

CONTENTS

1. ABSTRACT
2. REQUIREMENTS
 - 2.1 EQUIPMENT
 - 2.2 STORAGE
 - 2.3 PRELIMINARY PROGRAMS
3. LOADING PROCEDURE
4. STARTING PROCEDURE
 - 4.1 CONTROL SWITCH SETTINGS
 - 4.2 STARTING ADDRESS
 - 4.3 PROGRAM AND OPERATOR INTERACTION
5. OPERATING PROCEDURE
 - 5.1 OPERATIONAL SWITCH SETTINGS
 - 5.3 OPERATOR ACTION
6. ERRORS
 - 6.1 SUMMARY
 - 6.2 ERROR RECOVERY
7. RESTRICTIONS
 - 7.1 STARTING RESTRICTIONS
 - 7.2 OPERATING RESTRICTIONS
8. MISCELLANEOUS
 - 8.1 EXECUTION TIMES
 - 8.2 STACK POINTER
 - 8.3 PASS COUNT
 - 8.4 T-BIT TRAPPING
 - 8.5 SOFTWARE SWITCH REGISTER
 - 8.6 INTERRUPTS TEST
 - 8.7 ACT, APT AND XXOP COMPATIBILITY
9. PROGRAM DESCRIPTION
 - 9.1 DFFPBA
10. LISTING
 - 10.1 DFFPBA

1. ABSTRACT

 THE THREE PROGRAMS:

DFFPA DFFPB DFFPC

ARE DESIGN TO DETECT AND REPORT LOGIC FAULTS IN THE PDP 11/34 FP11-A 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 155 (OC) 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-A. IF THE PROGRAMS OR TESTS ARE NOT RUN IN ORDER THEN ERROR MESSAGES MAY NOT BE ACCURATE.

A. DFFPA

DFFPA TESTS:

LDFPS
 STFPS
 CFCC
 SETF, SETD, SETI AND SETL
 STST
 LDF AND LDD (ALL SOURCE MODES)
 STD (MODE 0 AND 1)
 ADDF, ADD AND SUBD (MOST CONDITIONS)

B. DFFPB

DFFPB TESTS:

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

C. DFFPC

DFFPC TESTS:

STF AND STD (ALL MODES)
 STCFD AND STCOF
 CLRD AND CLRF
 NEGF AND NEG0

 113
 114
 115
 116
 117
 118
 119
 120
 121
 122
 123
 124
 125
 126
 127
 128
 129
 130
 131
 132
 133
 134
 135
 136
 137
 138
 139
 140
 141
 142
 143
 144
 145
 146
 147
 148
 149
 150
 151
 152
 153
 154
 155
 156
 157
 158
 159
 160
 161
 162
 163
 164
 165
 166
 167
 168

169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220

ABSF AND ABSO
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 STCOI
STEXP
STST

2. REQUIREMENTS

2.1 EQUIPMENT

A PDP 11/34 (WITH OR WITHOUT CONSOLE), LA30 (OR EQUIVALENT) AND AN FP11-A 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 DFFPB 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 11/34 CENTRAL PROCESSOR IS FAULTLESS, THEREFORE WHEN IN DOUBT RUN THE PDP 11/34 PROCESSOR DIAGNOSTICS BEFORE THESE FP11-A DIAGNOSTICS.

3. LOADING PROCEDURE

THE PROGRAMS WILL BE SUPPLIED ON THE 11:34 DIAGNOSTIC MEDIA. REFER TO THE XXDP OPERATING MANUAL FOR FURTHER INFORMATION.

4. STARTING PROCEDURE

4.1 CONTROL SWITCH SETTINGS

SEE SECTION 5.1

4.2 PROGRAM AND OPERATOR ACTION

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

5. OPERATING PROCEDURE

5.1 OPERATIONAL SWITCH SETTINGS

THE SWITCH SETTING ARE:

	OCTAL	
SW<15>=1...	10000	HALT ON ERROR
SW<14>=1...	4000	LOOP ON CURRENT TEST
SW<13>=1...	2000	INHIBIT ERROR TYPE OUTS
SW<12>=1...	1000	INHIBIT 7-BIT TRAPPING
SW<11>=1...	400	INHIBIT ITERATIONS
SW<10>=1...	200	RING TTY BELL ON ERROR
SW<9>=1....	100	LOOP ON ERROR
SW<8>=1....	40	LOOP ON TEST SPECIFIED IN SW'S THROUGH SW<0>
SW<7>=1....	20	PRINT ERROR SUMMARY EVEN IF SW<13>=1. THIS APPLIES ONLY TO PROGRAM DFFPA.
SW<7>=1....	20	DESELECT CORRECT INTERRUPT TEST IN PROGRAM DFFPB. NOTE THAT THIS TEST WILL AUTOMATICALLY BE DESELECTED BY THE ABSENCE OF THE SPECIAL TEST EQUIPMENT DESIGNED TO CONDUCT THIS TEST. IF THIS EQUIPMENT IS NOT INSTALLED THERE IS NO NEED TO DESELECT THIS TEST. THIS APPLIES ONLY TO PROGRAM DFFPB!

6. ERRORS

6.1 SUMMARIES

IN PROGRAM DFFPA 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

225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280

281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336

(TYPED ON THE TTY). EVERY ERROR THOUGH IS LOGGED AND AN ERROR SUMMARY IS PRINTED WHEN THE TEST IS COMPLETE. NOTE THAT IS SW<13>=1 THIS SUMMARY WILL NOT BE TYPED UNLESS SW<7>=1. IN OTHER WORDS TO GET JUST AN ERROR SUMMARY FROM EITHER OF THESE TWO TESTS 1 AND 11 IN PROGRAM DFFPA 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

337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392

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 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 DFFPB 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

393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448TEST 1 ROUND\TRUNK TEST

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

TEST 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.

449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504

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.

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 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 WAS 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 ACD 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.

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

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).

10. LISTING

&

000266
000002

MNUMBER=266
PROGNUM=2

.LIST ME
.NLIST MD,MC,CND

```

617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634                .ENABL ABS
635
636
637
638                .TITLE MAINDEC-11-DFFPB-A      PDP 11/34 FPP DIAGNOSTIC PART 2
639                ;*COPYRIGHT (C) SEP 1976
640                ;*DIGITAL EQUIPMENT CORP.
641                ;*MAYNARD, MASS. 01754
642                ;*
643                ;*PROGRAM BY ANTHONY S. VEZZA
644                ;*
645                ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
646                ;*PACKAGE (MAINDEC-11-DZQAC-C2), SEPT 14, 1976.
647                ;*
648                $TN=1
649                $SWR=160000      ;;HALT ON ERROR, LOOP ON TEST, INHIBIT ERROR TYP0UT
650
651
652                FPVECT=244
653                $SWR=177400
654                $SWRMSK=200
655                TAB=11
656                CRLF=15
657
658                .SBTTL BASIC DEFINITIONS
659
660                ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
661                001100      STACK= 1100
662                .EQUIV EMT,ERROR      ;;BASIC DEFINITION OF ERROR CALL
663                .EQUIV IOT,SCOPE      ;;BASIC DEFINITION OF SCOPE CALL
664
665                ;*MISCELLANEOUS DEFINITIONS
666                000011      HT= 11      ;;CODE FOR HORIZONTAL TAB
667                000012      LF= 12      ;;CODE FOR LINE FEED
668                000015      CR= 15      ;;CODE FOR CARRIAGE RETURN
669                000200      CRLF= 200   ;;CODE FOR CARRIAGE RETURN-LINE FEED
670                177776      PS= 177776 ;;PROCESSOR STATUS WORD
671                .EQUIV PS,PSW
672                177774      STKLMT= 177774 ;;STACK LIMIT REGISTER

```

```

673      177772      PIRQ= 177772      ;; PROGRAM INTERRUPT REQUEST REGISTER
674      177570      DSWR= 177570      ;; HARDWARE SWITCH REGISTER
675      177570      DDISP= 177570      ;; HARDWARE DISPLAY REGISTER
676
677      ;*GENERAL PURPOSE REGISTER DEFINITIONS
678      000000      R0= %0      ;; GENERAL REGISTER
679      000001      R1= %1      ;; GENERAL REGISTER
680      000002      R2= %2      ;; GENERAL REGISTER
681      000003      R3= %3      ;; GENERAL REGISTER
682      000004      R4= %4      ;; GENERAL REGISTER
683      000005      R5= %5      ;; GENERAL REGISTER
684      000006      R6= %6      ;; GENERAL REGISTER
685      000007      R7= %7      ;; GENERAL REGISTER
686      000006      SP= %6      ;; STACK POINTER
687      000007      PC= %7      ;; PROGRAM COUNTER
688
689      ;*PRIORITY LEVEL DEFINITIONS
690      000000      PR0= 0      ;; PRIORITY LEVEL 0
691      000040      PR1= 40      ;; PRIORITY LEVEL 1
692      000100      PR2= 100      ;; PRIORITY LEVEL 2
693      000140      PR3= 140      ;; PRIORITY LEVEL 3
694      000200      PR4= 200      ;; PRIORITY LEVEL 4
695      000240      PR5= 240      ;; PRIORITY LEVEL 5
696      000300      PR6= 300      ;; PRIORITY LEVEL 6
697      000340      PR7= 340      ;; PRIORITY LEVEL 7
698
699      ;*"SWITCH REGISTER" SWITCH DEFINITIONS
700      100000      SW15= 100000
701      040000      SW14= 40000
702      020000      SW13= 20000
703      010000      SW12= 10000
704      004000      SW11= 4000
705      002000      SW10= 2000
706      001000      SW09= 1000
707      000400      SW08= 400
708      000200      SW07= 200
709      000100      SW06= 100
710      000040      SW05= 40
711      000020      SW04= 20
712      000010      SW03= 10
713      000004      SW02= 4
714      000002      SW01= 2
715      000001      SW00= 1
716      .EQUIV SW09, SW9
717      .EQUIV SW08, SW8
718      .EQUIV SW07, SW7
719      .EQUIV SW06, SW6
720      .EQUIV SW05, SW5
721      .EQUIV SW04, SW4
722      .EQUIV SW03, SW3
723      .EQUIV SW02, SW2
724      .EQUIV SW01, SW1
725      .EQUIV SW00, SW0
726
727      ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
728      100000      BIT15= 100000

```

BASIC DEFINITIONS

760 040000
761 020000
762 010000
763 004000
764 002000
765 001000
766 000400
767 000200
768 000100
769 000040
770 000020
771 000010
772 000004
773 000002
774 000001
775 000001
776 000001
777
778
779
780 000000
781
782
783
784 000174

BIT14= 40000
BIT13= 20000
BIT12= 10000
BIT11= 4000
BIT10= 2000
BIT09= 1000
BIT08= 400
BIT07= 200
BIT06= 100
BIT05= 40
BIT04= 20
BIT03= 10
BIT02= 4
BIT01= 2
BIT00= 1
.EQUIV BIT09,BIT9
.EQUIV BIT08,BIT8
.EQUIV BIT07,BIT7
.EQUIV BIT06,BIT6
.EQUIV BIT05,BIT5
.EQUIV BIT04,BIT4
.EQUIV BIT03,BIT3
.EQUIV BIT02,BIT2
.EQUIV BIT01,BIT1
.EQUIV BIT00,BIT0

.*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

.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
. =174

785 000174 000000
786 000176 000000
787
788 000200 000137 004336

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

.SBTTL COMMON TAGS

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

789
790
791
792
793
794
795 001100
796 001100
797 001100 000000
798 001102 000
799 001103 000
800 001104 000000
801 001106 000000
802 001110 000000
803 001112 000000
804 001114 000
805 001115 001
806 001116 000000
807 001120 000000
808 001122 000000
809 001124 000000
810 001126 000000
811 001130 000000
812 001132 000000
813 001134 000
814 001135 000
815 001136 000000
816 001140 177570
817 001142 177570
818 001144 177560
819 001146 177562
820 001150 177564
821 001152 177566
822 001154 000
823 001155 002
824 001156 012
825 001157 000
826 001160 000000
827
828 001162 000000
829 001164 000000
830 001166 000000
831 001170 000000
832 001172 000000
833 001174 000000
834 001176 000000
835 001200 000000
836 001202 000000
837 001204 000000
838 001206 000000
839 001210 000000
840 001212 000000
841 001214 000000
842 001216 000000
843 001220 000000
844 001222 000000

. =1100
\$CMTAG: .WORD 0
\$STNM: .BYTE 0
\$ERFLG: .BYTE 0
\$ICNT: .WORD 0
\$LPADR: .WORD 0
\$LPEPR: .WORD 0
\$ERTTL: .WORD 0
\$ITEMB: .BYTE 0
\$ERMAX: .BYTE 1
\$ERRPC: .WORD 0
\$GDADR: .WORD 0
\$BOARD: .WORD 0
\$GDDAT: .WORD 0
\$BODAT: .WORD 0
\$AUTOB: .BYTE 0
\$INTAG: .BYTE 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
\$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

:: 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
:: 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)

945 001224 000000
946 001226 000000
947 001230 000000
948 001232 000000
949 001234 000000
950 001236 000000
951 001238 000000
952 001240 000000
953 001242 000000
954 001244 000000
955 001246 000000
956 001248 000000
957 001250 000000
958 001252 000000
959 001254 000000
960 001256 000000
961 001258 000000
962 001260 000000
963 001262 000000
964 001264 000000
965 001266 000000
966 001270 000000
967 001272 000000
968 001274 000000
969 001276 000000
970 001300 000000
971 001302 000000
972 001304 000000
973 001306 177607
974 001312 077
975 001313 015
976 001314 000012
977
978
979
980 001316 000000
981 001320 000000
982 001322 000000
983 001324 000000
984 001326 000000
985 001330 000000
986 001332 000000
987 001334 000000
988 001336 000
989 001336 000
990 001337 000
991 001340 000000
992 001342 000000
993 001344 000000
994
995
996
997
998
999
000 001346 000

000377

\$REG21: .WORD 0 ::CONTAINS ((\$REGAC)+42)
\$REG22: .WORD 0 ::CONTAINS ((\$REGAD)+44)
\$REG23: .WORD 0 ::CONTAINS ((\$REGAD)+45)
\$TMP0: .WORD 0 ::USER DEFINED
\$TMP1: .WORD 0 ::USER DEFINED
\$TMP2: .WORD 0 ::USER DEFINED
\$TMP3: .WORD 0 ::USER DEFINED
\$TMP4: .WORD 0 ::USER DEFINED
\$TMP5: .WORD 0 ::USER DEFINED
\$TMP6: .WORD 0 ::USER DEFINED
\$TMP7: .WORD 0 ::USER DEFINED
\$TMP10: .WORD 0 ::USER DEFINED
\$TMP11: .WORD 0 ::USER DEFINED
\$TMP12: .WORD 0 ::USER DEFINED
\$TMP13: .WORD 0 ::USER DEFINED
\$TMP14: .WORD 0 ::USER DEFINED
\$TMP15: .WORD 0 ::USER DEFINED
\$TMP16: .WORD 0 ::USER DEFINED
\$TMP17: .WORD 0 ::USER DEFINED
\$TMP20: .WORD 0 ::USER DEFINED
\$TMP21: .WORD 0 ::USER DEFINED
\$TMP22: .WORD 0 ::USER DEFINED
\$TMP23: .WORD 0 ::USER DEFINED
\$TIMES: 0 ::MAX. NUMBER OF ITERATIONS
\$ESCAPE: 0 ::ESCAPE ON ERROR ADDRESS
\$BELL: .ASCIZ <207><377><377> ::CODE FOR BELL
\$QUES: .ASCIZ ??? ::QUESTION MARK
\$CARLF: .ASCIZ <15> ::CARRIAGE RETURN
\$LF: .ASCIZ <12> ::LINE FEED

::*****
:SBTTL APT MAILBOX-ETABLE

::*****

.EVEN
\$MAIL: .WORD 0 ::APT MAILBOX
\$MSGTY: .WORD 0 ::MESSAGE TYPE CODE
\$FATAL: .WORD 0 ::FATAL ERROR NUMBER
\$TESTN: .WORD 0 ::TEST NUMBER
\$PASS: .WORD 0 ::PASS COUNT
\$DEVCT: .WORD 0 ::DEVICE COUNT
\$UNIT: .WORD 0 ::I/O UNIT NUMBER
\$MSGAD: .WORD 0 ::MESSAGE ADDRESS
\$MSGLG: .WORD 0 ::MESSAGE LENGTH
\$ETABLE: .WORD 0 ::APT ENVIRONMENT TABLE
\$ENV: .BYTE 0 ::ENVIRONMENT BYTE
\$ENVM: .BYTE 0 ::ENVIRONMENT MODE BITS
\$SWREG: .WORD 0 ::APT SWITCH REGISTER
\$USWR: .WORD 0 ::USER SWITCHES
\$CPUOP: .WORD 0 ::CPU TYPE, OPTIONS
BITS 15-11=CPU TYPE
11/04=01, 11/05=02, 11/20=03, 11/40=04, 11 45=05
11/70=06, PDQ=07, Q=10
BIT 10=REAL TIME CLOCK
BIT 9=FLOATING POINT PROCESSOR
BIT 8=MEMORY MANAGEMENT
\$MAMS1: .BYTE 0 ::HIGH ADDRESS, M.S. BYTE

F02

901	001347	000	\$MTYP1:	.BYTE	AMTYP1	::MEM. TYPE, BLK#1
902			::*			MEM. TYPE BYTE -- (HIGH BYTE,
903			::*			900 NSEC CORE=001
904			::*			300 NSEC BIPOLAR=002
905			::*			500 NSEC MOS=003
906	001350	000000	\$MADR1:	.WORD	AMADR1	::HIGH ADDRESS, BLK#1
907			::*			MEM. LAST ADDR.=3 BYTES, THIS WORD AND LOW OF "TYPE" ABOVE
908	001352	000	\$MAMS2:	.BYTE	AMAMS2	::HIGH ADDRESS, M.S. BYTE
909	001353	000	\$MTYP2:	.BYTE	AMTYP2	::MEM. TYPE, BLK#2
910	001354	000000	\$MADR2:	.WORD	AMADR2	::MEM. LAST ADDRESS, BLK#2
911	001356	000	\$MAMS3:	.BYTE	AMAMS3	::HIGH ADDRESS, M.S. BYTE
912	001357	000	\$MTYP3:	.BYTE	AMTYP3	::MEM. TYPE, BLK#3
913	001360	000000	\$MADR3:	.WORD	AMADR3	::MEM. LAST ADDRESS, BLK#3
914	001362	000	\$MAMS4:	.BYTE	AMAMS4	::HIGH ADDRESS, M.S. BYTE
915	001363	000	\$MTYP4:	.BYTE	AMTYP4	::MEM. TYPE, BLK#4
916	001364	000000	\$MADR4:	.WORD	AMADR4	::MEM. LAST ADDRESS, BLK#4
917	001366	000000	\$VECT1:	.WORD	AVECT1	::INTERRUPT VECTOR#1, BUS PRIORITY#1
918	001370	000000	\$VECT2:	.WORD	AVECT2	::INTERRUPT VECTOR#2, BUS PRIORITY#2
919	001372	000000	\$BASE:	.WORD	ABASE	::BASE ADDRESS OF EQUIPMENT UNDER TEST
920	001374	000000	\$DEVN:	.WORD	ADEVN	::DEVICE MAP
921	001376	000000	\$CDW1:	.WORD	ACDW1	::CONTROLLER DESCRIPTION WORD#1
922	001400	000000	\$CDW2:	.WORD	ACDW2	::CONTROLLER DESCRIPTION WORD#2
923	001402	000000	\$DDW0:	.WORD	ADDW0	::DEVICE DESCRIPTOR WORD#0
924	001404	000000	\$DDW1:	.WORD	ADDW1	::DEVICE DESCRIPTOR WORD#1
925	001406	000000	\$DDW2:	.WORD	ADDW2	::DEVICE DESCRIPTOR WORD#2
926	001410	000000	\$DDW3:	.WORD	ADDW3	::DEVICE DESCRIPTOR WORD#3
927	001412	000000	\$DDW4:	.WORD	ADDW4	::DEVICE DESCRIPTOR WORD#4
928	001414	000000	\$DDW5:	.WORD	ADDW5	::DEVICE DESCRIPTOR WORD#5
929	001416	000000	\$DDW6:	.WORD	ADDW6	::DEVICE DESCRIPTOR WORD#6
930	001420	000000	\$DDW7:	.WORD	ADDW7	::DEVICE DESCRIPTOR WORD#7
931	001422	000000	\$DDW8:	.WORD	ADDW8	::DEVICE DESCRIPTOR WORD#8
932	001424	000000	\$DDW9:	.WORD	ADDW9	::DEVICE DESCRIPTOR WORD#9
933	001426	000000	\$DDW10:	.WORD	ADDW10	::DEVICE DESCRIPTOR WORD#10
934	001430	000000	\$DDW11:	.WORD	ADDW11	::DEVICE DESCRIPTOR WORD#11
935	001432	000000	\$DDW12:	.WORD	ADDW12	::DEVICE DESCRIPTOR WORD#12
936	001434	000000	\$DDW13:	.WORD	ADDW13	::DEVICE DESCRIPTOR WORD#13
937	001436	000000	\$DDW14:	.WORD	ADDW14	::DEVICE DESCRIPTOR WORD#14
938	001440	000000	\$DDW15:	.WORD	ADDW15	::DEVICE DESCRIPTOR WORD#15
939						
940						
941	001442		\$ETEND:			
942						

.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 SITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
:*NOTE1: IF SITEMB IS 0 THE ONLY PERTINENT DATA IS (SERAPP).
:*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

993
994
995
996
997
998
999
000
001
002
003
004
005
006
007
008
009
010
011
012
013
014
015
016
017
018
019
020
021
022
023
024
025
026
027
028
029
030
031
032
033
034
035
036
037
038
039
040
041
042
043
044
045
046
047
048
049
050
051
052
053
054
055
056
057
058
059
060
061
062
063
064
065
066
067
068
069
070
071
072
073
074
075
076
077
078
079
080
081
082
083
084
085
086
087
088
089
090
091
092
093
094
095
096
097
098
099

Index	SITEMB	DATA	EM	DH	DT	DF
001442						
001442	037660	366216	070540			
001450	070012					
001452	037712	066306	070576			
001460	070030					
001462	037746	066216	070540			
001470	070012					
001472	040053	066216	070540			
001500	070012					
001502	040160	066216	070540			
001510	070012					
001512	040265	066216	070540			
001520	070012					
001522	040372	066216	070540			
001530	070012					
001532	040477	066216	070540			
001540	070012					
001542	040604	066216	070540			
001550	070012					
001552	040713	066216	070540			
001560	070012					
001562	041022	066216	070540			
001570	070012					
001572	041136	066216	070540			
001600	070012					
001602	041250	066216	070540			
001610	070012					
001612	041362	066216	070540			

SERRTB:
:ITEM 1 .WORD EM1,DH1,DT1,DF1
:ITEM 2 .WORD EM2,DH2,DT2,DF2
:ITEM 3 .WORD EM3,DH3,DT3,DF3
:ITEM 4 .WORD EM4,DH4,DT4,DF4
:ITEM 5 .WORD EM5,DH5,DT5,DF5
:ITEM 6 .WORD EM6,DH6,DT6,DF6
:ITEM 7 .WORD EM7,DH7,DT7,DF7
:ITEM 10 .WORD EM10,DH10,DT10,DF10
:ITEM 11 .WORD EM11,DH11,DT11,DF11
:ITEM 12 .WORD EM12,DH12,DT12,DF12
:ITEM 13 .WORD EM13,DH13,DT13,DF13
:ITEM 14 .WORD EM14,DH14,DT14,DF14
:ITEM 15 .WORD EM15,DH15,DT15,DF15
:ITEM 16 .WORD EM16,DH16,DT16,DF16

999	001620	070012					
1000					: ITEM 17		
1001	001622	041457	066347	070624	. WORD	EM17, DH17, DT17, DF17	
1002	001630	070042					
1003					: ITEM 20		
1004	001632	041534	066216	070644	. WORD	EM20, DH20, DT20, DF20	
1005	001640	070051					
1006					: ITEM 21		
1007	001642	041566	066422	070666	. WORD	EM21, DH21, DT21, DF21	
1008	001650	070051					
1009					: ITEM 22		
1010	001652	041620	066511	070710	. WORD	EM22, DH22, DT22, DF22	
1011	001660	070051					
1012					: ITEM 23		
1013	001662	041701	066216	070726	. WORD	EM23, DH23, DT23, DF23	
1014	001670	070061					
1015					: ITEM 24		
1016	001672	041756	066216	070726	. WORD	EM24, DH24, DT24, DF24	
1017	001700	070061					
1018					: ITEM 25		
1019	001702	042054	066216	070726	. WORD	EM25, DH25, DT25, DF25	
1020	001710	070061					
1021					: ITEM 26		
1022	001712	042141	066216	070726	. WORD	EM26, DH26, DT26, DF26	
1023	001720	070061					
1024					: ITEM 27		
1025	001722	042054	066216	070726	. WORD	EM27, DH27, DT27, DF27	
1026	001730	070061					
1027					: ITEM 30		
1028	001732	042237	066216	070726	. WORD	EM30, DH30, DT30, DF30	
1029	001740	070061					
1030					: ITEM 31		
1031	001742	042311	066216	070726	. WORD	EM31, DH31, DT31, DF31	
1032	001750	070061					
1033					: ITEM 32		
1034	001752	041724	066216	070726	. WORD	EM32, DH32, DT32, DF32	
1035	001760	070061					
1036					: ITEM 33		
1037	001762	042356	066216	070726	. WORD	EM33, DH33, DT33, DF33	
1038	001770	070105					
1039					: ITEM 34		
1040	001772	042401	066216	070726	. WORD	EM34, DH34, DT34, DF34	
1041	002000	070105					
1042					: ITEM 35		
1043	002002	042433	066216	070726	. WORD	EM35, DH35, DT35, DF35	
1044	002010	070105					
1045					: ITEM 36		
1046	002012	042506	066216	070726	. WORD	EM36, DH36, DT36, DF36	
1047	002020	070105					
1048					: ITEM 37		
1049	002022	042554	066216	070726	. WORD	EM37, DH37, DT37, DF37	
1050	002030	070061					
1051					: ITEM 40		
1052	002032	042577	066216	070726	. WORD	EM40, DH40, DT40, DF40	
1053	002040	070061					
1054					: ITEM 41		

1055	002042	042631	066216	070726	.WORD	EM41,DM41,DT41,DF41
1056	002050	070061				
1057					:ITEM 42	
1058	002052	042704	066216	070726	.WORD	EM42,DM42,DT42,DF42
1059	002060	070061				
1060					:ITEM 43	
1061	002062	043035	066216	070726	.WORD	EM43,DM43,DT43,DF43
1062	002070	070061				
1063					:ITEM 44	
1064	002072	043166	066216	070726	.WORD	EM44,DM44,DT44,DF44
1065	002100	070061				
1066					:ITEM 45	
1067	002102	043234	066216	070726	.WORD	EM45,DM45,DT45,DF45
1068	002110	070061				
1069					:ITEM 46	
1070	002112	043307	066216	070726	.WORD	EM46,DM46,DT46,DF46
1071	002120	070105				
1072					:ITEM 47	
1073	002122	043364	066216	070726	.WORD	EM47,DM47,DT47,DF47
1074	002130	070105				
1075					:ITEM 50	
1076	002132	043520	066216	070726	.WORD	EM50,DM50,DT50,DF50
1077	002140	070105				
1078					:ITEM 51	
1079	002142	043573	066216	070726	.WORD	EM51,DM51,DT51,DF51
1080	002150	070105				
1081					:ITEM 52	
1082	002152	043641	066216	070726	.WORD	EM52,DM52,DT52,DF52
1083	002160	070105				
1084					:ITEM 53	
1085	002162	051471	066216	071054	.WORD	EM53,DM53,DT53,DF53
1086	002170	070201				
1087					:ITEM 54	
1088	002172	051522	066216	071054	.WORD	EM54,DM54,DT54,DF54
1089	002200	070201				
1090					:ITEM 55	
1091	002202	051437	066216	071054	.WORD	EM55,DM55,DT55,DF55
1092	002210	070201				
1093					:ITEM 56	
1094	002212	051552	066216	071054	.WORD	EM56,DM56,DT56,DF56
1095	002220	070201				
1096					:ITEM 57	
1097	002222	051643	066216	071054	.WORD	EM57,DM57,DT57,DF57
1098	002230	070201				
1099					:ITEM 60	
1100	002232	051733	066216	071054	.WORD	EM60,DM60,DT60,DF60
1101	002240	070201				
1102					:ITEM 61	
1103	002242	052035	066216	071054	.WORD	EM61,DM61,DT61,DF61
1104	002250	070201				
1105					:ITEM 62	
1106	002252	052231	066216	071054	.WORD	EM62,DM62,DT62,DF62
1107	002260	070201				
1108					:ITEM 63	
1109	002262	052425	066216	071054	.WORD	EM63,DM63,DT63,DF63
1110	002270	070201				

1111					: ITEM 64	
1112	002272	052536	066216	071054	.WORD	EM64, DH64, DT64, DF64
1113	002300	070201				
1114					: ITEM 65	
1115	002302	052655	066216	071054	.WORD	EM65, DH65, DT65, DF65
1116	002310	070201				
1117					: ITEM 66	
1118	002312	052770	066216	071054	.WORD	EM66, DH66, DT66, DF66
1119	002320	070201				
1120					: ITEM 67	
1121	002322	053037	066216	071054	.WORD	EM67, DH67, DT67, DF67
1122	002330	070201				
1123					: ITEM 70	
1124	002332	053122	066216	071054	.WORD	EM70, DH70, DT70, DF70
1125	002340	070233				
1126					: ITEM 71	
1127	002342	053153	066216	071054	.WORD	EM71, DH71, DT71, DF71
1128	002350	070233				
1129					: ITEM 72	
1130	002352	053203	066216	071054	.WORD	EM72, DH72, DT72, DF72
1131	002360	070233				
1132					: ITEM 73	
1133	002362	053203	066216	071054	.WORD	EM73, DH73, DT73, DF73
1134	002370	070233				
1135					: ITEM 74	
1136	002372	053235	066216	071054	.WORD	EM74, DH74, DT74, DF74
1137	002400	070233				
1138					: ITEM 75	
1139	002402	053344	066216	071054	.WORD	EM75, DH75, DT75, DF75
1140	002410	070233				
1141					: ITEM 76	
1142	002412	053435	066216	071054	.WORD	EM76, DH76, DT76, DF76
1143	002420	070233				
1144					: ITEM 77	
1145	002422	053525	066216	071054	.WORD	EM77, DH77, DT77, DF77
1146	002430	070233				
1147					: ITEM 100	
1148	002432	053635	066216	071054	.WORD	EM100, DH100, DT100, DF100
1149	002440	070233				
1150					: ITEM 101	
1151	002442	053720	066216	071054	.WORD	EM101, DH101, DT101, DF101
1152	002450	070233				
1153					: ITEM 102	
1154	002452	054114	066216	071054	.WORD	EM102, DH102, DT102, DF102
1155	002460	070233				
1156					: ITEM 103	
1157	002462	054310	066216	071054	.WORD	EM103, DH103, DT103, DF103
1158	002470	070233				
1159					: ITEM 104	
1160	002472	054422	066216	071054	.WORD	EM104, DH104, DT104, DF104
1161	002500	070233				
1162					: ITEM 105	
1163	002502	054567	066216	071054	.WORD	EM105, DH105, DT105, DF105
1164	002510	070233				
1165					: ITEM 106	
1166	002512	054676	066216	071054	.WORD	EM106, DH106, DT106, DF106

1167	002520	070233					
1168					: ITEM 107		
1169	002522	055005	066216	071054	.WORD	EM107, DH107, DT107, DF107	
1170	002530	070233					
1171					: ITEM 110		
1172	002532	055114	066216	071054	.WORD	EM110, DH110, DT110, DF110	
1173	002540	070233					
1174					: ITEM 111		
1175	002542	043731	066216	070726	.WORD	EM111, DH111, DT111, DF111	
1176	002550	070061					
1177					: ITEM 112		
1178	002552	044006	066216	070726	.WORD	EM112, DH112, DT112, DF112	
1179	002560	070061					
1180					: ITEM 113		
1181	002562	044064	066216	070726	.WORD	EM113, DH113, DT113, DF113	
1182	002570	070061					
1183					: ITEM 114		
1184	002572	044142	066216	070726	.WORD	EM114, DH114, DT114, DF114	
1185	002600	070061					
1186					: ITEM 115		
1187	002602	044221	066557	071000	.WORD	EM115, DH115, DT115, DF115	
1188	002610	070131					
1189					: ITEM 116		
1190	002612	044277	066557	071000	.WORD	EM116, DH116, DT116, DF116	
1191	002620	070131					
1192					: ITEM 117		
1193	002622	044356	066557	071000	.WORD	EM117, DH117, DT117, DF117	
1194	002630	070131					
1195					: ITEM 120		
1196	002632	044514	066557	071000	.WORD	EM120, DH120, DT120, DF120	
1197	002640	070131					
1198					: ITEM 121		
1199	002642	044652	066557	071000	.WORD	EM121, DH121, DT121, DF121	
1200	002650	070131					
1201					: ITEM 122		
1202	002652	045007	066557	071000	.WORD	EM122, DH122, DT122, DF122	
1203	002660	070131					
1204					: ITEM 123		
1205	002662	045144	066216	070726	.WORD	EM123, DH123, DT123, DF123	
1206	002670	070105					
1207					: ITEM 124		
1208	002672	045222	066216	070726	.WORD	EM124, DH124, DT124, DF124	
1209	002700	070105					
1210					: ITEM 125		
1211	002702	045301	066216	070726	.WORD	EM125, DH125, DT125, DF125	
1212	002710	070105					
1213					: ITEM 126		
1214	002712	045357	066216	070726	.WORD	EM126, DH126, DT126, DF126	
1215	002720	070105					
1216					: ITEM 127		
1217	002722	045436	066557	071000	.WORD	EM127, DH127, DT127, DF127	
1218	002730	070155					
1219					: ITEM 130		
1220	002732	045514	066557	071000	.WORD	EM130, DH130, DT130, DF130	
1221	002740	070155					
1222					: ITEM 131		

1223	002742	045573	066557	071000	.WORD	EM131,DH131,DT131,DF131
1224	002750	070155				
1225					; ITEM 132	
1226	002752	045731	066216	070726	.WORD	EM132,DH132,DT132,DF132
1227	002760	070105				
1228					; ITEM 133	
1229	002762	046067	066557	071000	.WORD	EM133,DH133,DT133,DF133
1230	002770	070155				
1231					; ITEM 134	
1232	002772	046225	066557	071000	.WORD	EM134,DH134,DT134,DF134
1233	003000	070155				
1234					; ITEM 135	
1235	003002	046362	066216	070726	.WORD	EM135,DH135,DT135,DF135
1236	003010	070105				
1237					; ITEM 136	
1238	003012	046517	066557	071000	.WORD	EM136,DH136,DT136,DF136
1239	003020	070155				
1240					; ITEM 137	
1241	003022	046654	066557	071000	.WORD	EM137,DH137,DT137,DF137
1242	003030	070131				
1243					; ITEM 140	
1244	003032	046742	066557	071000	.WORD	EM140,DH140,DT140,DF140
1245	003040	070131				
1246					; ITEM 141	
1247	003042	043731	066557	071000	.WORD	EM141,DH141,DT141,DF141
1248	003050	070131				
1249					; ITEM 142	
1250	003052	044006	066557	071000	.WORD	EM142,DH142,DT142,DF142
1251	003060	070131				
1252					; ITEM 143	
1253	003062	047031	066557	071000	.WORD	EM143,DH143,DT143,DF143
1254	003070	070131				
1255					; ITEM 144	
1256	003072	047107	066557	071000	.WORD	EM144,DH144,DT144,DF144
1257	003100	070131				
1258					; ITEM 145	
1259	003102	047166	066216	070726	.WORD	EM145,DH145,DT145,DF145
1260	003110	070061				
1261					; ITEM 146	
1262	003112	047333	066216	070726	.WORD	EM146,DH146,DT146,DF146
1263	003120	070061				
1264					; ITEM 147	
1265	003122	047500	066216	070726	.WORD	EM147,DH147,DT147,DF147
1266	003130	070061				
1267					; ITEM 150	
1268	003132	047644	066216	070726	.WORD	EM150,DH150,DT150,DF150
1269	003140	070061				
1270					; ITEM 151	
1271	003142	050010	066557	071000	.WORD	EM151,DH151,DT151,DF151
1272	003150	070155				
1273					; ITEM 152	
1274	003152	050076	066557	071000	.WORD	EM152,DH152,DT152,DF152
1275	003160	070155				
1276					; ITEM 153	
1277	003162	045144	066557	071000	.WORD	EM153,DH153,DT153,DF153
1278	003170	070155				

1279						;ITEM 154	
1280	003172	045222	066557	071000		.WORD	EM154,DH154,DT154,DF154
1281	003200	070155					
1282							
1283	003202	050165	066557	071000		;ITEM 155	
1284	003210	070155				.WORD	EM155,DH155,DT155,DF155
1285							
1286	003212	050243	066557	071000		;ITEM 156	
1287	003220	070155				.WORD	EM156,DH156,DT156,DF156
1288							
1289	003222	050322	066216	070726		;ITEM 157	
1290	003230	070105				.WORD	EM157,DH157,DT157,DF157
1291							
1292	003232	050467	066557	071000		;ITEM 160	
1293	003240	070155				.WORD	EM160,DH160,DT160,DF160
1294							
1295	003242	050625	066216	070726		;ITEM 161	
1296	003250	070105				.WORD	EM161,DH161,DT161,DF161
1297							
1298	003252	050772	066216	070726		;ITEM 162	
1299	003260	070105				.WORD	EM162,DH162,DT162,DF162
1300							
1301	003262	051136	066557	071000		;ITEM 163	
1302	003270	070155				.WORD	EM163,DH163,DT163,DF163
1303							
1304	003272	051273	066216	070726		;ITEM 164	
1305	003300	070105				.WORD	EM164,DH164,DT164,DF164
1306							
1307	003302	055224	066557	071142		;ITEM 165	
1308	003310	070265				.WORD	EM165,DH165,DT165,DF165
1309							
1310	003312	055310	066557	071142		;ITEM 166	
1311	003320	070265				.WORD	EM166,DH166,DT166,DF166
1312							
1313	003322	055373	066557	071142		;ITEM 167	
1314	003330	070265				.WORD	EM167,DH167,DT167,DF167
1315							
1316	003332	055425	066557	071142		;ITEM 170	
1317	003340	070265				.WORD	EM170,DH170,DT170,DF170
1318							
1319	003342	055513	066557	071142		;ITEM 171	
1320	003350	070265				.WORD	EM171,DH171,DT171,DF171
1321							
1322	003352	055636	066557	071142		;ITEM 172	
1323	003360	070265				.WORD	EM172,DH172,DT172,DF172
1324							
1325	003362	055723	066557	071142		;ITEM 173	
1326	003370	070265				.WORD	EM173,DH173,DT173,DF173
1327							
1328	003372	056071	066557	071142		;ITEM 174	
1329	003400	070265				.WORD	EM174,DH174,DT174,DF174
1330							
1331	003402	056237	066557	071142		;ITEM 175	
1332	003410	070265				.WORD	EM175,DH175,DT175,DF175
1333							
1334	003412	056351	066557	071142		;ITEM 176	
						.WORD	EM176,DH176,DT176,DF176

1335	003420	070317				
1336					; ITEM 177	
1337	003422	056435	066557	071142	.WORD	EM177, DH177, DT177, DF177
1338	003430	070317				
1339					; ITEM 200	
1340	003432	056520	066557	071142	.WORD	EM200, DH200, DT200, DF200
1341	003440	070317				
1342					; ITEM 201	
1343	003442	056552	066557	071142	.WORD	EM201, DH201, DT201, DF201
1344	003450	070317				
1345					; ITEM 202	
1346	003452	056641	066557	071142	.WORD	EM202, DH202, DT202, DF202
1347	003460	070317				
1348					; ITEM 203	
1349	003462	057003	066557	071142	.WORD	EM203, DH203, DT203, DF203
1350	003470	070317				
1351					; ITEM 204	
1352	003472	057145	066557	071142	.WORD	EM204, DH204, DT204, DF204
1353	003500	070317				
1354					; ITEM 205	
1355	003502	057233	066557	071142	.WORD	EM205, DH205, DT205, DF205
1356	003510	070317				
1357					; ITEM 206	
1358	003512	057401	066557	071142	.WORD	EM206, DH206, DT206, DF206
1359	003520	070317				
1360					; ITEM 207	
1361	003522	057547	066716	071252	.WORD	EM207, DH207, DT207, DF207
1362	003530	070361				
1363					; ITEM 210	
1364	003532	057611	067006	071230	.WORD	EM210, DH210, DT210, DF210
1365	003540	070351				
1366					; ITEM 211	
1367	003542	057653	067006	071252	.WORD	EM211, DH211, DT211, DF211
1368	003550	070361				
1369					; ITEM 212	
1370	003552	060017	066716	071324	.WORD	EM212, DH212, DT212, DF212
1371	003560	070405				
1372					; ITEM 213	
1373	003562	060203	066716	071324	.WORD	EM213, DH213, DT213, DF213
1374	003570	070405				
1375					; ITEM 214	
1376	003572	060367	066716	071324	.WORD	EM214, DH214, DT214, DF214
1377	003600	070405				
1378					; ITEM 215	
1379	003602	060553	066716	071324	.WORD	EM215, DH215, DT215, DF215
1380	003610	070405				
1381					; ITEM 216	
1382	003612	060737	067006	071252	.WORD	EM216, DH216, DT216, DF216
1383	003620	070361				
1384					; ITEM 217	
1385	003622	061121	067047	071406	.WORD	EM217, DH217, DT217, DF217
1386	003630	070435				
1387					; ITEM 220	
1388	003632	061163	066656	071252	.WORD	EM220, DH220, DT220, DF220
1389	003640	070361				
1390					; ITEM 221	

1391	003642	061413	067006	071252	.WORD	EM221,DM221,DT221,DF221
1392	003650	070361				
1393					:ITEM 222	
1394	003652	061657	066656	071252	.WORD	EM222,DM222,DT222,DF222
1395	003660	070361				
1396					:ITEM 223	
1397	003662	062110	067006	071252	.WORD	EM223,DM223,DT223,DF223
1398	003670	070361				
1399					:ITEM 224	
1400	003672	062355	066656	071252	.WORD	EM224,DM224,DT224,DF224
1401	003700	070361				
1402					:ITEM 225	
1403	003702	062606	067006	071252	.WORD	EM225,DM225,DT225,DF225
1404	003710	070361				
1405					:ITEM 226	
1406	003712	063053	067122	071324	.WORD	EM226,DM226,DT226,DF226
1407	003720	070405				
1408					:ITEM 227	
1409	003722	063206	067211	071324	.WORD	EM227,DM227,DT227,DF227
1410	003730	070405				
1411					:ITEM 230	
1412	003732	063341	067122	071324	.WORD	EM230,DM230,DT230,DF230
1413	003740	070405				
1414					:ITEM 231	
1415	003742	063475	067211	071230	.WORD	EM231,DM231,DT231,DF231
1416	003750	070405				
1417					:ITEM 232	
1418	003752	057611	067211	071252	.WORD	EM232,DM232,DT232,DF232
1419	003760	070405				
1420					:ITEM 233	
1421	003762	063631	067300	071426	.WORD	EM233,DM233,DT233,DF233
1422	003770	070444				
1423					:ITEM 234	
1424	003772	063670	067341	071462	.WORD	EM234,DM234,DT234,DF234
1425	004000	070461				
1426					:ITEM 235	
1427	004002	063754	067427	071502	.WORD	EM235,DM235,DT235,DF235
1428	004010	070470				
1429					:ITEM 236	
1430	004012	064022	067467	071462	.WORD	EM236,DM236,DT236,DF236
1431	004020	070461				
1432					:ITEM 237	
1433	004022	064055	067341	071462	.WORD	EM237,DM237,DT237,DF237
1434	004030	070461				
1435					:ITEM 240	
1436	004032	064141	067555	071462	.WORD	EM240,DM240,DT240,DF240
1437	004040	070461				
1438					:ITEM 241	
1439	004042	064175	067300	071426	.WORD	EM241,DM241,DT241,DF241
1440	004050	070444				
1441					:ITEM 242	
1442	004052	064300	067555	071462	.WORD	EM242,DM242,DT242,DF242
1443	004060	070461				
1444					:ITEM 243	
1445	004062	064334	067300	071426	.WORD	EM243,DM243,DT243,DF243
1446	004070	070444				

1447					: ITEM 244	
1448	004072	064437	067300	071426	.WORD	EM244, DM244, DT244, DF244
1449	004100	070444				
1450					: ITEM 245	
1451	004102	064476	067300	071426	.WORD	EM245, DM245, DT245, DF245
1452	004110	070444				
1453					: ITEM 246	
1454	004112	043341	066216	070726	.WORD	EM246, DM246, DT246, DF246
1455	004120	070105				
1456					: ITEM 247	
1457	004122	064560	067645	071514	.WORD	EM247, DM247, DT247, DF247
1458	004130	070474				
1459					: ITEM 250	
1460	004132	064614	067712	071532	.WORD	EM250, DM250, DT250, DF250
1461	004140	070474				
1462					: ITEM 251	
1463	004142	064646	067712	071532	.WORD	EM251, DM251, DT251, DF251
1464	004150	070474				
1465					: ITEM 252	
1466	004152	064701	066306	071544	.WORD	EM252, DM252, DT252, DF252
1467	004160	070502				
1468					: ITEM 253	
1469	004162	064772	066216	071556	.WORD	EM253, DM253, DT253, DF253
1470	004170	070506				
1471					: ITEM 254	
1472	004172	065056	066216	071556	.WORD	EM254, DM254, DT254, DF254
1473	004200	070506				
1474					: ITEM 255	
1475	004202	065143	066216	071556	.WORD	EM255, DM255, DT255, DF255
1476	004210	070506				
1477					: ITEM 256	
1478	004212	065231	066216	071556	.WORD	EM256, DM256, DT256, DF256
1479	004220	070506				
1480					: ITEM 257	
1481	004222	065320	066216	071556	.WORD	EM257, DM257, DT257, DF257
1482	004230	070506				
1483					: ITEM 260	
1484	004232	065406	066216	071556	.WORD	EM260, DM260, DT260, DF260
1485	004240	070506				
1486					: ITEM 261	
1487	004242	065475	066216	071556	.WORD	EM261, DM261, DT261, DF261
1488	004250	070506				
1489					: ITEM 262	
1490	004252	065565	066216	071556	.WORD	EM262, DM262, DT262, DF262
1491	004260	070506				
1492					: ITEM 263	
1493	004262	065656	066216	071556	.WORD	EM263, DM263, DT263, DF263
1494	004270	070506				
1495					: ITEM 264	
1496	004272	065743	066216	071556	.WORD	EM264, DM264, DT264, DF264
1497	004300	070506				
1498					: ITEM 265	
1499	004302	066030	066216	071556	.WORD	EM265, DM265, DT265, DF265
1500	004310	070506				
1501					: ITEM 266	
1502	004312	066115	067752	071544	.WORD	EM266, DM266, DT266, DF266

```

1503 004320 070502
1504
1505
1506
1507
1508
1509
1510 004322
1511 000046
1512 000046
1513 000046
1514 000052
1515 000000
1516 004322
1517
1518
1519
1520
1521 004322
1522 000024
1523 000024
1524 000044
1525 000044
1526 004322
1527 004322
1528
1529
1530
1531 004322
1532 004322 000000
1533 004324 001316
1534 004326 000010
1535 004330 000040
1536 004332 000000
1537 004334 000052
1538
1539
1540 004336
1541
1542
1543 004336 012706 001100
1544 004342 005026
1545 004344 022706 001140
1546 004350 001374
1547 004352 012706 001100
1548
1549 004356 012737 033442 000020
1550 004364 012737 000340 000022
1551 004372 012737 033722 000030
1552 004400 012737 000340 000032
1553 004406 012737 035670 000034
1554 004414 012737 000340 000036
1555 004422 012737 035754 000024
1556 004430 012737 000340 000026
1557 004436 016767 026542 026532
1558 004444 005067 174632

```

```

.SBTTL ACT11 HOOKS
*****
:HOOKS REQUIRED BY ACT11
.SX=          ;SAVE PC
.=46
SENDAD       ;;1)SET LOC.46 TO ADDRESS OF SENDAD IN .SECP
.=52
.WORD 0      ;;2)SET LOC.52 TO ZERO
.=SSVPC      ;; RESTORE PC
.SBTTL APT PARAMETER BLOCK
*****
:SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
*****
.SX=          ;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
.=.SX       ;RESET LOCATION COUNTER
*****
:SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
:INTERFACE SPEC.

$APTHD:
$SHIBTS: .WORD 0          ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
$MBAADR: .WORD $MAIL     ;;ADDRESS OF APT MAILBOX (BITS 0-15)
$STMT:   .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 ($CHTAG) AREA
MOV      #SCHTAG,R6          ;;FIRST LOCATION TO BE CLEARED
CLR      (R6)+              ;;CLEAR MEMORY LOCATION
CMP      #SMR,R6           ;;DONE?
BNE     -6                  ;;LOOP BACK IF NO
MOV      #STACK,SP         ;;SETUP THE STACK POINTER
:;INITIALIZE A FEW VECTORS
MOV      #SCOPE,#IOTVEC    ;;IOT VECTOR FOR SCOPE ROUTINE
MOV      #340,#IOTVEC+2    ;;LEVEL 7
MOV      #ERROR,#EMTVEC   ;;EMT VECTOR FOR ERROR ROUTINE
MOV      #340,#EMTVEC+2    ;;LEVEL 7
MOV      #TRAP,#TRAPVEC   ;;TRAP VECTOR FOR TRAP CALLS
MOV      #340,#TRAPVEC+2  ;;LEVEL 7
MOV      #PWARN,#PWAVEC   ;;POWER FAILURE VECTOR
MOV      #340,#PWAVEC+2   ;;LEVEL 7
MOV      #ENDCT,#EOPCT    ;;SETUP END-OF-PROGRAM COUNTER
CLR      $TIMES           ;;INITIALIZE NUMBER OF ITERATIONS

```

```

1553 004450 005067 174630 CLR $ESCAPE ;; CLEAR THE ESCAPE ON ERROR ADDRESS
1554 004454 112767 000001 174433 MOVB #1,$ERMAX ;; ALLOW ONE ERROR PER TEST
1555 ;; INITIALIZE THE "T-BIT" TRAP VECTOR. THEN LOAD LOCATION "$RTRN".
1556 ;; THE "END-OF-PASS" ($EOP) ROUTINE, WITH A "RTI" OR "RTT".
1557 004462 012737 033426 000014 MOV #RTRN,$TBITVEC ;; SET "T" BIT VECTOR TO $RTRN
1558 004470 012737 000340 000016 MOV #340,$TBITVEC+2 ;; LEVEL 7
1559 004476 012767 000002 026722 MOV #RTI,$RTRN ;; SET $RTRN TO A RTI
1560 004504 012737 004532 000010 MOV #65,$RESVEC ;; TRY TO DO A RTT
1561 004512 005046 CLR -(SP) ;; DUMMY PS
1562 004514 012746 004522 MOV #64,-(SP) ;; AND PC
1563 004520 000006 RTT ;; TRY THE RTT
1564 004522 012767 000006 026676 64$: MOV #RTI,$RTRN ;; RTT IS LEGAL--SET $RTRN TO A RTT
1565 004530 000402 BR 56$
1566 004532 062706 000010 65$: ADD #10,SP ;; RTT ILLEGAL--CLEAN OFF THE STACK
1567 004536 012737 000012 000010 66$: MOV #RESVEC+2,$RESVEC ;; RESTORE TRAP CATCHER
1568 004544 005067 026664 CLR $TBIT ;; CLEAR "T" BIT SWITCH
1569 004550 012767 004550 174330 MOV #,$LPADR ;; INITIALIZE THE LOOP ADDRESS FOR SCOPE
1570 004556 012767 004556 174324 MOV #,$LPERR ;; SETUP THE ERROR LOOP ADDRESS
1571 ;; SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
1572 ;; EQUAL TO A "-1" SETUP FOR A SOFTWARE SWITCH REGISTER.
1573 004564 013746 000004 MOV #ERRVEC,-(SP) ;; SAVE ERROR VECTOR
1574 004570 012737 004624 000004 MOV #67,$ERRVEC ;; SET UP ERROR VECTOR
1575 004576 012767 177570 174334 MOV #DSWR,$SWR ;; SETUP FOR A HARDWARE SWICH REGISTER
1576 004604 012767 177570 174330 MOV #DISP,$DISPLAY ;; AND A HARDWARE DISPLAY REGISTER
1577 004612 022777 177777 174320 CMP #-1,$SWR ;; TRY TO REFERENCE HARDWARE SWR
1578 004620 001012 BNE 69$ ;; BRANCH IF NO TIMEOUT TRAP OCCURRED
1579 58$: BR 58$ ;; AND THE HARDWARE SWR IS NOT = -1
1580 004622 000403 BR 58$ ;; BRANCH IF NO TIMEOUT
1581 004624 012716 004632 67$: MOV #68,$(SP) ;; SET UP FOR TRAP RETURN
1582 004630 000002 RTI
1583 004632 012767 000176 174300 68$: MOV #SWREG,$SWR ;; POINT TO SOFTWARE SWR
1584 004640 012767 000174 174274 MOV #DISPREG,$DISPLAY
1585 004646 012637 000004 69$: MOV (SP)+,$ERRVEC ;; RESTORE ERROR VECTOR
1586 1592 CLR $PASS ;; CLEAR PASS COUNT
1587 004652 005067 174446 BITB #APTSIZE,$ENVM ;; TEST USER SIZE UNDER APT
1588 004656 132767 000200 174453 BEQ 70$ ;; YES, USE NON-APT SWITCH
1589 004664 001403 MOV #SWREG,$SWR ;; NO, USE APT SWITCH REGISTER
1590 004666 012767 001340 174244 70$:
1591 .SBTTL TYPE PROGRAM NAME
1592 ;; TYPE THE NAME OF THE PROGRAM IF FIRST PASS
1593 004674 005227 177777 INC #-1 ;; FIRST TIME?
1594 004700 001055 BNE 71$ ;; BRANCH IF NO
1595 004702 022737 033362 000042 CMP #SENDAD,$#42 ;; ACT-11?
1596 004710 001451 BEQ 71$ ;; BRANCH IF YES
1597 004712 104401 004760 TYPE 72$ ;; TYPE ASCIZ STRING
1598 .SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
1599 004716 005737 000042 TST $#42 ;; ARE WE RUNNING UNDER XXDP/ACT?
1600 004722 001012 BNE 73$ ;; BRANCH IF YES
1601 004724 126727 174406 000001 CMPB $ENV,$#1 ;; ARE WE RUNNING UNDER APT?
1602 004732 001406 BEQ 73$ ;; BRANCH IF YES
1603 004734 026727 174200 000176 CMP $SWR,$SWREG ;; SOFTWARE SWITCH REG SELECTED?
1604 004742 001005 BNE 74$ ;; BRANCH IF NO
1605 004744 104405 GTSWR ;; GET SOFT-SWR SETTINGS
1606 004746 000403 BR 74$
1607 004750 112767 000001 174156 73$: MOVB #1,$AUTOB ;; SET AUTO-MODE INDICATOR

```

F03

```

1615 004756
1616 004756 000426
1617
1618 005034
1619
1620 005034
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634 005034 000004
1635
1636
1637
1638 005036
1639 005036 104413
1640 005040 012704 003200
1641 005044 170104
1642 005046 012737 005066 001236
1643 005054 012700 006616
1644 005060 172410
1645 005062 012700 006626
1646 005066 172010
1647 005070 170205
1648 005072 012700 006606
1649 005076 174010
1650 005100 012701 006636
1651 005104 012702 000004
1652 005110 022021
1653 005112 001415
1654 005114 012700 006606
1655 005120 012701 006646
1656 005124 012702 000004
1657 005130 022021
1658 005132 001402
1659 005134 000137 005626
1660 005140 077205
1661 005142 000137 005674
1662 005146 077220
1663 005150 020405
1664 005152 001402
1665 005154 000137 005742
1666
1667
1668
1669
1670
  
```

```

745:
BR 715 ;GET OVER THE ASCIZ
;725: .ASCIZ \CRLF>*DFFPB, FPI1-A 11-34 FPP DIAGNOSTIC PART 2* CRLF
;15:
LOOP:

;*****
;*TEST 1 ROUND\TRUNK TEST
;*
;* THIS IS A TEST OF THE ROUND\TRUNK
;* FLOWS. IN PARTICULAR TWO THINGS ARE TESTED:
;* FIRST A CONDITION IN WHICH ROUNDING
;* RESULTS IN THE NEED FOR RENORMALIZATION, AND
;* SECOND THE PSW CONDITION CODES N AND
;* Z BIT COMBINATIONS
;*****
;ST1: SCOPE
;ROUND AND NORMALIZE TEST

HH1: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
MOV #3200,R4 ;SET FIU, FIV, AND FO
LDFPS R4
MOV #HH2,@STMP2
MOV #HH0,R0 ;SET ACO OPERAND
LDD (R0),ACO
MOV #HH1,R0 ;FSPC
ADD (R0),ACO ;TEST INSTRUCTION
STFPS RS ;GET FPS
MOV #HH0,R0 ;GET THE RESULT
STD ACO(R0)
MOV #HH2,R1 ;IS IT CORRECT
MOV #4,R2
CMP (R0)+,(R1)+
BEQ HH5
MOV #HH0,R0 ;DID FLOW GO
MOV #HH3,R1 ;FROM STATE 663
MOV #4,R2 ;TO 313 INSTEAD
CMP (R0)+,(R1)+ ;OF TO 353
BEQ HH5
JMP @HHERO

HH5: SOB R2,HH4
JMP @HHER1

HH6: SOB R2,HH3
CMP R4,R5 ;FPS CORRECT?
BEQ HH7
JMP @HHER0

;THIS IS A TEST OF THE ABILITY
;OF NORMALIZE TO PRODUCE A ZERO EXP. AND
;OF THE RNT ALGORITHM TO PROPERLY SET THE FPS
  
```

```

1671 005160
1672 005160 104413
1673 005162 012704 043200
1674
1675 005166 170104
1676 005170 012737 005216 001236
1677 005176 012737 006536 00C244
1678 005204 012700 006666
1679 005210 172410
1680 005212 012700 006676
1681 005216 172010
1682 005220 170205
1683 005222 012700 006606
1684 005226 174010
1685 005230 012701 006656
1686 005234 012702 000004
1687 005240 022021
1688 005242 001402
1689 005244 000137 006010
1690 005250 077205
1691 005252 052704 100004
1692 005256 020405
1693 005260 001402
1694 005262 000137 006056
1695
1696
1697 005266
1698 005266 104413
1699
1700 005270 012704 043200
1701 005274 170104
1702 005276 012737 005316 001236
1703 005304 012700 006716
1704 005310 172410
1705 005312 012700 006726
1706 005316 172010
1707 005320 170205
1708 005322 012700 006606
1709 005326 174010
1710 005330 012701 006706
1711 005334 012702 000004
1712 005340 022021
1713 005342 001415
1714 005344 012700 006606
1715 005350 012701 006656
1716 005354 012702 000004
1717 005360 022021
1718 005362 001402
1719 005364 000137 006124
1720 005370 077205
1721 005372 000137 006172
1722 005376 077220
1723 005400 052704 100014
1724 005404 020405
1725 005406 001402
1726 005410 000137 006240

HH7:
LPERR ;SET JP THE LOOP ON ERROR ADDRESS.
MOV #043200,R4 ;SET FIV,FIV,AND FD
;FID
LDFPS R4
MOV #HH8,@STMP2 ;IN CASE UNDERFLOW
MOV #HHTRAP,@FPVECT ;TRAP OCCURS
MOV #HHP5,R0 ;SET ACO OPERAND
LDD (R0),ACO
MOV #HHP6,R0 ;FSPC
ADD (R0),ACO ;TEST INSTRUCTION
STFPS R5 ;GET FPS
MOV #HHDATA,R0 ;GET THE RESULT
STD ACO,(R0)
MOV #HHP4,R1 ;IS IT CORRECT
MOV #4,R2
HH9: CMP (R0)+,(R1)+
BEQ HH10
JMP @HHER2
HH10: SOB R2,HH9
BIS #100004,R4 ;FPS CORRECT?
CMP R4,R5
BEQ HH11
JMP @HHER3
;THIS IS A TEST OF THE RNT ALGORITHM'S
;ABILITY TO SET BOTH N AND Z ON A - C RESULT.
HH11:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
;SET FIV, FIV, AND FD
MOV #043200,R4
LDFPS R4
MOV #HH12,@STMP2
MOV #HHP8,R0 ;SET ACO OPERAND
LDD (R0),ACO
MOV #HHP9,R0 ;FSPC
ADD (R0),ACO ;TEST INSTRUCTION
STFPS R5 ;GET FPS
MOV #HHDATA,R0 ;GET THE RESULT
STD ACO,(R0)
MOV #HHP7,R1 ;IS IT CORRECT
MOV #4,R2
HH13: CMP (R0)+,(R1)+
BEQ HH16
MOV #HHDATA,R0
MOV #HHP4,R1
MOV #4,R2
HH14: CMP (R0)+,(R1)+
BEQ HH15
JMP @HHER4
HH15: SOB R2,HH14
JMP @HHER5
HH16: SOB R2,HH13
BIS #100014,R4 ;FPS CORRECT?
CMP R4,R5
BEQ HH17
JMP @HHER6
    
```


H03

```

1751 ;TEST THAT CC ARE CLEARED BY R/T
1752 HH17:
1753 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
1754 MOV #00200,R4 ;SET FIV, FIV, AND FD
1755 LDFPS R4
1756 MOV #HH18,@#STMP2
1757 MOV #FPSPUR,@#FPVECT
1758 MOV #HHP8,R0 ;SET ACC OPERAND
1759 LDD (R0),ACC
1760 MOV #HHP8,R0 ;FSPC
1761 HH18: ADDD (R0),ACC ;TEST INSTRUCTION
1762 STFPS R5 ;GET FPS
1763 MOV #HHDATA,R0 ;GET THE RESULT
1764 STD ACC,(R0)
1765 MOV #HHP10,R1 ;IS IT CORRECT
1766 HH19: MOV #4,R2
1767 CMP (R0)+,(R1)+
1768 BEQ HH20
1769 JMP @#HHER7
1770 HH20: SOB R2,HH19
1771 BIS #00000,R4 ;FPS CORRECT?
1772 CMP R4,R5
1773 BEQ HH21
1774 JMP @#HHER8
1775 ;TEST THAT N IS SET BY R/T
1776 HH21:
1777 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
1778 MOV #3200,R4 ;SET FIV, FIV, AND FD
1779 LDFPS R4
1780 MOV #HH22,@#STMP2
1781 MOV #HHP5,R0 ;SET ACC OPERAND
1782 LDD (R0),ACC
1783 MOV #HHP5,R0 ;FSPC
1784 HH22: ADDD (R0),ACC ;TEST INSTRUCTION
1785 STFPS R5 ;GET FPS
1786 MOV #HHDATA,R0 ;GET THE RESULT
1787 STD ACC,(R0)
1788 MOV #HHP11,R1 ;IS IT CORRECT
1789 HH23: MOV #4,R2
1790 CMP (R0)+,(R1)+
1791 BEQ HH24
1792 JMP @#HHER9
1793 HH24: SOB R2,HH23
1794 BIS #10,R4 ;FPS CORRECT?
1795 CMP R4,R5
1796 BEQ HH25
1797 JMP @#HHER10
1798 HH25: JMP @#HHDONE
1799 HHERO:
1800 MOV R5,@#STMP10
1801 MOV R4,@#STMP11
1802 MOV #HHP1,@#STMP3
1803 MOV #HHP0,@#STMP4
1804 MOV #HHDATA,@#STMP5
1805 MOV #HHP2,@#STMP6
1806 1$: ERROR 207
  
```

1783	005670	000137	006756		JMP	2#HHDONE
1784	005674			HHER1:		
1785	005674	010537	001252		MOV	R5,2#STMP10
1786	005700	010437	001254		MOV	R4,2#STMP11
1787	005704	012737	006626	001240	MOV	#HHP1,2#STMP3
1788	005712	012737	006616	001242	MOV	#HHP0,2#STMP4
1789	005720	012737	006606	001244	MOV	#HHDAT0,2#STMP5
1790	005726	012737	006636	001246	MOV	#HHP2,2#STMP6
1791	005734	104211		1\$:	ERROR	211
1792	005736	000137	006756		JMP	2#HHDONE
1793	005742			HHER00:		
1794	005742	010537	001252		MOV	R5,2#STMP10
1795	005746	010437	001254		MOV	R4,2#STMP11
1796	005752	012737	006626	001240	MOV	#HHP1,2#STMP3
1797	005760	012737	006616	001242	MOV	#HHP0,2#STMP4
1798	005766	012737	006606	001244	MOV	#HHDAT0,2#STMP5
1799	005774	012737	006636	001246	MOV	#HHP2,2#STMP6
1800	006002	104210		1\$:	ERROR	210
1801	006004	000137	006756		JMP	2#HHDONE
1802	006010			HHER2:		
1803	006010	010537	001252		MOV	R5,2#STMP10
1804	006014	010437	001254		MOV	R4,2#STMP11
1805	006020	012737	006676	001240	MOV	#HHP6,2#STMP3
1806	006026	012737	006666	001242	MOV	#HHP5,2#STMP4
1807	006034	012737	006606	001244	MOV	#HHDAT0,2#STMP5
1808	006042	012737	006656	001246	MOV	#HHP4,2#STMP6
1809	006050	104207		1\$:	ERROR	207
1810	006052	000137	006756		JMP	2#HHDONE
1811	006056			HHER3:		
1812	006056	010537	001252		MOV	R5,2#STMP10
1813	006062	010437	001254		MOV	R4,2#STMP11
1814	006066	012737	006676	001240	MOV	#HHP6,2#STMP3
1815	006074	012737	006666	001242	MOV	#HHP5,2#STMP4
1816	006102	012737	006606	001244	MOV	#HHDAT0,2#STMP5
1817	006110	012737	006656	001246	MOV	#HHP4,2#STMP6
1818	006116	104214		1\$:	ERROR	214
1819	006120	000137	006756		JMP	2#HHDONE
1820	006124			HHER4:		
1821	006124	010537	001252		MOV	R5,2#STMP10
1822	006130	010437	001254		MOV	R4,2#STMP11
1823	006134	012737	006726	001240	MOV	#HHP9,2#STMP3
1824	006142	012737	006716	001242	MOV	#HHP8,2#STMP4
1825	006150	012737	006606	001244	MOV	#HHDAT0,2#STMP5
1826	006156	012737	006706	001246	MOV	#HHP7,2#STMP6
1827	006164	104207		1\$:	ERROR	207
1828	006166	000137	006756		JMP	2#HHDONE
1829	006172			HHER5:		
1830	006172	010537	001252		MOV	R5,2#STMP10
1831	006176	010437	001254		MOV	R4,2#STMP11
1832	006202	012737	006726	001240	MOV	#HHP9,2#STMP3
1833	006210	012737	006716	001242	MOV	#HHP8,2#STMP4
1834	006216	012737	006606	001244	MOV	#HHDAT0,2#STMP5
1835	006224	012737	006706	001246	MOV	#HHP7,2#STMP6
1836	006232	104216		1\$:	ERROR	216
1837	006234	000137	006756		JMP	2#HHDONE
1838	006240			HHER6:		

1839	006240	010537	001252		MOV	R5, @STMP10	
1840	006244	010437	001254		MOV	R4, @STMP11	
1841	006250	012737	006726	001240	MOV	#HHP9, @STMP3	
1842	006256	012737	006716	001242	MOV	#HHP8, @STMP4	
1843	006264	012737	006606	001244	MOV	#HDATA0, @STMP5	
1844	006272	012737	006706	001246	MOV	#HHP7, @STMP6	
1845	006300	104215			IS:	ERROR	215
1846	006302	000137	006756		JMP	@HHDONE	
1847	006306				HHER7:		
1848	006306	010537	001252		MOV	R5, @STMP10	
1849	006312	010437	001254		MOV	R4, @STMP11	
1850	006316	012737	006716	001240	MOV	#HHP8, @STMP3	
1851	006324	012737	006716	001242	MOV	#HHP8, @STMP4	
1852	006332	012737	006606	001244	MOV	#HDATA0, @STMP5	
1853	006340	012737	006736	001246	MOV	#HHP10, @STMP6	
1854	006346	104207			IS:	ERROR	207
1855	006350	000137	006756		JMP	@HHDONE	
1856	006354				HHER8:		
1857	006354	010537	001252		MOV	R5, @STMP10	
1858	006360	010437	001254		MOV	R4, @STMP11	
1859	006364	012737	006716	001240	MOV	#HHP8, @STMP3	
1860	006372	012737	006716	001242	MOV	#HHP8, @STMP4	
1861	006400	012737	006606	001244	MOV	#HDATA0, @STMP5	
1862	006406	012737	006736	001246	MOV	#HHP10, @STMP6	
1863	006414	104212			IS:	ERROR	212
1864	006416	000137	006756		JMP	@HHDONE	
1865	006422				HHER9:		
1866	006422	010537	001252		MOV	R5, @STMP10	
1867	006426	010437	001254		MOV	R4, @STMP11	
1868	006432	012737	006666	001240	MOV	#HHP5, @STMP3	
1869	006440	012737	006666	001242	MOV	#HHP5, @STMP4	
1870	006446	012737	006606	001244	MOV	#HDATA0, @STMP5	
1871	006454	012737	006746	001246	MOV	#HHP11, @STMP6	
1872	006462	104207			IS:	ERROR	207
1873	006464	000137	006756		JMP	@HHDONE	
1874	006470				HHER10:		
1875	006470	010537	001252		MOV	R5, @STMP10	
1876	006474	010437	001254		MOV	R4, @STMP11	
1877	006500	012737	006666	001240	MOV	#HHP5, @STMP3	
1878	006506	012737	006666	001242	MOV	#HHP5, @STMP4	
1879	006514	012737	006606	001244	MOV	#HDATA0, @STMP5	
1880	006522	012737	006746	001246	MOV	#HHP11, @STMP6	
1881	006530	104213			IS:	ERROR	213
1882	006532	000137	006756		JMP	@HHDONE	
1883	006536	013703	001236		HHTRAP:	MOV	@STMP2, R3
1884	006542	062703	000002		ADD	#2, R3	: WAS THE TRAP TO 244
1885	006546	020316			CMP	R3, (SP)	: ON THE INSTRUCTION
1886	006550	001402			BEQ	IS	: BEING TESTED?
1887	006552	000137	036614		JMP	@FSPUR	
1888	006556	011637	001236		IS:	MOV	(SP), @STMP2
1889							: FAILURE OF FPS INTERRUPT
1890	006562	022626			CMP	(SP)+, (SP)+	: DISABLE BIT (FID=1)
1891	006564	170201			STFPS	R1	: TO INHIBIT TRAP.
1892	006566	010137	001240		MOV	R1, @STMP3	
1893	006572	170301			STST	R1	
1894	006574	010137	001242		MOV	R1, @STMP4	

11

K03

MAINDEC-11-DEFPB-A
DEFPB.A.P11

01-NOV-76

PDP 11 34 FPP DIAGNOSTIC PART 2 MACY11 27(1006) 21:06

T1 ROUND-TRUNK TEST

01-NOV-76 21:12 PAGE 36

1895 006600 104217
 1896 006602 000137
 1897 006606 000000
 1898 006610 000000
 1899 006612 000000
 1900 006614 000000
 1901 006616 000452
 1902 006620 125252
 1903 006622 125252
 1904 006624 125253
 1905 006626 000252
 1906 006630 125252
 1907 006632 125252
 1908 006634 125252
 1909 006636 000600
 1910 006640 000000
 1911 006642 000000
 1912 006644 000000
 1913 006646 000400
 1914 006650 000000
 1915 006652 000000
 1916 006654 000000
 1917 006656 000000
 1918 006660 000000
 1919 006662 000000
 1920 006664 000000
 1921 006666 100200
 1922 006670 000000
 1923 006672 000000
 1924 006674 000000
 1925 006676 000300
 1926 006700 000000
 1927 006702 000000
 1928 006704 000000
 1929 006706 100000
 1930 006710 000000
 1931 006712 000000
 1932 006714 000000
 1933 006716 000200
 1934 006720 000000
 1935 006722 000000
 1936 006724 000000
 1937 006726 100300
 1938 006730 000000
 1939 006732 000000
 1940 006734 000000
 1941 006736 000400
 1942 006740 000000
 1943 006742 000000
 1944 006744 000000
 1945 006746 100400
 1946 006750 000000
 1947 006752 000000
 1948 006754 000000
 1949 006756
 1950 006756 104412

00E756

25: ERROR 217
 JMP 0#HHDONE
 HMDATO: 0
 0
 0
 0
 HHP0: 452
 125252
 125252
 125253
 HHP1: 252
 125252
 125252
 125252
 HHP2: 600
 0
 0
 0
 HHP3: 400
 0
 0
 0
 0
 HHP4: 0
 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 + HHP6

;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 WAS
:THE USER TYPED CONTROL G?).

1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006

006760 000004

006762
006762 104413
006764 012704 000200
006770 170104
006772 012737 007020 001236
007000 012737 010052 000244
007006 012700 011624
007012 172410
007014 012700 011624
007020 172010
007022 170205
007024 012700 011554
007030 174010
007032 012701 011634
007036 012702 000004
007042 022021
007044 001402
007046 000137 010150
007052 077205
007054 052704 000006
007060 020405
007062 001402
007064 000137 010216

007070
007070 104413
007072 012704 001200
007076 170104
007100 012737 007126 001236
007106 012737 007144 000244
007114 012700 011624
007120 172410

*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).
*

†S2: 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 #GGPS,AC ;SET ACD OPERAND
LDD (R0),ACL
MOV #GGPS,R0 ;FSRC
GG2: ADD (R0),ACU ;TEST INSTRUCTION
STFPS RS ;GET FPS
MOV #GGDAT0,R0 ;GET THE RESULT
STD ACD,(R0)
MOV #GGPS,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 #GGPS,R0 ;SET ACD OPERAND
LDD (R0),ACD

```

2007 007122 012700 011624      MOV      #GGP5,R0      ;FSPC
2008 007126 172010      ADDO    (R0),AC0      ;TEST INSTRUCTION
2009 007130 170000      CFCC
2010 007132 012700 011554      MOV      #GGDAT0,R0
2011 007136 174010      STD     AC0,(R0)
2012 007140 000137 010264      JMP     @#GGER3
2013 007144 013703 001236      GG7:    MOV     @#STMP2,R3
2014 007150 062703 000002      ADD     #2,R3
2015 007154 020316      CMP     R3,(SP)
2016 007156 001402      BEQ     1$
2017 007160 000137 036614      JMP     @#FPSPUR
2018 007164 011637 001236      1$:    MOV     (SP),@#STMP2
2019 007170 022626      CMP     (SP)+,(SP)+
2020 007172 170205      STFPS  R5
2021 007174 012700 011554      MOV     #GGDAT0,R0      ;GET THE RESULT
2022 007200 174010      STD     AC0,(R0)
2023 007202 012701 011634      MOV     #GGP6,R1      ;IS IT CORRECT
2024 007206 012702 000004      MOV     #4,R2
2025 007212 022021      GG8:    CMP     (R0)+,(R1)+
2026 007214 001402      BEQ     GG9
2027 007216 000137 010332      JMP     @#GGER4
2028 007222 077205      GG9:    SOB     R2,GG8
2029 007224 052704 100006      BIS     #100006,R4
2030 007230 020405      CMP     R4,R5      ;FPS CORRECT?
2031 007232 001402      BEQ     1$
2032 007234 000137 010402      JMP     @#GGER6
2033 007240 012704 000010      1$:    MOV     #10,R4
2034      ;CHECK FEC
2035 007244 170305      STST   R5
2036 007246 020405      CMP     R4,R5
2037 007250 001402      BEQ     GG10
2038 007252 000137 010334      JMP     @#GGER5
2039      ;CHECK UNDER FLOW CONDITION WITH
2040      ;TRAPS DISABLED (FIU = 0)
2041      GG10:
2042 007256 104413      LPERR
2043 007260 012704 000200      MOV     #0200,R4      ;SET UP THE LOOP ON ERROR ADDRESS.
2044 007264 170104      LDFPS  R4      ;SET FIU, FIV. AND FD
2045 007266 012737 007314 001236      MOV     #GG11,@#STMP2
2046 007274 012737 010450 000244      MOV     #GGER7,@#FPVEXT
2047 007302 012700 011574      MOV     #GGP2,R0
2048 007306 172410      LDO    (R0),AC0      ;SET AC0 OPERAND
2049 007310 012700 011604      MOV     #GGP3,R0      ;FSRC
2050 007314 172010      GG11:  ADDO   (R0),AC0      ;TEST INSTRUCTION
2051 007316 170205      STFPS  R5      ;GET FPS
2052 007320 012700 011554      MOV     #GGDAT0,R0      ;GET THE RESULT
2053 007324 174010      STD     AC0,(R0)
2054 007326 012701 011634      MOV     #GGP6,R1      ;IS IT CORRECT
2055 007332 012702 000004      MOV     #4,R2
2056 007336 022021      GG12:  CMP     (R0)+,(R1)+
2057 007340 001402      BEQ     GG13
2058 007342 000137 010546      JMP     @#GGER8
2059 007346 077205      GG13:  SOB     R2,GG12
2060 007350 052704 000004      BIS     #4,R4      ;FPS CORRECT?
2061 007354 020405      CMP     R4,R5
2062 007356 001402      BEQ     GG14
  
```


2119	007612	170205		STFPS	R5		:GET FPS
2120	007614	012700	011554	MOV	#GGDAT0,R0		:GET THE RESULT
2121	007620	174010		STD	AC0,(R0)		
2122	007622	012701	011634	MOV	#GGP6,R1		:IS IT CORRECT
2123	007626	012702	000004	MOV	#4,R2		
2124	007632	022021		GG21: CMP	(R0)+,(R1)+		
2125	007634	001402		BEQ	GG22		
2126	007636	000137	011210	JMP	#GGER15		
2127	007642	077205		GG22: SOB	R2,GG21		
2128	007644	052704	000004	BIS	#4,R4		:FPS CORRECT?
2129	007650	020405		CMP	R4,R5		
2130	007652	001402		BEQ	GG23		
2131	007654	000137	011256	JMP	#GGER16		
2132							:CHECK UNDERFLOW CONDITION WITH TRAP
2133							:ENABLED (FIU = 1)
2134	007660			GG23:			
2135	007660	104413		LPERR			:SET UP THE LOOP ON ERROR ADDRESS.
2136	007662	012704	002200	MOV	#2200,R4		:SET FIU, FIV, AND FD
2137	007666	170104		LDFPS	R4		
2138	007670	012737	007716	MOV	#GG24,#STMP2		
2139	007676	012737	007734	MOV	#GG25,#FPVECT		
2140	007704	012700	011574	MOV	#GGP2,R0		:SET ACC OPERAND
2141	007710	172410		LDD	(R0),AC0		
2142	007712	012700	011654	MOV	#GGP8,R0		:FSRC
2143	007716	172010		GG24: ADD	(R0),AC0		:TEST INSTRUCTION
2144	007720	170000		CFCC			
2145	007722	012700	011554	MOV	#GGDAT0,R0		
2146	007726	174010		STD	AC0,(R0)		
2147	007730	000137	011324	JMP	#GGER17		
2148	007734	013700	001236	GG25: MOV	#STMP2,R0		
2149	007740	062700	000002	AC0	R2,R0		
2150	007744	020016		CMP	R0,(SP)		
2151	007746	001402		BEQ	1\$		
2152	007750	000137	036614	JMP	#FPSPUR		
2153	007754	011637	001236	1\$: MOV	(SP),#STMP2		
2154	007760	022626		CMP	(SP)+,(SP)+		
2155	007762	170205		STFPS	R5		:GET FPS
2156	007764	012700	011554	MOV	#GGDAT0,R0		:GET THE RESULT
2157	007770	174010		STD	AC0,(R0)		
2158	007772	012701	011664	MOV	#GGP9,R1		:IS IT CORRECT
2159	007776	012702	000004	MOV	#4,R2		
2160	010002	022021		GG26: CMP	(R0)+,(R1)+		
2161	010004	001402		BEQ	GG27		
2162	010006	000137	011372	JMP	#GGER18		
2163	010012	077205		GG27: SOB	R2,GG26		
2164	010014	052704	100004	BIS	#100004,R4		
2165	010020	020405		CMP	R4,R5		:FPS CORRECT?
2166	010022	001402		BEQ	1\$		
2167	010024	000167	001456	JMP	GGER20		
2168	010030	012704	000012	1\$: MOV	#12,R4		
2169							:CHECK FEC
2170	010034	170305		STST	R5		
2171	010036	020405		CMP	R4,R5		
2172	010040	001402		BEQ	GG28		
2173	010042	000167	001372	JMP	GGER19		
2174							

2175	010046	000137	011674		GG28:	JMP	2#GGDONE
2176							
2177	010052	013701	001236		GGER0:	MOV	2#STMP2,R.
2178	010056	062701	000002			ADD	2,R1
2179	010062	020116				CMP	R1,(SP)
2180	010064	001402				BEG	10\$
2181	010066	000137	036614		5\$:	JMP	2#FSPUR
2182	010072				10\$:		
2183	010072	170301				STST	R1
2184	010074	020127	000010			CMP	R1,#10
2185	010100	001372				BNE	5\$
2186	010102	022626				CMP	(SP)+,(SP)+
2187	010104	012700	011554			MOV	2#GGDAT0,R0
2188	010110	174010				STD	ACD,(R0)
2189	010112	012737	011624	001240		MOV	2#GGP5,2#STMP3
2190	010120	012737	011624	001242		MOV	2#GGP5,2#STMP4
2191	010126	012737	011554	001244		MOV	2#GGDAT0,2#STMP5
2192	010134	012737	011634	001246		MOV	2#GGP6,2#STMP6
2193	010142	104220			1\$:	ERROR	220
2194	010144	000137	011674			JMP	2#GGDONE
2195							
2196	010150				GGER1:		
2197	010150	010537	001252			MOV	R5,2#STMP10
2198	010154	010437	001254			MOV	R4,2#STMP11
2199	010160	012737	011624	001240		MOV	2#GGP5,2#STMP3
2200	010166	012737	011624	001242		MOV	2#GGP5,2#STMP4
2201	010174	012737	011554	001244		MOV	2#GGDAT0,2#STMP5
2202	010202	012737	011634	001246		MOV	2#GGP6,2#STMP6
2203	010210	104207			1\$:	ERROR	207
2204	010212	000137	011674			JMP	2#GGDONE
2205							
2206	010216				GGER2:		
2207	010216	010537	001252			MOV	R5,2#STMP10
2208	010222	010437	001254			MOV	R4,2#STMP11
2209	010226	012737	011624	001240		MOV	2#GGP5,2#STMP3
2210	010234	012737	011624	001242		MOV	2#GGP5,2#STMP4
2211	010242	012737	011554	001244		MOV	2#GGDAT0,2#STMP5
2212	010250	012737	011634	001246		MOV	2#GGP6,2#STMP6
2213	010256	104232			1\$:	ERROR	232
2214	010260	000137	011674			JMP	2#GGDONE
2215							
2216	010264				GGER3:		
2217	010264	010537	001252			MOV	R5,2#STMP10
2218	010270	010437	001254			MOV	R4,2#STMP11
2219	010274	012737	011624	001240		MOV	2#GGP5,2#STMP3
2220	010302	012737	011624	001242		MOV	2#GGP5,2#STMP4
2221	010310	012737	011554	001244		MOV	2#GGDAT0,2#STMP5
2222	010316	012737	011634	001246		MOV	2#GGP6,2#STMP6
2223	010324	104221			1\$:	ERROR	221
2224	010326	000137	011674			JMP	2#GGDONE
2225							
2226	010332	000706			GGER4:	BR	GGER1
2227							
2228	010334				GGER5:		
2229	010334	010537	001252			MOV	R5,2#STMP10
2230	010340	010437	001254			MOV	R4,2#STMP11

2231	010344	012737	011624	001240		MOV	#GGP5, @STMP3
2232	010352	012737	011624	001242		MOV	#GGP5, @STMP4
2233	010360	012737	011554	001244		MOV	#GGDAT0, @STMP5
2234	010366	012737	011634	001246		MOV	#GGP6, @STMP6
2235	010374	104226			15:	ERROR	226
2236	010376	000137	011674			JMP	@GGDONE
2237							
2238	010402				GGER6:		
2239	010402	010537	001252			MOV	R5, @STMP10
2240	010406	010437	001254			MOV	R4, @STMP11
2241	010412	012737	011624	001240		MOV	#GGP5, @STMP3
2242	010420	012737	011624	001242		MOV	#GGP5, @STMP4
2243	010426	012737	011554	001244		MOV	#GGDAT0, @STMP5
2244	010434	012737	011634	001246		MOV	#GGP6, @STMP6
2245	010442	104227			15:	ERROR	227
2246	010444	000137	011674			JMP	@GGDONE
2247							
2248	010450	013701	001236		GGER7:	MOV	@STMP2, R1
2249	010454	062701	000002			ADD	#2, R1
2250	010460	020116				CMP	R1, (SP)
2251	010462	001402				BEQ	105
2252	010464	000137	036614		55:	JMP	@FFSPUR
2253	010470				105:		
2254	010470	170301				STST	R1
2255	010472	020127	000012			CMP	R1, #12
2256	010476	001372				BNE	55
2257	010500	022626				CMP	(SP)+, (SP)+
2258	010502	012700	011554			MOV	#GGDAT0, R0
2259	010506	174010				STD	ACD, (R0)
2260	010510	012737	011604	001240		MOV	#GGP3, @STMP3
2261	010516	012737	011574	001242		MOV	#GGP2, @STMP4
2262	010524	012737	011554	001244		MOV	#GGDAT0, @STMP5
2263	010532	012737	011634	001246		MOV	#GGP6, @STMP6
2264	010540	104224			15:	ERROR	224
2265	010542	000137	011674			JMP	@GGDONE
2266							
2267	010546				GGER8:		
2268	010546	010537	001252			MOV	R5, @STMP10
2269	010552	010437	001254			MOV	R4, @STMP11
2270	010556	012737	011604	001240		MOV	#GGP3, @STMP3
2271	010564	012737	011574	001242		MOV	#GGP2, @STMP4
2272	010572	012737	011554	001244		MOV	#GGDAT0, @STMP5
2273	010600	012737	011634	001246		MOV	#GGP6, @STMP6
2274	010606	104207			15:	ERROR	207
2275	010610	000137	011674			JMP	@GGDONE
2276							
2277	010614				GGER9:		
2278	010614	010537	001252			MOV	R5, @STMP10
2279	010620	010437	001254			MOV	R4, @STMP11
2280	010624	012737	011604	001240		MOV	#GGP3, @STMP3
2281	010632	012737	011574	001242		MOV	#GGP2, @STMP4
2282	010640	012737	011554	001244		MOV	#GGDAT0, @STMP5
2283	010646	012737	011634	001246		MOV	#GGP6, @STMP6
2284	010654	104232			15:	ERROR	232
2285	010656	000137	011674			JMP	@GGDONE
2286							

2287	010662					GGER10:	
2288	010662	010537	001252			MOV	R5, @STMP10
2289	010666	010437	001254			MOV	R4, @STMP11
2290	010672	012737	011604	001240		MOV	@GGP3, @STMP3
2291	010700	012737	011574	001242		MOV	@GGP2, @STMP4
2292	010706	012737	011554	001244		MOV	@GGDAT0, @STMP5
2293	010714	012737	011644	001246		MOV	@GGP7, @STMP6
2294	010722	104225			1S:	ERROR	225
2295	010724	000137	011674			JMP	@GGDONE
2296							
2297	010730					GGER11:	
2298	010730	010537	001252			MOV	R5, @STMP10
2299	010734	010437	001254			MOV	R4, @STMP11
2300	010740	012737	011604	001240		MOV	@GGP3, @STMP3
2301	010746	012737	011574	001242		MOV	@GGP2, @STMP4
2302	010754	012737	011554	001244		MOV	@GGDAT0, @STMP5
2303	010762	012737	011644	001246		MOV	@GGP7, @STMP6
2304	010770	104207			1S:	ERROR	207
2305	010772	000137	011674			JMP	@GGDONE
2306							
2307	010776					GGER12:	
2308	010776	010537	001252			MOV	R5, @STMP10
2309	011002	010437	001254			MOV	R4, @STMP11
2310	011006	012737	011604	001240		MOV	@GGP3, @STMP3
2311	011014	012737	011574	001242		MOV	@GGP2, @STMP4
2312	011022	012737	011554	001244		MOV	@GGDAT0, @STMP5
2313	011030	012737	011644	001246		MOV	@GGP7, @STMP6
2314	011036	104231			1S:	ERROR	231
2315	011040	000137	011674			JMP	@GGDONE
2316							
2317	011044					GGER13:	
2318	011044	010537	001252			MOV	R5, @STMP10
2319	011050	010437	001254			MOV	R4, @STMP11
2320	011054	012737	011604	001240		MOV	@GGP3, @STMP3
2321	011062	012737	011574	001242		MOV	@GGP2, @STMP4
2322	011070	012737	011554	001244		MOV	@GGDAT0, @STMP5
2323	011076	012737	011644	001246		MOV	@GGP7, @STMP6
2324	011104	104230			1S:	ERROR	230
2325	011106	000137	011674			JMP	@GGDONE
2326							
2327	011112	013701	001236			GGER14:	MOV @STMP2, R1
2328	011116	062701	000002			ADD	#2, R1
2329	011122	020116				CMP	R1, (SP)
2330	011124	001402				BEQ	10\$
2331	011126	000137	036614		5S:	JMP	@FPSPUR
2332	011132				10\$:		
2333	011132	170301				STST	R1
2334	011134	020127	000012			CMP	R1, #12
2335	011140	001372				BNE	5\$
2336	011142	022626				CMP	(SP)+, (SP)+
2337	011144	012700	011554			MOV	@GGDAT0, R0
2338	011150	174010				STD	AC0, (R0)
2339							
2340	011152	012737	011564	001240		MOV	@GGP1, @STMP3
2341	011160	012737	011604	001242		MOV	@GGP3, @STMP4
2342	011166	012737	011554	001244		MOV	@GGDAT0, @STMP5

2343	011174	012737	011634	001246	MOV	#GGP6,2#STMP6
2344	011202	104222			15: ERROR	222
2345	011204	000137	011674		JMP	2#GGDONE
2346						
2347	011210				GGER15:	
2348	011210	010537	001252		MOV	R5,2#STMP10
2349	011214	010437	001254		MOV	R4,2#STMP11
2350	011220	012737	011574	001240	MOV	#GGP2,2#STMP3
2351	011226	012737	011654	001242	MOV	#GGP8,2#STMP4
2352	011234	012737	011554	001244	MOV	#GGDAT0,2#STMP5
2353	011242	012737	011634	001246	MOV	#GGP6,2#STMP6
2354	011250	104207			15: ERROR	207
2355	011252	000137	011674		JMP	2#GGDONE
2356						
2357	011256				GGER16:	
2358	011256	010537	001252		MOV	R5,2#STMP10
2359	011262	010437	001254		MOV	R4,2#STMP11
2360	011266	012737	011574	001240	MOV	#GGP2,2#STMP3
2361	011274	012737	011654	001242	MOV	#GGP8,2#STMP4
2362	011302	012737	011554	001244	MOV	#GGDAT0,2#STMP5
2363	011310	012737	011634	001246	MOV	#GGP6,2#STMP6
2364	011316	104232			15: ERROR	232
2365	011320	000137	011674		JMP	2#GGDONE
2366						
2367	011324				GGER17:	
2368	011324	010537	001252		MOV	R5,2#STMP10
2369	011330	010437	001254		MOV	R4,2#STMP11
2370	011334	012737	011574	001240	MOV	#GGP2,2#STMP3
2371	011342	012737	011654	001242	MOV	#GGP8,2#STMP4
2372	011350	012737	011554	001244	MOV	#GGDAT0,2#STMP5
2373	011356	012737	011664	001246	MOV	#GGP9,2#STMP6
2374	011364	104223			15: ERROR	223
2375	011366	000137	011674		JMP	2#GGDONE
2376						
2377	011372				GGER18:	
2378	011372	010537	001252		MOV	R5,2#STMP10
2379	011376	010437	001254		MOV	R4,2#STMP11
2380	011402	012737	011574	001240	MOV	#GGP2,2#STMP3
2381	011410	012737	011654	001242	MOV	#GGP8,2#STMP4
2382	011416	012737	011554	001244	MOV	#GGDAT0,2#STMP5
2383	011424	012737	011664	001246	MOV	#GGP9,2#STMP6
2384	011432	104207			15: ERROR	207
2385	011434	000137	011674		JMP	2#GGDONE
2386						
2387	011440				GGER19:	
2388	011440	010537	001252		MOV	R5,2#STMP10
2389	011444	010437	001254		MOV	R4,2#STMP11
2390	011450	012737	011574	001240	MOV	#GGP2,2#STMP3
2391	011456	012737	011654	001242	MOV	#GGP8,2#STMP4
2392	011464	012737	011554	001244	MOV	#GGDAT0,2#STMP5
2393	011472	012737	011664	001246	MOV	#GGP9,2#STMP6
2394	011500	104230			15: ERROR	230
2395	011502	000137	011674		JMP	2#GGDONE
2396						
2397	011506				GGER20:	
2398	011506	010537	001252		MOV	R5,2#STMP10

```

2399 011512 010437 001254      MOV      R4, @STMP11
2400 011516 012737 011574 001240      MOV      @GGP2, @STMP3
2401 011524 012737 011654 001242      MOV      @GGP8, @STMP4
2402 011532 012737 011554 001244      MOV      @GGDATO, @STMP5
2403 011540 012737 011664 001246      MOV      @GGP9, @STMP6
2404 011546 104231 1S:      ERROR  23:
2405 011550 000137 011674      JMP      @GGDONE
2406
2407 011554 000000      GGDATO: 0
2408 011556 000000      0
2409 011560 000000      0
2410 011562 000000      0
2411
2412 011564 000300      GGP1:   300
2413 011566 000000      0
2414 011570 000000      0
2415 011572 000000      0
2416 011574 100200      GGP2:  100200
2417 011576 000000      0
2418 011600 000000      0
2419 011602 000000      0
2420 011604 000200      GGP3:   200
2421 011606 000000      0
2422 011610 000000      0
2423 011612 000001      1
2424 011614 010200      GGP4:  10200
2425 011616 000000      0
2426 011620 000000      0
2427 011622 000000      0
2428 011624 077600      GGP5:  77600      :OVER FLOW = GGP5 + GGP5
2429 011626 000000      0
2430 011630 000000      0
2431 011632 000000      0
2432 011634 000000      GGP6:   0      :OVERFLOW RESULT
2433 011636 000000      0      :UNDERFLOW RESULT
2434 011640 000000      0      :GGP6 = GGP4 + GGP5
2435 011642 000000      0      :      = GGP3 + GGP2 (FIU = 0)
2436      0      :      = GGP3 + GGP1
2437 011644 062400      GGP7:  62400      :GGP7 = GGP3 + GGP2 (FIU = 1)
2438 011646 000000      0
2439 011650 000000      0
2440 011652 000000      0
2441 011654 000340      GGP8:   340
2442 011656 000000      0
2443 011660 000000      0
2444 011662 000000      0
2445 011664 000100      GGP9:   100
2446 011666 000000      0
2447 011670 000000      0
2448 011672 000000      0
2449 011674 000000      GGDONE:
2450 011674 104412      RSETJP
2451
2452
2453
2454
  
```

```

: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?).
  
```

```

2455
2456
2457
2458
2459
2460
2461
2462
2463 011576 000004
2464
2465 011700
2466 011700 104413
2467 011702 012704 000200
2468 011706 170104
2469 011710 012700 013314
2470 011714 172410
2471 011716 012700 013324
2472 011722 012737 011730 001236
2473 011730 177420
2474 011732 020027 013330
2475 011736 001402
2476 011740 000137 012672
2477 011744
2478 011744 170205
2479 011746 012700 013304
2480 011752 174010
2481 011754 012701 013374
2482 011760 012702 000004
2483 011764 022120
2484 011766 001415
2485 011770 012701 013324
2486 011774 012700 013304
2487 012000 012702 000004
2488 012004 022120
2489 012006 001402
2490 012010 000137 012732
2491 012014 077205
2492 012016 000137 012762
2493 012022 077220
2494 012024 012704 000200
2495 012030 020405
2496 012032 001402
2497 012034 000137 013030
2498
2499 012040
2500 012040 104413
2501 012042 012704 000200
2502 012046 170104
2503
2504 012050 012700 013314
2505 012054 172410
2506
2507 012056 012700 013324
2508 012062 012737 012072 001236
2509
2510 012070 170001

```

```

*****
: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),AC0
MOV #HXP2,R0
MOV #HX2,@STMP2
HX2: LDCFD (R0)+,AC0 ;IS R0 CORRECT
CMP R0,#HXP2+4
BEQ HX3
JMP @HXER1
HX3: STFPS R5 ;GET FPS
MOV #HXDAT0,R0
STD AC0,(R0) ;GET AC0
MOV #HXP7,R1 ;SEE IF RESULT IS
MOV #4,R2 ;CORRECT
HX4: CMP (R1)+,(R0)+
BEQ HX7
MOV #HXP2,R1 ;DID FD GET
MOV #HXDAT0,R0 ;COMPLIMENTED?
MOV #4,R2
HX5: CMP (R1)+,(R0)+
BEQ HX6
JMP @HXER2
HX6: SOB R2,HX5
JMP @HXER3
HX7: SOB R2,HX4 ;FPS CORRECT?
MOV #200,R4
CMP R4,R5
BEQ HX8
JMP @HXER8
:NOW
HX8: TEST LDCDF
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
MOV #200,R4
LDFPS R4
MOV #HXP1,R0
LDD (R0),AC0
MOV #HXP2,R0
MOV #HX9,@STMP2
SETF

```

2511											
2512	012072	177420									
2513											
2514	012074	020027	013334								
2515	012100	001402									
2516	012102	000137	012712								
2517											
2518	012106										
2519	012106	170205									
2520	012110	012700	013304								
2521	012114	170011									
2522	012116	174010									
2523	012120	012701	013404								
2524	012124	012702	000004								
2525	012130	022120									
2526	012132	001415									
2527											
2528	012134	012701	013374								
2529	012140	012700	013304								
2530	012144	012702	000004								
2531	012150	022110									
2532	012152	001402									
2533	012154	000137	013046								
2534	012160	077205									
2535	012162	000137	013076								
2536											
2537	012166	077220									
2538											
2539	012170	012704	000000								
2540	012174	020405									
2541	012176	001402									
2542	012200	000137	013030								
2543											
2544											
2545											
2546	012204										
2547	012204	104413									
2548											
2549	012206	012704	000200								
2550	012212	170104									
2551	012214	012737	012232	001236							
2552	012222	012737	013126	000004							
2553	012230	005001									
2554	012232	177427	043243								
2555	012236	005201									
2556	012240	005201									
2557	012242	005201									
2558	012244	012737	036646	000004							
2559	012252	020127	000003								
2560	012256	001402									
2561	012260	000137	013162								
2562											
2563	012264										
2564	012264	104413									
2565											
2566	012266	012704	000200								

```

HX9:  LDCDF  (R0)+,AC0      ;TEST INSTRUCTION
      CMP    R0,#HXP2+10    ;WAS A GOOD
      BEQ    HX10           ;CONSTANT USED
      JMP    @#HXERS        ;TO INCREMENT R0?

HX10: STFPS  R5
      MOV    #HXDAT0,R0
      SETD
      STD    ACO,(R0)      ;GET RESULT
      MOV    #HXP8,R1
      MOV    #4,R2
HX11:  CMP    (R1)+,(R0)+   ;IS IT CORRECT?
      BEQ    HX14

      MOV    #HXP7,R1
      MOV    #HXDAT0,R0
      MOV    #4,R2
HX12:  CMP    (R1)+,(R0)   ;DID FD FAIL TO GET
      BEQ    HX13          ;COMPLIMENTED?
      JMP    @#HXER6

HX13:  SOB   R2,HX12
      JMP    @#HXER7

HX14:  SOB   R2,HX11

      MOV    #0,R4          ;FPS CORRECT?
      CMP    R4,R5
      BEQ    HX15
      JMP    @#HXER8

;TEST GR7 IMMEDIATE MODE CONSTANT
HX15:  LPERR
      ;SET UP THE LOOP ON ERROR ADDRESS.

      MOV    #200,R4
      LDFPS R4              ;SET FD
      MOV    #HX16,@#STMP2
      MOV    #HXER9,@#ERRVECT
      CLR    R1
HX16:  LDCFD  #5201,ACO
HX165: INC    R1
      INC    R1
      INC    R1
      MOV    #CPSPUR,@#ERRVECT
      CMP    R1,#3          ;SEE IF PC WAS
      BEQ    HX17          ;CORRECT
      JMP    @#HXER10

HX17:  LPERR
      ;SET UP THE LOOP ON ERROR ADDRESS.

      MOV    #200,R4

```

2567	012272	170104			LDFPS	R4	
2568	012274	012737	012322	001236	MOV	#HX18, @#STMP2	
2569	012302	012700	013364		MOV	#HXP6, R0	
2570	012306	172410			LDD	(R0), ACO	
2571	012310	012737	036646	000004	MOV	#CPSUR, @#ERRVECT	
2572	012316	012700	013324		MOV	#HXP2, R0	
2573	012322	177410			HX18:	LDCFD (R0), ACO	
2574							
2575	012324	012700	013304		MOV	#HXDAT0, R0	
2576	012330	174010			STD	ACO, (R0)	; GET RESULT.
2577	012332	012701	013374		MOV	#HXP7, R1	
2578	012336	012702	000004		MOV	#4, R2	
2579	012342	022021			HX19:	CMP (R0)+, (R1)+	; IS RESULT CORRECT?
2580	012344	001402			BEQ	HX20	
2581	012346	000137	012732		JMP	@#HXER2	
2582	012352	077205			HX20:	SQB R2, HX19	
2583							
2584							
2585	012354						
2586	012354	104413					
2587	012356	012704	000200		LPERR		; SET UP THE LOOP ON ERROR ADDRESS.
2588	012362	170104			MOV	#200, R4	
2589	012364	012737	012404	001236	LDFPS	R4	
2590	012372	012700	013364		MOV	#HX22, @#STMP2	
2591	012376	172410			MOV	#HXP6, R0	
2592	012400	012700	013344		LDD	(R0), ACO	
2593	012404	177410			MOV	#HXP4, R0	
2594					HX22:	LDCFD (R0), ACO	
2595	012406	012700	013304		MOV	#HXDAT0, R0	
2596	012412	174010			STD	ACO, (R0)	; GET RESULT
2597							
2598	012414	012701	013354		MOV	#HXP5, R1	
2599	012420	012702	000004		MOV	#4, R2	
2600	012424	022120			HX23:	CMP (R1)+, (R0)+	
2601	012426	001415			BEQ	HX26	
2602							
2603	012430	012701	013374		MOV	#HXP7, R1	
2604	012434	012700	013304		MOV	#HXDAT0, R0	
2605	012440	012702	000004		MOV	#4, R2	
2606	012444	022120			HX24:	CMP (R1)+, (R0)+	; WAS SIGN INCORRECT
2607	012446	001402			BEQ	HX25	
2608	012450	000137	013214		JMP	@#HXER11	
2609	012454	077205			HX25:	SQB R2, HX24	
2610	012456	000137	013234		JMP	@#HXER12	
2611							
2612	012462	077220			HX26:	SQB R2, HX23	
2613							
2614							
2615							
2616	012464						
2617	012464	104413					
2618	012466	012704	000200		LPERR		; SET UP THE LOOP ON ERROR ADDRESS.
2619	012472	170104			MOV	#200, R4	
2620					LDFPS	R4	
2621	012474	012700	013314		MOV	#HXP1, R0	
2622	012500	172410			LDD	(R0), ACO	

K04

 MAINDEC-11-OFFPB-A PDP 11 34 FPP DIAGNOSTIC PART 2 MACY11 27(1006) 01-NOV-76 21:12 PAGE 49
 OFFPBA.P11 01-NOV-76 21:06 T3 LDCFD AND LDCDF TEST

2623	012502	172010			ADDC	(R0),AC0	
2624							
2625	012504	012737	012516	001236	MOV	#HX28,2#STMP2	
2626	012512	012700	013314		MOV	#HXP1,R0	
2627	012516	177410			HX28:	LDCFD	(R0),AC0
2628							
2629	012520	170205			STFPS	R5	
2630							
2631	012522	012700	013304		MOV	#HXDAT0,R0	
2632	012526	174010			STD	AC0,(R0)	;GET RESULT
2633							
2634	012530	012701	013314		MOV	#HXP1,R1	
2635	012534	012702	000004		MOV	#4,R2	
2636	012540	022120			HX29:	CMP	(R1)+,(R0)+ ;IS IT 0?
2637	012542	001402				BEQ	HX30
2638	012544	000137	013264			JMP	2#HXER13
2639	012550	077205			HX30:	SOB	R2,HX29
2640							
2641	012552	012704	000204		MOV	#204,R4	;FPS CORRECT
2642	012556	020405			CMP	R4,R5	
2643	012560	001402			BEQ	HX31	
2644	012562	000137	013012		JMP	2#HXER4	
2645							
2646					:TEST	LDCFD	0
2647							
2648	012566				HX31:		
2649	012566	104413			LPERR		;SET UP THE LOOP ON ERROR ADDRESS.
2650	012570	012704	000200		MOV	#200,R4	
2651	012574	170104			LDFPS	R4	
2652							
2653	012576	012700	013364		MOV	#HXP6,R0	
2654	012602	172410			LDD	(R0),AC0	
2655							
2656	012604	012737	012616	001236	MOV	#HX32,2#STMP2	
2657	012612	012700	013314		MOV	#HXP1,R0	
2658	012616	177410			HX32:	LDCFD	(R0),AC0
2659							
2660	012620	170205			STFPS	R5	
2661							
2662	012622	012700	013304		MOV	#HXDAT0,R0	
2663	012626	174010			STD	AC0,(R0)	;GET RESULT
2664							
2665	012630	012701	013314		MOV	#HXP1,R1	
2666	012634	012702	000004		MOV	#4,R2	
2667	012640	022120			HX33:	CMP	(R1)+,(R0)+ ;IS IT ZERO?
2668	012642	001402				BEQ	HX34
2669	012644	000137	013264			JMP	2#HXER13
2670	012650	077205			HX34:	SOB	R2,HX33
2671							
2672	012652	012704	000204		MOV	#204,R4	;FPS CORRECT
2673	012656	020405			CMP	R4,R5	
2674	012660	001402			BEQ	HX35	
2675	012662	000137	013012		JMP	2#HXER4	
2676	012666	000137	013414		HX35:	JMP	2#HXDONE
2677							
2678							

;RO INCORRECT

2679									
2680	012672	012737	013330	001242	HXER1:	MOV	#HXP2+4, @#STMP4		
2681	012700	010037	001240			MOV	R0, @#STMP3		
2682	012704	104234			1\$:	ERROR	234		
2683	012706	000137	013414			JMP	@#HXDONE		
2684									
2685	012712	012737	013334	001242	HXER5:	MOV	#HXP2+10, @#STMP4		
2686	012720	010037	001240			MOV	R0, @#STMP3		
2687	012724	104237			1\$:	ERROR	237		
2688	012726	000137	013414			JMP	@#HXDONE		
2689									
2690	012732	012737	013324	001244	HXER2:	MOV	#HXP2, @#STMP5		
2691	012740	012737	013374	001250		MOV	#HXP7, @#STMP7		
2692	012746	012737	013304	001246	HXER22:	MOV	#HXDAT0, @#STMP6		
2693	012754	104233			1\$:	ERROR	233		
2694	012756	000137	013414			JMP	@#HXDONE		
2695									
2696	012762	012737	013324	001244	HXER3:	MOV	#HXP2, @#STMP5		
2697	012770	012737	013374	001250		MOV	#HXP7, @#STMP7		
2698	012776	012737	013304	001246	HXER33:	MOV	#HXDAT0, @#STMP6		
2699	013004	104241			1\$:	ERROR	241		
2700	013006	000137	013414			JMP	@#HXDONE		
2701									
2702	013012	010537	001240		HXER4:	MOV	R5, @#STMP3		
2703	013016	010437	001242			MOV	R4, @#STMP4		
2704	013022	104240			1\$:	ERROR	240		
2705	013024	000137	013414			JMP	@#HXDONE		
2706									
2707	013030	010537	001240		HXER8:	MOV	R5, @#STMP3		
2708	013034	010437	001242			MOV	R4, @#STMP4		
2709	013040	104242			1\$:	ERROR	242		
2710	013042	000137	013414			JMP	@#HXDONE		
2711	013046	012737	013324	001244	HXER6:	MOV	#HXP2, @#STMP5		
2712	013054	012737	013404	001250		MOV	#HXP8, @#STMP7		
2713	013062	012737	013304	001246	HXER66:	MOV	#HXDAT0, @#STMP6		
2714	013070	104244			1\$:	ERROR	244		
2715	013072	000137	013414			JMP	@#HXDONE		
2716									
2717	013076	012737	013324	001244	HXER7:	MOV	#HXP2, @#STMP5		
2718	013104	012737	013404	001250		MOV	#HXP8, @#STMP7		
2719	013112	012737	013304	001246		MOV	#HXDAT0, @#STMP6		
2720	013120	104243			1\$:	ERROR	243		
2721	013122	000137	013414			JMP	@#HXDONE		
2722									
2723	013126	032716	000001		HXER9:	BIT	#1, (SP)		:SEE IF IT
2724	013132	001005				BNE	1\$:AN ODD ADDRESS
2725	013134	022716	012236			CMP	#HX165, (SP)		
2726	013140	001402				BEG	1\$		
2727	013142	000137	036646			JMP	@#CPSPUR		
2728									
2729	013146	011637	001236		1\$:	MOV	(SP), @#STMP2		
2730	013152	022626				CMP	(SP)+, (SP)+		
2731	013154	104235			2\$:	ERROR	235		
2732	013156	000137	013414			JMP	@#HXDONE		
2733									
2734	013162	162701	000003		HXER10:	SUB	#3, R1		

2735	013166	006301			ASL	R1
2736	013170	012702	012236		MOV	#HX165 R2
2737	013174	010237	001242		MOV	R2,@STMP4
2738	013200	160102			SUB	R1,R2
2739	013202	010237	001240		MOV	R2,@STMP3
2740	013206	104236		15:	ERROR	236
2741	013210	000137	013414		JMP	@HXDONE
2742						
2743	013214	012737	013344	001244	HXR11: MOV	#HXP4,@STMP5
2744	013222	012737	013354	001250	MOV	#HXP5,@STMP7
2745	013230	000137	012746		JMP	@HXER22
2746	013234	012737	013344	001244	HXR12: MOV	#HXP4,@STMP5
2747	013242	012737	013354	001250	MOV	#HXP5,@STMP7
2748	013250	012737	013304	001246	MOV	#HXDAT0,@STMP6
2749	013256	104245			15: ERROR	245
2750	013260	000137	013414		JMP	@HXDONE
2751						
2752	013264	012737	013314	001244	HXR13: MOV	#HXP1,@STMP5
2753	013272	012737	013314	001250	MOV	#HXP1,@STMP7
2754	013300	000137	012746		JMP	@HXER22
2755						
2756	013304	000000			HXDAT0: 0	
2757	013306	000000			0	
2758	013310	000000			0	
2759	013312	000000			0	
2760						
2761	013314	000000			HXP1: 0	
2762	013316	000000			0	
2763	013320	000000			0	
2764	013322	000000			0	
2765						
2766	013324	000577			HXP2: 577	
2767	013326	177776			177776	
2768	013330	177777			177777	
2769	013332	177776			177776	
2770	013334	005201			HXP3: 5201	
2771	013336	000000			0	
2772	013340	000000			0	
2773	013342	000000			0	
2774	013344	100577			HXP4: 100577	
2775	013346	177776			177776	
2776	013350	177777			177777	
2777	013352	177776			177776	
2778	013354	100577			HXP5: 100577	
2779	013356	177776			177776	
2780	013360	000000			0	
2781	013362	000000			0	
2782	013364	000252			HXP6: 252	
2783	013366	125252			125252	
2784	013370	125252			125252	
2785	013372	125252			125252	
2786						
2787	013374	000577			HXP7: 577	
2788	013376	177776			177776	
2789	013400	000000			0	
2790	013402	000000			0	

2791 013404 000577
2792 013406 177777
2793 013410 000000
2794 013412 000000
2795
2796 013414
2797 013414 104412
2798
2799
2800
2801
2802
2803
2804
2805
2806
2807
2808
2809
2810
2811
2812
2813
2814 013416 000004
2815
2816
2817 013420
2818 013420 104413
2819 013422 004737 014232
2820 013426 000000 000000 000000
2821 013434 000000
2822 013436 000000 000000 000000
2823 013444 000000
2824 013446 000200
2825 013450 000204
2826 013452 000200
2827 013454 104001
2828
2829
2830
2831 013456
2832 013456 104413
2833 013460 004737 014232
2834 013464 000000 000000 000000
2835 013472 000000
2836 013474 025252
2837 013476 052525
2838 013500 125252
2839 013502 052525
2840 013504 000200
2841 013506 000200
2842 013510 000210
2843 013512 104003
2844
2845
2846 013514

HXPB: 577
177777
0
0
HXDONE:
RSETUP

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

:TEST 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
:*

:TEST4: SCOPE

:TEST THE CMPD INSTRUCTION WITH (FSRC=AC=0)
AAA1:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,2#CMPSUB
1\$: .WORD 0,0,0,0 ;ACD
2\$: .WORD 0,0,0,0 ;FSRC
3\$: 200 ;FPS BEFORE EXECUTION
204 ;FPS AFTER EXECUTION
200 ;ERROR FPS
4\$: ERROR 1 ;FPS ERROR

:TEST CMPD WITH (AC=0) AND FSRC POSITIVE.
AAA2:

LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,2#CMPSUB
1\$: .WORD 0,0,0,0 ;AC
2\$: 25252 ;FSRC
52525
125252
3\$: 200 ;FPS BEFORE EXECUTION
200 ;FPS AFTER EXECUTION
210 ;ERROR FPS
4\$: ERROR 3 ;FPS ERROR

:TEST CMPD WITH (AC=0) AND FSRC NEGATIVE
AAA3:

```

2847 013514 104413 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
2848 013516 004737 014232 JSR PC, @CMPSUB
2849 013522 000000 000000 000000 1$: .WORD 0.0,0.0 ;AC
2850 013530 000000 2$: ;FSRC
2851 013532 125252
2852 013534 125252
2853 013536 052525 52525
2854 013540 125252 125252
2855 013542 000200 3$: 200 ;FPS BEFORE EXECUTION
2856 013544 000210 210 ;FPS AFTER EXECUTION
2857 013546 000200 200 ;ERROR FPS
2858 013550 104004 4$: ERROR 4 ;FPS ERROR.
2859
2860 ;TEST CMPD WITH (FSRC=0) AND AC POSITIVE
2861 013552 AAA4:
2862 013552 104413 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
2863 013554 004737 014232 JSR PC, @CMPSUB
2864 013560 025252 1$: 052525 ;AC
2865 013562 052525 52525
2866 013564 125252 125252
2867 013566 052525 52525
2868 013570 000000 000000 000000 2$: .WORD 0.0,0.0 ;FSRC
2869 013576 000000 3$: 200 ;FPS BEFORE EXECUTION
2870 013600 000200 210 ;FPS AFTER EXECUTION
2871 013602 000210 200 ;ERROR FPS
2872 013604 000200 4$: 200 ;FPS ERROR
2873 013606 104005 ERROR 5
2874
2875 ;TEST CMPD WITH (FSRC=0) AND AC NEGATIVE
2876 AAA5:
2877 013610
2878 013610 104413 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
2879 013612 004737 014232 JSR PC, @CMPSUB
2880 013616 125252 1$: 125252 ;AC
2881 013620 125252 125252
2882 013622 052525 52525
2883 013624 125252 125252
2884 013626 000000 000000 000000 2$: .WORD 0.0,0.0 ;FSRC
2885 013634 000000 3$: 200 ;FPS BEFORE EXECUTION
2886 013636 000200 200 ;FPS AFTER EXECUTION
2887 013640 000200 210 ;ERROR FPS
2888 013642 000210 4$: 210 ;FPS ERROR
2889 013644 104006 ERROR 6
2890
2891 ;TEST CMPD WITH AC POSITIVE AND FSRC NEGATIVE
2892 AAA6:
2893 013646 104413 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
2894 013650 004737 014232 JSR PC, @CMPSUB
2895 013654 052525 1$: 52525 ;AC
2896 013656 125252 125252
2897 013660 052525 52525
2898 013662 125252 125252
2899 013664 125252 2$: 125252 ;:FSRC
2900 013666 052525 52525
2901 013670 125252 125252
2902 013672 052525 52525

```

```

2903 013674 000200 35: 200 :FPS BEFORE EXECUTION
2904 013676 000210 :FPS AFTER EXECUTION
2905 013700 000200 :ERROR FPS
2906 013702 104007 45: ERROR 7 :FPS ERROR

```

:TEST CMPD WITH AC NEGATIVE AND FSRC POSITIVE

```

2908
2909
2910 013704
2911 013704 104413 014232 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
2912 013706 004737 JSR PC,20CMP5UB
2913 013712 125252 15: 125252 ;AC
2914 013714 052525 52525
2915 013716 125252 125252
2916 013720 052525 52525
2917 013722 052525 25: 52525 ;FSRC
2918 013724 125252 125252
2919 013726 052525 52525
2920 013730 125252 125252
2921 013732 000200 35: 200 :FPS BEFORE EXECUTION
2922 013734 000200 :FPS AFTER EXECUTION
2923 013736 000210 :ERROR FPS
2924 013740 104010 45: ERROR 10 :FPS ERROR.

```

:TEST CMPD WITH AC POSITIVE AND FSRC POSITIVE AND EAC LESS THAN EFSRC.

```

2925
2926
2927
2928 013742
2929 013742 104413 014232 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
2930 013744 004737 JSR PC,20CMP5UB
2931 013750 012345 15: 12345 ;AC
2932 013752 067654 67654
2933 013754 032101 32101
2934 013756 023456 23456
2935 013760 023456 25: 23456 ;FSRC
2936 013762 076543 76543
2937 013764 021012 21012
2938 013766 034567 34567
2939 013770 000200 35: 200 :FPS BEFORE EXECUTION
2940 013772 000200 :FPS AFTER EXECUTION
2941 013774 000210 :ERROR FPS
2942 013776 104011 45: ERROR 11 :FPS ERROR

```

:TEST CMPD WITH AC POSITIVE, FSRC POSITIVE AND EAC GREATER THAN EFSRC

```

2943
2944
2945
2946 014000
2947 014000 104413 014232 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
2948 014002 004737 JSR PC,20CMP5UB
2949 014006 045676 15: 45676 ;AC
2950 014010 054321 54321
2951 014012 012345 12345
2952 014014 067654 67654
2953 014016 034567 25: 34567 ;FSRC
2954 014020 065432 65432
2955 014022 101234 101234
2956 014024 056765 56765
2957 014026 000200 35: 200 :FPS BEFORE EXECUTION
2958 014030 000210 :FPS AFTER EXECUTION

```

```

2959 014032 000200          200          :ERROR FPS
2960 014034 104013          45:  ERROR 12
2961 014036 104013          :TEST CMPD WITH AC POSITIVE, FSRC POSITIVE AND AC EQUAL TO FSRC
2962 014036 104013          AAA10:
2963 014036 104413          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
2964 014036 004737 014232          JSR  PC,2@CMPSUB
2965 014040 004737          15: 12345          :AC
2966 014044 012345          :FSRC
2967 014046 067012          :FPS BEFORE EXECUTION
2968 014050 034567          :FPS AFTER EXECUTION
2969 014052 012345          :ERROR FPS
2970 014054 012345          :FPS ERROR
2971 014056 067012          25: 12345          :FSRC
2972 014060 034567          :FPS BEFORE EXECUTION
2973 014062 012345          :FPS AFTER EXECUTION
2974 014064 000200          35: 200          :ERROR FPS
2975 014066 000204          :FPS ERROR
2976 014070 000200          45: 200          :FPS BEFORE EXECUTION
2977 014072 104013          :FPS AFTER EXECUTION
2978 014072 104013          :ERROR FPS
2979 014072 104013          :FPS ERROR
2980 014072 104013          :TEST CMPD WITH AC POSITIVE, FSRC POSITIVE, EAC EQUAL TO EFSRC,
2981 014074 104013          :AND FSRC GREATER THAN AC.
2982 014074 104413          AAA11:
2983 014076 004737 014232          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
2984 014102 012345          JSR  PC,2@CMPSUB
2985 014104 067012          15: 12345          :AC
2986 014106 034567          :FSRC
2987 014110 012345          :FPS BEFORE EXECUTION
2988 014112 012345          :FPS AFTER EXECUTION
2989 014114 070123          25: 70123          :ERROR FPS
2990 014116 045670          :FPS ERROR
2991 014120 123456          :FPS BEFORE EXECUTION
2992 014122 000200          :FPS AFTER EXECUTION
2993 014124 000200          35: 200          :ERROR FPS
2994 014126 000210          :FPS ERROR
2995 014130 104014          45: 210          :FPS BEFORE EXECUTION
2996 014130 104014          :FPS AFTER EXECUTION
2997 014130 104014          :ERROR FPS
2998 014130 104014          :FPS ERROR
2999 014132 104013          :TEST CMPD WITH AC POSITIVE, FSRC POSITIVE, EAC EQUAL TO EFSRC,
3000 014132 104413          :AND AC GREATER THAN FSRC.
3001 014134 004737 014232          AAA12:
3002 014140 054321          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
3003 014142 076543          JSR  PC,2@CMPSUB
3004 014144 021076          15: 54321          :AC
3005 014146 054321          :FSRC
3006 014150 054321          :FPS BEFORE EXECUTION
3007 014152 065432          :FPS AFTER EXECUTION
3008 014154 107654          25: 65432          :ERROR FPS
3009 014156 032107          :FPS ERROR
3010 014160 000200          :FPS BEFORE EXECUTION
3011 014162 000210          :FPS AFTER EXECUTION
3012 014164 000200          35: 210          :ERROR FPS
3013 014166 104015          :FPS ERROR
3014 014166 104015          :FPS ERROR

```

E05

```

3015 :TEST CMPD WITH AC NEGATIVE, FSRC NEGATIVE, EAC EQUAL TO EFSRC.
3016 :AND AC GREATER THAN FSRC
3017 014170 AA013: LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
3018 014170 104413 JSR PC, @#CMPSUB ;AC
3019 014172 004737 014232 1S: 112345
3020 014176 112345 43210
3021 014200 043210 76543
3022 014202 076543 21076
3023 014204 021076 2S: 112345 ;FSRC
3024 014206 112345 54321
3025 014210 054321 07654
3026 014212 007654 32107
3027 014214 032107 3S: 200 ;FPS BEFORE EXECUTION
3028 014216 000200 210 ;FPS AFTER EXECUTION
3029 014220 000210 200 ;ERROR FPS
3030 014222 000200 4S: ERROR 16 ;FPS ERROR
3031 014224 104016
3032
3033
3034 014226 000137 014422 JMP @#AAADONE ;FINISHED CMPD TEST.
3035
3036
3037
3038
3039
3040
3041
3042
3043
3044
3045
3046
3047
3048
3049
3050
3051
3052
3053
3054
3055
3056
3057
3058
3059

```

: THIS SUBROUTINE, CMPSUB, IS CALLED TO SET UP, EXECUTE
 : AND CHECK THE RESULTS OF A CMPD INSTRUCTION.
 : IT IS CALLED THUS:

```

JSR PC, @#CMPSUB
ACARG: .WORD X,X,X,X ;AC OPERAND
FSRCARG: .WORD X,X,X,X ;FSRC OPERAND
FPSB: .WORD X ;FPS BEFORE EXECUTION
FPSA: .WORD X ;FPS AFTER EXECUTION
FPSE: .WORD X ;ERROR FPS
ERR: ERROR X ;FPS ERROR
CONT: ;RETURN ADDRESS

```

: THE OPERANDS ARE SET UP (USING ACO FOR THE AC OPERAND). THEN
 : FPSB IS LOADED INTO THE FPS. THE INSTRUCTION, CMPD, IS EXECUTED.
 : AFTER THE EXECUTION THE FPS IS CHECKED AGAINST FPSA. IF IT IS A MATCH
 : THEN THERE WAS NO ERROR AND CONTROL IS RETURNED TO CONT. IF
 : THE FPS IS INCORRECT IT IS COMPARED WITH FPSE IN AN ATTEMPT TO ANALYSE
 : THE FAILURE. IF THE FPS IS THE SAME AS FPSE THEN CONTROL IS
 : RETURNED TO THE ERROR CALL AT LOCATION ERR. IF THE FPS WAS
 : NOT CORRECT BUT DIDN'T MATCH FPSE A GENERAL ERROR IS REPORTED
 : AND CONTROL IS PASSED TO CONT.

```

3060 014232 012601 CMPSUB: MOV (SP)+, R1 ;PICK UP A POINTER TO THE
3061 3061 ;ARGUMENTS.
3062 014234 016100 000020 MOV 20(R1), R0 ;GET THE FPS BEFORE EXECUTION.
3063 014240 170100 LDFPS R0 ;LOAD IT INTO THE FPS.
3064
3065 014242 012737 014264 001236 MOV #15, @#STMP2 ;SAVE ADDRESS OF CMPD INSTRUCTION.
3066 014250 010100 MOV R1, R0 ;GET ADDRESS OF AC OPERAND.
3067 014252 172410 LDD (R0), ACO ;LOAD ACO OPERAND
3068
3069 014254 010100 MOV R1, R0 ;COMPUTE FSRC OPERAND
3070 014256 062700 000010 ADD #10, R0 ;ADDRESS

```


3127
3128
3129
3130
3131
3132
3133
3134
3135
3136
3137
3138
3139
3140
3141
3142
3143
3144
3145
3146
3147
3148
3149
3150
3151
3152
3153
3154
3155
3156
3157
3158
3159
3160
3161
3162
3163
3164
3165
3166
3167
3168
3169
3170
3171
3172
3173
3174
3175
3176
3177
3178
3179
3180
3181
3182

```

*****
*TEST 5      DIVD WITH (FSRC=0) AND (BUT FC) TEST
*
*THIS IS A TEST OF THE DIVD INSTRUCTION WITH A
*ZERO DIVISOR. THE CONDITION IS CHECKED WITH BOTH
*TRAP ENABLED AND TRAPS DISABLED.
*
*****
†STS:  SCOPE
:FIRST TEST DIVD WITH (FSRC=AC=0) AND TRAPS DISABLED.
3142 014426 104413          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
3143 014430 012704 040200 BB80:  MOV      #40200,R4 ;SET UP FPS
;WITH INTERRUPTS
;DISABLED.
3146 014434 170104          LDFPS      R4
3147 014436 012737 014700 000244 MOV      #BB80R1,2#FPVCT;SET UP FOR ANY FP INTERRUPTS.
3148 014444 012737 014464 001236 MOV      #BB81,2#STMP2
3149 014452 012700 015104 MOV      #BB8P1,R0 ;SET UP ACC = 0
3150 014456 172410          LDD      (R0),ACC
3151 014460 012701 015104 MOV      #BB8P1,R1 ;FSRC = 0
3153 014464 174411          BB81:  DIVD      (R1),ACC ;TEST INSTRUCTION
3155 014466 170205          STFPS     R5 ;GET FPS
3156 014470 170303          STST      R3 ;GET FEC
3158 014472 012704 140204 MOV      #140204,R4 ;EXPECTED FPS.
3159 014476 020405          CMP      R4,R5 ;IS FPS CORRECT.
3160 014500 001131          BNE      BB8E2 ;IF INCORRECT BRANCH.
3162 014502 012702 000004 MOV      #4,R2 ;EXPECTED FEC.
3163 014506 020203          CMP      R2,R3 ;IS FEC CORRECT?
3164 014510 001140          BNE      BB8E3 ;IF INCORRECT BRANCH.
3166          ;TEST DIVD WITH (FSRC=0) AND TRAPS DISABLED.
3167 014512          BB82:
3168 014512 104413          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
3169 014514 012704 040200 MOV      #40200,R4 ;LOAD FPS WITH TRAPS DISABLED.
3170 014520 170104          LDFPS      R4
3172 014522 012737 014542 001236 MOV      #BB83,2#STMP2
3173 014530 012700 015114 MOV      #BB8P2,R0 ;SET UP ACC OPERAND (NON ZERO).
3174 014534 172410          LDD      (R0),ACC
3175 014536 012700 015104 MOV      #BB8P1,R0 ;FSRC=0
3176 014542 174410          BB83:  DIVD      (R0),ACC
3178 014544 170205          STFPS     R5 ;GET FPS.
3179 014546 170303          STST      R3 ;GET FEC.
3181 014550 012704 140200 MOV      #140200,R4 ;EXPECTED FPS.
3182 014554 020405          CMP      R4,R5 ;IS FPS CORRECT?
  
```

H05

```

3183 014556 001102      BNE      BBBER2      ;IF INCORRECT BRANCH.
3184
3185 014560 012702 000004      MOV      #4,R2      ;EXPECTED FEC.
3186 014564 020203      CMP      R2,R3      ;WAS FEC CORRECT?
3187 014566 001111      BNE      BBBER3      ;IF INCORRECT BRANCH.
3188
3189      ;TEST DIVD WITH FSRC=0) AND TRAPS ENABLED.
3190      BBB4:
3191 014570      LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.
3192 014572 104413 000200      MOV      #200,R4    ;SET UP FPS. TRAP ENABLED.
3193 014574 170104      LD FPS     R4
3194
3195 014600 012737 014626 001236      MOV      #BBB5,2#STMP2
3196 014606 012700 015114      MOV      #BBB2,R0    ;SET UP ACC OPERAND (NON ZERO).
3197 014612 172410      LDD      (R0),ACC
3198
3199 014614 012737 014634 000244      MOV      #BBB6,2#FPVECT ;SET UP FOR THE EXPECTED INTERRUPT.
3200 014622 012700 015104      MOV      #BBB1,R0    ;FSRC=0
3201
3202 014626 174410      BBB5: DIVD      (R0),ACC ;TEST INSTRUCTION (SHOULD RESULT IN TRAP).
3203 014630 170000      CFCC
3204
3205 014632 000502      BR       BBBER4      ;GO REPORT FAILURE, NO TRAP.
3206
3207 014634 022716 014630      BBB6: CMP      #BBB5+2,(SP) ;TRAP TO HERE WHEN THE DIVISION BY 0
3208      ;OCCURS. FIRST SEE IF THE ADDRESS OF
3209      ;THE TRAP IS 2+THE ADDRESS OF THE TEST
3210      ;DIVD INSTRUCTION.
3211 014640 001402      BEQ      1$
3212 014642 000137 036614      JMP      2#FPSPUR    ;IF NOT THEN REPORT AN UNEXPECTED
3213      ;FP TRAP.
3214 014646 170205      1$: STFPS     R5      ;GET FPS.
3215 014650 170303      STST     R3      ;GET FEC.
3216 014652 022626      CMP      (SP)+,(SP)+ ;RESET THE STACK.
3217
3218 014654 012704 100200      MOV      #100200,R4  ;EXPECTED FPS.
3219 014660 020405      CMP      R4,R5      ;IS FPS CORRECT?
3220 014662 001040      BNE      BBBER2      ;IF INCORRECT BRANCH.
3221
3222 014664 012702 000004      MOV      #4,R2      ;EXPECTED FEC.
3223 014670 020203      CMP      R2,R3      ;IS FEC CORRECT?
3224 014672 001047      BNE      BBBER3      ;IF INCORRECT BRANCH.
3225
3226 014674 000137 015124      JMP      2#BBBDONE   ;OTHERWISE GO TO NEXT TEST.
3227
3228
3229      ;TRAP HERE IF AN UNEXPECTED INTERRUPT OCCURS.
3230 014700 062737 000002 001236      BBBER1: ADD     #2,2#STMP2 ;SEE IF THE INTERRUPT OCCURRED
3231      ;DURING THE EXECUTION OF THE DIVD
3232      ;INSTRUCTION BEING TESTED.
3233 014706 021637 001236      CMP      (SP),2#STMP2
3234 014712 001402      BEQ      1$
3235 014714 000137 036614      JMP      2#FPSPUR    ;IF NOT REPORT UNEXPECTED FP TRAP.
3236
3237 014720 022626      1$:  CMP      (SP)+,(SP)+ ;RESET THE STACK.
3238 014722 170303      STST     R3      ;GET FEC.
  
```

```

3239 014724 170205          STFPS  R5          ;GET FPS.
3240 014726 012737 000004 001240  MOV    #4,0#STMP3 ;EXPECTED FEC.
3241 014734 010337 001242          MOV    R3,0#STMP4
3242 014740 010537 001244          MOV    R5,0#STMP5
3243 014744 010037 001250          MOV    R0,0#STMP7
3244 014750 012737 140200 001246  MOV    #140200,0#STMP6
3245 014756 104017          2S:   ERROR  17          ;REPORT (BUT FD, FAILED RESULTING IN AN FP TRAP
3246                                     ;WITH TRAPS DISABLED.
3247 014760 000137 015124          JMP    0#BBBDONE
3248
3249          ;REPORT FPS INCORRECT:
3250 014764 010537 001242  BBBER2: MOV    R5,0#STMP4
3251 014770 010437 001244          MOV    R4,0#STMP5
3252 014774 010037 001246          MOV    R0,0#STMP6
3253 015000 010137 001250          MOV    R1,0#STMP7
3254 015004 104020          1S:   ERROR  20
3255 015006 000137 015124          JMP    0#BBBDONE
3256
3257          ;REPORT FEC INCORRECT:
3258 015012 010337 001242  BBBER3: MOV    R3,0#STMP4
3259 015016 010237 001240          MOV    R2,0#STMP3
3260 015022 010037 001246          MOV    R0,0#STMP6
3261 015026 010137 001250          MOV    R1,0#STMP7
3262 015032 104021          1S:   ERROR  21
3263 015034 000137 015124          JMP    0#BBBDONE
3264
3265          ;REPORT NO TRAP OCCURRED AFTER TRYING TO DIVIDE
3266          ;BY ZERO WITH ALL TRAPS ENABLED.
3267 015040 170303  BBBER4: STST  R3          ;GET FEC.
3268 015042 170205          STFPS  R5          ;GET FPS.
3269 015044 012737 000004 001242  MOV    #4,0#STMP4
3270 015052 010337 001240          MOV    R3,0#STMP3
3271 015056 010537 001244          MOV    R5,0#STMP5
3272 015062 012737 100200 001246  MOV    #100200,0#STMP6
3273 015070 010037 001250          MOV    R0,0#STMP7
3274 015074 010137 001252          MOV    R1,0#STMP10
3275 015100 104022          1S:   ERROR  22
3276 015102 000410          BR     BBBBDONE
3277
3278 015104 000000 000000 000000  BBBP1: .WORD  0,0,0,0
3279 015112 000000
3280 015114 012345 054321 023456  BBBP2: .WORD  12345,54321,23456,76543
3281 015122 076543
3282
3283
3284
3285          BBBBDONE:
3286 015124          RSETUP          ;GO INITIALIZE THE FPS AND STACK; AND
3287          ;SEE IF THE USER HAS EXPRESSED
3288          ;THE DESIRE TO CHANGE THE SOFTWARE
3289          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
3290          ;THE USER TYPED CONTROL G?).
3291
3292
3293
3294
; ; *****

```

3295
3296
3297
3298
3299
3300
3301
3302 015126 000004
3303
3304
3305 015130
3306 015130 104413
3307 015132 004767 000552
3308 015136 000000 000000
3309 015142 012345 067012
3310 015146 000000 000000
3311 015152 000000
3312 015154 000004
3313 015156 012345 067012
3314 015162 104023
3315
3316
3317 015164
3318 015164 104413
3319 015166 004737 015710
3320 015172 065652 125252
3321 015176 065600 000000
3322 015202 040252 125252
3323 015206 003000
3324 015210 003000
3325 015212 040052 125252
3326 015216 104024
3327
3328
3329 015220
3330 015220 104413
3331 015222 004767 000462
3332 015226 076400 000000
3333 015232 076400 000000
3334 015236 040200 000000
3335 015242 001000
3336 015244 001000
3337 015246 140200 000000
3338
3339 015252 104025
3340
3341
3342 015254
3343 015254 104413
3344 015256 004737 015710
3345 015262 056777 177777
3346 015266 054200 000000
3347 015272 042777 177777
3348 015276 000000
3349 015300 000000
3350 015302 002000 002000

```

;*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.
;*
*****
↑ST6: SCOPE

;CHECK DIVF WITH (AC=0).
CCC1:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC, DIVFSUB
1$: .WORD 0,0 ;AC
2$: .WORD 12345,67012 ;FSRC
3$: .WORD 0,0 ;RES
4$: 0 ;FPS BEFORE EXECUTION.
;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.
;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.
;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.
;FPS AFTER EXECUTION
5$: .WORD 2000,2000 ;ERROR RES.

```

```

3351 015306 104023      6$:  ERROR 23
3352
3353      :TEST THE DIVF INSTRUCTION:
3354      CCC5:
3354 015310      LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.
3355 015310 104413      JSR      PC,2#DIVFSUB
3356 015312 004737 015710      .WORD 12377,177777      ;AC
3357 015316 012377 177777      .WORD 12300,0          ;FSRC
3358 015322 012300 000000      .WORD 40252,125252     ;RES
3359 015326 040252 125252      .WORD 0                ;FPS BEFORE EXECUTION.
3360 015332 000000      .WORD 0                ;FPS AFTER EXECUTION.
3361 015334 000000      .WORD -1,-1           ;ERROR RES.
3362 015336 177777 177777      .WORD -1,-1
3363 015342 104023      6$:  ERROR 23
3364
3365      :TEST DIVIDE ALGORITHM. TEST ROUND CONSTANT.
3366      CCC6:
3367 015344      LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.
3368 015346 104413      JSR      PC,2#DIVFSUB
3369 015352 004737 015710      .WORD 64600,1          ;AC
3370 015356 064600 000001      .WORD 66600,0          ;FSRC
3371 015362 036200 000001      .WORD 36200,1          ;RES
3372 015366 000000      .WORD 0                ;FPS BEFORE EXECUTION.
3373 015370 000000      .WORD 0                ;FPS AFTER EXECUTION.
3374 015372 003000 003000      .WORD 3000,3000       ;ERROR RES.
3375 015376 104023      6$:  ERROR 23
3376
3377      :TEST DIVF.
3378      CCC7:
3379 015400      LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.
3380 015402 104413      JSR      PC,2#DIVFSUB
3381 015406 004737 015710      .WORD 34577,177776     ;AC
3382 015412 034577 177776      .WORD 23400,0          ;FSRC
3383 015416 023400 000000      .WORD 51377,177776     ;RES
3384 015422 051377 177776      .WORD 17                ;FPS BEFORE EXECUTION.
3385 015424 000017      .WORD 0                ;FPS AFTER EXECUTION.
3386 015426 000000      .WORD 0                ;ERROR RES.
3387 015428 003400 003400      .WORD 3400,3400
3388 015432 104023      6$:  ERROR 23
3389
3390      :DIVF TEST.
3391      CCC8:
3392 015434      LPERR      ;SET UP THE LOOP ON ERROR ADDRESS
3393 015436 104413      JSR      PC,2#DIVFSUB
3394 015442 004737 015710      .WORD 67652,125252     ;AC
3395 015446 067652 125252      .WORD 56500,0          ;FSRC
3396 015452 056500 000000      .WORD 51343,107070     ;RES
3397 015456 051343 107070      .WORD 0                ;FPS BEFORE EXECUTION.
3398 015460 000000      .WORD 0                ;FPS AFTER EXECUTION.
3399 015462 000000      .WORD 0                ;ERROR RES.
3400 015466 051543 107070      .WORD 51543,107070
3401 015468 104026      6$:  ERROR 26      ;DIDN'T INCREMENT THE EXPONENT
3402      ;AFTER DIVID NORMALIZATION.
3403
3404      :DIVF WITH AC NEGATIVE, FSRC NEGATIVE.
3405      CCC9:
3406 015470      LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.
3407 015472 104413      JSR      PC,2#DIVFSUB
3408 015476 004737 015710
    
```

LOS

3407	015476	140400	000000	15:	.WORD	140400,0	;AC
3408	015502	140500	000000	25:	.WORD	140500,0	;FSRC
3409	015506	040052	125253	35:	.WORD	040052,125253	;RES
3410	015512	000000		45:	0		;FPS BEFORE EXECUTION.
3411	015514	000000			0		;FPS AFTER EXECUTION.
3412	015516	140052	125253	55:	.WORD	140052,125253	;ERROR RES.
3413	015522	104027		65:	ERROR	27	;BAD SIGN.
3414							
3415							;DIVF WITH AC NEGATIVE AND FSRC POSITIVE.
3416	015524			CCC10:			
3417	015524	104413			LPERR		;SET UP THE LOOP ON ERROR ADDRESS.
3418	015526	004737	015710		JSR	PC,2#DIVFSUB	
3419	015532	160077	000000	15:	.WORD	160077,0	;AC
3420	015536	040277	000000	25:	.WORD	40277,0	;FSRC
3421	015542	160000	000000	35:	.WORD	160000,0	;RES
3422	015546	000007		45:	7		;FPS BEFORE EXECUTION.
3423	015550	000010			10		;FPS AFTER EXECUTION.
3424	015552	060000	000000	55:	.WORD	60000,0	;ERROR RES.
3425	015556	104027		65:	ERROR	27	;BAD SIGN.
3426							
3427							;DIVF WITH AC POSITIVE AND FSRC NEGATIVE.
3428	015560			CCC11:			
3429	015560	104413			LPERR		;SET UP THE LOOP ON ERROR ADDRESS.
3430	015562	004737	015710		JSR	PC,2#DIVFSUB	
3431	015566	040400	000000	15:	.WORD	40400,0	;AC
3432	015572	140500	000000	25:	.WORD	140500,0	;FSRC
3433	015576	140052	125253	35:	.WORD	140052,125253	;RES
3434	015602	000017		45:	17		;FPS BEFORE EXECUTION.
3435	015604	000010			10		;FPS AFTER EXECUTION.
3436	015606	040052	125253	55:	.WORD	40052,125253	;ERROR RES.
3437	015612	104027		65:	ERROR	27	;BAD SIGN.
3438							
3439							
3440							;TEST DIVF BOTH OPERANDS POSITIVE AND TRUNCATE MODE.
3441	015614			CCC12:			
3442	015614	104413			LPERR		;SET UP THE LOOP ON ERROR ADDRESS.
3443	015616	004737	015710		JSR	PC,2#DIVFSUB	
3444	015622	060100	000001	15:	.WORD	60100,1	;AC
3445	015626	040300	000000	25:	.WORD	40300,0	;FSRC
3446	015632	060000	000000	35:	.WORD	60000,0	;RES
3447	015636	000052		45:	52		;FPS BEFORE EXECUTION.
3448	015640	000040			40		;FPS AFTER EXECUTION.
3449	015642	060000	000001	55:	.WORD	60000,1	;ERROR RES.
3450	015646	104030		65:	ERROR	30	;TRUNCATION ERROR
3451							
3452							;DIVF WITH POSITIVE OPERANDS AND ROUND MODE.
3453	015650			CCC13:			
3454	015650	104413			LPERR		;SET UP THE LOOP ON ERROR ADDRESS.
3455	015652	004767	000032		JSR	PC,DIVFSUB	
3456	015656	060100	000001	15:	.WORD	60100,1	;AC
3457	015662	040300	000000	25:	.WORD	40300,0	;FSRC
3458	015666	060000	000001	35:	.WORD	60000,1	;RES
3459	015672	000005		45:	5		;FPS BEFORE EXECUTION.
3460	015674	000000			0		;FPS AFTER EXECUTION.
3461	015676	060000	000000	55:	.WORD	60000,0	;ERROR RES.
3462	015702	104031		65:	ERROR	31	;ROUND ERROR.

3463
 3464 015704 000137 016134
 3465
 3466
 3467
 3468
 3469
 3470
 3471
 3472
 3473
 3474
 3475
 3476
 3477
 3478
 3479
 3480
 3481
 3482
 3483
 3484
 3485
 3486
 3487
 3488
 3489
 3490
 3491
 3492 015710 012601
 3493 015712 012700 000200
 3494 015716 170100
 3495 015720 010100
 3496 015722 172410
 3497 015724 016100 000014
 3498 015730 170100
 3499 015732 012737 015746 001236
 3500 015740 010100
 3501 015742 062700 000004
 3502
 3503 015746 174410
 3504
 3505 015750 170204
 3506 015752 012700 000200
 3507 015756 170100
 3508
 3509 015760 012700 016124
 3510 015764 174010
 3511
 3512 015766 010102
 3513 015770 010237 001240
 3514 015774 062702 000004
 3515 016000 010237 001242
 3516 016004 062702 000004
 3517 016010 010237 001244
 3518 016014 012737 016124 001246

JMP @CCCCDONE ;GO TO NEXT TEST.

: 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 ACD 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),ACD
          MOV      14(R1),R0    ;LOAD THE FPS
          LDFPS   R0
          MOV      #15,@#STMP2
          MOV      R1,R0
          ADD     #4,R0          ;ESTABLISH A POINTER TO FSRC.
          15:  DIVF   (R0),ACD   ;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     ACD,(R0)
          MOV      R1,R2        ;SAVE THE DATA IN CASE OF ERROR.
          MOV      R2,@#STMP3
          ADD     #4,R2
          MOV      R2,@#STMP4
          ADD     #4,R2
          MOV      R2,@#STMP5
          MOV      #DIVFT,@#STMP6
    
```



```

3519 016022 010437 001250          MOV      R4,#STMP7
3520 016026 016137 000016 001252    MOV      16(R1),#STMP10
3521
3522 016034 021061 000010          CMP      (R0),10(R1)      ;IS THE RESULT CORRECT?
3523 016040 001011          BNE     10$              ;IF INCORRECT BRANCH.
3524 016042 026061 000002 000012    CMP      2(R0),12(R1)
3525 016050 001005          BNE     10$
3526
3527 016052 026104 000016          CMP      16(R1),R4      ;IS FPS CORRECT?
3528 016056 001020          BNE     15$              ;IF INCORRECT BRANCH.
3529 016060 000161 000026          JMP      26(R1)         ;IF NO ERRORS OCCURRED RETURN.
3530
3531 016064 021061 000020          10$:    CMP      (R0),20(R1)   ;DOES THE INCORRECT RESULT
3532 016070 001010          BNE     11$              ;MATCH THE ANTICIPATED INCORRECT RESULT.
3533 016072 026061 000002 000022    CMP      2(R0),22(R1)
3534 016100 001004          BNE     11$              ;BRANCH IF NO.
3535
3536 016102 010102          MOV      R1,R2          ;IT MATCHED SO RETURN TO THE ERROR
3537                                     ;REPORT AT THE CALLING ROUTINE.
3538 016104 062702 000024          ADD      #24,R2
3539 016110 000112          JMP      (R2)
3540
3541 016112          11$:                                     ;REPORT RESULT INCORRECT.
3542 016112 104023          12$:    ERROR    23
3543 016114 000161 000026          13$:    JMP      26(R1)
3544
3545 016120          15$:                                     ;REPORT FPS INCORRECT.
3546 016120 104032          16$:    ERROR    32
3547 016122 000774          BR      13$
3548
3549 016124 000000 000000 000000    DIVFT:  .WORD    0,0,0,0
3550 016132 000000
3551
3552 016134          CCCDONE:
3553 016134 104412          RSETUP
3554                                     ;GO INITIALIZE THE FPS AND STACK; AND
3555                                     ;SEE IF THE USER HAS EXPRESSED
3556                                     ;THE DESIRE TO CHANGE THE SOFTWARE
3557                                     ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
3558                                     ;THE USER TYPED CONTROL G?).
3559
3560
3561
3562
3563
3564
3565
3566
3567
3568 016136 000004          *****
3569                                     ;*TEST 7      DIVD TEST
3570                                     ;*
3571                                     ;*THIS IS A TEST OF THE DIVD INSTRUCTION. NOTE THAT A SUBROUTINE IS
3572                                     ;*USED TO SET UP THE OPERANDS, EXECUTE THE INSTRUCTION AND CHECK THE RESULTS.
3573                                     ;*
3574                                     ;*****
3575
3576
3577
3578
3579
3580
3581
3582
3583
3584
3585
3586
3587
3588
3589
3590
3591
3592
3593
3594
3595
3596
3597
3598
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
3632
3633
3634
3635
3636
3637
3638
3639
3640
3641
3642
3643
3644
3645
3646
3647
3648
3649
3650
3651
3652
3653
3654
3655
3656
3657
3658
3659
3660
3661
3662
3663
3664
3665
3666
3667
3668
3669
3670
3671
3672
3673
3674
3675
3676
3677
3678
3679
3680
3681
3682
3683
3684
3685
3686
3687
3688
3689
3690
3691
3692
3693
3694
3695
3696
3697
3698
3699
3700
3701
3702
3703
3704
3705
3706
3707
3708
3709
3710
3711
3712
3713
3714
3715
3716
3717
3718
3719
3720
3721
3722
3723
3724
3725
3726
3727
3728
3729
3730
3731
3732
3733
3734
3735
3736
3737
3738
3739
3740
3741
3742
3743
3744
3745
3746
3747
3748
3749
3750
3751
3752
3753
3754
3755
3756
3757
3758
3759
3760
3761
3762
3763
3764
3765
3766
3767
3768
3769
3770
3771
3772
3773
3774
3775
3776
3777
3778
3779
3780
3781
3782
3783
3784
3785
3786
3787
3788
3789
3790
3791
3792
3793
3794
3795
3796
3797
3798
3799
3800
3801
3802
3803
3804
3805
3806
3807
3808
3809
3810
3811
3812
3813
3814
3815
3816
3817
3818
3819
3820
3821
3822
3823
3824
3825
3826
3827
3828
3829
3830
3831
3832
3833
3834
3835
3836
3837
3838
3839
3840
3841
3842
3843
3844
3845
3846
3847
3848
3849
3850
3851
3852
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
3882
3883
3884
3885
3886
3887
3888
3889
3890
3891
3892
3893
3894
3895
3896
3897
3898
3899
3900
3901
3902
3903
3904
3905
3906
3907
3908
3909
3910
3911
3912
3913
3914
3915
3916
3917
3918
3919
3920
3921
3922
3923
3924
3925
3926
3927
3928
3929
3930
3931
3932
3933
3934
3935
3936
3937
3938
3939
3940
3941
3942
3943
3944
3945
3946
3947
3948
3949
3950
3951
3952
3953
3954
3955
3956
3957
3958
3959
3960
3961
3962
3963
3964
3965
3966
3967
3968
3969
3970
3971
3972
3973
3974
3975
3976
3977
3978
3979
3980
3981
3982
3983
3984
3985
3986
3987
3988
3989
3990
3991
3992
3993
3994
3995
3996
3997
3998
3999
4000

```

3575	016154	000000								
3576	016156	040277	000000	000000	2\$:	.WORD	40277,0,0,0			;FSRC
3577	016164	000000								
3578	016166	034200	000000	000000	3\$:	.WORD	34200,0,0,0			;RES
3579	016174	000000								
3580	016176	000200			4\$:	200				;FPS BEFORE EXECUTION.
3581	016200	000200				200				;FPS AFTER EXECUTION.
3582	016202	177777	177777	177777	5\$:	.WORD	-1,-1,-1,-1			;ERROR RES.
3583	016210	177777								
3584	016212	104033			6\$:	ERROR	33			
3585										
3586										
3587	016214									;DIVD WITH AC NEGATIVE AND FSRC POSITIVE IN TRUNCATE MODE.
3588	016214	104413			0002:	LPERR				;SET UP THE LOOP ON ERROR ADDRESS.
3589	016216	004737	016630			JSR	PC,0#DIVDSUB			
3590	016222	134277	000000	000000	1\$:	.WORD	134277,0,0,0			;AC
3591	016230	000000								
3592	016232	040277	000000	000000	2\$:	.WORD	40277,0,0,0			;FSRC
3593	016240	000000								
3594	016242	134200	000000	000000	3\$:	.WORD	134200,0,0,0			;RES
3595	016250	000000								
3596	016252	000207			4\$:	207				;FPS BEFORE EXECUTION.
3597	016254	000210				210				;FPS AFTER EXECUTION.
3598	016256	177777	177777	177777	5\$:	.WORD	-1,-1,-1,-1			;ERROR RESULT
3599	016264	177777								
3600	016266	104033			6\$:	ERROR	33			
3601										
3602										
3603	016270				0003:					;DIVD TEST WITH OPERANDS BOTH NEGATIVE AND IN TRUNCATE MODE.
3604	016270	104413				LPERR				;SET UP THE LOOP ON ERROR ADDRESS.
3605	016272	004767	000332			JSR	PC,0#DIVDSUB			
3606	016276	134300	000000	000000	1\$:	.WORD	134300,0,0,1			;AC
3607	016304	000001								
3608	016306	140300	000000	000000	2\$:	.WORD	140300,0,0,0			;FSRC
3609	016314	000000								
3610	016316	034200	000000	000000	3\$:	.WORD	34200,0,0,0			;RES
3611	016324	000000								
3612	016326	000250			4\$:	250				;FPS BEFORE EXECUTION.
3613	016330	000240				240				;FPS AFTER EXECUTION.
3614	016332	034200	000000	000000	5\$:	.WORD	34200,0,0,1			;ERROR RES.
3615	016340	000001								
3616	016342	104035			6\$:	ERROR	35			;TRUNCATION ERROR.
3617										
3618										
3619	016344				0004:					;DIVD WITH AC POSITIVE AND FSRC NEGATIVE IN ROUND MODE.
3620	016344	104413				LPERR				;SET UP THE LOOP ON ERROR ADDRESS.
3621	016346	004737	016630			JSR	PC,0#DIVDSUB			
3622	016352	034300	000000	000000	1\$:	.WORD	34300,0,0,1			;AC
3623	016360	000001								
3624	016362	140300	000000	000000	2\$:	.WORD	140300,0,0,0			;FSRC
3625	016370	000000								
3626	016372	134200	000000	000000	3\$:	.WORD	134200,0,0,1			;RES
3627	016400	000001								
3628	016402	000207			4\$:	207				;FPS BEFORE EXECUTION.
3629	016404	000210				210				;FPS AFTER EXECUTION.
3630	016406	134200	000000	000000	5\$:	.WORD	134200,0,0,0			;ERROR RES.

```

3631 C16414 000000
3632 016416 104036          6$: ERROR 36          ;ROUND ERROR.
3633
3634          ;DIVD TEST.
3635          0005:
3636 016420          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
3637 016422 004737 016630 JSR PC, @DIVDSUB
3638 016426 100400 000000 .WORD 100400,0,0,0 ;AC
3639 016434 000000          1$:
3640 016436 000500 000000 2$: .WORD 500,0,0,0 ;FSRC
3641 016444 000000          3$:
3642 016446 140052 125252 .WORD 140052,125252 ;RES
3643 016452 125252 125252 .WORD 125252,125252
3644 016456 007647          4$: 7647 ;FPS BEFORE EXECUTION.
3645 016460 007650          .WORD 7650 ;FPS AFTER EXECUTION.
3646 016462 177777 177777 5$: .WORD -1,-1,-1,-1 ;ERROR RES.
3647 016470 177777
3648 016472 104033          6$: ERROR 33
3649
3650          ;DIVD TEST WITH AC POSITIVE AND FSRC NEGATIVE IN ROUND MODE.
3651          0006:
3652 C16474          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
3653 016474 104413 JSR PC, @DIVDSUB
3654 016476 004737 016630 .WORD 400,0,0,0 ;AC
3655 016502 000400 000000 1$:
3656 016510 000000          2$: .WORD 100500,0,0,0 ;FSRC
3657 016512 100500 000000 3$: .WORD 140052,125252 ;RES
3658 016520 000000          .WORD 125252,125253
3659 016522 140052 125252 4$: 7707 ;FPS BEFORE EXECUTION.
3660 016526 125252 125253 .WORD 7710 ;FPS AFTER EXECUTION.
3661 016532 007707          5$: .WORD -1,-1,-1,-1 ;ERROR RES.
3662 016534 007710
3663 016536 177777 177777 6$:
3664 016544 177777          ERROR 33
3665 016546 104033
3666
3667          ;DIVD TEST.
3668          0007:
3669 016550          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
3670 016552 004737 016630 JSR PC, @DIVDSUB
3671 016556 170360 170360 1$. .WORD 170360,170360 ;AC
3672 016562 170360 170360 .WORD 170360,170360 ;FSRC
3673 016566 170360 170360 2$. .WORD 170360,170360
3674 016572 170360 170360 .WORD 170360,170360
3675 016576 040200 000000 3$. .WORD 40200,0,0,0 ;RES
3676 016604 000000          4$: 7717 ;FPS BEFORE EXECUTION.
3677 016606 007717          .WORD 7700 ;FPS AFTER EXECUTION.
3678 016610 007700          5$: .WORD -1,-1,-1,-1 ;ERROR RES.
3679 016612 177777 177777 6$:
3680 016620 177777          ERROR 33
3681 016622 104033
3682
3683 C16624 000137 017070 JMP @DDDDONE ;GO TO NEXT TEST.
3684
3685
3686

```

;THIS SUBROUTINE, DIVDSUB, IS CALLED TO SET UP, EXECUTE

3657
3688
3689
3690
3691
3692
3693
3694
3695
3696
3697
3698
3699
3700
3701
3702
3703
3704
3705
3706
3707
3708
3709
3710
3711
3712
3713
3714
3715
3716
3717
3718
3719
3720
3721
3722
3723
3724
3725
3726
3727
3728
3729
3730
3731
3732
3733
3734
3735
3736
3737
3738
3739
3740
3741
3742

016630 012601
016632 012700 000200
016636 170100
016640 010100
016642 172410
016644 016100 000030
016650 170100
016652 012737 016656 DC1236
016660 010100
016662 062700 000010
016666 174410
016670 170204
016672 012700 000200
016676 170100
016700 012700 017060
016704 174010
016706 010102
016710 010237 001240
016714 062702 000010
016720 010237 001242
016724 062702 000010
016730 010237 001244
016734 012737 017060 001246
016742 010437 001250
016746 016137 000032 001252

AND CHECK THE RESULT OF A DIVD INSTRUCTION. IT IS CALLED THUS:

```

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

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

```

DIVDSUB:  MOV      (SP)+,R1           ;GET A POINTER TO THE ARGUMENTS.
          MOV      @200,R0           ;SET FD MODE.
          LDFPS   R0
          MOV      R1,R0             ;SET UP THE ACD OPERAND.
          LOD      (R0),ACD
          MOV      30(R1),R0         ;LOAD THE FPS.
          LDFPS   R0
          MOV      @18,@STMP2
          MOV      R1,R0             ;ESTABLISH A POINTER TO FSRC.
          ADD     @10,R0
          15:  DIVD   (R0),ACD        ;EXECUTE THE TEST INSTRUCTION.
          STFPS   R4                 ;GET THE FPS.
          MOV      @200,R0           ;SET FD MODE
          LDFPS   R0
          MOV      @DIVDT,R0         ;GET THE RESULT.
          STD     ACD,(R0)
          MOV      R1,R2             ;SAVE DATA IN CASE OF ERROR.
          MOV      R2,@STMP3
          ADD     @10,R2
          MOV      R2,@STMP4
          ADD     @10,R2
          MOV      R2,@STMP5
          MOV      @DIVDT,@STMP6
          MOV      R4,@STMP7
          MOV      32(R1),@STMP10
    
```

E06

```

3743
3744 016754 010102      MOV      R1,R2      ;CHECK THE RESULT.
3745 016756 062702 000020  ADD      #20,R2
3746 016762 012703 017060  MOV      #DIVDT,R3
3747 016766 012705 000004  MOV      #4,R5
3748 016772 022223      25:  CMP      (R2)+,(R3)+
3749 016774 001006      BNE     10$        ;BRANCH IF RESULT INCORRECT.
3750 016776 077503      SOB     R5,25$
3751
3752 017000 026104 000032  CMP      32(R1),R4  ;IS FPS CORRECT?
3753 017004 001023      BNE     15$        ;BRANCH IF INCORRECT.
3754 017006 000161 000046  JMP      46(R1)    ;RETURN.
3755
3756 017012 010102      10$:  MOV      R1,R2      ;WAS INCORRECT RESULT ANTICIPATED?
3757 017014 062702 000034  ADD      #34,R2
3758 017020 012703 017060  MOV      #DIVDT,R3
3759 017024 012705 000004  MOV      #4,R5
3760 017030 022223      11$:  CMP      (R2)+,(R3)+
3761 017032 001005      BNE     12$        ;BRANCH IF NO.
3762 017034 077503      SOB     R5,11$
3763 017036 010102      MOV      R1,R2      ;IF THE INCORRECT RESULT WAS
3764 017040 062702 000044  ADD      #44,R2    ;ANTICIPATED RETURN TO THE
3765                                     ;ERROR REPORT IN THE CALLING
3766 017044 000112      JMP      (R2)      ;ROUTINE.
3767
3768 017046      12$:
3769 017046 104033      13$:  ERROR   33        ;REPORT RESULT INCORRECT.
3770 017050 000161 000046  14$:  JMP      46(R1)
3771
3772 017054      15$:
3773 017054 104033      16$:  ERROR   34        ;REPORT FPS INCORRECT.
3774 017056 000774      BR      14$
3775
3776 017060 000000 100000 000000  DIVDT:  .WORD  0,0,0,0
3777 017066 000000
3778
3779 017070      DDDDONE:
3780 017070 104412      RSETUP
3781                                     ;GO INITIALIZE THE FPS AND STACK; AND
3782                                     ;SEE IF THE USER HAS EXPRESSED
3783                                     ;THE DESIRE TO CHANGE THE SOFTWARE
3784                                     ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
3785                                     ;THE USER TYPED CONTROL G?).
3786
3787
3788 ;*****
3789 ;*TEST 10      MULF TEST
3790 ;*
3791 ;*THIS IS A TEST OF THE MULF INSTRUCTION. IT MAKES USE OF A SUBROUTINE
3792 ;*TO SET UP THE OPERANDS, EXECUTE THE MULF INSTRUCTION AND CHECK THE
3793 ;*RESULTS.
3794 ;*
3795 ;*****
3796 017072 000004  TST10:  SCOPE
3797
3798 ;MULF WITH (FSRC=AC=0)

```

```

3799 017074
3800 017074 104413
3801 017076 004737 017654
3802 017102 000000 000000
3803 017106 000000 000000
3804 017112 000000 000000
3805 017116 007517
3806 017120 007504
3807 017122 177777 177777
3808 017126 104037
3809
3810
3811 017130
3812 017130 104413
3813 017132 004737 017654
3814 017136 071625 034435
3815 017142 000000 000000
3816 017146 000000 000000
3817 017152 000013
3818 017154 000004
3819 017156 177777 177777
3820 017162 104037
3821
3822
3823 017164
3824 017164 104413
3825 017166 004737 017654
3826 017172 000000 000000
3827 017176 071625 153443
3828 017202 000000 000000
3829 017206 007500
3830 017210 007504
3831 017212 177777 177777
3832 017216 104037
3833
3834
3835 017220
3836 017220 104413
3837 017222 004737 017654
3838 017226 040200 000000
3839 017232 040177 177777
3840 017236 040177 177777
3841 017242 000017
3842 017244 000000
3843 017246 140177 177777
3844 017252 104041
3845
3846
3847 017254
3848 017254 104413
3849 017256 004767 000372
3850 017262 040177 177777
3851 017266 040200 000000
3852 017272 040177 177777
3853 017276 000040
3854 017300 000040

```

```

EEE1:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC, @MULFSUB
1S: .WORD 0,0 ;AC
2S: .WORD 0,0 ;FSRC
3S: .WORD 0,0 ;RES
4S: 7517 ;FPS BEFORE EXECUTION.
7504 ;FPS AFTER EXECUTION.
5S: .WORD -1,-1
6S: ERROR 37

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

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

;MULF WITH AC POSITIVE AND FSRC POSITIVE IN ROUND MODE.
EEE4:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC, @MULFSUB
1S: .WORD 40200,0 ;AC
2S: .WORD 40177,-1 ;FSRC
3S: .WORD 40177,-1 ;RES
4S: 17 ;FPS BEFORE EXECUTION.
0 ;FPS AFTER EXECUTION.
5S: .WORD 140177,-1 ;ERROR RES.
6S: 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
1S: .WORD 40177,-1 ;AC
2S: .WORD 40200,0 ;FSRC
3S: .WORD 40177,-1 ;RES
4S: 40 ;FPS BEFORE EXECUTION.
40 ;FPS AFTER EXECUTION.

```

```

3855 017302 037777 177777 55: .WORD 37777,-1 ;ERROR RES.
3856 017306 104042 55: ERROR 42 ;ST 252 TO 044 INTO 444 (BUT Y62)
;MUL. NORMALIZATION FAILURE.
;MULF WITH BOTH OPERANDS POSITIVE NORMALIZE TEST.
EEEE:
3860 017310 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
3861 017310 104413 JSR PC,2#MULFSUB
3862 017312 004737 017654 15: .WORD 40100,0 ;AC
3863 017316 040100 000000 25: .WORD 40100,0 ;FSRC
3864 017322 040100 000000 35: .WORD 40020,0 ;RES
3865 017326 040020 000000 45: 12 ;FPS BEFORE EXECUTION.
3866 017332 000012 0 ;FPS AFTER EXECUTION.
3867 017334 000000 55: .WORD 42040,0 ;ERROR RES.
3868 017336 042040 000000 65: ERROR 43 ;ST 252 TO 444 INTO 042 (BUT Y62)
;MUL. NORMALIZATION FAILURE.
;MULF WITH BOTH OPERANDS POSITIVE IN ROUND MODE.
EEEE7:
3873 017344 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
3874 017344 104413 JSR PC,2#MULFSUB
3875 017346 004737 017654 15: .WORD 17500,0 ;AC
3876 017352 017500 000000 25: .WORD 23652,125252 ;FSRC
3877 017356 023652 125252 35: .WORD 3177,-1 ;RES
3878 017362 003177 177777 45: 7417 ;FPS BEFORE EXECUTION.
3879 017366 007417 7400 ;FPS AFTER EXECUTION.
3880 017370 007400 55: .WORD -1,-1 ;ERROR RES.
3881 017372 177777 177777 65: ERROR 37 ;BAD SIGN.
;MULF WITH AC POSITIVE AND FSRC NEGATIVE IN ROUND MODE.
EEEE8:
3885 017400 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
3886 017400 104413 JSR PC,2#MULFSUB
3887 017402 004737 017654 15: .WORD 40342,0 ;AC
3888 017406 040342 000000 25: .WORD 176542,0 ;FSRC
3889 017412 176542 000000 35: .WORD 176707,102000 ;RES
3890 017416 176707 102000 45: 7 ;FPS BEFORE EXECUTION.
3891 017422 000007 10 ;FPS AFTER EXECUTION.
3892 017424 000010 75: .WORD 76507,102000 ;ERROR RES.
3893 017426 076507 102000 65: ERROR 41 ;BAD SIGN.
;MULF WITH AC NEGATIVE AND FSRC POSITIVE IN ROUND MODE.
EEEE9:
3897 017434 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
3898 017434 104413 JSR PC,2#MULFSUB
3899 017436 004737 017654 15: .WORD 140200,0 ;AC
3900 017442 140200 000000 25: .WORD 7417,7417 ;FSRC
3901 017446 007417 007417 35: .WORD 107417,7417 ;RES
3902 017452 107417 007417 45: 0 ;FPS BEFORE EXECUTION.
3903 017456 000000 10 ;FPS AFTER EXECUTION.
3904 017460 000010 55: .WORD 7417,7417 ;ERROR RES.
3905 017462 007417 007417 65: ERROR 41 ;BAD SIGN.
;MULF WITH BOTH OPERANDS NEGATIVE IN ROUND MODE.
EEEE10:
3908 017470 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
3909 017470 104413

```

DFPB.F11 01-NOV-76 21:06

3911	017472	004737	017654
3912	017476	144600	000000
3913	017502	154000	000000
3914	017506	060400	000000
3915	017512	000017	
3916	017514	000000	
3917	017516	160400	000000
3918	017522	104041	
3919			
3920			
3921	017524		
3922	017524	104413	
3923	017526	004737	017654
3924	017532	140300	000000
3925	017536	160000	000001
3926	017542	060100	000002
3927	017546	000010	
3928	017550	000000	
3929	017552	060100	000001
3930	017556	104044	
3931			
3932			
3933	017560		
3934	017560	104413	
3935	017562	004737	017654
3936	017566	060000	000001
3937	017572	140300	000000
3938	017576	160100	000001
3939	017602	007547	
3940	017604	007550	
3941	017606	160100	000001
3942	017612	104045	
3943			
3944			
3945	017614		
3946	017614	104413	
3947	017616	004737	017654
3948	017622	040277	000000
3949	017626	060000	000001
3950	017632	060077	000001
3951	017636	000014	
3952	017640	000000	
3953	017642	060077	000002
3954	017646	104044	
3955			
3956	017650	000167	000224
3957			
3958			
3959			
3960			
3961			
3962			
3963			
3964			
3965			
3966			

```

      JSR      PC,@#MULFSJB
1$:      .WORD 144600,0      :AC
2$:      .WORD 154000,0      :FSRC
3$:      .WORD 60400,0       :RES
4$:      17                  :FPS BEFORE EXECUTION.
      0                      :FPS AFTER EXECUTION.
5$:      .WORD 160400,0      :ERROR RES.
6$:      ERROR 41           :BAD SIGN.

;MULF BOTH OPERANDS NEGATIVE IN ROUND MODE.
EEE11:
      LPERR                    ;SET UP THE LOOP ON ERROR ADDRESS.
      JSR      PC,@#MULFSUB
1$:      .WORD 140300,0      :AC
2$:      .WORD 160000,1      :FSRC
3$:      .WORD 60100,2       :RES
4$:      10                  :FPS BEFORE EXECUTION.
      0                      :FPS AFTER EXECUTION.
5$:      .WORD 60100,1      :ERROR RES.
6$:      ERROR 44           :ROUND FAILURE.

;MULF WITH AC POSITIVE AND FSRC NEGATIVE IN TRUNCATE MODE.
EEE12:
      LPERR                    ;SET UP THE LOOP ON ERROR ADDRESS.
      JSR      PC,@#MULFSUB
1$:      .WORD 60000,1       :AC
2$:      .WORD 140300,0      :FSRC
3$:      .WORD 160100,1      :RES
4$:      7547                :FPS BEFORE EXECUTION.
      7550                  :FPS AFTER EXECUTION.
5$:      .WORD 160100,1      :ERROR RES.
6$:      ERROR 45           :TRUNCATION ERROR.

;MULF WITH AC POSITIVE AND FSRC POSITIVE IN ROUND MODE.
EEE13:
      LPERR                    ;SET UP THE LOOP ON ERROR ADDRESS.
      JSR      PC,@#MULFSUB
1$:      .WORD 40277,0       :AC
2$:      .WORD 60000,1      :FSRC
3$:      .WORD 60077,1      :RES
4$:      14                  :FPS BEFORE EXECUTION.
      0                      :FPS AFTER EXECUTION.
5$:      .WORD 60077,2      :ERROR RES.
6$:      ERROR 44           :ROUND FAILURE. CONSTANT BAD.

      JMP      EEEDONE          ;GO TO THE NEXT TEST.

;THIS SUBROUTINE, MULFSUB, IS CALLED TO SET UP, EXECUTE
;AND CHECK THE RESULT OF A MULF INSTRUCTION. IT IS CALLED THUS:
:
:      JSR      PC,@#MULFSUB
:      ACARG:  .WORD X,X      :AC OPERAND
:      FSRCARG: .WORD X,X      :FSRC OPERAND
:      RES:    .WORD X,X      :EXPECTED RESULT
:      FPSB:   .WORD X        :FPS BEFORE EXECUTION
:      FPSA:   .WORD X        :FPS AFTER EXECUTION

```



```

3967 : ERRES: .WORD X,X ;ERROR RESULT
3968 : ERR: ERROR X ;RESULT ERROR
3969 : CONT: ;PETJRN ADDRESS
3970
3971
3972
3973
3974
3975
3976
3977
3978
3979
3980
3981
3982
3983
3984
3985
3986
3987
3988
3989
3990
3991
3992
3993
3994
3995
3996
3997
3998
3999
4000
4001
4002
4003
4004
4005
4006
4007
4008
4009
4010
4011
4012
4013
4014
4015
4016
4017
4018
4019
4020
4021
4022
    
```

```

: THE OPERANDS ARE SET UP (USING ACD 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.
    
```

```

3984 017654 012601 MULFSUB: MOV (SP)+,R1 ;GET A POINTER TO THE ARGUMENTS.
3985 017656 012700 000200 MOV #200,R0 ;SET FD MODE.
3986 017662 170100 LDFPS R0
3987 017664 010100 MOV R1,R0 ;LOAD THE AC OPERAND.
3988 017666 172410 LDD (R0),ACD
3989 017670 016100 000014 MOV 14(R1),R0 ;LOAD THE FPS
3990 017674 170100 LDFPS R0
3991 017676 012737 017712 001236 MOV #15,2#STMP2
3992 017704 010100 MOV R1,R0
3993 017706 062700 000004 ADD #4,R0 ;ESTABLISH A POINTER TO FSRC.
3994
3995 017712 171010 15: MULF (R0),ACD ;TEST INSTRUCTION.
3996
3997 017714 170204 STFPS R4 ;GET THE FPS.
3998 017716 012700 000200 MOV #200,R0 ;SET FD MODE
3999 017722 170100 LDFPS R0
4000
4001 017724 012700 020070 MOV #MULFT,R0 ;GET THE RESULT OF THE MULF.
4002 017730 174010 STD ACD,(R0)
4003
4004 017732 010102 MOV R1,R2 ;SAVE THE DATA IN CASE OF ERROR.
4005 017734 010237 001240 MOV R2,2#STMP3
4006 017740 062702 000004 ADD #4,R2
4007 017744 010237 001242 MOV R2,2#STMP4
4008 017750 062702 000004 ADD #4,R2
4009 017754 010237 001244 MOV R2,2#STMP5
4010 017760 012737 020070 001246 MOV #MULFT,2#STMP6
4011 017766 010437 001250 MOV R4,2#STMP7
4012 017772 016137 000016 001252 MOV 16(R1),2#STMP10
4013
4014 020000 021061 000010 CMP (R0),10(R1) ;IS THE RESULT CORRECT?
4015 020004 001011 BNE 10$ ;IF INCORRECT BRANCH.
4016 020006 026061 000002 000012 CMP 2(R0),12(R1)
4017 020014 001005 BNE 10$
4018
4019 020016 026104 000016 CMP 16(R1),R4 ;IS FPS CORRECT?
4020 020022 001020 BNE 15$ ;IF INCORRECT BRANCH.
4021 020024 000161 000026 JMP 26(R1) ;IF NO ERRORS OCCURRED RETURN.
4022
    
```


K06

```

4079
4080 ;MULD TEST WITH BOTH OPERANDS POSITIVE TRUNCATION TEST.
4081 FFF2:
4082 020160 104413 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
4083 020162 004767 000202 JSR PC,MULDSUB
4084 020166 065400 000000 000000 1$: .WORD 65400,0,0,1 ;AC
4085 020174 000001 2$: .WORD 37577,-1,-1,-2 ;FSRC
4086 020176 037577 177777 177777 2$: .WORD 37577,-1,-1,-2 ;FSRC
4087 020204 177776 3$: .WORD 64777,-1,-1,-1 ;RES
4088 020206 064777 177777 177777 4$: 247 ;FPS BEFORE EXECUTION.
4089 020214 177777 240 ;FPS AFTER EXECUTION.
4090 020216 000247 5$: .WORD 65000,0,0,0 ;ERROR RES.
4091 020220 000240 6$: ERROR 50 ;TRUNCATION ERROR.
4092 020222 065000 000000 000000
4093 020230 000000
4094 020232 104050
4095
4096 ;MULD TEST WITH BOTH OPERANDS NEGATIVE IN ROUND MODE.
4097 FFF3:
4098 020234 104413 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
4099 020236 004737 020370 JSR PC,MULDSUB
4100 020242 137577 177777 177777 1$: .WORD 137577,-1,-1,-2 ;AC
4101 020250 177776 2$: .WORD 165400,0,0,1 ;FSRC
4102 020252 165400 000000 000000 3$: .WORD 65000,0,0,0 ;RES
4103 020260 000001 4$: 7717 ;FPS BEFORE EXECUTION.
4104 020262 065000 000000 000000 7700 ;FPS AFTER EXECUTION.
4105 020270 000000 5$: .WORD 64777,-1,-1,-1 ;ERROR RES.
4106 020272 007717
4107 020274 007700 6$: ERROR 51 ;ROUND ERROR.
4108 020276 064777 177777 177777
4109 020304 177777
4110 020306 104051
4111
4112 ;MULD TEST WITH AC POSITIVE AND FSRC NEGATIVE IN ROUND MODE.
4113 FFF4:
4114 020310 104413 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
4115 020312 004737 020370 JSR PC,MULDSUB
4116 020316 017500 000000 000000 1$: .WORD 17500,0,0,0 ;AC
4117 020324 000000 2$: .WORD 123652,125252 ;FSRC
4118 020326 123652 125252 3$: .WORD 125252,125252 ;RES
4119 020332 125252 177777 177777 4$: 200 ;FPS BEFORE EXECUTION.
4120 020336 103177 177777 177777 210 ;FPS AFTER EXECUTION.
4121 020344 177777 5$: .WORD 103200,0,0,0 ;ERROR RES.
4122 020346 000200 6$: ERROR 52 ;ROUND ERROR (BAD CONSTANT).
4123 020350 000210 JMP FFFDONE
4124 020352 103200 000000 000000
4125 020360 000000
4126 020362 104052
4127
4128 020364 000167 000240
4129
4130 ;THIS SUBROUTINE, MULDSUB, IS CALLED TO SET UP, EXECUTE
4131 ;AND CHECK THE RESULT OF A MULD INSTRUCTION. IT IS CALLED THUS:
4132
4133
4134
4135
4136
4137
4138
4139
4140
4141
4142
4143
4144
4145
4146
4147
4148
4149
4150
4151
4152
4153
4154
4155
4156
4157
4158
4159
4160
4161
4162
4163
4164
4165
4166
4167
4168
4169
4170
4171
4172
4173
4174
4175
4176
4177
4178
4179
4180
4181
4182
4183
4184
4185
4186
4187
4188
4189
4190
4191
4192
4193
4194
4195
4196
4197
4198
4199
4200
4201
4202
4203
4204
4205
4206
4207
4208
4209
4210
4211
4212
4213
4214
4215
4216
4217
4218
4219
4220
4221
4222
4223
4224
4225
4226
4227
4228
4229
4230
4231
4232
4233
4234
4235
4236
4237
4238
4239
4240
4241
4242
4243
4244
4245
4246
4247
4248
4249
4250
4251
4252
4253
4254
4255
4256
4257
4258
4259
4260
4261
4262
4263
4264
4265
4266
4267
4268
4269
4270
4271
4272
4273
4274
4275
4276
4277
4278
4279
4280
4281
4282
4283
4284
4285
4286
4287
4288
4289
4290
4291
4292
4293
4294
4295
4296
4297
4298
4299
4300
4301
4302
4303
4304
4305
4306
4307
4308
4309
4310
4311
4312
4313
4314
4315
4316
4317
4318
4319
4320
4321
4322
4323
4324
4325
4326
4327
4328
4329
4330
4331
4332
4333
4334
4335
4336
4337
4338
4339
4340
4341
4342
4343
4344
4345
4346
4347
4348
4349
4350
4351
4352
4353
4354
4355
4356
4357
4358
4359
4360
4361
4362
4363
4364
4365
4366
4367
4368
4369
4370
4371
4372
4373
4374
4375
4376
4377
4378
4379
4380
4381
4382
4383
4384
4385
4386
4387
4388
4389
4390
4391
4392
4393
4394
4395
4396
4397
4398
4399
4400
4401
4402
4403
4404
4405
4406
4407
4408
4409
4410
4411
4412
4413
4414
4415
4416
4417
4418
4419
4420
4421
4422
4423
4424
4425
4426
4427
4428
4429
4430
4431
4432
4433
4434
4435
4436
4437
4438
4439
4440
4441
4442
4443
4444
4445
4446
4447
4448
4449
4450
4451
4452
4453
4454
4455
4456
4457
4458
4459
4460
4461
4462
4463
4464
4465
4466
4467
4468
4469
4470
4471
4472
4473
4474
4475
4476
4477
4478
4479
4480
4481
4482
4483
4484
4485
4486
4487
4488
4489
4490
4491
4492
4493
4494
4495
4496
4497
4498
4499
4500

```

```

4135 : FSRCARG: .WORD X,X,X,X ;FSRC OPERAND
4136 : RES: .WORD X,X,X,X ;EXPECTED RESULT
4137 : FPSB: .WORD X ;FPS BEFORE EXECUTION
4138 : FPSA: .WORD X ;FPS AFTER EXECUTION
4139 : ERRES: .WORD X,X,X,X ;ERROR RESULT
4140 : ERR: ERROR X ;RESULT ERROR
4141 : CONT: ;RETURN ADDRESS
    
```

```

4142 : THE OPERANDS ARE SET UP (USING ACO FOR THE AC OPERAND). THEN
4143 : FPSB IS LOADED INTO THE FPS. THE INSTRUCTION, MULDT IS EXECUTED.
4144 : AFTER THE EXECUTION THE RESULT IS CHECKED AGAINST THE
4145 : EXPECTED CORRECT RESULT, RES. IF IT IS CORRECT THEN THE FPS
4146 : IS CHECKED WITH THE EXPECTED CORRECT FPS, FPSA. IF THE FPS WAS
4147 : INCORRECT THEN IT IS REPORTED. IF THE RESULT WAS INCORRECT IT
4148 : IS COMPARED WITH ERRES IN AN ATTEMPT TO ANALYSE THE ERROR. IF
4149 : THE INCORRECT RESULT MATCHED ERRES THEN CONTROL IS PASSED TO
4150 : THE ERROR CALL AT ERR. IF THE INCORRECT RESULT DID NOT MATCH ERRES
4151 : THEN THE FAILURE IS REPORTED IN MULDSUB AND CONTROL IS PASSED TO
4152 : CONT. IF NO ERRORS ARE DETECTED THEN MULDSUB RETURNS CONTROL
4153 : TO CONT.
    
```

```

4154
4155
4156 020370 012601 000200 MULDSUB: MOV (SP)+,R1 ;GET A POINTER TO THE ARGUMENTS.
4157 020372 012700 000200 MOV #200,R0 ;SET FD MODE.
4158 020376 170100 LDFPS R0
4159
4160 020400 010100 MOV R1,R0 ;SET UP THE ACO OPERAND.
4161 020402 172410 LDD (R0),ACO
4162 020404 016100 000030 MOV 30(R1),R0 ;LOAD THE FPS.
4163 020410 17C100 LDFPS R0
4164
4165 020412 012737 020426 001236 MOV #15,0#STMP2
4166 020420 010100 MOV R1,R0 ;ESTABLISH A POINTER TO FSRC.
4167 020422 062700 000010 ADD #10,R0
4168
4169 020426 171010 15: MULDT (R0),ACO ;EXECUTE THE TEST INSTRUCTION.
4170
4171 020430 170204 STFPS R4 ;GET THE FPS.
4172 020432 012700 000200 MOV #200,R0 ;SET FD MODE.
4173 020436 170100 LDFPS R0
4174
4175 020440 012700 020620 MOV #MULDT,R0 ;GET THE RESULT.
4176 020444 174010 STD ACO,(R0)
4177
4178 020446 010102 MOV R1,R2 ;SAVE DATA IN CASE OF ERROR.
4179 020450 010237 001240 MOV R2,0#STMP3
4180 020454 062702 000010 ADD #10,R2
4181 020460 010237 001242 MOV R2,0#STMP4
4182 020464 062702 000010 ADD #10,R2
4183 020470 010237 001244 MOV R2,0#STMP5
4184 020474 012737 020620 001246 MOV #MULDT,0#STMP6
4185 020502 010437 001250 MOV R4,0#STMP7
4186 020506 016137 000032 001252 MOV 32(R1),0#STMP10
4187
4188 020514 010102 MOV R1,R2 ;CHECK THE RESULT.
4189 020516 062702 000020 ADD #20,R2
4190 020522 012703 020620 MOV #MULDT,R3
    
```

```

4191 020526 012705 000004      MOV      #4,R5
4192 020532 022223      2$:    CMP      (R2)+,(R3)+
4193 020534 001006      BNE     10$      ;BRANCH IF RESULT INCORRECT.
4194 020536 077503      SOB     R5,2$
4195
4196 020540 026104 000032      CMP      32(R1),R4      ;IS FPS CORRECT?
4197 020544 001023      BNE     15$      ;BRANCH IF INCORRECT.
4198 020546 000161 000046      JMP      46(R1)      ;RETURN.
4199
4200 020552 010102      10$:   MOV      R1,R2      ;WAS INCORRECT RESULT ANTICIPATED?
4201 020554 062702 000034      ADD     #34,R2
4202 020560 012703 020620      MOV     #MULDT,R3
4203 020564 012705 000004      MOV     #4,R5
4204 020570 022223      11$:   CMP      (R2)+,(R3)+
4205 020572 001005      BNE     12$      ;BRANCH IF NO.
4206 020574 077503      SOB     R5,11$
4207 020576 010102      MOV     R1,R2      ;IF THE INCORRECT RESULT WAS
4208 020600 062702 000044      ADD     #44,R2      ;ANTICIPATED RETURN TO THE
4209                                     ;ERROR REPORT IN THE CALLING
4210 020604 000112      JMP      (R2)      ;ROUTINE.
4211
4212 020606      12$:   ;REPORT RESULT INCORRECT.
4213 020606 104246      13$:   ERROR    246
4214 020610 000161 000046      14$:   JMP      46(R1)
4215
4216 020614      15$:   ;REPORT FPS INCORRECT.
4217 020614 104046      16$:   ERROR    46
4218 020616 000774      BR      14$
4219
4220 020620 000000 000000 000000 MULDT: .WORD 0,0,0,0
4221 020626 000000
4222
4223 020630      FFFDONE:
4224 020630 104412      RSETUP      ;GO INITIALIZE THE FPS AND STACK; AND
4225                                     ;SEE IF THE USER HAS EXPRESSED
4226                                     ;THE DESIRE TO CHANGE THE SOFTWARE
4227                                     ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
4228                                     ;THE USER TYPED CONTROL G?).
4229
4230
4231
4232
4233      ;*****
4234      ;*TEST 12      UNDER\OVER FLOW, USING MULF WITH TRAPS DISABLED. TEST*
4235      ;*
4236      ;*THIS IS A TEST OF THE OVERFLOW AND UNDERFLOW CONDITIONS USING
4237      ;*THE MULF INSTRUCTION WITH TRAPS DISABLED. NOTE THAT A SUBROUTINE
4238      ;*IS USED TO SET UP THE OPERANDS, EXECUTE THE MULF INSTRUCTION AND
4239      ;*CHECK THE RESULTS.
4240      ;*
4241      ;*****
4241 020632 000004      TEST12: SCOPE
4242
4243      ;UNDERFLOW, WITH EXPONENT OF RESULT = -129
4244 020634      III1:
4245 020634 104413      LPERR
4246 020636 004737 021060      JSR     PC,2#OVUNFNT      ;SET UP THE LOOP ON ERROR ADDRESS.

```

4247	020642	020200	000000	1\$:	.WORD	20200,0	;AC
4248	020646	020000	000000	2\$:	.WORD	20000,0	;FSRC
4249	020652	000000	000000	3\$:	.WORD	0,0	;RES
4250	020656	177777	177777	4\$:	.WORD	-1,-1	;ERROR RES.
4251	020662	000000		5\$:	0		;FPS BEFORE EXECUTION.
4252	020664	000004			4		;FPS AFTER EXECUTION.
4253	020666	000012		6\$:	12		;FEC
4254	020670	177777			-1		;FLAG
4255	020672	104117		7\$:	ERROR	117	;ST 331 TO 155 INTO 115 (BUT FIU)
4256	020674	000401			BR	8\$	
4257	020676	104114			ERROR	114	
4258	020700			8\$:			
4259							
4260							; UNDERFLOW, WITH EXPONENT OF RESULT = -193
4261	020700			1112:			
4262	020700	104413			LPERR		; SET UP THE LOOP ON ERROR ADDRESS.
4263	020702	004737	021060		JSR	PC, 2#OVUNFNT	
4264	020706	010200	000000	1\$:	.WORD	10200,0	;AC
4265	020712	010000	000000	2\$:	.WORD	10000,0	;FSRC
4266	020716	000000	000000	3\$:	.WORD	0,0	;RES
4267	020722	010000	000000	4\$:	.WORD	10000,0	;ERROR RES.
4268	020726	005013		5\$:	5013		;FPS BEFORE EXECUTION.
4269	020730	005004			5004		;FPS AFTER EXECUTION.
4270	020732	000012		6\$:	12		;FEC
4271	020734	177777			-1		;FLAG
4272	020736	104120		7\$:	ERROR	120	;SETTING FIUV OR FIV CAUSES TRAP
4273							;WITH FIU CLEAR.
4274	020740	000401			BR	8\$	
4275	020742	104114			ERROR	114	
4276	020744			8\$:			
4277							
4278							; OVERFLOW, EXPONENT OF RESULT = 128
4279	020744			1113:			
4280	020744	104413			LPERR		; SET UP THE LOOP ON ERROR ADDRESS.
4281	020746	004737	021060		JSR	PC, 2#OVUNFNT	
4282	020752	060200	000000	1\$:	.WORD	60200,0	;AC
4283	020756	000000	000000	2\$:	.WORD	60000,0	;FSRC
4284	020760	000000	000000	3\$:	.WORD	0,0	;RES
4285	020766	060200	000000	4\$:	.WORD	60000,0	;ERROR RES.
4286	020772	000000		5\$:	0		;FPS BEFORE EXECUTION.
4287	020776	000006			6		;FPS AFTER EXECUTION.
4288	020776	000010		6\$:	10		;FEC
4289	021000	000000			0		;FLAG
4290	021002	104121		7\$:	ERROR	121	;ST 333 TO 136 INTO 116 (BUT FIV).
4291	021004	000401			BR	8\$	
4292	021006	104113			ERROR	113	
4293	021010			8\$:			
4294							
4295							; OVERFLOW, EXPONENT OF RESULT = 130
4296	021010			1114:			
4297	021010	104413			LPERR		; SET UP THE LOOP ON ERROR ADDRESS.
4298	021012	004737	021060		JSR	PC, 2#OVUNFNT	
4299	021016	060200	000000	1\$:	.WORD	60200,0	;AC
4300	021022	060200	000000	2\$:	.WORD	60200,0	;FSRC
4301	021026	000000	000000	3\$:	.WORD	0,0	;RES
4302	021032	177777	177777	4\$:	.WORD	-1,-1	;ERROR RES.

```

4303 021036 006011 55: 6011 ;FPS BEFORE EXECUTION.
4304 021040 006006 ;FPS AFTER EXECUTION.
4305 021042 000010 65: 10 ;FEC
4306 021044 000000 ;FLAG
4307 021046 104122 75: ERROR 122 ;SETTING FIUV OR FIU WITH
;FIV CLEAR CAUSES TRAP.
4308
4309 021050 000401 BR 85
4310 021052 104113 ERROR 113
4311 021054 000167 000410 85: JMP IIIDONE ;GO TO NEXT TEST.
4312
4313
4314
4315
4316
4317
4318
4319
4320
4321
4322
4323
4324
4325
4326
4327
4328
4329
4330
4331
4332
4333
4334
4335
4336
4337
4338
4339
4340
4341
4342
4343
4344
4345
4346
4347
4348
4349
4350
4351
4352
4353 021060 012601
4354 021062 012700 000200
4355 021066 170100
4356
4357 021070 010100
4358 021072 172410
  
```

: 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
ERRRES: .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 ACD 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, ERRRES. IF THE FAILURE IN THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRRES 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. 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 ACD, OPERAND.
LDO (R0),ACD
  
```

4359										
4360	021074	010102				MOV	R1,R2			;SAVE THE DATA PATTERNS IN CASE OF
4361	021076	010237	001240			MOV	R2,@STMP3			;ERROR.
4362	021102	062702	000004			ADD	#4,R2			
4363	021106	010237	001242			MOV	R2,@STMP4			
4364	021112	062702	000004			ADD	#4,R2			
4365	021116	010237	001244			MOV	R2,@STMP5			
4366	021122	016137	000022	001252		MOV	22(R1),@STMP10			
4367	021130	012737	021460	001246		MOV	@OVFNTT,@STMP6			
4368										
4369	021136	016100	000020			MOV	20(R1),R0			;LOAD THE FPS.
4370	021142	170100				LDFPS	R0			
4371	021144	012737	021166	001236		MOV	#15,@STMP2			
4372	021152	012737	021352	000244		MOV	@255,@FPVECT			;SET UP THE FP TRAP VECTOR IN CASE
4373										;OF ERROR.
4374	021160	010100				MOV	R1,R0			;COMPUTE THE ADDRESS OF FSRC.
4375	021162	062700	000004			ADD	#4,R0			
4376										
4377	021166	171010			15:	MULF	(R0),R0			;TEST INSTRUCTION.
4378										
4379	021170	170204			25:	STFPS	R4			;GET FPS.
4380	021172	170305				STST	R5			;GET FEC.
4381	021174	012700	000200			MOV	@200,R0			;SET FD MODE.
4382	021200	170100				LDFPS	R0			
4383	021202	012700	021460			MOV	@OVFNTT,R0			;GET THE RESULT.
4384	021206	174010				STD	R0,(R0)			
4385	021210	010437	001250			MOV	R4,@STMP7			
4386	021214	010537	001254			MOV	R5,@STMP11			
4387										
4388	021220	012700	021460			MOV	@OVFNTT,R0			;CHECK THE RESULT.
4389	021224	010102				MOV	R1,R2			
4390	021226	062702	000010			ADD	#10,R2			
4391	021232	012703	000002			MOV	#2,R3			
4392	021236	022022			35:	CMP	(R0)+,(R2)+			
4393	021240	001015				BNE	155			;BRANCH IF INCORRECT.
4394	021242	077303				SOB	R3,35			
4395										
4396	021244	026104	000022			CMP	22(R1),R4			;WAS FPS CORRECT?
4397	021250	001002				BNE	105			;BRANCH IF FPS IS INCORRECT.
4398										
4399	021252	000161	000036		45:	JMP	36(R1)			;RETURN, TEST COMPLETED.
4400										
4401										;REPORT INCORRECT FPS.
4402	021256	005761	000026		105:	TST	26(R1)			;WAS THE RESULT OVER OR UNDER FLOW?
4403	021262	001002				BNE	125			;BRANCH IF UNDERFLOW.
4404										
4405										;REPORT FPS BAD AFTER OVERFLOW.
4406	021264	104111			115:	ERROR	111			
4407	021266	000771				BR	45			
4408										
4409	021270				125:					;REPORT FPS BAD AFTER UNDERFLOW.
4410	021270	104112			135:	ERROR	112			
4411	021272	000767				BR	45			
4412										
4413										;RESULT INCORRECT.
4414	021274	012700	021460		155:	MOV	@OVFNTT,R0			;SEE IF FAILURE IS ANTICIPATED


```

#15 021300 010102 MOV R1,R2 ;FAILURE.
#16 021302 062702 000014 ADD #14,R2
#17 021306 012703 000002 MOV #2,R3
#18 021312 022022 165: CMP (R0)+,(R2)+
#19 021314 001007 BNE 175 ;BRANCH IF NOT ANTICIPATED.
#20 021316 077303 SOB R3,165
#21
#22 021320 010102 MOV R1,R2 ;ERROR WAS ANTICIPATED SO RETURN
#23 021322 062702 000034 ADD #34,R2 ;TO THE ERROR REPORT IN THE CALLING
#24 021326 010237 001236 MOV R2,#STMP2 ;ROUTINE.
#25 021332 000112 JMP (R2)
#26
#27 021334 005761 000026 175: TST 26(R1) ;RESULT WAS NOT ANTICIPATED
#28 ;SO ERROR MUST BE REPORTED HERE.
#29 ;FIRST SEE IF ARGUMENTS SHOULD
#30 ;HAVE RESULTED IN OVERFLOW OR UNDER
#31 ;FLOW BY LOOKING AT THE FLAG.
#32 021340 001002 BNE 195 ;BRANCH IF UNDERFLOW EXPECTED.
#33
#34 021342 104113 185: ERROR 113 ;REPORT RESULT INCORRECT, EXPECTING
#35 021344 000742 BR 45 ;OVERFLOW.
#36
#37 021346 195: ;REPORT RESULT INCORRECT, EXPECTING
#38 021346 104114 205: ERROR 114 ;UNDERFLOW.
#39 021350 000740 BR 45
#40
#41 ;IF AN FP TRAP OCCURS COME HERE.
#42 021352 011602 255: MOV (SP),R2 ;GET ADDRESS OF TRAP.
#43 021354 022702 021170 CMP #25,R2 ;WAS THE TRAP DURING THE MULF INSTRUCTION?
#44 021360 001402 BEQ 265 ;BRANCH IF YES.
#45 021362 000137 036614 JMP @FFPSPUR ;OTHERWISE GO REPORT A SPURIOUS
#46 ;FP TRAP.
#47 021366 022626 265: CMP (SP)+,(SP)+ ;RESET THE STACK.
#48 021370 010237 001236 MOV R2,#STMP2 ;SAVE DATA FOR ERROR REPORT.
#49 021374 170204 STFPS R4 ;GET FPS.
#50 021376 170305 STST R5 ;GET FEC.
#51 021400 012700 000200 MOV #200,R0 ;SET FD MODE.
#52 021404 170100 LDFPS R0
#53 021406 012700 021460 MOV #OVFNIT,R0 ;GET THE RESULT.
#54 021412 174010 STD ACD,(R0)
#55 021414 010537 001254 MOV R5,#STMP11
#56 021420 020561 000024 CMP R5,24(R1) ;WAS THE FEC ANTICIPATED?
#57 021424 001004 BNE 275 ;BRANCH IF NOT ANTICIPATED.
#58
#59 021426 010102 MOV R1,R2 ;ERROR WAS ANTICIPATED SO
#60 021430 062702 000030 ADD #30,R2 ;RETURN TO THE ERROR REPORT OF THE
#61 ;CALLING ROUTINE.
#62
#63 021434 000112 JMP (R2)
#64
#65 021436 005761 000026 275: TST 26(R1) ;THE ERROR WAS NOT ANTICIPATED SO
#66 ;IT MUST BE REPORTED HERE. FIRST SEE IF EXPECTED
#67 ;OVERFLOW OR UNDER FLOW.
#68 021442 001003 BNE 295 ;BRANCH IF EXPECTING UNDERFLOW
#69
#70 ;REPORT TRAPPED ON OVERFLOW WITH FIV=0
    
```

E07

MINOEC-11-DFFPB-A PDP 11 34 FPP DIAGNOSTIC PART 2 MACY11 27(1006) 01-NOV-76 21:12 PAGE 82
 DFFPB.P11 01-NOV-76 21:05 T12 UNDER\OVER FLOW, USING MULF WITH TRAPS DISABLED, TEST

```

4171 021444 104115 28$: ERROR 115
4172 021446 000161 000036 JMP 36(R1)
4173 021450 29$:
4174 021450 104116 30$: ERROR 116 ;REPORT TRAPPED ON UNDER FLOW WITH FIU=C
4175 021454 000161 000036 JMP 36(R1)
4176 021460 000000 000000 000000 OVFNTT: .WORD 0,0,0,0
4177 021466 000000
4178 021470 IIIDONE:
4179 021470 104412 RSETJP ;GO INITIALIZE THE FPS AND STACK; AND
;SEE IF THE USER HAS EXPRESSED
;THE DESIRE TO CHANGE THE SOFTWARE
;VIRTUAL CONSOLE SWITCH REGISTER HAS
;THE USER TYPED CONTROL G?.

;*****
;TEST 13 UNDER\OVER FLOW, USING MULF WITH TRAP DISABLED, TEST
;
;THIS IS A TEST OF THE OVERFLOW AND UNDERFLOW CONDITIONS THAT CAN
;ARRISE USING THE MULF INSTRUCTION WITH TRAPS DISABLED. A SUBROUTINE IS
;USED TO SET UP THE OPERANDS, EXECUTE THE MULF INSTRUCTION AND
;CHECK THE RESULTS.
;
;*****
4180 021472 000004 TST13: SCOPE
4181
4182 ;UNDERFLOW, EXPONENT OF RESULT=-129
4183 JJJ1:
4184 021474 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
4185 021476 004737 022020 JSR PC, @#OVUNDNT
4186 021502 020200 000000 1$: .WORD 20200,0 ;AC
4187 021506 127272 000000 .WORD 127272,0
4188 021512 020000 000000 000000 2$: .WORD 20000,0,0,0 ;FSRC
4189 021520 000000
4190 021522 000000 000000 000000 3$: .WORD 0,0,0,0 ;RES
4191 021530 000000
4192 021532 000000 000000 4$: .WORD 0,0 ;ERROR RES.
4193 021536 127272 000000 .WORD 127272,0
4194 021542 000200 5$: 200 ;FPS BEFORE EXECUTION.
4195 021544 000204 .WORD 204 ;FPS AFTER EXECUTION.
4196 021546 000012 6$: 12 ;FEC
4197 021550 177777 -1 ;FLAG
4198 021552 104131 7$: ERROR 131 ;ST 331 TO 155 INTO 115 (BUT FIU)
4199 021554 000401 BR 85
4200 021556 104132 8$: ERROR 132 ;ST 115 (BUT FD)
4201 021560
4202
4203 ;UNDERFLOW, EXPONENT OF RESULT = -193
4204 JJJ2:
4205 021560 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
4206 021562 004737 022020 JSR PC, @#OVUNDNT
  
```

F07

MACROCC-11-DFFPB-R PDP 11 34 FPP DIAGNOSTIC PART 2 MACY11 27(1006) 01-NOV-76 21:12 PAGE 83
 DFFPBAL.F11 01-NOV-76 21:06 T13 UNDER/OVER FLOW, USING MULD WITH TRAP DISABLED. TEST

4527	021566	010200	000000		1\$:	.WORD	10200,0		;AC
4528	021572	123456	000000			.WORD	123456,0		
4529	021576	010000	000000	000000	2\$:	.WORD	10000,0,0,0		;FSRC
4530	021604	000000							
4531	021606	000000	000000	000000	3\$:	.WORD	0,0,0,0		;RES
4532	021614	000000							
4533	021616	000000	000000	123456	4\$:	.WORD	0,0,123456,0		;ERROR RES
4534	021624	000000							
4535	021626	005213			5\$:		5213		;FPS BEFORE EXECUTION.
4536	021630	005204					5204		;FPS AFTER EXECUTION.
4537	021632	000012			6\$:		12		;FEC
4538	021634	177777					-1		;FLAG
4539	021636	104133			7\$:	ERROR	133		;SETTING FIUV OR FIV BAD.
4540	021640	000401				BR	8\$		
4541	021642	104132				ERROR	132		;ST 115 (BUT FD)
4542	021644				8\$:				
4543									
4544									
4545	021644								;OVERFLOW, EXPONENT OF RESULT = 128
4546	021644	104413			JJJ3:	LPERR			;SET UP THE LOOP ON ERROR ADDRESS.
4547	021646	004737	022020			JSR	PC, 2#OVUNONT		
4548	021652	060200	000000		1\$:	.WORD	60200,0		;AC
4549	021656	065432	000000			.WORD	65432,0		
4550	021662	060000	000000	000000	2\$:	.WORD	60000,0,0,0		;FSRC
4551	021670	000000							
4552	021672	000000	000000	000000	3\$:	.WORD	0,0,0,0		;RES
4553	021700	000000							
4554	021702	000000	000000	065432	4\$:	.WORD	0,0,65432,0		;ERROR RES.
4555	021710	000000							
4556	021712	000200			5\$:		200		;FPS BEFORE EXECUTION.
4557	021714	000206					206		;FPS AFTER EXECUTION.
4558	021716	000010			6\$:		10		;FEC
4559	021720	000000					0		;FLAG
4560	021722	104134			7\$:	ERROR	134		;ST 333 TO 136 INTO 116 .BUT FIV
4561	021724	000401				BR	8\$		
4562	021726	104135				ERROR	135		;ST 116 (BUT FD)
4563	021730				8\$:				
4564									
4565									
4566									
4567	021730								;OVERFLOW, EXPONENT OF RESULT = 130
4568	021730	104413			JJJ4:	LPERR			;SET UP THE LOOP ON ERROR ADDRESS.
4569	021732	004737	022020			JSR	PC, 2#OVUNONT		
4570	021736	060200	000000		1\$:	.WORD	60200,0		;AC
4571	021742	125252	000000			.WORD	125252,0		
4572	021746	060200	000000	000000	2\$:	.WORD	60200,0,0,0		;FSRC
4573	021754	000000							
4574	021756	000000	000000	000000	3\$:	.WORD	0,0,0,0		;RES
4575	021764	000000							
4576	021766	000000	000000	125252	4\$:	.WORD	0,0,125252,0		;ERROR RES.
4577	021774	000000							
4578	021776	006211			5\$:		6211		;FPS BEFORE EXECUTION.
4579	022000	006206					6206		;FPS AFTER EXECUTION.
4580	022002	000010			6\$:		10		;FEC
4581	022004	000000					0		;FLAG
4582	022006	104136			7\$:	ERROR	136		;SETTING FIUV OR FIV BAD.

4583 022010 000401
4584 022012 104135
4585 022014 000137 022430

BR BS
ERROR 135 ;ST 116 (BUT FD)
JMP @#JJJDONE ;GO TO NEXT TEST.

: 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
: 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).

4626 022020 012601
4628 022022 012700 000200
4627 022026 170100
4629 022030 010100
4630 022032 172410
4631
4632 022034 010102
4633 022036 010237 001240
4634 022042 062702 000010
4635 022046 010237 001242
4636 022052 062702 000010
4637 022056 010237 001244
4638 022062 016137 000042 001252

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,@#STMP3 ;ERROR.
ADD #10,R2
MOV R2,@#STMP4
ADD #10,R2
MOV R2,@#STMP5
MOV 42(R1),@#STMP10

4639	022070	012737	022420	001246	MOV	#0VDNTT, 0#STMP6	
4640							
4641	022076	016100	000040		MOV	40(R1), R0	;LOAD THE FPS.
4642	022102	170100			LDFPS	R0	
4643	022104	012737	022126	001236	MOV	#15, 0#STMP2	
4644	022112	012737	022312	000244	MOV	#255, 0#FPVECT	;SET UP THE FP TRAP VECTOR IN CASE ;OF ERROR.
4645							;COMPUTE THE ADDRESS OF FSRC.
4646	022120	010100			MOV	R1, R0	
4647	022122	062700	000010		ADD	#10, R0	
4648							
4649	022126	171010			15: MUL0	(R0), ACO	;TEST INSTRUCTION.
4650							
4651	022130	170204			25: STFPS	R4	;GET FPS.
4652	022132	170305			STST	R5	;GET FEC.
4653	022134	012700	000200		MOV	#200, R0	;SET FD MODE.
4654	022140	170100			LDFPS	R0	
4655	022142	012700	022420		MOV	#0VDNTT, R0	;GET THE RESULT.
4656	022146	174010			STD	ACO, (R0)	
4657	022150	010437	001250		MOV	R4, 0#STMP7	
4658	022154	010537	001254		MOV	R5, 0#STMP11	
4659							
4660	022160	012700	022420		MOV	#0VDNTT, R0	;CHECK THE RESULT.
4661	022164	010102			MOV	R1, R2	
4662	022166	062702	000020		ADD	#20, R2	
4663	022172	012703	000004		MOV	#4, R3	
4664	022176	022022			35: CMP	(R0)+, (R2)+	
4665	022200	001015			BNE	155	;BRANCH IF INCORRECT.
4666	022202	077303			SOB	R3, 35	
4667							
4668	022204	026104	000042		CMP	42(R1), R4	;WAS FPS CORRECT?
4669	022210	001002			BNE	105	;BRANCH IF FPS IS INCORRECT.
4670							
4671	022212	000161	000056		45: JMP	56(R1)	;RETURN, TEST COMPLETED.
4672							
4673							;REPORT INCORRECT FPS.
4674	022216	005761	000046		105: TST	46(R1)	;WAS THE RESULT OVER OR UNDER FLOW?
4675	022222	001002			BNE	125	;BRANCH IF UNDERFLOW.
4676							
4677							;REPORT FPS BAD AFTER OVERFLOW.
4678	022224	104123			115: ERROR	123	
4679	022226	000771			BR	45	
4680							
4681	022230				125: ERROR	124	;REPORT FPS BAD AFTER UNDERFLOW.
4682	022230	104124			135: BR	45	
4683	022232	000767					
4684							
4685							;RESULT INCORRECT.
4686	022234	012700	022420		155: MOV	#0VDNTT, R0	;SEE IF FAILURE IS ANTICIPATED
4687	022240	010102			MOV	R1, R2	;FAILURE.
4688	022242	062702	000030		ADD	#30, R2	
4689	022246	012703	000004		MOV	#4, R3	
4690	022252	022022			165: CMP	(R0)+, (R2)+	
4691	022254	001007			BNE	175	;BRANCH IF NOT ANTICIPATED.
4692	022256	077303			SOB	R3, 165	
4693							
4694	022260	010102			MOV	R1, R2	;ERROR WAS ANTICIPATED SO RETURN

4695	022262	062702	000054		ADD	#54,R2		;TO THE ERROR REPORT IN THE CALLING
4696	022266	010237	001236		MOV	R2,#\$TMP2		;ROUTINE.
4697	022272	000112			JMP	(R2)		
4698								
4699	022274	005761	000046	17\$:	TST	46(R1)		;RESULT WAS NOT ANTICIPATED
4700								;SO ERROR MUST BE REPORTED HERE.
4701								;FIRST SEE IF ARGUMENTS SHOULD
4702								;HAVE RESULTED IN OVERFLOW OR UNDER
4703								;FLOW BY LOOKING AT THE FLAG.
4704	022300	001002			BNE	19\$;BRANCH IF UNDERFLOW EXPECTED.
4705								
4706								;REPORT RESULT INCORRECT, EXPECTING
4707	022302	104125		18\$:	ERROR	125		;OVERFLOW.
4708	022304	000742			BR	4\$		
4709								
4710	022306			19\$:				;REPORT RESULT INCORRECT, EXPECTING
4711	022306	104126		20\$:	ERROR	126		;UNDERFLOW.
4712	022310	000740			BR	4\$		
4713								
4714								;IF AN FP TRAP OCCURS COME HERE.
4715	022312	011602		25\$:	MOV	(SP),R2		;GET ADDRESS OF TRAP.
4716	022314	022702	022130		CMP	#2\$,R2		;WAS THE TRAP DURING THE MULF INSTRUCTION?
4717	022320	001402			BEQ	26\$;BRANCH IF YES.
4718	022322	000137	036614		JMP	#\$FSPUR		;OTHERWISE GO REPORT A SPURIOUS
4719								;FP TRAP.
4720	022326	022626		26\$:	CMP	(SP)+,(SP)+		;RESET THE STACK.
4721	022330	010237	001236		MOV	R2,#\$TMP2		;SAVE DATA FOR ERROR REPORT.
4722	022334	170204			STFPS	R4		;GET FPS.
4723	022336	170305			STST	R5		;GET FEC.
4724	022340	012700	000200		MOV	#200,R0		;SET FD MODE.
4725	022344	170100			LDFPS	R0		
4726	022346	012700	022420		MOV	#OVDNTT,R0		;GET THE RESULT.
4727	022352	174010			STD	AC0,(R0)		
4728	022354	010537	001254		MOV	R5,#\$TMP11		
4729	022360	020561	000044		CMP	R5,44(R1)		;WAS THE FEC ANTICIPATED?
4730	022364	001004			BNE	27\$;BRANCH IF NOT ANTICIPATED.
4731								
4732	022366	010102			MOV	R1,R2		;ERROR WAS ANTICIPATED SO
4733	022370	062702	000050		ADD	#50,R2		;RETURN TO THE ERROR REPORT OF THE
4734								;CALLING ROUTINE.
4735	022374	000112			JMP	(R2)		
4736								
4737	022376	005761	000026	27\$:	TST	26(R1)		;THE ERROR WAS NOT ANTICIPATED SO
4738								;IT MUST BE REPORTED HERE. FIRST SEE IF EXPECTED
4739								;OVERFLOW OR UNDER FLOW.
4740	022402	001003			BNE	29\$;BRANCH IF EXPECTING UNDERFLOW
4741								
4742								;REPORT TRAPPED ON OVERFLOW WITH FIV=0
4743	022404	104127		28\$:	ERROR	127		
4744	022406	000161	000056		JMP	56(R1)		
4745								
4746	022412			29\$:				;REPORT TRAPPED ON UNDER FLOW WITH FIU=0
4747	022412	104130		30\$:	ERROR	130		
4748	022414	000161	000056		JMP	56(R1)		
4749								
4750	022420	000000	000000	000000	OVDNTT: .WORD	0,0,0,0		

4751 022426 000000

4752

4753 022430

4754 022430 104412

4755

4756

4757

4758

4759

4760

4761

4762

4763

4764

4765

4766

4767

4768

4769

4770

4771

4772

4773

4774

4775 022432 000004

4776

4777

4778 022434

4779 022434 104413

4780 022436 004737 022660

4781 022442 020123 045676

4782 022446 020200 000000

4783 022452 000123 045676

4784 022456 177777 177777

4785 022462 002000

4786 022464 102004

4787 022466 000012

4788 022470 177777

4789 022472 104145

4790 022474 000401

4791 022476 104144

4792 022500

4793

4794

4795 022500

4796 022500 104413

4797 022502 004737 022660

4798 022506 010127 127272

4799 022512 010200 000000

4800 022516 060127 127272

4801 022522 177777 177777

4802 022526 007017

4803 022530 107000

4804 022532 000012

4805 022534 177777

4806 022536 104146

JJJDONE:

RSETUP

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

: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.
:*

TST14: SCOPE

:UNDERFLOW, EXPONENT OF RESULT = -129

KFK1:

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 FIJ) NO TRAP.
BR 8\$
8\$: ERROR 144

: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 ;FLAG
7\$: ERROR 146 ;ST 137 (BUT FIJ) NO TRAP.

K07

```

4807 022540 000401          BR      B5
4808 022542 104144          ERROR   144
4809 022544                B5:
4810
4811 ;OVERFLOW, EXPONENT OF THE RESULT = 128
4812 KKK4:
4813 022544 104413          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
4814 022546 004737 022660        JSR          PC, @OVUNFT
4815 022552 060252 125252        .WORD       60252,125252      ;AC
4816 022556 060000 000000        .WORD       60000,0          ;FSRC
4817 022562 000052 125252        .WORD       000052,125252   ;RES
4818 022566 177777 177777        .WORD       -1,-1          ;ERROR RES.
4819 022572 001000                .WORD       1000           ;FPS BEFORE EXECUTION.
4820 022574 101006                .WORD       101006        ;FPS AFTER EXECUTION.
4821 022576 000010                .WORD       10            ;FEC
4822 022600 000000                .WORD       0             ;FLAG
4823 022602 104147                ERROR      147            ;ST 333 (BUT FIV) NO TRAP
4824 022604 000401          BR      B5
4825 022606 104143          ERROR     143
4826 022610                B5:

```

```

4827
4828 ;OVERFLOW, EXPONENT OF RESULT = 130
4829 KKK5:
4830 022610 104413          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
4831 022612 004737 022660        JSR          PC, @OVUNFT
4832 022616 060345 067654        .WORD       60345,67654     ;AC
4833 022622 060200 000000        .WORD       60200,0        ;FSRC
4834 022626 000345 067654        .WORD       345,67654     ;RES
4835 022632 177777 177777        .WORD       -1,-1        ;ERROR RES.
4836 022636 007015                .WORD       7015          ;FPS BEFORE EXECUTION.
4837 022640 107002                .WORD       107002       ;FPS AFTER EXECUTION.
4838 022642 000010                .WORD       10            ;FEC
4839 022644 000000                .WORD       0             ;FLAG
4840 022646 104150                ERROR      150            ;ST 133 (BUT FIV) NO TRAP
4841 022650 000401          BR      B5
4842 022652 104143          ERROR     143
4843 022654 000167 000412        JMP         KKKDONE
4844

```

4845 ;THIS SUBROUTINE, OVUNFT, IS USED TO SET UP THE OPERANDS, EXECUTE
 4846 ;THE MULF INSTRUCTION AND CHECK THE RESULTS OF AN INSTRUCTION WITH
 4847 ;OPERANDS WHICH SHOULD RESULT IN EITHER OVERFLOW OR UNDERFLOW. A CALL
 4848 ;TO IT IS MADE THUS:

```

4849 :
4850 :          ACARG: .WORD X,X          ;AC OPERAND
4851 :          FSRCARG: .WORD X,X       ;FSRC OPERAND
4852 :          RES: .WORD X,X           ;EXPECTED RESULT
4853 :          ERRRES: .WORD X,X        ;ERROR RESULT
4854 :          FPSB: .WORD X            ;FPS BEFORE EXECUTION
4855 :          FPSA: .WORD X            ;FPS AFTER EXECUTION
4856 :          FEC: .WORD X             ;EXPECTED FEC
4857 :          FLAG: .WORD X            ;0/-1, OVER/UNDER FLOW FLAG
4858 :          ERR1: .ERROR X           ;TRAP ERROR.
4859 :          BR      CONT
4860 :          ERR2: .ERROR X
4861 :          CONT:

```



```

4863 ;THE OPERANDS ARE SET UP (USING ACO AS THE ACCUMULATOR). THEN
4864 ;THE MULF INSTRUCTION IS EXECUTED. IF THE TRAP OCCURS THEN THE
4865 ;RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
4866 ;COMPARED WITH FPSP IF THIS TOO IS CORRECT OVUNFT RETURNS CONTROL
4867 ;TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD OVUNFT
4868 ;REPORTS THIS FAILURE AND THEN RETURNS TO CONT. THE FEC IS TREATED
4869 ;IN THE SAME WAY. IF THE RESULT OF THE
4870 ;MULF IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
4871 ;ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
4872 ;THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN OVUNFT
4873 ;WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR2 OTHERWISE THE
4874 ;RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND OVUNFT WILL
4875 ;REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.
4876 ;IF NO TRAP OCCURS CONTROL IS PASSED TO ERR1.
4877 ;NOTE THAT OVUNFT USES THE FLAG
4878 ;TO TELL WHETHER OR NOT THESE PARTICULAR OPERANDS WILL RESULT IN
4879 ;UNDERFLOW (FLAG=-1) OR OVERFLOW (FLAG=0).
4880
4881 022660 012601 000200 022662 012700 000200 022666 170100
4882
4883
4884
4885 022670 010100 022672 172410
4886
4887
4888 022674 010102 022676 010237 001240 022702 062702 000004
4889 022706 010237 001242 022712 062702 000004
4890 022716 010237 001244 022722 016137 000022 001252
4891 022730 012737 023262 001246
4892
4893 022736 016100 000020 022742 170100
4894 022744 012737 022766 001236 022752 012737 022776 000244
4895
4896
4897 022760 010100 022762 062700 000004
4898
4899
4900
4901
4902 022766 171010 022770 170000
4903
4904
4905 022772 000137 023222
4906
4907
4908
4909
4910 022776 011602 023000 020227 022770 023004 001402
4911 023006 000137 036614
4912
4913
4914
4915 023012 022626 023014 170204 023016 170305
4916 023020 012700 000200
4917
4918

```

```

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

```

```

OVUNFT: MOV (SP)+,R1 ;GET A POINTER TO THE ARGUMENTS.
MOV #200,R0 ;SET FD MODE.
LOFPS R0
MOV R1,R0 ;LOAD ACO, OPERAND.
LDD (R0),ACO
MOV R1,R2 ;SAVE THE DATA PATTERNS IN CASE OF
MOV R2,#STMP3 ;ERROR.
ADD #4,R2
MOV R2,#STMP4
ADD #4,R2
MOV R2,#STMP5
MOV 22(R1),#STMP10
MOV #OVFTT,#STMP6
MOV 20(R1),R0 ;LOAD THE FPS.
LOFPS R0
MOV #15,#STMP2
MOV #505,#FPVECT ;SET UP THE FP TRAP VECTOR IN CASE
;OF ERROR.
MOV R1,R0 ;COMPUTE THE ADDRESS OF FSRC.
ADD #4,R0
1$: MULF (R0),ACO ;TEST INSTRUCTION. SHOULD CAUSE TRAP.
2$: CFCC
JMP #255 ;FAILURE, NO TRAP.
50$: MOV (SP),R2 ;TRAP TO HERE AND SEE IF THE PC OF THE
CMP R2,#25 ;TRAP WAS THAT OF THE MULF INSTRUCTION.
BEQ 51$ ;BRANCH IF YES.
JMP #FPSPUR ;OTHERWISE REPORT SPURIOUS FP ERROR.
51$: CMP (SP)+,(SP)+ ;RESET THE STACK
STFPS R4 ;GET FPS.
STST R5 ;GET FEC.
MOV #200,R0 ;SET FD MODE.

```

M07

MAINDEC-11-DFFPB-A PDP 11/34 FPP DIAGNOSTIC PART 2 MACY11 27(1006) 01-NOV-76 21:12 PAGE 90
 DFFPB.A.P11 01-NOV-76 21:06 T14 UNDER/OVER FLOW, USING MULF WITH TRAPS ENABLED, TEST

4919	023024	170100		LDFPS	RO		
4920	023026	012700	023262	MOV	#OVFTT,RO		;GET THE RESULT.
4921	023032	174010		STD	ACD,(RO)		
4922	023034	010437	001250	MOV	R4,0#STMP7		
4923	023040	010537	001254	MOV	R5,0#STMP11		
4924							
4925	023044	012700	023262	MOV	#OVFTT,RO		;CHECK THE RESULT.
4926	023050	010102		MOV	R1,R2		
4927	023052	062702	000010	ADD	#10,R2		
4928	023056	012703	000002	MOV	#2,R3		
4929	023062	022022		3\$: CMP	(RO)+,(R2)+		
4930	023064	001027		BNE	15\$;BRANCH IF INCORRECT.
4931	023066	077303		SQB	R3,3\$		
4932							
4933	023070	026104	000022	CMP	22(R1),R4		;WAS FPS CORRECT?
4934	023074	001014		BNE	10\$;BRANCH IF FPS IS INCORRECT.
4935							
4936	023076	026105	000024	CMP	24(R1),R5		;IS FEC CORRECT?
4937	023102	001002		BNE	5\$;IF INCORRECT BRANCH.
4938	023104	000161	000036	4\$: JMP	36(R1)		;RETURN, TEST COMPLETED.
4939							
4940				:REPORT	INCORRECT FEC.		
4941	023110	005761	000026	5\$: TST	26(R1)		;WAS THE RESULT OVERFLOW OR UNDERFLOW?
4942	023114	001002		BNE	7\$;BRANCH IF UNDERFLOW.
4943							
4944							;REPORT BAD FEC ON EXPECTED OVERFLOW.
4945	023116	104137		6\$: ERROR	137		
4946	023120	000771		BR	4\$		
4947							
4948	023122			7\$: ERROR	140		;REPORT BAD FEC ON EXPECTED UNDERFLOW.
4949	023122	104140		8\$: BR	4\$		
4950	023124	000767					
4951							
4952				:REPORT	INCORRECT FPS.		
4953	023126	005761	000026	10\$: TST	26(R1)		;WAS THE RESULT OVER OR UNDER FLOW?
4954	023132	001002		BNE	12\$;BRANCH IF UNDERFLOW.
4955							;REPORT FPS BAD AFTER OVERFLOW.
4956							
4957	023134	104141		11\$: ERROR	141		
4958	023136	000762		BR	4\$		
4959							
4960	023140			12\$: ERROR	142		;REPORT FPS BAD AFTER UNDERFLOW.
4961	023140	104142		13\$: BR	4\$		
4962	023142	000760					
4963							
4964				:RESULT	INCORRECT.		
4965	023144	012700	023262	15\$: MOV	#OVFTT,RO		;SEE IF FAILURE IS ANTICIPATED
4966	023150	010102		MOV	R1,R2		;FAILURE.
4967	023152	062702	000014	ADD	#14,R2		
4968	023156	012703	000002	MOV	#2,R3		
4969	023162	022022		16\$: CMP	(RO)+,(R2)+		
4970	023164	001007		BNE	17\$;BRANCH IF NOT ANTICIPATED.
4971	023166	077303		SQB	R3,16\$		
4972							
4973	023170	010102		MOV	R1,R2		;ERROR WAS ANTICIPATED SO RETURN
4974	023172	062702	000034	ADD	#34,R2		;TO THE ERROR REPORT IN THE CALLING

```

4975 023176 010237 001236          MOV  R2,2#STMP2      ;ROUTINE.
4976 023202 000112                JMP  (R2)
4977
4978 023204 005761 000026          17$: TST  26(R1)      ;RESULT WAS NOT ANTICIPATED
4979                                     ;SO ERROR MUST BE REPORTED HERE.
4980                                     ;FIRST SEE IF ARGUMENTS SHOULD
4981                                     ;HAVE RESULTED IN OVERFLOW OR UNDER
4982                                     ;FLOW BY LOOKING AT THE FLAG.
4983 023210 001002                BNE  19$             ;BRANCH IF UNDERFLOW EXPECTED.
4984
4985                                     ;REPORT RESULT INCORRECT, EXPECTING
4986 023212 104143          18$: ERROR  143      ;OVERFLOW.
4987 023214 000733                BR   4$
4988
4989                                     ;REPORT RESULT INCORRECT, EXPECTING
4990 023216 104144          20$: ERROR  144      ;UNDERFLOW.
4991 023220 000731                BR   4$
4992
4993                                     ;IF NO FP TRAP OCCURS COME HERE.
4994 023222 170204          25$: STFPS  R4          ;GET FPS.
4995 023224 170305                STST  R5          ;GET FEC.
4996 023226 012700 000200          MOV  #200,R0       ;SET FD MODE.
4997 023232 170100                LDFPS R0
4998 023234 012700 023262          MOV  #OVFTT,R0     ;GET THE RESULT.
4999 023240 174010                STD  ACO,(R0)
5000 023242 010437 001250          MOV  R4,2#STMP7
5001 023246 010537 001254          MOV  R5,2#STMP11
5002 023252 010102                MOV  R1,R2
5003 023254 062702 000030          ADD  #30,R2        ;ERROR WAS ANTICIPATED SO
5004                                     ;RETURN TO THE ERROR REPORT OF THE
5005                                     ;CALLING ROUTINE.
5006 023260 000112                JMP  (R2)
5007 023262 000000 000000 000000  OVFTT: .WORD  0,0,0,0
5008 023270 000000
5009
5010                                     KKKDONE:
5011 023272 104412                RSETUP             ;GO INITIALIZE THE FPS AND STACK; AND
5012                                     ;SEE IF THE USER HAS EXPRESSED
5013                                     ;THE DESIRE TO CHANGE THE SOFTWARE
5014                                     ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
5015                                     ;THE USER TYPED CONTROL G?).
5016
5017
5018
5019
5020
5021
5022
5023
5024
5025
5026
5027
5028 023274 000004          *ST15: SCOPE
5029
5030

```

;UNDERFLOW, EXPONENT OF RESULT = -129

```

5031 023276          LLL1:
5032 023276 104413      LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
5033 023300 004737 023622 JSR          PC, #OVUNDT
5034 023304 020052 125252 1S: .WORD      20052,125252 ;AC
5035 023310 125252 125252 .WORD      125252,125252 ;FSRC
5036 023314 020300 000000 2S: .WORD      20300,0,0,0
5037 023322 000000 .WORD
5038 023324 000177 177777 3S: .WORD      177,-1,-1,-1 ;RES
5039 023332 177777 4S: .WORD
5040 023334 000177 177777 .WORD      177,-1 ;ERROR RES.
5041 023340 125252 125252 .WORD      125252,125252
5042 023344 002200 5S: 2200 ;FPS BEFORE EXECUTION.
5043 023346 102204 .WORD      102204 ;FPS AFTER EXECUTION.
5044 023350 000012 6S: 12 ;FEC
5045 023352 177777 -1 ;FLAG
5046 023354 104157 7S: ERROR 157 ;ST 331 (BUT FIU) NO TRAP.
5047 023356 000401 BR 8S ;
5048 023360 104160 ERROR 160 ;ST 155 (BUT FD)
5049 023362
5050
5051 ;UNDERFLOW, EXPONENT OF THE RESULT = -193
5052 023362          LLL2:
5053 023362 104413      LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
5054 023364 004737 023622 JSR          PC, #OVUNDT
5055 023370 010327 127272 1S: .WORD      10327,127272 ;AC
5056 023374 036363 045454 .WORD      36363,45454 ;FSRC
5057 023400 010000 000000 2S: .WORD      10000,0,0,0
5058 023406 000000 .WORD
5059 023410 060127 127272 3S: .WORD      60127,127272 ;RES
5060 023414 036363 045454 .WORD      36363,45454
5061 023420 177777 177777 4S: .WORD      -1,-1,-1,-1 ;ERROR RES.
5062 023426 177777 5S:
5063 023430 007217 7217 ;FPS BEFORE EXECUTION.
5064 023432 107200 107200 ;FPS AFTER EXECUTION.
5065 023434 000012 6S: 12 ;FEC
5066 023436 177777 -1 ;FLAG
5067 023440 104161 7S: ERROR 161 ;ST 137 (BUT FIU) NO TRAP.
5068 023442 000401 BR 8S ;
5069 023444 104156 ERROR 156
5070 023446
5071
5072 ;OVERFLOW, EXPONENT OF THE RESULT = 128
5073 023446          LLL3:
5074 023446 104413      LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
5075 023450 004737 023622 JSR          PC, #OVUNDT
5076 023454 060252 125252 1S: .WORD      60252,125252 ;AC
5077 023460 125252 125252 .WORD      125252,125252 ;FSRC
5078 023464 160100 000000 2S: .WORD      160100,0,0,0 ;FSRC
5079 023472 000000 .WORD
5080 023474 100177 177777 3S: .WORD      100177,-1,-1,-1 ;RES
5081 023502 177777 4S:
5082 023504 100177 177777 .WORD      100177,-1 ;ERROR RES.
5083 023510 125252 125252 .WORD      125252,125252
5084 023514 001200 5S: 1200 ;FPS BEFORE EXECUTION.
5085 023516 101216 .WORD      101216 ;FPS AFTER EXECUTION.
5086 023520 000010 6S: 10 ;FEC
    
```

5087 023522 000000
 5088 023524 104162
 5089 023526 000401
 5090 023530 104163
 5091 023532
 5092
 5093
 5094 023532
 5095 023532 104410
 5096 023534 004737 023522
 5097 023540 060345 067654
 5098 023544 056765 045676
 5099 023550 060200 000100 000000
 5100 023556 000000
 5101 023560 000345 067654
 5102 023564 056765 045676
 5103 023570 177777 177777 177777
 5104 023576 177777
 5105 023600 007215
 5106 023602 107202
 5107 023604 000010
 5108 023606 000000
 5109 023610 104164
 5110 023612 000401
 5111 023614 104155
 5112 023616 000137 024234
 5113
 5114
 5115
 5116
 5117
 5118
 5119
 5120
 5121
 5122
 5123
 5124
 5125
 5126
 5127
 5128
 5129
 5130
 5131
 5132
 5133
 5134
 5135
 5136
 5137
 5138
 5139
 5140
 5141
 5142

```

0                                     :FLAG
7S:  ERROR 162                       ;ST 333 (BUT FIV) NO TRAP.
    BR 85
    ERROR 163                       ;ST 700 (BUT FD).
9S:
:OVERFLOW, EXPONENT OF THE RESULT = 130
LLL4:
    LPERR                               ;SET UP THE LOOP ON ERROR ADDRESS.
    JSR PC,OVUNDT
1S:  .WORD 60345,67654                 ;AC
    .WORD 56765,45676
2S:  .WORD 60200,0,0,0                 ;FSRC
    .WORD 345,67654
3S:  .WORD 56765,45676                 ;RES
    .WORD -1,-1,-1,-1
4S:  .WORD -1,-1,-1,-1                 ;ERROR RES.
5S:  7215                               ;FPS BEFORE EXECUTION.
    107202                              ;FPS AFTER EXECUTION.
6S:  10                                     ;FEC
    0                                     ;FLAG
7S:  ERROR 164                       ;ST 133 (BUT FIV) NO TRAP
    BR 85
    ERROR 155
8S:  JMP J#LLLDONE
  
```

: THIS SUBROUTINE, OVUNDT, 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 ACC AS THE ACCUMULATOR). THEN
 : THE MULD INSTRUCTION IS EXECUTED. IF THE TRAP OCCURS THEN THE
 : RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
 : COMPARED WITH FPSA IF THIS TOO IS CORRECT OVUNDT RETURNS CONTROL
 : TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD OVUNDT
 : REPORTS THIS FAILURE AND THEN RETURNS TO CONT. THE FEC IS TREATED
 : IN THE SAME WAY IF THE RESULT OF THE
 : MULF IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
 : ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
 : THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN OVUNDT
 : WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR2. OTHERWISE THE

: RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND OVUNDT WILL
 : REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT
 : IF NO TRAP OCCURS CONTROL IS PASSED TO ERR1.
 : NOTE THAT OVUNDT USES THE FLAG
 : TO TELL WHETHER OR NOT THESE PARTICULAR OPERANDS WILL RESULT IN
 : UNDERFLOW (FLAG=-1) OR OVERFLOW (FLAG=0).

S143
S144
S145
S146
S147
S148
S149
S150
S151
S152
S153
S154
S155
S156
S157
S158
S159
S160
S161
S162
S163
S164
S165
S166
S167
S168
S169
S170
S171
S172
S173
S174
S175
S176
S177
S178
S179
S180
S181
S182
S183
S184
S185
S186
S187
S188
S189
S190
S191
S192
S193
S194
S195
S196
S197
S198

023622	012601		
023624	012700	000200	
023630	170100		
023632	010100		
023634	172410		
023636	010102		
023640	010237	001240	
023644	062702	000010	
023650	010237	001242	
023654	062702	000010	
023660	010237	001244	
023664	016137	000042	001252
023672	012737	024224	001246
023700	016100	000040	
023704	170100		
023706	012737	023730	001236
023714	012737	023740	000244
023722	010100		
023724	062700	000010	
023730	171010		
023732	170000		
023734	000137	024164	
023740	011602		
023742	020227	023732	
023746	001402		
023750	000137	036614	
023754	022626		
023756	170204		
023760	170305		
023762	012700	000200	
023766	170100		
023770	012700	024224	
023774	174010		
023776	010437	001250	
024002	010537	001254	
024006	012700	024224	
024012	010102		
024014	062702	000020	
024020	012703	000004	
024024	022022		

```

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

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

          MOV R1,R2 ;SAVE THE DATA PATTERNS IN CASE OF
          MOV R2,@STMP3 ;ERROR.
          ADD #10,R2
          MOV R2,@STMP4
          ADD #10,R2
          MOV R2,@STMP5
          MOV 42(R1),@STMP10
          MOV #OVDTT,@STMP6

          MOV 40(R1),R0 ;LOAD THE FPS.
          LDFPS R0
          MOV #15,@STMP2
          MOV #50,@FPVECT ;SET UP THE FP TRAP VECTOR IN CASE
                           ;OF ERROR.
          MOV R1,R0 ;COMPUTE THE ADDRESS OF FSRC.
          ADD #10,R0

15: MULD (R0),ACO ;TEST INSTRUCTION. SHOULD CAUSE TRAP.
25: CFCC

          JMP @255 ;FAILURE, NO TRAP.

505: MOV (SP),R2 ;TRAP TO HERE AND SEE IF THE PC OF THE
          CMP R2,#25 ;TRAP WAS THAT OF THE MULD INSTRUCTION.
          BEQ 515 ;BRANCH IF YES.
          JMP @FPSPUR ;OTHERWISE REPORT SPURIOUS FP ERROR.

515: CMP (SP)+,(SP)+ ;RESET THE STACK
          STFPS R4 ;GET FPS.
          STST R5 ;GET FEC.
          MOV #200,R0 ;SET FD MODE.
          LDFPS R0
          MOV #OVDTT,R0 ;GET THE RESULT.
          STD ACO,(R0)
          MOV R4,@STMP7
          MOV R5,@STMP11

          MOV #OVDTT,R0 ;CHECK THE RESULT.
          MOV R1,R2
          ADD #20,R2
          MOV #4,R3
35: CMP (R0)+,(R2)+
    
```

5199	024026	001027		BNE	155		: BRANCH IF INCORRECT.
5200	024030	077303		SOB	R3,35		
5201							
5202	024032	026104	000042	CMP	42(R1),R4		: WAS FPS CORRECT?
5203	024036	001014		BNE	105		: BRANCH IF FPS IS INCORRECT.
5204							
5205	024040	026105	000044	CMP	44(R1),R5		: IS FEC CORRECT?
5206	024044	001002		BNE	55		: IF INCORRECT BRANCH.
5207	024046	000161	000056	45: JMP	56(R1)		: RETURN, TEST COMPLETED.
5208							
5209							: REPORT INCORRECT FEC.
5210	024052	005761	000046	55: TST	46(R1)		: WAS THE RESULT OVERFLOW OR UNDERFLOW?
5211	024056	001002		BNE	75		: BRANCH IF UNDERFLOW.
5212							
5213							: REPORT BAD FEC ON EXPECTED OVERFLOW.
5214	024060	104151		65: ERROR	151		
5215	024062	000771		BR	45		
5216							
5217	024064			75: ERROR	152		: REPORT BAD FEC ON EXPECTED UNDERFLOW.
5218	024064	104152		85: ERROR	152		
5219	024066	000767		BR	45		
5220							
5221							: REPORT INCORRECT FPS.
5222	024070	005761	000046	105: TST	46(R1)		: WAS THE RESULT OVER OR UNDER FLOW?
5223	024074	001002		BNE	125		: BRANCH IF UNDERFLOW.
5224							
5225							: REPORT FPS BAD AFTER OVERFLOW.
5226	024076	104153		115: ERROR	153		
5227	024100	000762		BR	45		
5228							
5229	024102			125: ERROR	154		: REPORT FPS BAD AFTER UNDERFLOW.
5230	024102	104154		135: ERROR	154		
5231	024104	000760		BR	45		
5232							
5233							: RESULT INCORRECT.
5234	024106	012700	024224	155: MOV	#0VDTT,RC		: SEE IF FAILURE IS ANTICIPATED
5235	024112	010102		MOV	R1,R2		: FAILURE.
5236	024114	062702	000030	ADD	#30,R2		
5237	024120	012703	000004	MOV	#4,R3		
5238	024124	022022		165: CMP	(R0)+,(R2)+		: BRANCH IF NOT ANTICIPATED.
5239	024126	001007		BNE	175		
5240	024130	077303		SOB	R3,165		
5241							
5242	024132	010102		MOV	R1,R2		: ERROR WAS ANTICIPATED SO RETURN
5243	024134	062702	000054	ADD	#54,R2		: TO THE ERROR REPORT IN THE CALLING
5244	024140	010237	001236	MOV	R2,#\$TMP2		: ROUTINE.
5245	024144	000112		JMP	(R2)		
5246							
5247	024146	005761	000046	175: TST	46(R1)		: RESULT WAS NOT ANTICIPATED
5248							: SO ERROR MUST BE REPORTED HERE.
5249							: FIRST SEE IF ARGUMENTS SHOULD
5250							: HAVE RESULTED IN OVERFLOW OR UNDER
5251							: FLOW BY LOOKING AT THE FLAG.
5252	024152	001002		BNE	195		: BRANCH IF UNDERFLOW EXPECTED.
5253							
5254							: REPORT RESULT INCORRECT, EXPECTING

F08

MAINDEC-11-OFFPB-A PDP 11 34 FPP DIAGNOSTIC PART 2 MACY11 27(1006) 01-NOV-76 21:12 PAGE 96
 OFFPB9.P11 01-NOV-76 21:06 T15 UNDER/OVER FLOW, USING MUL0 WITH TRAPS ENABLED. TEST

```

5255 024154 104155 185: ERROR 155 :OVERFLOW.
5256 024156 000733 BR 45
5257
5258 024160 195: :REPORT RESULT INCORRECT, EXPECTING
5259 024160 104156 205: ERROR 156 :UNDERFLOW.
5260 024162 000731 BR 45
5261
5262 :IF NO FP TRAP OCCURS COME HERE.
5263 024164 170204 255: STFPS R4 :GET FPS.
5264 024166 170305 STST R5 :GET FEC.
5265 024170 012700 000200 MOV #200,R0 :SET FD MODE.
5266 024174 170100 LDFPS R0
5267 024176 012700 024224 MOV #OVDTT,R0 :GET THE RESULT.
5268 024202 174010 STD ACO,(R0)
5269 024204 010437 001250 MOV R4,#STMP7
5270 024210 010537 001254 MOV R5,#STMP11
5271 024214 010102 MOV R1,R2 :ERROR WAS ANTICIPATED SO
5272 024216 062702 000050 ADD #50,R2 :RETURN TO THE ERROR REPORT OF THE
5273 :CALLING ROUTINE.
5274 024222 000112 JMP (R2)
5275
5276 024224 003000 000000 000000 OVDTT: .WORD 0,0,0,0
5277 024232 000000
5278
5279 024234 LLDONE:
5280 024234 104412 RSETUP :GO INITIALIZE THE FPS AND STACK; AND
5281 :SEE IF THE USER HAS EXPRESSED
5282 :THE DESIRE TO CHANGE THE SOFTWARE
5283 :VIRTUAL CONSOLE SWITCH REGISTER (HAS
5284 :THE USER TYPED CONTROL G?).
5285
5286
5287
5288
5289
5290
5291 :*****
5292 :*TEST 16 MODF TEST
5293 :*
5294 :*THIS IS A TEST OF THE MODF INSTRUCTION, WHICH MAKES USE OF
5295 :*A SUBROUTINE TO SET UP THE OPERANDS, EXECUTE THE MODF INSTRUCTION
5296 :*AND CHECK THE RESULTS.
5297 :*
5298 :*****
5299 †ST16: SCOPE
5300 :MODF WITH (FSRC=AC=0)
5301 GGG1:
5302 024240 104413 LPERR :SET UP THE LOOP ON ERPOP ADDRESS.
5303 024242 004737 025324 JSR PC,#MODFSUB
5304 024246 000000 000000 15: .WORD 0,0 :AC
5305 024252 000000 000000 25: .WORD 0,0 :FSRC
5306 024256 000000 000000 35: .WORD 0,0 :FRACTIONAL RES.
5307 024262 000000 000000 45: .WORD 0,0 :INTEGER RES.
5308 024266 177777 177777 55: .WORD -1,-1 :ERROR FRACTIONAL RES.
5309 024272 177777 177777 65: .WORD -1,-1 :ERROR INGETER RES.
5310 024276 000013 75: 13 :FPS BEFORE EXECUTION.
  
```



```

5311 024300 000004          4          :FPS AFTER EXECUTION.
5312 024302 104056      8S:  ERROR    56      :STORE SINGLE ZERO BAD.
5313 024304 000401          BR     95
5314 024306 104057          ERROR    57      :AC V 1 (= ZERO FAILED.
5315 024310
5316
5317
:MODF TEST, WITH (FSRC=0)
5318 024310      GGG2:
5319 024310 104413          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
5320 024312 004737 025324      JSR      PC,2#MODFSUB
5321 024316 123456 076543      1S:  .WORD 123456,76543      :AC
5322 024322 000000 000000      2S:  .WORD 0,0              :FSRC
5323 024326 000000 000000      3S:  .WORD 0,0              :FRACTIONAL RES.
5324 024332 000000 000000      4S:  .WORD 0,0              :INTEGER RESULT.
5325 024336 123456 076543      5S:  .WORD 123456,76543     :ERROR FRACTIONAL RES.
5326 024342 177777 177777      6S:  .WORD -1,-1          :ERROR INTEGER RES.
5327 024346 000000          7S:  0                      :FPS BEFORE EXECUTION.
5328 024350 000004          4          :FPS AFTER EXECUTION.
5329 024352 104056      8S:  ERROR    56      :STORE ZERO FAILURE.
5330 024354 000401          BR     95
5331 024356 104057          ERROR    57
5332 024360
5333
5334
:MODF TEST WITH (AC=0)
5335 024360      GGG3:
5336 024360 104413          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
5337 024362 004737 025324      JSR      PC,2#MODFSUB
5338 024366 000000 000000      1S:  .WORD 0,0              :AC
5339 024372 076543 021234      2S:  .WORD 76543,21234     :FSRC
5340 024376 000000 000000      3S:  .WORD 0,0              :FRACTIONAL RES.
5341 024402 000000 000000      4S:  .WORD 0,0              :INTEGER RES.
5342 024406 000000 000000      5S:  .WORD 0,0              :ERROR FRACTIONAL RES.
5343 024412 177777 177777      6S:  .WORD -1,-1          :ERROR INTEGER RES.
5344 024416 000003          7S:  3                      :FPS BEFORE EXECUTION.
5345 024420 000004          4          :FPS AFTER EXECUTION.
5346 024422 104053      8S:  ERROR    53      :RES.BAD
5347 024424 000401          BR     95
5348 024426 104057          ERROR    57
5349 024430
5350
5351
:MODF TEST WITH EXPONENT OF THE RESULT = 25
5352 024430      GGG4:
5353 024430 104413          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
5354 024432 004737 025324      JSR      PC,2#MODFSUB
5355 024436 046252 125252      1S:  .WORD 46252,125252     :AC
5356 024442 040300 000000      2S:  .WORD 40300,0         :FSRC
5357 024446 000000 000000      3S:  .WORD 0,0             :FRACTIONAL RES.
5358 024452 046377 177777      4S:  .WORD 46377,-1       :INTEGER RES.
5359 024456 046252 125252      5S:  .WORD 46252,125252     :ERROR FRACTIONAL RES.
5360 024462 040300 000000      6S:  .WORD 40300,0         :ERROR INTEGER RES.
5361 024466 000013          7S:  13                      :FPS BEFORE EXECUTION.
5362 024470 000004          4          :FPS AFTER EXECUTION.
5363 024472 104053      8S:  ERROR    53      :ST 134
5364 024474 000401          BR     95
5365 024476 104060          ERROR    60
5366 024500
    
```

```

5367
5368
5369 024500
5370 024500 104413
5371 024502 004737 025324
5372 024506 077652 125252
5373 024512 040300 000000
5374 024516 000000 000000
5375 024522 077777 177777
5376 024526 077652 125252
5377 024532 040300 000000
5378 024536 000000
5379 024540 000004
5380 024542 104053
5381 024544 000401
5382 024546 104060
5383 024550
5384
5385
5386 024550
5387 024550 104413
5388 024552 004737 025324
5389 024556 046200 000001
5390 024562 040340 000000
5391 024566 000000 000000
5392 024572 046340 000001
5393 024576 040000 000000
5394 024602 177777 177777
5395 024606 000013
5396 024610 000004
5397 024612 104061
5398
5399 024614 000401
5400 024616 104064
5401 024620
5402
5403
5404 024620
5405 024620 104413
5406 024622 004737 025324
5407 024626 046000 000001
5408 024632 040340 000000
5409 024636 040100 000000
5410 024642 046140 000001
5411 024646 000000 000000
5412 024652 177777 177777
5413 024656 000000
5414 024660 000000
5415 024662 104062
5416
5417 024664 000401
5418 024666 104054
5419 024670
5420
5421
5422 024670

```

```

:MODF TEST WITH EXPONENT OF THE RESULT = 127
GGG5:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS
JSR PC,2#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,2#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
BR 9$
ERROR 54
9$:

```

```

:MODF TEST WITH EXPONENT OF THE RESULT = 24
GGG7:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,2#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
BR 9$
ERROR 54
9$:

```

```

:MODF TEST WITH EXPONENT OF THE RESULT = 10
GGG8:

```

```

5423 024670 104413 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
5424 024672 004737 025324 JSR PC,3#MODFSUB
5425 024676 042577 177777 1%: .WORD 42577,-1 ;AC
5426 024702 040200 000000 2%: .WORD 40200,0 ;FSRC
5427 024706 040177 176000 3%: .WORD 40177,176000 ;FRACTIONAL RES.
5428 024712 042577 140000 4%: .WORD 42577,140000 ;INTEGER RES.
5429 024716 177777 177777 5%: .WORD -1,-1 ;ERROR FRACTIONAL RES.
5430 024722 177777 177777 6%: .WORD -1,-1 ;ERROR INTEGER RES.
5431 024726 000000 7%: 0 ;FPS BEFORE EXECUTION.
5432 024730 000000 8%: 0 ;FPS AFTER EXECUTION.
5433 024732 104053 BR 53
5434 024734 000401 BR 9%
5435 024736 104054 ERROR 54
5436 024740 9%:
    
```

:MODF TEST WITH THE EXPONENT OF THE RESULT = 10
 GGG9:

```

5440 024740 104413 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
5441 024742 004737 025324 JSR PC,3#MODFSUB
5442 024746 042577 140001 1%: .WORD 42577,140001 ;AC
5443 024752 040200 000000 2%: .WORD 40200,0 ;FSRC
5444 024756 034600 000000 3%: .WORD 34600,0 ;FRACTIONAL RES.
5445 024762 042577 140000 4%: .WORD 42577,140000 ;INTEGER RES.
5446 024766 000000 000000 5%: .WORD 0,0 ;ERROR FRACTIONAL RES.
5447 024772 177777 177777 6%: .WORD -1,-1 ;ERROR INTEGER RES.
5448 024776 000000 7%: 0 ;FPS BEFORE EXECUTION.
5449 025000 000000 8%: 0 ;FPS AFTER EXECUTION.
5450 025002 104063 BR 63
5451 025004 000401 BR 9%
5452 025006 104054 ERROR 54
5453 025010 9%:
    
```

:MODF TEST WITH EXPONENT OF THE RESULT = 9
 GGG10:

```

5456 025010 104413 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
5457 025012 004737 025324 JSR PC,3#MODFSUB
5458 025016 042377 100000 1%: .WORD 42377,100000 ;AC
5459 025022 040200 000000 2%: .WORD 40200,0 ;FSRC
5460 025026 000000 000000 3%: .WORD 0,0 ;FRACTIONAL RES.
5461 025032 042377 100000 4%: .WORD 42377,100000 ;INTEGER RES.
5462 025036 177777 177777 5%: .WORD -1,-1 ;ERROR FRACTIONAL RES.
5463 025042 177777 177777 6%: .WORD -1,-1 ;ERROR INTEGER RES.
5464 025046 000013 7%: 13 ;FPS BEFORE EXECUTION.
5465 025050 000004 8%: 4 ;FPS AFTER EXECUTION.
5466 025052 104053 BR 53
5467 025054 000401 BR 9%
5468 025056 104054 ERROR 54
5469 025060 9%:
    
```

:MODF TEST WITH EXPONENT OF THE RESULT = 0
 GGG11:

```

5473 025060 104413 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
5474 025062 004737 025324 JSR PC,3#MODFSUB
5475 025066 040177 177777 1%: .WORD 40177,-1 ;AC
5476 025072 040200 000000 2%: .WORD 40200,0 ;FSRC
5477 025076 040177 177777 3%: .WORD 40177,-1 ;FRACTIONAL RES.
    
```

5479	025102	000000	000000
5480	025106	000000	000000
5481	025112	040177	177777
5482	025116	000017	
5483	025120	000000	
5484	025122	104064	
5485	025124	000401	
5486	025126	104064	
5487	025130		
5488			
5489			
5490	025130		
5491	025130	104413	
5492	025132	004737	025324
5493	025136	034377	177777
5494	025142	040200	000000
5495	025146	034377	177777
5496	025152	000000	000000
5497	025156	000000	000000
5498	025162	034377	177777
5499	025166	000000	
5500	025170	000000	
5501	025172	104064	
5502	025174	000401	
5503	025176	104064	
5504	025200		
5505			
5506			
5507	025200		
5508	025200	104413	
5509	025202	004737	025324
5510	025206	020000	000001
5511	025212	040300	000000
5512	025216	020100	000002
5513	025222	000000	000000
5514	025226	020100	000001
5515	025232	000000	000000
5516	025236	000000	
5517	025240	000000	
5518	025242	104065	
5519	025244	000401	
5520	025246	104054	
5521	025250		
5522			
5523			
5524	025250		
5525	025250	104413	
5526	025252	004737	025324
5527	025256	142777	170000
5528	025262	040200	000000
5529	025266	140000	000000
5530	025272	142777	160000
5531	025276	040000	000000
5532	025302	042777	160000
5533	025306	000007	
5534	025310	000010	

```

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: LPERR ; SET UP THE LOOP ON ERRCR ADDRESS.
JSR PC,2#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: LPERR ; SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,2#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$: 0 ; FPS BEFORE EXECUTION.
0 ; FPS AFTER EXECUTION.
8$: 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,2#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.

```

5535 025312 104066
 5536 025314 000401
 5537 025316 104067
 5538 025320 000167 000266
 5539
 5540
 5541
 5542
 5543
 5544
 5545
 5546
 5547
 5548
 5549
 5550
 5551
 5552
 5553
 5554
 5555
 5556
 5557
 5558
 5559
 5560
 5561
 5562
 5563
 5564
 5565
 5566
 5567
 5568
 5569
 5570
 5571
 5572
 5573

95: ERROR 66 ;SIGN OF FRACTION.
 BR 95
 ERROR 67 ;SIGN OF INTEGER.
 95: JMP GGGDONE ;GO TO NEXT TEST.

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

```

ACARG: .WORD X,X ;AC OPERAND
FSRCARG: .WORD Y,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 ACC 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.

5574 025324 012601
 5575 025326 012700 000200
 5576 025332 170100
 5577 025334 010100
 5578 025336 172410
 5579 025340 012700 025702
 5580 025344 172510
 5581 025346 016100 000030
 5582 025352 170100
 5583 025354 012737 025370 001236
 5584 025362 010100
 5585 025364 062700 000004
 5586
 5587 025370 171410
 5588
 5589 025372 170204
 5590 025374 012700 000200

```

MODFSUB:  MOV (SP)+,R1 ;GET A POINTER TO THE ARGUMENTS
          MOV #200,R0 ;SET FD MODE.
          LDFPS R0
          MOV R1,R0 ;SET UP ACC
          LDD (R0),AC0
          MOV #MODF1,R0 ;PUT A BACKGROUND PATTERN INTO AC1.
          LDD (R0),AC1
          MOV 30(R1),R0 ;SET UP THE FPS.
          LDFPS R0
          MOV #15,2#STMP2
          MOV R1,R0 ;COMPUTE THE ADDRESS OF THE FSRC.
          ADD #4,R0
1$: MODF (R0),AC0 ;EXECUTE THE TEST INSTRUCTION.
          STFPS R4 ;GET THE FPS.
          MOV #200,R0 ;SET FD MODE.
    
```

L08

```

5591 025400 170100          LDFPS  R0
5592 025402 012700 025662  MOV  #MODFT0,R0      ;GET THE FRACTIONAL RESULT.
5593 025406 174010          STD   ACO,(R0)
5594 025410 012700 025672  MOV  #MODFT1,R0      ;GET THE INTEGER RESULT.
5595 025414 174110          STD   AC1,(R0)
5596
5597 025416 010102          MOV  R1,R2           ;SAVE THE DATA IN CASE OF ERROR.
5598 025420 010237 001240  MOV  R2,#STMP3
5599 025424 062702 000004  ADD  #4,R2
5600 025430 010237 001242  MOV  R2,#STMP4
5601 025434 062702 000004  ADD  #4,R2
5602 025440 010237 001244  MOV  R2,#STMP5
5603 025444 062702 000004  ADD  #4,R2
5604 025450 010237 001246  MOV  R2,#STMP6
5605 025454 012737 025662 001250  MOV  #MODFT0,#STMP7
5606 025462 012737 025672 001252  MOV  #MODFT1,#STMP10
5607 025470 010437 001254  MOV  R4,#STMP11
5608 025474 016137 000032 001256  MOV  32(R1),#STMP12
5609
5610 025502 012702 025662          MOV  #MODFT0,R2      ;CHECK THE FRACTIONAL RESULT.
5611 025506 026112 000010  CMP  10(R1),(R2)
5612 025512 001027          BNE  10$            ;BRANCH IF INCORRECT.
5613 025514 026162 000012 000002  CMP  12(R1),2(R2)
5614 025522 001016          BNE  10$
5615
5616 025524 012702 025672          MOV  #MODFT1,R2      ;CHECK THE INTEGER RESULT.
5617 025530 026112 000014  CMP  14(R1),(R2)
5618 025534 001026          BNE  15$            ;BRANCH IF INCORRECT.
5619 025536 026162 000016 000002  CMP  16(R1),2(R2)
5620 025544 001022          BNE  15$
5621
5622 025546 026104 000032          CMP  32(R1),R4       ;CHECK THE FPS.
5623 025552 001034          BNE  20$            ;BRANCH IF INCORRECT.
5624
5625 025554 000161 000042          9$: JMP  42(R1)         ;RETURN.
5626
5627          ;FRACTIONAL ERROR.
5628 025560 026112 000020          10$: CMP  20(R1),(R2)  ;WAS THE ERROR ANTICIPATED?
5629 025564 010101          BNE  11$            ;BRANCH IF NOT ANTICIPATED.
5630 025566 026162 000022 000002  CMP  22(R1),2(R2)
5631 025574 010104          BNE  11$
5632 025576 010102          MOV  R1,R2           ;THE ERROR WAS ANTICIPATED SO
5633 025600 062702 000034          ADD  #34,R2          ;RETURN TO THE ERROR REPORT AT THE
5634          ;CALLING ROUTINE.
5635 025604 000112          JMP  (R2)
5636
5637 025606          11$:              ;THE ERROR WAS NOT ANTICIPATED SO
5638 025606 104053          12$: ERROR 53        ;REPORT THE INCORRECT FRACTION HERE.
5639 025610 000761          BR   9$
5640
5641          ;INTEGER ERROR.
5642 025612 026112 000024          15$: CMP  24(R1),(R2)  ;WAS THIS ERROR ANTICIPATED?
5643 025616 001010          BNE  16$            ;BRANCH IF NOT.
5644 025620 026162 000026 000002  CMP  26(R1),2(R2)
5645 025626 001004          BNE  16$
5646 025630 010102          MOV  R1,R2          ;THE ERROR WAS ANTICIPATED SO RETURN
  
```

M08

```

5647 025632 062702 000040          ADD    #40,R2          ;TO THE ERROR REPORT IN THE CALLING
5648                                     ;ROUTINE.
5649 025636 000112          JMP    (R2)
5650
5651 025640          16$:          ;THE ERROR WAS NOT ANTICIPATED SO REPORT
5652 025640 104054          17$:  ERROR    54      ;THE INTEGER FAILURE HERE.
5653 025642 000744          BR     9$
5654
5655          ;FPS INCORRECT.
5656 025644 010437 001254          20$:  MOV     R4,#STMP11 ;REPORT INCORRECT FPS.
5657 025650 016137 000032 001256          MOV     32(R1),#STMP12
5658 025656 104055          21$:  ERROR    55
5659 025660 000735          BR     9$
5660
5661 025662 000000 000000 000000 MODFT0: .WORD 0,0,0,0
5662 025670 000000
5663
5664 025672 000000 000000 000000 MODFT1: .WORD 0,0,0,0
5665 025700 000000
5666
5667 025702 177777 177777 177777 MODP1: .WORD -1,-1,-1,-1
5668 025710 177777
5669
5670 025712          GGDONE:
5671 025712 104412          RSETUP          ;GO INITIALIZE THE FPS AND STACK; AND
5672                                     ;SEE IF THE USER HAS EXPRESSED
5673                                     ;THE DESIRE TO CHANGE THE SOFTWARE
5674                                     ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
5675                                     ;THE USER TYPED CONTROL G?).
5676
5677
5678
5679
5680          ;*****
5681          ;*TEST 17      MODD TEST
5682          ;*
5683          ;*THIS IS A TEST OF THE MODD INSTRUCTION. IT MAKES USE OF A SUBROUTINE
5684          ;*TO SET UP THE ARGUMENTS, EXECUTE THE INSTRUCTION AND CHECK THE
5685          ;*RESULTS.
5686          ;*
5687          ;*****
5688 025714 000004          †ST17: SCOPE
5689
5690          ;MODD WITH (FSRC=AC=0)
5691          MHM1:
5692 025716 104413          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
5693 025720 004737 027422          JSR     PC,#MODDSUB
5694 025724 000000 000000 000000 1$:  .WORD 0,0,0,0          ;AC
5695 025732 000000
5696 025734 000000 000000 000000 2$:  .WORD 0,0,0,0          ;FSRC
5697 025742 000000
5698 025744 000000 000000 000000 3$:  .WORD 0,0,0,0          ;FRACTIONAL RES.
5699 025752 000000
5700 025754 000000 000000 000000 4$:  .WORD 0,0,0,0          ;INTEGER RES.
5701 025762 000000
5702 025764 000000 000000 000000 5$:  .WORD 0,0,0,0          ;ERROR FRACTIONAL RES.

```



```

5759          :MODC TEST WITH EXPONENT OF THE RESULT = 57
5760 026216          *****
5761 026216          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
5762 026220 004737 027422          JSR          PC,2#MODC SUB
5763 026224 056252 125252          1$: .WORD          56252,125252          ;AC
5764 026230 125252 125250          .WORD          125252,125250
5765 026234 040300 000000 000000 2$: .WORD          40300,0,0,0          ;FSRC
5766 026242 000000          .WORD          0,0,0,0          ;FRACTIONAL RES.
5767 026244 000000 000000 000000 3$: .WORD          0,0,0,0
5768 026252 000000          .WORD          56377,-1,-1,-4          ;INTEGER RES.
5769 026254 056377 177777 177777 4$: .WORD          56377,-1,-1,-4
5770 026262 177774          .WORD          0,0          ;ERROR FRACTIONAL RES.
5771 026264 000000 000000          .WORD          125252,125252
5772 026270 125252 125252          .WORD          56377,-1,-1,-1          ;ERROR INTEGER RES.
5773 026274 056377 177777 177777 6$: .WORD          56377,-1,-1,-1
5774 026302 177777          7$:          213          ;FPS BEFORE EXECUTION.
5775 026304 000213          204          ;FPS AFTER EXECUTION.
5776 026306 000204          ERROR          77          ;ST 526 TO 134 INTO 135
5777 026310 104077          BR          95
5778 026312 000401          ERROR          77
5779 026314 104077          9$:
5780 026316

```

```

5781          :MODC TEST WITH EXPONENT OF THE RESULT = 79
5782          *****
5783 026316          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
5784 026316 104413          JSR          PC,2#MODC SUB
5785 026320 004737 027422          .WORD          140240,0,0,0          ;AC
5786 026324 140240 000000 000000 1$: .WORD          140240,0,0,0
5787 026332 000000          .WORD          63714,146314          ;FSRC
5788 026334 063714 146314          .WORD          133572,167737
5789 026340 133572 167737          .WORD          0,0,0,0          ;FRACTIONAL RES.
5790 026344 000000 000000 000000 3$: .WORD          0,0,0,0
5791 026352 000000          .WORD          163777,-1          ;INTEGER RES.
5792 026354 163777 177777          .WORD          162531,125726
5793 026360 162531 125726          .WORD          -1,-1,-1,-1          ;ERROR FRACTIONAL RES.
5794 026364 177777 177777 177777 5$: .WORD          -1,-1,-1,-1
5795 026372 177777          .WORD          63777,-1          ;ERROR INTEGER RES.
5796 026374 063777 177777          .WORD          162531,125726
5797 026400 162531 125726          7$:          210          ;FPS BEFORE EXECUTION.
5798 026404 000210          204          ;FPS AFTER EXECUTION.
5799 026406 000204          ERROR          70
5800 026410 104070          BR          95
5801 026412 000401          ERROR          100          ;ST 526 BAD SIGN
5802 026414 104100
5803 026416

```

```

5804          :MODC TEST WITH EXPONENT OF THE RESULT = 57
5805          *****
5806 026416          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
5807 026416 104413          JSR          PC,2#MODC SUB
5808 026420 004737 027422          .WORD          56200,0,0,1          ;AC
5809 026424 056200 000000 000000 1$: .WORD          56200,0,0,1
5810 026432 000001          .WORD          40340,0,0,0          ;FSRC
5811 026434 040340 000000 000000 2$: .WORD          40340,0,0,0
5812 026442 000000          .WORD          0,0,0,0          ;FRACTIONAL RES.
5813 026444 000000 000000 000000 3$: .WORD          0,0,0,0
5814 026452 000000

```

5815	026454	056340	000000	000000	45:	.WORD	56340,0,0,1	; INTEGER RES.
5816	026462	000001						
5817	026464	040000	000000	000000	55:	.WORD	40000,0,0,0	; ERROR FRACTIONAL RES.
5818	026472	700000						
5819	026474	056340	000000	000000	65:	.WORD	56340,0,0,1	; ERROR INTEGER RES.
5820	026500	000001						
5821	026504	000213			75:	213		; FPS BEFORE EXECUTION.
5822	026506	000204				204		; FPS AFTER EXECUTION.
5823	026510	104101			85:	ERROR	101	; CONSTANT BAD (NOT 56)
5824								; OR ST 525 TO 050 INTO 150
5825	026512	000401				BR	95	
5826	026514	104101				ERROR	101	
5827	026516				95:			
5828								
5829								
5830	026516							; MODC TEST WITH EXPONENT OF THE RESULT = 56
5831	026516	104413				LPERR		HHH7: ; SET UP THE LOOP ON ERROR ADDRESS.
5832	026520	004737	027422			JSR	PC, 2#MODDSUB	
5833	026524	056000	000000	000000	15:	.WORD	56000,0,0,1	; AC
5834	026532	000001						
5835	026534	040340	000000	000000	25:	.WORD	40340,0,0,0	; FSRC
5836	026542	000000						
5837	026544	040100	000000	000000	35:	.WORD	40100,0,0,0	; FRACTIONAL RES.
5838	026552	000000						
5839	026554	056140	000000	000000	45:	.WORD	56140,0,0,1	; INTEGER RES.
5840	026562	000001						
5841	026564	000000	000000	000000	55:	.WORD	0,0,0,0	; ERROR FRACTIONAL RES.
5842	026572	000000						
5843	026574	056140	000000	000000	65:	.WORD	56140,0,0,1	; ERROR INTEGER RES.
5844	026602	000001						
5845	026604	000213			75:	213		; FPS BEFORE EXECUTION.
5846	026606	000200				200		; FPS AFTER EXECUTION.
5847	026610	104102			85:	ERROR	102	; BAD CONSTANT (NOT 56) OR
5848								; ST 525 TO 150 INTO 050
5849	026612	000401				BR	95	
5850	026614	104102				ERROR	102	
5851	026616				95:			
5852								
5853								
5854	026616							; MODC TEST WITH EXPONENT OF THE RESULT = 36
5855	026616	104413				LPERR		HHH8: ; SET UP THE LOOP ON ERROR ADDRESS.
5856	026620	004737	027422			JSR	PC, 2#MODDSUB	
5857	026624	051177	177777	177777	15:	.WORD	51177,-1,-1,-1	; AC
5858	026632	177777						
5859	026634	040200	000000	000000	25:	.WORD	40200,0,0,0	; FSRC
5860	026642	000000						
5861	026644	040177	177760	000000	35:	.WORD	40177,-20,0,0	; FRACTIONAL RES.
5862	026652	000000						
5863	026654	051177	177777	177760	45:	.WORD	51177,-1,-20,0	; INTEGER RES.
5864	026662	000000						
5865	026664	177777	177777	177777	55:	.WORD	-1,-1,-1,-1	; ERROR FRACTIONAL RES.
5866	026672	177777						
5867	026674	177777	177777	177777	65:	.WORD	-1,-1,-1,-1	; ERROR INTEGER RES.
5868	026702	177777						
5869	026704	000217			75:	217		; FPS BEFORE EXECUTION.
5870	026706	000200				200		; FPS AFTER EXECUTION.

```

5871 026710 104070      BS:  ERROR 70
5872 026712 000401      BR    95
5873 026714 104071      ERROR 71
5874 026716
5875
5876      ;MODC TEST WITH EXPONENT OF THE RESULT = 30
5877      ###9:
5878 026716 104413      LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.
5879 026720 004737 027422      JSR      PC,2#MODDSUB
5880 026724 040200 000000 000000 1$: .WORD 40200,0,0,0 ;AC
5881 026732 000000
5882 026734 047577 177777      2$: .WORD 47577,-1 ;FSRC
5883 026740 176000 000001      .WORD 176000,1
5884 026744 031600 000000 000000 3$: .WORD 31600,0,0,0 ;FRACTIONAL RES.
5885 026752 000000
5886 026754 047577 177777      4$: .WORD 47577,-1 ;INTEGER RES.
5887 026760 176000 000000      .WORD 176000,0
5888 026764 000000 000000 000000 5$: .WORD 0,0,0,0 ;ERROR FRACTIONAL RES.
5889 026772 000000
5890 026774 047577 177777 177777 6$: .WORD 47577,-1,-1,-1 ;ERROR INTEGER RES.
5891 027002 177777
5892 027004 000200      7$: 200 ;FPS BEFORE EXECUTION.
5893 027006 000200      200 ;FPS AFTER EXECUTION.
5894 027010 104103      8$:  ERROR 103 ;(NORMALIZE) ST 532 TO 122
5895      BR    95 ;INTO NORM.
5896 027012 000401      ERROR 104 ;AC V 1 (<= X14
5897 027014 104104      ;OR ST 733 TO 156 INTO 157.
5898
5899 027016      9$:
5900
5901      ;MODC TEST WITH EXPONENT OF THE RESULT = 31
5902      ###10:
5903 027016 104413      LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.
5904 027020 004737 027422      JSR      PC,2#MODDSUB
5905 027024 047777 177777      1$: .WORD 47777,-1 ;AC
5906 027030 177000 000000      .WORD 177000,0
5907 027034 040200 000000 000000 2$: .WORD 40200,0,0,0 ;FSRC
5908 027042 000000
5909 027044 000000 000000 000000 3$: .WORD 0,0,0,0 ;FRACTIONAL RES.
5910 027052 000000
5911 027054 047777 177777      4$: .WORD 47777,-1 ;INTEGER RES.
5912 027060 177000 000000      .WORD 177000,0
5913 027064 000000 000000 177000 5$: .WORD 0,0,177000,0 ;ERROR FRACTIONAL RES.
5914 027072 000000
5915 027074 177777 177777 177777 6$: .WORD -1,-1,-1,-1 ;ERROR INTEGER RES.
5916 027102 177777
5917 027104 000213      7$: 213 ;FPS BEFORE EXECUTION.
5918 027106 000204      204 ;FPS AFTER EXECUTION.
5919 027110 104105      8$:  ERROR 105 ;(BUT FD) STORE X10
5920 027112 000401      BR    95
5921 027114 104071      ERROR 71
5922 027116
5923
5924      ;MODC TEST WITH EXPONENT OF THE RESULT = 0
5925      ###11:
5926 027116 104413      LPERR      ;SET UP THE LOOP ON ERROR ADDRESS.

```

5927	027120	004737	027422			JSR	PC, 2#MODSUB		
5928	027124	040200	000000	000000	1\$:	.WORD	40200,0,0,0	:	AC
5929	027132	000000							
5930	027134	040177	072727		2\$:	.WORD	40177,72727	:	FSRC
5931	027140	127272	072727			.WORD	127272,72727		
5932	027144	040177	072727		3\$:	.WORD	40177,72727	:	FRACTIONAL RES.
5933	027150	127272	072727			.WORD	127272,72727		
5934	027154	000000	000000	000000	4\$:	.WORD	0,0,0,0	:	INTEGER RES.
5935	027162	000000							
5936	027164	177777	177777	177777	5\$:	.WORD	-1,-1,-1,-1	:	ERROR FRACTIONAL RES.
5937	027172	177777							
5938	027174	000000	000000	177777	6\$:	.WORD	0,0,-1,-1	:	ERROR INTEGER RES.
5939	027202	177777							
5940	027204	000200			7\$:	200		:	FPS BEFORE EXECUTION.
5941	027206	000200				200		:	FPS AFTER EXECUTION.
5942	027210	104070			8\$:	ERROR	70		
5943	027212	000401				BR	9\$		
5944	027214	104106				ERROR	106	:	ST 246 TO 126 INTO 127 (BUT FD)
5945	027216				9\$:				
5946									
5947									
5948									
5949	027216	104413							
5950	027220	004737	027422			LPERR		:	SET UP THE LOOP ON ERROR ADDRESS.
5951	027224	003377	177777		1\$:	.WORD	3377,-1	:	AC
5952	027230	177777	052525			.WORD	-1,52525		
5953	027234	040200	000000	000000	2\$:	.WORD	40200,0,0,0	:	FSRC
5954	027242	000000							
5955	027244	003377	177777		3\$:	.WORD	3377,-1	:	FRACTIONAL RES.
5956	027250	177777	052525			.WORD	-1,52525		
5957	027254	000000	000000	000000	4\$:	.WORD	0,0,0,0	:	INTEGER RES.
5958	027262	000000							
5959	027264	177777	177777	177777	5\$:	.WORD	-1,-1,-1,-1	:	ERROR FRACTIONAL RES.
5960	027272	177777							
5961	027274	000000	000000	177777	6\$:	.WORD	0,0,-1,-1	:	ERROR INTEGER RES.
5962	027302	177777							
5963	027304	000200			7\$:	200		:	FPS BEFORE EXECUTION.
5964	027306	000200				200		:	FPS AFTER EXECUTION.
5965	027310	104070			8\$:	ERROR	70		
5966	027312	000401				BR	9\$		
5967	027314	104107				ERROR	107	:	ST 446 TO 126 INTO 127 (BUT FD)
5968	027316				9\$:				
5969									
5970									
5971	027316								
5972	027316	104413							
5973	027320	004737	027422			LPERR		:	SET UP THE LOOP ON ERROR ADDRESS.
5974	027324	040300	000000	000000	1\$:	.WORD	40300,0,0,0	:	AC
5975	027332	000000							
5976	027334	020200	000000	000000	2\$:	.WORD	20200,0,0,1	:	FSRC
5977	027342	000001							
5978	027344	020300	000000	000000	3\$:	.WORD	20300,0,0,2	:	FRACTIONAL RES.
5979	027352	000002							
5980	027354	000000	000000	000000	4\$:	.WORD	0,0,0,0	:	INTEGER RES.
5981	027362	000000							
5982	027364	000000	000000	177777	5\$:	.WORD	0,0,-1,-1	:	ERROR FRACTIONAL RES.

```

5983 027372 177777
5984 027374 177777 177777 177777 65: .WORD -1,-1,-1,-1 ;ERROR INTEGER RES.
5985 027402 177777
5986 027404 000200 75: 200 ;FPS BEFORE EXECUTION.
5987 027406 000200 200 ;FPS AFTER EXECUTION.
5988 027410 104110 85: ERROR 110 ;ST 127 INTO RND/IR
5989 027412 000401 BR 95
5990 027414 104071 ERROR 71
5991 027416 000137 030020 95: JMP 00000000 ;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 ACD 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.

```

6026 027422 012601 000200 MODDSUB: MOV (SP)+,R1 ;GET A POINTER TO THE ARGUMENTS
6027 027424 012700 MOV #200,R0 ;SET FD MODE.
6028 027430 170100 LDFPS R0
6029 027432 010100 MOV R1,R0 ;SET UP ACO
6030 027434 172410 LDD (R0),AC0
6031 027436 012700 MOV #MODP1,R0 ;PUT A BACKGROUND PATTERN INTO AC1.
6032 027442 172510 LDD (R0),AC1
6033 027444 016100 MOV 60(R1),R0 ;SET UP THE FPS.
6034 027450 170100 LDFPS R0
6035 027452 012737 027466 001236 MOV #15,0#STMP2
6036 027460 010100 MOV R1,R0 ;COMPUTE THE ADDRESS OF THE FSRC.
6037 027462 062700 000010 ADD #10,R0
6038
    
```

6039	027466	171410		1\$:	MODD	(R0),AC0		;EXECUTE THE TEST INSTRUCTION.
6040								
6041	027470	170204			STFPS	R4		;GET THE FPS.
6042	027472	012700	000200		MOV	#200,R0		;SET FD MODE.
6043	027476	170100			LDFPS	R0		
6044	027500	012700	030000		MOV	#MODDT0,R0		;GET THE FRACTIONAL RESULT.
6045	027504	174010			STD	AC0,(R0)		
6046	027506	012700	030010		MOV	#MODDT1,R0		;GET THE INTEGER RESULT.
6047	027512	174110			STD	AC1,(R0)		
6048								
6049	027514	010102			MOV	R1,R2		;SAVE THE DATA IN CASE OF ERROR.
6050	027516	010237	001240		MOV	R2,#STMP3		
6051	027522	062702	000010		ADD	#10,R2		
6052	027526	010237	001242		MOV	R2,#STMP4		
6053	027532	062702	000010		ADD	#10,R2		
6054	027536	010237	001244		MOV	R2,#STMP5		
6055	027542	062702	000010		ADD	#10,R2		
6056	027546	010237	001246		MOV	R2,#STMP6		
6057	027552	012737	030000	001250	MOV	#MODDT0,#STMP7		
6058	027560	012737	030010	001252	MOV	#MODDT1,#STMP10		
6059	027566	016137	000062	001256	MOV	62(R1),#STMP12		
6060	027574	010437	001254		MOV	R4,#STMP11		
6061								
6062	027600	012702	030000		MOV	#MODDT0,R2		;CHECK THE FRACTIONAL RESULT.
6063	027604	010103			MOV	R1,R3		
6064	027606	062703	000020		ADD	#20,R3		
6065	027612	012705	000004		MOV	#4,R5		
6066	027616	022223		2\$:	CMP	(R2)+,(R3)+		
6067	027620	001020			BNE	10\$;BRANCH IF INCORRECT.
6068	027622	077503			SOB	R5,2\$		
6069								
6070	027624	012702	030010		MOV	#MODDT1,R2		;CHECK THE INTEGER RESULT.
6071	027630	010103			MOV	R1,R3		
6072	027632	062703	000030		ADD	#30,R3		
6073	027636	012705	000004		MOV	#4,R5		
6074	027642	022223		3\$:	CMP	(R2)+,(R3)+		
6075	027644	001026			BNE	15\$;BRANCH IF INCORRECT.
6076	027646	077503			SOB	R5,3\$		
6077								
6078								
6079	027650	026104	000062		CMP	62(R1),R4		;CHECK THE FPS.
6080	027654	001042			BNE	20\$;BRANCH IF INCORRECT.
6081								
6082	027656	000161	000072	9\$:	JMP	72(R1)		;RETURN.
6083								
6084								
6085	027662	012702	030000					;FRACTIONAL ERROR.
6086	027666	010103		10\$:	MOV	#MODDT0,R2		;WAS THE FRACTIONAL ERROR ANTICIPATED?
6087	027670	062703	000040		MOV	R1,R3		
6088	027674	012705	000004		ADD	#40,R3		
6089	027700	022223			MOV	#4,R5		
6090	027702	001005		50\$:	CMP	(R2)+,(R3)+		
6091	027704	077503			BNE	11\$;BRANCH IF NOT ANTICIPATED.
6092	027706	010102			SOB	R5,50\$		
6093	027710	062702	000064		MOV	R1,R2		;THE ERROR WAS ANTICIPATED SO
6094					ADD	#64,R2		;RETURN TO THE ERROR REPORT AT THE
								;CALLING ROUTINE.

```

6095 027714 000112          JMP      (R2)
6096
6097 027716          15S:
6098 027716 104070          12S:  ERROR  70      ;THE ERROR WAS NOT ANTICIPATED SO
6099 027720 000756          BR      9S      ;REPORT THE INCORRECT FRACTION HERE.
6100
6101          ;INTEGER ERROR.
6102 027722 012702 030010          15S:  MOV      #MODD1,R2      ;WAS THE INTEGER ERROR ANTICIPATED?
6103 027726 010103          MOV      R1,R3
6104 027730 062703 000050          ADD      #50,R3
6105 027734 012705 000004          MOV      #4,R5
6106 027740 022223          60S:  CMP      (R2)+,(R3)+
6107 027742 001005          BNE     17S      ;BRANCH IF NOT ANTICIPATED.
6108 027744 077503          SOB     R5,60S
6109 027746 010102          MOV      R1,R2
6110 027750 062702 000070          ADD      #70,R2      ;THE ERROR WAS ANTICIPATED SO RETURN
;TO THE ERROR REPORT IN THE CALLING
;ROUTINE.
6111
6112 027754 000112          JMP      (R2)
6113
6114 027756          16S:
6115 027756 104071          17S:  ERROR  71      ;THE ERROR WAS NOT ANTICIPATED SO REPORT
6116 027760 000736          BR      9S      ;THE INTEGER FAILURE HERE.
6117
6118          ;FPS INCORRECT.
6119 027762 010437 001254          20S:  MOV      R4,#$TMP11      ;REPORT INCORRECT FPS.
6120 027766 016137 0000E2 001256          MOV      62(R1),#$TMP12
6121 027774 104072          21S:  ERROR  72
6122 027776 000727          BR      9S
6123
6124 030000 000000 000000 000000  MODD0: .WORD  0,0,0,0
6125 030006 000000
6126
6127 030010 000000 000000 000000  MODD1: .WORD  0,0,0,0
6128 030016 000000
6129
6130 030020          HHHDONE:
6131 030020 104412          RSETUP      ;GO INITIALIZE THE FPS AND STACK; AND
6132          ;SEE IF THE USER HAS EXPRESSED
6133          ;THE DESIRE TO CHANGE THE SOFTWARE
6134          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
6135          ;THE USER TYPED CONTROL G?).
6136
6137
6138
6139
6140          ;*****
6141          ;*TEST 20      UNDER/OVER FLOW, USING MODF WITH TRAPS DISABLED. TEST
6142          ;*
6143          ;*THIS IS A TEST OF THE MODF OVERFLOW AND UNDERFLOW CONDITIONS. IT MAKES
6144          ;*USE OF A SUBROUTINE TO SETUP THE OPERANDS, EXECUTE THE MODF INSTRUCTION
6145          ;*AND CHECK THE RESULTS. TRAPS ARE DISABLED DURING THIS TEST.
6146          ;*
6147          ;*****
6148 030022 000004          †ST20:  SCOPE
6149
6150          ;UNDERFLOW TEST, WITH EXPONENT OF THE RESULT = -129, FIU = 1, FID = 1

```

```

6151 030024
6152 030024 104413
6153 030026 004767 000320
6154 030032 020123 045676
6155 030036 020200 000000
6156 030042 000123 045676
6157 030046 000000 000000
6158 030052 177777 177777
6159 030056 177777 177777
6160 030062 042000
6161 030064 142004
6162 030066 000012
6163 030070 104170
6164 030072 000401
6165 030074 104171
6166 030076
6167
6168
6169 030076
6170 030076 104413
6171 030100 004737 030352
6172 030104 010200 000000
6173 030110 010000 000000
6174 030114 000000 000000
6175 030120 000000 000000
6176 030124 177777 177777
6177 030130 177777 177777
6178 030134 005013
6179 030136 005004
6180 030140 000012
6181 030142 000240
6182 030144 000401
6183 030146 104171
6184 030150
6185
6186
6187 030150
6188 030150 104413
6189 030152 004737 030352
6190 030156 060052 125252
6191 030162 060200 000000
6192 030166 000000 000000
6193 030172 000052 125252
6194 030176 000000 000000
6195 030202 000000 000000
6196 030206 041000
6197 030210 141006
6198 030212 000010
6199 030214 104172
6200 030216 000401
6201 030220 104173
6202
6203 030222
6204
6205
6206 030222

```

```

MMM1:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,MODFOV
1$: .WORD 20123,45678 ;AC
2$: .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$: 42000 ;FPS BEFORE EXECUTION.
142004 ;FPS AFTER EXECUTION.
12 ;FEC
9$: ERROR 170 ;FEC INCORRECT, UNDERFLOW.
BR 9$
ERROR 171 ;AC V 1 (2,3) <= ZERO, ST 126.
9$:

;UNDERFLOW EXP OF RESULT = -193, FIU = 0, FID = 1
MMM2:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,MODFOV
1$: .WORD 10200,0 ;AC
2$: .WORD 10000,0 ;FSRC
3$: .WORD 0,0 ;FRACTIONAL RES.
4$: .WORD 0,0 ;INTEGER RES.
5$: .WORD -1,-1 ;ERROR FRACTIONAL RES.
6$: .WORD -1,-1 ;ERROR INTEGER RES.
7$: 5013 ;FPS BEFORE EXECUTION.
5004 ;FPS AFTER EXECUTION.
12 ;FEC
8$: NOP
BR 9$
ERROR 171
9$:

;OVERFLOW TEST WITH EXPONENT OF THE RESULT = 128, FIV = 1, FID = 1
MMM3:
LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
JSR PC,MODFOV
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.
7$: 41000 ;FPS BEFORE EXECUTION.
141006 ;FPS AFTER EXECUTION.
10 ;FEC
8$: ERROR 172 ;BAD FEC ON OVERFLOW.
BR 9$
ERROR 173 ;ST 520 TO STORE ZERO TWICE
;INTO 162
9$:

;OVERFLOW TEST WITH EXPONENT OF THE RESULT = 130, FIV = 0, FID = 1
MMM4:

```


6207 030222 104413
 6208 030224 004737 030352
 6209 030230 060345 067654
 6210 030234 060200 000000
 6211 030240 000000 000000
 6212 030244 000000 000000
 6213 030250 000000 000000
 6214 030254 000345 067654
 6215 030260 006011
 6216 030262 006006
 6217 030264 000010
 6218 030266 000240
 6219 030270 000401
 6220 030272 104174
 6221 030274

LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
 JSR PC,2#MODFOV
 1\$: .WORD 60345,67654 ;AC
 2\$: .WORD 60200,0 ;FSRC
 3\$: .WORD 0,0 ;FRACTIONAL RES.
 4\$: .WORD 0,0 ;INTEGER RES.
 5\$: .WORD 0,0 ;ERROR FRACTIONAL RES.
 6\$: .WORD 345,67654 ;ERROR INTEGER RES.
 7\$: 6011 ;FPS BEFORE EXECUTION.
 6006 ;FPS AFTER EXECUTION.
 10 ;FEC
 8\$: NOP
 BR 9\$
 ERROR 174 ;ST 520 TO 162 INTO STORE ZERO TWICE.
 9\$:

;OVERFLOW TEST WITH EXPONENT OF THE RESULT = 128, RESULT NEGATIVE
 ;AND FIV = 1, FID = 1
 MMS:

6222
 6223
 6224
 6225 030274
 6226 030274 104413
 6227 030276 004737 030352
 6228 030302 160252 125252
 6229 030306 060000 000000
 6230 030312 000000 000000
 6231 030316 100052 125252
 6232 030322 000000 000000
 6233 030326 000052 125252
 6234 030332 041000
 6235 030334 141006
 6236 030336 000010
 6237 030340 104172
 6238 030342 000401
 6239 030344 104175
 6240 030346 000137 030746
 6241
 6242
 6243
 6244
 6245
 6246
 6247
 6248
 6249
 6250
 6251
 6252
 6253
 6254
 6255
 6256
 6257
 6258
 6259
 6260
 6261
 6262

LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
 JSR PC,2#MODFOV
 1\$: .WORD 160252,125252 ;AC
 2\$: .WORD 60000,0 ;FSRC
 3\$: .WORD 0,0 ;FRACTIONAL RES.
 4\$: .WORD 100052,125252 ;INTEGER RES.
 5\$: .WORD 0,0 ;ERROR FRACTIONAL RES.
 6\$: .WORD 52,125252 ;ERROR INTEGER RES.
 7\$: 41000 ;FPS BEFORE EXECUTION.
 141006 ;FPS AFTER EXECUTION.
 10 ;FEC
 8\$: ERROR 172
 BR 9\$
 ERROR 175 ;ST 517, BAD SIGN.
 9\$: JMP 2#MMDONE ;GO TO THE NEXT TEST.

;THIS SUBROUTINE, MODFOV, 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
FEC: .WORD X ;FEC
ERR1: ERROR X ;FEC 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

```

6263 : THEN THE INTEGER PART IS COMPARED WITH INTRES. IF BOTH OF THESE ARE CORRECT
6264 : THEN THE FPS IS COMPARED WITH FPSA. AFTER EXECUTION IF NO ERRORS OCCURRED
6265 : THEN MODFOV WILL RETURN TO CONT. IF THE FPS WAS INCORRECT
6266 : IT IS REPORTED HERE. IF THE FRACTION IS INCORRECT IT IS COMPARED WITH
6267 : THE ANTICIPATED BAD FRACTION, ERFRES. IF THIS DOESN'T MATCH
6268 : THE TRUE RESULT THEN THE ERROR IS REPORTED HERE. IF THE ANTICIPATED
6269 : FAILURE MATCHES THE TRUE RESULT THEN MODFOV PASSES CONTROL TO THE
6270 : ERROR CALL AT ERR1. LIKewise IF THE INTEGER PART OF THE RESULT IS
6271 : NOT CORRECT THEN IT IS COMPARED WITH THE ANTICIPATED INTEGER
6272 : FAILURE. IF THIS DOESN'T MATCH THEN THE ERROR IS REPORTED HERE.
6273 : IF A MATCH IS MADE HOWEVER, MODFOV WILL RETURN CONTROL TO THE ERROR
6274 : CALL AT ERR2.
6275
6276 030352 012601 MODFOV: MOV (SP)+,R1 ;GET A POINTER TO THE ARGUMENTS
6277 030354 012700 000230 MOV #200,R0 ;SET FD MODE.
6278 030360 170100 LDFPS R0
6279 030362 010100 MOV R1,R0 ;SET UP ACO
6280 030364 172410 LDD (R0),AC0
6281 030366 012700 025702 MOV #MODF1,R0 ;PUT A BACKGROUND PATTERN INTO AC1.
6282 030372 172510 LDD (R0),AC1
6283 030374 016100 000030 MOV 30(R1),R0 ;SET UP THE FPS.
6284 030400 170100 LDFPS R0
6285 030402 012737 030416 001236 MOV #15,@#STMP2
6286 030410 010100 MOV R1,R0 ;COMPUTE THE ADDRESS OF THE FSRC.
6287 030412 062700 000004 ADD #4,R0
6288
6289 030416 171410 15: MODF (R0),AC0 ;EXECUTE THE TEST INSTRUCTION.
6290
6291 030420 170204 STFPS R4 ;GET THE FPS.
6292 030422 170305 STST R5 ;GET FEC.
6293 030424 012700 000200 MOV #200,R0 ;SET FD MODE.
6294 030430 170100 LDFPS R0
6295 030432 012700 030726 MOV #MODF0,R0 ;GET THE FRACTIONAL RESULT.
6296 030436 174010 STD ACO,(R0)
6297 030440 012700 030736 MOV #MODF01,R0 ;GET THE INTEGER RESULT.
6298 030444 174110 STD AC1,(R0)
6299
6300 030446 010102 MOV R1,R2 ;SAVE THE DATA IN CASE OF ERROR.
6301 030450 010237 001240 MOV R2,@#STMP3
6302 030454 062702 000004 ADD #4,R2
6303 030460 010237 001242 MOV R2,@#STMP4
6304 030464 062702 000004 ADD #4,R2
6305 030470 010237 001244 MOV R2,@#STMP5
6306 030474 062702 000004 ADD #4,R2
6307 030500 010237 001246 MOV R2,@#STMP6
6308 030504 012737 030726 001250 MOV #MODF0,@#STMP7
6309 030512 012737 030736 001252 MOV #MODF01,@#STMP10
6310 030520 010437 001254 MOV R4,@#STMP11
6311 030524 016137 000032 001256 MOV 32(R1),@#STMP12
6312 030532 010537 001260 MOV R5,@#STMP13
6313 030536 016137 000034 001262 MOV 34(R1),@#STMP14
6314
6315 030544 012702 030726 MOV #MODF0,R2 ;CHECK THE FRACTIONAL RESULT.
6316 030550 026112 000010 CMP 10(R1),(R2)
6317 030554 001025 BNE 10$ ;BRANCH IF INCORRECT.
6318 030556 026162 000012 000002 CMP 12(R1),2(R2)
    
```

```

6319 030564 001021      BNE      10$
6320
6321 030566 012702 030736      MOV      #MODFD1,R2      ;CHECK THE INTEGER RESULT.
6322 030572 026112 000014      CMP      14(R1),R2
6323 030576 001016      BNE      15$      ;BRANCH IF INCORRECT.
6324 030600 026162 000016 000002      CMP      16(R1),2(R2)
6325 030606 001012      BNE      15$
6326
6327 030610 026104 000032      CMP      32(R1),R4      ;CHECK THE FPS.
6328 030614 001024      BNE      20$      ;BRANCH IF INCORRECT.
6329
6330 030616 026105 000034      CMP      34(R1),R5      ;CHECK THE FEC.
6331 030622 001030      BNE      25$      ;BRANCH IF INCORRECT.
6332
6333 030624 000161 000044      9$:     JMP      44(R1)      ;RETURN.
6334
6335      ;FRACTIONAL ERROR.
6336 030630      10$:
6337 030630 104165      12$:     ERROR    165      ;THE ERROR WAS NOT ANTICIPATED SO
6338 030632 000774      BR        9$      ;REPORT THE INCORRECT FRACTION HERE.
6339
6340      ;INTEGER ERROR.
6341 030634 026112 000024      15$:     CMP      24(R1),R2      ;WAS THIS ERROR ANTICIPATED?
6342 030640 001010      BNE      16$      ;BRANCH IF NOT.
6343 030642 026162 000026 000002      CMP      26(R1),2(R2)
6344 030650 001004      BNE      16$
6345 030652 010102      MOV      R1,R2      ;THE ERROR WAS ANTICIPATED SO RETURN
6346 030654 062702 000042      ADD      #4,R2      ;TO THE ERROR REPORT IN THE CALLING
6347      ;ROUTINE.
6348 030660 000112      JMP      (R2)
6349
6350 030662      16$:
6351 030662 104166      17$:     ERROR    166      ;THE ERROR WAS NOT ANTICIPATED SO REPORT
6352 030664 000757      BR        9$      ;THE INTEGER FAILURE HERE.
6353
6354      ;FPS INCORRECT.
6355 030666 010437 001254      20$:     MOV      R4,#STMP11      ;REPORT INCORRECT FPS.
6356 030672 016137 000032 001256      MOV      32(R1),#STMP12
6357 030700 104167      21$:     ERROR    167
6358 030702 000750      BR        9$
6359
6360      ;REPORT FEC ERROR.
6361 030704 010537 001260      25$:     MOV      R5,#STMP13
6362 030710 016137 000034 001262      MOV      34(R1),#STMP14
6363 030716 010102      MOV      R1,R2
6364 030720 062702 000036      ADD      #36,R2
6365 030724 000112      JMP      (R2)
6366
6367 030726 000000 000000 000000 MODF0: .WORD 0,0,0,0
6368 030734 000000
6369
6370 030736 000000 000000 000000 MODF1: .WORD 0,0,0,0
6371 030744 000000
6372
6373 030746      MMMDONE:
6374 030746 104412      RSETUP      ;GO INITIALIZE THE FPS AND STACK; AND
  
```

M09

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

6375
6376
6377
6378
6379
6380
6381
6382
6383
6384
6385
6386
6387
6388
6389
6390
6391
6392
6393
6394
6395
6396
6397
6398
6399
6400
6401
6402
6403
6404
6405
6406
6407
6408
6409
6410
6411
6412
6413
6414
6415
6416
6417
6418
6419
6420
6421
6422
6423
6424
6425
6426
6427
6428
6429
6430

030750 000004

030752
030752 104413
030754 004737 031366
030760 020252 125252
030764 125252 125252
030770 020100 000000 000000
030776 000000
031000 000177 177777 177777
031006 177777
031010 000000 000000 000000
031016 000000
031020 020252 125252
031024 125252 125252
031030 000000 000000 177777
031036 177777
031040 042200
031042 142204
031044 000012
031046 104201
031050 000401
031052 104202
031054

 *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.
 *

 †ST21: SCOPE

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

LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
 JSR PC,2#MODDOV
 1\$: .WORD 20252,125252 ;AC
 .WORD 125252,125252
 2\$: .WORD 20100,0,0,0 ;FSRC
 3\$: .WORD 177,-1,-1,-1 ;FRACTIONAL RES.
 4\$: .WORD 0,0,0,0 ;INTEGER RES.
 5\$: .WORD 20252,125252 ;ERROR FRACTIONAL RES.
 .WORD 125252,125252
 6\$: .WORD 0,0,-1,-1 ;ERROR INTEGER RES.
 7\$: 42200 ;FPS BEFORE EXECUTION.
 142204 ;FPS AFTER EXECUTION.
 12 ;FEC
 8\$: ERROR 201 ;FEC INCORRECT ON UNDERFLOW.
 BR 9\$
 9\$: ERROR 202 ;ST 155 (BUT FD)

:UNDERFLOW TEST WITH EXPONENT OF THE RESULT = -193, FIU = 0, FID = 1
 NNN2:

LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
 JSR PC,2#MODDOV
 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.

6431	031132	000000	000000		6\$:	.WORD	0,0		; ERROR INTEGER RES.
6432	031136	123456	000000			.WORD	123456,0		
6433	031142	005213			7\$:	5213			; FPS BEFORE EXECUTION.
6434	031144	005204				5204			; FPS AFTER EXECUTION.
6435	031146	000012				12			
6436	031150	000240			8\$:	NOP			
6437	031152	000401				BR	9\$		
6438	031154	104203				ERROR	203		; ST 047 (BUT FD).
6439	031156				9\$:				
6440									
6441									; OVERFLOW TEST WITH EXPONENT OF THE RESULT = 128, FIV = 1, FID = 1
6442	031156				NNN3:				
6443	031156	104413				LPERR			; SET UP THE LOOP ON ERROR ADDRESS.
6444	031160	004737	031366			JSR	PC, 2#MODDOV		
6445	031164	060252	125252		1\$:	.WORD	60252, 125252		; AC
6446	031170	125252	125252			.WORD	125252, 125252		
6447	031174	060100	000000	000000	2\$:	.WORD	60100, 0, 0, 0		; FSRC
6448	031202	000000							
6449	031204	000000	000000	000000	3\$:	.WORD	0, 0, 0, 0		; FRACTIONAL RES.
6450	031212	000000							
6451	031214	000177	177777	177777	4\$:	.WORD	177, -1, -1, -1		; INTEGER RES.
6452	031222	177777							
6453	031224	000000	000000	000000	5\$:	.WORD	0, 0, 0, 0		; ERROR FRACTIONAL RES.
6454	031232	000000							
6455	031234	000177	177777		6\$:	.WORD	177, -1		; ERROR INTEGER RES.
6456	031240	125252	125252			.WORD	125252, 125252		
6457	031244	041200			7\$:	41200			; FPS BEFORE EXECUTION.
6458	031246	141206				141206			; FPS AFTER EXECUTION.
6459	031250	000010				10			; FEC
6460	031252	104204			8\$:	ERROR	204		; FEC BAD ON OVERFLOW.
6461	031254	000401				BR	9\$		
6462	031256	104205				ERROR	205		; ST 520 TO 162 INTO 163 (BUT FD).
6463	031260				9\$:				
6464									
6465									; OVERFLOW TEST WITH EXPONENT OF THE RESULT = 130, FIV = 0, FID = 1
6466	031260				NNN4:				
6467	031260	104413				LPERR			; SET UP THE LOOP ON ERROR ADDRESS.
6468	031262	004737	031366			JSR	PC, 2#MODDOV		
6469	031266	060200	000000		1\$:	.WORD	60200, 0		; AC
6470	031272	125252	000000			.WORD	125252, 0		
6471	031276	060200	000000	000000	2\$:	.WORD	60200, 0, 0, 0		; FSRC
6472	031304	000000							
6473	031306	000000	000000	000000	3\$:	.WORD	0, 0, 0, 0		; FRACTIONAL RES.
6474	031314	000000							
6475	031316	000000	000000	000000	4\$:	.WORD	0, 0, 0, 0		; INTEGER RES.
6476	031324	000000							
6477	031326	000000	000000	000000	5\$:	.WORD	0, 0, 0, 0		; ERROR FRACTIONAL RES.
6478	031334	000000							
6479	031336	000400	000000		6\$:	.WORD	400, 0		; ERROR INTEGER RES.
6480	031342	125252	000000			.WORD	125252, 0		
6481	031346	006211			7\$:	6211			; FPS BEFORE EXECUTION.
6482	031350	006206				6206			; FPS AFTER EXECUTION.
6483	031352	000010				10			; FEC
6484	031354	000240			8\$:	NOP			
6485	031356	000401				BR	9\$		
6486	031358	104206				ERROR	206		; ST 520 TO 162 INTO STORE ZERO TWICE.

6487 031362 000137 031774
6488
6489
6490
6491
6492
6493
6494
6495
6496
6497
6498
6499
6500
6501
6502
6503
6504
6505
6506
6507
6508
6509
6510
6511
6512
6513
6514
6515
6516
6517
6518
6519
6520
6521
6522 031366 012601
6523 031370 012700 00020C
6524 031374 170100
6525 031376 010100
6526 031400 172410
6527 031402 012700 025702
6528 031406 172510
6529 031410 016100 000060
6530 031414 170100
6531 031416 012737 031432 001236
6532 031424 010100
6533 031426 062700 000010
6534
6535 031432 171410
6536
6537 031434 170305
6538 031436 170204
6539 031440 012700 000200
6540 031444 170100
6541 031446 012700 031754
6542 031452 174010

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

MODDOV: MOV (SP)+,R1 ;GET A POINTER TO THE ARGUMENTS
MOV #200,R0 ;SET FD MODE.
LDFPS R0
MOV R1,R0 ;SET UP ACD
LDD (R0),ACD
MOV #MODD,R0 ;PUT A BACKGROUND PATTERN INTO AC1.
LDD (R0),AC1
MOV 60(R1),R0 ;SET UP THE FPS.
LDFPS R0
MOV #15,@STMP2
MOV R1,R0 ;COMPUTE THE ADDRESS OF THE FSRC.
ADD #10,R0
15: MODD (R0),ACD ;EXECUTE THE TEST INSTRUCTION.
STST R5 ;GET THE FPS.
STFPS R4 ;GET THE FPS.
MOV #200,R0 ;SET FD MODE.
LDFPS R0
MOV #MODDDO,R0 ;GET THE FRACTIONAL RESULT.
STD ACD,(R0)

```

6543 031454 012700 031764      MOV      #MODD01,R0      ;GET THE INTEGER RESULT.
6544 031460 144110              STO      AC1,(R0)
6545
6546 031462 010102              MOV      R1,R2          ;SAVE THE DATA IN CASE OF ERROR.
6547 031464 010237 001240          MOV      R2,@STMP3
6548 031470 062702 000010          ADD      #10,R2
6549 031474 010237 001242          MOV      R2,@STMP4
6550 031500 062702 000010          ADD      #10,R2
6551 031504 010237 001244          MOV      R2,@STMP5
6552 031510 062702 000010          ADD      #10,R2
6553 031514 010237 001246          MOV      R2,@STMP6
6554 031520 012737 031754 001250          MOV      #MODD00,@STMP7
6555 031526 012737 031764 001252          MOV      #MODD01,@STMP10
6556 031534 010437 001254          MOV      R4,@STMP11
6557 031540 016137 000062 001256          MOV      62(R1),@STMP12
6558 031546 010537 001260          MOV      R5,@STMP13
6559 031552 016137 000064 001262          MOV      64(R1),@STMP14
6560
6561 031560 012702 031754          MOV      #MODD00,R2      ;CHECK THE FRACTIONAL RESULT.
6562 031564 010103          MOV      R1,R3
6563 031566 062703 000020          ADD      #20,R3
6564 031572 012700 000004          MOV      #4,R0
6565 031576 022223              2$:      CMP      (R2)+,(R3)+
6566 031600 001023              BNE      10$
6567 031602 077003              SOB      R0,2$
6568
6569 031604 012702 031764          MOV      #MODD01,R2      ;CHECK THE INTEGER RESULT.
6570 031610 010103          MOV      R1,R3
6571 031612 062703 000030          ADD      #30,R3
6572 031616 012700 000004          MOV      #4,R0
6573 031622 022223              3$:      CMP      (R2)+,(R3)+
6574 031624 001013              BNE      15$
6575 031626 077003              SOB      R0,3$
6576
6577
6578 031630 026104 000062          CMP      62(R1),R4      ;CHECK THE FPS.
6579 031634 001027              BNE      20$
6580
6581 031636 026105 000064          CMP      64(R1),R5      ;CHECK THE FEC.
6582 031642 001033              BNE      25$
6583
6584 031644 000161 000074          9$:      JMP      74(R1)      ;RETURN.
6585
6586      ;FRACTIONAL ERROR.
6587 031650 10$:
6588 031650 104176 12$:      ERROR 176
6589 031652 000774          BR      9$
6590
6591      ;INTEGER ERROR.
6592 031654 012702 031764          15$:      MOV      #MODD01,R2      ;WAS THE INTEGER ERROR ANTICIPATED?
6593 031660 010103          MOV      R1,R3
6594 031662 062703 000050          ADD      #50,R3
6595 031666 012705 000004          MOV      #4,R5
6596 031672 022223          60$:      CMP      (R2)+,(R3)+
6597 031674 001005              BNE      17$
6598 031676 077503              SOB      R5,60$

```

D10

MAINDEC-11-DFPBA POP 11 34 FPP DIAGNOSTIC PART 2 MACY11 27 (1006) 01-NOV-76 21:12 PAGE 20
 DFPBA.F11 01-NOV-76 21:06 T21 UNDER-OVER FLOW, USING MODE WITH TRAPS DISABLED, TEST

```

6599 031700 010102          MOV    R1,R2          ;THE ERROR WAS ANTICIPATED SO RETURN
6600 031702 062702 000072    ADD    #72,R2         ;TO THE ERROR REPORT IN THE CALLING
6601                                ;ROUTINE.
6602 031706 000112          JMP    (R2)
6603
6604 031710          16S:          ;THE ERROR WAS NOT ANTICIPATED SO REPORT
6605 031710 104177          17S:  ERROR  177    ;THE INTEGER FAILURE HERE.
6606 031712 000754          BR     95
6607
6608                                ;FPS INCORRECT.
6609 031714 010437 001254          20S:  MOV    R4,@STMP11 ;REPORT INCORRECT FPS.
6610 031720 016137 000062 001256    MOV    62(R1),@STMP12
6611 031726 104200          21S:  ERROR  200
6612 031730 000745          BR     95
6613
6614                                ;REPORT FEC ERROR.
6615 031732 010537 001260          25S:  MOV    R5,@STMP13
6616 031736 016137 000064 001262    MOV    64(R1),@STMP14
6617 031744 010102          MOV    R1,R2
6618 031746 062702 000066          ADD    #66,R2
6619 031752 000112          JMP    (R2)
6620
6621 031754 000000 000000 000000  MODDD0: .WORD 0,0,0,0
6622 031762 000000
6623
6624 031764 000000 000000 000000  MODDD1: .WORD 0,0,0,0
6625 031772 000000
6626
6627 031774          NNNDONE:
6628 031774 104412          RSETUP          ;GO INITIALIZE THE FPS AND STACK; AND
6629                                ;SEE IF THE USER HAS EXPRESSED
6630                                ;THE DESIRE TO CHANGE THE SOFTWARE
6631                                ;VIRTUAL CONSOLE SWITCH REGISTER HAS
6632                                ;THE USER TYPED CONTROL G?).
6633
6634
6635  ;*****
6636  ;*TEST 22          INTERRUPT CORRECT FLOWS TEST
6637  ;*
6638  ;*THIS IS A TEST OF THE 'CORRECT' FLOWS. THIS PART OF THE MICRO CODE
6639  ;*HAS AS ITS PURPOSE INSURING THAT INTERRUPT REQUESTS MADE DURING
6640  ;*CERTAIN LENGTHY FPP INSTRUCTIONS GET HONORED. THIS IS DONE
6641  ;*IN A WAY SUCH THAT IF AN INTERRUPT REQUEST OCCURS DURING ONE
6642  ;*OF THESE INSTRUCTIONS THE STATE OF THAT INSTRUCTION'S
6643  ;*EXECUTION WILL BE THE SAME AS IF THAT INSTRUCTION HAD NEVER
6644  ;*BEEN FETCHED AND ITS EXECUTION NEVER STARTED. THUS THE MICRO CODE
6645  ;*WILL RESTORE ALL REGISTERS, BACK UP THE PC AND LEAVE THE
6646  ;*FPS AND ACD THROUGH ACS UNMODIFIED.
6647  ;*THE INSTRUCTIONS FOR WHICH THIS IS NECESSARY ARE:
6648  ;*   ADD (OR SUB)
6649  ;*   DIV
6650  ;*   MUL
6651  ;*   MOD
6652  ;*(BOTH DOUBLE AND FLOATING)
6653  ;*ALL ADDRESSING MODES WILL BE TRIED WITH THE ADD INSTRUCTION. THEN
6654  ;*EACH OF THE OTHER INSTRUCTIONS WILL BE TRIED USING MODE 1.
  
```



```

6655 : *NOTE THAT THIS TEST NEEDS A SPECIAL INTERRUPT MODULE.
6656 : *WHICH WILL PROBABLY ONLY BE PRESENT IN DEC'S MANUFACTURING ENVIRONMENT.
6657 : *TO RUN. THIS SPECIAL EQUIPMENT IS DESIGNED TO RAISE AN
6658 : *INTERRUPT REQUEST IN THE PROCESSOR IF A BIT IS SET IN ITS STATUS
6659 : *REGISTER AND ONLY WHEN AN FPP INSTRUCTION IS ENCOUNTERED.
6660 : *THEREFORE THIS TEST WILL BE RUN CONDITIONALLY (DEPENDENT UPON WHETHER
6661 : *OR NOT THE STATUS REGISTER OF THE TEST EQUIPMENT TIMES OUT WHEN REFERENCED.
6662 : *THIS TEST CAN ALSO BE DESELECTED BY TURNING SWITCH 7 OF THE
6663 : *SWITCH REGISTER (PHYSICAL OR VIRTUAL) ON.
6664 : *THE TEST ASSUMES THAT THE TEST EQUIPMENT'S STATUS REGISTER IS AT
6665 : *LOCATION 77774 (NOTE THAT ALL REFERENCES TO THIS LOCATION ARE
6666 : *MADE INDIRECT THROUGH THIS PROGRAMS LOCATION CORINT, SO THAT
6667 : *IF THE USER HAS MODIFIED THE TEST EQUIPMENT'S STATUS REGISTER TO
6668 : *RESPOND TO A DIFFERENT ADDRESS LOCATION CORINT MUST BE
6669 : *MADE TO CONTAIN THAT STATUS REGISTER'S NEW ADDRESS).
6670 : *THIS PROGRAM ASSUMES THAT THE TRAP VECTOR FOR THE TEST EQUIPMENT IS :IC.
6671 : *AGAIN NOTE THAT ALL REFERENCES, TO THIS TRAP VECTOR ARE INDIRECT, THROUGH
6672 : *THIS PROGRAM'S LOCATION CORTAP (IF THE TEST EQUIPMENT IS
6673 : *MADE TO TRAP TO A DIFFERENT VECTOR LOCATION CORTAP MUST CONTAIN THE
6674 : *ADDRESS OF THIS VECTOR).
6675 : *
6676 : *
6677 : *****

```

```

6677 031776 000004 †ST22: SCOPE
6678
6679 032000 032777 000200 147132 BIT #200,2SWR ;SEE IF THE USER HAS DESELECTED THIS
6680 ;TEST USING THE SWITCH REGISTER.
6681 032006 001402 BEQ COR1 ;IF NOT SEE IF TEST EQUIPMENT IS PRESENT.
6682 032010 000137 033146 JMP #CORDONE ;ELSE DO NOT RUN TEST.
6683
6684 032014 012737 032040 000004 COR1: MOV #COR2,#ERRVECT ;SEE IF THE TEST EQUIPMENT'S STATUS
6685 032022 012777 000000 001112 MOV #0,CORINT ;REGISTER TIMES OUT.
6686 032030 012737 036646 000004 MOV #CPSPUR,#ERRVECT
6687 032036 000406 BR COR3 ;DIDN'T TIME OUT SO START TEST.
6688
6689 032040 022626 COR2: CMP (SP)+,(SP)+ ;IF THE REFERENCE TIMES OUT DO
6690 032042 012737 036646 000004 MOV #CPSPUR,#ERRVECT
6691 032050 000137 033146 JMP #CORDONE ;NOT RUN TEST.
6692
6693 :TEST ADDD MODE 0
6694 032054 COR3:
6695 032054 005227 177777 INC #-1
6696 032060 001002 BNE COR33
6697 032062 104401 TYPE
6698 032064 037621 .WORD CORMES
6699 032066 COR33:
6700 032066 104413 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
6701 032070 004737 032646 JSR PC,#CORSUB
6702 032074 040200 000100 000200 1$: .WORD 40200,100,200,300 ;ACD
6703 032102 000300
6704 032104 123456 2$: .WORD 123456 ;RD
6705 032106 000200 3$: 200 ;FPS
6706 032110 172000 4$: ADDD ACD,ACD ;TEST INSTRUCTION.
6707 032112 000240 NOP
6708 032114 005037 033130 CLR #CORFLG ;RESET INTERRUPT FLAG
6709 032120 104252 ERROR 252 ;NO INTERRUPT! TEST EQUIPMENT FAILED.
6710 032122 000401 BR 11$

```

F10

```

6711 032124 104253          55:  ERROR 253          ;INCORRECT STATE AT INTERRUPT.
6712 032126
6713
6714 032126          115:
        ;TEST ADD MODE 1
        COR4:
6715 032126 104413          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
6716 032130 004737 032646          JSR PC,2#CORSUB
6717 032134 040201 000555 077007 15:  .WORD 40201,555,77007,111111 ;ACC
6718 032142 111111
6719 032144 032134          25:  .WORD 15          ;RO
6720 032146 000217          35:  217          ;FPS
6721 032150 172010          45:  ADD (RO),ACC ;TEST INSTRUCTION
6722 032152 000240          NOP
6723 032154 005037 033130          CLR 2#CORFLG ;RESET INTERRUPT FLAG
6724 032160 104252          ERROR 252      ;REPORT FAILURE. NO INTERRUPT.
6725 032162 000401          BR 115
6726 032164 104254          ERROR 254
6727 032166          115:
6728
6729          ;TEST ADD MODE 2
        COR5:
6730 032166
6731 032166 104413          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
6732 032170 004737 032646          JSR PC,2#CORSUB
6733 032174 040202 111333 052525 15:  .WORD 40202,111333,52525,70707 ;ACC
6734 032202 070707
6735 032204 032174          25:  .WORD 15          ;RO
6736 032206 000205          35:  205          ;FPS
6737 032210 172020          45:  ADD (RO)+,ACC ;TEST INSTRUCTION
6738 032212 000240          NOP
6739 032214 005037 033130          CLR 2#CORFLG ;RESET THE INTERRUPT FLAG
6740 032220 104252          ERROR 252      ;REPORT FAILURE. NO INTERRUPT.
6741 032222 000401          BR 115
6742 032224 104255          ERROR 255
6743 032226          115: ;CORRECT FLOWS FAILED.
6744
6745          ;TEST ADD MODE 3
        COR6:
6746 032226
6747 032226 104413          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
6748 032230 004737 032646          JSR PC,2#CORSUB
6749 032234 040203 071735 072746 15:  .WORD 40203,71735,72746,1 ;ACC
6750 032242 000001
6751 032244 032270          25:  .WORD 105         ;RO
6752 032246 000206          35:  206         ;FPS
6753 032250 172030          45:  ADD 2(RO)+,ACC ;TEST INSTRUCTION
6754 032252 000240          NOP
6755 032254 005037 033130          CLR 2#CORFLG ;RESET THE INTERRUPT FLAG
6756 032260 104252          ERROR 252      ;REPORT FAILURE. NO INTERRUPT.
6757 032262 000403          BR 115
6758 032264 104256          55:  ERROR 256      ;CORRECT FLOWS FAILED.
6759 032266 000401          BR 115
6760 032270 032234          105: .WORD 15
6761
6762 032272          115:
6763          ;TEST ADD MODE 4
        COR7:
6764 032272
6765 032272 104413          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
6766 032274 004737 032646          JSR PC,2#CORSUB
  
```

G10

```

6767 032300 040204 123456 070123 1$: .WORD 40204,123456,70123,45671 ;ACC
6768 032306 045671
6769 032310 032310 2$: .WORD 15+10 ;RO
6770 032312 000212 3$: 212 ;FPS
6771 032314 172040 4$: ADD -(RO),ACC ;TEST INSTRUCTION
6772 032316 000240 NOP
6773 032320 005037 033130 CLR 2#CORFLG ;RESET THE INTERRUPT FLAG
6774 032324 104252 ERROR 252 ;REPORT FAILURE. NO INTERRUPT.
6775 032326 000401 BR 11$
6776 032330 104257 ERROR 257 ;CORRECT FLOWS FAILED
6777 032332 11$:
6778
6779 ;TEST ADD MODE 5
6780 032332 COR8:
6781 032332 104413 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
6782 032334 004737 032646 JSR PC,2#CORSUB
6783 032340 040205 076543 021076 1$: .WORD 40205,76543,21076,54321 ;ACC
6784 032346 054321
6785 032350 032376 2$: .WORD 10$+2 ;RO
6786 032352 000213 3$: 213 ;FPS
6787 032354 172050 4$: ADD 2-(RO),ACC ;TEST INSTRUCTION
6788 032356 000240 NOP
6789 032360 005037 033130 CLR 2#CORFLG ;RESET THE INTERRUPT FLAG
6790 032364 104252 ERROR 252 ;REPORT ERROR. NO INTERRUPT.
6791 032366 000403 BR 11$
6792 032370 104260 5$: ERROR 250 ;CORRECT FLOWS FAILED.
6793 032372 000401 BR 11$
6794 032374 032340 10$: .WORD 15
6795 032376 11$:
6796
6797 ;TEST ADD MODE 6
6798 032376 COR9:
6799 032376 104413 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
6800 032400 004737 032646 JSR PC,2#CORSUB
6801 032404 040206 034353 063730 1$: .WORD 40206,34353,63730,31323 ;ACC
6802 032412 031323
6803 032414 032403 2$: .WORD 15-1 ;RO
6804 032416 000214 3$: 214 ;FPS
6805 032420 172060 000001 4$: ADD 1(RO),ACC ;TEST INSTRUCTION
6806 032424 005037 033130 CLR 2#CORFLG
6807 032430 104252 ERROR 252 ;REPORT FAILURE NO TRAP.
6808 032432 000401 BR 11$
6809 032434 104261 5$: ERROR 261
6810 032436 11$:
6811
6812 ;TEST ADD MODE 7
6813 032436 COR10:
6814 032436 104413 LPERR ;SET UP THE LOOP ON ERROR ADDRESS.
6815 032440 004737 032646 JSR PC,2#CORSUB
6816 032444 040210 070107 062426 1$: .WORD 40210,70107,62426,55555 ;ACC
6817 032452 055555
6818 032454 032477 2$: .WORD 10$-1 ;RO
6819 032456 000204 3$: 204 ;FPS
6820 032460 172070 000001 4$: ADD 21(RO),ACC ;TEST INSTRUCTION
6821 032464 005037 033130 CLR 2#CORFLG
6822 032470 104252 ERROR 252 ;REPORT FAILURE NO TRAP
  
```

H10

```

6823 032472 000403          BR      115
6824 032474 104262          55:    ERROR 262          ;CORRECT FLOWS FAILED.
6825 032476 000401          BR      115
6826 032500 032444          105:   .WORD 15
6827 032502          115:
6828
6829          ;TEST DIVD MODE 1
6830 032502          COR11:
6831 032502 104413          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
6832 032504 004737 032646          JSR      PC,2#CORSUB
6833 032510 040211 033445 056677 15:   .WORD 40211,33445,56677,001122          ;ACO
6834 032516 001122          25:   .WORD 15          ;RO
6835 032520 032510          35:   205          ;FPS
6836 032522 000205          45:   DIVD (RO),ACO          ;TEST INSTRUCTION
6837 032524 174410          NOP
6838 032526 000240          CLR      2#CORFLG
6839 032530 005037 033130          ERROR 252          ;REPORT FAILURE, NO TRAP.
6840 032534 104252          BR      115
6841 032536 000401          55:    ERROR 263          ;CORRECT FLOWS FAILED.
6842 032540 104263          115:
6843 032542
6844
6845          ;TEST MULD MODE 1
6846 032542          COR12:
6847 032542 104413          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
6848 032544 004737 032646          JSR      PC,2#CORSUB
6849 032550 040212 165411 046252 15:   .WORD 40212,165411,46252,63650          ;ACO
6850 032556 063650          25:   .WORD 15          ;RO
6851 032560 032550          35:   .WORD 210          ;FPS
6852 032562 000210          45:   MULD (RO),ACO          ;TEST INSTRUCTION
6853 032564 171010          NOP
6854 032566 000240          CLR      2#CORFLG
6855 032570 005037 033130          ERROR 252          ;REPORT FAILURE, NO TRAP.
6856 032574 104252          BR      115
6857 032576 000401          55:    ERROR 264          ;CORRECT FLOWS FAILED.
6858 032600 104264          115:
6859 032602
6860
6861          ;TEST MODD MODE 1
6862 032602          COR13:
6863 032602 104413          LPERR          ;SET UP THE LOOP ON ERROR ADDRESS.
6864 032604 004737 032646          JSR      PC,2#CORSUB
6865 032610 040213 045654 054542 15:   .WORD 40213,45654,54542,171623          ;ACO
6866 032616 171623          25:   .WORD 15          ;RO
6867 032620 032610          35:   .WORD 412          ;FPS
6868 032622 000412          45:   MODD (RO),ACO          ;TEST INSTRUCTION.
6869 032624 171410          NOP
6870 032626 000240          CLR      2#CORFLG
6871 032630 005037 033130          ERROR 252          ;REPORT FAILURE NO TRAP.
6872 032634 104252          BR      115
6873 032636 000401          55:    ERROR 265          ;CORRECT FLOWS FAILED.
6874 032640 104265          115:   JMP      2#CORDONE          ;FINISHED TEST!
6875 032642 000137 033146
6876
6877          ;THIS SUBROUTINE, CORSUB, IS CALLED TO SET UP THE OPERANDS
6878          ;AND CHECK THE RESULTS IN THIS TEST. IT IS CALLED THUS:
    
```

6879
6880
6881
6882
6883
6884
6885
6886
6887
6888
6889
6890
6891
6892
6893
6894
6895
6896
6897
6898
6899
6900
6901
6902
6903
6904
6905
6906
6907
6908
6909
6910
6911
6912
6913
6914
6915
6916
6917
6918
6919
6920
6921
6922
6923
6924
6925
6926
6927
6928
6929
6930
6931
6932
6933
6934

032646 005037 033130
032652 012601
032654 010102
032656 012700 000200
032662 170100
032664 172412
032666 016100 000012
032672 170100
032674 016100 000010
032700 010102
032702 062702 000014
032706 010237 001236
032712 005037 177776
032716 012777 032744 000220
032724 012767 177777 000176
032732 012777 177777 000202
032740 000161 000014
032744 005137 033130
032750 001060
032752 012777 000000 000162
032760 170204

JSR PC,0#CORSUB
1\$: .WORD X,X,X,X ;ACD OPERAND
2\$: .WORD X ;RO
3\$: .WORD X ;FPS
4\$: INST ;TEST INSTRUCTION TO BE
;EXECUTED.
ADR ;AN ADDRESS OFFSET FOR
;CERTAIN MODES OR NOP.
ERROR 252 ;NO TRAP ERROR.
BR 11\$
5\$: ERROR N ;CORRECT FLOWS FAILURE.
BR 11\$;OPTIONAL FOR CERTAIN MODES.
10\$: .WORD ADDRESS ;OPTIONAL FOR CERTAIN MODES.
11\$:

:CORSUB WILL PICK UP A POINTER TO THE ARGUMENTS, IN R1. ACD, RO AND
:THE FPS WILL BE SET TO THE DESIGNATED VALUES. THEN THE TEST MODULE
:WILL BE SET UP TO INTERRUPT AND THE INSTRUCTION AT 4\$ EXECUTED. IF
:NO TRAP OCCURS THEN THE TEST MODULE IS FAULTY. WHEN THE TRAP OCCURS
:THE PC ON THE STACK SHOULD BE 4\$, AND ACD, RO AND THE FPS SHOULD NOT
:HAVE BEEN MODIFIED. IF EVERYTHING IS CORRECT CORSUB WILL RETURN TO
:5\$ PLUS TWO. IF AN ERROR IS DETECTED THEN CORSUB WILL RETURN TO THE
:ERROR REPORT AT 5\$.
:NOTE THAT A FLAG, CORFLG, IS SET TO -1 WHEN AN INTERRUPT IS PENDING.
:CORFLG IS ZERO OTHERWISE.

CORSUB: CLR 0#CORFLG ;SET FLAG TO INDICATE NO INTERRUPT
;PENDING.
MOV (SP)+,R1 ;GET A POINTER TO THE ARGUMENTS.
MOV R1,R2 ;SET ACD.
MOV #200,RO
LDFPS RO
LDD (R2),ACD
MOV 12(R1),RO ;SET UP THE FPS.
LDFPS RO
MOV 10(R1),RO ;SET UP RO.
MOV R1,R2
ADD #14,R2
MOV R2,0#STMP2 ;SAVE ADDRESS OF INSTRUCTION IN CASE
;OF ERROR.
CLR 0#PSW ;CLEAR THE PRIORITY TO ALLOW INTERRUPTS.
MOV #CORTV,0#CORTVP ;SET UP THE INTERRUPT VECTOR.
MOV #-1,CORFLG ;SET THE FLAG TO INDICATE
;AN INTERRUPT IS PENDING.
MOV #-1,0#CORINT ;ENABLE THE TEST EQUIPMENT'S
;TRAP FUNCTION AND GO
;EXECUTE THE INSTRUCTION.
JMP 14(R1)
:TRAP TO HERE WHEN THE INTERRUPT OCCURS.
CORTV: COM 0#CORFLG ;FIRST SEE IF AN INTERRUPT WAS PENDING.
BNE CORTV! ;IF NOT GO REPORT AN ERROR.
MOV #0,0#CORINT ;MAKE SURE THE TEST EQUIPMENT
;IS NOT INTERRUPT ENABLED.
STFPS R4 ;GET THE FPS.

6935	032762	012702	000200		MOV	#200,R2	:GET ACO
6936	032766	170102			LDFPS	R2	
6937	032770	012702	033132		MOV	#CORTMP,R2	
6938	032774	174012			STD	ACO,(R2)	
6939	032776	012737	033132	001240	MOV	#CORTMP,#STMP3	
6940	033004	010037	001244		MOV	RO,#STMP5	
6941	033010	010437	001250		MOV	R4,#STMP7	
6942	033014	011637	001254		MOV	(SP),#STMP11	
6943	033020	010102			MOV	R1,R2	
6944	033022	010237	001242		MOV	R2,#STMP4	
6945	033026	062702	000010		ADD	#10,R2	
6946	033032	012237	001246		MOV	(R2)+,#STMP6	
6947	033036	012237	001252		MOV	(R2)+,#STMP10	
6948	033042	010237	001256		MOV	R2,#STMP12	
6949	033046	021602			CMP	(SP),R2	:SEE IF THE TRAP OCCURRED :AT THE CORRECT ADDRESS.
6950							
6951	033050	001016			BNE	CORTVO	
6952	033052	022626			CMP	(SP)+,(SP)+	:RESET THE STACK.
6953	033054	020061	000010		CMP	RO,10(R1)	:SEE IF RO IS CORRECT.
6954	033060	001012			BNE	CORTVO	:BR IF NOT CORRECT.
6955	033062	010102			MOV	R1,R2	:SEE IF ACO WAS CORRECT
6956	033064	012703	033132		MOV	#CORTMP,R3	
6957	033070	012705	000004		MOV	#4,R5	
6958	033074	022223		1\$:	CMP	(R2)+,(R3)+	
6959	033076	001003			BNE	CORTVO	:BRANCH IF INCORRECT.
6960	033100	077503			SOB	R5,1\$	
6961	033102	000161	000032		JMP	32(R1)	:IF EVERYTHING IS CORRECT THEN RETURN.
6962							
6963	033106	000161	000030		CORTVO: JMP	30(R1)	:CORRECT FLOWS FAILED SO GO REPORT ERROR.
6964							
6965	033112	011637	001236		CORTV1: MOV	(SP),#STMP2	:AN INTERRUPT OCCURRED WHEN THE FLAG :CORFLG, DID NOT INDICATE THAT ONE WAS :PENDING SO REPORT SPURIOUS TRAP.
6966							
6967							
6968	033116	005037	033130		CLR	#CORFLG	
6969	033122	022626			CMP	(SP)+,(SP)+	
6970	033124	104266			ERROR	266	
6971	033126	000407			BR	CORDONE	
6972							
6973	033130	000000			CORFLG: .WORD	0	
6974	033132	000000	000000	000000	CORTMP: .WORD	0,0,0,0	
6975	033140	000000					
6976							
6977	033142	177774			CORINT: .WORD	177774	:THIS IS THE ADDRESS, 177774, OF THE :TEST EQUIPMENT'S STATUS REGISTER. :THE CONTENTS OF CORINT CAN BE MODIFIED :IF THIS STATUS REGISTER'S ADDRESS IS :CHANGED.
6978							
6979							
6980							
6981							
6982	033144	000110			CORTRP: .WORD	110	:THIS IS THE ADDRESS OF THE TEST EQUIPMENTS :TRAP VECTOR. LIKE THE STATUS REGISTER'S ADDRESS :DESCRIBED IMMEDIATELY ABOVE :THIS VECTOR CAN BE CHANGED, BUT THE :CONTENTS OF CORTRP MUST INDICATE THE :CHANGE.
6983							
6984							
6985							
6986							
6987							
6988							
6989	033146				CORDONE:		
6990	033146	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 -AS
:THE USER TYPED CONTROL G?).

6991
6992
6993
6994
6995
6996
6997
6998 03315C
6999
7000
7001
7002
7003
7004
7005
7006
7007
7008
7009
7010
7011 033150
7012 033150 000004
7013 033152 005067 145724
7014 033156 005067 146120
7015 033162 005267 146136
7016 033166 042767 100000 146130
7017 033174 005327
7018 033176 000001
7019 033200 003074
7020 033202 012737
7021 033204 000001
7022 033206 033176
7023 033210 104401 033216
7024 033214 000407
7025
7026 033234
7027 033234 016746 146064
7028
7029 033240 104403
7030 033242 006
7031 033243 000
7032 033244 104401 033252
7033 033250 000421
7034
7035 033314
7036 033314 016746 145572
7037
7038 033320 104403
7039 033322 006
7040 033323 000
7041 033324 104401 001313
7042 033330 005067 145556
7043 033334 013700 000042
7044 033340 001414
7045 033342 005046
7046 033344 012746 033352

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:

SCOPE
CLR \$STNM ;;ZERO THE TEST NUMBER
CLR \$TIMES ;;ZERO THE NUMBER OF ITERATIONS
INC \$PASS ;;INCREMENT THE PASS NUMBER
BIC #10000,\$PASS ;;DON'T ALLOW A NEG. NUMBER
DEC (PC)+ ;;LOOP?

\$EOPCT:

.WORD 1
BGT \$DOAGN ;;YES
MOV (PC)+,2(PC)+ ;;RESTORE COUNTER

\$ENDCT:

.WORD 1
\$EOPCT
TYPE \$55 ;;TYPE ASCIZ STRING
BR \$45 ;;GET OVER THE ASCIZ

;;\$55: .ASCIZ <12><15>/END PASS #/
\$45:

MOV \$PASS,-(SP) ;;SAVE \$PASS FOR TYPEOUT
;;TYPE PASS NUMBER IN OCTAL
;;GO TYPE--OCTAL ASCII
TYPE 6
;;TYPE 6 DIGITS
BYT 0
;;SUPPRESS LEADING ZEROS
TYPE \$75
;;TYPE ASCIZ STRING
BR \$65
;;GET OVER THE ASCIZ
;;\$75: .ASCIZ / TOTAL ERRORS SINCE LAST REPORT /
\$65:

MOV \$ERTTL,-(SP) ;;SAVE \$ERTTL FOR TYPEOUT
;;TOTAL NUMBER OF ERRORS IN OCTAL
;;GO TYPE--OCTAL ASCII
TYPE 6
;;TYPE 6 DIGITS
BYT 0
;;SUPPRESS LEADING ZEROS
TYPE \$CRLF
;;TYPE CARRIAGE RETURN, LINE FEED
CLR \$ERTTL
;;CLEAR ERROR TOTAL

\$GET42:

MOV \$42,R0
BGT \$DOAGN ;;BRANCH IF NO MONITOR
CLR -(SP) ;;INSURE THE "T" BIT IS CLEAR
MOV #CLR.T,-(SP) ;;SETUP FOR AN RTI OR RTT

```

7047 033350 000426 BR $RTRN ;;GO DO AN RTI OR RTT TO LOAD THE PSW
7048 ;;WITH A CLEARED "T" BIT
7049 033352 $CLR.T:
7050 033352 013700 000242 MOV $#42,R0 ;;INSURE R0 CONTAINS THE MONITORS
7051 033356 001405 SEQ $DOAGN ;;RETURN ADDRESS
7052 033360 000005 RESET ;;CLEAR THE WORLD
7053 033362 004710 $ENDAD: JSR PC,(R0) ;;GO TO MONITOR
7054 033364 000240 NOP ;;SAVE ROOM
7055 033366 000240 NOP ;;FOR
7056 033370 000240 NOP ;;ACT11
7057 033372 $DOAGN:
7058 033372 104400 TRAP ;;PUSH OLD PSW AND PC ON STACK
7059 033374 042716 000020 BIC #20,(SP) ;;CLEAR THE "T" BIT
7060 033400 032777 010000 145532 BIT #BIT12,$SWR ;;RUN WITH TRACE TRAP?
7061 033406 001005 IS ;;BR IF NO
7062 033410 005167 000020 COM $TBIT ;;IS IT TIME FOR TRACE TRAP
7063 033414 100402 BMI IS ;;BR IF NO
7064 033416 052716 000020 BIS #20,(SP) ;;SET TRACE TRAP
7065 033422 012746 033430 IS: MOV $SLOOP,-(SP) ;;JUMP TO START OF TEST
7066 033426 000002 $RTRN: RTI ;;RETURN--THIS IS CHANGED TO
;;AN "RTT" IF "RTT" IS A LEGAL
;;INSTRUCTION
7069 033430 $SLOOP:
7070 033430 000137 JMP $PC)+ ;;RETURN
7071 033432 005034 $RTNAD: .WORD LOOP
7072 033434 000000 $TBIT: .WORD 0 ;;;"T" BIT STATE INDICATOR
7073 033436 377 377 000 $ENULL: .BYTE -1,-1,0 ;;NULL CHARACTER STRING
7074 033442 .EVEN
7076 .SBTTL SCOPE HANDLER ROUTINE
7078 ;;*****
7079 ;;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
7080 ;;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
7081 ;;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
7082 ;;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
7083 ;;*SW14=1 LOOP ON TEST
7084 ;;*SW11=1 INHIBIT ITERATIONS
7085 ;;*SW09=1 LOOP ON ERROR
7086 ;;*SW08=1 LOOP ON TEST IN SWR<7:0>
7087 ;;*CALL
7088 ;;* SCOPE ;;SCOPE=IOT
7089
7090 033442 $SCOPE:
7091 033442 104406 040000 145466 IS: CKSWR ;;TEST FOR CHANGE IN SOFT-SWR
7092 033444 032777 BIT #BIT14,$SWR ;;LOOP ON PRESENT TEST?
7093 033452 001114 BNE $OVER ;;YES IF SW14=1
7094 ;;*****START OF CODE FOR THE XOR TESTER*****
7095 033454 000416 $XTSTR: BR $S ;;IF RUNNING ON THE "XOR" TESTER CHANGE
7096 ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
7097 033456 013746 000004 MOV $#ERRVEC,-(SP) ;;SAVE THE CONTENTS OF THE ERROR VECTOR
7098 033462 012737 033502 000004 MOV $S,$#ERRVEC ;;SET FOR TIMEOUT
7099 033470 005737 177060 TST $#177060 ;;TIME OUT ON XOR?
7100 033474 012637 000004 MOV (SP)+,$#ERRVEC ;;RESTORE THE ERROR VECTOR
7101 033500 000463 BR $SVLAD ;;GO TO THE NEXT TEST
7102 033502 022626 $$: CMP (SP)+,(SP)+ ;;CLEAR THE STACK AFTER A TIME OUT

```


M10

```

7103 033514 012637 000004      MOV      (SP)+, @#ERRVEC      ;; RESTORE THE ERROR VECTOR
7104 033510 000423              BR       7$                  ;; LOOP ON THE PRESENT TEST
7105 033512                    6$: *****END OF CODE FOR THE XOR TESTER*****
7106 033512 032777 000400 145420    BIT      #BIT08, @SWR        ;; LOOP ON SPEC. TEST?
7107 033520 001404              BEQ     2$                  ;; BR IF NO
7108 033522 127767 145412 145352    CMPB   @SWR, $STNM          ;; ON THE RIGHT TEST? SWR(7:0)
7109 033530 001465              BEQ     $OVER              ;; BR IF YES
7110 033532 105767 145345      2$:     TSTB   $ERFLG        ;; HAS AN ERROR OCCURRED?
7111 033536 001421              BEQ     3$                  ;; BR IF NO
7112 033540 126767 145351 145335    CMPB   $ERMAX, $ERFLG      ;; MAX. ERRORS FOR THIS TEST OCCURRED?
7113 033546 101015              BHI     3$                  ;; BR IF NO
7114 033550 032777 001000 145362    BIT      #BIT09, @SWR        ;; LOOP ON ERROR?
7115 033556 001404              BEQ     4$                  ;; BR IF NO
7116 033560 016767 145324 145320    7$:     MOV     $LPERR, $LPADR   ;; SET LOOP ADDRESS TO LAST SCOPE
7117 033566 000446              BR       $OVER
7118 033570 105067 145307      4$:     CLRB   $ERFLG          ;; ZERO THE ERROR FLAG
7119 033574 005067 145502              CLR     $TIMES             ;; CLEAR THE NUMBER OF ITERATIONS TO MAKE
7120 033600 000415              BR      1$                 ;; ESCAPE TO THE NEXT TEST
7121 033602 032777 004000 145330    3$:     BIT      #BIT11, @SWR    ;; INHIBIT ITERATIONS?
7122 033610 001011              BNE     1$                 ;; BR IF YES
7123 033612 005767 145506              TST    $PASS              ;; IF FIRST PASS OF PROGRAM
7124 033616 001406              BEQ     1$                 ;; INHIBIT ITERATIONS
7125 033620 005267 145260              INC     $ICNT              ;; INCREMENT ITERATION COUNT
7126 033624 026767 145452 145252    CMP     $TIMES, $ICNT       ;; CHECK THE NUMBER OF ITERATIONS MADE
7127 033632 002024              BGE     $OVER              ;; BR IF MORE ITERATION REQUIRED
7128 033634 012767 000001 145242    1$:     MOV     #1, $ICNT       ;; REINITIALIZE THE ITERATION COUNTER
7129 033642 016767 000052 145432      MOV     $MXCNT, $TIMES     ;; SET NUMBER OF ITERATIONS TO DO
7130 033650 105267 145226      $SVLAD: INCB   $STNM         ;; COUNT TEST NUMBERS
7131 033654 116767 145222 145440    MOVB   $STNM, $STNM        ;; SET TEST NUMBER IN APT MAILBOX
7132 033662 011667 145220      MOV     (SP), $LPADR       ;; SAVE SCOPE LOOP ADDRESS
7133 033666 011667 145216      MOV     (SP), $LPERR       ;; SAVE ERROR LOOP ADDRESS
7134 033672 005067 145406              CLR     $ESCAPE           ;; CLEAR THE ESCAPE FROM ERROR ADDRESS
7135 033676 112767 000001 145211    MOVB   #1, $ERMAX         ;; ONLY ALLOW ONE(1) ERROR ON NEXT TEST
7136 033704 016777 145172 145230    $OVER: MOV     $STNM, @DISPLAY  ;; DISPLAY TEST NUMBER
7137 033712 016716 145170      MOV     $LPADR, (SP)       ;; FUDGE RETURN ADDRESS
7138 033716 000002              RTI                        ;; FIXES PS
7139 033720 000001      $MXCNT: 1                 ;; MAX. NUMBER OF ITERATIONS
  
```

.SBTTL ERROR HANDLER ROUTINE

```

7140
7141
7142
7143 *****
7144 *THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
7145 *SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
7146 *AND GO TO ERTYPE ON ERROR
7147 *THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
7148 *SW15=1      HALT ON ERROR
7149 *SW13=1      INHIBIT ERROR TYPEOUTS
7150 *SW10=1      BELL ON ERROR
7151 *SW09=1      LOOP ON ERROR
7152 *CALL
7153 *          ERROR  N          ;; ERROR=EMT AND N=ERROR ITEM NUMBER
7154
7155 $ERROR:
7156 033722 104406      CKSWR              ;; TEST FOR CHANGE IN SOFT-SWR
7157 033724 105267 145153    7$:     INCB   $ERFLG          ;; SET THE ERROR FLAG
7158 033730 001775      BEQ     7$            ;; DON'T LET THE FLAG GO TO ZERO
  
```

```

7159 033732 016777 145144 145202      MOV      $STNM, @DISPLAY      ;; DISPLAY TEST NUMBER AND ERROR FLAG
7160 033740 032777 002000 145172      BIT      #BIT10, @SWR        ;; BELL ON ERROR?
7161 033746 001402                BEQ      1$                  ;; NO - SKIP
7162 033750 104401 001306                TYPE    $BELL                ;; RING BELL
7163 033754 005267 145132      1$:     INC      $ERTL        ;; COUNT THE NUMBER OF ERRORS
7164 033760 011667 145132      MOV      (SP), $ERRPC        ;; GET ADDRESS OF ERROR INSTRUCTION
7165 033764 162767 000002 145124      SUB      #2, $ERRPC
7166 033772 117767 145120 145114      MOV      @ERRPC, $ITEMB      ;; STRIP AND SAVE THE ERROR ITEM CODE
7167 034000 032777 020000 145132      BIT      #BIT13, @SWR        ;; SKIP TIMEOUT IF SET
7168 034006 001004                BNE     20$                  ;; SKIP TIMEOUTS
7169 034010 004767 002124      JSR     PC, ERTYPE          ;; GO TO USER ERROR ROUTINE
7170 034014 104401 001313      TYPE    , $CRLF
7171 034020                20$:
7172 034020 122767 000001 145310      CMPB    #APTENV, $ENV        ;; RUNNING IN APT MODE
7173 034026 001007                BNE     2$                    ;; NO, SKIP APT ERROR REPORT
7174 034030 116767 145060 000004      MOV      $ITEMB, 21$        ;; SET ITEM NUMBER AS ERROR NUMBER
7175 034036 004767 000740      JSR     PC, $ATY4          ;; REPORT FATAL ERROR TO APT
7176 034042 000                21$:     .BYTE    0
7177 034043 000                .BYTE    0
7178 034044 000777                22$:     BR      22$                    ;; APT ERROR LOOP
7179 034046 005777 145066      2$:     TST      @SWR            ;; HALT ON ERROR
7180 034052 100002                BPL     3$                    ;; SKIP IF CONTINUE
7181 034054 000000                HALT    ;; HALT ON ERROR!
7182 034056 104406                CKSWR  ;; TEST FOR CHANGE IN SOFT-SWR
7183 034060 032777 001000 145052      3$:     BIT      #BIT09, @SWR      ;; LOOP ON ERROR SWITCH SET?
7184 034066 001402                BEQ     4$                    ;; BR IF NO
7185 034070 016716 145014      MOV      $LPERR, (SP)       ;; FUDGE RETURN FOR LOOPING
7186 034074 005767 145204      4$:     TST      $ESCAPE        ;; CHECK FOR AN ESCAPE ADDRESS
7187 034100 001402                BEQ     5$                    ;; BR IF NONE
7188 034102 016716 145176      MOV      $ESCAPE, (SP)     ;; FUDGE RETURN ADDRESS FOR ESCAPE
7189 034106                5$:
7190 034106 022737 033362 000042      CMP      #$ENDAD, @#42      ;; ACT-11 AUTO-ACCEPT?
7191 034114 001001                BNE     6$                    ;; BRANCH IF NO
7192 034116 000000                HALT    ;; YES
7193 034120                6$:
7194 034120 032777 001000 145012      BIT      #BIT09, @SWR
7195 034126 001013                BNE     ERM10
7196 034130 011637 001162      MOV      (SP), @#$REGO      ; SEE IF ERROR #377
7197 034134 062737 177776 001162      ADD     #-2, @#$REGO
7198 034142 122777 000377 145012      CMPB    #377, @#$REGO
7199 034150 001002                BNE     ERM10
7200 034152 062716 000002      ADD     #2, (SP)
7201 034156 000002      ERM10: RTI
  
```

.SBTTL SAVE AND RESTORE RO-R5 ROUTINES

```

;*****
;SAVE RO-R5
;CALL:
;* SAVREG
;UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
;*
;*TOP---(+16)
;* +2---(+18)
;* +4---R5
  
```

7250
7251
7252
7253
7254
7255
7256
7257
7258
7259
7260
7261
7262
7263
7264
7265
7266
7267
7268
7269
7270
7271
7272
7273
7274
7275
7276
7277
7278
7279
7280
7281
7282
7283
7284
7285
7286
7287
7288
7289
7290
7291
7292
7293
7294
7295
7296
7297
7298
7299
7300
7301
7302
7303
7304
7305
7306
7307
7308
7309
7310
7311
7312
7313
7314
7315
7316
7317
7318
7319
7320
7321
7322
7323
7324
7325
7326
7327
7328
7329
7330
7331
7332
7333
7334
7335
7336
7337
7338
7339
7340
7341
7342
7343
7344
7345
7346
7347
7348
7349
7350
7351
7352
7353
7354
7355
7356
7357
7358
7359
7360
7361
7362
7363
7364
7365
7366
7367
7368
7369
7370
7371
7372
7373
7374
7375
7376
7377
7378
7379
7380
7381
7382
7383
7384
7385
7386
7387
7388
7389
7390
7391
7392
7393
7394
7395
7396
7397
7398
7399
7400
7401
7402
7403
7404
7405
7406
7407
7408
7409
7410
7411
7412
7413
7414
7415
7416
7417
7418
7419
7420
7421
7422
7423
7424
7425
7426
7427
7428
7429
7430
7431
7432
7433
7434
7435
7436
7437
7438
7439
7440
7441
7442
7443
7444
7445
7446
7447
7448
7449
7450
7451
7452
7453
7454
7455
7456
7457
7458
7459
7460
7461
7462
7463
7464
7465
7466
7467
7468
7469
7470
7471
7472
7473
7474
7475
7476
7477
7478
7479
7480
7481
7482
7483
7484
7485
7486
7487
7488
7489
7490
7491
7492
7493
7494
7495
7496
7497
7498
7499
7500
7501
7502
7503
7504
7505
7506
7507
7508
7509
7510
7511
7512
7513
7514
7515
7516
7517
7518
7519
7520
7521
7522
7523
7524
7525
7526
7527
7528
7529
7530
7531
7532
7533
7534
7535
7536
7537
7538
7539
7540
7541
7542
7543
7544
7545
7546
7547
7548
7549
7550
7551
7552
7553
7554
7555
7556
7557
7558
7559
7560
7561
7562
7563
7564
7565
7566
7567
7568
7569
7570
7571
7572
7573
7574
7575
7576
7577
7578
7579
7580
7581
7582
7583
7584
7585
7586
7587
7588
7589
7590
7591
7592
7593
7594
7595
7596
7597
7598
7599
7600
7601
7602
7603
7604
7605
7606
7607
7608
7609
7610
7611
7612
7613
7614
7615
7616
7617
7618
7619
7620
7621
7622
7623
7624
7625
7626
7627
7628
7629
7630
7631
7632
7633
7634
7635
7636
7637
7638
7639
7640
7641
7642
7643
7644
7645
7646
7647
7648
7649
7650
7651
7652
7653
7654
7655
7656
7657
7658
7659
7660
7661
7662
7663
7664
7665
7666
7667
7668
7669
7670
7671
7672
7673
7674
7675
7676
7677
7678
7679
7680
7681
7682
7683
7684
7685
7686
7687
7688
7689
7690
7691
7692
7693
7694
7695
7696
7697
7698
7699
7700
7701
7702
7703
7704
7705
7706
7707
7708
7709
7710
7711
7712
7713
7714
7715
7716
7717
7718
7719
7720
7721
7722
7723
7724
7725
7726
7727
7728
7729
7730
7731
7732
7733
7734
7735
7736
7737
7738
7739
7740
7741
7742
7743
7744
7745
7746
7747
7748
7749
7750
7751
7752
7753
7754
7755
7756
7757
7758
7759
7760
7761
7762
7763
7764
7765
7766
7767
7768
7769
7770
7771
7772
7773
7774
7775
7776
7777
7778
7779
7780
7781
7782
7783
7784
7785
7786
7787
7788
7789
7790
7791
7792
7793
7794
7795
7796
7797
7798
7799
7800
7801
7802
7803
7804
7805
7806
7807
7808
7809
7810
7811
7812
7813
7814
7815
7816
7817
7818
7819
7820
7821
7822
7823
7824
7825
7826
7827
7828
7829
7830
7831
7832
7833
7834
7835
7836
7837
7838
7839
7840
7841
7842
7843
7844
7845
7846
7847
7848
7849
7850
7851
7852
7853
7854
7855
7856
7857
7858
7859
7860
7861
7862
7863
7864
7865
7866
7867
7868
7869
7870
7871
7872
7873
7874
7875
7876
7877
7878
7879
7880
7881
7882
7883
7884
7885
7886
7887
7888
7889
7890
7891
7892
7893
7894
7895
7896
7897
7898
7899
7900
7901
7902
7903
7904
7905
7906
7907
7908
7909
7910
7911
7912
7913
7914
7915
7916
7917
7918
7919
7920
7921
7922
7923
7924
7925
7926
7927
7928
7929
7930
7931
7932
7933
7934
7935
7936
7937
7938
7939
7940
7941
7942
7943
7944
7945
7946
7947
7948
7949
7950
7951
7952
7953
7954
7955
7956
7957
7958
7959
7960
7961
7962
7963
7964
7965
7966
7967
7968
7969
7970
7971
7972
7973
7974
7975
7976
7977
7978
7979
7980
7981
7982
7983
7984
7985
7986
7987
7988
7989
7990
7991
7992
7993
7994
7995
7996
7997
7998
7999
8000

;*+6---R4
;*+8---R3
;*+10---R2
;*+12---R1
;*+14---R0

SSAVREG:
MOV R0, -(SP) ;; PUSH R0 ON STACK
MOV R1, -(SP) ;; PUSH R1 ON STACK
MOV R2, -(SP) ;; PUSH R2 ON STACK
MOV R3, -(SP) ;; PUSH R3 ON STACK
MOV R4, -(SP) ;; PUSH R4 ON STACK
MOV R5, -(SP) ;; PUSH R5 ON STACK
MOV 22(SP), -(SP) ;; SAVE PS OF MAIN FLOW
MOV 22(SP), -(SP) ;; SAVE PC OF MAIN FLOW
MOV 22(SP), -(SP) ;; SAVE PS OF CALL
MOV 22(SP), -(SP) ;; SAVE PC OF CALL
RTI

;*RESTORE R0-R5
;*CALL:
;* RESREG

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

.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: \$MULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
*NOTE2: \$FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
*NOTE3: \$FILLC CONTAINS THE CHARACTER TO FILL AFTER.
*

*CALL:
*1) USING A TRAP INSTRUCTION
* TYPE ,MESADR ;; MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
*OR
* TYPE
* MESADR
*

\$TYPE: TSTB \$TPFLG ;; IS THERE A TERMINAL?
BPL IS ;; BR IF YES
HALT ;; HALT HERE IF NO TERMINAL
BR 35 ;; LEAVE

7277	034266	010046		15:	MOV	RO, -(SP)	:: SAVE RO	
7278	034270	017600	000002		MOV	22(SP), RO	:: GET ADDRESS OF ASCIZ STRING	
7279	034274	122767	000001	145034	CMPB	#APTENV, \$ENV	:: RUNNING IN APT MODE	
7274	034302	001011			BNE	62\$:: NO GO CHECK FOR APT CONSOLE	
7275	034304	132757	000100	145025	BITB	#APTSPool, \$ENVM	:: SPOOL MESSAGE TO APT	
7276	034312	001405			BEG	62\$:: NO GO CHECK FOR CONSOLE	
7277	034314	010067	000004		MOV	RO, 61\$:: SETUP MESSAGE ADDRESS FOR APT	
7278	034320	004767	000446		JSR	PC, \$AT13	:: SPOOL MESSAGE TO APT	
7279	034324	000000		61\$:	.WORD	0	:: MESSAGE ADDRESS	
7280	034326	132757	000340	145003	62\$:	BITB	#APTCSUP, \$ENVM	
7281	034334	001003			BNE	60\$:: APT CONSOLE SUPPRESSED	
7282	034336	112046		2\$:	MOVB	(RO)+, -(SP)	:: YES, SKIP TYPE OUT	
7283	034340	001005			BNE	4\$:: BR IF IT ISN'T THE TERMINATOR	
7284	034342	005726			TST	(SP)+	:: IF TERMINATOR POP IT OFF THE STACK	
7285	034344	012600		50\$:	MOV	(SP)+, RO	:: RESTORE RO	
7286	034346	062716	000002	3\$:	ADD	62, (SP)	:: ADJUST RETURN PC	
7287	034352	000002			RTI		:: RETURN	
7288	034354	127716	000011	4\$:	CMPB	#HT, (SP)	:: BRANCH IF <HT>	
7289	034360	001430			BEG	8\$		
7290	034362	122716	000200		CMPB	#CRLF, (SP)	:: BRANCH IF NOT <CRLF>	
7291	034366	001006			BNE	5\$		
7292	034370	005726			TST	(SP)+	:: POP <CR><LF> EQUIV	
7293	034372	104401			TYPE		:: TYPE A CR AND LF	
7294	034374	001313			\$CRLF			
7295	034376	105067	000130		CLRB	\$CHARCNT	:: CLEAR CHARACTER COUNT	
7296	034402	000755			BR	2\$:: GET NEXT CHARACTER	
7297	034404	004767	000056	5\$:	JSR	PC, \$TYPEC	:: GO TYPE THIS CHARACTER	
7298	034410	126726	144542	6\$:	CMPB	\$FILLC, (SP)+	:: IS IT TIME FOR FILLER CHARS.?	
7299	034414	001350			BNE	2\$:: IF NO GO GET NEXT CHAR.	
7300	034416	016746	144532		MOV	\$NULL, -(SP)	:: GET # OF FILLER CHARS. NEEDED	
7301							:: AND THE NULL CHAR.	
7302	034422	105366	000001	7\$:	DECB	1(SP)	:: DOES A NULL NEED TO BE TYPED?	
7303	034426	002770			BLT	6\$:: BR IF NO--GO POP THE NULL OFF OF STACK	
7304	034430	004767	000032		JSR	PC, \$TYPEC	:: GO TYPE A NULL	
7305	034434	105367	000072		DECB	\$CHARCNT	:: DO NOT COUNT AS A COUNT	
7306	034440	000770			BR	7\$:: LOOP	
7307								
7308								
7309								
7310	034442	112716	000040	8\$:	MOVB	#' (SP)	:: REPLACE TAB WITH SPACE	
7311	034446	004767	000014	9\$:	JSR	PC, \$TYPEC	:: TYPE A SPACE	
7312	034452	132767	000007	000052	BITB	#7, \$CHARCNT	:: BRANCH IF NOT AT	
7313	034460	001372			BNE	9\$:: TAB STOP	
7314	034462	005726			TST	(SP)+	:: POP SPACE OFF STACK	
7315	034464	000724			BR	2\$:: GET NEXT CHARACTER	
7316	034466	105777	144456	\$TYPEC:	TSTB	2\$TPS	:: WAIT UNTIL PRINTER IS READY	
7317	034472	100375			BPL	\$TYPEC		
7318	034474	116677	000002	144450	MOVB	2(SP), 2\$TPB	:: LOAD CHAR TO BE TYPED INTO DATA REG.	
7319	034502	122766	000015	000002	CMPB	#CR, 2(SP)	:: IS CHARACTER A CARRIAGE RETURN?	
7320	034510	001003			BNE	1\$:: BRANCH IF NO	
7321	034512	105067	000014		CLRB	\$CHARCNT	:: YES--CLEAR CHARACTER COUNT	
7322	034516	000406			BR	\$TYPEX	:: EXIT	
7323	034520	122766	000012	000002	1\$:	CMPB	#LF, 2(SP)	:: IS CHARACTER A LINE FEED?
7324	034526	001402			BEG	\$TYPEX	:: BRANCH IF YES	
7325	034530	105227			INCB	(PC)+	:: COUNT THE CHARACTER	
7326	034532	000000			\$CHARCNT:	.WORD	0	

TYPE ROUTINE

03453 000207
 03454 000000
 03455 017646 000000
 03456 116667 000001 000211
 03457 112667 000207
 03458 062716 000002
 03459 000406
 03460 112767 000001 000171
 03461 112767 000006 000165
 03462 112767 000005 000154
 03463 010346
 03464 010446
 03465 010546
 03466 116704 000145
 03467 005404
 03468 062704 000006
 03469 110467 000132
 03470 116704 000125
 03471 016605 000012
 03472 005003
 03473 006105
 03474 000404
 03475 006105
 03476 006105
 03477 006105
 03478 010503
 03479 006103
 03480 105367 000076
 03481 100016
 03482 042703 177770

STYPEX: RTS PC

.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.
*STYPOS---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
*STYPOX---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
*STYPOS OR STYPOX
*CALL:
*   MOV   NUM,-(SP)      ;; NUMBER TO BE TYPED
*   TYPON          ;; CALL FOR TYPEOUT
*STYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
*CALL:
*   MOV   NUM,-(SP)      ;; NUMBER TO BE TYPED
*   TYPOC          ;; CALL FOR TYPEOUT
*STYPOS: MOV   2(SP),-(SP) ;; PICKUP THE MODE
*         MOVB 1(SP),SOFILL ;; LOAD ZERO FILL SWITCH
*         MOVB (SP)+,SOMODE+1 ;; NUMBER OF DIGITS TO TYPE
*         ADD  #2,(SP)      ;; ADJUST RETURN ADDRESS
*         BR   STYPOX
*STYPOC: MOVB #1,SOFILL    ;; SET THE ZERO FILL SWITCH
*         MOVB #6,SOMODE+1 ;; SET FOR SIX(6) DIGITS
*         MOVB #5,SOCNT    ;; SET THE ITERATION COUNT
*         MOV  R3,-(SP)    ;; SAVE R3
*         MOV  R4,-(SP)    ;; SAVE R4
*         MOV  R5,-(SP)    ;; SAVE R5
*         MOVB SOMODE+1,R4 ;; GET THE NUMBER OF DIGITS TO TYPE
*         NEG  R4
*         ADD  #6,R4      ;; SUBTRACT IT FOR MAX. ALLOWED
*         MOVB R4,SOMODE  ;; SAVE IT FOR USE
*         MOVB SOFILL,R4  ;; GET THE ZERO FILL SWITCH
*         MOV  12(SP),R5  ;; PICKUP THE INPUT NUMBER
*         CLR  R3         ;; CLEAR THE OUTPUT WORD
*         ROL  R5         ;; ROTATE MSB INTO "C"
*         BR   3$        ;; GO DO MSB
*         ROL  R5         ;; FORM THIS DIGIT
*         ROL  R5
*         ROL  R5
*         MOV  R5,R3
*         ROL  R3         ;; GET LSB OF THIS DIGIT
*         DECB SOMODE    ;; TYPE THIS DIGIT
*         BPL  7$        ;; BR IF NO
*         BIC  #177770,R3 ;; GET RID OF JUNK

```

```

7383 034672 001002      BNE      4$      ;; TEST FOR 0
7384 034674 005704      TST      R4      ;; SUPPRESS THIS 0?
7385 034676 001403      BEQ      5$      ;; BR IF YES
7386 034700 005204      4$: INC      R4      ;; DON'T SUPPRESS ANYMORE 0'S
7387 034702 052703 00006C    BIS      #'0,R3  ;; MAKE THIS DIGIT ASCII
7388 034706 052703 000040    5$: BIS      #' ,R3  ;; MAKE ASCII IF NOT ALREADY
7389 034712 110367 000040    MOV      R3,8$   ;; SAVE FOR TYPING
7390 034716 104401 034756    TYPE     8$      ;; GO TYPE THIS DIGIT
7391 034722 105367 000032    7$: DECB    $OCNT  ;; COUNT BY 1
7392 034726 003347      BGT      2$      ;; BR IF MORE TO DO
7393 034730 002402      BLT      6$      ;; BR IF DONE
7394 034732 005204      INC      R4      ;; INSURE LAST DIGIT ISN'T A BLANK
7395 034734 000744      BR       2$      ;; GO DO THE LAST DIGIT
7396 034736 012605    6$: MOV      (SP)+,R5  ;; RESTORE R5
7397 034740 012604      MOV      (SP)+,R4  ;; RESTORE R4
7398 034742 012603      MOV      (SP)+,R3  ;; RESTORE R3
7399 034744 016666 000002 000004    MOV      2(SP),4(SP) ;; SET THE STACK FOR RETURNING
7400 034752 012616      MOV      (SP)+,(SP)
7401 034754 000002      RTI          ;; RETURN
7402 034756      000      8$: .BYTE    0      ;; STORAGE FOR ASCII DIGIT
7403 034757      000      .BYTE    0      ;; TERMINATOR FOR TYPE ROUTINE
7404 034760      000      $OCNT:   .BYTE    0      ;; OCTAL DIGIT COUNTER
7405 034761      000      $OFILL: .BYTE    0      ;; ZERO FILL SWITCH
7406 034762 000000      $OMODE: .WORD    0      ;; NUMBER OF DIGITS TO TYPE
7407
7408      .SBTTL  APT COMMUNICATIONS ROUTINE
7409
7410      ..*****
7411 034764 112767 000001 000236  $ATY1:  MOV      #1,$FFLG  ;; TO REPORT FATAL ERROR
7412 034772 112767 000001 000226  $ATY3:  MOV      #1,$MFLG  ;; TO TYPE A MESSAGE
7413 035000 000403      BR       $ATYC
7414 035002 112767 000001 000220  $ATY4:  MOV      #1,$FFLG  ;; TO ONLY REPORT FATAL ERROR
7415 035010      $ATYC:
7416 035010 010046      MOV      R0,-(SP)  ;; PUSH R0 ON STACK
7417 035012 010146      MOV      R1,-(SP)  ;; PUSH R1 ON STACK
7418 035014 105767 000206      TST      $MFLG     ;; SHOULD TYPE A MESSAGE?
7419 035020 001450      BEQ      5$      ;; IF NOT: BR
7420 035022 122767 000001 144306      CMPB     #APTENV,$ENV  ;; OPERATING UNDER APT?
7421 035030 001031      BNE      3$      ;; IF NOT: BR
7422 035032 132767 000100 144277      BITB     #APTSPOOL,$ENVM  ;; SHOULD SPOOL MESSAGES?
7423 035040 001425      BEQ      3$      ;; IF NOT: BR
7424 035042 017600 000004      MOV      #4(SP),R0  ;; GET MESSAGE ADDR.
7425 035046 062766 000002 000004      ADD      #2,4(SP)   ;; BUMP RETURN ADDR.
7426 035054 005767 144236    1$: TST      $MSGTYPE  ;; SEE IF DONE W/ LAST XMISSION?
7427 035060 001375      BNE      1$      ;; IF NOT: WAIT
7428 035062 010067 144244      MOV      R0,$MSGAD  ;; PUT ADDR IN MAILBOX
7429 035066 105720    2$: TSTB    (R0)+     ;; FIND END OF MESSAGE
7430 035070 001376      BNE      2$
7431 035072 166700 144234      SUB      $MSGAD,R0  ;; SUB START OF MESSAGE
7432 035076 006200      ASR      R0        ;; GET MESSAGE LENGTH IN WORDS
7433 035100 010067 144230      MOV      R0,$MSGLGT  ;; PUT LENGTH IN MAILBOX
7434 035104 012767 000004 144204      MOV      #4,$MSGTYPE  ;; TELL APT TO TAKE MSG.
7435 035112 000413      BR       5$
7436 035114 017667 000004 000016    3$: MOV      #4(SP),4$  ;; PUT MSG ADDR IN JSR LINKAGE
7437 035122 062766 000002 000004      ADD      #2,4(SP)   ;; BUMP RETURN ADDRESS
7438 035130 016746 142642      MOV      177776,-(SP) ;; PUSH 177776 ON STACK

```

```

7439 035134 004767 177114          JSR    PC,STYPE      ;;CALL TYPE MACRO
7440 035140 000000          45:    .WORD        0
7441 035142          55:
7442 035142 105767 000062          105:   TSTB    $FFLG      ;; SHOULD REPORT FATAL ERRC?
7443 035146 001416          BEQ    125            ;; IF NOT: BR
7444 035150 005767 144162          TST    $ENV          ;; RUNNING UNDER APT?
7445 035154 001413          BEQ    125            ;; IF NOT: BR
7446 035156 005767 144134          115:   TST    $MSGTYPE   ;; FINISHED LAST MESSAGE?
7447 035162 001375          BNE    115            ;; IF NOT: WAIT
7448 035164 017667 000004 144126      MOV    24(SP), $FATAL ;; GET ERROR #
7449 035172 062766 000002 000004      ADD    #24(SP)        ;; BUMP RETURN ADDR.
7450 035200 005267 144112          INC    $MSGTYPE      ;; TELL APT TO TAKE ERROR
7451 035204 105067 00002C          125:   CLRB    $FFLG      ;; CLEAR FATAL FLAG
7452 035210 105067 000013          CLRB    $LFLG       ;; CLEAR LOG FLAG
7453 035214 105067 000006          CLRB    $MFLG       ;; CLEAR MESSAGE FLAG
7454 035220 012601          MOV    (SP)+,R1      ;; POP STACK INTO R1
7455 035222 012600          MOV    (SP)+,R0      ;; POP STACK INTO R0
7456 035224 000207          RTS     PC            ;; RETURN
7457 035226 000          $MFLG: .BYTE        0      ;; MESSG. FLAG
7458 035227 000          $LFLG: .BYTE        0      ;; LOG FLAG
7459 035230 000          $FFLG: .BYTE        0      ;; FATAL FLAG
7460          035232          .EVEN
7461          00020C          APTSIZE=200
7462          000001          APTENV=001
7463          00010C          APTSPool=100
7464          00004C          APTCSUP=040
7465
7466          .SBTTL  TTY INPUT ROUTINE
7467
7468          ;;*****
7469          .ENABL  LSB
7470
7471          ;;*****
7472          *SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
7473          *ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
7474          *SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
7475          *WHEN OPERATING IN TTY FLAG MODE.
7476 035232 022767 000176 143700  $CKSWR: CMP    #SWREG,SWR      ;; IS THE SOFT-SWR SELECTED?
7477 035240 001074          BNE    155            ;; BRANCH IF NO
7478 035242 105777 143676          TSTB   2$TKS         ;; CHAR THERE?
7479 035246 100071          BPL    155            ;; IF NO, DON'T WAIT AROUND
7480 035250 117746 143672          MOVB   2$TKB, -(SP)  ;; SAVE THE CHAR
7481 035254 042716 177600          BIC    #17, (SP)     ;; STRIP-OFF THE ASCII
7482 035260 022726 000007          CMP    #7, (SP)+     ;; IS IT A CONTROL G?
7483 035264 001062          BNE    155            ;; NO, RETURN TO USER
7484 035266 126727 143642 000001  $AUTOB, #1          ;; ARE WE RUNNING IN AUTO-MODE?
7485 035274 001456          BEQ    155            ;; BRANCH IF YES
7486
7487 035276 104401 035641          TYPE   , $CNTLG      ;; ECHO THE CONTROL-G (G)
7488 035302 104401 035646          $GTSWR: TYPE   $MSWR    ;; TYPE CURRENT CONTENTS
7489 035306 016746 142664          MOV    $WREG, -(SP)  ;; SAVE SWREG FOR TYPEOUT
7490 035312 104402          TYPOC          ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
7491 035314 104401 035657          TYPE   , $MNEW      ;; PROMPT FOR NEW SWR
7492 035320 005046          195:   CLR    -(SP)         ;; CLEAR COUNTER
7493 035322 005046          CLR    -(SP)         ;; THE NEW SWR
7494 035324 105777 143614          75:    TSTB   2$TKS     ;; CHAR THERE?

```

```

7495 035330 100375          BPL      75          ::IF NOT TRY AGAIN
7496
7497 035332 117746 143610    MOVB    25TKB, -(SP)  ::PICK UP CHAR
7498 035336 042716 177500    BIC     #10177, SP   ::MAKE IT 7-BIT ASCII
7499
7500
7501
7502 035342 021627 000025    95:     CMP     (SP), #25  ::IS IT A CONTROL-U?
7503 035346 001005          BNE     105          ::BRANCH IF NOT
7504 035350 104401 035534    TYPE    $CNTLU      ::YES, ECHO CONTROL-U (U)
7505 035354 062706 000006    205:    ADD     #6, SP     ::IGNORE PREVIOUS INPUT
7506 035360 000757          BR      195          ::LET'S TRY IT AGAIN
7507
7508
7509 035362 021627 000015    105:    CMP     (SP), #15   ::IS IT A <CR>?
7510 035366 001022          BNE     165          ::BRANCH IF NO
7511 035370 005766 000004    TST     4(SP)       ::YES, IS IT THE FIRST CHAR?
7512 035374 001403          BEQ     115          ::BRANCH IF YES
7513 035376 016677 000002 143534    MOV     2(SP), 25WR  ::SAVE NEW SWR
7514 035404 062706 000006    115:    ADD     #6, SP     ::CLEAR UP STACK
7515 035410 104401 001313    145:    TYPE    $SCRLF     ::ECHO <CR> AND <LF>
7516 035414 126727 143515 000001    CMPB   $INTAG, #1  ::RE-ENABLE TTY KBD INTERRUPTS?
7517 035422 001003          BNE     155          ::BRANCH IF NOT
7518 035424 012777 000100 143512    MOV     #100, 25TKS  ::RE-ENABLE TTY KBD INTERRUPTS
7519 035432 000002          RTI     ::RETURN
7520 035434 004767 177026    165:    JSR     PC, $TYPEC  ::ECHO CHAR
7521 035440 021627 000060    CMP     (SP), #60   ::CHAR < 0?
7522 035444 002420          BLT     185          ::BRANCH IF YES
7523 035446 021627 000067          CMP     (SP), #67   ::CHAR > ??
7524 035452 003015          BGT     185          ::BRANCH IF YES
7525 035454 042726 000060    BIC     #60, (SP)+  ::STRIP-OFF ASCII
7526 035460 005766 000002    TST     2(SP)       ::IS THIS THE FIRST CHAR
7527 035464 001403          BEQ     175          ::BRANCH IF YES
7528 035466 006316          ASL     (SP)        ::NO, SHIFT PRESENT
7529 035470 006316          ASL     (SP)        ::CHAR OVER TO MAKE
7530 035472 006316          ASL     (SP)        ::ROOM FOR NEW ONE.
7531 035474 005266 000002    175:    INC     2(SP)       ::KEEP COUNT OF CHAR
7532 035500 056616 177776    BIS     -2(SP), (SP) ::SET IN NEW CHAR
7533 035504 000707          BR      75          ::GET THE NEXT ONE
7534 035506 104401 001312    185:    TYPE    $QUES     ::TYPE ?<CR><LF>
7535 035512 000720          BR      205          ::SIMULATE CONTROL-U
7536 .DSABL  LSB
7537
7538
7539
7540
7541
7542
7543
7544
7545
7546
7547 035514 011646          $RDCHR: MOV    (SP), -(SP)  ::PUSH DOWN THE PC
7548 035516 016666 000004 000002    MOV    4(SP), 2(SP)  ::SAVE THE PS
7549 035524 105777 143414    15:     TSTB   25TKS       ::WAIT FOR
7550 035530 100375          BPL     15          ::A CHARACTER

```

```

::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

```



```

7551 035532 117766 143410 000004      MOVB  05TKB,4(SP)      ;; READ THE TTY
7552 035540 042766 177600 000004      BIC   #10<177>,4(SP)  ;; GET RID OF JUNK IF ANY
7553 035546 026627 000004 000023      CMP   4(SP),#23       ;; IS IT A CONTROL-S?
7554 035554 001013                BNE   35              ;; BRANCH IF NO
7555 035556 105777 143362      25:   TSTB  05TKS          ;; WAIT FOR A CHARACTER
7556 035562 100375                BPL   25              ;; LOOP UNTIL ITS THERE
7557 035564 117746 143356      MOVB  05TKB,-(SP)     ;; GET CHARACTER
7558 035570 042716 177600      BIC   #*C177,(SP)    ;; MAKE IT 7-BIT ASCII
7559 035574 022627 000021      CMP   (SP)+,#21      ;; IS IT A CONTROL-Q?
7560 035600 001366                BNE   25              ;; IF NOT DISCARD IT
7561 035602 000750                BR    15              ;; YES, RESUME
7562 035604 026627 000004 000140  35:   CMP   4(SP),#140     ;; IS IT UPPER CASE?
7563 035612 002407                BLT   45              ;; BRANCH IF YES
7564 035614 026627 000004 000175      CMP   4(SP),#175     ;; IS IT A SPECIAL CHAR?
7565 035622 003003                BGT   45              ;; BRANCH IF YES
7566 035624 042766 000040 000004      BIC   #40,4(SP)      ;; MAKE IT UPPER CASE
7567 035632 000002      45:   RTI                    ;; GO BACK TO USER
7568 035634 052536 005015      00C  $CNTLU: .ASCIZ /?U<15><12>  ;; CONTROL "U"
7569 035641 136 006507 000012  $CNTLG: .ASCIZ /?G<15><12>  ;; CONTROL "G"
7570 035646 005015 053523 020122  $MSWR: .ASCIZ <15><12>/SWR = /
7571 035654 020075 000 000012
7572 035657 040 047040 053505  $MNEW: .ASCIZ / NEW = /
7573 035664 036440 000040
7574
7575      .SBTTL TRAP DECODER
7576
7577      ;;*****
7578      ;;THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
7579      ;;AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
7580      ;;OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
7581      ;;GO TO THAT ROUTINE.
7582
7583 035670 010046                $TRAP: MOV   ,RO,-(SP)    ;; SAVE RO
7584 035672 016600 000002      MOV   2(SP),RO        ;; GET TRAP ADDRESS
7585 035676 005740                TST   -(RO)           ;; BACKUP BY 2
7586 035700 111000                MOVB  (RO),RO         ;; GET RIGHT BYTE OF TRAP
7587 035702 006300                ASL   RO              ;; POSITION FOR INDEXING
7588 035704 016000 035724      MOV   $TRAPD(RO),RO   ;; INDEX TO TABLE
7589 035710 000200                RTS   RO              ;; GO TO ROUTINE
7590
7591
7592      ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
7593
7594 035712 011646                $TRAP2: MOV  (SP),-(SP)  ;; MOVE THE PC DOWN
7595 035714 016666 000004 000002      MOV  4(SP),2(SP)     ;; MOVE THE PSW DOWN
7596 035722 000002                RTI                    ;; RESTORE THE PSW
7597
7598      .SBTTL TRAP TABLE
7599
7600      ;;THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
7601      ;;BY THE "TRAP" INSTRUCTION.
7602
7603      :      ROUTINE
7604      :      -----
7605 035724 035712                $TRAPD: .WORD  $TRAP2
7606 035726 034254                $TYPE  ;;CALL=TYPE TRAP+1(104401) TTY TYPEOUT ROUTINE

```

7607	035730	034562				\$TYPOC	::CALL=TYPOC	TRAP+2(104402)	TYPE OCTAL NUMBER WITH LEADING ZEROS
7608	035732	034536				\$TYPOS	::CALL=TYPOS	TRAP+3(104403)	TYPE OCTAL NUMBER (NO LEADING ZEROS)
7609	035734	034576				\$TYPON	::CALL=TYPON	TRAP+4(104404)	TYPE OCTAL NUMBER AS PER LAST CALL
7610									
7611	035736	035302				\$GTSWR	::CALL=GTSWR	TRAP+5(104405)	GET SOFT-SWR SETTING
7612									
7613	035740	035232				\$CKSWR	::CALL=CKSWR	TRAP+6(104406)	TEST FOR CHANGE IN SOFT-SWR
7614	035742	035514				\$RDCHR	::CALL=RDCHR	TRAP+7(104407)	TTY TYPEIN CHARACTER ROUTINE
7615	035744	034160				\$SAVREG	::CALL=SAVREG	TRAP+10(104410)	SAVE RD-RS ROUTINE
7616	035746	034216				\$RESREG	::CALL=RESREG	TRAP+11(104411)	RESTORE RD-RS ROUTINE
7617	035750	036710				.RSET	::CALL=RSETUP	TRAP+12(104412)	ROUTINE TO INITIALIZE AFTER EVERY TEST
7618	035752	036702				.LPER	::CALL=LPER	TRAP+13(104413)	ROUTINE TO SET LOOP ON ERROR ADDRESS
7619		00003C							

\$TERM=-.\$TRPAD

.SBTTL POWER DOWN AND UP ROUTINES

::*****
:POWER DOWN ROUTINE

7625	035754	012737	036132	000024		\$PWRDN:	MOV	#SILLUP, @#PWRVEC	::SET FOR FAST UP
7626	035762	012737	000340	000026			MOV	#340, @#PWRVEC+2	::PRIO:7
7627	035770	010046					MOV	R0, -(SP)	::PUSH R0 ON STACK
7628	035772	010146					MOV	R1, -(SP)	::PUSH R1 ON STACK
7629	035774	010246					MOV	R2, -(SP)	::PUSH R2 ON STACK
7630	035776	010346					MOV	R3, -(SP)	::PUSH R3 ON STACK
7631	036000	010446					MOV	R4, -(SP)	::PUSH R4 ON STACK
7632	036002	010546					MOV	R5, -(SP)	::PUSH R5 ON STACK
7633	036004	017746	143130				MOV	@SWR, -(SP)	::PUSH @SWR ON STACK
7634	036010	010667	000122				MOV	SP, \$SAVR6	::SAVE SP
7635	036014	012737	036026	000024			MOV	#PWRUP, @#PWRVEC	::SET UP VECTOR
7636	036022	000000					HALT		
7637	036024	000776					BR	.-2	::HANG UP

::*****
:POWER UP ROUTINE

7641	036026	012737	036132	000024		\$PWRUP:	MOV	#SILLUP, @#PWRVEC	::SET FOR FAST DOWN
7642	036034	016706	000076				MOV	\$SAVR6, SP	::GET SP
7643	036040	005067	000072				CLR	\$SAVR6	::WAIT LOOP FOR THE TTY
7644	036044	005267	000066			\$S:	INC	\$SAVR6	::WAIT FOR THE INC
7645	036050	001375					BNE	\$S	::OF WORD
7646	036052	012677	143062				MOV	(SP)+, @SWR	::POP STACK INTO @SWR
7647	036056	012605					MOV	(SP)+, R5	::POP STACK INTO R5
7648	036060	012604					MOV	(SP)+, R4	::POP STACK INTO R4
7649	036062	012603					MOV	(SP)+, R3	::POP STACK INTO R3
7650	036064	012602					MOV	(SP)+, R2	::POP STACK INTO R2
7651	036066	012601					MOV	(SP)+, R1	::POP STACK INTO R1
7652	036070	012600					MOV	(SP)+, R0	::POP STACK INTO R0
7653	036072	012737	035754	000024			MOV	#PWRDN, @#PWRVEC	::SET UP THE POWER DOWN VECTOR
7654	036100	012737	000340	000026			MOV	#340, @#PWRVEC+2	::PRIO:7
7655	036106	104401					TYPE		::REPORT THE POWER FAILURE
7656	036110	037104				\$PWRMG:	.WORD	POWERM	::POWER FAIL MESSAGE POINTER
7657	036112	012716					MOV	(PC)+, (SP)	::RESTART AT START
7658	036114	004336				\$PWRAD:	.WORD	START	::RESTART ADDRESS
7659	036116	042766	000020	000002			BIC	#20, 2(SP)	::CLEAR "T" BIT
7660	036124	005067	175304				CLR	\$TBIT	::CLEAR THE "T" BIT FLAG
7661	036130	000002					RTI		
7662	036132	000000				\$SILLUP:	HALT		::THE POWER UP SEQUENCE WAS STARTED

7663 036134 000776
 7664 036136 000000
 7665
 7666
 7667
 7668
 7669
 7670
 7671
 7672
 7673
 7674
 7675
 7676 036140 104401
 7677 036142 001313
 7679 036144 113737 001102 001232
 7679 036152 042737 177400 001232
 7680 036160 013737 001116 001234
 7681 036166 010046
 7682
 7683 036170 113700 001114
 7684 036174 042700 177400
 7685 036200 001005
 7686
 7687 036202 013746 001116
 7688 036206 104402
 7689 036210 000137 036610
 7690
 7691 036214 022700 000377
 7692 036220 001005
 7693 036222 016600 000004
 7694 036226 011000
 7695 036230 062700 000400
 7696 036234 005300
 7697 036236 006300
 7698 036240 006300
 7699 036242 006300
 7700 036244 062700 001442
 7701
 7702 036250 012037 036260
 7703 036254 001404
 7704 036256 104401
 7705 036260 000000
 7706 036262 104401
 7707 036264 001313
 7708
 7709 036266 012037 036276
 7710 036272 001404
 7711 036274 104401
 7712 036276 000000
 7713 036300 104401
 7714 036302 001313
 7715
 7716 036304 010146
 7717 036306 010246
 7718 036310 010346

BR .-2 :: BEFORE THE POWER DOWN WAS COMPLETE
 \$SAVR6: 0 :: PUT THE SP HERE

.SBTTL ERROR TYPE OUT ROUTINE

```

:*****
:*****
: THIS ROUTINE IS CALLED TO TYPE AN ERROR MESSAGE WHICH IS INCLUDED
: IN THE ERROR MESSAGE DATA TABLE. IT IS CALLED BY THE $ERROR ROUTINE
: OR BY FIRST SETTING $ITEMB EQUAL TO THE ERROR TABLE ITEM TO BE PRINTED
: OUT AND THEN EXECUTING A:
:
: JSR PC,ERTYPE
:
: ERTYPE: TYPE ;TYPE A CRLF
: .WORD $CRLF
: MOVB @($STNM,@$STMP0
: BIC @177400,@$STMP0
: MOV @$ERRPC,@$STMP1 ;GET PC OF CALL
: MOV RO,-(SP) ;SAVE RO
:
: MOVB @$ITEMB,RO ;GET THE ITEM NUMBER.
: BIC @177400,RO
: BNE IS
:
: MOV @$ERRPC,-(SP) ;IF ZERO THEN JUST
: TYPOC ;PRINT THE PC
: JMP @ERTS
:
: IS: CMP @377,RO
: BNE 20$
: MOV 4(SP),RO
: MOV (RO),RO
: ADD @400,RO
: 20$: DEC RO ;OTHERWISE MAKE RC AN
: ASL RO ;INDEX FOR THE TABLE.
: ASL RO
: ASL RO
: ADD @$ERRTB,RO
:
: MOV (RO)+,@25 ;PICK UP THE ADDRESS
: BEQ 3$ ;OF THE EM, ERROR MESSAGE
: TYPE
: 25: .WORD 0
: TYPE
: .WORD $CRLF
:
: 35: MOV (RO)+,@45 ;GET THE DH,DATA HEADER
: BEQ 5$
: TYPE
: 45: .WORD 0
: TYPE
: .WORD $CRLF
:
: 55: MOV R1,-(SP) ;SAVE R1,R2 AND R3
: MOV R2,-(SP)
: MOV R3,-(SP)

```

```

7719
7720 036312 012001          MOV      (R0)+,R1          ;GET THE ADDRESS OF THE
7721                                ;DATA TABLE.
7722 036314 001002          BNE      6$
7723 036316 000137 036576    JMP      @ERT4          ;RETURN IF NO DATA.
7724
7725 036322 011000          6$:      MOV      (R0),R0          ;GET A POINTER TO THE DATA
7726                                ;FORMAT TABLE.
7727 036324 105710          ERT1:    TSTB     (R0)          ;FORMAT ZERO?
7728 036326 001004          BNE      7$
7729
7730 036330 013146          MOV      @ (R1)+, -(SP)      ;FORMAT ZERO SO TYPE
7731 036332 104402          TYPOC   ;AN OCTAL NUMBER.
7732 036334 000137 036560    JMP      @ERT2
7733
7734 036340
7735 036340 122710 000002    7$:      CMPB     #2, (R0)          ;FORMAT TWO?
7736 036344 001011          BNE      8$
7737
7738 036346 013102          MOV      @ (R1)+, R2          ;FORMAT TWO SO TYPE TWO
7739 036350 012246          MOV      (R2)+, -(SP)      ;OCTAL NUMBERS.
7740 036352 104402          TYPOC
7741 036354 104401          TYPE
7742 036356 037153          .WORD   SPACE
7743 036360 011246          MOV      (R2), -(SP)
7744 036362 104402          TYPOC
7745 036364 000137 036560    JMP      @ERT2
7746
7747 036370 122710 000003    8$:      CMPB     #3, (R0)          ;FORMAT THREE?
7748 036374 001021          BNE      9$
7749
7750 036376 013102          MOV      @ (R1)+, R2          ;FORMAT THREE SO TYPE
7751 036400 012246          MOV      (R2)+, -(SP)      ;FOUR OCTAL NUMBERS.
7752 036402 104402          TYPOC
7753 036404 104401          TYPE
7754 036406 037153          .WORD   SPACE
7755 036410 012246          MOV      (R2)+, -(SP)
7756 036412 104402          TYPOC
7757 036414 104401          TYPE
7758 036416 037153          .WORD   SPACE
7759 036420 012246          MOV      (R2)+, -(SP)
7760 036422 104402          TYPOC
7761 036424 104401          TYPE
7762 036426 037153          .WORD   SPACE
7763 036430 011246          MOV      (R2), -(SP)
7764 036432 104402          TYPOC
7765 036434 000137 036560    JMP      @ERT2
7766
7767 036440 122710 000004    9$:      CMPB     #4, (R0)          ;FORMAT FOUR?
7768 036444 001005          BNE      10$
7769
7770 036446 013146          MOV      @ (R1)+, -(SP)      ;FORMAR FOUR SO TYPE
7771 036450 104403          TYPOS   ;AN OCTAL NUMBER
7772 036452 016             .BYTE   16                ;SUPPRESSING LEADING ZEROES.
7773 036453 000             .BYTE   0
7774 036454 000137 036560    JMP      @ERT2

```

```

7775
7776 036460 122710 000005      11$:  CMPB    #5,(R0)           ;FORMAT FIVE?
7777 036464 001006                BNE    13$
7778
7779 036466 012137 036474                MOV    (R1)+,2#12$          ;FORMAT FIVE SO TYPE AN
7780 036472 104401                TYPE   ;ASCIZ STRING.
7781 036474 000000      12$:  .WORD    0
7782 036476 000137 036564                JMP    2#ERT3
7783
7784 036502 122710 000011      13$:  CMPB    #11,(R0)         ;FORMAT ELEVEN?
7785 036506 001006                BNE    15$
7786
7787 036510 013137 036516                MOV    2(R1)+,2#14$        ;FORMAT ELEVEN SO PICK
7788 036514 104401                TYPE   ;A POINTER TO AN ASCIZ
7789 036516 000000      14$:  .WORD    0                ;STRING.
7790 036520 000137 036564                JMP    2#ERT3
7791
7792 036524 122710 000012      15$:  CMPB    #12,(R0)         ;FORMAT TWELVE?
7793 036530 001012                BNE    17$
7794
7795 036532 013102                MOV    2(R1)+,R2           ;FORMAT TWELVE SO TYPE
7796 036534 012703 000006      16$:  MOV     #6,R3             ;TYPE SIX OCTAL NUMBERS
7797 036540 012246                MOV    (R2)+,-(SP)
7798 036542 104402                TYPOC
7799 036544 104401                TYPE
7800 036546 037153                .WORD  SPACE
7801 036550 077305                SOB   R3,16$
7802 036552 000137 036560                JMP    2#ERT2
7803
7804 036556 000000      17$:  HALT                    ;UNDEFINED FORMAT FOR DATA????
7805
7806 036560 104401      ERT2:  TYPE                    ;PRINT A TAB AFTER TYPING
7807 036562 037151                .WORD  $TAB                ;AN DATA TABLE ENTRY
7808                                ;OF ALL FORMATS EXCEPT
7809                                ;ASCIZ, FORMATS 5 OR 1!
7810
7811 036564 005200      EPT3:  INC     R0                ;POINT TO THE NEXT FORMAT
7812 036566 005711                TST   (R1)                 ;END OF DATA TABLE.
7813 036570 001402                BEQ   ERT4
7814 036572 000137 036324                JMP    2#ERT1
7815
7816 036576 104401      ERT4:  TYPE                    ;DONE.
7817 036600 001313                .WORD  $CRLF
7818 036602 012603                MOV   (SP)+,R3             ;RESTORE R1,R2 AND R3
7819 036604 012602                MOV   (SP)+,R2
7820 036606 012601                MOV   (SP)+,R1
7821 036610 012600      ERT5:  MOV   (SP)+,R0           ;RESTORE R0.
7822 036612 000207                RTS    PC                  ;AND RETURN.
7823
7824
7825
7826
7827
7828
7829
7830

```

.SBTTL FPP SPURIOUS TRAP TO 244 HANDLER

```

;*****
;*****
;THIS ROUTINE HANDLES UNEXPECTED TRAPS TO THE FPP TRAP VECTOR AT 244.

```

```

7831      ;*THE LAST FPP INSTRUCTION EXECUTED AND ITS ADDRESS HAS BEEN RECORDED
7832      ;*THESE ALONG WITH THE FEC, FPS AND PC OF TRAP ARE REPORTED.
7833      ;*
7834 036614 011637 001236  FPPSPUR: MOV      (SP), 2#STMP2      ;SAVE PC OF TRAP.
7835 036620 022626          CMP      (SP)+, (SP)+      ;RESTORE SP.
7836 036622 170200          STFPS   RO              ;GET FPS
7837 036624 010037 001240  MOV      RO, 2#STMP3
7838 036630 170300          STST   RO              ;GET FEC
7839 036632 010037 001242  MOV      RO, 2#STMP4
7840 036636 104247          IS:    ERROR 247
7841 036640 104412          RSETUP
7842      ;GO INITIALIZE THE FPS AND STACK; AND
7843      ;SEE IF THE USER HAS EXPRESSED
7844      ;THE DESIRE TO CHANGE THE SOFTWARE
7845      ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
7846      ;THE USER TYPED CONTROL G?).
7846 036642 000137 033150  JMP      2#SEOP
7847
7848      .SBTTL CPU SPURIOUS TRAP TO 4 HANDLER
7849      ;*****
7850      ;*****
7851      ;*****
7852      ;*THIS ROUTINE REPORTS UNEXPECTED CPU TRAPS TO VECTOR 4.
7853      ;*
7854 036646 011637 001236  CPSPUR: MOV      (SP), 2#STMP2      ;SAVE PC OF TRAP.
7855 036652 022626          CMP      (SP)+, (SP)+
7856 036654 104250          IS:    ERROR 250
7857 036656 104412          RSETUP
7858      ;GO INITIALIZE THE FPS AND STACK; AND
7859      ;SEE IF THE USER HAS EXPRESSED
7860      ;THE DESIRE TO CHANGE THE SOFTWARE
7861      ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
7862      ;THE USER TYPED CONTROL G?).
7862 036660 000137 033150  JMP      2#SECP
7863
7864      .SBTTL CPU SPURIOUS TRAP TO 10 HANDLER
7865      ;*****
7866      ;*****
7867      ;*****
7868      ;*THIS ROUTINE REPORTS UNEXPECTED CPU TRAPS TO VECTOR 10.
7869      ;*
7870 036664 011637 001236  CPTWO: MOV      (SP), 2#STMP2      ;SAVE PC OF TRAP.
7871 036670 022626          CMP      (SP)+, (SP)+
7872 036672 104251          IS:    ERROR 251
7873 036674 104412          RSETUP
7874      ;GO INITIALIZE THE FPS AND STACK; AND
7875      ;SEE IF THE USER HAS EXPRESSED
7876      ;THE DESIRE TO CHANGE THE SOFTWARE
7877      ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
7878      ;THE USER TYPED CONTROL G?).
7878 036676 000137 033150  JMP      2#SEOP
7879
7880
7881
7882
7883
7884      .SBTTL SET LOOP ON ERROR ADDRESS ROUTINE
7885      ;*****
7886      ;*****

```

7887
7888 036702 011637 001110
7889 036706 000002
7890
7891
7892
7893
7894
7895
7896
7897
7898
7899
7900
7901 036710 023727 001140 177570
7902
7903 036716 001001
7904 036720 104406
7905
7906
7907
7908 036722 012737 036614 000244
7909 036730 012737 036646 000004
7910 036736 012737 036664 000010
7911 036744 011600
7912 036746 012706 001100
7913 036752 005004
7914 036754 170104
7915 036756 000110
7916
7917
7918

```
;*
.LPER: MOV (SP),@#SLPERR
RTI

.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.
;*
.RSET: CMP @#SWR,#177570 ;SEE IF THERE IS A PHYSICAL
;CONSOLE SWITCH REGISTER.
BNE IS ;BRANCH IF NO.
CKSWR ;OTHERWISE TYPE THE CONTENTS
;OF THE PROGRAM VIRTUAL SWITCH REGISTER
;AND GIVE THE USER A CHANCE TO
;MODIFY IT.
IS: MOV #FPSPUR,@#FPVCT
MOV #CPSPUR,@#ERRVCT
MOV #CPTWO,@#I0
MOV (SP),R0 ;SAVE RETURN ADDRESS.
MOV #STACK,SP ;RESET THE STACK POINTER.
CLR R4 ;CLEAR THE FPS.
LDFPS R4
JMP (R0) ;RETURN.
```

.NLIST BFX

;THESE ARE SPECIAL MESSAGES:

```
036760 051124 050101 042520 MSA1: .ASCIZ 'TRAPPED AT:'<TAB><TAB>
036776 054105 042520 052103 MSA2: .ASCIZ 'EXPECTED TRAP AT:'<TAB>
037021 107 052117 051040 MSA3: .ASCIZ 'GOT RO:'<TAB><TAB>
037033 105 050130 041505 MSA4: .ASCIZ 'EXPECTED RO:'<TAB>
037051 107 052117 040440 MSA5: .ASCIZ 'GOT ACO:'<TAB><TAB>
037064 054105 042520 052103 MSA6: .ASCIZ 'EXPECTED ACO:'<TAB><TAB>

037104 050200 053517 051105 POWERM: .ASCIZ <CRLF>'POWER FAILURE. PROGRAM RESTARTING.'<CRLF>
037151 011 000 $TAB: .ASCIZ <TAB>
037153 040 000040 SPACE: .ASCIZ ' '
037156 041501 047440 042520 MS1: .ASCIZ 'AC OPERAND:'<TAB><TAB>
037174 051506 041522 047440 MS2: .ASCIZ 'FSRC OPERAND:'<TAB><TAB>
037214 041501 020060 042502 MS3: .ASCIZ 'ACO BEFORE EXECUTION:'<TAB>
037243 101 030103 040440 MS4: .ASCIZ 'ACO AFTER EXECUTION:'<TAB>
037271 105 050130 041505 MS5: .ASCIZ 'EXPECTED RESULT:'<TAB>
037313 107 052117 051040 MS6: .ASCIZ 'GOT RESULT:'<TAB><TAB>
037331 106 040522 052103 MS7: .ASCIZ 'FRACTIONAL RESULT:'<TAB>
037355 111 052116 043505 MS10: .ASCIZ 'INTEGER RESULT:'<TAB>
037377 105 050130 041505 MS11: .ASCIZ 'EXPECTED FRACTION:'<TAB>
037423 105 050130 041505 MS12: .ASCIZ 'EXPECTED INTEGER:'<TAB>
```

```

03762: 00 0507 020124 042522 MS37: .ASCIZ 'LOADED DATA: '
03763: 00 0507 020124 042522 MS40: .ASCIZ 'READ DATA: '
03764: 00 0507 020124 042522 MS415: .ASCIZ 'EXPECTED DATA: '
03765: 00 0507 020124 042522 MS41: .ASCIZ 'DATA IN (R) FSRC: '
03766: 00 0507 020124 042522 MS42: .ASCIZ 'DATA IN ACC: '
03767: 00 0507 020124 042522 MS43: .ASCIZ 'GOT RESULT: '
03768: 00 0507 020124 042522 MS44: .ASCIZ 'EXPECTED RESULT: '
03769: 00 0507 020124 042522 COMES: .ASCIZ '<CR LF> TEST 22, TESTING INTERRUPTS. <CR LF>'

```

THESE ARE ERROR MESSAGES:

```

03766C 050106 020123 040502 EM1: .ASCIZ 'FPS BAD AFTER CMPD (R),A.'
0376712 041501 02006C 047515 EM2: .ASCIZ 'ACD MODIFIED BY CMPD (R),A.'
0376745 0376745 0376745 0376745 EM3:
037746 050106 020123 040502 .ASCII 'FPS BAD AFTER CMPD. <CR LF>'
037772 041050 052125 04244C .ASCIZ '(BUT ENBT) STATE 225 WENT TO 475 INSTEAD OF 075.'
040053 040053 106 051520 041040 EM4: .ASCII 'FPS BAD AFTER CMPD. <CR LF>'
040077 050 052502 020124 .ASCIZ '(BUT ENBT) STATE 225 WENT TO 075 INSTEAD OF 475.'
040160 040160 050106 020123 040502 EM5: .ASCII 'FPS BAD AFTER CMPD. <CR LF>'
040204 041050 052125 042440 .ASCIZ '(BUT ENBT) STATE 035 WENT TO 075 INSTEAD OF 475.'
040265 040265 106 051520 041040 EM6: .ASCII 'FPS BAD AFTER CMPD. <CR LF>'
040311 050 052502 020124 .ASCIZ '(BUT ENBT) STATE 035 WENT TO 475 INSTEAD OF 075.'
040372 040372 050106 020123 040502 EM7: .ASCII 'FPS BAD AFTER CMPD. <CR LF>'
040416 041050 052125 042440 .ASCIZ '(BUT ENBT Y8) STATE 777 SHOULD HAVE GONE TO 007.'
040477 040477 106 051520 041040 EM10: .ASCII 'FPS BAD AFTER CMPD. <CR LF>'
040523 050 052502 020124 .ASCIZ '(BUT ENBT Y8) STATE 777 SHOULD HAVE GONE TO 405.'
040604 040604 050106 020123 040502 EM11: .ASCII 'FPS BAD AFTER CMPD. <CR LF>'
040630 041050 052125 047040 .ASCIZ '(BUT NBIT ZBIT) STATE 456 SHOULD HAVE GONE TO 010.'
040713 040713 106 051520 041040 EM12: .ASCII 'FPS BAD AFTER CMPD. <CR LF>'
040737 050 052502 020124 .ASCIZ '(BUT NBIT ZBIT) STATE 456 SHOULD HAVE GONE TO 110.'
041022 041022 050106 020123 040502 EM13: .ASCII 'FPS BAD AFTER CMPD. <CR LF>'
041046 044504 047104 052047 .ASCIZ '/DIDN'T TAKE THE PATH: STATE 456, TO 012. TO 363 TO 120.'
041136 041136 050106 020123 040502 EM14: .ASCII 'FPS BAD AFTER CMPD. <CR LF>'
041162 041050 052125 054040 .ASCIZ '(BUT XNBT XZBT) STATE 363 WENT TO 140 INSTEAD OF 100.'
041250 041250 050106 020123 040502 EM15: .ASCII 'FPS BAD AFTER CMPD. <CR LF>'
041274 041050 052125 054040 .ASCIZ '(BUT XNBT XZBT) STATE 363 WENT TO 100 INSTEAD OF 140.'
041362 041362 050106 020123 040502 EM16: .ASCII 'FPS BAD AFTER CMPD. <CR LF>'
041406 044504 047104 052047 .ASCIZ '/DIDN'T TAKE THE PATH: STATE 777, TO 407./'
041457 104 053111 020104 EM17: .ASCIZ 'DIVD (R) A TRAPPED TO 244. FSRC=0 AND FID=1.'
041534 050106 020123 040502 EM20: .ASCIZ 'FPS BAD AFTER DIVD (R),A.'
041566 042506 020103 040502 EM21: .ASCIZ 'FEC BAD AFTER DIVD (R),A.'
041620 044504 042126 024040 EM22: .ASCIZ '/DIVD (R),A DIDN'T TRAP TO 244. FSRC=0 AND FID=0..'
041701 104 053111 020106 EM23: .ASCIZ 'DIVF (R),A FAILED.'
041724 050106 020123 040502 EM32: .ASCIZ 'FPS BAD AFTER DIVF (R),A.'
041756 041756 041756 041756 EM24:

```


0)	041756	044504	043126	024040	.ASCII	'DIVF (R),A FAILED.'
0)	042000	041050	052125	054440	.ASCIZ	'(BUT Y61) WENT TO STATE 006 INSTEAD OF 206.'
0)	042054				EM25:	
0)	042054	044504	043126	024040	.ASCII	'DIVF (R),A FAILED.'
0)	042076	047530	020122	043117	.ASCIZ	'XOR OF SIGN BITS FAILED STATE 470.'
0)	042141				EM26:	
0)	042141	104	053111	020106	.ASCII	'DIVF (R),A FAILED.'
0)	042163	050	052502	020124	.ASCIZ	'(BUT Y61) WENT TO STATE 206 INSTEAD OF 306.'
0)	042054				EM27=EM25	
0)	042237				EM30:	
0)	042237	104	053111	020106	.ASCII	'DIVF (R),A FAILED.'
0)	042261	124	052522	041516	.ASCIZ	'TRUNCATION ERROR. FT=1.'
0)	042311				EM31:	
0)	042311	104	053111	020106	.ASCII	'DIVF (R),A FAILED.'
0)	042333	122	052517	042116	.ASCIZ	'ROUND ERROR. FT=0.'
0)	042356	044504	042126	024040	EM33:	'DIVD (R),A FAILED.'
0)	042401	106	051520	041040	EM34:	'FPS BAD AFTER DIVD (R),A.'
0)	042433				EM35:	
0)	042433	104	053111	020104	.ASCII	'DIVD (R),A FAILED.<CRLF>
0)	042456	051124	047125	040503	.ASCIZ	'TRUNCATION ERROR. FT=1.'
0)	042506				EM36:	
0)	042506	044504	042126	024040	.ASCII	'DIVD (R),A FAILED.<CRLF>
0)	042531	122	052517	042116	.ASCIZ	'ROUND ERROR. FT=0.'
0)	042554	052515	043114	024040	EM37:	'MULF (R),A FAILED.'
0)	042577	106	051520	041040	EM40:	'FPS BAD AFTER MULF (R),A.'
0)	042631				EM41:	
0)	042631	115	046125	020106	.ASCII	'MULF (R),A FAILED.<CRLF>
0)	042654	044523	047107	041040	.ASCIZ	'SIGN BIT BAD STATE 511.'
0)	042704				EM42:	
0)	042704	052515	043114	024040	.ASCII	'MULF (R),A FAILED.<CRLF>
0)	042727	116	051117	040515	.ASCII	'NORMALIZATION FAILED.<CRLF>
0)	042755	050	052502	020124	.ASCIZ	'(BUT Y62) STATE 252 WENT TO 044 INSTEAD OF 444.'
0)	043035				EM43:	
0)	043035	115	046125	020106	.ASCII	'MULF (R),A FAILED.<CRLF>
0)	043060	047516	046522	046101	.ASCII	'NORMALIZATION FAILED.<CRLF>
0)	043106	041050	052125	054440	.ASCIZ	'(BUT Y62) STATE 252 WENT TO 444 INSTEAD OF 044.'
0)	043166				EM44:	
0)	043166	052515	043114	024040	.ASCII	'MULF (R),A FAILED.<CRLF>
0)	043211	122	052517	042116	.ASCIZ	'ROUND ERROR. FT=0.'
0)	043234				EM45:	
0)	043234	052515	043114	024040	.ASCII	'MULF (R),A FAILED.<CRLF>
0)	043257	124	052522	041516	.ASCIZ	'TRUNCATION ERROR. FT=1.'
0)	043307	106	051520	041040	EM46:	'FPS BAD AFTER MULD (R),A.'
0)	043341	115	046125	020104	EM246:	'MULD (R),A FAILED.'
0)	043364				EM47:	
0)	043364	052515	042114	024040	.ASCII	'MULD (R),A FAILED.<CRLF>
0)	043407	102	042101	041440	.ASCII	'BAD CONSTANT USED IN THE MUL ALGORITHM.'
0)	043456	052600	042523	020104	.ASCIZ	<CRLF>'USED 24 INSTEAD OF 56 STATE 020.'
0)	043520				EM50:	
0)	043520	052515	042114	024040	.ASCII	'MULD (R),A FAILED.<CRLF>
0)	043543	124	052522	041516	.ASCIZ	'TRUNCATION ERROR. FT=1.'
0)	043573				EM51:	
0)	043573	115	046125	020104	.ASCII	'MULD (R),A FAILED.<CRLF>
0)	043616	047522	047125	020104	.ASCIZ	'ROUND ERROR. FT=0.'
0)	043641				EM52:	
0)	043641	115	046125	020104	.ASCII	'MULD (R),A FAILED.<CRLF>

	043664	040502	020104	047503		.ASCIZ	'BAD CONSTANT USED IN ROUNDING, FI=0.'
	043731	106	051520	041040	EM111:	.ASCIZ	'FPS BAD AFTER MULF (R),A. EXPECTED OVERFLOW.'
	044006	050106	020123	040502	EM112:	.ASCIZ	'FPS BAD AFTER MULF (R),A. EXPECTED UNDERFLOW.'
	044064				EM113:		
	044064	052515	043114	024040		.ASCII	'MULF (R),A FAILED.' <crlf></crlf>
	044107	105	050130	041505		.ASCIZ	'EXPECTING OVERFLOW, FIV=0.'
	044142				EM114:		
	044142	052515	043114	024040		.ASCII	'MULF (R),A FAILED.' <crlf></crlf>
	044165	105	050130	041505		.ASCIZ	'EXPECTING UNDERFLOW, FIU=0.'
	044221	115	046125	020106	EM115:	.ASCIZ	'MULF (R),A TRAPPED TO 244 ON OVERFLOW. FIV=0.'
	044277	115	046125	020106	EM116:	.ASCIZ	'MULF (R),A TRAPPED TO 244 ON UNDERFLOW. FIU=0.'
	044356				EM117:		
	044356	052515	043114	024040		.ASCII	'MULF (R),A TRAPPED TO 244 ON UNDERFLOW. FIU=0.'
	044434	041050	052125	043040		.ASCIZ	'(BUT FIU) STATE 301 WENT TO 155 INSTEAD OF 115.'
	044514				EM120:		
	044514	052515	043114	024040		.ASCII	'MULF (R),A TRAPPED TO 244 ON UNDERFLOW. FIU=0.'
	044572	041050	052125	043040		.ASCIZ	'(BUT FIU) STATE 137 WENT TO 155 INSTEAD OF 115.'
	044652				EM121:		
	044652	052515	043114	024040		.ASCII	'MULF (R),A TRAPPED TO 244 ON OVERFLOW. FIV=0.'
	044727	050	052502	020124		.ASCIZ	'(BUT FIV) STATE 333 WENT TO 136 INSTEAD OF 116.'
	045007				EM122:		
	045007	115	046125	020106		.ASCII	'MULF (R),A TRAPPED TO 244 ON OVERFLOW. FIV=0.'
	045064	041050	052125	043040		.ASCIZ	'(BUT FIV) STATE 133 WENT TO 136 INSTEAD OF 116.'
	045144	050106	020123	040502	EM123:	.ASCIZ	'FPS BAD AFTER MULF (R),A. EXPECTING OVERFLOW.'
	045222	050106	020123	040502	EM124:	.ASCIZ	'FPS BAD AFTER MULF (R),A. EXPECTING UNDERFLOW.'
	045301				EM125:		
	045301	115	046125	020104		.ASCII	'MULD (R),A FAILED.' <crlf></crlf>
	045324	054105	042520	052103		.ASCIZ	'EXPECTING OVERFLOW, FIV=0.'
	045357				EM126:		
	045357	115	046125	020104		.ASCII	'MULD (R),A FAILED.' <crlf></crlf>
	045402	054105	042520	052103		.ASCIZ	'EXPECTING UNDERFLOW, FIU=0.'
	045436	052515	042114	024040	EM127:	.ASCIZ	'MULD (R),A TRAPPED TO 244 ON OVERFLOW. FIV=0.'
	045514	052515	042114	024040	EM130:	.ASCIZ	'MULD (R),A TRAPPED TO 244 ON UNDERFLOW. FIU=0.'
	045573				EM131:		
	045573	115	046125	020106		.ASCII	'MULF (R),A TRAPPED TO 244 ON UNDERFLOW. FIU=0.'
	045651	050	052502	020124		.ASCIZ	'(BUT FIU) STATE 331 WENT TO 155 INSTEAD OF 115.'
	045731				EM132:		
	045731	115	046125	020104		.ASCII	'MULD (R),A FAILED.' <crlf></crlf>
	045754	054105	042520	052103		.ASCII	'EXPECTING UNDERFLOW, FIU=0.'
	046007	200	041050	052125		.ASCIZ	<CRLF>'(BUT FD) STATE 115 WENT TO 424 INSTEAD OF 425.'
	046067				EM133:		
	046067	115	046125	020106		.ASCII	'MULF (R),A TRAPPED TO 244 ON UNDERFLOW. FIU=0.'
	046145	050	052502	020124		.ASCIZ	'(BUT FIU) STATE 137 WENT TO 155 INSTEAD OF 115.'
	046225				EM134:		
	046225	115	046125	020104		.ASCII	'MULD (R),A TRAPPED TO 244 ON OVERFLOW. FIV=0.'
	046302	041050	052125	043040		.ASCIZ	'(BUT FIV) STATE 333 WENT TO 136 INSTEAD OF 116.'
	046362				EM135:		
	046362	052515	042114	024040		.ASCII	'MULD (R),A FAILED.' <crlf></crlf>
	046405	105	050130	041505		.ASCII	'EXPECTING OVERFLOW, FIV=0.'
	046437	200	041050	052125		.ASCIZ	<CRLF>'(BUT FD) STATE 116 WENT TO 424 INSTEAD OF 425.'
	046517				EM136:		
	046517	115	046125	020104		.ASCII	'MULD (R),A TRAPPED TO 244 ON OVERFLOW. FIV=0.'
	046574	041050	052125	043040		.ASCIZ	'(BUT FIV) STATE 133 WENT TO 136 INSTEAD OF 116.'
	046654	042506	020103	040502	EM137:	.ASCIZ	'FEC BAD AFTER MULF (R),A. EXPECTING OVERFLOW, FEC=10.'
	046742	042506	020103	040502	EM140:	.ASCIZ	'FEC BAD AFTER MULF (R),A. EXPECTING UNDERFLOW, FEC=12.'
	043731				EM141=EM111		

	044006			EM142=EM112	
(0)	047031			EM143:	
(1)	047031	115	046125	020106	.ASCII 'MULF (R),A FAILED.' <crlf>< td=""> </crlf><>
	047054	054105	042520	052103	.ASCIZ 'EXPECTING OVERFLOW, FIV=1.'
(0)	047107			EM144:	
(1)	047107	115	046125	020106	.ASCII 'MULF (R),A FAILED.' <crlf>< td=""> </crlf><>
	047132	054105	042520	052103	.ASCIZ 'EXPECTING UNDERFLOW, FIU=1.'
(0)	047166			EM145:	
(1)	047166	052515	043114	024040	.ASCII 'MULF (R),A FAILED TO TRAP TO 244 ON UNDERFLOW, FIV=1.'
	047253	050	052502	020124	.ASCIZ '(BUT FIU) STATE 331 WENT TO 115 INSTEAD OF 155.'
(0)	047333			EM146:	
(1)	047333	115	046125	020106	.ASCII 'MULF (R),A FAILED TO TRAP TO 244 ON UNDERFLOW, FIU=1.'
	047420	041050	052125	043040	.ASCIZ '(BUT FIU) STATE 137 WENT TO 115 INSTEAD OF 155.'
(0)	047500			EM147:	
(1)	047500	052515	043114	024040	.ASCII 'MULF (R),A FAILED TO TRAP TO 244 ON OVERFLOW, FIV=1.'
	047564	041050	052125	043040	.ASCIZ '(BUT FIV) STATE 333 WENT TO 116 INSTEAD OF 136.'
(0)	047644			EM150:	
(1)	047644	052515	043114	024040	.ASCII 'MULF (R),A FAILED TO TRAP TO 244 ON OVERFLOW, FIV=1.'
	047730	041050	052125	043040	.ASCIZ '(BUT FIV) STATE 133 WENT TO 116 INSTEAD OF 136.'
	050010	042506	020103	040502	EM151: .ASCII 'FEC BAD AFTER MULF (R),A. EXPECTING OVERFLOW, FEC=10.'
	050076	042506	020103	040502	EM152: .ASCII 'FEC BAD AFTER MULF (R),A. EXPECTING UNDERFLOW, FEC=12.'
		045144			EM153=EM123
		045222			EM154=EM124
(0)	050165			EM155:	
(1)	050165	115	046125	020104	.ASCII 'MULD (R),A FAILED.' <crlf>< td=""> </crlf><>
	050210	054105	042520	052103	.ASCIZ 'EXPECTING OVERFLOW, FIV=1.'
(0)	050243			EM156:	
(1)	050243	115	046125	020104	.ASCII 'MULD (R),A FAILED.' <crlf>< td=""> </crlf><>
	050266	054105	042520	052103	.ASCIZ 'EXPECTING UNDERFLOW, FIU=1.'
(0)	050322			EM157:	
(1)	050322	052515	042114	024040	.ASCII 'MULD (R),A FAILED TO TRAP TO 244 ON UNDERFLOW, FIU=1.'
	050407	050	052502	020124	.ASCIZ '(BUT FIU) STATE 331 WENT TO 115 INSTEAD OF 155.'
(0)	050467			EM160:	
(1)	050467	115	046125	020104	.ASCII 'MULD (R),A FAILED.' <crlf>< td=""> </crlf><>
	050512	054105	042520	052103	.ASCII 'EXPECTING UNDERFLOW, FIU=1.'
	050545	200	041050	052125	.ASCIZ '<CRLF>'(BUT FD) STATE 155 WENT TO 426 INSTEAD OF 427.'
(0)	050625			EM161:	
(1)	050625	115	046125	020104	.ASCII 'MULD (R),A FAILED TO TRAP TO 244 ON UNDERFLOW, FIU=1.'
	050712	041050	052125	043040	.ASCIZ '(BUT FIU) STATE 137 WENT TO 115 INSTEAD OF 155.'
(0)	050772			EM162:	
(1)	050772	052515	042114	024040	.ASCII 'MULD (R),A FAILED TO TRAP TO 244 ON OVERFLOW, FIV=1.'
	051055	041050	052125	043040	.ASCIZ '(BUT FIV) STATE 333 WENT TO 116 INSTEAD OF 136.'
(0)	051136			EM163:	
(1)	051136	052515	042114	024040	.ASCII 'MULD (R),A FAILED.' <crlf>< td=""> </crlf><>
	051161	105	050130	041505	.ASCII 'EXPECTING OVERFLOW, FIV=1.'
	051213	200	041050	052125	.ASCIZ '<CRLF>'(BUT FD) STATE 700 WENT TO 426 INSTEAD OF 427.'
(0)	051273			EM164:	
(1)	051273	115	046125	020104	.ASCII 'MULD (R),A FAILED TO TRAP TO 244 ON OVERFLOW, FIV=1.'
	051357	050	052502	020124	.ASCIZ '(BUT FIV) STATE 133 WENT TO 116 INSTEAD OF 136.'
	051437	106	051520	041040	EM55: .ASCIZ 'FPS BAD AFTER MOOF (R),A.'
	051471	115	042117	020106	EM53: .ASCII 'MOOF (R),A FRACTION BAD.'
	051522	047515	043104	024040	EM54: .ASCIZ 'MOOF (R),A INTEGER BAD.'
(0)	051552			EM56:	
(1)	051552	047515	043104	024040	.ASCII 'MOOF (R),A FRACTION BAD.' <crlf>< td=""> </crlf><>
	051603	101	030103	042040	.ASCIZ 'ACO DID NOT GET 0 IN STATE 424.'
(0)	051643			EM57:	

01	051643	115	042117	020106	.ASCII	'MODF (R), A INTEGER BAD.'
	051673	101	030503	042040	.ASCIZ	'AC1 DID NOT GET 0 IN STATE 142.'
02	051733				EM69:	
03	051733	115	042117	020106	.ASCII	'MODF (R), A INTEGER BAD.'
	051763	101	030503	042040	.ASCIZ	'AC1 DID NOT GET THE INTEGER IN STATE 134.'
04	052035				EM61:	
	052035	115	042117	020106	.ASCII	'MODF (R), A FRACTION BAD.'
	052066	020101	040502	020104	.ASCII	'A BAD CONSTANT WAS USED (NOT 24) IN STATE 046.'
	052144	047600	020122	041050	.ASCIZ	<CRLF>'OR (BUT NBIT) STATE 525 WENT TO 050 INSTEAD OF 150.'
05	052231				EM62:	
	052231	115	042117	020106	.ASCII	'MODF (R), A FRACTION BAD.'
	052262	020101	040502	020104	.ASCII	'A BAD CONSTANT WAS USED (NOT 24) IN STATE 046.'
	052340	047600	020122	041050	.ASCIZ	<CRLF>'OR (BUT NBIT) STATE 525 WENT TO 150 INSTEAD OF 050.'
06	052426				EM63:	
	052426	115	042117	020106	.ASCII	'MODF (R), A FRACTION BAD.'
	052456	041050	052125	055040	.ASCIZ	'(BUT ZBT) STATE 532 WENT TO 102 INSTEAD OF 122.'
07	052536				EM64:	
	052536	047515	043104	024040	.ASCII	'MODF (R), A FRACTION BAD.'
	052567	050	052502	020124	.ASCIZ	'(BUT ENBT EZBT) STATE 041 WENT TO 046 INSTEAD OF 246.'
08	052655				EM65:	
	052655	115	042117	020106	.ASCII	'MODF (R), A FRACTION BAD.'
	052706	041050	052125	043040	.ASCIZ	'(BUT FT) STATE 126 SHOULD HAVE GONE TO 133. FT=0.'
09	052770				EM66:	
	052770	047515	043104	024040	.ASCII	'MODF (R), A FRACTION BAD.'
	053021	123	043511	020116	.ASCIZ	'SIGN BIT BAD.'
10	053037				EM67:	
	053037	115	042117	020106	.ASCII	'MODF (R), A INTEGER BAD.'
	053067	123	043511	020116	.ASCIZ	'SIGN BIT BAD IN STATE 733.'
	053122	047515	042104	024040	EM70:	'MODF (R), A FRACTION BAD.'
	053153	115	042117	020104	EM71:	'MODF (R), A INTEGER BAD.'
	053203	106	051520	041040	EM72:	'FPS BAD AFTER MODF (R), A.'
		053203			EM73=EM72	
					EM74:	
11	053235	115	042117	020104	.ASCII	'MODF (R), A INTEGER BAD.'
	053265	050	052502	020124	.ASCIZ	'(BUT FD) STATE 231 WENT TO 142 INSTEAD OF 143.'
12	053344				EM75:	
	053344	047515	042104	024040	.ASCII	'MODF (R), A FRACTION BAD.'
	053375	101	030103	043440	.ASCIZ	'AC0 GETS 0 IN STATE 425 FAILED.'
13	053435				EM76:	
	053435	115	042117	020104	.ASCII	'MODF (R), A INTEGER BAD.'
	053465	101	030503	043440	.ASCIZ	'AC1 GETS 0 IN STATE 143 FAILED.'
14	053525				EM77:	
	053525	115	042117	020104	.ASCII	'MODF (R), A FRACTION BAD.'
	053556	041050	052125	043040	.ASCIZ	'(BUT FD) STATE 526 WENT TO 134 INSTEAD OF 135.'
15	053635				EM100:	
	053635	115	042117	020104	.ASCII	'MODF (R), A INTEGER BAD.'
	053665	123	043511	020116	.ASCIZ	'SIGN BIT BAD IN STATE 526.'
16	053720				EM101:	
	053720	047515	042104	024040	.ASCII	'MODF (R), A FRACTION BAD.'
	053751	101	041040	042101	.ASCII	'A BAD CONSTANT WAS USED (NOT 56) IN STATE 046.'
	054027	200	051117	024040	.ASCIZ	<CRLF>'OR (BUT NBIT) STATE 525 WENT TO 050 INSTEAD OF 150.'
17	054114				EM102:	
	054114	047515	042104	024040	.ASCII	'MODF (R), A FRACTION BAD.'
	054145	101	041040	042101	.ASCII	'A BAD CONSTANT WAS USED (NOT 56) IN STATE 046.'
	054223	200	051117	024040	.ASCIZ	<CRLF>'OR (BUT NBIT) STATE 525 WENT TO 150 INSTEAD OF 050.'
18	054310				EM103:	

1)	054310	047515	042104	024040	.ASCII	'MODD (R),A FRACTION BAD.' <CRLF>
	054341	050	052502	020124	.ASCIZ	'(BUT ZBIT) STATE 532 WENT TO 122 INSTEAD OF 102.'
0)	054422				EM104:	
1)	054422	047515	042104	024040	.ASCII	'MODD (R),A INTEGER BAD.' <CRLF>
	054452	042523	020124	047111	.ASCII	'SET INTEGER IN AC1 FAILED.'
	054504	047600	020122	041050	.ASCIZ	<CRLF>'OR (BUT FD) STATE 733 WENT TO 156 INSTEAD OF 157.'
0)	054567				EM105:	
1)	054567	115	042117	020104	.ASCII	'MODD (R),A INTEGER BAD.' <CRLF>
	054617	050	052502	020124	.ASCIZ	'(BUT FD) STATE 122 WENT TO 424 INSTEAD OF 425.'
0)	054676				EM106:	
1)	054676	047515	042104	024040	.ASCII	'MODD (R),A INTEGER BAD.' <CRLF>
	054726	041050	052125	043040	.ASCIZ	'(BUT FD) STATE 246 WENT TO 126 INSTEAD OF 127.'
0)	055005				EM107:	
1)	055005	115	042117	020104	.ASCII	'MODD (R),A INTEGER BAD.' <CRLF>
	055035	050	052502	020124	.ASCIZ	'(BUT FD) STATE 446 WENT TO 126 INSTEAD OF 127.'
0)	055114				EM110:	
1)	055114	047515	042104	024040	.ASCII	'MODD (R),A FRACTION BAD.' <CRLF>
	055145	050	052502	020124	.ASCIZ	'(BUT FT) STATE 127 WENT TO 313 INSTEAD OF 113.'
0)	055224				EM165:	
1)	055224	047515	043104	024040	.ASCII	'/MODF (R),A FRACTION BAD. RESULT OVER OR UNDER FLOW.'
	055307	000			.BYTE	0
0)	055310				EM166:	
1)	055310	047515	043104	024040	.ASCII	'/MODF (R),A INTEGER BAD. RESULT OVER OR UNDER FLOW.'
	055372	000			.BYTE	0
0)	055373				EM167:	
1)	055373	106	051520	041040	.ASCII	'/FPS BAD AFTER MODF (R),A.'
	055424	000			.BYTE	0
0)	055425				EM170:	
1)	055425	106	041505	041040	.ASCII	'/FEC BAD AFTER MODF (R),A.'
	055456	054105	042520	052103	.ASCIZ	'EXPECTING UNDERFLOW, FEC=12.'
0)	055513				EM171:	
1)	055513	115	042117	020106	.ASCII	'/MODF (R),A INTEGER BAD. RESULT OVER OR UNDER FLOW.'
	055575	200	041501	020061	.ASCIZ	<CRLF>'AC1 GETS 0 IN STATE 126 FAILED.'
0)	055636				EM172:	
1)	055636	042506	020103	040502	.ASCII	'/FEC BAD AFTER MODF (R),A.'
	055667	105	050130	041505	.ASCIZ	'EXPECTING OVERFLOW, FEC=10.'
0)	055723				EM173:	
1)	055723	115	042117	020106	.ASCII	'/MODF (R),A INTEGER BAD. RESULT OVER OR UNDER FLOW.'
	056005	200	041050	052125	.ASCIZ	<CRLF>'(BUT FIV FD) STATE 520 WENT TO 142 INSTEAD OF 162.'
0)	056071				EM174:	
1)	056071	115	042117	020106	.ASCII	'/MODF (R),A INTEGER BAD. RESULT OVER OR UNDER FLOW.'
	056153	200	041050	052125	.ASCIZ	<CRLF>'(BUT FIV FD) STATE 520 WENT TO 162 INSTEAD OF 142.'
0)	056237				EM175:	
1)	056237	115	042117	020106	.ASCII	'/MODF (R),A INTEGER BAD. RESULT OVER OR UNDER FLOW.'
	056321	200	044523	047107	.ASCIZ	<CRLF>'SIGN BAD IN STATE 517.'
0)	056351				EM176:	
1)	056351	115	042117	020104	.ASCII	'/MODD (R),A FRACTION BAD. RESULT OVER OR UNDER FLOW.'
	056434	000			.BYTE	0
0)	056435				EM177:	
1)	056435	115	042117	020104	.ASCII	'/MODD (R),A INTEGER BAD. RESULT OVER OR UNDER FLOW.'
	056517	000			.BYTE	0
0)	056520				EM200:	
1)	056520	050106	020123	040502	.ASCII	'/FPS BAD AFTER MODD (R),A.'
	056551	000			.BYTE	0
0)	056552				EM201:	
1)	056552	042506	020103	040502	.ASCII	'/FEC BAD AFTER MODD (R),A.'

056603	200	04105	042520	.ASCIZ	<CR LF>'EXPECTING UNDERFLOW, FEC=12.'
056641				EM202:	
056641	115	042117	020104	.ASCII	/MODD (R),A INTEGER BAD. RESULT OVER OR UNDER FLOW./
056723	200	041050	052125	.ASCIZ	<CR LF>'(BUT FD) STATE 241 WENT TO 126 INSTEAD OF 127.'
057003				EM203:	
057003	115	042117	020104	.ASCII	/MODD (R),A INTEGER BAD. RESULT OVER OR UNDER FLOW./
057065	200	041050	052125	.ASCIZ	<CR LF>'(BUT FD) STATE 047 WENT TO 126 INSTEAD OF 127.'
057145				EM204:	
057145	136	041505	041040	.ASCII	/FEC BAD AFTER MODD (R),A./
057176	042600	050130	041505	.ASCIZ	<CR LF>'EXPECTING OVERFLOW, FEC=10.'
057233				EM205:	
057233	115	042117	020104	.ASCII	/MODD (R),A INTEGER BAD. RESULT OVER OR UNDER FLOW./
057315	200	041050	052125	.ASCIZ	<CR LF>'(BUT FIV FD) STATE 520 WENT TO 162 INSTEAD OF 163.'
057401				EM206:	
057401	115	042117	020104	.ASCII	/MODD (R),A INTEGER BAD. RESULT OVER OR UNDER FLOW./
057463	200	041050	052125	.ASCIZ	<CR LF>'(BUT FIV FD) STATE 520 WENT TO 162 INSTEAD OF 163.'
057547				* EM207:	
057547	101	042104	020104	.ASCIZ	/ADD (R),A PRODUCED A BAD RESULT./
057611				EM210:	
057611	124	042510	043040	.ASCIZ	/THE FPS WAS BAD AFTER ADD (R),A./
057653	101	042104	020104	EM211:	.ASCII /ADD (R),A FAILED IN THE ROUND\TRUNK FLOWS.'
057726	053600	047105	020124	.ASCII	<CR LF>'WENT FROM STATE 663 TO 313, <CR LF>'
057763	111	051516	042524	.ASCIZ	'INSTEAD OF FROM 663 TO 353.'
060017				EM212:	
060017	131	042104	020104	.ASCII	/ADD (R),A FAILED IN THE ROUND\TRUNK FLOWS.'
060072	052200	042510	043040	.ASCII	<CR LF>'THE FPS WAS BAD. <CR LF>'
060114	044504	020104	047516	.ASCII	'DID NOT TAKE THE PATH: <CR LF>'
060143	106	047522	020115	.ASCIZ	/FROM STATE 664, TO 505, TO 251./
060203				EM213:	
060203	101	042104	020104	.ASCII	/ADD (R),A FAILED IN THE ROUND\TRUNK FLOWS.'
060256	052200	042510	043040	.ASCII	<CR LF>'THE FPS WAS BAD. <CR LF>'
060300	044504	020104	047516	.ASCII	'DID NOT TAKE THE PATH: <CR LF>'
060327	106	047522	020115	.ASCIZ	/FROM STATE 664, TO 505, TO 253./
060367				EM214:	
060367	101	042104	020104	.ASCII	/ADD (R),A FAILED IN THE ROUND\TRUNK FLOWS.'
060442	052200	042510	043040	.ASCII	<CR LF>'THE FPS WAS BAD. <CR LF>'
060464	044504	020104	047516	.ASCII	'DID NOT TAKE THE PATH: <CR LF>'
060513	106	047522	020115	.ASCIZ	/FROM STATE 664, TO 705, TO 735./
060553				EM215:	
060553	101	042104	020104	.ASCII	/ADD (R),A FAILED IN THE ROUND\TRUNK FLOWS.'
060626	052200	042510	043040	.ASCII	<CR LF>'THE FPS WAS BAD. <CR LF>'
060650	044504	020104	047516	.ASCII	'DID NOT TAKE THE PATH: <CR LF>'
060677	106	047522	020115	.ASCIZ	/FROM STATE 664, TO 705, TO 737./
060737	124	042510	024040	EM216:	.ASCII /THE (BUT FIV FORK IN THE OVER\UNDER FLOWS FAILED. FIV =1.'
061030	053600	047105	020124	.ASCII	<CR LF>'WENT FROM STATE 331 TO 115. <CR LF>'
061065	111	051516	042524	.ASCIZ	'INSTEAD OF FROM 331 TO 155.'
061121	101	042104	020104	EM217:	.ASCIZ /ADD (R)A TRAPPED TO 244, FID=1.'
061163				EM220:	
061163	101	042104	020104	.ASCII	/ADD (R),A TRAPPED TO 244. <CR LF>'
061216	044124	020105	042522	.ASCII	'THE RESULT WAS AN OVERFLOW CONDITION BUT FIV= 0.'
061276	024200	052502	020124	.ASCIZ	<CR LF>'(BUT FIV) STATE 133 WENT TO 136 INSTEAD OF 116.'
061357	111	051516	042524	.ASCIZ	/INSTEAD OF FROM 133 TO 116./
061413				EM221:	
061413	101	042104	020104	.ASCII	/ADD (R),A FAILED TO TRAP TO 244. <CR LF>'

(1)	061455	124	042510	051040	.ASCII	'THE RESULT WAS A OVERFLOW CONDITION AND FIU=1.' CRLF
(1)	061534	044124	020105	041050	.ASCII	'THE (BUT FIU) FORK FAILED.' CRLF
(1)	061567	127	047105	020124	.ASCII	'WENT FROM STATE 133 TO 116.' CRLF
(1)	061623	111	051516	042524	.ASCIIZ	'INSTEAD OF FROM 133 TO 136.' CRLF
(0)	061657				EM222:	
(1)	061657	101	042104	020104	.ASCII	'ADD (R), A TRAPPED TO 244.' CRLF
(1)	061712	044124	020105	042522	.ASCII	'THE RESULT WAS AN UNDERFLOW CONDITION BUT FIU=0.' CRLF
(1)	061773	200	041050	052125	.ASCIIZ	'(BUT FIU) STATE 331 WENT TO 155 INSTEAD OF 115.' CRLF
(1)	062054	047111	052123	040505	.ASCIIZ	'INSTEAD OF FROM 331 TO 115.' CRLF
(0)	062110				EM223:	
(1)	062110	042101	042104	024040	.ASCII	'ADD (R), A FAILED TO TRAP TO 244.' CRLF
(1)	062152	044124	020105	042522	.ASCII	'THE RESULT WAS A UNDERFLOW CONDITION AND FIU=1.' CRLF
(1)	062232	044124	020105	041050	.ASCII	'THE (BUT FIU) FORK FAILED.' CRLF
(1)	062265	127	047105	020124	.ASCII	'WENT FROM STATE 331 TO 115.' CRLF
(1)	062321	111	051516	042524	.ASCIIZ	'INSTEAD OF FROM 331 TO 155.' CRLF
(0)	062355				EM224:	
(1)	062355	101	042104	020104	.ASCII	'ADD (R), A TRAPPED TO 244.' CRLF
(1)	062410	044124	020105	042522	.ASCII	'THE RESULT WAS AN UNDERFLOW CONDITION BUT FIU=0.' CRLF
(1)	062471	200	041050	052125	.ASCIIZ	'(BUT FIU) STATE 137 WENT TO 155 INSTEAD OF 115.' CRLF
(1)	062552	047111	052123	040505	.ASCIIZ	'INSTEAD OF FROM 137 TO 115.' CRLF
(0)	062606				EM225:	
(1)	062606	042101	042104	024040	.ASCII	'ADD (R), A FAILED TO TRAP TO 244.' CRLF
(1)	062650	044124	020105	042522	.ASCII	'THE RESULT WAS A UNDERFLOW CONDITION AND FIU=1.' CRLF
(1)	062730	044124	020105	041050	.ASCII	'THE (BUT FIU) FORK FAILED.' CRLF
(1)	062763	127	047105	020124	.ASCII	'WENT FROM STATE 137 TO 115.' CRLF
(1)	063017	111	051516	042524	.ASCIIZ	'INSTEAD OF FROM 137 TO 155.' CRLF
(0)	063053				EM226:	
(1)	063053	101	042104	020104	.ASCII	'ADD (R), A TRAPPED TO 244.' CRLF
(1)	063105	200	042502	040503	.ASCII	'(BUT FIU) STATE 137 WENT TO 155 INSTEAD OF 115.' CRLF
(1)	063161	102	052125	052040	.ASCIIZ	'BECAUSE OF AN EXPECTED OVERFLOW CONDITION, BUT THE FEC WAS BAD.' CRLF
(0)	063206				EM227:	
(1)	063206	042101	042104	024040	.ASCII	'ADD (R), A TRAPPED TO 244.' CRLF
(1)	063240	041200	041505	052501	.ASCII	'(BUT FIU) STATE 137 WENT TO 155 INSTEAD OF 115.' CRLF
(1)	063314	052502	020124	044124	.ASCIIZ	'BECAUSE OF AN EXPECTED OVERFLOW CONDITION, BUT THE FPS WAS BAD.' CRLF
(0)	063341				EM230:	
(1)	063341	101	042104	020104	.ASCII	'ADD (R), A TRAPPED TO 244.' CRLF
(1)	063373	200	042502	040503	.ASCII	'(BUT FIU) STATE 137 WENT TO 155 INSTEAD OF 115.' CRLF
(1)	063450	052502	020124	044124	.ASCIIZ	'BECAUSE OF AN EXPECTED UNDERFLOW CONDITION, BUT THE FEC WAS BAD.' CRLF
(0)	063475				EM231:	
(1)	063475	101	042104	020104	.ASCII	'ADD (R), A TRAPPED TO 244.' CRLF
(1)	063527	200	042502	040503	.ASCII	'(BUT FIU) STATE 137 WENT TO 155 INSTEAD OF 115.' CRLF
(1)	063604	052502	020124	044124	.ASCIIZ	'BECAUSE OF AN EXPECTED UNDERFLOW CONDITION, BUT THE FPS WAS BAD.' CRLF
		057611			EM232=EM210	
(0)	063631				EM233:	
(1)	063631	114	041504	042106	.ASCIIZ	'LDCFD (R)+, A RESULT INCORRECT.' CRLF
(0)	063670				EM234:	
(1)	063670	030122	041040	042101	.ASCII	'RO BAD AFTER LDCFD (R)+, A.' CRLF
(1)	063722	040600	041040	042101	.ASCIIZ	'(BUT FIU) STATE 137 WENT TO 155 INSTEAD OF 115.' CRLF
(0)	063754				EM235:	

(0)	063754	041520	041040	042101	EM236:	.ASCIZ	\PC BAD AFTER LDCFD #NUM,A. TRAP TO 4.
(0)	064022						
(1)	064022	041520	041040	042101	EM237:	.ASCIZ	\PC BAD AFTER LDCFD #NUM,A.\
(0)	064055						
(1)	064055	122	020060	040502		.ASCII	\RD BAD AFTER LDCDF (R)+,A.\
(0)	064107	200	020101	040502		.ASCIZ	<CR LF>'A BAD CONSTANT WAS USED.'
(0)	064141				EM240:		
(1)	064141	106	051520	041040		.ASCIZ	\FPS BAD AFTER LDCFD (R)+,A.\
(0)	064175				EM241:		
(1)	064175	114	041504	042106		.ASCII	\LDCFD (R)+,A FAILED.\
(1)	064221	200	044124	020105		.ASCII	<CR LF>'THE FD '
(1)	064231	102	052111	053440		.ASCII	'BIT WAS NOT COMPLIMENTED '
(1)	064262	047111	051440	040524		.ASCIZ	'IN STATE 017.'
(0)	064300				EM242:		
(1)	064300	050106	020123	040502		.ASCIZ	\FPS BAD AFTER LDCDF (R)+,A.\
(0)	064334				EM243:		
(1)	064334	042114	042103	020106		.ASCII	\LDCDF (R)+,A FAILED.\
(1)	064360	052200	042510	043040		.ASCII	<CR LF>'THE FD '
(1)	064370	044502	020124	040527		.ASCII	'BIT WAS NOT COMPLIMENTED '
(0)	064421	111	020116	052123		.ASCIZ	'IN STATE 017.'
(0)	064437				EM244:		
(1)	064437	114	041504	043104		.ASCIZ	\LDCDF (R)+,A RESULT INCORRECT.\
(0)	064476	042114	043103	020104	EM245:	.ASCII	'LDCFD (R),A FAILED.'
(0)	064521	200	042523	020124		.ASCII	<CR LF>'SET SIGN FAILED '
(0)	064542	047111	051440	040524		.ASCIZ	'IN STATE 512.'
(0)	064560	047125	054105	042520	EM247:	.ASCIZ	'UNEXPECTED FPP TRAP TO 244.'
(0)	064614	047125	054105	042520	EM250:	.ASCIZ	'UNEXPECTED CPU TRAP TO 4.'
(0)	064646	047125	054105	042520	EM251:	.ASCIZ	'UNEXPECTED CPU TRAP TO 10.'
(0)	064701	103	051117	042522	EM252:	.ASCIZ	'CORRECT FLOWS INTERRUPT TEST MODULE FAILED TO INTERRUPT '
(0)	064772				EM253:		
(1)	064772	042101	042104	040440		.ASCIZ	/ADD ACC,ACC FAILED IN THE INTERRUPT CORRECT FLOWS./
(0)	065056				EM254:		
(1)	065056	042101	042104	024040		.ASCIZ	/ADD (RD),ACC FAILED IN THE INTERRUPT CORRECT FLOWS./
(0)	065143				EM255:		
(1)	065143	101	042104	020104		.ASCIZ	/ADD (RD)+,ACC FAILED IN THE INTERRUPT CORRECT FLOWS.
(0)	065231				EM256:		
(1)	065231	101	042104	020104		.ASCIZ	/ADD 2(RD)+,ACC FAILED IN THE INTERRUPT CORRECT FLOWS.
(0)	065320				EM257:		
(1)	065320	042101	042104	026440		.ASCIZ	/ADD -(RD),ACC FAILED IN THE INTERRUPT CORRECT FLOWS.
(0)	065406				EM260:		
(1)	065406	042101	042104	040040		.ASCIZ	/ADD 2-(RD),ACC FAILED IN THE INTERRUPT CORRECT FLOWS.
(0)	065475				EM261:		
(1)	065475	101	042104	020104		.ASCIZ	/ADD NUM(RD),ACC FAILED IN THE INTERRUPT CORRECT FLOWS.
(0)	065565				EM262:		
(1)	065565	101	042104	020104		.ASCIZ	/ADD 3NUM(RD),ACC FAILED IN THE INTERRUPT CORRECT FLOWS.
(0)	065656				EM263:		
(1)	065656	044504	042126	024040		.ASCIZ	/DIVD (RD),ACC FAILED IN THE INTERRUPT CORRECT FLOWS.
(0)	065743				EM264:		
(1)	065743	115	046125	020104		.ASCIZ	/MULD (RD),ACC FAILED IN THE INTERRUPT CORRECT FLOWS.
(0)	066030				EM265:		
(1)	066030	047515	042104	024040		.ASCIZ	/MODD (RD),ACC FAILED IN THE INTERRUPT CORRECT FLOWS.

066115 103 051117 042522 EM266: .ASCIZ 'CORRECT FLOWS INTERRUPT TEST MODULE CAUSED UNEXPECTED INTERRUPT.'

: THESE ARE ERROR DATA TABLE HEADERS:

066216	020040	042524	052123	DH1:	.ASCIZ	'TEST.'	<TAB>	'CALL AT PC.'	<TAB>	'ERROR AT PC.'	<TAB>			
066257	107	052117	043040		.ASCIZ	'GOT FPS.'	<TAB>	'EXPECTED FPS.'						
066306	020040	042524	052123	DH2:	.ASCIZ	'TEST.'	<TAB>	'CALL AT PC.'	<TAB>	'ERROR AT PC.'				
066216				DH3=	DH1									
066216				DH4=	DH1									
066216				DH5=	DH1									
066216				DH6=	DH1									
066216				DH7=	DH1									
066216				DH10=	DH1									
066216				DH11=	DH1									
066216				DH12=	DH1									
066216				DH13=	DH1									
066216				DH14=	DH1									
066216				DH15=	DH1									
066216				DH16=	DH1									
066347	040	052040	051505	DH17:	.ASCIZ	'TEST.'	<TAB>	'PC OF CALL.'	<TAB>	'PC OF ERROR.'	<TAB>	'FEC.'	<TAB>	'FPS.'
066422	020040	042524	052123	DH20=	DH1									
066462	047507	020124	042506	DH21:	.ASCIZ	'TEST.'	<TAB>	'PC OF CALL.'	<TAB>	'PC OF ERROR.'	<TAB>			
066511	040	052040	051505	DH22:	.ASCIZ	'GOT FEC.'	<TAB>	'EXPECTED FEC.'						
066216				DH23=	DH1									
066216				DH32=	DH1									
066216				DH24=	DH1									
066216				DH25=	DH1									
066216				DH26=	DH1									
066216				DH27=	DH1									
066216				DH30=	DH1									
066216				DH31=	DH1									
066216				DH33=	DH1									
066216				DH34=	DH1									
066216				DH35=	DH1									
066216				DH36=	DH1									
066216				DH37=	DH1									
066216				DH40=	DH1									
066216				DH41=	DH1									
066216				DH42=	DH1									
066216				DH43=	DH1									
066216				DH44=	DH1									
066216				DH45=	DH1									
066216				DH246=	DH1									
066216				DH46=	DH1									
066216				DH47=	DH1									
066216				DH50=	DH1									
066216				DH51=	DH1									
066216				DH52=	DH1									
066216				DH111=	DH1									
066216				DH112=	DH1									
066216				DH113=	DH1									
066216				DH114=	DH1									
066557	040	052040	051505	DH115:	.ASCIZ	'TEST.'	<TAB>	'PC OF CALL.'	<TAB>	'PC OF ERROR.'	<TAB>			
066620	047507	020124	042506		.ASCIZ	'GOT FEC.'	<TAB>	'GOT FPS.'	<TAB>	'EXPECTED FPS.'				
066557				DH116=	DH115									

066557	DM117=DM115
066557	DM120=DM115
066557	DM121=DM115
066557	DM122=DM115
066216	DM123=DM1
066216	DM124=DM1
066216	DM125=DM1
066216	DM126=DM1
066557	DM127=DM115
066557	DM130=DM115
066557	DM131=DM115
066216	DM132=DM1
066557	DM133=DM115
066557	DM134=DM115
066216	DM135=DM1
066557	DM136=DM115
066557	DM137=DM115
066557	DM140=DM115
066557	DM141=DM115
066557	DM142=DM115
066557	DM143=DM115
066557	DM144=DM115
066216	DM145=DM1
066216	DM146=DM1
066216	DM147=DM1
066216	DM150=DM1
066557	DM151=DM115
066557	DM152=DM115
066557	DM153=DM115
066557	DM154=DM115
066557	DM155=DM115
066557	DM156=DM115
066216	DM157=DM1
066557	DM160=DM115
066216	DM161=DM1
066216	DM162=DM1
066557	DM163=DM115
066216	DM164=DM1
066216	DM53=DM1
066216	DM54=DM1
066216	DM55=DM1
066216	DM56=DM1
066216	DM57=DM1
066216	DM60=DM1
066216	DM61=DM1
066216	DM62=DM1
066216	DM63=DM1
066216	DM64=DM1
066216	DM65=DM1
066216	DM66=DM1
066216	DM67=DM1
066216	DM70=DM1
066216	DM71=DM1
066216	DM72=DM1
066216	DM73=DM1
066216	DM74=DM1

066216				DH75=DH1	
066216				DH76=DH1	
066216				DH77=DH1	
066216				DH100=DH1	
066216				DH101=DH1	
066216				DH102=DH1	
066216				DH103=DH1	
066216				DH104=DH1	
066216				DH105=DH1	
066216				DH106=DH1	
066216				DH107=DH1	
066216				DH110=DH1	
066557				DH165=DH115	
066557				DH166=DH115	
066557				DH167=DH115	
066557				DH170=DH115	
066557				DH171=DH115	
066557				DH172=DH115	
066557				DH173=DH115	
066557				DH174=DH115	
066557				DH175=DH115	
066557				DH176=DH115	
066557				DH177=DH115	
066557				DH200=DH115	
066557				DH201=DH115	
066557				DH202=DH115	
066557				DH203=DH115	
066557				DH204=DH115	
066557				DH205=DH115	
066557				DH206=DH115	
066656	020040	042524	052123	DH220: .ASCIZ	' TEST.'<TAB>'PC OF CALL.'<TAB>'PC OF TRAP.'
066716	020040	042524	052123	DH207: .ASCII	' TEST.'<TAB>'PC OF CALL.'<TAB>'PC OF ERROR.'
066756	043411	052117	043040	.ASCIZ	'<TAB>'GOT FPS.'<TAB>'EXPECTED FPS.'
067006	020040	042524	052123	DH210: .ASCIZ	' TEST.'<TAB>'PC OF CALL.'<TAB>'PC OF ERROR.'
	067006			DH211=DH210	
	066716			DH212=DH207	
	066716			DH213=DH207	
	066716			DH214=DH207	
	066716			DH215=DH207	
	067006			DH216=DH210	
067047	040	052040	051505	DH217: .ASCII	' TEST.'<TAB>'PC OF CALL.'<TAB>'PC OF ERROR.'
067107	011	050105	027123	.ASCIZ	'<TAB>'FPS.'<TAB>'FEC.'
	067006			DH221=DH210	
	066656			DH222=DH220	
	067006			DH223=DH210	
	066656			DH224=DH220	
	067006			DH225=DH210	
067122	020040	042524	052123	DH226: .ASCII	' TEST.'<TAB>'PC OF CALL.'<TAB>'PC OF TRAP.'
067161	011	047507	020124	.ASCIZ	'<TAB>'GOT FEC.'<TAB>'EXPECTED FEC.'
067211	040	052040	051505	DH227: .ASCII	' TEST.'<TAB>'PC OF CALL.'<TAB>'PC OF TRAP.'
067250	043411	052117	043040	.ASCIZ	'<TAB>'GOT FPS.'<TAB>'EXPECTED FPS.'
	067122			DH230=DH226	
	067211			DH231=DH227	
	067211			DH232=DH227	

```

067300 020040 042524 052123 DH233: .ASCIZ ' TEST.<TAB>'PC OF CALL.<TAB>'PC OF ERROR.'
067341 040 052040 051505 DH234: .ASCII ' TEST.<TAB>'PC OF CALL.<TAB>'PC OF ERROR.'
067401 011 047507 020124 .ASCIZ <TAB>'GOT RO.<TAB>'EXPECTED RO.'
067427 040 052040 051505 DH235: .ASCII ' TEST.<TAB>'PC OF CALL.<TAB>
067453 120 020103 043117 .ASCIZ 'PC OF TRAP.<TAB>'
067467 040 052040 051505 DH236: .ASCII ' TEST.<TAB>'PC OF CALL.<TAB>
067513 120 020103 043117 .ASCIZ 'PC OF ERROR.<TAB>'GOT PC.'
067537 011 054105 042520 .ASCIZ <TAB>'EXPECTED PC.'

067341 DH237=DH234
067555 040 052040 051505 DH240: .ASCII ' TEST.<TAB>'PC OF CALL.<TAB>
067601 120 020103 043117 .ASCIZ 'PC OF ERROR.<TAB>'
067616 047507 020124 050106 .ASCIZ 'GOT FPS.<TAB>'EXPECTED FPS.'

067300 DH241=DH233
067555 DH242=DH240
067300 DH243=DH233
067300 DH244=DH233
067300 DH245=DH233
067645 040 052040 051505 DH247: .ASCIZ ' TEST.<TAB>'PC OF CALL.<TAB>'PC OF TRAP.<TAB>'FEC.'
067712 020040 042524 052123 DH250: .ASCIZ ' TEST.<TAB>'PC OF CALL.<TAB>'PC OF TRAP.'
067712 DH251=DH250

066306 DH252=DH2
066216 DH253=DH1
066216 DH254=DH1
066216 DH255=DH1
066216 DH256=DH1
066216 DH257=DH1
066216 DH260=DH1
066216 DH261=DH1
066216 DH262=DH1
066216 DH263=DH1
066216 DH264=DH1
066216 DH265=DH1
067752 020040 042524 052123 DH266: .ASCIZ ' TEST.<TAB>'PC OF CALL.<TAB>'PC OF TRAP.'

```

: THESE ARE THE DATA FORMAT SPECIFIERS FOR THE DATA TABLE:

```

070012 004 000 005 DF1: .BYTE 4,0,5,0,5,0,5,0,5,5,3,5,5,3
070030 004 000 005 DF2: .BYTE 4,0,5,0,5,5,3,5,5,3
070012 DF3=DF1
070012 DF4=DF1
070012 DF5=DF1
070012 DF6=DF1
070012 DF7=DF1
070012 DF10=DF1
070012 DF11=DF1
070012 DF12=DF1
070012 DF13=DF1
070012 DF14=DF1
070012 DF15=DF1

```

070042	070012	000	005	DF16=DF1	
07005:	004	000	005	DF17: .BYTE	4,0,5,0,5,0,0
	070051			DF20: .BYTE	4,0,5,0,5,0,5 0
	070051			DF21=DF20	
070061	070051	000	005	DF22=DF20	
	004			DF23: .BYTE	4,0,5,0,5,0,5,0,5,5,2,5,5,2,5,5,2
	070061			DF24=DF23	
	070061			DF32=DF23	
	070061			DF25=DF23	
	070061			DF26=DF23	
	070061			DF27=DF23	
	070061			DF30=DF23	
070105	070061	000	005	DF31=DF23	
	004			DF33: .BYTE	4,0,5,0,5,0,5,0,5,5,3,5,5,3,5,5,3,5,5,2
	070105			DF34=DF33	
	070105			DF35=DF33	
	070105			DF36=DF33	
	070061			DF37=DF23	
	070061			DF40=DF23	
	070061			DF41=DF23	
	070061			DF42=DF23	
	070061			DF43=DF23	
	070061			DF44=DF23	
	070061			DF45=DF23	
	070105			DF246=DF33	
	070105			DF46=DF33	
	070105			DF47=DF33	
	070105			DF50=DF33	
	070105			DF51=DF33	
	070105			DF52=DF33	
	070061			DF111=DF23	
	070061			DF112=DF23	
	070061			DF113=DF23	
	070061			DF114=DF23	
070131	070061	000	005	DF115: .BYTE	4,0,5,0,5,0,0,0,5,5,2,5,5,2,5,5,2,5,5,2
	004			DF116=DF115	
	070131			DF117=DF115	
	070131			DF120=DF115	
	070131			DF121=DF115	
	070131			DF122=DF115	
	070105			DF123=DF33	
	070105			DF124=DF33	
	070105			DF125=DF33	
070155	070105	000	005	DF126=DF33	
	004			DF127: .BYTE	4,0,5,0,5,0,0,0,5,5,3,5,5,3,5,5,3,5,5,3
	070155			DF130=DF127	
	070155			DF131=DF127	
	070105			DF132=DF33	
	070155			DF133=DF127	
	070155			DF134=DF127	
	070105			DF135=DF33	
	070155			DF136=DF127	
	070131			DF137=DF115	
	070131			DF140=DF115	
	070131			DF141=DF115	
	070131			DF142=DF115	

070131			DF143=DF115	
070131			DF144=DF115	
070061			DF145=DF23	
070061			DF146=DF23	
070061			DF147=DF23	
070061			DF150=DF23	
070155			DF151=DF127	
070155			DF152=DF127	
070155			DF153=DF127	
070155			DF154=DF127	
070155			DF155=DF127	
070155			DF156=DF127	
070105			DF157=DF33	
070155			DF160=DF127	
070105			DF161=DF33	
070105			DF162=DF33	
070155			DF163=DF127	
070105			DF164=DF33	
070201	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
070201			DF54=DF53	
070201			DF55=DF53	
070201			DF56=DF53	
070201			DF57=DF53	
070201			DF60=DF53	
070201			DF61=DF53	
070201			DF62=DF53	
070201			DF63=DF53	
070201			DF64=DF53	
070201			DF65=DF53	
070201			DF66=DF53	
070201			DF67=DF53	
070233	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
070233			DF71=DF70	
070233			DF72=DF70	
070233			DF73=DF70	
070233			DF74=DF70	
070233			DF75=DF70	
070233			DF76=DF70	
070233			DF77=DF70	
070233			DF100=DF70	
070233			DF101=DF70	
070233			DF102=DF70	
070233			DF103=DF70	
070233			DF104=DF70	
070233			DF105=DF70	
070233			DF106=DF70	
070233			DF107=DF70	
070233			DF110=DF70	
070265	004	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
070265			DF166=DF165	
070265			DF167=DF165	
070265			DF170=DF165	
070265			DF171=DF165	
070265			DF172=DF165	
070265			DF173=DF165	
070265			DF174=DF165	

070317	070265	000	005	DF175=DF165	
	004			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
	070317			DF177=DF176	
	070317			DF200=DF176	
	070317			DF201=DF176	
	070317			DF202=DF176	
	070317			DF203=DF176	
	070317			DF204=DF176	
	070317			DF205=DF176	
	070317			DF206=DF176	
070351	004	000	005	DF210: .BYTE	4,0,5,0,5,0,5,0
070361	004	000	005	DF207: .BYTE	4,0,5,0,5,5,5,3,5,5,5,3,5,5,5,3,5,5,5,3
	070361			DF211=DF207	
070405	004	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
	070405			DF213=DF212	
	070405			DF214=DF212	
	070405			DF215=DF212	
	070361			DF216=DF207	
070435	004	000	005	DF217: .BYTE	4,0,5,0,5,0,0
	070361			DF220=DF207	
	070361			DF221=DF207	
	070361			DF222=DF207	
	070361			DF223=DF207	
	070361			DF224=DF207	
	070361			DF225=DF207	
	070405			DF226=DF212	
	070405			DF227=DF212	
	070405			DF230=DF212	
	070405			DF231=DF212	
	070405			DF232=DF212	
070444	004	000	005	DF233: .BYTE	4,0,5,0,5,5,3,5,5,3,5,5,3
070461	004	000	005	DF234: .BYTE	4,0,5,0,5,0,0
070470	004	000	005	DF235: .BYTE	4,0,5,0
	070461			DF236=DF234	
	070461			DF237=DF234	
	070461			DF240=DF234	
	070444			DF241=DF233	
	070461			DF242=DF234	
	070444			DF243=DF233	
	070444			DF244=DF233	
	070444			DF245=DF233	
070474	004	000	005	DF247: .BYTE	4,0,5,0,5,0
	070474			DF250=DF247	
	070474			DF251=DF247	
070502	004	000	005	DF252: .BYTE	4,0,5,0
070506	004	000	005	DF253: .BYTE	4,0,5,0,5,0,5,0,5,5,0,5,5,0,5,5,0,5,5,0,5,5,0,5,5,3,5,5,3
	070506			DF254=DF253	
	070506			DF255=DF253	
	070506			DF256=DF253	
	070506			DF257=DF253	
	070506			DF260=DF253	
	070506			DF261=DF253	
	070506			DF262=DF253	
	070506			DF263=DF253	

070506
070506
070502

DF264=DF253
DF265=DF253
DF266=DF252

.EVEN

* THESE ARE ERROR DATA TABLES (FORMATTED ABOVE):

070540	001232	001234	037151	DT1:	.WORD	STMP0, STMP1, STAB, STMP2, STAB, STMP5, STAB, STMP6
070560	001313	037156	001240		.WORD	SCRLF, MS1, STMP3, SCRLF, MS2, STMP4, 0
070576	001232	001234	037151	DT2:	.WORD	STMP0, STMP1, STAB, STMP2, SCRLF, MS3, STMP3, SCRLF, MS4, STMP4, 0
	070540			DT3=DT1		
	070540			DT4=DT1		
	070540			DT5=DT1		
	070540			DT6=DT1		
	070540			DT7=DT1		
	070540			DT10=DT1		
	070540			DT11=DT1		
	070540			DT12=DT1		
	070540			DT13=DT1		
	070540			DT14=DT1		
	070540			DT15=DT1		
	070540			DT16=DT1		
070624	001232	001234	037151	DT17:	.WORD	STMP0, STMP1, STAB, STMP2, STAB, STMP4, STMP5, 0
070644	001232	001234	037151	DT20:	.WORD	STMP0, STMP1, STAB, STMP2, STAB, STMP4, STAB, STMP5, 0
070666	001232	001234	037151	DT21:	.WORD	STMP0, STMP1, STAB, STMP2, STAB, STMP4, STAB, STMP3, 0
070710	001232	001234	037151	DT22:	.WORD	STMP0, STMP1, STAB, STMP2, STAB, STMP5, 0
070726	001232	001234	037151	DT23:	.WORD	STMP0, STMP1, STAB, STMP2, STAB, STMP7, STAB, STMP10, SCRLF, MS1, STMP3, SCRLF, MS2, SCRLF, MS5, STMP5, SCRLF, MS6, STMP6, 0
070762	001313	037271	001244		.WORD	
	070726			DT32=DT23		
	070726			DT24=DT23		
	070726			DT25=DT23		
	070726			DT26=DT23		
	070726			DT27=DT23		
	070726			DT30=DT23		
	070726			DT31=DT23		
	070726			DT33=DT23		
	070726			DT34=DT23		
	070726			DT35=DT23		
	070726			DT36=DT23		
	070726			DT37=DT23		
	070726			DT40=DT23		
	070726			DT41=DT23		
	070726			DT42=DT23		
	070726			DT43=DT23		
	070726			DT44=DT23		
	070726			DT45=DT23		
	070726			DT46=DT23		
	070726			DT46=DT23		
	070726			DT47=DT23		
	070726			DT50=DT23		
	070726			DT51=DT23		
	070726			DT52=DT23		
	070726			DT111=DT23		
	070726			DT112=DT23		

070726				DT113=DT23	
070726				DT114=DT23	
071000	001232	001234	037151	DY115: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,\$TAB,\$TMP11,\$TAB,\$TMP12
071022	001313	037156	001240	.WORD	\$CRLF,\$M51,\$TMP3,\$CRLF,\$M52,\$TMP4
071036	001313	037271	001244	.WORD	\$CRLF,\$M55,\$TMP5,\$CRLF,\$M56,\$TMP6.C
071000				DT116=DT115	
071000				DT117=DT115	
071000				DT120=DT115	
071000				DT121=DT115	
071000				DT122=DT115	
070726				DT123=DT23	
070726				DT124=DT23	
070726				DT125=DT23	
070726				DT126=DT23	
071000				DT127=DT115	
071000				DT130=DT115	
071000				DT131=DT115	
070726				DT132=DT23	
071000				DT133=DT115	
071000				DT134=DT115	
070726				DT135=DT23	
071000				DT136=DT115	
071000				DT137=DT115	
071000				DT140=DT115	
071000				DT141=DT115	
071000				DT142=DT115	
071000				DT143=DT115	
071000				DT144=DT115	
070726				DT145=DT23	
070726				DT146=DT23	
070726				DT147=DT23	
070726				DT150=DT23	
071000				DT151=DT115	
071000				DT152=DT115	
071000				DT153=DT115	
071000				DT154=DT115	
071000				DT155=DT115	
071000				DT156=DT115	
070726				DT157=DT23	
071000				DT160=DT115	
070726				DT161=DT23	
070726				DT162=DT23	
071000				DT163=DT115	
070726				DT164=DT23	
071054	001232	001234	037151	DT53: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,\$TAB,\$TMP11,\$TAB,\$TMP12
071074	001313	037156	001240	.WORD	\$CRLF,\$M51,\$TMP3,\$CRLF,\$M52,\$TMP4
071110	001313	037377	001244	.WORD	\$CRLF,\$M51,\$TMP5,\$CRLF,\$M52,\$TMP6
071124	001313	037331	001250	.WORD	\$CRLF,\$M57,\$TMP7,\$CRLF,\$M510,\$TMP10.O
071054				DT54=DT53	
071054				DT55=DT53	
071054				DT56=DT53	
071054				DT57=DT53	
071054				DT60=DT53	
071054				DT61=DT53	
071054				DT62=DT53	
071054				DT63=DT53	

	071054			DT64=DT53	
	071054			DT65=DT53	
	071054			DT66=DT53	
	071054			DT67=DT53	
	071054			DT70=DT53	
	071054			DT71=DT53	
	071054			DT72=DT53	
	071054			DT73=DT53	
	071054			DT74=DT53	
	071054			DT75=DT53	
	071054			DT76=DT53	
	071054			DT77=DT53	
	071054			DT100=DT53	
	071054			DT101=DT53	
	071054			DT102=DT53	
	071054			DT103=DT53	
	071054			DT104=DT53	
	071054			DT105=DT53	
	071054			DT106=DT53	
	071054			DT107=DT53	
	071054			DT110=DT53	
071142	001232	001234	037151	DT165: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,\$TAB,\$TMP13,\$TMP11,\$TMP12
071162	001313	037156	001240	.WORD	\$CRLF,\$M51,\$TMP3,\$CRLF,\$M52,\$TMP4
071176	001313	037377	001244	.WORD	\$CRLF,\$M51,\$TMP5,\$CRLF,\$M52,\$TMP6
071212	001313	037331	001250	.WORD	\$CRLF,\$M57,\$TMP7,\$CRLF,\$M510,\$TMP10,0
	071142			DT166=DT165	
	071142			DT167=DT165	
	071142			DT170=DT165	
	071142			DT171=DT165	
	071142			DT172=DT165	
	071142			DT173=DT165	
	071142			DT174=DT165	
	071142			DT175=DT165	
	071142			DT176=DT165	
	071142			DT177=DT165	
	071142			DT200=DT165	
	071142			DT201=DT165	
	071142			DT202=DT165	
	071142			DT203=DT165	
	071142			DT204=DT165	
	071142			DT205=DT165	
	071142			DT206=DT165	
071230	001232	001234	037151	DT210: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,\$TAB,\$TMP3
071244	037151	001242	000000	.WORD	\$TAB,\$TMP4,0
071252	001232	001234	037151	DT207: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,\$CRLF,\$M541,\$CRLF,\$TMP3
071272	001313	037544	001313	.WORD	\$CRLF,\$M542,\$CRLF,\$TMP4,\$CRLF,\$M543,\$CRLF,\$TMP5
071312	001313	037577	001313	.WORD	\$CRLF,\$M544,\$CRLF,\$TMP6,0
	071252			DT211=DT207	
071324	001232	001234	037151	DT212: .WORD	\$TMP0,\$TMP1,\$TAB,\$TMP2,\$TAB,\$TMP10,\$TAB,\$TMP11
071344	001313	037520	001313	.WORD	\$CRLF,\$M541,\$CRLF,\$TMP3,\$CRLF,\$M542,\$CRLF,\$TMP4
071364	001313	037562	001313	.WORD	\$CRLF,\$M543,\$CRLF,\$TMP5,\$CRLF,\$M544,\$CRLF,\$TMP6,0
	071324			DT213=DT212	
	071324			DT214=DT212	
	071324			DT215=DT212	
	071252			DT216=DT207	

H13

071406	001232	001234	037151	DT217: .WORD	STMP0,STMP1,STAB,STMP2,STAB,STMP3,STMP4,0
	071252			DT220=DT207	
	071252			DT221=DT207	
	071252			DT222=DT207	
	071252			DT223=DT207	
	071252			DT224=DT207	
	071252			DT225=DT207	
	071324			DT226=DT212	
	071324			DT227=DT212	
	071324			DT230=DT212	
	071230			DT231=DT210	
	071252			DT232=DT207	
071426	001232	001234	037151	DT233: .WORD	STMP0,STMP1,STAB,STMP2,SCRLF
071440	037446	001244	001313	.WORD	MS37,STMP5,SCRLF,MS40,STMP6
071452	001313	037520	001250	.WORD	SCRLF,MS41,STMP7,0
071462	001232	001234	037151	DT234: .WORD	STMP0,STMP1,STAB,STMP2,STAB,STMP3,STMP4,0
071502	001232	001234	037151	DT235: .WORD	STMP0,STMP1,STAB,STMP2,0
	071462			DT236=DT234	
	071462			DT237=DT234	
	071462			DT240=DT234	
	071426			DT241=DT233	
	071462			DT242=DT234	
	071426			DT243=DT233	
	071426			DT244=DT233	
	071426			DT245=DT233	
071514	001232	001234	037151	DT247: .WORD	STMP0,STMP1,STAB,STMP2,STAB,STMP3,0
071532	001232	001234	037151	DT250: .WORD	STMP0,STMP1,STAB,STMP2,0
	071532			DT251=DT250	
071544	001232	001234	037151	DT252: .WORD	STMP0,STMP1,STAB,STMP2,0
071556	001232	001234	037151	DT253: .WORD	STMP0,STMP1,STAB,STMP2,STAB,STMP7,STAB,STMP10
071576	001313	036760	001254	.WORD	SCRLF,MSA1,STMP11,SCRLF,MSA2,STMP12
071612	001313	037021	001244	.WORD	SCRLF,MSA3,STMP5,SCRLF,MSA4,STMP6
071626	001313	037051	001240	.WORD	SCRLF,MSA5,STMP3,SCRLF,MSA6,STMP4,0
	071556			DT254=DT253	
	071556			DT255=DT253	
	071556			DT256=DT253	
	071556			DT257=DT253	
	071556			DT260=DT253	
	071556			DT261=DT253	
	071556			DT262=DT253	
	071556			DT263=DT253	
	071556			DT264=DT253	
	071556			DT265=DT253	
	071544			DT266=DT252	

MAINDEC-11-OFFPB-A PDP 11 34 FPP DIAGNOSTIC PART 2 MACY11 27(1006) 01-NOV-76 21:12 PAGE 154
OFFPB.A.P11 01-NOV-76 21:06 FLAG RESET AND CONSOLE TEST ROUTINE

000001

:12345

.END

AA00M	014422	AMAMS3=	000000	BIT2	=	000004	CR	=	000015	CF144	=	07012.	
AA01	013420	AMAMS4=	000000	BIT3	=	000010	CRLF	=	000200	CF145	=	070061	
AA010	014036	AMSGAO=	000000	BIT4	=	000020	DDDD0M	017070		CF146	=	070061	
AA011	014074	AMSGLG=	000000	BIT5	=	000040	DCC1	016140		CF147	=	070061	
AA012	014132	AMSGTY=	000000	BIT6	=	000100	DD02	016214		CF15	=	070012	
AA013	014170	AMTYP1=	000000	BIT7	=	000200	DD03	016270		CF150	=	070061	
AA02	013456	AMTYP2=	000000	BIT8	=	000400	DD04	016344		CF151	=	070155	
AA03	013514	AMTYP3=	000000	BIT9	=	001000	DD05	016420		CF152	=	070155	
AA04	013552	AMTYP4=	000000	BPTVEC=	000014		DD06	016474		CF153	=	070155	
AA05	013610	APASS =	000000	CCCC0N	016134		DD07	016550		DF154	=	070155	
AA06	013646	APRIOR =	000000	CCC1	015130		DDISP	=	177570	DF155	=	070155	
AA07	013704	APTCU=	000040	CCC10	015524		DF1	=	070012	DF156	=	070155	
AA08	013742	APTENV=	000001	CCC11	015560		DF10	=	070012	DF157	=	070105	
AA09	014000	APTSIZ=	000200	CCC12	015614		DF100	=	070233	DF16	=	070012	
ABASE	=	000000	APTSP0=	000100	CCC13	015650	DF101	=	070233	DF160	=	070155	
ACDW1	=	000000	ASHREG=	000000	CCC2	015164	DF102	=	070233	DF161	=	070105	
ACDW2	=	000000	ATESTN=	000000	CCC3	015220	DF103	=	070233	DF162	=	070105	
ACPLCP=	000000	AUNIT =	000000	CCC4	015254		DF104	=	070233	DF163	=	070155	
ACC	=	%000000	AUSWR =	000000	CCC5	015310	DF105	=	070233	DF164	=	070105	
AC1	=	%000001	AVECT1=	000000	CCC6	015344	DF106	=	070233	DF165	=	070265	
AC2	=	%000002	AVECT2=	000000	CCC7	015400	DF107	=	070233	DF166	=	070265	
AC3	=	%000003	BB800M	015124	CCC8	015434	DF11	=	070012	DF167	=	070265	
AC4	=	%000004	BB8ER1	014700	CCC9	015470	DF110	=	070233	DF17	=	070042	
AC5	=	%000005	BB8ER2	014764	CKSWR =	104406	DF111	=	070061	DF170	=	070265	
AC6	=	%000006	BB8ER3	015012	CMPSUB	014232	DF112	=	070061	DF171	=	070265	
AC7	=	%000007	BB8ER4	015040	CHPTMP	014412	DF113	=	070061	DF172	=	070265	
ADDW0	=	000000	BB8P1	015104	CNT	=	000267	DF114	=	070061	DF173	=	070265
ADDW1	=	000000	BB8P2	015114	CORDON	033146	DF115	=	070131	DF174	=	070265	
ADDW10	=	000000	BB80	014430	CORFLG	033130	DF116	=	070131	DF175	=	070265	
ADDW11	=	000000	BB81	014464	CORINT	033142	DF117	=	070131	DF176	=	070317	
ADDW12	=	000000	BB82	014512	CORNES	037621	DF12	=	070012	DF177	=	070317	
ADDW13	=	000000	BB83	014542	CORSUB	032646	DF120	=	070131	DF2	=	070030	
ADDW14	=	000000	BB84	014570	CORTMP	033132	DF121	=	070131	DF20	=	070051	
ADDW15	=	000000	BB85	014626	CORTRP	033144	DF122	=	070131	DF200	=	070317	
ACDW2	=	000000	BB86	014634	CORTV	032744	DF123	=	070105	DF201	=	070317	
ACDW3	=	000000	BIT0	=	000001	CORTV0	033106	DF124	=	070105	DF202	=	070317
ACDW4	=	000000	BIT00	=	000001	CORTV1	033112	DF125	=	070105	DF203	=	070317
ACDW45	=	000000	BIT01	=	000002	COR1	032014	DF126	=	070105	DF204	=	070317
ACDW6	=	000000	BIT02	=	000004	COR10	032436	DF127	=	070155	DF205	=	070317
ACDW7	=	000000	BIT03	=	000010	COR11	032502	DF13	=	070012	DF206	=	070317
ACDW8	=	000000	BIT04	=	000020	COR12	032542	DF130	=	070155	DF207	=	070361
ACDW9	=	000000	BIT05	=	000040	COR13	032602	DF131	=	070155	DF21	=	070051
ACDYCT=	000000	BIT06	=	000100	COR2	032040	DF132	=	070105	DF210	=	070351	
ACDYM	=	000000	BIT07	=	000200	COR3	032054	DF133	=	070155	DF211	=	070361
ACENV	=	000000	BIT08	=	000400	COR33	032066	DF134	=	070155	DF212	=	070405
ACENVM	=	000000	BIT09	=	001000	COR4	032126	DF135	=	070105	DF213	=	070405
ACFATAL	=	000000	BIT1	=	000002	COR5	032166	DF136	=	070155	DF214	=	070405
AMADR1=	000000	BIT10	=	002000	COR6	032226	DF137	=	070131	DF215	=	070405	
AMADR2=	000000	BIT11	=	004000	COR7	032272	DF14	=	070012	DF216	=	070361	
AMADR3=	000000	BIT12	=	010000	COR8	032332	DF140	=	070131	DF217	=	070435	
AMADR4=	000000	BIT13	=	020000	COR9	032376	DF141	=	070131	DF22	=	070051	
AMAMS1=	000000	BIT14	=	040000	CPSPUR	036646	DF142	=	070131	DF220	=	070361	
AMAMS2=	000000	BIT15	=	100000	CPTWO	036664	DF143	=	070131	DF221	=	070361	

SYMBOL TABLE

DF222 = 070361	DF41 = 070061	DH12 = 066216	DH177 = 066557	DH255 = 066216
DF223 = 070361	DF42 = 070061	DH120 = 066557	DH2 = 066306	DH256 = 066216
DF224 = 070361	DF43 = 070061	DH121 = 066557	DH20 = 066216	DH257 = 066216
DF225 = 070361	DF44 = 070061	DH122 = 066557	DH200 = 066557	DH26 = 066216
DF226 = 070405	DF45 = 070061	DH123 = 066216	DH201 = 066557	DH260 = 066216
DF227 = 070405	DF46 = 070105	DH124 = 066216	DH202 = 066557	DH261 = 066216
DF23 = 070061	DF47 = 070105	DH125 = 066216	DH203 = 066557	DH262 = 066216
DF230 = 070405	DF5 = 070012	DH126 = 066216	DH204 = 066557	DH263 = 066216
DF231 = 070405	DF50 = 070105	DH127 = 066557	DH205 = 066557	DH264 = 066216
DF232 = 070405	DF51 = 070105	DH13 = 066216	DH206 = 066557	DH265 = 066216
DF233 = 070444	DF52 = 070105	DH130 = 066557	DH207 = 066716	DH266 = 067752
DF234 = 070461	DF53 = 070201	DH131 = 066557	DH21 = 066422	DH27 = 066216
DF235 = 070470	DF54 = 070201	DH132 = 066216	DH210 = 067006	DH3 = 066216
DF236 = 070461	DF55 = 070201	DH133 = 066557	DH211 = 067006	DH30 = 066216
DF237 = 070461	DF56 = 070201	DH134 = 066557	DH212 = 066716	DH31 = 066216
DF24 = 070061	DF57 = 070201	DH135 = 066216	DH213 = 066716	DH32 = 066216
DF240 = 070461	DF6 = 070012	DH136 = 066557	DH214 = 066716	DH33 = 066216
DF241 = 070444	DF60 = 070201	DH137 = 066557	DH215 = 066716	DH34 = 066216
DF242 = 070461	DF61 = 070201	DH14 = 066216	DH216 = 067006	DH35 = 066216
DF243 = 070444	DF62 = 070201	DH140 = 066557	DH217 = 067047	DH36 = 066216
DF244 = 070444	DF63 = 070201	DH141 = 066557	DH22 = 066511	DH37 = 066216
DF245 = 070444	DF64 = 070201	DH142 = 066557	DH220 = 066656	DH4 = 066216
DF246 = 070105	DF65 = 070201	DH143 = 066557	DH221 = 067006	DH4C = 066216
DF247 = 070474	DF66 = 070201	DH144 = 066557	DH222 = 066656	DH41 = 066216
DF25 = 070061	DF67 = 070201	DH145 = 066216	DH223 = 067006	DH42 = 066216
DF250 = 070474	DF7 = 070012	DH146 = 066216	DH224 = 066656	DH43 = 066216
DF251 = 070474	DF70 = 070233	DH147 = 066216	DH225 = 067006	DH44 = 066216
DF252 = 070502	DF71 = 070233	DH15 = 066216	DH226 = 067122	DH45 = 066216
DF253 = 070506	DF72 = 070233	DH150 = 066216	DH227 = 067211	DH46 = 066216
DF254 = 070506	DF73 = 070233	DH151 = 066557	DH23 = 066216	DH47 = 066216
DF255 = 070506	DF74 = 070233	DH152 = 066557	DH230 = 067122	DH5 = 066216
DF256 = 070506	DF75 = 070233	DH153 = 066557	DH231 = 067211	DH5C = 066216
DF257 = 070506	DF76 = 070233	DH154 = 066557	DH232 = 067211	DH51 = 066216
DF26 = 070061	DF77 = 070233	DH155 = 066557	DH233 = 067300	DH52 = 066216
DF260 = 070506	DH1 = 066216	DH156 = 066557	DH234 = 067341	DH53 = 066216
DF261 = 070506	DH10 = 066216	DH157 = 066216	DH235 = 067427	DH54 = 066216
DF262 = 070506	DH100 = 066216	DH16 = 066216	DH236 = 067467	DH55 = 066216
DF263 = 070506	DH101 = 066216	DH160 = 066557	DH237 = 067341	DH56 = 066216
DF264 = 070506	DH102 = 066216	DH161 = 066216	DH24 = 066216	DH57 = 066216
DF265 = 070506	DH103 = 066216	DH162 = 066216	DH240 = 067555	DH6 = 066216
DF266 = 070502	DH104 = 066216	DH163 = 066557	DH241 = 067300	DH60 = 066216
DF27 = 070061	DH105 = 066216	DH164 = 066216	DH242 = 067555	DH61 = 066216
DF3 = 070012	DH106 = 066216	DH165 = 066557	DH243 = 067300	DH62 = 066216
DF30 = 070061	DH107 = 066216	DH166 = 066557	DH244 = 067300	DH63 = 066216
DF31 = 070061	DH11 = 066216	DH167 = 066557	DH245 = 067300	DH64 = 066216
DF32 = 070061	DH110 = 066216	DH17 = 066347	DH246 = 066216	DH65 = 066216
DF33 = 070105	DH111 = 066216	DH170 = 066557	DH247 = 067645	DH66 = 066216
DF34 = 070105	DH112 = 066216	DH171 = 066557	DH25 = 066216	DH67 = 066216
DF35 = 070105	DH113 = 066216	DH172 = 066557	DH250 = 067712	DH7 = 066216
DF36 = 070105	DH114 = 066216	DH173 = 066557	DH251 = 067712	DH70 = 066216
DF37 = 070061	DH115 = 066557	DH174 = 066557	DH252 = 066306	DH71 = 066216
DF4 = 070012	DH116 = 066557	DH175 = 066557	DH253 = 066216	DH72 = 066216
DF40 = 070061	DH117 = 066557	DH176 = 066557	DH254 = 066216	DH73 = 066216

SYMBOL TABLE

DH74 = 066216
 DH75 = 066216
 DH76 = 066216
 DH77 = 066216
 DISPLA 001142
 DISPRE 000174
 DIVDSU 016630
 DIVDT 017060
 DIVFSU 015710
 DIVFT 016124
 JSWR = 177570
 DT10 = 070540
 DT100 = 071054
 DT101 = 071054
 DT102 = 071054
 DT103 = 071054
 DT104 = 071054
 DT105 = 071054
 DT106 = 071054
 DT107 = 071054
 DT11 = 070540
 DT110 = 071054
 DT111 = 070726
 DT112 = 070726
 DT113 = 070726
 DT114 = 070726
 DT115 = 071000
 DT116 = 071000
 DT117 = 071000
 DT12 = 070540
 DT120 = 071000
 DT121 = 071000
 DT122 = 071000
 DT123 = 070726
 DT124 = 070726
 DT125 = 070726
 DT126 = 070726
 DT127 = 071000
 DT13 = 070540
 DT130 = 071000
 DT131 = 071000
 DT132 = 070726
 DT133 = 071000
 DT134 = 071000
 DT135 = 070726
 DT136 = 071000
 DT137 = 071000
 DT14 = 070540
 DT140 = 071000
 DT141 = 071000
 DT142 = 071000
 DT143 = 071000

DT144 = 071000
 DT145 = 070726
 DT146 = 070726
 DT147 = 070726
 DT15 = 070540
 DT150 = 070726
 DT151 = 071000
 DT152 = 071000
 DT153 = 071000
 DT154 = 071000
 DT155 = 071000
 DT156 = 071000
 DT157 = 070726
 DT16 = 070540
 DT160 = 071000
 DT161 = 070726
 DT162 = 070726
 DT163 = 071000
 DT164 = 070726
 DT165 = 071142
 DT166 = 071142
 DT167 = 071142
 DT17 = 070624
 DT170 = 071142
 DT171 = 071142
 DT172 = 071142
 DT173 = 071142
 DT174 = 071142
 DT175 = 071142
 DT176 = 071142
 DT177 = 071142
 DT2 = 070576
 DT20 = 070644
 DT200 = 071142
 DT201 = 071142
 DT202 = 071142
 DT203 = 071142
 DT204 = 071142
 DT205 = 071142
 DT206 = 071142
 DT207 = 071252
 DT21 = 070666
 DT210 = 071230
 DT211 = 071252
 DT212 = 071324
 DT213 = 071324
 DT214 = 071324
 DT215 = 071324
 DT216 = 071252
 DT217 = 071406
 DT22 = 070710
 DT220 = 071252
 DT221 = 071252

DT222 = 071252
 DT223 = 071252
 DT224 = 071252
 DT225 = 071252
 DT226 = 071324
 DT227 = 071324
 DT23 = 070726
 DT230 = 071324
 DT231 = 071230
 DT232 = 071252
 DT233 = 071426
 DT234 = 071462
 DT235 = 071502
 DT236 = 071462
 DT237 = 071462
 DT24 = 070726
 DT240 = 071462
 DT241 = 071426
 DT242 = 071462
 DT243 = 071426
 DT244 = 071426
 DT246 = 070726
 DT247 = 071514
 DT25 = 070726
 DT250 = 071532
 DT251 = 071532
 DT252 = 071544
 DT253 = 071556
 DT254 = 071556
 DT255 = 071556
 DT256 = 071556
 DT257 = 071556
 DT26 = 070726
 DT260 = 071556
 DT261 = 071556
 DT262 = 071556
 DT263 = 071556
 DT264 = 071556
 DT265 = 071556
 DT266 = 071544
 DT27 = 070726
 DT3 = 070540
 DT30 = 070726
 DT31 = 070726
 DT32 = 070726
 DT33 = 070726
 DT34 = 070726
 DT35 = 070726
 DT36 = 070726
 DT37 = 070726
 DT4 = 070540
 DT40 = 070726

DT41 = 070726
 DT42 = 070726
 DT43 = 070726
 DT44 = 070726
 DT45 = 070726
 DT46 = 070726
 DT47 = 070726
 DT5 = 070540
 DT50 = 070726
 DT51 = 070726
 DT52 = 070726
 DT53 = 071054
 DT54 = 071054
 DT55 = 071054
 DT56 = 071054
 DT57 = 071054
 DT6 = 070540
 DT60 = 071054
 DT61 = 071054
 DT62 = 071054
 DT63 = 071054
 DT64 = 071054
 DT65 = 071054
 DT66 = 071054
 DT67 = 071054
 DT7 = 070540
 DT70 = 071054
 DT71 = 071054
 DT72 = 071054
 DT73 = 071054
 DT74 = 071054
 DT75 = 071054
 DT76 = 071054
 DT77 = 071054
 EEDOM 020100
 EEE1 017074
 EEE10 017470
 EEE11 017524
 EEE12 017560
 EEE13 017614
 EEE2 017130
 EEE3 017164
 EEE4 017220
 EEE5 017254
 EEE6 017310
 EEE7 017344
 EEE8 017400
 EEE9 017434
 EMTVEC= 000030
 EM1 037660
 EM10 040477
 EM100 053635
 EM101 053720

EM102 054100
 EM103 054310
 EM104 054422
 EM105 054567
 EM106 054676
 EM107 055005
 EM11 040604
 EM110 055114
 EM111 043731
 EM112 044006
 EM113 044064
 EM114 044142
 EM115 044221
 EM116 044277
 EM117 044356
 EM12 040713
 EM120 044514
 EM121 044652
 EM122 045007
 EM123 045144
 EM124 045222
 EM125 045301
 EM126 045357
 EM127 045436
 EM13 041022
 EM130 045514
 EM131 045573
 EM132 045731
 EM133 046067
 EM134 046225
 EM135 046362
 EM136 046517
 EM137 046654
 EM14 041136
 EM140 046742
 EM141 = 043731
 EM142 = 044006
 EM143 047031
 EM144 047107
 EM145 047166
 EM146 047333
 EM147 047500
 EM15 041250
 EM150 047644
 EM151 050010
 EM152 050076
 EM153 = 045144
 EM154 = 045222
 EM155 050165
 EM156 050243
 EM157 050322
 EM16 041362
 EM160 050467

EM161	050625	EM24	041756	EM57	051643	GGER6	010402	GG7	007144
EM162	050772	EM240	064141	EM6	040265	GGER7	010450	GG8	007212
EM163	051136	EM241	064175	EM60	051733	GGER8	010546	GG9	007222
EM164	051273	EM242	064300	EM61	052035	GGER9	010614	G-SWR =	104405
EM165	055224	EM243	064334	EM62	052231	GGGDON	025712	HHOATO	006606
EM166	055310	EM244	064437	EM63	052425	GGG1	024240	HHDONE	006756
EM167	055373	EM245	064476	EM64	052536	GGG10	025010	HHERO	005626
EM17	041457	EM246	043341	EM65	052655	GGG11	025060	HHERO0	005742
EM170	055425	EM247	064560	EM66	052770	GGG12	025130	HHER1	005674
EM171	055513	EM25	042054	EM67	053037	GGG13	025200	HHER10	006470
EM172	055636	EM250	064614	EM7	040372	GGG14	025250	HHER2	006010
EM173	055723	EM251	064646	EM70	053122	GGG2	024310	HHER3	006056
EM174	056071	EM252	064701	EM71	053153	GGG3	024360	HHER4	006124
EM175	056237	EM253	064772	EM72	053203	GGG4	024430	HHER5	006172
EM176	056351	EM254	065056	EM73	053203	GGG5	024500	HHER6	006240
EM177	056435	EM255	065143	EM74	053235	GGG6	024550	HHER7	006306
EM2	037712	EM256	065231	EM75	053344	GGG7	024620	HHER8	006354
EM20	041534	EM257	065320	EM76	053435	GGG8	024670	HHER9	006422
EM200	056520	EM26	042141	EM77	053525	GGG9	024740	HHHDON	030020
EM201	056552	EM260	065406	ERM10	054156	GGP1	011564	HHH1	025716
EM202	056641	EM261	065475	ERRVEC=	000004	GGP2	011574	HHH10	027016
EM203	057003	EM262	065565	ERTYPE	036140	GGP3	011604	HHH11	027116
EM204	057145	EM263	065656	ERT1	036324	GGP4	011614	HHH12	027216
EM205	057233	EM264	065743	ERT2	036560	GGP5	011624	HHH13	027316
EM206	057401	EM265	066030	ERT3	036564	GGP6	011634	HHH2	026016
EM207	057547	EM266	066115	ERT4	036576	GGP7	011644	HHH3	026116
EM21	041566	EM27 =	042054	ERT5	036610	GGP8	011654	HHH4	026216
EM210	057511	EM3	037746	FFFDON	020630	GGP9	011664	HHH5	026316
EM211	057653	EM30	042237	FFF1	020104	GG1	006762	HHH6	026416
EM212	060017	EM31	042311	FFF2	020160	GG10	007256	HHH7	026516
EM213	060203	EM32	041724	FFF3	020234	GG11	007314	HHH8	026616
EM214	060367	EM33	042356	FFF4	020310	GG12	007336	HHH9	026716
EM215	060553	EM34	042401	FPSPUR	036614	GG13	007346	HHP0	006616
EM216	060737	EM35	042433	FPVECT=	000244	GG14	007364	HHP1	006626
EM217	061121	EM36	042506	GGOATO	011554	GG15	007422	HHP10	006736
EM22	041620	EM37	042554	GGDONE	011674	GG16	007440	HHP11	006746
EM220	061163	EM4	040053	GGERO	010052	GG17	007506	HHP2	006636
EM221	061413	EM40	042577	GGER1	010150	GG18	007516	HHP3	006646
EM222	061657	EM41	042631	GGER10	010662	GG19	007552	HHP4	006656
EM223	062110	EM42	042704	GGER11	010730	GG2	007020	HHP5	006666
EM224	062355	EM43	043035	GGER12	010776	GG20	007610	HHP6	006676
EM225	062606	EM44	043166	GGER13	011044	GG21	007632	HHP7	006706
EM226	063053	EM45	043234	GGER14	011112	GG22	007642	HHP8	006716
EM227	063206	EM46	043307	GGER15	011210	GG23	007660	HHP9	006726
EM23	041701	EM47	043364	GGER16	011256	GG24	007716	HHTRAP	006536
EM230	063341	EM5	040160	GGER17	011324	GG25	007734	HH1	005036
EM231	063475	EM50	043520	GGER18	011372	GG26	010002	HH10	005250
EM232 =	057611	EM51	043573	GGER19	011440	GG27	010012	HH11	005266
EM233	063631	EM52	043641	GGER2	010216	GG28	010046	HH12	005316
EM234	063670	EM53	051471	GGER20	011506	GG3	007042	HH13	005340
EM235	063754	EM54	051522	GGER3	010264	GG4	007052	HH14	005360
EM236	064022	EM55	051437	GGER4	010332	GG5	007070	HH15	005370
EM237	064055	EM56	051552	GGERS	010334	GG6	007126	HH16	005376

HH17	005414	HX17	012264	MMM2	030076	OVFTT	023262	SW4	=	000020	
HH18	005452	HX18	012322	MMM3	030150	OVUNDN	022020	SW5	=	000040	
HH19	005474	HX19	012342	MMM4	030222	OVUNDT	023622	SW6	=	000100	
HH2	005066	HX2	011730	MMM5	030274	OVUNFN	021060	SW7	=	000200	
HH20	005504	HX20	012352	MNUMBE=	000266	OVUNFT	022660	SW8	=	000400	
HH21	005522	HX21	012354	MODDD0	031754	PIRQ	=	177772	SW9	=	001000
HH22	005552	HX22	012404	MODDD1	031764	PIQVE=	000240	TAB	=	000011	
HH23	005574	HX23	012424	MODDDOV	031366	POWERM	037104	TBITVE=	000014		
HH24	005604	HX24	012444	MODDSU	027422	PRUGNU=	000002	TKVEC	=	000060	
HH25	005622	HX25	012454	MODDT0	030000	PRO	=	000000	TPVEC	=	000064
HH3	005110	HX26	012462	MODDT1	030010	PR1	=	000040	TRAPVE=	000034	
HH4	005130	HX27	012464	MODF00	030726	PR2	=	000100	TRTVEC=	000014	
HH5	005140	HX28	012516	MODF01	030736	PR3	=	000140	TST1	005034	
HH6	005146	HX29	012540	MODFOV	030352	PR4	=	000200	TST10	017072	
HH7	005160	HX3	011744	MODFSU	025324	PR5	=	000240	TST11	020102	
HH8	005216	HX30	012550	MODFT0	025662	PR6	=	000300	TST12	020632	
HH9	005240	HX31	012566	MODFT1	025672	PR7	=	000340	TST13	021472	
HT	=	HX32	012616	MODP1	025702	PS	=	177776	TST14	022432	
HXDATO	013304	HX33	012640	MSA1	036760	PSW	=	177776	TST15	023274	
HXDONE	013414	HX34	012650	MSA2	036776	PWRVEC=	000024	TST16	024236		
HXER1	012672	HX35	012666	MSA3	037021	ROCHR	=	104407	TST17	025714	
HXER10	013162	HX4	011764	MSA4	037033	RESREG=	104411	TST2	006760		
HXER11	013214	HX5	012004	MSA5	037051	RESVEC=	000010	TST20	030022		
HXER12	013234	HX6	012014	MSA6	037064	RSETUP=	104412	TST21	030750		
HXER13	013264	HX7	012022	MS1	037156	R6	=	%000006	TST22	031776	
HXER2	012732	HX8	012040	MS10	037355	R7	=	%000007	TST23	033150	
HXER22	012746	HX9	012072	MS11	037377	SAVREG=	104410	TST3	011676		
HXER3	012762	IIIDON	021470	MS12	037423	SPACE	037153	TST4	013416		
HXER33	012776	III1	020634	MS2	037174	STACK	=	001100	TST5	014424	
HXER4	013012	III2	020700	MS3	037214	START	004336	TST6	015126		
HXER5	012712	III3	020744	MS37	037446	STKLMT=	177774	TST7	016136		
HXER6	013046	III4	021010	MS4	037243	SWR	001140	TYPE	=	104401	
HXERE6	013062	IOTVEC=	000020	MS40	037464	SWREG	000176	TYPOC	=	104402	
HXER7	013076	JJJDON	022430	MS41	037520	SWO	=	000001	TYPON	=	104404
HXER8	013030	JJJ1	021474	MS415	037500	SWO0	=	000001	TYPOS	=	104403
HXER9	013126	JJJ2	021560	MS42	037544	SWO1	=	000002	\$APTHO	004322	
HXP1	013314	JJJ3	021644	MS43	037562	SWO2	=	000004	\$ATYC	035010	
HXP2	013324	JJJ4	021730	MS44	037577	SWO3	=	000010	\$ATY1	034764	
HXP3	013334	KKKDON	023272	MS5	037271	SWO4	=	000020	\$ATY3	034772	
HXP4	013344	KKK1	022434	MS6	037313	SWO5	=	000040	\$ATY4	035002	
HXP5	013354	KKK3	022500	MS7	037331	SWO6	=	000100	\$AUTOB	001134	
HXP6	013364	KKK4	022544	MULDSU	020370	SWO7	=	000200	\$BASE	001372	
HXP7	013374	KKK5	022610	MULDT	020620	SWO8	=	000400	\$BDAOR	001122	
HXP8	013404	LF	=	MULFSU	017654	SWO9	=	001000	\$BODAT	001126	
HX1	011700	LLLLDON	024234	MULFT	020070	SW1	=	000002	\$BELL	001306	
HX10	012106	LLL1	023276	NMNDON	031774	SW10	=	002000	\$CDW1	001376	
HX11	012130	LLL2	023362	NNN1	030752	SW11	=	004000	\$CDW2	001400	
HX12	012150	LLL3	023446	NNN2	031054	SW12	=	010000	\$CHARC	034532	
HX13	012160	LLL4	023532	NNN3	031156	SW13	=	020000	\$CKSWR	035232	
HX14	012166	LOOP	005034	NNN4	031260	SW14	=	040000	\$CLR.T	033352	
HX15	012204	LPERR	=	OVDNTT	022420	SW15	=	!00000	\$CMTAG	001100	
HX16	012232	MMMDON	030746	OVDTT	024224	SW2	=	000004	\$CM1	=	000024
HX165	012236	MMM1	030024	OVFNNT	021460	SW3	=	000010	\$CM2	=	000050

\$CH3 = 000024	\$ERTYL 001112	\$MSGTY 001316	\$REG3 001170	\$TMP20 001272
\$CH4 = 000024	\$ESCAP 001304	\$MSWR 035646	\$REG4 001172	\$TMP21 001274
\$CNTLG 035641	\$ETABL 001336	\$MTYP1 001347	\$REG5 001174	\$TMP22 001276
\$CNTLL 035634	\$ETEND 001442	\$MTYP2 001353	\$REG6 001176	\$TMP23 001300
\$CPUOP 001344	\$FATAL 001320	\$MTYP3 001357	\$REG7 001200	\$TMP3 001240
\$CRLF 001313	\$FFLG 035230	\$MTYP4 001363	\$RESRE 034216	\$TMP4 001242
\$DOHC 001402	\$FILLC 001156	\$MXCNT 033720	\$RTNAD 033432	\$TMP5 001244
\$DOHI 001404	\$FILLS 001155	\$NULL 001154	\$RTAN 033426	\$TMP6 001246
\$DOHIC 001426	\$GDAOR 001120	\$NWTST= 000001	\$SAVRE 034160	\$TMP7 001250
\$DOHII 001430	\$GDOAT 001124	\$OCNT 034760	\$SAVR6 036136	\$TN = 000023
\$DOHIZ 001432	\$GET42 033334	\$OMODE 034762	\$SCOPE 033442	\$TPB 001152
\$DOHJ3 001434	\$GTSMR 035302	\$OVER 033704	\$SETUP= 000137	\$TPFLG 001157
\$DOHJ4 001436	\$HC = 000003	\$PASS 001324	\$STUP = 177777	\$TPS 001150
\$DOHJ5 001440	\$HIBTS 004322	\$PASTM 004330	\$SVLAD 033650	\$TRAP 035670
\$DOHJ2 001406	\$ICNT 001104	\$PWAD 036114	\$SVPC = 004322	\$TRAP2 035712
\$DOHJ3 001410	\$ILLUP 036132	\$PWADN 035754	\$SWR = 177400	\$TTP = 000014
\$DOHJ4 001412	\$INTAG 001135	\$PWARG 036110	\$SWREG 001340	\$TRPAD 035724
\$DOHJ5 001414	\$ITEMB 001114	\$PWUP 036026	\$SWRMK= 000000	\$TSTM 004326
\$DOHJ6 001416	\$LF 001314	\$QUES 001312	\$SWRMS= 000200	\$TSTM 001102
\$DOHJ7 001420	\$FLG 035227	\$RDCHR 035514	\$TAB 037151	\$TYPE 034254
\$DOHJ8 001422	\$LOOP 033430	\$RDSZ = 000001	\$TBIT 033434	\$TYPEC 034466
\$DOHJ9 001424	\$LPOR 001106	\$REGAD 001160	\$TERM = 000030	\$TYPEX 034534
\$DEVY 001326	\$LPERR 001110	\$REGO 001162	\$TESTN 001322	\$TYPOC 034562
\$DEVX 001374	\$MAOR1 001350	\$REG1 001164	\$TIMES 001302	\$TYPON 034576
\$DOAGN 033372	\$MAOR2 001354	\$REG10 001202	\$TKB 001146	\$TYPOS 034536
\$ENDAC 033362	\$MAOR3 001360	\$REG11 001204	\$TKS 001144	\$UNIT 001330
\$ENDCT 033204	\$MAOR4 001364	\$REG12 001206	\$TMP0 001232	\$UNITM 004332
\$ENULL 033436	\$MAIL 001316	\$REG13 001210	\$TMP1 001234	\$USWR 001342
\$ENV 001336	\$MAMS1 001346	\$REG14 001212	\$TMP10 001252	\$VECT1 001366
\$ENVX 001337	\$MAMS2 001352	\$REG15 001214	\$TMP11 001254	\$VECT2 001370
\$EOP 033150	\$MAMS3 001356	\$REG16 001216	\$TMP12 001256	\$XTSTR 033454
\$EOPCY 033176	\$MAMS4 001362	\$REG17 001220	\$TMP13 001260	\$SET4= 000001
\$ERFLG 001103	\$MBAOR 004324	\$REG2 001166	\$TMP14 001262	\$OFILL 034761
\$ERMAX 001115	\$FLG 035226	\$REG20 001222	\$TMP15 001264	. = 071644
\$ERROR 033722	\$MNEW 035657	\$REG21 001224	\$TMP16 001266	.LPER 036702
\$ERRPC 001116	\$MSGAD 001332	\$REG22 001226	\$TMP17 001270	.RSET 036710
\$ERRTB 001442	\$MSGLG 001334	\$REG23 001230	\$TMP2 001236	.SX = 004322

ABS. 071644 000

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

DSKZ:DFPBA.BIN,DSKZ:DFPBA.SEG/SOL+DFPBA.P11
RUN-TIME: 85 71 5 SECONDS
RUN-TIME RATIO: 215/162=1.3
CORE USED: 31K (61 PAGES)