

FP11-F

FP11F FLTG PNT PRT B
CKFPBBO

AH-F635B-MC
FICHE 1 OF 2

AUG 1981
COPYRIGHT © 79-81
MADE IN USA



The main body of the document is a large, dense grid of data. Each cell in the grid contains a small, complex diagram or schematic, likely representing a technical drawing or a specific data point. The diagrams are arranged in a regular, repeating pattern across the page. The text within these diagrams is too small to be legible, but they appear to be organized into columns and rows, possibly representing a sequence of operations or a set of related data points.

FP11-F

FP11F FLTG PNT PRT B
CKFPBBO

AH-F635B-MC
FICHE 2 OF 2

AUG 1981
COPYRIGHT © 79-81
MADE IN USA



.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
40
41
42
43
44
45
46

IDENTIFICATION

PRODUCT CODE: AC-F633B-MC
PRODUCT NAME: CKFPBB0 FP11F FLTG PNT PRT B
DATE CREATED: APRIL, 1981
MAINTAINER: DIAGNOSTIC ENGINEERING
AUTHOR: ANTHONY VEZZA, DAN MILLEVILLE

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

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

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

COPYRIGHT (C) 1979, 1981 BY DIGITAL EQUIPMENT CORPORATION

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

HISTORY

NO CHANGES TO THE 11/34 FLOATING POINT DIAGNOSTICS 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 - CHECKS THAT SR1 MATCHES WHAT ACTUALLY HAPPENED TO THE REGISTER OF THE INSTRUCTION, AND THAT THE VALUE OF AUTO INCREMENT/DECREMENT WAS PROPER.

ALL THREE PARTS WERE RE-RELEASED WITH NEW SCOPE AND ERROR ROUTINES THAT CHECK 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. 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.

CONTENTS

87	
88	
89	
90	1. ABSTRACT
91	
92	2. REQUIREMENTS
93	2.1 EQUIPMENT
94	2.2 STORAGE
95	2.3 PRELIMINARY PROGRAMS
96	
97	3. LOADING PROCEDURE
98	
99	4. STARTING PROCEDURE
100	4.1 CONTROL SWITCH SETTINGS
101	4.2 STARTING ADDRESS
102	4.3 PROGRAM AND OPERATOR INTERACTION
103	
104	5. OPERATING PROCEDURE
105	5.1 OPERATIONAL SWITCH SETTINGS
106	5.3 OPERATOR ACTION
107	
108	6. ERRORS
109	6.1 SUMMARY
110	6.2 ERROR RECOVERY
111	
112	7. RESTRICTIONS
113	7.1 STARTING RESTRICTIONS
114	7.2 OPERATING RESTRICTIONS
115	
116	8. MISCELLANEOUS
117	8.1 EXECUTION TIMES
118	8.2 STACK POINTER
119	8.3 PASS COUNT
120	8.4 T-BIT TRAPPING
121	8.5 SOFTWARE SWITCH REGISTER
122	8.6 INTERRUPTS TEST
123	8.7 ACT, APT AND XXDP COMPATIBILITY
124	
125	9. PROGRAM DESCRIPTION
126	9.1 CKFPBBO
127	
128	10. LISTING
129	10.1 CKFPBBO
130	
131	
132	
133	
134	
135	
136	

138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194

1.

ABSTRACT

THE THREE PROGRAMS:

CKFPACO CKFPBBO CKFPCCO

ARE DESIGN TO DETECT AND REPORT LOGIC FAULTS IN THE PDP 11/44 FP11-F FLOATING POINT PROCESSOR. THE DESIGN IS AN ATTEMPT TO REACH ALL ROM STATES, TAKE ALL BRANCH MICRO TESTS (BUT'S) AND VERIFY ALL THE LOGIC. THEY CONSIST OF 157 (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 CKFPBBO)
CMPD AND CMPF
DIVD AND DIVF
MULD AND MULF
MODD AND MODF

C. CKFPCCO

CKFPCCO TESTS:

STF AND STD (ALL MODES)
STCFD AND STCDF
CLRD AND CLRF

195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251

NEGF AND NEGD
ABSF AND ABSD
TSTF AND TSTD
NEGF, ABSF AND TSTF (ALL SOURCE MODES)
NEGF, ABSF AND TSTF (ALL SOURCE MODES)
LDFPS (ALL SOURCE MODES)
LDCIF AND LDCLF
LDCID AND LDCLD
LDEXP
STFPS (ALL DESTINATION MODES)
STCFL AND STCFI
STCDL AND STCDI
STEXP
STST

2. REQUIREMENTS

2.1 EQUIPMENT

A PDP 11/44 (WITH OR WITHOUT CONSOLE), LA30 (OR EQUIVALENT) 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 CKFPB80 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 11/44 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

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

1. LOAD PROGRAM INTO MEMORY
2. LOAD ADDRESS 200
3. SET CONSOLE SWITCHES (IF CONSOLE IS PRESENT)
4. PRESS START
ON FIRST PASS THE PROGRAM WILL IDENTIFY ITSELF. NOTE THAT IF THERE IS NO PHYSICAL CONSOLE THE PROGRAM WILL REQUEST THE OPERATOR FOR INITIAL VALUE FOR THE SOFTWARE SWITCH REGISTER (SEE SECTION 8.5). IF RUNNING UNDER ACT, APT OR CHAIN THIS DOES NOT APPLY.
5. THE PROGRAM WILL LOOP AND AN END OF PASS AND ERROR SUMMARY WILL BE TYPED AT THE END OF EVERY PASS.

5. OPERATING PROCEDURE

5.1 OPERATIONAL SWITCH SETTINGS

THE SWITCH SETTING ARE:

	OCTAL	
SW<15>=1....	100000	HALT ON ERROR
SW<14>=1....	40000	LOOP ON CURRENT TEST
SW<13>=1....	20000	INHIBIT ERROR TYPE OUTS
SW<12>=1....	10000	INHIBIT T-BIT TRAPPING
SW<11>=1....	4000	INHIBIT ITERATIONS
SW<10>=1....	2000	RING TTY BELL ON ERROR
SW<9>=1....	1000	LOOP ON ERROR
SW<8>=1....	400	LOOP ON TEST SPECIFIED IN SW<6> THROUGH SW<0>
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 CKFPBB0. IF APT IS SELECTING THE TEST, THE SWITCH REGISTER IS EXAMINED TO SEE IF THE USER HAS SELECTED THIS TEST BY A <1> IN SW<7>

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 IF SW<13>=1 THIS

309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365

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.

6.2 ERROR RECOVERY

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

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

7. RESTRICTIONS

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

EACH OF THE THREE PROGRAMS WILL RUN WITH OR WITHOUT A CONSOLE SWITCH REGISTER. IF A PHYSICAL CONSOLE SWITCH REGISTER IS PRESENT ON THE SYSTEM, THEN THESE PROGRAMS WILL GO AHEAD AND USE IT FOR THE SWITCH

366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422

FUNCTIONS DESCRIBED IN 5.1 ABOVE. IF HOWEVER THERE IS NO CONSOLE SWITCH REGISTER ON THE SYSTEM A SOFTWARE SWITCH REGISTER WILL BE USED. THIS SOFTWARE SWITCH REGISTER CAN BE EXAMINED OR MODIFIED AT ANY TIME BY THE USER IF HE TYPES CONTROL G WHILE THE PROGRAM IS RUNNING. THIS CONTROL G WILL CAUSE THE CONTENTS OF THE SOFTWARE SWITCH REGISTER TO BE TYPED ON THE TTY AND ASK THE USER FOR A NEW VALUE. WHEN THE USER TYPES A VALUE AND CARRIAGE RETURN THEN THE PROGRAM WILL RESUME TESTING AT THE SAME POINT AT WHICH IT LEFT OFF WHEN THE USER TYPED CONTROL G. NOTE THAT WHEN NOT RUNNING UNDER ACT, APT OR CHAIN THE USER WILL BE ASKED FOR A SOFTWARE SWITCH REGISTER VALUE AFTER LOADING ADDRESS 200 AND STARTING THE PROGRAM THE FIRST TIME THE PROGRAM IS RUN AFTER LOADING (ONLY IF NO CONSOLE SWITCH REGISTER IS ON THE SYSTEM).

8.6 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>=0 THIS TEST WILL BE RUN. IF SW<7>=1 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 ROUND\TRUNK TEST

423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479

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

TEST 2 OVER\UNDER TEST

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

TEST 3 LDCFD AND LDCDF TEST

THIS IS A TEST OF LDCFD AND LDCDF.

TEST 4 CMPD TEST

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

TEST 5 DIVD WITH (FSRC=0) AND (BUT FD) TEST

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

TEST 6 DIVF TEST

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

TEST 7 DIVD TEST

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

TEST 10 MULF TEST

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

480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536

TEST 11 MULD TEST

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

TEST 12 UNDER\OVER FLOW, USING MULF WITH TRAPS DISABLED, TEST

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

TEST 13 UNDER\OVER FLOW, USING MULD WITH TRAPS DISABLED, TEST

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

TEST 14 UNDER\OVER FLOW, USING MULF WITH TRAPS ENABLED, TEST

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

TEST 15 UNDER\OVER FLOW, USING MULD WITH TRAPS ENABLED, TEST

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

TEST 16 MODF TEST

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

TEST 17 MODD TEST

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

537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593

THE INSTRUCTION AND CHECK THE RESULTS.

TEST 20 UNDER\OVER FLOW, USING MODF WITH TRAPS DISABLED, TEST

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

TEST 21 UNDER\OVER FLOW, USING MODD WITH TRAPS DISABLED, TEST

THIS IS A TEST OF THE MODD INSTRUCTION'S OVER FLOW AND UNDER FLOW CONDITIONS. A SUBROUTINE IS USED TO SET UP THE OPERANDS, EXECUTE THE MODD INSTRUCTION AND CHECK THE RESULTS.

TEST 22 INTERRUPT CORRECT FLOWS TEST

THIS IS A TEST OF THE 'CORRECT' FLOWS. THIS PART OF THE MICRO CODE HAS AS ITS PURPOSE INSURING THAT INTERRUPT REQUESTS MADE DURING CERTAIN LENGTHY FPP INSTRUCTIONS GET HONORED. THIS IS DONE IN A WAY SUCH THAT IF AN INTERRUPT REQUEST OCCURS DURING ONE OF THESE INSTRUCTIONS THE STATE OF THAT INSTRUCTION'S EXECUTION WILL BE THE SAME AS IF THAT INSTRUCTION HAD NEVER BEEN FETCHED AND ITS EXECUTION NEVER STARTED. THUS THE MICRO CODE WILL RESTORE ALL REGISTERS, BACK UP THE PC AND LEAVE THE FPS AND ACO THROUGH ACS UNMODIFIED.
THE INSTRUCTIONS FOR WHICH THIS IS NECESSARY ARE:

ADD (OR SUB)
DIV
MUL
MOD

(BOTH DOUBLE AND FLOATING)

ALL ADDRESSING MODES WILL BE TRIED WITH THE ADDD INSTRUCTION. THEN EACH OF THE OTHER INSTRUCTIONS WILL BE TRIED USING MODE 1. NOTE THAT THIS TEST NEEDS A SPECIAL INTERRUPT MODULE, WHICH WILL PROBABLY ONLY BE PRESENT IN DEC'S MANUFACTURING ENVIRONMENT, TO RUN. THIS SPECIAL EQUIPMENT IS DESIGNED TO RAISE AN INTERRUPT REQUEST IN THE PROCESSOR IF A BIT IS SET IN ITS STATUS REGISTER AND ONLY WHEN AN FPP INSTRUCTION IS ENCOUNTERED. THEREFORE THIS TEST WILL BE RUN CONDITIONALLY (DEPENDENT UPON WHETHER OR NOT THE STATUS REGISTER OF THE TEST EQUIPMENT TIMES OUT WHEN REFERENCED). THIS TEST CAN ALSO BE SELECTED BY TURNING SWITCH 7 OF THE SWITCH REGISTER (PHYSICAL OR VIRTUAL) ON. THE TEST ASSUMES THAT THE TEST EQUIPMENT'S STATUS REGISTER IS AT LOCATION 777774 (NOTE THAT ALL REFERENCES TO THIS LOCATION ARE MADE INDIRECT

594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616

THROUGH THIS PROGRAMS LOCATION CORINT, SO THAT IF THE USER HAS MODIFIED THE TEST EQUIPMENT'S STATUS REGISTER TO RESPOND TO A DIFFERENT ADDRESS LOCATION CORINT MUST BE MADE TO CONTAIN THAT STATUS REGISTER'S NEW ADDRESS). THIS PROGRAM ASSUMES THAT THE TRAP VECTOR FOR THE TEST EQUIPMENT IS 110. AGAIN NOTE THAT ALL REFERENCES TO THIS TRAP VECTOR ARE INDIRECT, THROUGH THIS PROGRAM'S LOCATION CORTRP (IF THE TEST EQUIPMENT IS MADE TO TRAP TO A DIFFERENT VECTOR LOCATION CORTRP MUST CONTAIN THE ADDRESS OF THIS VECTOR).

10.

LISTING
-----8


```

617      000267      MNUMBER=267
618      000002      PROGNUM=2
619      .LIST      ME
620      .NLIST     MD
621      .NLIST     MC
622      .NLIST     CND
623      .NLIST     BEX
1287     .MCALL    .HEADER, .SWRHI, .EQUAT, .SETUP, .SCATCH, .SACT11, .SCMTAG
1288     .MCALL    NEWTST, $$NEWTEST, $EOP, $SAVE, $TYPE, $TYPOCT
1289     .MCALL    .STYPDEC, .STRAP, .$POWER, .$APTHDR, .$APTBL5
1290     .MCALL    .$TYPE, .$APTYPE, .$READ
1291     .MCALL    .EQUIV      ;REMOVE FOR PDP-10 ASSEMBLY
1292
        .TITLE   CKFPB80 FP11F FLTG PNT PRT B
        ;*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

1293
1294
1295     000244      FPVECT=244
1296     177400      $SWR=177400
1297     000200      $SWRMSK=200
1298     000011      TAB=11
1299     000015      CRLF=15
1300
1301     .SBTTL   BASIC DEFINITIONS
        ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
        STACK= 1100
        ERROR=EMT
        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
000040
000020
000010
000004
000002
000001

BIT6=BIT06
BIT5=BIT05
BIT4=BIT04
BIT3=BIT03
BIT2=BIT02
BIT1=BIT01
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
```

1302
1303
1304
1305
1306
1307
1308
1309
1310
1312
1313

000000
000001
000002
000003
000004
000005
000006
000007

000000

```
.SBTTL FPP REGISTER DEFINITIONS
AC0   =%0
AC1   =%1
AC2   =%2
AC3   =%3
AC4   =%4
AC5   =%5
AC6   =%6
AC7   =%7
```

```
.SBTTL TRAP CATCHER
.=0
```

```
;*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

```
.=174
DISPREG: .WORD 0          ;;SOFTWARE DISPLAY REGISTER
SWREG:   .WORD 0          ;;SOFTWARE SWITCH REGISTER
```

000200 000137 004346

```
.SBTTL STARTING ADDRESS(ES)
JMP @#START ;;JUMP TO STARTING ADDRESS OF PROGRAM
```


1314

.SBTTL COMMON TAGS

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

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

 .=1100
\$CMTAG: .WORD 0
\$TSTNM: .BYTE 0
\$ERFLG: .BYTE 0
\$ICNT: .WORD 0
\$LPADR: .WORD 0
\$LPERR: .WORD 0
\$ERTTL: .WORD 0
\$ITEMB: .BYTE 0
\$ERMAX: .BYTE 1
\$ERRPC: .WORD 0
\$GDADR: .WORD 0
\$BDADR: .WORD 0
\$GDDAT: .WORD 0
\$BDDAT: .WORD 0
 .WORD 0
 .WORD 0
\$AUTOB: .BYTE 0
\$INTAG: .BYTE 0
 .WORD 0
SWR: .WORD DSWR
DISPLAY: .WORD DDISP
\$TKS: 177560
\$TKB: 177562
\$TPS: 177564
\$TPB: 177566
\$NULL: .BYTE 0
\$FILLS: .BYTE 2
\$FILLC: .BYTE 12
\$TPFLG: .BYTE 0
\$REGAD: .WORD 0

:::START OF COMMON TAGS
:::CONTAINS THE TEST NUMBER
:::CONTAINS ERROR FLAG
:::CONTAINS SUBTEST ITERATION COUNT
:::CONTAINS SCOPE LOOP ADDRESS
:::CONTAINS SCOPE RETURN FOR ERRORS
:::CONTAINS TOTAL ERRORS DETECTED
:::CONTAINS ITEM CONTROL BYTE
:::CONTAINS MAX. ERRORS PER TEST
:::CONTAINS PC OF LAST ERROR INSTRUCTION
:::CONTAINS ADDRESS OF 'GOOD' DATA
:::CONTAINS ADDRESS OF 'BAD' DATA
:::CONTAINS 'GOOD' DATA
:::CONTAINS 'BAD' DATA
:::RESERVED--NOT TO BE USED
:::AUTOMATIC MODE INDICATOR
:::INTERRUPT MODE INDICATOR
:::ADDRESS OF SWITCH REGISTER
:::ADDRESS OF DISPLAY REGISTER
:::TTY KBD STATUS
:::TTY KBD BUFFER
:::TTY PRINTER STATUS REG. ADDRESS
:::TTY PRINTER BUFFER REG. ADDRESS
:::CONTAINS NULL CHARACTER FOR FILLS
:::CONTAINS # OF FILLER CHARACTERS REQUIRED
:::INSERT FILL CHARS. AFTER A 'LINE FEED'
:::'TERMINAL AVAILABLE' FLAG (BIT<07>=0=YES)
:::CONTAINS THE ADDRESS FROM
:::WHICH (\$REG0) WAS OBTAINED

001162 000024
001162 000000
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

 .REPT \$CM3
\$REG0: .WORD 0
\$REG1: .WORD 0
\$REG2: .WORD 0
\$REG3: .WORD 0
\$REG4: .WORD 0
\$REG5: .WORD 0
\$REG6: .WORD 0
\$REG7: .WORD 0
\$REG10: .WORD 0
\$REG11: .WORD 0
\$REG12: .WORD 0
\$REG13: .WORD 0
\$REG14: .WORD 0
\$REG15: .WORD 0
\$REG16: .WORD 0
\$REG17: .WORD 0
\$REG20: .WORD 0
\$REG21: .WORD 0
\$REG22: .WORD 0

:::CONTAINS ((\$REGAD)+0)
:::CONTAINS ((\$REGAD)+2)
:::CONTAINS ((\$REGAD)+4)
:::CONTAINS ((\$REGAD)+6)
:::CONTAINS ((\$REGAD)+10)
:::CONTAINS ((\$REGAD)+12)
:::CONTAINS ((\$REGAD)+14)
:::CONTAINS ((\$REGAD)+16)
:::CONTAINS ((\$REGAD)+20)
:::CONTAINS ((\$REGAD)+22)
:::CONTAINS ((\$REGAD)+24)
:::CONTAINS ((\$REGAD)+26)
:::CONTAINS ((\$REGAD)+30)
:::CONTAINS ((\$REGAD)+32)
:::CONTAINS ((\$REGAD)+34)
:::CONTAINS ((\$REGAD)+36)
:::CONTAINS ((\$REGAD)+40)
:::CONTAINS ((\$REGAD)+42)
:::CONTAINS ((\$REGAD)+44)


```
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 $UNIT: .WORD AUNIT ;;I/O UNIT NUMBER
001332 000000 $MSGAD: .WORD AMSGAD ;;MESSAGE ADDRESS
001334 000000 $MSGLG: .WORD AMSGLG ;;MESSAGE LENGTH
001336 $ETABLE: ;;APT ENVIRONMENT TABLE
001336 000 $ENV: .BYTE AENV ;;ENVIRONMENT BYTE
001337 000 $ENVM: .BYTE AENVM ;;ENVIRONMENT MODE BITS
001340 000000 $SWREG: .WORD ASWREG ;;APT SWITCH REGISTER
001342 000000 $USWR: .WORD AUSWR ;;USER SWITCHES
001344 000000 $CPUOP: .WORD ACPUOP ;;CPU TYPE, OPTIONS
*
* BIT 15-11=CPU TYPE
* 11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
* 11/70=06,PDQ=07,Q=10
*
* BIT 10=REAL TIME CLOCK
* BIT 9=FLOATING POINT PROCESSOR
* BIT 8=MEMORY MANAGEMENT
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  $DEVM:  .WORD  ADEVM   ::DEVICE MAP
001376 000000  $CDW1:  .WORD  ACDW1   ::CONTROLLER DESCRIPTION WORD#1
001400 000000  $CDW2:  .WORD  ACDW2   ::CONTROLLER DESCRIPTION WORD#2
001402 000000  $DDW0:  .WORD  ADDW0   ::DEVICE DESCRIPTOR WORD#0
001404 000000  $DDW1:  .WORD  ADDW1   ::DEVICE DESCRIPTOR WORD#1
001406 000000  $DDW2:  .WORD  ADDW2   ::DEVICE DESCRIPTOR WORD#2
001410 000000  $DDW3:  .WORD  ADDW3   ::DEVICE DESCRIPTOR WORD#3
001412 000000  $DDW4:  .WORD  ADDW4   ::DEVICE DESCRIPTOR WORD#4
001414 000000  $DDW5:  .WORD  ADDW5   ::DEVICE DESCRIPTOR WORD#5
001416 000000  $DDW6:  .WORD  ADDW6   ::DEVICE DESCRIPTOR WORD#6
001420 000000  $DDW7:  .WORD  ADDW7   ::DEVICE DESCRIPTOR WORD#7
001422 000000  $DDW8:  .WORD  ADDW8   ::DEVICE DESCRIPTOR WORD#8
001424 000000  $DDW9:  .WORD  ADDW9   ::DEVICE DESCRIPTOR WORD#9
001426 000000  $DDW10: .WORD  ADDW10  ::DEVICE DESCRIPTOR WORD#10
001430 000000  $DDW11: .WORD  ADDW11  ::DEVICE DESCRIPTOR WORD#11
001432 000000  $DDW12: .WORD  ADDW12  ::DEVICE DESCRIPTOR WORD#12
001434 000000  $DDW13: .WORD  ADDW13  ::DEVICE DESCRIPTOR WORD#13
001436 000000  $DDW14: .WORD  ADDW14  ::DEVICE DESCRIPTOR WORD#14
001440 000000  $DDW15: .WORD  ADDW15  ::DEVICE DESCRIPTOR WORD#15
001442          $ETEND:

```


.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

1318	001442	000267			\$ERRTB:	
1320					.REPT	MNUMBER
	001442	040044	066436	071024	:ITEM 1	.WORD EM1,DH1,DT1,DF1
	001452	040076	066526	071062	:ITEM 2	.WORD EM2,DH2,DT2,DF2
	001462	040132	066436	071024	:ITEM 3	.WORD EM3,DH3,DT3,DF3
	001472	040237	066436	071024	:ITEM 4	.WORD EM4,DH4,DT4,DF4
	001502	040344	066436	071024	:ITEM 5	.WORD EM5,DH5,DT5,DF5
	001512	040451	066436	071024	:ITEM 6	.WORD EM6,DH6,DT6,DF6
	001522	040556	066436	071024	:ITEM 7	.WORD EM7,DH7,DT7,DF7
	001532	040663	066436	071024	:ITEM 10	.WORD EM10,DH10,DT10,DF10
	001542	040770	066436	071024	:ITEM 11	.WORD EM11,DH11,DT11,DF11
	001552	041077	066436	071024	:ITEM 12	.WORD EM12,DH12,DT12,DF12
	001562	041206	066436	071024	:ITEM 13	.WORD EM13,DH13,DT13,DF13
	001572	041322	066436	071024	:ITEM 14	.WORD EM14,DH14,DT14,DF14
	001602	041434	066436	071024	:ITEM 15	.WORD EM15,DH15,DT15,DF15
	001612	041546	066436	071024	:ITEM 16	.WORD EM16,DH16,DT16,DF16
	001622	041643	066567	071110	:ITEM 17	.WORD EM17,DH17,DT17,DF17
	001632	041720	066436	071130	:ITEM 20	.WORD EM20,DH20,DT20,DF20
	001642	041752	066642	071152	:ITEM 21	.WORD EM21,DH21,DT21,DF21
	001652	042004	066731	071174	:ITEM 22	.WORD EM22,DH22,DT22,DF22
	001662	042065	066436	071212	:ITEM 23	.WORD EM23,DH23,DT23,DF23
	001672	042142	066436	071212	:ITEM 24	.WORD EM24,DH24,DT24,DF24
	001702	042240	066436	071212	:ITEM 25	.WORD EM25,DH25,DT25,DF25
	001712	042325	066436	071212	:ITEM 26	.WORD EM26,DH26,DT26,DF26
					:ITEM 27	

001722	042240	066436	071212		.WORD	EM27,DH27,DT27,DF27
001732	042423	066436	071212	:ITEM 30	.WORD	EM30,DH30,DT30,DF30
001742	042475	066436	071212	:ITEM 31	.WORD	EM31,DH31,DT31,DF31
001752	042110	066436	071212	:ITEM 32	.WORD	EM32,DH32,DT32,DF32
001762	042542	066436	071212	:ITEM 33	.WORD	EM33,DH33,DT33,DF33
001772	042565	066436	071212	:ITEM 34	.WORD	EM34,DH34,DT34,DF34
002002	042617	066436	071212	:ITEM 35	.WORD	EM35,DH35,DT35,DF35
002012	042672	066436	071212	:ITEM 36	.WORD	EM36,DH36,DT36,DF36
002022	042740	066436	071212	:ITEM 37	.WORD	EM37,DH37,DT37,DF37
002032	042763	066436	071212	:ITEM 40	.WORD	EM40,DH40,DT40,DF40
002042	043015	066436	071212	:ITEM 41	.WORD	EM41,DH41,DT41,DF41
002052	043070	066436	071212	:ITEM 42	.WORD	EM42,DH42,DT42,DF42
002062	043221	066436	071212	:ITEM 43	.WORD	EM43,DH43,DT43,DF43
002072	043352	066436	071212	:ITEM 44	.WORD	EM44,DH44,DT44,DF44
002102	043420	066436	071212	:ITEM 45	.WORD	EM45,DH45,DT45,DF45
002112	043473	066436	071212	:ITEM 46	.WORD	EM46,DH46,DT46,DF46
002122	043550	066436	071212	:ITEM 47	.WORD	EM47,DH47,DT47,DF47
002132	043704	066436	071212	:ITEM 50	.WORD	EM50,DH50,DT50,DF50
002142	043757	066436	071212	:ITEM 51	.WORD	EM51,DH51,DT51,DF51
002152	044025	066436	071212	:ITEM 52	.WORD	EM52,DH52,DT52,DF52
002162	051655	066436	071264	:ITEM 53	.WORD	EM53,DH53,DT53,DF53
002172	051706	066436	071264	:ITEM 54	.WORD	EM54,DH54,DT54,DF54
002202	051623	066436	071264	:ITEM 55	.WORD	EM55,DH55,DT55,DF55
002212	051736	066436	071264	:ITEM 56	.WORD	EM56,DH56,DT56,DF56
002222	052027	066436	071264	:ITEM 57	.WORD	EM57,DH57,DT57,DF57
002232	052117	066436	071264	:ITEM 60	.WORD	EM60,DH60,DT60,DF60
002242	052221	066436	071264	:ITEM 61	.WORD	EM61,DH61,DT61,DF61
002252	052415	066436	071264	:ITEM 62	.WORD	EM62,DH62,DT62,DF62
002262	052611	066436	071264	:ITEM 63	.WORD	EM63,DH63,DT63,DF63

002272	052722	066436	071264	:ITEM 64	.WORD	EM64,DH64,DT64,DF64
002302	053041	066436	071264	:ITEM 65	.WORD	EM65,DH65,DT65,DF65
002312	053154	066436	071264	:ITEM 66	.WORD	EM66,DH66,DT66,DF66
002322	053223	066436	071264	:ITEM 67	.WORD	EM67,DH67,DT67,DF67
002332	053306	066436	071264	:ITEM 70	.WORD	EM70,DH70,DT70,DF70
002342	053337	066436	071264	:ITEM 71	.WORD	EM71,DH71,DT71,DF71
002352	053367	066436	071264	:ITEM 72	.WORD	EM72,DH72,DT72,DF72
002362	053367	066436	071264	:ITEM 73	.WORD	EM73,DH73,DT73,DF73
002372	053421	066436	071264	:ITEM 74	.WORD	EM74,DH74,DT74,DF74
002402	053530	066436	071264	:ITEM 75	.WORD	EM75,DH75,DT75,DF75
002412	053621	066436	071264	:ITEM 76	.WORD	EM76,DH76,DT76,DF76
002422	053711	066436	071264	:ITEM 77	.WORD	EM77,DH77,DT77,DF77
002432	054021	066436	071264	:ITEM 100	.WORD	EM100,DH100,DT100,DF100
002442	054104	066436	071264	:ITEM 101	.WORD	EM101,DH101,DT101,DF101
002452	054300	066436	071264	:ITEM 102	.WORD	EM102,DH102,DT102,DF102
002462	054474	066436	071264	:ITEM 103	.WORD	EM103,DH103,DT103,DF103
002472	054606	066436	071264	:ITEM 104	.WORD	EM104,DH104,DT104,DF104
002502	054753	066436	071264	:ITEM 105	.WORD	EM105,DH105,DT105,DF105
002512	055062	066436	071264	:ITEM 106	.WORD	EM106,DH106,DT106,DF106
002522	055171	066436	071264	:ITEM 107	.WORD	EM107,DH107,DT107,DF107
002532	055300	066436	071264	:ITEM 110	.WORD	EM110,DH110,DT110,DF110
002542	044115	066436	071212	:ITEM 111	.WORD	EM111,DH111,DT111,DF111
002552	044172	066436	071212	:ITEM 112	.WORD	EM112,DH112,DT112,DF112
002562	044250	066436	071212	:ITEM 113	.WORD	EM113,DH113,DT113,DF113
002572	044326	066436	071212	:ITEM 114	.WORD	EM114,DH114,DT114,DF114
002602	044405	066777	071352	:ITEM 115	.WORD	EM115,DH115,DT115,DF115
002612	044463	066777	071352	:ITEM 116	.WORD	EM116,DH116,DT116,DF116
002622	044542	066777	071352	:ITEM 117	.WORD	EM117,DH117,DT117,DF117
				:ITEM 120		

002632	044700	066777	071352		.WORD	EM120,DH120,DT120,DF120
				:ITEM 121	.WORD	
002642	045036	066777	071352		.WORD	EM121,DH121,DT121,DF121
				:ITEM 122	.WORD	
002652	045173	066777	071352		.WORD	EM122,DH122,DT122,DF122
				:ITEM 123	.WORD	
002662	045330	066436	071212		.WORD	EM123,DH123,DT123,DF123
				:ITEM 124	.WORD	
002672	045406	066436	071212		.WORD	EM124,DH124,DT124,DF124
				:ITEM 125	.WORD	
002702	045465	066436	071212		.WORD	EM125,DH125,DT125,DF125
				:ITEM 126	.WORD	
002712	045543	066436	071212		.WORD	EM126,DH126,DT126,DF126
				:ITEM 127	.WORD	
002722	045622	066777	071352		.WORD	EM127,DH127,DT127,DF127
				:ITEM 130	.WORD	
002732	045700	066777	071352		.WORD	EM130,DH130,DT130,DF130
				:ITEM 131	.WORD	
002742	045757	066777	071352		.WORD	EM131,DH131,DT131,DF131
				:ITEM 132	.WORD	
002752	046115	066436	071212		.WORD	EM132,DH132,DT132,DF132
				:ITEM 133	.WORD	
002762	046253	066777	071352		.WORD	EM133,DH133,DT133,DF133
				:ITEM 134	.WORD	
002772	046411	066777	071352		.WORD	EM134,DH134,DT134,DF134
				:ITEM 135	.WORD	
003002	046546	066436	071212		.WORD	EM135,DH135,DT135,DF135
				:ITEM 136	.WORD	
003012	046703	066777	071352		.WORD	EM136,DH136,DT136,DF136
				:ITEM 137	.WORD	
003022	047040	066777	071352		.WORD	EM137,DH137,DT137,DF137
				:ITEM 140	.WORD	
003032	047126	066777	071352		.WORD	EM140,DH140,DT140,DF140
				:ITEM 141	.WORD	
003042	044115	066777	071352		.WORD	EM141,DH141,DT141,DF141
				:ITEM 142	.WORD	
003052	044172	066777	071352		.WORD	EM142,DH142,DT142,DF142
				:ITEM 143	.WORD	
003062	047215	066777	071352		.WORD	EM143,DH143,DT143,DF143
				:ITEM 144	.WORD	
003072	047273	066777	071352		.WORD	EM144,DH144,DT144,DF144
				:ITEM 145	.WORD	
003102	047352	066436	071212		.WORD	EM145,DH145,DT145,DF145
				:ITEM 146	.WORD	
003112	047517	066436	071212		.WORD	EM146,DH146,DT146,DF146
				:ITEM 147	.WORD	
003122	047664	066436	071212		.WORD	EM147,DH147,DT147,DF147
				:ITEM 150	.WORD	
003132	050030	066436	071212		.WORD	EM150,DH150,DT150,DF150
				:ITEM 151	.WORD	
003142	050174	066777	071352		.WORD	EM151,DH151,DT151,DF151
				:ITEM 152	.WORD	
003152	050262	066777	071352		.WORD	EM152,DH152,DT152,DF152
				:ITEM 153	.WORD	
003162	045330	066777	071352		.WORD	EM153,DH153,DT153,DF153
				:ITEM 154	.WORD	
003172	045406	066777	071352		.WORD	EM154,DH154,DT154,DF154

003202	050351	066777	071352	:ITEM 155	WORD	EM155,DH155,DT155,DF155
003212	050427	066777	071352	:ITEM 156	WORD	EM156,DH156,DT156,DF156
003222	050506	066436	071212	:ITEM 157	WORD	EM157,DH157,DT157,DF157
003232	050653	066777	071352	:ITEM 160	WORD	EM160,DH160,DT160,DF160
003242	051011	066436	071212	:ITEM 161	WORD	EM161,DH161,DT161,DF161
003252	051156	066436	071212	:ITEM 162	WORD	EM162,DH162,DT162,DF162
003262	051322	066777	071352	:ITEM 163	WORD	EM163,DH163,DT163,DF163
003272	051457	066436	071212	:ITEM 164	WORD	EM164,DH164,DT164,DF164
003302	055410	066777	071426	:ITEM 165	WORD	EM165,DH165,DT165,DF165
003312	055474	066777	071426	:ITEM 166	WORD	EM166,DH166,DT166,DF166
003322	055557	066777	071426	:ITEM 167	WORD	EM167,DH167,DT167,DF167
003332	055611	066777	071426	:ITEM 170	WORD	EM170,DH170,DT170,DF170
003342	055677	066777	071426	:ITEM 171	WORD	EM171,DH171,DT171,DF171
003352	056022	066777	071426	:ITEM 172	WORD	EM172,DH172,DT172,DF172
003362	056107	066777	071426	:ITEM 173	WORD	EM173,DH173,DT173,DF173
003372	056255	066777	071426	:ITEM 174	WORD	EM174,DH174,DT174,DF174
003402	056423	066777	071426	:ITEM 175	WORD	EM175,DH175,DT175,DF175
003412	056535	066777	071426	:ITEM 176	WORD	EM176,DH176,DT176,DF176
003422	056621	067076	071514	:ITEM 177	WORD	EM177,DH177,DT177,DF177
003432	056655	066777	071426	:ITEM 200	WORD	EM200,DH200,DT200,DF200
003442	056707	066777	071426	:ITEM 201	WORD	EM201,DH201,DT201,DF201
003452	056776	066777	071426	:ITEM 202	WORD	EM202,DH202,DT202,DF202
003462	057140	066777	071426	:ITEM 203	WORD	EM203,DH203,DT203,DF203
003472	057302	066777	071426	:ITEM 204	WORD	EM204,DH204,DT204,DF204
003502	057370	066777	071426	:ITEM 205	WORD	EM205,DH205,DT205,DF205
003512	057536	066777	071426	:ITEM 206	WORD	EM206,DH206,DT206,DF206
003522	057704	067177	071524	:ITEM 207	WORD	EM207,DH207,DT207,DF207
003532	057746	067267	071576	:ITEM 210	WORD	EM210,DH210,DT210,DF210
				:ITEM 211		

003542	060010	067267	071524		.WORD	EM211,DH211,DT211,DF211
				:ITEM 212		
003552	060154	067177	071620		.WORD	EM212,DH212,DT212,DF212
				:ITEM 213		
003562	060340	067177	071620		.WORD	EM213,DH213,DT213,DF213
				:ITEM 214		
003572	060524	067177	071620		.WORD	EM214,DH214,DT214,DF214
				:ITEM 215		
003602	060710	067177	071620		.WORD	EM215,DH215,DT215,DF215
				:ITEM 216		
003612	061074	067267	071524		.WORD	EM216,DH216,DT216,DF216
				:ITEM 217		
003622	061256	067330	071702		.WORD	EM217,DH217,DT217,DF217
				:ITEM 220		
003632	061320	067137	071524		.WORD	EM220,DH220,DT220,DF220
				:ITEM 221		
003642	061550	067267	071524		.WORD	EM221,DH221,DT221,DF221
				:ITEM 222		
003652	062014	067137	071524		.WORD	EM222,DH222,DT222,DF222
				:ITEM 223		
003662	062245	067267	071524		.WORD	EM223,DH223,DT223,DF223
				:ITEM 224		
003672	062512	067137	071524		.WORD	EM224,DH224,DT224,DF224
				:ITEM 225		
003702	062743	067267	071524		.WORD	EM225,DH225,DT225,DF225
				:ITEM 226		
003712	063210	067403	071620		.WORD	EM226,DH226,DT226,DF226
				:ITEM 227		
003722	063343	067472	071620		.WORD	EM227,DH227,DT227,DF227
				:ITEM 230		
003732	063476	067403	071620		.WORD	EM230,DH230,DT230,DF230
				:ITEM 231		
003742	063632	067472	071576		.WORD	EM231,DH231,DT231,DF231
				:ITEM 232		
003752	057746	067472	071524		.WORD	EM232,DH232,DT232,DF232
				:ITEM 233		
003762	063766	067561	071722		.WORD	EM233,DH233,DT233,DF233
				:ITEM 234		
003772	064025	067622	071756		.WORD	EM234,DH234,DT234,DF234
				:ITEM 235		
004002	064111	067710	071776		.WORD	EM235,DH235,DT235,DF235
				:ITEM 236		
004012	064157	067750	071756		.WORD	EM236,DH236,DT236,DF236
				:ITEM 237		
004022	064212	067622	071756		.WORD	EM237,DH237,DT237,DF237
				:ITEM 240		
004032	064276	070036	071756		.WORD	EM240,DH240,DT240,DF240
				:ITEM 241		
004042	064332	067561	071722		.WORD	EM241,DH241,DT241,DF241
				:ITEM 242		
004052	064435	070036	071756		.WORD	EM242,DH242,DT242,DF242
				:ITEM 243		
004062	064471	067561	071722		.WORD	EM243,DH243,DT243,DF243
				:ITEM 244		
004072	064574	067561	071722		.WORD	EM244,DH244,DT244,DF244
				:ITEM 245		
004102	064633	067561	071722		.WORD	EM245,DH245,DT245,DF245

004112	043525	066436	071212	:ITEM 246	.WORD	EM246,DH246,DT246,DF246
004122	064715	070126	072010	:ITEM 247	.WORD	EM247,DH247,DT247,DF247
004132	064751	070173	072026	:ITEM 250	.WORD	EM250,DH250,DT250,DF250
004142	065003	070173	072026	:ITEM 251	.WORD	EM251,DH251,DT251,DF251
004152	065036	066526	072040	:ITEM 252	.WORD	EM252,DH252,DT252,DF252
004162	065127	066436	072052	:ITEM 253	.WORD	EM253,DH253,DT253,DF253
004172	065213	066436	072052	:ITEM 254	.WORD	EM254,DH254,DT254,DF254
004202	065300	066436	072052	:ITEM 255	.WORD	EM255,DH255,DT255,DF255
004212	065366	066436	072052	:ITEM 256	.WORD	EM256,DH256,DT256,DF256
004222	065455	066436	072052	:ITEM 257	.WORD	EM257,DH257,DT257,DF257
004232	065543	066436	072052	:ITEM 260	.WORD	EM260,DH260,DT260,DF260
004242	065632	066436	072052	:ITEM 261	.WORD	EM261,DH261,DT261,DF261
004252	065722	066436	072052	:ITEM 262	.WORD	EM262,DH262,DT262,DF262
004262	066013	066436	072052	:ITEM 263	.WORD	EM263,DH263,DT263,DF263
004272	066100	066436	072052	:ITEM 264	.WORD	EM264,DH264,DT264,DF264
004302	066165	066436	072052	:ITEM 265	.WORD	EM265,DH265,DT265,DF265
004312	066252	070233	072040	:ITEM 266	.WORD	EM266,DH266,DT266,DF266
004322	066353	066777	071426	:ITEM 267	.WORD	EM267,DH267,DT267,DF267

1321
1322
1323

000046 004332
000052 000046
000000 033366
004332 000052
004332 000000

1324

000024 004332
000044 000024
000044 000200
000044 000044
004332 004332
004332 004332

.SBTTL ACT11 HOOKS

:HOOKS REQUIRED BY ACT11

 \$SVPC=. :SAVE PC
 .=46
 \$ENDAD :;1)SET LOC.46 TO ADDRESS OF \$ENDAD IN .\$EOP
 .=52
 .WORD 0 :;2)SET LOC.52 TO ZERO
 .=\$SVPC :; RESTORE PC

.SBTTL APT PARAMETER BLOCK

:SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT

 .\$X=. :;SAVE CURRENT LOCATION
 .=24 :;SET POWER FAIL TO POINT TO START OF PROGRAM
 200 :;FOR APT START UP
 .=44 :;POINT TO APT INDIRECT ADDRESS PNTR.
 \$APTHDR :;POINT TO APT HEADER BLOCK
 .=.\$X :;RESET LOCATION COUNTER


```

004332
004332 000000
004334 001316
004336 000010
004340 000040
004342 000000
004344 000052

1325
1326
1327 004346

004346 012706 001100
004352 005026
004354 022706 001140
004360 001374
004362 012706 001100

004366 012737 033446 000020
004374 012737 000340 000022
004402 012737 033764 000030
004410 012737 000340 000032
004416 012737 036120 000034
004424 012737 000340 000036
004432 012737 036204 000024
004440 012737 000340 000026
004446 013737 033210 033202
004454 005037 001302
004460 005037 001304
004464 112737 000001 001115

004472 012737 033432 000014
004500 012737 000340 000016
004506 012737 000002 033432
004514 012737 004542 000010
004522 005046
004524 012746 004532
004530 000006
004532 012737 000006 033432
004540 000402
004542 062706 000010
004546 012737 000012 000010
004554 005037 033440
004560 012737 004560 001106
004566 012737 004566 001110

004574 013746 000004
004600 012737 004634 000004
004606 012737 177570 001140
004614 012737 177570 001142
004622 022777 177777 174310
004630 001012
    
```

```

*****
:SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
:INTERFACE SPEC.
$APTHD:
$HIBTS: .WORD 0 ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
$MBADR: .WORD $MAIL ;;ADDRESS OF APT MAILBOX (BITS 0-15)
$TSTM: .WORD 10 ;;RUN TIM OF LONGEST TEST
$PASTM: .WORD 40 ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
$UNITM: .WORD 0 ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
        .WORD $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)

START:
.SBTTL INITIALIZE THE COMMON TAGS
:;CLEAR THE COMMON TAGS ($CMTAG) AREA
MOV     #$CMTAG,R6 ;;FIRST LOCATION TO BE CLEARED
CLR     (R6)+      ;;CLEAR MEMORY LOCATION
CMP     #SWR,R6   ;;DONE?
BNE     -6        ;;LOOP BACK IF NO
MOV     #STACK,SP ;;SETUP THE STACK POINTER

:;INITIALIZE A FEW VECTORS
MOV     #SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
MOV     #340,@#IOTVEC+2 ;;LEVEL 7
MOV     #ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
MOV     #340,@#EMTVEC+2 ;;LEVEL 7
MOV     #STRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
MOV     #340,@#TRAPVEC+2;LEVEL 7
MOV     #SPWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
MOV     #340,@#PWRVEC+2 ;;LEVEL 7
MOV     $ENDCT,$EOPCT ;;SETUP END-OF-PROGRAM COUNTER
CLR     $TIMES      ;;INITIALIZE NUMBER OF ITERATIONS
CLR     $ESCAPE     ;;CLEAR THE ESCAPE ON ERROR ADDRESS
MOVB   #1,$ERMAX   ;;ALLOW ONE ERROR PER TEST

:;INITIALIZE THE 'T-BIT' TRAP VECTOR. THEN LOAD LOCATION '$RTRN', IN
:;THE 'END-OF-PASS' ($EOP) ROUTINE, WITH A 'RTI' OR 'RTT'.
MOV     #$RTRN,@#TBITVEC ;;SET 'T' BIT VECTOR TO $RTRN
MOV     #340,@#TBITVEC+2 ;;LEVEL 7
MOV     #RTI,$RTRN   ;;SET $RTRN TO A RTI
MOV     #65$,@#RESVEC ;;TRY TO DO A RTT
CLR     -(SP)       ;;DUMMY PS
MOV     #64$,-(SP)  ;;AND PC
RTT     ;;TRY THE RTT
64$:   MOV     #RTT,$RTRN ;;RTT IS LEGAL--SET $RTRN TO A RTT
BR     66$
65$:   ADD     #10,SP ;;RTT ILLEGAL--CLEAN OFF THE STACK
66$:   MOV     #RESVEC+2,@#RESVEC ;;RESTORE TRAP CATCHER
CLR     $TBIT      ;;CLEAR 'T' BIT SWITCH
MOV     #,$SLPADR   ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
MOV     #,$SLPERR   ;;SETUP THE ERROR LOOP ADDRESS

:;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
:;EQUAL TO A '-1', SETUP FOR A SOFTWARE SWITCH REGISTER.
MOV     @#ERRVEC,-(SP) ;;SAVE ERROR VECTOR
MOV     #67$,@#ERRVEC ;;SET UP ERPOP VECTOR
MOV     #DSWR,SWR   ;;SETUP FOR A HARDWARE SWICH REGISTER
MOV     #DDISP,DISPLAY ;;AND A HARDWARE DISPLAY REGISTER
CMP     #-1,@SWR   ;;TRY TO REFERENCE HARDWARE SWR
BNE     69$       ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
    
```



```
004632 000403  
004634 012716 004642 67$: BR 68$ ::AND THE HARDWARE SWR IS NOT = -1  
004640 000002 RTI #68$, (SP) ::BRANCH IF NO TIMEOUT  
004642 012737 000176 001140 68$: MOV #SWREG, SWR ::POINT TO SOFTWARE SWR  
004650 012737 000174 001142 MOV #DISPREG, DISPLAY  
004656 012637 000004 69$: MOV (SP)+, @#ERRVEC ::RESTORE ERROR VECTOR  
004662 005037 001324 CLR $PASS ::CLEAR PASS COUNT  
004666 132737 000200 001337 BITB #APTSIZE, $ENVM ::TEST USER SIZE UNDER APT  
004674 001403 BEQ 70$ ::YES, USE NON-APT SWITCH  
004676 012737 001340 001140 MOV #$$SWREG, SWR ::NO, USE APT SWITCH REGISTER  
004704 70$:  
1328 .SBTTL TYPE PROGRAM NAME  
::TYPE THE NAME OF THE PROGRAM IF FIRST PASS  
004704 005227 177777 INC #-1 ::FIRST TIME?  
004710 001047 BNE 71$ ::BRANCH IF NO  
004712 022737 033366 000042 CMP #SENDAD, @#42 ::ACT-11?  
004720 001443 BEQ 71$ ::BRANCH IF YES  
004722 104401 004770 TYPE ,72$ ::TYPE ASCIZ STRING  
.SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER  
004726 005737 000042 TST @#42 ::ARE WE RUNNING UNDER XXDP/ACT?  
004732 001012 BNE 73$ ::BRANCH IF YES  
004734 123727 001336 000001 CMPB $ENV, #1 ::ARE WE RUNNING UNDER APT?  
004742 001406 BEQ 73$ ::BRANCH IF YES  
004744 023727 001140 000176 CMP SWR, #SWREG ::SOFTWARE SWITCH REG SELECTED?  
004752 001005 BNE 74$ ::BRANCH IF NO  
004754 104405 GTSWR ::GET SOFT-SWR SETTINGS  
004756 000403 BR 74$  
004760 112737 000001 001134 73$: MOVB #1, $AUTOB ::SET AUTO-MODE INDICATOR  
004766 74$:  
004766 000420 BR 71$ ::GET OVER THE ASCIZ  
005030 71$: .ASCIZ <CRLF>*CKFPBBO FP11F FLTG PNT PRT B*<CRLF>  
1329  
1330 005030 LOOP:  
1331  
1332
```


1386	005204	172410			LDD	(R0),ACO	
1387	005206	012700	006672		MOV	#HHP6,R0	:FSPC
1388	005212	172010		HH8:	ADDD	(R0),ACO	:TEST INSTRUCTION
1389	005214	170205			STFPS	R5	:GET FPS
1390	005216	012700	006602		MOV	#HHDATO,R0	:GET THE RESULT
1391	005222	174010			STD	ACO,(R0)	
1392	005224	012701	006652		MOV	#HHP4,R1	:IS IT CORRECT
1393	005230	012702	000004		MOV	#4,R2	
1394	005234	022021		HH9:	CMP	(R0)+,(R1)+	
1395	005236	001402			BEQ	HH10	
1396	005240	000137	006004		JMP	HHER2	
1397	005244	077205		HH10:	SOB	R2,HH9	
1398	005246	052704	100004		BIS	#100004,R4	:FPS CORRECT?
1399	005252	020405			CMP	R4,R5	
1400	005254	001402			BEQ	HH11	
1401	005256	000137	006052		JMP	HHER3	
1402							
1403							
1404	005262						
	005262	104413		HH11:	LPERR		:SET UP THE LOOP ON ERROR ADDRESS. :SET FIV, FIV, AND FD
1405							
1406	005264	012704	043200		MOV	#043200,R4	
1407	005270	170104			LDFPS	R4	
1408	005272	012737	005312	001236	MOV	#HH12,\$TMP2	
1409	005300	012700	006712		MOV	#HHP8,R0	:SET ACO OPERAND
1410	005304	172410			LDD	(R0),ACO	
1411	005306	012700	006722		MOV	#HHP9,R0	:FSPC
1412	005312	172010		HH12:	ADDD	(R0),ACO	:TEST INSTRUCTION
1413	005314	170205			STFPS	R5	:GET FPS
1414	005316	012700	006602		MOV	#HHDATO,R0	:GET THE RESULT
1415	005322	174010			STD	ACO,(R0)	
1416	005324	012701	006702		MOV	#HHP7,R1	:IS IT CORRECT
1417	005330	012702	000004		MOV	#4,R2	
1418	005334	022021		HH13:	CMP	(R0)+,(R1)+	
1419	005336	001415			BEQ	HH16	
1420	005340	012700	006602		MOV	#HHDATO,R0	
1421	005344	012701	006652		MOV	#HHP4,R1	
1422	005350	012702	000004		MOV	#4,R2	
1423	005354	022021		HH14:	CMP	(R0)+,(R1)+	
1424	005356	001402			BEQ	HH15	
1425	005360	000137	006120		JMP	HHER4	
1426	005364	077205		HH15:	SOB	R2,HH14	
1427	005366	000137	006166		JMP	HHER5	
1428	005372	077220		HH16:	SOB	R2,HH13	
1429	005374	052704	100014		BIS	#100014,R4	:FPS CORRECT?
1430	005400	020405			CMP	R4,R5	
1431	005402	001402			BEQ	HH17	
1432	005404	000137	006234		JMP	HHER6	
1433							
1434	005410						
	005410	104413		HH17:	LPERR		:SET UP THE LOOP ON ERROR ADDRESS. :SET FIV, FIV, AND FD
1435	005412	012704	000200		MOV	#00200,R4	
1436	005416	170104			LDFPS	R4	
1437	005420	012737	005446	001236	MOV	#HH18,\$TMP2	
1438	005426	012737	037000	000244	MOV	#FPSPUR,FPVECT	
1439	005434	012700	006712		MOV	#HHP8,R0	:SET ACO OPERAND
1440	005440	172410			LDD	(R0),ACO	

:THIS IS A TEST OF THE R\T ALGORITHM'S
:ABILITY TO SET BOTH N AND Z ON A - 0 RESULT.

:TEST THAT CC ARE CLEARED BY R\T

1441	005442	012700	006712			MOV	#HHP8,R0		:FSPC
1442	005446	172010			HH18:	ADDD	(R0),ACO		:TEST INSTRUCTION
1443	005450	170205				STFPS	R5		:GET FPS
1444	005452	012700	006602			MOV	#HHDATO,R0		:GET THE RESULT
1445	005456	174010				STD	ACO,(R0)		
1446	005460	012701	006732			MOV	#HHP10,R1		:IS IT CORRECT
1447	005464	012702	000004			MOV	#4,R2		
1448	005470	022021			HH19:	CMP	(R0)+,(R1)+		
1449	005472	001402				BEQ	HH20		
1450	005474	000137	006302			JMP	HHER7		
1451	005500	077205			HH20:	SOB	R2,HH19		
1452	005502	052704	000000			BIS	#00000,R4		:FPS CORRECT?
1453	005506	020405				CMP	R4,R5		
1454	005510	001402				BEQ	HH21		
1455	005512	000137	006350			JMP	HHER8		
1456									
1457	005516								
	005516	104413			HH21:	LPERR			:TEST THAT N IS SET BY R\T
1458	005520	012704	003200			MOV	#3200,R4		:SET UP THE LOOP ON ERROR ADDRESS.
1459	005524	170104				LDFPS	R4		:SET FIV, FIV, AND FD
1460	005526	012737	005546	001236		MOV	#HH22,\$TMP2		
1461	005534	012700	006662			MOV	#HHP5,R0		:SET ACO OPERAND
1462	005540	172410				LDD	(R0),ACO		
1463	005542	012700	006662			MOV	#HHP5,R0		:FSPC
1464	005546	172010			HH22:	ADDD	(R0),ACO		:TEST INSTRUCTION
1465	005550	170205				STFPS	R5		:GET FPS
1466	005552	012700	006602			MOV	#HHDATO,R0		:GET THE RESULT
1467	005556	174010				STD	ACO,(R0)		
1468	005560	012701	006742			MOV	#HHP11,R1		:IS IT CORRECT
1469	005564	012702	000004			MOV	#4,R2		
1470	005570	022021			HH23:	CMP	(R0)+,(R1)+		
1471	005572	001402				BEQ	HH24		
1472	005574	000137	006416			JMP	HHER9		
1473	005600	077205			HH24:	SOB	R2,HH23		
1474	005602	052704	000010			BIS	#10,R4		
1475	005606	020405				CMP	R4,R5		:FPS CORRECT?
1476	005610	001402				BEQ	HH25		
1477	005612	000137	006464			JMP	HHER10		
1478	005616	000137	006752			JMP	HHDONE		
1479	005622				HH25:				
	005622	010537	001252		HHERO:	MOV	R5,\$TMP10		
	005626	010437	001254			MOV	R4,\$TMP11		
	005632	012737	006622	001240		MOV	#HHP1,\$TMP3		
	005640	012737	006612	001242		MOV	#HHPO,\$TMP4		
	005646	012737	006602	001244		MOV	#HHDATO,\$TMP5		
	005654	012737	006632	001246		MOV	#HHP2,\$TMP6		
	005662	104207			1\$:	ERROR	+207		
	005664	000137	006752			JMP	HHDONE		
1480	005670				HHER1:				
	005670	010537	001252			MOV	R5,\$TMP10		
	005674	010437	001254			MOV	R4,\$TMP11		
	005700	012737	006622	001240		MOV	#HHP1,\$TMP3		
	005706	012737	006612	001242		MOV	#HHPO,\$TMP4		
	005714	012737	006602	001244		MOV	#HHDATO,\$TMP5		
	005722	012737	006632	001246		MOV	#HHP2,\$TMP6		
	005730	104211			1\$:	ERROR	+211		
	005732	000137	006752			JMP	HHDONE		

1481	005736				HHER00:		
	005736	010537	001252			MOV	R5,\$TMP10
	005742	010437	001254			MOV	R4,\$TMP11
	005746	012737	006622	001240		MOV	#HHP1,\$TMP3
	005754	012737	006612	001242		MOV	#HHP0,\$TMP4
	005762	012737	006602	001244		MOV	#HHDATO,\$TMP5
	005770	012737	006632	001246		MOV	#HHP2,\$TMP6
	005776	104210			1\$:	ERROR	+210
	006000	000137	006752			JMP	HHDONE
1482	006004				HHER2:		
	006004	010537	001252			MOV	R5,\$TMP10
	006010	010437	001254			MOV	R4,\$TMP11
	006014	012737	006672	001240		MOV	#HHP6,\$TMP3
	006022	012737	006662	001242		MOV	#HHP5,\$TMP4
	006030	012737	006602	001244		MOV	#HHDATO,\$TMP5
	006036	012737	006652	001246		MOV	#HHP4,\$TMP6
	006044	104207			1\$:	ERROR	+207
	006046	000137	006752			JMP	HHDONE
1483	006052				HHER3:		
	006052	010537	001252			MOV	R5,\$TMP10
	006056	010437	001254			MOV	R4,\$TMP11
	006062	012737	006672	001240		MOV	#HHP6,\$TMP3
	006070	012737	006662	001242		MOV	#HHP5,\$TMP4
	006076	012737	006602	001244		MOV	#HHDATO,\$TMP5
	006104	012737	006652	001246		MOV	#HHP4,\$TMP6
	006112	104214			1\$:	ERROR	+214
	006114	000137	006752			JMP	HHDONE
1484	006120				HHER4:		
	006120	010537	001252			MOV	R5,\$TMP10
	006124	010437	001254			MOV	R4,\$TMP11
	006130	012737	006722	001240		MOV	#HHP9,\$TMP3
	006136	012737	006712	001242		MOV	#HHP8,\$TMP4
	006144	012737	006602	001244		MOV	#HHDATO,\$TMP5
	006152	012737	006702	001246		MOV	#HHP7,\$TMP6
	006160	104207			1\$:	ERROR	+207
	006162	000137	006752			JMP	HHDONE
1485	006166				HHER5:		
	006166	010537	001252			MOV	R5,\$TMP10
	006172	010437	001254			MOV	R4,\$TMP11
	006176	012737	006722	001240		MOV	#HHP9,\$TMP3
	006204	012737	006712	001242		MOV	#HHP8,\$TMP4
	006212	012737	006602	001244		MOV	#HHDATO,\$TMP5
	006220	012737	006702	001246		MOV	#HHP7,\$TMP6
	006226	104216			1\$:	ERROR	+216
	006230	000137	006752			JMP	HHDONE
1486	006234				HHER6:		
	006234	010537	001252			MOV	R5,\$TMP10
	006240	010437	001254			MOV	R4,\$TMP11
	006244	012737	006722	001240		MOV	#HHP9,\$TMP3
	006252	012737	006712	001242		MOV	#HHP8,\$TMP4
	006260	012737	006602	001244		MOV	#HHDATO,\$TMP5
	006266	012737	006702	001246		MOV	#HHP7,\$TMP6
	006274	104215			1\$:	ERROR	+215
	006276	000137	006752			JMP	HHDONE
1487	006302				HHER7:		
	006302	010537	001252			MOV	R5,\$TMP10
	006306	010437	001254			MOV	R4,\$TMP11

006312	012737	006712	001240	MOV	#HHP8,\$TMP3	
006320	012737	006712	001242	MOV	#HHP8,\$TMP4	
006326	012737	006602	001244	MOV	#HHDATO,\$TMP5	
006334	012737	006732	001246	MOV	#HHP10,\$TMP6	
006342	104207			1\$: ERROR	+207	
006344	000137	006752		JMP	HHDONE	
1488 006350				HHER8:		
006350	010537	001252		MOV	R5,\$TMP10	
006354	010437	001254		MOV	R4,\$TMP11	
006360	012737	006712	001240	MOV	#HHP8,\$TMP3	
006366	012737	006712	001242	MOV	#HHP8,\$TMP4	
006374	012737	006602	001244	MOV	#HHDATO,\$TMP5	
006402	012737	006732	001246	MOV	#HHP10,\$TMP6	
006410	104212			1\$: ERROR	+212	
006412	000137	006752		JMP	HHDONE	
1489 006416				HHER9:		
006416	010537	001252		MOV	R5,\$TMP10	
006422	010437	001254		MOV	R4,\$TMP11	
006426	012737	006662	001240	MOV	#HHP5,\$TMP3	
006434	012737	006662	001242	MOV	#HHP5,\$TMP4	
006442	012737	006602	001244	MOV	#HHDATO,\$TMP5	
006450	012737	006742	001246	MOV	#HHP11,\$TMP6	
006456	104207			1\$: ERROR	+207	
006460	000137	006752		JMP	HHDONE	
1490 006464				HHER10:		
006464	010537	001252		MOV	R5,\$TMP10	
006470	010437	001254		MOV	R4,\$TMP11	
006474	012737	006662	001240	MOV	#HHP5,\$TMP3	
006502	012737	006662	001242	MOV	#HHP5,\$TMP4	
006510	012737	006602	001244	MOV	#HHDATO,\$TMP5	
006516	012737	006742	001246	MOV	#HHP11,\$TMP6	
006524	104213			1\$: ERROR	+213	
006526	000137	006752		JMP	HHDONE	
1491 006532	013703	001236		HHTRAP:	MOV \$TMP2,R3	:WAS THE TRAP TO 244
1492 006536	062703	000002		ADD	#2,R3	:ON THE INSTRUCTION
1493 006542	020316			CMP	R3,(SP)	:BEING TESTED?
1494 006544	001402			BEQ	1\$	
1495 006546	000137	037000		JMP	FPSPUR	
1496 006552	011637	001236		1\$: MOV	(SP),\$TMP2	:FAILURE OF FPS INTERRUPT
1497						:DISABLE BIT (FID=1)
1498 006556	022626			CMP	(SP)+,(SP)+	:TO INHIBIT TRAP.
1499 006560	170201			STFPS	R1	
1500 006562	010137	001240		MOV	R1,\$TMP3	
1501 006566	170301			STST	R1	
1502 006570	010137	001242		MOV	R1,\$TMP4	
1503 006574	104217			2\$: ERROR	+217	
1504 006576	000137	006752		JMP	HHDONE	
1505 006602	000000			HHDATO:	0	
1506 006604	000000				0	
1507 006606	000000				0	
1508 006610	000000				0	
1509 006612	000452			HHP0:	452	
1510 006614	125252				125252	
1511 006616	125252				125252	
1512 006620	125253				125253	
1513 006622	000252			HHP1:	252	
1514 006624	125252				125252	

1515 006626 125252
1516 006630 125252
1517 006632 000600
1518 006634 000000
1519 006636 000000
1520 006640 000000
1521 006642 000400
1522 006644 000000
1523 006646 000000
1524 006650 000000
1525 006652 000000
1526 006654 000000
1527 006656 000000
1528 006660 000000
1529 006662 100200
1530 006664 000000
1531 006666 000000
1532 006670 000000
1533 006672 000300
1534 006674 000000
1535 006676 000000
1536 006700 000000
1537 006702 100000
1538 006704 000000
1539 006706 000000
1540 006710 000000
1541 006712 000200
1542 006714 000000
1543 006716 000000
1544 006720 000000
1545 006722 100300
1546 006724 000000
1547 006726 000000
1548 006730 000000
1549 006732 000400
1550 006734 000000
1551 006736 000000
1552 006740 000000
1553 006742 100400
1554 006744 000000
1555 006746 000000
1556 006750 000000
1557 006752 104412
006752 104412

HHP2: 600
0
0
HHP3: 400
0
0
0
HHP4: 0
0
0
0
HHP5: 100200
0
0
0
HHP6: 300
0
0
0
HHP7: 100000
0
0
0
HHP8: 200
0
0
0
HHP9: 100300
0
0
0
HHP10: 400
0
0
0
HHP11: 100400
0
0
0
HHDONE: RSETUP

:HHP0 + HHP1 WITH
:PROPER NORMALIZATION

:HHP0 + HHP1 WITH
:BAD NORMALIZATION

:HHP7 = HHP8 + HHP9
: = HHP5 + HHP6

:HHP10 = HHP8 + HHP8

:HHP11 = HHP5 + HHP5

: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?).

1558
1572

1573

```
.SBTTL TEST # 2 - OVER\UNDER TEST
*****
:TEST 2 OVER\UNDER TEST
:
:THIS IS A PARTIAL TEST OF THE OVER\UNDER
:FLows. ONE OVERFLOW AND TWO UNDERFLOW
:CONDITIONS ARE CHECKED. THE REMAINING
:UNDERFLOW COND. AND THE REMAINING OVERFLOW
:COND. WILL BE CHECKED LATER USING THE
:XXX INSTRUCTION. HERE EACH CONDITION TESTED
:IS CHECKED BOTH WITH TRAPS ENABLED
:*(FIU=1 OR FIV=1) AND ALSO WITH TRAPS
:DISABLED (FIU=0 OR FIV=0).
:
*****
```

```
006754 000004
1574
1575
1576 006756
      C06756 104413
1577 006760 012704 000200
1578 006764 170104
1579 006766 012737 007014 001236
1580 006774 012737 010046 000244
1581 007002 012700 011620
1582 007006 172410
1583 007010 012700 011620
1584 007014 172010
1585 007016 170205
1586 007020 012700 011550
1587 007024 174010
1588 007026 012701 011630
1589 007032 012702 000004
1590 007036 022021
1591 007040 001402
1592 007042 000137 010144
1593 007046 077205
1594 007050 052704 000006
1595 007054 020405
1596 007056 001402
1597 007060 000137 010212
1598
1599
1600 007064
      007064 104413
1601 007066 012704 001200
1602 007072 170104
1603 007074 012737 007122 001236
1604 007102 012737 007140 000244
1605 007110 012700 011620
1606 007114 172410
1607 007116 012700 011620
1608 007122 172010
1609 007124 170000
1610 007126 012700 011550
1611 007132 174010
1612 007134 000137 010260
```

```
TST2: SCOPE
:TEST OVERFLOW CONDITION WITH TRAP DISABLER FIV=0
GG1:
      LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
      MOV #200,R4 ;CLEAR FIU, FIV, AND SET FD
      LDFPS R4
      MOV #GG2,$TMP2
      MOV #GGER0,FPVECT
      MOV #GGP5,R0 ;SET ACO OPERAND
      LDD (R0),ACO
      MOV #GGP5,R0 ;FSRC
      GG2: ADDD (R0),ACO ;TEST INSTRUCTION
      STFPS R5 ;GET FPS
      MOV #GGDAT0,R0 ;GET THE RESULT
      STD ACO,(R0)
      MOV #GGP6,R1 ;IS IT CORRECT
      MOV #4,R2
      GG3: CMP (R0)+,(R1)+
      BEQ GG4
      JMP GGER1
      GG4: SOB R2,GG3
      BIS #6,R4 ;FPS CORRECT?
      CMP R4,R5
      BEQ GG5
      JMP GGER2
      ;TEST OVERFLOW WITH TRAPS ENABLED
      ;FIV = 1
      GG5: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
      MOV #1200,R4 ;CLEAR FIU, SET FIV, AND FD
      LDFPS R4
      MOV #GG6,$TMP2
      MOV #GG7,FPVECT
      MOV #GGP5,R0 ;SET ACO OPERAND
      LDD (R0),ACO
      MOV #GGP5,R0 ;FSRC
      GG6: ADDD (R0),ACO ;TEST INSTRUCTION
      CFCC
      MOV #GGDAT0,R0
      STD ACO,(R0)
      JMP GGER3
```


1613	007140	013703	001236		GG7:	MOV	\$TMP2,R3	
1614	007144	062703	000002			ADD	#2,R3	
1615	007150	020316				CMP	R3,(SP)	
1616	007152	001402				BEQ	1\$	
1617	007154	000137	037000			JMP	FPSPUR	
1618	007160	011637	001236		1\$:	MOV	(SP),\$TMP2	
1619	007164	022626				CMP	(SP)+,(SP)+	
1620	007166	170205				STFPS	R5	
1621	007170	012700	011550			MOV	#GGDATO,R0	;GET THE RESULT
1622	007174	174010				STD	ACO,(R0)	
1623	007176	012701	011630			MOV	#GGP6,R1	;IS IT CORRECT
1624	007202	012702	000004			MOV	#4,R2	
1625	007206	022021			GG8:	CMP	(R0)+,(R1)+	
1626	007210	001402				BEQ	GG9	
1627	007212	000137	010326			JMP	GGER4	
1628	007216	077205			GG9:	SOB	R2,GG8	
1629	007220	052704	100006			BIS	#100006,R4	
1630	007224	020405				CMP	R4,R5	;FPS CORRECT?
1631	007226	001402				BEQ	1\$	
1632	007230	000137	010376			JMP	GGER6	
1633	007234	012704	000010		1\$:	MOV	#10,R4	
1634					;CHECK	FEC		
1635	007240	170305				STST	R5	
1636	007242	020405				CMP	R4,R5	
1637	007244	001402				BEQ	GG10	
1638	007246	000137	010330			JMP	GGER5	
1639					;CHECK	UNDER FLOW CONDITION WITH		
1640					;TRAPS	DISABLED (FIU = 0)		
1641	007252				GG10:	LPERR		;SET UP THE LOOP ON ERROR ADDRESS.
	007252	104413				MOV	#0200,R4	;SET FIU, FIV, AND FD
1642	007254	012704	000200			LDFPS	R4	
1643	007260	170104				MOV	#GG11,\$TMP2	
1644	007262	012737	007310	001236		MOV	#GGER7,FPVECT	
1645	007270	012737	010444	000244		MOV	#GGP2,R0	;SET ACO OPERAND
1646	007276	012700	011570			LDD	(R0),ACO	;FSRC
1647	007302	172410				MOV	#GGP3,R0	
1648	007304	012700	011600		GG11:	ADDD	(R0),ACO	;TEST INSTRUCTION
1649	007310	172010				STFPS	R5	;GET FPS
1650	007312	170205				MOV	#GGDATO,R0	;GET THE RESULT
1651	007314	012700	011550			STD	ACO,(R0)	
1652	007320	174010				MOV	#GGP6,R1	;IS IT CORRECT
1653	007322	012701	011630			MOV	#4,R2	
1654	007326	012702	000004		GGi2:	CMP	(R0)+,(R1)+	
1655	007332	022021				BEQ	GG13	
1656	007334	001402				JMP	GGER8	
1657	007336	000137	010542		GG13:	SOB	R2,GG12	
1658	007342	077205				BIS	#4,R4	;FPS CORRECT?
1659	007344	052704	000004			CMP	R4,R5	
1660	007350	020405				BEQ	GG14	
1661	007352	001402				JMP	GGER9	
1662	007354	000137	010610					
1663					;CHECK	UNDERFLOW CONDITION WITH		
1664					;TRAP	ENABLED (FIU = 1)		
1665	007360				GG14:	LPERR		;SET UP THE LOOP ON ERROR ADDRESS.
	007360	104413				MOV	#2200,R4	;SET FIU, FIV, AND FD
1666	007362	012704	002200			LDFPS	R4	
1667	007366	170104						


```

1668 007370 012737 007416 001236      MOV      #GG15,$TMP2
1669 007376 012737 007434 000244      MOV      #GG16,FPVECT
1670 007404 012700 011570      MOV      #GGP2,R0          ;SET ACO OPERAND
1671 007410 172410      LDD      (R0),ACO          ;FSPC
1672 007412 012700 011600      MOV      #GGP3,R0
1673 007416 172010      GG15:   ADDD     (R0),ACO          ;TEST INSTRUCTION
1674 007420 170000      CFCC
1675 007422 012700 011550      MOV      #GGDATO,R0
1676 007426 174010      STD      ACO,(R0)
1677 007430 000137 010656      JMP      GGER10
1678 007434 013703 001236      GG16:   MOV      $TMP2,R3
1679 007440 062703 000002      ADD      #2,R3
1680 007444 021603      CMP      (SP),R3
1681 007446 001402      BEQ      1$
1682 007450 000137 037000      JMP      FPSPUR
1683 007454 011637 001236      1$:     MOV      (SP),$TMP2
1684 007460 022626      CMP      (SP)+,(SP)+
1685 007462 170205      STFPS   R5          ;GET FPS
1686 007464 012700 011550      MOV      #GGDATO,R0          ;GET THE RESULT
1687 007470 174010      STD      ACO,(R0)
1688 007472 012701 011640      MOV      #GGP7,R1          ;IS IT CORRECT
1689 007476 012702 000004      MOV      #4,R2
1690 007502 022021      GG17:   CMP      (R0)+,(R1)+
1691 007504 001402      BEQ      GG18
1692 007506 000137 010724      JMP      GGER11
1693 007512 077205      GG18:   SOB      R2,GG17
1694 007514 052704 100000      BIS      #100000,R4
1695 007520 020405      CMP      R4,R5          ;FPS CORRECT?
1696 007522 001402      BEQ      2$
1697 007524 000137 010772      JMP      GGER12
1698 007530      2$:
1699 007530 012704 000012      1$:     MOV      #12,R4
1700      ;CHECK FEC
1701 007534 170305      STST   R5
1702 007536 020405      CMP      R4,R5
1703 007540 001402      BEQ      GG19
1704 007542 000177 001272      JMP      @GGER13
1705      ;CHECK UNDERFLOW CONDITION WITH TRAPS
1706      ;DISABLED (FIU = 0)
1707      GG19:
1708 007546 104413      LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
1709 007550 012704 000200      MOV      #0200,R4          ;SET FIU, FIV, AND FD
1710 007554 170104      LDFPS   R4
1711 007556 012737 007604 001236      MOV      #GG20,$TMP2
1712 007564 012737 011106 000244      MOV      #GGER14,FPVECT
1713 007572 012700 011570      MOV      #GGP2,R0          ;SET ACO OPERAND
1714 007600 012700 011650      LDD      (R0),ACO          ;FSPC
1715 007604 172010      GG20:   MOV      #GGP8,R0          ;TEST INSTRUCTION
1716 007606 170205      ADDD     (R0),ACO          ;GET FPS
1717 007610 012700 011550      STFPS   R5          ;GET THE RESULT
1718 007614 174010      MOV      #GGDATO,R0
1719 007616 012701 011630      STD      ACO,(R0)
1720 007622 012702 000004      MOV      #GGP6,R1          ;IS IT CORRECT
1721 007626 022021      GG21:   MOV      #4,R2
1722 007630 001402      CMP      (R0)+,(R1)+
1723 007632 000137 011204      BEQ      GG22
1723 007632 000137 011204      JMP      GGER15

```


1724 007636 077205
1725 007640 052704 000004
1726 007644 020405
1727 007646 001402
1728 007650 000137 011252

GG22: SOB R2,GG21
BIS #4,R4
CMP R4,R5
BEQ GG23
JMP GGER16

;FPS CORRECT?

CKFPBBO FP11F FLTG PNT PRT B
TEST # 2 - OVER\UNDER TEST

MACRO M1113 28-MAR-81 14:47 PAGE 11

M 3

SEQUENCE 38

1730

;CHECK UNDERFLOW CONDITION WITH TRAP


```

1732                                     ;ENABLED (FIU = 1)
1733 007654                               GG23: LPERR                               ;SET UP THE LOOP ON ERROR ADDRESS.
      007654 104413                       MOV #2200,R4                               ;SET FIU, FIV, AND FD
1734 007656 012704 002200                 LDFPS R4
1735 007662 170104                       MOV #GG24,$TMP2
1736 007664 012737 007712 001236         MOV #GG25,FPVECT
1737 007672 012737 007730 000244         MOV #GGP2,R0                               ;SET ACO OPERAND
1738 007700 012700 011570                 LDD (R0),ACO
1739 007704 172410                       MOV #GGP8,R0                               ;FSRC
1740 007706 012700 011650                 GG24: ADDD (R0),ACO                       ;TEST INSTRUCTION
1741 007712 172010                       CFCC
1742 007714 170000                       MOV #GGDATO,R0
1743 007716 012700 011550                 STD ACO,(R0)
1744 007722 174010                       JMP GGER17
1745 007724 000137 011320                 GG25: MOV $TMP2,R0
1746 007730 013700 001236                 ADD #2,R0
1747 007734 062700 000002                 CMP R0,(SP)
1748 007740 020016                       BEQ 1$
1749 007742 001402                       JMP FPSPUR
1750 C07744 000137 037000                 1$: MOV (SP),$TMP2
1751 007750 011637 001236                 CMP (SP)+,(SP)+
1752 007754 022626                       STFPS R5                               ;GET FPS
1753 007756 170205                       MOV #GGDATO,R0                             ;GET THE RESULT
1754 007760 012700 011550                 STD ACO,(R0)
1755 007764 174010                       MOV #GGP9,R1                               ;IS IT CORRECT
1756 007766 012701 011660                 MOV #4,R2
1757 007772 012702 000004                 GG26: CMP (R0)+,(R1)+
1758 007776 022021                       BEQ GG27
1759 010000 001402                       JMP GGER18
1760 010002 000137 011366                 GG27: SOB R2,GG26
1761 010006 077205                       BIS #100004,R4
1762 010010 052704 100004                 CMP R4,R5                               ;FPS CORRECT?
1763 010014 020405                       BEQ 1$
1764 010016 001402                       JMP GGER20
1765 010020 000137 011502                 1$: MOV #12,R4
1766 010024 012704 000012                 ;CHECK FEC
1767                                     STST R5
1768 010030 170305                       CMP R4,R5
1769 010032 020405                       BEQ GG28
1770 010034 001402                       JMP GGER19
1771 010036 000137 011434                 GG28: JMP GGDONE
1772                                     GG28:
1773 010042 000137 011670                 GG28: JMP GGDONE
1774                                     GG28:
1775 010046 013701 001236                 GGER0: MOV $TMP2,R1
1776 010052 062701 000002                 ADD #2,R1
1777 010056 020116                       CMP R1,(SP)
1778 010060 001402                       BEQ 10$
1779 010062 000137 037000                 5$: JMP FPSPUR
1780 010066                                     10$:
1781 010066 170301                       STST R1
1782 010070 020127 000010                 CMP R1,#10
1783 010074 001372                       BNE 5$
1784 010076 022626                       CMP (SP)+,(SP)+
1785 010100 012700 011550                 MOV #GGDATO,R0
1786 010104 174010                       STD ACO,(R0)
1787 010106 012737 011620 001240         MOV #GGP5,$TMP3
  
```

	010114	012737	011620	001242		MOV	#GGP5,\$TMP4
	010122	012737	011550	001244		MOV	#GGDATO,\$TMP5
	010130	012737	011630	001246		MOV	#GGP6,\$TMP6
	010136	104220			1\$:	ERROR	+220
	010140	000137	011670			JMP	GGDONE
1788							
1789	010144				GGER1:		
	010144	010537	001252			MOV	R5,\$TMP10
	010150	010437	001254			MOV	R4,\$TMP11
	010154	012737	011620	001240		MOV	#GGP5,\$TMP3
	010162	012737	011620	001242		MOV	#GGP5,\$TMP4
	010170	012737	011550	001244		MOV	#GGDATO,\$TMP5
	010176	012737	011630	001246		MOV	#GGP6,\$TMP6
	010204	104207			1\$:	ERROR	+207
	010206	000137	011670			JMP	GGDONE
1790							
1791	010212				GGER2:		
	010212	010537	001252			MOV	R5,\$TMP10
	010216	010437	001254			MOV	R4,\$TMP11
	010222	012737	011620	001240		MOV	#GGP5,\$TMP3
	010230	012737	011620	001242		MOV	#GGP5,\$TMP4
	010236	012737	011550	001244		MOV	#GGDATO,\$TMP5
	010244	012737	011630	001246		MOV	#GGP6,\$TMP6
	010252	104232			1\$:	ERROR	+232
	010254	000137	011670			JMP	GGDONE
1792							
1793	010260				GGER3:		
	010260	010537	001252			MOV	R5,\$TMP10
	010264	010437	001254			MOV	R4,\$TMP11
	010270	012737	011620	001240		MOV	#GGP5,\$TMP3
	010276	012737	011620	001242		MOV	#GGP5,\$TMP4
	010304	012737	011550	001244		MOV	#GGDATO,\$TMP5
	010312	012737	011630	001246		MOV	#GGP6,\$TMP6
	010320	104221			1\$:	ERROR	+221
	010322	000137	011670			JMP	GGDONE
1794							
1795	010326	000706			GGER4:	BR	GGER1
1796							
1797	010330				GGER5:		
	010330	010537	001252			MOV	R5,\$TMP10
	010334	010437	001254			MOV	R4,\$TMP11
	010340	012737	011620	001240		MOV	#GGP5,\$TMP3
	010346	012737	011620	001242		MOV	#GGP5,\$TMP4
	010354	012737	011550	001244		MOV	#GGDATO,\$TMP5
	010362	012737	011630	001246		MOV	#GGP6,\$TMP6
	010370	104226			1\$:	ERROR	+226
	010372	000137	011670			JMP	GGDONE
1798							
1799	010376				GGER6:		
	010376	010537	001252			MOV	R5,\$TMP10
	010402	010437	001254			MOV	R4,\$TMP11
	010406	012737	011620	001240		MOV	#GGP5,\$TMP3
	010414	012737	011620	001242		MOV	#GGP5,\$TMP4
	010422	012737	011550	001244		MOV	#GGDATO,\$TMP5
	010430	012737	011630	001246		MOV	#GGP6,\$TMP6
	010436	104227			1\$:	ERROR	+227
	010440	000137	011670			JMP	GGDONE

1800						
1801	010444	013701	001236		GGER7:	MOV \$TMP2,R1
1802	010450	062701	000002			ADD #2,R1
1803	010454	020116				CMP R1,(SP)
1804	010456	001402				BEQ 10\$
1805	010460	000137	037000		5\$:	JMP FPSPUR
1806	010464				10\$:	
1807	010464	170301				STST R1
1808	010466	020127	000012			CMP R1,#12
1809	010472	001372				BNE 5\$
1810	010474	022626				CMP (SP)+,(SP)+
1811	010476	012700	011550			MOV #GGDATO,RO
1812	010502	174010				STD ACO,(RO)
1813	010504	012737	011600	001240		MOV #GGP3,\$TMP3
	010512	012737	011570	001242		MOV #GGP2,\$TMP4
	010520	012737	011550	001244		MOV #GGDATO,\$TMP5
	010526	012737	011630	001246		MOV #GGP6,\$TMP6
	010534	104224			1\$:	ERROR +224
	010536	000137	011670			JMP GGDONE
1814						
1815	010542				GGER8:	
	010542	010537	001252			MOV R5,\$TMP10
	010546	010437	001254			MOV R4,\$TMP11
	010552	012737	011600	001240		MOV #GGP3,\$TMP3
	010560	012737	011570	001242		MOV #GGP2,\$TMP4
	010566	012737	011550	001244		MOV #GGDATO,\$TMP5
	010574	012737	011630	001246		MOV #GGP6,\$TMP6
	010602	104207			1\$:	ERROR +207
	010604	000137	011670			JMP GGDONE
1816						
1817	010610				GGER9:	
	010610	010537	001252			MOV R5,\$TMP10
	010614	010437	001254			MOV R4,\$TMP11
	010620	012737	011600	001240		MOV #GGP3,\$TMP3
	010626	012737	011570	001242		MOV #GGP2,\$TMP4
	010634	012737	011550	001244		MOV #GGDATO,\$TMP5
	010642	012737	011630	001246		MOV #GGP6,\$TMP6
	010650	104232			1\$:	ERROR +232
	010652	000137	011670			JMP GGDONE
1818						
1819	010656				GGER10:	
	010656	010537	001252			MOV R5,\$TMP10
	010662	010437	001254			MOV R4,\$TMP11
	010666	012737	011600	001240		MOV #GGP3,\$TMP3
	010674	012737	011570	001242		MOV #GGP2,\$TMP4
	010702	012737	011550	001244		MOV #GGDATO,\$TMP5
	010710	012737	011640	001246		MOV #GGP7,\$TMP6
	010716	104225			1\$:	ERROR +225
	010720	000137	011670			JMP GGDONE
1820						
1821	010724				GGER11:	
	010724	010537	001252			MOV R5,\$TMP10
	010730	010437	001254			MOV R4,\$TMP11
	010734	012737	011600	001240		MOV #GGP3,\$TMP3
	010742	012737	011570	001242		MOV #GGP2,\$TMP4
	010750	012737	011550	001244		MOV #GGDATO,\$TMP5
	010756	012737	011640	001246		MOV #GGP7,\$TMP6

	010764	104207		1\$:	ERROR	+207
	010766	000137	011670		JMP	GGDONE
1822						
1823	010772			GGER12:	MOV	R5,\$TMP10
	010772	010537	001252		MOV	R4,\$TMP11
	010776	010437	001254		MOV	#GGP3,\$TMP3
	011002	012737	011600	001240	MOV	#GGP2,\$TMP4
	011010	012737	011570	001242	MOV	#GGDATO,\$TMP5
	011016	012737	011550	001244	MOV	#GGP7,\$TMP6
	011024	012737	011640	001246		
	011032	104231		1\$:	ERROR	+231
	011034	000137	011670		JMP	GGDONE
1824						
1825	011040			GGER13:	MOV	R5,\$TMP10
	011040	010537	001252		MOV	R4,\$TMP11
	011044	010437	001254		MOV	#GGP3,\$TMP3
	011050	012737	011600	001240	MOV	#GGP2,\$TMP4
	011056	012737	011570	001242	MOV	#GGDATO,\$TMP5
	011064	012737	011550	001244	MOV	#GGP7,\$TMP6
	C11072	012737	011640	001246		
	011100	104230		1\$:	ERROR	+230
	011102	000137	011670		JMP	GGDONE
1826						
1827	011106	013701	001236	GGER14:	MOV	\$TMP2,R1
1828	011112	062701	000002		ADD	#2,R1
1829	011116	020116			CMP	R1,(SP)
1830	011120	001402			BEQ	10\$
1831	011122	000137	037000	5\$:	JMP	FPSPUR
1832	011126			10\$:		
1833	011126	170301			STST	R1
1834	011130	020127	000012		CMP	R1,#12
1835	011134	001372			BNE	5\$
1836	011136	022626			CMP	(SP)+,(SP)+
1837	011140	012700	011550		MOV	#GGDATO,R0
1838	011144	174010			STD	ACO,(R0)
1839						
1840	011146	012737	011560	001240	MOV	#GGP1,\$TMP3
	011154	012737	011600	001242	MOV	#GGP3,\$TMP4
	011162	012737	011550	001244	MOV	#GGDATO,\$TMP5
	011170	012737	011630	001246	MOV	#GGP6,\$TMP6
	011176	104222		1\$:	ERROR	+222
	011200	000137	011670		JMP	GGDONE
1841						
1842	011204			GGER15:	MOV	R5,\$TMP10
	011204	010537	001252		MOV	R4,\$TMP11
	011210	010437	001254		MOV	#GGP2,\$TMP3
	011214	012737	011570	001240	MOV	#GGP8,\$TMP4
	011222	012737	011650	001242	MOV	#GGDATO,\$TMP5
	011230	012737	011550	001244	MOV	#GGP6,\$TMP6
	011236	012737	011630	001246		
	011244	104207		1\$:	ERROR	+207
	011246	000137	011670		JMP	GGDONE
1843						
1844	011252			GGER16:	MOV	R5,\$TMP10
	011252	010537	001252		MOV	R4,\$TMP11
	011256	010437	001254		MOV	#GGP2,\$TMP3
	011262	012737	011570	001240		

	011270	012737	011650	001242		MOV	#GGP8,\$TMP4
	011276	012737	011550	001244		MOV	#GGDATO,\$TMP5
	011304	012737	011630	001246		MOV	#GGP6,\$TMP6
	011312	104232			1\$:	ERROR	+232
	011314	000137	011670			JMP	GGDONE
1845							
1846	011320				GGER17:		
	011320	010537	001252			MOV	R5,\$TMP10
	011324	010437	001254			MOV	R4,\$TMP11
	011330	012737	011570	001240		MOV	#GGP2,\$TMP3
	011336	012737	011650	001242		MOV	#GGP8,\$TMP4
	011344	012737	011550	001244		MOV	#GGDATO,\$TMP5
	011352	012737	011660	001246		MOV	#GGP9,\$TMP6
	011360	104223			1\$:	ERROR	+223
	011362	000137	011670			JMP	GGDONE
1847							
1848	011366				GGER18:		
	011366	010537	001252			MOV	R5,\$TMP10
	011372	010437	001254			MOV	R4,\$TMP11
	011376	012737	011570	001240		MOV	#GGP2,\$TMP3
	011404	012737	011650	001242		MOV	#GGP8,\$TMP4
	011412	012737	011550	001244		MOV	#GGDATO,\$TMP5
	011420	012737	011660	001246		MOV	#GGP9,\$TMP6
	011426	104207			1\$:	ERROR	+207
	011430	000137	011670			JMP	GGDONE
1849							
1850	011434				GGER19:		
	011434	010537	001252			MOV	R5,\$TMP10
	011440	010437	001254			MOV	R4,\$TMP11
	011444	012737	011570	001240		MOV	#GGP2,\$TMP3
	011452	012737	011650	001242		MOV	#GGP8,\$TMP4
	011460	012737	011550	001244		MOV	#GGDATO,\$TMP5
	011466	012737	011660	001246		MOV	#GGP9,\$TMP6
	011474	104230			1\$:	ERROR	+230
	011476	000137	011670			JMP	GGDONE
1851							
1852	011502				GGER20:		
	011502	010537	001252			MOV	R5,\$TMP10
	011506	010437	001254			MOV	R4,\$TMP11
	011512	012737	011570	001240		MOV	#GGP2,\$TMP3
	011520	012737	011650	001242		MOV	#GGP8,\$TMP4
	011526	012737	011550	001244		MOV	#GGDATO,\$TMP5
	011534	012737	011660	001246		MOV	#GGP9,\$TMP6
	011542	104231			1\$:	ERROR	+231
	011544	000137	011670			JMP	GGDONE
1853							
1854	011550	000000			GGDATO:	0	
1855	011552	000000				0	
1856	011554	000000				0	
1857	011556	000000				0	
1858							
1859	011560	000300			GGP1:	300	
1860	011562	000000				0	
1861	011564	000000				0	
1862	011566	000000				0	
1863	011570	100200			GGP2:	100200	
1864	011572	000000				0	

1865 011574 000000
1866 011576 000000
1867 011600 000200
1868 011602 000000
1869 011604 000000
1870 011606 000001
1871 011610 010200
1872 011612 000000
1873 011614 000000
1874 011616 000000
1875 011620 077600
1876 011622 000000
1877 011624 000000
1878 011626 000000
1879 011630 000000
1880 011632 000000
1881 011634 000000
1882 011636 000000
1883
1884 011640 062400
1885 011642 000000
1886 011644 000000
1887 011646 000000
1888 011650 000340
1889 011652 000000
1890 011654 000000
1891 011656 000000
1892 011660 000100
1893 011662 000000
1894 011664 000000
1895 011666 000000
1896 011670
011670 104412

0
0
GGP3: 200
0
0
GGP4: 10200
0
0
GGP5: 77600
0
0
GGP6: 0
0
0
GGP7: 62400
0
0
GGP8: 340
0
0
GGP9: 100
0
0
GGDONE: RSETUP

:OVER FLOW = GGP5 + GGP5

:OVERFLOW RESULT
:UNDERFLOW RESULT
:GGP6 = GGP4 + GGP5
: = GGP3 + GGP2 (FIU = 0)
: = GGP3 + GGP1
:GGP7 = GGP3 + GGP2 (FIU = 1)

: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?).

1897
1903

1904
 1905 011672 000004
 1906 011674
 1907 011674 104413
 1908 011676 012704 000200
 1909 011702 170104
 1910 011704 012700 013310
 1911 011710 172410
 1912 011712 012700 013320
 1913 011716 012737 011724 001236
 1914 011724 177420
 1915 011726 020027 013324
 1916 011732 001402
 1917 011734 000137 012666
 1918 011740
 1919 011740 170205
 1920 011742 012700 013300
 1921 011746 174010
 1922 011750 012701 013370
 1923 011754 012702 000004
 1924 011760 022120
 1925 011762 001415
 1926 011764 012701 013320
 1927 011770 012700 013300
 1928 011774 012702 000004
 1929 012000 022120
 1930 012002 001402
 1931 012004 000137 012726
 1932 012010 077205
 1933 012012 000137 012756
 1934 012016 077220
 1935 012020 012704 000200
 1936 012024 020405
 1937 012026 001402
 1938 012030 000137 013024
 1939 012034
 1940 012034 104413
 1941 012036 012704 000200
 1942 012042 170104
 1943 012044 012700 013310
 1944 012050 172410
 1945
 1946 012052 012700 013320
 1947 012056 012737 012066 001236
 1948
 1949 012064 170001
 1950
 1951 012066 177420

```

.SBTTL TEST # 3 - LDCFD AND LDCDF TEST
*****
*TEST 3          LDCFD AND LDCDF TEST
*
*THIS IS A TEST OF LDCFD AND LDCDF.
*
*****
TST3:  SCOPE
:TEST FOR CORRECT AUTO INCREMENT CONSTANT.
HX1:
      LPERR                                ;SET UP THE LOOP ON ERROR ADDRESS.
      MOV #200,R4
      LDFPS R4
      MOV #HXP1,R0
      LDD (R0),ACO
      MOV #HXP2,R0
      MOV #HX2,$TMP2
      LDCFD (R0)+,ACO
      CMP R0,#HXP2+4                       ;IS R0 CORRECT
      BEQ HX3
      JMP HXER1
HX2:
HX3:
      STFPS R5                               ;GET FPS
      MOV #HXDATO,R0
      STD ACO,(R0)                          ;GET ACO
      MOV #HXP7,R1                          ;SEE IF RESULT IS
      MOV #4,R2                              ;CORRECT
      CMP (R1)+,(R0)+
      BEQ HX7
      MOV #HXP2,R1                          ;DID FD GET
      MOV #HXDATO,R0                        ;COMPLIMENTED?
      MOV #4,R2
      CMP (R1)+,(R0)+
      BEQ HX6
      JMP HXER2
HX4:
HX5:
      SOB R2,HX5
      JMP HXER3
HX6:
HX7:
      SOB R2,HX4
      MOV #200,R4
      CMP R4,R5
      BEQ HX8
      JMP HXER8
      TEST LDCDF
      ;NOW
HX8:
      LPERR                                ;SET UP THE LOOP ON ERROR ADDRESS.
      MOV #200,R4
      LDFPS R4
      MOV #HXP1,R0
      LDD (R0),ACO
      MOV #HXP2,R0
      MOV #HX9,$TMP2
      SETF
HX9:
      LDCFD (R0)+,ACO                       ;TEST INSTRUCTION
  
```

1952									
1953	012070	020027	013330		CMP	R0,#HXP2+10		:WAS A GOOD	
1954	012074	001402			BEQ	HX10		:CONSTANT USED	
1955	012076	000137	012706		JMP	HXER5		:TO INCREMENT R0?	
1956									
1957	012102			HX10:	STFPS	R5			
1958	012102	170205			MOV	#HXDATO,R0			
1959	012104	012700	013300		SETD				
1960	012110	170011			STD	AC0,(R0)		:GET RESULT	
1961	012112	174010			MOV	#HXP8,R1			
1962	012114	012701	013400		MOV	#4,R2			
1963	012120	012702	000004	HX11:	CMP	(R1)+,(R0)+		:IS IT CORRECT?	
1964	012124	022120			BEQ	HX14			
1965	012126	001415							
1966									
1967	012130	012701	013370		MOV	#HXP7,R1			
1968	012134	012700	013300		MOV	#HXDATO,R0			
1969	012140	012702	000004		MOV	#4,R2			
1970	012144	022110		HX12:	CMP	(R1)+,(R0)		:DID FD FAIL TO GET	
1971	012146	001402			BEQ	HX13		:COMPLIMENTED?	
1972	012150	000137	013042		JMP	HXER6			
1973	012154	077205		HX13:	SOB	R2,HX12			
1974	012156	000137	013072		JMP	HXER7			
1975									
1976	012162	077220		HX14:	SOR	R2,HX11			
1977									
1978	012164	012704	000000		MOV	#0,R4		:FPS CORRECT?	
1979	012170	020405			CMP	R4,R5			
1980	012172	001402			BEQ	HX15			
1981	012174	000137	013024		JMP	HXER8			
1982									
1983									
1984									
1985	012200								
	012200	104413		HX15:	LPERR			:SET UP THE LOOP ON ERROR ADDRESS.	
1986									
1987	012202	012704	000200		MOV	#200,R4			
1988	012206	170104			LDFPS	R4		:SET FD	
1989	012210	012737	012226	001236	MOV	#HX16,\$TMP2			
1990	012216	012737	013122	000004	MOV	#HXER9,ERRVECT			
1991	012224	005001			CLR	R1			
1992	012226	177427	043243	HX16:	LDCFD	#5201,AC0			
1993	012232	005201		HX165:	INC	R1			
1994	012234	005201			INC	R1			
1995	012236	005201			INC	R1			
1996	012240	012737	037032	000004	MOV	#CPSPUR,ERRVECT			
1997	012246	020127	000003		CMP	R1,#3		:SEE IF PC WAS	
1998	012252	001402			BEQ	HX17		:CORRECT	
1999	012254	000137	013156		JMP	HXER10			
2000									
2001	012260								
	012260	104413		HX17:	LPERR			:SET UP THE LOOP ON ERROR ADDRESS.	
2002									
2003	012262	012704	000200		MOV	#200,R4			
2004	012266	170104			LDFPS	R4			
2005	012270	012737	012316	001236	MOV	#HX18,\$TMP2			
2006	012276	012700	013360		MOV	#HXP6,R0			


```

2007 012302 172410
2008 012304 012737 037032 000004
2009 012312 012700 013320
2010 012316 177410 HX18: LDCFD (R0),ACO
2011
2012 012320 012700 013300 MOV #HXDATO,R0
2013 012324 174010 STD ACO,(R0) ;GET RESULT.
2014 012326 012701 013370 MOV #HXP7,R1
2015 012332 012702 000004 MOV #4,R2
2016 012336 022021 HX19: CMP (R0)+,(R1)+ ;IS RESULT CORRECT?
2017 012340 001402 BEQ HX20
2018 012342 000137 012726 JMP HXER2
2019 012346 077205 HX20: SOB R2,HX19
2020
2021 ;TEST LDCFD WITH NEGATIVE OPERAND
2022 HX21: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
      MOV #200,R4
      LDFPS R4
2023 012352 012704 000200 MOV #HX22,$TMP2
2024 012356 170104 MOV #HXP6,R0
2025 012360 012737 012400 001236 MOV #HXP4,R0
2026 012366 012700 013360 LDD (R0),ACO
2027 012372 172410 HX22: LDCFD (R0),ACO
2028 012374 012700 013340
2029 012400 177410
2030
2031 012402 012700 013300 MOV #HXDATO,R0
2032 012406 174010 STD ACO,(R0) ;GET RESULT
2033
2034 012410 012701 013350 MOV #HXP5,R1
2035 012414 012702 000004 MOV #4,R2
2036 012420 022120 HX23: CMP (R1)+,(R0)+
2037 012422 001415 BEQ HX26
2038
2039 012424 012701 013370 MOV #HXP7,R1
2040 012430 012700 013300 MOV #HXDATO,R0
2041 012434 012702 000004 MOV #4,R2
2042 012440 022120 HX24: CMP (R1)+,(R0)+ ;WAS SIGN INCORRECT
2043 012442 001402 BEQ HX25
2044 012444 000137 013210 JMP HXER11
2045 012450 077205 HX25: SOB R2,HX24
2046 012452 000137 013230 JMP HXER12
2047
2048 012456 077220 HX26: SOB R2,HX23
2049
2050 ;TEST LDCFD 0
2051
2052 HX27: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
      MOV #200,R4
      LDFPS R4
2053 012462 012704 000200 MOV #HXP1,R0
2054 012466 170104 LDD (R0),ACO
2055
2056 012470 012700 013310 ADDD (R0),ACO
2057 012474 172410
2058 012476 172010
2059
2060 012500 012737 012512 001236 MOV #HX28,$TMP2
2061 012506 012700 013310 MOV #HXP1,R0
    
```

```

2062 012512 177410          HX28:  LDCFD  (R0),ACO
2063
2064 012514 170205          STFPS  R5
2065
2066 012516 012700 013300    MOV    #HXDATO,R0
2067 012522 174010          STD    ACO,(R0)          ;GET RESULT
2068
2069 012524 012701 013310    MOV    #HXP1,R1
2070 012530 012702 000004    MOV    #4,R2
2071 012534 022120          HX29:  CMP    (R1)+,(R0)+  ;IS IT 0?
2072 012536 001402          BEQ    HX30
2073 012540 000137 013260    JMP    HXER13
2074 012544 077205          HX30:  SOB    R2,HX29
2075
2076 012546 012704 000204    MOV    #204,R4          ;FPS CORRECT
2077 012552 020405          CMP    R4,R5
2078 012554 001402          BEQ    HX31
2079 012556 000137 013006    JMP    HXER4
2080
2081          ;TEST  LDCFD  0
2082
2083 012562          HX31:
      012562 104413          LPERR
2084 012564 012704 000200    MOV    #200,R4          ;SET UP THE LOOP ON ERROR ADDRESS.
2085 012570 170104          LDFPS  R4
2086
2087 012572 012700 013360    MOV    #HXP6,R0
2088 012576 172410          LDD    (R0),ACO
2089
2090 012600 012737 012612 001236    MOV    #HX32,$TMP2
2091 012606 012700 013310    MOV    #HXP1,R0
2092 012612 177410          HX32:  LDCFD  (R0),ACO
2093
2094 012614 170205          STFPS  R5
2095
2096 012616 012700 013300    MOV    #HXDATO,R0
2097 012622 174010          STD    ACO,(R0)          ;GET RESULT
2098
2099 012624 012701 013310    MOV    #HXP1,R1
2100 012630 012702 000004    MOV    #4,R2
2101 012634 022120          HX33:  CMP    (R1)+,(R0)+  ;IS IT ZERO?
2102 012636 001402          BEQ    HX34
2103 012640 000137 013260    JMP    HXER13
2104 012644 077205          HX34:  SOB    R2,HX33
2105
2106 012646 012704 000204    MOV    #204,R4          ;FPS CORRECT?
2107 012652 020405          CMP    R4,R5
2108 012654 001402          BEQ    HX35
2109 012656 000137 013006    JMP    HXER4
2110 012662 000137 013410    HX35:  JMP    HXDONE
2111
2112          ;RO INCORRECT
2113
2114 012666 012737 013324 001242    HXER1: MOV    #HXP2+4,$TMP4
2115 012674 010037 001240    MOV    R0,$TMP3
2116 012700 104234          1$:   ERROR  +234
2117 012702 000137 013410    JMP    HXDONE
  
```



```

2118
2119 012706 012737 013330 001242 HXER5: MOV #HXP2+10,$TMP4
2120 012714 010037 001240 MOV R0,$TMP3
2121 012720 104237 1$: ERROR +237
2122 012722 000137 013410 JMP HXDONE
2123 ;REPORT BAD DATA
2124 012726 012737 013320 001244 HXER2: MOV #HXP2,$TMP5
2125 012734 012737 013370 001250 MOV #HXP7,$TMP7
2126 012742 012737 013300 001246 HXER22: MOV #HXDAT0,$TMP6
2127 012750 104233 1$: ERROR +233
2128 012752 000137 013410 JMP HXDONE
2129
2130 012756 012737 013320 001244 HXER3: MOV #HXP2,$TMP5
2131 012764 012737 013370 001250 MOV #HXP7,$TMP7
2132 012772 012737 013300 001246 HXER33: MOV #HXDAT0,$TMP6
2133 013000 104241 1$: ERROR +241
2134 013002 000137 013410 JMP HXDONE
2135
2136 013006 010537 001240 HXER4: MOV R5,$TMP3
2137 013012 010437 001242 MOV R4,$TMP4
2138 013016 104240 1$: ERROR +240
2139 013020 000137 013410 JMP HXDONE
2140
2141 013024 010537 001240 HXER8: MOV R5,$TMP3
2142 013030 010437 001242 MOV R4,$TMP4
2143 013034 104242 1$: ERROR +242
2144 013036 000137 013410 JMP HXDONE
2145 013042 012737 013320 001244 HXER6: MOV #HXP2,$TMP5
2146 013050 012737 013400 001250 MOV #HXP8,$TMP7
2147 013056 012737 013300 001246 HXER66: MOV #HXDAT0,$TMP6
2148 013064 104244 1$: ERROR +244
2149 013066 000137 013410 JMP HXDONE
2150
2151 013072 012737 013320 001244 HXER7: MOV #HXP2,$TMP5
2152 013100 012737 013400 001250 MOV #HXP8,$TMP7
2153 013106 012737 013300 001246 MOV #HXDAT0,$TMP6
2154 013114 104243 1$: ERROR +243
2155 013116 000137 013410 JMP HXDONE
2156
2157 013122 032716 000001 HXER9: BIT #1,(SP) ;SEE IF IT
2158 013126 001005 BNE 1$ ;AN ODD ADDRESS
2159 013130 022716 012232 CMP #HX165,(SP)
2160 013134 001402 BEQ 1$
2161 013136 000137 037032 JMP CPSPUR
2162
2163 013142 011637 001236 1$: MOV (SP),$TMP2
2164 013146 022626 CMP (SP)+,(SP)+
2165 013150 104235 2$: ERROR +235
2166 013152 000137 013410 JMP HXDONE
2167
2168 013156 162701 000003 HXER10: SUB #3,R1
2169 013162 006301 ASL R1
2170 013164 012702 012232 MOV #HX165,R2
2171 013170 010237 001242 MOV R2,$TMP4
2172 013174 160102 SUB R1,R2
2173 013176 010237 001240 MOV R2,$TMP3
2174 013202 104236 1$: ERROR +236
    
```

```

2175 013204 000137 013410          JMP      HXDONE
2176
2177 013210 012737 013340 001244 HXER11: MOV    #HXP4,$TMP5
2178 013216 012737 013350 001250      MOV    #HXP5,$TMP7
2179 013224 000137 012742          JMP      HXER22
2180 013230 012737 013340 001244 HXER12: MOV    #HXP4,$TMP5
2181 013236 012737 013350 001250      MOV    #HXP5,$TMP7
2182 013244 012737 013300 001246      MOV    #HXDAT0,$TMP6
2183 013252 104245          1$:     ERROR  +245
2184 013254 000137 013410          JMP      HXDONE
2185
2186 013260 012737 013310 001244 HXER13: MOV    #HXP1,$TMP5
2187 013266 012737 013310 001250      MOV    #HXP1,$TMP7
2188 013274 000137 012742          JMP      HXER22
2189
2190 013300 000000          HXDAT0: 0
2191 013302 000000              0
2192 013304 000000              0
2193 013306 000000              0
2194
2195 013310 000000          HXP1:   0
2196 013312 000000              0
2197 013314 000000              0
2198 013316 000000              0
2199
2200 013320 000577          HXP2:   577
2201 013322 177776              177776
2202 013324 177777              177777
2203 013326 177776              177776
2204 013330 005201          HXP3:   5201
2205 013332 000000              0
2206 013334 000000              0
2207 013336 000000              0
2208 013340 100577          HXP4:  100577
2209 013342 177776              177776
2210 013344 177777              177777
2211 013346 177776              177776
2212 013350 100577          HXP5:  100577
2213 013352 177776              177776
2214 013354 000000              0
2215 013356 000000              0
2216 013360 000252          HXP6:   252
2217 013362 125252              125252
2218 013364 125252              125252
2219 013366 125252              125252
2220
2221 013370 000577          HXP7:   577
2222 013372 177776              177776
2223 013374 000000              0
2224 013376 000000              0
2225 013400 000577          HXP8:   577
2226 013402 177777              177777
2227 013404 000000              0
2228 013406 000000              0
2229
2230 013410          HXDONE:
      013410 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?).

2231
2232
2233
2241

2242

```
.SBTTL TEST # 4 - CMPD TEST  
:*****  
:*TEST 4      CMPD TEST  
:*  
:*THIS IS A TEST OF THE CMPD INSTRUCTION. NOTE THAT A SUBROUTINE  
:*IS USED TO SET UP OPERANDS, EXECUTE THE INSTRUCTION AND CHECK THE  
:*RESULTS  
:*  
:*****
```

```
013412 000004  
2243  
2244  
2245 013414  
013414 104413  
2246 013416 004737 014226  
2247 013422 000000 000000 000000 1$:  
2248 013432 000000 000000 000000 2$:  
2249 013442 000200 3$:  
2250 013444 000204  
2251 013446 000200  
2252 013450 104001 4$:
```

```
TST4:  SCOPE  
:TEST THE CMPD INSTRUCTION WITH (FSRC=AC=0)  
AAA1:  LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.  
        JSR          PC,CMPD  
        .WORD        0,0,0,0      ;AC0  
        .WORD        0,0,0,0      ;FSRC  
        200              ;FPS BEFORE EXECUTION  
        204              ;FPS AFTER EXECUTION  
        200              ;ERROR FPS  
        ERROR +1          ;FPS ERROR
```

```
2253  
2254  
2255  
2256 013452  
013452 104413  
2257 013454 004737 014226  
2258 013460 000000 000000 000000 1$:  
2259 013470 025252 2$:  
2260 013472 052525  
2261 013474 125252  
2262 013476 052525  
2263 013500 000200 3$:  
2264 013502 000200  
2265 013504 000210  
2266 013506 104003 4$:
```

```
:TEST CMPD WITH (AC=0) AND FSRC POSITIVE.  
AAA2:  LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.  
        JSR          PC,CMPD  
        .WORD        0,0,0,0      ;AC  
        25252          ;FSRC  
        52525  
        125252  
        52525  
        200              ;FPS BEFORE EXECUTION  
        200              ;FPS AFTER EXECUTION  
        210              ;ERROR FPS  
        ERROR +3          ;FPS ERROR
```

```
2267  
2268  
2269 013510  
013510 104413  
2270 013512 004737 014226  
2271 013516 000000 000000 000000 1$:  
2272 013526 125252 2$:  
2273 013530 125252  
2274 013532 052525  
2275 013534 125252  
2276 013536 000200 3$:  
2277 013540 000210  
2278 013542 000200  
2279 013544 104004 4$:
```

```
:TEST CMPD WITH (AC=0) AND FSRC NEGATIVE  
AAA3:  LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.  
        JSR          PC,CMPD  
        .WORD        0,0,0,0      ;AC  
        125252          ;FSRC  
        125252  
        52525  
        125252  
        200              ;FPS BEFORE EXECUTION  
        210              ;FPS AFTER EXECUTION  
        200              ;ERROR FPS  
        ERROR +4          ;FPS ERROR.
```

```
2280  
2281  
2282 013546  
013546 104413  
2283 013550 004737 014226  
2284 013554 025252  
2285 013556 052525
```

```
:TEST CMPD WITH (FSRC=0) AND AC POSITIVE  
AAA4:  LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.  
        JSR          PC,CMPD  
        25252          ;AC  
        52525
```



```

2286 013560 125252          125252
2287 013562 052525          52525
2288 013564 000000 000000 000000 2$: .WORD 0,0,0,0      ;FSRC
2289 013574 000200          200          ;FPS BEFORE EXECUTION
2290 013576 000210          210          ;FPS AFTER EXECUTION
2291 013600 000200          200          ;ERROR FPS
2292 013602 104005          4$: ERROR +5      ;FPS ERROR
2293
2294
2295      ;TEST CMPD WITH (FSRC=0) AND AC NEGATIVE
2296 013604 AAA5:          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
      013604 104413          JSR          PC,CMPSUB
2297 013606 004737 014226          1$: 125252          ;AC
2298 013612 125252          125252
2299 013614 125252          52525
2300 013616 052525          125252
2301 013620 125252          2$: .WORD 0,0,0,0      ;FSRC
2302 013622 000000 000000 000000 3$: 200          ;FPS BEFORE EXECUTION
2303 013632 000200          200          ;FPS AFTER EXECUTION
2304 013634 000200          210          ;ERROR FPS
2305 013636 000210          4$: ERROR +6      ;FPS ERROR
2306 013640 104006
2307
2308      ;TEST CMPD WITH AC POSITIVE AND FSRC NEGATIVE
2309 013642 AAA6:          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
      013642 104413          JSR          PC,CMPSUB
2310 013644 004737 014226          1$: 52525          ;AC
2311 013650 052525          125252
2312 013652 125252          52525
2313 013654 052525          125252
2314 013656 125252          2$: 125252          ;;FSRC
2315 013660 125252          52525
2316 013662 052525          125252
2317 013664 125252          52525
2318 013666 052525          3$: 200          ;FPS BEFORE EXECUTION
2319 013670 000200          210          ;FPS AFTER EXECUTION
2320 013672 000210          200          ;ERROR FPS
2321 013674 000200          4$: ERROR +7      ;FPS ERROR
2322 013676 104007
2323
2324
2325      ;TEST CMPD WITH AC NEGATIVE AND FSRC POSITIVE
2326 013700 AAA7:          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
      013700 104413          JSR          PC,CMPSUB
2327 013702 004737 014226          1$: 125252          ;AC
2328 013706 125252          52525
2329 013710 052525          125252
2330 013712 125252          52525
2331 013714 052525          2$: 52525          ;FSRC
2332 013716 052525          125252
2333 013720 125252          52525
2334 013722 052525          125252
2335 013724 125252          3$: 200          ;FPS BEFORE EXECUTION
2336 013726 000200          200          ;FPS AFTER EXECUTION
2337 013730 000200          210          ;ERROR FPS
2338 013732 000210          4$: ERROR +10     ;FPS ERROR.
2339 013734 104010

```

```
2340
2341      ;TEST CMPD WITH AC POSITIVE AND FSRC POSITIVE
2342      ;AND EAC LESS THAN EFSRC.
2343      AAA8:
2344      013736 104413      LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.
2345      013736 004737 014226 JSR      PC,CMPSUB
2346      013744 012345      1$:      12345      ;AC
2347      013746 067654      67654
2348      013750 032101      32101
2349      013752 023456      23456
2350      013754 023456      2$:      23456      ;FSRC
2351      013756 076543      76543
2352      013760 021012      21012
2353      013762 034567      34567
2354      013764 000200      3$:      200      ;FPS BEFORE EXECUTION
2355      013766 000200      200      ;FPS AFTER EXECUTION
2356      013770 000210      210      ;ERROR FPS
2357      013772 104011      4$:      ERROR +11 ;FPS ERROR
2358
2359      ;TEST CMPD WITH AC POSITIVE, FSRC POSITIVE AND EAC GREATER THAN EFSRC
2360      AAA9:
2361      013774 104413      LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.
2362      013776 004737 014226 JSR      PC,CMPSUB
2363      014002 045676      1$:      45676      ;AC
2364      014004 054321      54321
2365      014006 012345      12345
2366      014010 067654      67654
2367      014012 034567      2$:      34567      ;FSRC
2368      014014 065432      65432
2369      014016 101234      101234
2370      014020 056765      56765
2371      014022 000200      3$:      200      ;FPS BEFORE EXECUTION
2372      014024 000210      210      ;FPS AFTER EXECUTION
2373      014026 000200      200      ;ERROR FPS
2374      014030 104012      4$:      ERROR +12
2375
2376      ;TEST CMPD WITH AC POSITIVE, FSRC POSITIVE AND AC EQUAL TO FSRC
2377      AAA10:
2378      014032 104413      LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.
2379      014034 004737 014226 JSR      PC,CMPSUB
2380      014040 012345      1$:      12345      ;AC
2381      014042 067012      67012
2382      014044 034567      34567
2383      014046 012345      012345
2384      014050 012345      2$:      12345      ;FSRC
2385      014052 067012      67012
2386      014054 034567      34567
2387      014056 012345      012345
2388      014060 000200      3$:      200      ;FPS BEFORE EXECUTION
```



```
2388 014062 000204          204          :FPS AFTER EXECUTION
2389 014064 000200          200          :ERROR FPS
2390 014066 104013          4$: ERROR +13 :FPS ERROR
2391
2392          :TEST CMPD WITH AC POSITIVE, FSRC POSITIVE, EAC EQUAL TO EFSRC,
2393          :AND FSRC GREATER THAN AC.
2394 014070          AAA11:
      014070 104413          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
2395 014072 004737 014226          JSR PC,CMPSUB
2396 014076 012345          1$: 12345          :AC
      067012
      34567
2397 014100 067012          2$: 012345          :FSRC
      012345
2398 014102 034567          12345
2399 014104 012345          70123
2400 014106 012345          45670
2401 014110 070123          123456
2402 014112 045670          3$: 200          :FPS BEFORE EXECUTION
      200          :FPS AFTER EXECUTION
2403 014114 123456          210          :ERROR FPS
2404 014116 000200          4$: ERROR +14 :FPS ERROR
2405 014120 000200
2406 014122 000210
2407 014124 104014
2408
2409          :TEST CMPD WITH AC POSITIVE, FSRC POSITIVE, EAC EQUAL TO EFSRC,
2410          :AND AC GREATER THAN FSRC.
2411 014126          AAA12:
      014126 104413          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
2412 014130 004737 014226          JSR PC,CMPSUB
2413 014134 054321          1$: 54321          :AC
      76543
      21076
2414 014136 076543          2$: 54321          :FSRC
      54321
2415 014140 021076          65432
2416 014142 054321          107654
2417 014144 054321          3$: 200          :FPS BEFORE EXECUTION
      32107          :FPS AFTER EXECUTION
2418 014146 065432          210          :ERROR FPS
2419 014150 107654          4$: ERROR +15 :FPS ERROR
2420 014152 032107
2421 014154 000200
2422 014156 000210
2423 014160 000200
2424 014162 104015
2425
2426          :TEST CMPD WITH AC NEGATIVE, FSRC NEGATIVE, EAC EQUAL TO EFSRC,
2427          :AND AC GREATER THAN FSRC
2428 014164          AAA13:
      014164 104413          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
2429 014166 004737 014226          JSR PC,CMPSUB
2430 014172 112345          1$: 112345          :AC
      43210
      76543
2431 014174 043210          2$: 112345          :FSRC
      21076
2432 014176 076543          112345
2433 014200 021076          54321
2434 014202 112345          76543
2435 014204 054321          3$: 200          :FPS BEFORE EXECUTION
      07654          :FPS AFTER EXECUTION
2436 014206 007654          210          :ERROR FPS
2437 014210 032107          4$: ERROR +16 :FPS ERROR
2438 014212 000200
2439 014214 000210
2440 014216 000200
2441 014220 104016
```

```

2442
2443
2444 014222 000137 014416          JMP      AAADONE          ;FINISHED CMPD TEST.
2445
2446
2447          ;THIS SUBROUTINE, CMPSUB, IS CALLED TO SET UP, EXECUTE
2448          ;AND CHECK THE RESULTS OF A CMPD INSTRUCTION.
2449          ;IT IS CALLED THUS:
2450          :
2451          :          JSR      PC,CMPSUB
2452          :          ACARG:  .WORD  X,X,X,X          ;AC OPERAND
2453          :          FSRCARG: .WORD  X,X,X,X          ;FSRC OPERAND
2454          :          FPSB:   .WORD  X              ;FPS BEFORE EXECUTION
2455          :          FPSA:   .WORD  X              ;FPS AFTER EXECUTION
2456          :          FPSE:   .WORD  X              ;ERROR FPS
2457          :          ERR:    ERROR  +X             ;FPS ERROR
2458          :          CONT:   ;RETURN ADDRESS
2459          :
2460          ;THE OPERANDS ARE SET UP (USING ACO FOR THE AC OPERAND). THEN
2461          ;FPSB IS LOADED INTO THE FPS. THE INSTRUCTION, CMPD, IS EXECUTED.
2462          ;AFTER THE EXECUTION THE FPS IS CHECKED AGAINST FPSA. IF IT IS A MATCH
2463          ;THEN THERE WAS NO ERROR AND CONTROL IS RETURNED TO CONT. IF
2464          ;THE FPS IS INCORRECT IT IS COMPARED WITH FPSE IN AN ATTEMPT TO ANALYSE
2465          ;THE FAILURE. IF THE FPS IS THE SAME AS FPSE THEN CONTROL IS
2466          ;RETURNED TO THE ERROR CALL AT LOCATION ERR. IF THE FPS WAS
2467          ;NOT CORRECT BUT DIDN'T MATCH FPSE A GENERAL ERROR IS REPORTED
2468          ;AND CONTROL IS PASSED TO CONT.
2469
2470 014226 012601          CMPSUB: MOV      (SP)+,R1          ;PICK UP A POINTER TO THE
2471          ;ARGUMENTS.
2472 014230 016100 000020          MOV      20(R1),R0          ;GET THE FPS BEFORE EXECUTION.
2473 014234 170100          LDFPS   R0              ;LOAD IT INTO THE FPS.
2474
2475 014236 012737 014260 001236          MOV      #1$, $TMP2          ;SAVE ADDRESS OF CMPD INSTRUCTION.
2476 014244 010100          MOV      R1,R0          ;GET ADDRESS OF AC OPERAND.
2477 014246 172410          LDD     (R0),ACO          ;LOAD ACO OPERAND
2478
2479 014250 010100          MOV      R1,R0          ;COMPUTE FSRC OPERAND
2480 014252 062700 000010          ADD     #10,R0          ;ADDRESS
2481
2482 014256 000240          NOP
2483 014260 173410          1$:    CMPD   (R0),ACO          ;FOR SCOPING.
2484          ;EXECUTE THE TEST INSTRUCTION.
2485 014262 170205          STFPS   R5              ;SAVE FPS AFTER INSTRUCTION.
2486
2487 014264 016104 000022          MOV      22(R1),R4          ;GET EXPECTED FPS.
2488          ;IF INCORRECT SET UP FOR
2489 014270 010137 001240          MOV      R1,$TMP3          ;AN ERROR CALL.
2490 014274 010137 001242          MOV      R1,$TMP4
2491 014300 062737 000010 001242          ADD     #10,$TMP4
2492 014306 010537 001244          MOV      R5,$TMP5
2493 014312 010437 001246          MOV      R4,$TMP6
2494 014316 020405          CMP     R4,R5          ;WAS FPS CORRECT?
2495 014320 001410          BEQ     3$              ;BRANCH IF YES.
2496
2497
2498 014322 026105 000024          CMP     24(R1),R5          ;WAS THE FPS THE SAME

```



```

2499
2500 014326 001003          BNE      2$          ;AS THE EXPECTED INCORRECT FPS?
                                ;BRANCH IF NO MATCH.
2501
2502 014330 062701 000026  ADD      #26,R1      ;IF THE EXPECTED INCORRECT
2503                                ;FPS MATCHED THE RESULTANT FPS
2504 014334 000111          JMP      (R1)        ;RETURN TO THE ERROR CALL
2505                                ;IN THE CALLING ROUTINE.
2506
2507 014336 104001          2$:      ERROR    +1      ;OTHERWISE REPORT INCORRECT FPS
2508 014340 000411          BR       5$
2509
2510 014342 012700 014406  3$:      MOV      #CMPTMP,R0      ;IF FPS WAS CORRECT MAKE SURE
2511 014346 174010          STD      ACO,(R0)      ;ACO WAS NOT AFFECTED BY CMPD.
2512 014350 010102          MOV      R1,R2
2513 014352 012703 000004  MOV      #4,R3
2514 014356 022220          4$:      CMP      (R2)+,(R0)+
2515 014360 001003          BNE      6$
2516 014362 077303          SOB     R3,4$
2517
2518 C14364 000161 0C0030  5$:      JMP      30(R1)      ;RETURN
2519
2520 014370          6$:
2521 014370 010137 001240  MOV      R1,$TMP3      ;REPORT ACO MODIFIED BY CMPD
2522 014374 012737 014406 001242  MOV      #CMPTMP,$TMP4
2523 014402 104002          7$:      ERROR    +2
2524 014404 000767          BR       5$          ;RETURN
2525
2526 014406 000000 000000 000000  CMPTMP: .WORD 0,0,0,0
2527
2528
2529
2530 014416          AAADONE:
      014416 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?).

2531
2532
2533
2541
  
```

2542

```

.SBTTL TEST # 5 - DIVD WITH (FSRC=0) AND (BUT FD) TEST
*****
:TEST 5      DIVD WITH (FSRC=0) AND (BUT FD) TEST
:
:THIS IS A TEST OF THE DIVD INSTRUCTION WITH A
:ZERO DIVISOR. THE CONDITION IS CHECKED WITH BOTH
:TRAP ENABLED AND TRAPS DISABLED.
:
*****
TST5:  SCOPE
    
```

```

2543 014420 000004
2544
2545 014422 104413
2546 014424 012704 040200
2547
2548
2549 014430 170104
2550 014432 012737 014674 000244
2551 014440 012737 014460 001236
2552 014446 012700 015100
2553 014452 172410
2554 014454 012701 015100
2555
2556 014460 174411
2557
2558 014462 170205
2559 014464 170303
2560
2561 014466 012704 140204
2562 014472 020405
2563 014474 001131
2564
2565 014476 012702 000004
2566 014502 020203
2567 014504 001140
2568
2569
2570 014506
2571 014510 012704 040200
2572 014514 170104
2573
2574 014516 012737 014536 001236
2575 014524 012700 015110
2576 014530 172410
2577 014532 012700 015100
2578 014536 174410
2579
2580 014540 170205
2581 014542 170303
2582
2583 014544 012704 140200
2584 014550 020405
2585 014552 001102
2586
2587 014554 012702 000004
2588 014560 020203
    
```

```

:FIRST TEST DIVD WITH (FSRC=AC=0) AND TRAPS DISABLED.
BBB0:  LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.
      MOV        #40200,R4 ;SET UP FPS
      ;WITH INTERRUPTS
      ;DISABLED.
      LDFPS     R4
      MOV        #BBBER1,FPVECT;SET UP FOR ANY FP INTERRUPTS.
      MOV        #BBB1,$TMP2
      MOV        #BBBP1,R0 ;SET UP ACO = 0
      LDD        (R0),ACO
      MOV        #BBBP1,R1 ;FSRC = 0
BBB1:  DIVD      (R1),ACO ;TEST INSTRUCTION
      STFPS     R5 ;GET FPS
      STST      R3 ;GET FEC
      MOV        #140204,R4 ;EXPECTED FPS.
      CMP        R4,R5 ;IS FPS CORRECT.
      BNE       BBBER2 ;IF INCORRECT BRANCH.
      MOV        #4,R2 ;EXPECTED FEC.
      CMP        R2,R3 ;IS FEC CORRECT?
      BNE       BBBER3 ;IF INCORRECT BRANCH.
:TEST DIVD WITH (FSRC=0) AND TRAPS DISABLED.
BBB2:  LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.
      MOV        #40200,R4 ;LOAD FPS WITH TRAPS DISABLED.
      LDFPS     R4
      MOV        #BBB3,$TMP2
      MOV        #BBBP2,R0 ;SET UP ACO OPERAND (NON ZERO).
      LDD        (R0),ACO
      MOV        #BBBP1,R0 ;FSRC=0
BBB3:  DIVD      (R0),ACO
      STFPS     R5 ;GET FPS.
      STST      R3 ;GET FEC.
      MOV        #140200,R4 ;EXPECTED FPS.
      CMP        R4,R5 ;IS FPS CORRECT?
      BNE       BBBER2 ;IF INCORRECT BRANCH.
      MOV        #4,R2 ;EXPECTED FEC.
      CMP        R2,R3 ;WAS FEC CORRECT?
    
```



```

2589 014562 001111          BNE      BBBER3          ;IF INCORRECT BRANCH.
2590
2591          ;TEST DIVD WITH FSRC=0) AND TRAPS ENABLED.
2592 014564          BBB4:
      014564 104413          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
2593 014566 012704 000200    MOV      #200,R4          ;SET UP FPS. TRAP ENABLED.
2594 014572 170104          LDFPS     R4
2595
2596 014574 012737 014622 001236    MOV      #BBB5,$TMP2
2597 014602 012700 015110          MOV      #B3BP2,R0          ;SET UP ACO OPERAND (NON ZERO).
2598 014606 172410          LDD      (R0),ACO
2599
2600 014610 012737 014630 000244    MOV      #BBB6,FPVECT
2601 014616 012700 015100          MOV      #BBBP1,R0          ;SET UP FOR THE EXPECTED INTERRUPT.
2602          ;FSRC=0
2603 014622 174410          BBB5:  DIVD     (R0),ACO          ;TEST INSTRUCTION (SHOULD RESULT IN TRAP).
2604 014624 170000          CFCC
2605
2606 014626 000502          BR       BBBER4          ;GO REPORT FAILURE, NO TRAP.
2607
2608 014630 022716 014624          BBB6:  CMP      #BBB5+2,(SP)          ;TRAP TO HERE WHEN THE DIVISION BY 0
2609          ;OCCURS. FIRST SEE IF THE ADDRESS OF
2610          ;THE TRAP IS 2+THE ADDRESS OF THE TEST
2611          ;DIVD INSTRUCTION.
2612 014634 001402          BEQ     1$
2613 014636 000137 037000          JMP     FPSPUR          ;IF NOT THEN REPORT AN UNEXPECTED
2614          ;FP TRAP.
2615 014642 170205          1$:  STFPS    R5          ;GET FPS.
2616 014644 170303          STST   R3          ;GET FEC.
2617 014646 022626          CMP     (SP)+,(SP)+          ;RESET THE STACK.
2618
2619 014650 012704 100200          MOV     #100200,R4          ;EXPECTED FPS.
2620 014654 020405          CMP     R4,R5          ;IS FPS CORRECT?
2621 014656 001040          BNE     BBBER2          ;IF INCORRECT BRANCH.
2622
2623 014660 012702 000004          MOV     #4,R2          ;EXPECTED FEC.
2624 014664 020203          CMP     R2,R3          ;IS FEC CORRECT?
2625 014666 001047          BNE     BBBER3          ;IF INCORRECT BRANCH.
2626
2627 014670 000137 015120          JMP     BBBDONE          ;OTHERWISE GO TO NEXT TEST.
2628
2629
2630          ;TRAP HERE IF AN UNEXPECTED INTERRUPT OCCURS.
2631 014674 062737 000002 001236    BBBER1: ADD    #2,$TMP2          ;SEE IF THE INTERRUPT OCCURRED
2632          ;DURING THE EXECUTION OF THE DIVD
2633          ;INSTRUCTION BEING TESTED.
2634 014702 021637 001236          CMP     (SP),$TMP2
2635 014706 001402          BEQ     1$
2636 014710 000137 037000          JMP     FPSPUR          ;IF NOT REPORT UNEXPECTED FP TRAP.
2637
2638 014714 022626          1$:  CMP     (SP)+,(SP)+          ;RESET THE STACK.
2639 014716 170303          STST   R3          ;GET FEC.
2640 014720 170205          STFPS  R5          ;GET FPS.
2641 014722 012737 000004 001240    MOV     #4,$TMP3          ;EXPECTED FEC.
2642 014730 010337 001242          MOV     R3,$TMP4
2643 014734 010537 001244          MOV     R5,$TMP5
2644 014740 010037 001250          MOV     R0,$TMP7
    
```

```

2645 014744 012737 140200 001246      MOV    #140200,$TMP6
2646 014752 104017      2$:    ERROR  +17      ;REPORT (BUT FD) FAILED RESULTING IN AN FP TRAP
2647                                ;WITH TRAPS DISABLED.
2648 014754 000137 015120      JMP    BBBDONE
2649
2650                                ;REPORT FPS INCORRECT:
2651 014760 010537 001242      BBBER2: MOV    R5,$TMP4
2652 014764 010437 001244      MOV    R4,$TMP5
2653 014770 010037 001246      MOV    R0,$TMP6
2654 014774 010137 001250      MOV    R1,$TMP7
2655 015000 104020      1$:    ERROR  +20
2656 015002 000137 015120      JMP    BBBDONE
2657
2658                                ;REPORT FEC INCORRECT:
2659 015006 010337 001242      BBBER3: MOV    R3,$TMP4
2660 015012 010237 001240      MOV    R2,$TMP3
2661 015016 010037 001246      MOV    R0,$TMP6
2662 015022 010137 001250      MOV    R1,$TMP7
2663 015026 104021      1$:    ERROR  +21
2664 015030 000137 015120      JMP    BBBDONE
2665
2666                                ;REPORT NO TRAP OCCURRED AFTER TRYING TO DIVIDE
2667                                ;BY ZERO WITH ALL TRAPS ENABLED.
2668 015034 170303      BBBER4: STST  R3      ;GET FEC.
2669 015036 170205      STFPS  R5      ;GET FPS.
2670 015040 012737 000004 001242      MOV    #4,$TMP4
2671 015046 010337 001240      MOV    R3,$TMP3
2672 015052 010537 001244      MOV    R5,$TMP5
2673 015056 012737 100200 001246      MOV    #100200,$TMP6
2674 015064 010037 001250      MOV    R0,$TMP7
2675 015070 010137 001252      MOV    R1,$TMP10
2676 015074 104022      1$:    ERROR  +22
2677 015076 000410      BR     BBBDONE
2678
2679 015100 000000 000000 000000 BBBP1: .WORD  0,0,0,0
2680 015110 012345 054321 023456 BBBP2: .WORD  12345,54321,23456,76543
2681
2682
2683
2684 015120      BBBDONE:
      015120 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?).
  
```

2685
 2686
 2694

2695

```
.SBTTL TEST # 6 - DIVF TEST  
:*****  
:*TEST 6 DIVF TEST  
:*  
:*THIS IS A TEST OF THE DIVF INSTRUCTION. NOTE THAT A SUBROUTINE IS  
:*USED TO SET UP THE OPERANDS, EXECUTE THE INSTRUCTION AND CHECK THE  
:*RESULTS.  
:*  
:*****  
TST6: SCOPE
```

```
015122 000004  
2696  
2697  
2698 015124  
015124 104413  
2699 015126 004737 015704  
2700 015132 000000 000000  
2701 015136 012345 067012  
2702 015142 000000 000000  
2703 015146 000000  
2704 015150 000004  
2705 015152 012345 067012  
2706 015156 104023  
2707  
2708  
2709 015160  
015160 104413  
2710 015162 004737 015704  
2711 015166 065652 125252  
2712 015172 065600 000000  
2713 015176 040252 125252  
2714 015202 003000  
2715 015204 003000  
2716 015206 040052 125252  
2717 015212 104024  
2718  
2719  
2720 015214  
015214 104413  
2721 015216 004737 015704  
2722 015222 076400 000000  
2723 015226 076400 000000  
2724 015232 040200 000000  
2725 015236 001000  
2726 015240 001000  
2727 015242 140200 000000  
2728  
2729 015246 104025  
2730  
2731  
2732 015250  
015250 104413  
2733 015252 004737 015704  
2734 015256 056777 177777  
2735 015262 054200 000000  
2736 015266 042777 177777  
2737 015272 000000  
2738 015274 000000
```

```
:CHECK DIVF WITH (AC=0).  
CCC1:  
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.  
JSR PC,DIVFSUB  
1$: .WORD 0,0 ;AC  
2$: .WORD 12345,67012 ;FSRC  
3$: .WORD 0,0 ;RES  
4$: 0 ;FPS BEFORE EXECUTION.  
4 ;FPS AFTER EXECUTION  
5$: .WORD 12345,67012 ;ERROR RESULT  
6$: ERROR +23 ;RESULT BAD.  
  
:TEST DIVF WITH AC POSITIVE, FSRC POSITIVE AND IN ROUND MODE.  
CCC2:  
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.  
JSR PC,DIVFSUB  
1$: .WORD 65652,125252 ;AC  
2$: .WORD 65600,0 ;FSRC  
3$: .WORD 40252,125252 ;RES  
4$: 3000 ;FPS BEFORE EXECUTION.  
3000 ;FPS AFTER EXECUTION.  
5$: .WORD 40052,125252 ;ERROR RESULT.  
6$: ERROR +24 ;DIV NORMALIZE FAILURE.  
  
:TEST DIVF WITH AC POSITIVE, FSRC POSITIVE.  
CCC3:  
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.  
JSR PC,DIVFSUB  
1$: .WORD 76400,0 ;AC  
2$: .WORD 76400,0 ;FSRC  
3$: .WORD 40200,0 ;RES  
4$: 1000 ;FPS BEFORE EXECUTION.  
1000 ;FPS AFTER EXECUTION.  
5$: .WORD 140200,0 ;ERROR RES.  
6$: ERROR +25 ;SIGN BAD.  
  
:TEST DIVF WITH BOTH OPERANDS POSITIVE.  
CCC4:  
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.  
JSR PC,DIVFSUB  
1$: .WORD 56777,177777 ;AC  
2$: .WORD 54200,0 ;FSRC  
3$: .WORD 42777,177777 ;RES  
4$: 0 ;FPS BEFORE EXECUTION.  
0 ;FPS AFTER EXECUTION.
```

```
2739 015276 002000 002000 5$: .WORD 2000,2000 ;ERROR RES.
2740 015302 104023 6$: ERROR +23
2741
2742
2743 015304 ;TEST THE DIVF INSTRUCTION:
CCC5: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
015304 104413 JSR PC,DIVFSUB
2744 015306 004737 015704 1$: .WORD 12377,177777 ;AC
2745 015312 012377 177777 2$: .WORD 12300,0 ;FSRC
2746 015316 012300 000000 3$: .WORD 40252,125252 ;RES
2747 015322 040252 125252 4$: 0 ;FPS BEFORE EXECUTION.
2748 015326 000000 0 ;FPS AFTER EXECUTION.
2749 015330 000000 5$: .WORD -1,-1 ;ERROR RES.
2750 015332 177777 177777 6$: ERROR +23
2751 015336 104023
2752
2753 ;TEST DIVIDE ALGORITHM. TEST ROUND CONSTANT.
2754 015340 CCC6: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
015340 104413 JSR PC,DIVFSUB
2755 015342 004737 015704 1$: .WORD 64600,1 ;AC
2756 015346 064600 000001 2$: .WORD 66600,0 ;FSRC
2757 015352 066600 000000 3$: .WORD 36200,1 ;RES
2758 015356 036200 000001 4$: 0 ;FPS BEFORE EXECUTION.
2759 015362 000000 0 ;FPS AFTER EXECUTION.
2760 015364 000000 5$: .WORD 3000,3000 ;ERROR RES.
2761 015366 003000 003000 6$: ERROR +23
2762 015372 104023
2763
2764 ;TEST DIVF.
2765 015374 CCC7: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
015374 104413 JSR PC,DIVFSUB
2766 015376 004737 015704 1$: .WORD 34577,177776 ;AC
2767 015402 034577 177776 2$: .WORD 23400,0 ;FSRC
2768 015406 023400 000000 3$: .WORD 51377,177776 ;RES
2769 015412 051377 177776 4$: 17 ;FPS BEFORE EXECUTION.
2770 015416 000017 0 ;FPS AFTER EXECUTION.
2771 015420 000000 5$: .WORD 3400,3400 ;ERROR RES.
2772 015422 003400 003400 6$: ERROR +23
2773 015426 104023
2774
2775 ;DIVF TEST.
2776 CCC8: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
2777 015430 015430 104413 JSR PC,DIVFSUB
2778 015432 004737 015704 1$: .WORD 67652,125252 ;AC
2779 015436 067652 125252 2$: .WORD 56500,0 ;FSRC
2780 015442 056500 000000 3$: .WORD 51343,107070 ;RES
2781 015446 051343 107070 4$: 0 ;FPS BEFORE EXECUTION.
2782 015452 000000 0 ;FPS AFTER EXECUTION.
2783 015454 000000 5$: .WORD 51543,107070 ;ERROR RES.
2784 015456 051543 107070 6$: ERROR +26 ;DIDN'T INCREMENT THE EXPONENT
2785 015462 104026 ;AFTER DIVID NORMALIZATION.
2786
2787
2788 ;DIVF WITH AC NEGATIVE, FSRC NEGATIVE.
2789 015464 CCC9: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
015464 104413 JSR PC,DIVFSUB
2790 015466 004737 015704
```


2791 015472 140400 000000
2792 015476 140500 000000
2793 015502 040052 125253
2794 015506 000000
2795 015510 000000
2796 015512 140052 125253
2797 015516 104027
2798
2799
2800 015520
2801 015522 104413 015704
2802 015526 004737 000000
2803 015532 160077 000000
2804 015536 040277 000000
2805 015542 160000 000000
2806 015544 000007
2807 015546 000010
2808 015546 060000 000000
2809 015552 104027
2810
2811 015554
2812 015554 104413
2813 015556 004737 015704
2814 015562 040400 000000
2815 015566 140500 000000
2816 015572 040052 125253
2817 015576 000017
2818 015600 000010
2819 015602 040052 125253
2820 015606 104027
2821
2822
2823 015610
2824 015610 104413
2825 015612 004737 015704
2826 015616 060100 000001
2827 015622 040300 000000
2828 015626 060000 000000
2829 015632 000052
2830 015634 000040
2831 015636 060000 000001
2832 015642 104030
2833
2834 015644
2835 015644 104413
2836 015646 004737 015704
2837 015652 060100 000001
2838 015656 040300 000000
2839 015662 060000 000001
2840 015666 000005
2841 015670 000000
2842 015672 060000 000000
2843 015676 104031

1\$: .WORD 140400,0 ;AC
2\$: .WORD 140500,0 ;FSRC
3\$: .WORD 040052,125253 ;RES
4\$: 0 ;FPS BEFORE EXECUTION.
0 ;FPS AFTER EXECUTION.
5\$: .WORD 140052,125253 ;ERROR RES.
6\$: ERROR +27 ;BAD SIGN.
;DIVF WITH AC NEGATIVE AND FSRC POSITIVE.
CCC10: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,DIVFSUB
1\$: .WORD 160077,0 ;AC
2\$: .WORD 40277,0 ;FSRC
3\$: .WORD 160000,0 ;RES
4\$: 7 ;FPS BEFORE EXECUTION.
10 ;FPS AFTER EXECUTION.
5\$: .WORD 60000,0 ;ERROR RES.
6\$: ERROR +27 ;BAD SIGN.
;DIVF WITH AC POSITIVE AND FSRC NEGATIVE.
CCC11: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,DIVFSUB
1\$: .WORD 40400,0 ;AC
2\$: .WORD 140500,0 ;FSRC
3\$: .WORD 140052,125253 ;RES
4\$: 17 ;FPS BEFORE EXECUTION.
10 ;FPS AFTER EXECUTION.
5\$: .WORD 40052,125253 ;ERROR RES.
6\$: ERROR +27 ;BAD SIGN.
;TEST DIVF BOTH OPERANDS POSITIVE AND TRUNCATE MODE.
CCC12: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,DIVFSUB
1\$: .WORD 60100,1 ;AC
2\$: .WORD 40300,0 ;FSRC
3\$: .WORD 60000,0 ;RES
4\$: 52 ;FPS BEFORE EXECUTION.
40 ;FPS AFTER EXECUTION.
5\$: .WORD 60000,1 ;ERROR RES.
6\$: ERROR +30 ;TRUNCATION ERROR
;DIVF WITH POSITIVE OPERANDS AND ROUND MODE.
CCC13: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,DIVFSUB
1\$: .WORD 60100,1 ;AC
2\$: .WORD 40300,0 ;FSRC
3\$: .WORD 60000,1 ;RES
4\$: 5 ;FPS BEFORE EXECUTION.
0 ;FPS AFTER EXECUTION.
5\$: .WORD 60000,0 ;ERROR RES.
6\$: ERROR +31 ;ROUND ERROR.

2844 015700 000137 016130

JMP CCCDONE ;GO TO NEXT TEST.

2845
2846
2847
2848
2849
2850
2851
2852
2853
2854
2855
2856
2857
2858
2859
2860
2861
2862
2863
2864
2865
2866
2867
2868
2869
2870
2871
2872 015704 012601
2873 015706 012700 000200
2874 015712 170100
2875 015714 010100
2876 015716 172410
2877 015720 016100 000014
2878 015724 170100
2879 015726 012737 015742 001236
2880 015734 010100
2881 015736 062700 000004
2882
2883 015742 174410
2884
2885 015744 170204
2886 015746 012700 000200
2887 015752 170100
2888
2889 015754 012700 016120
2890 015760 174010
2891
2892 015762 010102
2893 015764 010237 001240
2894 015770 062702 000004
2895 015774 010237 001242
2896 016000 062702 000004
2897 016004 010237 001244
2898 016010 012737 016120 001246
2899 016016 010437 001250
2900 016022 016137 000016 001252

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

```

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

```

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

```

DIVFSUB:  MOV      (SP)+,R1      ;GET A POINTER TO THE ARGUMENTS.
          MOV      #200,R0      ;SET FD MODE.
          LDFPS   R0
          MOV      R1,R0        ;LOAD THE AC OPERAND.
          LDD     (R0),ACO
          MOV      14(R1),R0    ;LOAD THE FPS
          LDFPS   R0
          MOV      #1$, $TMP2
          MOV      R1,R0
          ADD     #4,R0         ;ESTABLISH A POINTER TO FSRC.
1$:      DIVF    (R0),ACO      ;TEST INSTRUCTION.
          STFPS   R4           ;GET THE FPS.
          MOV      #200,R0      ;SET FD MODE
          LDFPS   R0
          MOV      #DIVFT,R0    ;GET THE RESULT OF THE DIVF.
          STD     ACO,(R0)
          MOV      R1,R2        ;SAVE THE DATA IN CASE OF ERROR.
          MOV      R2,$TMP3
          ADD     #4,R2
          MOV      R2,$TMP4
          ADD     #4,R2
          MOV      R2,$TMP5
          MOV      #DIVFT,$TMP6
          MOV      R4,$TMP7
          MOV      16(R1),$TMP10

```



```
2901
2902 016030 021061 000010          CMP      (R0),10(R1)      ;IS THE RESULT CORRECT?
2903 016034 001011          BNE      10$             ;IF INCORRECT BRANCH.
2904 016036 026061 000002 000012  CMP      2(R0),12(R1)
2905 016044 001005          BNE      10$
2906
2907 016046 026104 000016          CMP      16(R1),R4       ;IS FPS CORRECT?
2908 016052 001020          BNE      15$             ;IF INCORRECT BRANCH.
2909 016054 000161 000026          JMP      26(R1)         ;IF NO ERRORS OCCURRED RETURN.
2910
2911 016060 021061 000020          10$:    CMP      (R0),20(R1) ;DOES THE INCORRECT RESULT
2912 016064 001010          BNE      11$             ;MATCH THE ANTICIPATED INCORRECT RESULT.
2913 016066 026061 000002 000022  CMP      2(R0),22(R1)
2914 016074 001004          BNE      11$             ;BRANCH IF NO.
2915
2916 016076 010102          MOV      R1,R2          ;IT MATCHED SO RETURN TO THE ERROR
2917                                ;REPORT AT THE CALLING ROUTINE.
2918 016100 062702 000024          ADD      #24,R2
2919 016104 000112          JMP      (R2)
2920
2921 016106          11$:
2922 016106 104023          12$:    ERROR    +23
2923 016110 000161 000026          13$:    JMP      26(R1)
2924
2925 016114          15$:
2926 016114 104032          16$:    ERROR    +32
2927 016116 000774          BR      13$
2928
2929 016120 000000 000000 000000  DIVFT:  .WORD  0,0,0,0
2930
2931 016130          CCCDONE:
      016130 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?).

2932
2933
2940
```

2941

```
.SBTTL TEST # 7 - DIVD TEST  
:*****  
:*TEST 7 DIVD TEST  
:*  
:*THIS IS A TEST OF THE DIVD INSTRUCTION. NOTE THAT A SUBROUTINE IS  
:*USED TO SET UP THE OPERANDS, EXECUTE THE INSTRUCTION AND CHECK THE RESULTS.  
:*  
:*****
```

```
016132 000004  
2942  
2943  
2944 016134  
016134 104413  
2945 016136 004737 016624  
2946 016142 034277 000000 000000 1$:  
2947 016152 040277 000000 000000 2$:  
2948 016162 034200 000000 000000 3$:  
2949 016172 000200 4$:  
2950 016174 000200  
2951 016176 177777 177777 177777 5$:  
2952 016206 104033 6$:
```

```
TST7: SCOPE  
;DIVD TEST WITH POSITIVE OPERANDS AND IN ROUND MODE.  
DDD1:  
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.  
JSR PC,DIVDSUB  
.WORD 34277,0,0,0 ;AC  
.WORD 40277,0,0,0 ;FSRC  
.WORD 34200,0,0,0 ;RES  
200 ;FPS BEFORE EXECUTION.  
200 ;FPS AFTER EXECUTION.  
.WORD -1,-1,-1,-1 ;ERROR RES.  
ERROR +33
```

```
2953  
2954  
2955 016210  
016210 104413  
2956 016212 004737 016624  
2957 016216 134277 000000 000000 1$:  
2958 016226 040277 000000 000000 2$:  
2959 016236 134200 000000 000000 3$:  
2960 016246 000207 4$:  
2961 016250 000210  
2962 016252 177777 177777 177777 5$:  
2963 016262 104033 6$:
```

```
;DIVD WITH AC NEGATIVE AND FSRC POSITIVE IN TRUNCATE MODE.  
DDD2:  
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.  
JSR PC,DIVDSUB  
.WORD 134277,0,0,0 ;AC  
.WORD 40277,0,0,0 ;FSRC  
.WORD 134200,0,0,0 ;RES  
207 ;FPS BEFORE EXECUTION.  
210 ;FPS AFTER EXECUTION.  
.WORD -1,-1,-1,-1 ;ERROR RESULT.  
ERROR +33
```

```
2964  
2965  
2966 016264  
016264 104413  
2967 016266 004737 016624  
2968 016272 134300 000000 000000 1$:  
2969 016302 140300 000000 000000 2$:  
2970 016312 034200 000000 000000 3$:  
2971 016322 000250 4$:  
2972 016324 000240  
2973 016326 034200 000000 000000 5$:  
2974 016336 104035 6$:
```

```
;DIVD TEST WITH OPERANDS BOTH NEGATIVE AND IN TRUNCATE MODE.  
DDD3:  
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.  
JSR PC,DIVDSUB  
.WORD 134300,0,0,1 ;AC  
.WORD 140300,0,0,0 ;FSRC  
.WORD 34200,0,0,0 ;RES  
250 ;FPS BEFORE EXECUTION.  
240 ;FPS AFTER EXECUTION.  
.WORD 34200,0,0,1 ;ERROR RES.  
ERROR +35 ;TRUNCATION ERROR.
```

```
2975  
2976  
2977 016340  
016340 104413  
2978 016342 004737 016624  
2979 016346 034300 000000 000000 1$:  
2980 016356 140300 000000 000000 2$:  
2981 016366 134200 000000 000000 3$:  
2982 016376 000207 4$:  
2983 016400 000210  
2984 016402 134200 000000 000000 5$:  
2985 016412 104036 6$:
```

```
;DIVD WITH AC POSITIVE AND FSRC NEGATIVE IN ROUND MODE.  
DDD4:  
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.  
JSR PC,DIVDSUB  
.WORD 34300,0,0,1 ;AC  
.WORD 140300,0,0,0 ;FSRC  
.WORD 134200,0,0,1 ;RES  
207 ;FPS BEFORE EXECUTION.  
210 ;FPS AFTER EXECUTION.  
.WORD 134200,0,0,0 ;ERROR RES.  
ERROR +36 ;ROUND ERROR.
```



```

2986
2987 ;DIVD TEST.
2988 016414 DDD5: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
      016414 104413 JSR PC,DIVDSUB
2989 016416 004737 016624 .WORD 100400,0,0,0 ;AC
2990 016422 100400 000000 000000 1$: .WORD 500,0,0,0 ;FSRC
2991 016432 000500 000000 000000 2$: .WORD 140052,125252 ;RES
2992 016442 140052 125252 .WORD 125252,125252
2993 016446 125252 125252
2994 016452 007647 4$: 7647 ;FPS BEFORE EXECUTION.
2995 016454 007650 7650 ;FPS AFTER EXECUTION.
2996 016456 177777 177777 177777 5$: .WORD -1,-1,-1,-1 ;ERROR RES.
2997 016466 104033 6$: ERROR +33

```

```

2998
2999 ;DIVD TEST WITH AC POSITIVE AND FSRC NEGATIVE IN ROUND MODE.
3000 DDD6: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
      016470 104413 JSR PC,DIVDSUB
3001 016470 004737 016624 .WORD 400,0,0,0 ;AC
3002 016472 004737 016624 000000 1$: .WORD 100500,0,0,0 ;FSRC
3003 016476 000400 000000 000000 2$: .WORD 140052,125252 ;RES
3004 016506 100500 000000 000000 3$: .WORD 125252,125253
3005 016516 140052 125252 .WORD 125252,125253
3006 016522 125252 125253
3007 016526 007707 4$: 7707 ;FPS BEFORE EXECUTION.
3008 016530 007710 7710 ;FPS AFTER EXECUTION.
3009 016532 177777 177777 177777 5$: .WORD -1,-1,-1,-1 ;ERROR RES.
3010 016542 104033 6$: ERROR +33

```

```

3011
3012 ;DIVD TEST.
3013 016544 DDD7: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
      016544 104413 JSR PC,DIVDSUB
3014 016546 004737 016624 .WORD 170360,170360 ;AC
3015 016552 170360 170360 1$: .WORD 170360,170360
3016 016556 170360 170360 .WORD 170360,170360 ;FSRC
3017 016562 170360 170360 2$: .WORD 170360,170360
3018 016566 170360 170360 .WORD 170360,170360
3019 016572 040200 000000 000000 3$: .WORD 40200,0,0,0 ;RES
3020 016602 007717 4$: 7717 ;FPS BEFORE EXECUTION.
3021 016604 007700 7700 ;FPS AFTER EXECUTION.
3022 016606 177777 177777 177777 5$: .WORD -1,-1,-1,-1 ;ERROR RES.
3023 016616 104033 6$: ERROR +33

```

```

3024
3025 016620 000137 017064 JMP DDDDONE ;GO TO NEXT TEST.
3026
3027

```

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

```

3028
3029
3030
3031 :
3032 : JSR PC,DIVDSUB
3033 : ACARG: .WORD X,X,X,X ;AC OPERAND
3034 : FSRCARG: .WORD X,X,X,X ;FSRC OPERAND
3035 : RES: .WORD X,X,X,X ;EXPECTED RESULT
3036 : FPSB: .WORD X ;FPS BEFORE EXECUTION
3037 : FPSA: .WORD X ;FPS AFTER EXECUTION
3038 : ERRES: .WORD X,X,X,X ;ERROR RESULT
3039 : ERR: ERROR +X ;RESULT ERROR
: CONT: ;RETURN ADDRESS

```

3040
3041
3042

: THE OPERANDS ARE SET UP (USING ACO FOR THE AC OPERAND). THEN
: FPSB IS LOADED INTO THE FPS. THE INSTRUCTION, DIVD IS EXECUTED.


```

3044                                     :AFTER THE EXECUTION THE RESULT IS CHECKED AGAINST THE
3045                                     :EXPECTED CORRECT RESULT, RES. IF IT IS CORRECT THEN THE FPS
3046                                     :IS CHECKED WITH THE EXPECTED CORRECT FPS, FPSA. IF THE FPS WAS
3047                                     :INCORRECT THEN IT IS REPORTED. IF THE RESULT WAS INCORRECT IT
3048                                     :IS COMPARED WITH ERRES IN AN ATTEMPT TO ANALYSE THE ERROR. IF
3049                                     :THE INCORRECT RESULT MATCHED ERRES THEN CONTROL IS PASSED TO
3050                                     :THE ERROR CALL AT ERR. IF THE INCORRECT RESULT DID NOT MATCH ERRES
3051                                     :THEN THE FAILURE IS REPORTED IN DIVDSUB AND CONTROL IS PASSED TO
3052                                     :CONT. IF NO ERRORS ARE DETECTED THEN DIVDSUB RETURNS CONTROL
3053                                     :TO CONT.
3054
3055 016624 012601 000200 DIVDSUB:      MOV      (SP)+,R1      ;GET A POINTER TO THE ARGUMENTS.
3056 016626 012700      MOV      #200,R0      ;SET FD MODE.
3057 016632 170100      LDFPS   R0
3058
3059 016634 010100      MOV      R1,R0      ;SET UP THE ACO OPERAND.
3060 016636 172410      LDD      (R0),ACO
3061 016640 016100 000030      MOV      30(R1),R0 ;LOAD THE FPS.
3062 016644 170100      LDFPS   R0
3063
3064 016646 012737 016662 001236      MOV      #1$, $TMP2
3065 016654 010100      MOV      R1,R0      ;ESTABLISH A POINTER TO FSRC.
3066 016656 062700 000010      ADD      #10,R0
3067
3068 016662 174410      1$:     DIVD      (R0),ACO ;EXECUTE THE TEST INSTRUCTION.
3069
3070 016664 170204      STFPS   R4          ;GET THE FPS.
3071 016666 012700 000200      MOV      #200,R0    ;SET FD MODE.
3072 016672 170100      LDFPS   R0
3073
3074 016674 012700 017054      MOV      #DIVDT,R0   ;GET THE RESULT.
3075 016700 174010      STD      ACO,(R0)
3076
3077 016702 010102      MOV      R1,R2      ;SAVE DATA IN CASE OF ERROR.
3078 016704 010237 001240      MOV      R2,$TMP3
3079 016710 062702 000010      ADD      #10,R2
3080 016714 010237 001242      MOV      R2,$TMP4
3081 016720 062702 000010      ADD      #10,R2
3082 016724 010237 001244      MOV      R2,$TMP5
3083 016730 012737 017054 001246      MOV      #DIVDT,$TMP6
3084 016736 010437 001250      MOV      R4,$TMP7
3085 016742 016137 000032 001252      MOV      32(R1),$TMP10
3086
3087 016750 010102      MOV      R1,R2      ;CHECK THE RESULT.
3088 016752 062702 000020      ADD      #20,R2
3089 016756 012703 017054      MOV      #DIVDT,R3
3090 016762 012705 000004      MOV      #4,R5
3091 016766 022223      2$:     CMP      (R2)+,(R3)+ ;BRANCH IF RESULT INCORRECT.
3092 016770 001006      BNE     10$
3093 016772 077503      SOB     R5,2$
3094
3095 016774 026104 000032      CMP      32(R1),R4   ;IS FPS CORRECT?
3096 017000 001023      BNE     15$         ;BRANCH IF INCORRECT.
3097 017002 000161 000046      JMP     46(R1)      ;RETURN.
3098
3099 017006 010102      10$:    MOV      R1,R2      ;WAS INCORRECT RESULT ANTICIPATED?
3100 017010 062702 000034      ADD      #34,R2

```

3101	017014	012703	017054		MOV	#DIVDT,R3	
3102	017020	012705	000004		MOV	#4,R5	
3103	017024	022223		11\$:	CMP	(R2)+,(R3)+	
3104	017026	001005			BNE	12\$:BRANCH IF NO.
3105	017030	077503			SOB	R5,11\$	
3106	017032	010102			MOV	R1,R2	:IF THE INCORRECT RESULT WAS
3107	017034	062702	000044		ADD	#44,R2	:ANTICIPATED RETURN TO THE
3108							:ERROR REPORT IN THE CALLING
3109	017040	000112			JMP	(R2)	:ROUTINE.
3110							
3111	017042			12\$:			:REPORT RESULT INCORRECT.
3112	017042	104033		13\$:	ERROR	+33	
3113	017044	000161	000046	14\$:	JMP	46(R1)	
3114							
3115	017050			15\$:			:REPORT FPS INCORRECT.
3116	017050	104034		16\$:	ERROR	+34	
3117	017052	000774			BR	14\$	
3118							
3119	017054	000000	000000	000000	DIVDT:	.WORD	0,0,0,0
3120							
3121	017064				DDDDONE:		
	017064	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?).

3122
 3123
 3131

3132

```
.SBTTL TEST # 10 - MULF TEST
:*****
:*TEST 10      MULF TEST
:*
:*THIS IS A TEST OF THE MULF INSTRUCTION. IT MAKES USE OF A SUBROUTINE
:*TO SET UP THE OPERANDS, EXECUTE THE MULF INSTRUCTION AND CHECK THE
:*RESULTS.
:*
:*****
```

```
017066 000004
3133
3134
3135 017070
017070 104413
3136 017072 004737 017650
3137 017076 000000 000000
3138 017102 000000 000000
3139 017106 000000 000000
3140 017112 007517
3141 017114 007504
3142 017116 177777 177777
3143 017122 104037
3144
3145
3146 017124
017124 104413
3147 017126 004737 017650
3148 017132 071625 034435
3149 017136 000000 000000
3150 017142 000000 000000
3151 017146 000013
3152 017150 000004
3153 017152 177777 177777
3154 017156 104037
3155
3156
3157 017160
017160 104413
3158 017162 004737 017650
3159 017166 000000 000000
3160 017172 071625 153443
3161 017176 000000 000000
3162 017202 007500
3163 017204 007504
3164 017206 177777 177777
3165 017212 104037
3166
3167
3168 017214
017214 104413
3169 017216 004737 017650
3170 017222 040200 000000
3171 017226 040177 177777
3172 017232 040177 177777
3173 017236 000017
3174 017240 000000
3175 017242 140177 177777
```

```
TST10: SCOPE

:MULF WITH (FSRC=AC=0)
EEE1:
LPERR JSR PC,MULFSUB ;SET UP THE LOOP ON ERROR ADDRESS.
1$: .WORD 0,0 ;AC
2$: .WORD 0,0 ;FSRC
3$: .WORD 0,0 ;RES
4$: 7517 ;FPS BEFORE EXECUTION.
7504 ;FPS AFTER EXECUTION.
5$: .WORD -1,-1
6$: ERROR +37

:MULF WITH (FSRC=0).
EEE2:
LPERR JSR PC,MULFSUB ;SET UP THE LOOP ON ERROR ADDRESS.
1$: .WORD 71625,34435 ;AC
2$: .WORD 0,0 ;FSRC
3$: .WORD 0,0 ;RES
4$: 13 ;FPS BEFORE EXECUTION.
4 ;FPS AFTER EXECUTION.
5$: .WORD -1,-1 ;ERROR RES.
6$: ERROR +37

:MULF WITH (AC=0)
EEE3:
LPERR JSR PC,MULFSUB ;SET UP THE LOOP ON ERROR ADDRESS.
1$: .WORD 0,0 ;AC
2$: .WORD 071625,153443 ;FSRC
3$: .WORD 0,0 ;RES
4$: 7500 ;FPS BEFORE EXECUTION.
7504 ;FPS AFTER EXECUTION.
5$: .WORD -1,-1 ;ERROR RES.
6$: ERROR +37

:MULF WITH AC POSITIVE AND FSRC POSITIVE IN ROUND MODE.
EEE4:
LPERR JSR PC,MULFSUB ;SET UP THE LOOP ON ERROR ADDRESS.
1$: .WORD 40200,0 ;AC
2$: .WORD 40177,-1 ;FSRC
3$: .WORD 40177,-1 ;RES
4$: 17 ;FPS BEFORE EXECUTION.
0 ;FPS AFTER EXECUTION.
5$: .WORD 140177,-1 ;ERROR RES.
```

3176 017246 104041
3177
3178
3179 017250
017250 104413
3180 017252 004737 017650
3181 017256 040177 177777
3182 017262 040200 000000
3183 017266 040177 177777
3184 017272 000040
3185 017274 000040
3186 017276 037777 177777
3187 017302 104042
3188
3189
3190
3191 017304
017304 104413
3192 017306 004737 017650
3193 017312 040100 000000
3194 017316 040100 000000
3195 017322 040020 000000
3196 017326 000012
3197 017330 000000
3198 017332 042040 000000
3199 017336 104043
3200
3201
3202
3203 017340
017340 104413
3204 017342 004737 017650
3205 017346 017500 000000
3206 017352 023652 125252
3207 017356 003177 177777
3208 017362 007417
3209 017364 007400
3210 017366 177777 177777
3211 017372 104037
3212
3213
3214 017374
017374 104413
3215 017376 004737 017650
3216 017402 040342 000000
3217 017406 176542 000000
3218 017412 176707 102000
3219 017416 000007
3220 017420 000010
3221 017422 076507 102000
3222 017426 104041
3223
3224
3225 017430
017430 104413
3226 017432 004737 017650
3227 017436 140200 000000

6\$: ERROR +41 ;BAD SIGN.
:MULF WITH AC POSITIVE AND FSRC POSITIVE IN TRUNCATE MODE.
EEE5: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,MULFSUB
1\$: .WORD 40177,-1 ;AC
2\$: .WORD 40200,0 ;FSRC
3\$: .WORD 40177,-1 ;RES
4\$: 40 ;FPS BEFORE EXECUTION.
40 ;FPS AFTER EXECUTION.
5\$: .WORD 37777,-1 ;ERROR RES.
6\$: ERROR +42 ;ST 252 TO 044 INTO 444 (BUT Y62)
;MUL. NORMALIZATION FAILURE.
:MULF WITH BOTH OPERANDS POSITIVE NORMALIZE TEST.
EEE6: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,MULFSUB
1\$: .WORD 40100,0 ;AC
2\$: .WORD 40100,0 ;FSRC
3\$: .WORD 40020,0 ;RES
4\$: 12 ;FPS BEFORE EXECUTION.
0 ;FPS AFTER EXECUTION.
5\$: .WORD 42040,0 ;ERROR RES.
6\$: ERROR +43 ;ST 252 TO 444 INTO 042 (BUT Y62)
;MUL. NORMALIZATION FAILURE.
:MULF WITH BOTH OPERANDS POSITIVE IN ROUND MODE.
EEE7: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,MULFSUB
1\$: .WORD 17500,0 ;AC
2\$: .WORD 23652,125252 ;FSRC
3\$: .WORD 3177,-1 ;RES
4\$: 7417 ;FPS BEFORE EXECUTION.
7400 ;FPS AFTER EXECUTION.
5\$: .WORD -1,-1 ;ERROR RES.
6\$: ERROR +37 ;BAD SIGN.
:MULF WITH AC POSITIVE AND FSRC NEGATIVE IN ROUND MODE.
EEE8: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,MULFSUB
1\$: .WORD 40342,0 ;AC
2\$: .WORD 176542,0 ;FSRC
3\$: .WORD 176707,102000 ;RES
4\$: 7 ;FPS BEFORE EXECUTION.
10 ;FPS AFTER EXECUTION.
7\$: .WORD 76507,102000 ;ERROR RES.
6\$: ERROR +41 ;BAD SIGN.
:MULF WITH AC NEGATIVE AND FSRC POSITIVE IN ROUND MODE.
EEE9: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,MULFSUB
1\$: .WORD 140200,0 ;AC


```

3228 017442 007417 007417      2$:      .WORD      7417,7417      :FSRC
3229 017446 107417 007417      3$:      .WORD      107417,7417     :RES
3230 017452 000000                4$:      0                          :FPS BEFORE EXECUTION.
3231 017454 000010                4$:      10                         :FPS AFTER EXECUTION.
3232 017456 007417 007417      5$:      .WORD      7417,7417     :ERROR RES.
3233 017462 104041                6$:      ERROR      +41          :BAD SIGN.
3234
3235
3236 017464                :MULF WITH BOTH OPERANDS NEGATIVE IN ROUND MODE.
      104413      EEE10:
      004737      LPERR                      ;SET UP THE LOOP ON ERROR ADDRESS.
      017650      JSR      PC,MULFSUB
3237 017466 004737 017650      1$:      .WORD      144600,0      :AC
3238 017472 144600 000000      2$:      .WORD      154000,0      :FSRC
3239 017476 154000 000000      3$:      .WORD      60400,0      :RES
3240 017502 060400 000000      4$:      17                          :FPS BEFORE EXECUTION.
3241 017506 000017                4$:      0                          :FPS AFTER EXECUTION.
3242 017510 000000                5$:      .WORD      160400,0      :ERROR RES.
3243 017512 160400 000000      6$:      ERROR      +41          :BAD SIGN.
3244 017516 104041
3245
3246                :MULF BOTH OPERANDS NEGATIVE IN ROUND MODE.
3247 017520                EEE11:
      017520 104413      LPERR                      ;SET UP THE LOOP ON ERROR ADDRESS.
      004737 017650      JSR      PC,MULFSUB
3248 017522 004737 017650      1$:      .WORD      140300,0      :AC
3249 017526 140300 000000      2$:      .WORD      160000,1      :FSRC
3250 017532 160000 000001      3$:      .WORD      60100,2      :RES
3251 017536 060100 000002      4$:      10                          :FPS BEFORE EXECUTION.
3252 017542 000010                4$:      0                          :FPS AFTER EXECUTION.
3253 017544 000000                5$:      .WORD      60100,1      :ERROR RES.
3254 017546 060100 000001      6$:      ERROR      +44          :ROUND FAILURE.
3255 017552 104044
3256
3257                :MULF WITH AC POSITIVE AND FSRC NEGATIVE IN TRUNCATE MODE.
3258 017554                EEE12:
      017554 104413      LPERR                      ;SET UP THE LOOP ON ERROR ADDRESS.
      004737 017650      JSR      PC,MULFSUB
3259 017556 004737 017650      1$:      .WORD      60000,1      :AC
3260 017562 060000 000001      2$:      .WORD      140300,0      :FSRC
3261 017566 140300 000000      3$:      .WORD      160100,1      :RES
3262 017572 160100 000001      4$:      7547                          :FPS BEFORE EXECUTION.
3263 017576 007547                4$:      7550                          :FPS AFTER EXECUTION.
3264 017600 007550                5$:      .WORD      160100,1      :ERROR RES.
3265 017602 160100 000001      6$:      ERROR      +45          :TRUNCATION ERROR.
3266 017606 104045
3267
3268                :MULF WITH AC POSITIVE AND FSRC POSITIVE IN ROUND MODE.
3269 017610                EEE13:
      017610 104413      LPERR                      ;SET UP THE LOOP ON ERROR ADDRESS.
      004737 017650      JSR      PC,MULFSUB
3270 017612 004737 017650      1$:      .WORD      40277,0      :AC
3271 017616 040277 000000      2$:      .WORD      60000,1      :FSRC
3272 017622 060000 000001      3$:      .WORD      60077,1      :RES
3273 017626 060077 000001      4$:      14                          :FPS BEFORE EXECUTION.
3274 017632 000014                4$:      0                          :FPS AFTER EXECUTION.
3275 017634 000000                5$:      .WORD      60077,2      :ERROR RES.
3276 017636 060077 000002      6$:      ERROR      +44          :ROUND FAILURE. CONSTANT BAD.
3277 017642 104044
3278
3279 017644 000137 020074      JMP      EEEDONE          ;GO TO THE NEXT TEST.
3280

```

3281
3282
3283
3284
3285
3286
3287
3288
3289
3290
3291
3292
3293
3294
3295
3296
3297
3298
3299
3300
3301
3302
3303
3304
3305
3306
3307
3308
3309
3310
3311
3312
3313
3314
3315
3316
3317
3318
3319
3320
3321
3322
3323
3324
3325
3326
3327
3328
3329
3330
3331
3332
3333
3334
3335
3336
3337

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

```

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

```

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

```

MULFSUB:  MOV      (SP)+,R1      ;GET A POINTER TO THE ARGUMENTS.
:      MOV      #200,R0        ;SET FD MODE.
:      LDFPS    R0
:      MOV      R1,R0         ;LOAD THE AC OPERAND.
:      LDD      (R0),ACO
:      MOV      14(R1),R0     ;LOAD THE FPS
:      LDFPS    R0
:      MOV      #1$, $TMP2
:      MOV      R1,R0
:      ADD      #4,R0         ;ESTABLISH A POINTER TO FSRC.
:
1$:      MULF    (R0),ACO      ;TEST INSTRUCTION.
:
:      STFPS    R4            ;GET THE FPS.
:      MOV      #200,R0      ;SET FD MODE
:      LDFPS    R0
:
:      MOV      #MULFT,R0    ;GET THE RESULT OF THE MULF.
:      STD      ACO,(R0)
:
:      MOV      R1,R2        ;SAVE THE DATA IN CASE OF ERROR.
:      MOV      R2,$TMP3
:      ADD      #4,R2
:      MOV      R2,$TMP4
:      ADD      #4,R2
:      MOV      R2,$TMP5
:      MOV      #MULFT,$TMP6
:      MOV      R4,$TMP7
:      MOV      16(R1),$TMP10
:
:      CMP      (R0),10(R1)  ;IS THE RESULT CORRECT?

```

012601
012700 000200
170100
010100
172410
016100 000014
170100
012737 017706 001236
010100
062700 000004
171010
170204
012700 000200
170100
012700 020064
174010
010102
010237 001240
062702 000004
010237 001242
062702 000004
010237 001244
012737 020064 001246
010437 001250
016137 000016 001252
021061 000010


```

TEST # 10 - MULF TEST

3338 020000 001011          BNE      10$          ;IF INCORRECT BRANCH.
3339 020002 026061 000002 000012  CMP      2(R0),12(R1)
3340 020010 001005          BNE      10$
3341
3342 020012 026104 000016          CMP      16(R1),R4    ;IS FPS CORRECT?
3343 020016 001020          BNE      15$          ;IF INCORRECT BRANCH.
3344 020020 000161 000026          JMP      26(R1)       ;IF NO ERRORS OCCURRED RETURN.
3345
3346 020024 021061 000020          10$:    CMP      (R0),20(R1) ;DOES THE INCORRECT RESULT
3347 020030 001010          BNE      11$          ;MATCH THE ANTICIPATED INCORRECT RESULT.
3348 020032 026061 000002 000022  CMP      2(R0),22(R1)
3349 020040 001004          BNE      11$          ;BRANCH IF NO.
3350
3351 020042 010102          MOV      R1,R2        ;IT MATCHED SO RETURN TO THE ERROR
3352                                ;REPORT AT THE CALLING ROUTINE.
3353 020044 062702 000024          ADD      #24,R2
3354 020050 000112          JMP      (R2)
3355
3356 020052          11$:
3357 020052 104037          12$:    ERROR   +37
3358 020054 000161 000026          13$:    JMP      26(R1)
3359
3360 020060          15$:
3361 020060 104040          16$:    ERROR   +40
3362 020062 000774          BR      13$
3363
3364 020064 000000 000000 000000  MULFT:  .WORD  0,0,0,0
3365
3366 020074          EEEDONE:
      020074 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?).

3367
3368
3376

```

3377

```
.SBTTL TEST # 11 - MULD TEST
:*****
:*TEST 11      MULD TEST
:*
:*THIS IS A TEST OF THE MULD INSTRUCTION. NOTE THAT A SUBROUTINE IS
:*USED TO SET UP THE OPERANDS, EXECUTE THE MULD INSTRUCTION AND
:*CHECK THE RESULTS.
:*
:*****
```

020076 000004

TST11: SCOPE

3378

```
:MULD TEST WITH AC POSITIVE AND FSRC POSITIVE.
```

3379

FFF1:

3380 020100

```
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
```

020100 104413

```
JSR PC,MULDSUB
```

3381 020102 004737 020364

```
.WORD 40200,0,0,0 ;AC
```

3382 020106 040200 000000 000000

```
.WORD 23777,-1,-1,-1 ;FSRC
```

3383 020116 023777 177777 177777

```
.WORD 23777,-1,-1,-1 ;RES
```

3384 020126 023777 177777 177777

```
217 ;FPS BEFORE EXECUTION.
```

3385 020136 000217

```
200 ;FPS AFTER EXECUTION.
```

3386 020140 000200

```
.WORD 23777,-1,0,0 ;ERROR RES.
```

3387 020142 023777 177777 000000

```
ERROR +47 ;BAD CONSTANT USED IN ALGORITHM
```

3388 020152 104047

```
 ;USED 24 INSTEAD OF 56.
```

3389

3390

3391

```
:MULD TEST WITH BOTH OPERANDS POSITIVE TRUNCATION TEST.
```

3392 020154

FFF2:

020154 104413

```
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
```

3393 020156 004737 020364

```
JSR PC,MULDSUB
```

3394 020162 065400 000000 000000

```
.WORD 65400,0,0,1 ;AC
```

3395 020172 037577 177777 177777

```
.WORD 37577,-1,-1,-2 ;FSRC
```

3396 020202 064777 177777 177777

```
.WORD 64777,-1,-1,-1 ;RES
```

3397 020212 000247

```
247 ;FPS BEFORE EXECUTION.
```

3398 020214 000240

```
240 ;FPS AFTER EXECUTION.
```

3399 020216 065000 000000 000000

```
.WORD 65000,0,0,0 ;ERROR RES.
```

3400 020226 104050

```
ERROR +50 ;TRUNCATION ERROR.
```

3401

3402

3403 020230

```
:MULD TEST WITH BOTH OPERANDS NEGATIVE IN ROUND MODE.
```

FFF3:

020230 104413

```
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
```

3404 020232 004737 020364

```
JSR PC,MULDSUB
```

3405 020236 137577 177777 177777

```
.WORD 137577,-1,-1,-2 ;AC
```

3406 020246 165400 000000 000000

```
.WORD 165400,0,0,1 ;FSRC
```

3407 020256 065000 000000 000000

```
.WORD 65000,0,0,0 ;RES
```

3408 020266 007717

```
7717 ;FPS BEFORE EXECUTION.
```

3409 020270 007700

```
7700 ;FPS AFTER EXECUTION.
```

3410 020272 064777 177777 177777

```
.WORD 64777,-1,-1,-1 ;ERROR RES.
```

3411 020302 104051

```
ERROR +51 ;ROUND ERROR.
```

3412

3413

3414 020304

```
:MULD TEST WITH AC POSITIVE AND FSRC NEGATIVE IN ROUND MODE.
```

FFF4:

020304 104413

```
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
```

3415 020306 004737 020364

```
JSR PC,MULDSUB
```

3416 020312 017500 000000 000000

```
.WORD 17500,0,0,0 ;AC
```

3417 020322 123652 125252

```
.WORD 123652,125252 ;FSRC
```

3418 020326 125252 125252

```
.WORD 125252,125252
```

3419 020332 103177 177777 177777

```
.WORD 103177,-1,-1,-1 ;RES
```

3420 020342 000200

```
200 ;FPS BEFORE EXECUTION.
```



```

3421 020344 000210          210          :FPS AFTER EXECUTION.
3422 020346 103200 000000 000000 5$: .WORD 103200,0,0,0 :ERROR RES.
3423 020356 104052          6$: ERROR +52      :ROUND ERROR (BAD CONSTANT).

```

```

3424
3425 020360 000137 020624      JMP      FFFDONE
3426

```

```

3427      ;THIS SUBROUTINE, MULDSUB, IS CALLED TO SET UP, EXECUTE
3428      ;AND CHECK THE RESULT OF A MULD INSTRUCTION. IT IS CALLED THUS:
3429

```

```

3430      :
3431      :
3432      :
3433      :
3434      :
3435      :
3436      :
3437      :
3438      :
3439      :
3440      :
3441      :
3442      :
3443      :
3444      :
3445      :
3446      :
3447      :
3448      :
3449      :
3450      :
3451      :
3452      :

```

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

```

3440      ;THE OPERANDS ARE SET UP (USING ACO FOR THE AC OPERAND). THEN
3441      ;FPSB IS LOADED INTO THE FPS. THE INSTRUCTION, MULD IS EXECUTED.
3442      ;AFTER THE EXECUTION THE RESULT IS CHECKED AGAINST THE
3443      ;EXPECTED CORRECT RESULT, RES. IF IT IS CORRECT THEN THE FPS
3444      ;IS CHECKED WITH THE EXPECTED CORRECT FPS, FPSA. IF THE FPS WAS
3445      ;INCORRECT THEN IT IS REPORTED. IF THE RESULT WAS INCORRECT IT
3446      ;IS COMPARED WITH ERRES IN AN ATTEMPT TO ANALYSE THE ERROR. IF
3447      ;THE INCORRECT RESULT MATCHED ERRES THEN CONTROL IS PASSED TO
3448      ;THE ERROR CALL AT ERR. IF THE INCORRECT RESULT DID NOT MATCH ERRES
3449      ;THEN THE FAILURE IS REPORTED IN MULDSUB AND CONTROL IS PASSED TO
3450      ;CONT. IF NO ERRORS ARE DETECTED THEN MULDSUB RETURNS CONTROL
3451      ;TO CONT.

```

```

3452
3453 020364 012601          MULDSUB:  MOV      (SP)+,R1      ;GET A POINTER TO THE ARGUMENTS.
3454 020366 012700 000200      MOV      #200,R0      ;SET FD MODE.
3455 020372 170100          LDFPS   R0
3456
3457 020374 010100          MOV      R1,R0        ;SET UP THE ACO OPERAND.
3458 020376 172410          LDD     (R0),ACO
3459 020400 016100 000030      MOV      30(R1),R0    ;LOAD THE FPS.
3460 020404 170100          LDFPS   R0
3461
3462 020406 012737 020422 001236  MOV      #1$, $TMP2
3463 020414 010100          MOV      R1,R0        ;ESTABLISH A POINTER TO FSRC.
3464 020416 062700 000010      ADD     #10,R0
3465
3466 020422 171010          1$:    MULD   (R0),ACO    ;EXECUTE THE TEST INSTRUCTION.
3467
3468 020424 170204          STFPS  R4             ;GET THE FPS.
3469 020426 012700 000200      MOV      #200,R0      ;SET FD MODE.
3470 020432 170100          LDFPS   R0
3471
3472 020434 012700 020614      MOV      #MULDT,R0    ;GET THE RESULT.
3473 020440 174010          STD     ACO,(R0)
3474
3475 020442 010102          MOV      R1,R2        ;SAVE DATA IN CASE OF ERROR.
3476 020444 010237 001240      MOV      R2,$TMP3
3477 020450 062702 000010      ADD     #10,R2

```

```

3478 020454 010237 001242      MOV    R2,$TMP4
3479 020460 062702 000010      ADD    #10,R2
3480 020464 010237 001244      MOV    R2,$TMP5
3481 020470 012737 020614 001246      MOV    #MULDT,$TMP6
3482 020476 010437 001250      MOV    R4,$TMP7
3483 020502 016137 000032 001252      MOV    32(R1),$TMP10
3484
3485 020510 010102      MOV    R1,R2      ;CHECK THE RESULT.
3486 020512 062702 000020      ADD    #20,R2
3487 020516 012703 020614      MOV    #MULDT,R3
3488 020522 012705 000004      MOV    #4,R5
3489 020526 022223      2$:    CMP    (R2)+,(R3)+
3490 020530 001006      BNE   10$      ;BRANCH IF RESULT INCORRECT.
3491 020532 077503      SOB   R5,2$
3492
3493 020534 026104 000032      CMP    32(R1),R4  ;IS FPS CORRECT?
3494 020540 001023      BNE   15$      ;BRANCH IF INCORRECT.
3495 020542 000161 000046      JMP   46(R1)    ;RETURN.
3496
3497 020546 010102      10$:   MOV    R1,R2      ;WAS INCORRECT RESULT ANTICIPATED?
3498 020550 062702 000034      ADD    #34,R2
3499 020554 012703 020614      MOV    #MULDT,R3
3500 020560 012705 000004      MOV    #4,R5
3501 020564 022223      11$:   CMP    (R2)+,(R3)+
3502 020566 001005      BNE   12$      ;BRANCH IF NO.
3503 020570 077503      SOB   R5,11$
3504 020572 010102      MOV    R1,R2      ;IF THE INCORRECT RESULT WAS
3505 020574 062702 000044      ADD    #44,R2    ;ANTICIPATED RETURN TO THE
3506                                     ;ERROR REPORT IN THE CALLING
3507 020600 000112      JMP   (R2)      ;ROUTINE.
3508
3509 020602      12$:   ;REPORT RESULT INCORRECT.
3510 020602 104246      13$:   ERROR  +246
3511 020604 000161 000046      14$:   JMP   46(R1)
3512
3513 020610      15$:   ;REPORT FPS INCORRECT.
3514 020610 104046      16$:   ERROR  +46
3515 020612 000774      BR    14$
3516
3517 020614 000000 000000 000000 MULDT: .WORD 0,0,0,0
3518
3519 020624      FFFDONE:
      020624 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?).

3520
3521
3530
3531      ;TEST TITLE:UNDER/OVERFLOW, USING MULF WITH TRAPS DISABLED

```


3532

```
.SBTTL TEST # 12 - SEE COMMENT ABOVE FOR TEST TITLE
:*****
:*TEST 12 SEE COMMENT ABOVE FOR TEST TITLE
:*
:*THIS IS A TEST OF THE OVERFLOW AND UNDERFLOW CONDITIONS USING
:*THE MULF INSTRUCTION WITH TRAPS DISABLED. NOTE THAT A SUBROUTINE
:*IS USED TO SET UP THE OPERANDS, EXECUTE THE MULF INSTRUCTION AND
:*CHECK THE RESULTS.
:*
:*****
```

```
020626 000004
3533
3534
3535 020630
020630 104413
3536 020632 004737 021054
3537 020636 020200 000000
3538 020642 020000 000000
3539 020646 000000 000000
3540 020652 177777 177777
3541 020656 000000
3542 020660 000004
3543 020662 000012
3544 020664 177777
3545 020666 104117
3546 020670 000401
3547 020672 104114
3548 020674
```

```
TST12: SCOPE
:UNDERFLOW, WITH EXPONENT OF RESULT = -129
III1:
LPERR JSR PC,OVUNFNT ;SET UP THE LOOP ON ERROR ADDRESS.
1$: .WORD 20200,0 ;AC
2$: .WORD 20000,0 ;FSRC
3$: .WORD 0,0 ;RES
4$: .WORD -1,-1 ;ERROR RES.
5$: 0 ;FPS BEFORE EXECUTION.
4 ;FPS AFTER EXECUTION.
6$: 12 ;FEC
-1 ;FLAG
7$: ERROR +117 ;ST 331 TO 155 INTO 115 (BUT FIU)
BR 8$
ERROR +114
8$:
```

```
3549
3550
3551 020674
020674 104413
3552 020676 004737 021054
3553 020702 010200 000000
3554 020706 010000 000000
3555 020712 000000 000000
3556 020716 010000 000000
3557 020722 005013
3558 020724 005004
3559 020726 000012
3560 020730 177777
3561 020732 104120
3562
3563 020734 000401
3564 020736 104114
3565 020740
3566
3567
```

```
:UNDERFLOW, WITH EXPONENT OF RESULT = -193
III2:
LPERR JSR PC,OVUNFNT ;SET UP THE LOOP ON ERROR ADDRESS.
1$: .WORD 10200,0 ;AC
2$: .WORD 10000,0 ;FSRC
3$: .WORD 0,0 ;RES
4$: .WORD 10000,0 ;ERROR RES.
5$: 5013 ;FPS BEFORE EXECUTION.
5004 ;FPS AFTER EXECUTION.
6$: 12 ;FEC
-1 ;FLAG
7$: ERROR +120 ;SETTING FIUV OR FIV CAUSES TRAP
BR 8$ ;WITH FIU CLEAR.
ERROR +114
8$:
```

```
3568 020740
020740 104413
3569 020742 004737 021054
3570 020746 060200 000000
3571 020752 060000 000000
3572 020756 000000 000000
3573 020762 060000 000000
3574 020766 000000
3575 020770 000006
```

```
:OVERFLOW, EXPONENT OF RESULT = 128
III3:
LPERR JSR PC,OVUNFNT ;SET UP THE LOOP ON ERROR ADDRESS.
1$: .WORD 60200,0 ;AC
2$: .WORD 60000,0 ;FSRC
3$: .WORD 0,0 ;RES
4$: .WORD 60000,0 ;ERROR RES.
5$: 0 ;FPS BEFORE EXECUTION.
6 ;FPS AFTER EXECUTION.
```

3576 020772 000010
 3577 020774 000000
 3578 020776 104121
 3579 021000 000401
 3580 021002 104113
 3581 021004
 3582
 3583
 3584 021004 104413
 021004 104413
 3585 021006 004737 021054
 3586 021012 060200 000000
 3587 021016 060200 000000
 3588 021022 000000 000000
 3589 021026 177777 177777
 3590 021032 006011
 3591 021034 006006
 3592 021036 000010
 3593 021040 000000
 3594 021042 104122
 3595
 3596 021044 000401
 3597 021046 104113
 3598 021050 000137 021464
 3599
 3600
 3601
 3602
 3603
 3604
 3605
 3606
 3607
 3608
 3609
 3610
 3611
 3612
 3613
 3614
 3615
 3616
 3617
 3618
 3619
 3620
 3621
 3622
 3623
 3624
 3625
 3626
 3627
 3628
 3629
 3630
 3631

6\$: 10 ;FEC
 0 ;FLAG
 7\$: ERROR +121 ;ST 333 TO 136 INTO 116 (BUT FIV).
 BR 8\$
 ERROR +113
 8\$:
 ;OVERFLOW, EXPONENT OF RESULT = 130
 III4:
 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
 JSR PC,OVUNFNT
 1\$: .WORD 60200,0 ;AC
 2\$: .WORD 60200,0 ;FSRC
 3\$: .WORD 0,0 ;RES
 4\$: .WORD -1,-1 ;ERROR RES.
 5\$: 6011 ;FPS BEFORE EXECUTION.
 6006 ;FPS AFTER EXECUTION.
 6\$: 10 ;FEC
 0 ;FLAG
 7\$: ERROR +122 ;SETTING FIUV OR FIU WITH
 ;FIV CLEAR CAUSES TRAP.
 BR 8\$
 ERROR +113
 8\$: JMP IIIDONE ;GO TO NEXT TEST.

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

```

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

;THE OPERANDS ARE SET UP (USING ACO AS THE ACCUMULATOR). THEN
 ;THE MULF INSTRUCTION IS EXECUTED. IF NO TRAP OCCURS THEN THE
 ;RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
 ;COMPARED WITH FPSA IF THIS TOO IS CORRECT OVUNFNT RETURNS CONTROL
 ;TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD OVUNFNT
 ;REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE
 ;MULF IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
 ;ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
 ;THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN OVUNFNT
 ;WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR2. OTHERWISE THE
 ;RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND OVUNFNT WILL
 ;REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.

3632
 3633
 3634
 3635
 3636
 3637
 3638
 3639
 3640 021054 012601
 3641 021056 012700 000200
 3642 021062 170100
 3643
 3644 021064 010100
 3645 021066 172410
 3646
 3647 021070 010102
 3648 021072 010237 001240
 3649 021076 062702 000004
 3650 021102 010237 001242
 3651 021106 062702 000004
 3652 021112 010237 001244
 3653 021116 016137 000022 001252
 3654 021124 012737 021454 001246
 3655
 3656 021132 016100 000020
 3657 021136 170100
 3658 021140 012737 021162 001236
 3659 021146 012737 021346 000244
 3660
 3661 021154 010100
 3662 021156 062700 000004
 3663
 3664 021162 171010
 3665
 3666 021164 170204
 3667 021166 170305
 3668 021170 012700 000200
 3669 021174 170100
 3670 021176 012700 021454
 3671 021202 174010
 3672 021204 010437 001250
 3673 021210 010537 001254
 3674
 3675 021214 012700 021454
 3676 021220 010102
 3677 021222 062702 000010
 3678 021226 012703 000002
 3679 021232 022022
 3680 021234 001015
 3681 021236 077303
 3682
 3683 021240 026104 000022
 3684 021244 001002
 3685
 3686 021246 000161 000036
 3687
 3688

```

:IF A TRAP OCCURS (IT SHOULD NOT) THEN OVUNFNT WILL READ THE FEC.
:SHOULD THE FEC MATCH THE ANTICIPATED FEC OVUNFNT WILL
:STORE ALL DATA AND TRANSFER CONTROL TO THE ERROR CALL AT ERR1. IF THE
:FEC IS NOT THE SAME AS THE ANTICIPATED FEC OVUNFNT WILL REPORT
:THE ERROR AND RETURN TO CONT. NOTE THAT OVUNFNT USES THE FLAG
:TO TELL WHETHER OR NOT THESE PARTICULAR OPERANDS WILL RESULT IN
:UNDERFLOW (FLAG=-1) OR OVERFLOW (FLAG=0).

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

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

              MOV      R1,R2       ;SAVE THE DATA PATTERNS IN CASE OF
              MOV      R2,$TMP3     ;ERROR.
              ADD      #4,R2
              MOV      R2,$TMP4
              ADD      #4,R2
              MOV      R2,$TMP5
              MOV      22(R1),$TMP10
              MOV      #OVFNNT,$TMP6

              MOV      20(R1),R0    ;LOAD THE FPS.
              LDFPS    R0
              MOV      #1,$$TMP2
              MOV      #25$,FPVECT ;SET UP THE FP TRAP VECTOR IN CASE
              ;OF ERROR.
              MOV      R1,R0       ;COMPUTE THE ADDRESS OF FSRC.
              ADD      #4,R0

3$:           MULF      (R0),ACO    ;TEST INSTRUCTION.

2$:           STFPS    R4          ;GET FPS.
              STST     R5          ;GET FEC.
              MOV      #200,R0     ;SET FD MODE.
              LDFPS    R0
              MOV      #OVFNNT,R0  ;GET THE RESULT.
              STD      ACO,(R0)
              MOV      R4,$TMP7
              MOV      R5,$TMP11

              MOV      #OVFNNT,R0  ;CHECK THE RESULT.
              MOV      R1,R2
              ADD      #10,R2
              MOV      #2,R3
3$:           CMP      (R0)+,(R2)+ ;BRANCH IF INCORRECT.
              BNE      15$
              SOB      R3,3$

              CMP      22(R1),R4   ;WAS FPS CORRECT?
              BNE      10$         ;BRANCH IF FPS IS INCORRECT.

4$:           JMP      36(R1)      ;RETURN, TEST COMPLETED.

:REPORT INCORRECT FPS.
    
```

3689	021252	005761	000026	10\$:	TST	26(R1)	:WAS THE RESULT OVER OR UNDER FLOW?
3690	021256	001002			BNE	12\$:BRANCH IF UNDERFLOW.
3691							
3692							:REPORT FPS BAD AFTER OVERFLOW.
3693	021260	104111		11\$:	ERROR	+111	
3694	021262	000771			BR	4\$	
3695							
3696	021264			12\$:			:REPORT FPS BAD AFTER UNDERFLOW.
3697	021264	104112		13\$:	ERROR	+112	
3698	021266	000767			BR	4\$	
3699							


```

3701                                     :RESULT INCORRECT.
3702 021270 012700 021454 15$: MOV #OVFNTR,R0 ;SEE IF FAILURE IS ANTICIPATED
3703 021274 010102 MOV R1,R2 ;FAILURE.
3704 021276 062702 000014 ADD #14,R2
3705 021302 012703 000002 MOV #2,R3
3706 021306 022022 16$: CMP (R0)+,(R2)+ ;BRANCH IF NOT ANTICIPATED.
3707 021310 001007 BNE 17$
3708 021312 077303 SOB R3,16$
3709
3710 021314 010102 MOV R1,R2 ;ERROR WAS ANTICIPATED SO RETURN
3711 021316 062702 000034 ADD #34,R2 ;TO THE ERROR REPORT IN THE CALLING
3712 021322 010237 001236 MOV R2,$TMP2 ;ROUTINE.
3713 021326 000112 JMP (R2)
3714
3715 021330 005761 000026 17$: TST 26(R1) ;RESULT WAS NOT ANTICIPATED
3716 ;SO ERROR MUST BE REPORTED HERE.
3717 ;FIRST SEE IF ARGUMENTS SHOULD
3718 ;HAVE RESULTED IN OVERFLOW OR UNDER
3719 ;FLOW BY LOOKING AT THE FLAG.
3720 021334 001002 BNE 19$ ;BRANCH IF UNDERFLOW EXPECTED.
3721
3722 ;REPORT RESULT INCORRECT, EXPECTING
3723 021336 104113 18$: ERROR +113 ;OVERFLOW.
3724 021340 000742 BR 4$
3725
3726 021342 19$: ;REPORT RESULT INCORRECT, EXPECTING
3727 021342 104114 20$: ERROR +114 ;UNDERFLOW.
3728 021344 000740 BR 4$
3729
3730 ;IF AN FP TRAP OCCURS COME HERE.
3731 021346 011602 25$: MOV (SP),R2 ;GET ADDRESS OF TRAP.
3732 021350 022702 021164 CMP #2$,R2 ;WAS THE TRAP DURING THE MULF INSTRUCTION?
3733 021354 001402 BEQ 26$ ;BRANCH IF YES.
3734 021356 000137 037000 JMP FPSPUR ;OTHERWISE GO REPORT A SPURIOUS
3735 ;FP TRAP.
3736 021362 022626 26$: CMP (SP)+,(SP)+ ;RESET THE STACK.
3737 021364 010237 001236 MOV R2,$TMP2 ;SAVE DATA FOR ERROR REPORT.
3738 021370 170204 STFPS R4 ;GET FPS.
3739 021372 170305 STST R5 ;GET FEC.
3740 021374 012700 000200 MOV #200,R0 ;SET FD MODE.
3741 021400 170100 LDFPS R0
3742 021402 012700 021454 MOV #OVFNTR,R0 ;GET THE RESULT.
3743 021406 174010 STD ACO,(R0)
3744 021410 010537 001254 MOV R5,$TMP11
3745 021414 020561 000024 CMP R5,24(R1) ;WAS THE FEC ANTICIPATED?
3746 021420 001004 BNE 27$ ;BRANCH IF NOT ANTICIPATED.
3747
3748 021422 010102 MOV R1,R2 ;ERROR WAS ANTICIPATED SO
3749 021424 062702 000030 ADD #30,R2 ;RETURN TO THE ERROR REPORT OF THE
3750 ;CALLING ROUTINE.
3751 021430 000112 JMP (R2)
3752
3753 021432 005761 000026 27$: TST 26(R1) ;THE ERROR WAS NOT ANTICIPATED SO
3754 ;IT MUST BE REPORTED HERE. FIRST SEE IF EXPECTED
3755 ;OVERFLOW OR UNDER FLOW.
3756 021436 001003 BNE 29$ ;BRANCH IF EXPECTING UNDERFLOW
3757
    
```

TEST # 12 - SEE COMMENT ABOVE FOR TEST TITLE

```

3758
3759 021440 104115      28$:  ERROR  +115
3760 021442 000161 000036  JMP    36(R1)
3761
3762 021446  -----  29$:
3763 021446 104116  30$:  ERROR  +116
3764 021450 000161 000036  JMP    36(R1)
3765
3766 021454 000000 000000 000000 OVFNTT: .WORD 0,0,0,0
3767
3768 021464      IIIDONE:
      021464 104412      RSETUP

```

:REPORT TRAPPED ON OVERFLOW WITH FIV=0

:REPORT TRAPPED ON UNDER FLOW WITH FIU=0

```

: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?).

```

```

3769
3770
3771
3780
3781

```

:TEST TITLE:UNDER/OVERFLOW, USING MULD WITH TRAP DISABLED

3782

```
.SBTTL TEST # 13 - SEE COMMENT ABOVE FOR TEST TITLE
:*****
:TEST 13 SEE COMMENT ABOVE FOR TEST TITLE
:
:*THIS IS A TEST OF THE OVERFLOW AND UNDERFLOW CONDITIONS THAT CAN
:*ARRISE USING THE MULD INSTRUCTION WITH TRAPS DISABLED. A SUBROUTINE IS
:*USED TO SET UP THE OPERANDS, EXECUTE THE MULD INSTRUCTION AND
:*CHECK THE RESULTS.
:
:*****
```

```
021466 000004
3783
3784
3785 021470
021470 104413
3786 021472 004737 022014
3787 021476 020200 000000
3788 021502 127272 000000
3789 021506 020000 000000 000000
3790 021516 000000 000000 000000
3791 021526 000000 000000
3792 021532 127272 000000
3793 021536 000200
3794 021540 000204
3795 021542 000012
3796 021544 177777
3797 021546 104131
3798 021550 000401
3799 021552 104132
3800 021554
3801
3802
```

```
TST13: SCOPE
:UNDERFLOW, EXPONENT OF RESULT=-129
JJJ1:
LPERR JSR PC,OVUNDNT ;SET UP THE LOOP ON ERROR ADDRESS.
1$: .WORD 20200,0 ;AC
.WORD 127272,0
2$: .WORD 20000,0,0,0 ;FSRC
3$: .WORD 0,0,0,0 ;RES
4$: .WORD 0,0 ;ERROR RES.
.WORD 127272,0
5$: 200 ;FPS BEFORE EXECUTION.
204 ;FPS AFTER EXECUTION.
6$: 12 ;FEC
-1 ;FLAG
7$: ERROR +131 ;ST 331 TO 155 INTO 115 (BUT FIU)
BR 8$
ERROR +132 ;ST 115 (BUT FD)
8$:
```

```
3803 021554
021554 104413
3804 021556 004737 022014
3805 021562 010200 000000
3806 021566 123456 000000
3807 021572 010000 000000 000000
3808 021602 000000 000000 000000
3809 021612 000000 000000 123456
3810 021622 005213
3811 021624 005204
3812 021626 000012
3813 021630 177777
3814 021632 104133
3815 021634 000401
3816 021636 104132
3817 021640
3818
3819
```

```
:UNDERFLOW, EXPONENT OF RESULT = -193
JJJ2:
LPERR JSR PC,OVUNDNT ;SET UP THE LOOP ON ERROR ADDRESS.
1$: .WORD 10200,0 ;AC
.WORD 123456,0
2$: .WORD 10000,0,0,0 ;FSRC
3$: .WORD 0,0,0,0 ;RES
4$: .WORD 0,0,123456,0 ;ERROR RES
5$: 5213 ;FPS BEFORE EXECUTION.
5204 ;FPS AFTER EXECUTION.
6$: 12 ;FEC
-1 ;FLAG
7$: ERROR +133 ;SETTING FIUV OR FIV BAD.
BR 8$
ERROR +132 ;ST 115 (BUT FD)
8$:
```

```
3820 021640
021640 104413
3821 021642 004737 022014
3822 021646 060200 000000
3823 021652 065432 000000
3824 021656 060000 000000 000000
3825 021666 000000 000000 000000
```

```
:OVERFLOW, EXPONENT OF RESULT = 128
JJJ3:
LPERR JSR PC,OVUNDNT ;SET UP THE LOOP ON ERROR ADDRESS.
1$: .WORD 60200,0 ;AC
.WORD 65432,0
2$: .WORD 60000,0,0,0 ;FSRC
3$: .WORD 0,0,0,0 ;RES
```

```

3826 021676 000000 000000 065432 4$: .WORD 0,0,65432,0 ;ERROR RES.
3827 021706 000200 5$: 200 ;FPS BEFORE EXECUTION.
3828 021710 000206 ;206 ;FPS AFTER EXECUTION.
3829 021712 000010 6$: 10 ;FEC
3830 021714 000000 0 ;FLAG
3831 021716 104134 7$: ERROR +134 ;ST 333 TO 136 INTO 116 (BUT FIV)
3832 021720 000401 8$: BR 8$
3833 021722 104135 ;ERROR +135 ;ST 116 (BUT FD)
3834 021724 8$:
3835
3836
3837
3838
    
```

;OVERFLOW, EXPONENT OF RESULT = 130
 JJJ4:

```

3839 021724 104413 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
3839 021726 004737 022014 JSR PC,OVUNDNT
3840 021732 060200 000000 1$: .WORD 60200,0 ;AC
3841 021736 125252 000000 .WORD 125252,0
3842 021742 060200 000000 000000 2$: .WORD 60200,0,0,0 ;FSRC
3843 021752 000000 000000 000000 3$: .WORD 0,0,0,0 ;RES
3844 021762 000000 000000 125252 4$: .WORD 0,0,125252,0 ;ERROR RES.
3845 021772 006211 5$: 6211 ;FPS BEFORE EXECUTION.
3846 021774 006206 ;6206 ;FPS AFTER EXECUTION.
3847 021776 000010 6$: 10 ;FEC
3848 022000 000000 0 ;FLAG
3849 022002 104136 7$: ERROR +136 ;SETTING FIUV OR FIV BAD.
3850 022004 000401 8$: BR 8$
3851 022006 104135 ;ERROR +135 ;ST 116 (BUT FD)
3852 022010 000137 022424 8$: JMP JJJDONE ;GO TO NEXT TEST.
3853
3854
3855
3856
3857
3858
3859
3860
3861
3862
3863
3864
3865
3866
3867
3868
3869
3870
3871
3872
3873
3874
3875
3876
3877
3878
3879
3880
3881
    
```

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

```

ACARG: .WORD X,X,X,X ;AC OPERAND
FSRCARG: .WORD X,X,X,X ;FSRC 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
FLAG: .WORD X ;0/-1,OVER/UNDER FLOW FLAG
ERR1: ERROR +X ;TRAP ERROR.
BR CONT
ERR2: ERROR +X ;DATA, RESULT ERROR
CONT: ;RETURN ADDRESS
    
```

;THE OPERANDS ARE SET UP (USING ACO AS THE ACCUMULATOR). THEN
 ;THE MULD INSTRUCTION IS EXECUTED. IF NO TRAP OCCURS THEN THE
 ;RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
 ;COMPARED WITH FPSA IF THIS TOO IS CORRECT OVUNDNT RETURNS CONTROL
 ;TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD OVUNDNT
 ;REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE
 ;MULD 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 OVUNDNT
 ;WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR2. OTHERWISE THE

3882
 3883
 3884
 3885
 3886
 3887
 3888
 3889
 3890
 3891
 3892 022014 012601
 3893 022016 012700 000200
 3894 022022 170100
 3895
 3896 022024 010100
 3897 022026 172410
 3898
 3899 022030 010102
 3900 022032 010237 001240
 3901 022036 062702 000010
 3902 022042 010237 001242
 3903 022046 062702 000010
 3904 022052 010237 001244
 3905 022056 016137 000042 001252
 3906 022064 012737 022414 001246
 3907
 3908 022072 016100 000040
 3909 022076 170100
 3910 022100 012737 022122 001236
 3911 022106 012737 022306 000244
 3912
 3913 022114 010100
 3914 022116 062700 000010
 3915
 3916 022122 171010
 3917
 3918 022124 170204
 3919 022126 170305
 3920 022130 012700 000200
 3921 022134 170100
 3922 022136 012700 022414
 3923 022142 174010
 3924 022144 010437 001250
 3925 022150 010537 001254
 3926
 3927 022154 012700 022414
 3928 022160 010102
 3929 022162 062702 000020
 3930 022166 012703 000004
 3931 022172 022022
 3932 022174 001015
 3933 022176 077303
 3934
 3935 022200 026104 000042
 3936 022204 001002
 3937
 3938 022206 000161 000056

:RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND OVUNDNT WILL
 :REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.
 :IF A TRAP OCCURS (IT SHOULD NOT) THEN OVUNDNT WILL READ THE FEC.
 :SHOULD THE FEC MATCH THE ANTICIPATED FEC OVUNDNT WILL
 :STORE ALL DATA AND TRANSFER CONTROL TO THE ERROR CALL AT ERR1. IF THE
 :FEC IS NOT THE SAME AS THE ANTICIPATED FEC OVUNDNT WILL REPORT
 :THE ERROR AND RETURN TO CONT. NOTE THAT OVUNDNT USES THE FLAG
 :TO TELL WHETHER OR NOT THESE PARTICULAR OPERANDS WILL RESULT IN
 :UNDERFLOW (FLAG=-1) OR OVERFLOW (FLAG=0).

```

OVUNDNT:      MOV      (SP)+,R1      ;GET A POINTER TO THE ARGUMENTS.
              MOV      #200,R0      ;SET FD MODE.
              LDFPS   R0
              MOV      R1,R0        ;LOAD ACO, OPERAND.
              LDD     (R0),ACO
              MOV      R1,R2        ;SAVE THE DATA PATTERNS IN CASE OF
              MOV      R2,$TMP3     ;ERROR.
              ADD     #10,R2
              MOV      R2,$TMP4
              ADD     #10,R2
              MOV      R2,$TMP5
              MOV      42(R1),$TMP10
              MOV      #OVDNTT,$TMP6
              MOV      40(R1),R0     ;LOAD THE FPS.
              LDFPS   R0
              MOV      #1,$TMP2
              MOV      #25$,FPVECT  ;SET UP THE FP TRAP VECTOR IN CASE
              ;OF ERROR.
              MOV      R1,R0        ;COMPUTE THE ADDRESS OF FSRC.
              ADD     #10,R0
              1$:    MULD   (R0),ACO  ;TEST INSTRUCTION.
              2$:    STFPS  R4        ;GET FPS.
                   STST  R5        ;GET FEC.
                   MOV   #200,R0    ;SET FD MODE.
                   LDFPS  R0
                   MOV   #OVDNTT,R0 ;GET THE RESULT.
                   STD   ACO,(R0)
                   MOV   R4,$TMP7
                   MOV   R5,$TMP11
                   MOV   #OVDNTT,R0 ;CHECK THE RESULT.
                   MOV   R1,R2
                   ADD   #20,R2
                   MOV   #4,R3
                   3$:    CMP   (R0)+,(R2)+ ;BRANCH IF INCORRECT.
                   BNE  15$
                   SOB  R3,3$
                   CMP   42(R1),R4 ;WAS FPS CORRECT?
                   BNE  10$         ;BRANCH IF FPS IS INCORRECT.
                   4$:    JMP   56(R1) ;RETURN, TEST COMPLETED.
    
```

```

3939
3940          ;REPORT INCORRECT FPS.
3941 022212 005761 000046 10$:  TST 46(R1)      ;WAS THE RESULT OVER OR UNDER FLOW?
3942 022216 001002          BNE 12$      ;BRANCH IF UNDERFLOW.
3943
3944          ;REPORT FPS BAD AFTER OVERFLOW.
3945 022220 104123 11$:  ERROR +123
3946 022222 000771          BR 4$
3947
3948 022224 12$:
3949 022224 104124 13$:  ERROR +124
3950 022226 000767          BR 4$
3951
3952          ;RESULT INCORRECT.
3953 022230 012700 022414 15$:  MOV #OVDNTT,R0      ;SEE IF FAILURE IS ANTICIPATED
3954 022234 010102          MOV R1,R2          ;FAILURE.
3955 022236 062702 000030          ADD #30,R2
3956 022242 012703 000004          MOV #4,R3
3957 022246 022022 16$:  CMP (R0)+,(R2)+
3958 022250 001007          BNE 17$      ;BRANCH IF NOT ANTICIPATED.
3959 022252 077303          SOB R3,16$
3960
3961 022254 010102          MOV R1,R2      ;ERROR WAS ANTICIPATED SO RETURN
3962 022256 062702 000054          ADD #54,R2      ;TO THE ERROR REPORT IN THE CALLING
3963 022262 010237 001236          MOV R2,$TMP2    ;ROUTINE.
3964 022266 000112          JMP (R2)
3965
3966 022270 005761 000046 17$:  TST 46(R1)      ;RESULT WAS NOT ANTICIPATED
3967          ;SO ERROR MUST BE REPORTED HERE.
3968          ;FIRST SEE IF ARGUMENTS SHOULD
3969          ;HAVE RESULTED IN OVERFLOW OR UNDER
3970          ;FLOW BY LOOKING AT THE FLAG.
3971 022274 001002          BNE 19$      ;BRANCH IF UNDERFLOW EXPECTED.
3972
3973          ;REPORT RESULT INCORRECT, EXPECTING
3974 022276 104125 18$:  ERROR +125
3975 022300 000742          BR 4$      ;OVERFLOW.
3976
3977 022302 19$:
3978 022302 104126 20$:  ERROR +126
3979 022304 000740          BR 4$      ;REPORT RESULT INCORRECT, EXPECTING
3980          ;UNDERFLOW.
3981
3982          ;IF AN FP TRAP OCCURS COME HERE.
3983 022306 011602 25$:  MOV (SP),R2      ;GET ADDRESS OF TRAP.
3984 022310 022702 022124          CMP #2$,R2      ;WAS THE TRAP DURING THE MULF INSTRUCTION?
3985 022314 001402          BEQ 26$      ;BRANCH IF YES.
3986 022316 000137 037000          JMP FPSPUR ;OTHERWISE GO REPORT A SPURIOUS
3987          ;FP TRAP.
3988 022322 022626 26$:  CMP (SP)+,(SP)+
3989 022324 010237 001236          MOV R2,$TMP2    ;RESET THE STACK.
3990 022330 170204          STFPS R4        ;SAVE DATA FOR ERROR REPORT.
3991 022332 170305          STST R5        ;GET FPS.
3992 022334 012700 000200          MOV #200,R0     ;GET FEC.
3993 022340 170100          LDFPS R0        ;SET FD MODE.
3994 022342 012700 022414          MOV #OVDNTT,R0
3995 022346 174010          STD ACO,(R0)
3995 022350 010537 001254          MOV R5,$TMP11
    
```



```
3996 022354 020561 000044      CMP      R5,44(R1)      ;WAS THE FEC ANTICIPATED?  
3997 022360 001004      BNE      27$           ;BRANCH IF NOT ANTICIPATED.  
3998  
3999 022362 010102      MOV      R1,R2        ;ERROR WAS ANTICIPATED SO  
4000 022364 062702 000050      ADD      #50,R2       ;RETURN TO THE ERROR REPORT OF THE  
4001                               ;CALLING ROUTINE.  
4002 022370 000112      JMP      (R2)  
4003  
4004 022372 005761 000026      27$:    TST      26(R1)  ;THE ERROR WAS NOT ANTICIPATED SO  
4005                               ;IT MUST BE REPORTED HERE. FIRST SEE IF EXPECTED  
4006                               ;OVERFLOW OR UNDER FLOW.  
4007 022376 001003      BNE      29$           ;BRANCH IF EXPECTING UNDERFLOW  
4008  
4009                               ;REPORT TRAPPED ON OVERFLOW WITH FIV=0  
4010 022400 104127      28$:    ERROR    +127  
4011 022402 000161 000056      JMP      56(R1)  
4012                               ;REPORT TRAPPED ON UNDER FLOW WITH FIU=0  
4013 022406      29$:  
4014 022406 104130      30$:    ERROR    +130  
4015 022410 000161 000056      JMP      56(R1)  
4016  
4017 022414 000000 000000 000000 OVDNNT: .WORD 0,0,0,0  
4018  
4019 022424      JJJDONE:  
      022424 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?).
```

4020
4021
4022
4034
4035

;TEST TITLE:UNDER/OVERFLOW, USING MULF WITH TRAPS ENABLED

4036

```
.SBTTL TEST # 14 - SEE COMMENT ABOVE FOR TEST TITLE
:*****
:*TEST 14 SEE COMMENT ABOVE FOR TEST TITLE
:*
:*THIS IS A TEST OF THE UNDERFLOW AND OVERFLOW
:*CONDITIONS THAT CAN OCCUR USING THE MULF INSTRUCTION.
:*A SUBROUTINE IS CALLED TO SET UP THE OPERANDS,
:* EXECUTE THE MULF INSTRUCTION AND CHECK
:*THE RESULTS. HERE THE PARTICULAR INTERRUPT,
:*EITHER OVERFLOW OR UNDERFLOW, IS ENABLED SO A TRAP SHOULD
:*OCCUR.
:*
:*****
TST14: SCOPE
```

```
022426 000004
4037
4038
4039 022430
022430 104413
4040 022432 004737 022654
4041 022436 020123 045676
4042 022442 020200 000000
4043 022446 000123 045676
4044 022452 177777 177777
4045 022456 002000
4046 022460 102004
4047 022462 000012
4048 022464 177777
4049 022466 104145
4050 022470 000401
4051 022472 104144
4052 022474
4053
4054
4055 022474
022474 104413
4056 022476 004737 022654
4057 022502 010127 127272
4058 022506 010200 000000
4059 022512 060127 127272
4060 022516 177777 177777
4061 022522 007017
4062 022524 107000
4063 022526 000012
4064 022530 177777
4065 022532 104146
4066 022534 000401
4067 022536 104144
4068 022540
4069
4070
4071 022540
022540 104413
4072 022542 004737 022654
4073 022546 060252 125252
4074 022552 060000 000000
4075 022556 000052 125252
4076 022562 177777 177777
```

```
;UNDERFLOW, EXPONENT OF RESULT = -129
KKK1:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,OVUNFT
1$: .WORD 20123,45676 ;AC
2$: .WORD 20200,0 ;FSRC
3$: .WORD 123,45676 ;RES
4$: .WORD -1,-1 ;ERROR RES.
5$: 2000 ;FPS BEFORE EXECUTION.
102004 ;FPS AFTER EXECUTION.
6$: 12 ;FEC
-1 ;FLAG
7$: ERROR +145 ;ST 331 (BUT FIU) NO TRAP.
BR 8$
ERROR +144
8$:

;UNDERFLOW, EXPONENT OF THE RESULT = -193
KKK3:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,OVUNFT
1$: .WORD 10127,127272 ;AC
2$: .WORD 10200,0 ;FSRC
3$: .WORD 60127,127272 ;RES
4$: .WORD -1,-1 ;ERROR RES.
5$: 7017 ;FPS BEFORE EXECUTION.
107000 ;FPS AFTER EXECUTION.
6$: 12 ;FEC
-1
7$: ERROR +146 ;ST 137 (BUT FIU) NO TRAP.
BR 8$
ERROR +144
8$:

;OVERFLOW, EXPONENT OF THE RESULT = 128
KKK4:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,OVUNFT
1$: .WORD 60252,125252 ;AC
2$: .WORD 60000,0 ;FSRC
3$: .WORD 000052,125252 ;RES
4$: .WORD -1,-1 ;ERROR RES.
```


4077 022566 001000
 4078 022570 101006
 4079 022572 000010
 4080 022574 000000
 4081 022576 104147
 4082 022600 000401
 4083 022602 104143
 4084 022604
 4085
 4086
 4087 022604 104413
 4088 022606 004737 022654
 4089 022612 060345 067654
 4090 022616 060200 000000
 4091 022622 000345 067654
 4092 022626 177777 177777
 4093 022632 007015
 4094 022634 107002
 4095 022636 000010
 4096 022640 000000
 4097 022642 104150
 4098 022644 000401
 4099 022646 104143
 4100 022650 000137 023266
 4101
 4102
 4103
 4104
 4105
 4106
 4107
 4108
 4109
 4110
 4111
 4112
 4113
 4114
 4115
 4116
 4117
 4118
 4119
 4120
 4121
 4122
 4123
 4124
 4125
 4126
 4127
 4128
 4129
 4130
 4131
 4132

5\$: 1000 ;FPS BEFORE EXECUTION.
 101006 ;FPS AFTER EXECUTION.
 6\$: 10 ;FEC
 0 ;FLAG
 7\$: ERROR +147 ;ST 333 (BUT FIV) NO TRAP
 BR 8\$
 ERROR +143
 8\$:
 ;OVERFLOW, EXPONENT OF RESULT = 130
 KKK5:
 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
 JSR PC_OVUNFT
 1\$: .WORD 60345,67654 ;AC
 2\$: .WORD 60200,0 ;FSRC
 3\$: .WORD 345,67654 ;RES
 4\$: .WORD -1,-1 ;ERROR RES.
 5\$: 7015 ;FPS BEFORE EXECUTION.
 107002 ;FPS AFTER EXECUTION.
 6\$: 10 ;FEC
 0 ;FLAG
 7\$: ERROR +150 ;ST 133 (BUT FIV) NO TRAP
 BR 8\$
 ERROR +143
 8\$: JMP KKKDONE

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

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

;THE OPERANDS ARE SET UP (USING ACO AS THE ACCUMULATOR). THEN
 ;THE MULF INSTRUCTION IS EXECUTED. IF THE TRAP OCCURS THEN THE
 ;RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
 ;COMPARED WITH FPSA IF THIS TOO IS CORRECT OVUNFT RETURNS CONTROL
 ;TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD OVUNFT
 ;REPORTS THIS FAILURE AND THEN RETURNS TO CONT. THE FEC IS TREATED
 ;IN THE SAME WAY. IF THE RESULT OF THE
 ;MULF IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
 ;ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
 ;THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN OVUNFT
 ;WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR2. OTHERWISE THE
 ;RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND OVUNFT WILL
 ;REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.

```

4133                                     :IF NO TRAP OCCURS CONTROL IS PASSED TO ERR1.
4134                                     :NOTE THAT OVUNFT USES THE FLAG
4135                                     :TO TELL WHETHER OR NOT THESE PARTICULAR OPERANDS WILL RESULT IN
4136                                     :UNDERFLOW (FLAG=-1) OR OVERFLOW (FLAG=0).
4137
4138 022654 012601 OVUNFT: MOV (SP)+,R1 ;GET A POINTER TO THE ARGUMENTS.
4139 022656 012700 000200 MOV #200,R0 ;SET FD MODE.
4140 022662 170100 LDFPS R0
4141
4142 022664 010100 MOV R1,R0 ;LOAD ACO, OPERAND.
4143 022666 172410 LDD (R0),ACO
4144
4145 022670 010102 MOV R1,R2 ;SAVE THE DATA PATTERNS IN CASE OF
4146 022672 010237 001240 MOV R2,$TMP3 ;ERROR.
4147 022676 062702 000004 ADD #4,R2
4148 022702 010237 001242 MOV R2,$TMP4
4149 022706 062702 000004 ADD #4,R2
4150 022712 010237 001244 MOV R2,$TMP5
4151 022716 016137 000022 001252 MOV 22(R1),$TMP10
4152 022724 012737 023256 001246 MOV #OVFTT,$TMP6
4153
4154 022732 016100 000020 MOV 20(R1),R0 ;LOAD THE FPS.
4155 022736 170100 LDFPS R0
4156 022740 012737 022762 001236 MOV #1$,$TMP2
4157 022746 012737 022772 000244 MOV #50$,FPVECT ;SET UP THE FP TRAP VECTOR IN CASE
4158                                     ;OF ERROR.
4159 022754 010100 MOV R1,R0 ;COMPUTE THE ADDRESS OF FSRC.
4160 022756 062700 000004 ADD #4,R0
4161
4162 022762 171010 1$: MULF (R0),ACO ;TEST INSTRUCTION. SHOULD CAUSE TRAP.
4163 022764 170000 2$: CFCC
4164
4165 022766 000137 023216 JMP 25$ ;FAILURE, NO TRAP.
4166
4167 022772 011602 50$: MOV (SP),R2 ;TRAP TO HERE AND SEE IF THE PC OF THE
4168 022774 020227 022764 CMP R2,#2$ ;TRAP WAS THAT OF THE MULF INSTRUCTION.
4169 023000 001402 BEQ 51$ ;BRANCH IF YES.
4170 023002 000137 037000 JMP FPSPUR ;OTHERWISE REPORT SPURIOUS FP ERROR.
4171
4172 023006 022626 51$: CMP (SP)+,(SP)+ ;RESET THE STACK
4173 023010 170204 STFPS R4 ;GET FPS.
4174 023012 170305 STST R5 ;GET FEC.
4175 023014 012700 000200 MOV #200,R0 ;SET FD MODE.
4176 023020 170100 LDFPS R0
4177 023022 012700 023256 MOV #OVFTT,R0 ;GET THE RESULT.
4178 023026 174010 STD ACO,(R0)
4179 023030 010437 001250 MOV R4,$TMP7
4180 023034 010537 001254 MOV R5,$TMP11
4181
4182 023040 012700 023256 MOV #OVFTT,R0 ;CHECK THE RESULT.
4183 023044 010102 MOV R1,R2
4184 023046 062702 000010 ADD #10,R2
4185 023052 012703 000002 MOV #2,R3
4186 023056 022022 3$: CMP (R0)+,(R2)+ ;BRANCH IF INCORRECT.
4187 023060 001027 BNE 15$
4188 023062 077303 SOB R3,3$
4189
    
```


TEST # 14 - SEE COMMENT ABOVE FOR TEST TITLE

4190	023064	026104	000022		CMP	22(R1),R4		:WAS FPS CORRECT?
4191	023070	001014			BNE	10\$:BRANCH IF FPS IS INCORRECT.
4192								
4193	023072	026105	000024		CMP	24(R1),R5		:IS FEC CORRECT?
4194	023076	001002			BNE	5\$:IF INCORRECT BRANCH.
4195	023100	000161	000036	4\$:	JMP	36(R1)		:RETURN, TEST COMPLETED.
4196								
4197					:REPORT INCORRECT FEC.			
4198	023104	005761	000026	5\$:	TST	26(R1)		:WAS THE RESULT OVERFLOW OR UNDERFLOW?
4199	023110	001002			BNE	7\$:BRANCH IF UNDERFLOW.
4200								:REPORT BAD FEC ON EXPECTED OVERFLOW.
4201								
4202	023112	104137		6\$:	ERROR	+137		
4203	023114	000771			BR	4\$		
4204								:REPORT BAD FEC ON EXPECTED UNDERFLOW.
4205	023116			7\$:				
4206	023116	104140		8\$:	ERROR	+140		
4207	023120	000767			BR	4\$		
4208								
4209					:REPORT INCORRECT FPS.			
4210	023122	005761	000026	10\$:	TST	26(R1)		:WAS THE RESULT OVER OR UNDER FLOW?
4211	023126	001002			BNE	12\$:BRANCH IF UNDERFLOW.
4212								:REPORT FPS BAD AFTER OVERFLOW.
4213								
4214	023130	104141		11\$:	ERROR	+141		
4215	023132	000762			BR	4\$		
4216								:REPORT FPS BAD AFTER UNDERFLOW.
4217	023134			12\$:				
4218	023134	104142		13\$:	ERROR	+142		
4219	023136	000760			BR	4\$		
4220								
4221					:RESULT INCORRECT.			
4222	023140	012700	023256	15\$:	MOV	#OVFTT,R0		:SEE IF FAILURE IS ANTICIPATED
4223	023144	010102			MOV	R1,R2		:FAILURE.
4224	023146	062702	000014		ADD	#14,R2		
4225	023152	012703	000002		MOV	#2,R3		
4226	023156	022022		16\$:	CMP	(R0)+,(R2)+		:BRANCH IF NOT ANTICIPATED.
4227	023160	001007			BNE	17\$		
4228	023162	077303			SOB	R3,16\$		
4229								:ERROR WAS ANTICIPATED SO RETURN
4230	023164	010102			MOV	R1,R2		:TO THE ERROR REPORT IN THE CALLING
4231	023166	062702	000034		ADD	#34,R2		:ROUTINE.
4232	023172	010237	001236		MOV	R2,\$TMP2		
4233	023176	000112			JMP	(R2)		
4234								:RESULT WAS NOT ANTICIPATED
4235	023200	005761	000026	17\$:	TST	26(R1)		:SO ERROR MUST BE REPORTED HERE.
4236								:FIRST SEE IF ARGUMENTS SHOULD
4237								:HAVE RESULTED IN OVERFLOW OR UNDER
4238								:FLOW BY LOOKING AT THE FLAG.
4239								:BRANCH IF UNDERFLOW EXPECTED.
4240	023204	001002			BNE	19\$		
4241								:REPORT RESULT INCORRECT, EXPECTING
4242								:OVERFLOW.
4243	023206	104143		18\$:	ERROR	+143		
4244	023210	000733			BR	4\$		
4245								:REPORT RESULT INCORRECT, EXPECTING
4246	023212			19\$:				

```
4247 023212 104144          20$:  ERROR  +144          ;UNDERFLOW.
4248 023214 000731          BR      4$
4249
4250          ;IF NO FP TRAP OCCURS COME HERE.
4251 023216 170204          25$:  STFPS  R4          ;GET FPS.
4252 023220 170305          STST   R5          ;GET FEC.
4253 023222 012700 000200  MOV    #200,R0     ;SET FD MODE.
4254 023226 170100          LDFPS  R0
4255 023230 012700 023256  MOV    #OVFTT,R0   ;GET THE RESULT.
4256 023234 174010          STD    ACO,(R0)
4257 023236 010437 001250  MOV    R4,$TMP7
4258 023242 010537 001254  MOV    R5,$TMP11
4259 023246 010102          MOV    R1,R2
4260 023250 062702 000030  ADD    #30,R2      ;ERROR WAS ANTICIPATED SO
                                        ;RETURN TO THE ERROR REPORT OF THE
                                        ;CALLING ROUTINE.
4261
4262 023254 000112          JMP    (R2)
4263
4264 023256 000000 000000 000000 OVFTT: .WORD 0,0,0,0
4265
4266 023266 104412          KKKDONE: 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?).

4267
4268
4269
4277
4278          ;TEST TITLE:UNDER/OVERFLOW, USING MULD WITH TRAPS ENABLED
```


4279

```
.SBTTL TEST # 15 - SEE COMMENT ABOVE FOR TEST TITLE
:*****
:*TEST 15 SEE COMMENT ABOVE FOR TEST TITLE
:*
:*THIS IS A TEST OF THE OVER FLOW AND UNDER FLOW CONDITIONS USING THE
:*MULD INSTRUCTION WITH TRAPS ENABLED. A SUBROUTINE IS USED TO SET UP
:*THE OPERANDS, EXECUTE THE MULD INSTRUCTION AND CHECK THE RESULTS.
:*
:*****
TST15: SCOPE
```

```
023270 000004
4280
4281
4282 023272
023272 104413
4283 023274 004737 023616
4284 023300 020052 125252
4285 023304 125252 125252
4286 023310 020300 000000 000000
4287 023320 000177 177777 177777
4288 023330 000177 177777
4289 023334 125252 125252
4290 023340 002200
4291 023342 102204
4292 023344 000012
4293 023346 177777
4294 023350 104157
4295 023352 000401
4296 023354 104160
4297 023356
4298
4299
4300 023356
023356 104413
4301 023360 004737 023616
4302 023364 010327 127272
4303 023370 036363 045454
4304 023374 010000 000000 000000
4305 023404 060127 127272
4306 023410 036363 045454
4307 023414 177777 177777 177777
4308 023424 007217
4309 023426 107200
4310 023430 000012
4311 023432 177777
4312 023434 104161
4313 023436 000401
4314 023440 104156
4315 023442
4316
4317
4318 023442
023442 104413
4319 023444 004737 023616
4320 023450 060252 125252
4321 023454 125252 125252
4322 023460 160100 000000 000000
4323 023470 100177 177777 177777
```

```
:UNDERFLOW, EXPONENT OF RESULT = -129
LLL1:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,OVUNDT
1$: .WORD 20052,125252 ;AC
.WORD 125252,125252
2$: .WORD 20300,0,0,0 ;FSRC
3$: .WORD 177,-1,-1,-1 ;RES
4$: .WORD 177,-1 ;ERROR RES.
.WORD 125252,125252
5$: 2200 ;FPS BEFORE EXECUTION.
102204 ;FPS AFTER EXECUTION.
6$: 12 ;FEC
-1 ;FLAG
7$: ERROR +157 ;ST 331 (BUT FIU) NO TRAP.
BR 8$
ERROR +160 ;ST 155 (BUT FD)
8$:
```

```
:UNDERFLOW, EXPONENT OF THE RESULT = -193
LLL2:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,OVUNDT
1$: .WORD 10327,127272 ;AC
.WORD 36363,45454
2$: .WORD 10000,0,0,0 ;FSRC
3$: .WORD 60127,127272 ;RES
.WORD 36363,45454
4$: .WORD -1,-1,-1,-1 ;ERROR RES.
5$: 7217 ;FPS BEFORE EXECUTION.
107200 ;FPS AFTER EXECUTION.
6$: 12 ;FEC
-1 ;FLAG
7$: ERROR +161 ;ST 137 (BUT FIU) NO TRAP.
BR 8$
ERROR +156
8$:
```

```
:OVERFLOW, EXPONENT OF THE RESULT = 128
LLL3:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,OVUNDT
1$: .WORD 60252,125252 ;AC
.WORD 125252,125252 ;FSRC
2$: .WORD 160100,0,0,0 ;FSRC
3$: .WORD 100177,-1,-1,-1 ;RES
```

```

4324 023500 100177 177777 4$: .WORD 100177,-1 ;ERROR RES.
4325 023504 125252 125252 .WORD 125252,125252
4326 023510 001200 5$: 1200 ;FPS BEFORE EXECUTION.
4327 023512 101216 101216 ;FPS AFTER EXECUTION.
4328 023514 000010 6$: 10 ;FEC
4329 023516 000000 0 ;FLAG
4330 023520 104162 7$: ERROR +162 ;ST 333 (BUT FIV) NO TRAP.
4331 023522 000401 BR 8$
4332 023524 104163 ERROR +163 ;ST 700 (BUT FD).
4333 023526 8$:
4334
4335 ;OVERFLOW, EXPONENT OF THE RESULT = 130
4336 023526 LLL4: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
023526 104413 JSR PC,OVUNDT
4337 023530 004737 023616 1$: .WORD 60345,67654 ;AC
4338 023534 060345 067654 .WORD 56765,45676
4339 023540 056765 045676 2$: .WORD 60200,0,0,0 ;FSRC
4340 023544 060200 000000 000000 3$: .WORD 345,67654 ;RES
4341 023554 000345 067654 .WORD 56765,45676
4342 023560 056765 045676 4$: .WORD -1,-1,-1,-1 ;ERROR RES.
4343 023564 177777 177777 177777 5$: 7215 ;FPS BEFORE EXECUTION.
4344 023574 007215 107202 ;FPS AFTER EXECUTION.
4345 023576 107202 6$: 10 ;FEC
4346 023600 000010 0 ;FLAG
4347 023602 000000 7$: ERROR +164 ;ST 133 (BUT FIV) NO TRAP
4348 023604 104164 BR 8$
4349 023606 000401 ERROR +155
4350 023610 104155 JMP LLLDONE
4351 023612 000137 024230 8$:
4352
4353 ;THIS SUBROUTINE, OVUNDT, IS USED TO SET UP THE OPERANDS, EXECUTE
4354 ;THE MULD INSTRUCTION AND CHECK THE RESULTS OF AN INSTRUCTION WITH
4355 ;OPERANDS WHICH SHOULD RESULT IN EITHER OVERFLOW OR UNDERFLOW. A CALL
4356 ;TO IT IS MADE THUS:
4357
4358 ; ACARG: .WORD X,X,X,X ;AC OPERAND
    
```


4360	:	FSRCARG: .WORD	X,X,X,X	:FSRC OPERAND
4361	:	RES: .WORD	X,X,X,X	:EXPECTED RESULT
4362	:	ERRES: .WORD	X,X,X,X	:ERROR RESULT
4363	:	FPSB: .WORD	X	:FPS BEFORE EXECUTION
4364	:	FPSA: .WORD	X	:FPS AFTER EXECUTION
4365	:	FEC: .WORD	X	:EXPECTED FEC
4366	:	FLAG: .WORD	X	:0/-1,OVER/UNDER FLOW FLAG
4367	:	ERR1: ERROR	+X	:TRAP ERROR.
4368	:	BR	CONT	
4369	:	ERR2: ERROR	+X	:DATA, RESULT ERROR
4370	:	CONT:		:RETURN ADDRESS

4371
 4372 :THE OPERANDS ARE SET UP (USING ACO AS THE ACCUMULATOR). THEN
 4373 :THE MULD INSTRUCTION IS EXECUTED. IF THE TRAP OCCURS THEN THE
 4374 :RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
 4375 :COMPARED WITH FPSA IF THIS TOO IS CORRECT OVUNDT RETURNS CONTROL
 4376 :TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD OVUNDT
 4377 :REPORTS THIS FAILURE AND THEN RETURNS TO CONT. THE FEC IS TREATED
 4378 :IN THE SAME WAY. IF THE RESULT OF THE
 4379 :MULF IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
 4380 :ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
 4381 :THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN OVUNDT
 4382 :WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR2. OTHERWISE THE
 4383 :RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND OVUNDT WILL
 4384 :REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.
 4385 :IF NO TRAP OCCURS CONTROL IS PASSED TO ERR1.
 4386 :NOTE THAT OVUNDT USES THE FLAG
 4387 :TO TELL WHETHER OR NOT THESE PARTICULAR OPERANDS WILL RESULT IN
 4388 :UNDERFLOW (FLAG=-1) OR OVERFLOW (FLAG=0).
 4389

4390	023616	012601		OVUNDT: MOV	(SP)+,R1	:GET A POINTER TO THE ARGUMENTS.
4391	023620	012700	000200	MOV	#200,R0	:SET FD MODE.
4392	023624	170100		LDFPS	R0	
4393						
4394	023626	010100		MOV	R1,R0	:LOAD ACO, OPERAND.
4395	023630	172410		LDD	(R0),ACO	
4396						
4397	023632	010102		MOV	R1,R2	:SAVE THE DATA PATTERNS IN CASE OF
4398	023634	010237	001240	MOV	R2,\$TMP3	:ERROR.
4399	023640	062702	000010	ADD	#10,R2	
4400	023644	010237	001242	MOV	R2,\$TMP4	
4401	023650	062702	000010	ADD	#10,R2	
4402	023654	010237	001244	MOV	R2,\$TMP5	
4403	023660	016137	000042 001252	MOV	42(R1),\$TMP10	
4404	023666	012737	024220 001246	MOV	#OVDTT,\$TMP6	
4405						
4406	023674	016100	000040	MOV	40(R1),R0	:LOAD THE FPS.
4407	023700	170100		LDFPS	R0	
4408	023702	012737	023724 001236	MOV	#1\$,\$TMP2	
4409	023710	012737	023734 000244	MOV	#50\$,FPVECT	:SET UP THE FP TRAP VECTOR IN CASE
4410						:OF ERROR.
4411	023716	010100		MOV	R1,R0	:COMPUTE THE ADDRESS OF FSRC.
4412	023720	062700	000010	ADD	#10,R0	
4413						
4414	023724	171010		1\$: MULD	(R0),ACO	:TEST INSTRUCTION. SHOULD CAUSE TRAP.
4415	023726	170000		2\$: CFCC		
4416						

```

4417 023730 000137 024160          JMP      25$          ;FAILURE, NO TRAP.
4418
4419 023734 011602          50$:    MOV      (SP),R2      ;TRAP TO HERE AND SEE IF THE PC OF THE
4420 023736 020227 023726          CMP      R2,#2$        ;TRAP WAS THAT OF THE MULF INSTRUCTION.
4421 023742 001402          BEQ      51$          ;BRANCH IF YES.
4422 023744 000137 037000          JMP      FPSPUR ;OTHERWISE REPORT SPURIOUS FP ERROR.
4423
4424 023750 022626          51$:    CMP      (SP)+,(SP)+    ;RESET THE STACK
4425 023752 170204          STFPS   R4            ;GET FPS.
4426 023754 170305          STST    R5            ;GET FEC.
4427 023756 012700 000200          MOV      #200,R0      ;SET FD MODE.
4428 023762 170100          LDFPS   R0            ;
4429 023764 012700 024220          MOV      #OVDTT,R0    ;GET THE RESULT.
4430 023770 174010          STD     ACO,(R0)
4431 023772 010437 001250          MOV      R4,$TMP7
4432 023776 010537 001254          MOV      R5,$TMP11
4433
4434 024002 012700 024220          MOV      #OVDTT,R0    ;CHECK THE RESULT.
4435 024006 010102          MOV      R1,R2
4436 024010 062702 0C0020          ADD     #20,R2
4437 024014 012703 000004          MOV      #4,R3
4438 024020 022022          3$:    CMP      (R0)+,(R2)+    ;
4439 024022 001027          BNE     15$          ;BRANCH IF INCORRECT.
4440 024024 077303          SOB     R3,3$
4441
4442 024026 026104 000042          CMP      42(R1),R4    ;WAS FPS CORRECT?
4443 024032 001014          BNE     10$          ;BRANCH IF FPS IS INCORRECT.
4444
4445 024034 026105 000044          CMP      44(R1),R5    ;IS FEC CORRECT?
4446 024040 001002          BNE     5$           ;IF INCORRECT BRANCH.
4447 024042 000161 000056          4$:    JMP      56(R1)      ;RETURN. TEST COMPLETED.
4448
4449          ;REPORT INCORRECT FEC.
4450 024046 005761 000046          5$:    TST     46(R1)      ;WAS THE RESULT OVERFLOW OR UNDERFLOW?
4451 024052 001002          BNE     7$           ;BRANCH IF UNDERFLOW.
4452
4453          ;REPORT BAD FEC ON EXPECTED OVERFLOW.
4454 024054 104151          6$:    ERROR  +151
4455 024056 000771          BR      4$
4456
4457 024060          7$:
4458 024060 104152          8$:    ERROR  +152
4459 024062 000767          BR      4$
4460
4461          ;REPORT INCORRECT FPS.
4462 024064 005761 000046          10$:   TST     46(R1)      ;WAS THE RESULT OVER OR UNDER FLOW?
4463 024070 001002          BNE     12$          ;BRANCH IF UNDERFLOW.
4464
4465          ;REPORT FPS BAD AFTER OVERFLOW.
4466 024072 104153          11$:   ERROR  +153
4467 024074 000762          BR      4$
4468
4469 024076          12$:
4470 024076 104154          13$:   ERROR  +154
4471 024100 000760          BR      4$
4472
4473          ;RESULT INCORRECT.

```



```

4474 024102 012700 024220      15$:  MOV    #OVDTT,R0      ;SEE IF FAILURE IS ANTICIPATED
4475 024106 010102              MOV    R1,R2          ;FAILURE.
4476 024110 062702 000030      ADD    #30,R2
4477 024114 012703 000004      MOV    #4,R3
4478 024120 022022      16$:  CMP    (R0)+,(R2)+    ;BRANCH IF NOT ANTICIPATED.
4479 024122 001007              BNE    17$
4480 024124 077303              SOB    R3,16$
4481
4482 024126 010102              MOV    R1,R2          ;ERROR WAS ANTICIPATED SO RETURN
4483 024130 062702 000054      ADD    #5,R2          ;TO THE ERROR REPORT IN THE CALLING
4484 024134 010237 001236      MOV    R2,$TMP2      ;ROUTINE.
4485 024140 000112              JMP    (R2)
4486
4487 024142 005761 000046      17$:  TST    46(R1)        ;RESULT WAS NOT ANTICIPATED
4488                          ;SO ERROR MUST BE REPORTED HERE.
4489                          ;FIRST SEE IF ARGUMENTS SHOULD
4490                          ;HAVE RESULTED IN OVERFLOW OR UNDER
4491                          ;FLOW BY LOOKING AT THE FLAG.
4492 024146 001002              BNE    19$           ;BRANCH IF UNDERFLOW EXPECTED.
4493
4494                          ;REPORT RESULT INCORRECT, EXPECTING
4495 024150 104155      18$:  ERROR  +155         ;OVERFLOW.
4496 024152 000733              BR     4$
4497
4498 024154      19$:                          ;REPORT RESULT INCORRECT, EXPECTING
4499 024154 104156      20$:  ERROR  +156         ;UNDERFLOW.
4500 024156 000731              BR     4$
4501
4502                          ;IF NO FP TRAP OCCURS COME HERE.
4503 024160 170204      25$:  STFPS  R4            ;GET FPS.
4504 024162 170305              STST  R5            ;GET FEC.
4505 024164 012700 000200      MOV    #200,R0       ;SET FD MODE.
4506 024170 170100              LDFPS R0
4507 024172 012700 024220      MOV    #OVDTT,R0     ;GET THE RESULT.
4508 024176 174010              STD   ACO,(R0)
4509 024200 010437 001250      MOV    R4,$TMP7
4510 024204 010537 001254      MOV    R5,$TMP11
4511 024210 010102              MOV    R1,R2
4512 024212 062702 000050      ADD    #50,R2
4513                          ;ERROR WAS ANTICIPATED SO
4514 024216 000112              JMP    (R2)          ;RETURN TO THE ERROR REPORT OF THE
4515                          ;CALLING ROUTINE.
4516 024220 000000 000000 000000 OVDTT: .WORD 0,0,0,0
4517
4518 024230      LLLDONE:
         024230 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?).

```

4519
 4520
 4521
 4522
 4530

4531

```
.SBTTL TEST # 16 - MODF TEST
:*****
:*TEST 16      MODF TEST
:*
:*THIS IS A TEST OF THE MODF INSTRUCTION, WHICH MAKES USE OF
:*A SUBROUTINE TO SET UP THE OPERANDS, EXECUTE THE MODF INSTRUCTION
:*AND CHECK THE RESULTS.
:*
:*****
```

```
024232 000004
4532
4533
4534 024234
024234 104413
4535 024236 004737 025320
4536 024242 000000 000000
4537 024246 000000 000000
4538 024252 000000 000000
4539 024256 000000 000000
4540 024262 177777 177777
4541 024266 177777 177777
4542 024272 000013
4543 024274 000004
4544 024276 104056
4545 024300 000401
4546 024302 104057
4547 024304
4548
4549
4550 024304
024304 104413
4551 024306 004737 025320
4552 024312 123456 076543
4553 024316 000000 000000
4554 024322 000000 000000
4555 024326 000000 000000
4556 024332 123456 076543
4557 024336 177777 177777
4558 024342 000000
4559 024344 000004
4560 024346 104056
4561 024350 000401
4562 024352 104057
4563 024354
4564
4565
4566 024354
024354 104413
4567 024356 004737 025320
4568 024362 000000 000000
4569 024366 076543 021234
4570 024372 000000 000000
4571 024376 000000 000000
4572 024402 000000 000000
4573 024406 177777 177777
4574 024412 000003
4575 024414 000004
```

```
TST16: SCOPE
:MODF WITH (FSRC=AC=0)
GGG1:
LPERR JSR PC,MODFSUB ;SET UP THE LOOP ON ERROR ADDRESS.
1$: .WORD 0,0 ;AC
2$: .WORD 0,0 ;FSRC
3$: .WORD 0,0 ;FRACTIONAL RES.
4$: .WORD 0,0 ;INTEGER RES.
5$: .WORD -1,-1 ;ERROR FRACTIONAL RES.
6$: .WORD -1,-1 ;ERROR INGETER RES.
7$: 13 ;FPS BEFORE EXECUTION.
4 ;FPS AFTER EXECUTION.
8$: ERROR +56 ;STORE SINGLE ZERO BAD.
BR 9$
ERROR +57 ;AC V 1 <= ZERO FAILED.
9$:

:MODF TEST, WITH (FSRC=0)
GGG2:
LPERR JSR PC,MODFSUB ;SET UP THE LOOP ON ERROR ADDRESS.
1$: .WORD 123456,76543 ;AC
2$: .WORD 0,0 ;FSRC
3$: .WORD 0,0 ;FRACTIONAL RES.
4$: .WORD 0,0 ;INTEGER RESULT.
5$: .WORD 123456,76543 ;ERROR FRACTIONAL RES.
6$: .WORD -1,-1 ;ERROR INTEGER RES.
7$: 0 ;FPS BEFORE EXECUTION.
4 ;FPS AFTER EXECUTION.
8$: ERROR +56 ;STORE ZERO FAILURE.
BR 9$
ERROR +57
9$:

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


4576 024416 104053
 4577 024420 000401
 4578 024422 104057
 4579 024424
 4580
 4581
 4582 024424
 024424 104413
 4583 024426 004737 025320
 4584 024432 046252 125252
 4585 024436 040300 000000
 4586 024442 000000 000000
 4587 024446 046377 177777
 4588 024452 046252 125252
 4589 024456 040300 000000
 4590 024462 000013
 4591 024464 000004
 4592 024466 104053
 4593 024470 000401
 4594 024472 104060
 4595 024474
 4596
 4597
 4598 024474
 024474 104413
 4599 024476 004737 025320
 4600 024502 077652 125252
 4601 024506 040300 000000
 4602 024512 000000 000000
 4603 024516 077777 177777
 4604 024522 077652 125252
 4605 024526 040300 000000
 4606 024532 000000
 4607 024534 000004
 4608 024536 104053
 4609 024540 000401
 4610 024542 104060
 4611 024544
 4612
 4613
 4614 024544
 024544 104413
 4615 024546 004737 025320
 4616 024552 046200 000001
 4617 024556 040340 000000
 4618 024562 000000 000000
 4619 024566 046340 000001
 4620 024572 040000 000000
 4621 024576 177777 177777
 4622 024602 000013
 4623 024604 000004
 4624 024606 104061
 4625
 4626 024610 000401
 4627 024612 104054
 4628 024614
 4629

```

8$:  ERROR +53 ;RES.BAD
      BR 9$
      ERROR +57
9$:
;MODF TEST WITH EXPONENT OF THE RESULT = 25
GGG4:
      LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
      JSR PC,MODFSUB
1$: .WORD 46252,125252 ;AC
2$: .WORD 40300,0 ;FSRC
3$: .WORD 0,0 ;FRACTIONAL RES.
4$: .WORD 46377,-1 ;INTEGER RES.
5$: .WORD 46252,125252 ;ERROR FRACTIONAL RES.
6$: .WORD 40300,0 ;ERROR INTEGER RES.
7$: 13 ;FPS BEFORE EXECUTION.
      4 ;FPS AFTER EXECUTION.
8$:  ERROR +53 ;ST 134
      BR 9$
      ERROR +60
9$:
;MODF TEST WITH EXPONENT OF THE RESULT = 127
GGG5:
      LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
      JSR PC,MODFSUB
1$: .WORD 77652,125252 ;AC
2$: .WORD 40300,0 ;FSRC
3$: .WORD 0,0 ;FRACTIONAL RES.
4$: .WORD 77777,-1 ;INTEGER RES.
5$: .WORD 77652,125252 ;ERROR FRACTIONAL RES.
6$: .WORD 40300,0 ;ERROR INTEGER RES.
7$: 0 ;FPS BEFORE EXECUTION.
      4 ;FPS AFTER EXECUTION.
8$:  ERROR +53
      BR 9$
      ERROR +60
9$:
;MODF TEST WITH EXPONENT OF RESULT = 25
GGG6:
      LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
      JSR PC,MODFSUB
1$: .WORD 46200,1 ;AC
2$: .WORD 40340,0 ;FSRC
3$: .WORD 0,0 ;FRACTIONAL RES.
4$: .WORD 46340,1 ;INTEGER RES.
5$: .WORD 40000,0 ;ERROR FRACTIONAL RES.
6$: .WORD -1,-1 ;ERROR INTEGER RES.
7$: 13 ;FPS BEFORE EXECUTION.
      4 ;FPS AFTER EXECUTION.
8$:  ERROR +61 ;BAD CONSTANT (NOT 24).
      BR 9$ ;OR ST 525 TO 050 INTO 150.
      ERROR +54
9$:
    
```

4630
4631 024614
024614 104413
4632 024616 004737 025320
4633 024622 046000 000001
4634 024626 040340 000000
4635 024632 040100 000000
4636 024636 046140 000001
4637 024642 000000 000000
4638 024646 177777 177777
4639 024652 000000
4640 024654 000000
4641 024656 104062
4642
4643 024660 000401
4644 024662 104054
4645 024664
4646
4647
4648 024664
024664 104413
4649 024666 004737 025320
4650 024672 042577 177777
4651 024676 040200 000000
4652 024702 040177 176000
4653 024706 042577 140000
4654 024712 177777 177777
4655 024716 177777 177777
4656 024722 000000
4657 024724 000000
4658 024726 104053
4659 024730 000401
4660 024732 104054
4661 024734
4662
4663
4664 024734
024734 104413
4665 024736 004737 025320
4666 024742 042577 140001
4667 024746 040200 000000
4668 024752 034600 000000
4669 024756 042577 140000
4670 024762 000000 000000
4671 024766 177777 177777
4672 024772 000000
4673 024774 000000
4674 024776 104063
4675 025000 000401
4676 025002 104054
4677 025004
4678
4679
4680 025004
025004 104413
4681 025006 004737 025320
4682 025012 042377 100000

:MODF TEST WITH EXPONENT OF THE RESULT = 24
GGG7: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,MODFSUB
1\$: .WORD 46000,1 ;AC
2\$: .WORD 40340,0 ;FSRC
3\$: .WORD 40100,0 ;FRACTIONAL RES.
4\$: .WORD 46140,1 ;INTEGER RESULT.
5\$: .WORD 0,0 ;ERROR FRACTIONAL RES.
6\$: .WORD -1,-1 ;ERROR INTEGER RES.
7\$: 0 ;FPS BEFORE EXECUTION.
0 ;FPS AFTER EXECUTION.
8\$: ERROR +62 ;BAD CONSTANT USED (NOT 24)
BR 9\$;OR ST 525 TO 150 INTO 050
ERROR +54
9\$:

:MODF TEST WITH EXPONENT OF THE RESULT = 10
GGG8: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,MODFSUB
1\$: .WORD 42577,-1 ;AC
2\$: .WORD 40200,0 ;FSRC
3\$: .WORD 40177,176000 ;FRACTIONAL RES.
4\$: .WORD 42577,140000 ;INTEGER RES.
5\$: .WORD -1,-1 ;ERROR FRACTIONAL RES.
6\$: .WORD -1,-1 ;ERROR INTEGER RES.
7\$: 0 ;FPS BEFORE EXECUTION.
0 ;FPS AFTER EXECUTION.
8\$: ERROR +53
BR 9\$
ERROR +54
9\$:

:MODF TEST WITH THE EXPONENT OF THE RESULT = 10
GGG9: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,MODFSUB
1\$: .WORD 42577,140001 ;AC
2\$: .WORD 40200,0 ;FSRC
3\$: .WORD 34600,0 ;FRACTIONAL RES.
4\$: .WORD 42577,140000 ;INTEGER RES.
5\$: .WORD 0,0 ;ERROR FRACTIONAL RES.
6\$: .WORD -1,-1 ;ERROR INTEGER RES.
7\$: 0 ;FPS BEFORE EXECUTION.
0 ;FPS AFTER EXECUTION.
8\$: ERROR +63 ;ST 532 TO 122 INTO NORMALIZE.
BR 9\$
ERROR +54
9\$:

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

4683 025016 040200 000000
4684 025022 000000 000000
4685 025026 042377 100000
4686 025032 177777 177777
4687 025036 177777 177777
4688 025042 000013
4689 025044 000004
4690 025046 104053
4691 025050 000401
4692 025052 104054
4693 025054
4694
4695

2\$: .WORD 40200,0 :FSRC
3\$: .WORD 0,0 :FRACTIONAL RES.
4\$: .WORD 42377,100000 :INTEGER RES.
5\$: .WORD -1,-1 :ERROR FRACTIONAL RES.
6\$: .WORD -1,-1 :ERROR INTEGER RES.
7\$: 13 :FPS BEFORE EXECUTION.
4 :FPS AFTER EXECUTION.
8\$: ERROR +53
BR 9\$
ERROR +54
9\$:

:MODF TEST WITH EXPONENT OF THE RESULT = 0

GGG11:
4696 025054 104413
025054 004737 025320
4697 025056 040177 177777
4698 025062 040177 177777
4699 025066 040200 000000
4700 025072 040177 177777
4701 025076 000000 000000
4702 025102 000000 000000
4703 025106 040177 177777
4704 025112 000017
4705 025114 000000
4706 025116 104064
4707 025120 000401
4708 025122 104064
4709 025124
4710
4711

LPERR :SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,MODFSUB
1\$: .WORD 40177,-1 :AC
2\$: .WORD 40200,0 :FSRC
3\$: .WORD 40177,-1 :FRACTIONAL RES.
4\$: .WORD 0,0 :INTEGER RES.
5\$: .WORD 0,0 :ERROR FRACTIONAL RES.
6\$: .WORD 40177,-1 :ERROR INTEGER RES.
7\$: 17 :FPS BEFORE EXECUTION.
0 :FPS AFTER EXECUTION.
8\$: ERROR +64 :ST 041 TO 046 INTO 246.
BR 9\$
ERROR +64
9\$:

:MODF TEST WITH EXPONENT OF THE RESULT = -15

GGG12:
4712 025124 104413
025124 004737 025320
4713 025126 034377 177777
4714 025132 040200 000000
4715 025136 034377 177777
4716 025142 000000 000000
4717 025146 000000 000000
4718 025152 000000 000000
4719 025156 034377 177777
4720 025162 000000
4721 025164 000000
4722 025166 104064
4723 025170 000401
4724 025172 104064
4725 025174
4726
4727

LPERR :SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,MODFSUB
1\$: .WORD 34377,-1 :AC
2\$: .WORD 40200,0 :FSRC
3\$: .WORD 34377,-1 :FRACTIONAL RES.
4\$: .WORD 0,0 :INTEGER RES.
5\$: .WORD 0,0 :ERROR FRACTIONAL RES.
6\$: .WORD 34377,-1 :ERROR INTEGER RES.
7\$: 0 :FPS BEFORE EXECUTION.
0 :FPS AFTER EXECUTION.
8\$: ERROR +64
BR 9\$
ERROR +64
9\$:

:MODF TEST WITH EXPONENT OF RESULT = -64, IN ROUND MODE

GGG13:
4728 025174 104413
025174 004737 025320
4729 025176 020000 000001
4730 025202 040300 000000
4731 025206 020100 000002
4732 025212 000000 000000
4733 025216 020100 000001
4734 025222 000000 000000
4735 025226 000000 000000
4736 025232 000000

LPERR :SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,MODFSUB
1\$: .WORD 20000,1 :AC
2\$: .WORD 40300,0 :FSRC
3\$: .WORD 20100,2 :FRACTIONAL RES.
4\$: .WORD 0,0 :INTEGER RES.
5\$: .WORD 20100,1 :ERROR FRACTIONAL RES.
6\$: .WORD 0,0 :ERROR INTEGER RES.
7\$: 0 :FPS BEFORE EXECUTION.

4737 025234 000000
 4738 025236 104065
 4739 025240 000401
 4740 025242 104054
 4741 025244
 4742
 4743
 4744 025244 104413
 4745 025246 004737 025320
 4746 025252 142777 170000
 4747 025256 040200 000000
 4748 025262 140000 000000
 4749 025266 142777 160000
 4750 025272 040000 000000
 4751 025276 042777 160000
 4752 025302 000007
 4753 025304 000010
 4754 025306 104066
 4755 025310 000401
 4756 025312 104067
 4757 025314 000137 025706
 4758
 4759
 4760
 4761
 4762
 4763
 4764
 4765
 4766
 4767
 4768
 4769
 4770
 4771
 4772
 4773
 4774
 4775
 4776
 4777
 4778
 4779
 4780
 4781
 4782
 4783
 4784
 4785
 4786
 4787
 4788
 4789
 4790
 4791
 4792

```

0                                     ;FPS AFTER EXECUTION.
8$:  ERROR  +65                       ;ROUND TRUNK, ST 126 INTO ROUND.
    BR      9$
    ERROR  +54
9$:

;MODF TEST WITH EXPONENT OF RESULT = 11
GGG14:
    LPERR                                     ;SET UP THE LOOP ON ERROR ADDRESS.
    JSR    PC,MODFSUB
1$:  .WORD 142777,170000                   ;AC
2$:  .WORD 40200,0                         ;FSRC
3$:  .WORD 140000,0                       ;FRACTIONAL RES.
4$:  .WORD 142777,160000                  ;INTEGER RES.
5$:  .WORD 40000,0                       ;ERROR FRACTIONAL RES.
6$:  .WORD 42777,160000                  ;ERROR INTEGER RES.
7$:  7                                     ;FPS BEFORE EXECUTION.
    10                                     ;FPS AFTER EXECUTION.
8$:  ERROR  +66                           ;SIGN OF FRACTION.
    BR      9$
    ERROR  +67
9$:  JMP    GGGDONE                       ;SIGN OF INTEGER.
    GGGDONE                               ;GO TO NEXT TEST.
    
```

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

```

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

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


```

4793 025320 012601          MODFSUB:  MOV      (SP)+,R1      ;GET A POINTER TO THE ARGUMENTS
4794 025322 012700 000200      MOV      #200,R0      ;SET FD MODE.
4795 025326 170100          LDFPS   R0
4796 025330 010100          MOV      R1,R0      ;SET UP ACO
4797 025332 172410          LDD     (R0),AC0
4798 025334 012700 025676      MOV      #MODP1,R0   ;PUT A BACKGROUND PATTERN INTO AC1.
4799 025340 172510          LDD     (R0),AC1
4800 025342 016100 000030      MOV      30(R1),R0   ;SET UP THE FPS.
4801 025346 170100          LDFPS   R0
4802 025350 012737 025364 001236  MOV      #1$, $TMP2
4803 025356 010100          MOV      R1,R0      ;COMPUTE THE ADDRESS OF THE FSRC.
4804 025360 062700 000004      ADD     #4,R0
4805
4806 025364 171410          1$:   MODF     (R0),AC0   ;EXECUTE THE TEST INSTRUCTION.
4807
4808 025366 170204          STFPS   R4          ;GET THE FPS.
4809 025370 012700 000200      MOV      #200,R0      ;SET FD MODE.
4810 025374 170100          LDFPS   R0
4811 025376 012700 025656      MOV      #MODFT0,R0  ;GET THE FRACTIONAL RESULT.
4812 025402 174010          STD     ACO,(R0)
4813 025404 012700 025666      MOV      #MODFT1,R0  ;GET THE INTEGER RESULT.
4814 025410 174110          STD     AC1,(R0)
4815
4816 025412 010102          MOV     R1,R2      ;SAVE THE DATA IN CASE OF ERROR.
4817 025414 010237 001240      MOV     R2,$TMP3
4818 025420 062702 000004      ADD     #4,R2
4819 025424 010237 001242      MOV     R2,$TMP4
4820 025430 062702 000004      ADD     #4,R2
4821 025434 010237 001244      MOV     R2,$TMP5
4822 025440 062702 000004      ADD     #4,R2
4823 025444 010237 001246      MOV     R2,$TMP6
4824 025450 012737 025656 001250  MOV     #MODFT0,$TMP7
4825 025456 012737 025666 001252  MOV     #MODFT1,$TMP10
4826 025464 010437 001254      MOV     R4,$TMP11
4827 025470 016137 000032 001256  MOV     32(R1),$TMP12
4828
4829 025476 012702 025656      MOV     #MODFT0,R2   ;CHECK THE FRACTIONAL RESULT.
4830 025502 026112 000010      CMP     10(R1),(R2)
4831 025506 001022          BNE     10$         ;BRANCH IF INCORRECT.
4832 025510 026162 000012 000002  CMP     12(R1),2(R2)
4833 025516 001016          BNE     10$
4834
4835 025520 012702 025666      MOV     #MODFT1,R2   ;CHECK THE INTEGER RESULT.
4836 025524 026112 000014      CMP     14(R1),(R2)
4837 025530 001026          BNE     15$         ;BRANCH IF INCORRECT.
4838 025532 026162 000016 000002  CMP     16(R1),2(R2)
4839 025540 001022          BNE     15$
4840
4841 025542 026104 000032      CMP     32(R1),R4   ;CHECK THE FPS.
4842 025546 001034          BNE     20$         ;BRANCH IF INCORRECT.
4843
4844 025550 000161 000042          9$:   JMP     42(R1)      ;RETURN.
4845
4846          ;FRACTIONAL ERROR.
4847 025554 026112 000020          10$:  CMP     20(R1),(R2)  ;WAS THE ERROR ANTICIPATED?
4848 025560 001010          BNE     11$         ;BRANCH IF NOT ANTICIPATED.
4849 025562 026162 000022 000002  CMP     22(R1),2(R2)

```

```
4850 025570 001004          BNE      11$
4851 025572 010102          MOV      R1,R2
4852 025574 062702 000034   ADD      #34,R2          ;THE ERROR WAS ANTICIPATED SO
4853                                     ;RETURN TO THE ERROR REPORT AT THE
4854 025600 000112          JMP      (R2)           ;CALLING ROUTINE.
4855
4856 025602          11$:
4857 025602 104053          12$:  ERROR  +53
4858 025604 000761          BR      9$             ;THE ERROR WAS NOT ANTICIPATED SO
4859                                     ;REPORT THE INCORRECT FRACTION HERE.
4860          ;INTEGER ERROR.
4861 025606 026112 000024   15$:  CMP      24(R1),(R2) ;WAS THIS ERROR ANTICIPATED?
4862 025612 001010          BNE      16$           ;BRANCH IF NOT.
4863 025614 026162 000026 000002   CMP      26(R1),2(R2)
4864 025622 001004          BNE      16$
4865 025624 010102          MOV      R1,R2
4866 025626 062702 000040   ADD      #40,R2
4867                                     ;THE ERROR WAS ANTICIPATED SO RETURN
4868 025632 000112          JMP      (R2)           ;TO THE ERROR REPORT IN THE CALLING
4869                                     ;ROUTINE.
4870          16$:
4871 025634 104054          17$:  ERROR  +54
4872 025636 000744          BR      9$             ;THE ERROR WAS NOT ANTICIPATED SO REPORT
4873                                     ;THE INTEGER FAILURE HERE.
4874          ;FPS INCORRECT.
4875 025640 010437 001254   20$:  MOV      R4,$TMP11 ;REPORT INCORRECT FPS.
4876 025644 016137 000032 001256   MOV      32(R1),$TMP12
4877 025652 104055          21$:  ERROR  +55
4878 025654 000735          BR      9$
4879
4880 025656 000000 000000 000000 MODFT0: .WORD 0,0,0,0
4881
4882 025666 000000 000000 000000 MODFT1: .WORD 0,0,0,0
4883
4884 025676 177777 177777 177777 MODP1:  .WORD -1,-1,-1,-1
4885
4886 025706          GGGDONE:
      025706 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?).
```

4887
4888
4889
4897

4898

```
.SBTTL TEST # 17 - MODD TEST
:*****
:*TEST 17      MODD TEST
:*
:*THIS IS A TEST OF THE MODD INSTRUCTION. IT MAKES USE OF A SUBROUTINE
:*TO SET UP THE ARGUMENTS, EXECUTE THE INSTRUCTION AND CHECK THE
:*RESULTS.
:*
:*****
```

4899 025710 000004

TST17: SCOPE

4900

;MODD WITH (FSRC=AC=0)

4901

HHH1:

025712

LPERR JSR PC,MODDSUB

;SET UP THE LOOP ON ERROR ADDRESS.

025712 104413

4902 025714 004737 027416

.WORD 0,0,0,0

;AC

4903 025720 000000 000000 000000

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

;FSRC

4904 025730 000000 000000 000000

2\$: .WORD 0,0,0,0

;FRACTIONAL RES.

4905 025740 000000 000000 000000

3\$: .WORD 0,0,0,0

;INTEGER RES.

4906 025750 000000 000000 000000

4\$: .WORD 0,0,0,0

;ERROR FRACTIONAL RES.

4907 025760 000000 000000 000000

5\$: .WORD 0,0,0,0

;ERROR INTEGER RES.

4908 025770 000000 000000 177777

6\$: .WORD 0,0,-1,-1

;FPS BEFORE EXECUTION.

4909 026000 000200

7\$: 200

;FPS AFTER EXECUTION.

4910 026002 000204

204

4911 026004 104070

8\$: ERROR +70

4912 026006 000401

BR 9\$

4913 026010 104074

9\$: ERROR +74

;ST 231 TO 142 INTO 143

4914 026012

4915

4916

;MODD TEST WITH FSRC=0

4917

HHH2:

026012

LPERR JSR PC,MODDSUB

;SET UP THE LOOP ON ERROR ADDRESS.

026012 104413

4918 026014 004737 027416

.WORD 012345,67012

;AC

4919 026020 012345 067012

1\$: .WORD 34567,012345

4920 026024 034567 012345

.WORD 0,0,0,0

;FSRC

4921 026030 000000 000000 000000

2\$: .WORD 0,0,0,0

;FRACTIONAL RES.

4922 026040 000000 000000 000000

3\$: .WORD 0,0,0,0

;INTEGER RES.

4923 026050 000000 000000 000000

4\$: .WORD 0,0,0,0

;ERROR FRACTIONAL RES.

4924 026060 012345 067012

5\$: .WORD 012345,67012

;ERROR INTEGER RES.

4925 026064 034567 012345

.WORD 34567,012345

;FPS BEFORE EXECUTION.

4926 026070 177777 177777 177777

6\$: .WORD -1,-1,-1,-1

;FPS AFTER EXECUTION.

4927 026100 000213

7\$: 213

;STORE DOUBLE ZERO

4928 026102 000204

204

4929 026104 104075

8\$: ERROR +75

4930 026106 000401

BR 9\$

4931 026110 104076

9\$: ERROR +76

;AC V 1 <= ZERO ST 143

4932 026112

4933

4934

;MODD TEST WITH (AC=0)

4935

HHH3:

026112

LPERR JSR PC,MODDSUB

;SET UP THE LOOP ON ERROR ADDRESS.

026112 104413

4936 026114 004737 027416

.WORD 0,0,0,0

;AC

4937 026120 000000 000000 000000

1\$: .WORD 72727,127272

;FSRC

4938 026130 072727 127272

2\$: .WORD 72727,127272

;FRACTIONAL RES.

4939 026134 072727 127272

.WORD 0,0,0,0

;INTEGER RES.

4940 026140 000000 000000 000000

3\$: .WORD 0,0,0,0

;ERROR FRACTIONAL RES.

4941 026150 000000 000000 000000

4\$: .WORD 0,0,0,0

4942 026160 177777 177777 177777

5\$: .WORD -1,-1,-1,-1

```

4943 026170 177777 177777 177777 6$: .WORD -1,-1,-1,-1 ;ERROR INTEGER RES.
4944 026200 000213 7$: 213 ;FPS BEFORE EXECUTION.
4945 026202 000204 204 ;FPS AFTER EXECUTION.
4946 026204 104070 8$: ERROR +70
4947 026206 000401 BR 9$
4948 026210 104071 ERROR +71
4949 026212 9$:
4950
4951 ;MODD TEST WITH EXPONENT OF THE RESULT = 57
4952 026212 HHH4:
      026212 104413 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
4953 026214 004737 027416 JSR PC,MODDSUB
4954 026220 056252 125252 1$: .WORD 56252,125252 ;AC
4955 026224 125252 125250 .WORD 125252,125250
4956 026230 040300 000000 000000 2$: .WORD 40300,0,0,0 ;FSRC
4957 026240 000000 000000 000000 3$: .WORD 0,0,0,0 ;FRACTIONAL RES.
4958 026250 056377 177777 177777 4$: .WORD 56377,-1,-1,-4 ;INTEGER RES.
4959 026260 000000 000000 5$: .WORD 0,0 ;ERROR FRACTIONAL RES.
4960 026264 125252 125252 .WORD 125252,125252
4961 026270 056377 177777 177777 6$: .WORD 56377,-1,-1,-1 ;ERROR INTEGER RES.
4962 026300 000213 7$: 213 ;FPS BEFORE EXECUTION.
4963 026302 000204 204 ;FPS AFTER EXECUTION.
4964 026304 104077 8$: ERROR +77 ;ST 526 TO 134 INTO 135
4965 026306 000401 BR 9$
4966 026310 104077 ERROR +77
4967 026312 9$:
4968
4969 ;MODD TEST WITH EXPONENT OF THE RESULT = 79
4970 026312 HHH5:
      026312 104413 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
4971 026314 004737 027416 JSR PC,MODDSUB
4972 026320 140240 000000 000000 1$: .WORD 140240,0,0,0 ;AC
4973 026330 063714 146314 2$: .WORD 63714,146314 ;FSRC
4974 026334 133572 167737 .WORD 133572,167737
4975 026340 000000 000000 000000 3$: .WORD 0,0,0,0 ;FRACTIONAL RES.
4976 026350 153777 177777 4$: .WORD 163777,-1 ;INTEGER RES.
4977 026354 162531 125726 .WORD 162531,125726
4978 026360 177777 177777 177777 5$: .WORD -1,-1,-1,-1 ;ERROR FRACTIONAL RES.
4979 026370 063777 177777 6$: .WORD 63777,-1 ;ERROR INTEGER RES.
4980 026374 162531 125726 .WORD 162531,125726
4981 026400 000210 7$: 210 ;FPS BEFORE EXECUTION.
4982 026402 000204 204 ;FPS AFTER EXECUTION.
4983 026404 104070 8$: ERROR +70
4984 026406 000401 BR 9$
4985 026410 104100 ERROR +100 ;ST 526 BAD SIGN
4986 026412 9$:
4987
4988 ;MODD TEST WITH EXPONENT OF THE RESULT = 57
4989 026412 HHH6:
      026412 104413 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
4990 026414 004737 027416 JSR PC,MODDSUB
4991 026420 056200 000000 000000 1$: .WORD 56200,0,0,1 ;AC
4992 026430 040340 000000 000000 2$: .WORD 40340,0,0,0 ;FSRC
4993 026440 000000 000000 000000 3$: .WORD 0,0,0,0 ;FRACTIONAL RES.
4994 026450 056340 000000 000000 4$: .WORD 56340,0,0,1 ;INTEGER RES.
4995 026460 040000 000000 000000 5$: .WORD 40000,0,0,0 ;ERROR FRACTIONAL RES.
4996 026470 056340 000000 000000 6$: .WORD 56340,0,0,1 ;ERROR INTEGER RES.

```



```

4997 026500 000213      7$:      213      ;FPS BEFORE EXECUTION.
4998 026502 000204      ;FPS AFTER EXECUTION.
4999 026504 104101      8$:      ERROR    +101  ;CONSTANT BAD (NOT 56)
5000                                     ;OR ST 525 TO 050 INTO 150
5001 026506 000401      BR        9$
5002 026510 104101      ERROR    +101
5003 026512
5004
5005
5006 026512      ;MODD TEST WITH EXPONENT OF THE RESULT = 56
      026512 104413      HHH7:
      004737 027416      LPERR
      056000 000000 1$:      JSR      PC,MODDSUB      ;SET UP THE LOOP ON ERROR ADDRESS.
      040340 000000 2$:      .WORD    56000,0,0,1      ;AC
      040100 000000 3$:      .WORD    40340,0,0,0      ;FSRC
      056140 000000 4$:      .WORD    40100,0,0,0      ;FRACTIONAL RES.
      000000 000000 5$:      .WORD    56140,0,0,1      ;INTEGER RES.
      000000 000000 6$:      .WORD    0,0,0,0      ;ERROR FRACTIONAL RES.
      000000 000000 7$:      .WORD    56140,0,0,1      ;ERROR INTEGER RES.
      213      ;FPS BEFORE EXECUTION.
      200      ;FPS AFTER EXECUTION.
5013 026570 056140 000000 8$:      ERROR    +102  ;BAD CONSTANT (NOT 56) OR
5014 026600 000213      BR        9$      ;ST 525 TO 150 INTO 050
5015 026602 000200      ERROR    +102
5016 026604 104102
5017
5018 026606 000401      BR        9$
5019 026610 104102      ERROR    +102
5020 026612
5021
5022
5023 026612      ;MODD TEST WITH EXPONENT OF THE RESULT = 36
      026612 104413      HHH8:
      004737 027416      LPERR
      051177 177777 177777 1$:      JSR      PC,MODDSUB      ;SET UP THE LOOP ON ERROR ADDRESS.
      040200 000000 000000 2$:      .WORD    51177,-1,-1,-1      ;AC
      040177 177760 000000 3$:      .WORD    40200,0,0,0      ;FSRC
      051177 177777 177760 4$:      .WORD    40177,-20,0,0      ;FRACTIONAL RES.
      177777 177777 177777 5$:      .WORD    51177,-1,-20,0      ;INTEGER RES.
      177777 177777 177777 6$:      .WORD    -1,-1,-1,-1      ;ERROR FRACTIONAL RES.
      177777 177777 177777 7$:      .WORD    -1,-1,-1,-1      ;ERROR INTEGER RES.
      217      ;FPS BEFORE EXECUTION.
      200      ;FPS AFTER EXECUTION.
5031 026700 000217
5032 026702 000200
5033 026704 104070      8$:      ERROR    +70
5034 026706 000401      BR        9$
5035 026710 104071      ERROR    +71
5036 026712
5037
5038
5039 026712      ;MODD TEST WITH EXPONENT OF THE RESULT = 30
      026712 104413      HHH9:
      004737 027416      LPERR
      040200 000000 000000 1$:      JSR      PC,MODDSUB      ;SET UP THE LOOP ON ERROR ADDRESS.
      047577 177777 2$:      .WORD    40200,0,0,0      ;AC
      176000 000001 3$:      .WORD    47577,-1      ;FSRC
      031600 000000 000000 4$:      .WORD    176000,1      ;FRACTIONAL RES.
      047577 177777 5$:      .WORD    31600,0,0,0      ;INTEGER RES.
      176000 000000 6$:      .WORD    47577,-1      ;ERROR FRACTIONAL RES.
      000000 000000 000000 7$:      .WORD    176000,0      ;ERROR INTEGER RES.
      000000 000000 000000 8$:      .WORD    0,0,0,0      ;FPS BEFORE EXECUTION.
      047577 177777 177777 9$:      .WORD    47577,-1,-1,-1      ;FPS AFTER EXECUTION.
      200
      200
5049 027000 000200
5050 027002 000200

```

```

5051 027004 104103      8$:      ERROR      +103      :(NORMALIZE) ST 532 TO 122
5052                                     :INTO NORM.
5053 027006 000401      BR          9$
5054 027010 104104      ERROR      +104      :AC V 1 <= X14
5055                                     :OR ST 733 TO 156 INTO 157.
5056 027012      9$:
5057
5058 ;MODD TEST WITH EXPONENT OF THE RESULT = 31
5059 HHH10:
      LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.
      JSR        PC,MODDSUB
      .WORD      47777,-1      ;AC
      .WORD      177000,0
      .WORD      40200,0,0,0   ;FSRC
      .WORD      0,0,0,0      ;FRACTIONAL RES.
      .WORD      47777,-1     ;INTEGER RES.
      .WORD      177000,0
      .WORD      0,0,177000,0  ;ERROR FRACTIONAL RES.
      .WORD      -1,-1,-1,-1  ;ERROR INTEGER RES.
      213         ;FPS BEFORE EXECUTION.
      204         ;FPS AFTER EXECUTION.
      8$:      ERROR      +105      ;(BUT FD) STORE X10
      BR          9$
      ERROR      +71
      9$:
5076 ;MODD TEST WITH EXPONENT OF THE RESULT = 0
5077 HHH11:
      LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.
      JSR        PC,MODDSUB
      .WORD      40200,0,0,0   ;AC
      .WORD      40177,72727  ;FSRC
      .WORD      127272,72727
      .WORD      40177,72727  ;FRACTIONAL RES.
      .WORD      127272,72727
      .WORD      0,0,0,0      ;INTEGER RES.
      .WORD      -1,-1,-1,-1  ;ERROR FRACTIONAL RES.
      .WORD      0,0,-1,-1    ;ERROR INTEGER RES.
      200         ;FPS BEFORE EXECUTION.
      200         ;FPS AFTER EXECUTION.
      8$:      ERROR      +70
      BR          9$
      ERROR      +106      ;ST 246 TO 126 INTO 127 (BUT FD)
      9$:
5094 ;MODD TEST WITH EXPONENT OF THE RESULT = -115
5095 HHH12:
      LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.
      JSR        PC,MODDSUB
      .WORD      3377,-1      ;AC
      .WORD      -1,52525
      .WORD      40200,0,0,0   ;FSRC
      .WORD      3377,-1      ;FRACTIONAL RES.
      .WORD      -1,52525
      .WORD      0,0,0,0      ;INTEGER RES.
      .WORD      -1,-1,-1,-1  ;ERROR FRACTIONAL RES.
      .WORD      0,0,-1,-1    ;ERROR INTEGER RES.

```


5105 027300 000200
 5106 027302 000200
 5107 027304 104070
 5108 027306 000401
 5109 027310 104107
 5110 027312
 5111
 5112
 5113 027312 104413
 5114 027314 004737 027416
 5115 027320 040300 000000 000000
 5116 027330 020200 000000 000000
 5117 027340 020300 000000 000000
 5118 027350 000000 000000 000000
 5119 027360 000000 000000 177777
 5120 027370 177777 177777 177777
 5121 027400 000200
 5122 027402 000200
 5123 027404 104110
 5124 027406 000401
 5125 027410 104071
 5126 027412 000137 030014
 5127
 5128
 5129
 5130
 5131
 5132
 5133
 5134
 5135
 5136
 5137
 5138
 5139
 5140
 5141
 5142
 5143
 5144
 5145
 5146
 5147
 5148
 5149
 5150
 5151
 5152
 5153
 5154
 5155
 5156
 5157
 5158
 5159
 5160

7\$: 200 :FPS BEFORE EXECUTION.
 200 :FPS AFTER EXECUTION.
 8\$: ERROR +70
 BR 9\$
 ERROR +107 ;ST 446 TO 126 INTO 127 (BUT FD)

;MODD TEST WITH EXPONENT OF THE RESULT = -63, IN ROUND MODE.

HHH13: ;SET UP THE LOOP ON ERROR ADDRESS.

LPERR
 JSR PC,MODDSUB
 1\$: .WORD 40300,0,0,0 :AC
 2\$: .WORD 20200,0,0,1 :FSRC
 3\$: .WORD 20300,0,0,2 :FRACTIONAL RES.
 4\$: .WORD 0,0,0,0 :INTEGER RES.
 5\$: .WORD 0,0,-1,-1 :ERROR FRACTIONAL RES.
 6\$: .WORD -1,-1,-1,-1 :ERROR INTEGER RES.
 7\$: 200 :FPS BEFORE EXECUTION.
 200 :FPS AFTER EXECUTION.
 8\$: ERROR +110 :ST 127 INTO RND/TR
 BR 9\$
 ERROR +71
 9\$: JMP HHHDONE ;GO TO THE NEXT TEST.

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

```

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

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

5161	027416	012601		MODDSUB:	MOV	(SP)+,R1	:GET A POINTER TO THE ARGUMENTS
5162	027420	012700	000200		MOV	#200,R0	:SET FD MODE.
5163	027424	170100			LDFPS	R0	
5164	027426	010100			MOV	R1,R0	:SET UP ACO
5165	027430	172410			LDD	(R0),ACO	
5166	027432	012700	025676		MOV	#MODP1,R0	:PUT A BACKGROUND PATTERN INTO AC1.
5167	027436	172510			LDD	(R0),AC1	
5168	027440	016100	000060		MOV	60(R1),R0	:SET UP THE FPS.
5169	027444	170100			LDFPS	R0	
5170	027446	012737	027462	001236	MOV	#1\$, \$TMP2	
5171	027454	010100			MOV	R1,R0	:COMPUTE THE ADDRESS OF THE FSRC.
5172	027456	062700	000010		ADD	#10,R0	
5173							
5174	027462	171410		1\$:	MODD	(R0),ACO	:EXECUTE THE TEST INSTRUCTION.
5175							
5176	027464	170204			STFPS	R4	:GET THE FPS.
5177	027466	012700	000200		MOV	#200,R0	:SET FD MODE.
5178	027472	170100			LDFPS	R0	
5179	027474	012700	027774		MOV	#MODDT0,R0	:GET THE FRACTIONAL RESULT.
5180	027500	174010			STD	ACO,(R0)	
5181	027502	012700	030004		MOV	#MODDT1,R0	:GET THE INTEGER RESULT.
5182	027506	174110			STD	AC1,(R0)	
5183							
5184	027510	010102			MOV	R1,R2	:SAVE THE DATA IN CASE OF ERROR.
5185	027512	010237	001240		MOV	R2,\$TMP3	
5186	027516	062702	000010		ADD	#10,R2	
5187	027522	010237	001242		MOV	R2,\$TMP4	
5188	027526	062702	000010		ADD	#10,R2	
5189	027532	010237	001244		MOV	R2,\$TMP5	
5190	027536	062702	000010		ADD	#10,R2	
5191	027542	010237	001246		MOV	R2,\$TMP6	
5192	027546	012737	027774	001250	MOV	#MODDT0,\$TMP7	
5193	027554	012737	030004	001252	MOV	#MODDT1,\$TMP10	
5194	027562	016137	000062	001256	MOV	62(R1),\$TMP12	
5195	027570	010437	001254		MOV	R4,\$TMP11	
5196							
5197	027574	012702	027774		MOV	#MODDT0,R2	:CHECK THE FRACTIONAL RESULT.
5198	027600	010103			MOV	R1,R3	
5199	027602	062703	000020		ADD	#20,R3	
5200	027606	012705	000004		MOV	#4,R5	
5201	027612	022223		2\$:	CMP	(R2)+,(R3)+	
5202	027614	001020			BNE	10\$:BRANCH IF INCORRECT.
5203	027616	077503			SOB	R5,2\$	
5204							
5205	027620	012702	030004		MOV	#MODDT1,R2	:CHECK THE INTEGER RESULT.
5206	027624	010103			MOV	R1,R3	
5207	027626	062703	000030		ADD	#30,R3	
5208	027632	012705	000004		MOV	#4,R5	
5209	027636	022223		3\$:	CMP	(R2)+,(R3)+	
5210	027640	001026			BNE	15\$:BRANCH IF INCORRECT.
5211	027642	077503			SOB	R5,3\$	
5212							
5213							
5214	027644	026104	000062		CMP	62(R1),R4	:CHECK THE FPS.
5215	027650	001042			BNE	20\$:BRANCH IF INCORRECT.
5216							
5217	027652	000161	000072	9\$:	JMP	72(R1)	:RETURN.


```

5218
5219
5220 027656 012702 027774
5221 027662 010103
5222 027664 062703 000040
5223 027670 012705 000004
5224 027674 022223
5225 027676 001005
5226 027700 077503
5227 027702 010102
5228 027704 062702 000064
5229
5230 027710 000112
5231
5232 027712
5233 027712 104070
5234 027714 000756
5235
5236
5237 027716 012702 030004
5238 027722 010103
5239 027724 062703 000050
5240 027730 012705 000004
5241 027734 022223
5242 027736 001005
5243 027740 077503
5244 027742 010102
5245 027744 062702 000070
5246
5247 027750 000112
5248
5249 027752
5250 027752 104071
5251 027754 000736
5252
5253
5254 027756 010437 001254
5255 027762 016137 000062 001256
5256 027770 104072
5257 027772 000727
5258
5259 027774 000000 000000 000000 MODDT0: .WORD 0,0,0,0
5260
5261 030004 000000 000000 000000 MODDT1: .WORD 0,0,0,0
5262
5263 030014
030014 104412 HHHDONE:
RSETUP

:FRACTIONAL ERROR.
10$: MOV #MODDT0,R2 ;WAS THE FRACTIONAL ERROR ANTICIPATED?
MOV R1,R3
ADD #40,R3
MOV #4,R5
50$: CMP (R2)+,(R3)+ ;BRANCH IF NOT ANTICIPATED.
BNE 11$
SOB R5,50$
MOV R1,R2 ;THE ERROR WAS ANTICIPATED SO
ADD #64,R2 ;RETURN TO THE ERROR REPORT AT THE
;CALLING ROUTINE.
JMP (R2)

11$:
12$: ERROR +70 ;THE ERROR WAS NOT ANTICIPATED SO
BR 9$ ;REPORT THE INCORRECT FRACTION HERE.

:INTEGER ERROR.
15$: MOV #MODDT1,R2 ;WAS THE INTEGER ERROR ANTICIPATED?
MOV R1,R3
ADD #50,R3
MOV #4,R5
60$: CMP (R2)+,(R3)+ ;BRANCH IF NOT ANTICIPATED.
BNE 17$
SOB R5,60$
MOV R1,R2 ;THE ERROR WAS ANTICIPATED SO RETURN
ADD #70,R2 ;TO THE ERROR REPORT IN THE CALLING
;ROUTINE.
JMP (R2)

16$:
17$: ERROR +71 ;THE ERROR WAS NOT ANTICIPATED SO REPORT
BR 9$ ;THE INTEGER FAILURE HERE.

:FPS INCORRECT.
20$: MOV R4,$TMP11 ;REPORT INCORRECT FPS.
MOV 62(R1),$TMP12
21$: ERROR +72
BR 9$

MODDT0: .WORD 0,0,0,0
MODDT1: .WORD 0,0,0,0

HHHDONE:
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?).

```

5264
5265
5266
5274
5275

;TEST TITLE:UNDER/OVERFLOW. USING MODF WITH TRAPS DISABLED

5276

```
.SBTTL TEST # 20 - SEE COMMENT ABOVE FOR TEST TITLE
:*****
:*TEST 20 SEE COMMENT ABOVE FOR TEST TITLE
:*
:*THIS IS A TEST OF THE MODF OVERFLOW AND UNDERFLOW CONDITIONS. IT MAKES
:*USE OF A SUBROUTINE TO SETUP THE OPERANDS, EXECUTE THE MODF INSTRUCTION
:*AND CHECK THE RESULTS. TRAPS ARE DISABLED DURING THIS TEST.
:*
:*****
```

5277 030016 000004

```
TST20: SCOPE
:UNDERFLOW TEST, WITH EXPONENT OF THE RESULT = -129, FIU = 1, FID = 1
MMM1:
```

5278 030020 104413
 5279 030022 004737 030346
 5280 030026 020123 045676
 5281 030032 020200 000000
 5282 030036 000123 045676
 5283 030042 000000 000000
 5284 030046 177777 177777
 5285 030052 177777 177777
 5286 030056 042000
 5287 030060 142004
 5288 030062 000012
 5289 030064 104170
 5290 030066 000401
 5291 030070 104171
 5292 030072

```
LPERR JSR PC,MODFOV ;SET UP THE LOOP ON ERROR ADDRESS.
1$: .WORD 20123,45676 ;AC
2$: .WORD 20200,0 ;FSRC
3$: .WORD 123,45676 ;FRACTIONAL RES.
4$: .WORD 0,0 ;INTEGER RES.
5$: .WORD -1,-1 ;ERROR FRACTIONAL RES.
6$: .WORD -1,-1 ;ERROR INTEGER RES.
7$: 42000 ;FPS BEFORE EXECUTION.
142004 ;FPS AFTER EXECUTION.
12 ;FEC
8$: ERROR +170 ;FEC INCORRECT, UNDERFLOW.
BR 9$
ERROR +171 ;AC V 1 (2,3) <= ZERO, ST 126.
9$:
```

5293 030072

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

5294
 5295
 5296 030072 104413
 5297 030074 004737 030346
 5298 030100 010200 000000
 5299 030104 010000 000000
 5300 030110 000000 000000
 5301 030114 000000 000000
 5302 030120 177777 177777
 5303 030124 177777 177777
 5304 030130 005013
 5305 030132 005004
 5306 030134 000012
 5307 030136 000240
 5308 030140 000401
 5309 030142 104171
 5310 030144

```
MMM2: LPERR JSR PC,MODFOV ;SET UP THE LOOP ON ERROR ADDRESS.
1$: .WORD 10200,0 ;AC
2$: .WORD 10000,0 ;FSRC
3$: .WORD 0,0 ;FRACTIONAL RES.
4$: .WORD 0,0 ;INTEGER RES.
5$: .WORD -1,-1 ;ERROR FRACTIONAL RES.
6$: .WORD -1,-1 ;ERROR INTEGER RES.
7$: 5013 ;FPS BEFORE EXECUTION.
5004 ;FPS AFTER EXECUTION.
12 ;FEC
8$: NOP
BR 9$
ERROR +171
9$:
```

5311

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

5312
 5313 030144 104413
 5314 030146 004737 030346
 5315 030152 060052 125252
 5316 030156 060200 000000
 5317 030162 000000 000000
 5318 030166 000052 125252
 5319 030172 000000 000000
 5320 030176 000000 000000

```
MMM3: LPERR JSR PC,MODFOV ;SET UP THE LOOP ON ERROR ADDRESS.
1$: .WORD 60052,125252 ;AC
2$: .WORD 60200,0 ;FSRC
3$: .WORD 0,0 ;FRACTIONAL RES.
4$: .WORD 52,125252 ;INTEGER RES.
5$: .WORD 0,0 ;ERROR FRACTIONAL RES.
6$: .WORD 0,0 ;ERROR INTEGER RES.
```



```

5321 030202 041000      7$:      41000      :FPS BEFORE EXECUTION.
5322 030204 141006      :      141006      :FPS AFTER EXECUTION.
5323 030206 000010      :      10          :FEC
5324 030210 104172      8$:      ERROR      +172      :BAD FEC ON OVERFLOW.
5325 030212 000401      :      BR          9$
5326 030214 104173      :      ERROR      +173      :ST 520 TO STORE ZERO TWICE
5327                                     :INTO 162
5328 030216      9$:
5329
5330      :OVERFLOW TEST WITH EXPONENT OF THE RESULT = 130, FIV = 0, FID = 1
5331 030216      MMM4:
5332 030216 104413      LPERR      :SET UP THE LOOP ON ERROR ADDRESS.
5333 030220 004737 030346      JSR      PC,MODFOV
5334 030224 060345 067654      1$:      .WORD      60345,67654      :AC
5335 030230 060200 000000      2$:      .WORD      60200,0          :FSRC
5336 030234 000000 000000      3$:      .WORD      0,0            :FRACTIONAL RES.
5337 030240 000000 000000      4$:      .WORD      0,0            :INTEGER RES.
5338 030244 000000 000000      5$:      .WORD      0,0            :ERROR FRACTIONAL RES.
5339 030250 000345 067654      6$:      .WORD      345,67654      :ERROR INTEGER RES.
5340 030254 006011      7$:      6011          :FPS BEFORE EXECUTION.
5341 030256 006006      :      6006          :FPS AFTER EXECUTION.
5342 030260 000010      :      10          :FEC
5343 030262 000240      8$:      NOP
5344 030264 000401      :      BR          9$
5345 030266 104174      :      ERROR      +174      :ST 520 TO 162 INTO STORE ZERO TWICE.
5346      9$:
5347      :OVERFLOW TEST WITH EXPONENT OF THE RESULT = 128, RESULT NEGATIVE
5348      :AND FIV = 1, FID = 1
5349 030270      MMM5:
5350 030270 104413      LPERR      :SET UP THE LOOP ON ERROR ADDRESS.
5351 030272 004737 030346      JSR      PC,MODFOV
5352 030276 160252 125252      1$:      .WORD      160252,125252      :AC
5353 030302 060000 000000      2$:      .WORD      60000,0          :FSRC
5354 030306 000000 000000      3$:      .WORD      0,0            :FRACTIONAL RES.
5355 030312 100052 125252      4$:      .WORD      100052,125252      :INTEGER RES.
5356 030316 000000 000000      5$:      .WORD      0,0            :ERROR FRACTIONAL RES.
5357 030322 000052 125252      6$:      .WORD      52,125252      :ERROR INTEGER RES.
5358 030326 041000      7$:      41000          :FPS BEFORE EXECUTION.
5359 030330 141006      :      141006          :FPS AFTER EXECUTION.
5360 030332 000010      :      10          :FEC
5361 030334 104172      8$:      ERROR      +172
5362 030336 000401      :      BR          9$
5363 030340 104175      :      ERROR      +175      :ST 517, BAD SIGN.
5364 030342 000137 030742      9$:      JMP      MMMDONE ;GO TO THE NEXT TEST.
5365      :THIS SUBROUTINE, MODFOV, IS CALLED TO SETUP THE
5366      :OPERANDS, EXECUTE THE MODF INSTRUCTION AND CHECK THE RESULTS.
5367      :IT IS CALLED THUS:
5368      :
5369      :ACARG: .WORD      X,X      :AC OPERAND
5370      :FSRCARG: .WORD      X,X      :FSRC OPERAND
5371      :FRES: .WORD      X,X      :FRACTIONAL RESULT
5372      :INTRES: .WORD      X,X      :INTEGER RESULT
5373      :ERFRES: .WORD      X,X      :ERROR FRACTIONAL RESULT
5374      :ERINTRES: .WORD      X,X      :ERROR INTEGER RESULT
5375      :FPSB: .WORD      X      :FPS BEFORE EXECUTION
    
```


5433	030514	010437	001254		MOV	R4,\$TMP11	
5434	030520	016137	000032	001256	MOV	32(R1),\$TMP12	
5435	030526	010537	001260		MOV	R5,\$TMP13	
5436	030532	016137	000034	001262	MOV	34(R1),\$TMP14	
5437							
5438	030540	012702	030722		MOV	#MODFDO,R2	;CHECK THE FRACTIONAL RESULT.
5439	030544	026112	000010		CMP	10(R1),(R2)	
5440	030550	001025			BNE	10\$;BRANCH IF INCORRECT.
5441	030552	026162	000012	000002	CMP	12(R1),2(R2)	
5442	030560	001021			BNE	10\$	
5443							
5444	030562	012702	030732		MOV	#MODFD1,R2	;CHECK THE INTEGER RESULT.
5445	030566	026112	000014		CMP	14(R1),(R2)	
5446	030572	001016			BNE	15\$;BRANCH IF INCORRECT.
5447	030574	026162	000016	000002	CMP	16(R1),2(R2)	
5448	030602	001012			BNE	15\$	
5449							
5450	030604	026104	000032		CMP	32(R1),R4	;CHECK THE FPS.
5451	030610	001024			BNE	20\$;BRANCH IF INCORRECT.
5452							
5453	030612	026105	000034		CMP	34(R1),R5	;CHECK THE FEC.
5454	030616	001030			BNE	25\$;BRANCH IF INCORRECT.
5455							
5456	030620	000161	000044		9\$: JMP	44(R1)	;RETURN.
5457							
5458							
5459	030624						
5460	030624	104165			10\$:		;THE ERROR WAS NOT ANTICIPATED SO
5461	030626	000774			12\$:	ERROR +165	;REPORT THE INCORRECT FRACTION HERE.
5462						BR 9\$	
5463							
5464	030630	026112	000024				
5465	030634	001010					
5466	030636	026162	000026	000002			
5467	030644	001004					
5468	030646	010102					
5469	030650	062702	000042				
5470							
5471	030654	000112					
5472							
5473	030656				16\$:		;THE ERROR WAS NOT ANTICIPATED SO REPORT
5474	030656	104166			17\$:	ERROR +166	;THE INTEGER FAILURE HERE.
5475	030660	000757				BR 9\$	
5476							
5477							
5478	030662	010437	001254				
5479	030666	016137	000032	001256			
5480	030674	104167					
5481	030676	000750					
5482							
5483							
5484	030700	010537	001260				
5485	030704	016137	000034	001262			
5486	030712	010102					
5487	030714	062702	000036				
5488	030720	000112					
5489							

5490 030722 000000 000000 000000 MODFD0: .WORD 0,0,0,0
5491
5492 030732 000000 000000 000000 MODFD1: .WORD 0,0,0,0
5493
5494 030742 MMDONE:
030742 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?).

5495
5496
5497
5505
5506

;TEST TITLE:UNDER/OVERFLOW, USING MODD WITH TRAPS DISABLED

5507

```
.SBTTL TEST # 21 - SEE COMMENT ABOVE FOR TEST TITLE  
:*****  
:*TEST 21 SEE COMMENT ABOVE FOR TEST TITLE  
:*  
:*THIS IS A TEST OF THE MODD INSTRUCTION'S OVER FLOW AND UNDER FLOW  
:*CONDITIONS. A SUBROUTINE IS USED TO SET UP THE OPERANDS, EXECUTE THE  
:*MODD INSTRUCTION AND CHECK THE RESULTS.  
:*  
:*****
```

```
030744 000004  
5508  
5509  
5510 030746  
030746 104413  
5511 030750 004737 031362  
5512 030754 020252 125252  
5513 030760 125252 125252  
5514 030764 020100 000000 000000  
5515 030774 000177 177777 177777  
5516 C31004 000000 000000 000000  
5517 031014 020252 125252  
5518 031020 125252 125252  
5519 031024 000000 000000 177777  
5520 031034 042200  
5521 031036 142204  
5522 031040 000012  
5523 031042 104201  
5524 031044 000401  
5525 031046 104202  
5526 031050  
5527
```

```
TST21: SCOPE  
;UNDERFLOW TEST WITH EXPONENT OF THE RESULT = -129, FIU = 1, FID = 1  
NNN1:  
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.  
JSR PC,MODDOV ;AC  
1$: .WORD 20252,125252 ;AC  
.WORD 125252,125252  
2$: .WORD 20100,0,0,0 ;FSRC  
3$: .WORD 177,-1,-1,-1 ;FRACTIONAL RES.  
4$: .WORD 0,0,0,0 ;INTEGER RES.  
5$: .WORD 20252,125252 ;ERROR FRACTIONAL RES.  
.WORD 125252,125252  
6$: .WORD 0,0,-1,-1 ;ERROR INTEGER RES.  
7$: 42200 ;FPS BEFORE EXECUTION.  
142204 ;FPS AFTER EXECUTION.  
12 ;FEC  
8$: ERROR +201 ;FEC INCORRECT ON UNDERFLOW.  
BR 9$  
ERROR +202 ;ST 155 (BUT FD)  
9$:
```

```
5528  
5529 031050  
031050 104413  
5530 031052 004737 031362  
5531 031056 010000 000000  
5532 031062 123456 000000  
5533 031066 010200 000000 000000  
5534 031076 000000 000000 000000  
5535 031106 000000 000000 000000  
5536 031116 000000 000000 000000  
5537 031126 000000 000000 000000  
5538 031132 123456 000000  
5539 031136 005213  
5540 031140 005204  
5541 031142 000012  
5542 031144 000240  
5543 031146 000401  
5544 031150 104203  
5545 031152  
5546  
5547
```

```
;UNDERFLOW TEST WITH EXPONENT OF THE RESULT = -193, FIU = 0, FID = 1  
NNN2:  
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.  
JSR PC,MODDOV ;AC  
1$: .WORD 10000,0 ;AC  
.WORD 123456,0  
2$: .WORD 10200,0,0,0 ;FSRC  
3$: .WORD 0,0,0,0 ;FRACTIONAL RES.  
4$: .WORD 0,0,0,0 ;INTEGER RES.  
5$: .WORD 0,0,0,0 ;ERROR FRACTIONAL RES.  
6$: .WORD 0,0 ;ERROR INTEGER RES.  
.WORD 123456,0  
7$: 5213 ;FPS BEFORE EXECUTION.  
5204 ;FPS AFTER EXECUTION.  
12  
8$: NOP  
BR 9$  
ERROR +203 ;ST 047 (BUT FD).  
9$:
```

```
5548 031152  
031152 104413  
5549 031154 004737 031362  
5550 031160 060252 125252  
5551 031164 125252 125252
```

```
;OVERFLOW TEST WITH EXPONENT OF THE RESULT = 128, FIV = 1, FID = 1  
NNN3:  
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.  
JSR PC,MODDOV ;AC  
1$: .WORD 60252,125252 ;AC  
.WORD 125252,125252
```

```

5552 031170 060100 000000 000000 2$: .WORD 60100,0,0,0 ;FSRC
5553 031200 000000 000000 000000 3$: .WORD 0,0,0,0 ;FRACTIONAL RES.
5554 031210 000177 177777 177777 4$: .WORD 177,-1,-1,-1 ;INTEGER RES.
5555 031220 000000 000000 000000 5$: .WORD 0,0,0,0 ;ERROR FRACTIONAL RES.
5556 031230 000177 177777 000000 6$: .WORD 177,-1 ;ERROR INTEGER RES.
5557 031234 125252 125252 .WORD 125252,125252
5558 031240 041200 7$: 41200 ;FPS BEFORE EXECUTION.
5559 031242 141206 141206 ;FPS AFTER EXECUTION.
5560 031244 000010 10 ;FEC
5561 031246 104204 8$: ERROR +204 ;FEC BAD ON OVERFLOW.
5562 031250 000401 BR 9$
5563 031252 104205 ERROR +205 ;ST 520 TO 162 INTO 163 (BUT FD).
5564 031254 9$:
5565
5566 ;OVERFLOW TEST WITH EXPONENT OF THE RESULT = 130, FIV = 0, FID = 1
5567 031254 NNN4:
031254 104413 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
5568 031256 004737 031362 JSR PC,MODDOV
5569 031262 060200 000000 1$: .WORD 60200,0 ;AC
5570 031266 125252 000000 .WORD 125252,0
5571 031272 060200 000000 000000 2$: .WORD 60200,0,0,0 ;FSRC
5572 031302 000000 000000 000000 3$: .WORD 0,0,0,0 ;FRACTIONAL RES.
5573 031312 000000 000000 000000 4$: .WORD 0,0,0,0 ;INTEGER RES.
5574 031322 000000 000000 000000 5$: .WORD 0,0,0,0 ;ERROR FRACTIONAL RES.
5575 031332 000400 000000 000000 6$: .WORD 400,0 ;ERROR INTEGER RES.
5576 031336 125252 000000 .WORD 125252,0
5577 031342 006211 7$: 6211 ;FPS BEFORE EXECUTION.
5578 031344 006206 6206 ;FPS AFTER EXECUTION.
5579 031346 000010 10 ;FEC
5580 031350 000240 8$: NOP
5581 031352 000401 BR 9$
5582 031354 104206 ERROR +206 ;ST 520 TO 162 INTO STORE ZERO TWICE.
5583 031356 000137 031770 9$: JMP NNNDONE ;GO TO NEXT TEST.

```

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

```

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

```

; THE OPERANDS ARE SET UP (USING ACO FOR THE AC ARGUMENT). THE MODD
 ; INSTRUCTION IS EXECUTED. THEN THE RESULTS ARE RETRIEVED.
 ; THE FRACTION PART OF THE RESULT IS COMPARED WITH FRES. IF THIS IS CORRECT
 ; THEN THE INTEGER PART IS COMPARED WITH INTRES. IF BOTH OF THESE ARE CORRECT
 ; THEN THE FPS IS COMPARED WITH FPSA. AFTER EXECUTION IF NO ERRORS OCCURRED
 ; THEN MODDOV WILL RETURN TO CONT. IF THE FPS WAS INCORRECT

5584
5585
5586
5587
5588
5589
5590
5591
5592
5593
5594
5595
5596
5597
5598
5599
5600
5601
5602
5603
5604
5605
5606
5607


```

5608                                     :IT IS REPORTED HERE. IF THE FRACTION IS INCORRECT IT IS COMPARED WITH
5609                                     :THE ANTICIPATED BAD FRACTION, ERFRES. IF THIS DOESN'T MATCH
5610                                     :THE TRUE RESULT THEN THE ERROR IS REPORTED HERE. IF THE ANTICIPATED
5611                                     :FAILURE MATCHES THE TRUE RESULT THEN MODDOV PASSES CONTROL TO THE
5612                                     :ERROR CALL AT ERR1. LIKewise IF THE INTEGER PART OF THE RESULT IS
5613                                     :NOT CORRECT THEN IT IS COMPARED WITH THE ANTICIPATED INTEGER
5614                                     :FAILURE. IF THIS DOESN'T MATCH THEN THE ERROR IS REPORTED HERE.
5615                                     :IF A MATCH IS MADE HOWEVER, MODDOV WILL RETURN CONTROL TO THE ERROR
5616                                     :CALL AT ERR2.
5617
5618 031362 012601 MODDOV: MOV (SP)+,R1 ;GET A POINTER TO THE ARGUMENTS
5619 031364 012700 000200 MOV #200,R0 ;SET FD MODE.
5620 031370 170100 LDFPS R0
5621 031372 010100 MOV R1,R0 ;SET UP ACO
5622 031374 172410 LDD (R0),ACO
5623 031376 012700 025676 MOV #MODP1,R0 ;PUT A BACKGROUND PATTERN INTO AC1.
5624 031402 172510 LDD (R0),AC1
5625 031404 016100 000060 MOV 60(R1),R0 ;SET UP THE FPS.
5626 031410 170100 LDFPS R0
5627 031412 012737 031426 001236 MOV #1$, $TMP2
5628 031420 010100 MOV R1,R0 ;COMPUTE THE ADDRESS OF THE FSRC.
5629 031422 062700 000010 ADD #10,R0
5630
5631 031426 171410 1$: MODD (R0),ACO ;EXECUTE THE TEST INSTRUCTION.
5632
5633 031430 170305 STST R5 ;GET THE FPS.
5634 031432 170204 STFPS R4 ;GET THE FPS.
5635 031434 012700 000200 MOV #200,R0 ;SET FD MODE.
5636 031440 170100 LDFPS R0
5637 031442 012700 031750 MOV #MODDD0,R0 ;GET THE FRACTIONAL RESULT.
5638 031446 174010 STD ACO,(R0)
5639 031450 012700 031760 MOV #MODDD1,R0 ;GET THE INTEGER RESULT.
5640 031454 174110 STD AC1,(R0)
5641
5642 031456 010102 MOV R1,R2 ;SAVE THE DATA IN CASE OF ERROR.
5643 031460 010237 001240 MOV R2,$TMP3
5644 031464 062702 000010 ADD #10,R2
5645 031470 010237 001242 MOV R2,$TMP4
5646 031474 062702 000010 ADD #10,R2
5647 031500 010237 001244 MOV R2,$TMP5
5648 031504 062702 000010 ADD #10,R2
5649 031510 010237 001246 MOV R2,$TMP6
5650 031514 012737 031750 001250 MOV #MODDD0,$TMP7
5651 031522 012737 031760 001252 MOV #MODDD1,$TMP10
5652 031530 010437 001254 MOV R4,$TMP11
5653 031534 016137 000062 001256 MOV 62(R1),$TMP12
5654 031542 010537 001260 MOV R5,$TMP13
5655 031546 016137 000064 001262 MOV 64(R1),$TMP14
5656
5657 031554 012702 031750 MOV #MODDD0,R2 ;CHECK THE FRACTIONAL RESULT.
5658 031560 010103 MOV R1,R3
5659 031562 062703 000020 ADD #20,R3
5660 031566 012700 000004 MOV #4,R0
5661 031572 022223 2$: CMP (R2)+,(R3)+
5662 031574 001023 BNE 10$ ;BRANCH IF INCORRECT.
5663 031576 077003 SOB R0,2$
5664
    
```

5665	031600	012702	031760	MOV	#MODDD1,R2	:CHECK THE INTEGER RESULT.
5666	031604	010103		MOV	R1,R3	
5667	031606	062703	000030	ADD	#30,R3	
5668	031612	012700	000004	MOV	#4,R0	
5669	031616	022223		3\$: CMP	(R2)+,(R3)+	:BRANCH IF INCORRECT.
5670	031620	001013		BNE	15\$	
5671	031622	077003		SOB	R0,3\$	

5766

```

.SBTTL TEST # 22 - INTERRUPT CORRECT FLOWS TEST
:*****
:*TEST 22          INTERRUPT CORRECT FLOWS TEST
:*
:*THIS IS A TEST OF THE 'CORRECT' FLOWS. THIS PART OF THE MICRO CODE
:*HAS AS ITS PURPOSE INSURING THAT INTERRUPT REQUESTS MADE DURING
:*CERTAIN LENGTHY FPP INSTRUCTIONS GET HONORED. THIS IS DONE
:*IN A WAY SUCH THAT IF AN INTERRUPT REQUEST OCCURS DURING ONE
:*OF THESE INSTRUCTIONS THE STATE OF THAT INSTRUCTION'S
:*EXECUTION WILL BE THE SAME AS IF THAT INSTRUCTION HAD NEVER
:*BEEN FETCHED AND ITS EXECUTION NEVER STARTED. THUS THE MICRO CODE
:*WILL RESTORE ALL REGISTERS, BACK UP THE PC AND LEAVE THE
:*FPS AND ACO THROUGH ACS UNMODIFIED.
:*THE INSTRUCTIONS FOR WHICH THIS IS NECESSARY ARE:
:*
:*   ADD (OR SUB)
:*   DIV
:*   MUL
:*   MOD
:*(BOTH DOUBLE AND FLOATING)
:*ALL ADDRESSING MODES WILL BE TRIED WITH THE ADD INSTRUCTION. THEN
:*EACH OF THE OTHER INSTRUCTIONS WILL BE TRIED USING MODE 1.
:*NOTE THAT THIS TEST NEEDS A SPECIAL INTERRUPT MODULE,
:*WHICH WILL PROBABLY ONLY BE PRESENT IN DEC'S MANUFACTURING ENVIRONMENT,
:*TO RUN. THIS SPECIAL EQUIPMENT IS DESIGNED TO RAISE AN
:*INTERRUPT REQUEST IN THE PROCESSOR IF A BIT IS SET IN ITS STATUS
:*REGISTER AND ONLY WHEN AN FPP INSTRUCTION IS ENCOUNTERED.
:*THEREFORE THIS TEST WILL BE RUN CONDITIONALLY (DEPENDENT UPON WHETHER
:*OR NOT THE STATUS REGISTER OF THE TEST EQUIPMENT TIMES OUT WHEN REFERENCED).
:*THIS TEST CAN ALSO BE DESELECTED BY TURNING SWITCH 7 OF THE
:*SWITCH REGISTER (PHYSICAL OR VIRTUAL) ON.
:*THE TEST ASSUMES THAT THE TEST EQUIPMENT'S STATUS REGISTER IS AT
:*LOCATION 777774 (NOTE THAT ALL REFERENCES TO THIS LOCATION ARE
:*MADE INDIRECT THROUGH THIS PROGRAMS LOCATION CORINT, SO THAT
:*IF THE USER HAS MODIFIED THE TEST EQUIPMENT'S STATUS REGISTER TO
:*RESPOND TO A DIFFERENT ADDRESS LOCATION CORINT MUST BE
:*MADE TO CONTAIN THAT STATUS REGISTER'S NEW ADDRESS).
:*THIS PROGRAM ASSUMES THAT THE TRAP VECTOR FOR THE TEST EQUIPMENT IS 110.
:*AGAIN NOTE THAT ALL REFERENCES TO THIS TRAP VECTOR ARE INDIRECT, THROUGH
:*THIS PROGRAM'S LOCATION CORTRP (IF THE TEST EQUIPMENT IS
:*MADE TO TRAP TO A DIFFERENT VECTOR LOCATION CORTRP MUST CONTAIN THE
:*ADDRESS OF THIS VECTOR).
    
```

```

:*****
:TST22:  SCOPE
5767 031772 000004
5768 031774 132737 000200 001337      BITB   #200,$ENVM      ;SEE IF APT IS SELECTING TEST.
5769 032002 001406 000200 147126      BEQ    COR1          ;BRANCH TO AUTOSIZE IF NOT.
5770 032004 032777 000200 147126      BIT    #200,@SWR    ;SEE IF THE USER HAS SELECTED THIS
5771                                     ;TEST USING THE SWITCH REGISTER.
5772 032012 001022 033152              BNE    COR3          ;IF SO, PERFORM TEST.
5773 032014 000137 033152              JMP    CORDONE ;ELSE DO NOT RUN TEST.
5774
5775 032020 012737 032044 000004  COR1:  MOV    #COR2,ERRVECT ;SEE IF THE TEST EQUIPMENT'S STATUS
5776 032026 012777 000000 001112      MOV    #C,@CORINT  ;REGISTER TIMES OUT.
5777 032034 012737 037032 000004      MOV    #CPSPUR,ERRVECT
5778 032042 000406                          BR     COR3          ;DIDN'T TIME OUT SO START TEST.
5779
    
```



```

5780 032044 022626 COR2: CMP (SP)+,(SP)+ ;IF THE REFERENCE TIMES OUT DO
5781 032046 012737 037032 000004 MOV #CPSPUR,ERRVCT
5782 032054 000137 033152 JMP CORDONE ;NOT RUN TEST.
5783
5784 ;TEST ADDD MODE 0
5785 032060 COR3:
5786 032060 005227 177777 INC #-1
5787 032064 001002 BNE COR33
5788 032066 104401 TYPE
5789 032070 040005 .WORD CORMES
5790 032072 COR33:
5791 032072 104413 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
5792 032074 004737 032652 JSR PC,CORSUB
5793 032100 040200 000100 000200 1$: .WORD 40200,100,200,300 ;ACO
5794 032110 123456 2$: .WORD 123456 ;RO
5795 032112 000200 3$: 200 ;FPS
5796 032114 172000 4$: ADDD ACO,ACO ;TEST INSTRUCTION.
5797 032116 000240 NOP
5798 032120 005037 033134 CLR CORFLG ;RESET INTERRUPT FLAG
5799 032124 104252 ERROR +252 ;NO INTERRUPT! TEST EQUIPMENT FAILED.
5800 032126 000401 BR 11$
5801 032130 104253 5$: ERROR +253 ;INCORRECT STATE AT INTERRUPT.
5802 032132 11$:
5803 ;TEST ADDD MODE 1
5804 032132 COR4:
5805 032132 104413 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
5806 032134 004737 032652 JSR PC,CORSUB
5807 032140 040201 000555 077007 1$: .WORD 40201,555,77007,111111 ;ACO
5808 032150 032140 2$: .WORD 1$ ;RO
5809 032152 000217 3$: 217 ;FPS
5810 032154 172010 4$: ADDD (RO),ACO ;TEST INSTRUCTION
5811 032156 000240 NOP
5812 032160 005037 033134 CLR CORFLG ;RESET INTERRUPT FLAG
5813 032164 104252 ERROR +252 ;REPORT FAILURE. NO INTERRUPT.
5814 032166 000401 BR 11$
5815 032170 104254 ERROR +254
5816 11$:
5817 ;TEST ADDD MODE 2
5818 032172 COR5:
5819 032172 104413 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
5820 032174 004737 032652 JSR PC,CORSUB
5821 032200 040202 111333 052525 1$: .WORD 40202,111333,52525,70707 ;ACO
5822 032210 032200 2$: .WORD 1$ ;RO
5823 032212 000205 3$: 205 ;FPS
5824 032214 172020 4$: ADDD (RO)+,ACO ;TEST INSTRUCTION
5825 032216 000240 NOP
5826 032220 005037 033134 CLR CORFLG ;RESET THE INTERRUPT FLAG
5827 032224 104252 ERROR +252 ;REPORT FAILURE. NO INTERRUPT.
5828 032226 000401 BR 11$
5829 032230 104255 ERROR +255 ;CORRECT FLOWS FAILED.
5830 11$:
5831 ;TEST ADDD MODE 3
5832 032232 COR6:
5833 032232 104413 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
5833 032234 004737 032652 JSR PC,CORSUB
    
```

```

5834 032240 040203 071735 072746 1$: .WORD 40203,71735,72746,1 ;ACO
5835 032250 032274 2$: .WORD 10$ ;RO
5836 032252 000206 3$: 206 ;FPS
5837 032254 172030 4$: ADDD @ (RO)+,ACO ;TEST INSTRUCTION
5838 032256 000240 NOP
5839 032260 005037 033134 CLR CORFLG ;RESET THE INTERRUPT FLAG
5840 032264 104252 ERROR +252 ;REPORT FAILURE, NO INTERRUPT.
5841 032266 000403 BR 11$
5842 032270 104256 5$: ERROR +256 ;CORRECT FLOWS FAILED.
5843 032272 000401 BR 11$
5844 032274 032240 10$: .WORD 1$ ;USED FOR THE ADDRESSING OF THE OPERAND
5845 ;IN THIS MODE.
5846 032276 11$:
5847 ;TEST ADDD MODE 4
5848 032276 COR7: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
032276 104413 JSR PC,CORSUB
5849 032300 004737 032652 070123 1$: .WORD 40204,123456,70123,45671 ;ACO
5850 032304 040204 123456 2$: .WORD 1$+10 ;RO
5851 032314 032314 3$: 212 ;FPS
5852 032316 000212 4$: ADDD -(RO),ACO ;TEST INSTRUCTION
5853 032320 172040 NOP
5854 032322 000240 CLR CORFLG ;RESET THE INTERRUPT FLAG
5855 032324 005037 033134 ERROR +252 ;REPORT FAILURE. NO INTERRUPT.
5856 032330 104252 BR 11$
5857 032332 000401 ERROR +257 ;CORRECT FLOWS FAILED
5858 032334 104257 11$:
5859 032336 ;TEST ADDD MODE 5
5860 COR8: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
5861 032336 104413 JSR PC,CORSUB
5862 032336 004737 032652 021076 1$: .WORD 40205,76543,21076,54321 ;ACO
5863 032340 040205 076543 2$: .WORD 10$+2 ;RO
5864 032344 032402 3$: 213 ;FPS
5865 032354 032402 4$: ADDD @-(RO),ACO ;TEST INSTRUCTION
5866 032356 000213 NOP
5867 032360 172050 CLR CORFLG ;RESET THE INTERRUPT FLAG
5868 032362 000240 ERROR +252 ;REPORT ERROR. NO INTERRUPT.
5869 032364 005037 033134 BR 11$
5870 032370 104252 5$: ERROR +260 ;CORRECT FLOWS FAILED.
5871 032372 000403 BR 11$
5872 032374 104260 10$: .WORD 1$
5873 032376 000401 11$:
5874 032400 032344 ;TEST ADDD MODE 6
5875 032402 COR9: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
5876 032402 104413 JSR PC,CORSUB
5877 032402 004737 032652 063730 1$: .WORD 40206,34353,63730,31323 ;ACO
5878 032402 040206 034353 2$: .WORD 1$-1 ;RO
5879 032410 032407 3$: 214 ;FPS
5880 032420 000214 4$: ADDD 1 (RO),ACO ;TEST INSTRUCTION
5881 032422 000214 CLR CORFLG
5882 032424 172060 000001 033134 ERROR +252 ;REPORT FAILURE NO TRAP.
5883 032424 005037 BR 11$
5884 032430 005037 033134 ERROR +261
5885 032434 104252
5886 032436 000401
5887 032440 104261
    
```



```

5888 032442      11$:
5889
5890
5891 032442      ;TEST ADDD MODE 7
      032442 104413      COR10:
      004737 032652      LPERR
      040210 070107 062426 1$:      JSR      PC,CORSUB      ;SET UP THE LOOP ON ERROR ADDRESS.
      032460 032503      .WORD    40210,70107,62426,55555 ;ACO
      032462 000204      .WORD    10$-1      ;RO
      032464 172070 000001 3$:      204      ;FPS
      032470 005037 033134 4$:      ADDD    @1,(RO),ACO ;TEST INSTRUCTION
      032474 104252      CLR      CORFLG
      032476 000403      ERROR   +252      ;REPORT FAILURE NO TRAP
      032500 104262      BR      11$
      032502 000401      5$:      ERROR   +262      ;CORRECT FLOWS FAILED.
      032504 032450      BR      11$
      032506      10$:      .WORD    1$
      032506      11$:
5904
5905
5906 032506      ;TEST DIVD MODE 1
      032506 104413      COR11:
      004737 032652      LPERR
      040211 033445 056677 1$:      JSR      PC,CORSUB      ;SET UP THE LOOP ON ERROR ADDRESS.
      032524 032514      .WORD    40211,33445,56677,001122 ;ACO
      032526 000205      .WORD    1$      ;RO
      032530 174410      3$:      205      ;FPS
      032532 000240      4$:      DIVD    (RO),ACO ;TEST INSTRUCTION
      032534 005037 033134      NOP
      032540 104252      CLR      CORFLG
      032542 000401      ERROR   +252      ;REPORT FAILURE, NO TRAP.
      032544 104263      BR      11$
      032546      5$:      ERROR   +263      ;CORRECT FLOWS FAILED.
      032546      11$:
5918
5919
5920 032546      ;TEST MULD MODE 1
      032546 104413      COR12:
      004737 032652      LPERR
      040212 165411 046252 1$:      JSR      PC,CORSUB      ;SET UP THE LOOP ON ERROR ADDRESS.
      032564 032554      .WORD    40212,165411,46252,63650 ;ACO
      032570 171010      2$:      .WORD    1$      ;RO
      032572 000240      3$:      .WORD    210      ;FPS
      032574 005037 033134 4$:      MULD    (RO),ACO ;TEST INSTRUCTION
      032600 104252      NOP
      032602 000401      CLR      CORFLG
      032604 104264      ERROR   +252      ;REPORT FAILURE, NO TRAP.
      032606      BR      11$
      032606      5$:      ERROR   +264      ;CORRECT FLOWS FAILED.
      032606      11$:
5932
5933
5934 032606      ;TEST MODD MODE 1
      032606 104413      COR13:
      004737 032652      LPERR
      040213 045654 054542 1$:      JSR      PC,CORSUB      ;SET UP THE LOOP ON ERROR ADDRESS.
      032624 032614      .WORD    40213,45654,54542,171623 ;ACO
      032626 000412      2$:      .WORD    1$      ;RO
      032630 171410      3$:      .WORD    412      ;FPS
      032632 000240      4$:      MODD    (RO),ACO ;TEST INSTRUCTION.
      032632      NOP
  
```



```

5998                                     ;TRAP TO HERE WHEN THE INTERRUPT OCCURS.
5999 032750 005137 033134                CORTV: COM   CORFLG ;FIRST SEE IF AN INTERRUPT WAS PENDING.
6000 032754 001060                       BNE   CORTV1 ;IF NOT GO REPORT AN ERROR.
6001 032756 012777 000000 000162        MOV   #0,@CORINT ;MAKE SURE THE TEST EQUIPMENT
6002                                     ;IS NOT INTERRUPT ENABLED.
6003
6004 032764 170204                       STFPS  R4      ;GET THE FPS.
6005 032766 012702 000200                MOV   #200,R2 ;GET ACO
6006 032772 170102                       LDFPS  R2
6007 032774 012702 033136                MOV   #CORTMP,R2
6008 033000 174012                       STD   ACO,(R2)
6009 033002 012737 033136 001240        MOV   #CORTMP,$TMP3
6010 033010 010037 001244                MOV   R0,$TMP5
6011 033014 010437 001250                MOV   R4,$TMP7
6012 033020 011637 001254                MOV   (SP),$TMP11
6013 033024 010102                       MOV   R1,R2
6014 033026 010237 001242                MOV   R2,$TMP4
6015 033032 062702 000010                ADD   #10,R2
6016 033036 012237 001246                MOV   (R2)+,$TMP6
6017 033042 012237 001252                MOV   (R2)+,$TMP10
6018 033046 010237 001256                MOV   R2,$TMP12
6019 033052 021602                       CMP   (SP),R2 ;SEE IF THE TRAP OCCURRED
6020                                     ;AT THE CORRECT ADDRESS.
6021 033054 001016                       BNE   CORTV0
6022 033056 022626                       CMP   (SP)+,(SP)+ ;RESET THE STACK.
6023 033060 020061 000010                CMP   R0,10(R1) ;SEE IF R0 IS CORRECT.
6024 033064 001012                       BNE   CORTV0 ;BR IF NOT CORRECT.
6025 033066 010102                       MOV   R1,R2 ;SEE IF ACO WAS CORRECT
6026 033070 012703 033136                MOV   #CORTMP,R3
6027 033074 012705 000004                MCV   #4,R5
6028 033100 022223 1$: CMP   (R2)+,(R3)+ ;BRANCH IF INCORRECT.
6029 033102 001003                       BNE   CORTV0
6030 033104 077503                       SOB   R5,1$
6031 033106 000161 000032                JMP   32(R1) ;IF EVERYTHING IS CORRECT THEN RETURN.
6032
6033 033112 000161 000030                CORTV0: JMP   30(R1) ;CORRECT FLOWS FAILED SO GO REPORT ERROR.
6034
6035 033116 011637 001236                CORTV1: MOV   (SP),$TMP2 ;AN INTERRUPT OCCURRED WHEN THE FLAG
6036                                     ;CORFLG, DID NOT INDICATE THAT ONE WAS
6037                                     ;PENDING SO REPORT SPURIOUS TRAP.
6038 033122 005037 033134                CLR   CORFLG
6039 033126 022626                       CMP   (SP)+,(SP)+
6040 033130 104266                       ERROR +266
6041 033132 000407                       BR    CORDONE
6042
6043 033134 000000 000000 000000        CORFLG: .WORD 0
6044 033136 000000 000000 000000        CORTMP: .WORD 0,0,0,0
6045
6046 033146 177774                       CORINT: .WORD 177774 ;THIS IS THE ADDRESS, 177774, OF THE
6047                                     ;TEST EQUIPMENT'S STATUS REGISTER.
6048                                     ;THE CONTENTS OF CORINT CAN BE MODIFIED
6049                                     ;IF THIS STATUS REGISTER'S ADDRESS IS
6050                                     ;CHANGED.
6051 033150 000110                       CORTRP: .WORD 110 ;THIS IS THE ADDRESS OF THE TEST EQUIPMENTS
6052                                     ;TRAP VECTOR. LIKE THE STATUS REGISTER'S ADDRESS
6053                                     ;DESCRIBED IMMEDIATELY ABOVE
6054                                     ;THIS VECTOR CAN BE CHANGED, BUT THE
  
```

6055
 6056
 6057
 6058 033152
 033152 104412

CORDONE:
 RSETUP

:CONTENTS OF CORTRP MUST INDICATE THE
 :CHANGE.

: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?).

6059
 6060
 6061
 6062 033154
 6063
 6064
 6065
 6066

TST23:

.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:

033154
 033154 000004
 033156 005037 001102
 033162 005037 001302
 033166 005237 001324
 033172 042737 100000 001324
 033200 005327
 033202 000001
 033204 003074
 033206 012737
 033210 000001
 033212 033202
 033214 104401 033222
 033220 000407

 033240
 033240 013746 001324

 033244 104403
 033246 006
 033247 000
 033250 104401 033256
 033254 000421

 033320
 033320 013746 001112

 033324 104403
 033326 006
 033327 000
 033330 104401 001313
 033334 005037 001112
 033340 013700 000042
 033344 001414

SCOPE
 CLR \$STNM ::ZERO THE TEST NUMBER
 CLR \$TIMES ::ZERO THE NUMBER OF ITERATIONS
 INC \$PASS ::INCREMENT THE PASS NUMBER
 BIC #100000,\$PASS ::DON'T ALLOW A NEG. NUMBER
 DEC (PC)+ ::LOOP?
 \$EOPCT: .WORD 1
 BGT \$DOAGN ::YES
 MOV (PC)+,@(PC)+ ::RESTORE COUNTER
 \$ENDCT: .WORD 1
 \$EOPCT
 TYPE ,65\$::TYPE ASCIZ STRING
 BR 64\$::GET OVER THE ASCIZ
 ::65\$: .ASCIZ <12><15>/END PASS #/
 64\$:
 MOV \$PASS,-(SP) ::SAVE \$PASS FOR TYPEOUT
 ::TYPE PASS NUMBER IN OCTAL
 ::GO TYPE--OCTAL ASCII
 TYPOS
 .BYTE 6 ::TYPE 6 DIGITS
 .BYTE 0 ::SUPPRESS LEADING ZEROS
 TYPE ,67\$::TYPE ASCIZ STRING
 BR 66\$::GET OVER THE ASCIZ
 ::67\$: .ASCIZ / TOTAL ERRORS SINCE LAST REPORT /
 66\$:
 MOV \$ERTTL,-(SP) ::SAVE \$ERTTL FOR TYPEOUT
 ::TOTAL NUMBER OF ERRORS IN OCTAL
 ::GO TYPE--OCTAL ASCII
 TYPOS
 .BYTE 6 ::TYPE 6 DIGITS
 .BYTE 0 ::SUPPRESS LEADING ZEROS
 TYPE ,\$CRLF ::TYPE CARRIAGE RETURN, LINE FEED
 CLR \$ERTTL ::CLEAR ERROR TOTAL
 \$GET42: MOV @#42,R0 ::GET MONITOR ADDRESS
 BEQ \$DOAGN ::BR NCH IF NO MONITOR


```
033346 005046          CLR      -(SP)          ;;INSURE THE 'T' BIT IS CLEAR
033350 012746 033356  MOV      #$CLR.T,-(SP)  ;;SETUP FOR AN RTI OR RTT
033354 000426          BR       $RTRN         ;;GO DO AN RTI OR RTT TO LOAD THE PSW
                                ;;WITH A CLEARED 'T' BIT

033356          $CLR.T:
033356 013700 000042  MOV      @#42,R0        ;;INSURE R0 CONTAINS THE MONITORS
033362 001405          BEQ      $DOAGN        ;;RETURN ADDRESS
033364 000005          RESET          ;;CLEAR THE WORLD
033366 004710          $ENDAD: JSR      PC,(R0)  ;;GO TO MONITOR
033370 000240          NOP              ;;SAVE ROOM
033372 000240          NOP              ;;FOR
033374 000240          NOP              ;;ACT11
033376          $DOAGN:
033376 104400          TRAP          ;;PUSH OLD PSW AND PC ON STACK
033400 042716 000020  BIC      #20,(SP)      ;;CLEAR THE 'T' BIT
033404 032777 010000 145526  BIT      #BIT12,@SWR   ;;RUN WITH TRACE TRAP?
033412 001005          BNE      1$          ;;BR IF NO
033414 005137 033440  COM      $TBIT         ;;IS IT TIME FOR TRACE TRAP
033420 100402          BMI      1$          ;;BR IF NO
033422 052716 000020  BIS      #20,(SP)      ;;SET TRACE TRAP
033426 012746 033434  1$:      MOV      #$LOOP,-(SP)  ;;JUMP TO START OF TEST
033432 000002          $RTRN: RT!        ;;RETURN--THIS IS CHANGED TO
                                ;;AN 'RTT' IF 'RTT' IS A LEGAL
                                ;;INSTRUCTION

033434          $LOOP:
033434 000137          JMP      @(PC)+        ;;RETURN
033436 005030          $RTNAD: .WORD    LOOP
033440 000000          $TBIT:  .WORD    0          ;;'T' BIT STATE INDICATOR
033442      377      000          $ENULL: .BYTE   -1,-1,0      ;;NULL CHARACTER STRING
                                .EVEN
```

6067
6068

.SBTTL SCOPE HANDLER ROUTINE

```
*****
*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
*AND LOAD THE TEST NUMBER($STNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
*AND LOAD THE ERROR FLAG ($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
```

```
033446          $SCOPE:
033446 104406          CKSWR
033450 032777 040000 145462 1$:  BIT      #BIT14,@SWR  ;;TEST FOR CHANGE IN SOFT-SWR
033456 001131          BNE      $OVER      ;;LOOP ON PRESENT TEST?
                                ;;YES IF SW14=1
033460 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)
033462 013746 000004          MOV      @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
033466 012737 033506 000004  MOV      #5$,@#ERRVEC  ;;SET FOR TIMEOUT
033474 005737 177060          TST      @#177060    ;;TIME OUT ON XOR?
033500 012637 000004          MOV      (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
033504 000500          BR       $SVLAD    ;;GO TO THE NEXT TEST
```



```
033506 022626          5$:    CMP      (SP)+,(SP)+      ;;CLEAR THE STACK AFTER A TIME OUT
033510 012637 000004    MOV      (SP)+,@#ERRVEC    ;;RESTORE THE ERROR VECTOR
033514 000440          BR       7$                ;;LOOP ON THE PRESENT TEST
033516          6$:;#####END OF CODE FOR THE XOR TESTER#####
033516 032777 000400 145414    BIT      #BIT08,@SWR      ;;LOOP ON SPEC. TEST?
033524 001404          BEQ      2$                ;;BR IF NO
033526 127737 145406 001102    CMPB    @SWR,$STNM      ;;ON THE RIGHT TEST? SWR<7:0>
033534 001502          BEQ      $OVER          ;;BR IF YES
033536 013737 177766 033760 2$:    MOV      177766,CPSAVE    ;;MOVE CPU ERR REG VALUE TO LOC FOR TST ;DPM001
033544 032737 000001 033760    BIT      #BIT00,CPSAVE    ;;SEE IF THE POWER MONITOR BIT IS ON ;DPM001
033552 001406          BEQ      2000$          ;;BRANCH TO CONTINUE ROUTINE IF CLEAR ;DPM001
033554 042737 000001 177766    BIC      #BIT00,177766    ;;CLEAR THE BIT FOUND TO BE SET ;DPM001
033562 104177          EMT      +177            ;;CALL SPECIAL POWER FAIL BIT ERROR CALL ;DPM001
033564 105037 001103          CLRB    $ERFLG          ;;CLEAR THE ERROR FLAG FOR NEXT TEST ;DPM001
033570 105737 001103    2000$:  TSTB    $ERFLG          ;;WAS THERE AN ERROR?
033574 001421          BEQ      3$                ;;BR IF NO
033576 123737 001115 001103    CMPB    $ERMAX,$ERFLG    ;;MAX. ERRORS FOR THIS TEST OCCURRED?
033604 101015          BHI      3$                ;;BR IF NO
033606 032777 001000 145324    BIT      #BIT09,@SWR      ;;LOOP ON ERROR?
033614 001404          BEQ      4$                ;;BR IF NO
033616 013737 001110 001106 7$:    MOV      $LPERR,$LPADR    ;;SET LOOP ADDRESS TO LAST SCOPE
033624 000446          BR       $OVER          ;;
033626 105037 001103          4$:    CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
033632 005037 001302          CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
033636 000415          BR       1$                ;;ESCAPE TO THE NEXT TEST
033640 032777 004000 145272 3$:    BIT      #BIT11,@SWR      ;;INHIBIT ITERATIONS?
033646 001011          BNE      1$                ;;BR IF YES
033650 005737 001324          TST     $PASS          ;;IF FIRST PASS OF PROGRAM
033654 001406          BEQ      1$                ;;INHIBIT ITERATIONS
033656 005237 001104          INC     $ICNT          ;;INCREMENT ITERATION COUNT
033662 023737 001302 001104    CMP     $TIMES,$ICNT     ;;CHECK THE NUMBER OF ITERATIONS MADE
033670 002024          BGE     $OVER          ;;BR IF MORE ITERATION REQUIRED
033672 012737 000001 001104 1$:    MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
033700 013737 033756 001302    MOV     $MXCNT,$TIMES    ;;SET NUMBER OF ITERATIONS TO DO
033706 105237 001102          $SVLAD: INCB    $STNM          ;;COUNT TEST NUMBERS
033712 113737 001102 001322    MOVB   $STNM,$TESTN     ;;SET TEST NUMBER IN APT MAILBOX
033720 011637 001106          MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
033724 011637 001110          MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
033730 005037 001304          CLR     $ESCAPE        ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
033734 112737 000001 001115    MOVB   #1,$ERMAX        ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
033742 013777 001102 145172 $OVER:  MOV     $STNM,@DISPLAY   ;;DISPLAY TEST NUMBER
033750 013716 001106          MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
033754 000002          RTI                    ;;FIXES PS
033756 000001          $MXCNT: 1                ;;MAX. NUMBER OF ITERATIONS
033760 000000          CPSAVE: .WORD 0        ;;LOCATION TO SAVE CPU ERR REG CONTENTS ;DPM001
```

6069
6070

.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
033762 000000      IBSAVE: .WORD      0      ;LOC'N TO HOLD $ERRPC DURING DUAL ERR ;DPM001
033764      $ERROR:
033764 104406      CKSWR      ;;TEST FOR CHANGE IN SOFT-SWR
033766 105237 001103 7$:      INCB      $ERFLG      ;;SET THE ERROR FLAG
033772 001775      BEQ      7$      ;;DON'T LET THE FLAG GO TO ZERO
033774 013777 001102 145140      MOV      $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
034002 032777 002000 145130      BIT      #BIT10,@SWR    ;;BELL ON ERROR?
034010 001402      BEQ      1$      ;;NO - SKIP
034012 104401 001306      TYPE     ,SBELL      ;;RING BELL
034016 005237 001112 1$:      INC      $ERTTL      ;;COUNT THE NUMBER OF ERRORS
034022 011637 001116      MOV      (SP),$ERRPC   ;;GET ADDRESS OF ERROR INSTRUCTION
034026 162737 000002 001116      SUB      #2,$ERRPC
034034 117737 145056 001114      MOV      @ERRPC,$ITEMB ;;STRIP AND SAVE THE ERROR ITEM CODE
034042 122737 000177 001114      CMPB    #177,$ITEMB   ;;SEE IF THIS IS THE POWER FAIL CALL ;DPM001
034050 001421      BEQ      1000$      ;;BRANCH AROUND ROUTINE IF IT IS ;DPM001
034052 013737 177766 033760      MOV      177766,CPSAVE ;;MOVE CPU ERR REG TO CPSAVE FOR TEST ;DPM001
034060 032737 000001 033760      BIT      #BIT00,CPSAVE ;;SEE IF POWER MONITOR BIT IS SET ;DPM001
034066 001412      BEQ      1000$      ;;BRANCH IF OK ;DPM001
034070 042737 000001 177766      BIC      #BIT00,177766 ;;CLEAR THE BIT FOUND SET ;DPM001
034076 013737 001116 033762      MOV      $ERRPC,IBSAVE ;;SAVE $ERRPC ;DPM001
034104 104177      ERROR   +177      ;;CALL SPECIAL POWER MON BIT ERROR ;DPM001
034106 013737 033762 001116      MOV      IBSAVE,$ERRPC ;;RESTORE $ERRPC
034114
034114 032777 020000 145016 1000$:  BIT      #BIT13,@SWR    ;;SKIP TYPEOUT IF SET
034122 001004      BNE     20$      ;;SKIP TYPEOUTS
034124 004737 036370      JSR     PC,ERTYPE   ;;GO TO USER ERROR ROUTINE
034130 104401 001313      TYPE     ,SCLRF
034134
034134 122737 000001 001336 20$:   CMPB    #APTENV,$ENV  ;;RUNNING IN APT MODE
034142 001007      BNE     2$      ;;NO,SKIP APT ERROR REPORT
034144 113737 001114 034156      MOV      $ITEMB,21$   ;;SET ITEM NUMBER AS ERROR NUMBER
034152 004737 035222      JSR     PC,$ATY4     ;;REPORT FATAL ERROR TO APT
034156 000      21$:   .BYTE   0
034157 000      .BYTE   0
034160 000777      22$:   BR      22$      ;;APT ERROR LOOP
034162 005737 033762 2$:   TST     IBSAVE      ;;SEE IF POWER FAIL ERROR CALL ;DPM001
034166 001005      BNE     3$      ;;BRANCH IF NOT - HALT NOT ALLOWED ;DPM001
034170 005777 144744      TST     @SWR
034174 100002      BPL     3$      ;;SKIP IF CONTINUE
034176 000000      HALT
034200 104406      CKSWR      ;;TEST FOR CHANGE IN SOFT-SWR
034202 032777 001000 144730 3$:   BIT      #BIT09,@SWR  ;;LOOP ON ERROR SWITCH SET?
034210 001405      BEQ     4$      ;;BR IF NO
034212 005737 033762      TST     IBSAVE      ;;SEE IF THIS IS THE PWR MNTR BIT ERROR ;DPM001
034216 001002      BNE     4$      ;;BRANCH IF SO - NO FUDGING ALLOWED ;DPM001
034220 013716 001110      MOV     $LPERR,(SP)  ;;FUDGE RETURN FOR LOOPING
034224 005737 001304 4$:   TST     $ESCAPE     ;;CHECK FOR AN ESCAPE ADDRESS
034230 001405      BEQ     5$      ;;BR IF NONE
034232 005737 033762      TST     IBSAVE      ;;SEE IF THIS IS THE PWR MNTR BIT ERROR ;DPM001
034236 001002      BNE     5$      ;;BRANCH IF SO - NO FUDGING ALLOWED ;DPM001
034240 013716 001304      MOV     $ESCAPE,(SP) ;;FUDGE RETURN ADDRESS FOR ESCAPE
034244
034244 022737 033366 000042 5$:   CMP     #SENDAD,42   ;;ACT-11 AUTO-ACCEPT?
034252 001001      BNE     6$      ;;BRANCH IF NO
  
```

6071
6072

```
034254 000000          HALT          ;;YES
034256 032777 001000 144654 6$:      BIT      #BIT09,@SWR
034264 001013          BNE      ERM10
034266 011637 001162          MOV      (SP), $REGO          ;SEE IF ERROR #377
034272 062737 177776 001162          ADD      #-2, $REGO
034300 122777 000377 144654          CMPB    #377,@$REGO
034306 001002          BNE      ERM10
034310 062716 000002          ADD      #2,(SP)
034314 000002          ERM10:    RTI
```

```
.SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
:*****
:*SAVE R0-R5
:*CALL:
:*      SAVREG
:*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
:*
:*TOP---(+16)
:* +2---(+18)
:* +4---R5
:* +6---R4
:* +8---R3
:*+10---R2
:*+12---R1
:*+14---R0
```

```
034316 010046          $SAVREG:  MOV      R0,-(SP)          ;;PUSH R0 ON STACK
034320 010146          MOV      R1,-(SP)          ;;PUSH R1 ON STACK
034322 010246          MOV      R2,-(SP)          ;;PUSH R2 ON STACK
034324 010346          MOV      R3,-(SP)          ;;PUSH R3 ON STACK
034326 010446          MOV      R4,-(SP)          ;;PUSH R4 ON STACK
034330 010546          MOV      R5,-(SP)          ;;PUSH R5 ON STACK
034332 016646 000022          MOV      22(SP),-(SP)     ;;SAVE PS OF MAIN FLOW
034336 016646 000022          MOV      22(SP),-(SP)     ;;SAVE PC OF MAIN FLOW
034342 016646 000022          MOV      22(SP),-(SP)     ;;SAVE PS OF CALL
034346 016646 000022          MOV      22(SP),-(SP)     ;;SAVE PC OF CALL
034352 000002          RTI
```

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

6073
6074

```
.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

034412	105737	001157	\$TYPE:	TSTB	\$TPFLG	:: IS THERE A TERMINAL?
034416	100002			BPL	1\$:: BR IF YES
034420	000000			HALT		:: HALT HERE IF NO TERMINAL
034422	000430			BR	3\$:: LEAVE
034424	010046		1\$:	MOV	RO,-(SP)	:: SAVE RO
034426	017600	000002		MOV	@2(SP),RO	:: GET ADDRESS OF ASCIZ STRING
034432	122737	000001 001336		CMPB	#APTENV,\$ENV	:: RUNNING IN APT MODE
034440	001011			BNE	62\$:: NO,GO CHECK FOR APT CONSOLE
034442	132737	000100 001337		BITB	#APTSPOOL,\$ENVM	:: SPOOL MESSAGE TO APT
034450	001405			BEQ	62\$:: NO,GO CHECK FOR CONSOLE
034452	010037	034462		MOV	RO,61\$:: SETUP MESSAGE ADDRESS FOR APT
034456	004737	035212		JSR	PC,\$ATY3	:: SPOOL MESSAGE TO APT
034462	000000		61\$:	.WORD	0	:: MESSAGE ADDRESS
034464	132737	000040 001337	62\$:	BITB	#APTCSUP,\$ENVM	:: APT CONSOLE SUPPRESSED
034472	001003			BNE	60\$:: YES,SKIP TYPE OUT
034474	112046		2\$:	MOVB	(RO)+,-(SP)	:: PUSH CHARACTER TO BE TYPED ONTO STACK
034476	001005			BNE	4\$:: BR IF IT ISN'T THE TERMINATOR
034500	005726			TST	(SP)+	:: IF TERMINATOR POP IT OFF THE STACK
034502	012600		60\$:	MOV	(SP)+,RO	:: RESTORE RO
034504	062716	000002	3\$:	ADD	#2,(SP)	:: ADJUST RETURN PC
034510	000002			RTI		:: RETURN
034512	122716	000011	4\$:	CMPB	#HT,(SP)	:: BRANCH IF <HT>
034516	001430			BEQ	8\$	
034520	122716	000200		CMPB	#CRLF,(SP)	:: BRANCH IF NOT <CRLF>
034524	001006			BNE	5\$	
034526	005726			TST	(SP)+	:: POP <CR><LF> EQUIV
034530	104401			TYPE		:: TYPE A CR AND LF
034532	001313			\$CRLF		
034534	105037	034752		CLRB	\$CHARCNT	:: CLEAR CHARACTER COUNT
034540	000755			BR	2\$:: GET NEXT CHARACTER
034542	004737	034624	5\$:	JSR	PC,\$TYPEC	:: GO TYPE THIS CHARACTER
034546	123726	001156	6\$:	CMPB	\$FILLC,(SP)+	:: IS IT TIME FOR FILLER CHARS.?
034552	001350			BNE	2\$:: IF NO GO GET NEXT CHAR.
034554	013746	001154		MOV	\$NULL,-(SP)	:: GET # OF FILLER CHARS. NEEDED
						:: AND THE NULL CHAR.
034560	105366	000001	7\$:	DECB	1(SP)	:: DOES A NULL NEED TO BE TYPED?
034564	002770			BLT	6\$:: BR IF NO--GO POP THE NULL OFF OF STACK
034566	004737	034624		JSR	PC,\$TYPEC	:: GO TYPE A NULL
034572	105337	034752		DECB	\$CHARCNT	:: DO NOT COUNT AS A COUNT
034576	000770			BR	7\$:: LOOP
				:HORIZONTAL TAB PROCESSOR		
034600	112716	000040	8\$:	MOVB	#'(SP)	:: REPLACE TAB WITH SPACE
034604	004737	034624	9\$:	JSR	PC,\$TYPEC	:: TYPE A SPACE
034610	132737	000007 034752		BITB	#7,\$CHARCNT	:: BRANCH IF NOT AT
034616	001372			BNE	9\$:: TAB STOP

```

034620 005726          TST      (SP)+      ;;POP SPACE OFF STACK
034622 000724          BR        2$          ;;GET NEXT CHARACTER
034624          $TYPEPC: TSTB     @$TKS      ;;CHAR IN KYBD BUFFER?      :MJD001
034624 105777 144314  BPL      10$          ;;BR IF NOT                :MJD001
034630 100022          MOV      @$TKB,-(SP)  ;;GET CHAR                  :MJD001
034632 017746 144310  BIC      #177600,(SP) ;;STRIP EXTRANEIOUS BITS  :MJD001
034636 042716 177600  CMPB     #$XOFF,(SP)  ;;WAS CHAR XOFF           :MJD001
034642 122716 000023  BNE      102$        ;;BR IF NOT                :MJD001
034646 001012          101$: TSTB     @$TKS      ;;WAIT FOR CHAR            :MJD001
034650          BPL      101$          ;;BR IF NOT                :MJD001
034650 105777 144270  MOVB     @$TKB,(SP)  ;;GET CHAR                  :MJD001
034654 100375          BIC      #177600,(SP) ;;STRIP IT                  :MJD001
034656 117716 144264  CMPB     #$XON,(SP)  ;;WAS IT XON?             :MJD001
034662 042716 177600  BNE      101$        ;;BR IF NOT                :MJD001
034666 122716 000021  102$: TST      (SP)+      ;;FIX STACK                :MJD001
034672 001366          10$: TSTB     @$TPS      ;;WAIT UNTIL PRINTER IS READY :MJD001
034674 005726          BPL      10$          ;;BR IF NOT                :MJD001
034676 105777 144246  CMPB     2(SP),#$XON  ;;IS CHARACTER A RANDOM XON? :RAN001
034702 100375          BEQ     $TYPEX      ;;BRANCH IF YES           :RAN001
034704 126627 000002 000021  MOVB     2(SP),@$TPB  ;;LOAD CHAR TO BE TYPED INTO DATA REG.
034712 001420          CMPB     #CR,2(SP)  ;;IS CHARACTER A CARRIAGE RETURN?
034714 116677 000002 144230  BNE      1$          ;;BRANCH IF NO
034722 122766 000015 000002  CLRB     $CHARCNT   ;;YES--CLEAR CHARACTER COUNT
034730 001003          BR      $TYPEX      ;;EXIT
034732 105037 034752  1$: CMPB     #LF,2(SP)  ;;IS CHARACTER A LINE FEED?
034736 000406          BEQ     $TYPEX      ;;BRANCH IF YES
034740 122766 000012 000002  INCB     (PC)+      ;;COUNT THE CHARACTER
034746 001402          $CHARCNT: .WORD 0  ;;CHARACTER COUNT STORAGE
034750 105227          $TYPEX: RTS      PC
034752 000000
034754 000207

```

6075
6076

```

.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

```



```

034756 017646 000000 $TYPOS: MOV @ (SP),-(SP) ;;PICKUP THE MODE
034762 116637 000001 035201 MOV 1(SP), $OFILL ;;LOAD ZERO FILL SWITCH
034770 112637 035203 MOV (SP)+, $OMODE+1 ;;NUMBER OF DIGITS TO TYPE
034774 062716 000002 ADD #2, (SP) ;;ADJUST RETURN ADDRESS
035000 000406 BR $TYPON
035002 112737 000001 035201 $TYPOC: MOV #1, $OFILL ;;SET THE ZERO FILL SWITCH
035010 112737 000006 035203 MOV #6, $OMODE+1 ;;SET FOR SIX(6) DIGITS
035016 112737 000005 035200 $TYPON: MOV #5, $OCNT ;;SET THE ITERATION COUNT
035024 010346 MOV R3, -(SP) ;;SAVE R3
035026 010446 MOV R4, -(SP) ;;SAVE R4
035030 010546 MOV R5, -(SP) ;;SAVE R5
035032 113704 035203 MOV $OMODE+1, R4 ;;GET THE NUMBER OF DIGITS TO TYPE
035036 005404 NEG R4
035040 062704 000006 ADD #6, R4 ;;SUBTRACT IT FOR MAX. ALLOWED
035044 110437 035202 MCVB R4, $OMODE ;;SAVE IT FOR USE
035050 113704 035201 MOV $OFILL, R4 ;;GET THE ZERO FILL SWITCH
035054 016605 000012 MOV 12(SP), R5 ;;PICKUP THE INPUT NUMBER
035060 005003 CLR R3 ;;CLEAR THE OUTPUT WORD
035062 006105 1$: ROL R5 ;;ROTATE MSB INTO 'C'
035064 000404 BR 3$ ;;GO DO MSB
035066 006105 2$: ROL R5 ;;FORM THIS DIGIT
035070 006105 ROL R5
035072 006105 ROL R5
035074 010503 MOV R5, R3
035076 006103 3$: ROL R3 ;;GET LSB OF THIS DIGIT
035100 105337 035202 DECB $OMODE ;;TYPE THIS DIGIT?
035104 100016 BPL 7$ ;;BR IF NO
035106 042703 177770 BIC #177770, R3 ;;GET RID OF JUNK
035112 001002 BNE 4$ ;;TEST FOR 0
035114 005704 TST R4 ;;SUPPRESS THIS 0?
035116 001403 BEQ 5$ ;;BR IF YES
035120 005204 4$: INC R4 ;;DON'T SUPPRESS ANYMORE 0'S
035122 052703 000060 BIS #'0, R3 ;;MAKE THIS DIGIT ASCII
035126 052703 000040 5$: BIS #' , R3 ;;MAKE ASCII IF NOT ALREADY
035132 110337 035176 MOV R3, 8$ ;;SAVE FOR TYPING
035136 104401 035176 TYPE , 8$ ;;GO TYPE THIS DIGIT
035142 105337 035200 7$: DECB $OCNT ;;COUNT BY 1
035146 003347 BGT 2$ ;;BR IF MORE TO DO
035150 002402 BLT 6$ ;;BR IF DONE
035152 005204 INC R4 ;;INSURE LAST DIGIT ISN'T A BLANK
035154 000744 BR 2$ ;;GO DO THE LAST DIGIT
035156 012605 6$: MOV (SP)+, R5 ;;RESTORE R5
035160 012604 MOV (SP)+, R4 ;;RESTORE R4
035162 012603 MOV (SP)+, R3 ;;RESTORE R3
035164 016666 000002 000004 MOV 2(SP), 4(SP) ;;SET THE STACK FOR RETURNING
035172 012616 MOV (SP)+, (SP)
035174 000002 RTI ;;RETURN
035176 000 8$: .BYTE 0 ;;STORAGE FOR ASCII DIGIT
035177 000 .BYTE 0 ;;TERMINATOR FOR TYPE ROUTINE
035200 000 $OCNT: .BYTE 0 ;;OCTAL DIGIT COUNTER
035201 000 $OFILL: .BYTE 0 ;;ZERO FILL SWITCH
035202 000000 $OMODE: .WORD 0 ;;NUMBER OF DIGITS TO TYPE
    
```

6077
6078

.SBTTL APT COMMUNICATIONS ROUTINE

```

035204 112737 000001 035450 $ATV1: MOV #1, $FFLG ;;TO REPORT FATAL ERROR
035212 112737 000001 035446 $ATV3: MOV #1, $MFLG ;;TO TYPE A MESSAGE
    
```

```
035220 000403 BR $ATYC
035222 112737 000001 035450 $ATY4: MOVB #1,$FFLG ::TO ONLY REPORT FATAL ERROR
035230 $ATYC:
035230 010046 MOV R0,-(SP) ::PUSH R0 ON STACK
035232 010146 MOV R1,-(SP) ::PUSH R1 ON STACK
035234 105737 035446 TSTB $MFLG ::SHOULD TYPE A MESSAGE?
035240 001450 BEQ 5$ ::IF NOT: BR
035242 122737 000001 001336 CMPB #APTENV,$ENV ::OPERATING UNDER APT?
035250 001031 BNE 3$ ::IF NOT: BR
035252 132737 000100 001337 BITB #APTSPool,$ENVM ::SHOULD SPOOL MESSAGES?
035260 001425 BEQ 3$ ::IF NOT: BR
035262 017600 000004 MOV @4(SP),R0 ::GET MESSAGE ADDR.
035266 062766 000002 000004 ADD #2,4(SP) ::BUMP RETURN ADDR.
035274 005737 001316 1$: TST $MSGTYPE ::SEE IF DONE W/ LAST XMISSION?
035300 001375 BNE 1$ ::IF NOT: WAIT
035302 010037 001332 MOV R0,$MSGAD ::PUT ADDR IN MAILBOX
035306 105720 2$: TSTB (R0)+ ::FIND END OF MESSAGE
035310 001376 BNE 2$
035312 163700 001332 SUB $MSGAD,R0 ::SUB START OF MESSAGE
035316 006200 ASR R0 ::GET MESSAGE LNGTH IN WORDS
035320 010037 001334 MOV R0,$MSGGLT ::PUT LENGTH IN MAILBOX
035324 012737 000004 001316 MOV #4,$MSGTYPE ::TELL APT TO TAKE MSG.
035332 000413 BR 5$
035334 017637 000004 035360 3$: MOV @4(SP),4$ ::PUT MSG ADDR IN JSR LINKAGE
035342 062766 000002 000004 ADD #2,4(SP) ::BUMP RETURN ADDRESS
035350 013746 177776 MOV 177776,-(SP) ::PUSH 177776 ON STACK
035354 004737 034412 JSR PC,$TYPE ::CALL TYPE MACRO
035360 000000 4$: .WORD 0
035362 5$:
035362 105737 035450 10$: TSTB $FFLG ::SHOULD REPORT FATAL ERROR?
035366 001416 BEQ 12$ ::IF NOT: BR
035370 005737 001336 TST $ENV ::RUNNING UNDER APT?
035374 001413 BEQ 12$ ::IF NOT: BR
035376 005737 001316 11$: TST $MSGTYPE ::FINISHED LAST MESSAGE?
035402 001375 BNE 11$ ::IF NOT: WAIT
035404 017637 000004 001320 MOV @4(SP),$FATAL ::GET ERROR #
035412 062766 000002 000004 ADD #2,4(SP) ::BUMP RETURN ADDR.
035420 005237 001316 INC $MSGTYPE ::TELL APT TO TAKE ERROR
035424 105037 035450 12$: CLRB $FFLG ::CLEAR FATAL FLAG
035430 105037 035447 CLRB $LFLG ::CLEAR LOG FLAG
035434 105037 035446 CLRB $MFLG ::CLEAR MESSAGE FLAG
035440 012601 MOV (SP)+,R1 ::POP STACK INTO R1
035442 012600 MOV (SP)+,R0 ::POP STACK INTO R0
035444 000207 RTS PC ::RETURN
035446 000 $MFLG: .BYTE 0 ::MESSG. FLAG
035447 000 $LFLG: .BYTE 0 ::LOG FLAG
035450 000 $FFLG: .BYTE 0 ::FATAL FLAG
.EVEN
```

000200
000001
000100
000040
APTSIZE=200
APTENV=001
APTSPool=100
APTCSUP=040

6079
6080

```
.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.

035452	022737	000176	001140	\$CKSWR:	CMP	#SWREG,SWR	:: IS THE SOFT-SWR SELECTED?
035460	001074				BNE	15\$:: BRANCH IF NO
035462	105777	143456			TSTB	@\$TKS	:: CHAR THERE?
035466	100071				BPL	15\$:: IF NO, DON'T WAIT AROUND
035470	117746	143452			MOVB	@\$TKB,-(SP)	:: SAVE THE CHAR
035474	042716	177600			BIC	#^C177,(SP)	:: STRIP-OFF THE ASCII
035500	022726	000007			CMP	#7,(SP)+	:: IS IT A CONTROL G?
035504	001062				BNE	15\$:: NO, RETURN TO USER
035506	123727	001134	000001		CMPB	\$AUTOB,#1	:: ARE WE RUNNING IN AUTO-MODE?
035514	001456				BEQ	15\$:: BRANCH IF YES
035516	104401	036071			TYPE	,\$CNTLG	:: ECHO THE CONTROL-G (^G)
035522	104401	036076		\$GTSWR:	TYPE	,\$MSWR	:: TYPE CURRENT CONTENTS
035526	013746	000176			MOV	SWREG,-(SP)	:: SAVE SWREG FOR TYPEOUT
035532	104402				TYPOC		:: GO TYPE--OCTAL ASCII(ALL DIGITS)
035534	104401	036107			TYPE	,\$MNEW	:: PROMPT FOR NEW SWR
035540	005046			19\$:	CLR	-(SP)	:: CLEAR COUNTER
035542	005046				CLR	-(SP)	:: THE NEW SWR
035544	105777	143374		7\$:	TSTB	@\$TKS	:: CHAR THERE?
035550	100375				BPL	7\$:: IF NOT TRY AGAIN
035552	117746	143370			MOVB	@\$TKB,-(SP)	:: PICK UP CHAR
035556	042716	177600			BIC	#^C177,(SP)	:: MAKE IT 7-BIT ASCII
035562	021627	000025		9\$:	CMP	(SP),#25	:: IS IT A CONTROL-U?
035566	001005				BNE	10\$:: BRANCH IF NOT
035570	104401	036064			TYPE	,\$CNTLU	:: YES, ECHO CONTROL-U (^U)
035574	062706	000006		20\$:	ADD	#6,SP	:: IGNORE PREVIOUS INPUT
035600	000757				BR	19\$:: LET'S TRY IT AGAIN
035602	021627	000015		10\$:	CMP	(SP),#15	:: IS IT A <CR>?
035606	001022				BNE	16\$:: BRANCH IF NO
035610	005766	000004			TST	4(SP)	:: YES, IS IT THE FIRST CHAR?
035614	001403				BEQ	11\$:: BRANCH IF YES
035616	016677	000002	143314		MOV	2(SP),@SWR	:: SAVE NEW SWR
035624	062706	000006		11\$:	ADD	#6,SP	:: CLEAR UP STACK
035630	104401	001313		14\$:	TYPE	,\$CRLF	:: ECHO <CR> AND <LF>
035634	123727	001135	000001		CMPB	\$INTAG,#1	:: RE-ENABLE TTY KBD INTERRUPTS?
035642	001003				BNE	15\$:: BRANCH IF NOT
035644	012777	000100	143272		MOV	#100,@\$TKS	:: RE-ENABLE TTY KBD INTERRUPTS
035652	000002			15\$:	RTI		:: RETURN
035654	004737	034624		16\$:	JSR	PC,\$TYPEC	:: ECHO CHAR
035660	021627	000060			CMP	(SP),#60	:: CHAR < 0?
035664	002420				BLT	18\$:: BRANCH IF YES
035666	021627	000067			CMP	(SP),#67	:: CHAR > 7?
035672	003015				BGT	18\$:: BRANCH IF YES
035674	042726	000060			BIC	#60,(SP)+	:: STRIP-OFF ASCII
035700	005766	000002			TST	2(SP)	:: IS THIS THE FIRST CHAR
035704	001403				BEQ	17\$:: BRANCH IF YES
035706	006316				ASL	(SP)	:: NO, SHIFT PRESENT
035710	006316				ASL	(SP)	:: CHAR OVER TO MAKE
035712	006316				ASL	(SP)	:: ROOM FOR NEW ONE.
035714	005266	000002		17\$:	INC	2(SP)	:: KEEP COUNT OF CHAR
035720	056616	177776			BIS	-2(SP),(SP)	:: SET IN NEW CHAR
035724	000707				BR	7\$:: GET THE NEXT ONE
035726	104401	001312		18\$:	TYPE	,\$QUES	:: TYPE ?<CR><LF>
035732	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
```

035734	011646			\$RDCHR: MOV	(SP),-(SP)	::PUSH DOWN THE PC	
035736	016666	000004	000002	MOV	4(SP),2(SP)	::SAVE THE PS	
035744	105777	143174		1\$: TSTB	@\$TKS	::WAIT FOR	
035750	100375			BPL	1\$::A CHARACTER	
035752	117766	143170	000004	MOVB	@\$TKB,4(SP)	::READ THE TTY	
035760	042766	177600	000004	BIC	#\$C<177>,4(SP)	::GET RID OF JUNK IF ANY	
035766	026627	000004	000023	CMP	4(SP),#23	::IS IT A CONTROL-S?	
035774	001013			BNE	3\$::BRANCH IF NO	
035776	105777	143142		2\$: TSTB	@\$TKS	::WAIT FOR A CHARACTER	
036002	100375			BPL	2\$::LOOP UNTIL ITS THERE	
036004	117746	143136		MOVB	@\$TKB,-(SP)	::GET CHARACTER	
036010	042716	177600		BIC	#\$C177,(SP)	::MAKE IT 7-BIT ASCII	
036014	022627	000021		CMP	(SP)+,#21	::IS IT A CONTROL-Q?	
036020	001366			BNE	2\$::IF NOT DISCARD IT	
036022	000750			BR	1\$::YES, RESUME	
036024	026627	000004	000021	3\$: CMP	4(SP),#\$XON	::IS IT A RANDOM XON?	:RAN001
036032	001744			BEQ	1\$::BRANCH IF YES	:RAN001
036034	026627	000004	000140	CMP	4(SP),#140	::IS IT UPPER CASE?	
036042	002407			BLT	4\$::BRANCH IF YES	
036044	026627	000004	000175	CMP	4(SP),#175	::IS IT A SPECIAL CHAR?	
036052	003003			BGT	4\$::BRANCH IF YES	
036054	042766	000040	000004	BIC	#\$40,4(SP)	::MAKE IT UPPER CASE	
036062	000002			4\$: RTI		::GO BACK TO USER	
036064	136	125	015	\$CNTLU: .ASCIZ	/^U/<15><12>	::CONTROL 'U'	
036071	136	107	015	\$CNTLG: .ASCIZ	/^G/<15><12>	::CONTROL 'G'	
036076	015	012	123	\$MSWR: .ASCIZ	<15><12>/SWR = /		
036107	040	040	116	\$MNEW: .ASCIZ	/ NEW = /		

6081
6082

```
.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.
```

036120	010046			\$TRAP: MOV	R0,-(SP)	::SAVE R0		
036122	016600	000002		MOV	2(SP),R0	::GET TRAP ADDRESS		
036126	005740			TST	-(R0)	::BACKUP BY 2		
036130	111000			MOVB	(R0),R0	::GET RIGHT BYTE OF TRAP		
036132	006300			ASL	R0	::POSITION FOR INDEXING		
036134	016000	036154		MOV	\$TRPAD(R0),R0	::INDEX TO TABLE		
036140	000200			RTS	R0	::GO TO ROUTINE		
				::THIS IS USE TO HANDLE THE 'GETPRI' MACRO				
036142	011646			\$TRAP2: MOV	(SP),-(SP)	::MOVE THE PC DOWN		
036144	016666	000004	000002	MOV	4(SP),2(SP)	::MOVE THE PSW DOWN		
036152	000002			RTI		::RESTORE THE PSW		

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


036154 036142
036156 034412
036160 035002
036162 034756
036164 035016
036166 035522
036170 035452
036172 035734
036174 034316
036176 034354
6083 036200 037074
6084 036202 037066
6085 000030
6086
6087

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

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

036204 012737 036362 000024
036212 012737 000340 000026
036220 010046
036222 010146
036224 010246
036226 010346
036230 010446
036232 010546
036234 017746 142700
036240 010637 036366
036244 012737 036256 000024
036252 000000
036254 000776

```
:POWER DOWN ROUTINE  
$PWRDN: MOV $ILLUP,@#PWRVEC ::SET FOR FAST UP  
MOV #340,@#PWRVEC+2 ::PRIO:7  
MOV R0,-(SP) ::PUSH R0 ON STACK  
MOV R1,-(SP) ::PUSH R1 ON STACK  
MOV R2,-(SP) ::PUSH R2 ON STACK  
MOV R3,-(SP) ::PUSH R3 ON STACK  
MOV R4,-(SP) ::PUSH R4 ON STACK  
MOV R5,-(SP) ::PUSH R5 ON STACK  
MOV @SWR,-(SP) ::PUSH @SWR ON STACK  
MOV SP,$SAVR6 ::SAVE SP  
MOV #PWRUP,@#PWRVEC ::SET UP VECTOR  
HALT  
BR -2 ::HANG UP  
:*****
```

036256 012737 036362 000024
036264 013706 036366
036270 005037 036366
036274 005237 036366
036300 001375
036302 012677 142632
036306 012605
036310 012604
036312 012603
036314 012602
036316 012601
036320 012600
036322 012737 036204 000024
036330 012737 000340 000026
036336 104401
036340 037270
036342 012716
036344 004346
036346 042766 000020 000002
036354 005037 033440
036360 000002
036362 000000
036364 000776
036366 000000

```
:POWER UP ROUTINE  
$PWRUP: MOV $ILLUP,@#PWRVEC ::SET FOR FAST DOWN  
MOV $SAVR6,SP ::GET SP  
CLR $SAVR6 ::WAIT LOOP FOR THE TTY  
1$: INC $SAVR6 ::WAIT FOR THE INC  
BNE 1$ ::OF WORD  
MOV (SP)+,@SWR ::POP STACK INTO @SWR  
MOV (SP)+,R5 ::POP STACK INTO R5  
MOV (SP)+,R4 ::POP STACK INTO R4  
MOV (SP)+,R3 ::POP STACK INTO R3  
MOV (SP)+,R2 ::POP STACK INTO R2  
MOV (SP)+,R1 ::POP STACK INTO R1  
MOV (SP)+,R0 ::POP STACK INTO R0  
MOV #PWRDN,@#PWRVEC ::SET UP THE POWER DOWN VECTOR  
MOV #340,@#PWRVEC+2 ::PRIO:7  
TYPE ::REPORT THE POWER FAILURE  
$PWRMG: .WORD POWERM ::POWER FAIL MESSAGE POINTER  
MOV (PC)+,(SP) ::RESTART AT START  
$PWRAD: .WORD START ::RESTART ADDRESS  
BIC #20,2(SP) ::CLEAR 'T' BIT  
CLR $TBIT ::CLEAR THE 'T' BIT FLAG  
RTI  
$ILLUP: HALT ::THE POWER UP SEQUENCE WAS STARTED  
BR -2 ::BEFORE THE POWER DOWN WAS COMPLETE  
$SAVR6: 0 ::PUT THE SP HERE
```


6088
6089
6090
6091

6092
6093
6094
6095
6096
6097

6098	036370	104401	001313	
6099	036374	113737	001102	001232
6100	036402	042737	177400	001232
6101	036410	013737	001116	001234
6102	036416	010046		
6103	036420	113700	001114	
6104	036424	042700	177400	
6105	036430	001005		
6106	036432	013746	001116	
6107	036436	104402		
6108	036440	000137	036740	
6109	036444	022700	000377	
6110	036450	001005		
6111	036452	016600	000004	
6112	036456	011000		
6113	036460	062700	000400	
6114	036464	010037	001320	
6115	036470	005300		
6116	036472	006300		
6117	036474	006300		
6118	036476	006300		
6119	036500	062700	001442	
6120	036504	012037	036514	
6121	036510	001404		
6122	036512	104401		
6123	036514	000000		
6124	036516	104401	001313	
6125	036522	012037	036532	
6126	036526	001404		
6127	036530	104401		
6128	036532	000000		
6129	036534	104401	001313	
6130	036540	010146		
6131	036542	010246		
6132	036544	010346		
6133	036546	012001		
6134	036550	001503		
6135	036552	011000		
6136	036554	105710		
6137	036556	001003		
6138	036560	013146		
6139	036562	104402		
6140	036564	000463		
6141	036566	122710	000002	
6142	036572	001005		
6143	036574	004737	036764	

.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      , $CRLF      ;TYPE A CRLF
        MOVVB   $TSTNM,$TMPO   ;MOVE TEST NUMBER TO $TMPO
        BIC     #177400,$TMPO  ;CLEAR THE UPPER BYTE
        MOV     $ERRPC,$TMP1   ;GET PC OF CALL
        MOV     R0,-(SP)       ;SAVE R0
        MOVVB  $ITEMB,R0      ;GET THE ITEM NUMBER.
        BIC     #177400,R0     ;CLEAR THE UPPER BYTE
        BNE     1$            ;BRANCH IF ITEM IS ZERO
        MOV     $ERRPC,-(SP)   ;MOVE THE ERROR PC TO THE STACK FOR PRINTING
        TYPOC                    ;PRINT THE PC
        JMP     22$           ;JUMP TO EXIT - ALL DONE
1$:     CMP     #377,R0        ;SEE IF ERROR # IS 377
        BNE     3$            ;BRANCH IF NOT
        MOV     4(SP),R0       ;MOVE ITEM ADDRESS ON STACK TO R0
        MOV     (R0),R0        ;MOVE ITEM TO R0
        ADD     #400,R0        ;ADD 400 TO R0
3$:     MOV     R0,$FATAL      ;SET ITEM NUMBER IN $FATAL FOR APT      ;DPM001
        DEC     R0             ;DECREMENT R0 AND
        ASL     R0             ;SHIFT THREE TIMES
        ASL     R0             ;TO FORM AN INDEX
        ASL     R0             ;FOR THE TABLE.
        ADD     #$ERRTB,R0     ;ADD ERROR TABLE START TO R0 - FORMS STARTING ADDRESS
4$:     MOV     (R0)+,$5$      ;PICK UP THE ADDRESS OF THE ERROR MESSAGE
        BEQ     6$            ;BRANCH IF NONE
        TYPE                    ;TYPE THE MESSAGE
5$:     .WORD   0              ;LOCATION FOR ASCII MESSAGE ADDRESS
        TYPE      , $CRLF      ;TYPE A <CRLF>
6$:     MOV     (R0)+,$7$      ;GET THE DATA HEADER
        BEQ     8$            ;BRANCH IF NO HEADER
        TYPE                    ;TYPE THE HEADER
7$:     .WORD   0              ;LOCATION FOR HEADER ADDRESS
        TYPE      , $CRLF      ;TYPE A <CRLF>
8$:     MOV     R1,-(SP)       ;SAVE R1
        MOV     R2,-(SP)       ;SAVE R2
        MOV     R3,-(SP)       ;SAVE R3
        MOV     (R0)+,R1       ;GET VHE ADDRESS OF THE DATA TABLE.
        BEQ     23$           ;BRANCH TO RETURN IF NO DATA.
        MOV     (R0),R0        ;GET A POINTER TO THE DATA FORMAT TABLE.
9$:     TSTB   (R0)           ;IS THE FORMAT ZERO?
        BNE     10$          ;BRANCH TO CHECK FOR FORMAT 2 IF NOT
        MOV     @R1+,-(SP)    ;FORMAT ZERO. SO TYPE
        TYPOC                    ;AN OCTAL NUMBER.
        BR     21$           ;BRANCH TO PRINT A TAB CHARACTER AFTER TABLE ENTRY
10$:    CMPB   #2,(R0)        ;IS THE FORMAT TWO?
        BNE     11$          ;BRANCH TO CHECK FOR FORMAT 3 IF NOT
        JSR     PC,TOCTNM     ;TYPE 1ST OCTAL NUMBER

```


6144	036600	004737	036764		JSR	PC,TOCTNM	:TYPE 2ND OCTAL NUMBER
6145	036604	000453			BR	21\$:BRANCH TO PRINT A TAB CHARACTER AFTER TABLE ENTRY
6146	036606	122710	000003	11\$:	CMPB	#3,(R0)	:IS THE FORMAT THREE?
6147	036612	001006			BNE	13\$:BRANCH TO CHECK FOR FORMAT 4 IF NOT
6148	036614	012703	000004		MOV	#4,R3	:SET LOOP COUNTER TO TYPE 4 OCTAL NUMBERS
6149	036620	004737	036764	12\$:	JSR	PC,TOCTNM	:TYPE AN OCTAL NUMBER
6150	036624	077303			SOB	R3,12\$:SUBTRACT 1 AND BRANCH IF NOT DONE
6151	036626	000442			BR	21\$:BRANCH TO PRINT A TAB CHARACTER AFTER TABLE ENTRY
6152							
6153	036630	122710	000004	13\$:	CMPB	#4,(R0)	:IS THE FORMAT FOUR?
6154	036634	001004			BNE	14\$:BRANCH TO CHECK FOR FORMAT 5 IF NOT
6155	036636	013146			MOV	@(R1)+,-(SP)	:MOVE THE DATA TO THE STACK FOR TYPING
6156	036640	104403			TYPOS		:GO TYPE AN OCTAL NUMBER
6157	036642	016			.BYTE	16	:TYPE UP TO 16 DIGITS
6158	036643	000			.BYTE	0	:SUPPRESSING LEADING ZEROES.
6159	036644	000433			BR	21\$:BRANCH TO PRINT A TAB CHARACTER AFTER TABLE ENTRY
6160	036646	122710	000005	14\$:	CMPB	#5,(R0)	:IS THE FORMAT FIVE?
6161	036652	001005			BNE	16\$:BRANCH TO CHECK FOR FORMAT 11 IF NOT
6162	036654	012137	036662		MOV	(R1)+,15\$:PUT THE ADDRESS OF THE ASCII STRING IN THE LOCATION
6163	036660	104401			TYPE		:TYPE THE ASCIZ STRING.
6164	036662	000000		15\$:	.WORD	0	:LOCATION FOR THE ADDRESS OF THE ASCIZ STRING
6165	036664	000425			BR	22\$:BRANCH TO INCREMENT R0 AND CONTINUE
6166	036666	122710	000011	16\$:	CMPB	#11,(R0)	:IS THE FORMAT ELEVEN?
6167	036672	001005			BNE	18\$:BRANCH TO CHECK FOR FORMAT 12 IF NOT
6168	036674	013137	036702		MOV	@(R1)+,17\$:MOVE THE DATA TO THE STACK FOR TYPING
6169	036700	104401			TYPE		:TYPE THE ASCIZ STRING.
6170	036702	000000		17\$:	.WORD	0	:LOCATION FOR THE ADDRESS OF THE ASCIZ STRING
6171	036704	000415			BR	22\$:BRANCH TO INCREMENT R0 AND CONTINUE
6172	036706	122710	000012	18\$:	CMPB	#12,(R0)	:IS THE FORMAT TWELVE?
6173	036712	001007			BNE	20\$:BRANCH TO HALT IF NOT - FORMAT NOT RECOGNIZED
6174	036714	013102			MOV	@(R1)+,R2	:MOVE THE DATA TO THE STACK FOR TYPING
6175	036716	012703	000006		MOV	#6,R3	:TYPE SIX OCTAL NUMBERS
6176	036722	004737	036764	19\$:	JSR	PC,TOCTNM	:TYPE AN OCTAL NUMBER
6177	036726	077303			SOB	R3,19\$:SUBTRACT 1 AND BRANCH IF NOT DONE YET
6178	036730	000401			BR	21\$:BRANCH TO PRINT A TAB CHARACTER AFTER TABLE ENTRY
6179	036732	000000		20\$:	HALT		:UNDEFINED FORMAT FOR DATA?????
6180	036734	104401	037335	21\$:	TYPE	,\$TAB	:PRINT A TAB AFTER TYPING A DATA TABLE ENTRY
6181							:OF ALL FORMATS EXCEPT FORMATS 5 OR 11
6182	036740	005200		22\$:	INC	R0	:POINT TO THE NEXT FORMAT
6183	036742	005711			TST	(R1)	:HAS THE END OF THE DATA TABLE BEEN REACHED?
6184	036744	001303			BNE	9\$:BRANCH BACK IF NOT
6185	036746	104401	001313		TYPE	,\$CRLF	:DONE.
6186	036752	012603			MOV	(SP)+,R3	:RESTORE R1,R2 AND R3
6187	036754	012602			MOV	(SP)+,R2	
6188	036756	012601			MOV	(SP)+,R1	
6189	036760	012600		23\$:	MOV	(SP)+,R0	:RESTORE R0.
6190	036762	000207			RTS	PC	:AND RETURN.
6191							
6192	036764	013102		TOCTNM:	MOV	@(R1)+,R2	:FORMAT TWO SO TYPE TWO
6193	036766	012246			MOV	(R2)+,-(SP)	:OCTAL NUMBERS.
6194	036770	104402			TYPOC		:TYPE THE OCTAL NUMBER
6195	036772	104401	037337		TYPE	,\$SPACE	:TYPE A SPACE CHARACTER
6196	036776	000207			RTS	PC	:EXIT

6197
6198

6199
6200
6201
6202
6203 037000 011637 001236
6204 037004 022626
6205 037006 170200
6206 037010 010037 001240
6207 037014 170300
6208 037016 010037 001242
6209 037022 104247
6210 037024 104412

```
.SBTTL FPP SPURIOUS TRAP TO 244 HANDLER
:*****
:*****
:THIS ROUTINE HANDLES UNEXPECTED TRAPS TO THE FPP TRAP VECTOR AT 244.
:THE LAST FPP INSTRUCTION EXECUTED AND ITS ADDRESS HAS BEEN RECORDED
:THESE ALONG WITH THE FEC, FPS AND PC OF TRAP ARE REPORTED.
:*
FPPSPUR: MOV      (SP), $TMP2          ;SAVE PC OF TRAP.
          CMP      (SP)+, (SP)+      ;RESTORE SP.
          STFPS    R0                ;GET FPS
          MOV      R0, $TMP3         ;GET FEC
          STST     R0
          MOV      R0, $TMP4
1$:      ERROR    +247
          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
```

6211 037026 000137 033154
6212
6213
6214
6215

```
.SBTTL CPU SPURIOUS TRAP TO 4 HANDLER
:*****
:*****
:THIS ROUTINE REPORTS UNEXPECTED CPU TRAPS TO VECTOR 4.
:*
CPSPUR:  MOV      (SP), $TMP2          ;SAVE PC OF TRAP.
          CMP      (SP)+, (SP)+
1$:      ERROR    +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?).

          JMP      $EOP
```

6222 037044 000137 033154
6223
6224
6225
6226

```
.SBTTL CPU SPURIOUS TRAP TO 10 HANDLER
:*****
:*****
:THIS ROUTINE REPORTS UNEXPECTED CPU TRAPS TO VECTOR 10.
:*
CPTWO:   MOV      (SP), $TMP2          ;SAVE PC OF TRAP.
          CMP      (SP)+, (SP)+
1$:      ERROR    +251
          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
```

6233 037062 000137 033154
6234
6235
6236
6237
6238

JMP \$EOP

6239
6240

.SBTTL SET LOOP ON ERROR ADDRESS ROUTINE

```
*****  
*****  
*  
.LPER: MOV (SP), $LPERR  
RTI
```

6241
6242 037066 011637 001110
6243 037072 000002
6244
6245
6246

.SBTTL FLAG RESET AND CONSOLE TEST ROUTINE

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

6247
6248
6249
6250
6251
6252
6253
6254 037074 023727 001140 177570
6255
6256 037102 001001
6257 037104 104406
6258
6259
6260
6261 037106 012737 037000 000244 1\$:
6262 037114 012737 037032 000004
6263 037122 012737 037050 000010
6264 037130 011600
6265 037132 012706 001100
6266 037136 005004
6267 037140 170104
6268 037142 000110
6269
6270
6271

```
.RSET: CMP SWR, #177570 ;SEE IF THERE IS A PHYSICAL  
;CONSOLE SWITCH REGISTER.  
BNE 1$ ;BRANCH IF NO.  
CKSWR ;OTHERWISE TYPE THE CONTENTS  
;OF THE PROGRAM VIRTUAL SWITCH REGISTER  
;AND GIVE THE USER A CHANCE TO  
;MODIFY IT.  
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.
```

6272
6273 037144 124 122 101
6274 037162 105 130 120
6275 037205 107 117 124
6276 037217 105 130 120
6277 037235 107 117 124
6278 037250 105 130 120
6279 037270 200 120 117
6280 037335 011 000
6281 037337 040 040 000
6282 037342 101 103 040
6283 037360 106 123 122
6284 037400 101 103 060
6285 037427 101 103 060
6286 037455 105 130 120
6287 037477 107 117 124
6288 037515 106 122 101
6289 037541 111 116 124
6290 037563 105 130 120
6291 037607 105 130 120
6292 037632 114 117 101
6293 037650 122 105 101

```
.SBTTL SPECIAL MESSAGES  
.ASCIZ 'TRAPPED AT:'<TAB><TAB>  
.ASCIZ 'EXPECTED TRAP AT:'<TAB>  
.ASCIZ 'GOT R0:'<TAB><TAB>  
.ASCIZ 'EXPECTED R0:'<TAB>  
.ASCIZ 'GOT ACO:'<TAB><TAB>  
.ASCIZ 'EXPECTED ACO:'<TAB><TAB>  
POWERM: .ASCIZ <CRLF>'POWER FAILURE. PROGRAM RESTARTING.'<CRLF>  
$TAB: .ASCIZ <TAB>  
SPACE: .ASCIZ ' '  
MS1: .ASCIZ 'AC OPERAND:'<TAB><TAB>  
MS2: .ASCIZ 'FSRC OPERAND:'<TAB><TAB>  
MS3: .ASCIZ 'ACO BEFORE EXECUTION:'<TAB>  
MS4: .ASCIZ 'ACO AFTER EXECUTION:'<TAB>  
MS5: .ASCIZ 'EXPECTED RESULT:'<TAB>  
MS6: .ASCIZ 'GOT RESULT:'<TAB><TAB>  
MS7: .ASCIZ 'FRACTIONAL RESULT:'<TAB>  
MS10: .ASCIZ 'INTEGER RESULT:'<TAB>  
MS11: .ASCIZ 'EXPECTED FRACTION:'<TAB>  
MS12: .ASCIZ 'EXPECTED INTEGER:'<TAB>  
MS37: .ASCIZ 'LOADED DATA: '  
MS40: .ASCIZ 'READ DATA: '
```

6294	037664	105	130	120	MS415:	.ASCIZ	'EXPECTED DATA: '
6295	037704	104	101	124	MS41:	.ASCIZ	'DATA IN (RO) FSRC: '
6296	037730	104	101	124	MS42:	.ASCIZ	'DATA IN ACO: '
6297	037746	107	117	124	MS43:	.ASCIZ	'GOT RESULT: '
6298	037763	105	130	120	MS44:	.ASCIZ	'EXPECTED RESULT: '
6299	040005	200	124	105	CORMES:	.ASCIZ	<CRLF>'TEST 22, TESTING INTERRUPTS.'<CRLF>

Line No.	Code	Col 1	Col 2	Col 3	Col 4	Label	Message
6300						.SBTTL	ERROR MESSAGES
6301	040044	106	120	123	EM1:	.ASCIZ	'FPS BAD AFTER CMPD (R),A.'
6302	040076	101	103	060	EM2:	.ASCIZ	'ACO MODIFIED BY CMPD (R),A.'
6306	040132				EM3:		
	040132	106	120	123		.ASCII	'FPS BAD AFTER CMPD.'<CRLF>
6307	040156	050	102	125		.ASCIZ	'(BUT ENBT) STATE 225 WENT TO 475 INSTEAD OF 075.'
6308	040237				EM4:		
	040237	106	120	123		.ASCII	'FPS BAD AFTER CMPD.'<CRLF>
6309	040263	050	102	125		.ASCIZ	'(BUT ENBT) STATE 225 WENT TO 075 INSTEAD OF 475.'
6310	040344				EM5:		
	040344	106	120	123		.ASCII	'FPS BAD AFTER CMPD.'<CRLF>
6311	040370	050	102	125		.ASCIZ	'(BUT ENBT) STATE 035 WENT TO 075 INSTEAD OF 475.'
6312	040451				EM6:		
	040451	106	120	123		.ASCII	'FPS BAD AFTER CMPD.'<CRLF>
6313	040475	050	102	125		.ASCIZ	'(BUT ENBT) STATE 035 WENT TO 475 INSTEAD OF 075.'
6314	040556				EM7:		
	040556	106	120	123		.ASCII	'FPS BAD AFTER CMPD.'<CRLF>
6315	040602	050	102	125		.ASCIZ	'(BUT ENBT Y8) STATE 777 SHOULD HAVE GONE TO 007.'
6316	040663				EM10:		
	040663	106	120	123		.ASCII	'FPS BAD AFTER CMPD.'<CRLF>
6317	040707	050	102	125		.ASCIZ	'(BUT ENBT Y8) STATE 777 SHOULD HAVE GONE TO 405.'
6318	040770				EM11:		
	040770	106	120	123		.ASCII	'FPS BAD AFTER CMPD.'<CRLF>
6319	041014	050	102	125		.ASCIZ	'(BUT NBIT ZBIT) STATE 456 SHOULD HAVE GONE TO 010.'
6320	041077				EM12:		
	041077	106	120	123		.ASCII	'FPS BAD AFTER CMPD.'<CRLF>
6321	041123	050	102	125		.ASCIZ	'(BUT NBIT ZBIT) STATE 456 SHOULD HAVE GONE TO 110.'
6322	041206				EM13:		
	041206	106	120	123		.ASCII	'FPS BAD AFTER CMPD.'<CRLF>
6323	041232	104	111	104		.ASCIZ	/DIDN'T TAKE THE PATH: STATE 456, TO 012, TO 363 TO 120./
6324	041322				EM14:		
	041322	106	120	123		.ASCII	'FPS BAD AFTER CMPD.'<CRLF>
6325	041346	050	102	125		.ASCIZ	'(BUT XNBT XZBT) STATE 363 WENT TO 140 INSTEAD OF 100.'
6326	041434				EM15:		
	041434	106	120	123		.ASCII	'FPS BAD AFTER CMPD.'<CRLF>
6327	041460	050	102	125		.ASCIZ	'(BUT XNBT XZBT) STATE 363 WENT TO 100 INSTEAD OF 140.'
6328	041546				EM16:		
	041546	106	120	123		.ASCII	'FPS BAD AFTER CMPD.'<CRLF>
6329	041572	104	111	104		.ASCIZ	/DIDN'T TAKE THE PATH: STATE 777, TO 407./
6330	041643	104	111	126	EM17:	.ASCIZ	'DIVD (R),A TRAPPED TO 244. FSRC=0 AND FID=1.'
6331	041720	106	120	123	EM20:	.ASCIZ	'FPS BAD AFTER DIVD (R),A.'
6332	041752	106	105	103	EM21:	.ASCIZ	'FEC BAD AFTER DIVD (R),A.'
6333	042004	104	111	126	EM22:	.ASCIZ	/DIVD (R),A DIDN'T TRAP TO 244. FSRC=0 AND FID=0./
6334	042065	104	111	126	EM23:	.ASCIZ	'DIVF (R),A FAILED.'
6335	042110	106	120	123	EM32:	.ASCIZ	'FPS BAD AFTER DIVF (R),A.'
6339	042142				EM24:		
	042142	104	111	126		.ASCII	'DIVF (R),A FAILED.'
6340	042164	050	102	125		.ASCIZ	'(BUT Y61) WENT TO STATE 006 INSTEAD OF 206.'
6341	042240				EM25:		
	042240	104	111	126		.ASCII	'DIVF (R),A FAILED.'
6342	042262	130	117	122		.ASCIZ	'XOR OF SIGN BITS FAILED STATE 470.'
6343	042325				EM26:		
	042325	104	111	126		.ASCII	'DIVF (R),A FAILED.'
6344	042347	050	102	125		.ASCIZ	'(BUT Y61) WENT TO STATE 206 INSTEAD OF 006.'
6345		042240			EM27=EM25		
6346	042423				EM30:		
	042423	104	111	126		.ASCII	'DIVF (R),A FAILED.'

6347	042445	124	122	125		.ASCIZ	'TRUNCATION ERROR. FT=1.'
6348	042475				EM31:		
	042475	104	111	126		.ASCII	'DIVF (R),A FAILED.'
6349	042517	122	117	125		.ASCIZ	'ROUND ERROR. FT=0.'
6350	042542	104	111	126	EM33:	.ASCIZ	'DIVD (R),A FAILED.'
6351	042565	106	120	123	EM34:	.ASCIZ	'FPS BAD AFTER DIVD (R),A.'
6355	042617				EM35:		
	042617	104	111	126		.ASCII	'DIVD (R),A FAILED.<CRLF>
6356	042642	124	122	125		.ASCIZ	'TRUNCATION ERROR. FT=1.'
6357	042672				EM36:		
	042672	104	111	126		.ASCII	'DIVD (R),A FAILED.<CRLF>
6358	042715	122	117	125		.ASCIZ	'ROUND ERROR. FT=0.'
6359	042740	115	125	114	EM37:	.ASCIZ	'MULF (R),A FAILED.'
6363	042763	106	120	123	EM40:	.ASCIZ	'FPS BAD AFTER MULF (R),A.'
6364	043015				EM41:		
	043015	115	125	114		.ASCII	'MULF (R),A FAILED.<CRLF>
6365	043040	123	111	107		.ASCIZ	'SIGN BIT BAD STATE 511.'
6366	043070				EM42:		
	043070	115	125	114		.ASCII	'MULF (R),A FAILED.<CRLF>
6367	043113	116	117	122		.ASCII	'NORMALIZATION FAILED.<CRLF>
6368	043141	050	102	125		.ASCIZ	'(BUT Y62) STATE 252 WENT TO 044 INSTEAD OF 444.'
6369	043221				EM43:		
	043221	115	125	114		.ASCII	'MULF (R),A FAILED.<CRLF>
6370	043244	116	117	122		.ASCII	'NORMALIZATION FAILED.<CRLF>
6371	043272	050	102	125		.ASCIZ	'(BUT Y62) STATE 252 WENT TO 444 INSTEAD OF 044.'
6372	043352				EM44:		
	043352	115	125	114		.ASCII	'MULF (R),A FAILED.<CRLF>
6373	043375	122	117	125		.ASCIZ	'ROUND ERROR. FT=0.'
6374	043420				EM45:		
	043420	115	125	114		.ASCII	'MULF (R),A FAILED.<CRLF>
6375	043443	124	122	125		.ASCIZ	'TRUNCATION ERROR. FT=1.'
6376	043473	106	120	123	EM46:	.ASCIZ	'FPS BAD AFTER MULD (R),A.'
6380	043525	115	125	114	EM246:	.ASCIZ	'MULD (R),A FAILED.'
6381	043550				EM47:		
	043550	115	125	114		.ASCII	'MULD (R),A FAILED.<CRLF>
6382	043573	102	101	104		.ASCII	'BAD CONSTANT USED IN THE MUL ALGORITHM.'
6383	043642	200	125	123		.ASCIZ	<CRLF>'USED 24 INSTEAD OF 56 STATE 020.'
6384	043704				EM50:		
	043704	115	125	114		.ASCII	'MULD (R),A FAILED.<CRLF>
6385	043727	124	122	125		.ASCIZ	'TRUNCATION ERROR. FT=1.'
6386	043757				EM51:		
	043757	115	125	114		.ASCII	'MULD (R),A FAILED.<CRLF>
6387	044002	122	117	125		.ASCIZ	'ROUND ERROR. FT=0.'
6388	044025				EM52:		
	044025	115	125	114		.ASCII	'MULD (R),A FAILED.<CRLF>
6389	044050	102	101	104		.ASCIZ	'BAD CONSTANT USED IN ROUNDING. FT=0.'
6390	044115	106	120	123	EM111:	.ASCIZ	'FPS BAD AFTER MULF (R),A. EXPECTED OVERFLOW.'
6391	044172	106	120	123	EM112:	.ASCIZ	'FPS BAD AFTER MULF (R),A. EXPECTED UNDERFLOW.'
6392	044250				EM113:		
	044250	115	125	114		.ASCII	'MULF (R),A FAILED.<CRLF>
6393	044273	105	130	120		.ASCIZ	'EXPECTING OVERFLOW, FIV=0.'
6394	044326				EM114:		
	044326	115	125	114		.ASCII	'MULF (R),A FAILED.<CRLF>
6395	044351	105	130	120		.ASCIZ	'EXPECTING UNDERFLOW, FIU=0.'
6396	044405	115	125	114	EM115:	.ASCIZ	'MULF (R),A TRAPPED TO 244 ON OVERFLOW. FIV=0.'
6397	044463	115	125	114	EM116:	.ASCIZ	'MULF (R),A TRAPPED TO 244 ON UNDERFLOW. FIU=0.'
6404	044542				EM117:		

6405	044542	115	125	114	.ASCII	'MULF (R),A TRAPPED TO 244 ON UNDERFLOW. FIU=0.'	
6406	044620	050	102	125	.ASCIZ	'(BUT FIU) STATE 331 WENT TO 155 INSTEAD OF 115.'	
6407	044700	115	125	114	EM120:	.ASCII	'MULF (R),A TRAPPED TO 244 ON UNDERFLOW. FIU=0.'
6408	044756	050	102	125	.ASCIZ	'(BUT FIU) STATE 137 WENT TO 155 INSTEAD OF 115.'	
6409	045036	115	125	114	EM121:	.ASCII	'MULF (R),A TRAPPED TO 244 ON OVERFLOW. FIV=0.'
6410	045113	050	102	125	.ASCIZ	'(BUT FIV) STATE 333 WENT TO 136 INSTEAD OF 116.'	
6411	045173	115	125	114	EM122:	.ASCII	'MULF (R),A TRAPPED TO 244 ON OVERFLOW. FIV=0.'
6412	045250	050	102	125	.ASCIZ	'(BUT FIV) STATE 133 WENT TO 136 INSTEAD OF 116.'	
6413	045330	106	120	123	EM123:	.ASCIZ	'FPS BAD AFTER MULF (R),A. EXPECTING OVERFLOW.'
6414	045406	106	120	123	EM124:	.ASCIZ	'FPS BAD AFTER MULF (R),A. EXPECTING UNDERFLOW.'
6415	045465	115	125	114	EM125:	.ASCII	'MULD (R),A FAILED.<CRLF>
6416	045510	105	130	120	.ASCIZ	'EXPECTING OVERFLOW, FIV=0.'	
6417	045543	115	125	114	EM126:	.ASCII	'MULD (R),A FAILED.<CRLF>
6418	045622	105	130	120	.ASCIZ	'EXPECTING UNDERFLOW, FIU=0.'	
6419	045700	115	125	114	EM127:	.ASCIZ	'MULD (R),A TRAPPED TO 244 ON OVERFLOW. FIV=0.'
6426	045757	115	125	114	EM130:	.ASCIZ	'MULD (R),A TRAPPED TO 244 ON UNDERFLOW. FIU=0.'
6427	046035	050	102	125	EM131:	.ASCII	'MULF (R),A TRAPPED TO 244 ON UNDERFLOW. FIU=0.'
6428	046115	115	125	114	.ASCIZ	'(BUT FIU) STATE 331 WENT TO 155 INSTEAD OF 115.'	
6429	046140	105	130	120	EM132:	.ASCII	'MULD (R),A FAILED.<CRLF>
6430	046173	200	050	102	.ASCIZ	'EXPECTING UNDERFLOW, FIU=0.'	
6431	046253	115	125	114	.ASCIZ	<CRLF>'(BUT FD) STATE 115 WENT TO 424 INSTEAD OF 425.'	
6432	046331	050	102	125	EM133:	.ASCII	'MULF (R),A TRAPPED TO 244 ON UNDERFLOW. FIU=0.'
6433	046411	115	125	114	.ASCIZ	'(BUT FIU) STATE 137 WENT TO 155 INSTEAD OF 115.'	
6434	046466	050	102	125	EM134:	.ASCII	'MULD (R),A TRAPPED TO 244 ON OVERFLOW. FIV=0.'
6435	046546	115	125	114	.ASCIZ	'(BUT FIV) STATE 333 WENT TO 136 INSTEAD OF 116.'	
6436	046571	105	130	120	EM135:	.ASCII	'MULD (R),A FAILED.<CRLF>
6437	046623	200	050	102	.ASCIZ	'EXPECTING OVERFLOW, FIV=0.'	
6438	046703	115	125	114	.ASCIZ	<CRLF>'(BUT FD) STATE 116 WENT TO 424 INSTEAD OF 425.'	
6439	046760	050	102	125	EM136:	.ASCII	'MULD (R),A TRAPPED TO 244 ON OVERFLOW. FIV=0.'
6440	047040	106	105	103	.ASCIZ	'(BUT FIV) STATE 133 WENT TO 136 INSTEAD OF 116.'	
6441	047126	106	105	103	EM137:	.ASCIZ	'FEC BAD AFTER MULF (R),A. EXPECTING OVERFLOW, FEC=10.'
6442	044115				EM140:	.ASCIZ	'FEC BAD AFTER MULF (R),A. EXPECTING UNDERFLOW, FEC=12.'
6443	044172				EM141=EM111		
6444	047215				EM142=EM112		
6445	047240	115	125	114	EM143:	.ASCII	'MULF (R),A FAILED.<CRLF>
6446	047273	105	130	120	.ASCIZ	'EXPECTING OVERFLOW, FIV=1.'	
6447	047316	115	125	114	EM144:	.ASCII	'MULF (R),A FAILED.<CRLF>
6454	047352	105	130	120	.ASCIZ	'EXPECTING UNDERFLOW, FIU=1.'	
6455	047437	115	125	114	EM145:	.ASCII	'MULF (R),A FAILED TO TRAP TO 244 ON UNDERFLOW. FIU=1.'
6456	047517	050	102	125	.ASCIZ	'(BUT FIU) STATE 331 WENT TO 115 INSTEAD OF 155.'	
6457	047604	115	125	114	EM146:	.ASCII	'MULF (R),A FAILED TO TRAP TO 244 ON UNDERFLOW. FIU=1.'
		050	102	125	.ASCIZ	'(BUT FIU) STATE 137 WENT TO 115 INSTEAD OF 155.'	

6458	047664				EM147:	.ASCII	'MULF (R),A FAILED TO TRAP TO 244 ON OVERFLOW. FIV=1.'
	047664	115	125	114		.ASCII	'(BUT FIV) STATE 333 WENT TO 116 INSTEAD OF 136.'
6459	047750	050	102	125		.ASCII	
6460	050030				EM150:	.ASCII	'MULF (R),A FAILED TO TRAP TO 244 ON OVERFLOW. FIV=1.'
	050030	115	125	114		.ASCII	'(BUT FIV) STATE 133 WENT TO 116 INSTEAD OF 136.'
6461	050114	050	102	125		.ASCII	
6462	050174	106	105	103	EM151:	.ASCII	'FEC BAD AFTER MULF (R),A. EXPECTING OVERFLOW, FEC=10.'
6463	050262	106	105	103	EM152:	.ASCII	'FEC BAD AFTER MULF (R),A. EXPECTING UNDERFLOW, FEC=12.'
6464		045330			EM153=EM123		
6465		045406			EM154=EM124		
6466	050351				EM155:	.ASCII	'MULF (R),A FAILED.'<CRLF>
	050351	115	125	114		.ASCII	'EXPECTING OVERFLOW, FIV=1.'
6467	050374	105	130	120		.ASCII	
6468	050427				EM156:	.ASCII	'MULF (R),A FAILED.'<CRLF>
	050427	115	125	114		.ASCII	'EXPECTING UNDERFLOW, FIU=1.'
6469	050452	105	130	120		.ASCII	
6476	050506				EM157:	.ASCII	'MULF (R),A FAILED TO TRAP TO 244 ON UNDERFLOW. FIU=1.'
	050506	115	125	114		.ASCII	'(BUT FIU) STATE 331 WENT TO 115 INSTEAD OF 155.'
6477	050573	050	102	125		.ASCII	
6478	050653				EM160:	.ASCII	'MULF (R),A FAILED.'<CRLF>
	050653	115	125	114		.ASCII	'EXPECTING UNDERFLOW, FIU=1.'
6479	050676	105	130	120		.ASCII	<CRLF>'(BUT FD) STATE 155 WENT TO 426 INSTEAD OF 427.'
6480	050731	200	050	102		.ASCII	
6481	051011				EM161:	.ASCII	'MULF (R),A FAILED TO TRAP TO 244 ON UNDERFLOW. FIU=1.'
	051011	115	125	114		.ASCII	'(BUT FIU) STATE 137 WENT TO 115 INSTEAD OF 155.'
6482	051076	050	102	125		.ASCII	
6483	051156				EM162:	.ASCII	'MULF (R),A FAILED TO TRAP TO 244 ON OVERFLOW. FIV=1.'
	051156	115	125	114		.ASCII	'(BUT FIV) STATE 333 WENT TO 116 INSTEAD OF 136.'
6484	051242	050	102	125		.ASCII	
6485	051322				EM163:	.ASCII	'MULF (R),A FAILED.'<CRLF>
	051322	115	125	114		.ASCII	'EXPECTING OVERFLOW, FIV=1.'
6486	051345	105	130	120		.ASCII	<CRLF>'(BUT FD) STATE 700 WENT TO 426 INSTEAD OF 427.'
6487	051377	200	050	102		.ASCII	
6488	051457				EM164:	.ASCII	'MULF (R),A FAILED TO TRAP TO 244 ON OVERFLOW. FIV=1.'
	051457	115	125	114		.ASCII	'(BUT FIV) STATE 133 WENT TO 116 INSTEAD OF 136.'
6489	051543	050	102	125		.ASCII	
6490	051623	106	120	123	EM55:	.ASCII	'FPS BAD AFTER MODF (R),A.'
6491	051655	115	117	104	EM53:	.ASCII	'MODF (R),A FRACTION BAD.'
6492	051706	115	117	104	EM54:	.ASCII	'MODF (R),A INTEGER BAD.'
6499	051736				EM56:	.ASCII	'MODF (R),A FRACTION BAD.'<CRLF>
	051736	115	117	104		.ASCII	'ACO DID NOT GET 0 IN STATE 424.'
6500	051767	101	103	060		.ASCII	
6501	052027				EM57:	.ASCII	'MODF (R),A INTEGER BAD.'<CRLF>
	052027	115	117	104		.ASCII	'AC1 DID NOT GET 0 IN STATE 142.'
6502	052057	101	103	061		.ASCII	
6503	052117				EM60:	.ASCII	'MODF (R),A INTEGER BAD.'<CRLF>
	052117	115	117	104		.ASCII	'AC1 DID NOT GET THE INTEGER IN STATE 134.'
6504	052147	101	103	061		.ASCII	
6505	052221				EM61:	.ASCII	'MODF (R),A FRACTION BAD.'<CRLF>
	052221	115	117	104		.ASCII	'A BAD CONSTANT WAS USED (NOT 24) IN STATE 046.'
6506	052252	101	040	102		.ASCII	<CRLF>'OR (BUT NBIT) STATE 525 WENT TO 050 INSTEAD OF 150.'
6507	052330	200	117	122		.ASCII	
6508	052415				EM62:	.ASCII	'MODF (R),A FRACTION BAD.'<CRLF>
	052415	115	117	104		.ASCII	'A BAD CONSTANT WAS USED (NOT 24) IN STATE 046.'
6509	052446	101	040	102		.ASCII	<CRLF>'OR (BUT NBIT) STATE 525 WENT TO 150 INSTEAD OF 050.'
6510	052524	200	117	122		.ASCII	
6511	052611				EM63:		

6512	052611	115	117	104		.ASCII	'MODF (R),A FRACTION BAD.'<CRLF>
6513	052642	050	102	125		.ASCIZ	'(BUT ZBT) STATE 532 WENT TO 102 INSTEAD OF 122.'
6514	052722	115	117	104	EM64:	.ASCII	'MODF (R),A FRACTION BAD.'<CRLF>
6515	052722	050	102	125		.ASCIZ	'(BUT ENBT EZBT) STATE 041 WENT TO 046 INSTEAD OF 246.'
6516	053041	115	117	104	EM65:	.ASCII	'MODF (R),A FRACTION BAD.'<CRLF>
6517	053072	050	102	125		.ASCIZ	'(BUT FT) STATE 126 SHOULD HAVE GONE TO 133. FT=0.'
6518	053154	115	117	104	EM66:	.ASCII	'MODF (R),A FRACTION BAD.'<CRLF>
6519	053205	123	111	107		.ASCIZ	'SIGN BIT BAD.'
6520	053223	115	117	104	EM67:	.ASCII	'MODF (R),A INTEGER BAD.'<CRLF>
6521	053253	123	111	107		.ASCIZ	'SIGN BIT BAD IN STATE 733.'
6522	053306	115	117	104	EM70:	.ASCIZ	'MODD (R),A FRACTION BAD.'
6523	053337	115	117	104	EM71:	.ASCIZ	'MODD (R),A INTEGER BAD.'
6524	053367	106	120	123	EM72:	.ASCIZ	'FPS BAD AFTER MODD (R),A.'
6531	053421	053367			EM73=EM72		
6532	053421	115	117	104	EM74:	.ASCII	'MODD (R),A INTEGER BAD.'<CRLF>
6533	053451	050	102	125		.ASCIZ	'(BUT FD) STATE 231 WENT TO 142 INSTEAD OF 143.'
6534	053530	115	117	104	EM75:	.ASCII	'MODD (R),A FRACTION BAD.'<CRLF>
6535	053561	101	103	060		.ASCIZ	'AC0 GETS 0 IN STATE 425 FAILED.'
6536	053621	115	117	104	EM76:	.ASCII	'MODD (R),A INTEGER BAD.'<CRLF>
6537	053651	101	103	061		.ASCIZ	'AC1 GETS 0 IN STATE 143 FAILED.'
6538	053711	115	117	104	EM77:	.ASCII	'MODD (R),A FRACTION BAD.'<CRLF>
6539	053742	050	102	125		.ASCIZ	'(BUT FD) STATE 526 WENT TO 134 INSTEAD OF 135.'
6540	054021	115	117	104	EM100:	.ASCII	'MODD (R),A INTEGER BAD.'<CRLF>
6541	054051	123	111	107		.ASCIZ	'SIGN BIT BAD IN STATE 526.'
6542	054104	115	117	104	EM101:	.ASCII	'MODD (R),A FRACTION BAD.'<CRLF>
6543	054135	101	040	102		.ASCII	'A BAD CONSTANT WAS USED (NOT 56) IN STATE 046.'
6544	054213	200	117	122		.ASCIZ	<CRLF>'OR (BUT NBIT) STATE 525 WENT TO 050 INSTEAD OF 150.'
6545	054300	115	117	104	EM102:	.ASCII	'MODD (R),A FRACTION BAD.'<CRLF>
6546	054331	101	040	102		.ASCII	'A BAD CONSTANT WAS USED (NOT 56) IN STATE 046.'
6547	054407	200	117	122		.ASCIZ	<CRLF>'OR (BUT NBIT) STATE 525 WENT TO 150 INSTEAD OF 050.'
6548	054474	115	117	104	EM103:	.ASCII	'MODD (R),A FRACTION BAD.'<CRLF>
6549	054525	050	102	125		.ASCIZ	'(BUT ZBIT) STATE 532 WENT TO 122 INSTEAD OF 102.'
6550	054606	115	117	104	EM104:	.ASCII	'MODD (R),A INTEGER BAD.'<CRLF>
6551	054636	123	105	124		.ASCII	'SET INTEGER IN AC1 FAILED.'
6552	054670	200	117	122		.ASCIZ	<CRLF>'OR (BUT FD) STATE 733 WENT TO 156 INSTEAD OF 157.'
6553	054753	115	117	104	EM105:	.ASCII	'MODD (R),A INTEGER BAD.'<CRLF>
6554	055003	050	102	125		.ASCIZ	'(BUT FD) STATE 122 WENT TO 424 INSTEAD OF 425.'
6555	055062	115	117	104	EM106:	.ASCII	'MODD (R),A INTEGER BAD.'<CRLF>
6556	055112	050	102	125		.ASCIZ	'(BUT FD) STATE 246 WENT TO 126 INSTEAD OF 127.'
6557	055171	115	117	104	EM107:	.ASCII	'MODD (R),A INTEGER BAD.'<CRLF>
	055221	050	102	125		.ASCIZ	'(BUT FD) STATE 446 WENT TO 126 INSTEAD OF 127.'

6558	055300				EM110:	.ASCII	'MODD (R),A FRACTION BAD.<CRLF>
	055300	115	117	104		.ASCIIZ	'(BUT FT) STATE 127 WENT TO 313 INSTEAD OF 113.'
6559	055331	050	102	125			
6572	055410				EM165:	.ASCII	/MODF (R),A FRACTION BAD. RESULT OVER OR UNDER FLOW./
	055410	115	117	104		.BYTE	0
6573	055473	000					
6574	055474				EM166:	.ASCII	/MODF (R),A INTEGER BAD. RESULT OVER OR UNDER FLOW./
	055474	115	117	104		.BYTE	0
6575	055556	000					
6576	055557				EM167:	.ASCII	/FPS BAD AFTER MODF (R),A./
	055557	106	120	123		.BYTE	0
6577	055610	000					
6578	055611				EM170:	.ASCII	/FEC BAD AFTER MODF (R),A./
	055611	106	105	103		.ASCIIZ	'EXPECTING UNDERFLOW, FEC=12.'
6579	055642	105	130	120			
6580	055677				EM171:	.ASCII	/MODF (R),A INTEGER BAD. RESULT OVER OR UNDER FLOW./
	055677	115	117	104		.ASCIIZ	<CRLF>'ACI GETS 0 IN STATE 126 FAILED.'
6581	055761	200	101	103			
6582	056022				EM172:	.ASCII	/FEC BAD AFTER MODF (R),A./
	056022	106	105	103		.ASCIIZ	'EXPECTING OVERFLOW, FEC=10.'
6583	056053	105	130	120			
6584	056107				EM173:	.ASCII	/MODF (R),A INTEGER BAD. RESULT OVER OR UNDER FLOW./
	056107	115	117	104		.ASCIIZ	<CRLF>'(BUT FIV FD) STATE 520 WENT TO 142 INSTEAD OF 162.'
6585	056171	200	050	102			
6586	056255				EM174:	.ASCII	/MODF (R),A INTEGER BAD. RESULT OVER OR UNDER FLOW./
	056255	115	117	104		.ASCIIZ	<CRLF>'(BUT FIV FD) STATE 520 WENT TO 162 INSTEAD OF 142.'
6587	056337	200	050	102			
6588	056423				EM175:	.ASCII	/MODF (R),A INTEGER BAD. RESULT OVER OR UNDER FLOW./
	056423	115	117	104		.ASCIIZ	<CRLF>'SIGN BAD IN STATE 517.'
6589	056505	200	123	111			
6590	056535				EM176:	.ASCII	/MODD (R),A FRACTION BAD. RESULT OVER OR UNDER FLOW./
	056535	115	117	104		.BYTE	0
6591	056620	000					
6592	056621	120	117	127	EM177:	.ASCIIZ	'POWER MONITOR BIT FOUND SET'
6593	056655				EM200:	.ASCII	/FPS BAD AFTER MODD (R),A./
	056655	106	120	123		.BYTE	0
6594	056706	000					
6595	056707				EM201:	.ASCII	/FEC BAD AFTER MODD (R),A./
	056707	106	105	103		.ASCIIZ	<CRLF>'EXPECTING UNDERFLOW, FEC=12.'
6596	056740	200	105	130			
6597	056776				EM202:	.ASCII	/MODD (R),A INTEGER BAD. RESULT OVER OR UNDER FLOW./
	056776	115	117	104		.ASCIIZ	<CRLF>'(BUT FD) STATE 241 WENT TO 126 INSTEAD OF 127.'
6598	057060	200	050	102			
6599	057140				EM203:	.ASCII	/MODD (R),A INTEGER BAD. RESULT OVER OR UNDER FLOW./
	057140	115	117	104		.ASCIIZ	<CRLF>'(BUT FD) STATE 047 WENT TO 126 INSTEAD OF 127.'
6600	057222	200	050	102			
6601	057302				EM204:	.ASCII	/FEC BAD AFTER MODD (R),A./
	057302	106	105	103		.ASCIIZ	<CRLF>'EXPECTING OVERFLOW, FEC=10.'
6602	057333	200	105	130			
6603	057370				EM205:	.ASCII	/MODD (R),A INTEGER BAD. RESULT OVER OR UNDER FLOW./
	057370	115	117	104		.ASCIIZ	<CRLF>'(BUT FIV FD) STATE 520 WENT TO 162 INSTEAD OF 163.'
6604	057452	200	050	102			
6605	057536				EM206:	.ASCII	/MODD (R),A INTEGER BAD. RESULT OVER OR UNDER FLOW./
	057536	115	117	104		.ASCIIZ	<CRLF>'(BUT FIV FD) STATE 520 WENT TO 162 INSTEAD OF 143.'
6606	057620	200	050	102			
6607							
6608							

;

6701	057704				EM207:	
	057704	101	104	104	.ASCIZ	/ADDD (R),A PRODUCED A BAD RESULT./
6702	057746				EM210:	
	057746	124	110	105	.ASCIZ	/THE FPS WAS BAD AFTER ADDD (R),A./
6703	060010	101	104	104	EM211:	'ADDD (R),A FAILED IN THE ROUND\TRUNK FLOWS.'
6704	060063	200	127	105	.ASCII	<CRLF>'WENT FROM STATE 663 TO 313,'<CRLF>
6705	060120	111	116	123	.ASCIZ	'INSTEAD OF FROM 663 TO 353.'
6730	060154				EM212:	
	060154	101	104	104	.ASCII	'ADDD (R),A FAILED IN THE ROUND\TRUNK FLOWS.'
	060227	200	124	110	.ASCII	<CRLF>'THE FPS WAS BAD.'<CRLF>
	060251	104	111	104	.ASCII	'DID NOT TAKE THE PATH:'<CRLF>
	060300	106	122	117	.ASCIZ	/FROM STATE 664, TO 505, TO 251./
6731	060340				EM213:	
	060340	101	104	104	.ASCII	'ADDD (R),A FAILED IN THE ROUND\TRUNK FLOWS.'
	060413	200	124	110	.ASCII	<CRLF>'THE FPS WAS BAD.'<CRLF>
	060435	104	111	104	.ASCII	'DID NOT TAKE THE PATH:'<CRLF>
	060464	106	122	117	.ASCIZ	/FROM STATE 664, TO 505, TO 253./
6732	060524				EM214:	
	060524	101	104	104	.ASCII	'ADDD (R),A FAILED IN THE ROUND\TRUNK FLOWS.'
	060577	200	124	110	.ASCII	<CRLF>'THE FPS WAS BAD.'<CRLF>
	060621	104	111	104	.ASCII	'DID NOT TAKE THE PATH:'<CRLF>
	060650	106	122	117	.ASCIZ	/FROM STATE 664, TO 705, TO 735./
6733	060710				EM215:	
	060710	101	104	104	.ASCII	'ADDD (R),A FAILED IN THE ROUND\TRUNK FLOWS.'
	060763	200	124	110	.ASCII	<CRLF>'THE FPS WAS BAD.'<CRLF>
	061005	104	111	104	.ASCII	'DID NOT TAKE THE PATH:'<CRLF>
	061034	106	122	117	.ASCIZ	/FROM STATE 664, TO 705, TO 737./
6734	061074	124	110	105	EM216:	'THE (BUT FIU FORK IN THE OVER\UNDER FLOWS FAILED. FIU =1.'
6735	061165	200	127	105	.ASCII	<CRLF>'WENT FROM STATE 331 TO 115.'<CRLF>
6736	061222	111	116	123	.ASCIZ	'INSTEAD OF FROM 331 TO 155.'
6737	061256	101	104	104	EM217:	'ADDD (R)A TRAPPED TO 244,.. FID=1.'
6738	061320				EM220:	
	061320	101	104	104	.ASCII	/ADDD (R),A TRAPPED TO 244./<CRLF>
	061353	124	110	105	.ASCII	'THE RESULT WAS AN OVERFLOW CONDITION BUT FIV= 0.'
	061433	200	050	102	.ASCIZ	<CRLF>/ (BUT FIV) STATE 133 WENT TO 136 INSTEAD OF 116./
	061514	111	116	123	.ASCIZ	/INSTEAD OF FROM 133 TO 116./
6739	061550				EM221:	
	061550	101	104	104	.ASCII	/ADDD (R),A FAILED TO TRAP TO 244./<CRLF>
	061612	124	110	105	.ASCII	'THE RESULT WAS A OVERFLOW CONDITION AND FIV=1.'<CRLF>
	061671	124	110	105	.ASCII	/THE (BUT FIV) FORK FAILED./<CRLF>
	061724	127	105	116	.ASCII	/WENT FROM STATE 133 TO 116./<CRLF>
	061760	111	116	123	.ASCIZ	/INSTEAD OF FROM 133 TO 136./
6740	062014				EM222:	
	062014	101	104	104	.ASCII	/ADDD (R),A TRAPPED TO 244./<CRLF>
	062047	124	110	105	.ASCII	'THE RESULT WAS AN UNDERFLOW CONDITION BUT FIU= 0.'
	062130	200	050	102	.ASCIZ	<CRLF>/ (BUT FIU) STATE 331 WENT TO 155 INSTEAD OF 115./
	062211	111	116	123	.ASCIZ	/INSTEAD OF FROM 331 TO 115./
6741	062245				EM223:	
	062245	101	104	104	.ASCII	/ADDD (R),A FAILED TO TRAP TO 244./<CRLF>
	062307	124	110	105	.ASCII	'THE RESULT WAS A UNDERFLOW CONDITION AND FIU=1.'<CRLF>
	062367	124	110	105	.ASCII	/THE (BUT FIU) FORK FAILED./<CRLF>
	062422	127	105	116	.ASCII	/WENT FROM STATE 331 TO 115./<CRLF>
	062456	111	116	123	.ASCIZ	/INSTEAD OF FROM 331 TO 155./
6742	062512				EM224:	
	062512	101	104	104	.ASCII	/ADDD (R),A TRAPPED TO 244./<CRLF>
	062545	124	110	105	.ASCII	'THE RESULT WAS AN UNDERFLOW CONDITION BUT FIU= 0.'
	062626	200	050	102	.ASCIZ	<CRLF>/ (BUT FIU) STATE 137 WENT TO 155 INSTEAD OF 115./

6743	062707	111	116	123	EM225:	.ASCIZ	/INSTEAD OF FROM 137 TO 115./
	062743					.ASCII	/ADDD (R),A FAILED TO TRAP TO 244./<CRLF>
	062743	101	104	104		.ASCII	'THE RESULT WAS A UNDERFLOW CONDITION AND FIU=1.'<CRLF>
	063005	124	110	105		.ASCII	/THE (BUT FIU) FORK FAILED./<CRLF>
	063065	124	110	105		.ASCII	/WENT FROM STATE 137 TO 115./<CRLF>
	063120	127	105	116		.ASCIZ	/INSTEAD OF FROM 137 TO 155./
	063154	111	116	123			
6744	063210				EM226:	.ASCII	/ADDD (R),A TRAPPED TO 244./
	063210	101	104	104		.ASCII	<CRLF>'BECAUSE OF AN EXPECTED OVERFLOW CONDITION,'<CRLF>
	063242	200	102	105		.ASCIZ	'BUT THE FEC WAS BAD.'
	063316	102	125	124			
6745	063343				EM227:	.ASCII	/ADDD (R),A TRAPPED TO 244./
	063343	101	104	104		.ASCII	<CRLF>'BECAUSE OF AN EXPECTED OVERFLOW CONDITION,'<CRLF>
	063375	200	102	105		.ASCIZ	'BUT THE FPS WAS BAD.'
	063451	102	125	124			
6746	063476				EM230:	.ASCII	/ADDD (R),A TRAPPED TO 244./
	063476	101	104	104		.ASCII	<CRLF>'BECAUSE OF AN EXPECTED UNDERFLOW CONDITION,'<CRLF>
	063530	200	102	105		.ASCIZ	'BUT THE FEC WAS BAD.'
	063605	102	125	124			
6747	063632				EM231:	.ASCII	/ADDD (R),A TRAPPED TO 244./
	063632	101	104	104		.ASCII	<CRLF>'BECAUSE OF AN EXPECTED UNDERFLOW CONDITION,'<CRLF>
	063664	200	102	105		.ASCIZ	'BUT THE FPS WAS BAD.'
	063741	102	125	124			
6748		057746			EM232=EM210		
6749							
6750							
6751							
6756							
6760							
6767							
6777							
6778	063766				EM233:	.ASCIZ	\LDCFD (R)+,A RESULT INCORRECT.\
	063766	114	104	103			
6779	064025				EM234:	.ASCII	\RO BAD AFTER LDCFD (R)+,A.\
	064025	122	060	040		.ASCIZ	<CRLF>'A BAD CONSTANT WAS USED.'
	064057	200	101	040			
6780	064111				EM235:	.ASCIZ	\PC BAD AFTER LDCFD #NUM,A. TRAP TO 4.\
	064111	120	103	040			
6781	064157				EM236:	.ASCIZ	\PC BAD AFTER LDCFD #NUM,A.\
	064157	120	103	040			
6782	064212				EM237:	.ASCII	\RO BAD AFTER LDCDF (R)+,A.\
	064212	122	060	040		.ASCIZ	<CRLF>'A BAD CONSTANT WAS USED.'
	064244	200	101	040			
6783	064276				EM240:	.ASCIZ	\FPS BAD AFTER LDCFD (R)+,A.\
	064276	106	120	123			
6784	064332				EM241:	.ASCII	\LDCFD (R)+,A FAILED.\
	064332	114	104	103		.ASCII	<CRLF>'THE FD '
	064356	200	124	110		.ASCII	'BIT WAS NOT COMPLIMENTED '
	064366	102	111	124		.ASCIZ	'IN STATE 017.'
	064417	111	116	040			
6785	064435				EM242:	.ASCIZ	\FPS BAD AFTER LDCDF (R)+,A.\
	064435	106	120	123			
6786	064471				EM243:	.ASCII	\LDCDF (R)+,A FAILED.\
	064471	114	104	103		.ASCII	<CRLF>'THE FD '
	064515	200	124	110		.ASCII	'BIT WAS NOT COMPLIMENTED '
	064525	102	111	124		.ASCIZ	'IN STATE 017.'
	064556	111	116	040			

6787	064574				EM244:	.ASCIZ	\LDCDF (R)+,A RESULT INCORRECT.\
	064574	114	104	103			
6788							
6789	064633	114	104	103	EM245:	.ASCII	'LDCFD (R),A FAILED.'
6790	064656	200	123	105		.ASCII	<CRLF>'SET SIGN FAILED '
6791	064677	111	116	040		.ASCIZ	'IN STATE 512.'
6792	064715	125	116	105	EM247:	.ASCIZ	'UNEXPECTED FPP TRAP TO 244.'
6793	064751	125	116	105	EM250:	.ASCIZ	'UNEXPECTED CPU TRAP TO 4.'
6794	065003	125	116	105	EM251:	.ASCIZ	'UNEXPECTED CPU TRAP TO 10.'
6795							
6796	065036	103	117	122	EM252:	.ASCIZ	'CORRECT FLOWS INTERRUPT TEST MODULE FAILED TO INTERRUPT.'
6797							
6801							
6802	065127				EM253:		
	065127	101	104	104		.ASCIZ	/ADDD ACO,ACO FAILED IN THE INTERRUPT CORRECT FLOWS./
6803	065213				EM254:		
	065213	101	104	104		.ASCIZ	/ADDD (RO),ACO FAILED IN THE INTERRUPT CORRECT FLOWS./
6804	065300				EM255:		
	065300	101	104	104		.ASCIZ	/ADDD (RO)+,ACO FAILED IN THE INTERRUPT CORRECT FLOWS./
6805	065366				EM256:		
	065366	101	104	104		.ASCIZ	/ADDD @ (RO)+,ACO FAILED IN THE INTERRUPT CORRECT FLOWS./
6806	065455				EM257:		
	065455	101	104	104		.ASCIZ	/ADDD -(RO),ACO FAILED IN THE INTERRUPT CORRECT FLOWS./
6807	065543				EM260:		
	065543	101	104	104		.ASCIZ	/ADDD @-(RO),ACO FAILED IN THE INTERRUPT CORRECT FLOWS./
6808	065632				EM261:		
	065632	101	104	104		.ASCIZ	/ADDD NUM(RO),ACO FAILED IN THE INTERRUPT CORRECT FLOWS./
6809	065722				EM262:		
	065722	101	104	104		.ASCIZ	/ADDD @NUM(RO),ACO FAILED IN THE INTERRUPT CORRECT FLOWS./
6810	066013				EM263:		
	066013	104	111	126		.ASCIZ	/DIVD (RO),ACO FAILED IN THE INTERRUPT CORRECT FLOWS./
6811	066100				EM264:		
	066100	115	125	114		.ASCIZ	/MULD (RO),ACO FAILED IN THE INTERRUPT CORRECT FLOWS./
6812	066165				EM265:		
	066165	115	117	104		.ASCIZ	/MODD (RO),ACO FAILED IN THE INTERRUPT CORRECT FLOWS./
6813							
6814	066252	103	117	122	EM266:	.ASCIZ	'CORRECT FLOWS INTERRUPT TEST MODULE CAUSED UNEXPECTED INTERRUPT.'
6815	066353				EM267:		
	066353	115	117	104		.ASCII	/MODD (R),A INTEGER BAD. RESULT OVER OR UNDER FLOW./
6816	066435	000				.BYTE	0

.SBTTL ERROR DATA TABLE HEADERS									
6817									
6818									
6819	066436	040	040	124	DH1:	.ASCII	' TEST.'	'<TAB>'CALL AT PC.'	'<TAB>'ERROR AT PC.'
6820	066477	107	117	124		.ASCIZ	'GOT FPS.'	'<TAB>'EXPECTED FPS.'	
6821	066526	040	040	124	DH2:	.ASCIZ	' TEST.'	'<TAB>'CALL AT PC.'	'<TAB>'ERROR AT PC.'
6822	066436				DH3=	DH1			
6823	066436				DH4=	DH1			
6824	066436				DH5=	DH1			
6825	066436				DH6=	DH1			
6826	066436				DH7=	DH1			
6827	066436				DH10=	DH1			
6828	066436				DH11=	DH1			
6829	066436				DH12=	DH1			
6830	066436				DH13=	DH1			
6831	066436				DH14=	DH1			
6832	066436				DH15=	DH1			
6833	066436				DH16=	DH1			
6834	066567	040	040	124	DH17:	.ASCIZ	' TEST.'	'<TAB>'PC OF CALL.'	'<TAB>'PC OF ERROR.'
6835	066436				DH20=	DH1			
6836	066642	040	040	124	DH21:	.ASCII	' TEST.'	'<TAB>'PC OF CALL'	'<TAB>'PC OF ERROR.'
6837	066702	107	117	124		.ASCIZ	'GOT FEC.'	'<TAB>'EXPECTED FEC.'	
6838	066731	040	040	124	DH22:	.ASCIZ	' TEST.'	'<TAB>'PC OF CALL.'	'<TAB>'PC OF ERROR.'
6839	066436				DH23=	DH1			
6840	066436				DH32=	DH1			
6841	066436				DH24=	DH1			
6842	066436				DH25=	DH1			
6843	066436				DH26=	DH1			
6844	066436				DH27=	DH1			
6845	066436				DH30=	DH1			
6846	066436				DH31=	DH1			
6847	066436				DH33=	DH1			
6848	066436				DH34=	DH1			
6849	066436				DH35=	DH1			
6850	066436				DH36=	DH1			
6851	066436				DH37=	DH1			
6852	066436				DH40=	DH1			
6853	066436				DH41=	DH1			
6854	066436				DH42=	DH1			
6855	066436				DH43=	DH1			
6856	066436				DH44=	DH1			
6857	066436				DH45=	DH1			
6858	066436				DH246=	DH1			
6859	066436				DH46=	DH1			
6860	066436				DH47=	DH1			
6861	066436				DH50=	DH1			
6862	066436				DH51=	DH1			
6863	066436				DH52=	DH1			
6864	066436				DH111=	DH1			
6865	066436				DH112=	DH1			
6866	066436				DH113=	DH1			
6867	066436				DH114=	DH1			
6868	066777	040	040	124	DH115:	.ASCII	' TEST.'	'<TAB>'PC OF CALL.'	'<TAB>'PC OF ERROR.'
6869	067040	107	117	124		.ASCIZ	'GOT FEC.'	'GOT FPS.'	'EXPECTED FPS.'
6870	066777				DH116=	DH115			
6871	066777				DH117=	DH115			
6872	066777				DH120=	DH115			
6873	066777				DH121=	DH115			

6874	066777	DH122=DH115
6875	066436	DH123=DH1
6876	066436	DH124=DH1
6877	066436	DH125=DH1
6878	066436	DH126=DH1
6879	066777	DH127=DH115
6880	066777	DH130=DH115
6881	066777	DH131=DH115
6882	066436	DH132=DH1
6883	066777	DH133=DH115
6884	066777	DH134=DH115
6885	066436	DH135=DH1
6886	066777	DH136=DH115
6887	066777	DH137=DH115
6888	066777	DH140=DH115
6889	066777	DH141=DH115
6890	066777	DH142=DH115
6891	066777	DH143=DH115
6892	066777	DH144=DH115
6893	066436	DH145=DH1
6894	066436	DH146=DH1
6895	066436	DH147=DH1
6896	066436	DH150=DH1
6897	066777	DH151=DH115
6898	066777	DH152=DH115
6899	066777	DH153=DH115
6900	066777	DH154=DH115
6901	066777	DH155=DH115
6902	066777	DH156=DH115
6903	066436	DH157=DH1
6904	066777	DH160=DH115
6905	066436	DH161=DH1
6906	066436	DH162=DH1
6907	066777	DH163=DH115
6908	066436	DH164=DH1
6909	066436	DH53=DH1
6910	066436	DH54=DH1
6911	066436	DH55=DH1
6912	066436	DH56=DH1
6913	066436	DH57=DH1
6914	066436	DH60=DH1
6915	066436	DH61=DH1
6916	066436	DH62=DH1
6917	066436	DH63=DH1
6918	066436	DH64=DH1
6919	066436	DH65=DH1
6920	066436	DH66=DH1
6921	066436	DH67=DH1
6922	066436	DH70=DH1
6923	066436	DH71=DH1
6924	066436	DH72=DH1
6925	066436	DH73=DH1
6926	066436	DH74=DH1
6927	066436	DH75=DH1
6928	066436	DH76=DH1
6929	066436	DH77=DH1
6930	066436	DH100=DH1

6931	066436				DH101=DH1	
6932	066436				DH102=DH1	
6933	066436				DH103=DH1	
6934	066436				DH104=DH1	
6935	066436				DH105=DH1	
6936	066436				DH106=DH1	
6937	066436				DH107=DH1	
6938	066436				DH110=DH1	
6939	066777				DH165=DH115	
6940	066777				DH166=DH115	
6941	066777				DH167=DH115	
6942	066777				DH170=DH115	
6943	066777				DH171=DH115	
6944	066777				DH172=DH115	
6945	066777				DH173=DH115	
6946	066777				DH174=DH115	
6947	066777				DH175=DH115	
6948	066777				DH176=DH115	
6949	067076	040	040	124	DH177: .ASCIZ	: TEST:<TAB>:ERR PC:<TAB>:CPU ERROR REGISTER:
6950	066777				DH200=DH115	
6951	066777				DH201=DH115	
6952	066777				DH202=DH115	
6953	066777				DH203=DH115	
6954	066777				DH204=DH115	
6955	066777				DH205=DH115	
6956	066777				DH206=DH115	
6957	067137	040	040	124	DH220: .ASCIZ	' TEST.'<TAB>'PC OF CALL.'<TAB>'PC OF TRAP.'
6958	067177	040	040	124	DH207: .ASCII	' TEST.'<TAB>'PC OF CALL.'<TAB>'PC OF ERROR.'
6959	067237	011	107	117	.ASCIZ	<TAB>'GOT FPS.'<TAB>'EXPECTED FPS.'
6960	067267	040	040	124	DH210: .ASCIZ	' TEST.'<TAB>'PC OF CALL.'<TAB>'PC OF ERROR.'
6961	067267				DH211=DH210	
6962	067177				DH212=DH207	
6963	067177				DH213=DH207	
6964	067177				DH214=DH207	
6965	067177				DH215=DH207	
6966	067267				DH216=DH210	
6967	067330	040	040	124	DH217: .ASCII	' TEST.'<TAB>'PC OF CALL.'<TAB>'PC OF ERROR.'
6968	067370	011	106	120	.ASCIZ	<TAB>'FPS.'<TAB>'FEC.'
6969	067267				DH221=DH210	
6970	067137				DH222=DH220	
6971	067267				DH223=DH210	
6972	067137				DH224=DH220	
6973	067267				DH225=DH210	
6974	067403	040	040	124	DH226: .ASCII	' TEST.'<TAB>'PC OF CALL.'<TAB>'PC OF TRAP.'
6975	067442	011	107	117	.ASCIZ	<TAB>'GOT FEC.'<TAB>'EXPECTED FEC.'
6976	067472	040	040	124	DH227: .ASCII	' TEST.'<TAB>'PC OF CALL.'<TAB>'PC OF TRAP.'
6977	067531	011	107	117	.ASCIZ	<TAB>'GOT FPS.'<TAB>'EXPECTED FPS.'
6978	067403				DH230=DH226	
6979	067472				DH231=DH227	
6980	067472				DH232=DH227	
6981						
6982						
6983	067561	040	040	124	DH233: .ASCIZ	' TEST.'<TAB>'PC OF CALL.'<TAB>'PC OF ERROR.'
6984						
6985	067622	040	040	124	DH234: .ASCII	' TEST.'<TAB>'PC OF CALL.'<TAB>'PC OF ERROR.'
6986	067662	011	107	117	.ASCIZ	<TAB>'GOT RO.'<TAB>'EXPECTED RO.'
6987						

6988	067710	040	040	124	DH235:	.ASCII	' TEST.' <tab>'PC OF CALL.'<tab></tab></tab>
6989	067734	120	103	040		.ASCII	'PC OF TRAP.' <tab></tab>
6990							
6991	067750	040	040	124	DH236:	.ASCII	' TEST.' <tab>'PC OF CALL.'<tab></tab></tab>
6992	067774	120	103	040		.ASCII	'PC OF ERROR.' <tab>'GOT PC.'</tab>
6993	070020	011	105	130		.ASCIZ	<TAB>'EXPECTED PC.'
6994							
6995	067622						DH237=DH234
6996							
6997	070036	040	040	124	DH240:	.ASCII	' TEST.' <tab>'PC OF CALL.'<tab></tab></tab>
6998	070062	120	103	040		.ASCII	'PC OF ERROR.' <tab></tab>
6999	070077	107	117	124		.ASCIZ	'GOT FPS.' <tab>'EXPECTED FPS.'</tab>
7000							
7001	067561						DH241=DH233
7002	070036						DH242=DH240
7003	067561						DH243=DH233
7004	067561						DH244=DH233
7005	067561						DH245=DH233
7006	070126	040	040	124	DH247:	.ASCIZ	' TEST.' <tab>'PC OF CALL.'<tab>'PC OF TRAP.'<tab>'FEC.'</tab></tab></tab>
7007	070173	040	040	124	DH250:	.ASCIZ	' TEST.' <tab>'PC OF CALL.'<tab>'PC OF TRAP.'</tab></tab>
7008	070173						DH251=DH250
7009							
7010	066526						DH252=DH2
7011	066436						DH253=DH1
7012	066436						DH254=DH1
7013	066436						DH255=DH1
7014	066436						DH256=DH1
7015	066436						DH257=DH1
7016	066436						DH260=DH1
7017	066436						DH261=DH1
7018	066436						DH262=DH1
7019	066436						DH263=DH1
7020	066436						DH264=DH1
7021	066436						DH265=DH1
7022	070233	040	040	124	DH266:	.ASCIZ	' TEST.' <tab>'PC OF CALL.'<tab>'PC OF TRAP.'</tab></tab>
7023	066777						DH267=DH115

Line	Key	DF	Value	DF	DF	DF
7024					.SBTTL	DATA FORMAT SPECIFIERS
7025	070273	004	000	005	DF1:	.BYTE 4,0,5,0,5,0,5,0,5,5,3,5,5,3
7026	070311	004	000	005	DF2:	.BYTE 4,0,5,0,5,5,3,5,5,3
7027		070273			DF3=DF1	
7028		070273			DF4=DF1	
7029		070273			DF5=DF1	
7030		070273			DF6=DF1	
7031		070273			DF7=DF1	
7032		070273			DF10=DF1	
7033		070273			DF11=DF1	
7034		070273			DF12=DF1	
7035		070273			DF13=DF1	
7036		070273			DF14=DF1	
7037		070273			DF15=DF1	
7038		070273			DF16=DF1	
7039	070323	004	000	005	DF17:	.BYTE 4,0,5,0,5,0,0
7040	070332	004	000	005	DF20:	.BYTE 4,0,5,0,5,0,5,0
7041		070332			DF21=DF20	
7042		070332			DF22=DF20	
7043	070342	004	000	005	DF23:	.BYTE 4,0,5,0,5,0,5,0,5,5,2,5,5,2,5,5,2,5,5,2
7044		070342			DF24=DF23	
7045		070342			DF32=DF23	
7046		070342			DF25=DF23	
7047		070342			DF26=DF23	
7048		070342			DF27=DF23	
7049		070342			DF30=DF23	
7050		070342			DF31=DF23	
7051	070366	004	000	005	DF33:	.BYTE 4,0,5,0,5,0,5,0,5,5,3,5,5,3,5,5,3,5,5,3
7052		070366			DF34=DF33	
7053		070366			DF35=DF33	
7054		070366			DF36=DF33	
7055		070342			DF37=DF23	
7056		070342			DF40=DF23	
7057		070342			DF41=DF23	
7058		070342			DF42=DF23	
7059		070342			DF43=DF23	
7060		070342			DF44=DF23	
7061		070342			DF45=DF23	
7062		070366			DF246=DF33	
7063		070366			DF46=DF33	
7064		070366			DF47=DF33	
7065		070366			DF50=DF33	
7066		070366			DF51=DF33	
7067		070366			DF52=DF33	
7068		070342			DF111=DF23	
7069		070342			DF112=DF23	
7070		070342			DF113=DF23	
7071		070342			DF114=DF23	
7072	070412	004	000	005	DF115:	.BYTE 4,0,5,0,5,0,0,0,5,5,2,5,5,2,5,5,2,5,5,2
7073		070412			DF116=DF115	
7074		070412			DF117=DF115	
7075		070412			DF120=DF115	
7076		070412			DF121=DF115	
7077		070412			DF122=DF115	
7078		070366			DF123=DF33	
7079		070366			DF124=DF33	
7080		070366			DF125=DF33	

7081		070366			DF126=DF33	
7082	070436	004	000	005	DF127: .BYTE	4,0,5,0,5,0,0,0,5,5,3,5,5,3,5,5,3,5,5,3
7083		070436			DF130=DF127	
7084		070436			DF131=DF127	
7085		070366			DF132=DF33	
7086		070436			DF133=DF127	
7087		070436			DF134=DF127	
7088		070366			DF135=DF33	
7089		070436			DF136=DF127	
7090		070412			DF137=DF115	
7091		070412			DF140=DF115	
7092		070412			DF141=DF115	
7093		070412			DF142=DF115	
7094		070412			DF143=DF115	
7095		070412			DF144=DF115	
7096		070342			DF145=DF23	
7097		070342			DF146=DF23	
7098		070342			DF147=DF23	
7099		070342			DF150=DF23	
7100		070436			DF151=DF127	
7101		070436			DF152=DF127	
7102		070436			DF153=DF127	
7103		070436			DF154=DF127	
7104		070436			DF155=DF127	
7105		070436			DF156=DF127	
7106		070366			DF157=DF33	
7107		070436			DF160=DF127	
7108		070366			DF161=DF33	
7109		070366			DF162=DF33	
7110		070436			DF163=DF127	
7111		070366			DF164=DF33	
7112	070462	004	000	005	DF53: .BYTE	4,0,5,0,5,0,5,0,5,5,2,5,5,2,5,5,2,5,5,2,5,5,2,5,5,2
7113		070462			DF54=DF53	
7114		070462			DF55=DF53	
7115		070462			DF56=DF53	
7116		070462			DF57=DF53	
7117		070462			DF60=DF53	
7118		070462			DF61=DF53	
7119		070462			DF62=DF53	
7120		070462			DF63=DF53	
7121		070462			DF64=DF53	
7122		070462			DF65=DF53	
7123		070462			DF66=DF53	
7124		070462			DF67=DF53	
7125	070514	004	000	005	DF70: .BYTE	4,0,5,0,5,0,5,0,5,5,3,5,5,3,5,5,3,5,5,3,5,5,3,5,5,3
7126		070514			DF71=DF70	
7127		070514			DF72=DF70	
7128		070514			DF73=DF70	
7129		070514			DF74=DF70	
7130		070514			DF75=DF70	
7131		070514			DF76=DF70	
7132		070514			DF77=DF70	
7133		070514			DF100=DF70	
7134		070514			DF101=DF70	
7135		070514			DF102=DF70	
7136		070514			DF103=DF70	
7137		070514			DF104=DF70	

7138	070514			DF105=DF70	
7139	070514			DF106=DF70	
7140	070514			DF107=DF70	
7141	070514			DF110=DF70	
7142	070546	000	005	DF165: .BYTE	4,0,5,0,5,0,0,0,5,5,2,5,5,2,5,5,2,5,5,2,5,5,2,5,5,2
7143	070546			DF166=DF165	
7144	070546			DF167=DF165	
7145	070546			DF170=DF165	
7146	070546			DF171=DF165	
7147	070546			DF172=DF165	
7148	070546			DF173=DF165	
7149	070546			DF174=DF165	
7150	070546			DF175=DF165	
7151	070600	000	005	DF176: .BYTE	4,0,5,0,5,0,0,0,5,5,3,5,5,3,5,5,3,5,5,3,5,5,3,5,5,3
7152	070632	000	000	DF177: .BYTE	4,0,0
7153	070600			DF200=DF176	
7154	070600			DF201=DF176	
7155	070600			DF202=DF176	
7156	070600			DF203=DF176	
7157	070600			DF204=DF176	
7158	070600			DF205=DF176	
7159	070600			DF206=DF176	
7160	070635	000	005	DF210: .BYTE	4,0,5,0,5,0,5,0
7161	070645	000	005	DF207: .BYTE	4,0,5,0,5,5,5,3,5,5,5,3,5,5,5,3,5,5,5,3
7162	070645			DF211=DF207	
7163	070671	000	005	DF212: .BYTE	4,0,5,0,5,0,5,0,5,5,5,3,5,5,5,3,5,5,5,3,5,5,5,3
7164	070671			DF213=DF212	
7165	070671			DF214=DF212	
7166	070671			DF215=DF212	
7167	070645			DF216=DF207	
7168	070721	000	005	DF217: .BYTE	4,0,5,0,5,0,0
7169	070645			DF220=DF207	
7170	070645			DF221=DF207	
7171	070645			DF222=DF207	
7172	070645			DF223=DF207	
7173	070645			DF224=DF207	
7174	070645			DF225=DF207	
7175	070671			DF226=DF212	
7176	070671			DF227=DF212	
7177	070671			DF230=DF212	
7178	070671			DF231=DF212	
7179	070671			DF232=DF212	
7180	070730	000	005	DF233: .BYTE	4,0,5,0,5,5,3,5,5,3,5,5,3
7181	070745	000	005	DF234: .BYTE	4,0,5,0,5,0,0
7182					
7183	070754	000	005	DF235: .BYTE	4,0,5,0
7184	070745			DF236=DF234	
7185	070745			DF237=DF234	
7186	070745			DF240=DF234	
7187	070730			DF241=DF233	
7188	070745			DF242=DF234	
7189	070730			DF243=DF233	
7190	070730			DF244=DF233	
7191	070730			DF245=DF233	
7192	070760	000	005	DF247: .BYTE	4,0,5,0,5,0
7193	070760			DF250=DF247	
7194	070760			DF251=DF247	


```
7211 .SBTTL ERROR DATA TABLES
7212 071024 001232 001234 037335 DT1: .WORD $TMP0,$TMP1,$TAB,$TMP2,$TAB,$TMP5,$TAB,$TMP6
7213 071044 001313 037342 001240 .WORD $CRLF,MS1,$TMP3,$CRLF,MS2,$TMP4,0
7214 071062 001232 001234 037335 DT2: .WORD $TMP0,$TMP1,$TAB,$TMP2,$CRLF,MS3,$TMP3,$CRLF,MS4,$TMP4,0
7215 071024 DT3=DT1
7216 071024 DT4=DT1
7217 071024 DT5=DT1
7218 071024 DT6=DT1
7219 071024 DT7=DT1
7220 071024 DT10=DT1
7221 071024 DT11=DT1
7222 071024 DT12=DT1
7223 071024 DT13=DT1
7224 071024 DT14=DT1
7225 071024 DT15=DT1
7226 071024 DT16=DT1
7227 071110 001232 001234 037335 DT17: .WORD $TMP0,$TMP1,$TAB,$TMP2,$TAB,$TMP4,$TMP5,0
7228 071130 001232 001234 037335 DT20: .WORD $TMP0,$TMP1,$TAB,$TMP2,$TAB,$TMP4,$TAB,$TMP5,0
7229 071152 001232 001234 037335 DT21: .WORD $TMP0,$TMP1,$TAB,$TMP2,$TAB,$TMP4,$TAB,$TMP3,0
7230 071174 001232 001234 037335 DT22: .WORD $TMP0,$TMP1,$TAB,$TMP2,$TAB,$TMP5,0
7231 071212 001232 001234 037335 DT23: .WORD $TMP0,$TMP1,$TAB,$TMP2,$TAB,$TMP7,$TAB,$TMP10,$CRLF,MS1,$TMP3,$CRLF,MS2,$TMP
7232 071246 001313 037455 001244 .WORD $CRLF,MS5,$TMP5,$CRLF,MS6,$TMP6,0
7233 071212 DT32=DT23
7234 071212 DT24=DT23
7235 071212 DT25=DT23
7236 071212 DT26=DT23
7237 071212 DT27=DT23
7238 071212 DT30=DT23
7239 071212 DT31=DT23
7240 071212 DT33=DT23
7241 071212 DT34=DT23
7242 071212 DT35=DT23
7243 071212 DT36=DT23
7244 071212 DT37=DT23
7245 071212 DT40=DT23
7246 071212 DT41=DT23
7247 071212 DT42=DT23
7248 071212 DT43=DT23
7249 071212 DT44=DT23
7250 071212 DT45=DT23
7251 071212 DT46=DT23
7252 071212 DT47=DT23
7253 071212 DT50=DT23
7254 071212 DT51=DT23
7255 071212 DT52=DT23
7256 071264 001232 001234 037335 DT53: .WORD $TMP0,$TMP1,$TAB,$TMP2,$TAB,$TMP11,$TAB,$TMP12
7257 071304 001313 037342 001240 .WORD $CRLF,MS1,$TMP3,$CRLF,MS2,$TMP4
7258 071320 001313 037563 001244 .WORD $CRLF,MS11,$TMP5,$CRLF,MS12,$TMP6
7259 071334 001313 037515 001250 .WORD $CRLF,MS7,$TMP7,$CRLF,MS10,$TMP10,0
7260 071264 DT54=DT53
7261 071264 DT55=DT53
7262 071264 DT56=DT53
7263 071264 DT57=DT53
7264 071264 DT60=DT53
7265 071264 DT61=DT53
7266 071254 DT62=DT53
7267 071264 DT63=DT53
```


7268	071264			DT64=DT53
7269	071264			DT65=DT53
7270	071264			DT66=DT53
7271	071264			DT67=DT53
7272	071264			DT70=DT53
7273	071264			DT71=DT53
7274	071264			DT72=DT53
7275	071264			DT73=DT53
7276	071264			DT74=DT53
7277	071264			DT75=DT53
7278	071264			DT76=DT53
7279	071264			DT77=DT53
7280	071264			DT100=DT53
7281	071264			DT101=DT53
7282	071264			DT102=DT53
7283	071264			DT103=DT53
7284	071264			DT104=DT53
7285	071264			DT105=DT53
7286	071264			DT106=DT53
7287	071264			DT107=DT53
7288	071264			DT110=DT53
7289	071212			DT111=DT23
7290	071212			DT112=DT23
7291	071212			DT113=DT23
7292	071212			DT114=DT23
7293	071352	001232	001234 037335	DT115: .WORD
7294	071374	001313	037342 001240	.WORD
7295	071410	001313	037455 001244	.WORD
7296	071352			DT116=DT115
7297	071352			DT117=DT115
7298	071352			DT120=DT115
7299	071352			DT121=DT115
7300	071352			DT122=DT115
7301	071212			DT123=DT23
7302	071212			DT124=DT23
7303	071212			DT125=DT23
7304	071212			DT126=DT23
7305	071352			DT127=DT115
7306	071352			DT130=DT115
7307	071352			DT131=DT115
7308	071212			DT132=DT23
7309	071352			DT133=DT115
7310	071352			DT134=DT115
7311	071212			DT135=DT23
7312	071352			DT136=DT115
7313	071352			DT137=DT115
7314	071352			DT140=DT115
7315	071352			DT141=DT115
7316	071352			DT142=DT115
7317	071352			DT143=DT115
7318	071352			DT144=DT115
7319	071212			DT145=DT23
7320	071212			DT146=DT23
7321	071212			DT147=DT23
7322	071212			DT150=DT23
7323	071352			DT151=DT115
7324	071352			DT152=DT115

\$TMP0,\$TMP1,\$TAB,\$TMP2,\$TAB,\$TMP11,\$TMP7,\$TAB,\$TMP10
\$CRLF,\$MS1,\$TMP3,\$CRLF,\$MS2,\$TMP4
\$CRLF,\$MS5,\$TMP5,\$CRLF,\$MS6,\$TMP6,0

7325	071352			DT153=DT115	
7326	071352			DT154=DT115	
7327	071352			DT155=DT115	
7328	071352			DT156=DT115	
7329	071212			DT157=DT23	
7330	071352			DT160=DT115	
7331	071212			DT161=DT23	
7332	071212			DT162=DT23	
7333	071352			DT163=DT115	
7334	071212			DT164=DT23	
7335	071426	001232	001234 037335	DT165: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,\$TAB,\$TMP13,\$TMP11,\$TMP12
7336	071446	001313	037342 001240	.WORD	\$CRLF,MS1,\$TMP3,\$CRLF,MS2,\$TMP4
7337	071462	001313	037563 001244	.WORD	\$CRLF,MS11,\$TMP5,\$CRLF,MS12,\$TMP6
7338	071476	001313	037515 001250	.WORD	\$CRLF,MS7,\$TMP7,\$CRLF,MS10,\$TMP10,0
7339	071426			DT166=DT165	
7340	071426			DT167=DT165	
7341	071426			DT170=DT165	
7342	071426			DT171=DT165	
7343	071426			DT172=DT165	
7344	071426			DT173=DT165	
7345	071426			DT174=DT165	
7346	071426			DT175=DT165	
7347	071426			DT176=DT165	
7348	071514	001232	001234 033760	DT177: .WORD	\$TMP0,\$TMP1,CPSAVE,0
7349	071426			DT200=DT165	
7350	071426			DT201=DT165	
7351	071426			DT202=DT165	
7352	071426			DT203=DT165	
7353	071426			DT204=DT165	
7354	071426			DT205=DT165	
7355	071426			DT206=DT165	
7356	071524	001232	001234 037335	DT207: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,\$CRLF,MS41,\$CRLF,\$TMP3
7357	071544	001313	037730 001313	.WORD	\$CRLF,MS42,\$CRLF,\$TMP4,\$CRLF,MS43,\$CRLF,\$TMP5
7358	071564	001313	037763 001313	.WORD	\$CRLF,MS44,\$CRLF,\$TMP6,0
7359	071576	001232	001234 037335	DT210: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,\$TAB,\$TMP3
7360	071612	037335	001242 000000	.WORD	\$TAB,\$TMP4,0
7361	071524			DT211=DT207	
7362	071620	001232	001234 037335	DT212: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,\$TAB,\$TMP10,\$TAB,\$TMP11
7363	071640	001313	037704 001313	.WORD	\$CRLF,MS41,\$CRLF,\$TMP3,\$CRLF,MS42,\$CRLF,\$TMP4
7364	071660	001313	037746 001313	.WORD	\$CRLF,MS43,\$CRLF,\$TMP5,\$CRLF,MS44,\$CRLF,\$TMP6,0
7365	071620			DT213=DT212	
7366	071620			DT214=DT212	
7367	071620			DT215=DT212	
7368	071524			DT216=DT207	
7369	071702	001232	001234 037335	DT217: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,\$TAB,\$TMP3,\$TMP4,0
7370	071524			DT220=DT207	
7371	071524			DT221=DT207	
7372	071524			DT222=DT207	
7373	071524			DT223=DT207	
7374	071524			DT224=DT207	
7375	071524			DT225=DT207	
7376	071620			DT226=DT212	
7377	071620			DT227=DT212	
7378	071620			DT230=DT212	
7379	071576			DT231=DT210	
7380	071524			DT232=DT207	
7381	071722	001232	001234 037335	DT233: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,\$CRLF

7382	071734	037632	001244	001313	.WORD	MS37,\$TMP5,\$CRLF,MS40,\$TMP6
7383	071746	001313	037704	001250	.WORD	\$CRLF,MS41,\$TMP7,0
7384	071756	001232	001234	037335	DT234: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,\$TAB,\$TMP3,\$TMP4,0
7385	071776	001232	001234	037335	DT235: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,0
7386		071756			DT236=DT234	
7387		071756			DT237=DT234	
7388		071756			DT240=DT234	
7389		071722			DT241=DT233	
7390		071756			DT242=DT234	
7391		071722			DT243=DT233	
7392		071722			DT244=DT233	
7393		071722			DT245=DT233	
7394		071212			DT246=DT23	
7395	072010	001232	001234	037335	DT247: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,\$TAB,\$TMP3,0
7396	072026	001232	001234	037335	DT250: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,0
7397		072026			DT251=DT250	
7398	072040	001232	001234	037335	DT252: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,0
7399	072052	001232	001234	037335	DT253: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,\$TAB,\$TMP7,\$TAB,\$TMP10
7400	072072	001313	037144	001254	.WORD	\$CRLF,MSA1,\$TMP11,\$CRLF,MSA2,\$TMP12
7401	072106	001313	037205	001244	.WORD	\$CRLF,MSA3,\$TMP5,\$CRLF,MSA4,\$TMP6
7402	072122	001313	037235	001240	.WORD	\$CRLF,MSA5,\$TMP3,\$CRLF,MSA6,\$TMP4,0
7403		072052			DT254=DT253	
7404		072052			DT255=DT253	
7405		072052			DT256=DT253	
7406		072052			DT257=DT253	
7407		072052			DT260=DT253	
7408		072052			DT261=DT253	
7409		072052			DT262=DT253	
7410		072052			DT263=DT253	
7411		072052			DT264=DT253	
7412		072052			DT265=DT253	
7413		072040			DT266=DT252	
7414		071426			DT267=DT165	
7415	072140	007070	007070	007070	.WORD	7070,7070,7070,7070 ;KLUDGE FOR APT LOAD PROBLEM
7416		000001			.END	

SYMBOL TABLE

AAADON 014416
AAA1 013414
AAA10 014032
AAA11 014070
AAA12 014126
AAA13 014164
AAA2 013452
AAA3 013510
AAA4 013546
AAA5 013604
AAA6 013642
AAA7 013700
AAA8 013736
AAA9 013774
ABASE = 000000
ACDW1 = 000000
ACDW2 = 000000
ACPUOP= 000000
ACO =%000000
AC1 =%000001
AC2 =%000002
AC3 =%000003
AC4 =%000004
AC5 =%000005
AC6 =%000006
AC7 =%000007
ADDW0 = 000000
ADDW1 = 000000
ADDW10= 000000
ADDW11= 000000
ADDW12= 000000
ADDW13= 000000
ADDW14= 000000
ADDW15= 000000
ADDW2 = 000000
ADDW3 = 000000
ADDW4 = 000000
ADDW5 = 000000
ADDW6 = 000000
ADDW7 = 000000
ADDW8 = 000000
ADDW9 = 000000
ADEVCT= 000000
ADEVM = 000000
AENV = 000000
AENVM = 000000
AFATAL= 000000
AMADR1= 000000
AMADR2= 000000
AMADR3= 000000
AMADR4= 000000
AMAMS1= 000000
AMAMS2= 000000
AMAMS3= 000000
AMAMS4= 000000
AMSGAD= 000000
AMSGLG= 000000

AMSGTY= 000000
AMTYP1= 000000
AMTYP2= 000000
AMTYP3= 000000
AMTYP4= 000000
APASS = 000000
APRIOR= 000000
APTCSU= 000040
APTENV= 000001
APTSIZ= 000200
APTSPO= 000100
ASWREG= 000000
ATESTN= 000000
AUNIT = 000000
AUSWR = 000000
AVECT1= 000000
AVECT2= 000000
BBBDON 015120
BBBER1 014674
BBBER2 014760
BBBER3 015006
BBBER4 015034
BBBP1 015100
BBBP2 015110
BBB0 014424
BBB1 014460
BBB2 014506
BBB3 014536
BBB4 014564
BBB5 014622
BBB6 014630
BIT0 = 000001
BIT00 = 000001
BIT01 = 000002
BIT02 = 000004
BIT03 = 000010
BIT04 = 000020
BIT05 = 000040
BIT06 = 000100
BIT07 = 000200
BIT08 = 000400
BIT09 = 001000
BIT1 = 000002
BIT10 = 002000
BIT11 = 004000
BIT12 = 010000
BIT13 = 020000
BIT14 = 040000
BIT15 = 100000
BIT2 = 000004
BIT3 = 000010
BIT4 = 000020
BIT5 = 000040
BIT6 = 000100
BIT7 = 000200
BIT8 = 000400
BIT9 = 001000

BPTVEC= 000014
CCCDON 016130
CCC1 015124
CCC10 015520
CCC11 015554
CCC12 015610
CCC13 015644
CCC2 015160
CCC3 015214
CCC4 015250
CCC5 015304
CCC6 015340
CCC7 015374
CCC8 015430
CCC9 015464
CKSWR = 104406
CMPSUB 014226
CMPTMP 014406
CNT = 000270
CORDON 033152
CORFLG 033134
CORINT 033146
CORMES 040005
CORSUB 032652
CORTMP 033136
CORTRP 033150
CORTV 032750
CORTV0 033112
CORTV1 033116
COR1 032020
COR10 032442
COR11 032506
COR12 032546
COR13 032606
COR2 032044
COR3 032060
COR33 032072
COR4 032132
COR5 032172
COR6 032232
COR7 032276
COR8 032336
COR9 032402
CPSAVE 033760
CPSPUR 037032
CPTWO 037050
CR = 000015
CRLF = 000200
DDDDON 017064
DDD1 016134
DDD2 016210
DDD3 016264
DDD4 016340
DDD5 016414
DDD6 016470
DDD7 016544
DDISP = 177570

DF1 = 070273
DF10 = 070273
DF100 = 070514
DF101 = 070514
DF102 = 070514
DF103 = 070514
DF104 = 070514
DF105 = 070514
DF106 = 070514
DF107 = 070514
DF11 = 070273
DF110 = 070514
DF111 = 070342
DF112 = 070342
DF113 = 070342
DF114 = 070342
DF115 = 070412
DF116 = 070412
DF117 = 070412
DF12 = 070273
DF120 = 070412
DF121 = 070412
DF122 = 070412
DF123 = 070366
DF124 = 070366
DF125 = 070366
DF126 = 070366
DF127 = 070436
DF13 = 070273
DF130 = 070436
DF131 = 070436
DF132 = 070366
DF133 = 070436
DF134 = 070436
DF135 = 070366
DF136 = 070436
DF137 = 070412
DF14 = 070273
DF140 = 070412
DF141 = 070412
DF142 = 070412
DF143 = 070412
DF144 = 070412
DF145 = 070342
DF146 = 070342
DF147 = 070342
DF15 = 070273
DF150 = 070342
DF151 = 070436
DF152 = 070436
DF153 = 070436
DF154 = 070436
DF155 = 070436
DF156 = 070436
DF157 = 070366
DF16 = 070273
DF160 = 070436

DF161 = 070366
DF162 = 070366
DF163 = 070436
DF164 = 070366
DF165 = 070546
DF166 = 070546
DF167 = 070546
DF17 = 070323
DF170 = 070546
DF171 = 070546
DF172 = 070546
DF173 = 070546
DF174 = 070546
DF175 = 070546
DF176 = 070600
DF177 = 070632
DF2 = 070311
DF20 = 070332
DF200 = 070600
DF201 = 070600
DF202 = 070600
DF203 = 070600
DF204 = 070600
DF205 = 070600
DF206 = 070600
DF207 = 070645
DF21 = 070352
DF210 = 070635
DF211 = 070645
DF212 = 070671
DF213 = 070671
DF214 = 070671
DF215 = 070671
DF216 = 070645
DF217 = 070721
DF22 = 070332
DF220 = 070645
DF221 = 070645
DF222 = 070645
DF223 = 070645
DF224 = 070645
DF225 = 070645
DF226 = 070671
DF227 = 070671
DF23 = 070342
DF230 = 070671
DF231 = 070671
DF232 = 070671
DF233 = 070730
DF234 = 070745
DF235 = 070754
DF236 = 070745
DF237 = 070745
DF24 = 070342
DF240 = 070745
DF241 = 070730
DF242 = 070745

DF243 = 070730
DF244 = 070730
DF245 = 070730
DF246 = 070366
DF247 = 070760
DF25 = 070342
DF250 = 070760
DF251 = 070760
DF252 = 070766
DF253 = 070772
DF254 = 070772
DF255 = 070772
DF256 = 070772
DF257 = 070772
DF26 = 070342
DF260 = 070772
DF261 = 070772
DF262 = 070772
DF263 = 070772
DF264 = 070772
DF265 = 070772
DF266 = 070766
DF267 = 070600
DF27 = 070342
DF3 = 070273
DF30 = 070342
DF31 = 070342
DF32 = 070342
DF33 = 070366
DF34 = 070366
DF35 = 070366
DF36 = 070366
DF37 = 070342
DF4 = 070273
DF40 = 070342
DF41 = 070342
DF42 = 070342
DF43 = 070342
DF44 = 070342
DF45 = 070342
DF46 = 070366
DF47 = 070366
DF5 = 070273
DF50 = 070366
DF51 = 070366
DF52 = 070366
DF53 = 070462
DF54 = 070462
DF55 = 070462
DF56 = 070462
DF57 = 070462
DF6 = 070273
DF60 = 070462
DF61 = 070462
DF62 = 070462
DF63 = 070462
DF64 = 070462

DF65 = 070462
DF66 = 070462
DF67 = 070462
DF7 = 070273
DF70 = 070514
DF71 = 070514
DF72 = 070514
DF73 = 070514
DF74 = 070514
DF75 = 070514
DF76 = 070514
DF77 = 070514
DH1 = 066436
DH10 = 066436
DH100 = 066436
DH101 = 066436
DH102 = 066436
DH103 = 066436
DH104 = 066436
DH105 = 066436
DH106 = 066436
DH107 = 066436
DH11 = 066436
DH110 = 066436
DH111 = 066436
DH112 = 066436
DH113 = 066436
DH114 = 066436
DH115 = 066777
DH116 = 066777
DH117 = 066777
DH12 = 066436
DH120 = 066777
DH121 = 066777
DH122 = 066777
DH123 = 066436
DH124 = 066436
DH125 = 066436
DH126 = 066436
DH127 = 066777
DH13 = 066436
DH130 = 066777
DH131 = 066777
DH132 = 066436
DH133 = 066777
DH134 = 066777
DH135 = 066436
DH136 = 066777
DH137 = 066777
DH14 = 066436
DH140 = 066777
DH141 = 066777
DH142 = 066777
DH143 = 066777
DH144 = 066777
DH145 = 066436
DH146 = 066436

DH147 = 066436
DH15 = 066436
DH150 = 066436
DH151 = 066777
DH152 = 066777
DH153 = 066777
DH154 = 066777
DH155 = 066777
DH156 = 066777
DH157 = 066436
DH16 = 066436
DH160 = 066777
DH161 = 066436
DH162 = 066436
DH163 = 066777
DH164 = 066436
DH165 = 066777
DH166 = 066777
DH167 = 066777
DH17 = 066567
DH170 = 066777
DH171 = 066777
DH172 = 066777
DH173 = 066777
DH174 = 066777
DH175 = 066777
DH176 = 066777
DH177 = 067076
DH2 = 066526
DH20 = 066436
DH200 = 066777
DH201 = 066777
DH202 = 066777
DH203 = 066777
DH204 = 066777
DH205 = 066777
DH206 = 066777
DH207 = 067177
DH21 = 066642
DH210 = 067267
DH211 = 067267
DH212 = 067177
DH213 = 067177
DH214 = 067177
DH215 = 067177
DH216 = 067267
DH217 = 067330
DH22 = 066731
DH220 = 067137
DH221 = 067267
DH222 = 067137
DH223 = 067267
DH224 = 067137
DH225 = 067267
DH226 = 067403
DH227 = 067472
DH23 = 066436

DH230 = 067403
DH231 = 067472
DH232 = 067472
DH233 = 067561
DH234 = 067622
DH235 = 067710
DH236 = 067750
DH237 = 067622
DH24 = 066436
DH240 = 070036
DH241 = 067561
DH242 = 070036
DH243 = 067561
DH244 = 067561
DH245 = 067561
DH246 = 066436
DH247 = 070126
DH25 = 066436
DH250 = 070173
DH251 = 070173
DH252 = 066526
DH253 = 066436
DH254 = 066436
DH255 = 066436
DH256 = 066436
DH257 = 066436
DH26 = 066436
DH260 = 066436
DH261 = 066436
DH262 = 066436
DH263 = 066436
DH264 = 066436
DH265 = 066436
DH266 = 070233
DH267 = 066777
DH27 = 066436
DH3 = 066436
DH30 = 066436
DH31 = 066436
DH32 = 066436
DH33 = 066436
DH34 = 066436
DH35 = 066436
DH36 = 066436
DH37 = 066436
DH4 = 066436
DH40 = 066436
DH41 = 066436
DH42 = 066436
DH43 = 066436
DH44 = 066436
DH45 = 066436
DH46 = 066436
DH47 = 066436
DH5 = 066436
DH50 = 066436
DH51 = 066436

DH52 = 066436
DH53 = 066436
DH54 = 066436
DH55 = 066436
DH56 = 066436
DH57 = 066436
DH6 = 066436
DH60 = 066436
DH61 = 066436
DH62 = 066436
DH63 = 066436
DH64 = 066436
DH65 = 066436
DH66 = 066436
DH67 = 066436
DH7 = 066436
DH70 = 066436
DH71 = 066436
DH72 = 066436
DH73 = 066436
DH74 = 066436
DH75 = 066436
DH76 = 066436
DH77 = 066436
DISPLA = 001142
DISPRE = 000174
DIVDSU = 016624
DIVDT = 017054
DIVFSU = 015704
DIVFT = 016120
DSWR = 177570
DT1 = 071024
DT10 = 071024
DT100 = 071264
DT101 = 071264
DT102 = 071264
DT103 = 071264
DT104 = 071264
DT105 = 071264
DT106 = 071264
DT107 = 071264
DT11 = 071024
DT110 = 071264
DT111 = 071212
DT112 = 071212
DT113 = 071212
DT114 = 071212
DT115 = 071352
DT116 = 071352
DT117 = 071352
DT12 = 071024
DT120 = 071352
DT121 = 071352
DT122 = 071352
DT123 = 071212
DT124 = 071212
DT125 = 071212

DT126 = 071212
 DT127 = 071352
 DT13 = 071024
 DT130 = 071352
 DT131 = 071352
 DT132 = 071212
 DT133 = 071352
 DT134 = 071352
 DT135 = 071212
 DT136 = 071352
 DT137 = 071352
 DT14 = 071024
 DT140 = 071352
 DT141 = 071352
 DT142 = 071352
 DT143 = 071352
 DT144 = 071352
 DT145 = 071212
 DT146 = 071212
 DT147 = 071212
 DT15 = 071024
 DT150 = 071212
 DT151 = 071352
 DT152 = 071352
 DT153 = 071352
 DT154 = 071352
 DT155 = 071352
 DT156 = 071352
 DT157 = 071212
 DT16 = 071024
 DT160 = 071352
 DT161 = 071212
 DT162 = 071212
 DT163 = 071352
 DT164 = 071212
 DT165 = 071426
 DT166 = 071426
 DT167 = 071426
 DT17 = 071110
 DT170 = 071426
 DT171 = 071426
 DT172 = 071426
 DT173 = 071426
 DT174 = 071426
 DT175 = 071426
 DT176 = 071426
 DT177 = 071514
 DT2 = 071062
 DT20 = 071130
 DT200 = 071426
 DT201 = 071426
 DT202 = 071426
 DT203 = 071426
 DT204 = 071426
 DT205 = 071426
 DT206 = 071426
 DT207 = 071524

DT21 = 071152
 DT210 = 071576
 DT211 = 071524
 DT212 = 071620
 DT213 = 071620
 DT214 = 071620
 DT215 = 071620
 DT216 = 071524
 DT217 = 071702
 DT22 = 071174
 DT220 = 071524
 DT221 = 071524
 DT222 = 071524
 DT223 = 071524
 DT224 = 071524
 DT225 = 071524
 DT226 = 071620
 DT227 = 071620
 DT23 = 071212
 DT230 = 071620
 DT231 = 071576
 DT232 = 071524
 DT233 = 071722
 DT234 = 071756
 DT235 = 071776
 DT236 = 071756
 DT237 = 071756
 DT24 = 071212
 DT240 = 071756
 DT241 = 071722
 DT242 = 071756
 DT243 = 071722
 DT244 = 071722
 DT245 = 071722
 DT246 = 071212
 DT247 = 072010
 DT25 = 071212
 DT250 = 072026
 DT251 = 072026
 DT252 = 072040
 DT253 = 072052
 DT254 = 072052
 DT255 = 072052
 DT256 = 072052
 DT257 = 072052
 DT26 = 071212
 DT260 = 072052
 DT261 = 072052
 DT262 = 072052
 DT263 = 072052
 DT264 = 072052
 DT265 = 072052
 DT266 = 072040
 DT267 = 071426
 DT27 = 071212
 DT3 = 071024
 DT30 = 071212

DT31 = 071212
 DT32 = 071212
 DT33 = 071212
 DT34 = 071212
 DT35 = 071212
 DT36 = 071212
 DT37 = 071212
 DT4 = 071024
 DT40 = 071212
 DT41 = 071212
 DT42 = 071212
 DT43 = 071212
 DT44 = 071212
 DT45 = 071212
 DT46 = 071212
 DT47 = 071212
 DT5 = 071024
 DT50 = 071212
 DT51 = 071212
 DT52 = 071212
 DT53 = 071264
 DT54 = 071264
 DT55 = 071264
 DT56 = 071264
 DT57 = 071264
 DT6 = 071024
 DT60 = 071264
 DT61 = 071264
 DT62 = 071264
 DT63 = 071264
 DT64 = 071264
 DT65 = 071264
 DT66 = 071264
 DT67 = 071264
 DT7 = 071024
 DT70 = 071264
 DT71 = 071264
 DT72 = 071264
 DT73 = 071264
 DT74 = 071264
 DT75 = 071264
 DT76 = 071264
 DT77 = 071264
 EEEDON = 020074
 EEE1 = 017070
 EEE10 = 017464
 EEE11 = 017520
 EEE12 = 017554
 EEE13 = 017610
 EEE2 = 017124
 EEE3 = 017160
 EEE4 = 017214
 EEE5 = 017250
 EEE6 = 017304
 EEE7 = 017340
 EEE8 = 017374
 EEE9 = 017430

EMTVEC = 000030
 EM1 = 040044
 EM10 = 040663
 EM100 = 054021
 EM101 = 054104
 EM102 = 054300
 EM103 = 054474
 EM104 = 054606
 EM105 = 054753
 EM106 = 055062
 EM107 = 055171
 EM11 = 040770
 EM110 = 055300
 EM111 = 044115
 EM112 = 044172
 EM113 = 044250
 EM114 = 044326
 EM115 = 044405
 EM116 = 044463
 EM117 = 044542
 EM12 = 041077
 EM120 = 044700
 EM121 = 045036
 EM122 = 045173
 EM123 = 045330
 EM124 = 045406
 EM125 = 045465
 EM126 = 045543
 EM127 = 045622
 EM13 = 041206
 EM130 = 045700
 EM131 = 045757
 EM132 = 046115
 EM133 = 046253
 EM134 = 046411
 EM135 = 046546
 EM136 = 046703
 EM137 = 047040
 EM14 = 041322
 EM140 = 047126
 EM141 = 044115
 EM142 = 044172
 EM143 = 047215
 EM144 = 047273
 EM145 = 047352
 EM146 = 047517
 EM147 = 047664
 EM15 = 041434
 EM150 = 050030
 EM151 = 050174
 EM152 = 050262
 EM153 = 045330
 EM154 = 045406
 EM155 = 050351
 EM156 = 050427
 EM157 = 050506
 EM16 = 041546

EM160 = 050653
 EM161 = 051011
 EM162 = 051156
 EM163 = 051322
 EM164 = 051457
 EM165 = 055410
 EM166 = 055474
 EM167 = 055557
 EM17 = 041643
 EM170 = 055611
 EM171 = 055677
 EM172 = 056022
 EM173 = 056107
 EM174 = 056255
 EM175 = 056423
 EM176 = 056535
 EM177 = 056621
 EM2 = 040076
 EM20 = 041720
 EM200 = 056655
 EM201 = 056707
 EM202 = 056776
 EM203 = 057140
 EM204 = 057302
 EM205 = 057370
 EM206 = 057536
 EM207 = 057704
 EM21 = 041752
 EM210 = 057746
 EM211 = 060010
 EM212 = 060154
 EM213 = 060340
 EM214 = 060524
 EM215 = 060710
 EM216 = 061074
 EM217 = 061256
 EM22 = 042004
 EM220 = 061320
 EM221 = 061550
 EM222 = 062014
 EM223 = 062245
 EM224 = 062512
 EM225 = 062743
 EM226 = 063210
 EM227 = 063343
 EM23 = 042065
 EM230 = 063476
 EM231 = 063632
 EM232 = 057746
 EM233 = 063766
 EM234 = 064025
 EM235 = 064111
 EM236 = 064157
 EM237 = 064212
 EM24 = 042142
 EM240 = 064276
 EM241 = 064332

EM242	064435	EM64	052722	GGG5	024474	HHHDON	030014	HXER11	013210
EM243	064471	EM65	053041	GGG6	024544	HHH1	025712	HXER12	013230
EM244	064574	EM66	053154	GGG7	024614	HHH10	027012	HXER13	013260
EM245	064633	EM67	053223	GGG8	024664	HHH11	027112	HXER2	012726
EM246	043525	EM7	040556	GGG9	024734	HHH12	027212	HXER22	012742
EM247	064715	EM70	053306	GGP1	011560	HHH13	027312	HXER3	012756
EM25	042240	EM71	053337	GGP2	011570	HHH2	026012	HXER33	012772
EM250	064751	EM72	053367	GGP3	011600	HHH3	026112	HXER4	013006
EM251	065003	EM73 =	053367	GGP4	011610	HHH4	026212	HXER5	012706
EM252	065036	EM74	053421	GGP5	011620	HHH5	026312	HXER6	013042
EM253	065127	EM75	053530	GGP6	011630	HHH6	026412	HXER66	013056
EM254	065213	EM76	053621	GGP7	011640	HHH7	026512	HXER7	013072
EM255	065300	EM77	053711	GGP8	011650	HHH8	026612	HXER8	013024
EM256	065366	ERM10	034314	GGP9	011660	HHH9	026712	HXER9	013122
EM257	065455	ERROR =	104000	GG1	006756	HHP0	006612	HXP1	013310
EM26	042325	ERRVEC =	000004	GG10	007252	HHP1	006622	HXP2	013320
EM260	065543	ERTYPE	036370	GG11	007310	HHP10	006732	HXP3	013330
EM261	065632	FFFDON	020624	GG12	007332	HHP11	006742	HXP4	013340
EM262	065722	FFF1	020100	GG13	007342	HHP2	006632	HXP5	013350
EM263	066013	FFF2	020154	GG14	007360	HHP3	006642	HXP6	013360
EM264	066100	FFF3	020230	GG15	007416	HHP4	006652	HXP7	013370
EM265	066165	FFF4	020304	GG16	007434	HHP5	006662	HXP8	013400
EM266	066252	FPSPUR	037000	GG17	007502	HHP6	006672	HX1	011674
EM267	066353	FPVECT =	000244	GG18	007512	HHP7	006702	HX10	012102
EM27 =	042240	GGDATO	011550	GG19	007546	HHP8	006712	HX11	012124
EM3	040132	GGDONE	011670	GG2	007014	HHP9	006722	HX12	012144
EM30	042423	GGERO	010046	GG20	007604	HHTRAP	006532	HX13	012154
EM31	042475	GGER1	010144	GG21	007626	HH1	005032	HX14	012162
EM32	042110	GGER10	010656	GG22	007636	HH10	005244	HX15	012200
EM33	042542	GGER11	010724	GG23	007654	HH11	005262	HX16	012226
EM34	042565	GGER12	010772	GG24	007712	HH12	005312	HX165	012232
EM35	042617	GGER13	011040	GG25	007730	HH13	005334	HX17	012260
EM36	042672	GGER14	011106	GG26	007776	HH14	005354	HX18	012316
EM37	042740	GGER15	011204	GG27	010006	HH15	005364	HX19	012336
EM4	040237	GGER16	011252	GG28	010042	HH16	005372	HX2	011724
EM40	042763	GGER17	011320	GG3	007036	HH17	005410	HX20	012346
EM41	043015	GGER18	011366	GG4	007046	HH18	005446	HX21	012350
EM42	043070	GGER19	011434	GG5	007064	HH19	005470	HX22	012400
EM43	043221	GGER2	010212	GG6	007122	HH2	005062	HX23	012420
EM44	043352	GGER20	011502	GG7	007140	HH20	005500	HX24	012440
EM45	043420	GGER3	010260	GG8	007206	HH21	005516	HX25	012450
EM46	043473	GGER4	010326	GG9	007216	HH22	005546	HX26	012456
EM47	043550	GGER5	010330	GTSWR =	104405	HH23	005570	HX27	012460
EM5	040344	GGER6	010376	HHDATO	006602	HH24	005600	HX28	012512
EM50	043704	GGER7	010444	HHDONE	006752	HH25	005616	HX29	012534
EM51	043757	GGER8	010542	HHERO	005622	HH3	005104	HX3	011740
EM52	044025	GGER9	010610	HHER00	005736	HH4	005124	HX30	012544
EM53	051655	GGGDON	025706	HHER1	005670	HH5	005134	HX31	012562
EM54	051706	GGG1	024234	HHER10	006464	HH6	005142	HX32	012612
EM55	051623	GGG10	025004	HHER2	006004	HH7	005154	HX33	012634
EM56	051736	GGG11	025054	HHER3	006052	HH8	005212	HX34	012644
EM57	052027	GGG12	025124	HHER4	006120	HH9	005234	HX35	012662
EM6	040451	GGG13	025174	HHER5	006166	HT =	000011	HX4	011760
EM60	052117	GGG14	025244	HHER6	006234	HXDATO	013300	HX5	012000
EM61	052221	GGG2	024304	HHER7	006302	HXDONE	013410	HX6	012010
EM62	052415	GGG3	024354	HHER8	006350	HXER1	012666	HX7	012016
EM63	052611	GGG4	024424	HHER9	006416	HXER10	013156	HX8	012034

HX9	012066	MS3	037400	SWREG	000176	\$APTHD	004332	\$ETABL	001336
IBSAVE	033762	MS37	037632	SWO =	000001	\$ATYC	035230	\$ETEND	001442
IIIDON	021464	MS4	037427	SW00 =	000001	\$ATY1	035204	\$FATAL	001320
III1	020630	MS40	037650	SW01 =	000002	\$ATY3	035212	\$FFLG	035450
III2	020674	MS41	037704	SW02 =	000004	\$ATY4	035222	\$FILLC	001156
III3	020740	MS415	037664	SW03 =	000010	\$AUTOB	001134	\$FILLS	001155
III4	021004	MS42	037730	SW04 =	000020	\$BASE	001372	\$GDADR	001120
IOTVEC=	000020	MS43	037746	SW05 =	000040	\$BDADR	001122	\$GDDAT	001124
JJJDON	022424	MS44	037763	SW06 =	000100	\$BDDAT	001126	\$GET42	033340
JJJ1	021470	MS5	037455	SW07 =	000200	\$BELL	001306	\$GTSWR	035522
JJJ2	021554	MS6	037477	SW08 =	000400	\$CDW1	001376	\$HD =	000003
JJJ3	021640	MS7	037515	SW09 =	001000	\$CDW2	001400	\$HIBTS	004332
JJJ4	021724	MULDSU	020364	SW1 =	000002	\$CHARC	034752	\$ICNT	001104
KKKDON	023266	MULDT	020614	SW10 =	002000	\$CKSWR	035452	\$ILLUP	036362
KKK1	022430	MULFSU	017650	SW11 =	004000	\$CLR.T	033356	\$INTAG	001135
KKK3	022474	MULFT	020064	SW12 =	010000	\$CMTAG	001100	\$ITEMB	001114
KKK4	022540	NNNDON	031770	SW13 =	020000	\$CM1 =	000024	\$LF	001314
KKK5	022604	NNN1	030746	SW14 =	040000	\$CM2 =	000050	\$LFLG	035447
LF =	000012	NNN2	031050	SW15 =	100000	\$CM3 =	000024	\$LOOP	033434
LLLON	024230	NNN3	031152	SW2 =	000004	\$CM4 =	000024	\$LPADR	001106
LLL1	023272	NNN4	031254	SW3 =	000010	\$CNTLG	036071	\$LPERR	001110
LLL2	023356	OVDNTT	022414	SW4 =	000020	\$CNTLU	036064	\$MADR1	001350
LLL3	023442	OVDTT	024220	SW5 =	000040	\$CPUOP	001344	\$MADR2	001354
LLL4	023526	OVFNTT	021454	SW6 =	000100	\$CRLF	001313	\$MADR3	001360
LOOP	005030	OVFTT	023256	SW7 =	000200	\$DDW0	001402	\$MADR4	001364
LPERR =	104413	OVUNDN	022014	SW8 =	000400	\$DDW1	001404	\$MAIL	001316
MMMDON	030742	OVUNDT	023616	SW9 =	001000	\$DDW10	001426	\$MAMS1	001346
MM1	030020	OVUNFN	021054	TAB =	000011	\$DDW11	001430	\$MAMS2	001352
MM2	030072	OVUNFT	022654	TBITVE=	000014	\$DDW12	001432	\$MAMS3	001356
MM3	030144	PIRQ =	177772	TKVEC =	000060	\$DDW13	001434	\$MAMS4	001362
MM4	030216	PIRQVE=	000240	TOCTNM	036764	\$DDW14	001436	\$MBADR	004334
MM5	030270	POWERM	037270	TPVEC =	000064	\$DDW15	001440	\$MFLG	035446
MNUMBE=	000267	PROGNUM=	000002	TRAPVE=	000034	\$DDW2	001406	\$MNEW	036107
MODDDO	031750	PRO =	000000	TRTVEC=	000014	\$DDW3	001410	\$MSGAD	001332
MODDD1	031760	PR1 =	000040	TST1	005030	\$DDW4	001412	\$MSGLG	001334
MODDOV	031362	PR2 =	000100	TST10	017066	\$DDW5	001414	\$MSGTY	001316
MODDSU	027416	PR3 =	000140	TST11	020076	\$DDW6	001416	\$MSWR	036076
MODDTO	027774	PR4 =	000200	TST12	020626	\$DDW7	001420	\$MTYP1	001347
MODDT1	030004	PR5 =	000240	TST13	021466	\$DDW8	001422	\$MTYP2	001353
MODFDO	030722	PR6 =	000300	TST14	022426	\$DDW9	001424	\$MTYP3	001357
MODFD1	030732	PR7 =	000340	TST15	023270	\$DEVCT	001326	\$MTYP4	001363
MODFOV	030346	PS =	177776	TST16	024232	\$DEVN	001374	\$MXCNT	033756
MODFSU	025320	PSW =	177776	TST17	025710	\$DOAGN	033376	\$NULL	001154
MODFTO	025656	PWRVEC=	000024	TST2	006754	\$ENDAD	033366	\$NWTST=	000001
MODFT1	025666	RDCHR =	104407	TST20	030016	\$ENDCT	033210	\$OCNT	035200
MODP1	025676	RESREG=	104411	TST21	030744	\$ENULL	033442	\$OMODE	035202
MSA1	037144	RESVEC=	000010	TST22	031772	\$ENV	001336	\$OVER	033742
MSA2	037162	RSETUP=	104412	TST23	033154	\$ENVM	001337	\$PASS	001324
MSA3	037205	R6 =	%000006	TST3	011672	\$EOP	033154	\$PASTM	004340
MSA4	037217	R7 =	%000007	TST4	013412	\$EOPCT	033202	\$PWRAD	036344
MSA5	037235	SAVREG=	104410	TST5	014420	\$ERFLG	001103	\$PWRDN	036204
MSA6	037250	SCOPE =	000004	TST6	015122	\$ERMAX	001115	\$PWRMG	036340
MS1	037342	SPACE	037337	TST7	016132	\$ERROR	033764	\$PWRUP	036256
MS10	037541	STACK =	001100	TYPE =	104401	\$ERRPC	001116	\$QUES	001312
MS11	037563	START	004346	TYPOC =	104402	\$ERRTB	001442	\$RDCHR	035734
MS12	037607	STKLMT=	177774	TYPON =	104404	\$ERTTL	001112	\$RDSZ =	000001
MS2	037360	SWR	001140	TYPOS =	104403	\$ESCAP	001304	\$REGAD	001160

\$REG0	001162	\$REG6	001176	\$TERM =	000030	\$TMP22	001276	\$TYPEC	034624
\$REG1	001164	\$REG7	001200	\$TESTN	001322	\$TMP23	001300	\$TYPEX	034754
\$REG10	001202	\$RESRE	034354	\$TIMES	001302	\$TMP3	001240	\$TYPOC	035002
\$REG11	001204	\$RTNAD	033436	\$TKB	001146	\$TMP4	001242	\$TYPON	035016
\$REG12	001206	\$RTRN	033432	\$TKS	001144	\$TMP5	001244	\$TYPOS	034756
\$REG13	001210	\$SAVRE	034316	\$TMP0	001232	\$TMP6	001246	\$UNIT	001330
\$REG14	001212	\$SAVR6	036366	\$TMP1	001234	\$TMP7	001250	\$UNITM	004342
\$REG15	001214	\$SCOPE	033446	\$TMP10	001252	\$TN =	000023	\$USWR	001342
\$REG16	001216	\$SETUP=	000137	\$TMP11	001254	\$TPB	001152	\$VECT1	001366
\$REG17	001220	\$STUP =	177777	\$TMP12	001256	\$TPFLG	001157	\$VECT2	001370
\$REG2	001166	\$SVLAD	033706	\$TMP13	001260	\$TPS	001150	\$XOFF =	000023
\$REG20	001222	\$SVPC =	004332	\$TMP14	001262	\$TRAP	036120	\$XON =	000021
\$REG21	001224	\$SWR =	177400	\$TMP15	001264	\$TRAP2	036142	\$XTSTR	033460
\$REG22	001226	\$SWREG	001340	\$TMP16	001266	\$TRP =	000014	\$GET4=	000001
\$REG23	001230	\$SWRMK=	000000	\$TMP17	001270	\$TRPAD	036154	\$OFILL	035201
\$REG3	001170	\$SWRMS=	000200	\$TMP2	001236	\$TSTM	004336	.LPER	037066
\$REG4	001172	\$TAB	037335	\$TMP20	001272	\$TSTM	001102	.RSET	037074
\$REG5	001174	\$TBIT	033440	\$TMP21	001274	\$TYPE	034412	.\$X =	004332

. ABS. 072150 000
000000 001

ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 59552 WORDS (233 PAGES)
DYNAMIC MEMORY: 20034 WORDS (77 PAGES)
ELAPSED TIME: 00:08:19
CKFPBB.BIN,CKFPBB/CR/-SP/NL:TOC=CKFPBB.MLB/ML,CKFPBB.P11

SYMBOL CROSS REFERENCE

SYMBOL	CROSS REFERENCE	VALUE	REFERENCES
AAADON		014416	15-2444 #15-2530
AAA1		013414	#14-2245
AAA10		014032	#14-2376
AAA11		014070	#15-2394
AAA12		014126	#15-2411
AAA13		014164	#15-2428
AAA2		013452	#14-2256
AAA3		013510	#14-2269
AAA4		013546	#14-2282
AAA5		013604	#14-2296
AAA6		013642	#14-2309
AAA7		013700	#14-2326
AAA8		013736	#14-2343
AAA9		013774	#14-2360
ABASE	=	000000	7-1314 7-1314
ACDW1	=	000000	7-1314 7-1314
ACDW2	=	000000	7-1314 7-1314
ACPUOP	=	000000	7-1314 7-1314
ACO	=	%000000	#6-1303 *9-1352 *9-1354 9-1357 *9-1386 *9-1388 9-1391 *9-1410 *9-1412
			9-1415 *9-1440 *9-1442 9-1445 *9-1462 *9-1464 9-1467 *10-1582 *10-1584
			10-1587 *10-1606 *10-1608 10-1611 10-1622 *10-1647 *10-1649 10-1652 *10-1671
			*10-1673 10-1676 10-1687 *10-1713 *10-1715 10-1718 *12-1739 *12-1741 12-1744
			12-1755 12-1786 12-1812 12-1838 *13-1910 *13-1913 13-1920 *13-1944 *13-1951
			13-1961 *13-1992 *13-2007 *13-2010 13-2013 *13-2027 *13-2029 13-2032 *13-2057
			*13-2058 *13-2062 13-2067 *13-2088 *13-2092 13-2097 *15-2477 15-2483 15-2511
			*16-2553 *16-2556 *16-2576 *16-2578 *16-2598 *16-2603 *17-2876 *17-2883 17-2890
			*19-3060 *19-3068 19-3075 *20-3311 *20-3318 20-3325 *21-3458 *21-3466 21-3473
			*22-3645 *22-3664 22-3671 23-3743 *24-3897 *24-3916 24-3923 24-3994 *25-4143
			*25-4162 25-4178 25-4256 *27-4395 *27-4414 27-4430 27-4508 *28-4797 *28-4806
			28-4812 *29-5165 *29-5174 29-5180 *30-5403 *30-5412 30-5419 *31-5622 *31-5631
			31-5638 33-5796 *33-5796 *33-5809 *33-5823 *33-5837 *33-5853 *33-5867 *33-5883
			*33-5896 *33-5911 *33-5925 *33-5939 *33-5982 33-6008
			*6-1304 *28-4799 28-4814 *29-5167 *30-5405 30-5421 *31-5624 31-5640
AC1	=	%000001	#6-1304
AC2	=	%000002	#6-1305
AC3	=	%000003	#6-1306
AC4	=	%000004	#6-1307
AC5	=	%000005	#6-1308
AC6	=	%000006	#6-1309
AC7	=	%000007	#6-1310
ADDW0	=	000000	7-1314 7-1314
ADDW1	=	000000	7-1314 7-1314
ADDW10	=	000000	7-1314 7-1314
ADDW11	=	000000	7-1314 7-1314
ADDW12	=	000000	7-1314 7-1314
ADDW13	=	000000	7-1314 7-1314
ADDW14	=	000000	7-1314 7-1314
ADDW15	=	000000	7-1314 7-1314
ADDW2	=	000000	7-1314 7-1314
ADDW3	=	000000	7-1314 7-1314
ADDW4	=	000000	7-1314 7-1314
ADDW5	=	000000	7-1314 7-1314
ADDW6	=	000000	7-1314 7-1314

SYMBOL CROSS REFERENCE		REFERENCES			
SYMBOL	VALUE				
ADDW7	= 000000	7-1314	7-1314		
ADDW8	= 000000	7-1314	7-1314		
ADDW9	= 000000	7-1314	7-1314		
ADEVCT	= 000000	7-1314	7-1314		
ADEVM	= 000000	7-1314	7-1314		
AENV	= 000000	7-1314	7-1314		
AENVM	= 000000	7-1314	7-1314		
AFATAL	= 000000	7-1314	7-1314		
AMADR1	= 000000	7-1314	7-1314		
AMADR2	= 000000	7-1314	7-1314		
AMADR3	= 000000	7-1314	7-1314		
AMADR4	= 000000	7-1314	7-1314		
AMAMS1	= 000000	7-1314	7-1314		
AMAMS2	= 000000	7-1314	7-1314		
AMAMS3	= 000000	7-1314	7-1314		
AMAMS4	= 000000	7-1314	7-1314		
AMSGAD	= 000000	7-1314	7-1314		
AMSLG	= 000000	7-1314	7-1314		
AMSGTY	= 000000	7-1314	7-1314		
AMTYP1	= 000000	7-1314	7-1314		
AMTYP2	= 000000	7-1314	7-1314		
AMTYP3	= 000000	7-1314	7-1314		
AMTYP4	= 000000	7-1314	7-1314		
APASS	= 000000	7-1314	7-1314		
APRIOR	= 000000	7-1314			
APTCSU	= 000040	33-6074	#33-6078		
APTENV	= 000001	33-6070	33-6074	33-6078	#33-6078
APTSIZ	= 000200	8-1327	#33-6078		
APTSP0	= 000100	33-6074	33-6078	#33-6078	
ASWREG	= 000000	7-1314	7-1314		
ATESTN	= 000000	7-1314	7-1314		
AUNIT	= 000000	7-1314	7-1314		
AUSWR	= 000000	7-1314	7-1314		
AVECT1	= 000000	7-1314	7-1314		
AVECT2	= 000000	7-1314	7-1314		
BBBDON	015120	16-2627	16-2648	16-2656	16-2664 16-2677 #16-2684
BBBER1	014674	16-2550	#16-2631		
BBBER2	014760	16-2563	16-2585	16-2621	#16-2651
BBBER3	015006	16-2567	16-2589	16-2625	#16-2659
BBBER4	015034	16-2606	#16-2668		
BBBP1	015100	16-2552	16-2554	16-2577	16-2601 #16-2679
BBBP2	015110	16-2575	16-2597	#16-2680	
BBB0	014424	#16-2546			
BBB1	014460	16-2551	#16-2556		
BBB2	014506	#16-2570			
BBB3	014536	16-2574	#16-2578		
BBB4	014564	#16-2592			
BBB5	014622	16-2596	#16-2603	16-2608	
BBB6	014630	16-2600	#16-2608		
BIT0	= 000001	#6-1301			
BIT00	= 000001	#6-1301	6-1301	33-6068	33-6068 33-6070 33-6070
BIT01	= 000002	#6-1301	6-1301		

SYMBOL	CROSS REFERENCE VALUE	REFERENCES	REFERENCES	REFERENCES	REFERENCES	REFERENCES	REFERENCES	REFERENCES	REFERENCES
CORDON	033152	33-5773	33-5782	33-5945	33-6041	#33-6058			
CORFLG	033134	*33-5798	*33-5811	*33-5825	*33-5839	*33-5855	*33-5869	*33-5884	*33-5897
		*33-5927	*33-5941	*33-5976	*33-5992	*33-5999	*33-6038	#33-6043	
CORINT	033146	33-5776	33-5994	33-6001	#33-6046				
CORMES	040005	33-5789	#34-6299						
CORSUB	032652	33-5792	33-5805	33-5819	33-5833	33-5849	33-5863	33-5879	33-5892
		33-5921	33-5935	#33-5976					33-5907
		33-6007	33-6009	33-6026	#33-6044				
CORTMP	033136	33-5991	#33-6051						
CORTRP	033150	33-5991	#33-5999						
CORTV	032750	33-6021	33-6024	33-6029	#33-6033				
CORTV0	033112	33-6000	#33-6035						
CORTV1	033116	33-5769	#33-5775						
COR1	032020	#33-5891							
COR10	032442	#33-5906							
COR11	032506	#33-5920							
COR12	032546	#33-5934							
COR13	032606	33-5775	#33-5780						
COR2	032044	33-5772	33-5778	#33-5785					
COR3	032060	33-5787	#33-5790						
COR33	032072	#33-5804							
COR4	032132	#33-5818							
COR5	032172	#33-5832							
COR6	032232	#33-5848							
COR7	032276	#33-5862							
COR8	032336	#33-5878							
COR9	032402	*33-6068	33-6068	#33-6068	*33-6070	33-6070	38-7348		
CPSAVE	033760	13-1996	13-2008	13-2161	33-5777	33-5781	#34-6218	34-6262	
CPSPUR	037032	#34-6229	34-6263						
CPTWO	037050	#6-1301	33-6074	33-6074					
CR	= 000015	#6-1299	#6-1301	8-1328	8-1328	33-6074	33-6074	34-6279	34-6279
CRLF	= 000200	34-6299	35-6306	35-6308	35-6310	35-6312	35-6314	35-6316	35-6318
		35-6322	35-6324	35-6326	35-6328	35-6355	35-6357	35-6364	35-6366
		35-6369	35-6370	35-6372	35-6374	35-6381	35-6383	35-6384	35-6386
		35-6392	35-6394	35-6414	35-6416	35-6428	35-6430	35-6435	35-6437
		35-6446	35-6466	35-6468	35-6478	35-6480	35-6485	35-6487	35-6499
		35-6503	35-6505	35-6507	35-6508	35-6510	35-6511	35-6513	35-6515
		35-6519	35-6531	35-6533	35-6535	35-6537	35-6539	35-6541	35-6543
		35-6546	35-6547	35-6549	35-6551	35-6552	35-6554	35-6556	35-6558
		35-6585	35-6587	35-6589	35-6596	35-6598	35-6600	35-6602	35-6604
		35-6704	35-6704	35-6730	35-6730	35-6730	35-6731	35-6731	35-6731
		35-6732	35-6732	35-6733	35-6733	35-6733	35-6735	35-6735	35-6735
		35-6739	35-6739	35-6739	35-6739	35-6740	35-6740	35-6741	35-6741
		35-6741	35-6742	35-6742	35-6743	35-6743	35-6743	35-6743	35-6744
		35-6745	35-6745	35-6746	35-6746	35-6747	35-6747	35-6779	35-6782
		35-6786	35-6790						
DDDDON	017064	18-3025	#19-3121						
DDD1	016134	#18-2944							
DDD2	016210	#18-2955							
DDD3	016264	#18-2966							
DDD4	016340	#18-2977							
DDD5	016414	#18-2988							

SYMBOL	CROSS REFERENCE	REFERENCES								
SYMBOL	VALUE									
DDD6	016470	#18-3001								
DDD7	016544	#18-3013								
DDISP	= 177570	#6-1301	7-1314	8-1327						
DF1	070273	8-1320	#37-7025	37-7027	37-7028	37-7029	37-7030	37-7031	37-7032	37-7033
		37-7034	37-7035	37-7036	37-7037	37-7038				
DF10	= 070273	8-1320	#37-7032							
DF100	= 070514	8-1320	#37-7133							
DF101	= 070514	8-1320	#37-7134							
DF102	= 070514	8-1320	#37-7135							
DF103	= 070514	8-1320	#37-7136							
DF104	= 070514	8-1320	#37-7137							
DF105	= 070514	8-1320	#37-7138							
DF106	= 070514	8-1320	#37-7139							
DF107	= 070514	8-1320	#37-7140							
DF11	= 070273	8-1320	#37-7033							
DF110	= 070514	8-1320	#37-7141							
DF111	= 070342	8-1320	#37-7068							
DF112	= 070342	8-1320	#37-7069							
DF113	= 070342	8-1320	#37-7070							
DF114	= 070342	8-1320	#37-7071							
DF115	070412	8-1320	#37-7072	37-7073	37-7074	37-7075	37-7076	37-7077	37-7090	37-7091
		37-7092	37-7093	37-7094	37-7095					
DF116	= 070412	8-1320	#37-7073							
DF117	= 070412	8-1320	#37-7074							
DF12	= 070273	8-1320	#37-7034							
DF120	= 070412	8-1320	#37-7075							
DF121	= 070412	8-1320	#37-7076							
DF122	= 070412	8-1320	#37-7077							
DF123	= 070366	8-1320	#37-7078							
DF124	= 070366	8-1320	#37-7079							
DF125	= 070366	8-1320	#37-7080							
DF126	= 070366	8-1320	#37-7081							
DF127	070436	8-1320	#37-7082	37-7083	37-7084	37-7086	37-7087	37-7089	37-7100	37-7101
		37-7102	37-7103	37-7104	37-7105	37-7107	37-7110			
DF13	= 070273	8-1320	#37-7035							
DF130	= 070436	8-1320	#37-7083							
DF131	= 070436	8-1320	#37-7084							
DF132	= 070366	8-1320	#37-7085							
DF133	= 070436	8-1320	#37-7086							
DF134	= 070436	8-1320	#37-7087							
DF135	= 070366	8-1320	#37-7088							
DF136	= 070436	8-1320	#37-7089							
DF137	= 070412	8-1320	#37-7090							
DF14	= 070273	8-1320	#37-7036							
DF140	= 070412	8-1320	#37-7091							
DF141	= 070412	8-1320	#37-7092							
DF142	= 070412	8-1320	#37-7093							
DF143	= 070412	8-1320	#37-7094							
DF144	= 070412	8-1320	#37-7095							
DF145	= 070342	8-1320	#37-7096							
DF146	= 070342	8-1320	#37-7097							
DF147	= 070342	8-1320	#37-7098							

SYMBOL	CROSS REFERENCE	REFERENCES								
SYMBOL	VALUE									
DF221	= 070645	8-1320	#37-7170							
DF222	= 070645	8-1320	#37-7171							
DF223	= 070645	8-1320	#37-7172							
DF224	= 070645	8-1320	#37-7173							
DF225	= 070645	8-1320	#37-7174							
DF226	= 070671	8-1320	#37-7175							
DF227	= 070671	8-1320	#37-7176							
DF23	070342	8-1320	#37-7043	37-7044	37-7045	37-7046	37-7047	37-7048	37-7049	37-7050
		37-7055	37-7056	37-7057	37-7058	37-7059	37-7060	37-7061	37-7068	37-7069
		37-7070	37-7071	37-7096	37-7097	37-7098	37-7099			
DF230	= 070671	8-1320	#37-7177							
DF231	= 070671	8-1320	#37-7178							
DF232	= 070671	8-1320	#37-7179							
DF233	070730	8-1320	#37-7180	37-7187	37-7189	37-7190	37-7191			
DF234	070745	8-1320	#37-7181	37-7184	37-7185	37-7186	37-7188			
DF235	070754	8-1320	#37-7183							
DF236	= 070745	8-1320	#37-7184							
DF237	= 070745	8-1320	#37-7185							
DF24	= 070342	8-1320	#37-7044							
DF240	= 070745	8-1320	#37-7186							
DF241	= 070730	8-1320	#37-7187							
DF242	= 070745	8-1320	#37-7188							
DF243	= 070730	8-1320	#37-7189							
DF244	= 070730	8-1320	#37-7190							
DF245	= 070730	8-1320	#37-7191							
DF246	= 070366	8-1320	#37-7062							
DF247	070760	8-1320	#37-7192	37-7193	37-7194					
DF25	= 070342	8-1320	#37-7046							
DF250	= 070760	8-1320	#37-7193							
DF251	= 070760	8-1320	#37-7194							
DF252	070766	8-1320	#37-7196	37-7208						
DF253	070772	8-1320	#37-7197	37-7198	37-7199	37-7200	37-7201	37-7202	37-7203	37-7204
		37-7205	37-7206	37-7207						
DF254	= 070772	8-1320	#37-7198							
DF255	= 070772	8-1320	#37-7199							
DF256	= 070772	8-1320	#37-7200							
DF257	= 070772	8-1320	#37-7201							
DF26	= 070342	8-1320	#37-7047							
DF260	= 070772	8-1320	#37-7202							
DF261	= 070772	8-1320	#37-7203							
DF262	= 070772	8-1320	#37-7204							
DF263	= 070772	8-1320	#37-7205							
DF264	= 070772	8-1320	#37-7206							
DF265	= 070772	8-1320	#37-7207							
DF266	= 070766	8-1320	#37-7208							
DF267	= 070600	8-1320	#37-7209							
DF27	= 070342	8-1320	#37-7048							
DF3	= 070273	8-1320	#37-7027							
DF30	= 070342	8-1320	#37-7049							
DF31	= 070342	8-1320	#37-7050							
DF32	= 070342	8-1320	#37-7045							
DF33	070366	8-1320	#37-7051	37-7052	37-7053	37-7054	37-7062	37-7063	37-7064	37-7065

SYMBOL	CROSS REFERENCE VALUE	REFERENCES	37-7066	37-7067	37-7070	37-7079	37-7080	37-7081	37-7085	37-7088	37-7106
DF 34	= 070366	8-1320	37-7066	37-7067	37-7070	37-7079	37-7080	37-7081	37-7085	37-7088	37-7106
DF 35	= 070366	8-1320	37-7108	37-7109	37-7111						
DF 36	= 070366	8-1320	#37-7052								
DF 37	= 070342	8-1320	#37-7053								
DF 4	= 070273	8-1320	#37-7054								
DF 40	= 070342	8-1320	#37-7055								
DF 41	= 070342	8-1320	#37-7028								
DF 42	= 070342	8-1320	#37-7056								
DF 43	= 070342	8-1320	#37-7057								
DF 44	= 070342	8-1320	#37-7058								
DF 45	= 070342	8-1320	#37-7059								
DF 46	= 070366	8-1320	#37-7060								
DF 47	= 070366	8-1320	#37-7061								
DF 5	= 070273	8-1320	#37-7063								
DF 50	= 070366	8-1320	#37-7064								
DF 51	= 070366	8-1320	#37-7029								
DF 52	= 070366	8-1320	#37-7065								
DF 53	070462	8-1320	#37-7066								
		37-7120	#37-7067								
DF 54	= 070462	8-1320	#37-7112	37-7113	37-7114	37-7115	37-7116	37-7117	37-7118	37-7119	
DF 55	= 070462	8-1320	37-7121	37-7122	37-7123	37-7124					
DF 56	= 070462	8-1320	#37-7113								
DF 57	= 070462	8-1320	#37-7114								
DF 6	= 070273	8-1320	#37-7115								
DF 60	= 070462	8-1320	#37-7116								
DF 61	= 070462	8-1320	#37-7030								
DF 62	= 070462	8-1320	#37-7117								
DF 63	= 070462	8-1320	#37-7118								
DF 64	= 070462	8-1320	#37-7119								
DF 65	= 070462	8-1320	#37-7120								
DF 66	= 070462	8-1320	#37-7121								
DF 67	= 070462	8-1320	#37-7122								
DF 7	= 070273	8-1320	#37-7123								
DF 70	070514	8-1320	#37-7124								
		37-7133	#37-7031								
DF 71	= 070514	8-1320	#37-7125	37-7126	37-7127	37-7128	37-7129	37-7130	37-7131	37-7132	
DF 72	= 070514	8-1320	37-7134	37-7135	37-7136	37-7137	37-7138	37-7139	37-7140	37-7141	
DF 73	= 070514	8-1320	#37-7126								
DF 74	= 070514	8-1320	#37-7127								
DF 75	= 070514	8-1320	#37-7128								
DF 76	= 070514	8-1320	#37-7129								
DF 77	= 070514	8-1320	#37-7130								
DH1	066436	8-1320	#37-7131								
		36-6819	#37-7132								
		36-6829	#36-6819	36-6822	36-6823	36-6824	36-6825	36-6826	36-6827	36-6828	
		36-6842	36-6830	36-6831	36-6832	36-6833	36-6835	36-6839	36-6840	36-6841	
		36-6851	36-6843	36-6844	36-6845	36-6846	36-6847	36-6848	36-6849	36-6850	
		36-6860	36-6852	36-6853	36-6854	36-6855	36-6856	36-6857	36-6858	36-6859	
		36-6876	36-6861	36-6862	36-6863	36-6864	36-6865	36-6866	36-6867	36-6875	
		36-6903	36-6877	36-6878	36-6882	36-6885	36-6893	36-6894	36-6895	36-6896	
		36-6914	36-6905	36-6906	36-6908	36-6909	36-6910	36-6911	36-6912	36-6913	
			36-6915	36-6916	36-6917	36-6918	36-6919	36-6920	36-6921	36-6922	

SYMBOL CROSS REFERENCE		REFERENCES		CREF V01						
SYMBOL	VALUE									
		36-6923	36-6924	36-6925	36-6926	36-6927	36-6928	36-6929	36-6930	36-6931
		36-6932	36-6933	36-6934	36-6935	36-6936	36-6937	36-6938	36-7011	36-7012
		36-7013	36-7014	36-7015	36-7016	36-7017	36-7018	36-7019	36-7020	36-7021
DH10	= 066436	8-1320	#36-6827							
DH100	= 066436	8-1320	#36-6930							
DH101	= 066436	8-1320	#36-6931							
DH102	= 066436	8-1320	#36-6932							
DH103	= 066436	8-1320	#36-6933							
DH104	= 066436	8-1320	#36-6934							
DH105	= 066436	8-1320	#36-6935							
DH106	= 066436	8-1320	#36-6936							
DH107	= 066436	8-1320	#36-6937							
DH11	= 066436	8-1320	#36-6828							
DH110	= 066436	8-1320	#36-6938							
DH111	= 066436	8-1320	#36-6864							
DH112	= 066436	8-1320	#36-6865							
DH113	= 066436	8-1320	#36-6866							
DH114	= 066436	8-1320	#36-6867							
DH115	= 066777	8-1320	#36-6868	36-6870	36-6871	36-6872	36-6873	36-6874	36-6879	36-6880
		36-6881	36-6883	36-6884	36-6886	36-6887	36-6888	36-6889	36-6890	36-6891
		36-6892	36-6897	36-6898	36-6899	36-6900	36-6901	36-6902	36-6904	36-6907
		36-6939	36-6940	36-6941	36-6942	36-6943	36-6944	36-6945	36-6946	36-6947
		36-6948	36-6950	36-6951	36-6952	36-6953	36-6954	36-6955	36-6956	36-7023
DH116	= 066777	8-1320	#36-6870							
DH117	= 066777	8-1320	#36-6871							
DH12	= 066436	8-1320	#36-6829							
DH120	= 066777	8-1320	#36-6872							
DH121	= 066777	8-1320	#36-6873							
DH122	= 066777	8-1320	#36-6874							
DH123	= 066436	8-1320	#36-6875							
DH124	= 066436	8-1320	#36-6876							
DH125	= 066436	8-1320	#36-6877							
DH126	= 066436	8-1320	#36-6878							
DH127	= 066777	8-1320	#36-6879							
DH13	= 066436	8-1320	#36-6830							
DH130	= 066777	8-1320	#36-6880							
DH131	= 066777	8-1320	#36-6881							
DH132	= 066436	8-1320	#36-6882							
DH133	= 066777	8-1320	#36-6883							
DH134	= 066777	8-1320	#36-6884							
DH135	= 066436	8-1320	#36-6885							
DH136	= 066777	8-1320	#36-6886							
DH137	= 066777	8-1320	#36-6887							
DH14	= 066436	8-1320	#36-6831							
DH140	= 066777	8-1320	#36-6888							
DH141	= 066777	8-1320	#36-6889							
DH142	= 066777	8-1320	#36-6890							
DH143	= 066777	8-1320	#36-6891							
DH144	= 066777	8-1320	#36-6892							
DH145	= 066436	8-1320	#36-6893							
DH146	= 066436	8-1320	#36-6894							
DH147	= 066436	8-1320	#36-6895							

SYMBOL	CROSS REFERENCE	VALUE	REFERENCES				
DH15	=	066436	8-1320	#36-6832			
DH150	=	066436	8-1320	#36-6896			
DH151	=	066777	8-1320	#36-6897			
DH152	=	066777	8-1320	#36-6898			
DH153	=	066777	8-1320	#36-6899			
DH154	=	066777	8-1320	#36-6900			
DH155	=	066777	8-1320	#36-6901			
DH156	=	066777	8-1320	#36-6902			
DH157	=	066436	8-1320	#36-6903			
DH16	=	066436	8-1320	#36-6833			
DH160	=	066777	8-1320	#36-6904			
DH161	=	066436	8-1320	#36-6905			
DH162	=	066436	8-1320	#36-6906			
DH163	=	066777	8-1320	#36-6907			
DH164	=	066436	8-1320	#36-6908			
DH165	=	066777	8-1320	#36-6939			
DH166	=	066777	8-1320	#36-6940			
DH167	=	066777	8-1320	#36-6941			
DH17		066567	8-1320	#36-6834			
DH170	=	066777	8-1320	#36-6942			
DH171	=	066777	8-1320	#36-6943			
DH172	=	066777	8-1320	#36-6944			
DH173	=	066777	8-1320	#36-6945			
DH174	=	066777	8-1320	#36-6946			
DH175	=	066777	8-1320	#36-6947			
DH176	=	066777	8-1320	#36-6948			
DH177		067076	8-1320	#36-6949			
DH2		066526	8-1320	#36-6821	36-7010		
DH20	=	066436	8-1320	#36-6835			
DH200	=	066777	8-1320	#36-6950			
DH201	=	066777	8-1320	#36-6951			
DH202	=	066777	8-1320	#36-6952			
DH203	=	066777	8-1320	#36-6953			
DH204	=	066777	8-1320	#36-6954			
DH205	=	066777	8-1320	#36-6955			
DH206	=	066777	8-1320	#36-6956			
DH207		067177	8-1320	#36-6958	36-6962	36-6963	36-6964 36-6965
DH21		066642	8-1320	#36-6836			
DH210		067267	8-1320	#36-6960	36-6961	36-6966	36-6969 36-6971 36-6973
DH211	=	067267	8-1320	#36-6961			
DH212	=	067177	8-1320	#36-6962			
DH213	=	067177	8-1320	#36-6963			
DH214	=	067177	8-1320	#36-6964			
DH215	=	067177	8-1320	#36-6965			
DH216	=	067267	8-1320	#36-6966			
DH217		067330	8-1320	#36-6967			
DH22		066731	8-1320	#36-6838			
DH220		067137	8-1320	#36-6957	36-6970	36-6972	
DH221	=	067267	8-1320	#36-6969			
DH222	=	067137	8-1320	#36-6970			
DH223	=	067267	8-1320	#36-6971			
DH224	=	067137	8-1320	#36-6972			

SYMBOL CROSS REFERENCE		REFERENCES					
SYMBOL	VALUE						
DH225	= 067267	8-1320	#36-6973				
DH226	= 067403	8-1320	#36-6974	36-6978			
DH227	= 067472	8-1320	#36-6976	36-6979	36-6980		
DH23	= 066436	8-1320	#36-6839				
DH230	= 067403	8-1320	#36-6978				
DH231	= 067472	8-1320	#36-6979				
DH232	= 067472	8-1320	#36-6980				
DH233	= 067561	8-1320	#36-6983	36-7001	36-7003	36-7004	36-7005
DH234	= 067622	8-1320	#36-6985	36-6995			
DH235	= 067710	8-1320	#36-6988				
DH236	= 067750	8-1320	#36-6991				
DH237	= 067622	8-1320	#36-6995				
DH24	= 066436	8-1320	#36-6841				
DH240	= 070036	8-1320	#36-6997	36-7002			
DH241	= 067561	8-1320	#36-7001				
DH242	= 070036	8-1320	#36-7002				
DH243	= 067561	8-1320	#36-7003				
DH244	= 067561	8-1320	#36-7004				
DH245	= 067561	8-1320	#36-7005				
DH246	= 066436	8-1320	#36-6858				
DH247	= 070126	8-1320	#36-7006				
DH25	= 066436	8-1320	#36-6842				
DH250	= 070173	8-1320	#36-7007	36-7008			
DH251	= 070173	8-1320	#36-7008				
DH252	= 066526	8-1320	#36-7010				
DH253	= 066436	8-1320	#36-7011				
DH254	= 066436	8-1320	#36-7012				
DH255	= 066436	8-1320	#36-7013				
DH256	= 066436	8-1320	#36-7014				
DH257	= 066436	8-1320	#36-7015				
DH26	= 066436	8-1320	#36-6843				
DH260	= 066436	8-1320	#36-7016				
DH261	= 066436	8-1320	#36-7017				
DH262	= 066436	8-1320	#36-7018				
DH263	= 066436	8-1320	#36-7019				
DH264	= 066436	8-1320	#36-7020				
DH265	= 066436	8-1320	#36-7021				
DH266	= 070233	8-1320	#36-7022				
DH267	= 066777	8-1320	#36-7023				
DH27	= 066436	8-1320	#36-6844				
DH3	= 066436	8-1320	#36-6822				
DH30	= 066436	8-1320	#36-6845				
DH31	= 066436	8-1320	#36-6846				
DH32	= 066436	8-1320	#36-6840				
DH33	= 066436	8-1320	#36-6847				
DH34	= 066436	8-1320	#36-6848				
DH35	= 066436	8-1320	#36-6849				
DH36	= 066436	8-1320	#36-6850				
DH37	= 066436	8-1320	#36-6851				
DH4	= 066436	8-1320	#36-6823				
DH40	= 066436	8-1320	#36-6852				
DH41	= 066436	8-1320	#36-6853				

SYMBOL CROSS REFERENCE

SYMBOL	CROSS REFERENCE VALUE	REFERENCES							
DH42	= 066436	8-1320	#36-6854						
DH43	= 066436	8-1320	#36-6855						
DH44	= 066436	8-1320	#36-6856						
DH45	= 066436	8-1320	#36-6857						
DH46	= 066436	8-1320	#36-6859						
DH47	= 066436	8-1320	#36-6860						
DH5	= 066436	8-1320	#36-6824						
DH50	= 066436	8-1320	#36-6861						
DH51	= 066436	8-1320	#36-6862						
DH52	= 066436	8-1320	#36-6863						
DH53	= 066436	8-1320	#36-6909						
DH54	= 066436	8-1320	#36-6910						
DH55	= 066436	8-1320	#36-6911						
DH56	= 066436	8-1320	#36-6912						
DH57	= 066436	8-1320	#36-6913						
DH6	= 066436	8-1320	#36-6825						
DH60	= 066436	8-1320	#36-6914						
DH61	= 066436	8-1320	#36-6915						
DH62	= 066436	8-1320	#36-6916						
DH63	= 066436	8-1320	#36-6917						
DH64	= 066436	8-1320	#36-6918						
DH65	= 066436	8-1320	#36-6919						
DH66	= 066436	8-1320	#36-6920						
DH67	= 066436	8-1320	#36-6921						
DH7	= 066436	8-1320	#36-6826						
DH70	= 066436	8-1320	#36-6922						
DH71	= 066436	8-1320	#36-6923						
DH72	= 066436	8-1320	#36-6924						
DH73	= 066436	8-1320	#36-6925						
DH74	= 066436	8-1320	#36-6926						
DH75	= 066436	8-1320	#36-6927						
DH76	= 066436	8-1320	#36-6928						
DH77	= 066436	8-1320	#36-6929						
DISPLA	001142	#7-1314	*8-1327	*8-1327	33-6068	33-6070			
DISPRE	000174	#6-1313	8-1327						
DIVDSU	016624	18-2945	18-2956	18-2967	18-2978	18-2989	18-3002	18-3014	#19-3055
DIVDT	017054	19-3074	19-3083	19-3089	19-3101	#19-3119			
DIVFSU	015704	17-2699	17-2710	17-2721	17-2733	17-2744	17-2755	17-2766	17-2778 17-2790
		17-2801	17-2812	17-2824	17-2835	#17-2872			
DIVFT	016120	17-2889	17-2898	#17-2929					
DSWR	= 177570	#6-1301	7-1314	8-1327					
DT1	071024	8-1320	#38-7212	38-7215	38-7216	38-7217	38-7218	38-7219	38-7220 38-7221
		38-7222	38-7223	38-7224	38-7225	38-7226			
DT10	= 071024	8-1320	#38-7220						
DT100	= 071264	8-1320	#38-7280						
DT101	= 071264	8-1320	#38-7281						
DT102	= 071264	8-1320	#38-7282						
DT103	= 071264	8-1320	#38-7283						
DT104	= 071264	8-1320	#38-7284						
DT105	= 071264	8-1320	#38-7285						
DT106	= 071264	8-1320	#38-7286						
DT107	= 071264	8-1320	#38-7287						

CKFPBB

CREATED BY MACRO ON 28-MAR-81 AT 14:50

PAGE 14

SEQUENCE 187

CREF V01

SYMBOL CROSS REFERENCE

SYMBOL	VALUE	REFERENCES								
DT11	= 071024	8-1320	#38-7221							
DT110	= 071264	8-1320	#38-7288							
DT111	= 071212	8-1320	#38-7289							
DT112	= 071212	8-1320	#38-7290							
DT113	= 071212	8-1320	#38-7291							
DT114	= 071212	8-1320	#38-7292							
DT115	= 071352	8-1320	#38-7293	38-7296	38-7297	38-7298	38-7299	38-7300	38-7305	38-7306
		38-7307	38-7309	38-7310	38-7312	38-7313	38-7314	38-7315	38-7316	38-7317
		38-7318	38-7323	38-7324	38-7325	38-7326	38-7327	38-7328	38-7330	38-7333
DT116	= 071352	8-1320	#38-7296							
DT117	= 071352	8-1320	#38-7297							
DT12	= 071024	8-1320	#38-7222							
DT120	= 071352	8-1320	#38-7298							
DT121	= 071352	8-1320	#38-7299							
DT122	= 071352	8-1320	#38-7300							
DT123	= 071212	8-1320	#38-7301							
DT124	= 071212	8-1320	#38-7302							
DT125	= 071212	8-1320	#38-7303							
DT126	= 071212	8-1320	#38-7304							
DT127	= 071352	8-1320	#38-7305							
DT13	= 071024	8-1320	#38-7223							
DT130	= 071352	8-1320	#38-7306							
DT131	= 071352	8-1320	#38-7307							
DT132	= 071212	8-1320	#38-7308							
DT133	= 071352	8-1320	#38-7309							
DT134	= 071352	8-1320	#38-7310							
DT135	= 071212	8-1320	#38-7311							
DT136	= 071352	8-1320	#38-7312							
DT137	= 071352	8-1320	#38-7313							
DT14	= 071024	8-1320	#38-7224							
DT140	= 071352	8-1320	#38-7314							
DT141	= 071352	8-1320	#38-7315							
DT142	= 071352	8-1320	#38-7316							
DT143	= 071352	8-1320	#38-7317							
DT144	= 071352	8-1320	#38-7318							
DT145	= 071212	8-1320	#38-7319							
DT146	= 071212	8-1320	#38-7320							
DT147	= 071212	8-1320	#38-7321							
DT15	= 071024	8-1320	#38-7225							
DT150	= 071212	8-1320	#38-7322							
DT151	= 071352	8-1320	#38-7323							
DT152	= 071352	8-1320	#38-7324							
DT153	= 071352	8-1320	#38-7325							
DT154	= 071352	8-1320	#38-7326							
DT155	= 071352	8-1320	#38-7327							
DT156	= 071352	8-1320	#38-7328							
DT157	= 071212	8-1320	#38-7329							
DT16	= 071024	8-1320	#38-7226							
DT160	= 071352	8-1320	#38-7330							
DT161	= 071212	8-1320	#38-7331							
DT162	= 071212	8-1320	#38-7332							
DT163	= 071352	8-1320	#38-7333							

SYMBOL	CROSS REFERENCE	REFERENCES								
SYMBOL	VALUE									
DT164	= 071212	8-1320	#38-7334							
DT165	071426	8-1320	#38-7335	38-7339	38-7340	38-7341	38-7342	38-7343	38-7344	38-7345
		38-7346	38-7347	38-7349	38-7350	38-7351	38-7352	38-7353	38-7354	38-7355
		38-7414								
DT166	= 071426	8-1320	#38-7339							
DT167	= 071426	8-1320	#38-7340							
DT17	071110	8-1320	#38-7227							
DT170	= 071426	8-1320	#38-7341							
DT171	= 071426	8-1320	#38-7342							
DT172	= 071426	8-1320	#38-7343							
DT173	= 071426	8-1320	#38-7344							
DT174	= 071426	8-1320	#38-7345							
DT175	= 071426	8-1320	#38-7346							
DT176	= 071426	8-1320	#38-7347							
DT177	071514	8-1320	#38-7348							
DT2	071062	8-1320	#38-7214							
DT20	071130	8-1320	#38-7228							
DT200	= 071426	8-1320	#38-7349							
DT201	= 071426	8-1320	#38-7350							
DT202	= 071426	8-1320	#38-7351							
DT203	= 071426	8-1320	#38-7352							
DT204	= 071426	8-1320	#38-7353							
DT205	= 071426	8-1320	#38-7354							
DT206	= 071426	8-1320	#38-7355							
DT207	071524	8-1320	#38-7356	38-7361	38-7368	38-7370	38-7371	38-7372	38-7373	38-7374
		38-7375	38-7380							
DT21	071152	8-1320	#38-7229							
DT210	071576	8-1320	#38-7359	38-7379						
DT211	= 071524	8-1320	#38-7361							
DT212	071620	8-1320	#38-7362	38-7365	38-7366	38-7367	38-7376	38-7377	38-7378	
DT213	= 071620	8-1320	#38-7365							
DT214	= 071620	8-1320	#38-7366							
DT215	= 071620	8-1320	#38-7367							
DT216	= 071524	8-1320	#38-7368							
DT217	071702	8-1320	#38-7369							
DT22	071174	8-1320	#38-7230							
DT220	= 071524	8-1320	#38-7370							
DT221	= 071524	8-1320	#38-7371							
DT222	= 071524	8-1320	#38-7372							
DT223	= 071524	8-1320	#38-7373							
DT224	= 071524	8-1320	#38-7374							
DT225	= 071524	8-1320	#38-7375							
DT226	= 071620	8-1320	#38-7376							
DT227	= 071620	8-1320	#38-7377							
DT23	071212	8-1320	#38-7231	38-7233	38-7234	38-7235	38-7236	38-7237	38-7238	38-7239
		38-7240	38-7241	38-7242	38-7243	38-7244	38-7245	38-7246	38-7247	38-7248
		38-7249	38-7250	38-7251	38-7252	38-7253	38-7254	38-7255	38-7289	38-7290
		38-7291	38-7292	38-7301	38-7302	38-7303	38-7304	38-7308	38-7311	38-7319
		38-7320	38-7321	38-7322	38-7329	38-7331	38-7332	38-7334	38-7394	
DT230	= 071620	8-1320	#38-7378							
DT231	= 071576	8-1320	#38-7379							
DT232	= 071524	8-1320	#38-7380							

SYMBOL CROSS REFERENCE

SYMBOL	VALUE	REFERENCES								
DT5	= 071024	8-1320	#38-7217							
DT50	= 071212	8-1320	#38-7253							
DT51	= 071212	8-1320	#38-7254							
DT52	= 071212	8-1320	#38-7255							
DT53	071264	8-1320	#38-7256	38-7260	38-7261	38-7262	38-7263	38-7264	38-7265	38-7266
		38-7267	38-7268	38-7269	38-7270	38-7271	38-7272	38-7273	38-7274	38-7275
		38-7276	38-7277	38-7278	38-7279	38-7280	38-7281	38-7282	38-7283	38-7284
		38-7285	38-7286	38-7287	38-7288					
DT54	= 071264	8-1320	#38-7260							
DT55	= 071264	8-1320	#38-7261							
DT56	= 071264	8-1320	#38-7262							
DT57	= 071264	8-1320	#38-7263							
DT6	= 071024	8-1320	#38-7218							
DT60	= 071264	8-1320	#38-7264							
DT61	= 071264	8-1320	#38-7265							
DT62	= 071264	8-1320	#38-7266							
DT63	= 071264	8-1320	#38-7267							
DT64	= 071264	8-1320	#38-7268							
DT65	= 071264	8-1320	#38-7269							
DT66	= 071264	8-1320	#38-7270							
DT67	= 071264	8-1320	#38-7271							
DT7	= 071024	8-1320	#38-7219							
DT70	= 071264	8-1320	#38-7272							
DT71	= 071264	8-1320	#38-7273							
DT72	= 071264	8-1320	#38-7274							
DT73	= 071264	8-1320	#38-7275							
DT74	= 071264	8-1320	#38-7276							
DT75	= 071264	8-1320	#38-7277							
DT76	= 071264	8-1320	#38-7278							
DT77	= 071264	8-1320	#38-7279							
EEEDON	020074	20-3279	#20-3366							
EEE1	017070	#20-3135								
EEE10	017464	#20-3236								
EEE11	017520	#20-3247								
EEE12	017554	#20-3258								
EEE13	017610	#20-3269								
EEE2	017124	#20-3146								
EEE3	017160	#20-3157								
EEE4	017214	#20-3168								
EEE5	017250	#20-3179								
EEE6	017304	#20-3191								
EEE7	017340	#20-3203								
EEE8	017374	#20-3214								
EEE9	017430	#20-3225								
EMTVEC	= 000030	#6-1301	*8-1327	*8-1327						
EM1	040044	8-1320	#35-6301							
EM10	040663	8-1320	#35-6316							
EM100	054021	8-1320	#35-6539							
EM101	054104	8-1320	#35-6541							
EM102	054300	8-1320	#35-6544							
EM103	054474	8-1320	#35-6547							
EM104	054606	8-1320	#35-6549							

SYMBOL CROSS REFERENCE		REFERENCES	
SYMBOL	VALUE		
EM105	054753	8-1320	#35-6552
EM106	055062	8-1320	#35-6554
EM107	055171	8-1320	#35-6556
EM11	040770	8-1320	#35-6318
EM110	055300	8-1320	#35-6558
EM111	044115	8-1320	#35-6390
EM112	044172	8-1320	#35-6391
EM113	044250	8-1320	#35-6392
EM114	044326	8-1320	#35-6394
EM115	044405	8-1320	#35-6396
EM116	044463	8-1320	#35-6397
EM117	044542	8-1320	#35-6404
EM12	041077	8-1320	#35-6320
EM120	044700	8-1320	#35-6406
EM121	045036	8-1320	#35-6408
EM122	045173	8-1320	#35-6410
EM123	045330	8-1320	#35-6412
EM124	045406	8-1320	#35-6413
EM125	045465	8-1320	#35-6414
EM126	045543	8-1320	#35-6416
EM127	045622	8-1320	#35-6418
EM13	041206	8-1320	#35-6322
EM130	045700	8-1320	#35-6419
EM131	045757	8-1320	#35-6426
EM132	046115	8-1320	#35-6428
EM133	046253	8-1320	#35-6431
EM134	046411	8-1320	#35-6433
EM135	046546	8-1320	#35-6435
EM136	046703	8-1320	#35-6438
EM137	047040	8-1320	#35-6440
EM14	041322	8-1320	#35-6324
EM140	047126	8-1320	#35-6441
EM141	= 044115	3-1320	#35-6442
EM142	= 044172	8-1320	#35-6443
EM143	047215	8-1320	#35-6444
EM144	047273	8-1320	#35-6446
EM145	047352	8-1320	#35-6454
EM146	047517	8-1320	#35-6456
EM147	047664	8-1320	#35-6458
EM15	041434	8-1320	#35-6326
EM150	050030	8-1320	#35-6460
EM151	050174	8-1320	#35-6462
EM152	050262	8-1320	#35-6463
EM153	= 045330	8-1320	#35-6464
EM154	= 045406	8-1320	#35-6465
EM155	050351	8-1320	#35-6466
EM156	050427	8-1320	#35-6468
EM157	050506	8-1320	#35-6476
EM16	041546	8-1320	#35-6328
EM160	050653	8-1320	#35-6478
EM161	051011	8-1320	#35-6481
EM162	051156	8-1320	#35-6483

35-6442
35-6443

35-6464
35-6465

SYMBOL	CROSS REFERENCE	REFERENCES
SYMBOL	VALUE	
EM163	051322	8-1320 #35-6485
EM164	051457	8-1320 #35-6488
EM165	055410	8-1320 #35-6572
EM166	055474	8-1320 #35-6574
EM167	055557	8-1320 #35-6576
EM17	041643	8-1320 #35-6330
EM170	055611	8-1320 #35-6578
EM171	055677	8-1320 #35-6580
EM172	056022	8-1320 #35-6582
EM173	056107	8-1320 #35-6584
EM174	056255	8-1320 #35-6586
EM175	056423	8-1320 #35-6588
EM176	056535	8-1320 #35-6590
EM177	056621	8-1320 #35-6592
EM2	040076	8-1320 #35-6302
EM20	041720	8-1320 #35-6331
EM200	056655	8-1320 #35-6593
EM201	056707	8-1320 #35-6595
EM202	056776	8-1320 #35-6597
EM203	057140	8-1320 #35-6599
EM204	057302	8-1320 #35-6601
EM205	057370	8-1320 #35-6603
EM206	057536	8-1320 #35-6605
EM207	057704	8-1320 #35-6701
EM21	041752	8-1320 #35-6332
EM210	057746	8-1320 #35-6702
EM211	060010	8-1320 #35-6703
EM212	060154	8-1320 #35-6730
EM213	060340	8-1320 #35-6731
EM214	060524	8-1320 #35-6732
EM215	060710	8-1320 #35-6733
EM216	061074	8-1320 #35-6734
EM217	061256	8-1320 #35-6737
EM22	042004	8-1320 #35-6333
EM220	061320	8-1320 #35-6738
EM221	061550	8-1320 #35-6739
EM222	062014	8-1320 #35-6740
EM223	062245	8-1320 #35-6741
EM224	062512	8-1320 #35-6742
EM225	062743	8-1320 #35-6743
EM226	063210	8-1320 #35-6744
EM227	063343	8-1320 #35-6745
EM23	042065	8-1320 #35-6334
EM230	063476	8-1320 #35-6746
EM231	063632	8-1320 #35-6747
EM232	= 057746	8-1320 #35-6748
EM233	063766	8-1320 #35-6778
EM234	064025	8-1320 #35-6779
EM235	064111	8-1320 #35-6780
EM236	064157	8-1320 #35-6781
EM237	064212	8-1320 #35-6782
EM24	042142	8-1320 #35-6339

35-6748

SYMBOL	CROSS REFERENCE	VALUE	REFERENCES
EM240		064276	8-1320 #35-6783
EM241		064332	8-1320 #35-6784
EM242		064435	8-1320 #35-6785
EM243		064471	8-1320 #35-6786
EM244		064574	8-1320 #35-6787
EM245		064633	8-1320 #35-6789
EM246		043525	8-1320 #35-6380
EM247		064715	8-1320 #35-6792
EM25		042240	8-1320 #35-6341
EM250		064751	8-1320 #35-6793
EM251		065003	8-1320 #35-6794
EM252		065036	8-1320 #35-6796
EM253		065127	8-1320 #35-6802
EM254		065213	8-1320 #35-6803
EM255		065300	8-1320 #35-6804
EM256		065366	8-1320 #35-6805
EM257		065455	8-1320 #35-6806
EM26		042325	8-1320 #35-6343
EM260		065543	8-1320 #35-6807
EM261		065632	8-1320 #35-6808
EM262		065722	8-1320 #35-6809
EM263		066013	8-1320 #35-6810
EM264		066100	8-1320 #35-6811
EM265		066165	8-1320 #35-6812
EM266		066252	8-1320 #35-6814
EM267		066353	8-1320 #35-6815
EM27	=	042240	8-1320 #35-6345
EM3		040132	8-1320 #35-6306
EM30		042423	8-1320 #35-6346
EM31		042475	8-1320 #35-6348
EM32		042110	8-1320 #35-6335
EM33		042542	8-1320 #35-6350
EM34		042565	8-1320 #35-6351
EM35		042617	8-1320 #35-6355
EM36		042672	8-1320 #35-6357
EM37		042740	8-1320 #35-6359
EM4		040237	8-1320 #35-6308
EM40		042763	8-1320 #35-6363
EM41		043015	8-1320 #35-6364
EM42		043070	8-1320 #35-6366
EM43		043221	8-1320 #35-6369
EM44		043352	8-1320 #35-6372
EM45		043420	8-1320 #35-6374
EM46		043473	8-1320 #35-6376
EM47		043550	8-1320 #35-6381
EM5		040344	8-1320 #35-6310
EM50		043704	8-1320 #35-6384
EM51		043757	8-1320 #35-6386
EM52		044025	8-1320 #35-6388
EM53		051655	8-1320 #35-6491
EM54		051706	8-1320 #35-6492
EM55		051623	8-1320 #35-6490

35-6345

SYMBOL CROSS REFERENCE

SYMBOL VALUE
 EM56 051736
 EM57 052027
 EM6 040451
 EM60 052117
 EM61 052221
 EM62 052415
 EM63 052611
 EM64 052722
 EM65 053041
 EM66 053154
 EM67 053223
 EM7 040556
 EM70 053306
 EM71 053337
 EM72 053367
 EM73 = 053367
 EM74 053421
 EM75 053530
 EM76 053621
 EM77 053711
 ERM10 034314
 ERROR = 10400C

REFERENCES

8-1320	#35-6499							
8-1320	#35-6501							
8-1320	#35-6312							
8-1320	#35-6503							
8-1320	#35-6505							
8-1320	#35-6508							
8-1320	#35-6511							
8-1320	#35-6513							
8-1320	#35-6515							
8-1320	#35-6517							
8-1320	#35-6519							
8-1320	#35-6314							
8-1320	#35-6521							
8-1320	#35-6522							
8-1320	#35-6523	35-6524						
8-1320	#35-6524							
8-1320	#35-6531							
8-1320	#35-6533							
8-1320	#35-6535							
8-1320	#35-6537							
33-6070	33-6070	#33-6070						
#6-1301	9-1479	9-1480	9-1481	9-1482	9-1483	9-1484	9-1485	9-1486
9-1487	9-1488	9-1489	9-1490	9-1503	12-1787	12-1789	12-1791	12-1793
12-1797	12-1799	12-1813	12-1815	12-1817	12-1819	12-1821	12-1823	12-1825
12-1840	12-1842	12-1844	12-1846	12-1848	12-1850	12-1852	13-2116	13-2121
13-2127	13-2133	13-2138	13-2143	13-2148	13-2154	13-2165	13-2174	13-2183
14-2252	14-2266	14-2279	14-2292	14-2306	14-2322	14-2339	14-2356	14-2373
15-2390	15-2407	15-2424	15-2441	15-2507	15-2523	16-2646	16-2655	16-2663
16-2676	17-2706	17-2717	17-2729	17-2740	17-2751	17-2762	17-2773	17-2785
17-2797	17-2808	17-2819	17-2831	17-2842	17-2922	17-2926	18-2952	18-2963
18-2974	18-2985	18-2997	18-3010	18-3023	19-3112	19-3116	20-3143	20-3154
20-3165	20-3176	20-3187	20-3199	20-3211	20-3222	20-3233	20-3244	20-3255
20-3266	20-3277	20-3357	20-3361	21-3388	21-3400	21-3411	21-3423	21-3510
21-3514	22-3545	22-3547	22-3561	22-3564	22-3578	22-3580	22-3594	22-3597
22-3693	22-3697	23-3723	23-3727	23-3759	23-3763	24-3797	24-3799	24-3814
24-3816	24-3831	24-3833	24-3849	24-3851	24-3945	24-3949	24-3974	24-3978
24-4010	24-4014	25-4049	25-4051	25-4065	25-4067	25-4081	25-4083	25-4097
25-4099	25-4202	25-4206	25-4214	25-4218	25-4243	25-4247	26-4294	26-4296
26-4312	26-4314	26-4330	26-4332	26-4348	26-4350	27-4454	27-4458	27-4466
27-4470	27-4495	27-4499	28-4544	28-4546	28-4560	28-4562	28-4576	28-4578
28-4592	28-4594	28-4608	28-4610	28-4624	28-4627	28-4641	28-4644	28-4658
28-4660	28-4674	28-4676	28-4690	28-4692	28-4706	28-4708	28-4722	28-4724
28-4738	28-4740	28-4754	28-4756	28-4857	28-4871	28-4877	29-4911	29-4913
29-4929	29-4931	29-4946	29-4948	29-4964	29-4966	29-4983	29-4985	29-4999
29-5002	29-5016	29-5019	29-5033	29-5035	29-5051	29-5054	29-5071	29-5073
29-5089	29-5091	29-5107	29-5109	29-5123	29-5125	29-5233	29-5250	29-5256
30-5290	30-5292	30-5309	30-5324	30-5326	30-5344	30-5360	30-5362	30-5460
30-5474	30-5480	31-5523	31-5525	31-5544	31-5561	31-5563	31-5582	32-5685
32-5702	32-5708	33-5799	33-5801	33-5812	33-5814	33-5826	33-5828	33-5840
33-5842	33-5856	33-5858	33-5870	33-5872	33-5885	33-5887	33-5898	33-5900
33-5914	33-5916	33-5928	33-5930	33-5942	33-5944	33-6040	33-6070	34-6209
34-6220	34-6231							

SYMBOL CROSS REFERENCE

SYMBOL	VALUE	REFERENCES								
GGG7	024614	#28-4631								
GGG8	024664	#28-4648								
GGG9	024734	#28-4664								
GGP1	011560	12-1840	#12-1859							
GGP2	011570	10-1646	10-1670	10-1712	12-1738	12-1813	12-1815	12-1817	12-1819	12-1821
		12-1823	12-1825	12-1842	12-1844	12-1846	12-1848	12-1850	12-1852	#12-1863
GGP3	011600	10-1648	10-1672	12-1813	12-1815	12-1817	12-1819	12-1821	12-1823	12-1825
		12-1840	#12-1867							
GGP4	011610	#12-1871								
GGP5	011620	10-1581	10-1583	10-1605	10-1607	12-1787	12-1787	12-1789	12-1789	12-1791
		12-1791	12-1793	12-1793	12-1797	12-1797	12-1799	12-1799	#12-1875	
GGP6	011630	10-1588	10-1623	10-1653	10-1719	12-1787	12-1789	12-1791	12-1793	12-1797
		12-1799	12-1813	12-1815	12-1817	12-1840	12-1842	12-1844	#12-1879	
GGP7	011640	10-1688	12-1819	12-1821	12-1823	12-1825	#12-1884			
GGP8	011650	10-1714	12-1740	12-1842	12-1844	12-1846	12-1848	12-1850	12-1852	#12-1888
GGP9	011660	12-1756	12-1846	12-1848	12-1850	12-1852	#12-1892			
GG1	006756	#10-1576								
GG10	007252	10-1637	#10-1641							
GG11	007310	10-1644	#10-1649							
GG12	007332	#10-1655	10-1658							
GG13	007342	10-1656	#10-1658							
GG14	007360	10-1661	#10-1665							
GG15	007416	10-1668	#10-1673							
GG16	007434	10-1669	#10-1678							
GG17	007502	#10-1690	10-1693							
GG18	007512	10-1691	#10-1693							
GG19	007546	10-1703	#10-1707							
GG2	007014	10-1579	#10-1584							
GG20	007604	10-1710	#10-1715							
GG21	007626	#10-1721	10-1724							
GG22	007636	10-1722	#10-1724							
GG23	007654	10-1727	#12-1733							
GG24	007712	12-1736	#12-1741							
GG25	007730	12-1737	#12-1746							
GG26	007776	#12-1758	12-1761							
GG27	010006	12-1759	#12-1761							
GG28	010042	12-1770	#12-1773							
GG3	007036	#10-1590	10-1593							
GG4	007046	10-1591	#10-1593							
GG5	007064	10-1596	#10-1600							
GG6	007122	10-1603	#10-1608							
GG7	007140	10-1604	#10-1613							
GG8	007206	#10-1625	10-1628							
GG9	007216	10-1626	#10-1628							
GNS	= *****	6-1313	6-1313	8-1328	33-6066	33-6066	33-6082	33-6082	33-6082	33-6082
		33-6082	33-6082	33-6082	33-6082	33-6082	33-6082	33-6082	33-6082	33-6082
		33-6082	33-6082	33-6082	33-6082	33-6082	33-6083	33-6083	33-6084	33-6084
GTSWR	= 104405	8-1328	#33-6082							
HHDATO	006602	9-1356	9-1362	9-1390	9-1414	9-1420	9-1444	9-1466	9-1479	9-1480
		9-1481	9-1482	9-1483	9-1484	9-1485	9-1486	9-1487	9-1488	9-1489
		9-1490	#9-1505							
HHDONE	006752	9-1478	9-1479	9-1480	9-1481	9-1482	9-1483	9-1484	9-1485	9-1486

SYMBOL CROSS REFERENCE

SYMBOL	VALUE	REFERENCES								
HH19	005470	#9-1448	9-1451							
HH2	005062	9-1350	#9-1354							
HH20	005500	9-1449	#9-1451							
HH21	005516	9-1454	#9-1457							
HH22	005546	9-1460	#9-1464							
HH23	005570	#9-1470	9-1473							
HH24	005600	9-1471	#9-1473							
HH25	005616	9-1476	#9-1478							
HH3	005104	#9-1360	9-1370							
HH4	005124	#9-1365	9-1368							
HH5	005134	9-1366	#9-1368							
HH6	005142	9-1361	#9-1370							
HH7	005154	9-1372	#9-1379							
HH8	005212	9-1383	#9-1388							
HH9	005234	#9-1394	9-1397							
HT	= 000011	#6-1301	33-6074	33-6074						
HXDATO	013300	13-1919	13-1926	13-1959	13-1968	13-2012	13-2031	13-2040	13-2066	13-2096
		13-2126	13-2132	13-2147	13-2153	13-2182	#13-2190			
HXDONE	013410	13-2110	13-2117	13-2122	13-2128	13-2134	13-2139	13-2144	13-2149	13-2155
		13-2166	13-2175	13-2184	#13-2230					
HXER1	012666	13-1916	#13-2114							
HXER10	013156	13-1999	#13-2168							
HXER11	013210	13-2044	#13-2177							
HXER12	013230	13-2046	#13-2180							
HXER13	013260	13-2073	13-2103	#13-2186						
HXER2	012726	13-1930	13-2018	#13-2124						
HXER22	012742	#13-2126	13-2179	13-2188						
HXER3	012756	13-1932	#13-2130							
HXER33	012772	#13-2132								
HXER4	013006	13-2079	13-2109	#13-2136						
HXER5	012706	13-1955	#13-2119							
HXER6	013042	13-1972	#13-2145							
HXER66	013056	#13-2147								
HXER7	013072	13-1974	#13-2151							
HXER8	013024	13-1937	13-1981	#13-2141						
HXER9	013122	13-1990	#13-2157							
HXP1	013310	13-1909	13-1943	13-2056	13-2061	13-2069	13-2091	13-2099	13-2186	13-2187
		#13-2195								
HXP2	013320	13-1911	13-1914	13-1925	13-1946	13-1953	13-2009	13-2114	13-2119	13-2124
		13-2130	13-2145	13-2151	#13-2200					
HXP3	013330	#13-2204								
HXP4	013340	13-2028	13-2177	13-2180	#13-2208					
HXP5	013350	13-2034	13-2178	13-2181	#13-2212					
HXP6	013360	13-2006	13-2026	13-2087	#13-2216					
HXP7	013370	13-1921	13-1967	13-2014	13-2039	13-2125	13-2131	#13-2221		
HXP8	013400	13-1962	13-2146	13-2152	#13-2225					
HX1	011674	#13-1906								
HX10	012102	13-1954	#13-1957							
HX11	012124	#13-1964	13-1976							
HX12	012144	#13-1970	13-1973							
HX13	012154	13-1971	#13-1973							
HX14	012162	13-1965	#13-1976							

SYMBOL CROSS REFERENCE

SYMBOL	VALUE	REFERENCES				
HX15	012200	13-1980	#13-1985			
HX16	012226	13-1989	#13-1992			
HX165	012232	#13-1993	13-2159	13-2170		
HX17	012260	13-1998	#13-2001			
HX18	012316	13-2005	#13-2010			
HX19	012336	#13-2016	13-2019			
HX2	011724	13-1912	#13-1913			
HX20	012346	13-2017	#13-2019			
HX21	012350	#13-2022				
HX22	012400	13-2025	#13-2029			
HX23	012420	#13-2036	13-2048			
HX24	012440	#13-2042	13-2045			
HX25	012450	13-2043	#13-2045			
HX26	012456	13-2037	#13-2048			
HX27	012460	#13-2052				
HX28	012512	13-2060	#13-2062			
HX29	012534	#13-2071	13-2074			
HX3	011740	13-1915	#13-1917			
HX30	012544	13-2072	#13-2074			
HX31	012562	13-2078	#13-2083			
HX32	012612	13-2090	#13-2092			
HX33	012634	#13-2101	13-2104			
HX34	012644	13-2102	#13-2104			
HX35	012662	13-2108	#13-2110			
HX4	011760	#13-1923	13-1933			
HX5	012000	#13-1928	13-1931			
HX6	012010	13-1929	#13-1931			
HX7	012016	13-1924	#13-1933			
HX8	012034	13-1936	#13-1939			
HX9	012066	13-1947	#13-1951			
IBSAVE	033762	#33-6070	*33-6070	33-6070	33-6070	33-6070
IIIDON	021464	22-3598	#23-3768			
III1	020630	#22-3535				
III2	020674	#22-3551				
III3	020740	#22-3568				
III4	021004	#22-3584				
IOTVEC	= 000020	#6-1301	*8-1327	*8-1327		
JJJDON	022424	24-3852	#24-4019			
JJJ1	021470	#24-3785				
JJJ2	021554	#24-3803				
JJJ3	021640	#24-3820				
JJJ4	021724	#24-3838				
KKKDON	023266	25-4100	#25-4266			
KKK1	022430	#25-4039				
KKK3	022474	#25-4055				
KKK4	022540	#25-4071				
KKK5	022604	#25-4087				
LF	= 000012	#6-1301	33-6074	33-6074		
LLLDON	024230	26-4351	#27-4518			
LLL1	023272	#26-4282				
LLL2	023356	#26-4300				
LLL3	023442	#26-4318				

SYMBOL CROSS REFERENCE

SYMBOL	CROSS REFERENCE	REFERENCES
LLL4	023526	#26-4336
LOOP	005030	#8-1330 33-6066
LPERR	= 104413	9-1347 9-1379 9-1404 9-1434 9-1457 10-1576 10-1600 10-1641 10-1665
		10-1707 12-1733 13-1906 13-1939 13-1985 13-2001 13-2022 13-2052 13-2083
		14-2245 14-2256 14-2269 14-2282 14-2296 14-2309 14-2326 14-2343 14-2360
		14-2376 15-2394 15-2411 15-2428 16-2545 16-2570 16-2592 17-2698 17-2709
		17-2720 17-2732 17-2743 17-2754 17-2765 17-2777 17-2789 17-2800 17-2811
		17-2823 17-2834 18-2944 18-2955 18-2966 18-2977 18-2988 18-3001 18-3013
		20-3135 20-3146 20-3157 20-3168 20-3179 20-3191 20-3203 20-3214 20-3225
		20-3236 20-3247 20-3258 20-3269 21-3380 21-3392 21-3403 21-3414 22-3535
		22-3551 22-3568 22-3584 24-3785 24-3803 24-3820 24-3838 25-4039 25-4055
		25-4071 25-4087 26-4282 26-4300 26-4318 26-4336 28-4534 28-4550 28-4566
		28-4582 28-4598 28-4614 28-4631 28-4648 28-4664 28-4680 28-4696 28-4712
		28-4728 28-4744 29-4901 29-4917 29-4935 29-4952 29-4970 29-4989 29-5006
		29-5023 29-5039 29-5059 29-5077 29-5095 29-5113 30-5279 30-5296 30-5313
		30-5331 30-5349 31-5510 31-5529 31-5548 31-5567 33-5791 33-5804 33-5818
		33-5832 33-5848 33-5862 33-5878 33-5891 33-5906 33-5920 33-5934 #33-6084
		30-5363 #30-5494
MMMDON	030742	30-5279
MMM1	030020	#30-5296
MMM2	030072	#30-5313
MMM3	030144	#30-5331
MMM4	030216	#30-5349
MMM5	030270	#6-617
MNUMBE	= 000267	8-1318
MODDDO	031750	31-5637 31-5650 31-5657 #32-5718
MODDD1	031760	31-5639 31-5651 31-5665 32-5689 #32-5720
MODDOV	031362	31-5511 31-5530 31-5549 31-5568 #31-5618
MODDSU	027416	29-4902 29-4918 29-4936 29-4953 29-4971 29-4990 29-5007 29-5024 29-5040
		29-5060 29-5078 29-5096 29-5114 #29-5161
		29-5179 29-5192 29-5197 29-5220 #29-5259
		29-5181 29-5193 29-5205 29-5237 #29-5261
		30-5418 30-5431 30-5438 #30-5490
		30-5420 30-5432 30-5444 #30-5492
		30-5280 30-5297 30-5314 30-5332 30-5350 #30-5399
		28-4535 28-4551 28-4567 28-4583 28-4599 28-4615 28-4632 28-4649 28-4665
		28-4681 28-4697 28-4713 28-4729 28-4745 #28-4793
		28-4811 28-4824 28-4829 #28-4880
		28-4813 28-4825 28-4835 #28-4882
		28-4798 #28-4884 29-5166 30-5404 31-5623
MODFTO	025656	
MODFT1	025666	
MODP1	025676	
MSA1	037144	#34-6273 38-7400
MSA2	037162	#34-6274 38-7400
MSA3	037205	#34-6275 38-7401
MSA4	037217	#34-6276 38-7401
MSA5	037235	#34-6277 38-7402
MSA6	037250	#34-6278 38-7402
MS1	037342	#34-6282 38-7213 38-7231 38-7257 38-7294 38-7336
MS10	037541	#34-6289 38-7259 38-7338
MS11	037563	#34-6290 38-7258 38-7337
MS12	037607	#34-6291 38-7258 38-7337
MS2	037360	#34-6283 38-7213 38-7231 38-7257 38-7294 38-7336
MS3	037400	#34-6284 38-7214
MS37	037632	#34-6292 38-7382

SYMBOL	CROSS REFERENCE	REFERENCES									
SYMBOL	VALUE										
TRAPVE	= 000034	#6-1301	*8-1327	*8-1327							
TRTVEC	= 000014	#6-1301									
TST1	005030	#9-1343									
TST10	017066	#20-3132									
TST11	020076	#21-3377									
TST12	020626	#22-3532									
TST13	021466	#24-3782									
TST14	022426	#25-4036									
TST15	023270	#26-4279									
TST16	024232	#28-4531									
TST17	025710	#29-4898									
TST2	006754	#10-1573									
TST20	030016	#30-5276									
TST21	030744	#31-5507									
TST22	031772	#33-5766									
TST23	033154	#33-6062									
TST3	011672	#13-1904									
TST4	013412	#14-2242									
TST5	014420	#16-2542									
TST6	015122	#17-2695									
TST7	016132	#18-2941									
TYPE	= 104401	8-1328	33-5788	33-6066	33-6066	33-6066	33-6066	33-6070	33-6070	33-6074	33-6076
		33-6080	33-6080	33-6080	33-6080	33-6080	33-6080	33-6080	#33-6082	33-6087	33-6098
		33-6122	33-6124	33-6127	33-6129	33-6163	33-6169	33-6169	33-6180	33-6185	33-6195
		33-6080	#33-6082	33-6107	33-6139	33-6194					
TYPOC	= 104402	#33-6082									
TYPON	= 104404	33-6066	33-6066	#33-6082	33-6156						
TYPOS	= 104403	8-1324	#8-1324								
\$APTHD	004332	33-6078	33-6078								
\$ASTAT	= *****	33-6078	#33-6078								
\$ATYC	035230	#33-6078									
\$ATY1	035204	33-6074	#33-6078								
\$ATY3	035212	33-6070	#33-6078								
\$ATY4	035222	#7-1314	*8-1328	33-6080	33-6080	33-6080					
\$AUTOB	001134	#7-1314									
\$BASE	001372	#7-1314									
\$BDADR	001122	#7-1314									
\$BDDAT	001126	#7-1314									
\$BELL	001306	#7-1314	33-6070	33-6070	33-6070						
\$CDW1	001376	#7-1314									
\$CDW2	001400	#7-1314									
\$CHARC	034752	*33-6074	*33-6074	33-6074	*33-6074	#33-6074					
\$CKSWR	035452	#33-6080	33-6082	33-6082							
\$CLR.T	033356	33-6066	#33-6066								
\$CMTAG	001100	#7-1314	8-1327	8-1327	8-1327	8-1327	8-1327	8-1327	8-1327	8-1327	8-1327
\$CM1	= 000024	#7-1314	7-1314	7-1314	#7-1314	7-1314	7-1314	7-1314	#7-1314	7-1314	7-1314
		#7-1314	7-1314	7-1314	#7-1314	7-1314	7-1314	7-1314	#7-1314	7-1314	7-1314
		#7-1314	7-1314	7-1314	#7-1314	7-1314	7-1314	7-1314	#7-1314	7-1314	7-1314
		#7-1314	7-1314	7-1314	#7-1314	7-1314	7-1314	7-1314	#7-1314	7-1314	7-1314
		#7-1314	7-1314	7-1314	#7-1314	7-1314	7-1314	7-1314	#7-1314	7-1314	7-1314
		#7-1314	7-1314	7-1314	#7-1314	7-1314	7-1314	7-1314	#7-1314	7-1314	7-1314
		#7-1314	7-1314	7-1314	#7-1314	7-1314	7-1314	7-1314	#7-1314	7-1314	7-1314
		#7-1314	7-1314	7-1314	#7-1314	7-1314	7-1314	7-1314	#7-1314	7-1314	7-1314
\$CM2	= 000050	#7-1314	7-1314	7-1314	#7-1314	7-1314	7-1314	7-1314	#7-1314	7-1314	7-1314

SYMBOL	CROSS REFERENCE	REFERENCES	SEQUENCE	CREF	V01					
SYMBOL	VALUE									
\$ERMAX	001115	#7-1314	*8-1327	33-6068	*33-6068	33-6068	33-6068			
\$ERROR	033764	8-1327	#33-6070							
\$ERRPC	001116	#7-1314	*33-6070	*33-6070	33-6070	33-6070	*33-6070	33-6070	33-6070	33-6101
		33-6106								
\$ERRTB	001442	#8-1314	33-6119							
\$ERTTL	001112	#7-1314	33-6066	*33-6066	*33-6070	33-6070	33-6070	33-6070		
\$ESCAP	001304	#7-1314	*8-1327	*33-6068	33-6070	33-6070	33-6070	33-6070		
\$ETABL	001336	#7-1314								
\$ETEND	001442	#7-1314	8-1324							
\$FATAL	001320	#7-1314	*33-6078	*33-6114						
\$FFLG	035450	*33-6078	*33-6078	33-6078	*33-6078	#33-6078				
\$FILLC	001156	#7-1314	33-6074	33-6074	33-6074					
\$FILLS	001155	#7-1314	33-6074	33-6074						
\$GDADR	001120	#7-1314								
\$GDDAT	001124	#7-1314								
\$GET42	033340	#33-6066								
\$GTSWR	035522	#33-6080	33-6082	33-6082						
\$HD	= 000003	6-1292	6-1292	6-1292						
\$HIBTS	004332	#8-1324								
\$ICNT	001104	#7-1314	*33-6068	33-6068	*33-6068	33-6068	33-6068			
\$ILLUP	036362	33-6087	33-6087	#33-6087						
\$INTAG	001135	#7-1314	33-6080	33-6080	33-6080					
\$ITEMB	001114	#7-1314	*33-6070	33-6070	33-6070	33-6070	33-6070	33-6103		
\$LF	001314	#7-1314	33-6070	33-6070	33-6074	33-6074				
\$LFLG	035447	*33-6078	#33-6078							
\$LOOP	033434	33-6066	#33-6066							
\$LPADR	001106	#7-1314	*8-1327	*33-6068	*33-6068	33-6068	33-6068	33-6068	33-6068	
\$LPERR	001110	#7-1314	*8-1327	33-6068	*33-6068	33-6068	33-6068	33-6070	*34-6242	
\$MADR1	001350	#7-1314								
\$MADR2	001354	#7-1314								
\$MADR3	001360	#7-1314								
\$MADR4	001364	#7-1314								
\$MAIL	001316	#7-1314	8-1324	8-1324	8-1327	8-1328	33-6068	33-6070	33-6074	
\$MAMS1	001346	#7-1314								
\$MAMS2	001352	#7-1314								
\$MAMS3	001356	#7-1314								
\$MAMS4	001362	#7-1314								
\$MBADR	004334	#8-1324								
\$MFLG	035446	*33-6078	33-6078	*33-6078	#33-6078					
\$MNEW	036107	33-6080	#33-6080							
\$MSGAD	001332	#7-1314	*33-6078	33-6078						
\$MSGLG	001334	#7-1314	*33-6078							
\$MSGTY	001316	#7-1314	33-6078	*33-6078	33-6078	*33-6078				
\$MSWR	036076	33-6080	#33-6080							
\$MTYP1	001347	#7-1314								
\$MTYP2	001353	#7-1314								
\$MTYP3	001357	#7-1314								
\$MTYP4	001363	#7-1314								
\$MXCNT	033756	33-6068	33-6068	33-6068	#33-6068					
\$NULL	001154	#7-1314	33-6074	33-6074	33-6074					
\$NWTST	= 000001	#8-1343	8-1343	#9-1343	9-1343	#9-1573	9-1573	#10-1573	10-1573	#12-1904
		12-1904	#13-1904	13-1904	#13-2242	13-2242	#14-2242	14-2242	#15-2542	15-2542

SYMBOL CROSS REFERENCE
SYMBOL VALUE

REFERENCES

SYMBOL	CROSS REFERENCE VALUE	REFERENCES	REFERENCES	REFERENCES	REFERENCES	REFERENCES	REFERENCES	REFERENCES	REFERENCES	REFERENCES
		#16-2542	16-2542	#16-2695	16-2695	#17-2695	17-2695	#17-2941	17-2941	#18-2941
		18-2941	#19-3132	19-3132	#20-3132	20-3132	#20-3377	20-3377	#21-3377	21-3377
		#21-3532	21-3532	#22-3532	22-3532	#23-3782	23-3782	#24-3782	24-3782	#24-4036
		24-4036	#25-4036	25-4036	#25-4279	25-4279	#26-4279	26-4279	#27-4531	27-4531
		#28-4531	28-4531	#28-4898	28-4898	#29-4898	29-4898	#29-5276	29-5276	#30-5276
		30-5276	#30-5507	30-5507	#31-5507	31-5507	#32-5766	32-5766	#33-5766	33-5766
\$OCNT	035200	*33-6076	*33-6076	#33-6076						
\$OMODE	035202	*33-6076	*33-6076	33-6076	*33-6076	*33-6076	#33-6076			
\$OVER	033742	33-6068	33-6068	33-6068	33-6068	#33-6068				
\$PASS	001324	#7-1314	*8-1327	*33-6066	*33-6066	33-6066	33-6066	33-6066	33-6068	33-6068
		33-6068								
\$PASTM	004340	#8-1324								
\$PWRAD	036344	#33-6087								
\$PWRDN	036204	8-1327	#33-6087	33-6087						
\$PWRMG	036340	#33-6087								
\$PWRUP	036256	33-6087	#33-6087							
\$QUES	001312	#7-1314	33-6070	33-6070	33-6074	33-6074	33-6080			
\$RDCHR	035734	#33-6080	33-6082	33-6082						
\$RDDEC	= *****	33-6082								
\$RDLIN	= *****	33-6082								
\$RDOCT	= *****	33-6082								
\$RDSZ	= 000001	#33-6080	33-6080							
\$REGAD	001160	#7-1314								
\$REG0	001162	#7-1314	*33-6070	*33-6070	33-6070					
\$REG1	001164	#7-1314								
\$REG10	001202	#7-1314								
\$REG11	001204	#7-1314								
\$REG12	001206	#7-1314								
\$REG13	001210	#7-1314								
\$REG14	001212	#7-1314								
\$REG15	001214	#7-1314								
\$REG16	001216	#7-1314								
\$REG17	001220	#7-1314								
\$REG2	001166	#7-1314								
\$REG20	001222	#7-1314								
\$REG21	001224	#7-1314								
\$REG22	001226	#7-1314								
\$REG23	001230	#7-1314								
\$REG3	001170	#7-1314								
\$REG4	001172	#7-1314								
\$REG5	001174	#7-1314								
\$REG6	001176	#7-1314								
\$REG7	001200	#7-1314								
\$RESRE	034354	#33-6072	33-6082							
\$RTNAD	033436	#33-6066								
\$RTRN	033432	8-1327	*8-1327	*8-1327	33-6066	#33-6066				
\$R2A	= *****	33-6082								
\$SAVRE	034316	#33-6072	33-6082	33-6082						
\$SAVR6	036366	*33-6087	33-6087	*33-6087	*33-6087	#33-6087				
\$SCOPE	033446	8-1327	#33-6068							
\$SETUP	= 000137	#6-1311	6-1311	#6-1311	6-1311	#6-1311	6-1311	#6-1311	6-1311	#6-1311
		6-1311	#6-1311	6-1311	#6-1311	8-1327	8-1327	8-1327	8-1327	8-1327

SYMBOL	CROSS REFERENCE VALUE	REFERENCES	CREF	V01						
\$STUP	= 177777	8-1327 8-1328 33-6080 #6-1311 #6-1311 33-6068	8-1327 33-6066 33-6087 #6-1311 #6-1311 #33-6068	8-1327 33-6066 6-1311 6-1311	8-1327 33-6068 #6-1311 #6-1311	8-1327 33-6070 #6-1311 #6-1311	8-1327 33-6070 6-1311 6-1311	8-1327 33-6070 #6-1311 #6-1311	8-1328 33-6070 #6-1311 #6-1311	8-1328 33-6080 6-1311 6-1311
\$SVLAD	033706	#8-1323	8-1323							
\$SVPC	= 004332	6-1292	#6-1292	#6-1296	7-1314	7-1314	7-1314	8-1327	8-1327	8-1327
\$SWR	= 177400	8-1327	8-1327	9-1343	10-1573	13-1904	14-2242	16-2542	17-2695	18-2941
		20-3132	21-3377	22-3532	24-3782	25-4036	26-4279	28-4531	29-4898	30-5276
		31-5507	33-5766	33-6066	33-6066	33-6066	33-6066	33-6066	33-6068	33-6068
		33-6068	33-6068	33-6068	33-6068	33-6068	33-6068	33-6068	33-6068	33-6068
		33-6068	33-6068	33-6068	33-6068	33-6068	33-6068	33-6068	33-6068	33-6068
		33-6070	33-6070	33-6070	33-6070	33-6070	33-6070	33-6070	33-6070	33-6070
		33-6070	33-6070	33-6087						
\$SWREG	001340	#7-1314	8-1327							
\$SWRMK	= 000000	33-6068	33-6068	33-6068	33-6068	33-6068	33-6068	33-6068	33-6068	33-6068
		33-6068								
\$SWRMS	= 000200	#6-1297								
\$TAB	037335	33-6180	#34-6280	38-7212	38-7212	38-7212	38-7214	38-7227	38-7227	38-7228
		38-7228	38-7228	38-7229	38-7229	38-7229	38-7230	38-7230	38-7231	38-7231
		38-7231	38-7256	38-7256	38-7256	38-7293	38-7293	38-7293	38-7335	38-7335
		38-7356	38-7359	38-7359	38-7360	38-7362	38-7362	38-7362	38-7369	38-7369
		38-7381	38-7384	38-7384	38-7385	38-7395	38-7395	38-7395	38-7398	38-7399
		38-7399	38-7399							
\$TBIT	033440	*8-1327	*33-6066	33-6066	33-6066	#33-6066	*33-6087			
\$TERM	= 000030	#33-6085								
\$TESTN	001322	#7-1314	*33-6068							
\$TIMES	001302	#7-1314	*8-1327	*33-6066	*33-6068	33-6068	*33-6068	33-6068	33-6068	
\$TKB	001146	#7-1314	33-6074	33-6074	33-6074	33-6074	33-6080	33-6080	33-6080	33-6080
		33-6080	33-6080							
\$TKS	001144	#7-1314	33-6074	33-6074	33-6074	33-6074	33-6080	33-6080	33-6080	33-6080
		33-6080	33-6080	33-6080						
\$TMP0	001232	#7-1314	*33-6099	*33-6100	38-7212	38-7214	38-7227	38-7228	38-7229	38-7230
		38-7231	38-7256	38-7293	38-7335	38-7348	38-7356	38-7359	38-7362	38-7369
		38-7381	38-7384	38-7385	38-7395	38-7396	38-7398	38-7399		
\$TMP1	001234	#7-1314	*33-6101	38-7212	38-7214	38-7227	38-7228	38-7229	38-7230	38-7231
		38-7256	38-7293	38-7335	38-7348	38-7356	38-7359	38-7362	38-7369	38-7381
		38-7384	38-7385	38-7395	38-7396	38-7398	38-7399			
\$TMP10	001252	#7-1314	*9-1479	*9-1480	*9-1481	*9-1482	*9-1483	*9-1484	*9-1485	*9-1486
		*9-1487	*9-1488	*9-1489	*9-1490	*12-1789	*12-1791	*12-1793	*12-1797	*12-1799
		*12-1815	*12-1817	*12-1819	*12-1821	*12-1823	*12-1825	*12-1842	*12-1844	*12-1846
		*12-1848	*12-1850	*12-1852	*16-2675	*17-2900	*19-3085	*20-3335	*21-3483	*22-3653
		*24-3905	*25-4151	*27-4403	*28-4825	*29-5193	*30-5432	*31-5651	*33-6017	38-7231
		38-7259	38-7293	38-7338	38-7362	38-7399				
\$TMP11	001254	#7-1314	*9-1479	*9-1480	*9-1481	*9-1482	*9-1483	*9-1484	*9-1485	*9-1486
		*9-1487	*9-1488	*9-1489	*9-1490	*12-1789	*12-1791	*12-1793	*12-1797	*12-1799
		*12-1815	*12-1817	*12-1819	*12-1821	*12-1823	*12-1825	*12-1842	*12-1844	*12-1846
		*12-1848	*12-1850	*12-1852	*22-3673	*23-3744	*24-3925	*24-3995	*25-4180	*25-4258
		*27-4432	*27-4510	*28-4826	*28-4875	*29-5195	*29-5254	*30-5433	*30-5478	*31-5652
		*32-5706	*33-6012	38-7256	38-7293	38-7335	38-7362	38-7400		
\$TMP12	001256	#7-1314	*28-4827	*28-4876	*29-5194	*29-5255	*30-5434	*30-5479	*31-5653	*32-5707

SYMBOL CROSS REFERENCE

SYMBOL VALUE
\$TMP7 001250

\$TN = 000023

\$TPB 001152

\$TPFLG 001157

\$TPS 001150

\$TRAP 036120

\$TRAP2 036142

\$TRP = 000014

\$TRPAD 036154

\$STM 004336

\$STNM 001102

\$TYPBN = *****

\$TYPDS = *****

\$TYPE 034412

\$TYPE_C 034624

\$TYPEX 034754

\$TYPOC 035002

\$TYPON 035016

\$TYPOS 034756

\$UNIT 001330

\$UNITM 004342

\$USWR 001342

\$VECT1 001366

\$VECT2 001370

\$XOFF = 000023

\$XON = 000021

\$XTSTR 033460

\$GET4 = 000001

\$OFILL 035201

\$OCAT = *****

.LPER 037066

.RSET 037074

.\$ASTA = *****

.\$X = 004332

REFERENCES

#7-1314	*13-2125	*13-2131	*13-2146	*13-2152	*13-2178	*13-2181	*13-2187	*16-2644
*16-2654	*16-2662	*16-2674	*17-2899	*19-3084	*20-3334	*21-3482	*22-3672	*24-3924
*25-4179	*25-4257	*27-4431	*27-4509	*28-4824	*29-5192	*30-5431	*31-5650	*33-6011
38-7231	38-7259	38-7293	38-7338	38-7383	38-7399			
6-1292	#6-1292	8-1343	9-1343	#9-1343	9-1573	10-1573	#10-1573	12-1904
13-1904	#13-1904	13-2242	14-2242	#14-2242	15-2542	16-2542	#16-2542	16-2695
17-2695	#17-2695	17-2941	18-2941	#18-2941	19-3132	20-3132	#20-3132	20-3377
21-3377	#21-3377	21-3532	22-3532	#22-3532	23-3782	24-3782	#24-3782	24-4036
25-4036	#25-4036	25-4279	26-4279	#26-4279	27-4531	28-4531	#28-4531	28-4898
29-4898	#29-4898	29-5276	30-5276	#30-5276	30-5507	31-5507	#31-5507	32-5766
33-5766	#33-5766	33-6062						
#7-1314	33-6074	33-6074	33-6074					
#7-1314	33-6074	33-6074	33-6074					
#7-1314	33-6074	33-6074	33-6074					
8-1327	#33-6082							
#33-6082	33-6082							
#33-6082	33-6082	33-6082	33-6082	33-6082	#33-6082	33-6082	33-6082	33-6082
33-6082	#33-6082	33-6082	33-6082	33-6082	33-6082	#33-6082	33-6082	33-6082
33-6082	33-6082	#33-6082	33-6082	33-6082	33-6082	33-6082	#33-6082	33-6082
33-6082	33-6082	33-6082	#33-6082	33-6082	33-6082	33-6082	33-6082	#33-6082
33-6082	33-6082	33-6082	33-6082	#33-6082	33-6082	33-6082	33-6082	33-6082
#33-6082	33-6083	33-6083	33-6083	33-6083	#33-6083	33-6084	33-6084	33-6084
33-6084	#33-6084							
33-6082	#33-6082	33-6085						
#8-1324								
#7-1314	*33-6066	33-6068	33-6068	*33-6068	33-6068	33-6068	33-6068	33-6068
33-6070	33-6070	33-6070	33-6099					
33-6082								
33-6082								
#33-6074	33-6078	33-6082	33-6082					
33-6074	33-6074	33-6074	#33-6074	33-6080				
33-6074	33-6074	33-6074	#33-6074					
#33-6076	33-6082	33-6082						
33-6076	#33-6076	33-6082						
#33-6076	33-6082							
#7-1314								
#8-1324								
#7-1314								
#7-1314								
33-6074	33-6074							
33-6074	33-6074	33-6074	33-6080					
#33-6068								
#33-6066	#33-6066	33-6066						
*33-6076	*33-6076	33-6076	#33-6076					
33-6068	33-6070							
33-6084	#34-6242							
33-6083	#34-6254							
33-6078	33-6078							
#8-1324	8-1324							

MACRO CROSS REFERENCE

MACRO NAME	REFERENCES	8-1320	8-1320	8-1320	8-1320						
LCM1	#35-6752	8-1320	8-1320	8-1320	8-1320						
LCM2	#35-6757	35-6779	35-6778	35-6782	35-6787						
LCM3	#35-6761	35-6784	35-6784	35-6786	35-6786						
LCM4	#35-6768	#35-6783	#35-6783	#35-6785	#35-6785						
LCM5	#35-6771	#35-6780	#35-6780	#35-6780	#35-6780						
LDM1	#35-6626										
LDM2	#35-6632										
LDM3	#35-6636										
LDM4	#35-6640										
LDM5	#35-6644										
LDM6	#35-6647	#35-6774	35-6781	35-6781	35-6781						
LDM7	#35-6651										
LDM8	#35-6655										
LDM9	#35-6659										
LD1M1	#35-6613										
LD1M2	#35-6617										
LD1M3	#35-6623										
LOADTP	#6-1058										
LPER	#6-1275	9-1347	9-1379	9-1404	9-1434	9-1457	10-1576	10-1600	10-1641	10-1665	
	10-1707	12-1733	13-1906	13-1939	13-1985	13-2001	13-2022	13-2052	13-2083	14-2245	
	14-2256	14-2269	14-2282	14-2296	14-2309	14-2326	14-2343	14-2360	14-2376	15-2394	
	15-2411	15-2428	16-2545	16-2570	16-2592	17-2698	17-2709	17-2720	17-2732	17-2743	
	17-2754	17-2765	17-2777	17-2789	17-2800	17-2811	17-2823	17-2834	18-2944	18-2955	
	18-2966	18-2977	18-2988	18-3001	18-3013	20-3135	20-3146	20-3157	20-3168	20-3179	
	20-3191	20-3203	20-3214	20-3225	20-3236	20-3247	20-3258	20-3269	21-3380	21-3392	
	21-3403	21-3414	22-3535	22-3551	22-3568	22-3584	24-3785	24-3803	24-3820	24-3838	
	25-4039	25-4055	25-4071	25-4087	26-4282	26-4300	26-4318	26-4336	28-4534	28-4550	
	28-4566	28-4582	28-4598	28-4614	28-4631	28-4648	28-4664	28-4680	28-4696	28-4712	
	28-4728	28-4744	29-4901	29-4917	29-4935	29-4952	29-4970	29-4989	29-5006	29-5023	
	29-5039	29-5059	29-5077	29-5095	29-5113	30-5279	30-5296	30-5313	30-5331	30-5349	
	31-5510	31-5529	31-5548	31-5567	33-5791	33-5804	33-5818	33-5832	33-5848	33-5862	
	33-5878	33-5891	33-5906	33-5920	33-5934	33-5934					
MDO1	#35-6420	#35-6433	#35-6438	#35-6438	#35-6438						
MDO2	#35-6470	35-6483	35-6488	35-6488	35-6488						
MDU1	#35-6423	#35-6426	#35-6431	#35-6431	#35-6431						
MDU2	#35-6473	35-6476	35-6481	35-6481	35-6481						
MERROR	#6-1178										
MF M4	#35-6560	35-6572	35-6590	35-6590	35-6590						
MF M5	#35-6563	35-6574	35-6580	35-6580	35-6584	35-6586	35-6588	35-6597	35-6599	35-6603	35-6605
	35-6815										
MF M6	#35-6566	#35-6576	#35-6593	#35-6593	#35-6593						
MF M7	#35-6569	#35-6578	#35-6582	#35-6582	#35-6595	#35-6601					
MF O1	#35-6398	35-6408	35-6410	35-6410	35-6410						
MF O2	#35-6448	#35-6458	#35-6460	#35-6460	#35-6460						
MF U1	#35-6401	35-6404	35-6406	35-6406	35-6406						
MF U2	#35-6451	35-6454	35-6456	35-6456	35-6456						
MM1	#35-6360	35-6364	35-6366	35-6369	35-6372	35-6374	35-6392	35-6394	35-6444	35-6446	
MM2	#35-6377	35-6381	35-6384	35-6386	35-6388	35-6414	35-6416	35-6428	35-6435	35-6466	
	35-6468	35-6478	35-6485	35-6485	35-6485						
MODDM1	#35-6525	#35-6533	#35-6537	#35-6541	#35-6544	#35-6547	#35-6558	#35-6558	#35-6558	#35-6558	
MODDM2	#35-6528	35-6531	35-6535	35-6539	35-6549	35-6552	35-6554	35-6556	35-6556	35-6556	

MACRO CROSS REFERENCE

MACRO NAME	REFERENCES									
MODM1	#35-6493	#35-6499	#35-6505	#35-6508	#35-6511	#35-6513	#35-6515	#35-6517		
MODM2	#35-6496	#35-6501	#35-6503	#35-6519						
MSG	#8-1333	9-1343	#9-1559	10-1573	#12-1898	13-1904	#13-2234	14-2242	#15-2534	16-2542
	#16-2687	17-2695	#17-2934	18-2941	#19-3124	20-3132	#20-3369	21-3377	#21-3522	22-3532
	#23-3772	24-3782	#24-4023	25-4036	#25-4270	26-4279	#27-4523	28-4531	#28-4890	29-4898
	#29-5267	30-5276	#30-5498	31-5507	#32-5724	33-5766				
MULT	#6-1301									
NAMEP	#6-1072	8-1328								
NEWTST	#6-1288	#6-1301	#8-1343	#9-1573	#12-1904	#13-2242	#15-2542	#16-2695	#17-2941	#19-3132
	#20-3377	#21-3532	#23-3782	#24-4036	#25-4279	#27-4531	#28-4898	#29-5276	#30-5507	#32-5766
NTMPM	#6-1068									
OU1	#35-6712	#35-6738	#35-6740	#35-6742						
OU2	#35-6718	35-6739	35-6741	35-6743						
OU3	#35-6725	#35-6744	#35-6745	#35-6746	#35-6747					
POP	#6-1301	#33-6072	#33-6078	#33-6078	#33-6087	#33-6087				
PUSH	#6-1301	33-6072	33-6078	33-6078	33-6078	33-6087	33-6087			
REPORT	#6-1301									
ROM	#6-1101									
ROMAC	#6-1233									
ROM1	#6-1104									
ROM10	#6-1125									
ROM11	#6-1128									
ROM12	#6-1131									
ROM13	#6-1134									
ROM14	#6-1137									
ROM15	#6-1140									
ROM16	#6-1143									
ROM17	#6-1146									
ROM2	#6-1107									
ROM20	#6-1149									
ROM3	#6-1110									
ROM30	#6-1095									
ROM31	#6-1098									
ROM4	#6-1113									
ROM5	#6-1116									
ROM6	#6-1119									
ROM7	#6-1122									
RSET	#6-1075	9-1557	12-1896	13-2230	15-2530	16-2684	17-2931	19-3121	20-3366	21-3519
	23-3768	24-4019	25-4266	27-4518	28-4886	29-5263	30-5494	32-5722	33-6058	34-6210
	34-6221	34-6232								
RTMAC	#35-6706	35-6730	35-6731	35-6732	35-6733					
RTMPM	#6-1063									
SETPRI	#6-1301									
SETTRA	#33-6082	#33-6082	#33-6082	#33-6082	#33-6082	#33-6082	#33-6082	#33-6082	#33-6082	#33-6082
	#33-6083	#33-6084								
SETUP	#6-1301	8-1327								
SKIP	#6-1301									
SLASH	#6-1301									
SPACE	#6-1281	#6-1301								
SSKAD	#6-1272									
STARS	#6-1301	7-1314	7-1314	7-1314	8-1323	8-1324	8-1324	8-1324	9-1343	9-1343
	10-1573	10-1573	13-1904	13-1904	14-2242	14-2242	16-2542	16-2542	17-2695	17-2695

MACRO CROSS REFERENCE

MACRO NAME	REFERENCES	18-2941	20-3132	20-3132	21-3377	21-3377	22-3532	22-3532	24-3782	24-3782
	18-2941	18-2941	20-3132	20-3132	21-3377	21-3377	22-3532	22-3532	24-3782	24-3782
	25-4036	25-4036	26-4279	26-4279	28-4531	28-4531	29-4898	29-4898	30-5276	30-5276
	31-5507	31-5507	33-5766	33-5766	33-6066	33-6068	33-6070	33-6072	33-6074	33-6076
	33-6078	33-6080	33-6080	33-6080	33-6082	33-6087	33-6087	33-6091	34-6198	34-6215
	34-6226	34-6240	34-6246							
SUMMAC	#6-1260									
SWRSU	#6-1301	#8-1327	#8-1327							
TMAC	#6-1092	33-6062								
TRMTRP	#33-6082	#33-6085								
TYPBIN	#6-1301									
TYPDEC	#6-1301									
TYPNAM	#6-1301	#8-1328								
TYPNUM	#6-1301									
TYPOCS	#6-1301	#33-6066	#33-6066							
TYPOCT	#6-1301	33-6080								
TYPTXT	#6-1301	33-6066	33-6066							
\$\$CMRE	#6-1314	#7-1314	#7-1314	#7-1314	#7-1314	#7-1314	#7-1314	#7-1314	#7-1314	#7-1314
	#7-1314	#7-1314	#7-1314	#7-1314	#7-1314	#7-1314	#7-1314	#7-1314	#7-1314	#7-1314
	#7-1314									
\$\$CMTM	#6-1314	#7-1314	#7-1314	#7-1314	#7-1314	#7-1314	#7-1314	#7-1314	#7-1314	#7-1314
	#7-1314	#7-1314	#7-1314	#7-1314	#7-1314	#7-1314	#7-1314	#7-1314	#7-1314	#7-1314
	#7-1314									
\$\$ESCA	#6-1301									
\$\$NEWT	#6-1288	#6-1301	#8-1343	#9-1573	#12-1904	#13-2242	#15-2542	#16-2695	#17-2941	#19-3132
	#20-3377	#21-3532	#23-3782	#24-4036	#25-4279	#27-4531	#28-4898	#29-5276	#30-5507	#32-5766
\$\$SET	#33-6082	#33-6082	#33-6082	#33-6082	#33-6082	#33-6082	#33-6082	#33-6082	#33-6082	#33-6082
	#33-6083	#33-6084								
\$\$SETM	#8-1327	8-1327								
\$\$SETU	#8-1327	#8-1327								
\$\$SKIP	#6-1301									
.EQUAT	#6-1287	#6-1301								
.EQUIV	#6-1278	#6-1291	6-1301	6-1301	6-1301	6-1301	6-1301	6-1301	6-1301	6-1301
	6-1301	6-1301	6-1301	6-1301	6-1301	6-1301	6-1301	6-1301	6-1301	6-1301
	6-1301	6-1301	6-1301	6-1301	6-1301	6-1301	6-1301	6-1301	6-1301	6-1301
.HEADE	#6-1287	#6-1292								
.SETUP	#6-1287	#6-1311								
.SWRH:	#6-1287									
.\$ACT1	#6-1287	8-1323								
.\$APT8	#6-1289	#7-1314	7-1314							
.\$APTH	#6-1289	8-1324								
.\$APTY	#6-1290	33-6078								
.\$CATC	#6-1287	#6-1313								
.\$CMTA	#6-1287	#6-1314								
.\$EOP	#6-1288	33-6066								
.\$ERRO	#6-876	33-6070								
.\$POWE	#6-1289	33-6087								
.\$READ	#6-1290	#33-6080								
.\$SAVE	#6-1288	33-6072								
.\$SCOP	#6-625	33-6068								
.\$TRAP	#6-1289	#33-6082								
.\$TYPD	#6-1289									
.\$TYPE	#6-1288	#6-1290	#33-6074							

CKFPBB

CREATED BY MACRO ON 28-MAR-81 AT 14:50

PAGE 41

G 1

SEQUENCE 214

REF V01

MACRO CROSS REFERENCE

MACRO NAME

REFERENCES

.STYPO

#6-1288 #33-6076