

# KE11-F

INSTR TEST  
MD-11-DBKEA-B

EP DBKEA-B-DL-A  
COPYRIGHT © 1976  
FICHE 1 OF 1

NOV 1976  
**digital**  
MADE IN USA

The microfiche card contains a grid of 48 frames, arranged in 8 rows and 6 columns. Each frame displays a small table or chart, likely representing test data or a log. The text within the frames is too small to read clearly but appears to consist of columns of numbers and some headers. The frames are separated by a grid pattern.



IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DBKEA-B-D  
PRODUCT NAME: KE11F (PDP-11 FIS) INSTRUCTION TESTS  
DATE CREATED: APRIL 1973  
MAINTAINER: DIAGNOSTIC GROUP  
AUTHOR: KEN CHAPMAN

COPYRIGHT (C) 1972, 1973  
DIGITAL EQUIPMENT CORPORATION  
MAYNARD, MASSACHUSETTS 01754  
.subtitle Table of Contents



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/or operator action
- 5. OPERATING PROCEDURE
  - 5.1 Operational switch settings
  - 5.2 Subroutine abstracts
- 6. ERRORS
  - 6.1 Error printout
  - 6.2 Error recovery
  - 6.3 Error counter
- 7. RESTRICTIONS
- 8. MISCELLANEOUS
  - 8.1 Execution time
  - 8.2 Stack pointer
  - 8.3 Pass counter
  - 8.4 Power fail
- 9. PROGRAM DESCRIPTION



.indent -10 1. Abstract

This program tests the KE11F (PDP-11 Floating Instruction Set (FADD, FSUB, FMUL, and FDIV)) option with fixed number patterns, using each register at least once as the stack pointer. It also checks stack overflow and that the floating instructions can be interrupted (by the console teletype). The program should be run for at least 2 passes with all switches down.

2. REQUIREMENTS

2.1 Equipment

PDP-11 (KD11A) standard computer with KE11F option

2.2 Storage

Program Storage - the routines use memory 0 - 17500

2.3 Preliminary programs

None

3. LOADING PROCEDURE

Use standard procedure for ABS tapes.

4. STARTING PROCEDURE

4.1 Control switch settings

See 5.1.1 (all down for worst case testing)

4.2 Starting address

The program should always be started at 200.

4.3 Program and/or operator action .blank 1 1) Load program into memory using ABS loader.  
2) Load address 200.  
3) Set switches (see sec 5.1.1) All down for worst case  
4) Press start.  
5) The interrupt test section will type three random length lines of @'s on the console teletype every pass.



- 6) The program will loop and bell will ring once every pass.  
7) A minimum of two passes should always be run.  
141 .blank 2

5. OPERATING PROCEDURE

5.1 Operational switch settings

At SA 200, all switches down is worst case testing. Each subtest will be looped upon until completion of 256 passes of that subtest. The bell will ring upon completion of a pass of the entire program. Alternate pass will run with the T-bit set.

5.1.1 Switch settings are:

SW<15> = 1 ..... HALT ON ERROR  
SW<14> = 1 ..... SCOPE LOOP  
SW<13> = 1 ..... INHIBIT PRINTOUT  
SW<12> = 1 ..... INHIBIT TRACE TRAPPING  
SW<11> = 1 ..... INHIBIT ITERATIONS OF SUBTEST  
SW<10> = 1 ..... BELL ON ERROR  
          0 ..... BELL ON PASS COMPLETE  
SW<09> = 1 ..... LOOP ON ERROR  
SW<08> = 1 ..... LOOP ON TEST IN SW<7:0>

Caution: SW<8:0> are also used for ROM word match with KM11 maintenance card.

5.2 Subroutine Abstracts

5.2.1 SCOPE

This subroutine call (via a TRAP instruction) is placed between each subtest in the instruction section. It records the starting address of each subtest as it is being entered in location "LADS". If a scope loop is requested, the current subtest will be looped upon. SW<11> on a 1 inhibits iteration of subtests. The contents of "LADS" may be used to determine the last subtest successfully completed.

5.2.2 HLT

This routine (called by an EMT instruction) prints out an error message (See 6.1.) If SW<9> is on a 1 and a HLT is executed, the subtest will be looped upon until 256 consecutive good passes are completed. To inhibit typeouts, put SW<13> on a 1. To ring the bell on an error, put SW<10> on a 1.



5.2.3 NOP

A NOP is placed just before each FIS instruction. This allows the operator to patch in a HALT for debugging purposes.

5.2.4 TRTRAP

If SW<12> is on a 0, the T-bit will be set on alternate passes. When the T-bit is set, the processor traps after each instruction. The first instruction executed upon trapping is an "RTT" which returns to the interrupted sequence of instructions. This sequence is continued until the end of the program is reached.

5.2.5 TRAPCATCHER

A ".+2" - "HALT" sequence is repeated from 0 - 776 to catch any unexpected traps. Thus any unexpected traps or interrupts will HALT at the vector + 2.

5.2.6 FLOATING ERROR TRAP (to 244) .blank 1 If a floating point error (overflow, underflow, or divide by zero) was expected, the vector will point to a unique ISR within the subtest where the error occurred which checks the data on the stack(s). If an error was not anticipated, an erroneous trap will be detected in TRAPER.

6. ERRORS

6.1 Error printout

The format is as follows:

ADR PS SP ANS1 ANS2 ANS3 ANS4 ANS5 ANS6

Where:

ADR = Address of error HLT  
PS = Processor Status  
SP = Contents of Stack Pointer Register  
ANS1-6 = Error data read from the STACK(s). From 0 to 6 of these may be typed depending on the number following the HLT; e.g., HLT+3 would type ANS1 thru ANS3, HLT (by itself) would

To find the failing test, look at the listing above the address typed. In most cases the comment beside the HLT tells what was being checked and what was expected.



6.2 Error recovery

Restart at 200 .blank 2  
6.3 Error counter

An error count is kept in "ERRORS" (LOC 1002). It can only be cleared from the console or by reloading the program.

7. RESTRICTIONS

None

8. MISCELLANEOUS

8.1 Execution time

Due to the random characteristic of the interrupt tests, the execution time can be half a minute or more. However, normally a bell will ring within 15 seconds with all switches down.

8.2 Stack Pointer

Stack is initially set to 500

8.3 Pass count

A 32 bit (2 words) pass count is kept in "PASSES" (LOC 1004,1006). It can only be cleared from the console or by reloading the program.

8.4 Power Fail

Each test can be power failed with no errors. To use, start the test as usual and power down then up at any time. The program should type "POWER" and continue to run from where the power fail inter

9. PROGRAM DESCRIPTION

This program tests all the instructions of the KE11F (FADD, FSUB, FMUL, and FDIV) ALL REGISTERS ARE CHECK TO SEE IF THEY FUNCTION PROPERLY AS THE STACK POINTER. THE PROGRAM HAS MANY SUBTESTS (THE CODE BETWEEN 2 SCOPE statements) which are run 256 times before continuing to the next. dbkea-b ke11f (pdg-11 fis) instruction tests. macyl 27(732) 20-sep-76 13:41 page 9 SW<11> on a 1 causes each subtest to



be run only once. SW(9) on a 1 enables loop on error. The address ICNT (LOC 1000) contains the iteration count in the left byte and the test number in the right byte. All the subtests should be run sequentially by starting at 200 not by starting at the beginning of the subtest. To loop on a particular subtest, put the test number (see listing) in the right byte of the switch register and SW(8) on a 1. This test will be looped upon until SW(8) is put on a 0 or the right byte is changed. If the test is non-existent, the program will be run as usual.

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

```
.TITLE MAINDEC-11-DBKEA-B      KE11F (PDP-11 FIS) INSTRUCTION TESTS.
.ENABL ABS
;COPYRIGHT 1972, DIGITAL EQUIPMENT CORP., MAYNARD, MASS
;PROGRAM BY KEN CHAPMAN
```

SWITCH	USE
8	LOOP ON TEST IN SW<7:0>
9	LOOP ON ERROR
10	0 - BELL ON PASS COMPLETE 1 - BELL ON ERROR
11	INHIBIT ITERATIONS
12	INHIBIT TRACE TRAP
13	INHIBIT ERROR TYPEOUTS
14	LOOP ON TEST
15	HALT ON ERROR

ERROR MESSAGE FORMAT:

```
ADR PSW SP ANS1 ANS2 ANS3 ANS4 ANS5 ANS6
```

WHERE ADR = ADDRESS OF "HLT" INSTRUCTION + 2  
PSW = PROCESSOR STATUS WORD  
SP = STACK POINTER

ANS1 THRU ANS6 = DATA OFF THE STACK(S)  
NOTE: ANS1 THRU ANS6 ARE NOT ALWAYS TYPED, DEPENDING ON THE NUMBER ADDED TO THE "HLT". "HLT" ALONE TYPES NONE, "HLT+1" TYPES ANS1, "HLT+2" TYPES ANS1 AND ANS2, ETC.

104400  
104000  
000004  
177776  
177570  
177570  
000007  
000000  
000001  
000002  
000003  
000004  
000005  
000005  
000006

```
SCOPE= TRAP
HLT= EMT
TYPE= IOT
PS= 177776
SWR= 177570
DISPLAY=SWR
BELL= 7
RO= %0
R1= %1
R2= %2
R3= %3
R4= %4
R5= %5
TTY= %5
SP= %6
```



387	000007	PC=	%7
388	100000	SW15=	100000
389	040000	SW14=	40000
390	020000	SW13=	20000
391	010000	SW12=	10000
392	004000	SW11=	4000
393	002000	SW10=	2000
394	001000	SW09=	1000
395	000400	SW08=	400



```

396          000000          . = 0          ;TRAP CATCHER FROM 0 - 776
397
398          000200          . = 200
399
400 000200 000167 000604          JMP BEGIN          ;JUMP TO STARTING ADDRESS OF PROGRAM
401
402          000600          . = 600
403 000600 000000          $PSW: 0          ;PROCESSOR STATUS WORD
404 000602 000000          $SP: 0          ;STACK POINTER
405 000604 000000          ANS1: 0          ;FIRST ANSWER (SEE CODE)
406 000606 000000          ANS2: 0
407 000610 000000          ANS3: 0
408 000612 000000          ANS4: 0
409 000614 000000          ANS5: 0
410 000616 000000          ANS6: C
411 000620 000000 000000 000000          0,0,0,0          ;NON-%6 STACK BUFFER
412 000626 000000
413 000630 000000          STACK0: 0          ;NON-%6 STACK NORMAL LIMIT
414 000632 000000          STACK2: 0
415 000634 000000          STACK4: 0
416 000636 000000          STACK6: 0
417 000640 000000 000000 000000          STACK8: 0,0,0,0          ;NON-%6 STACK BUFFER
418 000646 000000
419          000631          STACK1 = STACK0+1
420
421 000650 000244          FISVEC: 244          ;FIS TRAP VECTOR ADDRESS
422 000652 000246          FISLVL: 246
423
424 000654 177564          TPS: 177564          ;TELEPRINTER STATUS
425 000656 177566          TPB: 177566          ;TELEPRINTER BUFFER
426
427          001000          . = 1000
428 001000 000000          ICNT: 0          ;ITERATION COUNT - LH TEST NO. - RH
429 001002 000000          ERRORS: 0          ;ERROR COUNT
430 001004 000000 000000          PASSES: 0,0          ;PASS COUNTER
431
432 001010 000005          BEGIN: RESET
433 001012 012706 000500          MOV #500, SP
434 001016 012737 015772 000014          MOV #YESRT, @#14          ;SET TRACE TRAP VECTOR
435 001024 012777 017124 016250          MOV #PDOWN$, @PDVECS          ;SET UP POWER FAIL VECTOR
436 001032 012777 000340 016244          MOV #340, @PDVECS+2
437 001040 012737 017322 000020          MOV #.IOT, @#20          ;SET UP VECTOR 20
438 001046 012700 000030          MOV #30, RO          ;SET RO TO VECTOR 30
439 001052 012720 016600          MOV #HLTS, (0)+          ;SET EMT VECTOR
440 001056 012720 000340          MOV #340, (0)+
441 001062 012720 015774          MOV #SCOPES, (0)+          ;SET TRAP VECTOR
442 001066 012710 000340          MOV #340, (0)
443 001072 012737 000006 000004 15:          MOV #6, @#4          ;RESTORE TIME-OUT VECTOR
444 001100 005067 177674          CLR ICNT
445 001104 005067 015016          CLR LADS          ;CLEAR LOOP ADDRESS
446 001110 012767 000377 015012          MOV #377, TIMES          ;INITIALIZE NUMBER OF ITERATIONS
447 001116 104400          SCOPE

```

```

448
449
450
451
452
453
454
455 001120 004567 015164
456 001124 000000 000000
457 001130 000000 000000
458 001134 000000
459 001136 016574 000340
460 001142 012700 000630
461
462 001146 000240
463 001150 075000
464
465 001152 004767 015164
466 001156 010067 177420
467 001162 022767 000004 177410
468 001170 001401
469 001172 104000
470
471 001174 022767 000634 177400
472 001202 001401
473 001204 104000
474
475 001206 005767 177372
476 001212 001401
477 001214 104002
478
479 001216 005767 177364
480 001222 001401
481 001224 104002
482
483 001226 122767 000001 177544 END1:
484 001234 001401
485 001236 104000
486
487 001240 104400
488

```

```

*****
TEST 1: FADD (KE11F FLOATING ADD INSTRUCTION)
000000,000000 + 000000,000000 = 000000,000000
PS = 004, STACK POINTER = R0
*****
TST1: JSR R5, PUSHR ;PUSH 4 WORDS ONTO R0 STACK, SET PRIORITY
.WORD 000000,000000 ;SECOND OPERAND ON TOP
.WORD 000000,000000 ;FIRST OPERAND ON BOTTOM
.WORD 000 ;PROCESSOR PRIORITY LEVEL
.WORD TRAPER,340 ;FIS TRAP VECTOR
MOV #STACK0,R0 ;SET UP STACK POINTER

NOP
FADD+ R0 ;FLOATING ADD ON THE R0 STACK

JSR PC, POPR ;POP THE ANSWER
MOV R0, SSP ;SAVE "STACK POINTER"
CMP #004, SPSW ;CHECK PS (EXCEPT T BIT)
BEQ .+4 ;BRANCH IF OK
HLT ;PS NOT EQUAL TO 004

CMP #STACK4,SSP ;CHECK THE STACK POINTER (R0)
BEQ .+4 ;BRANCH IF OK
HLT ;STACK POINTER (R0) NOT EQUAL TO #STACK4

TST ANS1 ;CHECK FIRST HALF OF ANSWER
BEQ .+4 ;BRANCH IF OK
HLT+2 ;ANS1 NOT EQUAL TO 000000

TST ANS2 ;CHECK SECOND HALF OF ANSWER
BEQ .+4 ;BRANCH IF OK
HLT+2 ;ANS2 NOT EQUAL TO 000000

CMPB #1, ICNT ;CHECK THE TEST NUMBER
BEQ .+4 ;BRANCH IF OK
HLT ;WRONG TEST! PC MUST HAVE FOULED UP.

SCOPE

```



489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504  
505  
506  
507  
508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518  
519  
520  
521  
522  
523  
524  
525  
526  
527  
528  
529

```
*****
:TEST 2:      FADD (KE11F FLOATING ADD INSTRUCTION)
:      040200,000000 + 040200,000000 = 040400,000000
:      PS = 040,      STACK POINTER = R1
*****
```

```
TST2:  JSR      R5,      PUSHR      ;PUSH 4 WORDS ONTO R1 STACK, SET PRIORITY
        .WORD   040200,000000      ;SECOND OPERAND ON TOP
        .WORD   040200,000000      ;FIRST OPERAND ON BOTTOM
        .WORD   040                ;PROCESSOR PRIORITY LEVEL
        .WORD   TRAPER,340         ;FIS TRAP VECTOR
        MOV     #STACK0,R1         ;SET UP STACK POINTER

        NOP
        FADD+  R1                  ;FLOATING ADD ON THE R1 STACK

        JSR     PC,      POPR      ;POP THE ANSWER
        MOV     R1,      $SP       ;SAVE "STACK POINTER"
        CMP     #040,    $PSW      ;CHECK PS (EXCEPT T BIT)
        BEQ    .+4              ;BRANCH IF OK
        HLT
        ;PS NOT EQUAL TO 040

        CMP     #STACK4,$SP       ;CHECK THE STACK POINTER (R1)
        BEQ    .+4              ;BRANCH IF OK
        HLT
        ;STACK POINTER (R1) NOT EQUAL TO #STACK4

        CMP     #040400,ANS1      ;CHECK FIRST HALF OF ANSWER
        BEQ    .+4              ;BRANCH IF OK
        HLT+2
        ;ANS1 NOT EQUAL TO 040400

        TST    ANS2
        BEQ    .+4              ;CHECK SECOND HALF OF ANSWER
        HLT+2
        ;BRANCH IF OK
        ;ANS2 NOT EQUAL TO 000000

        END2:  CMPB    #2,      ICNT ;CHECK THE TEST NUMBER
        BEQ    .+4              ;BRANCH IF OK
        HLT
        ;WRONG TEST! PC MUST HAVE FOULED UP.

        SCOPE
```

177266

177256

177246

177240

177420

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

\*\*\*\*\*  
:TEST 3: FADD (KE11F FLOATING ADD INSTRUCTION)  
: 077777,177777 + 177777,177777 = 000000,000000  
: PS = 104, STACK POINTER = R2  
:\*\*\*\*\*

TST3: JSR R5, PUSHR ;PUSH 4 WORDS ONTO R2 STACK, SET PRIORITY  
;SECOND OPERAND ON TOP  
.WORD 177777,177777  
;FIRST OPERAND ON BOTTOM  
.WORD 077777,177777  
;PROCESSOR PRIORITY LEVEL  
.WORD 100  
;FIS TRAP VECTOR  
MOV #STACK0,R2 ;SET UP STACK POINTER  
  
NOP  
FADD+ R2 ;FLOATING ADD ON THE R2 STACK  
  
JSR PC, POPR ;POP THE ANSWER  
MOV R2, SSP ;SAVE "STACK POINTER"  
CMP #104, SPSW ;CHECK PS (EXCEPT T BIT)  
BEQ .+4 ;BRANCH IF OK  
HLT ;PS NOT EQUAL TO 104  
  
CMP #STACK4,SSP ;CHECK THE STACK POINTER (R2)  
BEQ .+4 ;BRANCH IF OK  
HLT ;STACK POINTER (R2) NOT EQUAL TO #STACK4  
  
TST ANS1 ;CHECK FIRST HALF OF ANSWER  
BEQ .+4 ;BRANCH IF OK  
HLT+2 ;ANS1 NOT EQUAL TO 000000  
  
TST ANS2 ;CHECK SECOND HALF OF ANSWER  
BEQ .+4 ;BRANCH IF OK  
HLT+2 ;ANS2 NOT EQUAL TO 000000  
  
END3: CMPB #3, ICNT ;CHECK THE TEST NUMBER  
BEQ .+4 ;BRANCH IF OK  
HLT ;WRONG TEST! PC MUST HAVE FOULED UP.  
  
SCOPE



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

```
*****
;TEST 4:          FADD (KE11F FLOATING ADD INSTRUCTION)
;          052525 052525 + 152525 052524 = 044600,000000
;          PS = 200,          STACK POINTER = SP
*****
```

```
TST4:  JSR      R5,      PUSH5      ;PUSH 4 WORDS ONTO STACK, SET PRIORITY
        .WORD   152525,052524      ;SECOND OPERAND ON TOP
        .WORD   052525,052525      ;FIRST OPERAND ON BOTTOM
        .WORD   217                ;PROCESSOR PRIORITY LEVEL
        .WORD   TRAPER,340         ;FIS TRAP VECTOR

        NOP
        FADD+   SP                ;FLOATING ADD ON THE STACK

        JSR      PC,      POPS      ;POP THE ANSWER
        CMP      #500,    SP        ;CHECK THE STACK POINTER
        BEQ     TSA4      ;BRANCH IF OK
        MOV      #500,    SP        ;RESTORE STACK POINTER
        HLT     ;STACK POINTER FOULED UP
        BR      END4          ;SKIP REST OF TEST

        CMP      #200,    $PSW      ;CHECK PS (EXCEPT T BIT)
        BEQ     .+4         ;BRANCH IF OK
        HLT     ;PS NOT EQUAL TO 200

        CMP      #044600,ANS1      ;CHECK FIRST HALF OF ANSWER
        BEQ     .+4         ;BRANCH IF OK
        HLT+2  ;ANS1 NOT EQUAL TO 044600

        TST     ANS2          ;CHECK SECOND HALF OF ANSWER
        BEQ     .+4         ;BRANCH IF OK
        HLT+2  ;ANS2 NOT EQUAL TO 000000

        CMPB    #4,      ICNT      ;CHECK THE TEST NUMBER
        BEQ     .+4         ;BRANCH IF OK
        HLT     ;WRONG TEST! PC MUST HAVE FOULED UP.

        SCOPE
```

612  
613  
614  
615  
616  
617  
618  
619  
620  
621  
622  
623  
624  
625  
626  
627  
628  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639  
640  
641  
642  
643  
644  
645  
646  
647  
648  
649  
650  
651  
652

```

*****
:TEST 5:      FADD (KE11F FLOATING ADD INSTRUCTION)
:      125200,000000 + 025177,177777 = 117200,000000
:      PS = 310,      STACK POINTER = SP
*****
    
```

```

TSTS:  JSR      R5,      PUSH5      ;PUSH 4 WORDS ONTO STACK, SET PRIORITY
        .WORD   025177,177777      ;SECOND OPERAND ON TOP
        .WORD   125200,000000      ;FIRST OPERAND ON BOTTOM
        .WORD   307                  ;PROCESSOR PRIORITY LEVEL
        .WORD   TRAPER,340          ;FIS TRAP VECTOR

        NOP
        FADD+   SP                  ;FLOATING ADD ON THE STACK

        JSR      PC,      POPS       ;POP THE ANSWER
        CMP      #500,     SP        ;CHECK THE STACK POINTER
        BEQ      TSAS,     SP        ;BRANCH IF OK
        MOV      #500,     SP        ;RESTORE STACK POINTER
        HLT      TSAS,     SP        ;STACK POINTER FOULED UP
        BR       ENDS              ;SKIP REST OF TEST

        CMP      #310,     SPSW      ;CHECK PS (EXCEPT T BIT)
        BEQ      .+4          ;BRANCH IF OK
        HLT      .+4          ;PS NOT EQUAL TO 310

        CMP      #117200,ANS1       ;CHECK FIRST HALF OF ANSWER
        BEQ      .+4          ;BRANCH IF OK
        HLT+2    .+4          ;ANS1 NOT EQUAL TO 117200

        TST      ANS2
        BEQ      .+4          ;CHECK SECOND HALF OF ANSWER
        HLT+2    .+4          ;BRANCH IF OK
        HLT+2    .+4          ;ANS2 NOT EQUAL TO 000000

        CMPB     #5,      ICNT       ;CHECK THE TEST NUMBER
        BEQ      .+4          ;BRANCH IF OK
        HLT      .+4          ;WRONG TEST! PC MUST HAVE FOULED UP.

        SCOPE
    
```

```

001630 004567 014276
001634 025177 177777
001640 125200 000000
001644 000307
001646 016574 000340

001652 000240
001654 075006

001656 004767 014310
001662 022706 000500
001666 001404
001670 012706 000500
001674 104000
001676 000416

001700 022767 000310 176672 TSAS:
001706 001401
001710 104000

001712 022767 117200 176664
001720 001401
001722 104002

001724 005767 176656
001730 001401
001732 104002

001734 122767 000005 177036 ENDS:
001742 001401
001744 104000

001746 104400
    
```



653  
654  
655  
656  
657  
658  
659  
660  
661  
662  
663  
664  
665  
666  
667  
668  
669  
670  
671  
672  
673  
674  
675  
676  
677  
678  
679  
680  
681  
682  
683  
684  
685  
686  
687  
688  
689  
690  
691  
692  
693

001750 004567 014334  
001754 100125 052525  
001760 135753 024642  
001764 000347  
001766 016574 000340  
001772 012705 000630  
001776 000240  
002000 075005  
002002 004767 014334  
002006 010567 176570  
002012 022767 000350 176560  
002020 001401  
002022 104000  
002024 022767 000634 176550  
002032 001401  
002034 104000  
002036 022767 135753 176540  
002044 001401  
002046 104002  
002050 022767 024642 176530  
002056 001401  
002060 104002  
002062 122767 000006 176710  
002070 001401  
002072 104000  
002074 104400

\*\*\*\*\*  
TEST 6: FADD (KE11F FLOATING ADD INSTRUCTION)  
135753,024642 + 100125,052525 = 135753,024642  
PS = 350, STACK POINTER = R5  
\*\*\*\*\*

TST6: JSR R5, PUSHR ; PUSH 4 WORDS ONTO R5 STACK, SET PRIORITY  
.WORD 100125,052525 ; SECOND OPERAND ON TOP  
.WORD 135753,024642 ; FIRST OPERAND ON BOTTOM  
.WORD 347 ; PROCESSOR PRIORITY LEVEL  
.WORD TRAPER,340 ; FIS TRAP VECTOR  
MOV #STACK0,R5 ; SET UP STACK POINTER  
  
NOP  
FADD+ R5 ; FLOATING ADD ON THE R5 STACK  
  
JSR PC, POPR ; POP THE ANSWER  
MOV R5, SSP ; SAVE "STACK POINTER"  
CMP #350, SPSW ; CHECK PS (EXCEPT T BIT)  
BEQ .+4 ; BRANCH IF OK  
HLT ; PS NOT EQUAL TO 350  
  
CMP #STACK4,SSP ; CHECK THE STACK POINTER (R5)  
BEQ .+4 ; BRANCH IF OK  
HLT ; STACK POINTER (R5) NOT EQUAL TO #STACK4  
  
CMP #135753,ANS1 ; CHECK FIRST HALF OF ANSWER  
BEQ .+4 ; BRANCH IF OK  
HLT+2 ; ANS1 NOT EQUAL TO 135753  
  
CMP #024642,ANS2 ; CHECK SECOND HALF OF ANSWER  
BEQ .+4 ; BRANCH IF OK  
HLT+2 ; ANS2 NOT EQUAL TO 024642  
  
END6: CMPB #6, ICNT ; CHECK THE TEST NUMBER  
BEQ .+4 ; BRANCH IF OK  
HLT ; WRONG TEST! PC MUST HAVE FOULED UP.  
  
SCOPE

694  
695  
696  
697  
698  
699  
700  
701  
702  
703  
704  
705  
706  
707  
708  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
730  
731  
732  
733  
734

\*\*\*\*\*  
:TEST 7: FADD (KE11F FLOATING ADD INSTRUCTION)  
: 000052,125252 + 001357,024642 = 001357,024642  
: PS = 240, STACK POINTER = R1  
\*\*\*\*\*

TST7:	JSR	R5	PUSHR	:	PUSH 4 WORDS ONTO R1 STACK, SET PRIORITY
	.WORD	001357,024642		:	SECOND OPERAND ON TOP
	.WORD	000052,125252		:	FIRST OPERAND ON BOTTOM
	.WORD	257		:	PROCESSOR PRIORITY LEVEL
	.WORD	TRAPER,340		:	FIS TRAP VECTOR
	MOV	#STACK0,R1		:	SET UP STACK POINTER
	NOP				
	FADD+	R1		:	FLOATING ADD ON THE R1 STACK
	JSR	PC,	POPR	:	POP THE ANSWER
	MOV	R1,	SSP	:	SAVE "STACK POINTER"
	CMP	#240,	SPSW	:	CHECK PS (EXCEPT T BIT)
	BEQ	+.4		:	BRANCH IF OK
	HLT			:	PS NOT EQUAL TO 240
	CMP	#STACK4,SSP		:	CHECK THE STACK POINTER (R1)
	BEQ	+.4		:	BRANCH IF OK
	HLT			:	STACK POINTER (R1) NOT EQUAL TO #STACK4
	CMP	#001357,ANS1		:	CHECK FIRST HALF OF ANSWER
	BEQ	+.4		:	BRANCH IF OK
	HLT+2			:	ANS1 NOT EQUAL TO 001357
	CMP	#024642,ANS2		:	CHECK SECOND HALF OF ANSWER
	BEQ	+.4		:	BRANCH IF OK
	HLT+2			:	ANS2 NOT EQUAL TO 024642
	CMPB	#7,	ICNT	:	CHECK THE TEST NUMBER
END7:	BEQ	+.4		:	BRANCH IF OK
	HLT			:	WRONG TEST! PC MUST HAVE FOULED UP.
	SCOPE				



```

735
736
737
738
739
740
741
742 002224 004567 014060
743 002230 000200 000000
744 002234 100400 000000
745 002240 000140
746 002242 016574 000340
747 002246 012705 000630
748
749 002252 000240
750 002254 075005
751
752 002256 004767 014060
753 002262 010567 176314
754 002266 022767 000150 176304
755 002274 001401
756 002276 104000
757
758 002300 022767 000634 176274
759 002306 001401
760 002310 104000
761
762 002312 022767 100200 176264
763 002320 001401
764 002322 104002
765
766 002324 005767 176256
767 002330 001401
768 002332 104002
769
770 002334 122767 000010 176436 END10:
771 002342 001401
772 002344 104000
773
774 002346 104400
775

```

```

*****
:TEST 10: FADD (KE11F FLOATING ADD INSTRUCTION)
:100400,000000 + 000200,000000 = 100200,000000
:PS = 150, STACK POINTER = R5
*****

```

```

TST10: JSR R5, PUSHR ;PUSH 4 WORDS ONTO R5 STACK, SET PRIORITY
        .WORD 000200,000000 ;SECOND OPERAND ON TOP
        .WORD 100400,000000 ;FIRST OPERAND ON BOTTOM
        .WORD 140 ;PROCESSOR PRIORITY LEVEL
        .WORD TRAPER,340 ;FIS TRAP VECTOR
        MOV #STACK0,R5 ;SET UP STACK POINTER

        NOP
        FADD+ R5 ;FLOATING ADD ON THE R5 STACK

        JSR PC, POPR ;POP THE ANSWER
        MOV R5, $SP ;SAVE "STACK POINTER"
        CMP #150, $PSW ;CHECK PS (EXCEPT T BIT)
        BEQ .+4 ;BRANCH IF OK
        HLT ;PS NOT EQUAL TO 150

        CMP #STACK4,$SP ;CHECK THE STACK POINTER (R5)
        BEQ .+4 ;BRANCH IF OK
        HLT ;STACK POINTER (R5) NOT EQUAL TO #STACK4

        CMP #100200,ANS1 ;CHECK FIRST HALF OF ANSWER
        BEQ .+4 ;BRANCH IF OK
        HLT+2 ;ANS1 NOT EQUAL TO 100200

        TST ANS2 ;CHECK SECOND HALF OF ANSWER
        BEQ .+4 ;BRANCH IF OK
        HLT+2 ;ANS2 NOT EQUAL TO 000000

        CMPB #10, ICNT ;CHECK THE TEST NUMBER
        BEQ .+4 ;BRANCH IF OK
        HLT ;WRONG TEST! PC MUST HAVE FOULED UP.

```

SCOPE

776  
777  
778  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
810  
811  
812  
813  
814  
815  
816  
817  
818  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
830  
831

```

*****
TEST 11: FADD (KE11F FLOATING ADD INSTRUCTION)
          100200,000000 + 000377,177777 ==> UNDERFLOW
          PS(ON STACK) = 152, STACK POINTER = R3
*****

TST11: JSR R5, PUSHR ; PUSH 4 WORDS ONTO R3 STACK, SET PRIORITY
        .WORD 000377,177777 ; SECOND OPERAND ON TOP
        .WORD 100200,000000 ; FIRST OPERAND ON BOTTOM
        .WORD 157 ; PROCESSOR PRIORITY LEVEL
        .WORD ISR11, 000 ; FIS TRAP VECTOR
        MOV #STACK0,R3 ; SET UP R3 AS STACK POINTER

NOP
FADD+ R3 ; FLOATING ADD ON THE R3 STACK

RTA11: JSR %7, POPR ; POP THE "ANSWER"
        MOV R3, SSP ; SAVE STACK POINTER (R3)
        HLT+2 ; FIS TRAP DIDN'T OCCURE!
        BR END11

ISR11: JSR %7, POPER ; POP ALL DATA OFF THE STACKS
        MOV R3, SSP ; SAVE STACK POINTER (R3)
        TST SPSW ; CHECK PS AFTER FIS TRAP
        BEQ .+4 ; BRANCH IF OK
        HLT ; PS AFTER FIS TRAP NOT EQUAL TO 000

        CMP #STACK0,SSP ; CHECK THE STACK POINTER (R3)
        BEQ .+4 ; BRANCH IF OK
        HLT ; STACK POINTER (R3) NOT EQUAL TO #STACK0

        CMP #RTA11,ANS1 ; CHECK FIS TRAP RETURN ADDRESS
        BEQ .+4 ; BRANCH IF OK
        HLT+1 ; FIS TRAP AT WRONG ADDRESS

        CMP #152,ANS2 ; CHECK PS BEFORE FIS TRAP
        BEQ .+4 ; BRANCH IF OK
        HLT+2 ; PS AT FIS TRAP TIME NOT 152

        CMP #000377,ANS3 ; CHECK DATA FROM THE STACK
        BEQ .+4 ; BRANCH IF OK
        HLT+4 ; DATA ON STACK (000377) CHANGED

        CMP #177777,ANS4 ; CHECK DATA FROM STACK
        BEQ .+4 ; BRANCH IF OK
        HLT+4 ; DATA ON STACK (177777) CHANGED

        CMP #100200,ANS5 ; CHECK DATA FROM STACK
        BEQ .+4 ; BRANCH IF OK
        HLT+6 ; DATA ON STACK (100200) CHANGED

        TST ANS6 ; CHECK DATA FROM STACK
        BEQ .+4 ; BRANCH IF OK
        HLT+6 ; DATA ON STACK (000000) CHANGED
    
```

```

002350 004567 013734
002354 000377 177777
002360 100200 000000
002364 000157
002366 002416 000000
002372 012703 000630

002376 000240
002400 075003

002402 004767 013734
002406 010367 176170
002412 104002
002414 000452

002416 004767 013752
002422 010367 176154
002426 005767 176146
002432 001401
002434 104000

002436 022767 000630 176136
002444 001401
002446 104000

002450 022767 002402 176126
002456 001401
002460 104001

002462 022767 000152 176116
002470 001401
002472 104002

002474 022767 000377 176106
002502 001401
002504 104004

002506 022767 177777 176076
002514 001401
002516 104004

002520 022767 100200 176066
002526 001401
002530 104006

002532 005767 176060
002536 001401
002540 104006
    
```



```

832 002542 122767 000011 176230 END11:  CMPB  #11,  ICNT  ;CHECK THE TEST NUMBER
833 002550 001401          BEQ    .+4          ;BRANCH IF OK
834 002552 104000          HLT                    ;WRONG TEST! PC MUST HAVE FOULED UP.
835
836 002554 104400          SCOPE
837
838
839
840
841
842
843
844
845 002556 004567 013526  TST12: JSR    R5,  PUSH  ;PUSH 4 WORDS ONTO R4 STACK, SET PRIORITY
846 002562 100252 125252  .WORD 100252,125252 ;SECOND OPERAND ON TOP
847 002566 000425 052525  .WORD 000425,052525 ;FIRST OPERAND ON BOTTOM
848 002572 000217          .WORD 217           ;PROCESSOR PRIORITY LEVEL
849 002574 016574 000340  .WORD TRAPER,340    ;FIS TRAP VECTOR
850 002600 012704 000630  MOV    #STACK0,R4   ;SET UP STACK POINTER
851
852 002604 000240          NOP
853 002606 075004          FADD+  R4           ;FLOATING ADD ON THE R4 STACK
854
855 002610 004767 013526  JSR    PC,  POPR   ;POP THE ANSWER
856 002614 010467 175762  MOV    R4,  $SP   ;SAVE "STACK POINTER"
857 002620 022767 000200 175752  CMP    #200, $PSW ;CHECK PS (EXCEPT T BIT)
858 002626 001401          BEQ    .+4          ;BRANCH IF OK
859 002630 104000          HLT                    ;PS NOT EQUAL TO 200
860
861 002632 022767 000634 175742  CMP    #STACK4,$SP ;CHECK THE STACK POINTER (R4)
862 002640 001401          BEQ    .+4          ;BRANCH IF OK
863 002642 104000          HLT                    ;STACK POINTER (R4) NOT EQUAL TO #STACK4
864
865 002644 022767 000200 175732  CMP    #000200,ANS1 ;CHECK FIRST HALF OF ANSWER
866 002652 001401          BEQ    .+4          ;BRANCH IF OK
867 002654 104002          HLT+2              ;ANS1 NOT EQUAL TO 000200
868
869 002656 005767 175724  TST    ANS2        ;CHECK SECOND HALF OF ANSWER
870 002662 001401          BEQ    .+4          ;BRANCH IF OK
871 002664 104002          HLT+2              ;ANS2 NOT EQUAL TO 000000
872
873 002666 122767 000012 176104 END12:  CMPB  #12,  ICNT  ;CHECK THE TEST NUMBER
874 002674 001401          BEQ    .+4          ;BRANCH IF OK
875 002676 104000          HLT                    ;WRONG TEST! PC MUST HAVE FOULED UP.
876
877 002700 104400          SCOPE
878

```

```

*****
:TEST 12:      FADD (KE11F FLOATING ADD INSTRUCTION)
:              000425,052525 + 100252,125252 = 000200,000000
:              PS = 200,      STACK POINTER = R4
*****

```

879  
880  
881  
882  
883  
884  
885  
886  
887  
888  
889  
890  
891  
892  
893  
894  
895  
896  
897  
898  
899  
900  
901  
902  
903  
904  
905  
906  
907  
908  
909  
910  
911  
912  
913  
914  
915  
916  
917  
918  
919  
920  
921  
922  
923  
924  
925  
926  
927  
928  
929  
930  
931  
932  
933  
934

```
*****
:TEST 13:      FADD (KE11F FLOATING ADD INSTRUCTION)
:              000200,000000 + 100377,177777 ==> UNDERFLOW
:              PS(ON STACK) = 252,      STACK POINTER = SP
*****
```

```
TST13: JSR      RS,      PUSHS      ;PUSH 4 WORDS ONTO STACK, SET PRIORITY
        .WORD   100377,177777      ;SECOND OPERAND ON TOP
        .WORD   000200,000000      ;FIRST OPERAND ON BOTTOM
        .WORD   257                  ;PROCESSOR PRIORITY LEVEL
        .WORD   ISR13, 340          ;FIS TRAP VECTOR
```

```
        NOP
        FADD+   SP                  ;FLOATING ADD ON THE STACK
```

```
RTA13: JSR      %7,      POPS      ;POP THE "ANSWER"
        HLT+2
        MOV     #500, SP          ;FIS TRAP DIDN'T OCCURE!
        BR      END13            ;RESTORE THE STACK POINTER
```

```
ISR13: JSR      %7,      POPES     ;POP ALL DATA OFF THE STACK
        CMP     #500, SP          ;CHECK THE STACK POINTER
        BEQ     ISA13            ;BRANCH IF OK
        MOV     #500, SP          ;RESTORE THE STACK POINTER
        HLT
        BR      END13            ;STACK POINTER FOULED UP
                                        ;SKIP REST OF TEST
```

```
ISA13: CMP     #340, SPSW         ;CHECK PS AFTER FIS TRAP
        BEQ     .+4              ;BRANCH IF OK
        HLT                                     ;PS AFTER FIS TRAP NOT EQUAL TO 340
```

```
        CMP     #RTA13, ANS1      ;CHECK FIS TRAP RETURN ADDRESS
        BEQ     .+4              ;BRANCH IF OK
        HLT+1                      ;FIS TRAP AT WRONG ADDRESS
```

```
        CMP     #252, ANS2        ;CHECK PS BEFORE FIS TRAP
        BEQ     .+4              ;BRANCH IF OK
        HLT+2                      ;PS AT FIS TRAP TIME NOT 252
```

```
        CMP     #100377,ANS3      ;CHECK DATA FROM THE STACK
        BEQ     .+4              ;BRANCH IF OK
        HLT+4                      ;DATA ON STACK (100377) CHANGED
```

```
        CMP     #177777,ANS4     ;CHECK DATA FROM STACK
        BEQ     .+4              ;BRANCH IF OK
        HLT+4                      ;DATA ON STACK (177777) CHANGED
```

```
        CMP     #000200,ANS5     ;CHECK DATA FROM STACK
        BEQ     .+4              ;BRANCH IF OK
        HLT+6                      ;DATA ON STACK (000200) CHANGED
```

```
TST     ANS6                      ;CHECK DATA FROM STACK
        BEQ     .+4              ;BRANCH IF OK
        HLT+6                      ;DATA ON STACK (000000) CHANGED
```



```

935 003072 122767 000013 175700 END13:  CMPB  #13,  ICNT  ;CHECK THE TEST NUMBER
936 003100 001401  BEQ   .+4   ;BRANCH IF OK
937 003102 104000  HLT   ;WRONG TEST! PC MUST HAVE FOULED UP.
938
939 003104 104400  SCOPE
940
941
942
943
944
945
946
947
948 003106 004567 013020  TST14: JSR   R5,  PUSH5 ;PUSH 4 WORDS ONTO STACK, SET PRIORITY
949 003112 000252 125252  .WORD 000252,125252 ;SECOND OPERAND ON TOP
950 003116 100425 052525  .WORD 100425,052525 ;FIRST OPERAND ON BOTTOM
951 003122 000307  .WORD 307 ;PROCESSOR PRIORITY LEVEL
952 003124 016574 000340  .WORD TRAPER,340 ;FIS TRAP VECTOR
953
954 003130 000240  NOP
955 003132 075006  FADD+ SP ;FLOATING ADD ON THE STACK
956
957 003134 004767 013032  JSR   PC,  POPS ;POP THE ANSWER
958 003140 022706 000500  CMP   #500, SP ;CHECK THE STACK POINTER
959 003144 001404  BEQ   TSA14 ;BRANCH IF OK
960 003146 012706 000500  MOV   #500, SP ;RESTORE STACK POINTER
961 003152 104000  HLT   ;STACK POINTER FOULED UP
962 003154 000416  BR    END14 ;SKIP REST OF TEST
963
964 003156 022767 000310 175414 TSA14: CMP   #310, $PSW ;CHECK PS (EXCEPT T BIT)
965 003164 001401  BEQ   .+4   ;BRANCH IF OK
966 003166 104000  HLT   ;PS NOT EQUAL TO 310
967
968 003170 022767 100200 175406  CMP   #100200,ANS1 ;CHECK FIRST HALF OF ANSWER
969 003176 001401  BEQ   .+4   ;BRANCH IF OK
970 003200 104002  HLT+2 ;ANS1 NOT EQUAL TO 100200
971
972 003202 005767 175400  TST   ANS2 ;CHECK SECOND HALF OF ANSWER
973 003206 001401  BEQ   .+4   ;BRANCH IF OK
974 003210 104002  HLT+2 ;ANS2 NOT EQUAL TO 000000
975
976 003212 122767 000014 175560 END14: CMPB  #14,  ICNT  ;CHECK THE TEST NUMBER
977 003220 001401  BEQ   .+4   ;BRANCH IF OK
978 003222 104000  HLT   ;WRONG TEST! PC MUST HAVE FOULED UP.
979
980 003224 104400  SCOPE
981

```

```

*****
:TEST 14: FADD (KE11F FLOATING ADD INSTRUCTION)
: 100425 052525 + 000252 125252 = 100200,000000
: PS = 310, STACK POINTER = SP
*****

```

982  
983  
984  
985  
986  
987  
988  
989  
990  
991  
992  
993  
994  
995  
996  
997  
998  
999  
1000  
1001  
1002  
1003  
1004  
1005  
1006  
1007  
1008  
1009  
1010  
1011  
1012  
1013  
1014  
1015  
1016  
1017  
1018  
1019  
1020  
1021  
1022

```
*****
:TEST 15: FADD (KE11F FLOATING ADD INSTRUCTION)
: 077652,125252 + 077452,125252 = 077777,177777
: PS = 240, STACK POINTER = SP
*****
```

```
TST15: JSR R5, PUSH5 ;PUSH 4 WORDS ONTO STACK, SET PRIORITY
        .WORD 077452,125252 ;SECOND OPERAND ON TOP
        .WORD 077652,125252 ;FIRST OPERAND ON BOTTOM
        .WORD 257 ;PROCESSOR PRIORITY LEVEL
        .WORD TRAPER,340 ;FIS TRAP VECTOR

        NOP
        FADD+ SP ;FLOATING ADD ON THE STACK

        JSR PC, POPS ;POP THE ANSWER
        CMP #500, SP ;CHECK THE STACK POINTER
        BEQ TSA15 ;BRANCH IF OK
        MOV #500, SP ;RESTORE STACK POINTER
        HLT ;STACK POINTER FOULED UP
        BR END15 ;SKIP REST OF TEST

1005 003276 022767 000240 175274 TSA15: CMP #240, SPSW ;CHECK PS (EXCEPT T BIT)
1006 003304 001401 ;BEQ .+4 ;BRANCH IF OK
1007 003306 104000 ;HLT ;PS NOT EQUAL TO 240

1009 003310 022767 077777 175266 CMP #077777,ANS1 ;CHECK FIRST HALF OF ANSWER
1010 003316 001401 ;BEQ .+4 ;BRANCH IF OK
1011 003320 104002 ;HLT+2 ;ANS1 NOT EQUAL TO 077777

1013 003322 022767 177777 175256 CMP #177777,ANS2 ;CHECK SECOND HALF OF ANSWER
1014 003330 001401 ;BEQ .+4 ;BRANCH IF OK
1015 003332 104002 ;HLT+2 ;ANS2 NOT EQUAL TO 177777

1017 003334 122767 000015 175436 END15: CMPB #15, ICNT ;CHECK THE TEST NUMBER
1018 003342 001401 ;BEQ .+4 ;BRANCH IF OK
1019 003344 104000 ;HLT ;WRONG TEST! PC MUST HAVE FOULED UP.

1021 003346 104400 SCOPE
```



```

1023
1024
1025
1026
1027
1028
1029
1030 003350 004567 012734
1031 003354 177652 125252
1032 003360 177452 125253
1033 003364 000105
1034 003366 003416 000252
1035 003372 012701 000630
1036
1037 003376 000240
1038 003400 075001
1039
1040 003402 004767 012734
1041 003406 010167 175170
1042 003412 104002
1043 003414 000454
1044
1045 003416 004767 012752
1046 003422 010167 175154
1047 003426 022767 000252 175144
1048 003434 001401
1049 003436 104000
1050
1051 003440 022767 000630 175134
1052 003446 001401
1053 003450 104000
1054
1055 003452 022767 003402 175124
1056 003460 001401
1057 003462 104001
1058
1059 003464 022767 000102 175114
1060 003472 001401
1061 003474 104002
1062
1063 003476 022767 177652 175104
1064 003504 001401
1065 003506 104004
1066
1067 003510 022767 125252 175074
1068 003516 001401
1069 003520 104004
1070
1071 003522 022767 177452 175064
1072 003530 001401
1073 003532 104006
1074
1075 003534 022767 125253 175054
1076 003542 001401
1077 003544 104006
1078

*****
TEST 16: FADD (KE11F FLOATING ADD INSTRUCTION)
          177452,125253 + 177652,125252 ==> OVERFLOW
          PS(ON STACK) = 102, STACK POINTER = R1
*****

TST16: JSR R5, PUSHR ;PUSH 4 WORDS ONTO R1 STACK, SET PRIORITY
        .WORD 177652,125252 ;SECOND OPERAND ON TOP
        .WORD 177452,125253 ;FIRST OPERAND ON BOTTOM
        .WORD 105 ;PROCESSOR PRIORITY LEVEL
        .WORD ISR16, 252 ;FIS TRAP VECTOR
        MOV #STACK0,R1 ;SET UP R1 AS STACK POINTER

NOP
FADD+ R1 ;FLOATING ADD ON THE R1 STACK

RTA16: JSR %7, POPR ;POP THE "ANSWER"
        MOV R1, SSP ;SAVE STACK POINTER (R1)
        HLT+2 ;FIS TRAP DIDN'T OCCURE!
        BR END16

ISR16: JSR %7, POPER ;POP ALL DATA OFF THE STACKS
        MOV R1, SSP ;SAVE STACK POINTER (R1)
        CMP #252, SPSW ;CHECK PS AFTER FIS TRAP
        BEQ .+4 ;BRANCH IF OK
        HLT ;PS AFTER FIS TRAP NOT EQUAL TO 252

CMP #STACK0,SSP ;CHECK THE STACK POINTER (R1)
BEQ .+4 ;BRANCH IF OK
HLT ;STACK POINTER (R1) NOT EQUAL TO #STACK0

CMP #RTA16, ANS1 ;CHECK FIS TRAP RETURN ADDRESS
BEQ .+4 ;BRANCH IF OK
HLT+1 ;FIS TRAP AT WRONG ADDRESS

CMP #102, ANS2 ;CHECK PS BEFORE FIS TRAP
BEQ .+4 ;BRANCH IF OK
HLT+2 ;PS AT FIS TRAP TIME NOT 102

CMP #177652,ANS3 ;CHECK DATA FROM THE STACK
BEQ .+4 ;BRANCH IF OK
HLT+4 ;DATA ON STACK (177652) CHANGED

CMP #125252,ANS4 ;CHECK DATA FROM STACK
BEQ .+4 ;BRANCH IF OK
HLT+4 ;DATA ON STACK (125252) CHANGED

CMP #177452,ANS5 ;CHECK DATA FROM STACK
BEQ .+4 ;BRANCH IF OK
HLT+6 ;DATA ON STACK (177452) CHANGED

CMP #125253,ANS6 ;CHECK DATA FROM STACK
BEQ .+4 ;BRANCH IF OK
HLT+6 ;DATA ON STACK (125253) CHANGED
    
```

```

1079 003546 122767 000016 175224 END16: CMPB #16, ICNT ;CHECK THE TEST NUMBER
1080 003554 001401 BEQ .+4 ;BRANCH IF OK
1081 003556 104000 HLT ;WRONG TEST! PC MUST HAVE FOULED UP.
1082
1083 003560 104400 SCOPE
1084
1085
1086
1087
1088
1089
1090
1091
1092 003562 004567 012522 TST17: JSR R5, PUSHR ;PUSH 4 WORDS ONTO R4 STACK, SET PRIORITY
1093 003566 177652 125252 .WORD 177652,125252 ;SECOND OPERAND ON TOP
1094 003572 177452 125252 .WORD 177452,125252 ;FIRST OPERAND ON BOTTOM
1095 003576 000357 .WORD 357 ;PROCESSOR PRIORITY LEVEL
1096 003600 016574 000340 .WORD TRAPER,340 ;FIS TRAP VECTOR
1097 003604 012704 000630 MOV #STACK0,R4 ;SET UP STACK POINTER
1098
1099 003610 000240 NOP
1100 003612 075004 FADD+ R4 ;FLOATING ADD ON THE R4 STACK
1101
1102 003614 004767 012522 JSR PC, POPR ;POP THE ANSWER
1103 003620 010467 174756 MOV R4, $SP ;SAVE "STACK POINTER"
1104 003624 022767 000350 174746 CMP #350, $PSW ;CHECK PS (EXCEPT T BIT)
1105 003632 001401 BEQ .+4 ;BRANCH IF OK
1106 003634 104000 HLT ;PS NOT EQUAL TO 350
1107
1108 003636 022767 000634 174736 CMP #STACK4,$SP ;CHECK THE STACK POINTER (R4)
1109 003644 001401 BEQ .+4 ;BRANCH IF OK
1110 003646 104000 HLT ;STACK POINTER (R4) NOT EQUAL TO #STACK4
1111
1112 003650 022767 177777 174726 CMP #177777,ANS1 ;CHECK FIRST HALF OF ANSWER
1113 003656 001401 BEQ .+4 ;BRANCH IF OK
1114 003660 104002 HLT+2 ;ANS1 NOT EQUAL TO 177777
1115
1116 003662 022767 177777 174716 CMP #177777,ANS2 ;CHECK SECOND HALF OF ANSWER
1117 003670 001401 BEQ .+4 ;BRANCH IF OK
1118 003672 104002 HLT+2 ;ANS2 NOT EQUAL TO 177777
1119
1120 003674 122767 000017 175076 END17: CMPB #17, ICNT ;CHECK THE TEST NUMBER
1121 003702 001401 BEQ .+4 ;BRANCH IF OK
1122 003704 104000 HLT ;WRONG TEST! PC MUST HAVE FOULED UP.
1123
1124 003706 104400 SCOPE
1125

```

```

*****
:TEST 17: FADD (KE11F FLOATING ADD INSTRUCTION)
:177452,125252 + 177652,125252 = 177777,177777
:PS = 350, STACK POINTER = R4
*****

```



```

1126
1127
1128
1129
1130
1131
1132
1133 003710 004567 012216
1134 003714 077452 125252
1135 003720 077652 125253
1136 003724 000003
1137 003726 003752 000344
1138
1139 003732 000240
1140 003734 075006
1141
1142 003736 004767 012230
1143 003742 104002
1144 003744 012706 000500
1145 003750 000454
1146
1147 003752 004767 012250
1148 003756 022706 000500
1149 003762 001404
1150 003764 012706 000500
1151 003770 104000
1152 003772 000443
1153
1154 003774 022767 000344 174576
1155 004002 001401
1156 004004 104000
1157
1158 004006 022767 003736 174570
1159 004014 001401
1160 004016 104001
1161
1162 004020 022767 000002 174560
1163 004026 001401
1164 004030 104002
1165
1166 004032 022767 077452 174550
1167 004040 001401
1168 004042 104004
1169
1170 004044 022767 125252 174540
1171 004052 001401
1172 004054 104004
1173
1174 004056 022767 077652 174530
1175 004064 001401
1176 004066 104006
1177
1178 004070 022767 125253 174520
1179 004076 001401
1180 004100 104006
1181

```

```

*****
:TEST 20: FADD (KE11F FLOATING ADD INSTRUCTION)
: 077652,125253 + 077452,125252 ==> OVERFLOW
: PS(ON STACK) = 002, STACK POINTER = SP
*****
TST20: JSR R5, PUSH5 ;PUSH 4 WORDS ONTO STACK, SET PRIORITY
: .WORD 077452,125252 ;SECOND OPERAND ON TOP
: .WORD 077652,125253 ;FIRST OPERAND ON BOTTOM
: .WORD 003 ;PROCESSOR PRIORITY LEVEL
: .WORD ISR20, 344 ;FIS TRAP VECTOR

NOP
FADD+ SP ;FLOATING ADD ON THE STACK

RTA20: JSR %7, POPS ;POP THE "ANSWER"
: HLT+2 ;FIS TRAP DIDN'T OCCURE!
: MOV #500, SP ;RESTORE THE STACK POINTER
: BR END20

ISR20: JSR %7, POPES ;POP ALL DATA OFF THE STACK
: CMP #500, SP ;CHECK THE STACK POINTER
: BEQ ISA20 ;BRANCH IF OK
: MOV #500, SP ;RESTORE THE STACK POINTER
: HLT ;STACK POINTER FOULED UP
: BR END20 ;SKIP REST OF TEST

ISA20: CMP #344, SPSW ;CHECK PS AFTER FIS TRAP
: BEQ .+4 ;BRANCH IF OK
: HLT ;PS AFTER FIS TRAP NOT EQUAL TO 344

CMP #RTA20, ANS1 ;CHECK FIS TRAP RETURN ADDRESS
: BEQ .+4 ;BRANCH IF OK
: HLT+1 ;FIS TRAP AT WRONG ADDRESS

CMP #002, ANS2 ;CHECK PS BEFORE FIS TRAP
: BEQ .+4 ;BRANCH IF OK
: HLT+2 ;PS AT FIS TRAP TIME NOT 002

CMP #077452, ANS3 ;CHECK DATA FROM THE STACK
: BEQ .+4 ;BRANCH IF OK
: HLT+4 ;DATA ON STACK (077452) CHANGED

CMP #125252, ANS4 ;CHECK DATA FROM STACK
: BEQ .+4 ;BRANCH IF OK
: HLT+4 ;DATA ON STACK (125252) CHANGED

CMP #077652, ANS5 ;CHECK DATA FROM STACK
: BEQ .+4 ;BRANCH IF OK
: HLT+6 ;DATA ON STACK (077652) CHANGED

CMP #125253, ANS6 ;CHECK DATA FROM STACK
: BEQ .+4 ;BRANCH IF OK
: HLT+6 ;DATA ON STACK (125253) CHANGED

```

```

1182 004102 122767 000020 174670 END20: CMPB #20, ICNT ;CHECK THE TEST NUMBER
1183 004110 001401 BEQ .+4 ;BRANCH IF OK
1184 004112 104000 HLT ;WRONG TEST! PC MUST HAVE FOULED UP.
1185
1186 004114 104400 SCOPE
1187
1188
1189 ;*****
1190 ;TEST 21: FSUB (KE11F FLOATING SUBTRACT INSTRUCTION)
1191 ; 177520,017552 - 135352,051107 = 177520,017552
1192 ; PS = 050, STACK POINTER = R1
1193 ;*****
1194
1195 004116 004567 012166 TST21: JSR R5, PUSHR ;PUSH 4 WORDS ONTO R1 STACK, SET PRIORITY
1196 004122 135352 051107 .WORD 135352,051107 ;SECOND OPERAND ON TOP
1197 004126 177520 017552 .WORD 177520,017552 ;FIRST OPERAND ON BOTTOM
1198 004132 000040 .WORD 040 ;PROCESSOR PRIORITY LEVEL
1199 004134 016574 000340 .WORD TRAPER, 340 ;FIS TRAP VECTOR
1200 004140 012701 000630 MOV #STACK0,R1 ;SET UP STACK POINTER
1201
1202 004144 000240 NOP
1203 004146 075011 FSUB+ R1 ;FLOATING SUBTRACT ON THE R1 STACK
1204
1205 004150 004767 012166 JSR PC, POPR ;POP THE ANSWER
1206 004154 010167 174422 MOV R1, SSP ;SAVE "STACK POINTER"
1207 004160 022767 000050 174412 CMP #050, SPSW ;CHECK PS (EXCEPT T BIT)
1208 004166 001401 BEQ .+4 ;BRANCH IF OK
1209 004170 104000 HLT ;PS NOT EQUAL TO 050
1210
1211 004172 022767 000634 174402 CMP #STACK4,SSP ;CHECK THE STACK POINTER (R1)
1212 004200 001401 BEQ .+4 ;BRANCH IF OK
1213 004202 104000 HLT ;STACK POINTER (R1) NOT EQUAL TO #STACK4
1214
1215 004204 022767 177520 174372 CMP #177520,ANS1 ;CHECK FIRST HALF OF ANSWER
1216 004212 001401 BEQ .+4 ;BRANCH IF OK
1217 004214 104002 HLT+2 ;ANS1 NOT EQUAL TO 177520
1218
1219 004216 022767 017552 174362 CMP #017552,ANS2 ;CHECK SECOND HALF OF ANSWER
1220 004224 001401 BEQ .+4 ;BRANCH IF OK
1221 004226 104002 HLT+2 ;ANS2 NOT EQUAL TO 017552
1222
1223 004230 122767 000021 174542 END21: CMPB #21, ICNT ;CHECK THE TEST NUMBER
1224 004236 001401 BEQ .+4 ;BRANCH IF OK
1225 004240 104000 HLT ;WRONG TEST! PC MUST HAVE FOULED UP.
1226
1227 004242 104400 SCOPE
1228
    
```



1229  
1230  
1231  
1232  
1233  
1234  
1235  
1236  
1237  
1238  
1239  
1240  
1241  
1242  
1243  
1244  
1245  
1246  
1247  
1248  
1249  
1250  
1251  
1252  
1253  
1254  
1255  
1256  
1257  
1258  
1259  
1260  
1261  
1262  
1263  
1264  
1265  
1266  
1267  
1268  
1269

\*\*\*\*\*  
TEST 22: FSUB (KE11F FLOATING SUBTRACT INSTRUCTION)  
125252,125252 - 125252,125253 = 017400,000000  
PS = 040, STACK POINTER = R0  
\*\*\*\*\*

TST22: JSR R5, PUSHR ; PUSH 4 WORDS ONTO R0 STACK, SET PRIORITY  
; SECOND OPERAND ON TOP  
; FIRST OPERAND ON BOTTOM  
; PROCESSOR PRIORITY LEVEL  
; FIS TRAP VECTOR  
; SET UP STACK POINTER  
.WORD 125252,125253  
.WORD 125252,125252  
.WORD 047  
.WORD TRAPER, 340  
MOV #STACK0,R0  
  
NOP  
FSUB+ R0 ; FLOATING SUBTRACT ON THE R0 STACK  
  
JSR PC, POPR ; POP THE ANSWER  
MOV R0, SSP ; SAVE "STACK POINTER"  
CMP #040, SPSW ; CHECK PS (EXCEPT T BIT)  
BEQ .+4 ; BRANCH IF OK  
HLT ; PS NOT EQUAL TO 040  
  
CMP #STACK4,SSP ; CHECK THE STACK POINTER (R0)  
BEQ .+4 ; BRANCH IF OK  
HLT ; STACK POINTER (R0) NOT EQUAL TO #STACK4  
  
CMP #017400,ANS1 ; CHECK FIRST HALF OF ANSWER  
BEQ .+4 ; BRANCH IF OK  
HLT+2 ; ANS1 NOT EQUAL TO 017400  
  
TST ANS2 ; CHECK SECOND HALF OF ANSWER  
BEQ .+4 ; BRANCH IF OK  
HLT+2 ; ANS2 NOT EQUAL TO 000000  
  
END22: CMPB #22, ICNT ; CHECK THE TEST NUMBER  
BEQ .+4 ; BRANCH IF OK  
HLT ; WRONG TEST! PC MUST HAVE FOULED UP.  
  
SCOPE

1270  
1271  
1272  
1273  
1274  
1275  
1276  
1277  
1278  
1279  
1280  
1281  
1282  
1283  
1284  
1285  
1286  
1287  
1288  
1289  
1290  
1291  
1292  
1293  
1294  
1295  
1296  
1297  
1298  
1299  
1300  
1301  
1302  
1303  
1304  
1305  
1306  
1307  
1308  
1309  
1310

004370 004567 011536  
004374 100177 177777  
004400 002460 123456  
004404 000015  
004406 016574 000340  
004412 000240  
004414 075016  
004416 004767 011550  
004422 022706 000500  
004426 001404  
004430 012706 000500  
004434 104000  
004436 000417  
004440 022767 000000 174132  
004446 001401  
004450 104000  
004452 022767 002460 174124  
004460 001401  
004462 104002  
004464 022767 123456 174114  
004472 001401  
004474 104002  
004476 122767 000023 174274  
004504 001401  
004506 104000  
004510 104400

\*\*\*\*\*  
:TEST 23: FSUB (KE11F FLOATING SUBTRACT INSTRUCTION)  
: 002460,123456 - 100177,177777 = 002460,123456  
: PS = 000, STACK POINTER = SP  
:\*\*\*\*\*

TST23: JSR RS, PUSH5 ;PUSH 4 WORDS ONTO STACK, SET PRIORITY  
.WORD 100177,177777 ;SECOND OPERAND ON TOP  
.WORD 002460,123456 ;FIRST OPERAND ON BOTTOM  
.WORD 015 ;PROCESSOR PRIORITY LEVEL  
.WORD TRAPER, 340 ;FIS TRAP VECTOR  
  
NOP  
FSUB+ SP ;FLOATING SUBTRACT ON THE STACK  
  
JSR PC, POPS ;POP THE ANSWER  
CMP #500, SP ;CHECK THE STACK POINTER  
BEQ TSA23 ;BRANCH IF OK  
MOV #500, SP ;RESTORE STACK POINTER  
HLT ;STACK POINTER FOULED UP  
BR END23 ;SKIP REST OF TEST  
  
TSA23: CMP #000, SPSW ;CHECK PS (EXCEPT T BIT)  
BEQ .+4 ;BRANCH IF OK  
HLT ;PS NOT EQUAL TO 000  
  
CMP #002460,ANS1 ;CHECK FIRST HALF OF ANSWER  
BEQ .+4 ;BRANCH IF OK  
HLT+2 ;ANS1 NOT EQUAL TO 002460  
  
CMP #123456,ANS2 ;CHECK SECOND HALF OF ANSWER  
BEQ .+4 ;BRANCH IF OK  
HLT+2 ;ANS2 NOT EQUAL TO 123456  
  
END23: CMPB #23, ICNT ;CHECK THE TEST NUMBER  
BEQ .+4 ;BRANCH IF OK  
HLT ;WRONG TEST! PC MUST HAVE FOULED UP.  
  
SCOPE



```

1311
1312
1313
1314
1315
1316
1317
1318 004512 004567 011572
1319 004516 000252 125252
1320 004522 000425 052525
1321 004526 000217
1322 004530 016574 000340
1323 004534 012704 000630
1324
1325 004540 000240
1326 004542 075014
1327
1328 004544 004767 011572
1329 004550 010467 174026
1330 004554 022767 000200 174016
1331 004562 001401
1332 004564 104000
1333
1334 004566 022767 000634 174006
1335 004574 001401
1336 004576 104000
1337
1338 004600 022767 000200 173776
1339 004606 001401
1340 004610 104002
1341
1342 004612 005767 173770
1343 004616 001401
1344 004620 104002
1345
1346 004622 122767 000024 174150 END24:
1347 004630 001401
1348 004632 104000
1349
1350 004634 104400
1351

```

```

*****
:TEST 24: FSUB (KE11F FLOATING SUBTRACT INSTRUCTION)
: 000425,052525 - 000252,125252 = 000200,000000
: PS = 200, STACK POINTER = R4
*****
TST24: JSR R5, PUSHR ;PUSH 4 WORDS ONTO R4 STACK, SET PRIORITY
        .WORD 000252,125252 ;SECOND OPERAND ON TOP
        .WORD 000425,052525 ;FIRST OPERAND ON BOTTOM
        .WORD 217 ;PROCESSOR PRIORITY LEVEL
        .WORD TRAPER, 340 ;FIS TRAP VECTOR
        MOV #STACK0,R4 ;SET UP STACK POINTER

        NOP
        FSUB+ R4 ;FLOATING SUBTRACT ON THE R4 STACK

        JSR PC, POPR ;POP THE ANSWER
        MOV R4, SSP ;SAVE "STACK POINTER"
        CMP #200, SPSW ;CHECK PS (EXCEPT T BIT)
        BEQ .+4 ;BRANCH IF OK
        HLT ;PS NOT EQUAL TO 200

        CMP #STACK4,SSP ;CHECK THE STACK POINTER (R4)
        BEQ .+4 ;BRANCH IF OK
        HLT ;STACK POINTER (R4) NOT EQUAL TO #STACK4

        CMP #000200,ANS1 ;CHECK FIRST HALF OF ANSWER
        BEQ .+4 ;BRANCH IF OK
        HLT+2 ;ANS1 NOT EQUAL TO 000200

        TST ANS2 ;CHECK SECOND HALF OF ANSWER
        BEQ .+4 ;BRANCH IF OK
        HLT+2 ;ANS2 NOT EQUAL TO 000000

        CMPB #24, ICNT ;CHECK THE TEST NUMBER
        BEQ .+4 ;BRANCH IF OK
        HLT ;WRONG TEST! PC MUST HAVE FOULED UP.

        SCOPE

```

1352  
1353  
1354  
1355  
1356  
1357  
1358  
1359  
1360  
1361  
1362  
1363  
1364  
1365  
1366  
1367  
1368  
1369  
1370  
1371  
1372  
1373  
1374  
1375  
1376  
1377  
1378  
1379  
1380  
1381  
1382  
1383  
1384  
1385  
1386  
1387  
1388  
1389  
1390  
1391  
1392  
1393  
1394  
1395  
1396  
1397  
1398  
1399  
1400  
1401  
1402  
1403  
1404  
1405  
1406  
1407

004636 004567 011270  
004642 000252 125253  
004646 000425 052525  
004652 000257  
004654 004700 000340  
004660 000240  
004662 075016  
004664 004767 011302  
004670 104002  
004672 012706 000500  
004676 000454  
004700 004767 011322  
004704 022706 000500  
004710 001404  
004712 012706 000500  
004716 104000  
004720 000443  
004722 022767 000340 173650  
004730 001401  
004732 104000  
004734 022767 004664 173642  
004742 001401  
004744 104001  
004746 022767 000252 173632  
004754 001401  
004756 104002  
004760 022767 000252 173622  
004766 001401  
004770 104004  
004772 022767 125253 173612  
005000 001401  
005002 104004  
005004 022767 000425 173602  
005012 001401  
005014 104006  
005016 022767 052525 173572  
005024 001401  
005026 104006

```
*****
TEST 25:      FSUB (KE11F FLOATING SUBTRACT INSTRUCTION)
              000425,052525 - 000252,125253 ==> UNDERFLOW
              PS(ON STACK) = 252,      STACK POINTER = SP
*****
TST25:  JSR      R5,      PUSH5      ;PUSH 4 WORDS ONTO STACK, SET PRIORITY
        .WORD    000252,125253      ;SECOND OPERAND ON TOP
        .WORD    000425,052525      ;FIRST OPERAND ON BOTTOM
        .WORD    257                ;PROCESSOR PRIORITY LEVEL
        .WORD    ISR25, 340         ;FIS TRAP VECTOR

        NOP
        FSUB+   SP                ;FLOATING SUBTRACT ON THE STACK

RTA25:  JSR      %7,      POPS       ;POP THE "ANSWER"
        HLT+2
        MOV     #500,     SP         ;FIS TRAP DIDN'T OCCURE!
        BR     END25              ;RESTORE THE STACK POINTER

ISR25:  JSR      %7,      POPES      ;POP ALL DATA OFF THE STACK
        CMP     #500,     SP         ;CHECK THE STACK POINTER
        BEQ    ISA25        ;BRANCH IF OK
        MOV     #500,     SP         ;RESTORE THE STACK POINTER
        HLT
        BR     END25              ;STACK POINTER FOULED UP
                                        ;SKIP REST OF TEST

ISA25:  CMP     #340,     SPSW       ;CHECK PS AFTER FIS TRAP
        BEQ    .+4          ;BRANCH IF OK
        HLT                ;PS AFTER FIS TRAP NOT EQUAL TO 340

        CMP     #RTA25,   ANS1      ;CHECK FIS TRAP RETURN ADDRESS
        BEQ    .+4          ;BRANCH IF OK
        HLT+1              ;FIS TRAP AT WRONG ADDRESS

        CMP     #252,     ANS2      ;CHECK PS BEFORE FIS TRAP
        BEQ    .+4          ;BRANCH IF OK
        HLT+2              ;PS AT FIS TRAP TIME NOT 252

        CMP     #000252,  ANS3      ;CHECK DATA FROM THE STACK
        BEQ    .+4          ;BRANCH IF OK
        HLT+4              ;DATA ON STACK (000252) CHANGED

        CMP     #125253,  ANS4      ;CHECK DATA FROM STACK
        BEQ    .+4          ;BRANCH IF OK
        HLT+4              ;DATA ON STACK (125253) CHANGED

        CMP     #000425,  ANS5      ;CHECK DATA FROM STACK
        BEQ    .+4          ;BRANCH IF OK
        HLT+6              ;DATA ON STACK (000425) CHANGED

        CMP     #052525,  ANS6      ;CHECK DATA FROM STACK
        BEQ    .+4          ;BRANCH IF OK
        HLT+6              ;DATA ON STACK (052525) CHANGED
```



```

1408 005030 122767 000025 173742 END25: CMPB #25, ICNT ;CHECK THE TEST NUMBER
1409 005036 001401 BEQ .+4 ;BRANCH IF OK
1410 005040 104000 HLT ;WRONG TEST! PC MUST HAVE FOULED UP.
1411
1412 005042 104400 SCOPE
1413
1414
1415
1416 :*****
1417 :TEST 26: FSUB (KE11F FLOATING SUBTRACT INSTRUCTION)
1418 : 177452,125252 - 077652,125252 = 177777,177777
1419 : PS = 350, STACK POINTER = SP
1420 :*****
1421 005044 004567 011062 TST26: JSR R5, PUSHS ;PUSH 4 WORDS ONTO STACK, SET PRIORITY
1422 005050 077652 125252 .WORD 077652,125252 ;SECOND OPERAND ON TOP
1423 005054 177452 125252 .WORD 177452,125252 ;FIRST OPERAND ON BOTTOM
1424 005060 000357 .WORD 357 ;PROCESSOR PRIORITY LEVEL
1425 005062 016574 000340 .WORD TRAPER, 340 ;FIS TRAP VECTOR
1426
1427 005066 000240 NOP
1428 005070 075016 FSUB+ SP ;FLOATING SUBTRACT ON THE STACK
1429
1430 005072 004767 011074 JSR PC, POPS ;POP THE ANSWER
1431 005076 022706 000500 CMP #500, SP ;CHECK THE STACK POINTER
1432 005102 001404 BEQ TSA26 ;BRANCH IF OK
1433 005104 012706 000500 MOV #500, SP ;RESTORE STACK POINTER
1434 005110 104000 HLT ;STACK POINTER FOULED UP
1435 005112 000417 BR END26 ;SKIP REST OF TEST
1436
1437 005114 022767 000350 173456 TSA26: CMP #350, SPSW ;CHECK PS (EXCEPT T BIT)
1438 005122 001401 BEQ .+4 ;BRANCH IF OK
1439 005124 104000 HLT ;PS NOT EQUAL TO 350
1440
1441 005126 022767 177777 173450 CMP #177777,ANS1 ;CHECK FIRST HALF OF ANSWER
1442 005134 001401 BEQ .+4 ;BRANCH IF OK
1443 005136 104002 HLT+2 ;ANS1 NOT EQUAL TO 177777
1444
1445 005140 022767 177777 173440 CMP #177777,ANS2 ;CHECK SECOND HALF OF ANSWER
1446 005146 001401 BEQ .+4 ;BRANCH IF OK
1447 005150 104002 HLT+2 ;ANS2 NOT EQUAL TO 177777
1448
1449 005152 122767 000026 173620 END26: CMPB #26, ICNT ;CHECK THE TEST NUMBER
1450 005160 001401 BEQ .+4 ;BRANCH IF OK
1451 005162 104000 HLT ;WRONG TEST! PC MUST HAVE FOULED UP.
1452
1453 005164 104400 SCOPE
1454

```

```

1455
1456
1457
1458
1459
1460
1461
1462 005166 004567 011116
1463 005172 177452 125252
1464 005176 077652 125253
1465 005202 000015
1466 005204 005234 000344
1467 005210 012703 000630
1468
1469 005214 000240
1470 005216 075013
1471
1472 005220 004767 011116
1473 005224 010367 173352
1474 005230 104002
1475 005232 000454
1476
1477 005234 004767 011134
1478 005240 010367 173336
1479 005244 022767 000344 173326
1480 005252 001401
1481 005254 104000
1482
1483 005256 022767 000630 173316
1484 005264 001401
1485 005266 104000
1486
1487 005270 022767 005220 173306
1488 005276 001401
1489 005300 104001
1490
1491 005302 022767 000002 173276
1492 005310 001401
1493 005312 104002
1494
1495 005314 022767 177452 173266
1496 005322 001401
1497 005324 104004
1498
1499 005326 022767 125252 173256
1500 005334 001401
1501 005336 104004
1502
1503 005340 022767 077652 173246
1504 005346 001401
1505 005350 104006
1506
1507 005352 022767 125253 173236
1508 005360 001401
1509 005362 104006
1510
    
```

```

*****
TEST 27:      FSUB (KE11F FLOATING SUBTRACT INSTRUCTION)
              077652,125253 - 177452,125252 ==> OVERFLOW
              PS(ON STACK) = 002,      STACK POINTER = R3
*****
TST27:  JSR      R5,      PUSHR      ;PUSH 4 WORDS ONTO R3 STACK, SET PRIORITY
        .WORD    177452,125252      ;SECOND OPERAND ON TOP
        .WORD    077652,125253      ;FIRST OPERAND ON BOTTOM
        .WORD    015                ;PROCESSOR PRIORITY LEVEL
        .WORD    ISR27, 344          ;FIS TRAP VECTOR
        MOV      #STACK0,R3         ;SET UP R3 AS STACK POINTER

        NOP
        FSUB+   R3                  ;FLOATING SUBTRACT ON THE R3 STACK

RTA27:  JSR      %7,      POPR        ;POP THE "ANSWER"
        MOV      R3,      $SSP       ;SAVE STACK POINTER (R3)
        HLT+2
        BR
        END27

ISR27:  JSR      %7,      POPER       ;POP ALL DATA OFF THE STACKS
        MOV      R3,      $SSP       ;SAVE STACK POINTER (R3)
        CMP      #344,      $PSW     ;CHECK PS AFTER FIS TRAP
        BEQ
        .+4                          ;BRANCH IF OK
        HLT
        ;PS AFTER FIS TRAP NOT EQUAL TO 344

        CMP      #STACK0,$SSP        ;CHECK THE STACK POINTER (R3)
        BEQ
        .+4                          ;BRANCH IF OK
        HLT
        ;STACK POINTER (R3) NOT EQUAL TO #STACK0

        CMP      #RTA27, ANS1        ;CHECK FIS TRAP RETURN ADDRESS
        BEQ
        .+4                          ;BRANCH IF OK
        HLT+1
        ;FIS TRAP AT WRONG ADDRESS

        CMP      #002,      ANS2     ;CHECK PS BEFORE FIS TRAP
        BEQ
        .+4                          ;BRANCH IF OK
        HLT+2
        ;PS AT FIS TRAP TIME NOT 002

        CMP      #177452,ANS3        ;CHECK DATA FROM THE STACK
        BEQ
        .+4                          ;BRANCH IF OK
        HLT+4
        ;DATA ON STACK (177452) CHANGED

        CMP      #125252,ANS4        ;CHECK DATA FROM STACK
        BEQ
        .+4                          ;BRANCH IF OK
        HLT+4
        ;DATA ON STACK (125252) CHANGED

        CMP      #077652,ANS5        ;CHECK DATA FROM STACK
        BEQ
        .+4                          ;BRANCH IF OK
        HLT+6
        ;DATA ON STACK (077652) CHANGED

        CMP      #125253,ANS6        ;CHECK DATA FROM STACK
        BEQ
        .+4                          ;BRANCH IF OK
        HLT+6
        ;DATA ON STACK (125253) CHANGED
    
```



```

1511 005364 122767 000027 173406 END27:  CMPB  #27,  ICNT  ;CHECK THE TEST NUMBER
1512 005372 001401                BEQ    .+4    ;BRANCH IF OK
1513 005374 104000                HLT                    ;WRONG TEST! PC MUST HAVE FOULED UP.
1514
1515 005376 104400                SCOPE
1516
1517
1518
1519
1520
1521
1522
1523
1524 005400 004567 010704  TST30:  JSR    R5,  PUSHR  ;PUSH 4 WORDS ONTO R3 STACK, SET PRIORITY
1525 005404 043125 052525        .WORD  043125,052525 ;SECOND OPERAND ON TOP
1526 005410 035152 125252        .WORD  035152,125252 ;FIRST OPERAND ON BOTTOM
1527 005414 000147                .WORD  147           ;PROCESSOR PRIORITY LEVEL
1528 005416 016574 000340        .WORD  TRAPER, 340   ;FIS TRAP VECTOR
1529 005422 012703 000630        MOV    #STACK0,R3   ;SET UP STACK POINTER
1530
1531 005426 000240                NOP
1532 005430 075013                FSUB+  R3           ;FLOATING SUBTRACT ON THE R3 STACK
1533
1534 005432 004767 010704        JSR    PC,  POPR    ;POP THE ANSWER
1535 005436 010367 173140        MOV    R3,  SSP    ;SAVE "STACK POINTER"
1536 005442 022767 000150 173130    CMP    #150, SPSW  ;CHECK PS (EXCEPT T BIT)
1537 005450 001401                BEQ    .+4    ;BRANCH IF OK
1538 005452 104000                HLT                    ;PS NOT EQUAL TO 150
1539
1540 005454 022767 000634 173120    CMP    #STACK4,SSP ;CHECK THE STACK POINTER (R3)
1541 005462 001401                BEQ    .+4    ;BRANCH IF OK
1542 005464 104000                HLT                    ;STACK POINTER (R3) NOT EQUAL TO #STACK4
1543
1544 005466 022767 143125 173110    CMP    #143125,ANS1 ;CHECK FIRST HALF OF ANSWER
1545 005474 001401                BEQ    .+4    ;BRANCH IF OK
1546 005476 104002                HLT+2          ;ANS1 NOT EQUAL TO 143125
1547
1548 005500 022767 052524 173100    CMP    #052524,ANS2 ;CHECK SECOND HALF OF ANSWER
1549 005506 001401                BEQ    .+4    ;BRANCH IF OK
1550 005510 104002                HLT+2          ;ANS2 NOT EQUAL TO 052524
1551
1552 005512 122767 000030 173260 END30:  CMPB  #30,  ICNT  ;CHECK THE TEST NUMBER
1553 005520 001401                BEQ    .+4    ;BRANCH IF OK
1554 005522 104000                HLT                    ;WRONG TEST! PC MUST HAVE FOULED UP.
1555
1556 005524 104400                SCOPE
1557

```

```

:*****
:TEST 30:  FSUB (KE11F FLOATING SUBTRACT INSTRUCTION)
:          035152,125252 - 043125,052525 = 143125,052524
:          PS = 150,          STACK POINTER = R3
:*****

```

```

1558
1559
1560
1561
1562
1563
1564
1565 005526 004567 010556
1566 005532 135152 125252
1567 005536 143325 052525
1568 005542 000243
1569 005544 016574 000340
1570 005550 012700 000630
1571
1572 005554 000240
1573 005556 075010
1574
1575 005560 004767 010556
1576 005564 010067 173012
1577 005570 022767 000250 173002
1578 005576 001401
1579 005600 104000
1580
1581 005602 022767 000634 172772
1582 005610 001401
1583 005612 104000
1584
1585 005614 022767 143325 172762
1586 005622 001401
1587 005624 104002
1588
1589 005626 022767 052525 172752
1590 005634 001401
1591 005636 104002
1592
1593 005640 122767 000031 173132 END31:
1594 005646 001401
1595 005650 104000
1596
1597 005652 104400
1598

```

```

*****
:TEST 31: FSUB (KE11F FLOATING SUBTRACT INSTRUCTION)
: 143325,052525 - 135152,125252 = 143325,052525
: PS = 250, STACK POINTER = R0
*****
TST31: JSR R5, PUSHR ;PUSH 4 WORDS ONTO R0 STACK, SET PRIORITY
        .WORD 135152,125252 ;SECOND OPERAND ON TOP
        .WORD 143325,052525 ;FIRST OPERAND ON BOTTOM
        .WORD 243 ;PROCESSOR PRIORITY LEVEL
        .WORD TRAPER, 340 ;FIS TRAP VECTOR
        MOV #STACK0,R0 ;SET UP STACK POINTER

        NOP
        FSUB+ R0 ;FLOATING SUBTRACT ON THE R0 STACK

        JSR PC, POPR ;POP THE ANSWER
        MOV R0, $SP ;SAVE "STACK POINTER"
        CMP #250, $PSW ;CHECK PS (EXCEPT T BIT)
        BEQ .+4 ;BRANCH IF OK
        HLT ;PS NOT EQUAL TO 250

        CMP #STACK4,$SP ;CHECK THE STACK POINTER (R0)
        BEQ .+4 ;BRANCH IF OK
        HLT ;STACK POINTER (R0) NOT EQUAL TO #STACK4

        CMP #143325,ANS1 ;CHECK FIRST HALF OF ANSWER
        BEQ .+4 ;BRANCH IF OK
        HLT+2 ;ANS1 NOT EQUAL TO 143325

        CMP #052525,ANS2 ;CHECK SECOND HALF OF ANSWER
        BEQ .+4 ;BRANCH IF OK
        HLT+2 ;ANS2 NOT EQUAL TO 052525

        CMPB #31, ICNT ;CHECK THE TEST NUMBER
        BEQ .+4 ;BRANCH IF OK
        HLT ;WRONG TEST! PC MUST HAVE FOULED UP.

        SCOPE

```



1599  
1600  
1601  
1602  
1603  
1604  
1605  
1606  
1607  
1608  
1609  
1610  
1611  
1612  
1613  
1614  
1615  
1616  
1617  
1618  
1619  
1620  
1621  
1622  
1623  
1624  
1625  
1626  
1627  
1628  
1629  
1630  
1631  
1632  
1633  
1634  
1635  
1636  
1637  
1638  
1639

005654 004567 010430  
005660 143325 052525  
005664 135152 125252  
005670 000357  
005672 016574 000340  
005676 012705 000630  
  
005702 000240  
005704 075015  
  
005706 004767 010430  
005712 010567 172664  
005716 022767 000340 172654  
005724 001401  
005726 104000  
  
005730 022767 000634 172644  
005736 001401  
005740 104000  
  
005742 022767 043325 172634  
005750 001401  
005752 104002  
  
005754 022767 052525 172624  
005762 001401  
005764 104002  
  
005766 122767 000032 173004  
005774 001401  
005776 104000  
  
006000 104400

\*\*\*\*\*  
:TEST 32: FSUB (KE11F FLOATING SUBTRACT INSTRUCTION)  
: 135152,125252 - 143325,052525 = 043325,052525  
: PS = 340, STACK POINTER = R5  
:\*\*\*\*\*

TST32: JSR R5, PUSHR ;PUSH 4 WORDS ONTO R5 STACK, SET PRIORITY  
.WORD 143325,052525 ;SECOND OPERAND ON TOP  
.WORD 135152,125252 ;FIRST OPERAND ON BOTTOM  
.WORD 357 ;PROCESSOR PRIORITY LEVEL  
.WORD TRAPER, 340 ;FIS TRAP VECTOR  
MOV #STACK0,R5 ;SET UP STACK POINTER  
  
NOP  
FSUB+ R5 ;FLOATING SUBTRACT ON THE R5 STACK  
  
JSR PC, POPR ;POP THE ANSWER  
MOV R5, \$SP ;SAVE "STACK POINTER"  
CMP #340, \$PSW ;CHECK PS (EXCEPT T BIT)  
BEQ .+4 ;BRANCH IF OK  
HLT ;PS NOT EQUAL TO 340  
  
CMP #STACK4,\$SP ;CHECK THE STACK POINTER (R5)  
BEQ .+4 ;BRANCH IF OK  
HLT ;STACK POINTER (R5) NOT EQUAL TO #STACK4  
  
CMP #043325,ANS1 ;CHECK FIRST HALF OF ANSWER  
BEQ .+4 ;BRANCH IF OK  
HLT+2 ;ANS1 NOT EQUAL TO 043325  
  
CMP #052525,ANS2 ;CHECK SECOND HALF OF ANSWER  
BEQ .+4 ;BRANCH IF OK  
HLT+2 ;ANS2 NOT EQUAL TO 052525  
  
END32: CMPB #32, ICNT ;CHECK THE TEST NUMBER  
BEQ .+4 ;BRANCH IF OK  
HLT ;WRONG TEST! PC MUST HAVE FOULED UP.  
  
SCOPE

1640  
1641  
1642  
1643  
1644  
1645  
1646  
1647  
1648  
1649  
1650  
1651  
1652  
1653  
1654  
1655  
1656  
1657  
1658  
1659  
1660  
1661  
1662  
1663  
1664  
1665  
1666  
1667  
1668  
1669  
1670  
1671  
1672  
1673  
1674  
1675  
1676  
1677  
1678  
1679  
1680

```

*****
:TEST 33:          FSUB (KE11F FLOATING SUBTRACT INSTRUCTION)
:          043125,052525 - 035152,125252 = 043125,052524
:          PS = 040,          STACK POINTER = R2
*****
TST33: JSR      R5,          PUSHR      ;PUSH 4 WORDS ONTO R2 STACK, SET PRIORITY
        .WORD   035152,125252        ;SECOND OPERAND ON TOP
        .WORD   043125,052525        ;FIRST OPERAND ON BOTTOM
        .WORD   040                    ;PROCESSOR PRIORITY LEVEL
        .WORD   TRAPER, 340          ;FIS TRAP VECTOR
        MOV     #STACK0,R2          ;SET UP STACK POINTER

        NOP
        FSUB+  R2                    ;FLOATING SUBTRACT ON THE R2 STACK

        JSR     PC,          POPR      ;POP THE ANSWER
        MOV     R2,          $SP      ;SAVE "STACK POINTER"
        CMP     #040,        $PSW     ;CHECK PS (EXCEPT T BIT)
        BEQ     .+4           ;BRANCH IF OK
        HLT
        ;PS NOT EQUAL TO 040

        CMP     #STACK4,$SP          ;CHECK THE STACK POINTER (R2)
        BEQ     .+4           ;BRANCH IF OK
        HLT
        ;STACK POINTER (R2) NOT EQUAL TO #STACK4

        CMP     #043125,ANS1        ;CHECK FIRST HALF OF ANSWER
        BEQ     .+4           ;BRANCH IF OK
        HLT+2
        ;ANS1 NOT EQUAL TO 043125

        CMP     #052524,ANS2        ;CHECK SECOND HALF OF ANSWER
        BEQ     .+4           ;BRANCH IF OK
        HLT+2
        ;ANS2 NOT EQUAL TO 052524

        EN33: CMPB    #33,          ICNT ;CHECK THE TEST NUMBER
        BEQ     .+4           ;BRANCH IF OK
        HLT
        ;WRONG TEST! PC MUST HAVE FOULED UP.

        SCOPE
    
```



1681  
1682  
1683  
1684  
1685  
1686  
1687  
1688  
1689  
1690  
1691  
1692  
1693  
1694  
1695  
1696  
1697  
1698  
1699  
1700  
1701  
1702  
1703  
1704  
1705  
1706  
1707  
1708  
1709  
1710  
1711  
1712  
1713  
1714  
1715  
1716  
1717  
1718  
1719  
1720  
1721

006130 004567 010154  
006134 000000 000000  
006140 000000 000000  
006144 000111  
006146 016574 000340  
006152 012704 000630  
  
006156 000240  
006160 075024  
  
006162 004767 010154  
006166 010467 172410  
006172 022767 000104 172400  
006200 001401  
006202 104000  
  
006204 022767 000634 172370  
006212 001401  
006214 104000  
  
006216 005767 172362  
006222 001401  
006224 104002  
  
006226 005767 172354  
006232 001401  
006234 104002  
  
006236 122767 000034 172534  
006244 001401  
006246 104000  
  
006250 104400

\*\*\*\*\*  
TEST 34: FMUL (KE11F FLOATING MULTIPLY INSTRUCTION)  
000000,000000 \* 000000,000000 = 000000,000000  
PS = 104, STACK POINTER = R4  
\*\*\*\*\*

TST34: JSR R5, PUSHR ;PUSH 4 WORDS ONTO R4 STACK, SET PRIORITY  
.WORD 000000,000000 ;SECOND OPERAND ON TOP  
.WORD 000000,000000 ;FIRST OPERAND ON BOTTOM  
.WORD 111 ;PROCESSOR PRIORITY LEVEL  
.WORD TRAPER, 340 ;FIS TRAP VECTOR  
MOV #STACK0,R4 ;SET UP STACK POINTER  
  
NOP  
FMUL+ R4 ;FLOATING MULTIPLY ON THE R4 STACK  
  
JSR PC, POPR ;POP THE ANSWER  
MOV R4, SSP ;SAVE "STACK POINTER"  
CMP #104, SPSW ;CHECK PS (EXCEPT T BIT)  
BEQ .+4 ;BRANCH IF OK  
HLT ;PS NOT EQUAL TO 104  
  
CMP #STACK4,SSP ;CHECK THE STACK POINTER (R4)  
BEQ .+4 ;BRANCH IF OK  
HLT ;STACK POINTER (R4) NOT EQUAL TO #STACK4  
  
TST ANS1 ;CHECK FIRST HALF OF ANSWER  
BEQ .+4 ;BRANCH IF OK  
HLT+2 ;ANS1 NOT EQUAL TO 000000  
  
TST ANS2 ;CHECK SECOND HALF OF ANSWER  
BEQ .+4 ;BRANCH IF OK  
HLT+2 ;ANS2 NOT EQUAL TO 000000  
  
END34: CMPB #34, ICNT ;CHECK THE TEST NUMBER  
BEQ .+4 ;BRANCH IF OK  
HLT ;WRONG TEST! PC MUST HAVE FOULED UP.  
  
SCOPE

1722  
1723  
1724  
1725  
1726  
1727  
1728  
1729  
1730  
1731  
1732  
1733  
1734  
1735  
1736  
1737  
1738  
1739  
1740  
1741  
1742  
1743  
1744  
1745  
1746  
1747  
1748  
1749  
1750  
1751  
1752  
1753  
1754  
1755  
1756  
1757  
1758  
1759  
1760  
1761  
1762

006252 004567 010032  
006256 052345 123456  
006262 140200 000000  
006266 000343  
006270 016574 000340  
006274 012702 000630  
  
006300 000240  
006302 075022  
  
006304 004767 010032  
006310 010267 172266  
006314 022767 000350 172256  
006322 001401  
006324 104000  
  
006326 022767 000634 172246  
006334 001401  
006336 104000  
  
006340 022767 152345 172236  
006346 001401  
006350 104002  
  
006352 022767 123456 172226  
006360 001401  
006362 104002  
  
006364 122767 000035 172406  
006372 001401  
006374 104000  
  
006376 104400

```

*****
TEST 35:      FMUL (KE11F FLOATING MULTIPLY INSTRUCTION)
              140200,000000 * 052345,123456 = 152345,123456
              PS = 350,      STACK POINTER = R2
*****
TST35:  JSR      R5,      PUSH4      ;PUSH 4 WORDS ONTO R2 STACK, SET PRIORITY
        .WORD   052345,123456      ;SECOND OPERAND ON TOP
        .WORD   140200,000000      ;FIRST OPERAND ON BOTTOM
        .WORD   343                ;PROCESSOR PRIORITY LEVEL
        .WORD   TRAPER, 340        ;FIS TRAP VECTOR
        MOV     #STACK0,R2        ;SET UP STACK POINTER

        NOP
        FMUL+  R2                ;FLOATING MULTIPLY ON THE R2 STACK

        JSR     PC,      POPR      ;POP THE ANSWER
        MOV     R2,      $SP      ;SAVE "STACK POINTER"
        CMP     #350,    $PSW     ;CHECK PS (EXCEPT T BIT)
        BEQ    .+4            ;BRANCH IF OK
        HLT
        ;PS NOT EQUAL TO 350

        CMP     #STACK4,$SP      ;CHECK THE STACK POINTER (R2)
        BEQ    .+4            ;BRANCH IF OK
        HLT
        ;STACK POINTER (R2) NOT EQUAL TO #STACK4

        CMP     #152345,ANS1     ;CHECK FIRST HALF OF ANSWER
        BEQ    .+4            ;BRANCH IF OK
        HLT+2
        ;ANS1 NOT EQUAL TO 152345

        CMP     #123456,ANS2     ;CHECK SECOND HALF OF ANSWER
        BEQ    .+4            ;BRANCH IF OK
        HLT+2
        ;ANS2 NOT EQUAL TO 123456

        EN35:  CMPB    #35,      ICNT ;CHECK THE TEST NUMBER
        BEQ    .+4            ;BRANCH IF OK
        HLT
        ;WRONG TEST! PC MUST HAVE FOULED UP.

        SCOPE
    
```



1763  
 1764  
 1765  
 1766  
 1767  
 1768  
 1769  
 1770 006400 004567 007704  
 1771 006404 135753 024642  
 1772 006410 100125 052525  
 1773 006414 000117  
 1774 006416 016574 000340  
 1775 006422 012705 000630  
 1776  
 1777 006426 000240  
 1778 006430 075025  
 1779  
 1780 006432 004767 007704  
 1781 006436 010567 172140  
 1782 006442 022767 000104 172130  
 1783 006450 001401  
 1784 006452 104000  
 1785  
 1786 006454 022767 000634 172120  
 1787 006462 001401  
 1788 006464 104000  
 1789  
 1790 006466 005767 172112  
 1791 006472 001401  
 1792 006474 104002  
 1793  
 1794 006476 005767 172104  
 1795 006502 001401  
 1796 006504 104002  
 1797  
 1798 006506 122767 000036 172264 END36:  
 1799 006514 001401  
 1800 006516 104000  
 1801  
 1802 006520 104400  
 1803

```

*****
:TEST 36:      FMUL (KE11F FLOATING MULTIPLY INSTRUCTION)
:      100125,052525 * 135753,024642 = 000000,000000
:      PS = 104,      STACK POINTER = R5
*****
TST36:  JSR      R5,      PUSHR      ;PUSH 4 WORDS ONTO R5 STACK, SET PRIORITY
        .WORD    135753,024642      ;SECOND OPERAND ON TOP
        .WORD    100125,052525      ;FIRST OPERAND ON BOTTOM
        .WORD    117                  ;PROCESSOR PRIORITY LEVEL
        .WORD    TRAPER, 340         ;FIS TRAP VECTOR
        MOV      #STACK0,R5         ;SET UP STACK POINTER

        NOP
        FMUL+   R5                  ;FLOATING MULTIPLY ON THE R5 STACK

        JSR      PC,      POPR      ;POP THE ANSWER
        MOV      R5,      $SP      ;SAVE "STACK POINTER"
        CMP      #104,     $PSW     ;CHECK PS (EXCEPT T BIT)
        BEQ      .+4              ;BRANCH IF OK
        HLT
        ;PS NOT EQUAL TO 104

        CMP      #STACK4,$SP      ;CHECK THE STACK POINTER (R5)
        BEQ      .+4              ;BRANCH IF OK
        HLT
        ;STACK POINTER (R5) NOT EQUAL TO #STACK4

        TST      ANS1
        BEQ      .+4              ;CHECK FIRST HALF OF ANSWER
        HLT+2
        ;BRANCH IF OK
        ;ANS1 NOT EQUAL TO 000000

        TST      ANS2
        BEQ      .+4              ;CHECK SECOND HALF OF ANSWER
        HLT+2
        ;BRANCH IF OK
        ;ANS2 NOT EQUAL TO 000000

        CMPB    #36,      ICNT     ;CHECK THE TEST NUMBER
        BEQ      .+4              ;BRANCH IF OK
        HLT
        ;WRONG TEST! PC MUST HAVE FOULED UP.

        SCOPE
    
```

```

1804
1805
1806
1807
1808
1809
1810
1811 006522 004567 007562
1812 006526 000052 125252
1813 006532 161616 161616
1814 006536 000217
1815 006540 016574 000340
1816 006544 012703 000630
1817
1818 006550 000240
1819 006552 075023
1820
1821 006554 004767 007562
1822 006560 010367 172016
1823 006564 022767 000204 172006
1824 006572 001401
1825 006574 104000
1826
1827 006576 022767 000634 171776
1828 006604 001401
1829 006606 104000
1830
1831 006610 005767 171770
1832 006614 001401
1833 006616 104002
1834
1835 006620 005767 171762
1836 006624 001401
1837 006626 104002
1838
1839 006630 122767 000037 172142 END37:
1840 006636 001401
1841 006640 104000
1842
1843 006642 104400
1844

```

```

*****
:TEST 37: FMUL (KE11F FLOATING MULTIPLY INSTRUCTION)
:      161616,161616 * 000052,125252 = 000000,000000
:      PS = 204,          STACK POINTER = R3
*****
TST37: JSR      R5,      PUSHR      ;PUSH 4 WORDS ONTO R3 STACK, SET PRIORITY
        .WORD    000052,125252      ;SECOND OPERAND ON TOP
        .WORD    161616,161616      ;FIRST OPERAND ON BOTTOM
        .WORD    217                  ;PROCESSOR PRIORITY LEVEL
        .WORD    TRAPER, 340         ;FIS TRAP VECTOR
        MOV      #STACK0,R3         ;SET UP STACK POINTER

        NOP
        FMUL+   R3                  ;FLOATING MULTIPLY ON THE R3 STACK

        JSR      PC,      POPR       ;POP THE ANSWER
        MOV      R3,      SSP        ;SAVE "STACK POINTER"
        CMP      #204,    SPSW       ;CHECK PS (EXCEPT T BIT)
        BEQ      .+4               ;BRANCH IF OK
        HLT
        ;PS NOT EQUAL TO 204

        CMP      #STACK4,SSP        ;CHECK THE STACK POINTER (R3)
        BEQ      .+4               ;BRANCH IF OK
        HLT
        ;STACK POINTER (R3) NOT EQUAL TO #STACK4

        TST      ANS1
        BEQ      .+4               ;CHECK FIRST HALF OF ANSWER
        HLT+2
        ;BRANCH IF OK
        ;ANS1 NOT EQUAL TO 000000

        TST      ANS2
        BEQ      .+4               ;CHECK SECOND HALF OF ANSWER
        HLT+2
        ;BRANCH IF OK
        ;ANS2 NOT EQUAL TO 000000

        CMPB    #37,      ICNT       ;CHECK THE TEST NUMBER
        BEQ      .+4               ;BRANCH IF OK
        HLT
        ;WRONG TEST! PC MUST HAVE FOULED UP.

        SCOPE

```



1845  
1846  
1847  
1848  
1849  
1850  
1851  
1852  
1853  
1854  
1855  
1856  
1857  
1858  
1859  
1860  
1861  
1862  
1863  
1864  
1865  
1866  
1867  
1868  
1869  
1870  
1871  
1872  
1873  
1874  
1875  
1876  
1877  
1878  
1879  
1880  
1881  
1882  
1883  
1884  
1885

006644 004567 007262  
006650 041500 000000  
006654 176452 125252  
006660 000357  
006662 016574 000340  
  
006666 000240  
006670 075026  
  
006672 004767 007274  
006676 022706 000500  
006702 001404  
006704 012706 000500  
006710 104000  
006712 000417  
  
006714 022767 000350 171656  
006722 001401  
006724 104000  
  
006726 022767 177777 171650  
006734 001401  
006736 104002  
  
006740 022767 177777 171640  
006746 001401  
006750 104002  
  
006752 122767 000040 172020  
006760 001401  
006762 104000  
  
006764 104400

```

*****
:TEST 40:      FMUL (KE11F FLOATING MULTIPLY INSTRUCTION)
:      176452,125252 * 041500,000000 = 177777,177777
:      PS = 350,      STACK POINTER = SP
*****
    
```

```

TST40: JSR      R5,      PUSH5      ;PUSH 4 WORDS ONTO STACK, SET PRIORITY
        .WORD    041500,000000      ;SECOND OPERAND ON TOP
        .WORD    176452,125252      ;FIRST OPERAND ON BOTTOM
        .WORD    357                  ;PROCESSOR PRIORITY LEVEL
        .WORD    TRAPER, 340         ;FIS TRAP VECTOR

        NOP
        FMUL+   SP                  ;FLOATING MULTIPLY ON THE STACK

        JSR      PC,      POPS
        CMP      #500,     SP        ;POP THE ANSWER
        BEQ      TSA40,     SP        ;CHECK THE STACK POINTER
        MOV      #500,     SP        ;BRANCH IF OK
        HLT
        BR       END40              ;RESTORE STACK POINTER
                                        ;STACK POINTER FOULED UP
                                        ;SKIP REST OF TEST

        CMP      #350,     SPSW      ;CHECK PS (EXCEPT T BIT)
        BEQ      .+4,
        HLT
                                        ;BRANCH IF OK
                                        ;PS NOT EQUAL TO 350

        CMP      #177777,ANS1       ;CHECK FIRST HALF OF ANSWER
        BEQ      .+4,
        HLT+2
                                        ;BRANCH IF OK
                                        ;ANS1 NOT EQUAL TO 177777

        CMP      #177777,ANS2       ;CHECK SECOND HALF OF ANSWER
        BEQ      .+4,
        HLT+2
                                        ;BRANCH IF OK
                                        ;ANS2 NOT EQUAL TO 177777

        CMPB     #40,      ICNT      ;CHECK THE TEST NUMBER
        BEQ      .+4,
        HLT
                                        ;BRANCH IF OK
                                        ;WRONG TEST! PC MUST HAVE FOULED UP.

        SCOPE
    
```

1886  
1887  
1888  
1889  
1890  
1891  
1892  
1893  
1894  
1895  
1896  
1897  
1898  
1899  
1900  
1901  
1902  
1903  
1904  
1905  
1906  
1907  
1908  
1909  
1910  
1911  
1912  
1913  
1914  
1915  
1916  
1917  
1918  
1919  
1920  
1921  
1922  
1923  
1924  
1925  
1926  
1927  
1928  
1929  
1930  
1931  
1932  
1933  
1934  
1935  
1936  
1937  
1938  
1939  
1940  
1941

\*\*\*\*\*  
:TEST 41: FMUL (KE11F FLOATING MULTIPLY INSTRUCTION)  
: 076452,125252 \* 041500,000001 ==> OVERFLOW  
: PS(ON STACK) = 102, STACK POINTER = SP  
:\*\*\*\*\*

TST41: JSR R5, PUSH5 ;PUSH 4 WORDS ONTO STACK, SET PRIORITY  
.WORD 041500,000001 ;SECOND OPERAND ON TOP  
.WORD 076452,125252 ;FIRST OPERAND ON BOTTOM  
.WORD 105 ;PROCESSOR PRIORITY LEVEL  
.WORD ISR41, 357 ;FIS TRAP VECTOR

NOP  
FMUL+ SP ;FLOATING MULTIPLY ON THE STACK

RTA41: JSR %7, POPS ;POP THE "ANSWER"  
HLT+2 ;FIS TRAP DIDN'T OCCURE!  
MOV #500, SP ;RESTORE THE STACK POINTER  
BR END41

ISR41: JSR %7, POPES ;POP ALL DATA OFF THE STACK  
CMP #500, SP ;CHECK THE STACK POINTER  
BEQ ISA41 ;BRANCH IF OK  
MOV #500, SP ;RESTORE THE STACK POINTER  
HLT ;STACK POINTER FOULED UP  
BR END41 ;SKIP REST OF TEST

ISA41: CMP #357, SPSW ;CHECK PS AFTER FIS TRAP  
BEQ .+4 ;BRANCH IF OK  
HLT ;PS AFTER FIS TRAP NOT EQUAL TO 357

CMP #RTA41, ANS1 ;CHECK FIS TRAP RETURN ADDRESS  
BEQ .+4 ;BRANCH IF OK  
HLT+1 ;FIS TRAP AT WRONG ADDRESS

CMP #102, ANS2 ;CHECK PS BEFORE FIS TRAP  
BEQ .+4 ;BRANCH IF OK  
HLT+2 ;PS AT FIS TRAP TIME NOT 102

CMP #041500, ANS3 ;CHECK DATA FROM THE STACK  
BEQ .+4 ;BRANCH IF OK  
HLT+4 ;DATA ON STACK (041500) CHANGED

CMP #000001, ANS4 ;CHECK DATA FROM STACK  
BEQ .+4 ;BRANCH IF OK  
HLT+4 ;DATA ON STACK (000001) CHANGED

CMP #076452, ANS5 ;CHECK DATA FROM STACK  
BEQ .+4 ;BRANCH IF OK  
HLT+6 ;DATA ON STACK (076452) CHANGED

CMP #125252, ANS6 ;CHECK DATA FROM STACK  
BEQ .+4 ;BRANCH IF OK  
HLT+6 ;DATA ON STACK (125252) CHANGED



```

1942 007160 122767 000041 171612 END41: CMPB #41, ICNT ;CHECK THE TEST NUMBER
1943 007166 001401 BEQ .+4 ;BRANCH IF OK
1944 007170 104000 HLT ;WRONG TEST! PC MUST HAVE FOULED UP.
1945
1946 007172 104400 SCOPE
1947
1948
1949
1950 ;*****
1951 ;TEST 42: FMUL (KE11F FLOATING MULTIPLY INSTRUCTION)
1952 ; 124252,125252 * 114100,000001 = 000200,000000
1953 ; PS = 200, STACK POINTER = R1
1954 ;*****
1955 007174 004567 007110 TST42: JSR R5, PUSHR ;PUSH 4 WORDS ONTO R1 STACK, SET PRIORITY
1956 007200 114100 000001 .WORD 114100,000001 ;SECOND OPERAND ON TOP
1957 007204 124252 125252 .WORD 124252,125252 ;FIRST OPERAND ON BOTTOM
1958 007210 000200 .WORD 200 ;PROCESSOR PRIORITY LEVEL
1959 007212 016574 000340 .WORD TRAPER, 340 ;FIS TRAP VECTOR
1960 007216 012701 000630 MOV #STACK0,R1 ;SET UP STACK POINTER
1961
1962 007222 000240 NOP
1963 007224 075021 FMUL+ R1 ;FLOATING MULTIPLY ON THE R1 STACK
1964
1965 007226 004767 007110 JSR PC, POPR ;POP THE ANSWER
1966 007232 010167 171344 MOV R1, $SP ;SAVE "STACK POINTER"
1967 007236 022767 000200 171334 CMP #200, $PSW ;CHECK PS (EXCEPT T BIT)
1968 007244 001401 BEQ .+4 ;BRANCH IF OK
1969 007246 104000 HLT ;PS NOT EQUAL TO 200
1970
1971 007250 022767 000634 171324 CMP #STACK4,$SP ;CHECK THE STACK POINTER (R1)
1972 007256 001401 BEQ .+4 ;BRANCH IF OK
1973 007260 104000 HLT ;STACK POINTER (R1) NOT EQUAL TO #STACK4
1974
1975 007262 022767 000200 171314 CMP #000200,ANS1 ;CHECK FIRST HALF OF ANSWER
1976 007270 001401 BEQ .+4 ;BRANCH IF OK
1977 007272 104002 HLT+2 ;ANS1 NOT EQUAL TO 000200
1978
1979 007274 005767 171306 TST ANS2 ;CHECK SECOND HALF OF ANSWER
1980 007300 001401 BEQ .+4 ;BRANCH IF OK
1981 007302 104002 HLT+2 ;ANS2 NOT EQUAL TO 000000
1982
1983 007304 122767 000042 171466 END42: CMPB #42, ICNT ;CHECK THE TEST NUMBER
1984 007312 001401 BEQ .+4 ;BRANCH IF OK
1985 007314 104000 HLT ;WRONG TEST! PC MUST HAVE FOULED UP.
1986
1987 007316 104400 SCOPE
1988

```

1989  
 1990  
 1991  
 1992  
 1993  
 1994  
 1995  
 1996 007320 004567 006764  
 1997 007324 114100 000000  
 1998 007330 024252 125252  
 1999 007334 000305  
 2000 007336 007366 000057  
 2001 007342 012700 000630  
 2002  
 2003 007346 000240  
 2004 007350 075020  
 2005  
 2006 007352 004767 006764  
 2007 007356 010067 171220  
 2008 007362 104002  
 2009 007364 000453  
 2010  
 2011 007366 004767 007002  
 2012 007372 010067 171204  
 2013 007376 022767 000057 171174  
 2014 007404 001401  
 2015 007406 104000  
 2016  
 2017 007410 022767 000630 171164  
 2018 007416 001401  
 2019 007420 104000  
 2020  
 2021 007422 022767 007352 171154  
 2022 007430 001401  
 2023 007432 104001  
 2024  
 2025 007434 022767 000312 171144  
 2026 007442 001401  
 2027 007444 104002  
 2028  
 2029 007446 022767 114100 171134  
 2030 007454 001401  
 2031 007456 104004  
 2032  
 2033 007460 005767 171126  
 2034 007464 001401  
 2035 007466 104004  
 2036  
 2037 007470 022767 024252 171116  
 2038 007476 001401  
 2039 007500 104006  
 2040  
 2041 007502 022767 125252 171106  
 2042 007510 001401  
 2043 007512 104006  
 2044

```

*****
TEST 43:      FMUL (KE11F FLOATING MULTIPLY INSTRUCTION)
              024252,125252 * 114100,000000 ==> UNDERFLOW
              PS(ON STACK) = 312,      STACK POINTER = R0
*****
    
```

```

TST43:  JSR      R5,      PUSHR      ;PUSH 4 WORDS ONTO R0 STACK, SET PRIORITY
          .WORD    114100,000000      ;SECOND OPERAND ON TOP
          .WORD    024252,125252      ;FIRST OPERAND ON BOTTOM
          .WORD    305                  ;PROCESSOR PRIORITY LEVEL
          .WORD    ISR43, 057          ;FIS TRAP VECTOR
          MOV      #STACK0,R0          ;SET UP R0 AS STACK POINTER

          NOP
          FMUL+   R0                  ;FLOATING MULTIPLY ON THE R0 STACK

RTA43:   JSR      %7,      POPR       ;POP THE "ANSWER"
          MOV      R0,      SSP       ;SAVE STACK POINTER (R0)
          HLT+2
          BR       END43             ;FIS TRAP DIDN'T OCCURE!

ISR43:   JSR      %7,      POPER      ;POP ALL DATA OFF THE STACKS
          MOV      R0,      SSP       ;SAVE STACK POINTER (R0)
          CMP      #057,      SPSW    ;CHECK PS AFTER FIS TRAP
          BEQ      .+4              ;BRANCH IF OK
          HLT      ;PS AFTER FIS TRAP NOT EQUAL TO 057

          CMP      #STACK0,SSP        ;CHECK THE STACK POINTER (R0)
          BEQ      .+4              ;BRANCH IF OK
          HLT      ;STACK POINTER (R0) NOT EQUAL TO #STACK0

          CMP      #RTA43, ANS1       ;CHECK FIS TRAP RETURN ADDRESS
          BEQ      .+4              ;BRANCH IF OK
          HLT+1      ;FIS TRAP AT WRONG ADDRESS

          CMP      #312, ANS2        ;CHECK PS BEFORE FIS TRAP
          BEQ      .+4              ;BRANCH IF OK
          HLT+2      ;PS AT FIS TRAP TIME NOT 312

          CMP      #114100,ANS3       ;CHECK DATA FROM THE STACK
          BEQ      .+4              ;BRANCH IF OK
          HLT+4      ;DATA ON STACK (114100) CHANGED

          TST      ANS4              ;CHECK DATA FROM STACK
          BEQ      .+4              ;BRANCH IF OK
          HLT+4      ;DATA ON STACK (000000) CHANGED

          CMP      #024252,ANS5       ;CHECK DATA FROM STACK
          BEQ      .+4              ;BRANCH IF OK
          HLT+6      ;DATA ON STACK (024252) CHANGED

          CMP      #125252,ANS6       ;CHECK DATA FROM STACK
          BEQ      .+4              ;BRANCH IF OK
          HLT+6      ;DATA ON STACK (125252) CHANGED
    
```



```

2045 007514 122767 000043 171256 END43: CMPB #43, ICNT ;CHECK THE TEST NUMBER
2046 007522 001401 BEQ .+4 ;BRANCH IF OK
2047 007524 104000 HLT ;WRONG TEST! PC MUST HAVE FOULED UP.

```

```

2048
2049 007526 104400 SCOPE
2050
2051

```

```

*****
:TEST 44: FDIV (KE11F FLOATING DIVIDE INSTRUCTION)
: 125252,125252 / 140200,000000 = 025252,125252
: PS = 200, STACK POINTER = SP
*****

```

```

2057
2058 007530 004567 006376 TST44: JSR R5, PUSHS ;PUSH 4 WORDS ONTO STACK, SET PRIORITY
2059 007534 140200 000000 .WORD 140200,000000 ;SECOND OPERAND ON TOP
2060 007540 125252 125252 .WORD 125252,125252 ;FIRST OPERAND ON BOTTOM
2061 007544 000217 .WORD 217 ;PROCESSOR PRIORITY LEVEL
2062 007546 016574 000340 .WORD TRAPEX, 340 ;FIS TRAP VECTOR
2063

```

```

2064 007552 000240 NOP
2065 007554 075036 FDIV+ SP ;FLOATING DIVIDE ON THE STACK
2066

```

```

2067 007556 004767 006410 JSR PC, POPS ;POP THE ANSWER
2068 007562 022706 000500 CMP #500, SP ;CHECK THE STACK POINTER
2069 007566 001404 BEQ TSA44 ;BRANCH IF OK
2070 007570 012706 000500 MOV #500, SP ;RESTORE STACK POINTER
2071 007574 104000 HLT ;STACK POINTER FOULED UP
2072 007576 000417 BR END44 ;SKIP REST OF TEST
2073

```

```

2074 007600 022767 000200 170772 TSA44: CMP #200, SPSW ;CHECK PS (EXCEPT T BIT)
2075 007606 001401 BEQ .+4 ;BRANCH IF OK
2076 007610 104000 HLT ;PS NOT EQUAL TO 200
2077

```

```

2078 007612 022767 025252 170764 CMP #025252,ANS1 ;CHECK FIRST HALF OF ANSWER
2079 007620 001401 BEQ .+4 ;BRANCH IF OK
2080 007622 104002 HLT+2 ;ANS1 NOT EQUAL TO 025252
2081

```

```

2082 007624 022767 125252 170754 CMP #125252,ANS2 ;CHECK SECOND HALF OF ANSWER
2083 007632 001401 BEQ .+4 ;BRANCH IF OK
2084 007634 104002 HLT+2 ;ANS2 NOT EQUAL TO 125252
2085

```

```

2086 007636 122767 000044 171134 END44: CMPB #44, ICNT ;CHECK THE TEST NUMBER
2087 007644 001401 BEQ .+4 ;BRANCH IF OK
2088 007646 104000 HLT ;WRONG TEST! PC MUST HAVE FOULED UP.
2089

```

```

2090 007650 104400 SCOPE
2091

```





FDIV TEST SECTION

```

2148 010050 122767 000045 170722 END45: CMPB #45, ICNT ;CHECK THE TEST NUMBER
2149 010056 001401 BEQ .+4 ;BRANCH IF OK
2150 010060 104000 HLT ;WRONG TEST! PC MUST HAVE FOULED UP.
2151
2152 010062 104400 SCOPE
2153
2154
2155
2156 :*****
2157 :TEST 46: FDIV (KE11F FLOATING DIVIDE INSTRUCTION)
2158 : 167452,125251 / 127652,125252 = 077777,177776
2159 : PS = 100, STACK POINTER = RO
2160 :*****
2161 010064 004567 006220 TST46: JSR R5, PUSHR ;PUSH 4 WORDS ONTO RO STACK, SET PRIORITY
2162 010070 127652 125252 .WORD 127652,125252 ;SECOND OPERAND ON TOP
2163 010074 167452 125251 .WORD 167452,125251 ;FIRST OPERAND ON BOTTOM
2164 010100 000111 .WORD 111 ;PROCESSOR PRIORITY LEVEL
2165 010102 016574 000340 .WORD TRAPER, 340 ;FIS TRAP VECTOR
2166 010106 012700 000630 MOV #STACK0,RO ;SET UP STACK POINTER
2167
2168 010112 000240 NOP
2169 010114 075030 FDIV+ RO ;FLOATING DIVIDE ON THE RO STACK
2170
2171 010116 004767 006220 JSR PC, POPR ;POP THE ANSWER
2172 010122 010067 170454 MOV RO, $SP ;SAVE "STACK POINTER"
2173 010126 022767 000100 170444 CMP #100, $PSW ;CHECK PS (EXCEPT T BIT)
2174 010134 001401 BEQ .+4 ;BRANCH IF OK
2175 010136 104000 HLT ;PS NOT EQUAL TO 100
2176
2177 010140 022767 000634 170434 CMP #STACK4,$SP ;CHECK THE STACK POINTER (RO)
2178 010146 001401 BEQ .+4 ;BRANCH IF OK
2179 010150 104000 HLT ;STACK POINTER (RO) NOT EQUAL TO #STACK4
2180
2181 010152 022767 077777 170424 CMP #077777,ANS1 ;CHECK FIRST HALF OF ANSWER
2182 010160 001401 BEQ .+4 ;BRANCH IF OK
2183 010162 104002 HLT+2 ;ANS1 NOT EQUAL TO 077777
2184
2185 010164 022767 177776 170414 CMP #177776,ANS2 ;CHECK SECOND HALF OF ANSWER
2186 010172 001401 BEQ .+4 ;BRANCH IF OK
2187 010174 104002 HLT+2 ;ANS2 NOT EQUAL TO 177776
2188
2189 010176 122767 000046 170574 END46: CMPB #46, ICNT ;CHECK THE TEST NUMBER
2190 010204 001401 BEQ .+4 ;BRANCH IF OK
2191 010206 104000 HLT ;WRONG TEST! PC MUST HAVE FOULED UP.
2192
2193 010210 104400 SCOPE
2194

```

```

2195
2196
2197
2198
2199
2200
2201
2202 010212 004567 006072
2203 010216 127652 125252
2204 010222 067452 125252
2205 010226 000242
2206 010230 010260 000357
2207 010234 012704 000630
2208
2209 010240 000240
2210 010242 075034
2211
2212 010244 004767 006072
2213 010250 010467 170326
2214 010254 104002
2215 010256 000454
2216
2217 010260 004767 006110
2218 010264 010467 170312
2219 010270 022767 000357 170302
2220 010276 001401
2221 010300 104000
2222
2223 010302 022767 000630 170272
2224 010310 001401
2225 010312 104000
2226
2227 010314 022767 010244 170262
2228 010322 001401
2229 010324 104001
2230
2231 010326 022767 000242 170252
2232 010334 001401
2233 010336 104002
2234
2235 010340 022767 127652 170242
2236 010346 001401
2237 010350 104004
2238
2239 010352 022767 125252 170232
2240 010360 001401
2241 010362 104004
2242
2243 010364 022767 067452 170222
2244 010372 001401
2245 010374 104006
2246
2247 010376 022767 125252 170212
2248 010404 001401
2249 010406 104006
2250

```

```

*****
:TEST 47: FDIV (KE11F FLOATING DIVIDE INSTRUCTION)
: 067452,125252 / 127652,125252 ==> OVERFLOW
: PS(ON STACK) = 242, STACK POINTER = R4
*****
TST47: JSR R5, PUSHR ;PUSH 4 WORDS ONTO R4 STACK, SET PRIORITY
: .WORD 127652,125252 ;SECOND OPERAND ON TOP
: .WORD 067452,125252 ;FIRST OPERAND ON BOTTOM
: .WORD 242 ;PROCESSOR PRIORITY LEVEL
: .WORD ISR47, 357 ;FIS TRAP VECTOR
MOV #STACK0,R4 ;SET UP R4 AS STACK POINTER

NOP
FDIV+ R4 ;FLOATING DIVIDE ON THE R4 STACK

RTA47: JSR %7, POPR ;POP THE "ANSWER"
MOV R4, $SP ;SAVE STACK POINTER (R4)
HLT+2 ;FIS TRAP DIDN'T OCCURE!
BR END47

ISR47: JSR %7, POPER ;POP ALL DATA OFF THE STACKS
MOV R4, $SP ;SAVE STACK POINTER (R4)
CMP #357, $PSW ;CHECK PS AFTER FIS TRAP
BEQ .+4 ;BRANCH IF OK
HLT ;PS AFTER FIS TRAP NOT EQUAL TO 357

CMP #STACK0,$SP ;CHECK THE STACK POINTER (R4)
BEQ .+4 ;BRANCH IF OK
HLT ;STACK POINTER (R4) NOT EQUAL TO #STACK0

CMP #RTA47, ANS1 ;CHECK FIS TRAP RETURN ADDRESS
BEQ .+4 ;BRANCH IF OK
HLT+1 ;FIS TRAP AT WRONG ADDRESS

CMP #242, ANS2 ;CHECK PS BEFORE FIS TRAP
BEQ .+4 ;BRANCH IF OK
HLT+2 ;PS AT FIS TRAP TIME NOT 242

CMP #127652,ANS3 ;CHECK DATA FROM THE STACK
BEQ .+4 ;BRANCH IF OK
HLT+4 ;DATA ON STACK (127652) CHANGED

CMP #125252,ANS4 ;CHECK DATA FROM STACK
BEQ .+4 ;BRANCH IF OK
HLT+4 ;DATA ON STACK (125252) CHANGED

CMP #067452,ANS5 ;CHECK DATA FROM STACK
BEQ .+4 ;BRANCH IF OK
HLT+6 ;DATA ON STACK (067452) CHANGED

CMP #125252,ANS6 ;CHECK DATA FROM STACK
BEQ .+4 ;BRANCH IF OK
HLT+6 ;DATA ON STACK (125252) CHANGED

```



```

2251 010410 122767 000047 170362 END47:  CMPB  #47,  ICNT  ;CHECK THE TEST NUMBER
2252 010416 001401  BEQ   .+4   ;BRANCH IF OK
2253 010420 104000  HLT   ;WRONG TEST! PC MUST HAVE FOULED UP.
2254
2255 010422 104400  SCOPE
2256
2257

```

```

*****
:TEST 50:  FDIV (KE11F FLOATING DIVIDE INSTRUCTION)
:          167452,125252 / 027652,125253 = 177777,177777
:          PS = 310,  STACK POINTER = SP
*****

```

```

2263
2264 010424 004567 005502  TST50:  JSR   R5,  PUSHS  ;PUSH 4 WORDS ONTO STACK, SET PRIORITY
2265 010430 027652 125253  .WORD  027652,125253 ;SECOND OPERAND ON TOP
2266 010434 167452 125252  .WORD  167452,125252 ;FIRST OPERAND ON BOTTOM
2267 010440 000300  .WORD  300           ;PROCESSOR PRIORITY LEVEL
2268 010442 016574 000340  .WORD  TRAPER, 340   ;FIS TRAP VECTOR
2269
2270 010446 000240  NOP
2271 010450 075036  FDIV+  SP           ;FLOATING DIVIDE ON THE STACK
2272
2273 010452 004767 005514  JSR   PC,  POPS   ;POP THE ANSWER
2274 010456 022706 000500  CMP   #500, SP    ;CHECK THE STACK POINTER
2275 010462 001404  BEQ   TSA50      ;BRANCH IF OK
2276 010464 012706 000500  MOV   #500, SP   ;RESTORE STACK POINTER
2277 010470 104000  HLT   ;STACK POINTER FOULED UP
2278 010472 000417  BR    END50     ;SKIP REST OF TEST
2279
2280 010474 022767 000310 170076  TSA50:  CMP   #310, SPSW  ;CHECK PS (EXCEPT T BIT)
2281 010502 001401  BEQ   .+4       ;BRANCH IF OK
2282 010504 104000  HLT   ;PS NOT EQUAL TO 310
2283
2284 010506 022767 177777 170070  CMP   #177777,ANS1 ;CHECK FIRST HALF OF ANSWER
2285 010514 001401  BEQ   .+4       ;BRANCH IF OK
2286 010516 104002  HLT+2 ;ANS1 NOT EQUAL TO 177777
2287
2288 010520 022767 177777 170060  CMP   #177777,ANS2 ;CHECK SECOND HALF OF ANSWER
2289 010526 001401  BEQ   .+4       ;BRANCH IF OK
2290 010530 104002  HLT+2 ;ANS2 NOT EQUAL TO 177777
2291
2292 010532 122767 000050 170240  END50:  CMPB  #50,  ICNT  ;CHECK THE TEST NUMBER
2293 010540 001401  BEQ   .+4   ;BRANCH IF OK
2294 010542 104000  HLT   ;WRONG TEST! PC MUST HAVE FOULED UP.
2295
2296 010544 104400  SCOPE
2297

```

2298  
2299  
2300  
2301  
2302  
2303  
2304  
2305  
2306  
2307  
2308  
2309  
2310  
2311  
2312  
2313  
2314  
2315  
2316  
2317  
2318  
2319  
2320  
2321  
2322  
2323  
2324  
2325  
2326  
2327  
2328  
2329  
2330  
2331  
2332  
2333  
2334  
2335  
2336  
2337  
2338  
2339  
2340  
2341  
2342  
2343  
2344  
2345  
2346  
2347  
2348  
2349  
2350  
2351  
2352  
2353

010546 004567 005360  
010552 014377 177777  
010556 154200 000000  
010562 000246  
010564 010610 000103  
  
010570 000240  
010572 075036  
  
010574 004767 005372  
010600 104002  
010602 012706 000500  
010606 000453  
  
010610 004767 005412  
010614 022706 000500  
010620 001404  
010622 012706 000500  
010626 104000  
010630 000442  
  
010632 022767 000103 167740  
010640 001401  
010642 104000  
  
010644 022767 010574 167732  
010652 001401  
010654 104001  
  
010656 022767 000242 167722  
010664 001401  
010666 104002  
  
010670 022767 014377 167712  
010676 001401  
010700 104004  
  
010702 022767 177777 167702  
010710 001401  
010712 104004  
  
010714 022767 154200 167672  
010722 001401  
010724 104006  
  
010726 005767 167664  
010732 001401  
010734 104006

```
*****
:TEST 51:          FDIV (KE11F FLOATING DIVIDE INSTRUCTION)
:          154200,000000 / 014377,177777 ==> OVERFLOW
:          PS(ON STACK) = 242,   STACK POINTER = SP
*****

TST51:  JSR      R5,      PUSHS   ;PUSH 4 WORDS ONTO STACK, SET PRIORITY
:          .WORD   014377,177777 ;SECOND OPERAND ON TOP
:          .WORD   154200,000000 ;FIRST OPERAND ON BOTTOM
:          .WORD   246           ;PROCESSOR PRIORITY LEVEL
:          .WORD   ISR51, 103    ;FIS TRAP VECTOR

:          NOP
:          FDIV+  SP           ;FLOATING DIVIDE ON THE STACK

RTA51:  JSR      %7,      POPS    ;POP THE "ANSWER"
:          HLT+2                    ;FIS TRAP DIDN'T OCCURE!
:          MOV      #500, SP      ;RESTORE THE STACK POINTER
:          BR       ENDS1

ISR51:  JSR      %7,      POPES   ;POP ALL DATA OFF THE STACK
:          CMP      #500, SP      ;CHECK THE STACK POINTER
:          BEQ     ISA51          ;BRANCH IF OK
:          MOV      #500, SP      ;RESTORE THE STACK POINTER
:          HLT     BR           ;STACK POINTER FOULED UP
:          BR       ENDS1       ;SKIP REST OF TEST

ISA51:  CMP      #103,  $PSW     ;CHECK PS AFTER FIS TRAP
:          BEQ     .+4           ;BRANCH IF OK
:          HLT     BR           ;PS AFTER FIS TRAP NOT EQUAL TO 103

:          CMP      #RTA51, ANS1  ;CHECK FIS TRAP RETURN ADDRESS
:          BEQ     .+4           ;BRANCH IF OK
:          HLT+1                    ;FIS TRAP AT WRONG ADDRESS

:          CMP      #242,  ANS2   ;CHECK PS BEFORE FIS TRAP
:          BEQ     .+4           ;BRANCH IF OK
:          HLT+2                    ;PS AT FIS TRAP TIME NOT 242

:          CMP      #014377,ANS3  ;CHECK DATA FROM THE STACK
:          BEQ     .+4           ;BRANCH IF OK
:          HLT+4                    ;DATA ON STACK (014377) CHANGED

:          CMP      #177777,ANS4  ;CHECK DATA FROM STACK
:          BEQ     .+4           ;BRANCH IF OK
:          HLT+4                    ;DATA ON STACK (177777) CHANGED

:          CMP      #154200,ANS5  ;CHECK DATA FROM STACK
:          BEQ     .+4           ;BRANCH IF OK
:          HLT+6                    ;DATA ON STACK (154200) CHANGED

TST     ANS6                    ;CHECK DATA FROM STACK
:          BEQ     .+4           ;BRANCH IF OK
:          HLT+6                    ;DATA ON STACK (000000) CHANGED
```



```

2354 010736 122767 000051 170034 ENDS1:  CMPB  #51,  ICNT  ;CHECK THE TEST NUMBER
2355 010744 001401  BEQ   .+4,          ;BRANCH IF OK
2356 010746 104000  HLT                   ;WRONG TEST! PC MUST HAVE FOULED UP.
2357
2358 010750 104400  SCOPE
2359
2360
2361
2362 *****
2363 :TEST 52:          FDIV (KE11F FLOATING DIVIDE INSTRUCTION)
2364 :          125252,125252 / 065252,125252 = 100200,000000
2365 :          PS = 210,          STACK POINTER = R2
2366 :*****
2367 010752 004567 005332  TST52:  JSR    R5,    PUSHR  ;PUSH 4 WORDS ONTO R2 STACK, SET PRIORITY
2368 010756 065252 125252  .WORD  065252,125252 ;SECOND OPERAND ON TOP
2369 010762 125252 125252  .WORD  125252,125252 ;FIRST OPERAND ON BOTTOM
2370 010766 000217  .WORD  217           ;PROCESSOR PRIORITY LEVEL
2371 010770 016574 000340  .WORD  TRAPER, 340   ;FIS TRAP VECTOR
2372 010774 012702 000630  MOV    #STACK0,R2   ;SET UP STACK POINTER
2373
2374 011000 000240  NOP
2375 011002 075032  FDIV+  R2           ;FLOATING DIVIDE ON THE R2 STACK
2376
2377 011004 004767 005332  JSR    PC,    POPR   ;POP THE ANSWER
2378 011010 010267 167566  MOV    R2,    $SP   ;SAVE "STACK POINTER"
2379 011014 022767 000210 167556  CMP    #210,  $PSW  ;CHECK PS (EXCEPT T BIT)
2380 011022 001401  BEQ    .+4,          ;BRANCH IF OK
2381 011024 104000  HLT                   ;PS NOT EQUAL TO 210
2382
2383 011026 022767 000634 167546  CMP    #STACK4,$SP ;CHECK THE STACK POINTER (R2)
2384 011034 001401  BEQ    .+4,          ;BRANCH IF OK
2385 011036 104000  HLT                   ;STACK POINTER (R2) NOT EQUAL TO #STACK4
2386
2387 011040 022767 100200 167536  CMP    #100200,ANS1 ;CHECK FIRST HALF OF ANSWER
2388 011046 001401  BEQ    .+4,          ;BRANCH IF OK
2389 011050 104002  HLT+2          ;ANS1 NOT EQUAL TO 100200
2390
2391 011052 005767 167530  TST    ANS2
2392 011056 001401  BEQ    .+4,          ;CHECK SECOND HALF OF ANSWER
2393 011060 104002  HLT+2          ;BRANCH IF OK
2394                      ;ANS2 NOT EQUAL TO 000000
2395 011062 122767 000052 167710  ENDS2:  CMPB  #52,  ICNT  ;CHECK THE TEST NUMBER
2396 011070 001401  BEQ    .+4,          ;BRANCH IF OK
2397 011072 104000  HLT                   ;WRONG TEST! PC MUST HAVE FOULED UP.
2398
2399 011074 104400  SCOPE
2400
    
```

2401  
2402  
2403  
2404  
2405  
2406  
2407  
2408  
2409  
2410  
2411  
2412  
2413  
2414  
2415  
2416  
2417  
2418  
2419  
2420  
2421  
2422  
2423  
2424  
2425  
2426  
2427  
2428  
2429  
2430  
2431  
2432  
2433  
2434  
2435  
2436  
2437  
2438  
2439  
2440  
2441  
2442  
2443  
2444  
2445  
2446  
2447  
2448  
2449  
2450  
2451  
2452  
2453  
2454  
2455  
2456

```
*****
:TEST 53:      FDIV (KE11F FLOATING DIVIDE INSTRUCTION)
:              025252,125251 / 065252,125252 ==> UNDERFLOW
:              PS(ON STACK) = 012,      STACK POINTER = R1
*****
```

011076	004567	005206		TST53:	JSR	R5,	PUSHR	;PUSH 4 WORDS ONTO R1 STACK, SET PRIORITY
011102	065252	125252			.WORD	065252,125252	;SECOND OPERAND ON TOP	
011106	025252	125251			.WORD	025252,125251	;FIRST OPERAND ON BOTTOM	
011112	000015				.WORD	015	;PROCESSOR PRIORITY LEVEL	
011114	011144	000300			.WORD	ISR53, 300	;FIS TRAP VECTOR	
011120	012701	000630			MOV	#STACK0,R1	;SET UP R1 AS STACK POINTER	
011124	000240				NOP			
011126	075031				FDIV+	R1	;FLOATING DIVIDE ON THE R1 STACK	
011130	004767	005206		RTA53:	JSR	%7,	POPR	;POP THE "ANSWER"
011134	010167	167442			MOV	R1,	SSP	;SAVE STACK POINTER (R1)
011140	104002				HLT+2			;FIS TRAP DIDN'T OCCURE!
011142	000454				BR	END53		
011144	004767	005224		ISR53:	JSR	%7,	POPER	;POP ALL DATA OFF THE STACKS
011150	010167	167426			MOV	R1,	SSP	;SAVE STACK POINTER (R1)
011154	022767	000300	167416		CMP	#300,	SPSW	;CHECK PS AFTER FIS TRAP
011162	001401				BEQ	+.4		;BRANCH IF OK
011164	104000				HLT			;PS AFTER FIS TRAP NOT EQUAL TO 300
011166	022767	000630	167406		CMP	#STACK0,\$SP		;CHECK THE STACK POINTER (R1)
011174	001401				BEQ	+.4		;BRANCH IF OK
011176	104000				HLT			;STACK POINTER (R1) NOT EQUAL TO #STACK0
011200	022767	011130	167376		CMP	#RTA53, ANS1		;CHECK FIS TRAP RETURN ADDRESS
011206	001401				BEQ	+.4		;BRANCH IF OK
011210	104001				HLT+1			;FIS TRAP AT WRONG ADDRESS
011212	022767	000012	167366		CMP	#012, ANS2		;CHECK PS BEFORE FIS TRAP
011220	001401				BEQ	+.4		;BRANCH IF OK
011222	104002				HLT+2			;PS AT FIS TRAP TIME NOT 012
011224	022767	065252	167356		CMP	#065252,ANS3		;CHECK DATA FROM THE STACK
011232	001401				BEQ	+.4		;BRANCH IF OK
011234	104004				HLT+4			;DATA ON STACK (065252) CHANGED
011236	022767	125252	167346		CMP	#125252,ANS4		;CHECK DATA FROM STACK
011244	001401				BEQ	+.4		;BRANCH IF OK
011246	104004				HLT+4			;DATA ON STACK (125252) CHANGED
011250	022767	025252	167336		CMP	#025252,ANS5		;CHECK DATA FROM STACK
011256	001401				BEQ	+.4		;BRANCH IF OK
011260	104006				HLT+6			;DATA ON STACK (025252) CHANGED
011262	022767	125251	167326		CMP	#125251,ANS6		;CHECK DATA FROM STACK
011270	001401				BEQ	+.4		;BRANCH IF OK
011272	104006				HLT+6			;DATA ON STACK (125251) CHANGED



```

2557 011274 122767 000053 167476 END53:  CMPB  #53,  ICNT  ;CHECK THE TEST NUMBER
2558 011302 001401          BEQ   .+4    ;BRANCH IF OK
2559 011304 104000          HLT                   ;WRONG TEST!  PC MUST HAVE FOULED UP.
2560
2561 011306 104400          SCOPE

```

```

*****
:TEST 54:      FDIV (KE11F FLOATING DIVIDE INSTRUCTION)
:      000000,000000 / 140670,123456 = 000000,000000
:      PS = 104,      STACK POINTER = R3
*****

```

```

2570 011310 004567 004774  TST54:  JSR   R5,    PUSHR  ;PUSH 4 WORDS ONTO R3 STACK, SET PRIORITY
2571 011314 140670 123456      .WORD  140670,123456 ;SECOND OPERAND ON TOP
2572 011320 000000 000000      .WORD  000000,000000 ;FIRST OPERAND ON BOTTOM
2573 011324 000105          .WORD  105           ;PROCESSOR PRIORITY LEVEL
2574 011326 016574 000340      .WORD  TRAPER, 340   ;FIS TRAP VECTOR
2575 011332 012703 000630      MOV   #STACK0,R3    ;SET UP STACK POINTER
2576
2577 011336 000240          NOP
2578 011340 075033          FDIV+  R3           ;FLOATING DIVIDE ON THE R3 STACK
2579
2580 011342 004767 004774  JSR   PC,    POPR   ;POP THE ANSWER
2581 011346 010367 167230      MOV   R3,    $SP   ;SAVE "STACK POINTER"
2582 011352 022767 000104 167220  CMP   #104,  $PSW  ;CHECK PS (EXCEPT T BIT)
2583 011360 001401          BEQ   .+4    ;BRANCH IF OK
2584 011362 104000          HLT                   ;PS NOT EQUAL TO 104
2585
2586 011364 022767 000634 167210  CMP   #STACK4,$SP  ;CHECK THE STACK POINTER (R3)
2587 011372 001401          BEQ   .+4    ;BRANCH IF OK
2588 011374 104000          HLT                   ;STACK POINTER (R3) NOT EQUAL TO #STACK4
2589
2590 011376 005767 167202  TST   ANS1        ;CHECK FIRST HALF OF ANSWER
2591 011402 001401          BEQ   .+4    ;BRANCH IF OK
2592 011404 104002          HLT+2       ;ANS1 NOT EQUAL TO 000000
2593
2594 011406 005767 167174  TST   ANS2        ;CHECK SECOND HALF OF ANSWER
2595 011412 001401          BEQ   .+4    ;BRANCH IF OK
2596 011414 104002          HLT+2       ;ANS2 NOT EQUAL TO 000000
2597
2598 011416 122767 000054 167354  END54:  CMPB  #54,  ICNT  ;CHECK THE TEST NUMBER
2599 011424 001401          BEQ   .+4    ;BRANCH IF OK
2500 011426 104000          HLT                   ;WRONG TEST!  PC MUST HAVE FOULED UP.
2501
2502 011430 104400          SCOPE
2503

```

2504  
2505  
2506  
2507  
2508  
2509  
2510  
2511  
2512  
2513  
2514  
2515  
2516  
2517  
2518  
2519  
2520  
2521  
2522  
2523  
2524  
2525  
2526  
2527  
2528  
2529  
2530  
2531  
2532  
2533  
2534  
2535  
2536  
2537  
2538  
2539  
2540  
2541  
2542  
2543  
2544  
2545  
2546  
2547  
2548  
2549  
2550  
2551  
2552  
2553  
2554  
2555  
2556  
2557  
2558  
2559

\*\*\*\*\*  
TEST 55: FDIV (KE11F FLOATING DIVIDE INSTRUCTION)  
100052,052525 / 000006,123456 ==> DIVIDE BY ZERO  
PS(ON STACK) = 353, STACK POINTER = SP  
\*\*\*\*\*

```

TST55: JSR    R5,    PUSHS    ;PUSH 4 WORDS ONTO STACK, SET PRIORITY
        .WORD  000006,123456 ;SECOND OPERAND ON TOP
        .WORD  100052,052525 ;FIRST OPERAND ON BOTTOM
        .WORD  357          ;PROCESSOR PRIORITY LEVEL
        .WORD  ISR55, 311   ;FIS TRAP VECTOR

        NOP
        FDIV+ SP          ;FLOATING DIVIDE ON THE STACK

RTA55: JSR    %7,    POPS     ;POP THE "ANSWER"
        HLT+2          ;FIS TRAP DIDN'T OCCURE!
        MOV    #500, SP    ;RESTORE THE STACK POINTER
        BR    END55

ISR55: JSR    %7,    POPES   ;POP ALL DATA OFF THE STACK
        CMP    #500, SP    ;CHECK THE STACK POINTER
        BEQ   ISA55      ;BRANCH IF OK
        MOV    #500, SP    ;RESTORE THE STACK POINTER
        HLT          ;STACK POINTER FOULED UP
        BR    END55      ;SKIP REST OF TEST

ISA55: CMP    #311, SPSW   ;CHECK PS AFTER FIS TRAP
        BEQ   .+4        ;BRANCH IF OK
        HLT          ;PS AFTER FIS TRAP NOT EQUAL TO 311

        CMP    #RTA55, ANS1 ;CHECK FIS TRAP RETURN ADDRESS
        BEQ   .+4        ;BRANCH IF OK
        HLT+1          ;FIS TRAP AT WRONG ADDRESS

        CMP    #353, ANS2  ;CHECK PS BEFORE FIS TRAP
        BEQ   .+4        ;BRANCH IF OK
        HLT+2          ;PS AT FIS TRAP TIME NOT 353

        CMP    #000006, ANS3 ;CHECK DATA FROM THE STACK
        BEQ   .+4        ;BRANCH IF OK
        HLT+4          ;DATA ON STACK (000006) CHANGED

        CMP    #123456, ANS4 ;CHECK DATA FROM STACK
        BEQ   .+4        ;BRANCH IF OK
        HLT+4          ;DATA ON STACK (123456) CHANGED

        CMP    #100052, ANS5 ;CHECK DATA FROM STACK
        BEQ   .+4        ;BRANCH IF OK
        HLT+6          ;DATA ON STACK (100052) CHANGED

        CMP    #052525, ANS6 ;CHECK DATA FROM STACK
        BEQ   .+4        ;BRANCH IF OK
        HLT+6          ;DATA ON STACK (052525) CHANGED

```



```

2560 011624 122767 000055 167146 END55:  CMPB  #55,  ICNT  ;CHECK THE TEST NUMBER
2561 011632 001401          BEQ    .+4      ;BRANCH IF OK
2562 011634 104000          HLT                    ;WRONG TEST! PC MUST HAVE FOULED UP.
2563
2564 011636 104400          SCOPE
2565
2566
2567
2568
2569
2570
2571
2572
2573 011640 004567 004622          TST56:  JSR    RS,      PUSH7  ;PUSH 4 WORDS ONTO STACK, SET PRIORITY
2574 011644 011670          .WORD  STK56      ;TOP OF STACK
2575 011646 104000 104000    .WORD  104000,104000 ;SECOND OPERAND ON TOP
2576 011652 004000 105004    .WORD  004000,105004 ;FIRST OPERAND ON BOTTOM
2577 011656 000144          .WORD  144          ;PROCESSOR PRIORITY LEVEL
2578 011660 016574 000340    .WORD  TRAPER,340   ;FIS TRAP VECTOR
2579
2580 011664 000240          NOP
2581 011666 075007          STK56:  FADD+  PC      ;FLOATING ADD ON FOLLOWING 4 WORDS
2582 011670 104000    104000    .WORD  104000      ;SHOULD CONTAIN 104000
2583 011672 104000    104000    .WORD  104000      ;SHOULD CONTAIN 104000
2584 011674 004000    004000    .WORD  004000      ;BEFORE FADD, 004000; AFTER, 000401
2585 011676 105004    105004    .WORD  105004      ;BEFORE FADD, 105004; AFTER, 000000
2586
2587 011700 004767 004620          JSR    PC,      POP7   ;POP THE ANSWER
2588 011704 022767 000140 166666    CMP    #140,    SPSW   ;CHECK PS (EXCEPT T BIT)
2589 011712 001401          BEQ    .+4      ;BRANCH IF OK
2590 011714 104000          HLT                    ;PS NOT EQUAL TO 140
2591
2592 011716 022767 104000 166660    CMP    #104000,ANS1   ;CHECK FIRST HALF OF INPUT DATA (STK56)
2593 011724 001401          BEQ    .+4      ;BRANCH IF OK
2594 011726 104002          HLT+2         ;ANS1 NOT EQUAL TO 104000
2595
2596 011730 022767 104000 166650    CMP    #104000,ANS2   ;CHECK SECOND HALF OF INPUT DATA (STK56+2)
2597 011736 001401          BEQ    .+4      ;BRANCH IF OK
2598 011740 104002          HLT+2         ;ANS2 NOT EQUAL TO 104000
2599
2600 011742 022767 000401 166640    CMP    #000401,ANS3   ;CHECK FIRST HALF OF ANSWER
2601 011750 001401          BEQ    .+4      ;BRANCH IF OK
2602 011752 104004          HLT+4         ;ANS3 NOT EQUAL TO 000401
2603
2604 011754 005767 166632          TST    ANS4        ;CHECK SECOND HALF OF ANSWER
2605 011760 001401          BEQ    .+4      ;BRANCH IF OK
2606 011762 104004          HLT+4         ;ANS4 NOT EQUAL TO 000000
2607
2608 011764 122767 000056 167006 END56:  CMPB  #56,  ICNT  ;CHECK THE TEST NUMBER
2609 011772 001401          BEQ    .+4      ;BRANCH IF OK
2610 011774 104000          HLT                    ;WRONG TEST! PC MUST HAVE FOULED UP.
2611
2612 011776 104400          SCOPE
2613
2614
2615

```

# E05

MAINDEC-11-DBKEA-8  
DBKEAB.P11

KE11F (PDP-11 FIS) INSTRUCTION TESTS.  
TEST OF FIS USING REGISTER 7 (PC)

MACY11 27(732) 20-SEP-76 13:41 PAGE 58

```

2616                                     :TEST 57:      FSUB (KE11F FLOATING SUBTRACT INSTRUCTION)
2617                                     :             104000,105004 - 104000,104000 = 100401,000000
2618                                     :             PS = 250,      STACK POINTER = PC
2619                                     :*****
2620
2621 012000 004567 004462   TST57: JSR    R5,      PUSH7   ;PUSH 4 WORDS ONTO STACK, SET PRIORITY
2622 012004 012030                .WORD  STK57      ;TOP OF STACK
2623 012006 104000 104000    .WORD  104000,104000 ;SECOND OPERAND ON TOP
2624 012012 104000 105004    .WORD  104000,105004 ;FIRST OPERAND ON BOTTOM
2625 012016 000252                .WORD  252          ;PROCESSOR PRIORITY LEVEL
2626 012020 016574 000340    .WORD  TRAPER,340   ;FIS TRAP VECTOR
2627
2628 012024 000240                NOP
2629 012026 075017                FSUB+  PC          ;FLOATING SUBTRACT ON FOLLOWING 4 WORDS
2630 012030 104000   STK57: 104000    ;SHOULD CONTAIN 104000
2631 012032 104000    104000    ;SHOULD CONTAIN 104000
2632 012034 104000    104000    ;BEFORE FSUB, 104000; AFTER, 100401
2633 012036 105004    105004    ;BEFORE FSUB, 105004; AFTER, 000000
2634
2635 012040 004767 004460   JSR    PC,      POP7   ;POP THE ANSWER
2636 012044 022767 000250 166526  CMP    #250,    SPSW   ;CHECK PS (EXCEPT T BIT)
2637 012052 001401                BEQ    .+4         ;BRANCH IF OK
2638 012054 104000                HLT
2639                                     ;PS NOT EQUAL TO 250
2640 012056 022767 104000 166520  CMP    #104000,ANS1  ;CHECK FIRST HALF OF INPUT DATA (STK57)
2641 012064 001401                BEQ    .+4         ;BRANCH IF OK
2642 012066 104002                HLT+2            ;ANS1 NOT EQUAL TO 104000
2643
2644 012070 022767 104000 166510  CMP    #104000,ANS2  ;CHECK SECOND HALF OF INPUT DATA (STK57+2)
2645 012076 001401                BEQ    .+4         ;BRANCH IF OK
2646 012100 104002                HLT+2            ;ANS2 NOT EQUAL TO 104000
2647
2648 012102 022767 100401 166500  CMP    #100401,ANS3  ;CHECK FIRST HALF OF ANSWER
2649 012110 001401                BEQ    .+4         ;BRANCH IF OK
2650 012112 104004                HLT+4            ;ANS3 NOT EQUAL TO 100401
2651
2652 012114 005767 166472   TST    ANS4        ;CHECK SECOND HALF OF ANSWER
2653 012120 001401                BEQ    .+4         ;BRANCH IF OK
2654 012122 104004                HLT+4            ;ANS4 NOT EQUAL TO 000000
2655
2656 012124 122767 000057 166646  END57: CMPB   #57,      ICNT   ;CHECK THE TEST NUMBER
2657 012132 001401                BEQ    .+4         ;BRANCH IF OK
2658 012134 104000                HLT
2659                                     ;WRONG TEST! PC MUST HAVE FOULED UP.
2660 012136 104400                SCOPE
2661

```



```

2662
2663
2664
2665
2666
2667
2668
2669 012140 004567 004322
2670 012144 012170
2671 012146 104000 104000
2672 012152 134600 073601
2673 012156 000246
2674 012160 016574 000340
2675
2676 012164 000240
2677 012166 075027
2678 012170 104000
2679 012172 104000
2680 012174 134600
2681 012176 073601
2682
2683 012200 004767 004320
2684 012204 022767 000240 166366
2685 012212 001401
2686 012214 104000
2687
2688 012216 022767 104000 166360
2689 012224 001401
2690 012226 104002
2691
2692 012230 022767 104000 166350
2693 012236 001401
2694 012240 104002
2695
2696 012242 022767 000401 166340
2697 012250 001401
2698 012252 104004
2699
2700 012254 005767 166332
2701 012260 001401
2702 012262 104004
2703
2704 012264 122767 000060 166506 END60:
2705 012272 001401
2706 012274 104000
2707
2708 012276 104400
2709

```

```

:*****
:TEST 60: FMUL (KE11F FLOATING MULTIPLY INSTRUCTION)
:      134600,073601 * 104000,104000 = 000401,000000
:      PS = 240, STACK POINTER = PC
:*****
TST60: JSR R5, PUSH7 ;PUSH 4 WORDS ONTO STACK, SET PRIORITY
        .WORD STK60 ;TOP OF STACK
        .WORD 104000,104000 ;SECOND OPERAND ON TOP
        .WORD 134600,073601 ;FIRST OPERAND ON BOTTOM
        .WORD 246 ;PROCESSOR PRIORITY LEVEL
        .WORD TRAPER,340 ;FIS TRAP VECTOR

STK60: NOP
        FMUL+ PC ;FLOATING MULTIPLY ON FOLLOWING 4 WORDS
        104000 ;SHOULD CONTAIN 104000
        104000 ;SHOULD CONTAIN 104000
        134600 ;BEFORE FMUL, 134600; AFTER, 000401
        073601 ;BEFORE FMUL, 073601; AFTER, 000000

        JSR PC, POP7 ;POP THE ANSWER
        CMP #240, $PSW ;CHECK PS (EXCEPT T BIT)
        BEQ .+4 ;BRANCH IF OK
        HLT ;PS NOT EQUAL TO 240

        CMP #104000,ANS1 ;CHECK FIRST HALF OF INPUT DATA (STK60)
        BEQ .+4 ;BRANCH IF OK
        HLT+2 ;ANS1 NOT EQUAL TO 104000

        CMP #104000,ANS2 ;CHECK SECOND HALF OF INPUT DATA (STK60+2)
        BEQ .+4 ;BRANCH IF OK
        HLT+2 ;ANS2 NOT EQUAL TO 104000

        CMP #000401,ANS3 ;CHECK FIRST HALF OF ANSWER
        BEQ .+4 ;BRANCH IF OK
        HLT+4 ;ANS3 NOT EQUAL TO 000401

        TST ANS4 ;CHECK SECOND HALF OF ANSWER
        BEQ .+4 ;BRANCH IF OK
        HLT+4 ;ANS4 NOT EQUAL TO 000000

        CMPB #60, ICNT ;CHECK THE TEST NUMBER
        BEQ .+4 ;BRANCH IF OK
        HLT ;WRONG TEST! PC MUST HAVE FOULED UP.

SCOPE

```

2710  
 2711  
 2712  
 2713  
 2714  
 2715  
 2716  
 2717 012300 004567 004162  
 2718 012304 012330  
 2719 012306 104000 104000  
 2720 012312 102500 146000  
 2721 012316 000357  
 2722 012320 016574 000340  
 2723  
 2724 012324 000240  
 2725 012326 075037  
 2726 012330 104000  
 2727 012332 104000  
 2728 012334 102500  
 2729 012336 146000  
 2730  
 2731 012340 004767 004160  
 2732 012344 022767 000340 166226  
 2733 012352 001401  
 2734 012354 104000  
 2735  
 2736 012356 022767 104000 166220  
 2737 012364 001401  
 2738 012366 104002  
 2739  
 2740 012370 022767 104000 166210  
 2741 012376 001401  
 2742 012400 104002  
 2743  
 2744 012402 022767 036700 166200  
 2745 012410 001401  
 2746 012412 104004  
 2747  
 2748 012414 005767 166172  
 2749 012420 001401  
 2750 012422 104004  
 2751  
 2752 012424 122767 000061 166346  
 2753 012432 001401  
 2754 012434 104000  
 2755  
 2756 012436 104400  
 2757

```

*****
:TEST 61:      FDIV (KE11F FLOATING DIVIDE INSTRUCTION)
:      102500,146000 / 104000,104000 = 036700,000000
:      PS = 340,      STACK POINTER = PC
*****

```

```

TST61:  JSR      R5,      PUSH7      ;PUSH 4 WORDS ONTO STACK, SET PRIORITY
:      .WORD    STK61      ;TOP OF STACK
:      .WORD    104000,104000 ;SECOND OPERAND ON TOP
:      .WORD    102500,146000 ;FIRST OPERAND ON BOTTOM
:      .WORD    357        ;PROCESSOR PRIORITY LEVEL
:      .WORD    TRAPER,340 ;FIS TRAP VECTOR

```

```

STK61:  NOP
:      FDIV+   PC          ;FLOATING DIVIDE ON FOLLOWING 4 WORDS
:      104000  ;SHOULD CONTAIN 104000
:      104000  ;SHOULD CONTAIN 104000
:      102500  ;BEFORE FDIV, 102500; AFTER, 036700
:      146000  ;BEFORE FDIV, 146000; AFTER, 000000

```

```

JSR      PC,      POP7      ;POP THE ANSWER
CMP      #340,    $PSW      ;CHECK PS (EXCEPT T BIT)
BEQ      .+4      ;BRANCH IF OK
HLT      ;PS NOT EQUAL TO 340

```

```

CMP      #104000,ANS1      ;CHECK FIRST HALF OF INPUT DATA (STK61)
BEQ      .+4      ;BRANCH IF OK
HLT+2    ;ANS1 NOT EQUAL TO 104000

```

```

CMP      #104000,ANS2      ;CHECK SECOND HALF OF INPUT DATA (STK61+2)
BEQ      .+4      ;BRANCH IF OK
HLT+2    ;ANS2 NOT EQUAL TO 104000

```

```

CMP      #036700,ANS3      ;CHECK FIRST HALF OF ANSWER
BEQ      .+4      ;BRANCH IF OK
HLT+4    ;ANS3 NOT EQUAL TO 036700

```

```

TST      ANS4      ;CHECK SECCND HALF OF ANSWER
BEQ      .+4      ;BRANCH IF OK
HLT+4    ;ANS4 NOT EQUAL TO 000000

```

```

END61:  CMPB     #61,      ICNT      ;CHECK THE TEST NUMBER
BEQ      .+4      ;BRANCH IF OK
HLT      ;WRONG TEST! PC MUST HAVE FOULED UP.

```

SCOPE



```

2758
2759
2760
2761
2762
2763
2764
2765
2766
2767 012440 012704 000640
2768 012444 012744 107070
2769 012450 012744 134343
2770 012454 012744 065432
2771 012460 012744 032107
2772 012464 012744 123456
2773 012470 012744 045670
2774 012474 012744 125252
2775 012500 012744 135252
2776 012504 012744 016161
2777 012510 012744 040616
2778 012514 012737 000144 177776
2779
2780 012522 000240
2781 012524 075014
2782 012526 075034
2783 012530 075024
2784 012532 075004
2785
2786 012534 013767 177776 166036
2787 012542 042767 000020 166030
2788 012550 012467 166030
2789 012554 012467 166026
2790 012560 010467 166016
2791 012564 022767 000150 166006
2792 012572 001401
2793 012574 104000
2794
2795 012576 022767 000640 165776
2796 012604 001401
2797 012606 104000
2798
2799 012610 022767 137201 165766
2800 012616 001401
2801 012620 104002
2802
2803 012622 022767 115230 165756
2804 012630 001401
2805 012632 104002
2806
2807 012634 122767 000062 166136 END62:
2808 012642 001401
2809 012644 104000
2810
2811 012646 104400

```

```

*****
:TEST 62: TEST ALL INSTRUCTION TOGETHER
:          032107,065432 * 045670,123456
:          134343,107070 + ----- = 137201,115230
:          (135252,125252 - 040616,016161)
:          PS=150, STACK POINTER=R4
*****

```

```

TST62:  MOV    #STACK8,R4      ;SET STACK POINTER
        MOV    #107070,-(R4)  ;LOAD DATA ONTO STACK
        MOV    #134343,-(R4)
        MOV    #065432,-(R4)
        MOV    #032107,-(R4)
        MOV    #123456,-(R4)
        MOV    #045670,-(R4)
        MOV    #125252,-(R4)
        MOV    #135252,-(R4)
        MOV    #016161,-(R4)
        MOV    #040616,-(R4)
        MOV    #144, 2#PS     ;SET PROCESSOR STATUS

        NOP
        FSUB+  R4           ;135252,125252-040616,016161=140616,017434
        FDIV+  R4           ;045670,123456/140616,017434=145246,047065
        FMUL+  R4           ;032107,065432*145246,047065=137201,106137
        FADD+  R4           ;134343,107070+137201,106137=137201,115230

        MOV    2#PS, $PSW    ;SAVE FINAL PS
        BIC    #20, $PSW     ;CLR T-BIT
        MOV    (R4)+, ANS1   ;SAVE FIRST HALF OF ANSWER
        MOV    (R4)+, ANS2   ;SAVE SECOND HALF OF ANSWER
        MOV    R4, $SP       ;SAVE STACK POINTER
        CMP    #150, $PSW    ;CHECK PS (EXCEPT T BIT)
        BEQ   .+4           ;BRANCH IF OK
        HLT                    ;PS NOT EQUAL TO 150

        CMP    #STACK8,$SP   ;CHECK THE STACK POINTER (R4)
        BEQ   .+4           ;BRANCH IF OK
        HLT                    ;STACK POINTER (R4) NOT EQUAL TO 674

        CMP    #137201,ANS1  ;CHECK FIRST HALF OF ANSWER
        BEQ   .+4           ;BRANCH IF OK
        HLT+2                ;ANS1 NOT EQUAL TO 137201

        CMP    #115230,ANS2  ;CHECK SECOND HALF OF ANSWER
        BEQ   .+4           ;BRANCH IF OK
        HLT+2                ;ANS2 NOT EQUAL TO 115230

        CMPB   #62, ICNT     ;CHECK THE TEST NUMBER
        BEQ   .+4           ;BRANCH IF OK
        HLT                    ;WRONG TEST! PC MUST HAVE FOULED UP.

        SCOPE

```

2812  
2813  
2814  
2815  
2816  
2817  
2818  
2819  
2820  
2821  
2822  
2823  
2824  
2825  
2826  
2827  
2828  
2829  
2830  
2831  
2832  
2833  
2834  
2835  
2836  
2837  
2838  
2839  
2840  
2841  
2842  
2843  
2844  
2845  
2846  
2847  
2848  
2849  
2850  
2851  
2852  
2853  
2854  
2855  
2856  
2857  
2858  
2859  
2860  
2861  
2862  
2863  
2864  
2865  
2866  
2867

```
*****  
:TEST 63: TEST THAT YELLOW ZONE STACK OVERFLOW WORKS WITH FIS  
:STACK POINTER = SP = 356  
:*****
```

```
TST63: MOV #356, R1 ;SET UP STACK  
MOV #035152, (R1)+ ;PUT DATA ON THE STACK  
MOV #125252, (R1)+  
MOV #043125, (R1)+  
MOV #052525, (R1)+  
MOV #35, J#4 ;SETUP STACK ERROR VECTOR  
MOV #300, J#6  
MOV J#PS, R0 ;SAVE THE T-BIT  
MOV #340, -(SP) ;CLR T-BIT  
MOV #15, -(SP)  
RTI  
15: MOV #356, SP ;OVERFLOW THE STACK  
NOP  
FADD+ SP ;DO 043125 FLOATING POINT ADD  
25: MOV PS, $PSW ;SAVE PS FOR TYPING  
MOV SP, $SP ;SAVE STACK POINTER  
MOV #500, SP ;RESTORE THE STACK  
HLT ;STACK OVERFLOW DIDN'T TRAP  
BR 45  
35: MOV PS, $PSW ;SAVE THE PS  
MOV SP, $SP ;SAVE THE STACK POINTER  
MOV SP, R1  
MOV #500, SP ;RESTORE THE STACK  
MOV #ANS1, R2 ;TOP OF ANSWER TABLE  
MOV (R1)+, (R2)+ ;SAVE THE STACK DATA  
MOV (R1)+, (R2)+  
MOV (R1)+, (R2)+  
MOV (R1)+, (R2)+  
CMP #300, $PSW ;CHECK THE PS AFTER THE TRAP  
BEQ .+4 ;BRANCH IF OK  
HLT ;PS NOT EQUAL TO 300  
CMP #356, $SP ;CHECK FOR SP AT RIGHT SPOT  
BEQ .+4 ;BRANCH IF OK  
HLT ;STACK POINTER FOULED UP  
CMP #25, ANS1 ;CHECK TOP OF STACK FOR RTI ADR.  
BEQ .+4 ;BRANCH IF OK  
HLT+1 ;RTI ADDRESS NOT EQUAL TO #25  
CMP #340, ANS2 ;CHECK STACK DATA FOR RTI PS  
BEQ .+4 ;BRANCH IF OK  
HLT+2 ;RTI PS NOT EQUAL TO 340  
CMP #043125, ANS3 ;CHECK FIRST HALF OF ANSWER  
BEQ .+4 ;BRANCH IF OK
```

000004  
000006

165034 165634

165012 165612

000300 165556

000356 165546

012736 165536

000340 165526

043125 165516



```

2868 013074 104004          HLT+4          ;ANS3 NOT EQUAL TO 043125
2869
2870 013076 022767 052526 165506      CMP          #052526,ANS4      ;CHECK SECOND HALF OF ANSWER
2871 013104 001401          BEQ          .+4              ;BRANCH IF OK
2872 013106 104004          HLT+4          ;ANS4 NOT EQUAL TO 052526
2873
2874 013110 010046          4$:  MOV      R0,          -(SP)      ;RESTORE THE T-BIT
2875 013112 012746 013120      MOV      #5$,          -(SP)
2876 013116 000002          RTI
2877 013120 122767 000063 165652 5$:  CMPB     #63,          ICNT      ;CHECK THE TEST NUMBER
2878 013126 001401          BEQ          .+4              ;BRANCH IF OK
2879 013130 104000          HLT          ;WRONG TEST! PC MUST HAVE FOULED UP.
2880
2881 013132 104400          SCOPE
2882
2883
2884
2885 ;*****
2886 ;TEST 64:      TEST THAT RED ZONE STACK OVERFLOW WORKS WITH FIS
2887 ;      STACK POINTER = SP = 0
2888 ;*****
2889 013134 012701 000332      TST64:  MOV      #332,      R1          ;SET UP STACK
2890 013140 012721 025177      MOV      #025177,(R1)+      ;PUT DATA ON THE STACK
2891 013144 012721 177777      MOV      #177777,(R1)+
2892 013150 012721 125200      MOV      #125200,(R1)+
2893 013154 012721 000000      MOV      #000000,(R1)+
2894 013160 012737 013244 000004      MOV      #3$,          J#4          ;SETUP STACK ERROR VECTOR
2895 013166 012737 000300 000006      MOV      #300,         J#6
2896 013174 013700 177776      MOV      J#PS,         R0          ;SAVE THE T-BIT
2897 013200 012746 000340      MOV      #340,         -(SP)      ;CLR T-BIT
2898 013204 012746 013212      MOV      #1$,          -(SP)
2899 013210 000002          RTI
2900 013212 012706 000332      1$:  MOV      #332,         SP          ;OVERFLOW THE STACK
2901
2902 013216 000240          NOP
2903 013220 075006      FADD+   SP          ;DO 125200 FLOATING POINT ADD
2904
2905 013222 016767 164550 165350 2$:  MOV      PS,          $PSW      ;SAVE PS FOR TYPING
2906 013230 010667 165346      MOV      SP,          $$P      ;SAVE STACK POINTER
2907 013234 012706 000500      MOV      #500,        SP      ;RESTORE THE STACK
2908 013240 104000          HLT          ;STACK OVERFLOW DIDN'T TRAP
2909
2910 013242 000473          BR          4$
2911
2912 013244 016767 164526 165326 3$:  MOV      PS,          $PSW      ;SAVE THE PS
2913 013252 010667 165324      MOV      SP,          $$P      ;SAVE THE STACK POINTER
2914 013256 012706 000500      MOV      #500,        SP      ;RESTORE THE STACK
2915 013262 013767 000000 165314      MOV      J#0,          ANS1     ;SAVE RETURN ADR
2916 013270 013767 000002 165310      MOV      J#2,          ANS2     ;SAVE RETURN STATUS
2917 013276 012701 000332      MOV      #332,        R1        ;POINT TO TOP OF ORIGINAL STACK
2918 013302 012702 000610      MOV      #ANS3,       R2        ;TOP OF ANSWER TABLE
2919 013306 012122          MOV      (R1)+,       (R2)+     ;SAVE THE STACK DATA
2920 013310 012122          MOV      (R1)+,       (R2)+
2921 013312 012122          MOV      (R1)+,       (R2)+
2922 013314 012122          MOV      (R1)+,       (R2)+
2923 013316 022767 000300 165254      CMP      #300,        $PSW      ;CHECK THE PS AFTER THE TRAP
    
```





2964  
2965  
2966  
2967  
2968  
2969  
2970  
2971  
2972  
2973  
2974  
2975  
2976  
2977  
2978  
2979  
2980  
2981  
2982  
2983  
2984  
2985  
2986  
2987  
2988  
2989  
2990  
2991  
2992  
2993  
2994  
2995  
2996  
2997  
2998  
2999  
3000  
3001  
3002  
3003  
3004  
3005  
3006  
3007  
3008  
3009  
3010  
3011  
3012  
3013  
3014  
3015  
3016  
3017  
3018  
3019

013456 012701 000326  
013462 012721 100125  
013466 012721 052525  
013472 012721 135753  
013476 012721 024642  
013502 012737 013566 000004  
013510 012737 000300 000006  
013516 013700 177776  
013522 012746 000340  
013526 012746 013534  
013532 000002  
013534 012706 000326  
013540 000240  
013542 075006  
013544 016767 164226 165026 2\$:  
013552 010667 165024  
013556 012706 000500  
013562 104000  
013564 000474  
013566 016767 164204 165004 3\$:  
013574 010667 165002  
013600 012706 000500  
013604 013767 000000 164772  
013612 013767 000002 164766  
013620 012701 000326  
013624 012702 000610  
013630 012122  
013632 012122  
013634 012122  
013636 012122  
013640 022767 000300 164732  
013646 001401  
013650 104000  
013652 005767 164724  
013656 001401  
013660 104000  
013662 022767 013544 164714  
013670 001401  
013672 104001  
013674 022767 000340 164704  
013702 001401  
013704 104002

```
*****  
:TEST 65: TEST THAT RED ZONE STACK OVERFLOW WORKS WITH FIS  
: STACK POINTER = SP = 0  
*****  
TST65: MOV #326, R1 ;SET UP STACK  
MOV #100125, (R1)+ ;PUT DATA ON THE STACK  
MOV #052525, (R1)+  
MOV #135753, (R1)+  
MOV #024642, (R1)+  
MOV #3$, @#4 ;SETUP STACK ERROR VECTOR  
MOV #300, @#6  
MOV @#PS, R0 ;SAVE THE T-BIT  
MOV #340, -(SP) ;CLR T-BIT  
MOV #1$, -(SP)  
RTI  
1$: MOV #326, SP ;OVERFLOW THE STACK  
NOP  
FADD+ SP ;DO 135753 FLOATING POINT ADD  
2$: MOV PS, $PSW ;SAVE PS FOR TYPING  
MOV SP, $SP ;SAVE STACK POINTER  
MOV #500, SP ;RESTORE THE STACK  
HLT ;STACK OVERFLOW DIDN'T TRAP  
BR 4$  
3$: MOV PS, $PSW ;SAVE THE PS  
MOV SP, $SP ;SAVE THE STACK POINTER  
MOV #500, SP ;RESTORE THE STACK  
MOV @#0, ANS1 ;SAVE RETURN ADR  
MOV @#2, ANS2 ;SAVE RETURN STATUS  
MOV #326, R1 ;POINT TO TOP OF ORIGINAL STACK  
MOV #ANS3, R2 ;TOP OF ANSWER TABLE  
MOV (R1)+, (R2)+ ;SAVE THE STACK DATA  
MOV (R1)+, (R2)+  
MOV (R1)+, (R2)+  
MOV (R1)+, (R2)+  
CMP #300, $PSW ;CHECK THE PS AFTER THE TRAP  
BEQ .+4 ;BRANCH IF OK  
HLT ;PS NOT EQUAL TO 300  
TST $SP ;CHECK FOR SP AT RIGHT SPOT  
BEQ .+4 ;BRANCH IF OK  
HLT ;STACK POINTER FOULED UP  
CMP #2$, ANS1 ;CHECK TOP OF STACK FOR RTI ADR.  
BEQ .+4 ;BRANCH IF OK  
HLT+1 ;RTI ADDRESS NOT EQUAL TO #2$  
CMP #340, ANS2 ;CHECK STACK DATA FOR RTI PS  
BEQ .+4 ;BRANCH IF OK  
HLT+2 ;RTI PS NOT EQUAL TO 340
```







```

3101
3102 014214 122767 000066 164556 END66:  CMPB  #66,  ICNT  ;CHECK THE TEST NUMBER
3103 014222 001401          BEQ    .+4    ;BRANCH IF OK
3104 014224 104000          HLT                    ;WRONG TEST! PC MUST HAVE FOULED UP.
3105
3106 014226 104400          SCOPE
3107
3108
3109
3110
3111
3112
3113
3114 014230 012737 014300 000004 TST67:  MOV    #ISR67,  #4    ;SET UP ADDRESS TRAP VECTOR
3115 014236 012737 000340 000006      MOV    #340,   #6
3116 014244 012737 000202 177776      MOV    #202,   #PS
3117 014252 012705 160000      MOV    #160000,R5 ;SET UP R5 AS STACK POINTER
3118
3119 014256 000240          NOP
3120 014260 075025          FMUL+  R5    ;FLOATING MULTIPLY ON THE R5 STACK
3121
3122 014262 013767 177776 164310 RTA67:  MOV    #PS,   SPSW  ;SAVE THE PSW
3123 014270 010567 164306      MOV    R5,    SSP   ;SAVE STACK POINTER (R5)
3124 014274 104000          HLT                    ;FIS TRAP DIDN'T OCCURE!
3125 014276 000430          BR     END67
3126
3127 014300 004767 002070          ISR67:  JSR    %7,    POPER  ;POP ALL DATA OFF THE STACKS
3128 014304 010567 164272      MOV    R5,    SSP   ;SAVE STACK POINTER (R5)
3129 014310 022767 000340 164262      CMP    #340,   SPSW  ;CHECK PS AFTER ADR. ERR. TRAP
3130 014316 001401          BEQ    .+4    ;BRANCH IF OK
3131 014320 104000          HLT                    ;PS AFTER TRAP NOT EQUAL TO 340
3132
3133 014322 022767 160000 164252      CMP    #160000,SSP  ;CHECK THE STACK POINTER (R5)
3134 014330 001401          BEQ    .+4    ;BRANCH IF OK
3135 014332 104000          HLT                    ;STACK POINTER (R5) NOT EQUAL TO #160000
3136
3137 014334 022767 014262 164242      CMP    #RTA67, ANS1  ;CHECK FIS TRAP RETURN ADDRESS
3138 014342 001401          BEQ    .+4    ;BRANCH IF OK
3139 014344 104001          HLT+1          ;FIS TRAP AT WRONG ADDRESS
3140
3141 014346 022767 000210 164232      CMP    #210,   ANS2  ;CHECK PS BEFORE FIS TRAP
3142 014354 001401          BEQ    .+4    ;BRANCH IF OK
3143 014356 104002          HLT+2          ;PS AT FIS TRAP TIME NOT 210
3144
3145 014360 122767 000067 164412 END67:  CMPB  #67,  ICNT  ;CHECK THE TEST NUMBER
3146 014366 001401          BEQ    .+4    ;BRANCH IF OK
3147 014370 104000          HLT                    ;WRONG TEST! PC MUST HAVE FOULED UP.
3148
3149 014372 104400          SCOPE
3150
3151 014374 012737 000006 000004      MOV    #6,    #4    ;RESTORE TIME-OUT VECTOR
3152 014402 005037 000006

```

```

:*****
:TEST 67: TEST THAT STACK POINTER ADDRESS ERROR CAUSES ABORT
:INSTRUCTION = FMUL, STACK POINTER = R5
:*****

```



```

3153 014406 012767 000003 001514      MOV    #3,      TIMES ;REDUCE NUMBER OF ITERATIONS
3154
3155                                     ;*****
3156                                     ;TEST 70:      TEST THAT FIS ABORTS PROPERLY WHEN INTERRUPTED
3157                                     ;             101010,020202 - 00GJ00,000000 = 101010,020202
3158                                     ;             PS = 144,      STACK POINTER = R1
3159                                     ;*****
3160
3161 014414 012737 014514 000064 TST70: MOV    #ISR70, 3#64 ;SET UP TELEPRINTER INTERRUPT VECTOR
3162 014422 012737 000200 000066      MOV    #200, 3#66
3163 014430 000004 017426      TYPE,  RETURN ;TYPE CARRIAGE RETURN, LINE FEED
3164 014434 012767 014442 001464      MOV    #.+6,  LADS ;RESET LOOP ADDRESS
3165 014442 012777 000100 164204      MOV    #100, 3TPS ;SET TTY INTERRUPT ENABLE
3166 014450 012777 000100 164200      MOV    #100, 3TPB ;TYPE "3"
3167 014456 004567 001626      JSR    R5,     PUSHR ;PUSH 4 WORDS ONTO R1 STACK, SET PRIORITY
3168 014462 000000 000000      .WORD 000000,000000 ;SECOND OPERAND ON TOP
3169 014466 101010 020202      .WORD 101010,020202 ;FIRST OPERAND ON BOTTOM
3170 014472 000143      .WORD 143 ;PROCESSOR PRIORITY LEVEL
3171 014474 016574 000340      .WORD TRAPER, 340 ;FIS TRAP VECTOR
3172 014500 012701 000630      MOV    #STACK0,R1 ;SET UP STACK POINTER
3173
3174 014504 000240
3175 014506 075011      RTA70: FSUB+  R1 ;FLOATING SUBTRACT ON THE STACK
3176 014510 024141      CMP    -(R1), -(R1) ;RESET THE STACK POINTER FOR NEXT PASS
3177 014512 000775      BR     RTA70 ;REPEAT UNTIL INTERRUPTED
3178
3179 014514 022716 014506      ISR70: CMP    #RTA70, (SP) ;CHECK IF INTERRUPT AT FIS INSTR.
3180 014520 001410      BEQ   1$ ;BRANCH IF IT DID
3181 014522 022766 014506 000004      CMP    #RTA70, 4(SP) ;CHECK FOR INTERRUPT WITH T-BIT SET
3182 014530 001407      BEQ   2$ ;BRANCH IF IT DID
3183 014532 012777 000100 164116      MOV    #100, 3TPB ;CONTINUE TO TYPE "3"
3184 014540 000002      RTI
3185
3186 014542 004767 001626      1$:  JSR    PC,     POPER ;SAVE ALL THE STUFF ON THE STACK
3187 014546 000406      BR     3$
3188
3189 014550 013767 177776 164022 2$:  MOV    3#PS,  SPSW ;SAVE THE SPW
3190 014556 022626      CMP    (SP)+, (SP)+ ;RESET THE STACK TO IGNORE THE TRACE TRAP
3191 014560 004767 001616      JSR    PC,     POPER1 ;POP ALL THE STUFF OFF THE STACK
3192 014564 005077 164064      3$:  CLR    3TPS ;CLR INTERRUPT ENABLE
3193 014570 022706 000500      CMP    #500,  SP ;CHECK THE STACK POINTER
3194 014574 001406      BEQ   ISA70 ;BRANCH IF OK
3195 014576 010667 164000      MOV    SP,     SSP ;SAVE FOR TYPING
3196 014602 012706 000500      MOV    #500,  SP ;RESTORE THE STACK POINTER
3197 014606 104000      HLT ;STACK POINTER FOULED UP
3198 014610 000450      BR     END70 ;SKIP REST OF TEST
3199
3200 014612 010167 163764      ISA70: MOV    R1,     SSP ;SAVE STACK POINTER
3201 014616 022767 000204 163754      CMP    #204,  SPSW ;CHECK PS AFTER INTERRUPT
3202 014624 001401      BEQ   .+4 ;BRANCH IF OK
3203 014626 104000      HLT ;PS AFTER INTERRUPT NOT EQUAL TO LVLA
3204
3205 014630 022767 000630 163744      CMP    #STACK0,SSP ;CHECK THE STACK POINTER (R1)
3206 014636 001401      BEQ   .+4 ;BRANCH IF OK
3207 014640 104000      HLT ;STACK POINTER (R1) NOT EQUAL TO #STACK0
3208

```

```

3209 014642 022767 014506 163734      CMP      #RTA70, ANS1      ;CHECK FIS TRAP RETURN ADDRESS
3210 014650 001401                      BEQ      .+4              ;BRANCH IF OK
3211 014652 104001                      HLT+1                    ;FIS TRAP AT WRONG ADDRESS
3212
3213 014654 022767 000144 163724      CMP      #144,   ANS2      ;CHECK PS BEFORE INTERUPT
3214 014662 001401                      BEQ      .+4              ;BRANCH IF OK
3215 014664 104002                      HLT+2                    ;PS AT INTERUPT TIME NOT 144
3216
3217 014666 005767 163716                      TST      ANS3              ;CHECK DATA FROM THE STACK
3218 014672 001401                      BEQ      .+4              ;BRANCH IF OK
3219 014674 104004                      HLT+4                    ;DATA ON STACK (000000) CHANGED
3220
3221 014676 005767 163710                      TST      ANS4              ;CHECK DATA FROM STACK
3222 014702 001401                      BEQ      .+4              ;BRANCH IF OK
3223 014704 104004                      HLT+4                    ;DATA ON STACK (000000) CHANGED
3224
3225 014706 022767 101010 163700      CMP      #101010,ANS5     ;CHECK DATA FROM STACK
3226 014714 001401                      BEQ      .+4              ;BRANCH IF OK
3227 014716 104006                      HLT+6                    ;DATA ON STACK (101010) CHANGED
3228
3229 014720 022767 020202 163670      CMP      #020202,ANS6     ;CHECK DATA FROM STACK
3230 014726 001401                      BEQ      .+4              ;BRANCH IF OK
3231 014730 104006                      HLT+6                    ;DATA ON STACK (020202) CHANGED
3232
3233 014732 122767 000070 164040  END70:  CMPB     #70,   ICNT      ;CHECK THE TEST NUMBER
3234 014740 001401                      BEQ      .+4              ;BRANCH IF OK
3235 014742 104000                      HLT                                  ;WRONG TEST! PC MUST HAVE FOULED UP.
3236
3237 014744 104400                      SCOPE

```

```

3238
3239
3240
3241 *****
3242 TEST 71:      TEST THAT FIS ABORTS PROPERLY WHEN INTERUPTED
3243             123456,123456 / 040200,000000 = 123456,123456
3244             PS = 051,      STACK POINTER = R4
3245 *****

```

```

3246 014746 012737 015046 000064  TST71:  MOV      #ISR71, R#64      ;SET UP TELEPRINTER INTERUPT VECTOR
3247 014754 012737 000200 000066      MOV      #200,   R#66
3248 014762 000004 017426                      TYPE,   RETURN            ;TYPE CARRIAGE RETURN, LINE FEED
3249 014766 012767 014774 001132      MOV      #.+6,   LADS      ;RESET LOOP ADDRESS
3250 014774 012777 000100 163652      MOV      #100,   #TPS      ;SET TTY INTERUPT ENABLE
3251 015002 012777 000100 163646      MOV      #100,   #TPB      ;TYPE "3"
3252 015010 004567 001274                      JSR      R5,   PUSHR       ;PUSH 4 WORDS ONTO R4 STACK, SET PRIORITY
3253 015014 040200 000000                      .WORD   040200,000000     ;SECOND OPERAND ON TOP
3254 015020 123456 123456                      .WORD   123456,123456     ;FIRST OPERAND ON BOTTOM
3255 015024 000040                      .WORD   040              ;PROCESSOR PRIORITY LEVEL
3256 015026 016574 000340                      .WORD   TRAPER, 340       ;FIS TRAP VECTOR
3257 015032 012704 000630      MOV      #STACK0,R4      ;SET UP STACK POINTER
3258
3259 015036 000240
3260 015040 075034      RTA71:  FDIV+   R4          ;FLOATING DIVIDE ON THE STACK
3261 015042 024444      CMP      -(R4), -(R4)     ;RESET THE STACK POINTER FOR NEXT PASS
3262 015044 000775      BR       RTA71           ;REPEAT UNTIL INTERUPTED
3263
3264 015046 022716 015040      ISR71:  CMP      #RTA71, (SP) ;CHECK IF INTERUPT AT FIS INSTR.

```



3265	015052	001410				BEQ	1\$				;BRANCH IF IT DID
3266	015054	022766	015040	000004		CMP	#RTA71,	4(SP)			;CHECK FOR INTERUPT WITH T-BIT SET
3267	015062	001407				BEQ	2\$				;BRANCH IF IT DID
3268	015064	012777	000100	163564		MOV	#100,	ATPB			;CONTINUE TO TYPE "3"
3269	015072	000002				RTI					
3270											
3271	015074	004767	001274		1\$:	JSR	PC,	POPER			;SAVE ALL THE STUFF ON THE STACK
3272	015100	000406				BR	3\$				
3273											
3274	015102	013767	177776	163470	2\$:	MOV	2#PS,	SPSW			;SAVE THE SPW
3275	015110	022626				CMP	(SP)+,	(SP)+			;RESET THE STACK TO IGNORE THE TRACE TRAP
3276	015112	004767	001264			JSR	PC,	POPER1			;POP ALL THE STUFF OFF THE STACK
3277	015116	005077	163532		3\$:	CLR	ATPS				;CLR INTERUPT ENABLE
3278	015122	022706	000500			CMP	#500,	SP			;CHECK THE STACK POINTER
3279	015126	001406				BEQ	ISA71				;BRANCH IF OK
3280	015130	010667	163446			MOV	SP,	SSP			;SAVE FOR TYPING
3281	015134	012706	000500			MOV	#500,	SP			;RESTORE THE STACK POINTER
3282	015140	104000				HLT					;STACK POINTER FOULED UP
3283	015142	000451				BR	END71				;SKIP REST OF TEST
3284											
3285	015144	010467	163432		ISA71:	MOV	R4,	SSP			;SAVE STACK POINTER
3286	015150	022767	000204	163422		CMP	#204,	SPSW			;CHECK PS AFTER INTERUPT
3287	015156	001401				BEQ	.+4				;BRANCH IF OK
3288	015160	104000				HLT					;PS AFTER INTERUPT NOT EQUAL TO LVLA
3289											
3290	015162	022767	000630	163412		CMP	#STACK0,	SSP			;CHECK THE STACK POINTER (R4)
3291	015170	001401				BEQ	.+4				;BRANCH IF OK
3292	015172	104000				HLT					;STACK POINTER (R4) NOT EQUAL TO #STACK0
3293											
3294	015174	022767	015040	163402		CMP	#RTA71,	ANS1			;CHECK FIS TRAP RETURN ADDRESS
3295	015202	001401				BEQ	.+4				;BRANCH IF OK
3296	015204	104001				HLT+1					;FIS TRAP AT WRONG ADDRESS
3297											
3298	015206	022767	000051	163372		CMP	#051,	ANS2			;CHECK PS BEFORE INTERUPT
3299	015214	001401				BEQ	.+4				;BRANCH IF OK
3300	015216	104002				HLT+2					;PS AT INTERUPT TIME NOT 051
3301											
3302	015220	022767	040200	163362		CMP	#040200,	ANS3			;CHECK DATA FROM THE STACK
3303	015226	001401				BEQ	.+4				;BRANCH IF OK
3304	015230	104004				HLT+4					;DATA ON STACK (040200) CHANGED
3305											
3306	015232	005767	163354			TST	ANS4				;CHECK DATA FROM STACK
3307	015236	001401				BEQ	.+4				;BRANCH IF OK
3308	015240	104004				HLT+4					;DATA ON STACK (000000) CHANGED
3309											
3310	015242	022767	123456	163344		CMP	#123456,	ANS5			;CHECK DATA FROM STACK
3311	015250	001401				BEQ	.+4				;BRANCH IF OK
3312	015252	104006				HLT+6					;DATA ON STACK (123456) CHANGED
3313											
3314	015254	022767	123456	163334		CMP	#123456,	ANS6			;CHECK DATA FROM STACK
3315	015262	001401				BEQ	.+4				;BRANCH IF OK
3316	015264	104006				HLT+6					;DATA ON STACK (123456) CHANGED
3317											
3318	015266	122767	000071	163504	END71:	CMPB	#71,	ICNT			;CHECK THE TEST NUMBER
3319	015274	001401				BEQ	.+4				;BRANCH IF OK
3320	015276	104000				HLT					;WRONG TEST! PC MUST HAVE FOULED UP.

3321  
3322 015300 104400

SCOPE

\*\*\*\*\*  
:TEST 72: TEST THAT FIS ABORTS PROPERLY WHEN INTERRUPTED  
: 107070,070707 \* 040200,000000 = 107070,070707  
: PS = .111, STACK POINTER = RO  
:\*\*\*\*\*

3331	015302	012737	015402	000064	TST72:	MOV	#ISR72, 0#64		;SET UP TELEPRINTER INTERUPT VECTOR
3332	015310	012737	000200	000066		MOV	#200, 0#66		
3333	015316	000004	017426			TYPE,	RETURN		;TYPE CARRIAGE RETURN, LINE FEED
3334	015322	012767	015330	000576		MOV	#.+6, LADS		;RESET LOOP ADDRESS
3335	015330	012777	000100	163316		MOV	#100, 0TPS		;SET TTY INTERUPT ENABLE
3336	015336	012777	000100	163312		MOV	#100, 0TPB		;TYPE "0"
3337	015344	004567	000740			JSR	RS, PUSHR		;PUSH 4 WORDS ONTO RO STACK, SET PRIORITY
3338	015350	040200	000000			.WORD	040200,000000		;SECOND OPERAND ON TOP
3339	015354	107070	070707			.WORD	107070,070707		;FIRST OPERAND ON BOTTOM
3340	015360	000100				.WORD	100		;PROCESSOR PRIORITY LEVEL
3341	015362	016574	000340			.WORD	TRAPER, 340		;FIS TRAP VECTOR
3342	015366	012700	000630			MOV	#STACK0,RO		;SET UP STACK POINTER
3343									
3344	015372	000240				NOP			
3345	015374	075020			RTA72:	FMUL+	RO		;FLOATING MULTIPLY ON THE STACK
3346	015376	024040				CMP	-(RO), -(RO)		;RESET THE STACK POINTER FOR NEXT PASS
3347	015400	000775				BR	RTA72		;REPEAT UNTIL INTERRUPTED
3348									
3349	015402	022716	015374		ISR72:	CMP	#RTA72, (SP)		;CHECK IF INTERRUPT AT FIS INSTR.
3350	015406	001410				BEG	1\$		;BRANCH IF \$\$\$ \$\$\$
3351	015410	022766	015374	000004		CMP	#RTA72, 4(SP)		;CHECK FOR INTERRUPT WITH T-BIT SET
3352	015416	001407				BEG	2\$		;BRANCH IF IT DID
3353	015420	012777	000100	163230		MOV	#100, 0TPB		;CONTINUE TO TYPE "0"
3354	015426	000002				\$\$\$			
3355									
3356	015430	004767	000740		1\$:	JSR	PC, POPER		;SAVE ALL THE STUFF ON THE STACK
3357	015434	000406				BR	3\$		
3358									
3359	015436	013767	177776	163134	2\$:	MOV	0#PS, \$PSW		;SAVE THE SPW
3360	015444	022626				CMP	(SP)+, (SP)+		;RESET THE STACK TO IGNORE THE TRACE TRAP
3361	015446	004767	000730			JSR	PC, POPER1		;POP ALL THE STUFF OFF THE STACK
3362	015452	005077	163176		3\$:	CLR	0TPS		;CLR INTERUPT ENABLE
3363	015456	022706	000500			CMP	#500, SP		;CHECK THE STACK POINTER
3364	015462	001406				BEG	ISA72		;BRANCH IF OK
3365	015464	010667	163112			MOV	SP, \$SP		;SAVE FOR TYPING
3366	015470	012706	000500			MOV	#500, SP		;RESTORE THE STACK POINTER
3367	015474	104000				HLT			;STACK POINTER FOULED UP
3368	015476	000451				BR	END72		;SKIP REST OF TEST
3369									
3370	015500	010067	163076		ISA72:	MOV	RO, \$SP		;SAVE STACK POINTER
3371	015504	022767	000204	163066		CMP	#204, \$PSW		;CHECK PS AFTER INTERUPT
3372	015512	001401				BEG	+.4		;BRANCH IF OK
3373	015514	104000				HLT			;PS AFTER INTERUPT NOT EQUAL TO LVLA
3374									
3375	015516	022767	000630	163056		CMP	#STACK0,\$SP		;CHECK THE STACK POINTER (RO)
3376	015524	001401				BEG	+.4		;BRANCH IF OK





3413	015656	000240			DONE:	NOP		
3414	015660	032737	002000	177570		BIT	#SW10, @SWR	; RING THE BELL?
3415	015666	001002				BNE	1\$	; NO!
3416	015670	000004	000007			TYPE	, BELL	
3417	015674	005046			1\$:	CLR	-(6)	; CLEAR TRACE TRAP
3418	015676	032737	010000	177570		BIT	#SW12, @SWR	; RUN WITH TRT?
3419	015704	001010				BNE	2\$	
3420	015706	005167	000056			COM	, TBIT	
3421	015712	100005				BPL	2\$	
3422	015714	052716	000020			BIS	#20, (6)	; SET TRACE TRAP
3423	015720	012746	015752			MOV	#3\$, -(6)	; JUMP TO START OF TEST
3424	015724	000002				RTI		
3425	015726	012746	015734		2\$:	MOV	#4\$, -(6)	; JUMP TO START OF TEST
3426	015732	000002				RTI		; RETURN
3427	015734	013700	000042		4\$:	MOV	@#42, R0	; GET MONITOR ADDRESS
3428	015740	001404				BEQ	3\$	; IF NONE
3429	015742	004710				JSR	7, (0)	; GO TO MONITOR
3430	015744	000240				NOP		
3431	015746	000240				NOP		
3432	015750	000240				NOP		
3433	015752	062767	000001	163026	3\$:	ADD	#1, PASSES+2	; INC PASS COUNTER
3434	015760	005567	163020			ADC	PASSES	
3435	015764	000137	000200			JMP	@#200	; RETURN
3436								
3437	015770	000000			.REZZ:	0		
3438								
3439	015772	000006			YESRT:	RTT		; RETURN FROM TRACE TRAP
3440	015774	032737	000400	177570	SCOPE\$:	BIT	#SW08, @SWR	; KILL LOUB OR LOOP ON SPEC. TEST
3441	016002	001404				BEQ	1\$	
3442	016004	123767	177570	162766		CMPB	@SWR, ICNT	; ON RIGHT TEST? *SW7-0*
3443	016012	001434				BEQ	OVERS	
3444	016014	032737	040000	177570	1\$:	BIT	#SW14, @SWR	; LOOP ON TEST
3445	016022	001026				BNE	KITS	
3446	016024	032737	004000	177570		BIT	#SW11, @SWR	; KILL ITERATIONS
3447	016032	001012				BNE	SVLAD\$	
3448	016034	105767	162741			TSTB	ICNT+1	
3449	016040	001404				BEQ	2\$	; BRANCH IF FIRST
3450	016042	126767	000062	162731		CMPB	TIMES, ICNT+1	; DONE?
3451	016050	001013				BNE	KITS	; BRANCH IF NOT
3452	016052	112767	000001	162721	2\$:	MOVB	#1, ICNT+1	; FIRST ITERATION
3453	016060	105267	162714		SVLAD\$:	INCB	ICNT	; COUNT TEST NUMBERS
3454	016064	011667	000036			MOV	(6), LAD\$	; SAVE LOOP ADDRESS
3455	016070	016737	162704	177570		MOV	ICNT, @#DISPLAY	; DISPLAY TEST NO. AND ITERATION COUNT
3456	016076	000002				RTI		; RETURN
3457								
3458	016100	105267	162675		KITS:	INCB	ICNT+1	
3459	016104	016737	162670	177570	OVERS:	MOV	ICNT, @#DISPLAY	; SET UP DISPLAY
3460	016112	005767	000010			TST	LAD\$	; FIRST ONE?
3461	016116	001760				BEQ	SVLAD\$	
3462	016120	016716	000002			MOV	LAD\$, (6)	; FUDGE RETURN ADDRESS
3463	016124	000002				RTI		; FIXES PS
3464								
3465	016126	000000			LAD\$:	0		; LOOP ADDRESS
3466	016130	000377			TIMES:	377		; RUN 377 TIMES



```

3467
3468 ;SUBROUTINE TO PUSH 4 WORDS ONTO THE STACK
3469
3470 016132 005726          PUSHES: TST      (SP)+      ;POP STACK BY 1
3471 016134 062705 000010 ADD      #10,      R5        ;POINT TO END OF DATA
3472 016140 014546          MOV      -(R5),    -(SP)    ;PUSH DATA ONTO THE STACK
3473 016142 014546          MOV      -(R5),    -(SP)    ;PUSH DATA ONTO THE STACK
3474 016144 014546          MOV      -(R5),    -(SP)    ;PUSH DATA ONTO THE STACK
3475 016146 014546          MOV      -(R5),    -(SP)    ;PUSH DATA ONTO THE STACK
3476 016150 062705 000010 ADD      #10,      R5        ;POINT TO END OF DATA
3477 016154 012537 177776 MOV      (R5)+,    @#PS     ;SET THE PROCESSOR STATUS
3478 016160 012577 162464 MOV      (R5)+,    @FISVEC  ;SET UP FIS ERROR TRAP VECTOR
3479 016164 012577 162462 MOV      (R5)+,    @FISLVL  ;TRAP STATUS
3480 016170 000115          JMP      (R5)          ;RETURN
3481
3482
3483 ;SUBROUTINE TO POP 2 WORDS OFF THE STACK
3484 ;ALSO SAVES THE PROCESSOR STATUS WORD (EXCEPT T BIT)
3485
3486 016172 013767 177776 162400 POPS:  MOV      @#PS,    SPSW    ;SAVE PROCESSOR STATUS WORD
3487 016200 042767 000020 162372 BIC      #20,      SPSW    ;CLEAR T-BIT
3488 016206 012604          MOV      (SP)+,    R4      ;SAVE RTS ADDRESS
3489 016210 012667 162370 MOV      (SP)+,    ANS1    ;SAVE THE ANSWER
3490 016214 012667 162366 MOV      (SP)+,    ANS2    ;
3491 016220 010667 162356 MOV      SP,      $SP     ;SAVE THE STACK POINTER
3492 016224 000114          JMP      (R4)          ;RETURN
3493
3494
3495 ;SUBROUTINE TO POP 6 WORDS OFF THE STACK.
3496 ;THE FIRST TWO WERE PUT ON BY THE ERROR TRAP,
3497 ;THE LAST FOUR WERE THE ORIGINAL INPUT DATA.
3498 ;ALSO SAVES THE PS AND STACK POINTER.
3499
3500 016226 013767 177776 162344 POPES:  MOV      @#PS,    SPSW    ;SAVE PROCESSOR STATUS WORD
3501 016234 012604          MOV      (SP)+,    R4      ;SAVE RTS ADDRESS
3502 016236 012667 162342 MOV      (SP)+,    ANS1    ;SAVE RTI ADDRESS
3503 016242 011667 162340 MOV      (SP),     ANS2    ;SAVE RTI STATUS
3504 016246 042767 000020 162332 BIC      #20,      ANS2    ;CLEAR THE T-BIT
3505 016254 012746 016262 MOV      #1$,     -(SP)   ;
3506 016260 000002          RTI          ;RESTORE THE PROCESSOR STATUS
3507 016262 012667 162322 IS:    MOV      (SP)+,    ANS3    ;SAVE DATA
3508 016266 012667 162320 MOV      (SP)+,    ANS4    ;
3509 016272 012667 162316 MOV      (SP)+,    ANS5    ;
3510 016276 012667 162314 MOV      (SP)+,    ANS6    ;
3511 016302 010667 162274 MOV      SP,      $SP     ;SAVE SP
3512 016306 000114          JMP      (R4)          ;RTS
3513
3514 ;SUBROUTINE TO PUSH 4 WORDS ONTO THE STACK
3515
3516 016310 012704 000630 PUSHR:  MOV      #STACK0,R4   ;SET R4 TO STACK
3517 016314 012524          MOV      (R5)+,    (R4)+  ;PUT DATA ON STACK
3518 016316 012524          MOV      (R5)+,    (R4)+  ;
3519 016320 012524          MOV      (R5)+,    (R4)+  ;
3520 016322 012524          MOV      (R5)+,    (R4)+  ;
3521 016324 012537 177776 MOV      (R5)+,    @#PS     ;SET THE PROCESSOR STATUS
3522 016330 012577 162314 MOV      (R5)+,    @FISVEC ;SET UP FIS ERROR TRAP VECTOR

```

```

3523 016334 012577 162312      MOV    (R5)+, @FISLVL ;TRAP STATUS
3524 016340 000205              RTS    R5             ;RETURN
3525
3526
3527 ;SUBROUTINE TO POP 2 WORDS OFF THE STACK
3528 ;ALSO SAVES THE PROCESSOR STATUS WORD (EXCEPT T BIT)
3529
3530 016342 013767 177776 162230 POPR:  MOV    @#PS,   $PSW   ;SAVE PROCESSOR STATUS WORD
3531 016350 042767 000020 162222      BIC    #20,   $PSW   ;CLEAR T-BIT
3532 016356 016767 162252 162220      MOV    STACK4, ANS1  ;SAVE THE ANSWER
3533 016364 016767 162246 162214      MOV    STACK6, ANS2  ;
3534 016372 000207              RTS    %7
3535
3536
3537 ;SUBROUTINE TO POP 6 WORDS OFF THE STACKS.
3538 ;THE TWO OFF THE R6 STACK WERE PUT ON BY THE ERROR TRAP.
3539 ;THE FOUR OFF THE SOFTWARE STACK WERE THE ORIGINAL INPUT DATA.
3540 ;ALSO SAVES THE PS AND STACK POINTER AFTER THE FIS TRAP.
3541
3542 016374 013767 177776 162176 POPER: MOV    @#PS,   $PSW   ;SAVE PROCESSOR STATUS WORD
3543 016402 012667 000056 POPER1: MOV   (SP)+,  SAVRTS ;SAVE RTS ADDRESS
3544 016406 012667 162172      MOV   (SP)+,  ANS1    ;SAVE RTI ADDRESS
3545 016412 011667 162170      MOV   (SP)+,  ANS2    ;SAVE RTI STATUS
3546 016416 042767 000020 162162      BIC   #20,   ANS2    ;CLEAR THE T-BIT
3547 016424 012746 016432      MOV   #18,   -(SP)
3548 016430 000002              RTI
3549 016432 016767 162172 162150 JS:    MOV    STACK0, ANS3  ;RESTORE PROCESSOR STATUS
3550 016440 016767 162166 162144      MOV    STACK2, ANS4  ;SAVE DATA
3551 016446 016767 162162 162140      MOV    STACK4, ANS5
3552 016454 016767 162156 162134      MOV    STACK6, ANS6
3553 016462 000137              JMP    @(%7)+
3554 016464 000000      SAVRTS: 0 ;SIMULATED RTS
3555 ;RTS ADDRESS SAVE
3556
3557 ;SUBROUTINE TO PUSH 4 WORDS ONTO THE PC STACK
3558 016466 012504      PUSH7:  MOV    (R5)+,  R4    ;SET R4 TO STACK
3559 016470 012524      MOV    (R5)+,  (R4)+  ;PUT DATA ON STACK
3560 016472 012524      MOV    (R5)+,  (R4)+
3561 016474 012524      MOV    (R5)+,  (R4)+
3562 016476 012524      MOV    (R5)+,  (R4)+
3563 016500 042737 177757 177776      BIC   #177757, @#PS  ;CLEAR STATUS EXCEPT T-BIT
3564 016506 052537 177776      BIS   (R5)+,  @#PS  ;SET THE PROCESSOR STATUS
3565 016512 012577 162132      MOV   (R5)+,  @FISVEC ;SET UP FIS ERROR TRAP VECTOR
3566 016516 012577 162130      MOV   (R5)+,  @FISLVL ;TRAP STATUS
3567 016522 000205              RTS    R5             ;RETURN
3568
3569 ;SUBROUTINE TO POP 4 WORDS OFF THE PC "STACK"
3570 ;ALSO SAVES THE PROCESSOR STATUS WORD (EXCEPT T BIT)
3571
3572 016524 013767 177776 162046 POP7:  MOV    @#PS,   $PSW   ;SAVE PROCESSOR STATUS WORD
3573 016532 042767 000020 162040      BIC   #20,   $PSW   ;CLEAR T-BIT
3574 016540 011600      MOV   (SP),   R0    ;GET RETURN ADDRESS
3575 016542 162700 000014      SUB   #14,   R0    ;POINT TO TOP OF "PC STACK"
3576 016546 012067 162032      MOV   (R0)+,  ANS1  ;SAVE 1ST HALF INPUT DATA
3577 016552 012067 162030      MOV   (R0)+,  ANS2  ;SAVE 2ND HALF INPUT DATA
3578 016556 010067 162020      MOV   R0,    $SP   ;SAVE ASSUMED END PC "STACK POINTER"
    
```



```

3579 016562 012067 162022          MOV      (R0)+, ANS3      ;SAVE 1ST HALF OF ANSWER
3580 016566 012067 162020          MOV      (R0)+, ANS4      ;SAVE 2ND HALF OF ANSWER
3581 016572 000207                    RTS      %7
3582
3583                                ;ERRONIOUS TRAP SERVICE ROUTINE
3584
3585 016574 104000          TRAPER: HLT              ;FIS SHOULDN'T HAVE TRAPED
3586 016576 000002          RTI
3587
3588 016600 032737 002000 177570 HLT$: BIT      #SW10, @#SWR      ;BELL ON ERROR?
3589 016606 001402                    BEQ      1$              ;NO - SKIP
3590 016610 000004 000007                    TYPE    , BELL          ;RING BELL
3591 016614 005267 162162                    INC      ERRORS          ;COUNT THE NUMBER OF ERRORS
3592 016620 032737 020000 177570 1$: BIT      #SW13, @#SWR      ;SKIP TYPEOUT IF SET
3593 016626 001017                    BNE     2$              ;SKIP TYPEOUTS
3594 016630 000004 017426                    TYPE    , RETURN
3595 016634 011667 000060                    MOV     (6), HLTADS      ;PUT ADDRESS OF INSTRUCTION ON STACK
3596 016640 162767 000002 000052          SUB     #2, HLTADS
3597 016646 016705 000046                    MOV     HLTADS, TTY
3598 016652 004767 000106                    JSR    %7, PRINTR
3599 016656 000004 017434                    TYPE    , SPACE+3
3600 016662 004767 000034                    JSR    PC, ERRORS
3601 016666 005737 177570          2$: TST     @#SWR
3602 016672 100001                    BPL     .+4
3603 016674 000000                    HALT
3604 016676 032737 001000 177570          BIT     #SW09, @#SWR
3605 016704 001001                    BNE     .+4
3606 016706 000002          RTI
3607 016710 105067 162065          CLRB   ICNT+1
3608 016714 000167 177160          JMP     KITS
3609
3610 016720 000000          HLTADS: 0
3611
3612 016722 117767 177772 000032 ERRORS: MOVB   @HLTADS, TYPCNT ;TYPE COUNT IS LOW BYTE OF HLT
3613 016730 105267 000026                    INCB   TYPCNT          ;TYPE COUNT = X+1
3614 016734 012703 000600                    MOV    #SPSW, R3      ;TOP OF DATA TO BE TYPED
3615 016740          ERR1$:
3616 016740 012305                    MOV    (R3)+, TTY
3617 016742 004767 000016                    JSR    %7, PRINTR
3618 016746 000004 017435                    TYPE    , SPACE+4
3619 016752 105367 000004                    DECB   TYPCNT
3620 016756 100370                    BPL    ERR1$
3621 016760 000207          RTS      PC
3622
3623 016762 000000          TYPCNT: 0

```

3624	016764	112767	000001	000130	PRINTR: MOVB	#1, .PR	;SET ZERO FILL SWITCH
3625	016772	000402			BR	+.6	;SKIP
3626	016774	005067	000122		PRINTS: CLR	.PR	;SUPPRESS LEADING ZERO'S
3627	017000	112767	177772	000115	MOVB	#-6, .PR+1	;SET COUNT
3628	017006	010446			MOV	R4, -(6)	;SAVE R4
3629	017010	012704	017112		MOV	#.PRBUF, R4	;SET POINTER TO FIRST ASCII CHAR.
3630	017014	105014			CLRB	(4)	;CLEAR FIRST BYTE
3631	017016	000405			BR	.PRF	;ROTATE FIRST BIT
3632	017020	105014			.PRL: CLRB	(4)	;CLEAR BYTE OF CHARACTER
3633	017022	006105			ROL	TTY	;ROTATE BIT INTO C
3634	017024	106114			ROLB	(4)	;PACK IT
3635	017026	006105			ROL	TTY	;ROTATE BIT INTO C
3636	017030	106114			ROLB	(4)	;PACK IT
3637	017032	006105			.PRF: ROL	TTY	;ROTATE BIT INTO C
3638	017034	106114			ROLB	(4)	;PACK IT
3639	017036	105714			TSTB	(4)	;IS IT ZERO?
3640	017040	001402			BEQ	+.6	;SKIP INC
3641	017042	105267	000054		INCB	.PR	;SET FILL SWITCH
3642	017046	105767	000050		TSTB	.PR	;CHECK FILL SWITCH
3643	017052	001402			BEQ	+.6	;SKIP BITSET
3644	017054	152724	000060		BISB	#'0, (4)+	;MAKE INTO ASCII CHAR
3645	017060	105267	000037		INCB	.PR+1	;INC COUNT
3646	017064	001355			BNE	.PRL	;REPEAT
3647	017066	022704	017112		CMP	#.PRBUF, R4	;EMPTY BUFFER?
3648	017072	001002			BNE	+.6	;SKIP IF NOT
3649	017074	112724	000060		MOVB	#'0, (4)+	;LOAD 1 ZERO
3650	017100	105014			CLRB	(4)	;NULL TERMINATOR
3651	017102	000004	017112		TYPE	.PRBUF	;TYPE IT
3652	017106	012604			MOV	(6)+, R4	;RESTORE R4
3653	017110	000207			RTS	PC	;RETURN
3654							
3655	017112	000004			.PRBUF: .BLKW	4	;OUTPUT BUFFER
3656	017122	000000			.PR: 0		;COUNT AND SWITCH



```

3657 017124 012777 017272 000154 PDOWN: MOV #ILLUP, @PUVECS ;SET FOR FAST UP
3658 017132 012777 000340 000150 MOV #340, @PUVECS+2 ;PRIO:7
3659 017140 017767 161510 000132 MOV @TPS, SAVTPS ;SAVE TELEPRINTER STATUS
3660 017146 010046 MOV R0, -(6) ;PUSH R0 ON STACK
3661 017150 010146 MOV R1, -(6) ;PUSH R1 ON STACK
3662 017152 010246 MOV R2, -(6) ;PUSH R2 ON STACK
3663 017154 010346 MOV R3, -(6) ;PUSH R3 ON STACK
3664 017156 010446 MOV R4, -(6) ;PUSH R4 ON STACK
3665 017160 010546 MOV R5, -(6) ;PUSH R5 ON STACK
3666 017162 010667 000110 MOV SP, SAVR6 ;SAVE SP
3667 017166 012777 017176 000112 MOV #PUPS, @PUVECS ;SET UP VECTOR
3668 017174 000000 HALT
3669
3670 017176 016706 000074 PUPS: MOV .SAVR6, SP ;GET SP
3671 017202 005001 CLR R1 ;WAIT LOOP FOR THE TTY
3672 017204 005201 15: INC R1 ;WAIT FOR THE INC
3673 017206 001376 BNE 15 ;OF A WORD
3674 017210 012605 MOV (6)+, R5 ;POP STACK INTO R5
3675 017212 012604 MOV (6)+, R4 ;POP STACK INTO R4
3676 017214 012603 MOV (6)+, R3 ;POP STACK INTO R3
3677 017216 012602 MOV (6)+, R2 ;POP STACK INTO R2
3678 017220 012601 MOV (6)+, R1 ;POP STACK INTO R1
3679 017222 012600 MOV (6)+, R0 ;POP STACK INTO R0
3680 017224 012777 017124 000050 MOV #PDOWN, @PDVECS ;SET UP THE POWER DOWN VECTOR
3681 017232 012777 000340 000044 MOV #340, @PDVECS+2 ;PRIO:7
3682 017240 000004 017312 TYPE POWERS
3683 017244 032767 000100 000026 BIT #100, SAVTPS ;CHECK INT ENB BIT
3684 017252 001406 BEG 25 ;BRANCH IF NOT
3685 017254 012777 000100 161372 MOV #100, @TPS ;SET INT ENB
3686 017262 012777 000100 161366 MOV #100, @TPB ;TYPE AN "3"
3687 017270 000002 25: RTI
3688
3689 017272 000000 ILLUP: HALT ;THE POWER UP SEQUENCE WAS STARTED
3690 017274 000776 BR .-2 ;BEFORE THE POWER DOWN WAS COMPLETE
3691
3692 017276 000000 .SAVR6: 0 ;PUT THE SP HERE
3693 017300 000000 SAVTPS: 0 ;LOC TO SAVE TELEPRINTER STATUS
3694 017302 000024 000026 PDVECS: 24, 26 ;POWER DOWN VECTOR
3695 017306 000024 000026 PUVECS: 24, 26 ;POWER UP VECTOR
3696 017312 005015 047520 042527 POWERS: .ASCIZ <15><12>"POWER"
3697 017320 000122
3698 .EVEN
    
```

3699	017322	010546			.IOT:	MOV	TTY, -(6)		;SAVE TTY
3700	017324	017605	000002			MOV	2(6), TTY		;GET ADDRESS TO BE TYPED
3701	017330	032705	177400			BIT	#177400, TTY		;IS IT A TYPEN?
3702	017334	001004				BNE	1\$		;NO
3703	017336	010567	000076			MOV	TTY, .TYPE		;GET THE CHARACTER
3704	017342	012705	017440			MOV	#.TYPE, TTY		;FUDGE THE ADDRESS
3705	017346	105715			1\$:	TSTB	(TTY)		;TERMINATOR?
3706	017350	001406				BEQ	2\$		;GET OUT IF SO
3707	017352	112537	177566			MOVB	(TTY)+, 2#177566		;LOAD AND TYPE THE CHARACTER
3708	017356	105737	177564			TSTB	2#177564		;IS THE PRINTER READY
3709	017362	100375				BPL	:-4		;WAIT UNTIL IT IS
3710	017364	000770				BR	1\$		;GET THE NEXT CHARACTER
3711	017366	017646	000002		2\$:	MOV	2(6), -(6)		;GET ADDRESS TO BE TYPED
3712	017372	062766	000002	000004		ADD	#2, 4(6)		;ADD 2 TO THE ADDRESS
3713	017400	022666	000002			CMP	(6)+, 2(6)		;IS IT .+2?
3714	017404	001006				BNE	3\$		;NO
3715	017406	062705	000002			ADD	#2, TTY		;ADD 2 TO THE ADDRESS
3716	017412	042705	000001			BIC	#1, TTY		;BACK UP TO AN EVEN BYTE
3717	017416	010566	000002			MOV	TTY, 2(6)		;RESTORE ADDRESS
3718	017422	012605			3\$:	MOV	(6)+, TTY		;RESTORE TTY
3719	017424	000002				RTI			;RETURN
3720									
3721	017426	005015	000		RETURN:	.ASCIZ	<15><12>		;RETURN AND LINEFEED
3722	017431	015	020012	020040	SPACE:	.ASCIZ	<15><12>" "		;RETURN AND 3 SPACES
3723	017436	000							
3724		017440			.EVEN				
3725	017440	000000			.TYPE:	0			;CHARACTER TYPE LOCATION
3726		000001			.END				









		2546	2550	2554	2558	2562	2590	2594	2598	2602	2606	2610	2638	2642
		2646	2650	2654	2658	2686	2690	2694	2698	2702	2706	2734	2738	2742
		2746	2750	2754	2793	2797	2801	2805	2809	2837	2852	2856	2860	2864
		2868	2872	2879	2908	2925	2929	2933	2937	2941	2945	2949	2953	2960
		2989	3006	3010	3014	3018	3022	3026	3030	3034	3041	3065	3072	3076
		3080	3084	3088	3092	3096	3100	3104	3124	3131	3135	3139	3143	3147
		3197	3203	3207	3211	3215	3219	3223	3227	3231	3235	3282	3288	3292
		3296	3300	3304	3308	3312	3316	3320	3367	3373	3377	3381	3385	3389
		3393	3397	3401	3405	3585								
HLTADS	016720	3595*	3596*	3597	3610#	3612								
HLTS	016600	439	3588#											
ICNT	001000	428#	444*	483	524	565	606	647	688	729	770	832	873	935
		976	1017	1079	1120	1182	1223	1264	1305	1346	1408	1449	1511	1552
		1593	1634	1675	1716	1757	1798	1839	1880	1942	1983	2045	2086	2148
		2189	2251	2292	2354	2395	2457	2498	2560	2608	2656	2704	2752	2807
		2877	2958	3039	3102	3145	3233	3318	3403	3442	3448	3450	3452*	3453*
		3455	3458*	3459	3607*									
ILLUP	017272	3657	3689#											
ISA13	002766	902	907#											
ISA20	003774	1149	1154#											
ISA25	004722	1375	1380#											
ISA41	007052	1909	1914#											
ISA51	010632	2321	2326#											
ISA55	011516	2527	2532#											
ISA70	014612	3194	3200#											
ISA71	015144	3279	3285#											
ISA72	015500	3364	3370#											
ISR11	002416	787	798#											
ISR13	002744	890	900#											
ISR16	003416	1034	1045#											
ISR20	003752	1137	1147#											
ISR25	004700	1363	1373#											
ISR27	005234	1466	1477#											
ISR41	007030	1897	1907#											
ISR43	007366	2000	2011#											
ISR45	007720	2103	2114#											
ISR47	010260	2206	2217#											
ISR51	010610	2309	2319#											
ISR53	011144	2412	2423#											
ISR55	011474	2515	2525#											
ISR66	014064	3051	3068#											
ISR67	014300	3114	3127#											
ISR70	014514	3161	3179#											
ISR71	015046	3246	3264#											
ISR72	015402	3331	3349#											
KITS	016100	3445	3451	3458#	3608									
LADS	016126	445*	3164*	3249*	3334*	3454*	3460	3462	3465*					
N	= 000073	372#	448	489#	530#	571#	612#	653#	694#	735#	776#	838#	879#	941#
		982#	1023#	1085#	1126#	1188#	1229#	1270#	1311#	1352#	1414#	1455#	1517#	1558#
		1599#	1640#	1681#	1722#	1763#	1804#	1845#	1886#	1948#	1989#	2051#	2092#	2154#
		2195#	2257#	2298#	2360#	2401#	2463#	2504#	2566#	2614#	2662#	2710#	2758#	2782
		2812#	2883#	2964#	3045#	3108#	3151#	3154	3239#	3324#	3409#			
OVERS	016104	3443	3459#											
PASSES	001004	430#	3433*	3434*										
PC	=%000007	387#	465*	506*	547*	587*	628*	670*	711*	752*	855*	957*	998*	1102*
		1205*	1246*	1286*	1328*	1430*	1534*	1575*	1616*	1657*	1698*	1739*	1780*	1821*

# E07

MAINDEC-11-DBKEA-B  
DBKEAB.P11

KE11F (PDP-11 FIS) INSTRUCTION TESTS.  
CROSS REFERENCE TABLE -- USER SYMBOLS

MACY11 27(732) 20-SEP-76 13:41 PAGE 85

		1861*	1965*	2067*	2171*	2273*	2377*	2480*	2581*	2587*	2629*	2635*	2677*	2683*
		2725*	2731*	3186*	3191*	3271*	3276*	3356*	3361*	3600*	3621*	3653*		
PDOWN\$	017124	435	3657*	3680										
PVECS	017302	435*	436*	3680*	3681*	3694*								
POPER	016374	798	1045	1477	2011	2114	2217	2423	3068	3127	3186	3271	3356	3542*
POPER1	016402	3191	3276	3361	3543*									
POPES	016226	900	1147	1373	1907	2319	2525	3500*						
POPR	016342	465	506	547	670	711	752	793	855	1040	1102	1205	1246	1328
		1472	1534	1575	1616	1657	1698	1739	1780	1821	1965	2006	2109	2171
POPS	016172	2212	2377	2418	2480	3063	3530*							
		587	628	895	957	998	1142	1286	1368	1430	1861	1902	2067	2273
POP7	016524	2314	2520	3486*										
POWERS	017312	2587	2635	2683	2731	3572*								
PRINTR	016764	3682	3696*											
PRINTS	016774	3598	3617	3624*										
PS =	177776	3626*												
		375*	2778*	2786	2825	2834	2841	2896	2905	2912	2977	2986	2993	3116*
PUPS	017176	3122	3189	3274	3359	3477*	3486	3500	3521*	3530	3542	3563*	3564*	3572
PUSHR	016310	3667	3670*											
		455	496	537	660	701	742	783	845	1030	1092	1195	1236	1318
		1462	1524	1565	1606	1647	1688	1729	1770	1811	1955	1996	2099	2161
PUSH\$	016132	2202	2367	2408	2470	3053	3167	3252	3337	3516*				
		578	619	886	948	989	1133	1277	1359	1421	1852	1893	2058	2264
		2305	2511	3470*										
PUSH7	016466	2573	2621	2669	2717	3558*								
PVECS	017306	3657*	3658*	3667*	3695*									
RETURN	017426	3163	3248	3333	3594	3721*								
RTA11	002402	793*	808											
RTA13	002730	895*	911											
RTA16	003402	1040*	1055											
RTA20	003736	1142*	1158											
RTA25	004664	1368*	1384											
RTA27	005220	1472*	1487											
RTA41	007014	1902*	1918											
RTA43	007352	2006*	2021											
RTA45	007704	2109*	2124											
RTA47	010244	2212*	2227											
RTA51	010574	2314*	2330											
RTA53	011130	2418*	2433											
RTA55	011460	2520*	2536											
RTA66	014050	3063*	3078											
RTA67	014262	3122*	3137											
RTA70	014506	3175*	3177	3179	3181	3209								
RTA71	015040	3260*	3262	3264	3266	3294								
RTA72	015374	3345*	3347	3349	3351	3379								
RO =%	000000	379*	438*	460*	463*	466	1241*	1244*	1247	1570*	1573*	1576	2001*	2004*
		2007	2012	2166*	2169*	2172	2825*	2874	2896*	2955	2977*	3036	3342*	3345*
R1 =%	000001	3346	3370	3427*	3574*	3575*	3576	3577	3578	3579	3580	3660	3679*	3745*
		380*	501*	504*	507	706*	709*	712	1035*	1038*	1041	1046	1200*	1203*
		1206	1960*	1963*	1966	2413*	2416*	2419	2424	2818*	2819*	2820*	2821*	2822*
		2843*	2846	2847	2848	2849	2889*	2890*	2891*	2892*	2893*	2917*	2919	2920
		2921	2922	2970*	2971*	2972*	2973*	2974*	2998*	3000	3001	3002	3003	3172*
		3175*	3176	3200	3661	3671*	3672*	3678*						
R2 =%	000002	381*	542*	545*	548	1652*	1655*	1658	1734*	1737*	1740	2372*	2375*	2378
		2845*	2846*	2847*	2848*	2849*	2918*	2919*	2920*	2921*	2922*	2999*	3000*	3001*
		3002*	3003*	3058*	3061*	3064	3069	3662	3677*					















DUMP	343#	3597	3615					
POP	343#	3674						
PRINT	343#							
PUSH	343#	3660						
SDUMP	343#							
SADDER	448#	776	1023					
SADDES	448#	879	1126					
SADDR	448#	489	530	653	694	735	838	1085
SADDS	448#	571	612	941	982			
SADRER	448#	3045	3108					
SALL	448#	2758						
SALL2	448#	2782						
SCATCH	343#	396						
SDIVER	448#	2092	2195	2401				
SDIVES	448#	2298	2504					
SDIVR	448#	2154	2360	2463				
SDIVS	448#	2051	2257					
SEND	343#	3413						
SEQUAT	343#	372						
SFINT	448#	3154	3239	3324				
SFIS7	448#	2566	2614	2662	2710			
SFLT	343#	3588						
SMULR	448#	1989						
SMULES	448#	1886						
SMULR	448#	1681	1722	1763	1804	1948		
SMULS	448#	1845						
SOCTAL	343#	3624						
SSCOPE	343#	3440						
SSLRED	448#	2883	2964					
SSLRYE	448#	2812						
SSUBR	448#	1455						
SSUBES	448#	1352						
SSUBR	448#	1188	1229	1311	1517	1558	1599	1640
SSUBS	448#	1270	1414					
SSWDOC	343#	348						
STYPE	343#	3699						





	2850	2854	2858	2862	2866	2870	2923	2931	2935	2939	2943	2947	3004	3012	3016
	3020	3024	3028	3032	3070	3074	3078	3082	3086	3090	3094	3098	3129	3133	3137
	3141	3176	3179	3181	3190	3193	3201	3205	3209	3213	3225	3229	3261	3264	3266
	3275	3278	3286	3290	3294	3298	3302	3310	3314	3346	3349	3351	3360	3363	3371
	3375	3379	3383	3387	3395	3399	3647	3713							
CMPB	483	524	565	606	647	688	729	770	832	873	935	976	1017	1079	1120
	1182	1223	1264	1305	1346	1408	1449	1511	1552	1593	1634	1675	1716	1757	1798
	1839	1880	1942	1983	2045	2086	2148	2189	2251	2292	2354	2395	2457	2498	2560
	2608	2656	2704	2752	2807	2877	2958	3039	3102	3145	3233	3318	3403	3442	3450
COM	3420														
DECB	3619														
EMT	373														
FADD	463	504	545	585	626	668	709	750	791	853	893	955	996	1038	1100
	1140	2581	2784	2832	2903	2984	3061								
FDIV	2065	2107	2169	2210	2271	2312	2375	2416	2478	2518	2725	2782	3260		
FMUL	1696	1737	1778	1819	1859	1900	1963	2004	2677	2783	3120	3345			
FSUB	1203	1244	1284	1326	1366	1428	1470	1532	1573	1614	1655	2629	2781	3175	
HALT	397	3603	3668	3689											
INC	3591	3672													
INCB	3453	3458	3613	3641	3645										
IOT	374														
JMP	400	3435	3480	3492	3512	3553	3608								
JSR	455	465	496	506	537	547	578	587	619	628	660	670	701	711	742
	752	783	793	798	845	855	886	895	900	948	957	989	998	1030	1040
	1045	1092	1102	1133	1142	1147	1195	1205	1236	1246	1277	1286	1318	1328	1359
	1368	1373	1421	1430	1462	1472	1477	1524	1534	1565	1575	1606	1616	1647	1657
	1688	1698	1729	1739	1770	1780	1811	1821	1852	1861	1893	1902	1907	1955	1965
	1996	2006	2011	2058	2067	2099	2109	2114	2161	2171	2202	2212	2217	2264	2273
	2305	2314	2319	2367	2377	2408	2418	2423	2470	2480	2511	2520	2525	2573	2587
	2621	2635	2669	2683	2717	2731	3053	3063	3068	3127	3167	3186	3191	3252	3271
MOV	3276	3337	3356	3361	3429	3598	3600	3617							
	433	434	435	436	437	438	439	440	441	442	443	446	460	466	501
	507	542	548	590	631	665	671	706	712	747	753	788	794	799	850
	856	897	903	960	1001	1035	1041	1046	1097	1103	1144	1150	1200	1206	1241
	1247	1289	1323	1329	1370	1376	1433	1467	1473	1478	1529	1535	1570	1576	1611
	1617	1652	1658	1693	1699	1734	1740	1775	1781	1816	1822	1864	1904	1910	1960
	1966	2001	2007	2012	2070	2104	2110	2115	2166	2172	2207	2213	2218	2276	2316
	2322	2372	2378	2413	2419	2424	2475	2481	2522	2528	2767	2768	2769	2770	2771
	2772	2773	2774	2775	2776	2777	2778	2786	2788	2789	2790	2818	2819	2820	2821
	2822	2823	2824	2825	2826	2827	2829	2834	2835	2836	2841	2842	2843	2844	2845
	2846	2847	2848	2849	2874	2875	2889	2890	2891	2892	2893	2894	2895	2896	2897
	2898	2900	2905	2906	2907	2912	2913	2914	2915	2916	2917	2918	2919	2920	2921
	2922	2955	2956	2970	2971	2972	2973	2974	2975	2976	2977	2978	2979	2981	2986
	2987	2988	2993	2994	2995	2996	2997	2998	2999	3000	3001	3002	3003	3036	3037
	3051	3052	3058	3064	3069	3114	3115	3116	3117	3122	3123	3128	3151	3153	3161
	3162	3164	3165	3166	3172	3183	3189	3195	3196	3200	3246	3247	3249	3250	3251
	3257	3268	3274	3280	3281	3285	3331	3332	3334	3335	3336	3342	3353	3359	3365
	3366	3370	3409	3410	3423	3425	3427	3454	3455	3459	3462	3472	3473	3474	3475
	3477	3478	3479	3486	3488	3489	3490	3491	3500	3501	3502	3503	3505	3507	3508
	3509	3510	3511	3516	3517	3518	3519	3520	3521	3522	3523	3530	3532	3533	3542
	3543	3544	3545	3547	3549	3550	3551	3552	3558	3559	3560	3561	3562	3565	3566
	3572	3574	3576	3577	3578	3579	3580	3595	3597	3614	3616	3628	3629	3652	3657
	3658	3659	3660	3661	3662	3663	3664	3665	3666	3667	3670	3674	3675	3676	3677
	3678	3679	3680	3681	3685	3686	3699	3700	3703	3704	3711	3717	3718		
MOV8	3452	3612	3624	3627	3649	3707									
NOP	462	503	544	584	625	667	708	749	790	852	892	954	995	1037	1099



	1139	1202	1243	1283	1325	1365	1427	1469	1531	1572	1613	1654	1695	1736	1777
	1818	1858	1899	1962	2003	2064	2106	2168	2209	2270	2311	2374	2415	2477	2517
	2580	2628	2676	2724	2780	2831	2902	2983	3060	3119	3174	3259	3344	3413	3430
RESET	3431	3432													
ROL	422														
ROLB	3633	3635	3637												
RTI	3634	3636	3638												
	2828	2876	2899	2957	2980	3038	3184	3269	3354	3424	3426	3456	3463	3506	3548
RTS	3586	3606	3687	3719											
RTT	3524	3534	3567	3581	3621	3653									
SUB	3439														
TRAP	3575	3596													
TST	372														
	475	479	520	557	561	602	643	766	800	828	869	931	972	1260	1342
	1708	1712	1790	1794	1831	1835	1979	2033	2350	2391	2490	2494	2604	2652	2700
TSTB	2748	2927	2951	3008	3217	3221	3306	3391	3460	3470	3601				
.ASCIZ	3448	3639	3642	3705	3708										
.BLKW	3696	3721	3722												
.ENABL	3655														
.END	345														
.ENDC	3726														
	352	353	355	356	357	358	359	360	401	468	476	480	509	517	521
	550	558	562	599	603	640	644	673	681	685	714	722	726	755	763
	767	801	813	817	821	825	829	858	866	870	916	920	924	928	932
	969	973	1010	1014	1048	1060	1064	1068	1072	1076	1105	1113	1117	1153	1167
	1171	1175	1179	1208	1216	1220	1249	1257	1261	1298	1302	1331	1339	1343	1389
	1393	1397	1401	1405	1442	1446	1480	1492	1496	1500	1504	1508	1537	1545	1549
	1578	1586	1590	1619	1627	1631	1660	1668	1672	1701	1709	1713	1742	1750	1754
	1783	1791	1795	1824	1832	1836	1873	1923	1927	1931	1935	1939	1968	1976	1980
	2014	2026	2030	2034	2038	2042	2075	2079	2083	2117	2129	2133	2137	2141	2145
	2174	2182	2186	2220	2232	2236	2240	2244	2248	2281	2285	2289	2335	2339	2343
	2347	2351	2380	2388	2392	2426	2438	2442	2446	2450	2454	2483	2491	2495	2541
	2545	2549	2553	2557	2589	2593	2597	2601	2605	2637	2641	2645	2649	2653	2685
	2689	2693	2697	2701	2733	2737	2741	2745	2749	2792	2800	2804	2867	2871	2940
	2944	2948	2952	3021	3025	3029	3033	3058	3066	3071	3083	3087	3091	3095	3099
	3102	3117	3125	3130	3142	3145	3214	3218	3222	3226	3230	3299	3303	3307	3311
	3315	3384	3388	3392	3396	3400	3416	3417	3427	3592	3594	3601	3609	3632	3705
	3720														
.EVEN	3698	3724													
.IF	351	352	353	355	356	357	358	359	397	467	475	479	508	516	520
	549	557	561	598	602	639	643	672	680	684	713	721	725	754	762
	766	800	812	816	820	824	828	857	865	869	915	919	923	927	931
	968	972	1009	1013	1047	1059	1063	1067	1071	1075	1104	1112	1116	1162	1166
	1170	1174	1178	1207	1215	1219	1248	1256	1260	1297	1301	1330	1338	1342	1388
	1392	1396	1400	1404	1441	1445	1479	1491	1495	1499	1503	1507	1536	1544	1548
	1577	1585	1589	1618	1626	1630	1659	1667	1671	1700	1708	1712	1741	1749	1753
	1782	1790	1794	1823	1831	1835	1872	1922	1926	1930	1934	1938	1967	1975	1979
	2013	2025	2029	2033	2037	2041	2074	2078	2082	2116	2128	2132	2136	2140	2144
	2173	2181	2185	2219	2231	2235	2239	2243	2247	2280	2284	2288	2334	2338	2342
	2346	2350	2379	2387	2391	2425	2437	2441	2445	2449	2453	2482	2490	2494	2540
	2544	2548	2552	2556	2588	2592	2596	2600	2604	2636	2640	2644	2648	2652	2684
	2688	2692	2696	2700	2732	2736	2740	2744	2748	2791	2799	2803	2866	2870	2939
	2943	2947	2951	3020	3024	3028	3032	3070	3082	3086	3090	3094	3098	3129	3141
	3145	3213	3217	3221	3225	3229	3298	3302	3306	3310	3314	3383	3387	3391	3395
	3399	3414	3415	3417	3588	3592	3600	3604	3632	3701	3711				
.IFF	467	476	480	508	516	521	549	558	562	598	603	639	644	672	680



	684	713	721	725	754	762	767	801	812	816	820	824	829	857	865
	870	915	919	923	927	932	968	973	1009	1013	1047	1059	1063	1067	1071
	1075	1104	1112	1116	1162	1166	1170	1174	1178	1207	1215	1219	1248	1256	1261
	1297	1301	1330	1338	1343	1388	1392	1396	1400	1404	1441	1445	1479	1491	1495
	1499	1503	1507	1536	1544	1548	1577	1585	1589	1618	1626	1630	1659	1667	1671
	1700	1709	1713	1741	1749	1753	1782	1791	1795	1823	1832	1836	1872	1922	1926
	1930	1934	1938	1967	1975	1980	2013	2025	2029	2034	2037	2041	2074	2078	2082
	2116	2128	2132	2136	2140	2144	2173	2181	2185	2219	2231	2235	2239	2243	2247
	2280	2284	2288	2334	2338	2342	2346	2351	2379	2387	2392	2425	2437	2441	2445
	2449	2453	2482	2491	2495	2540	2544	2548	2552	2556	2588	2592	2596	2600	2605
	2636	2640	2644	2648	2653	2684	2688	2692	2696	2701	2732	2736	2740	2744	2749
	2791	2799	2803	2866	2870	2939	2943	2947	2952	3020	3024	3028	3032	3058	3066
	3070	3082	3086	3090	3094	3098	3116	3122	3129	3141	3145	3213	3218	3222	3225
	3229	3298	3302	3307	3310	3314	3383	3387	3392	3395	3399	3416	3588	3604	3720
.IFNE	3053	3063	3086	3116	3122	3145									
.IIF	3657														
.IRP	3660	3674													
.LIST	343	372	396	397	448	489	530	571	612	653	694	735	776	838	879
	941	982	1023	1085	1126	1188	1229	1270	1311	1352	1414	1455	1517	1558	1599
	1640	1681	1722	1763	1804	1845	1886	1948	1989	2051	2092	2154	2195	2257	2298
	2360	2401	2463	2504	2566	2614	2662	2710	2758	2812	2883	2964	3045	3108	3151
	3153	3239	3324	3409	3413	3467	3588	3624	3657	3699					
.MACRO	343	448													
.NLIST	343	372	396	397	448	489	530	571	612	653	694	735	776	838	879
	941	982	1023	1085	1126	1188	1229	1270	1311	1352	1414	1455	1517	1558	1599
	1640	1681	1722	1763	1804	1845	1886	1948	1989	2051	2092	2154	2195	2257	2298
	2360	2401	2463	2504	2566	2614	2662	2710	2758	2812	2883	2964	3045	3108	3151
	3153	3239	3324	3409	3413	3467	3588	3624	3657	3699					
.PAGE	396	489	530	571	612	653	694	735	776	879	982	1023	1126	1229	1270
	1311	1352	1455	1558	1599	1640	1681	1722	1763	1804	1845	1886	1989	2092	2195
	2298	2401	2504	2662	2710	2758	2812	2964	3045	3153	3413	3467	3624	3657	3699
.REM	1														
.REPT	397														
.SBTTL	343	396	448	1188	1681	2051	2566	2758	2812	3045	3153	3413	3467	3588	3624
	3657	3699													
.TITLE	344														
.WORD	456	457	458	459	497	498	499	500	538	539	540	541	579	580	581
	582	620	621	622	623	661	662	663	664	702	703	704	705	743	744
	745	746	784	785	786	787	846	847	848	849	887	888	889	890	949
	950	951	952	990	991	992	993	1031	1032	1033	1034	1093	1094	1095	1096
	1134	1135	1136	1137	1196	1197	1198	1199	1237	1238	1239	1240	1278	1279	1280
	1281	1319	1320	1321	1322	1360	1361	1362	1363	1422	1423	1424	1425	1463	1464
	1465	1466	1525	1526	1527	1528	1566	1567	1568	1569	1607	1608	1609	1610	1648
	1649	1650	1651	1689	1690	1691	1692	1730	1731	1732	1733	1771	1772	1773	1774
	1812	1813	1814	1815	1853	1854	1855	1856	1894	1895	1896	1897	1956	1957	1958
	1959	1997	1998	1999	2000	2059	2060	2061	2062	2100	2101	2102	2103	2162	2163
	2164	2165	2203	2204	2205	2206	2265	2266	2267	2268	2306	2307	2308	2309	2368
	2369	2370	2371	2409	2410	2411	2412	2471	2472	2473	2474	2512	2513	2514	2515
	2574	2575	2576	2577	2578	2622	2623	2624	2625	2626	2670	2671	2672	2673	2674
	2718	2719	2720	2721	2722	3054	3055	3056	3057	3168	3169	3170	3171	3253	3254
	3255	3256	3338	3339	3340	3341									

ERRORS DETECTED: 0

808

MAINDEC-11-DBKEA-B KE11F (PDP-11 FIS) INSTRUCTION TESTS.  
DBKEAB.P11 CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

MACY11 27(732) 20-SEP-76 13:41 PAGE 97

DEFAULT GLOBALS GENERATED: 0

\* DBKEAB.SEO/SOL/CRF/PAGNUM=DBKEAB

RUN-TIME: 25 38 5 SECONDS

RUN-TIME RATIO: 318/69=4.6

CORE USED: 19K (37 PAGES)



