ;MOVE 'START' VALUE INTO R4.
JSR PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
TSTF (R1) ;FDST-NOTCLR PAGE 21.
IOT ;FORCE A TRAP.
JSR PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
CLRF (R1) ;FDST MODES PAGE 27.
IOT ;FORCE A TRAP.
JSR PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
LDCIF (R1),ACO ;SOURCE MODES PAGE 28.
IOT ;FORCE A TRAP.
JSR PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
LDF (R1),ACO ;FSRC MODES PAGE 4.
IOT ;FORCE A TRAP.
JSR PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
STST (R1) ;DEST MODES PAGE 33.
IOT ;FORCE A TRAP.
CLR MMRO ;TURN OFF MEMORY MANAGEMENT.
MOV R0,-(SP) ;SAVE R0 FOR NEXT TEST
MOV R2,-(SP) ;SAVE R2 FOR NEXT TEST
MOV SAVIOT,IOTRAP ;RESTORE SCOPE TRAP VECTOR
```


8029

```

.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.
*
*****
    
```

```

      042650 000004
8030 042652 012737 044006 000020      MOV    #FALTRP,IOTRAP;SET UP FOR FAILURE OF TRAPS FOR THIS TEST.
8031 042660 012602                    MOV    (SP)+,R2      ;RESTORE R2 FOR THIS TEST
8032 042662 012600                    MOV    (CP)+,R0      ;RESTORE R0 FOR THIS TEST
8033 042664 010001                    MOV    R0,R1        ;CORRECT R1
8034 042666 004737 042430              JSR    PC,SETERL    ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
8035 042672 170521                    TSTF   (R1)+        ;FDST-NOTCLR PAGE 21.
8036 042674 000004 LABEL1: IOT          ;FORCE A TRAP.
8037
8038 042676 004737 042430              JSR    PC,SETERL    ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
8039 042702 010001                    MOV    R0,R1        ;CORRECT R1.
8040 042704 170421                    CLRF   (R1)+        ;FDST MODES PAGE 27.
8041 042706 000004 IOT          ;FORCE A TRAP.
8042
8043 042710 004737 042430              JSR    PC,SETERL    ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
8044 042714 010001                    MOV    R0,R1        ;CORRECT R1.
8045 042716 177021                    LDCIF (R1)+,ACO     ;SOURCE MODES PAGE 28.
8046 042720 000004 IOT          ;FORCE A TRAP.
8047
8048 042722 004737 042430              JSR    PC,SETERL    ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
8049 042726 010001                    MOV    R0,R1        ;CORRECT R1.
8050 042730 172421                    LDF   (R1)+,ACO     ;FSRC MODES PAGE 4.
8051 042732 000004 IOT          ;FORCE A TRAP.
8052
8053 042734 004737 042430              JSR    PC,SETERL    ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
8054 042740 010001                    MOV    R0,R1        ;CORRECT R1.
8055 042742 170321                    STST  (R1)+        ;DEST MODES PAGE 33.
8056 042744 000004 IOT          ;FORCE A TRAP.
8057 042746 005037 177572              CLR    MMRO         ;TURN OFF MEMORY MANAGEMENT.
8058 042752 013737 042426 000020      MOV    SAVIOT,IOTRAP;RESTORE SCOPE TRAP VECTOR
8059 042760 010046                    MOV    R0,-(SP)     ;SAVE R0 FOR NEXT TEST
8060 042762 010246                    MOV    R2,-(SP)     ;SAVE R2 FOR NEXT TEST
    
```

8061

```

.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
MOV      #FALTRP,IOTRAP;SET UP FOR FAILURE OF TRAPS FOR THIS TEST.
MOV      (SP)+,R2      ;RESTORE R2 FOR THIS TEST
MOV      (SP)+,R0      ;RESTORE R0 FOR THIS TEST
JSR      PC,SETERL     ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
MOV      R2,R1         ;SET UP R1 FOR MODE 3.
MOV      R2,R4         ;MOVE 'START' VALUE INTO R4.
TSTF     @ (R1)+       ;FDST-NOTCLR PAGE 21.
IOT      ;FORCE A TRAP.
JSR      PC,SETERL     ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
TSTF     NODAT
IOT      ;FORCE A TRAP.
JSR      PC,SETERL     ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
MOV      R2,R1         ;CORRECT R1.
CLRF     @ (R1)+       ;FDST MODES PAGE 27.
IOT      ;FORCE A TRAP.
JSR      PC,SETERL     ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
CLRF     NODAT
IOT      ;FORCE A TRAP.
JSR      PC,SETERL     ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
MOV      R2,R1         ;CORRECT R1.
LDCIF   @ (R1)+,ACO    ;SOURCE MODES PAGE 28.
IOT      ;FORCE A TRAP.
JSR      PC,SETERL     ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
LDCIF   NODAT,ACO
IOT      ;FORCE A TRAP.
JSR      PC,SETERL     ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
MOV      R2,R1         ;CORRECT R1.
LDF     @ (R1)+,ACO    ;FSRC MODES PAGE 4.
IOT      ;FORCE A TRAP.
JSR      PC,SETERL     ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
LDF     NODAT,ACO
IOT      ;FORCE A TRAP.
JSR      PC,SETERL     ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
MOV      R2,R1         ;CORRECT R1.
STST    @ (R1)+       ;DEST MODES PAGE 33.
IOT      ;FORCE A TRAP.
JSR      PC,SETERL     ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
STST    NODAT
    
```

8062	042764	000004	044006	000020
8063	042774	012737		
8064	042776	012602		
8065	043000	004737	042430	
8066	043004	010201		
8067	043006	010204		
8068	043010	170531		
8069	043012	000004		
8070	043014	004737	042430	
8071	043020	170537	045216	
8072	043024	000004		
8073				
8074	043026	004737	042430	
8075	043032	010201		
8076	043034	170431		
8077	043036	000004		
8078	043040	004737	042430	
8079	043044	170437	045216	
8080	043050	000004		
8081				
8082	043052	004737	042430	
8083	043056	010201		
8084	043060	177031		
8085	043062	000004		
8086	043064	004737	042430	
8087	043070	177037	045216	
8088	043074	000004		
8089				
8090	043076	004737	042430	
8091	043102	010201		
8092	043104	172431		
8093	043106	000004		
8094	043110	004737	042430	
8095	043114	172437	045216	
8096	043120	000004		
8097				
8098	043122	004737	042430	
8099	043126	010201		
8100	043130	170331		
8101	043132	000004		
8102	043134	004737	042430	
8103	043140	170337	045216	

8104	043144	000004		IOT			:FORCE A TRAP.
8105	043146	005037	177572	CLR	MMRO		:TURN OFF MEMORY MANAGEMENT.
8106	043152	010046		MOV	R0,-(SP)		:SAVE R0 FOR NEXT TEST
8107	043154	010246		MOV	R2,-(SP)		:SAVE R2 FOR NEXT TEST
8108	043156	013737	042426 000020	MOV	SAVIOT,IOTRAP		:RESTORE SCOPE TRAP VECTOR

8109

```

.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.
:
:*****
    
```

```

8110 043164 000004 044006 000020 TST111: SCOPE
8111 043166 012737 MOV #FALTRP,IOTRAP;SET UP FOR FAILURE OF TRAPS FOR THIS TEST.
8112 043174 012602 MOV (SP)+,R2 ;RESTORE R2 FOR THIS TEST
8113 043200 004737 MOV (SP)+,R0 ;RESTORE R0 FOR THIS TEST
8114 043204 010201 JSR PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
8115 043206 170541 MOV R2,R1 ;SET UP R1 FOR MODE 4.
8116 043210 000004 TSTF -(R1) ;FDST-NOTCLR PAGE 21.
8117 IOT ;FORCE A TRAP.
8118 043212 004737 JSR PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
8119 043216 010201 MOV R2,R1 ;CORRECT R1.
8120 043220 170441 CLRf -(R1) ;FDST MODES PAGE 27.
8121 043222 000004 IOT ;FORCE A TRAP.
8122
8123 043224 004737 JSR PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
8124 043230 010201 MOV R2,R1 ;CORRECT R1.
8125 043232 17704 LDCIF -(R1),ACO ;SOURCE MODES PAGE 28.
8126 043234 000004 IOT ;FORCE A TRAP.
8127
8128 043236 004737 JSR PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
8129 043242 010201 MOV R2,R1 ;CORRECT R1.
8130 043244 172441 LDF -(R1),ACO ;FSRC MODES PAGE 4.
8131 043246 000004 IOT ;FORCE A TRAP.
8132
8133 043250 004737 JSR PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
8134 043254 010201 MOV R2,R1 ;CORRECT R1.
8135 043256 170341 STST -(R1) ;DEST MODES PAGE 33.
8136 043260 000004 IOT ;FORCE A TRAP.
8137 043262 005037 CLR MMRO ;TURN OFF MEMORY MANAGEMENT.
8138 043266 010046 MOV R0,-(SP) ;SAVE R0 FOR NEXT TEST
8139 043270 010246 MOV R2,-(SP) ;SAVE R2 FOR NEXT TEST
8140 043272 013737 MOV SAVIOT,IOTRAP ;RESTORE SCOPE TRAP VECTOR
    
```

8141

```
.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-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.
*
*****
```

```
TST112: SCOPE
8142 043300 000004      MOV      #FALTRP,IOTRAP;SET UP FOR FAILURE OF TRAPS FOR THIS TEST.
8143 043302 012737 044006 000020      MOV      (SP)+,R2      ;RESTORE R2 FOR THIS TEST
8144 043310 012602      MOV      (SP)+,R0      ;RESTORE R0 FOR THIS TEST
8145 043312 012600      JSR      PC,SETERL    ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
8146 043314 004737 042430      MOV      R3,R1      ;SET UP R1 FOR MODE 5.
8147 043320 010301      MOV      R3,R4      ;MOVE 'START' VALUE INTO R4.
8148 043322 010304      TSTF    a-(R1)      ;FDST-NOTCLR PAGE 21.
8149 043324 170551      IOT      ;FORCE A TRAP.
8150 043326 000004
8151 043330 004737 042430      JSR      PC,SETERL    ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
8152 043334 010301      MOV      R3,R1      ;CORRECT R1.
8153 043336 170451      CLRF    a-(R1)      ;FDST MODES PAGE 27.
8154 043340 000004      IOT      ;FORCE A TRAP.
8155
8156 043342 004737 042430      JSR      PC,SETERL    ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
8157 043346 010301      MOV      R3,R1      ;CORRECT R1.
8158 043350 177051      LDCIF   a-(R1),ACO  ;SOURCE MODES PAGE 28.
8159 043352 000004      IOT      ;FORCE A TRAP.
8160
8161 043354 004737 042430      JSR      PC,SETERL    ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
8162 043360 010301      MOV      R3,R1      ;CORRECT R1.
8163 043362 172451      LDF     a-(R1),ACO  ;FSRC MODES PAGE 4.
8164 043364 000004      IOT      ;FORCE A TRAP.
8165
8166 043366 004737 042430      JSR      PC,SETERL    ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
8167 043372 010301      MOV      R3,R1      ;CORRECT R1.
8168 043374 170351      STST    a-(R1)      ;DEST MODES PAGE 33.
8169 043376 000004      IOT      ;FORCE A TRAP.
8170 043400 005037 177572      CLR      MMRO      ;TURN OFF MEMORY MANAGEMENT.
8171 043404 013737 042426 000020      MOV      SAVIOT,IOTRAP;RESTORE SCOPE TRAP VECTOR
```

8172

```
.SBTTL 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.
:*
```

```
043412 000004
8173 043412 000004 134376 TST113: SCOPE
8174 043414 012767 044006 .DSABL AMA ;DISABLE MODE 6 TO MODE 3 CONVERSIONS
8175 043422 004767 177002 MOV #FALTRP,IOTRAP;SET UP FOR FAILURE OF TRAPS FOR THIS TEST.
8176 043426 010301 JSR PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
8177 043430 010304 MOV R3,R1 ;SET UP R1 FOR MODE 6.
8178 043432 170561 000000 MOV R3,R4 ;MOVE 'START' VALUE INTO R4.
8179 043436 000004 TSTF 0(R1) ;FDST-NOTCLR PAGE 21.
8180 043440 004767 176764 IOT ;FORCE A TRAP.
8181 043444 170567 001546 JSR PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
8182 043450 000004 TSTF NODAT ;FDST-NOTCLR PAGE 21.
8183 043452 004767 176752 IOT ;FORCE A TRAP.
8184 043456 170461 000000 JSR PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
8185 043462 000004 CLRF 0(R1) ;FDST MODES PAGE 27.
8186 043464 004767 176740 IOT ;FORCE A TRAP.
8187 043470 170467 001522 JSR PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
8188 043474 000004 CLRF NODAT ;FDST MODES PAGE 27.
8189 043476 004767 176726 IOT ;FORCE A TRAP.
8190 043502 177061 000000 JSR PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
8191 043506 000004 LDCIF 0(R1),ACO ;SOURCE MODES PAGE 28.
8192 043510 004767 176714 IOT ;FORCE A TRAP.
8193 043514 177067 001476 JSR PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
8194 043520 000004 LDCIF NODAT,ACO ;SOURCE MODES PAGE 28.
8195 043522 004767 176702 IOT ;FORCE A TRAP.
8196 043526 172461 000000 JSR PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
8197 043532 000004 LDF 0(R1),ACO ;FSRC MODES PAGE 4.
8198 043534 004767 176670 IOT ;FORCE A TRAP.
8199 043540 172467 001452 JSR PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
8200 043544 000004 LDF NODAT,ACO ;FSRC MODES PAGE 4.
8201 043546 004767 176656 IOT ;FORCE A TRAP.
8202 043552 170361 000000 JSR PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
8203 043556 000004 STST 0(R1) ;DEST MODES PAGE 33.
8204 043560 004767 176644 IOT ;FORCE A TRAP.
8205 043564 170367 001426 JSR PC,SETERL ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
8206 043570 000004 STST NODAT ;DEST MODES PAGE 33.
8207 043572 005067 133774 IOT ;FORCE A TRAP.
8208 043576 010246 CLR MMRO ;TURN OFF MEMORY MANAGEMENT.
8209 043600 016767 176622 134212 MOV R2,-(SP) ;SAVE R2 FOR NEXT TEST
MOV SAVIOT,IOTRAP ;RESTORE SCOPE TRAP VECTOR
.ENABL AMA ;REENABLE MODE 6 TO MODE 3 CONVERSIONS
```

8211

.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.
*
*****
    
```

```

TST114: SCOPE
8212 043606 000004 044006 000020  MOV    #FALTRP,IOTRAP;SET UP FOR FAILURE OF TRAPS FOR THIS TEST.
8213 043610 012737 044006 000020  MOV    (SP)+,R2      ;RESTORE R2 FOR THIS TEST
8214 043620 004737 042430 000000  JSR    PC,SETERL    ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
8215 043624 010201 042430 000000  MOV    R2,R1        ;SET UP R1 FOR MODE 7.
8216 043626 010204 042430 000000  MOV    R2,R4        ;MOVE 'START' VALUE TO R4.
8217 043630 170571 000000  TSTF   @0(R1)       ;FDST-NOTCLR PAGE 21.
8218 043634 000004 042430 000000  IOT     ;FORCE A TRAP.
8219 043636 004737 042430 000000  JSR    PC,SETERL    ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
8220 043642 170577 001360 000000  TSTF   @NODAT+10   ;FDST-NOTCLR PAGE 21.
8221 043646 000004 042430 000000  IOT     ;FORCE A TRAP.
8222 043650 004737 042430 000000  JSR    PC,SETERL    ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
8223 043654 170471 000000  CLRF   @0(R1)       ;FDST MODES PAGE 27.
8224 043660 000004 042430 000000  IOT     ;FORCE A TRAP.
8225 043662 004737 042430 000000  JSR    PC,SETERL    ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
8226 043666 170477 001334 000000  CLRF   @NODAT+10   ;FDST MODES PAGE 27.
8227 043672 000004 042430 000000  IOT     ;FORCE A TRAP.
8228 043674 004737 042430 000000  JSR    PC,SETERL    ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
8229 043700 177071 000000  LDCIF  @0(R1),ACO   ;SOURCE MODES PAGE 28.
8230 043704 000004 042430 000000  IOT     ;FORCE A TRAP.
8231 043706 004737 042430 000000  JSR    PC,SETERL    ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
8232 043712 177077 001310 000000  LDCIF  @NODAT+10,ACO ;SOURCE MODES PAGE 28.
8233 043716 000004 042430 000000  IOT     ;FORCE A TRAP.
8234 043720 004737 042430 000000  JSR    PC,SETERL    ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
8235 043724 172471 000000  LDF    @0(R1),ACO   ;FSRC MODES PAGE 4.
8236 043730 000004 042430 000000  IOT     ;FORCE A TRAP.
8237 043732 004737 042430 000000  JSR    PC,SETERL    ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
8238 043736 172477 001264 000000  LDF    @NODAT+10,ACO ;FSRC MODES PAGE 4.
8239 043742 000004 042430 000000  IOT     ;FORCE A TRAP.
8240 043744 004737 042430 000000  JSR    PC,SETERL    ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
8241 043750 170371 000000  STST   @0(R1)       ;DEST MODES PAGE 33.
8242 043754 000004 042430 000000  IOT     ;FORCE A TRAP.
8243 043756 004737 042430 000000  JSR    PC,SETERL    ;GO SET ERROR LOOP TO ADRS OF NEXT INST ;DPM002
8244 043762 170377 001240 000000  STST   @NODAT+10,  ;DEST MODES PAGE 33.
8245 043766 000004 042430 000000  IOT     ;FORCE A TRAP.
8246 043770 005037 177572 000020  CLR    MMRO         ;TURN OFF MEMORY MANAGEMENT.
8247 043774 013737 042426 000020  MOV    SAVIOT,IOTRAP ;RESTORE SCOPE TRAP VECTOR
8248 044002 000137 045376 000020  JMP    ENDTES       ;JUMP TO END TEST.
    
```

8249	044006	005037	177572			FALTRP: CLR	MMRO	;TURN OFF MEMORY MANAGEMENT.
8250	044012	011637	001260			MOV	(SP), \$TMP13	;MOVE NEXT INSTRUCTION ADDRESS TO \$TMP13.
8251								
8252								:THIS NEXT SECTION NOW CORRECTS THE CONTENTS OF \$TMP13 SO THAT IT POINTS
8253								:TO THE PREVIOUS FPP INSTRUCTION. IT DOES THIS BY SUBTRACTING 2 FROM THE
8254								:ADDRESS IN \$TMP13, REPLACING THE 170000 THAT THE BIC INSTRUCTION USES,
8255								:AND BIT CLEARING THE INSTRUCTION WITH 170000. IF THE INSTRUCTION THAT
8256								:\$TMP13 IS POINTING TO IS NOT AN FPP INSTRUCTION, THE 170000 WILL NOT
8257								:CLEAR, SATISFYING THE NEXT BRANCH. THE ADDRESS IS AGAIN CORRECTED,
8258								:AND THE TESTING PROCESS STARTS OVER. THIS CONTINUES UNTIL \$TMP13 IS
8259								:POINTING TO AN FPP INSTRUCTION, AND NORMALLY WILL NOT BE EXECUTED MORE
8260								:THAN THREE TIMES BEFORE FINDING THE INSTRUCTION.
8261	044016	162737	000002	001260	1\$:	SUB	#2, \$TMP13	;SUBTRACT 2 FROM \$TMP13.
8262	044024	012737	170000	044036		MOV	#170000, 2\$+4	;SET UP BIC DATA LOCATION.
8263	044032	047727	135222	170000	2\$:	BIC	@\$TMP13, #170000	;TEST TO SEE IF FPP INSTRUCTION.
8264	044040	001366				BNE	1\$;BRANCH BACK FOR ANOTHER TRY IF NOT.
8265	044042	012737	000364	001272		MOV	#364, \$TMP20	;MOVE FAILURE TO ABORT ERROR TO \$TMP20.
8266	044050	000137	045104			JMP	MULTER	;JUMP TO MULTIPLE ERROR HANDLER.

8267	044054	013737	177574	001236	TRPV:	MOV	SR1,\$TMP2	:MOVE SR1 TO \$TMP2 FOR TESTING.
8268	044062	013737	177576	001260		MOV	MMR2,\$TMP13	:TRANSFER ADDRESS OF INST. CAUSING TRAP TO \$TMP13.
8269	044070	005037	177572			CLR	MMR0	:TURN OFF MEMORY MANAGEMENT.
8270	044074	112737	000365	001272		MOVB	#365,\$TMP20	:MOVE 365, THE MODE 0 ERROR, TO LOWER BYTE IN ERROR POINTER.
8271	044102	022737	042556	001260		CMP	#MODE1,\$TMP13	:SEE IF INSTRUCTION CAUSING TRAP IS BEFORE MODE 1 (MODE 0).
8272	044110	002402				BLT	1\$:BRANCH AROUND MODE 0 ERROR JUMP IF NOT.
8273	044112	001137	045104			JMP	MULTER	:JUMP TO ERROR NEST.
8274	044116	017737	135136	001266	1\$:	MOV	@\$TMP13,\$TMP16	:MOVE INSTRUCTION CAUSING TRAP TO \$TMP16.
8275	044124	112737	000363	001272		MOVB	#363,\$TMP20	:MOVE 363, SR1 WRONG ERROR, TO LOWER BYTE IN ERROR POINTER.
8276	044132	005037	001240			CLR	\$TMP3	:CLEAR CALCULATED LOCATION.
8277	044136	012737	044330	001244		MOV	#65\$,\$TMP5	:MOVE NEXT CHECK ADDRESS TO \$TMP5.
8278	044144	022737	042674	001260		CMP	#LABEL1,\$TMP13	:SEE IF TRAP IS BEFORE MODE 2 REG 1 CLRF INST.
8279	044152	100053				BPL	61\$:BRANCH TO SR1=0 TEST IF SO.
8280	044154	012737	000060	044166		MOV	#60,200\$:SET UP BIC DATA POSITION.
8281	044162	043727	001266		2\$:	BIC	\$TMP16,(PC)+	:TEST TO SEE IF MODE 6 OR 7 INSTRUCTION.
8282	044166	000060			200\$:	.WORD	60	:LOCATION TO HOLD 60
8283	044170	001444				BEQ	61\$:BRANCH DIRECTLY TO BYTE TABLE TESTING IF SO.
8284								:THIS NEXT ROUTINE DETERMINES WHICH REGISTER WAS IN THE INSTRUCTION, AND
8285								:LOADS THE START AND END VALUES OF EITHER R1 OR R7 (PROGRAM COUNTER) INTO
8286								:\$TMP17 AND \$TMP3 RESPECTIVELY. THEY ARE THEN SUBTRACTED TO FIND THE
8287								:DIFFERENCE THAT ACTUALLY OCCURED. IF NO DIFFERENCE WAS FOUND, THE TEST
8288								:FOR ZERO IN SR1 IS ACCOMPLISHED. IF A DIFFERENCE IS FOUND, THE DIFFERENCE
8289								:IS SHIFTED LEFT 3 PLACES, THE TOP BYTE IS CLEARED, AND THE REGISTER
8290								:OF THE INSTRUCTION IS ADDED. \$TMP3 NOW CONTAINS WHAT SHOULD APPEAR
8291								:IN SR1, ACCORDING TO WHAT ACTUALLY HAPPENED TO THE REGISTER.
8292	044172	042737	177770	001266	4\$:	BIC	#177770,\$TMP16	:BIT CLEAR THE INSTRUCTION, LEAVING THE REG EXPOSED.
8293	044200	023727	001266	000007		CMP	\$TMP16,#7	:COMPARE REGISTER TO DETERMINE IF IT IS REG 7.
8294	044206	001405				BEQ	5\$:BRANCH TO THE REG 7 SETUP IF EQUAL TO REG 7.
8295	044210	010437	001270			MOV	R4,\$TMP17	:MOVE THE START VALUE TO \$TMP17.
8296	044214	010137	001240			MOV	R1,\$TMP3	:MOVE THE END VALUE TO \$TMP3.
8297	044220	000410				BR	6\$:BRANCH TO CONTINUE.
8298	044222	013737	001260	001270	5\$:	MOV	\$TMP13,\$TMP17	:MOVE THE START VALUE TO \$TMP17.
8299	044230	062737	000002	001270		ADD	#2,\$TMP17	:ADD 2 TO START VALUE FOR NORMAL INCREMENTING.
8300	044236	011637	001240			MOV	(SP),\$TMP3	:MOVE THE END VALUE TO \$TMP3.
8301	044242	163737	001270	001240	6\$:	SUB	\$TMP17,\$TMP3	:FIND THE DIFFERENCE THAT OCCURED.
8302	044250	001414				BEQ	61\$:BRANCH TO TEST FOR SR1=0 IF NO DIFFERENCE.
8303	044252	006337	001240			ASL	\$TMP3	:ARITHMETIC SHIFT LEFT \$TMP3 3
8304	044256	006337	001240			ASL	\$TMP3	:PLACES TO PUT DIFFERENCE FOUND
8305	044262	006337	001240			ASL	\$TMP3	:IN BITS 3 THROUGH 7.
8306	044266	042737	177400	001240		BIC	#177400,\$TMP3	:BIT CLEAR UPPER BYTE OF \$TMP3.
8307	044274	063737	001266	001240		ADD	\$TMP16,\$TMP3	:ADD THE REGISTER THAT WAS CHANGED, AND
8308	044302	111537	001276		61\$:	MOVB	(R5),\$TMP22	:MOVE EXPECTED DATA TO \$TMP22.
8309	044306	123725	001236			CMPB	\$TMP2,(R5)+	:COMPARE SR1 WITH TABLE DATA.
8310	044312	001004				BNE	62\$:BRANCH TO ERROR JUMP IF WRONG.
8311	044314	005305				DEC	R5	:CORRECT R5 BEFORE NEXT COMPARE.
8312	044316	123725	001240			CMPB	\$TMP3,(R5)+	:COMPARE CALCULATED WITH TABLE DATA.
8313	044322	001402				BEQ	65\$:BRANCH AROUND ERROR JUMP IF OK.
8314	044324	000137	045002		62\$:	JMP	7\$:JUMP TO ERROR REPORT IF INCORRECT.
8315	044330	132737	000001	001273	65\$:	BITB	#1,\$TMP20+1	:TEST TO SEE IF BIT 8 IS SET.
8316	044336	001402				BEQ	66\$:BRANCH AROUND AC SKIP JUMP IF NOT.
8317	044340	000137	045166			JMP	RETURN	:JUMP TO RETURN - AC TESTS ARE TO BE SKIPPED.
8318	044344	112737	000366	001272	66\$:	MOVB	#366,\$TMP20	:MOVE 366, AC LOAD ERROR, TO ERROR POINTER.
8319	044352	010037	045310			MOV	R0,STORE+56	:STORE R0 FOR USE LATER IN THIS ROUTINE.
8320	044356	005037	001236			CLR	\$TMP2	:MOVE A '0' IN 'AC CHANGED' LOCATION.
8321	044362	012737	044426	001244		MOV	#101\$,\$TMP5	:MOVE RETURN TO \$TMP5.
8322	044370	173437	045232			CMPF	STORE,ACO	:SEE IF ACO WAS CHANGED.
8323	044374	170000				CFCC		:COPY FPP CONDITION CODES TO CPU CODES.

8324	044376	001413				BEQ	101\$:BRANCH TO NEXT TEST IF OK.
8325	044400	174037	045300			STF	ACO,STORE+46		:STORE ACTUAL ACO FOR ERROR PRINTING.
8326	044404	012700	045232			MOV	#STORE,RO		:MOVE ADDRESS OF EXPECTED ACC TO RO.
8327									:THE NEXT TWO INSTRUCTIONS TRY TO RESTORE THE ACCUMULATOR AND CHECK THE ACCUMULATOR
8328									:TO MAKE SURE IT WAS RESTORED PROPERLY FOR THE NEXT RUN THROUGH THIS TRAP HANDLER.
8329									:IT IS *IMPORTANT* TO REALIZE THAT IF THE "CMPF" FINDS A DIFFERENCE, THAT THE
8330									:*FLOATING*POINT*STATUS* IS BEING CHANGED MISTAKENLY. AN ERROR IN THE MICROC'DE
8331									:HAS BEEN FOUND TO CAUSE THIS, SO CHECK THE REVISION OF THE ROM/PROM SET IN THE
8332									:FPP YOU HAVE. IF YOU DO HAVE WHAT *SEEMS* TO BE THE LATEST REV, A NEW REV WI.'
8333									:BE COMING OUT TO CORRECT THIS PROBLEM. THIS SAME 'LDF/CMPF' SET OF RESTORE/
8334									:CHECK INSTRUCTIONS IS ACCOMPLISHED FOR EACH ACCUMULATOR CHECK. IT IS ALSO
8335									:IMPORTANT TO NOTE THAT IF AN ACCUMULATOR FAILS TO RESTORE PROPERLY, SUBSEQUENT
8336									:PASSES THROUGH THE TRAP HANDLER WILL SKIP THE ACCUMULATOR CHECKS DUE TO THE
8337									:BIT TEST #400 ABOVE. FOR EXAMPLE, IF ACO FAILS TO LOAD PROPERLY, AC1 THROUGH
8338									:AC3 WILL STILL BE CHECKED. AS SOON AS ANOTHER FPP INSTRUCTION TRAPS IN THE
8339									:MAIN TEST, ALL *FURTHER* ACO-AC3 CHECKS WILL BE SKIPPED.
8340	044410	172437	045232			LDF	STORE,ACO		:RESTORE ACO.
8341	044414	173437	045232			CMPF	STORE,ACO		:SEE IF IT WAS RESTORED PROPERLY.
8342	044420	170000				CFCC			:COPY FPP CONDITION CODES TO CPU CODES.
8343	044422	001567				BEQ	7\$:BRANCH TO ERROR CALL IF OK.
8344	044424	000476				BR	113\$:BRANCH TO ERROR SETUP ROUTINE.
8345	044426	012737	000001	001236	101\$:	MOV	#1,\$TMP2		:PUT A '1' IN 'AC CHANGED' LOCATION.
8346	044434	012737	044500	001244		MOV	#102\$,\$TMP5		:MOVE RETURN TO \$TMP5.
8347	044442	173537	045242			CMPF	STORE+10,AC1		:SEE IF AC1 WAS CHANGED.
8348	044446	170000				CFCC			:COPY FPP CONDITION CODES TO CPU CODES.
8349	044450	001413				BEQ	102\$:BRANCH TO NEXT TEST IF OK.
8350	044452	174137	045300			STF	AC1,STORE+46		:STORE ACTUAL AC1 FOR ERROR PRINTING.
8351	044456	012700	045242			MOV	#STORE+10,RO		:MOVE ADDRESS OF EXPECTED AC1 TO RO.
8352	044462	172537	045242			LDF	STORE+10,AC1		:RESTORE AC1.
8353	044466	173537	045242			CMPF	STORE+10,AC1		:SEE IF IT WAS RESTORED PROPERLY.
8354	044472	170000				CFCC			:COPY FPP CONDITION CODES TO CPU CODES.
8355	044474	001542				BEQ	7\$:BRANCH TO ERROR CALL IF OK.
8356	044476	000451				BR	113\$:BRANCH TO ERROR SETUP ROUTINE.
8357	044500	012737	000002	001236	102\$:	MOV	#2,\$TMP2		:PUT A '2' IN 'AC CHANGED' LOCATION.
8358	044506	012737	044552	001244		MOV	#103\$,\$TMP5		:MOVE RETURN TO \$TMP5.
8359	044514	173637	045252			CMPF	STORE+20,AC2		:SEE IF AC2 WAS CHANGED.
8360	044520	170000				CFCC			:COPY FPP CONDITION CODES TO CPU CODES.
8361	044522	001413				BEQ	103\$:BRANCH TO NEXT TEST IF OK.
8362	044524	174237	045300			STF	AC2,STORE+46		:STORE ACTUAL AC2 FOR ERROR PRINTING.
8363	044530	012700	045252			MOV	#STORE+20,RO		:MOVE ADDRESS OF EXPECTED AC2 TO RO.
8364	044534	172637	045252			LDF	STORE+20,AC2		:RESTORE AC2.
8365	044540	173637	045252			CMPF	STORE+20,AC2		:SEE IF IT WAS RESTORED PROPERLY.
8366	044544	170000				CFCC			:COPY FPP CONDITION CODES TO CPU CODES.
8367	044546	001515				BEQ	7\$:BRANCH TO ERROR CALL IF OK.
8368	044550	000424				BR	113\$:BRANCH TO ERROR SETUP ROUTINE.
8369	044552	012737	000003	001236	103\$:	MOV	#3,\$TMP2		:PUT A '3' IN 'AC CHANGED' LOCATION.
8370	044560	012737	044632	001244		MOV	#100\$,\$TMP5		:MOVE RETURN TO \$TMP5.
8371	044566	173737	045262			CMPF	STORE+30,AC3		:SEE IF AC3 WAS CHANGED.
8372	044572	170000				CFCC			:COPY FPP CONDITION CODES TO CPU CODES.
8373	044574	001416				BEQ	100\$:BRANCH TO NEXT TEST IF OK.
8374	044576	174337	045300			STF	AC3,STORE+46		:STORE ACTUAL AC3 FOR ERROR PRINTING.
8375	044602	012700	045262			MOV	#STORE+30,RO		:MOVE ADDRESS OF EXPECTED AC3 TO RO.
8376	044606	172737	045262			LDF	STORE+30,AC3		:RESTORE AC3.
8377	044612	173737	045262			CMPF	STORE+30,AC3		:SEE IF IT WAS RESTORED PROPERLY.
8378	044616	170000				CFCC			:COPY FPP CONDITION CODES TO CPU CODES.
8379	044620	001470				BEQ	7\$:BRANCH TO ERROR CALL IF OK.
8380	044622	012737	000770	001272	113\$:	MOV	#770,\$TMP20		:MOVE 370 FOR AC LOAD FAILURE, & SET BIT 8 OF ERROR POINTER.

8381	044630	000464			BR	7\$:BRANCH TO ERROR CALL.
8382	044632	005037	001236		CLR	\$TMP2		:CLEAR 'REGISTER CHANGED' LOCATION.
8383	044636	112737	000367	001272	MOVB	#367,\$TMP20		:MOVE 367, GENERAL REGISTER CHANGED ERROR, TO POINTER.
8384	044644	012737	044700	001244	MOV	#120,\$TMP5		:MOVE RETURN TO \$TMP5.
8385	044652	023700	045272		CMP	STORE+40,R0		:SEE IF R0 WAS CHANGED.
8386	044656	001410			BEQ	120\$:BRANCH TO NEXT TEST IF OK.
8387	044660	010037	001246		MOV	R0,\$TMP6		:MOVE ACTUAL R0 TO LOCATION FOR ERROR PRINTING.
8388	044664	013737	045272	001240	MOV	STORE+40,\$TMP3		:MOVE EXPECTED TO LOCATION FOR ERROR PRINTING.
8389	044672	013700	045272		MOV	STORE+40,R0		:RESTORE R0.
8390	044676	000441			BR	7\$:BRANCH TO ERROR CALL.
8391	044700	012737	000002	001236	MOV	#2,\$TMP2		:PUT A '2' IN 'REGISTER CHANGED' LOCATION.
8392	044706	012737	044742	001244	MOV	#130,\$TMP5		:MOVE RETURN TO \$TMP5.
8393	044714	023702	045274		CMP	STORE+42,R2		:SEE IF R2 WAS CHANGED.
8394	044720	001410			BEQ	130\$:BRANCH TO NEXT TEST IF OK.
8395	044722	010237	001246		MOV	R2,\$TMP6		:MOVE ACTUAL R2 TO LOCATION FOR ERROR PRINTING.
8396	044726	013737	045274	001240	MOV	STORE+42,\$TMP3		:MOVE EXPECTED TO LOCATION FOR ERROR PRINTING.
8397	044734	013702	045274		MOV	STORE+42,R2		:RESTORE R2.
8398	044740	000420			BR	7\$:BRANCH TO ERROR CALL.
8399	044742	012737	000003	001260	MOV	#3,\$TMP13		:PUT A '3' IN 'REGISTER CHANGED' LOCATION.
8400	044750	012737	045166	001244	MOV	#RETURN,\$TMP5		:MOVE RETURN TO \$TMP5.
8401	044756	023703	045276		CMP	STORE+44,R3		:SEE IF R3 WAS CHANGED.
8402	044762	001501			BEQ	RETURN		:BRANCH TO RETURN IF OK.
8403	044764	010337	001246		MOV	R3,\$TMP6		:MOVE ACTUAL R3 TO LOCATION FOR ERROR PRINTING.
8404	044770	013737	045276	001240	MOV	STORE+44,\$TMP3		:MOVE EXPECTED TO LOCATION FOR ERROR PRINTING.
8405	044776	013703	045276		MOV	STORE+44,R3		:RESTORE R3.
8406	045002	116537	177777	045374	MOVB	-1(R5),EXPCTD		:MOVE DATA FROM TABLE FOR POSSIBLE USE ;DPM001
8407	045010	122737	000370	001272	CMPB	#370,\$TMP20		:TEST TO SEE IF AC INFO NEEDS TO BE STORED.
8408	045016	001404			BEQ	71\$:BRANCH TO INFO STORE ROUTINE IF SO.
8409	045020	122737	000366	001272	CMPB	#366,\$TMP20		:TEST TO SEE IF AC INFO NEEDS TO BE STORED.
8410	045026	001026			BNE	MULTER		:SKIP AC INFO ROUTINE IF NOT.
8411	045030	012037	001240		MOV	(R0)+,\$TMP3		:MOVE 1ST WORD OF ACTUAL AC DATA TO \$TMP3.
8412	045034	012037	001242		MOV	(R0)+,\$TMP4		:MOVE 2ND WORD OF ACTUAL AC DATA TO \$TMP4.
8413	045040	012037	001246		MOV	(R0)+,\$TMP6		:MOVE 3RD WORD OF ACTUAL AC DATA TO \$TMP6.
8414	045044	012037	001250		MOV	(R0)+,\$TMP7		:MOVE 4TH WORD OF ACTUAL AC DATA TO \$TMP7.
8415	045050	013700	045310		MOV	STORE+56,R0		:RESTORE R0 TO WHAT IT HAD AT BEGINNING OF TRAP.
8416	045054	013737	045300	001252	MOV	STORE+46,\$TMP10		:MOVE 1ST WORD OF EXPECTED AC DATA TO \$TMP10.
8417	045062	013737	045302	001254	MOV	STORE+50,\$TMP11		:MOVE 2ND WORD OF EXPECTED AC DATA TO \$TMP11.
8418	045070	013737	045304	001256	MOV	STORE+52,\$TMP12		:MOVE 3RD WORD OF EXPECTED AC DATA TO \$TMP12.
8419	045076	013737	045306	001274	MOV	STORE+54,\$TMP21		:MOVE 4TH WORD OF EXPECTED AC DATA TO \$TMP21.
8420	045104	012637	001266		MULTER:	MOV	(SP)+,\$TMP16	:SAVE 1ST CONTENTS OF STACK AND POP IT ONCE.
8421	045110	012637	001270		MOV	(SP)+,\$TMP17		:SAVE 2ND CONTENTS OF STACK AND POP IT AGAIN.
8422	045114	142737	000377	045130	BICB	#377,74\$:CLEAR OUT LAST ERROR OFFSET FROM ERROR INSTRUCTION.
8423	045122	153737	001272	045130	BISB	\$TMP20,74\$:PUT ERROR NUMBER TO BE ACCOMPLISHED IN ERROR INSTRUCTION.
8424								:THIS ERROR IS DEFINED BY THE CONTENTS OF THE LOWER BYTE OF LOCATION [1272]
8425	045130	104000			74\$:	ERROR	+0	
8426	045132	013746	001270		MOV	\$TMP17,-(SP)		:PUSH 2ND CONTENTS BACK ON THE STACK.
8427	045136	013746	001266		MOV	\$TMP16,-(SP)		:PUSH 1ST CONTENTS BACK ON THE STACK.
8428	045142	022737	000364	001272	CMP	#364,\$TMP20		:SEE IF RETURN ROUTINE IS TO BE SKIPPED.
8429	045150	001417			BEQ	RTI		:BRANCH TO RTI IF SO.
8430	045152	022737	000365	001272	CMP	#365,\$TMP20		:SEE IF RETURN ROUTINE IS TO BE SKIPPED.
8431	045160	001413			BEQ	RTI		:BRANCH TO RTI IF SO.
8432	045162	000177	134056		JMP	@\$TMP5		:JUMP TO CONTINUE CHECKING.
8433	045166	022776	000004	000000	RETURN:	CMP	#4,@(SP)	:SEE IF INSTRUCTION IS THE IOT.
8434	045174	001403			BEQ	9\$:BRANCH IF THE IOT HAS BEEN FOUND.
8435	045176	062716	000002		ADD	#2,(SP)		:CORRECT PC RETURN.
8436	045202	000771			BR	RETURN		:BRANCH BACK FOR ANOTHER TRY.
8437	045204	062716	000002		9\$:	ADD	#2,(SP)	:CORRECT PC RETURN TO POINT AFTER IOT FOUND.

```

8438 045210 005237 177572      RTI:   INC   MMRO      ;TURN ON MEMORY MANAGEMENT, AND
8439 045214 000002              RTI      ;RETURN FROM INTERRUPT.
8440
8441 045216      NODAT: .BLKW 6      ;LOCATION IN NON-RES. D-SPACE USED TO FORCE A TRAP.
8442      ;THE 'STORE' LOCATION BELOW IS PARTITIONED TO RESERVE *4* WORDS FOR EACH FP
8443      ;ACCUMULATOR, EVEN THOUGH ONLY 2 ARE REQUIRED FOR STORING A FLOATING NUMBER.
8444      ;THIS IS BECAUSE *IF* THE FPS IS CHANGED BY A PROBLEM IN THE FPP, SO THAT A
8445      ;*DOUBLE* IS STORED, *4* WORDS RESERVED WILL GUARANTEE THAT THE NEXT DATA BLOCK
8446      ;WILL NOT BE DISTURBED. PARTITIONING IS AS FOLLOWS:
8447      :          WORD(S)      USE
8448      :          -----      -----
8449      :          1 - 4          STORE AC0
8450      :          5 -10          STORE AC1
8451      :          11-14         STORE AC2
8452      :          15-20         STORE AC3
8453      :          21            STORE R0
8454      :          22            STORE R2
8455      :          23            STORE R3
8456      :          24-27         STORE ACTUAL AC
8457      :          30            STORE ACTUAL R0 SO R0 CAN BE USED IN AC ERROR CALLS
8458 045232      STORE: .BLKW 30      ;STORAGE LOCATIONS FOR THE FLOATING ACCUMULATORS & DATA.
8459      ;THE FOLLOWING BYTE TABLE WILL BE USED TO CHECK THE VALUES OF SR1 AND THE CALCULATED
8460      ;VALUES. SR1 MAY TRACK THE ACTIVE REGISTER PROPERLY, BUT IF THE *VALUE* OF THE
8461      ;INCREMENT/DECREMENT IS WRONG, AN ERROR STILL EXISTS.
8462 045312      000      000      000      BYTABL: .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
8463 045342      341      000      361      .BYTE 341,0,361,361,361,361,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
8464 045374      000000      EXPCTD: .WORD 0      ;LOCATION TO HOLD DATA FROM BYTE TABLE ;DPM001
8465 045376      005037 177572      ENDTES: CLR   MMRO      ;TURN OFF MEMORY MANAGEMENT.
8466 045402      013737 001262 000250      MOV   $TMP14,MMVECT    ;RESTORE MMVECT CONTENTS.
8467 045410      013737 042426 000020      MOV   SAVIOT,IOTRAP    ;RESTORE IOTRAP CONTENTS.
8468 045416      104412      DIDONE:      ;GO INITIALIZE THE FPS AND STACK; AND
      RSETUP      ;SEE IF THE USER HAS EXPRESSED
      ;THE DESIRE TO CHANGE THE SOFTWARE
      ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
      ;THE USER TYPED CONTROL G?).

8469
8470 045420      TST115:
8471
8472
    
```

8474

.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:
    
```

```

045420          SCOPE
045420 000004    CLR      $STNM      ;;ZERO THE TEST NUMBER
045422 005037 001102 CLR      $TIMES      ;;ZERO THE NUMBER OF ITERATIONS
045426 005037 001302 INC      $PASS        ;;INCREMENT THE PASS NUMBER
045432 005237 001324 BPL     1000$        ;BRANCH IF STILL PLUS ;DPM002
045436 100004    CLR      $PASS        ;CLEAR THE PASS COUNTER ;DPM002
045440 005037 001324 INC     $PASS2       ;INCREMENT OVERFLOW PASS COUNTER ;DPM002
045444 005237 046166 1000$: DEC    (PC)+    ;;LOOP?
045450 005327 $EOPCT: .WORD 1
045452 000001    BLE     999$          ;NO ;DPM002
045454 003402    JMP     $DOAGN        ;;YES
045456 000137 046114 999$: MOV    (PC)+,a(PC)+ ;;RESTORE COUNTER
045462 012737 $ENDCT: .WORD 1
045464 000001 $EOPCT
045466 045452    TST     $ERTTL      ;SEE IF ANY ERRORS THIS PASS ;DPM002
045470 005737 001112 BNE     5000$        ;BRANCH IF SO TO PRINT THE EOP ;DPM002
045474 001007    TST     EPENDS       ;SEE IF EOP MSGS ARE DISABLED ;DPM002
045476 005737 046164 BNE     $GET42      ;BRANCH IF SO ;DPM002
045502 001120    BIT     #777,$PASS  ;PRINT EOP EVERY 1000TH PASS ;DPM002
045504 032737 000777 001324 BNE     $GET42      ;BRANCH IF NOT MULTIPLE OF 1000 ;DPM002
045512 001114 5000$:
045514          TYPE    ,65$      ;;TYPE ASCIZ STRING
045514 104401 045522 BR      64$          ;;GET OVER THE ASCIZ
045520 000407 ;;65$: .ASCIZ <12><15>/END PASS # /
045540          TST     $PASS2     ;SEE IF OVERFLOW HAS NON-ZERO VALUE ;DPM002
045540 005737 046166 BEQ     4900$        ;BRANCH IF ZERO
045544 001440    MOV     $PASS2,-(SP) ;SAVE $PASS2 FOR TYPEOUT
045546 013746 046166 ;;TYPE OVERFLOW PASS NUMBER IN OCTAL ;DPM002
045552 104403    TYPOS
045554 006      .BYTE 6      ;;GO TYPE--OCTAL ASCII
045555 000      .BYTE 0      ;;TYPE 6 DIGITS
045556 005737 001324 TST     $PASS        ;;SUPPRESS LEADING ZEROS
045562 001007    BNE     3000$        ;SEE IF PASS COUNT IS ZERO ;DPM002
045564 104401 045572 TYPE    ,67$        ;BRANCH IF NOT ;DPM002
045570 000403 BR      66$          ;;TYPE ASCIZ STRING
;;67$: .ASCIZ '00000!'
;;66$:
045600          BR      4910$        ;GO TEST $ERTTL ;DPM002
045600 000426    MOV     #70000,$TMP5 ;CHECK 5TH OCTAL DIGIT FIRST ;DPM002
045602 012737 070000 001244 3000$: BIT    $TMP5,$PASS ;CHECK TO SEE IF OCTAL DIGIT IS ZERO ;DPM002
045610 033737 001244 001324 4000$: BNE     4900$        ;BRANCH OUT IF ZERO ;DPM002
045616 001013    TYPE    ,69$        ;;TYPE ASCIZ STRING
045620 104401 045626 BR      68$          ;;GET OVER THE ASCIZ
045624 000401 ;;69$: .ASCIZ '0!'
045630          ASR     $TMP5        ;SHIFT THE THREE BITS RIGHT 3 PLACES ;DPM002
045630 006237 001244 ASR     $TMP5
045634 006237 001244
    
```

```

045640 006237 001244 ASR $TMP5 ; ;DPM002
045644 000761 BR 4000$ ;BRANCH BACK TO CHECK $PASS ;DPM002
045646 013746 001324 4900$: MOV $PASS,-(S?) ;:SAVE $PASS FOR TYPEOUT
045652 104403 TYPOS ;:TYPE PASS NUMBER IN OCTAL
045654 006 .BYTE 6 ;:GO TYPE--OCTAL ASCII
045655 000 .BYTE 0 ;:TYPE 6 DIGITS
045656 005737 001112 4910$: TST $ERTTL ;:SUPPRESS LEADING ZEROS
045662 001426 BEQ 5001$ ;SEE IF ANY ERRORS THIS PASS ;DPM002
045664 104401 045672 70$: .ASCIZ / TOTAL ERRORS THIS PASS / ;DPM002
045670 000415 BR 70$ ;GET OVER THE ASCIZ
045724 013746 001112 70$: MOV $ERTTL,-(SP) ;:SAVE $ERTTL FOR TYPEOUT
045730 104403 TYPOS ;:TOTAL NUMBER OF ERRORS IN OCTAL
045732 006 .BYTE 6 ;:GO TYPE--OCTAL ASCII
045733 000 .BYTE 0 ;:TYPE 6 DIGITS
045734 005037 001112 CLR $ERTTL ;:SUPPRESS LEADING ZEROS
045740 104401 001313 5001$: TYPE $CRLF ;:CLEAR ERROR TOTAL
045744 105777 133174 $GET42: TSTB @STKS ;:TYPE CARRIAGE RETURN, LINE FEED
045750 100042 BPL $GT42C ;IS A CHARACTER WAITING? ;DPM002
045752 013737 001146 001244 MOV $TKB,$TMP5 ;BRANCH IF NOT ;DPM002
045760 005737 046164 TST $ENDS ;WASTE THE CHARACTER, CLEARING READY ;DPM002
045764 001017 BNE $DISAB ;SEE WHICH STATE ENABLE/DISABLE IS IN ;DPM002
045766 005237 046164 INC $ENDS ;BRANCH IF EOP'S DISABLED ;DPM002
045772 104401 046000 TYPE $65$ ;SET FLAG DISABLING PRINTOUTS ;DPM002
045776 000411 BR 64$ ;TYPE ASCIZ STRING
;:GET OVER THE ASCIZ
;:65$: .ASCIZ <CRLF>!EOP'S DISABLED!<CRLF>
64$: BR $GT42C ;BRANCH OVER ENABLE ROUTINE ;DPM002
046022 000415 $DISAB: CLR $ENDS ;CLEAR FLAG ENABLING PRINTOUTS ;DPM002
046024 005037 046164 TYPE $65$ ;TYPE ASCIZ STRING
046030 104401 046036 BR 64$ ;GET OVER THE ASCIZ
046034 000410 ;:65$: .ASCIZ <CRLF>!EOP'S ENABLED!<CRLF>
64$: MOV @#42,R0 ;GET MONITOR ADDRESS
046056 013700 000042 $GT42C: BEQ $DOAGN ;BRANCH IF NO MONITOR
046062 001414 CLR -(SP) ;INSURE THE 'T' BIT IS CLEAR
046064 005046 MOV $CLR.T,-(SP) ;SETUP FOR AN RTI OR RTT
046066 012746 046074 BR $RTRN ;GO DO AN RTI OR RTT TO LOAD THE PSW
046072 000426 ;:WITH A CLEARED 'T' BIT
$CLR.T: MOV @#42,R0 ;INSURE R0 CONTAINS THE MONITORS
046074 013700 000042 BEQ $DOAGN ;RETURN ADDRESS
046100 001405 RESET ;CLEAR THE WORLD
046102 000005 SENDAD: JSR PC,(R0) ;GO TO MONITOR
046104 004710 NOP ;SAVE ROOM
046106 000240 NOP ;FOR
046110 000240 NOP ;ACT11
046112 000240 $DOAGN: TRAP ;PUSH OLD PSW AND PC ON STACK
046114 104400 BIC #20,(SP) ;CLEAR THE 'T' BIT
046116 042716 000020 BIT #BIT12,@SWR ;RUN WITH TRACE TRAP?
046122 032777 010000 133010 BNE 1$ ;BR IF NO
046130 001005 COM $TBIT ;IS IT TIME FOR TRACE TRAP
046132 005137 046156

```

END OF PASS ROUTINE

046136 100:02
046140 052716 000020
046144 012746 046152
046150 000002

BMI 1\$
BIS #20,(SP)
1\$: MOV #SLOOP,-(SP)
\$RTRN: RTI

::BR IF NO
::SET TRACE TRAP
::JUMP TO START OF TEST
::RETURN--THIS IS CHANGED TO
::AN 'RTT' IF 'RTT' IS A LEGAL
::INSTRUCTION

046152
046152 000137
046154 006764
046156 000000
046'60 377 377 000

\$LOOP:
JMP @PC)+
\$RTNAD: .WORD LOOP
\$TBIT: .WORD 0
\$ENULL: .BYTE -1,-1,0
.EVEN

::RETURN
::'T' BIT STATE INDICATOR
::NULL CHARACTER STRING

046164 000000
046166 000000

\$PENDS: .WORD 0
\$PASS2: .WORD 0

:LOCATION FOR EOP PRINT FLAG
:LOCATION FOR PASS COUNT OVERFLOW

:DPM002
:DPM002

8476

.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
    
```

```

046170          $SCOPE:
046170 104406          CKSWR          ;;TEST FOR CHANGE I% SOFT-SWR
046172 032777 040000 132740 1$: BIT #BIT14,@SWR          ;;LOOP ON PRESENT TEST?
046200 001134          BNE $OVER          ;;YES IF SW14=1
046202 000416          ;#####START OF CODE FOR THE XOR TESTER#####
          XTSTR: BR 6$          ;;IF RUNNING ON THE 'XOR' TESTER CHANGE
          ;;THIS INSTRUCTION TO A 'NOP' (NOP=240)
046204 013746 000004          MOV ERRVEC,-(SP)          ;;SAVE THE CONTENTS OF THE ERROR VECTOR
046210 012737 046230 000004          MOV #5$,ERRVEC          ;;SET FOR TIMEOUT
046216 005737 177060          TST 177060          ;;TIME OUT ON XOR?
046222 012637 000004          MOV (SP)+,ERRVEC          ;;RESTORE THE ERROR VECTOR
046226 000503          BR $SVLAD          ;;GO TO THE NEXT TEST
046230 022626          5$: CMP (SP)+,(SP)+          ;;CLEAR THE STACK AFTER A TIME OUT
046232 012637 000004          MOV (SP)+,ERRVEC          ;;RESTORE THE ERROR VECTOR
046236 000443          BR 7$          ;;LOOP ON THE PRESENT TEST
046240          6$:#####END OF CODE FOR THE XOR TESTER#####
046240 032777 000400 132672          BIT #BIT08,@SWR          ;;LOOP ON SPEC. TEST?
046246 001404          BEQ 2$          ;;BR IF NO
046250 127737 132664 001102          CMPB @SWR,$TSTNM          ;;ON THE RIGHT TEST? SWR<7:0>
046256 001505          BEQ $OVER          ;;BR IF YES
046260 013737 177766 046506 2$: MOV 177766,CPSAVE          ;;MOVE CPU ERR REG VALUE TO LOC FOR TST ;DPM001
046266 032737 000001 046506          BIT #BIT00,CPSAVE          ;;SEE IF THE POWER MONITOR BIT IS ON ;DPM001
046274 001411          BEQ 2000$          ;;BRANCH TO CONTINUE ROUTINE IF CLEAR ;DPM001
046276 042737 000001 177766          BIC #BIT00,177766          ;;CLEAR THE BIT FOUND TO BE SET ;DPM001
046304 012737 177777 001320          MOV #-1,$FATAL          ;;MOVE -1 TO $FATAL INDICATING PWR MON ER ;DPM001
046312 104177          ERROR +177          ;;CALL SPECIAL POWER FAIL BIT ERROR CALL ;DPM001
046314 105037 001103          CLRB $ERFLG          ;;CLEAR ERROR FLAG FOR CHECK BELOW ;DPM001
046320 105737 001103          2000$: TSTB $ERFLG          ;;HAS AN ERROR OCCURRED?
046324 001421          BEQ 3$          ;;BR IF NO
046326 123737 001115 001103          CMPB $ERMAX,$ERFLG          ;;MAX. ERRORS FOR THIS TEST OCCURRED?
046334 101015          BHI 3$          ;;BR IF NO
046336 032777 001000 132574          BIT #BIT09,@SWR          ;;LOOP ON ERROR?
046344 001404          BEQ 4$          ;;BR IF NO
046346 013737 001110 001106 7$: MOV $LPERR,$LPADR          ;;SET LOOP ADDRESS TO LAST SCOPE
046354 000446          BR $OVER
046356 105037 001103          4$: CLRB $ERFLG          ;;ZERO THE ERROR FLAG
046362 005037 001302          CLR $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
046366 000415          BR 1$          ;;ESCAPE TO THE NEXT TEST
046370 032777 004000 132542 3$: BIT #BIT11,@SWR          ;;INHIBIT ITERATIONS?
046376 001011          BNE 1$          ;;BR IF YES
046400 005737 001324          TST $PASS          ;;IF FIRST PASS OF PROGRAM
046404 001406          BEQ 1$          ;;INHIBIT ITERATIONS
046406 005237 001104          INC $ICNT          ;;INCREMENT ITERATION COUNT
    
```


SCOPE HANDLER ROUTINE

046412	023737	001302	001104	CMP	\$TIMES,\$ICNT	::CHECK THE NUMBER OF ITERATIONS MADE	
046420	002024			BGE	\$OVER	::BR IF MORE ITERATION REQUIRED	
046422	012737	000001	001104	1\$: MOV	#1,\$ICNT	::REINITIALIZE THE ITERATION COUNTER	
046430	013737	046510	001302	MOV	\$MXCNT,\$TIMES	::SET NUMBER OF ITERATIONS TO DO	
046436	105237	001102		\$SVLAD: INCB	\$STNM	::COUNT TEST NUMBERS	
046442	113737	001102	001322	MCVB	\$STNM,\$TESTN	::SET TEST NUMBER IN APT MAILBOX	
046450	011637	001106		MOV	(SP),\$LPADR	::SAVE SCOPE LOOP ADDRESS	
046454	011637	001110		MOV	(SP),\$LPERR	::SAVE ERROR LOOP ADDRESS	
046460	005037	001304		CLR	\$ESCAPE	::CLEAR THE ESCAPE FROM ERROR ADDRESS	
046464	112737	000001	001115	MOVB	#1,\$ERMAX	::ONLY ALLOW ONE(1) ERROR ON NEXT TEST	
046472	013777	001102	132442	\$OVER: MOV	\$STNM,@DISPLAY	::DISPLAY TEST NUMBER	
046500	013716	001106		MOV	\$LPADR,(SP)	::FUDGE RETURN ADDRESS	
046504	000002			RTI		::FIXES PS	
046506	000000			CPSAVE: .WORD	0	::LOCATION TO SAVE CPU ERR REG CONTENTS	:DPM001
046510	000001			\$MXCNT: 1		::MAX. NUMBER OF ITERATIONS	

8478

.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
  
```

```

046512 000000      IBSAVE: .WORD      0      ;LOC'N TO HOLD $ERRPC DURING DUAL ERR      ;DPM001
046514      $ERROR:
046514 104406      CKSWR      ;:TEST FOR CHANGE IN SOFT-SWR
046516 105237 001103 7$:      INCB      $ERFLG      ;:SET THE ERROR FLAG
046522 001775      BEQ      7$      ;:DON'T LET THE FLAG GO TO ZERO
046524 013777 001102 132410      MOV      $STNM,@DISPLAY ;:DISPLAY TEST NUMBER AND ERROR FLAG
046532 032777 002000 132400      BIT      #BIT10,@SWR   ;:BELL ON ERROR?
046540 001402      BEQ      1$      ;:NO - SKIP
046542 104401 001306      TYPE      $BELL      ;:RING BELL
046546 005237 001112      1$:      INC      $ERTTL     ;:COUNT THE NUMBER OF ERRORS
046552 011637 001116      MOV      (SP), $ERRPC  ;:GET ADDRESS OF ERROR INSTRUCTION
046556 162737 000002 001116      SUB      #2, $ERRPC
046564 117737 132326 001114      MOVB    @ $ERRPC, $ITEMB ;:STRIP AND SAVE THE ERROR ITEM CODE
046572 122737 000177 001114      CMPB    #177, $ITEMB   ;:SEE IF PWR MON ERROR CALL
046600 001421      BEQ      1000$      ;:BRANCH IF SO TO CALL THE ERROR      ;DPM001
046602 013737 177766 046506      MOV      177766, CPSAVE ;:MOVE CPU ERR REG TO CPSAVE FOR TEST ;DPM001
046610 032737 000001 046506      BIT      #BIT00, CPSAVE ;:SEE IF POWER MONITOR BIT IS SET   ;DPM001
046616 001412      BEQ      1000$      ;:BRANCH IF OK                       ;DPM001
046620 042737 000001 177766      BIC      #BIT00, 177766 ;:CLEAR THE BIT FOUND SET           ;DPM001
046626 013737 001116 046512      MOV      $ERRPC, IBSAVE ;:SAVE $ERROR                         ;DPM001
046634 104177      ERROR    +177      ;:CALL POWER MONITOR BIT ERROR      ;DPM001
046636 013737 046512 001116      MOV      IBSAVE, $ERRPC ;:RESTORE $ERROR                      ;DPM001
046644      1000$:
046644 032777 020000 132266      BIT      #BIT13, @SWR   ;:SKIP TYPEOUT IF SET
046652 001004      BNE      20$      ;:SKIP TYPEOUTS
046654 004737 051152      JSR      PC, ERTYPE   ;:GO TO USER ERROR ROUTINE
046660 104401 001313      TYPE      , $CRLF
046664      20$:
046664 122737 000001 001336      CMPB    #APTENV, $ENV  ;:RUNNING IN APT MODE
046672 001023      BNE      2$      ;:NO, SKIP APT ERROR REPORT
046674 005037 046736      CLR      21$      ;:CLEAR ANY PREVIOUS NUMBER FROM 21$ ;DPM001
046700 113737 001114 046736      MOVB    $ITEMB, 21$   ;:SET ITEM NUMBER AS ERROR NUMBER
046706 122737 000377 001114      CMPB    #377, $ITEMB  ;:SEE IF ITEM # IS OVER 400
046714 001006      BNE      900$      ;:BRANCH IF NOT
046716 012737 000400 046736      MOV      #400, 21$   ;:MOVE BASE OF 400 TO 21$
046724 067737 132166 046736      ADD      @ $ERRPC, 21$ ;:ADD NUMBER OVER 400 TO 21$
046732 004737 050006      900$:      JSR      PC, $ATY4    ;:REPORT FATAL ERROR TO APT
046736      21$:      .BYTE    0
046737      .BYTE    0
046740 000777      22$:      BR      22$      ;:APT ERROR LOOP
046742 005737 046512      2$:      TST      IBSAVE     ;:SEE IF POWER FAIL ERROR CALL      ;DPM001
046746 001005      BNE      3$      ;:BRANCH IF NOT - HALT NOT ALLOWED ;DPM001
046750 005777 132164      TST      @SWR      ;:HALT ON ERROR
  
```

```

046754 100002          BPL      3$          ;;SKIP IF CONTINUE
046756 000000          HALT                    ;;HALT ON ERROR!
046760 104406          CKSWR                    ;;TEST FOR CHANGE IN SOFT-SWR
046762 032777 001000 132150 3$: BIT      #BIT09,@SWR      ;;LOOP ON ERROR SWITCH SET?
046770 001405          BEQ      4$          ;;BR IF NO
046772 005737 046512          TST      IBSAVE        ;;SEE IF ERROR IS PWR MONITOR BIT ERROR :DPM001
046776 001002          BNE      4$          ;;BRANCH IF SO - DON'T FUDGE RETURN :DPM001
047000 013716 001110          MOV      $LPERR,(SP)   ;;FUDGE RETURN FOR LOOPING
047004 005737 001304          TST      $ESCAPE      ;;CHECK FOR AN ESCAPE ADDRESS
047010 001405          BEQ      5$          ;;BR IF NONE
047012 005737 046512          TST      IBSAVE        ;;SEE IF ERROR IS PWR MONITOR BIT ERROR :DPM001
047016 001002          BNE      5$          ;;BRANCH IF SO - DON'T FUDGE RETURN :DPM001
047020 013716 001304          MOV      $ESCAPE,(SP) ;;FUDGE RETURN ADDRESS FOR ESCAPE
047024          5$:          CMP      #$ENDAD,42    ;;ACT-11 AUTO-ACCEPT?
047024 022737 046104 000042          BNE      6$          ;;BRANCH IF NO
047032 001001          HALT                    ;;YES
047034 000000
047036          6$:          BIT      #BIT09,@SWR
047036 032777 001000 132074          BNE      ERM10
047044 001013          MOV      (SP), $REGO   ;SEE IF ERROR #377
047046 011637 001162          ADD      #-2,$REGO
047052 062737 177776 001162          CMPB    #377,@$REGO
047060 122777 000377 132074          BNE      ERM10
047066 001002          ADD      #2,(SP)
047070 062716 000002          ERM10: RTI
047074 000002
  
```

8480

.SBTTL SAVE AND RESTORE RO-R5 ROUTINES
 :*****

*SAVE RO-R5
 *CALL:
 * SAVREG
 *UPON RETURN FROM \$SAVREG THE STACK WILL LOOK LIKE:

- *TOP---(+16)
- * +2---(+18)
- * +4---R5
- * +6---R4
- * +8---R3
- *+10---R2
- *+12---R1
- *+14---R0

047076			\$SAVREG:		
047076	010046		MOV	R0,-(SP)	::PUSH R0 ON STACK
047100	010146		MOV	R1,-(SP)	::PUSH R1 ON STACK
047102	010246		MOV	R2,-(SP)	::PUSH R2 ON STACK
047104	010346		MOV	R3,-(SP)	::PUSH R3 ON STACK
047106	010446		MOV	R4,-(SP)	::PUSH R4 ON STACK
047110	010546		MOV	R5,-(SP)	::PUSH R5 ON STACK
047112	016646	000022	MOV	22(SP),-(SP)	::SAVE PS OF MAIN FLOW
047116	016646	000022	MOV	22(SP),-(SP)	::SAVE PC OF MAIN FLOW
047122	016646	000022	MOV	22(SP),-(SP)	::SAVE PS OF CALL
047126	016646	000022	MOV	22(SP),-(SP)	::SAVE PC OF CALL
047132	000002		RTI		

*RESTORE RO-R5

*CALL:
 * RESREG

047134			\$RESREG:		
047134	012666	000022	MOV	(SP)+,22(SP)	::RESTORE PC OF CALL
047140	012666	000022	MOV	(SP)+,22(SP)	::RESTORE PS OF CALL
047144	012666	000022	MOV	(SP)+,22(SP)	::RESTORE PC OF MAIN FLOW
047150	012666	000022	MOV	(SP)+,22(SP)	::RESTORE PS OF MAIN FLOW
047154	012605		MOV	(SP)+,R5	::POP STACK INTO R5
047156	012604		MOV	(SP)+,R4	::POP STACK INTO R4
047160	012603		MOV	(SP)+,R3	::POP STACK INTO R3
047162	012602		MOV	(SP)+,R2	::POP STACK INTO R2
047164	012601		MOV	(SP)+,R1	::POP STACK INTO R1
047166	012600		MOV	(SP)+,R0	::POP STACK INTO R0
047170	000002		RTI		

8482

.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

047172	000000		EOASCI:	.WORD	0		;LOC TO HOLD ADRS OF TERMINATOR BYTE ;DPM001
047174	105737	001157	\$TYPE:	TSTB	\$TPFLG		;:IS THERE A TERMINAL?
047200	100002			BPL	1\$;:BR IF YES
047202	000000			HALT			;:HALT HERE IF NO TERMINAL
047204	000432			BR	3\$;:LEAVE
047206	010046		1\$:	MOV	RO,-(SP)		;:SAVE RO
047210	017600	000002		MOV	@2(SP),RO		;:GET ADDRESS OF ASCIZ STRING
047214	122737	000001 001336		CMPB	#APTENV,\$ENV		;:RUNNING IN APT MODE
047222	001011			BNE	62\$;:NO,GO CHECK FOR APT CONSOLE
047224	132737	000100 001337		BITB	#APTSPOOL,\$ENVM		;:SPOOL MESSAGE TO APT
047232	001405			BEQ	62\$;:NO,GO CHECK FOR CONSOLE
047234	010037	047244		MOV	RO,61\$;:SETUP MESSAGE ADDRESS FOR APT
047240	004737	047776		JSR	PC,\$ATY3		;:SPOOL MESSAGE TO APT
047244	000000		61\$:	.WORD	0		;:MESSAGE ADDRESS
047246	132737	000040 001337	62\$:	BITB	#APTCSUP,\$ENVM		;:APT CONSOLE SUPPRESSED
047254	001005			BNE	60\$;:YES,SKIP TYPE OUT
047256	112046		2\$:	MOVB	(RO)+,-(SP)		;:PUSH CHARACTER TO BE TYPED ONTO STACK
047260	001007			BNE	4\$;:BR IF IT ISN'T THE TERMINATOR
047262	005726			TST	(SP)+		;:IF TERMINATOR POP IT OFF THE STACK
047264	010037	047172		MOV	RO,EOASCI		;:SAVE ADRS OF TERMINATOR - POSSIBLY USED;DPM001
047270	012600		60\$:	MOV	(SP)+,RO		;:RESTORE RO
047272	062716	000002	3\$:	ADD	#2,(SP)		;:ADJUST RETURN PC
047276	000002			RTI			;:RETURN
047300	122716	000011	4\$:	CMPB	#HT,(SP)		;:BRANCH IF <HT>
047304	001430			BEQ	8\$		
047306	122716	000200		CMPB	#CRLF,(SP)		;:BRANCH IF NOT <CRLF>
047312	001006			BNE	5\$		
047314	005726			TST	(SP)+		;:POP <CR><LF> EQUIV
047316	104401			TYPE			;:TYPE A CR AND LF
047320	001313			\$CRLF			
047322	105037	047530		CLRB	\$CHARCNT		;:CLEAR CHARACTER COUNT
047326	000753			BR	2\$;:GET NEXT CHARACTER
047330	004737	047412	5\$:	JSR	PC,\$TYPEC		;:GO TYPE THIS CHARACTER
047334	123726	001156	6\$:	CMPB	\$FILLC,(SP)+		;:IS IT TIME FOR FILLER CHARS.?
047340	001346			BNE	2\$;:IF NO GO GET NEXT CHAR.
047342	013746	001154		MOV	\$NULL,-(SP)		;:GET # OF FILLER CHARS. NEEDED
							;:AND THE NULL CHAR.
047346	105366	000001	7\$:	DECB	1(SP)		;:DOES A NULL NEED TO BE TYPED?
047352	002770			BLT	6\$;:BR IF NO--GO POP THE NULL OFF OF STACK
047354	004737	047412		JSR	PC,\$TYPEC		;:GO TYPE A NULL
047360	105337	047530		DECB	\$CHARCNT		;:DO NOT COUNT AS A COUNT

047364 000770

BR 7\$::LOOP

;HORIZONTAL TAB PROCESSOR

```

047366 112716 000040      8$:  MOVB  #' , (SP)      ;;REPLACE TAB WITH SPACE
047372 004737 047412      9$:  JSR   PC, $TYPEC      ;;TYPE A SPACE
047376 132737 000007 047,30 BITB  #7, $CHARCNT      ;;BRANCH IF NOT AT
047404 001372              BNE   9$                ;;TAB STOP
047406 005726              TST   (SP)+              ;;POP SPACE OFF STACK
047410 000722              BR    2$                ;;GET NEXT CHARACTER
047412                                $TYPEC:
047412 105777 131526              TSTB  @ $TKS              ;;CHAR IN KYBD BUFFER?      ;MJD001
047416 100022              BPL   10$              ;;BR IF NOT                  ;MJD001
047420 017746 131522              MOV   @ $TKB, -(SP)      ;;GET CHAR                   ;MJD001
047424 042716 177600              BIC   #177600, (SP)     ;;STRIP EXTRANEIOUS BITS   ;MJD001
047430 122716 000023              CMPB  # $XOFF, (SP)     ;;WAS CHAR XOFF            ;MJD001
047434 001012              BNE   102$             ;;BR IF NOT                  ;MJD001
047436                                101$:
047436 105777 131502              TSTB  @ $TKS              ;;WAIT FOR CHAR              ;MJD001
047442 100375              BPL   101$             ;;MJD001
047444 117716 131476              MOVB  @ $TKB, (SP)      ;;GET CHAR                   ;MJD001
047450 042716 177600              BIC   #177600, (SP)     ;;STRIP IT                   ;MJD001
047454 122716 000021              CMPB  # $XON, (SP)     ;;WAS IT XON?               ;MJD001
047460 001366              BNE   101$             ;;BR IF NOT                  ;MJD001
047462                                102$:
047462 005726              TST   (SP)+              ;;FIX STACK                  ;MJD001
047464                                10$:
047464 105777 131460              TSTB  @ $TPS              ;;WAIT UNTIL PRINTER IS READY ;MJD001
047470 100375              BPL   10$                ;MJD001
047472 116677 000002 131452      MOVB  2(SP), @ $TPB      ;;LOAD CHAR TO BE TYPED INTO DATA REG.
047500 122766 000015 000002      CMPB  #CR, 2(SP)        ;;IS CHARACTER A CARRIAGE RETURN?
047506 001003              BNE   1$                ;;BRANCH IF NO
047510 105037 047530              CLRB  $CHARCNT          ;;YES--CLEAR CHARACTER COUNT
047514 000406              BR    $TYPEX            ;;EXIT
047516 122766 000012 000002 1$:  CMPB  #LF, 2(SP)        ;;IS CHARACTER A LINE FEED?
047524 001402              BEQ   $TYPEX            ;;BRANCH IF YES
047526 105227              INCB  (PC)+              ;;COUNT THE CHARACTER
047530 000000      $CHARCNT: .WORD 0      ;;CHARACTER COUNT STORAGE
047532 000207      $TYPEX: RTS            PC

```

8484

.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
    
```

```

047534 017646 000000          $TYPOS: MOV      @(SP),-(SP)      ;;PICKUP THE MODE
047540 116637 000001 047765  MOVVB   1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
047546 112637 047767          MOVVB   (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
047552 062716 000002          ADD      #2,(SP)      ;;ADJUST RETURN ADDRESS
047556 000406          BR       $TYPON
047560 112737 000001 047765  $TYPOC: MOVVB   #1,$OFILL    ;;SET THE ZERO FILL SWITCH
047566 112737 000006 047767  MOVVB   #6,$OMODE+1  ;;SET FOR SIX(6) DIGITS
047574 112737 000005 047764  $TYPON: MOVVB   #5,$OCNT    ;;SET THE ITERATION COUNT
047602 010346          MOV      R3,-(SP)    ;;SAVE R3
047604 010446          MOV      R4,-(SP)    ;;SAVE R4
047606 010546          MOV      R5,-(SP)    ;;SAVE R5
047610 113704 047767          MOVVB   $OMODE+1,R4  ;;GET THE NUMBER OF DIGITS TO TYPE
047614 005404          NEG      R4
047616 062704 000C06          ADD      #6,R4      ;;SUBTRACT IT FOR MAX. ALLOWED
047622 110437 047766          MOVVB   R4,$OMODE    ;;SAVE IT FOR USE
047626 113704 047765          MOVVB   $OFILL,R4    ;;GET THE ZERO FILL SWITCH
047632 016605 000012          MOV      12(SP),R5  ;;PICKUP THE INPUT NUMBER
047636 005003          CLR      R3        ;;CLEAR THE OUTPUT WORD
047640 006105          1$:  ROL      R5      ;;ROTATE MSB INTO 'C'
047642 000404          BR       3$
047644 006105          2$:  ROL      R5      ;;GO DO MSB
047646 006105          ROL      R5        ;;FORM THIS DIGIT
047650 006105          ROL      R5
047652 010503          MOV      R5,R3
047654 006103          3$:  ROL      R3      ;;GET LSB OF THIS DIGIT
047656 105337 047766          DECB   $OMODE      ;;TYPE THIS DIGIT?
047662 100921          BPL     7$          ;;BR IF NO
047664 042703 177770          BIC     #177770,R3  ;;GET RID OF JUNK
047670 001002          BNE     4$          ;;TEST FOR 0
047672 005704          TST     R4          ;;SUPPRESS THIS 0?
047674 001403          BEQ     5$          ;;BR IF YES
047676 005204          4$:  INC     R4      ;;DON'T SUPPRESS ANYMORE 0'S
    
```

047700	052703	000060		BIS	#'0,R3	::MAKE THIS DIGIT ASCII	
047704	052703	000040	5\$:	BIS	#',R3	::MAKE ASCII IF NOT ALREADY	
047710	122703	000040		CMPB	#',R3	::IS THIS A SPACE CHARACTER?	:DPM002
047714	001404			BEQ	7\$::BRANCH IF SO - DON'T TYPE	:DPM002
047716	110337	047762		MOVB	R3,8\$::SAVE FOR TYPING	
047722	104401	047762		TYPE	.8\$::GO TYPE THIS DIGIT	
047726	105337	047764	7\$:	DECB	\$OCNT	::COUNT BY 1	
047732	003344			BGT	2\$::BR IF MORE TO DO	
047734	002402			BLT	6\$::BR IF DONE	
047736	005204			INC	R4	::INSURE LAST DIGIT ISN'T A BLANK	
047740	000741			BR	2\$::GO DO THE LAST DIGIT	
047742	012605		6\$:	MOV	(SP)+,R5	::RESTORE R5	
047744	012604			MOV	(SP)+,R4	::RESTORE R4	
047746	012603			MOV	(SP)+,R3	::RESTORE R3	
047750	016666	000002 000004		MOV	2(SP),4(SP)	::SET THE STACK FOR RETURNING	
047756	012616			MOV	(SP)+,(SP)		
047760	000002			RTI		::RETURN	
047762	000		8\$:	.BYTE	0	::STORAGE FOR ASCII DIGIT	
047763	000			.BYTE	0	::TERMINATOR FOR TYPE ROUTINE	
047764	000		\$OCNT:	.BYTE	0	::OCTAL DIGIT COUNTER	
047765	000		\$OFILL:	.BYTE	0	::ZERO FILL SWITCH	
047766	000000		\$OMODE:	.WORD	0	::NUMBER OF DIGITS TO TYPE	

8486

.SBTTL APT COMMUNICATIONS ROUTINE

```

*****
047770 112737 000001 050234 $ATY1: MOVB #1,$FFLG ;;TO REPORT FATAL ERROR
047776 112737 000001 050232 $ATY3: MOVB #1,$MFLG ;;TO TYPE A MESSAGE
050004 000403 000001 050234 BR $ATYC
050006 112737 000001 050234 $ATY4: MOVB #1,$FFLG ;;TO ONLY REPORT FATAL ERROR
050014 $ATYC:
050014 010046 MOV R0,-(SP) ;;PUSH R0 ON STACK
050016 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
050020 105737 050232 TSTB $MFLG ;;SHOULD TYPE A MESSAGE?
050024 001450 BEQ 5$ ;;IF NOT: BR
050026 122737 000001 001336 CMPB #APTENV,$ENV ;;OPERATING UNDER APT?
050034 001031 BNE 3$ ;;IF NOT: BR
050036 132737 000100 001337 BITB #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
050044 001425 BEQ 3$ ;;IF NOT: BR
050046 017600 000004 MOV @4(SP),R0 ;;GET MESSAGE ADDR.
050052 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDR.
050060 005737 001316 1$: TST $MSGTYPE ;;SEE IF DONE W/ LAST XMISSION?
050064 001375 BNE 1$ ;;IF NOT: WAIT
050066 010037 001332 MOV R0,$MSGAD ;;PUT ADDR IN MAILBOX
050072 105720 2$: TSTB (R0)+ ;;FIND END OF MESSAGE
050074 001376 BNE 2$
050076 163700 001332 SUB $MSGAD,R0 ;;SUB START OF MESSAGE
050102 006200 ASR R0 ;;GET MESSAGE LNTH IN WORDS
050104 010037 001334 MOV R0,$MSGGLT ;;PUT LENGTH IN MAILBOX
050110 012737 000004 001316 MOV #4,$MSGTYPE ;;TELL APT TO TAKE MSG.
050116 000413 BR 5$
050120 017637 000004 050144 3$: MOV @4(SP),4$ ;;PUT MSG ADDR IN JSR LINKAGE
050126 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDRESS
050134 013746 177776 MOV 177776,-(SP) ;;PUSH 177776 ON STACK
050140 004737 047174 JSR PC,$TYPE ;;CALL TYPE MACRO
050144 000000 4$: .WORD 0
050146 5$:
050146 105737 050234 10$: TSTB $FFLG ;;SHOULD REPORT FATAL ERROR?
050152 001416 BEQ 12$ ;;IF NOT: BR
050154 005737 001336 TST $ENV ;;RUNNING UNDER APT?
050160 001413 BEQ 12$ ;;IF NOT: BR
050162 005737 001316 11$: TST $MSGTYPE ;;FINISHED LAST MESSAGE?
050166 001375 BNE 11$ ;;IF NOT: WAIT
050170 017637 000004 001320 MOV @4(SP),$FATAL ;;GET ERROR #
050176 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDR.
050204 005237 001316 INC $MSGTYPE ;;TELL APT TO TAKE ERROR
050210 105037 050234 12$: CLRB $FFLG ;;CLEAR FATAL FLAG
050214 105037 050233 CLRB $LFLG ;;CLEAR LOG FLAG
050220 105037 050232 CLRB $MFLG ;;CLEAR MESSAGE FLAG
050224 012601 MOV (SP)+,R1 ;;POP STACK INTO R1
050226 012600 MOV (SP)+,R0 ;;POP STACK INTO R0
050230 000207 RTS PC ;;RETURN
050232 000 $MFLG: .BYTE 0 ;;MESSG. FLAG
050233 000 $LFLG: .BYTE 0 ;;LOG FLAG
050234 000 $FFLG: .BYTE 0 ;;FATAL FLAG
.EVEN
000200 APTSIZE=200
000001 APTENV=001
000100 APTSPOOL=100
000040 APTCSUP=040

```

8488

```

.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.
050236 022737 000176 001140 $CKSWR: CMP #SWREG,SWR ;; IS THE SOFT-SWR SELECTED?
050244 001074 BNE 15$ ;; BRANCH IF NO
050246 105777 130672 TSTB @STKS ;; CHAR THERE?
050252 100071 BPL 15$ ;; IF NO, DON'T WAIT AROUND
050254 117746 130666 MOVB @STKB,-(SP) ;; SAVE THE CHAR
050260 042716 177600 BIC #^C177,(SP) ;; STRIP-OFF THE ASCII
050264 022726 000007 CMP #7,(SP)+ ;; IS IT A CONTROL G?
050270 001062 BNE 15$ ;; NO, RETURN TO USER
050272 123727 001134 000001 CMPB $AUTOB,#1 ;; ARE WE RUNNING IN AUTO-MODE?
050300 001456 BEQ 15$ ;; BRANCH IF YES
050302 104401 050655 TYPE , $CNTLG ;; ECHO THE CONTROL-G (^G)
050306 104401 050662 $GTSWR: TYPE , $MSWR ;; TYPE CURRENT CONTENTS
050312 013746 000176 MOV SWREG,-(SP) ;; SAVE SWREG FOR TYPEOUT
050316 104402 TYPOC ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
050320 104401 050673 TYPE , $MNEW ;; PROMPT FOR NEW SWR
050324 005046 19$: CLR -(SP) ;; CLEAR COUNTER
050326 005046 CLR -(SP) ;; THE NEW SWR
050330 105777 130610 7$: TSTB @STKS ;; CHAR THERE?
050334 100375 BPL 7$ ;; IF NOT TRY AGAIN
050336 117746 130604 MOVB @STKB,-(SP) ;; PICK UP CHAR
050342 042716 177600 BIC #^C177,(SP) ;; MAKE IT 7-BIT ASCII
050346 021627 000025 9$: CMP (SP),#25 ;; IS IT A CONTROL-U?
050352 001005 BNE 10$ ;; BRANCH IF NOT
050354 104401 050650 TYPE , $CNTLU ;; YES, ECHO CONTROL-U (^U)
050360 062706 000006 20$: ADD #6,SP ;; IGNORE PREVIOUS INPUT
050364 000757 BR 19$ ;; LET'S TRY IT AGAIN
050366 021627 000015 10$: CMP (SP),#15 ;; IS IT A <CR>?
050372 001022 BNE 16$ ;; BRANCH IF NO
050374 005766 000004 TST 4(SP) ;; YES, IS IT THE FIRST CHAR?
050400 001403 BEQ 11$ ;; BRANCH IF YES
050402 016677 000002 130530 MOV 2(SP),@SWR ;; SAVE NEW SWR
050410 062706 000006 11$: ADD #6,SP ;; CLEAR UP STACK
050414 104401 001313 14$: TYPE , $CRLF ;; ECHO <CR> AND <LF>
050420 123727 001135 000001 CMPB $INTAG,#1 ;; RE-ENABLE TTY KBD INTERRUPTS?
050426 001003 BNE 15$ ;; BRANCH IF NOT
050430 012777 000100 130506 MOV #100,@STKS ;; RE-ENABLE TTY KBD INTERRUPTS
050436 000002 15$: RTI ;; RETURN
050440 004737 047412 16$: JSR PC,$TYPEC ;; ECHO CHAR
050444 021627 000060 CMP (SP),#60 ;; CHAR < 0?
050450 002420 BLT 18$ ;; BRANCH IF YES
050452 021627 000067 CMP (SP),#67 ;; CHAR > 7?
050456 003015 BGT 18$ ;; BRANCH IF YES
050460 042726 000060 BIC #60,(SP)+ ;; STRIP-OFF ASCII
050464 005766 000002 TST 2(SP) ;; IS THIS THE FIRST CHAR
050470 001403 BEQ 17$ ;; BRANCH IF YES
050472 006316 ASL (SP) ;; NO, SHIFT PRESENT
050474 006316 ASL (SP) ;; CHAR OVER TO MAKE
050476 006316 ASL (SP) ;; ROOM FOR NEW ONE.
050500 005266 000002 17$: INC 2(SP) ;; KEEP COUNT OF CHAR

```

```
050504 056616 177776          BIS      -2(SP),(SP)    ::SET IN NEW CHAR
050510 000707          BR       7$           ::GET THE NEXT ONE
050512 104401 001312    18$:     TYPE      $QUES    ::TYPE ?<CR><LF>
050516 000720          BR       20$          ::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
          *
050520 011646          $RDCHR: MOV      (SP),-(SP)    ::PUSH DOWN THE PC
050522 016666 000004 000002    MOV      4(SP),2(SP)    ::SAVE THE PS
050530 105777 130410    1$:     TSTB     @$TKS      ::WAIT FOR
050534 100375          BPL      1$           ::A CHARACTER
050536 117766 130404 000004    MOVB     @$TKB,4(SP)    ::READ THE TTY
050544 042766 177600 000004    BIC      #^C<17>,4(SP) ::GET RID OF JUNK IF ANY
050552 026627 000004 000023    CMP      4(SP),#23     ::IS IT A CONTROL-S?
050560 001013          BNE      3$           ::BRANCH IF NO
050562 105777 130356    2$:     TSTB     @$TKS      ::WAIT FOR A CHARACTER
050566 100375          BPL      2$           ::LOOP UNTIL ITS THERE
050570 117746 130352    MOVB     @$TKB,-(SP)   ::GET CHARACTER
050574 042716 177600    BIC      #^C177,(SP)  ::MAKE IT 7-BIT ASCII
050600 022627 000021    CMP      (SP)+,#21     ::IS IT A CONTROL-Q?
050604 001366          BNE      2$           ::IF NOT DISCARD IT
050606 000750          BR       1$           ::YES, RESUME
050610 026627 000004 000021    3$:     CMP      4(SP),#$XON ::IS IT A RANDOM XON?
050616 001744          BEQ     1$           ::BRANCH IF YES
050620 026627 000004 000140    CMP      4(SP),#140   ::IS IT 'JPPER CASE?
050626 002407          BLT     4$           ::BRANCH IF YES
050630 026627 000004 000175    CMP      4(SP),#175   ::IS IT A SPECIAL CHAR?
050636 003003          BGT     4$           ::BRANCH IF YES
050640 042766 000040 000004    BIC      #40,4(SP)    ::MAKE IT UPPER CASE
050646 000002          RTI          ::GO BACK TO USER
050650          136      125      015  $CNTLU: .ASCIZ /^U/<15><12> ::CONTROL 'U'
050655          136      107      015  $CNTLG: .ASCIZ /^G/<15><12> ::CONTROL 'G'
050662          015      012      123  $MSWR:  .ASCIZ <15><12>/SWR = /
050673          040      040      116  $MNEW:  .ASCIZ / NEW = /
```

8490

.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.

050704 010046
 050706 016600 000002
 050712 005740
 050714 111000
 050716 006300
 050720 016000 050740
 050724 000200

```
$TRAP:  MOV    RO,-(SP)      ;;SAVE RO
        MOV    2(SP),RO    ;;GET TRAP ADDRESS
        TST   -(RO)       ;;BACKUP BY 2
        MOVJ  (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 USF 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

050726 011646
 050730 016666 000004 000002
 050736 000002

.SBTTL TRAP TABLE

*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
 *BY THE 'TRAP' INSTRUCTION.
 : ROUTINE
 :-----

050740 050726
 050742 047174
 050744 047560
 050746 047534
 050750 047574
 050752 050306
 050754 050236
 050756 050520
 050760 047076
 050762 047134
 8491 050764 052050
 8492 050764 000026

```
$TRPAD: .WORD  $TRAP2
        $TYPE  ;;CALL=TYPE      TRAP+1(104401) TTY TIMEOUT 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
```

\$TERM=-.\$TRPAD

8494

.SBTTL POWER DOWN AND UP ROUTINES

```

*****
:POWER DOWN ROUTINE
050766 012737 051144 000024 $PWRDN: MOV    #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
050774 012737 000340 000026      MOV    #340,@#PWRVEC+2 ;;PRIO:7
051002 010046      MOV    R0,-(SP)      ;;PUSH R0 ON STACK
051004 010146      MOV    R1,-(SP)      ;;PUSH R1 ON STACK
051006 010246      MOV    R2,-(SP)      ;;PUSH R2 ON STACK
051010 010346      MOV    R3,-(SP)      ;;PUSH R3 ON STACK
051012 010446      MOV    R4,-(SP)      ;;PUSH R4 ON STACK
051014 010546      MOV    R5,-(SP)      ;;PUSH R5 ON STACK
051016 017746 130116      MOV    @SWR,-(SP)    ;;PUSH @SWR ON STACK
051022 010637 051150      MOV    SP,$SAVR6    ;;SAVE SP
051026 012737 051040 000024      MOV    #$PWRUP,@#PWRVEC ;;SET UP VECTOR
051034 000000      HALT
051036 000776      BR     .-2          ;;HANG UP
*****
:POWER UP ROUTINE
051040 012737 051144 000024 $PWRUP: MOV    #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
051046 013706 051150      MOV    $SAVR6,SP    ;;GET SP
051052 005037 051150      CLR    $SAVR6      ;;WAIT LOOP FOR THE TTY
051056 005237 051150      1$:  INC    $SAVR6    ;;WAIT FOR THE INC
051062 001375      BNE   $SAVR6      ;;OF WORD
051064 012677 130050      MOV    (SP)+,@SWR   ;;POP STACK INTO @SWR
051070 012605      MOV    (SP)+,R5     ;;POP STACK INTO R5
051072 012604      MOV    (SP)+,R4     ;;POP STACK INTO R4
051074 012603      MOV    (SP)+,R3     ;;POP STACK INTO R3
051076 012602      MOV    (SP)+,R2     ;;POP STACK INTO R2
051100 012601      MOV    (SP)+,R1     ;;POP STACK INTO R1
051102 012600      MOV    (SP)+,R0     ;;POP STACK INTO R0
051104 012737 050766 000024      MOV    #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
051112 012737 000340 000026      MOV    #340,@#PWRVEC+2 ;;PRIO:7
051120 104401      TYPE
051122 052120      $PWRMG: .WORD  POWERM    ;;REPORT THE POWER FAILURE
051124 012716      MOV    (PC)+,(SP)   ;;POWER FAIL MESSAGE POINTER
051126 006116      $PWRAD: .WORD  START    ;;RESTART AT START
051130 042766 000020 000002      BIC    #20,2(SP)    ;;RESTART ADDRESS
051136 005037 046156      CLR    $TBIT       ;;CLEAR 'T' BIT
051142 000002      RTI                ;;CLEAR THE 'T' BIT FLAG
051144 000000      $!LLUP: HALT      ;;THE POWER UP SEQUENCE WAS STARTED
051146 000776      BR     .-2          ;; BEFORE THE POWER DOWN WAS COMPLETE
051150 000000      $SAVR6: 0          ;;PUT THE SP HERE

```

8496
8497
8498

8499
8500
8501
8502
8503
8504
8505 051152 104401
8506 051154 001313
8507 051156 113737 001102 001232
8508 051164 042737 177400 001232
8509 051172 013737 001116 001234
8510 051200 010046
8511 051202 117700 130026
8512 051206 042700 177400
8513 051212 001006
8514 051214 013746 001116
8515 051220 104402
8516 051222 104401 001313
8517 051226 000561
8518 051230 022700 000377
8519 051234 001004
8520 051236 017600 000004
8521 051242 062700 000400
8522 051246 010037 051576
8523 051252 005300
8524 051254 006300
8525 051256 006300
8526 051260 006300
8527 051262 062700 001442
8528 051266 012037 051276
8529 051272 001406
8530 051274 104401
8531 051276 000000
8532 051300 004737 051600
8533 051304 104401 001313
8534 051310 012037 051320
8535 051314 001406
8536 051316 104401
8537 051320 000000
8538 051322 004737 051600
8539 051326 104401
8540 051330 001313
8541 051332 010146
8542 051334 010246
8543 051336 010346
8544 051340 012001
8545 051342 001506
8546 051344 011000
8547 051346 105710
8548 051350 001003
8549 051352 013146
8550 051354 104402
8551 051356 000473

```

.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
*      RO,-(SP) ;SAVE RO
*      MOVB  @$TMP1,RO ;GET THE ITEM NUMBER.
*      BIC   #177400,RO
*      BNE   21$
*      MOV   $ERRPC,-(SP) ;MOVE ERROR PC TO STACK FOR PRINTING
*      TYPOC ;GO TYPE THE ERROR PC
*      TYPE  , $CRLF ;TYPE A <CRLF>
*      BR    ERT5 ;BRANCH TO EXIT
21$:  *      CMP   #377,RO
*      BNE   20$
*      MOV   @4(SP),RO
*      ADD   #400,RO
20$:  *      MOV   RO,ERRNUM ;SAVE THE ERROR # FOR POSSIBLE USE ;DPM001
*      DEC   RO ;OTHERWISE MAKE RO AN
*      ASL   RO ;INDEX FOR THE TABLE.
*      ASL   RO
*      ASL   RO
*      ADD   #$ERRTB,RO
22$:  *      MOV   (RO)+,2$ ;PICK UP THE ADDRESS
*      BEQ   3$ ;OF THE EM, ERROR MESSAGE
2$:  *      TYPE
*      .WORD  0
*      JSR   PC,PASCIZ ;GO CHECK FOR AND PRINT ANY ADDTL ASCIZ ;DPM001
*      TYPE  , $CRLF
3$:  *      MOV   (RO)+,4$ ;GET THE DH,DATA HEADER
*      BEQ   5$
4$:  *      TYPE
*      .WORD  0
*      JSR   PC,PASCIZ ;GO CHECK FOR AND PRINT ANY ADDTL ASCIZ ;DPM001
*      TYPE
*      .WORD  $CRLF
5$:  *      MOV   R1,-(SP) ;SAVE R1,R2 AND R3
*      MOV   R2,-(SP)
*      MOV   R3,-(SP)
*      MOV   (RO)+,R1 ;GET THE ADDRESS OF THE DATA TABLE.
*      BEQ   ERT4 ;RETURN IF NO DATA.
*      MOV   (RO),RO ;GET A POINTER TO THE DATA FORMAT TABLE.
ERT1: *      TSTB  (RO) ;FORMAT ZERO?
*      BNE   7$
*      MOV   @ (R1)+,-(SP) ;FORMAT ZERO SO TYPE
*      TYPOC ;AN OCTAL NUMBER.
*      BR    ERT2

```

```

8552 051360 122710 000002 7$:  CMPB  #2,(R0)      ;FORMAT TWO?
8553 051364 001010      BNE  9$
8554 051366 013102      MOV  @ (R1)+,R2      ;FORMAT TWO SO TYPE TWO
8555 051370 012246      MOV  (R2)+,-(SP)    ;OCTAL NUMBERS.
8556 051372 104402      TYPOC
8557 051374 104401      TYPE
3558 051376 052164      .WORD  SPACE
8559 051400 011246      MOV  (R2)+,-(SP)
8560 051402 104402      TYPOC
8561 051404 000460      BR   ERT2
8562 051406 122710 000003 9$:  CMPB  #3,(R0)      ;FORMAT THREE?
8563 051412 001011      BNE  10$
8564 051414 013102      MOV  @ (R1)+,R2      ;FORMAT THREE SO TYPE 4 OCTAL NUMBERS.
8565 051416 012703 000004      MOV  #4,R3          ;LOOP COUNTER
8566 051422 012246      MOV  (R2)+,-(SP)    ;MOVE DATA TO THE STACK
8567 051424 104402      TYPOC              ;TYPE AN OCTAL NUMBER
8568 051426 104401      TYPE              ;TYPE A SPACE
8569 051430 052164      .WORD  SPACE
8570 051432 077305      SOB  R3,90$        ;SUBTRACT 1 AND BRANCH IF NOT DONE YET ;DPM001
8571 051434 000444      BR   ERT2          ;EXIT
8572 051436 122710 000004 10$:  CMPB  #4,(R0)      ;FORMAT FOUR?
8573 051442 001004      BNE  11$
8574 051444 013146      MOV  @ (R1)+,-(SP)    ;FORMAR FOUR SO TYPE
8575 051446 104403      TYPOS             ;AN OCTAL NUMBER
8576 051450 016        .BYTE  16          ;SUPPRESSING LEADING ZEROES.
8577 051451 000        .BYTE  0
8578 051452 000435      BR   ERT2
8579 051454 122710 000005 11$:  CMPB  #5,(R0)      ;FORMAT FIVE?
8580 051460 001005      BNE  13$
8581 051462 012137 051470      MOV  (R1)+,12$      ;FORMAT FIVE SO TYPE AN
8582 051466 104401      TYPE              ;ASCIZ STRING.
8583 051470 000000      .WORD  0
8584 051472 000427      BR   ERT3
8585 051474 122710 000011 13$:  CMPB  #11,(R0)     ;FORMAT ELEVEN?
8586 051500 001005      BNE  15$
8587 051502 013137 051510      MOV  @ (R1)+,14$    ;FORMAT ELEVEN SO PICK
8588 051506 104401      TYPE              ;A POINTER TO AN ASCIZ
8589 051510 000000      .WORD  0           ;STRING.
8590 051512 000417      BR   ERT3
8591 051514 122710 000012 15$:  CMPB  #12,(R0)     ;FORMAT TWELVE?
8592 051520 001011      BNE  17$
8593 051522 013102      MOV  @ (R1)+,R2      ;FORMAT TWELVE SO TYPE
8594 051524 012703 000006      MOV  #6,R3          ;TYPE SIX OCTAL NUMBERS
8595 051530 012246      MOV  (R2)+,-(SP)
8596 051532 104402      TYPOC
8597 051534 104401      TYPE
8598 051536 052164      .WORD  SPACE
8599 051540 077305      SOB  R3,16$
8600 051542 000401      BR   ERT2
8601 051544 000000      17$:  HALT
8602 051546 104401      ERT2:  TYPE        ;UNDEFINED FORMAT FOR DATA????
8603 051550 052167      .WORD  $TAB        ;PRINT A TAB AFTER TYPING A DATA TABLE ENTRY
8604 051552 005200      ERT3:  INC  R0      ;OF ALL FORMATS EXCEPT ASCIZ, FORMATS 5 OR 11
8605 051554 005711      TST  (R1)          ;POINT TO THE NEXT FORMAT
8606 051556 001273      BNE  ERT1          ;END OF DATA TABLE.
8607 051560 104401      ERT4:  TYPE        ;DONE.
8608 051562 001313      .WORD  $CRLF

```

CKFPCDO FP11F FLTG PNT PRT C
ERROR TYPE OUT ROUTINE

MACRO M1113 30-OCT-81 11:15 PAGE 110-2

SEQUENCE 224

8609 051564 012603
8610 051566 012602
8611 051570 012601
8612 051572 012600
8613 051574 000207
8614
8615 051576 000000

MOV (SP)+,R3 ;RESTORE R1,R2 AND R3
MOV (SP)+,R2
MOV (SP)+,R1
ERT5: MOV (SP)+,RO ;RESTORE RO.
RTS PC ;AND RETURN.
ERRNUM: .WORD 0 ;LOCATION TO HOLD CURRENT ERROR NUMBER ;DPM001


```

      .SBTTL SUBROUTINE TO LOOK FOR ANY ADDITIONAL MSGS TO PRINT
8616      PASCIZ: MOV R4, -(SP) ;SAVE R4 ;DPM001
8617 051600 010446 1$: MOV EQUASCII, R4 ;MOVE END OF ASCII ADDRESS TO R4 ;DPM001
8618 051602 013704 047172 CMPB #377, (R4) ;SEE IF ADD'L MSGS NEED TO BE PRINTED ;DPM001
8619 051606 122714 000377 BNE 4$ ;BRANCH IF NOT ;DPM001
8620 051612 001013 INC R4 ;INCREMENT TO NEXT BYTE ;DPM001
8621 051614 005204 BIT #BIT00, R4 ;SEE IF ODD ADDRESS ;DPM001
8622 051616 032704 000001 BEQ 2$ ;BRANCH IF NOT ;DPM001
8623 051622 001401 INC R4 ;POINT TO WORD ADDRESS ;DPM001
8624 051624 005204 2$: MOV (R4)+, 3$ ;MOVE ADDRESS TO LOCATION FOR TYPING ;DPM001
8625 051626 012437 051636 BEQ 1$ ;GO CHK FOR MORE MSG ADR'S IF NO MORE ;DPM001
8626 051632 001763 TYPE ;TYPE THE MESSAGE, ADDRESS IN NEXT WORD ;DPM001
8627 051634 104401 .WORD 0 ;LOCATION FOR MESSAGE ADDRESS ;DPM001
8628 051636 000000 3$: BR 2$ ;BRANCH BACK TO LOAD OTHER MESSAGES ;DPM001
8629 051640 000772 4$: CMPB #376, (R4) ;SEE IF SPECIFIC MSGS NEED PRINTING ;DPM001
8630 051642 122714 000376 BNE 11$ ;BRANCH TO EXIT IF NOT ;DPM001
8631 051646 001027 INC R4 ;INCREMENT TO NEXT BYTE ;DPM001
8632 051650 005204 BIT #BIT00, R4 ;SEE IF ODD ADDRESS ;DPM001
8633 051652 032704 000001 BEQ 5$ ;BRANCH IF NOT ;DPM001
8634 051656 001401 INC R4 ;POINT TO WORD ADDRESS ;DPM001
8635 051660 005204 5$: CLR -(SP) ;CLEAR COUNTER LOCATION ON STACK ;DPM001
8636 051662 005046 6$: INC (SP) ;INCREMENT COUNTER ;DPM001
8637 051664 005216 051576 CMP ERRNUM, (R4)+ ;SEE IF ITEM NUMBER MATCHES ERROR ;DPM001
8638 051666 023724 BNE 6$ ;BRANCH IF NOT TO SEARCH AGAIN ;DPM001
8639 051672 001374 7$: TST (R4)+ ;SEE IF AT TERMINATOR YET ;DPM001
8640 051674 005724 BNE 7$ ;BRANCH BACK UNTIL IT IS ;DPM001
8641 051676 001376 TST -(R4) ;PUT R4 AT THE TERMINATOR ;DPM001
8642 051700 005744 8$: TST (R4)+ ;POINT R4 TO NEXT ADDRESS ;DPM001
8643 051702 005724 DEC (SP) ;SEE IF COUNTER DOWN TO ZERO YET ;DPM001
8644 051704 005316 BNE 8$ ;BRANCH BACK IF NOT TO ADVANCE R4 ;DPM001
8645 051706 001375 9$: TST (SP)+ ;POP COUNTER OFF STACK ;DPM001
8646 051710 005726 MOV (R4), 10$ ;MOVE ADDRESS TO LOCATION TO TYPE ;DPM001
8647 051712 011437 051722 BEQ 11$ ;BRANCH IF NO EXTRA MESSAGE TO PRINT ;DPM001
8648 051716 001403 TYPE ;TYPE THE ASCII MESSAGE ;DPM001
8649 051720 104401 .WORD 0 ;LOCATION FOR ASCII MESSAGE ADDRESS ;DPM001
8650 051722 000000 10$: BR 1$ ;GO BACK TO MAKE SURE NO MORE MSGS ;DPM001
8651 051724 000726 11$: MOV (SP)+, R4 ;RESTORE R4 ;DPM001
8652 051726 012604 RTS PC ;EXIT ;DPM001
8653 051730 000207

```

8654
8655

.SBTTL FPP SPURIOUS TRAP TO 244 HANDLER

*THIS ROUTINE HANDLES UNEXPECTED TRAPS TO THE FPP TRAP VECTOR AT 244.
*THE LAST FPP INSTRUCTION EXECUTED AND ITS ADDRESS HAS BEEN RECORDED
*THESE ALONG WITH THE FEC, FPS AND PC OF TRAP ARE REPORTED.

8656
8657
8658
8659

8660 051732 042737 000001 177572
8661 051740 011637 001236
8662 051744 022626
8663 051746 170200
8664 051750 010037 001240
8665 051754 170300
8666 051756 010037 001242
8667 051762 104377
8668 051764 000041
8669 051766 104412

*
FPPUR: BIC #BIT00,MMRO ;MAKE SURE MEMORY MANAGEMENT IS OFF ;DPM001
MOV (SP),\$TMP2 ;SAVE PC OF TRAP.
CMP (SP)+,(SP)+ ;RESTORE SP.
STFPS RO ;GET FPS
MOV RO,\$TMP3 ;MOVE IT TO \$TMP3
STST RO ;GET FEC
MOV RO,\$TMP4 ;MOVE IT TO \$TMP4
ERROR +377 ;CALL ERROR
.WORD 41 ;ERROR #441
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

8670 051770 000137 045420

8671
8672

.SBTTL CPU SPURIOUS TRAP TO 4 HANDLER

*THIS ROUTINE REPORTS UNEXPECTED CPU TRAPS TO VECTOR 4.

8673
8674

8675 051774 042737 000001 177572
8676 052002 011637 001236
8677 052006 022626
8678 052010 104377
8679 052012 000042
8680 052014 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?).
;JUMP TO \$EOP

8681 052016 000137 045420

JMP \$EOP

8682
8683

.SBTTL CPU SPURIOUS TRAP TO 10 HANDLER
:*****
:*****
:*THIS ROUTINE REPORTS UNEXPECTED CPU TRAPS TO VECTOR 10.
:*

8684
8685

8686 052022 042737 000001 177572
8687 052030 011637 001236
8688 052034 022626
8689 052036 104377
8690 052040 000043
8691 052042 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

8692 052044 000137 045420

JMP \$EOP

8693
8694

.SBTTL FLAG RESET AND CONSOLE TEST ROUTINE
 :*****
 :*****

8695
8696
8697
8698
8699
8700
8701

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

8702 052050 023727 001140 177570
 8703 052056 001001
 8704 052060 104406
 8705
 8706 052062 012737 051732 000244
 8707 052070 012737 051774 000004
 8708 052076 012737 052022 000010
 8709 052104 011600
 8710 052106 012706 001100
 8711 052112 005004
 8712 052114 170104
 8713 052116 000110

.RSET: CMP SWR,#177570 ;SEE IF THERE IS A PHYSICAL CONSOLE SWITCH REGISTER.
 BNE 1\$;BRANCH IF NOT
 CKSWR ;OTHERWISE TYPE THE CONTENTS OF THE PROGRAM VIRTUAL
 ;SWITCH REGISTER AND GIVE THE USER CHANCE TO MODIFY IT
 1\$: MOV #FPSPUR,FPVECT
 MOV #CPSPUR,ERRVECT
 MOV #CPTWO,10
 MOV (SP),R0 ;SAVE RETURN ADDRESS.
 MOV #STACK,SP ;RESET THE STACK POINTER.
 CLR R4 ;CLEAR THE FPS.
 LDFPS R4
 JMP (R0) ;RETURN.

8714					.SBTTL	SPECIAL MESSAGES
8715	052120	200	120	117	POWERM: .ASCIZ	<CRLF>'POWER FAILURE. PROGRAM RESTARTING.'
8716	052164	040	040	000	SPACE: .ASCIZ	' '
8717	052167	011	000		\$TAB: .ASCIZ	<TAB>
8718	052171	107	117	124	MS1: .ASCIZ	'GOT RESULT:'<TAB><TAB>
8719	052207	105	130	120	MS2: .ASCIZ	'EXPECTED RESULT:'<TAB>
8720	052231	101	103	040	MS3: .ASCIZ	'AC OPERAND:'<TAB><TAB>
8721	052247	123	117	125	MS4: .ASCIZ	'SOURCE OPERAND: '<TAB>
8722		052231			MS10=MS3	
8723	052271	105	130	120	MS11: .ASCIZ	'EXPONENT OPERAND:'<TAB>
8724	052314	114	117	101	MS20: .ASCIZ	'LOADED:'<TAB><TAB>
8725	052326	124	122	111	MS21: .ASCIZ	'TRIED TO LOAD:'<TAB>

Line	Address	Page	Page	Page	Page	Label	Code	Message
8726								.SBTTL ERROR MESSAGES
8746	052346	123	000			EM1:	.ASCIZ	:S:
8747	052350	377					.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
							.EVEN	
8748	052352	052362	052373	052412			.WORD	EM1B,EM1C,EM1D,0
8749	052362	124	106	040	EM1B:	.ASCIZ	:TF A,AC7:	
8750	052373	040	104	111	EM1C:	.ASCIZ	:DID NOT TRAP.:	
8751	052412	040	106	111	EM1D:	.ASCIZ	:FID=0.:	
8752	052422	123	000		EM2:	.ASCIZ	:S:	
8753	052424	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
							.EVEN	
8754	052426	052362	052436	052412			.WORD	EM1B,EM2B,EM1D,0
8755	052436	056	040	106	EM2B:	.ASCIZ	:FPS BAD.:	
8756	052452	123	000		EM3:	.ASCIZ	:S:	
8757	052454	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
							.EVEN	
8758	052456	052362	052466	052412			.WORD	EM1B,EM3B,EM1D,0
8759	052466	056	040	106	EM3B:	.ASCIZ	:FEC BAD.:	
8760	052502	123	000		EM4:	.ASCIZ	:S:	
8761	052504	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
							.EVEN	
8762	052506	052520	052531	052543			.WORD	EM4A,EM4B,EM4C,EM4D,0
8763	052520	124	106	040	EM4A:	.ASCIZ	:TF A,(R):	
8764	052531	056	040	122	EM4B:	.ASCIZ	:RO BAD.:	
8765	052543	056	040	106	EM4C:	.ASCIZ	:FDST:	
8766	052552	040	106	101	EM4D:	.ASCIZ	:FAILED.:	
8767	052563	123	000		EM5:	.ASCIZ	:S:	
8768	052565	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
							.EVEN	
8769	052566	052520	052552	000000			.WORD	EM4A,EM4D,0
8770	052574	123	000		EM6:	.ASCIZ	:S:	
8771	052576	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
							.EVEN	
8772	052600	052520	052543	052552			.WORD	EM4A,EM4C,EM4D,EM6C
8773	052610				EM6C:			
	052610	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
							.EVEN	
	052612	052653	052650	052662			.WORD	CRBUT,IFD,PRST,N7,N0,N7,WENTTO,N2,N4,N5,INSTOF,N2,N4,N4,0
8774	052650	106	104	000	IFD:	.ASCIZ	:FD:	
8775	052653	200	050	102	CRBUT:	.ASCIZ	:<CRLF>:(BUT :	
8776	052662	051	040	123	PRST:	.ASCIZ	:) ST :	
8777	052670	040	127	105	WENTTO:	.ASCIZ	: WENT TO :	
8778	052702	040	111	116	INSTOF:	.ASCIZ	: INSTEAD OF :	
8779	052717	060	000		N0:	.ASCIZ	:0:	
8780	052721	061	000		N1:	.ASCIZ	:1:	
8781	052723	062	000		N2:	.ASCIZ	:2:	
8782	052725	063	000		N3:	.ASCIZ	:3:	
8783	052727	064	000		N4:	.ASCIZ	:4:	
8784	052731	065	000		N5:	.ASCIZ	:5:	
8785	052733	066	000		N6:	.ASCIZ	:6:	
8786	052735	067	000		N7:	.ASCIZ	:7:	
8787	052737	123	000		EM7:	.ASCIZ	:S:	
8788	052741	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
							.EVEN	
8789	052742	052520	052756	052531			.WORD	EM4A,EM7B,EM4B,EM4C,EM4D,0
8790	052756	053	056	000	EM7B:	.ASCIZ	:+:	
8791	052761	123	000		EM10:	.ASCIZ	:S:	

ERROR MESSAGES

8792	052763	377					.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
							.EVEN		
8793	052764	052520	052756	052552			.WORD	EM4A,EM7B,EM4D,0	
8794	052774	123	000		EM11:	.ASCIZ	!S!		
8795	052776	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN			
8796	053000	053012	052531	052543		.WORD	EM11B,EM4B,EM4C,EM4D,0		
8797	053012	124	104	040	EM11B:	.ASCIZ	!TD A,(R)+!		
8798	053024	123	000		EM12:	.ASCIZ	!S!		
8799	053026	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN			
8800	053030	053012	052552	000000		.WORD	EM11B,EM4D,0		
8801	053036	123	000		EM13:	.ASCIZ	!S!		
8802	053040	377				.BYTF	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN			
8803	053042	053050	053060	000000		.WORD	EM13A,EM13B,0		
8804	053050	124	104	040	EM13A:	.ASCIZ	!TD A,#N!		
8805	053060	040	124	122	EM13B:	.ASCIZ	! TRAP TO 4 IN FDST.!		
8806		053036			EM14=EM13				
8807	053104	123	000		EM15:	.ASCIZ	!S!		
8808	053106	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN			
8809	053110	053050	052552	000000		.WORD	EM13A,EM4D,0		
8810	053116	120	103	040	EM16:	.ASCIZ	!PC BAD AFTER S!		
8811	053135	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN			
8812	053136	053050	000000			.WORD	EM13A,0		
8813	053142	123	000		EM17:	.ASCIZ	!S!		
8814	053144	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN			
8815	053146	053154	053060	000000		.WORD	EM17B,EM13B,0		
8816	053154	124	104	040	EM17B:	.ASCIZ	!TD A,-(R)!		
8817	053166	123	000		EM20:	.ASCIZ	!S!		
8818	053170	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN			
8819	053172	053154	052531	052543		.WORD	EM17B,EM4B,EM4C,EM4D,0		
8820	053204	123	000		EM21:	.ASCIZ	!S!		
8821	053206	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN			
8822	053210	053154	052552	000000		.WORD	EM17B,EM4D,0		
8823		053204			EM22=EM21				
8824	053216	123	000		EM23:	.ASCIZ	!S!		
8825	053220	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN			
8826	053222	053230	053060	000000		.WORD	EM23B,EM13B,0		
8827	053230	124	104	040	EM23B:	.ASCIZ	!TD A,@(R)+!		
8828	053243	123	000		EM24:	.ASCIZ	!S!		
8829	053245	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN			
8830	053246	053230	052531	052543		.WORD	EM23B,EM4B,EM4C,EM4D,0		
8831	053260	123	000		EM25:	.ASCIZ	!S!		
8832	053262	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN			
8833	053264	053230	052552	000000		.WORD	EM23B,EM4D,0		
8834	053272	123	000		EM26:	.ASCIZ	!S!		
8835	053274	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN			

8836	053276	053304	053060	000000		.WORD	EM26B,EM13B,0
8837	053304	124	104	040	EM26B:	.ASCIZ	:TD A,@-(R):
8838	053317	123	000		EM27:	.ASCIZ	:S:
8839	053321	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8840	053322	053304	052531	052543		.WORD	EM26B,EM4B,EM4C,EM4D,0
8841	053334	123	000		EM30:	.ASCIZ	:S:
8842	053336	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8843	053340	053304	052552	000000		.WORD	EM26B,EM4D,0
8844	053346	123	000		EM31:	.ASCIZ	:S:
8845	053350	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8846	053352	053360	053060	000000		.WORD	EM31B,EM13B,0
8847	053360	124	104	040	EM31B:	.ASCIZ	:TD A,N(R):
8848	053372	123	000		EM32:	.ASCIZ	:S:
8849	053374	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8850	053376	053360	052531	052543		.WORD	EM31B,EM4B,EM4C,EM4D,0
8851	053410	123	000		EM33:	.ASCIZ	:S:
8852	053412	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8853	053414	053360	052552	000000		.WORD	EM31B,EM4D,0
8854	053422	123	000		EM34:	.ASCIZ	:S:
8855	053424	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8856	053426	053434	053060	000000		.WORD	EM34B,EM13B,0
8857	053434	124	104	040	EM34B:	.ASCIZ	:TD A,@N(R):
8858	053447	123	000		EM35:	.ASCIZ	:S:
8859	053451	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8860	053452	053434	052531	052543		.WORD	EM34B,EM4B,EM4C,EM4D,0
8861	053464	123	000		EM36:	.ASCIZ	:S:
8862	053466	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8863	053470	053434	052552	000000		.WORD	EM34B,EM4D,0
8864	053476	123	124	103	EM37:	.ASCIZ	:STCFD A,(R) FAILED.:
8865	053522	376				.BYTE	376 ;FLAGS 'ERTYPE' TO PRINT ERR # SPECIFIC ASCIZ MSGS, ADRS 3 LINES DOWN
						.EVEN	
8866	053524	000037	000042	000044		.WORD	37,42,44,45,47,50,0
8867	053542	000000	053616	053731		.WORD	0,EM42B,EM44B,EM45B,EM47B,EM50B
8868	053556	123	124	103	EM40:	.ASCIZ	:STCFD A,(R):
8869	053572	376				.BYTE	376 ;FLAGS 'ERTYPE' TO PRINT ERR # SPECIFIC ASCIZ MSGS, ADRS 3 LINES DOWN
						.EVEN	
8870	053574	000040	000041	000043		.WORD	40,41,43,46,0
8871	053606	052436	052466	053652		.WORD	EM2B,EM3B,EM43B,EM46B
8872		053556			EM41=EM40		
8873		053476			EM42=EM37		
8874	053616	200	111	116	EM42B:	.ASCIZ	<CRLF>:INVERT FDFL ST 767 FAILED.:
8875		053556			EM43=EM40		
8876	053652	056	040	106	EM43B:	.ASCIZ	:FPS BAD.:
8877	053665	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
	053666	052653	053724	052662		.WORD	CRBUT,IEZBT,PRST,N5,N6,NO,WENTTO,NO,N6,N1,INSTOF,N2,N6,N1,0
8878	053724	105	132	102	IEZBT:	.ASCIZ	:EZBT:
8879		053476			EM44=EM37		
8880	053731	200	114	117	EM44B:	.ASCIZ	<CRLF>:LOW ORDER BITS OF X11 DID NOT GET 0 ST 766.:

ERROR MESSAGES

Line	Address	Page	Code	Macro	Text
8881	053476			EM45=EM37	
8882	054006	200	050	102	EM45B: .ASCIZ <CRLF>:(BUT OP1C) ST 251 FAILED.'
8883	053556			EM46=EM40	
8884	054041	056	040	106	EM46B: .ASCIZ : . FPS BAD.:
8885	054054	377			.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
					.EVEN
	054056	052653	053724	052662	.WORD CRBUT, IEZBT, PRST, N4, N2, N1, WENTTO, N2, N6, N2, INSTOF, N0, N6, N2, 0
8886	054114	053476			EM47=EM37
8887	054114	040	000		EM47B: .ASCIZ : :
8888	054116	377			.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
					.EVEN
	054120	052653	052650	052662	.WORD CRBUT, IFD, PRST, N1, N1, N3, WENTTO, N4, N1, N5, INSTOF, N4, N1, N4, 0
8889	054156	053476			EM50=EM37
8890	054156	040	123	111	EM50B: .ASCIZ : SIGN BAD.:
8891	054171	377			.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
					.EVEN
	054172	052653	054230	052662	.WORD CRBUT, IENBT, PRST, N5, N6, N7, WENTTO, N0, N6, N0, INSTOF, N4, N6, N0, 0
8892	054230	105	116	102	IENBT: .ASCIZ : IENBT:
8893	054235	123	124	103	EM51: .ASCIZ : STCDF A, (R):
8894	054251	376			.BYTE 376 ;FLAGS 'ERTYPE' TO PRINT ERR # SPECIFIC ASCIZ MSGS, ADRS 3 LINES DOWN
					.EVEN
	054252	000051	000054	000055	.WORD 51, 54, 55, 60, 0
8896	054264	052552	054336	054402	.WORD EM4D, EM54B, EM55B, EM60B
8897	054274	123	124	104	EM52: .ASCIZ : STD A, (R):
8898	054306	376			.BYTE 376 ;FLAGS 'ERTYPE' TO PRINT ERR # SPECIFIC ASCIZ MSGS, ADRS 3 LINES DOWN
					.EVEN
	054310	000052	000053	000056	.WORD 52, 53, 56, 57, 61, 0
8900	054324	052436	052466	054473	.WORD EM2B, EM3B, EM56B, EM57B, EM61B
8901	054274				EM53=EM52
8902	054235				EM54=EM51
8903	054336	040	106	101	EM54B: .ASCIZ : FAILED. :<CRLF>:INVERT FDFL ST 767 FAILED.:
8904	054235				EM55=EM51
8905	054402	200	122	117	EM55B: .ASCIZ <CRLF>:ROUND ERROR, OR:
8906	054423	377			.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
					.EVEN
	054424	052653	054462	052662	.WORD CRBUT, IBKOUT, PRST, N4, N0, N0, WENTTO, N7, N6, N6, INSTOF, N7, N6, N7, 0
8907	054462	102	122	105	IBKOUT: .ASCIZ :BREAKOUT:
8908	054274				EM56=EM52
8909	054473	056	040	106	EM56B: .ASCIZ : . FPS BAD.:
8910	054506	377			.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
					.EVEN
	054510	052653	053724	052662	.WORD CRBUT, IEZBT, PRST, N4, N2, N1, WENTTO, N0, N6, N2, INSTOF, N2, N6, N2, 0
8911	054546	054274			EM57=EM52
8912	054546	040	106	120	EM57B: .ASCIZ : FPS BAD. FIV=0.:
8913	054567	377			.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
					.EVEN
	054570	052653	054626	052662	.WORD CRBUT, IFIV, PRST, N2, N6, N2, WENTTO, N1, N2, N3, INSTOF, N1, N0, N3, 0
8914	054626	106	111	126	IFIV: .ASCIZ :FIV:
8915	054235				EM60=EM51
8916	054632	040	106	101	EM60B: .ASCIZ : FAILED. FIV=1.:
8917	054652	377			.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
					.EVEN
	054654	052653	054626	052662	.WORD CRBUT, IFIV, PRST, N2, N6, N2, WENTTO, N1, N0, N3, INSTOF, N1, N2, N3, 0
8918	054712	054274			EM61=EM52
8919	054712	056	040	106	EM61B: .ASCIZ : . FPS BAD.:
8920	054725	377			.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
					.EVEN

ERROR MESSAGES

8921	054726	052653	054764	052662		.WORD	CRBUT,IFLAG,PRST,N1,N4,N7,WENTTO,N3,N6,N1,INSTOF,N3,N6,N5,0
8922	054764	106	114	101	IFLAG:	.ASCIZ	:FLAG:
8923	054771	123	124	103	EM62:	.ASCIZ	:STCF:
8923	054776	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
8924	055000	055010	052436	055020		.EVEN	
8925	055010	104	040	101	EM62A:	.WORD	EM62A,EM2B,EM62B,0
8926	055020	040	000		EM62B:	.ASCIZ	:D A,AC6:
8927	055022	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
8928	055024	052653	055062	052662		.EVEN	
8929	055062	106	104	123	ST:	.WORD	CRBUT,IFDST,PRST,N7,N6,N7,WENTTO,N5,N6,N7,INSTOF,N5,N7,N7,0
8930	055067	123	124	103	EM63:	.ASCIZ	:FDST:
8930	055074	377				.ASCIZ	:STCF:
8931	055076	055010	052466	000000		.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
8932	055104	103	114	122	EM64:	.EVEN	
8933	055147	377				.WORD	EM62A,EM3B,0
8934	055150	052552	000000			.ASCIZ	:CLRD (R) FAILED:<CRLF>:ZERO X11 AT ST 770:
8938	055154				EM65:	.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
8939	055177	103	114	122		.WORD	EM4D,0
8940	055200	000				.ASCIZ	:CLRD (R). FPS BAD.:
8941	055204	103	114	122	EM6:	.BYTE	0
8942	055206	055220	052531	052543		.ASCIZ	:CLR:
8943	055220	104	040	050	EM67:	.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
8944	055226					.EVEN	
8945	055226	103	114	122		.WORD	EM66B,EM4B,EM4C,EM4D,0
8945	055251	377				.ASCIZ	:D (R):
8946	055252	052653	055062	052662		.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
8947	055310	103	114	122	EM70:	.EVEN	
8948	055333	116	105	107	EM71:	.WORD	CRBUT,IFDST,PRST,N7,N7,N0,WENTTO,N6,N0,N7,INSTOF,N6,N1,N7,0
8948	055342	377				.ASCIZ	:CLRD AC7. FEC BAD.:
8949	055344	052552	000000			.ASCIZ	:NEGF A:
8950	055350	116	105	107	EM72:	.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
8951	055357	377				.EVEN	
8952	055360	052436	000000			.WORD	EM4D,0
8953	055364	116	105	107	EM73:	.ASCIZ	:NEGF A:
8954	055370	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
8955	055372	055220	052552	000000		.EVEN	
8956	055400	116	105	107	EM74:	.WORD	EM66B,EM4D,0
8957	055404	377				.ASCIZ	:NEGF:
8958	055406	055220	052531	000000		.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
8959	055414	116	105	107	EM75:	.EVEN	
8960	055420	377				.WORD	EM66B,EM4B,0
8961	055422	055220	052436	000000		.ASCIZ	:NEGF:
8962	055430	101	102	123	EM76:	.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
8963	055434	377				.EVEN	
8963	055434					.WORD	EM66B,EM2B,0
8963	055434					.ASCIZ	:ABS:
8963	055434					.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN

ERROR MESSAGES

8964	055436	055444	055453	000000		.EVEN	
8965	055444	104	040	050	EM76A:	.WORD	EM76A,EM76B,0
8966	055453	040	124	122	EM76B:	.ASCIZ	:D (R)+!
8967	055503	101	102	123	EM77:	.ASCIZ	: TRAP TO 4 IN SRC MODE.!
8968	055507	377				.ASCIZ	:ABS!
						.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8969	055510	055444	052552	000000		.WORD	EM76A,EM4D,0
8970	055516	101	102	123	EM100:	.ASCIZ	:ABS!
8971	055522	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8972	055524	055444	052531	000000		.WORD	EM76A,EM4B,0
8973	055532	101	102	123	EM101:	.ASCIZ	:ABS!
8974	055536	377				.BYTF	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8975	055540	055444	052436	000000		.WORD	EM76A,EM2B,0
8976	055546	101	102	123	EM102:	.ASCIZ	:ABS!
8977	055552	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8978	055554	055562	055453	000000		.WORD	EM102B,EM76B,0
8979	055562	104	040	055	EM102B:	.ASCIZ	:D -(R)!
8980	055571	101	102	123	EM103:	.ASCIZ	:ABS!
8981	055575	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8982	055576	055562	052552	000000		.WORD	EM102B,EM4D,0
8983	055604	101	102	123	EM104:	.ASCIZ	:ABS!
8984	055610	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8985	055612	055562	052531	000000		.WORD	EM102B,EM4B,0
8986	055620	101	102	123	EM105:	.ASCIZ	:ABS!
8987	055624	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8988	055626	055562	052436	000000		.WORD	EM102B,EM2B,0
8989	055634	101	102	123	EM106:	.ASCIZ	:ABS!
8990	055640	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8991	055642	055650	055453	000000		.WORD	EM106B,EM76B,0
8992	055650	104	040	100	EM106B:	.ASCIZ	:D @ (R)+!
8993	055660	116	105	107	EM107:	.ASCIZ	:NEG!
8994	055664	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8995	055666	055220	055453	000000		.WORD	EM66B,EM76B,0
8996	055674	101	102	123	EM110:	.ASCIZ	:ABS!
8997	055700	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
8998	055702	055650	052552	000000		.WORD	EM106B,EM4D,0
8999	055710	101	102	123	EM111:	.ASCIZ	:ABS!
9000	055714	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9001	055716	055650	052531	000000		.WORD	EM106B,EM4B,0
9002	055724	101	102	123	EM112:	.ASCIZ	:ABS!
9003	055730	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9004	055732	055650	052436	000000		.WORD	EM106B,EM2B,0
9005	055740	116	105	107	EM113:	.ASCIZ	:NEG!
9006	055744	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	

ERROR MESSAGES

9007	055746	055754	055453	000000		.WORD	EM113B,EM76B,0
9008	055754	104	040	100	EM113B:	.ASCIZ	!D @-(R)!
9009	055764	116	105	107	EM114:	.ASCIZ	!NEG!
9010	055770	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9011	055772	055754	052552	000000		.WORD	EM113B,EM4D,0
9012	056000	116	105	107	EM115:	.ASCIZ	!NEG!
9013	056004	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9014	056006	055754	052531	000000		.WORD	EM113B,EM4B,0
9015	056014	116	105	107	EM116:	.ASCIZ	!NEG!
9016	056020	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9017	056022	055754	052436	000000		.WORD	EM113B,EM2B,0
9018	056030	101	102	123	EM117:	.ASCIZ	!ABS!
9019	056034	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9020	056036	056044	055453	000000		.WORD	EM117B,EM76B,0
9021	056044	104	040	116	EM117B:	.ASCIZ	!D N(R)!
9022	056053	101	102	123	EM120:	.ASCIZ	!ABS!
9023	056057	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9024	056060	056044	052552	000000		.WORD	EM117B,EM4D,0
9025	056066	101	102	123	EM121:	.ASCIZ	!ABS!
9026	056072	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9027	056074	056044	052531	000000		.WORD	EM117B,EM4B,0
9028	056102	101	102	123	EM122:	.ASCIZ	!ABS!
9029	056106	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9030	056110	056044	052436	000000		.WORD	EM117B,EM2B,0
9031	056116	116	105	107	EM123:	.ASCIZ	!NEG!
9032	056122	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9033	056124	056132	055453	000000		.WORD	EM123B,EM76B,0
9034	056132	104	040	100	EM123B:	.ASCIZ	!D @N(R)!
9035	056142	116	105	107	EM124:	.ASCIZ	!NEG!
9036	056146	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9037	056150	056132	052552	000000		.WORD	EM123B,EM4D,0
9038	056156	116	105	107	EM125:	.ASCIZ	!NEG!
9039	056162	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9040	056164	056132	052531	000000		.WORD	EM123B,EM4B,0
9041	056172	116	105	107	EM126:	.ASCIZ	!NEG!
9042	056176	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9043	056200	056132	052436	000000		.WORD	EM123B,EM2B,0
9044	056206	116	105	107	EM127:	.ASCIZ	!NEG!
9045	056212	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9046	056214	056222	055453	000000		.WORD	EM127B,EM76B,0
9047	056222	104	040	116	EM127B:	.ASCIZ	!D N(R?)!
9048	056232	116	105	107	EM130:	.ASCIZ	!NEG!
9049	056236	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9050	056240	056222	052552	000000		.WORD	EM127B,EM4D,0

ERROR MESSAGES

9051	056246	116	105	107	EM131:	.ASCIZ	!NEG'	
9052	056252	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9053	056254	056222	052436	000000		.WORD	EM127B,EM2B,0	
9054	056262	101	102	123	EM132:	.ASCIZ	!ABS:	
9055	056266	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9056	056270	056276	055453	000000		.WORD	EM132B,EM76B,0	
9057	056276	104	040	100	EM132B:	.ASCIZ	!D @N(R7);	
9058	056307	101	102	123	EM133:	.ASCIZ	!ABS:	
9059	056313	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9060	056314	056276	052552	000000		.WORD	EM132B,EM4D,0	
9061	056322	101	102	123	EM134:	.ASCIZ	!ABS:	
9062	056326	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9063	056330	056276	052436	000000		.WORD	EM132B,EM2B,0	
9064	056336	116	105	107	EM135:	.ASCIZ	!NEGD A FAILED.!<CRLF>!XOR SIGN BIT ST 336!	
9065	056401	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9066	056402	052552	000000			.WORD	EM4D,0	
9067	056406	116	105	107	EM136:	.ASCIZ	!NEGD A:	
9068	056415	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9069	056416	052552	000000			.WORD	EM4D,0	
9070	056422	116	105	107	EM137:	.ASCIZ	!NEGD A:	
9071	056431	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9072	056432	052436	000000			.WORD	EM2B,0	
9073	056436	116	105	107	EM140:	.ASCIZ	!NEG:	
9074	056442	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9075	056444	055220	052552	000000		.WORD	EM66B,EM4D,0	
9076	056452	116	105	107	EM141:	.ASCIZ	!NEG:	
9077	056456	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9078	056460	055220	052531	056472		.WORD	EM66B,EM4B,EM141C,EM4D,0	
9079	056472	040	123	120	EM141C:	.ASCIZ	!SPECIAL DEST:	
9080	056510	116	105	107	EM142:	.ASCIZ	!NEG:	
9081	056514	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9082	056516	055220	052436	000000		.WORD	EM66B,EM2B,0	
9083	056524	116	105	107	EM143:	.ASCIZ	!NEG:	
9084	056530	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9085	056532	055444	052552	000000		.WORD	EM76A,EM4D,0	
9086	056540	116	105	107	EM144:	.ASCIZ	!NEG:	
9087	056544	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9088	056546	055444	052531	056472		.WORD	EM76A,EM4B,EM141C,0	
9089	056556	116	105	107	EM145:	.ASCIZ	!NEG:	
9090	056562	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9091	056564	055444	052436	000000		.WORD	EM76A,EM2B,0	
9092	056572	116	105	107	EM146:	.ASCIZ	!NEG:	
9093	056576	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		

ERROR MESSAGES

9094	05660C	055562	052552	000000			.WORD	EM102B,EM4D,0
9095	056606	116	105	107	EM147:		.ASCIZ	!NEG!
9096	056612	377					.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
							.EVEN	
9097	056614	055562	052531	056472			.WORD	EM102B,EM4B,EM141C,EM4D,0
9098	056626	116	105	107	EM150:		.ASCIZ	!NEG!
9099	056632	377					.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
							.EVEN	
9100	056634	055562	052436	000000			.WORD	EM102B,EM2B,0
9101	056642	116	105	107	EM151:		.ASCIZ	!NEG!
9102	056646	377					.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
							.EVEN	
9103	056650	055650	052552	000000			.WORD	EM106B,EM4D,0
9104	056656	116	105	107	EM152:		.ASCIZ	!NEG!
9105	056662	377					.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
							.EVEN	
9106	056664	055650	052531	056472			.WORD	EM106B,EM4B,EM141C,EM4D,0
9107	056676	116	105	107	EM153:		.ASCIZ	!NEG!
9108	056702	377					.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
							.EVEN	
9109	056704	055650	052436	000000			.WORD	EM106B,EM2B,0
9110	056712	116	105	107	EM154:		.ASCIZ	!NEG!
9111	056716	377					.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
							.EVEN	
9112	056720	055754	052552	000000			.WORD	EM113B,EM4D,0
9113	056726	116	105	107	EM155:		.ASCIZ	!NEG!
9114	056732	377					.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
							.EVEN	
9115	056734	055754	052531	056472			.WORD	EM113B,EM4B,EM141C,EM4D,0
9116	056746	116	105	107	EM156:		.ASCIZ	!NEG!
9117	056752	377					.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
							.EVEN	
9118	056754	055754	052436	000000			.WORD	EM113B,EM2B,0
9119	056762	116	105	107	EM157:		.ASCIZ	!NEG!
9120	056766	377					.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
							.EVEN	
9121	056770	056776	052552	000000			.WORD	EM157B,EM4D,0
9122	056776	106	040	050	EM157B:		.ASCIZ	!F (R)+!
9123	057005	116	105	107	EM160:		.ASCIZ	!NEG!
9124	057011	377					.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
							.EVEN	
9125	057012	056776	052531	057026			.WORD	EM157B,EM4B,EM160B,EM141C,EM4D,0
9126	057026	056	040	102	EM160B:		.ASCIZ	! . BAD CONSTANT USED.!
9127	057053	116	105	107	EM161:		.ASCIZ	!NEG!
9128	057057	377					.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
							.EVEN	
9129	057060	056776	052436	000000			.WORD	EM157B,EM2B,0
9130	057066	116	105	107	EM162:		.ASCIZ	!NEG!
9131	057072	377					.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
							.EVEN	
9132	057074	057102	052552	000000			.WORD	EM162B,EM4D,0
9133	057102	104	040	050	EM162B:		.ASCIZ	!D (R7)+!
9134	057112	116	105	107	EM163:		.ASCIZ	!NEG!
9135	057116	377					.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
							.EVEN	
9136	057120	057102	052436	000000			.WORD	EM162B,EM2B,0
9137	057126	120	103	040	EM164:		.ASCIZ	!PC BAD AFTER NEG!

9138	057147	377					.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
							.EVEN		
9139	057150	057102	057026	000000			.WORD	EM162B,EM160B,0	
9140	057156	116	105	107	EM165:		.ASCIZ	!NEG!	
9141	057162	377					.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
							.EVEN		
9142	057164	055220	052552	000000			.WORD	EM66B,EM4D,0	
9143	057172	101	102	123	EM166:		.ASCIZ	!ABS!	
9144	057176	377					.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
							.EVEN		
9145	057200	055220	052552	000000			.WORD	EM66B,EM4D,0	
9146	057206	124	123	124	EM167:		.ASCIZ	!TST!	
9147	057212	377					.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
							.EVEN		
9148	057214	055220	052552	000000			.WORD	EM66B,EM4D,0	
9149	057222	116	105	107	EM170:		.ASCIZ	!NEG!	
9150	057226	377					.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
							.EVEN		
9151	057230	055220	052436	000000			.WORD	EM66B,EM2B,0	
9152	057236	101	102	123	EM171:		.ASCIZ	!ABS!	
9153	057242	377					.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
							.EVEN		
9154	057244	055220	052436	000000			.WORD	EM66B,EM2B,0	
9155	057252	124	123	124	EM172:		.ASCIZ	!TST!	
9156	057256	377					.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
							.EVEN		
9157	057260	055220	052436	000000			.WORD	EM66B,EM2B,0	
9158	057266	116	105	107	EM173:		.ASCIZ	!NEG!	
9159	057272	377					.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
							.EVEN		
9160	057274	055220	052466	000000			.WORD	EM66B,EM3B,0	
9161	057302	101	102	123	EM174:		.ASCIZ	!ABS!	
9162	057306	377					.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
							.EVEN		
9163	057310	055220	052466	000000			.WORD	EM66B,EM3B,0	
9164	057316	124	123	124	EM175:		.ASCIZ	!TST!	
9165	057322	377					.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
							.EVEN		
9166	057324	055220	052466	000000			.WORD	EM66B,EM3B,0	
9167	057332	116	105	107	EM176:		.ASCIZ	!NEG!	
9168	057336	377					.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
							.EVEN		
9169	057340	057346	052436	000000			.WORD	EM176B,EM2B,0	
9170	057346	106	040	101	EM176B:		.ASCIZ	!F AC7!	
9171	057354	120	117	127	EM177:		.ASCIZ	!POWER MONITOR BIT FOUND SET!	
9172	057410	116	105	107	EM200:		.ASCIZ	!NEG!	
9173	057414	377					.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
							.EVEN		
9174	057416	055220	052552	057426			.WORD	EM66B,EM4D,EM200C,0	
9175	057426	200	130	117	EM200C:		.ASCIZ	<CRLF>!XOR SIGN BIT FAILED ST 336.!	
9176	057463	116	105	107	EM201:		.ASCIZ	!NEG!	
9177	057467	377					.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
							.EVEN		
9178	057470	055220	052436				.WORD	EM66B,EM2B	
9179	057474	377					.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
							.EVEN		
057476	052653	054230	052662				.WORD	CRBUT, IENBT, PRST, N3, N3, N6, WENTTO, NO, N5, N3, INSTOF, N4, N5, N3, 0	

ERROR MESSAGES

9180	057534	116	105	107	EM202:	.ASCIZ	:NEG:	
9181	057540	377				.BYTE	377	; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9182	057542	055220	052436			.WORD	EM66B,EM2B	
9183	057546	377				.BYTE	377	; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
	057550	052653	054230	052662		.WORD	CRBUT,IENBT,PRST,N3,N3,N6,WENTTO,N4,N5,N3,INSTOF,N0,N5,N3,0	
9184	057606	101	102	123	EM203:	.ASCIZ	:ABS:	
9185	057612	377				.BYTE	377	; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9186	057614	055220	052552			.WORD	EM66B,EM4D	
9187	057620	377				.BYTE	377	; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
	057622	052653	057723	052662		.WORD	CRBUT,IOP1B,PRST,N0,N5,N5,WENTTO,N3,N3,N6,INSTOF,N3,N3,N5,OR	
9188	057660	052653	054230	052662		.WORD	CRBUT,IENBT,PRST,N3,N3,N5,WENTTO,N4,N5,N2,INSTOF,N0,N5,N2,0	
9189	057716	054	040	117	OR:	.ASCIZ	:OR:	
9190	057723	117	120	061	IOP1B:	.ASCIZ	:OP1B:	
9191	057730	101	102	123	EM204:	.ASCIZ	:ABS:	
9192	057734	377				.BYTE	377	; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9193	057736	055220	052552	057746		.WORD	EM66B,EM4D,EM204C,0	
9194	057746	200	130	117	EM204C:	.ASCIZ	<CRLF>;XOR SIGN BIT FAILED ST 452.;	
9195	060003	124	123	124	EM205:	.ASCIZ	:TST:	
9196	060007	377				.BYTE	377	; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9197	060010	055220	052552			.WORD	EM66B,EM4D	
9198	060014	377				.BYTE	377	; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
	060016	052653	057723	052662		.WORD	CRBUT,IOP1B,PRST,N0,N5,N5,WENTTO,N3,N3,N6,INSTOF,N3,N3,N4,0	
9199	060054	124	123	124	EM206:	.ASCIZ	:TST:	
9200	060060	377				.BYTE	377	; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9201	060062	055220	052436			.WORD	EM66B,EM2B	
9202	060066	377				.BYTE	377	; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
	060070	052653	054230	052662		.WORD	CRBUT,IENBT,PRST,N3,N3,N4,WENTTO,N4,N5,N3,INSTOF,N0,N5,N3,0	
9203	060126	124	123	124	EM207:	.ASCIZ	:TST:	
9204	060132	377				.BYTE	377	; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9205	060134	055220	052552			.WORD	EM66B,EM4D	
9206	060140	377				.BYTE	377	; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
	060142	052653	057723	052662		.WORD	CRBUT,IOP1B,PRST,N0,N5,N7,WENTTO,N3,N3,N5,INSTOF,N3,N3,N4,0	
9207	060200	124	123	124	EM210:	.ASCIZ	:TST:	
9208	060204	377				.BYTE	377	; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9209	060206	055220	052552			.WORD	EM66B,EM4D	
9210	060212	377				.BYTE	377	; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
	060214	052653	054230	052662		.WORD	CRBUT,IENBT,PRST,N3,N3,N4,WENTTO,N0,N5,N3,INSTOF,N4,N5,N3,0	
9211	060252	124	123	124	EM211:	.ASCIZ	:TST:	
9212	060256	377				.BYTE	377	; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9213	060260	055220	052552	052653		.WORD	EM66B,EM4D,CRBUT,IOP1B,PRST,N2,N5,N5,WENTTO,N3132,INSTOF,N3,N1,N0,0	
9214	060316	063	061	061	N3132:	.ASCIZ	:311 OR 312:	
9215	060331	124	123	124	EM212:	.ASCIZ	:TST:	
9216	060335	377				.BYTE	377	; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN

ERROR MESSAGES

Line	Address	Code	Label	Text	Comment
9217	060336	055220	052436	.EVEN	
9218	060342	377		.WORD	EM66B,EM2B
				.BYTE	377 ; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
				.EVEN	
9219	060344	052653	054230	.WORD	CRBUT,IENBT,PRST,N3,N1,NO,WENTTO,N4,NO,N2,INSTOF,NO,NO,N2,0
9220	060402	124	123	.ASCIZ	!TST!
9220	060406	377		.BYTE	377 ; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
				.EVEN	
9221	060410	055220	052436	.WORD	EM66B,EM2B,EM213C,EM213D
9222	060420	377	060460	.BYTE	377 ; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
				.EVEN	
9223	060422	052653	060506	.WORD	CRBUT,IFIUV,PRST,N2,N5,N7,WENTTO,N3,N5,N5,INSTOF,N2,N5,N5,0
9224	060460	040	106	.ASCIZ	!FIUV=0!
9224	060470	054	040	.ASCIZ	!OPERAND=-0.!
9225	060506	106	111	.ASCIZ	!FIUV!
9226	060513	124	123	.ASCIZ	!TST!
9227	060517	377		.BYTE	377 ; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
				.EVEN	
9228	060520	055220	052436	.WORD	EM66B,EM2B,EM214B,EM213D
9229	060530	377	060570	.BYTE	377 ; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
				.EVEN	
9230	060532	052653	060506	.WORD	CRBUT,IFIUV,PRST,N2,N5,N7,WENTTO,N2,N5,N5,INSTOF,N3,N5,N5,0
9230	060570	040	106	.ASCIZ	!FIUV=1!
9236	060600	120	000	.ASCIZ	!P!
9237	060602	377		.BYTE	377 ; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
				.EVEN	
9238	060604	060647	060635	.WORD	PCBAD,NEGDNR,BADCON,N746T2,BUTIN,EM141C,EM4D,0
9239	060624	067	064	.ASCIZ	!746 746.!
9240	060635	116	105	.ASCIZ	!NEGD N(R)!
9241	060647	103	040	.ASCIZ	!C BAD AFTER !
9242	060664	056	040	.ASCIZ	!BAD CONSTANT USED !
9243	060711	056	000	.ASCIZ	!..!
9244	060713	200	117	.ASCIZ	<CRLF>!OR (BUT FDST) IN!
9245	060735	116	105	.ASCIZ	!NEG!
9246	060741	377		.BYTE	377 ; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
				.EVEN	
9247	060742	056044	052552	.WORD	EM117B,EM4D,0
9248	060750	116	105	.ASCIZ	!NEG!
9249	060754	377	107	.BYTE	377 ; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
				.EVEN	
9250	060756	056044	052531	.WORD	EM117B,EM4B,EM141C,EM4D,0
9251	060770	116	105	.ASCIZ	!NEG!
9252	060774	377	107	.BYTE	377 ; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
				.EVEN	
9253	060776	056044	052436	.WORD	EM117B,EM2B,0
9254	061004	120	000	.ASCIZ	!P!
9255	061006	377		.BYTE	377 ; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
				.EVEN	
9256	061010	060647	061041	.WORD	PCBAD,NEGDAR,BADCON,N747T2,BUTIN,EM141C,EM4D,0
9257	061030	067	064	.ASCIZ	!747 747.!
9258	061041	116	105	.ASCIZ	!NEGD @N(R)!
9259	061054	116	105	.ASCIZ	!NEG!
9260	061060	377		.BYTE	377 ; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
				.EVEN	
9261	061062	056132	052552	.WORD	EM123B,EM4D,0
9262	061070	116	105	.ASCIZ	!NEG!
9263	061074	377	107	.BYTE	377 ; FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN

Address	Offset	PC	MACRO	Label	Instruction	Comment
9264	061076	056132	052531	056472	.EVEN	
9265	061110	116	105	107	.WORD	EM123B,EM4B,EM141C,EM4D,0
9266	061114	377		EM224:	.ASCIZ	!NEG!
					.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9267	061116	056132	052436	000000	.EVEN	
9268	061124	114	000		.WORD	EM123B,EM2B,0
9269	061126	377		EM225:	.ASCIZ	!L!
					.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9270	061130	061136	052531	000000	.EVEN	
9271	061136	104	106	120	.WORD	EM225B,EM4B,0
9272	061147	114	000		EM225B:	.ASCIZ !DFPS (R)!
9273	061151	377		EM226:	.ASCIZ	!L!
					.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9274	061152	061136	052436	000000	.EVEN	
9275	061160	114	000		.WORD	EM225B,EM2B,0
9276	061162	377		EM227:	.ASCIZ	!L!
					.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9277	061164	061136	061172	000000	.EVEN	
9278	061172	040	124	122	.WORD	EM225B,EM227B,0
9279	061211	114	000		EM227B:	.ASCIZ ! TRAPPED TO 4.!
9280	061213	377		EM230:	.ASCIZ	!L!
					.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9281	061214	061222	052531	000000	.EVEN	
9282	061222	104	106	120	.WORD	EM230B,EM4B,0
9283	061234	114	000		EM230B:	.ASCIZ !DFPS (R)+!
9284	061236	377		EM231:	.ASCIZ	!L!
					.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9285	061240	061222	052436	000000	.EVEN	
9286	061246	114	000		.WORD	EM230B,EM2B,0
9287	061250	377		EM232:	.ASCIZ	!L!
					.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9288	061252	061222	061172	000000	.EVEN	
9289	061260	114	000		.WORD	EM230B,EM227B,0
9290	061262	377		EM233:	.ASCIZ	!L!
					.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9291	061264	061272	052531	000000	.EVEN	
9292	061272	104	106	120	.WORD	EM233B,EM4B,0
9293	061304	114	000		EM233B:	.ASCIZ !DFPS -(R)!
9294	061306	377		EM234:	.ASCIZ	!L!
					.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9295	061310	061272	052436	000000	.EVEN	
9296	061316	114	000		.WORD	EM233B,EM2B,0
9297	061320	377		EM235:	.ASCIZ	!L!
					.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9298	061322	061272	061172	000000	.EVEN	
9299	061330	114	000		.WORD	EM233B,EM227B,0
9300	061332	377		EM236:	.ASCIZ	!L!
					.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9301	061334	061342	052531	000000	.EVEN	
9302	061342	104	106	120	.WORD	EM236B,EM4B,0
9303	061355	114	000		EM236B:	.ASCIZ !DFPS @ (R)+!
9304	061357	377		EM237:	.ASCIZ	!L!
					.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9305	061360	061342	052436	000000	.EVEN	
9306	061366	114	000		.WORD	EM236B,EM2B,0
9307	061370	377		EM240:	.ASCIZ	!L!
					.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN

.EVEN
9308 061372 061342 061172 000000
9309 061400 114 000
9310 061402 377
EM241: .WORD EM236B,EM227B,0
.ASCIZ !L!
.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
.EVEN
9311 061404 061412 052531 000000
9312 061412 104 106 120
9313 061425 114 000
9314 061427 377
EM241B: .WORD EM241B,EM4B,0
EM242: .ASCIZ !DFPS @-(R)!
.ASCIZ !L!
.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
.EVEN
9315 061430 061412 052436 000000
9316 061436 114 000
9317 061440 377
EM243: .WORD EM241B,EM2B,0
.ASCIZ !L!
.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
.EVEN
9318 061442 061412 061172 000000
9319 061450 114 000
9320 061452 377
EM244: .WORD EM241B,EM227B,0
.ASCIZ !L!
.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
.EVEN
9321 061454 061462 052531 000000
9322 061462 104 106 120
9323 061474 114 000
9324 061476 377
EM244B: .WORD EM244B,EM4B,0
EM245: .ASCIZ !DFPS N(R)!
.ASCIZ !L!
.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
.EVEN
9325 061500 061462 052436 000000
9326 061506 120 103 040
9327 061524 376
EM246: .WORD EM244B,EM2B,0
.ASCIZ !PC BAD AFTER!
.BYTE 376 ;FLAGS 'ERTYPE' TO PRINT ERR # SPECIFIC ASCIZ MSGS, ADRS 3 LINES DOWN
.EVEN
9328 061526 000246 000252 000254
9329 061536 061544 061626 061654
9330 061544 114 104 106
9331 061557 114 000
9332 061561 377
EM246B: .WORD 246,252,254,0
EM247: .WORD EM246B,EM252B,EM254B
.ASCIZ !LDFPS N(R)!
.ASCIZ !L!
.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
.EVEN
9333 061562 061462 061172 000000
9334 061570 114 000
9335 061572 377
EM250: .WORD EM244B,EM227B,0
.ASCIZ !L!
.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
.EVEN
9336 061574 061602 052531 000000
9337 061602 104 106 120
9338 061615 114 000
9339 061617 377
EM250B: .WORD EM250B,EM4B,0
EM251: .ASCIZ !DFPS @N(R)!
.ASCIZ !L!
.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
.EVEN
9340 061620 061602 052436 000000
9341 061506
9342 061626 114 104 106
9343 061642 114 000
9344 061644 377
EM252=EM246
EM252B: .WORD EM250B,EM2B,0
EM253: .ASCIZ !LDFPS @N(R)!
.ASCIZ !L!
.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
.EVEN
9345 061646 061602 061172 000000
9346 061506
9347 061654 114 104 103
9348 061672 114 104 103
9349 061710 377
EM254=EM246
EM254B: .WORD EM250B,EM227B,0
EM255: .ASCIZ !LDCLD (R7)+,A!
.ASCIZ !L!
.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
.EVEN
9350 061712 061172 000000
9351 061716 114 000
9352 061720 377
EM256: .WORD EM227B,0
.ASCIZ !L!
.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN

Line No.	Time	Address	Code	Macro	Text
9353	061722	061730	052531	000000	.EVEN
9354	061730	104	103	114	.WORD EM256B,EM4B,0
9355	061744	114	000		.ASCIZ :DCLD (R)+,A:
9356	061746	377			.ASCIZ :L:
					.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9357	061750	061730	052436	000000	.EVEN
9358	061756	114	000		.WORD EM256B,EM2B,0
9359	061760	377			.ASCIZ :L:
					.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9360	061762	061770	052552	000000	.EVEN
9361	061770	104	103	111	.WORD EM260B,EM4D,0
9362	062014	114	000		.ASCIZ :DCIF OR LDCLF (R),A:
9363	062016	377			.ASCIZ :L:
					.BYTF 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9364	062020	061770	052436	000000	.EVEN
9365	062026	114	000		.WORD EM260B,EM2B,0
9366	062030	377			.ASCIZ :L:
					.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9367	062032	062101	052552		.EVEN
9368	062036	377			.WORD EM262B,EM4D
					.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9369	062040	052653	062076	052662	.EVEN
9370	062076	106	114	000	.WORD CRBUT,IFL,PRST,N2,N7,N7,WENTTO,N3,NO,NO,INSTOF,N3,NO,N1,0
9371	062101	104	103	111	.ASCIZ :IFL:
9372	062114	114	000		.ASCIZ :DCIF (R),A:
9372	062116	377			.ASCIZ :L:
					.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9373	062120	062126	052436	000000	.EVEN
9374	062126	104	103	114	.WORD EM263B,EM2B,0
9375	062141	114	000		.ASCIZ :DCLF (R),A:
9376	062143	377			.ASCIZ :L:
					.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9377	062144	062101	052552	062154	.EVEN
9378	062154	200	125	123	.WORD EM262B,EM4D,EM264C,0
9379	062226	114	000		.ASCIZ <CRLF>;USED CONSTANT 237 INSTEAD OF 217 ST 107.:
9380	062230	377			.ASCIZ :L:
					.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9381	062232	061770	052552	062242	.EVEN
9382	062242	200	123	105	.WORD EM260B,EM4D,EM265C,0
9383	062277	114	000		.ASCIZ <CRLF>;SET SIGN BIT FAILED ST 146.:
9384	062301	377			.ASCIZ :L:
					.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9385	062302	061770	052552		.EVEN
9386	062306	377			.WORD EM260B,EM4D
					.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9387	062310	052653	062346	052662	.EVEN
9388	062346	130	116	102	.WORD CRBUT,IXNBT,PRST,N3,N7,N2,WENTTO,N1,N5,N2,INSTOF,N1,N1,N2,0
9389	062353	114	000		.ASCIZ :IXNBT:
9389	062355	377			.ASCIZ :L:
					.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9390	062356	062126	052552	062366	.EVEN
9391	062366	200	125	123	.WORD EM263B,EM4D,EM267C,0
9392	062440	114	000		.ASCIZ <CRLF>;USED CONSTANT 217 INSTEAD OF 237 ST 107.:
9393	062442	377			.ASCIZ :L:
					.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9394	062444	062126	052552	062454	.EVEN
					.WORD EM263B,EM4D,EM270C,0

9395	062454	040	122	117	EM270C:	.ASCIZ	: ROUND ERROR.:
9396	062472	114	000		EM271:	.ASCIZ	:L:
9397	062474	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9398	062476	062126	052552	062506		.WORD	EM263B,EM4D,EM271C,0
9399	062506	040	124	122	EM271C:	.ASCIZ	: TRUNCATION ERROR.:
9400	062531	114	000		EM272:	.ASCIZ	:L:
9401	062533	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9402	062534	061770	052552	062544		.WORD	EM260B,EM4D,EM272C,0
9403	062544	200	122	061	EM272C:	.ASCIZ	<CRLF>;R14 NOT INCREMENTED ST 630.:
9404	062601	114	000		EM273:	.ASCIZ	:L:
9405	062603	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9406	062604	062612	052552	000000		.WORD	EM273B,EM4D,0
9407	062612	104	103	111	EM273B:	.ASCIZ	:DCID OR LDCLD (R),A:
9408	062636	114	000		EM274:	.ASCIZ	:L:
9409	062640	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9410	062642	062612	052436	000000		.WORD	EM273B,EM2B,0
9411	062650	114	000		EM275:	.ASCIZ	:L:
9412	062652	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9413	062654	062720	052552			.WORD	EM275B,EM4D
9414	062660	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9415	062662	052653	062076	052662		.WORD	CRBUT,IFL,PRST,N2,N7,N7,WENTTO,N3,NO,NO,INSTOF,N3,NO,N1,0
9416	062720	104	103	111	EM275B:	.ASCIZ	:DCID (R),A:
9417	062733	114	000		EM276:	.ASCIZ	:L:
9418	062735	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9418	062736	062720	052552	062746		.WORD	EM275B,EM4D,EM276C,0
9419	062746	200	125	123	EM276C:	.ASCIZ	<CRLF>;USED CONSTANT 237 INSTEAD OF 217 ST 107.:
9420	063020	114	000		EM277:	.ASCIZ	:L:
9421	063022	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9422	063024	062720	052552	063034		.WORD	EM275B,EM4D,EM277C,0
9423	063034	200	123	105	EM277C:	.ASCIZ	<CRLF>;SET SIGN FAILED ST 146.:
9424	063065	114	104	103	EM300:	.ASCIZ	:LDCLD (R),A:
9425	063101	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9426	063102	052552	063110	000000		.WORD	EM4D,EM300C,0
9427	063110	200	125	123	EM300C:	.ASCIZ	<CRLF>;USED CONSTANT 217 INSTEAD OF 237 ST 107.:
9428	063162	114	000		EM301:	.ASCIZ	:L:
9429	063164	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9430	063166	063174	052552	000000		.WORD	EM301B,EM4D,0
9431	063174	104	105	130	EM301B:	.ASCIZ	:DEXP (R),A:
9432	063207	114	000		EM302:	.ASCIZ	:L:
9433	063211	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9434	063212	063174	052436	000000		.WORD	EM301B,EM2B,0
9435	063220	114	000		EM303:	.ASCIZ	:L:
9436	063222	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9437	063224	063174	052466	000000		.WORD	EM301B,EM3B,0
9438	063232	114	000		EM304:	.ASCIZ	:L:

ERROR MESSAGES

9439	063234		377						.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
									.EVEN		
9440	063236	063174	052552	063246					.WORD	EM301B,EM4D,EM304C,0	
9441	063246	200	105	130	EM304C:				.ASCIZ	<CRLF>;EXCESS 200 CALCULATION ST 624 BAD.:	
9442	063312	114	000		EM305:				.ASCIZ	!L!	
9443	063314	377							.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
									.EVEN		
9444	063316	063174	052436	063326					.WORD	EM301B,EM2B,EM305C,0	
9445	063326	200	050	102	EM305C:				.ASCIZ	<CRLF>;(BUT ENBT,EZBT,XNBT) ST 625 DID NOT GO TO 304.:	
9446	063406	114	000		EM306:				.ASCIZ	!L!	
9447	063410	377							.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
									.EVEN		
9448	063412	063174	052436						.WORD	EM301B,EM2B	
9449	063416	377							.BYTF	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
									.EVEN		
	063420	052653	053724	052662					.WORD	CRBUT,IEZBT,PRST,N5,N4,N4,WENTTO,N5,N0,N4,INSTOF,N7,N0,N4,OR	
9450	063456	052653	053724	052662					.WORD	CRBUT,IEZBT,PRST,N7,N0,N4,WENTTO,N2,N6,N4,INSTOF,N0,N6,N4,0	
9451	063514	114	000		EM307:				.ASCIZ	!L!	
9452	063516	377							.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
									.EVEN		
9453	063520	063174	052552						.WORD	EM301B,EM4D	
9454	063524	377							.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
									.EVEN		
	063526	052653	053724	052662					.WORD	CRBUT,IEZBT,PRST,N7,N0,N4,WENTTO,N0,N6,N4,INSTOF,N2,N6,N4,0	
9455	063564	114	000		EM310:				.ASCIZ	!L!	
9456	063566	377							.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
									.EVEN		
9457	063570	063174	052436						.WORD	EM301B,EM2B	
9458	063574	377							.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
									.EVEN		
	063576	052653	063634	052662					.WORD	CRBUT,IFIU,PRST,N2,N6,N4,WENTTO,N1,N1,N5,INSTOF,N1,N5,N5,0	
9459	063634	106	111	125	IFIU:				.ASCIZ	!FIU!	
9460	063640	114	000		EM311:				.ASCIZ	!L!	
9461	063642	377							.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
									.EVEN		
9462	063644	063174	052552						.WORD	EM301B,EM4D	
9463	063650	377							.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
									.EVEN		
	063652	052653	063634	052662					.WORD	CRBUT,IFIU,PRST,N2,N6,N4,WENTTO,N1,N5,N5,INSTOF,N1,N1,N5,0	
9464	063710	114	000		EM312:				.ASCIZ	!L!	
9465	063712	377							.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
									.EVEN		
9466	063714	063174	052552						.WORD	EM301B,EM4D	
9467	063720	377							.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
									.EVEN		
	063722	052653	053724	052662					.WORD	CRBUT,IEZBT,PRST,N5,N4,N4,WENTTO,N7,N0,N4,INSTOF,N5,N0,N4,0	
9468	063760	114	000		EM313:				.ASCIZ	!L!	
9469	063762	377							.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
									.EVEN		
9470	063764	063174	052552						.WORD	EM301B,EM4D	
9471	063770	377							.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
									.EVEN		
	063772	052653	063634	052662					.WORD	CRBUT,IFIU,PRST,N5,N0,N4,WENTTO,N1,N5,N5,INSTOF,N1,N1,N5,0	
9472	064030	114	000		EM314:				.ASCIZ	!L!	
9473	064032	377							.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
									.EVEN		
9474	064034	063174	052552						.WORD	EM301B,EM4D	

ERROR MESSAGES

9475	064040	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
	064042	052653	054626	052662		.EVEN		
9476	064100	114	000		EM315:	.WORD	CRBUT,IFIV,PRST,N1,N0,N4,WENTTO,N1,N1,N6,INSTOF,N1,N3,N6,0	
9477	064102	377				.ASCIZ	:L:	
						.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9478	064104	063174	052552			.WORD	EM301B,EM4D	
9479	064110	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
	064112	052653	054626	052662		.WORD	CRBUT,IFIV,PRST,N1,N0,N4,WENTTO,N1,N3,N6,INSTOF,N1,N1,N6,0	
9480	064150	114	000		EM316:	.ASCIZ	:L:	
9481	064152	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9482	064154	063174	052552			.WORD	EM301B,EM4D	
9483	064160	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
	064162	052653	054626	052662		.WORD	CRBUT,IFIV,PRST,N1,N4,N4,WENTTO,N1,N1,N6,INSTOF,N1,N3,N6,0	
9484	064220	114	000		EM317:	.ASCIZ	:L:	
9485	064222	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9486	064224	063174	052552			.WORD	EM301B,EM4D	
9487	064230	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
	064232	052653	054626	052662		.WORD	CRBUT,IFIV,PRST,N1,N4,N4,WENTTO,N1,N3,N6,INSTOF,N1,N1,N6,0	
9488	064270	114	000		EM320:	.ASCIZ	:L:	
9489	064272	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9490	064274	063174	052552			.WORD	EM301B,EM4D	
9491	064300	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
	064302	052653	054626	052662		.WORD	CRBUT,IFIV,PRST,N3,N4,N4,WENTTO,N1,N1,N6,INSTOF,N1,N3,N6,0	
9492	064340	114	000		EM321:	.ASCIZ	:L:	
9493	064342	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9494	064344	063174	052552			.WORD	EM301B,EM4D	
9495	064350	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
	064352	052653	054626	052662		.WORD	CRBUT,IFIV,PRST,N3,N4,N4,WENTTO,N1,N3,N6,INSTOF,N1,N1,N6,0	
9496	064410	123	000		EM322:	.ASCIZ	:S:	
9497	064412	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9498	064414	064422	052552	000000		.WORD	EM322B,EM4D,0	
9499	064422	124	103	104	EM322B:	.ASCIZ	:TCDI OR STCDL (R),A:	
9500	064446	123	000		EM323:	.ASCIZ	:S:	
9501	064450	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9502	064452	064422	052436	000000		.WORD	EM322B,EM2B,0	
9503	064460	123	000		EM324:	.ASCIZ	:S:	
9504	064462	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9505	064464	064422	052466	000000		.WORD	EM322B,EM3B,0	
9506	064472	123	000		EM325:	.ASCIZ	:S:	
9507	064474	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9508	064476	064506	052436	064521		.WORD	EM325B,EM2B,EM325C,0	
9509	064506	124	103	104	EM325B:	.ASCIZ	:TCDL (R),A:	
9510	064521	200	103	114	EM325C:	.ASCIZ	<CRLF>:CLEAR FLAG ST 774 FAILED, OR:	

Line	Address	Page	Line	Page	Code	Text
9511	064557	377			.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
					.EVEN	
	064560	052653	054764	052662	.WORD	CRBUT,IFLAG,PRST,N6,N6,N2,WENTTO,N3,N6,N5,INSTOF,N3,N6,N1,0
9512	064472				EM326=EM325	
9513	064616	123	000		EM327:	.ASCIZ !S!
9514	064620	377			.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
					.EVEN	
9515	064622	064506	052552		.WORD	EM325B,EM4D
9516	064626	377			.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
					.EVEN	
	064630	052653	054230	052662	.WORD	CRBUT,IENBT,PRST,N6,N3,N2,WENTTO,N4,N7,N3,INSTOF,N0,N7,N3,0
9517	064666	123	000		EM330:	.ASCIZ !S!
9518	064670	377			.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
					.EVEN	
9519	064672	064506	052436		.WORD	EM325B,EM2B
9520	064676	377			.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
					.EVEN	
	064700	052653	064736	052662	.WORD	CRBUT,IFIC,PRST,N0,N0,N4,WENTTO,N3,N0,N5,INSTOF,N3,N1,N5,0
9521	064736	106	111	103	IFIC:	.ASCIZ !FIC!
9522	064742	123	000		EM331:	.ASCIZ !S!
9523	064744	377			.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
					.EVEN	
9524	064746	064506	052436		.WORD	EM325b,EM2B
9525	064752	377			.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
					.EVEN	
	064754	052653	064736	052662	.WORD	CRBUT,IFIC,PRST,N0,N0,N4,WENTTO,N3,N1,N5,INSTOF,N3,N0,N5,0
9526	065012	123	000		EM332:	.ASCIZ !S!
9527	065014	377			.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
					.EVEN	
9528	065016	065026	052552	065041	.WORD	EM332B,EM4D,EM332C,0
9529	065026	124	103	104	EM332B:	.ASCIZ !TCDI (R),A!
9530	065041	200	125	123	EM332C:	.ASCIZ <CRLF>!USED CONSTANT 37 INSTEAD OF 17 ST 66.!
9531	064446				EM333=EM323	
9532	065110	123	000		EM334:	.ASCIZ !S!
9533	065112	377			.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
					.EVEN	
9534	065114	065026	052436	065124	.WORD	EM332B,EM2B,EM334C,0
9535	065124	200	125	123	EM334C:	.ASCIZ <CRLF>!USED CONSTANT 37 INSTEAD OF 17 ST 66.!
9536	065173	123	000		EM335:	.ASCIZ !S!
9537	065175	377			.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
					.EVEN	
9538	065176	065026	052552		.WORD	EM332B,EM4D
9539	065202	377			.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
					.EVEN	
	065204	052653	054230	052662	.WORD	CRBUT,IENBT,PRST,N6,N3,N2,WENTTO,N0,N7,N3,INSTOF,N4,N7,N3,0
9540	065242	123	000		EM336:	.ASCIZ !S!
9541	065244	377			.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
					.EVEN	
9542	065246	065026	052436	065260	.WORD	EM332B,EM2B,EM336C,EM4D,0
9543	065260	200	123	105	EM336C:	.ASCIZ <CRLF>!SET FN ST 473!
9544	065277	123	000		EM337:	.ASCIZ !S!
9545	065301	377			.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
					.EVEN	
9546	065302	064506	052552		.WORD	EM325B,EM4D
9547	065306	377			.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
					.EVEN	
	065310	052653	065346	052662	.WORD	CRBUT,ICOUT,PRST,N2,N7,N5,WENTTO,N0,N7,N4,INSTOF,N2,N7,N4,0

ERROR MESSAGES

9548	065346	103	117	125	ICOUT:	.ASCIZ	:COUT:	
9549	065353	123	000		EM340:	.ASCIZ	:S!	
9550	065355	377				.BYTE	377	:FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9551	065356	064506	052552			.WORD	EM325B,EM4D	
9552	065362	377				.BYTE	377	:FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
	065364	052653	065346	052662		.WORD	CRBUT,ICOUT,PRST,N2,N7,N5,WENTTO,N2,N7,N4,INSTOF,N0,N7,N4,0	
9553	065422	123	000		EM341:	.ASCIZ	:S!	
9554	065424	377				.BYTE	377	:FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9555	065426	064506	052436			.WORD	EM325B,EM2B	
9556	065432	377				.BYTE	377	:FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
	065434	052653	053724	052662		.WORD	CRBUT,IEZBT,PRST,N3,N7,N7,WENTTO,N6,N3,N3,INSTOF,N4,N3,N3,0	
9557	065472	123	000		EM342:	.ASCIZ	:S!	
9558	065474	377				.BYTE	377	:FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9559	065476	064506	052552			.WORD	EM325B,EM4D	
9560	065502	377				.BYTE	377	:FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
	065504	052653	065346	052662		.WORD	CRBUT,ICOUT,PRST,N3,N6,N0,WENTTO,N6,N5,N4,INSTOF,N4,N5,N4,0	
9561	065542	123	000		EM343:	.ASCIZ	:S!	
9562	065544	377				.BYTE	377	:FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9563	065546	064506	052552			.WORD	EM325B,EM4D	
9564	065552	377				.BYTE	377	:FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
	065554	052653	065612	052662		.WORD	CRBUT,INBIT,PRST,N6,N5,N4,WENTTO,N5,N3,N1,INSTOF,N4,N3,N1,0	
9565	065612	116	102	111	INBIT:	.ASCIZ	:INBIT:	
9566	065617	123	000		EM344:	.ASCIZ	:S!	
9567	065621	377				.BYTE	377	:FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9568	065622	064506	052552			.WORD	EM325B,EM4D	
9569	065626	377				.BYTE	377	:FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
	065630	052653	065346	052662		.WORD	CRBUT,ICOUT,PRST,N3,N6,N0,WENTTO,N4,N5,N4,INSTOF,N6,N5,N4,OR	
9570	065666	052653	065612	052662		.WORD	CRBUT,INBIT,PRST,N6,N5,N4,WENTTO,N4,N3,N1,INSTOF,N5,N3,N1,0	
9571	065724	123	000		EM345:	.ASCIZ	:S!	
9572	065726	377				.BYTE	377	:FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9573	065730	065026	052552			.WORD	EM332B,EM4D	
9574	065734	377				.BYTE	377	:FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
	065736	052653	062076	052662		.WORD	CRBUT,IFL,PRST,N6,N3,N3,WENTTO,N6,N5,N5,INSTOF,N6,N5,N4,0	
9575	065774	123	124	103	EM346:	.ASCIZ	:STCFL (R)A!	
9576	066010	377				.BYTE	377	:FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9577	066012	052552	066020	000000		.WORD	EM4D,EM346C,0	
9578	066020	200	132	105	EM346C:	.ASCIZ	<CRLF>;ZERO LOW ORDER PART OF X11 FAILED ST 773.!	
9579	066073	123	000		EM347:	.ASCIZ	:S!	
9580	066075	377				.BYTE	377	:FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9581	066076	066104	052552	000000		.WORD	EM347B,EM4D,0	
9582	066104	124	105	130	EM347B:	.ASCIZ	:TEXP A,(R)!	
9583	066117	123	000		EM350:	.ASCIZ	:S!	
9584	066121	377				.BYTE	377	:FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN

Address	Offset	PC	Next PC	Next PC	Next PC	Label	Instruction
9585	066122	066104	052436	000000			.EVEN
9586	066130	115	117	122	EM351:	.WORD	EM347B,EM2B,0
9587	066153	127	122	111		.ASCII	:MORE THAN ONE WORD :
9588	066224	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
9589	066226	052552	000000			.EVEN	
9590	066232	123	000		EM352:	.WORD	EM4D,0
9591	066234	377				.ASCII	:S!
						.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9592	066236	066104	052436			.EVEN	
9593	066242	377				.WORD	EM347B,EM2B
						.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
	066244	052653	054230	052662		.EVEN	
9594	066302	123	000		EM353:	.WORD	CRBUT, IENBT, PRST, N3, N7, N6, WENTTO, NO, N7, N1, INSTOF, N4, N7, N1, 0
9595	066304	377				.ASCII	:S!
						.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9596	066306	066104	052436			.EVEN	
9597	066312	377				.WORD	EM347B,EM2B
						.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
	066314	052653	053724	052662		.EVEN	
9598	066352	123	000		EM354:	.WORD	CRBUT, IEZBT, PRST, NO, N7, N1, WENTTO, NO, N7, N2, INSTOF, N2, N7, N2, 0
9599	066354	377				.ASCII	:S!
						.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9600	066356	066104	052436			.EVEN	
9601	066357	377				.WORD	EM347B,EM2B
						.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
	066364	052653	053724	052662		.EVEN	
9602	066422	123	000		EM355:	.WORD	CRBUT, IEZBT, PRST, NO, N7, N1, WENTTO, N2, N7, N2, INSTOF, NO, N7, N2, 0
9603	066424	377				.ASCII	:S!
						.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9604	066426	066104	052436			.EVEN	
9605	066432	377				.WORD	EM347B,EM2B
						.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
	066434	052653	054230	052662		.EVEN	
9606	066472	123	124	123	EM356:	.WORD	CRBUT, IENBT, PRST, N3, N7, N6, WENTTO, N4, N7, N1, INSTOF, NO, N7, N1, 0
9607	066521	377				.ASCII	:STST (R) GOT BAD FEC.;<CRLF>
						.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9608	066522	066526	000000			.EVEN	
9609	066526	101	106	124	EM356B:	.WORD	EM356B,0
9610	066576	123	124	123	EM357:	.ASCII	:AFTER EXECUTING AN ILLEGAL FPP OP CODE.;
9611	066625	377				.ASCII	:STST (R) GOT BAD FEA.;<CRLF>
						.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9612	066626	066526	000000			.EVEN	
9613	066632	117	116	114	EM360:	.WORD	EM356B,0
9614	066675	123	105	124		.ASCII	:ONLY ONE WORD WRITTEN BY STST (R). ;
9615	066715	377				.ASCII	:SET FDFL ST 636!
						.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9616	066716	052552	000000			.EVEN	
9617	066722	123	124	123	EM361:	.WORD	EM4D,0
9618	066733	377				.ASCII	:STST (R)!
						.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
9619	066734	052436	000000			.EVEN	
9620	066740	116	117	116	EM362:	.WORD	EM2B,0
9621	067006	111	115	120		.ASCII	:NON-RESIDENT MEMORY MANAGEMENT TRAP - ;
9622	067050	104	111	106	EM363:	.ASCII	:IMPROPER D-SPACE ACCESS ATTEMPTED;
9623	067116	106	120	120	EM364:	.ASCII	:DIFFERENCE BETWEEN SR1 AND CALCULATED;
						.ASCII	:FPP INSTRUCTION FAILED TO ABORT, NOT ;

9624	067163	101	114	114		.ASCIZ	:ALLOWNG EXAMINATION OF SR1:
9625	067216	115	117	104	EM365:	.ASCIZ	:MODE 0 INSTRUCTION ABORTED WHEN IT SHOULD NOT HAVE:
9626	067301	106	120	120	EM366:	.ASCIZ	:FPP ACCUMULATOR WAS CHANGED IN THE EXPECTED ABORT.:
9627	067364	107	105	116	EM367:	.ASCIZ	:GENERAL REGISTER WAS CHANGED IN THE EXPECTED ABORT.:
9628	067447	106	120	120	EM370:	.ASCIZ	:FPP UNABLE TO RESTORE AN AC:
9629	067503	123	124	105	EM371:	.ASCII	:STEXP AUTO INCREMENTED/DECREMENTED R0 INCORRECTLY:<CRLF>
9630	067565	115	067	060		.ASCIZ	:M7095 ECO # 10 MIGHT NOT BE INSTALLED:
9631	000000				EM372=0		
9632	000000				EM373=0		
9633	000000				EM374=0		
9634	000000				EM375=0		
9635	000000				EM376=0		
9636	000000				EM377=0		
9637	000000				EM400=0		
9638	067633	123	000		EM401:	.ASCIZ	:S:
9639	067635	377			.BYTE	377	:FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
					.EVEN		
9640	067636	067644	052531	000000	.WORD	EM401B,EM4B,0	
9641	067644	124	106	120	EM401B:	.ASCIZ	:TFPS (R):
9642	067655	123	000		EM402:	.ASCIZ	:S:
9643	067657	377			.BYTE	377	:FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
					.EVEN		
9644	067660	067644	052552	000000	.WORD	EM401B,EM4D,0	
9645	067666	115	000		EM403:	.ASCIZ	:M:
9646	067670	377			.BYTE	377	:FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
					.EVEN		
9647	067672	067746	070005	060711	.WORD	EM403B,EM403C,PERIOD	
9648	067700	052653	067736	052662	.WORD	CRBUT,IGR7FL,PRST,N3,N5,N7,WENTTO,N4,N1,N6,INSTOF,N4,N1,N7,0	
9649	067736	107	122	067	IGR7FL:	.ASCIZ	:GR7,-FL:
9650	067746	117	122	105	EM403B:	.ASCIZ	:ORE THAN ONE WORD WRITTEN BY S:
9651	070005	124	106	120	EM403C:	.ASCIZ	:TFPS (R):
9652	070016	123	000		EM404:	.ASCIZ	:S:
9653	070020	377			.BYTE	377	:FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
					.EVEN		
9654	070022	067644	061172	000000	.WORD	EM401B,EM227B,0	
9655	070030	123	000		EM405:	.ASCIZ	:S:
9656	070032	377			.BYTE	377	:FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
					.EVEN		
9657	070034	070005	052756	052531	.WORD	EM403C,EM7B,EM4B,0	
9658	070044	123	000		EM406:	.ASCIZ	:S:
9659	070046	377			.BYTE	377	:FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
					.EVEN		
9660	070050	070005	052756	052552	.WORD	EM403C,EM7B,EM4D,0	
9661	070060	115	000		EM407:	.ASCIZ	:M:
9662	070062	377			.BYTE	377	:FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
					.EVEN		
9663	070064	067746	070005	052756	.WORD	EM403B,EM403C,EM7B	
9664	070072	052653	067736	052662	.WORD	CRBUT,IGR7FL,PRST,N3,N5,N7,WENTTO,N4,N1,N6,INSTOF,N4,N1,N7,0	
9665	070130	123	000		EM410:	.ASCIZ	:S:
9666	070132	377			.BYTE	377	:FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
					.EVEN		
9667	070134	070005	052756	061172	.WORD	EM403C,EM7B,EM227B,0	
9668	070144	123	000		EM411:	.ASCIZ	:S:
9669	070146	377			.BYTE	377	:FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
					.EVEN		
9670	070150	070156	052531	000000	.WORD	EM411B,EM4B,0	
9671	070156	124	106	120	EM411B:	.ASCIZ	:TFPS -(R):

ERROR MESSAGES

9672	070170	123	000		EM412:	.ASCIZ	IS'	
9673	070172	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9674	070174	070156	052552	000000		.WORD	EM411B,EM4D,0	
9675	070202	115	000		EM413:	.ASCIZ	IM:	
9676	070204	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9677	070206	067746	070156	060711		.WORD	EM403B,EM411B,PERIOD	
9678	070214	052653	067736	052662		.WORD	CRBUT,IGR7FL,PRST,N3,N5,N7,WENTTO,N4,N1,N6,INSTOF,N4,N1,N7,0	
9679	070252	123	000		EM414:	.ASCIZ	IS:	
9680	070254	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9681	070256	070156	061172	000000		.WORD	EM411B,EM227B,0	
9682	070264	123	000		EM415:	.ASCIZ	IS:	
9683	070266	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9684	070270	070276	052531	000000		.WORD	EM415B,EM4B,0	
9685	070276	124	106	120	EM415B:	.ASCIZ	!TFPS @ (R)+!	
9686	070311	123	000		EM416:	.ASCIZ	IS:	
9687	070313	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9688	070314	070276	052552	000000		.WORD	EM415B,EM4D,0	
9689	070322	123	000		EM417:	.ASCIZ	IS:	
9690	070324	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9691	070326	070276	070334	000000		.WORD	EM415B,EM417B,0	
9692	070334	040	104	111	EM417B:	.ASCIZ	! DID NOT DEFER THE WRITE.!	
9693	070366	123	000		EM420:	.ASCIZ	IS:	
9694	070370	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9695	070372	070276	061172	000000		.WORD	EM415B,EM227B,0	
9696	070400	123	000		EM421:	.ASCIZ	IS:	
9697	070402	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9698	070404	070412	052531	000000		.WORD	EM421B,EM4B,0	
9699	070412	124	106	120	EM421B:	.ASCIZ	!TFPS @-(R)!	
9700	070425	123	000		EM422:	.ASCIZ	IS:	
9701	070427	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9702	070430	070412	052552	000000		.WORD	EM421B,EM4D,0	
9703	070436	123	000		EM423:	.ASCIZ	IS:	
9704	070440	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9705	070442	070412	070334	000000		.WORD	EM421B,EM417B,0	
9706	070450	123	000		EM424:	.ASCIZ	IS:	
9707	070452	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9708	070454	070412	061172	000000		.WORD	EM421B,EM227B,0	
9709	070462	123	000		EM425:	.ASCIZ	IS:	
9710	070464	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9711	070466	070474	052531	000000		.WORD	EM425B,EM4B,0	
9712	070474	124	106	120	EM425B:	.ASCIZ	!TFPS N(R)!	
9713	070506	123	000		EM426:	.ASCIZ	IS:	
9714	070510	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9715	070512	070474	052552	000000		.WORD	EM425B,EM4D,0	

ERROR MESSAGES

9716	070520	115	000		EM427:	.ASCIZ	!M!		
9717	070522	377				.BYTE	377	:FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN			
9718	070524	067746	070474	060711		.WORD	EM403B,EM425B,PERIOD		
9719	070532	377				.BYTE	377	:FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN			
	070534	052653	067736	052662		.WORD	CRBUT,IGR7FL,PRST,N3,N5,N7,WENTTO,N4,N1,N6,INSTOF,N4,N1,N7,0		
9720	070572	123	000		EM430:	.ASCIZ	!S!		
9721	070574	377				.BYTE	377	:FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN			
9722	070576	070474	061172	000000		.WORD	EM425B,EM227B,0		
9723	070604	120	103	040	EM431:	.ASCIZ	!PC BAD AFTER S!		
9724	070623	377				.BYTE	377	:FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN			
9725	070624	070474	057026	000000		.WORD	EM425B,EM160B,0		
9726	070632	123	000		EM432:	.ASCIZ	!S!		
9727	070634	377				.BYTE	377	:FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN			
9728	070636	070644	052531	000000		.WORD	EM432B,EM4B,0		
9729	070644	124	106	120	EM432B:	.ASCIZ	!TFPS @N(R)!		
9730	070657	123	000		EM433:	.ASCIZ	!S!		
9731	070661	377				.BYTE	377	:FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN			
9732	070662	070644	052552	000000		.WORD	EM432B,EM4D,0		
9733	070670	115	000		EM434:	.ASCIZ	!M!		
9734	070672	377				.BYTE	377	:FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN			
9735	070674	067746	070644	060711		.WORD	EM403B,EM432B,PERIOD		
9736	070702	052653	067736	052662		.WORD	CRBUT,IGR7FL,PRST,N3,N5,N7,WENTTO,N4,N1,N6,INSTOF,N4,N1,N7,0		
9737	070740	123	000		EM435:	.ASCIZ	!S!		
9738	070742	377				.BYTE	377	:FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN			
9739	070744	070644	061172	000000		.WORD	EM432B,EM227B,0		
9740	070752	120	103	040	EM436:	.ASCIZ	!PC BAD AFTER S!		
9741	070771	377				.BYTE	377	:FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN			
9742	070772	070644	057026	000000		.WORD	EM432B,EM160B,0		
9743	071000	123	124	103	EM437:	.ASCIZ	!STCDL A,(R)+!		
9744	071015	377				.BYTE	377	:FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN			
9745	071016	052531	000000			.WORD	EM4B,0		
9746	071022	123	124	103	EM440:	.ASCIZ	!STCDL A,-(R)!		
9747	071037	377				.BYTE	377	:FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN			
9748	071040	052531	000000			.WORD	EM4B,0		
9749	071044	125	116	105	EM441:	.ASCIZ	!UNEXPECTED !		
9750	071060	376				.BYTE	376	:FLAGS 'ERTYPE' TO PRINT ERR # SPECIFIC ASCIZ MSGS, ADRS 3 LINES DOWN	
						.EVEN			
9751	071062	000441	000442	000443		.WORD	441,442,443,0		
9752	071072	071100	071147	071147		.WORD	EM441B,EM442B,EM442B		
9753	071100	106	120	120	EM441B:	.ASCIZ	!FPP !		
9754	071105	377				.BYTE	377	:FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN			
9755	071106	071112	000000			.WORD	TRAPTO,0		
9756	071112	124	122	101	TRAPTO:	.ASCIZ	!TRAP TO !		
9757	071123	376				.BYTE	376	:FLAGS 'ERTYPE' TO PRINT ERR # SPECIFIC ASCIZ MSGS, ADRS 3 LINES DOWN	
						.EVEN			

9758	071124	000441	000442	000443		.WORD	441,442,443,0
9759	071134	071142	071162	071165		.WORD	N244,N04,N10
9760	071142	062	064	064	N244:	.ASCIZ	!244.!
9761		071044			EM442=EM441		
9762	071147	103	120	125	EM442B:	.ASCIZ	!CPU !
9763	071154	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9764	071156	071112	000000			.WORD	TRAPTO,0
9765	071162	064	056	000	N04:	.ASCIZ	!4.!
9766		071044			EM443=EM441		
9767	071165	061	060	056	N10:	.ASCIZ	!10.!
9768	071171	116	105	107	EM444:	.ASCIZ	!NEG!
9769	071175	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9770	071176	057346	052466	000000		.WORD	EM176B,EM3B,0

DATA TABLE HEADERS

Address	Line 1	Line 2	Line 3	Line 4	Line 5	Label	Content
9771						.SBTTL	DATA TABLE HEADERS
9772	071204	040	000			DH1:	.ASCIZ ;
9773	071206	377				.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
9774	071210	071220	071252	071261		.WORD	DH1B,DH1C,DH1D,0
9775	071220	040	124	105	DH1B:	.ASCIZ	;; TEST.;<TAB>:PC OF CALL.;<TAB>:PC OF ;
9776	071252	105	122	122	DH1C:	.ASCIZ	;; ERROR.;
9777	071261	011	106	120	DH1D:	.ASCIZ	<TAB>:FPS.;<TAB>:FEC.;
9778	071274	040	000		DH2:	.ASCIZ	;; ;
9779	071276	377			.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN	
9780	071300	071220	071252	071310		.WORD	DH1B,DH1C,DH2B,0
9781	071310	011	107	117	DH2B:	.ASCIZ	<TAB>:GOT FPS.;<TAB>:EXPECTED FPS.;
9782	071340	040	000		DH3:	.ASCIZ	;; ;
9783	071342	377			.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN	
9784	071344	071220	071252	071354		.WORD	DH1B,DH1C,DH3B,0
9785	071354	011	107	117	DH3B:	.ASCIZ	<TAB>:GOT FEC.;<TAB>:EXPECTED FEC.;
9786	071404	040	000		DH4:	.ASCIZ	;; ;
9787	071406	377			.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN	
9788	071410	071220	071252	071420		.WORD	DH1B,DH1C,DH4B,0
9789	071420	011	107	117	DH4B:	.ASCIZ	<TAB>:GOT RO. ;<TAB>:EXPECTED RO.;
9790	071447	040	000		DH5:	.ASCIZ	;; ;
9791	071451	377			.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN	
9792	071452	071220	071252	000000		.WORD	DH1B,DH1C,0
9793		071447			DH6=DH5		
9794		071404			DH7=DH4		
9795		071447			DH10=DH5		
9796		071404			DH11=DH4		
9797		071447			DH12=DH5		
9798	071460	040	040	124	DH13:	.ASCIZ	;; TEST.;<TAB>:PC OF CALL.;<TAB>:PC OF TRAP.;
9799		071460			DH14=DH13		
9800		071447			DH15=DH5		
9801	071520	040	000		DH16:	.ASCIZ	;; ;
9802	071522	377			.BYTE	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN	
						.EVEN	
9803	071524	071220	071252	071534		.WORD	DH1B,DH1C,DH16B,0
9804	071534	011	107	117	DH16B:	.ASCIZ	<TAB>:GOT PC. ;<TAB>:EXPECTED PC.;
9805		071460			DH17=DH13		
9806		071404			DH20=DH4		
9807		071447			DH21=DH5		
9808		071447			DH22=DH5		
9809		071460			DH23=DH13		
9810		071404			DH24=DH4		
9811		071447			DH25=DH5		
9812		071460			DH26=DH13		
9813		071404			DH27=DH4		
9814		071447			DH30=DH5		
9815		071460			DH31=DH13		
9816		071404			DH32=DH4		
9817		071447			DH33=DH5		
9818		071460			DH34=DH13		
9819		071404			DH35=DH4		
9820		071447			DH36=DH5		
9821	071563	040	000		DH37:	.ASCIZ	;; ;

9822	071565					.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINFs DOWN
						.EVEN		
9823	071566	071220	071252	071576		.WORD		DH1B,DH1C,DH37B,0
9824	071576	011	107	117	DH37B:	.ASCIZ		<TAB>:GOT FP<.:<TAB>:EXPECTED FPS.:
9825		071563			DH40=	DH37		
9826	071626	040	000		DH41:	.ASCIZ	! !	
9827	071630	377				.BYTE	377	;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN		
9828	071632	071220	071252	071642		.WORD		DH1B,DH1C,DH41B,0
9829	071642	011	106	120	DH41B:	.ASCIZ		<TAB>:FPS.:<TAB>:GOT FEC. EXPECTED FEC.:
9830		071563			DH42=	DH37		
9831		071563			DH43=	DH37		
9832		071563			DH44=	DH37		
9833		071563			DH45=	DH37		
9834		071563			DH46=	DH37		
9835		071563			DH47=	DH37		
9836		071563			DH50=	DH37		
9837		071563			DH51=	DH37		
9838		071563			DH52=	DH37		
9839		071626			DH53=	DH41		
9840		071563			DH54=	DH37		
9841		071563			DH55=	DH37		
9842		071563			DH56=	DH37		
9843		071563			DH57=	DH37		
9844		071563			DH60=	DH37		
9845		071563			DH61=	DH37		
9846		071274			DH62=	DH2		
9847		071340			DH63=	DH3		
9848		071447			DH64=	DH5		
9849		071274			DH65=	DH2		
9850		071404			DH66=	DH4		
9851		071274			DH67=	DH2		
9852		071340			DH70=	DH3		
9853		071274			DH176=	DH2		
9854	071677	124	105	123	DH177:	.ASCIZ		:TESTNO ERR PC CPUERR:
9855		071447			DH71=	DH5		
9856		071274			DH72=	DH2		
9857		071460			DH107=	DH13		
9858		071447			DH73=	DH5		
9859		071404			DH74=	DH4		
9860		071274			DH75=	DH2		
9861		071460			DH76=	DH107		
9862		071447			DH77=	DH5		
9863		071404			DH100=	DH4		
9864		071274			DH101=	DH2		
9865		071460			DH102=	DH107		
9866		071447			DH103=	DH5		
9867		071404			DH104=	DH4		
9868		071274			DH105=	DH2		
9869		071460			DH106=	DH107		
9870		071447			DH110=	DH5		
9871		071404			DH111=	DH4		
9872		071274			DH112=	DH2		
9873		071460			DH113=	DH107		
9874		071447			DH114=	DH5		
9875		071404			DH115=	DH4		
9876		071274			DH116=	DH2		

9877	071460	DH117=DH107
9878	071447	DH120=DH5
9879	071404	DH121=DH4
9880	071274	DH122=DH2
9881	071460	DH123=DH107
9882	071447	DH124=DH5
9883	071404	DH125=DH4
9884	071274	DH126=DH2
9885	071460	DH127=DH107
9886	071447	DH130=DH5
9887	071274	DH131=DH2
9888	071460	DH132=DH107
9889	071447	DH133=DH5
9890	071274	DH134=DH2
9891	071447	DH135=DH5
9892	071447	DH136=DH5
9893	071274	DH137=DH2
9894	071447	DH140=DH5
9895	071404	DH141=DH4
9896	071274	DH142=DH2
9897	071447	DH143=DH5
9898	071404	DH144=DH4
9899	071274	DH145=DH2
9900	071447	DH146=DH5
9901	071404	DH147=DH4
9902	071274	DH150=DH2
9903	071447	DH151=DH5
9904	071404	DH152=DH4
9905	071274	DH153=DH2
9906	071447	DH154=DH5
9907	071404	DH155=DH4
9908	071274	DH156=DH2
9909	071447	DH157=DH5
9910	071404	DH160=DH4
9911	071274	DH161=DH2
9912	071447	DH162=DH5
9913	071274	DH163=DH2
9914	071520	DH164=DH16
9915	071563	DH165=DH37
9916	071563	DH166=DH37
9917	071563	DH167=DH37
9918	071563	DH170=DH37
9919	071563	DH171=DH37
9920	071563	DH172=DH37
9921	071626	DH173=DH41
9922	071626	DH174=DH41
9923	071626	DH175=DH41
9924	071563	DH200=DH37
9925	071563	DH201=DH37
9926	071563	DH202=DH37
9927	071563	DH203=DH37
9928	071563	DH204=DH37
9929	071563	DH205=DH37
9930	071563	DH206=DH37
9931	071563	DH207=DH37
9932	071563	DH210=DH37
9933	071563	DH211=DH37

9934		071563				DH212=DH37					
9935		071563				DH213=DH37					
9936		071563				DH214=DH37					
9937		071520				DH215=DH16					
9938		071447				DH216=DH5					
9939		071404				DH217=DH4					
9940		071274				DH220=DH2					
9941		071520				DH221=DH16					
9942		071447				DH222=DH5					
9943		071404				DH223=DH4					
9944		071274				DH224=DH2					
9945		071404				DH225=DH4					
9946		071274				DH226=DH2					
9947	071726	040		000		DH227: .ASCIZ	;;				
9948	071730	377				.BYTE	377		:FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN		
						.EVEN					
9949	071732	071220	071740	000000		.WORD	DH1B,DH227B,0				
9950	071740	124	122	101		.ASCIZ	!TRAP.!				
9951		071404				DH227B: .ASCIZ					
9952		071274				DH230=DH4					
9953		071726				DH231=DH2					
9954		071404				DH232=DH227					
9955		071274				DH233=DH4					
9956		071726				DH234=DH2					
9957		071404				DH235=DH227					
9958		071274				DH236=DH4					
9959		071726				DH237=DH2					
9960		071404				DH240=DH227					
9961		071274				DH241=DH4					
9962		071726				DH242=DH2					
9963		071404				DH243=DH227					
9964		071274				DH244=DH4					
9965		071520				DH245=DH2					
9966		071726				DH246=DH16					
9967		071404				DH247=DH227					
9968		071274				DH250=DH4					
9969		071520				DH251=DH2					
9970		071726				DH252=DH16					
9971		071520				DH253=DH227					
9972		071726				DH254=DH16					
9973		071404				DH255=DH227					
9974		071274				DH256=DH4					
9975		071563				DH257=DH2					
9976		071563				DH260=DH37					
9977		071563				DH261=DH37					
9978		071563				DH262=DH37					
9979		071563				DH263=DH37					
9980		071563				DH264=DH37					
9981		071563				DH265=DH37					
9982		071563				DH266=DH37					
9983		071563				DH267=DH37					
9984		071563				DH270=DH37					
9985		071563				DH271=DH37					
9986		071563				DH272=DH37					
9987		071563				DH273=DH37					
9988		071563				DH274=DH37					
9989		071563				DH275=DH37					
						DH276=DH37					

9990	071563				DH277=DH37
9991	071563				DH300=DH37
9992	071563				DH301=DH37
9993	071563				DH302=DH37
9994	071626				DH303=DH41
9995	071563				DH304=DH37
9996	071563				DH305=DH37
9997	071563				DH306=DH37
9998	071563				DH307=DH37
9999	071563				DH310=DH37
10000	071563				DH311=DH37
10001	071563				DH312=DH37
10002	071563				DH313=DH37
10003	071563				DH314=DH37
10004	071563				DH315=DH37
10005	071563				DH316=DH37
10006	071563				DH317=DH37
10007	071563				DH320=DH37
10008	071563				DH321=DH37
10009	071563				DH322=DH37
10010	071563				DH323=DH37
10011	071626				DH324=DH41
10012	071563				DH325=DH37
10013	071563				DH326=DH37
10014	071563				DH327=DH37
10015	071563				DH330=DH37
10016	071563				DH331=DH37
10017	071563				DH332=DH37
10018	071563				DH333=DH37
10019	071563				DH334=DH37
10020	071563				DH335=DH37
10021	071563				DH336=DH37
10022	071563				DH337=DH37
10023	071563				DH340=DH37
10024	071563				DH341=DH37
10025	071563				DH342=DH37
10026	071563				DH343=DH37
10027	071563				DH344=DH37
10028	071563				DH345=DH37
10029	071563				DH346=DH37
10030	071563				DH347=DH37
10031	071563				DH350=DH37
10032	071460				DH351=DH13
10033	071563				DH352=DH37
10034	071563				DH353=DH37
10035	071563				DH354=DH37
10036	071563				DH355=DH37
10037	071404				DH356=DH11
10038	071746	040	000		DH357: .ASCIZ ! !
10039	071750	377			.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
					.EVEN
10040	071752	071220	071252	071762	.WORD DH1B,DH1C,DH357B,0
10041	071762	011	107	117	DH357B: .ASCIZ <TAB>!GOT FEA.!<TAB>!EXPECTED FEA.!
10042		071460			DH360=DH13
10043		071274			DH361=DH2
10044	072012	040	000		DH362: .ASCIZ ! !
10045	072014	377			.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN

Address	Field 1	Field 2	Field 3	Field 4	Field 5	Field 6	Field 7
10046	072016	071220	071252	072026			.EVEN
10047	072026	011	115	115	DH362B:		.WORD DH1B,DH1C,DH362B,0
10048	072034	040	000		DH363:		.ASCIZ <TAB>:MMR0:
10049	072036	377					.ASCIZ
							.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
							.EVEN
10050	072040	071220	071252	072050			.WORD DH1B,DH1C,DH363B,0
10051	072050	011	123	122	DH363B:		.ASCIZ <TAB>:SR1:<TAB>:CALCD:<TAB>:EXPECTED:
10052	072074	040	000		DH364:		.ASCIZ
10053	072076	377					.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
							.EVEN
10054	072100	071220	072106	000000			.WORD DH1B,DH364B,0
10055	072106	111	116	123	DH364B:		.ASCIZ ;INSTRUCTION FAILING TO ABORT:
10056	072143	040	000		DH365:		.ASCIZ
10057	072145	377					.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
							.EVEN
10058	072146	071220	071252	000000			.WORD DH1B,DH1C,0
10059	072154	040	000		DH366:		.ASCIZ
10060	072156	377					.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
							.EVEN
10061	072160	071220	071252	072170			.WORD DH1B,DH1C,DH366B,0
10062	072170	011	101	103	DH366B:		.ASCIZ <TAB>:AC # CHANGED:
10063	072206	040	000		DH367:		.ASCIZ
10064	072210	377					.BYTE 377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
							.EVEN
10065	072212	071220	071252	072222			.WORD DH1B,DH1C,DH367B,0
10066	072222	011	122	105	DH367B:		.ASCIZ <TAB>:REG #:<TAB>:RECEIVED:<TAB>:EXPECTED:
10067	072253	040	040	124	DH370:		.ASCIZ ; TEST:<TAB>:PC OF CALL:<TAB>:AC #:<TAB>:PC OF ERROR:
10068	072316	040	040	124	DH371:		.ASCIZ ; TEST:<TAB>:PC OF CALL:<TAB>:STEXP OP CODE:<TAB>
10069	072356	105	130	120			.ASCIZ ;EXPCTD RECVD PC OF ERROR:
10070		000000			DH372=0		
10071		000000			DH373=0		
10072		000000			DH374=0		
10073		000000			DH375=0		
10074		000000			DH376=0		
10075		000000			DH377=0		
10076		000000			DH400=0		
10077		071404			DH401=DH4		
10078		071274			DH402=DH2		
10079		071460			DH403=DH13		
10080		071726			DH404=DH227		
10081		071404			DH405=DH4		
10082		071274			DH406=DH2		
10083		071460			DH407=DH13		
10084		071726			DH410=DH227		
10085		071404			DH411=DH4		
10086		071274			DH412=DH2		
10087		071460			DH413=DH13		
10088		071726			DH414=DH227		
10089		071404			DH415=DH4		
10090		071274			DH416=DH2		
10091		071460			DH417=DH13		
10092		071726			DH420=DH227		
10093		071404			DH421=DH4		
10094		071274			DH422=DH2		
10095		071460			DH423=DH13		
10096		071726			DH424=DH227		

DATA TABLE HEADERS

10097		071404				DH425=DH4	
10098		071274				DH426=DH2	
10099		071460				DH427=DH13	
10100		071726				DH430=DH227	
10101		071460				DH431=DH13	
10102		071404				DH432=DH4	
10103		071274				DH433=DH2	
10104		071460				DH434=DH13	
10105		071726				DH435=DH227	
10106		071460				DH436=DH13	
10107		071404				DH437=DH4	
10108		071404				DH440=DH4	
10109	072412	040	000			DH441: .ASCIZ	::
10110	072414	377				.BYTF	377 ;FLAGS 'ERTYPE' TO PRINT ADDT'L ASCIZ MSGS, ADDRESSES 2 LINES DOWN
						.EVEN	
10111	072416	071220	071252	072426		.WORD	DH1B,DH1C,DH441B,0
10112	072426	011	106	105		.ASCIZ	<TAB>!FEC.!
10113		072143				DH442=DH365	
10114		072143				DH443=DH442	
10115		071340				DH444=DH3	

						.SBTTL		FORMAT SPECIFICATIONS FOR THE DATA TABLES	
10116									
10117	072434	004	000	005	DF1:	.BYTE	4,0,5,0,5,0,0		
10118	072443	004	000	005	DF2:	.BYTE	4,0,5,0,5,0,5,0		
10119		072443			DF3=DF2				
10120		072443			DF4=DF2				
10121	072453	004	000	005	DF5:	.BYTE	4,0,5,0,5,5,2,5,5,2		
10122	072465	004	000	005	DF6:	.BYTE	4,0,5,0		
10123		072443			DF7=DF4				
10124		072453			DF10=DF5				
10125		072443			DF11=DF4				
10126	072471	004	000	005	DF12:	.BYTE	4,0,5,0,5,5,3,5,5,3		
10127		072465			DF13=DF6				
10128		072465			DF14=DF6				
10129		072471			DF15=DF12				
10130		072443			DF16=DF2				
10131		072465			DF17=DF6				
10132		072443			DF20=DF2				
10133		072471			DF21=DF12				
10134		072471			DF22=DF12				
10135		072465			DF23=DF6				
10136		072443			DF24=DF2				
10137		072471			DF25=DF12				
10138		072465			DF26=DF6				
10139		072443			DF27=DF2				
10140		072471			DF30=DF12				
10141		072465			DF31=DF6				
10142		072443			DF32=DF2				
10143		072471			DF33=DF12				
10144		072465			DF34=DF6				
10145		072443			DF35=DF2				
10146		072471			DF36=DF12				
10147	072503	004	000	005	DF37:	.BYTE	4,0,5,0,5,0,5,0,5,5,3,5,5,3,5,5,3		
10148		072503			DF40=DF37				
10149	072524	004	000	005	DF41:	.BYTE	4,0,5,0,5,0,0,0,5,5,3,5,5,3,5,5,3		
10150		072503			DF42=DF37				
10151		072503			DF43=DF37				
10152		072503			DF44=DF37				
10153		072503			DF45=DF37				
10154		072503			DF46=DF37				
10155		072503			DF47=DF37				
10156		072503			DF50=DF37				
10157		072503			DF51=DF37				
10158		072503			DF52=DF37				
10159		072503			DF53=DF37				
10160		072503			DF54=DF37				
10161		072503			DF55=DF37				
10162		072503			DF56=DF37				
10163		072503			DF57=DF37				
10164		072503			DF60=DF37				
10165		072503			DF61=DF37				
10166		072443			DF62=DF2				
10167		072443			DF63=DF2				
10168		072453			DF64=DF5				
10169		072443			DF65=DF2				
10170		072443			DF66=DF2				
10171		072443			DF67=DF2				
10172		072443			DF70=DF2				

10173		072443			DF176=DF2	
10174	072545	004	000	000	DF177: .BYTE	4,0,0
10175	072550	004	000	005	DF71: .BYTE	4,0,5,0,5,5,3,5,5,3,5,5,3
10176		072443			DF72=DF2	
10177		072465			DF107=DF6	
10178		072550			DF73=DF71	
10179		072443			DF74=DF2	
10180		072443			DF75=DF2	
10181		072465			DF76=DF6	
10182		072550			DF77=DF71	
10183		072443			DF100=DF2	
10184		072443			DF101=DF2	
10185		072465			DF102=DF6	
10186		072550			DF103=DF71	
10187		072443			DF104=DF2	
10188		072443			DF105=DF2	
10189		072465			DF106=DF6	
10190		072550			DF110=DF71	
10191		072443			DF111=DF2	
10192		072443			DF112=DF2	
10193		072465			DF113=DF6	
10194		072550			DF114=DF71	
10195		072443			DF115=DF2	
10196		072443			DF116=DF2	
10197		072465			DF117=DF6	
10198		072550			DF120=DF71	
10199		072443			DF121=DF2	
10200		072443			DF122=DF2	
10201		072465			DF123=DF6	
10202		072550			DF124=DF71	
10203		072443			DF125=DF2	
10204		072443			DF126=DF2	
10205		072465			DF127=DF6	
10206		072550			DF130=DF71	
10207		072443			DF131=DF2	
10208		072465			DF132=DF6	
10209		072550			DF133=DF71	
10210		072443			DF134=DF2	
10211		072471			DF135=DF12	
10212		072471			DF136=DF12	
10213		072443			DF137=DF2	
10214		072471			DF140=DF12	
10215		072443			DF141=DF2	
10216		072443			DF142=DF2	
10217		072471			DF143=DF12	
10218		072443			DF144=DF2	
10219		072443			DF145=DF2	
10220		072471			DF146=DF12	
10221		072443			DF147=DF2	
10222		072443			DF150=DF2	
10223		072471			DF151=DF12	
10224		072443			DF152=DF2	
10225		072443			DF153=DF2	
10226		072471			DF154=DF12	
10227		072443			DF155=DF2	
10228		072443			DF156=DF2	
10229		072471			DF157=DF12	

10230	072443			DF160=DF2	
10231	072443			DF161=DF2	
10232	072471			DF162=DF12	
10233	072443			DF163=DF2	
10234	072443			DF164=DF2	
10235	072443			DF215=DF2	
10236	072471			DF216=DF12	
10237	072443			DF217=DF2	
10238	072443			DF220=DF2	
10239	072443			DF221=DF2	
10240	072471			DF222=DF12	
10241	072443			DF223=DF2	
10242	072443			DF224=DF2	
10243	072503			DF165=DF37	
10244	072503			DF166=DF37	
10245	072503			DF167=DF37	
10246	072503			DF170=DF37	
10247	072503			DF171=DF37	
10248	072503			DF172=DF37	
10249	072524			DF173=DF41	
10250	072524			DF174=DF41	
10251	072524			DF175=DF41	
10252	072503			DF200=DF37	
10253	072503			DF201=DF37	
10254	072503			DF202=DF37	
10255	072503			DF203=DF37	
10256	072503			DF204=DF37	
10257	072503			DF205=DF37	
10258	072503			DF206=DF37	
10259	072503			DF207=DF37	
10260	072503			DF210=DF37	
10261	072503			DF211=DF37	
10262	072503			DF212=DF37	
10263	072503			DF213=DF37	
10264	072503			DF214=DF37	
10265	072565	000	005	DF225: .BYTE	4.0.5.0.5.0.5.0
10266	072565			DF226=DF225	
10267	072575	000	005	DF227: .BYTE	4.0.5.0
10268	072565			DF230=DF225	
10269	072565			DF231=DF225	
10270	072575			DF232=DF227	
10271	072565			DF233=DF225	
10272	072565			DF234=DF225	
10273	072575			DF235=DF227	
10274	072565			DF236=DF225	
10275	072565			DF237=DF225	
10276	072575			DF240=DF227	
10277	072565			DF241=DF225	
10278	072565			DF242=DF225	
10279	072575			DF243=DF227	
10280	072565			DF244=DF225	
10281	072565			DF245=DF225	
10282	072565			DF246=DF225	
10283	072575			DF247=DF227	
10284	072565			DF250=DF225	
10285	072565			DF251=DF225	
10286	072565			DF252=DF225	

FORMAT SPECIFICATIONS FOR THE DATA TABLES

10287	072575			DF253=DF227	
10288	072565			DF254=DF225	
10289	072575			DF255=DF227	
10290	072565			DF256=DF225	
10291	072565			DF257=DF225	
10292	072601	000	005	DF260: .BYTE	4,0,5,0,5,0,5,0,5,5,2,5,5,2,5,5,2
10293	072601			DF261=DF260	
10294	072601			DF262=DF260	
10295	072601			DF263=DF260	
10296	072601			DF264=DF260	
10297	072601			DF265=DF260	
10298	072601			DF266=DF260	
10299	072601			DF267=DF260	
10300	072601			DF270=DF260	
10301	072601			DF271=DF260	
10302	072601			DF272=DF260	
10303	072622	000	005	DF273: .BYTE	4,0,5,0,5,0,5,0,5,5,2,5,5,3,5,5,3
10304	072622			DF274=DF273	
10305	072622			DF275=DF273	
10306	072622			DF276=DF273	
10307	072622			DF277=DF273	
10308	072622			DF300=DF273	
10309	072643	000	005	DF301: .BYTE	4,0,5,0,5,0,5,0,5,5,3,5,5,0,5,5,3,5,5,3
10310	072643			DF302=DF301	
10311	072667	000	005	DF303: .BYTE	4,0,5,0,5,0,0,0,5,5,3,5,5,0,5,5,3,5,5,3
10312	072643			DF304=DF301	
10313	072643			DF305=DF301	
10314	072643			DF306=DF301	
10315	072643			DF307=DF301	
10316	072643			DF310=DF301	
10317	072643			DF311=DF301	
10318	072643			DF312=DF301	
10319	072643			DF313=DF301	
10320	072643			DF314=DF301	
10321	072643			DF315=DF301	
10322	072643			DF316=DF301	
10323	072643			DF317=DF301	
10324	072643			DF320=DF301	
10325	072643			DF321=DF301	
10326	072713	000	005	DF322: .BYTE	4,0,5,0,5,0,5,0,5,5,3,5,5,2,5,5,2
10327	072713			DF323=DF322	
10328	072734	000	005	DF324: .BYTE	4,0,5,0,5,0,0,0,5,5,3,5,5,2,5,5,2
10329	072713			DF325=DF322	
10330	072713			DF326=DF322	
10331	072713			DF327=DF322	
10332	072713			DF330=DF322	
10333	072713			DF331=DF322	
10334	072713			DF332=DF322	
10335	072713			DF333=DF322	
10336	072713			DF334=DF322	
10337	072713			DF335=DF322	
10338	072713			DF336=DF322	
10339	072713			DF337=DF322	
10340	072713			DF340=DF322	
10341	072713			DF341=DF322	
10342	072713			DF342=DF322	
10343	072713			DF343=DF322	

10344	072713			DF344=DF322	
10345	072713			DF345=DF322	
10346	072713			DF346=DF322	
10347	072755	000	005	DF347: .BYTE	4,0,5,0,5,0,5,0,5,5,3,5,5,0,5,5,0
10348	072755			DF350=DF347	
10349	072575			DF351=DF227	
10350	072755			DF352=DF347	
10351	072755			DF353=DF347	
10352	072755			DF354=DF347	
10353	072755			DF355=DF347	
10354	072565			DF356=DF225	
10355	072565			DF357=DF225	
10356	072575			DF360=DF227	
10357	072776	000	005	DF361: .BYTF	4,0,5,0,5,0,5,0,5,5,0,5,5,0,5,5,0,5,5,0
10358	073022	000	005	DF362: .BYTE	4,0,5,0,5,0
10359	073030	000	005	DF363: .BYTE	4,0,5,0,5,0,0,0
10360	072465			DF364=DF6	
10361	072465			DF365=DF6	
10362	073040	000	005	DF366: .BYTE	4,0,5,0,5,0,5,5,0,0,0,0,5,5,0,0,0,0
10363	073062	000	005	DF367: .BYTE	4,0,5,0,5,0,0,0
10364	073072	000	005	DF370: .BYTE	4,0,5,0,0,5,5,0,0,0,0,5,5,0,0,0,0
10365	073062			DF371=DF367	
10366	000000			DF372=0	
10367	000000			DF373=0	
10368	000000			DF374=0	
10369	000000			DF375=0	
10370	000000			DF376=0	
10371	000000			DF377=0	
10372	000000			DF400=0	
10373	072565			DF401=DF225	
10374	072565			DF402=DF225	
10375	072575			DF403=DF227	
10376	072575			DF404=DF227	
10377	072565			DF405=DF225	
10378	072565			DF406=DF225	
10379	072575			DF407=DF227	
10380	072575			DF410=DF227	
10381	072565			DF411=DF225	
10382	072565			DF412=DF225	
10383	072575			DF413=DF227	
10384	072575			DF414=DF227	
10385	072565			DF415=DF225	
10386	072565			DF416=DF225	
10387	072575			DF417=DF227	
10388	072575			DF420=DF227	
10389	072565			DF421=DF225	
10390	072565			DF422=DF225	
10391	072575			DF423=DF227	
10392	072575			DF424=DF227	
10393	072565			DF425=DF225	
10394	072565			DF426=DF225	
10395	072575			DF427=DF227	
10396	072575			DF430=DF227	
10397	072575			DF431=DF227	
10398	072565			DF432=DF225	
10399	072565			DF433=DF225	
10400	072575			DF434=DF227	

10401	072575			DF435=DF227	
10402	072575			DF436=DF227	
10403	072565			DF437=DF225	
10404	072565			DF440=DF225	
10405	073113	004	000	DF441: .BYTE	4,0,5,0,5,0
10406	073113			DF442=DF441	
10407	073113			DF443=DF441	
10408	072443			DF444=DF2	
10409				.EVEN	

Line	Code	Code	Code	Code	Code	Label	Text
10410						.SBTTL	ERROR MESSAGE DATA TABLES
10411	073122	001232	001234	052167		DT1: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,\$TAB,\$TMP3,\$TMP4,0
10412	073142	001232	001234	052167		DT2: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,\$TAB,\$TMP3,\$TAB,\$TMP5,0
10413	073164	001232	001234	052167		DT3: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,\$TAB,\$TMP4,\$TAB,\$TMP6,0
10414	073206	001232	001234	052167		DT4: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,\$TAB,\$TMP4,\$TAB,\$TMP3,0
10415	073230	001232	001234	052167		DT5: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,\$CRLF,\$MS1,\$TMP3
10416	073246	001313	052207	001242		.WORD	\$CRLF,\$MS2,\$TMP4,0
10417	073256	001232	001234	052167		DT6: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,0
10418		073206				DT7=DT4	
10419		073230				DT10=DT5	
10420		073206				DT11=DT4	
10421		073230				DT12=DT5	
10422		073256				DT13=DT6	
10423		073256				DT14=DT6	
10424		073230				DT15=DT5	
10425	073270	001232	001234	052167		DT16: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,\$TAB,\$TMP4,\$TAB,\$TMP3,0
10426		073256				DT17=DT6	
10427		073270				DT20=DT16	
10428		073230				DT21=DT5	
10429		073230				DT22=DT5	
10430		073256				DT23=DT6	
10431		073270				DT24=DT16	
10432		073230				DT25=DT5	
10433		073256				DT26=DT6	
10434		073270				DT27=DT16	
10435		073230				DT30=DT5	
10436		073256				DT31=DT6	
10437		073270				DT32=DT16	
10438		073230				DT33=DT5	
10439		073256				DT34=DT6	
10440		073270				DT35=DT16	
10441		073230				DT36=DT5	
10442	073312	001232	001234	052167		DT37: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,\$TAB,\$TMP7,\$TAB,\$TMP10,\$CRLF
10443	073334	052247	001240	001313		.WORD	\$MS4,\$TMP3,\$CRLF,\$MS1,\$TMP4,\$CRLF,\$MS2,\$TMP5,0
10444		073312				DT40=DT37	
10445	073356	001232	001234	052167		DT41: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,\$TAB,\$TMP7,\$TMP11,\$TMP12
10446	073376	001313	052247	001240		.WORD	\$CRLF,\$MS4,\$TMP3,\$CRLF,\$MS1,\$TMP4,\$CRLF,\$MS2,\$TMP5,0
10447		073312				DT42=DT37	
10448		073312				DT43=DT37	
10449		073312				DT44=DT37	
10450		073312				DT45=DT37	
10451		073312				DT46=DT37	
10452		073312				DT47=DT37	
10453		073312				DT50=DT37	
10454		073312				DT51=DT37	
10455		073312				DT52=DT37	
10456		073356				DT53=DT41	
10457		073312				DT54=DT37	
10458		073312				DT55=DT37	
10459		073312				DT56=DT37	
10460		073312				DT57=DT37	
10461		073312				DT60=DT37	
10462		073312				DT61=DT37	
10463		073270				DT62=DT16	
10464		073270				DT63=DT16	
10465		073230				DT64=DT5	
10466		073270				DT65=DT16	

ERROR MESSAGE DATA TABLES

10467		073206				DT66=DT4	
10468		073206				DT67=DT4	
10469		073206				DT70=DT4	
10470		073206				DT176=DT4	
10471	073422	001232	001116	046506		DT177: .WORD	\$TMP0,\$ERRPC,CPSAVE,0
10472	073432	001232	001234	052167		DT71: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,\$CRLF,MS3,\$TMP3,\$CRLF,MS
10473	073454	001244	001313	052207		.WORD	\$TMP5,\$CRLF,MS2,\$TMP4,0
10474		073206				DT72=DT4	
10475		073256				DT107=DT6	
10476	073466	001232	001234	052167		DT73: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,\$CRLF,MS4,\$TMP4
10477	073504	001313	052171	001244		.WORD	\$CRLF,MS1,\$TMP5,\$CRLF,MS2,\$TMP3,0
10478		073206				DT74=DT4	
10479		073142				DT75=DT2	
10480		073256				DT76=DT6	
10481		073466				DT77=DT73	
10482		073206				DT100=DT4	
10483		073142				DT101=DT2	
10484		073256				DT102=DT6	
10485		073466				DT103=DT73	
10486		073206				DT104=DT4	
10487		073142				DT105=DT2	
10488		073256				DT106=DT6	
10489		073466				DT110=DT73	
10490		073206				DT111=DT4	
10491		073142				DT112=DT2	
10492		073256				DT113=DT6	
10493		073466				DT114=DT73	
10494		073206				DT115=DT4	
10495		073142				DT116=DT2	
10496		073256				DT117=DT6	
10497		073466				DT120=DT73	
10498		073206				DT121=DT4	
10499		073142				DT122=DT2	
10500		073256				DT123=DT6	
10501		073466				DT124=DT73	
10502		073206				DT125=DT4	
10503		073142				DT126=DT2	
10504		073256				DT127=DT6	
10505		073466				DT130=DT73	
10506		073142				DT131=DT2	
10507		073256				DT132=DT6	
10508		073466				DT133=DT73	
10509		073142				DT134=DT2	
10510		073230				DT135=DT5	
10511		073230				DT136=DT5	
10512		073270				DT137=DT16	
10513		073230				DT140=DT5	
10514		073206				DT141=DT4	
10515		073206				DT142=DT4	
10516		073230				DT143=DT5	
10517		073206				DT144=DT4	
10518		073206				DT145=DT4	
10519		073230				DT146=DT5	
10520		073206				DT147=DT4	
10521		073206				DT150=DT4	
10522		073230				DT151=DT5	
10523		073206				DT152=DT4	

10524	073206	DT153=DT4
10525	073230	DT154=DT5
10526	073206	DT155=DT4
10527	073206	DT156=DT4
10528	073230	DT157=DT5
10529	073206	DT160=DT4
10530	073206	DT161=DT4
10531	073230	DT162=DT5
10532	073206	DT163=DT4
10533	073206	DT164=DT4
10534	073206	DT215=DT4
10535	073230	DT216=DT5
10536	073206	DT217=DT4
10537	073206	DT220=DT4
10538	073206	DT221=DT4
10539	073230	DT222=DT5
10540	073206	DT223=DT4
10541	073206	DT224=DT4
10542	073312	DT165=DT37
10543	073312	DT166=DT37
10544	073312	DT167=DT37
10545	073312	DT170=DT37
10546	073312	DT171=DT37
10547	073312	DT172=DT37
10548	073356	DT173=DT41
10549	073356	DT174=DT41
10550	073356	DT175=DT41
10551	073312	DT200=DT37
10552	073312	DT201=DT37
10553	073312	DT202=DT37
10554	073312	DT203=DT37
10555	073312	DT204=DT37
10556	073312	DT205=DT37
10557	073312	DT206=DT37
10558	073312	DT207=DT37
10559	073312	DT210=DT37
10560	073312	DT211=DT37
10561	073312	DT212=DT37
10562	073312	DT213=DT37
10563	073312	DT214=DT37
10564	073206	DT225=DT4
10565	073206	DT226=DT4
10566	073256	DT227=DT6
10567	073206	DT230=DT4
10568	073206	DT231=DT4
10569	073256	DT232=DT6
10570	073206	DT233=DT4
10571	073206	DT234=DT4
10572	073256	DT235=DT6
10573	073206	DT236=DT4
10574	073206	DT237=DT4
10575	073256	DT240=DT6
10576	073206	DT241=DT4
10577	073206	DT242=DT4
10578	073256	DT243=DT6
10579	073206	DT244=DT4
10580	073206	DT245=DT4

10581	073206			DT246=DT4	
10582	073256			DT247=DT6	
10583	073206			DT250=DT4	
10584	073206			DT251=DT4	
10585	073206			DT252=DT4	
10586	073256			DT253=DT6	
10587	073206			DT254=DT4	
10588	073256			DT255=DT6	
10589	073206			DT256=DT4	
10590	073206			DT257=DT4	
10591	073312			DT260=DT37	
10592	073312			DT261=DT37	
10593	073312			DT262=DT37	
10594	073312			DT263=DT37	
10595	073312			DT264=DT37	
10596	073312			DT265=DT37	
10597	073312			DT266=DT37	
10598	073312			DT267=DT37	
10599	073312			DT270=DT37	
10600	073312			DT271=DT37	
10601	073312			DT272=DT37	
10602	073312			DT273=DT37	
10603	073312			DT274=DT37	
10604	073312			DT275=DT37	
10605	073312			DT276=DT37	
10606	073312			DT277=DT37	
10607	073312			DT300=DT37	
10608	073522	001234	052167	DT301: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,\$TAB,\$TMP7,\$TAB,\$TMP10
10609	073542	001313	052231	.WORD	\$CRLF,MS10,\$TMP3,\$CRLF,MS11,\$TMP4
10610	073556	001313	052171	.WORD	\$CRLF,MS1,\$TMP6,\$CRLF,MS2,\$TMP5,0
10611	073522			DT302=DT301	
10612	073574	001234	052167	DT303: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,\$TAB,\$TMP7,\$TMP11,\$TMP12
10613	073614	001313	052231	.WORD	\$CRLF,MS10,\$TMP3,\$CRLF,MS11,\$TMP4
10614	073630	001313	052171	.WORD	\$CRLF,MS1,\$TMP6,\$CRLF,MS2,\$TMP5,0
10615	073522			DT304=DT301	
10616	073522			DT305=DT301	
10617	073522			DT306=DT301	
10618	073522			DT307=DT301	
10619	073522			DT310=DT301	
10620	073522			DT311=DT301	
10621	073522			DT312=DT301	
10622	073522			DT313=DT301	
10623	073522			DT314=DT301	
10624	073522			DT315=DT301	
10625	073522			DT316=DT301	
10626	073522			DT317=DT301	
10627	073522			DT320=DT301	
10628	073522			DT321=DT301	
10629	073646	001234	052167	DT322: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,\$TAB,\$TMP7,\$TAB,\$TMP10
10630	073666	001313	052231	.WORD	\$CRLF,MS10,\$TMP3,\$CRLF,MS1,\$TMP4,\$CRLF,MS2,\$TMP5,0
10631	073646			DT323=DT322	
10632	073712	001234	052167	DT324: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,\$TAB,\$TMP7,\$TMP11,\$TMP12
10633	073732	001313	052231	.WORD	\$CRLF,MS10,\$TMP3,\$CRLF,MS1,\$TMP4,\$CRLF,MS2,\$TMP5,0
10634	073646			DT325=DT322	
10635	073646			DT326=DT322	
10636	073646			DT327=DT322	
10637	073646			DT330=DT322	

10638	073646				DT331=DT322		
10639	073646				DT332=DT322		
10640	073646				DT333=DT322		
10641	073646				DT334=DT322		
10642	073646				DT335=DT322		
10643	073646				DT336=DT322		
10644	073646				DT337=DT322		
10645	073646				DT340=DT322		
10646	073646				DT341=DT322		
10647	073646				DT342=DT322		
10648	073646				DT343=DT322		
10649	073646				DT344=DT322		
10650	073646				DT345=DT322		
10651	073646				DT346=DT322		
10652	073646				DT347=DT322		
10653	073646				DT350=DT322		
10654	073256				DT351=DT6		
10655	073646				DT352=DT322		
10656	073646				DT353=DT322		
10657	073646				DT354=DT322		
10658	073646				DT355=DT322		
10659	073142				DT356=DT2		
10660	073164				DT357=DT3		
10661	073256				DT360=DT6		
10662	073522				DT361=DT302		
10663	073756	001232	001234	052167	DT362: .WORD	STMP0,STMP1,STAB,STMP2,STAB,STMP3,0	
10664	073774	001232	001234	052167	DT363: .WORD	STMP0,STMP1,STAB,STMP13,STAB,STMP2,STMP3,EXPCD,0	
10665	074016	001232	001234	052167	DT364: .WORD	STMP0,STMP1,STAB,STMP13,0	
10666	073256				DT365=DT6		
10667	074030	001232	001234	052167	DT366: .WORD	STMP0,STMP1,STAB,STMP13,STAB,STMP2	
10668	074044	001313	052207	001240	.WORD	SCRLF,MS2,STMP3,STMP4,STMP6,STMP7	
10669	074060	001313	052171	001252	.WORD	SCRLF,MS1,STMP10,STMP11,STMP12,STMP21,0	
10670	074076	001232	001234	052167	DT367: .WORD	STMP0,STMP1,STAB,STMP2,STMP13,STAB,STMP6,STAB,STMP3,0	
10671	074122	001232	001234	052167	DT370: .WORD	STMP0,STMP1,STAB,STMP2,STMP13	
10672	074134	001313	052326	001240	.WORD	SCRLF,MS21,STMP3,STMP4,STMP6,STMP7	
10673	074150	001313	052314	001252	.WORD	SCRLF,MS20,STMP10,STMP11,STMP12,STMP21,0	
10674	074166	001232	001234	052167	DT371: .WORD	STMP0,STMP1,STAB,STMP13,STAB,STMP3,STMP4,STMP2,0	
10675	000000				DT372=0		
10676	000000				DT373=0		
10677	000000				DT374=0		
10678	000000				DT375=0		
10679	000000				DT376=0		
10680	000000				DT377=0		
10681	000000				DT400=0		
10682	073206				DT401=DT4		
10683	073206				DT402=DT4		
10684	073256				DT403=DT6		
10685	073256				DT404=DT6		
10686	073206				DT405=DT4		
10687	073206				DT406=DT4		
10688	073256				DT407=DT6		
10689	073256				DT410=DT6		
10690	073206				DT411=DT4		
10691	073206				DT412=DT4		
10692	073256				DT413=DT6		
10693	073256				DT414=DT6		
10694	073206				DT415=DT4		

10695		073206			DT416=DT4	
10696		073256			DT417=DT6	
10697		073256			DT420=DT6	
10698		073206			DT421=DT4	
10699		073206			DT422=DT4	
10700		073256			DT423=DT6	
10701		073256			DT424=DT6	
10702		073206			DT425=DT4	
10703		073206			DT426=DT4	
10704		073256			DT427=DT6	
10705		073256			DT430=DT6	
10706		073256			DT431=DT6	
10707		073206			DT432=DT4	
10708		073206			DT433=DT4	
10709		073256			DT434=DT6	
10710		073256			DT435=DT6	
10711		073256			DT436=DT6	
10712		073206			DT437=DT4	
10713		073206			DT440=DT4	
10714	074210	001232	001234	052167	DT441: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,\$TAB,\$TMP3,0
10715	074226	001232	001234	052167	DT442: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,0
10716		074226			DT443=DT442	
10717		073206			DT444=DT4	
10718		000001			.END	

ABASE = 000000	AUSWR = 000000	DF113 = 072465	DF176 = 072443	DF26 = 072465
ACDW1 = 000000	AVECT1 = 000000	DF114 = 072550	DF177 = 072545	DF260 = 072601
ACDW2 = 000000	AVECT2 = 000000	DF115 = 072443	DF2 = 072443	DF261 = 072601
ACPUOP = 000000	BADCON = 060664	DF116 = 072443	DF20 = 072443	DF262 = 072601
ACO = %000000	BIT0 = 000001	DF117 = 072465	DF200 = 072503	DF263 = 072601
AC1 = %000001	BIT00 = 000001	DF12 = 072471	DF201 = 072503	DF264 = 072601
AC2 = %000002	BIT01 = 000002	DF120 = 072550	DF202 = 072503	DF265 = 072601
AC3 = %000003	BIT02 = 000004	DF121 = 072443	DF203 = 072503	DF266 = 072601
AC4 = %000004	BIT03 = 000010	DF122 = 072443	DF204 = 072503	DF267 = 072601
AC5 = %000005	BIT04 = 000020	DF123 = 072465	DF205 = 072503	DF27 = 072443
AC6 = %000006	BIT05 = 000040	DF124 = 072550	DF206 = 072503	DF270 = 072601
AC7 = %000007	BIT06 = 000100	DF125 = 072443	DF207 = 072503	DF271 = 072601
ADDW0 = 000000	BIT07 = 000200	DF126 = 072443	DF21 = 072471	DF272 = 072601
ADDW1 = 000000	BIT08 = 000400	DF127 = 072465	DF210 = 072503	DF273 = 072622
ADDW10 = 000000	BIT09 = 001000	DF13 = 072465	DF211 = 072503	DF274 = 072622
ADDW11 = 000000	BIT1 = 000002	DF130 = 072550	DF212 = 072503	DF275 = 072622
ADDW12 = 000000	BIT10 = 002000	DF131 = 072443	DF213 = 072503	DF276 = 072622
ADDW13 = 000000	BIT11 = 004000	DF132 = 072465	DF214 = 072503	DF277 = 072622
ADDW14 = 000000	BIT12 = 010000	DF133 = 072550	DF215 = 072443	DF3 = 072443
ADDW15 = 000000	BIT13 = 020000	DF134 = 072443	DF216 = 072471	DF30 = 072471
ADDW2 = 000000	BIT14 = 040000	DF135 = 072471	DF217 = 072443	DF300 = 072622
ADDW3 = 000000	BIT15 = 100000	DF136 = 072471	DF22 = 072471	DF301 = 072643
ADDW4 = 000000	BIT2 = 000004	DF137 = 072443	DF220 = 072443	DF302 = 072643
ADDW5 = 000000	BIT3 = 000010	DF14 = 072465	DF221 = 072443	DF303 = 072667
ADDW6 = 000000	BIT4 = 000020	DF140 = 072471	DF222 = 072471	DF304 = 072643
ADDW7 = 000000	BIT5 = 000040	DF141 = 072443	DF223 = 072443	DF305 = 072643
ADDW8 = 000000	BIT6 = 000100	DF142 = 072443	DF224 = 072443	DF306 = 072643
ADDW9 = 000000	BIT7 = 000200	DF143 = 072471	DF225 = 072565	DF307 = 072643
ADEVCT = 000000	BIT8 = 000400	DF144 = 072443	DF226 = 072565	DF31 = 072465
ADEVN = 000000	BIT9 = 001000	DF145 = 072443	DF227 = 072575	DF310 = 072643
AENV = 000000	BPTVEC = 000014	DF146 = 072471	DF23 = 072465	DF311 = 072643
AENVN = 000000	BYTABL = 060713	DF147 = 072443	DF230 = 072565	DF312 = 072643
AFATAL = 000000	CKSWR = 104406	DF15 = 072471	DF231 = 072565	DF313 = 072643
AMADR1 = 000000	CNT = 000445	DF150 = 072443	DF232 = 072575	DF314 = 072643
AMADR2 = 000000	CPSAVE = 046506	DF151 = 072471	DF233 = 072565	DF315 = 072643
AMADR3 = 000000	CPSPUR = 051774	DF152 = 072443	DF234 = 072565	DF316 = 072643
AMADR4 = 000000	CPTWO = 052022	DF153 = 072443	DF235 = 072575	DF317 = 072643
AMAMS1 = 000000	CR = 000015	DF154 = 072471	DF236 = 072565	DF32 = 072443
AMAMS2 = 000000	CRBUT = 052653	DF155 = 072443	DF237 = 072565	DF320 = 072643
AMAMS3 = 000000	CRLF = 000200	DF156 = 072443	DF24 = 072443	DF321 = 072643
AMAMS4 = 000000	DATA = 117760	DF157 = 072471	DF240 = 072575	DF322 = 072713
AMSGAD = 000000	DDISP = 177570	DF16 = 072443	DF241 = 072565	DF323 = 072713
AMSGLG = 000000	DF1 = 072434	DF160 = 072443	DF242 = 072565	DF324 = 072734
AMSGTY = 000000	DF10 = 072453	DF161 = 072443	DF243 = 072575	DF325 = 072713
AMTYP1 = 000000	DF100 = 072443	DF162 = 072471	DF244 = 072565	DF326 = 072713
AMTYP2 = 000000	DF101 = 072443	DF163 = 072443	DF245 = 072565	DF327 = 072713
AMTYP3 = 000000	DF102 = 072465	DF164 = 072443	DF246 = 072565	DF33 = 072471
AMTYP4 = 000000	DF103 = 072550	DF165 = 072503	DF247 = 072575	DF330 = 072713
APASS = 000000	DF104 = 072443	DF166 = 072503	DF25 = 072471	DF331 = 072713
APRIOR = 000000	DF105 = 072443	DF167 = 072503	DF250 = 072565	DF332 = 072713
APTC SU = 000040	DF106 = 072465	DF17 = 072465	DF251 = 072565	DF333 = 072713
APTENV = 000001	DF107 = 072465	DF170 = 072503	DF252 = 072565	DF334 = 072713
APTSIZ = 000200	DF11 = 072443	DF171 = 072503	DF253 = 072575	DF335 = 072713
APTSPO = 000100	DF110 = 072550	DF172 = 072503	DF254 = 072565	DF336 = 072713
ASWREG = 000000	DF111 = 072443	DF173 = 072524	DF255 = 072575	DF337 = 072713
ATESTN = 000000	DF112 = 072443	DF174 = 072524	DF256 = 072565	DF34 = 072465
AUNIT = 000000		DF175 = 072524	DF257 = 072565	DF340 = 072713

DF341 = 072713	DF423 = 072575	DH102 = 071460	DH164 = 071520	DH244 = 071404
DF342 = 072713	DF424 = 072575	DH103 = 071447	DH165 = 071563	DH245 = 071274
DF343 = 072713	DF425 = 072565	DH104 = 071404	DH166 = 071563	DH246 = 071520
DF344 = 072713	DF426 = 072565	DH105 = 071274	DH167 = 071563	DH247 = 071726
DF345 = 072713	DF427 = 072575	DH106 = 071460	DH17 = 071460	DH25 = 071447
DF346 = 072713	DF43 = 072503	DH107 = 071460	DH170 = 071563	DH250 = 071404
DF347 = 072755	DF430 = 072575	DH11 = 071404	DH171 = 071563	DH251 = 071274
DF35 = 072443	DF431 = 072575	DH110 = 071447	DH172 = 071563	DH252 = 071520
DF350 = 072755	DF432 = 072565	DH111 = 071404	DH173 = 071626	DH253 = 071726
DF351 = 072575	DF433 = 072565	DH112 = 071274	DH174 = 071626	DH254 = 071520
DF352 = 072755	DF434 = 072575	DH113 = 071460	DH175 = 071626	DH255 = 071726
DF353 = 072755	DF435 = 072575	DH114 = 071447	DH176 = 071274	DH256 = 071404
DF354 = 072755	DF436 = 072575	DH115 = 071404	DH177 = 071677	DH257 = 071274
DF355 = 072755	DF437 = 072565	DH116 = 071274	DH2 = 071274	DH26 = 071460
DF356 = 072565	DF44 = 072503	DH117 = 071460	DH2B = 071310	DH260 = 071563
DF357 = 072565	DF440 = 072565	DH12 = 071447	DH20 = 071404	DH261 = 071563
DF36 = 072471	DF441 = 073113	DH120 = 071447	DH200 = 071563	DH262 = 071563
DF360 = 072575	DF442 = 073113	DH121 = 071404	DH201 = 071563	DH263 = 071563
DF361 = 072776	DF443 = 073113	DH122 = 071274	DH202 = 071563	DH264 = 071563
DF362 = 073022	DF444 = 072443	DH123 = 071460	DH203 = 071563	DH265 = 071563
DF363 = 073030	DF45 = 072503	DH124 = 071447	DH204 = 071563	DH266 = 071563
DF364 = 072465	DF46 = 072503	DH125 = 071404	DH205 = 071563	DH267 = 071563
DF365 = 072465	DF47 = 072503	DH126 = 071274	DH206 = 071563	DH27 = 071404
DF366 = 073040	DF5 = 072453	DH127 = 071460	DH207 = 071563	DH270 = 071563
DF367 = 073062	DF50 = 072503	DH13 = 071460	DH21 = 071447	DH271 = 071563
DF37 = 072503	DF51 = 072503	DH130 = 071447	DH210 = 071563	DH272 = 071563
DF370 = 073072	DF52 = 072503	DH131 = 071274	DH211 = 071563	DH273 = 071563
DF371 = 073062	DF53 = 072503	DH132 = 071460	DH212 = 071563	DH274 = 071563
DF372 = 000000	DF54 = 072503	DH133 = 071447	DH213 = 071563	DH275 = 071563
DF373 = 000000	DF55 = 072503	DH134 = 071274	DH214 = 071563	DH276 = 071563
DF374 = 000000	DF56 = 072503	DH135 = 071447	DH215 = 071520	DH277 = 071563
DF375 = 000000	DF57 = 072503	DH136 = 071447	DH216 = 071447	DH3 = 071340
DF376 = 000000	DF6 = 072465	DH137 = 071274	DH217 = 071404	DH3B = 071354
DF377 = 000000	DF60 = 072503	DH14 = 071460	DH22 = 071447	DH30 = 071447
DF4 = 072443	DF61 = 072503	DH140 = 071447	DH220 = 071274	DH300 = 071563
DF40 = 072503	DF62 = 072443	DH141 = 071404	DH221 = 071520	DH301 = 071563
DF400 = 000000	DF63 = 072443	DH142 = 071274	DH222 = 071447	DH302 = 071563
DF401 = 072565	DF64 = 072453	DH143 = 071447	DH223 = 071404	DH303 = 071626
DF402 = 072565	DF65 = 072443	DH144 = 071404	DH224 = 071274	DH304 = 071563
DF403 = 072575	DF66 = 072443	DH145 = 071274	DH225 = 071404	DH305 = 071563
DF404 = 072575	DF67 = 072443	DH146 = 071447	DH226 = 071274	DH306 = 071563
DF405 = 072565	DF7 = 072443	DH147 = 071404	DH227 = 071726	DH307 = 071563
DF406 = 072565	DF70 = 072443	DH15 = 071447	DH227B = 071740	DH31 = 071460
DF407 = 072575	DF71 = 072550	DH150 = 071274	DH23 = 071460	DH310 = 071563
DF41 = 072524	DF72 = 072443	DH151 = 071447	DH230 = 071404	DH311 = 071563
DF410 = 072575	DF73 = 072550	DH152 = 071404	DH231 = 071274	DH312 = 071563
DF411 = 072565	DF74 = 072443	DH153 = 071274	DH232 = 071726	DH313 = 071563
DF412 = 072565	DF75 = 072443	DH154 = 071447	DH233 = 071404	DH314 = 071563
DF413 = 072575	DF76 = 072465	DH155 = 071404	DH234 = 071274	DH315 = 071563
DF414 = 072575	DF77 = 072550	DH156 = 071274	DH235 = 071726	DH316 = 071563
DF415 = 072565	DH1 = 071204	DH157 = 071447	DH236 = 071404	DH317 = 071563
DF416 = 072565	DH1B = 071220	DH16 = 071520	DH237 = 071274	DH32 = 071404
DF417 = 072575	DH1C = 071252	DH16B = 071534	DH24 = 071404	DH320 = 071563
DF42 = 072503	DH1D = 071261	DH160 = 071404	DH240 = 071726	DH321 = 071563
DF420 = 072575	DH10 = 071447	DH161 = 071274	DH241 = 071404	DH322 = 071563
DF421 = 072565	DH100 = 071404	DH162 = 071447	DH242 = 071274	DH323 = 071563
DF422 = 072565	DH101 = 071274	DH163 = 071274	DH243 = 071726	DH324 = 071626

DH325 = 071563	DH40 = 071563	DH60 = 071563	DT137 = 073270	DT220 = 073206
DH326 = 071563	DH400 = 000000	DH61 = 071563	DT14 = 073256	DT221 = 073206
DH327 = 071563	DH401 = 071404	DH62 = 071274	DT140 = 073230	DT222 = 073230
DH33 = 071447	DH402 = 071274	DH63 = 071340	DT141 = 073206	DT223 = 073206
DH330 = 071563	DH403 = 071460	DH64 = 071447	DT142 = 073206	DT224 = 073206
DH331 = 071563	DH404 = 071726	DH65 = 071274	DT143 = 073230	DT225 = 073206
DH332 = 071563	DH405 = 071404	DH66 = 071404	DT144 = 073206	DT226 = 073206
DH333 = 071563	DH406 = 071274	DH67 = 071274	DT145 = 073206	DT227 = 073256
DH334 = 071563	DH407 = 071460	DH7 = 071404	DT146 = 073230	DT23 = 073256
DH335 = 071563	DH41 = 071626	DH70 = 071340	DT147 = 073206	DT230 = 073206
DH336 = 071563	DH41B = 071642	DH71 = 071447	DT15 = 073230	DT231 = 073206
DH337 = 071563	DH410 = 071726	DH72 = 071274	DT150 = 073206	DT232 = 073256
DH34 = 071460	DH411 = 071404	DH73 = 071447	DT151 = 073230	DT233 = 073206
DH340 = 071563	DH412 = 071274	DH74 = 071404	DT152 = 073206	DT234 = 073206
DH341 = 071563	DH413 = 071460	DH75 = 071274	DT153 = 073206	DT235 = 073256
DH342 = 071563	DH414 = 071726	DH76 = 071460	DT154 = 073230	DT236 = 073206
DH343 = 071563	DH415 = 071404	DH77 = 071447	DT155 = 073206	DT237 = 073206
DH344 = 071563	DH416 = 071274	DIDONE 045416	DT156 = 073206	DT24 = 073270
DH345 = 071563	DH417 = 071460	DISPLA 001142	DT157 = 073230	DT240 = 073256
DH346 = 071563	DH42 = 071563	DISPRE 000174	DT16 = 073270	DT241 = 073206
DH347 = 071563	DH420 = 071726	DSWR = 177570	DT160 = 073206	DT242 = 073206
DH35 = 071404	DH421 = 071404		DT161 = 073206	DT243 = 073256
DH350 = 071563	DH422 = 071274		DT162 = 073230	DT244 = 073206
DH351 = 071460	DH423 = 071460		DT163 = 073206	DT245 = 073206
DH352 = 071563	DH424 = 071726		DT164 = 073206	DT246 = 073206
DH353 = 071563	DH425 = 071404		DT165 = 073312	DT247 = 073256
DH354 = 071563	DH426 = 071274		DT166 = 073312	DT25 = 073230
DH355 = 071563	DH427 = 071460		DT167 = 073312	DT250 = 073206
DH356 = 071404	DH43 = 071563		DT17 = 073256	DT251 = 073206
DH357 = 071746	DH430 = 071726		DT170 = 073312	DT252 = 073206
DH357B = 071762	DH431 = 071460		DT171 = 073312	DT253 = 073256
DH36 = 071447	DH432 = 071404		DT172 = 073312	DT254 = 073206
DH360 = 071460	DH433 = 071274		DT173 = 073356	DT255 = 073256
DH361 = 071274	DH434 = 071460		DT174 = 073356	DT256 = 073206
DH362 = 072012	DH435 = 071726		DT175 = 073356	DT257 = 073206
DH362B = 072026	DH436 = 071460		DT176 = 073206	DT26 = 073256
DH363 = 072034	DH437 = 071404		DT177 = 073422	DT260 = 073312
DH363B = 072050	DH44 = 071563		DT2 = 073142	DT261 = 073312
DH364 = 072074	DH440 = 071404		DT20 = 073270	DT262 = 073312
DH364B = 072106	DH441 = 072412		DT200 = 073312	DT263 = 073312
DH365 = 072143	DH441B = 072426		DT201 = 073312	DT264 = 073312
DH366 = 072154	DH442 = 072143		DT202 = 073312	DT265 = 073312
DH366B = 072170	DH443 = 072143		DT203 = 073312	DT266 = 073312
DH367 = 072206	DH444 = 071340		DT204 = 073312	DT267 = 073312
DH367B = 072222	DH45 = 071563		DT205 = 073312	DT27 = 073270
DH37 = 071563	DH46 = 071563		DT206 = 073312	DT270 = 073312
DH37B = 071576	DH47 = 071563		DT207 = 073312	DT271 = 073312
DH370 = 072253	DH5 = 071447		DT21 = 073230	DT272 = 073312
DH371 = 072316	DH50 = 071563		DT210 = 073312	DT273 = 073312
DH372 = 000000	DH51 = 071563		DT211 = 073312	DT274 = 073312
DH373 = 000000	DH52 = 071563		DT212 = 073312	DT275 = 073312
DH374 = 000000	DH53 = 071626		DT213 = 073312	DT276 = 073312
DH375 = 000000	DH54 = 071563		DT214 = 073312	DT277 = 073312
DH376 = 000000	DH55 = 071563		DT215 = 073206	DT3 = 073164
DH377 = 000000	DH56 = 071563		DT216 = 073230	DT30 = 073230
DH4 = 071404	DH57 = 071563		DT217 = 073206	DT300 = 073312
DH4B = 071420	DH6 = 071447		DT22 = 073230	DT301 = 073522

DT302 = 073522	DT365 = 073256	DT47 = 073312	EM120 = 056053	EM172 = 057252
DT303 = 073574	DT366 = 074030	DT5 = 073230	EM121 = 056066	EM173 = 057266
DT304 = 073522	DT367 = 074076	DT50 = 073312	EM122 = 056102	EM174 = 057302
DT305 = 073522	DT37 = 073312	DT51 = 073312	EM123 = 056116	EM175 = 057316
DT306 = 073522	DT370 = 074122	DT52 = 073312	EM123B = 056132	EM176 = 057332
DT307 = 073522	DT371 = 074166	DT53 = 073356	EM124 = 056142	EM176B = 057346
DT31 = 073256	DT372 = 000000	DT54 = 073312	EM125 = 056156	EM177 = 057354
DT310 = 073522	DT373 = 000000	DT55 = 073312	EM126 = 056172	EM2 = 052422
DT311 = 073522	DT374 = 000000	DT56 = 073312	EM127 = 056206	EM2B = 052436
DT312 = 073522	DT375 = 000000	DT57 = 073312	EM127B = 056222	EM20 = 053166
DT313 = 073522	DT376 = 000000	DT6 = 073256	EM13 = 053036	EM200 = 057410
DT314 = 073522	DT377 = 000000	DT60 = 073312	EM13A = 053050	EM200C = 057426
DT315 = 073522	DT4 = 073206	DT61 = 073312	EM13B = 053060	EM201 = 057463
DT316 = 073522	DT40 = 073312	DT62 = 073270	EM130 = 056232	EM202 = 057534
DT317 = 073522	DT400 = 000000	DT63 = 073270	EM131 = 056246	EM203 = 057606
DT32 = 073270	DT401 = 073206	DT64 = 073230	EM132 = 056262	EM204 = 057730
DT320 = 073522	DT402 = 073206	DT65 = 073270	EM132B = 056276	EM204C = 057746
DT321 = 073522	DT403 = 073256	DT66 = 073206	EM133 = 056307	EM205 = 060003
DT322 = 073646	DT404 = 073256	DT67 = 073206	EM134 = 056322	EM206 = 060054
DT323 = 073646	DT405 = 073206	DT7 = 073206	EM135 = 056336	EM207 = 060126
DT324 = 073712	DT406 = 073206	DT70 = 073206	EM136 = 056406	EM21 = 053204
DT325 = 073646	DT407 = 073256	DT71 = 073432	EM137 = 056422	EM210 = 060200
DT326 = 073646	DT41 = 073356	DT72 = 073206	EM14 = 053036	EM211 = 060252
DT327 = 073646	DT410 = 073256	DT73 = 073466	EM140 = 056436	EM212 = 060331
DT33 = 073230	DT411 = 073206	DT74 = 073206	EM141 = 056452	EM213 = 060402
DT330 = 073646	DT412 = 073206	DT75 = 073142	EM141C = 056472	EM213C = 060460
DT331 = 073646	DT413 = 073256	DT76 = 073256	EM142 = 056510	EM213D = 060470
DT332 = 073646	DT414 = 073256	DT77 = 073466	EM143 = 056524	EM214 = 060513
DT333 = 073646	DT415 = 073206	EMTVEC = 000030	EM144 = 056540	EM214B = 060570
DT334 = 073646	DT416 = 073206	EM1 = 052346	EM145 = 056556	EM215 = 060600
DT335 = 073646	DT417 = 073256	EM1B = 052362	EM146 = 056572	EM216 = 060735
DT336 = 073646	DT42 = 073312	EM1C = 052373	EM147 = 056606	EM217 = 060750
DT337 = 073646	DT420 = 073256	EM1D = 052412	EM15 = 053104	EM22 = 053204
DT34 = 073256	DT421 = 073206	EM10 = 052761	EM150 = 056626	EM220 = 060770
DT340 = 073646	DT422 = 073206	EM100 = 055516	EM151 = 056642	EM221 = 061004
DT341 = 073646	DT423 = 073256	EM101 = 055532	EM152 = 056656	EM222 = 061054
DT342 = 073646	DT424 = 073256	EM102 = 055546	EM153 = 056676	EM223 = 061070
DT343 = 073646	DT425 = 073206	EM102B = 055562	EM154 = 056712	EM224 = 061110
DT344 = 073646	DT426 = 073206	EM103 = 055571	EM155 = 056726	EM225 = 061124
DT345 = 073646	DT427 = 073256	EM104 = 055604	EM156 = 056746	EM225B = 061136
DT346 = 073646	DT43 = 073312	EM105 = 055620	EM157 = 056762	EM226 = 061147
DT347 = 073646	DT430 = 073256	EM106 = 055634	EM157B = 056776	EM227 = 061160
DT35 = 073270	DT431 = 073256	EM106B = 055650	EM16 = 053116	EM227B = 061172
DT350 = 073646	DT432 = 073206	EM107 = 055660	EM160 = 057005	EM23 = 053216
DT351 = 073256	DT433 = 073206	EM11 = 052774	EM160B = 057026	EM23B = 053230
DT352 = 073646	DT434 = 073256	EM11B = 053012	EM161 = 057053	EM230 = 061211
DT353 = 073646	DT435 = 073256	EM110 = 055674	EM162 = 057066	EM230B = 061222
DT354 = 073646	DT436 = 073256	EM111 = 055710	EM162B = 057102	EM231 = 061234
DT355 = 073646	DT437 = 073206	EM112 = 055724	EM163 = 057112	EM232 = 061246
DT356 = 073142	DT44 = 073312	EM113 = 055740	EM164 = 057126	EM233 = 061260
DT357 = 073164	DT440 = 073206	EM113B = 055754	EM165 = 057156	EM233B = 061272
DT36 = 073230	DT441 = 074210	EM114 = 055764	EM166 = 057172	EM234 = 061304
DT360 = 073256	DT442 = 074226	EM115 = 056000	EM167 = 057206	EM235 = 061316
DT361 = 073522	DT443 = 074226	EM116 = 056014	EM17 = 053142	EM236 = 061330
DT362 = 073756	DT444 = 073206	EM117 = 056030	EM17B = 053154	EM236B = 061342
DT363 = 073774	DT45 = 073312	EM117B = 056044	EM170 = 057222	EM237 = 061355
DT364 = 074016	DT46 = 073312	EM12 = 053024	EM171 = 057236	EM24 = 053243

EM240	061366	EM38	052466	EM346	065774	EM415B	070276	EM6C	052610
EM241	061400	EM30	053334	EM346C	066020	EM416	070311	EM60	= 054235
EM241B	061412	EM300	063065	EM347	066073	EM417	070322	EM60B	054632
EM242	061425	EM300C	063110	EM347B	066104	EM417B	070334	EM61	= 054274
EM243	061436	EM301	063162	EM35	053447	EM42	= 053476	EM61B	054712
EM244	061450	EM301B	063174	EM350	066117	EM42B	053616	EM62	054771
EM244B	061462	EM302	063207	EM351	066130	EM420	070366	EM62A	055010
EM245	061474	EM303	063220	EM352	066232	EM421	070400	EM62B	055020
EM246	061506	EM304	063232	EM353	066302	EM421B	070412	EM63	055067
EM246B	061544	EM304C	063246	EM354	066352	EM422	070425	EM64	055104
EM247	061557	EM305	063312	EM355	066422	EM423	070436	EM65	055154
EM25	053260	EM305C	063326	EM356	066472	EM424	070450	EM66	055200
EM250	061570	EM306	063406	EM356B	066526	EM425	070462	EM66B	055220
EM250B	061602	EM307	063514	EM357	066576	EM425B	070474	EM67	055226
EM251	061615	EM31	053346	EM36	053464	EM426	070506	EM7	052737
EM252	= 061506	EM31B	053360	EM360	066632	EM427	070520	EM7B	052756
EM252B	061626	EM310	063564	EM361	066722	EM43	= 053556	EM70	055310
EM253	061642	EM311	063640	EM362	066740	EM43B	053652	EM71	055333
EM254	= 061506	EM312	063710	EM363	067050	EM430	070572	EM72	055350
EM254B	061654	EM313	063760	EM364	067116	EM431	070604	EM73	055364
EM255	061672	EM314	064030	EM365	067216	EM432	070632	EM74	055400
EM256	061716	EM315	064100	EM366	067301	EM432B	070644	EM75	055414
EM256B	061730	EM316	064150	EM367	067364	EM433	070657	EM76	055430
EM257	061744	EM317	064220	EM37	053476	EM434	070670	EM76A	055444
EM26	053272	EM32	053372	EM370	067447	EM435	070740	EM76B	055453
EM26B	053304	EM320	064270	EM371	067503	EM436	070752	EM77	055503
EM260	061756	EM321	064340	EM372	= 000000	EM437	071000	ENDTES	045376
EM260B	061770	EM322	064410	EM373	= 000000	EM44	= 053476	ENDTST	041776
EM261	062014	EM322B	064422	EM374	= 000000	EM44B	053731	EQASCI	047172
EM262	062026	EM323	064446	EM375	= 000000	EM440	071022	EPENDS	046164
EM262B	062101	EM324	064460	EM376	= 000000	EM441	071044	ERM10	047074
EM263	062114	EM325	064472	EM377	= 000000	EM441B	071100	ERRNUM	051576
EM263B	062126	EM325B	064506	EM4	052502	EM442	= 071044	ERROR	= 104000
EM264	062141	EM325C	064521	EM4A	052520	EM442B	071147	ERRVEC	= 000004
EM264C	062154	EM326	= 064472	EM4B	052531	EM443	= 071044	ERTYPC	051152
EM265	062226	EM327	064616	EM4C	052543	EM444	071171	ERT1	051346
EM265C	062242	EM33	053410	EM4D	052552	EM45	= 053476	ERT2	051546
EM266	062277	EM330	064666	EM40	053556	EM45B	054006	ERT3	051552
EM267	062353	EM331	064742	EM400	= 000000	EM46	= 053556	ERT4	051560
EM267C	062366	EM332	065012	EM401	067633	EM46B	054041	ERT5	051572
EM27	053317	EM332B	065026	EM401B	067644	EM47	= 053476	EXPCTD	045374
EM270	062440	EM332C	065041	EM402	067655	EM47B	054114	FALTRP	044006
EM270C	062454	EM333	= 064446	EM403	067666	EM5	052563	FPSPUR	051732
EM271	062472	EM334	065110	EM403B	067746	EM50	= 053476	FPVECT	= 000244
EM271C	062506	EM334C	065124	EM403C	070005	EM50B	054156	GTSWR	= 104405
EM272	062531	EM335	065173	EM404	070016	EM51	054235	HT	= 000011
EM272C	062544	EM336	065242	EM405	070030	EM52	054274	IBKOUT	054462
EM273	062601	EM336C	065260	EM406	070044	EM53	= 054274	IBSAVE	046512
EM273B	062612	EM337	065277	EM407	070060	EM54	= 054235	ICOUT	065346
EM274	062636	EM34	053422	EM41	= 053556	EM54B	054336	IDONE	042010
EM275	062650	EM34B	053434	EM410	070130	EM55	= 054235	IENBT	054230
EM275B	062720	EM340	065353	EM411	070144	EM55B	054402	IEZBT	053724
EM276	062733	EM341	065422	EM411B	070156	EM56	= 054274	IFD	052650
EM276C	062746	EM342	065472	EM412	070170	EM56B	054473	IFDST	055062
EM277	063020	EM343	065542	EM413	070202	EM57	= 054274	IFIC	064736
EM277C	063034	EM344	065617	EM414	070252	EM57B	054546	IFIU	063634
EM3	052452	EM345	065724	EM415	070264	EM6	052574	IFIUV	060506

IFIV	054626	N1	052721	SW03	= 000010	TST22	015110	SAPTHD	006102
IFL	062076	N10	071165	SW04	= 000020	TST23	015404	SATYC	050014
IFLAG	054764	N2	052723	SW05	= 000040	TST24	015700	SATY1	047770
IGR7FL	067736	N244	071142	SW06	= 000100	TST25	016204	SATY3	047776
IMBIT	065612	N3	052725	SW07	= 000200	TST26	016510	SATY4	050006
INSTOF	052702	N3132	060316	SW08	= 000400	TST27	017010	SAUTOB	001134
IOP18	057723	N4	052727	SW09	= 001000	TST3	007374	SBASE	001372
IOTRAP=	000020	N5	052731	SW1	= 000002	TST30	017316	SBDADR	001122
IOTVEC=	000020	N6	052733	SW10	= 002000	TST31	017564	SBDAT	001126
IXMBT	062346	N7	052735	SW11	= 004000	TST32	020042	SBELL	001306
KDPAR0=	172360	N746T2	060624	SW12	= 010000	TST33	020250	SCDW1	001376
KDPAR1=	172362	N747T2	061030	SW13	= 020000	TST34	020464	SCDW2	001400
KDPAR2=	172364	OR	057716	SW14	= 040000	TST35	020700	SCHARC	047530
KDPAR3=	172366	PASCIZ	051600	SW15	= 100000	TST36	021126	SCKSWR	050236
KDPAR4=	172370	PCBAD	060647	SW2	= 000004	TST37	021352	SCLR.T	046074
KDPAR7=	172376	PERIOD	060711	SW3	= 000010	TST4	007766	SCMTAG	001100
KDPDR0=	172320	PIRQ =	177772	SW4	= 000020	TST40	021600	SCM1 =	000024
KDPDR1=	172322	PIRQVE=	000240	SW5	= 000040	TST41	022014	SCM2 =	000050
KDPDR2=	172324	POWERM	052120	SW6	= 000100	TST42	022246	SCM3 =	000024
KDPDR3=	172326	PROGNU=	000003	SW7	= 000200	TST43	022514	SCM4 =	000024
KDPDR4=	172330	PRST	052662	SW8	= 000400	TST44	022772	SCNTLG	050655
KDPDR7=	172336	PRO =	000000	SW9	= 001000	TST45	024320	SCNTLU	050650
KIPAR0=	172340	PR1 =	000040	TAB	= 000011	TST46	024502	SCPUOP	001344
KIPAR1=	172342	PR2 =	000100	TBITVE=	000014	TST47	024664	SCRLF	001313
KIPAR2=	172344	PR3 =	000140	TKVEC =	000060	TST5	010314	SDDW0	001402
KIPAR3=	172346	PR4 =	000200	TPVEC =	000064	TST50	025064	SDDW1	001404
KIPAR4=	172350	PR5 =	000240	TRAPTO	071112	TST51	025300	SDDW10	001426
KIPAR7=	172356	PR6 =	000300	TRAPV	041714	TST52	025506	SDDW11	001430
KIPDR0=	172300	PR7 =	000340	TRAPVE=	000034	TST53	025730	SDDW12	001432
KIPDR1=	172302	PS =	177776	TRPV	044054	TST54	026166	SDDW13	001434
KIPDR2=	172304	PSW =	177776	TRTVEC=	000014	TST55	026306	SDDW14	001436
KIPDR3=	172306	PWRVEC=	000024	TST1	006764	TST56	026450	SDDW15	001440
KIPDR4=	172310	RDCMR =	104407	TST10	011360	TST57	030142	SDDW2	001406
KIPDR7=	172316	RESREG=	104411	TST100	041302	TST6	010630	SDDW3	001410
LABEL1	042674	RESVEC=	000010	TST101	041430	TST60	031230	SDDW4	001412
LF =	000012	RETURN	045166	TST102	041446	TST61	033176	SDDW5	001414
LOOP	006764	RSETUP=	104412	TST103	041524	TST62	033434	SDDW6	001416
MFR0 =	177572	RTI	045210	TST104	041602	TST63	033672	SDDW7	001420
MFR2 =	177576	R6 =	X000006	TST105	042446	TST64	034130	SDDW8	001422
MFR3 =	172516	R7 =	X000007	TST106	042542	TST65	034376	SDDW9	001424
MVTECT=	000250	SAVIOT	042426	TST107	042650	TST66	034646	SDEVCT	001326
MNUMBE=	000444	SAVREG=	104410	TST11	011620	TST67	035124	SDEVN	001374
MODE1	042556	SCOPE =	000004	TST110	042764	TST7	011104	SDISAB	046024
MS1	052171	SETERL	042430	TST111	043164	TST70	035414	SDOAGN	046114
MS10 =	052231	SPACE	052164	TST112	043300	TST71	035524	SENDAD	046104
MS11	052271	SR1 =	177574	TST113	043412	TST72	035634	SENDCT	045464
MS2	052207	STACK =	001100	TST114	043606	TST73	037402	SENULL	046160
MS20	052314	START	006116	TST115	045420	TST74	037456	SENV	001336
MS21	052326	STCSUB	037110	TST12	012076	TST75	040416	SENVN	001337
MS3	052231	STKLMT=	177774	TST13	012756	TST76	041104	SEOP	045420
MS4	052247	STORE	045232	TST14	013636	TST77	041160	SEOPCT	045452
MULTER	045104	SWR	001140	TST15	013742	TYPE =	104401	SERFLG	001103
NEGDAR	061041	SWREG	000176	TST16	014160	TYPOC =	104402	SERMAX	001115
NEGDNR	060635	SWO =	000001	TST17	014264	TYPON =	104404	SERROR	046514
NODAT	045216	SW00 =	000001	TST2	007130	TYPOS =	104403	SERRPC	001116
NO	052717	SW01 =	000002	TST20	014372	WENTTO	052670	SERRTB	001442
NO4	071162	SW02 =	000004	TST21	014614	WCDON	037400	SERTIL	001112

SYMBOL TABLE

\$ESCAP	001304	\$MAMS3	001356	\$REG0	001162	\$SWR =	177400	\$TMP7	001250
\$ETABL	001336	\$MAMS4	001362	\$REG1	001164	\$SWREG	001340	\$TN =	000115
\$ETEND	001442	\$MBADR	006104	\$REG10	001202	\$SWRMK=	000000	\$TPB	001152
\$FATAL	001320	\$MFLG	050232	\$REG11	001204	\$SWRMS=	000200	\$TPFLG	001157
\$FFLG	050234	\$MNEW	050673	\$REG12	001206	\$TAB	052167	\$TPS	001150
\$FILLC	001156	\$MSGAD	001332	\$REG13	001210	\$TBIT	046156	\$TRAP	050704
\$FILLS	001155	\$MSGLG	001334	\$REG14	001212	\$TERM =	000026	\$TRAP2	050726
\$GDADR	001120	\$MSGTY	001316	\$REG15	001214	\$TESTN	001322	\$TRP =	000013
\$GDDAT	001124	\$MSWR	050662	\$REG16	001216	\$TIMES	001302	\$TRPAD	050740
\$GET42	045744	\$MTYP1	001347	\$REG17	001220	\$TKB	001146	\$TSTM	006106
\$GTSWR	050306	\$MTYP2	001353	\$REG2	001166	\$TKS	001144	\$TSTNM	001102
\$GT42C	046056	\$MTYP3	001357	\$REG20	001222	\$TMP0	001232	\$TYPE	047174
\$HD =	000003	\$MTYP4	001363	\$REG21	001224	\$TMP1	001234	\$TYPEC	047412
\$HIBTS	006102	\$MXCNT	046510	\$REG22	001226	\$TMP10	001252	\$TYPEX	047532
\$ICNT	001104	\$NULL	001154	\$REG23	001230	\$TMP11	001254	\$TYPOC	047560
\$ILLUP	051144	\$NWTST=	000001	\$REG3	001170	\$TMP12	001256	\$TYPON	047574
\$INTAG	001135	\$OCNT	047764	\$REG4	001172	\$TMP13	001260	\$TYPOS	047534
\$ITEMB	001114	\$OMODE	047766	\$REG5	001174	\$TMP14	001262	\$UNIT	001330
\$LF	001314	\$OVER	046472	\$REG6	001176	\$TMP15	001264	\$UNITM	006112
\$LFLG	050233	\$PASS	001324	\$REG7	001200	\$TMP16	001266	\$USWR	001342
\$LOOP	046152	\$PASS2	046166	\$RESRE	047134	\$TMP17	001270	\$VECT1	001366
\$LPADR	001106	\$PASTM	006110	\$RTNAD	046154	\$TMP2	001236	\$VECT2	001370
\$LPERR	001110	\$PWRAD	051126	\$RTRN	046150	\$TMP20	001272	\$XOFF =	000023
\$MADR1	001350	\$PWRDN	050766	\$SAVRE	047076	\$TMP21	001274	\$XON =	000021
\$MADR2	001354	\$PWRMG	051122	\$SAVR6	051150	\$TMP22	001276	\$XTSTR	046202
\$MADR3	001360	\$PWRUP	051040	\$SCOPE	046170	\$TMP23	001300	\$SET4=	000001
\$MADR4	001364	\$QUES	001312	\$SETUP=	000137	\$TMP3	001240	\$OFILL	047765
\$MAIL	001316	\$RDCHR	050520	\$STUP =	177777	\$TMP4	001242	.RSET	052050
\$MAMS1	001346	\$RDSZ =	000001	\$SVLAD	046436	\$TMP5	001244	.SX =	006102
\$MAMS2	001352	\$REGAD	001160	\$SVPC =	006102	\$TMP6	001246		

. ABS. 074240 000
000000 001

ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 60768 WORDS (238 PAGES)
DYNAMIC MEMORY: 20034 WORDS (77 PAGES)
ELAPSED TIME: 00:17:12
CKFPCD.BIN,CKFPCD/CR/-SP/NL:TOC=CKFPCD.MLB/ML,CKFPCD.P11