

FP11

DIVIDE EXERCISER

MD-11-DCFPU-D

EP-DCFPU-D-DL-A

OCT 1976

COPYRIGHT © 1976

digital

FICHE 1 OF 1

Made in U.S.A.

This microfiche card contains a grid of frames. The first column on the left contains frames with titles such as 'EXECUTIVE SUMMARY', 'MANAGEMENT', 'SYSTEMS', 'PROCEDURES', 'APPENDICES', and 'REFERENCES'. The subsequent columns contain frames with various data tables, charts, and diagrams. The frames are arranged in a grid that is approximately 10 frames wide and 20 frames high. The data in the frames is too small to read clearly, but it appears to be organized into sections corresponding to the titles in the first column.

.REPT 0

IDENTIFICATION

PRODUCT CODE:	MAINDEC-11-DCFPD-D-D
PRODUCT NAME:	FP11 DIVIDE EXERCISER
DATE CREATED:	1-NOV-72
MAINTAINER:	DIAGNOSTIC GROUP
AUTHOR:	KEN CHAPMAN

COPYRIGHT (C) DIGITAL EQUIPMENT CORPORATION
1973

THIS MATERIAL IN THIS DOCUMENT IS FOR INFORMATION
PURPOSES ONLY AND IS SUBJECT TO CHANGE WITHOUT NOTICE.
DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY
FOR THE USE OF SOFTWARE ON EQUIPMENT WHICH IS NOT
SUPPLIED BY IT.
DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY
FOR ANY ERRORS WHICH MAY APPEAR IN THE DOCUMENT.

MAINDEC-11-DOFFU-D-D
TABLE OF CONTENTS

FP11 DIVIDE EXERCISER

PAGE 2

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 ABSTRACT
6. ERRORS
7. RESTRICTIONS
8. MISCELLANEOUS
 - 8.1 EXECUTION TIME
 - 8.2 STACK POINTER
 - 8.3 POWER FAIL
9. PROGRAM DESCRIPTION

MAINDEC-11-DCFPD-D-D
DESCRIPTION

FP11 DIVIDE EXERCISER

PAGE 3

1. ABSTRACT

THIS PROGRAM EXERCISES THE FP11 FLOATING POINT DIVIDE INSTRUCTIONS (DIVE AND DIVD) WITH RANDOM NUMBER PATTERNS. THE ANSWERS ARE CHECKED AGAINST RESULTS OBTAINED USING THE CORRESPONDING FORTRAN SOFTWARE ROUTINES.

2. REQUIREMENTS

2.1 EQUIPMENT

PDP11/45 STANDARD COMPUTER WITH FP11 OPTION

2.2 STORAGE

THE ROUTINES USE MEMORY LOCATIONS 0 - 17500. THE MAP AT THE END OF THE LISTINGS SHOWS THE ABSOLUTE LOCATIONS OF THE FORTRAN MATH ROUTINES WHICH WERE ASSEMBLED SEPARATELY AND LINKED TO THE MAIN PROGRAM VIA LNKX11 ON A DECSYSTEM-10.

2.3 PRELIMINARY PROGRAMS

DCFPA THRU DCFPL

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

- 1) LOAD PROGRAM INTO MEMORY USING ABS LOADER.
- 2) LOAD ADDRESS 200.

MAINDEC-11-DOFPU-D
DOFPU.D.F11

FLOATING POINT DIVIDE EXERCISER

E01

MACY11 27(732) 16-SEP-76 16:30 PAGE 4

101
102

3) SET SWITCHES (SEE 5.1.1) ALL DOWN FOR WORST CASE.
4) PRESS START.

MAINDEC-11-DCFPU-D-D FP11 DIVIDE EXERCISER
DESCRIPTION

PAGE 4

5) THE PROGRAM WILL LOOP AND BELL WILL RING ONCE EVERY PASS.
6) THE DISPLAY ON THE 11/45 WILL SHOW THE ITERATION COUNT IN
THE LEFT BYTE AND TEST NUMBER IN THE RIGHT. TO USE, SET THE
DATA DISPLAY SWITCH TO THE DISPLAY POSITION.

5. OPERATING PROCEDURE

5.1 OPERATIONAL SWITCH SETTINGS

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		CORE IMAGE TYPE-OUT (16 BIT WORDS)
		0		FLOATING POINT TYPE-OUT (SIGN, EXPONENT, MANTISSA)
SW<08>	=	1		LOOP ON TEST IN SW<7:0>
		0		LOAD SW<7:0> INTO US REGISTER

5.2 SUBROUTINE ABSTRACTS

5.2.1 SCOPE

THIS SUBROUTINE CALL IS PLACED BETWEEN EACH SUBTEST IN THE
TEST SECTION. IT RECORDS THE STARTING ADDRESS OF EACH
SUBTEST AS IT IS BEING ENTERED IN LOCATION "LAD". IF A
SCOPE LOOP IS REQUESTED, THE CURRENT SUBTEST WILL BE LOOPE
UPON. SW<11> ON A 1 INHIBITS ITERATION OF SUBTESTS. THE
CONTENTS OF LAD MAY BE USED TO DETERMINE THE LAST SUBTEST
SUCCESSFULLY COMPLETED. LAD IS UPDATED INSIDE EACH SUBTEST
AFTER THE FORTRAN ANSWER IS CALCULATED, SO THAT THE
ITERATIONS WILL INCLUDE ONLY THE FP11 PORTION OF THE TEST.

5.2.2 HLT

THIS ROUTINE PRINTS OUT AN ERROR MESSAGE (SEE 6.1). TO
INHIBIT TYPEOUTS, PUT SW<13> ON A 1.

5.2.3 TRTRAP

IF SW<12> IS ON A 0, THE T BIT WILL BE SET ON ALTERNATE
PASSES. WHEN SET, IT CAUSES A TRAP 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

MAINDEC-11-DOFPU-0
DOFPU-11

FLOATING POINT DIVIDE EXERCISER

G01

MACY11 27(732) 16-SEP-76 16:30 PAGE 6

199

REACHED.

S.2.4 TRAPCATCHER

A ".+2" - "HALT" SEQUENCE IS REPEATED FROM D - 776 TO CATCH ANY UNEXPECTED TRAPS. THUS ANY UNEXPECTED TRAPS OR INTERRUPTS WILL HALT AT THE VECTOR + 2.

5.2.5 FLOATING POINT TRAP (TO 244)

SINCE SOME OF THE SUBTESTS HAVE INTERRUPTS ENABLED, THE FLOATING POINT TRAP (FLTERR) CHECKS TO SEE IF FORTRAN ALSO GOT AN ERROR CONDITION. IF FORTRAN DIDN'T INDICATE AN ERROR, OR INTERRUPTS WERE DISABLED AN ERROR HLT OCCURS (SEE U.2.2). IF AN INTERRUPT WAS ANTICIPATED, BUT DIDN'T OCCUR, THE SUBTEST WILL DETECT THE ERROR.

B. ERRORS

5.1 ERROR PRINTOUT

THE FORMAT IS AS FOLLOWS:

ADDRESS, OPERAND, OPERATOR, OPERAND, EQUALS
FPP: ANSWER, FPS, FEC, FEA
FORTRAN: ANSWER, FPS, FEC, FEA

WHERE:

ADDRESS = ADDRESS OF ERROR HLT
 OPERAND = RANDOM FLOATING POINT NUMBER INPUTS
 OPERATOR = ARITHMETIC OPERATOR (+ OR -)
 EQUALS = (=)
 ANSWER = FLOATING POINT ANSWER
 FPS = FLOATING POINT STATUS
 FEC = FLOATING EXCEPTION CODES (ERROR CODES)
 FEA = FLOATING EXCEPTION ADDRESS (ERROR ADDRESS)

TO FIND THE FAILING TEST, LOOK AT THE LISTING ABOVE THE ADDRESS TYPED.

6.2 ERROR RECOVERY

RESTART AT 200

7. RESTRICTIONS

NONE

MAINDEC-11-DCFPD-D-D
DESCRIPTION

FP11 DIVIDE EXERCISER

PAGE 6

8. MISCELLANEOUS

8.1 EXECUTION TIME

A BELL WILL RING WITHIN 15 SECONDS WITH ALL SWITCHES DOWN.

8.2 STACK POINTER

STACK IS INITIALLY SET TO 600

8.3 POWER FAIL

EACH TEST CAN BE POWER FAILED WITH NO ERRORS EXCEPT ON THE
FEC AND FEA. TO USE, START THE TEST AS USUAL AND POWER DOWN
THEN UP AT ANY TIME. THE PROGRAM SHOULD TYPE "POWER" AND
CONTINUE TO RUN WITH NO OTHER TYPEOUTS.

9. PROGRAM DESCRIPTION

THIS PROGRAM TESTS THE DIVIDE INSTRUCTIONS ON THE FP11 IN
ROUND AND TRUNCATE MODES AND WITH INTERRUPTS ON AND OFF.
EACH PROGRAM HAS MANY SUBTESTS (THE CODE BETWEEN 2 SCOPE
STATEMENTS) WHICH ARE RUN 256 TIMES BEFORE CONTINUING TO THE
NEXT. SW<11> ON A 1 CAUSES EACH SUBTEST TO BE RUN ONLY
ONCE. THE ADDRESS ICNT (LOC 1000) AND DISPLAY REGISTER ON
THE 11/45 EACH CONTAIN 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-EXISTANT, THE
PROGRAM WILL BE RUN AS USUAL.

THE FORTRAN MATH ROUTINES WERE TAKEN UNMODIFIED FROM THE
PDP-11 FORTRAN PACKAGE AND ASSEMBLED AS SEPERATE MODULES.
THEY WERE LINKED TO THE MAIN PROGRAMS VIA LNKX11 ON A
DECSYSTEM-10 WHICH PRODUCES A BINARY TAPE IN THE NORMAL
ABSOLUTE FORMAT. THUS, THE PROGRAMS LOAD AND RUN JUST LIKE
ANY OTHER DIAGNOSTIC PROGRAM.

NOTE: SINCE THE FP11 LOAD, STORE, AND COMPARE
INSTRUCTIONS (LDF, LDD, STF, STD, CMPF, AND CMPD) ARE ALSO
USED IN THIS PROGRAM, IT IS POSSIBLE THAT THEY AND NOT THE
DIVIDE INSTRUCTIONS COULD CAUSE ERRORS.

.ENDR

000000

.TITLE MAINDEC-11-DCFPD-D FLOATING POINT DIVIDE EXERCISER
.ASECT
.GLOBL \$DVR,\$DVD,\$ERRA
:COPYRIGHT 1972, DIGITAL EQUIPMENT CORP., MAYNARD, MASS
:PROGRAM BY KEN CHAPMAN

SWITCH	USE
8	0 - LOAD UB REGISTER WITH SW<7:0>
9	1 - LOOP ON TEST IN SW<7:0>
	TTY OUTPUT FORMAT:
	0 - SIGN, EXPONENT, MANTISSA
	1 - CORE IMAGE (16 BIT WORDS)
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

:OUTPUT FORM:

: ADDRESS, OPERAND, OPERATOR, OPERAND, EQUALS
: FPP: ANSWER, FPS, FEC, FEA
: FORTRAN:

BIT	FPS	REASON	CODE	FEC	ERROR
0		CARRY	0		ADDRESS ERROR
1		OVERFLOW	2		OPCODE ERROR
2		ZERO	4		DIVIDE BY ZERO
3		NEGATIVE	6		CONVERSION ERROR
4		MAINTAINANCE MODE	10		OVERFLOW
5		TRUNCATE MODE	12		UNDERFLOW
6		LONG INTEGER MODE	14		UNDEFINED VARIABLE (-0)
7		DOUBLE PRECISION MODE	16		UBREAK TRAP
8		INTERUPT ON CONVERSION ERROR			
9		INTERUPT ON OVERFLOW			
10		INTERUPT ON UNDERFLOW			
11		INTERUPT ON UNDEFINED VARIABLE			
12					
13					
14		INTERUPT ENABLE			
15		ERROR FLAG			

000000			RO=	%0	
000001			R1=	%1	
000002			R2=	%2	
000003			R3=	%3	
000004			R4=	%4	
000005			R5=	%5	
000005			TTY=	%5	
000006			SP=	%6	
000007			PC=	%7	
000000			AC0=	%0	
000001			AC1=	%1	
000002			AC2=	%2	
000003			AC3=	%3	
000004			AC4=	%4	
000005			AC5=	%5	
000400			SW08=	000400	
001000			SW09=	001000	
002000			SW10=	002000	
004000			SW11=	004000	
010000			SW12=	010000	
020000			SW13=	020000	
040000			SW14=	040000	
100000			SW15=	100000	
177570			SWR=	177570	
177776			PS=	177776	
177570			DISPLAY=	SWR	
000000			DUMMY=	HALT	
000240			NOP=	240	
104400			SCOPE=	TRAP	
104000			HLT=	EMT	
000004			TYPE=	IOT	
000207			BELL=	207	
000000			. =	0	;TRAP CATCHER FROM 0 - 776
000200			. =	200	
000200	000167	000652		JMP	BEG
001000	001000		. =	1000	
001000	000000		ICNT:	0	
001002	000000		LONUM:	DUMMY	
001004	000000			DUMMY	
001006	000000			DUMMY	
001010	000000			DUMMY	
001012	000000		HINUM:	DUMMY	
001014	000000			DUMMY	
001016	000000			DUMMY	
001020	000000			DUMMY	

001022	000000			ANS1:	DUMMY	
001024	000000				DUMMY	
001026	000000				DUMMY	
001030	000000				DUMMY	
001032	000000			ANS2:	DUMMY	
001034	000000				DUMMY	
001036	000000				DUMMY	
001040	000000				DUMMY	
001042	000000			FPS:	0	:FLOATING POINT STATUS
001044	000000			FEC:	0	:FLOATING EXCEPTION CODES
001046	000000			FPC:	0	:FLOATING PC
001050	000000			\$FPS:	0	:FORTRAN FLOATING POINT STATUS
001052	000000			\$FEC:	0	:FORTRAN FLOATING EXCEPTION CODES
001054	000000			\$FPC:	0	:FORTRAN FLOATING PC
001056	012706	000600		BEG:	MOV	#600, SP
001062	012737	001104	000004		MOV	#M1120, 3#4
001070	005737	177772			TST	3#177772
001074	012767	000006	005034		MOV	#6, YESRT
001102	000403				BR	BEGIN
001104	016737	010050	000010	M1120:	MOV	FPTADR, 3#10
						:LOAD THE ILLEGAL INSTRUCTION VECTOR
						: WITH THE ADDRESS OF THE FPU.
						: THE FPU WILL HANDLE THE BAD OPCODES
						:RESET 4
001112	012737	000006	000004	BEGIN:	MOV	#6, 3#4
001120	012706	000600			MOV	#600, SP
001124	012737	006136	000014		MOV	#YESRT, 3#14
001132	012777	010606	010026		MOV	#POWDWN, 3DOWNVEC
001140	012777	000340	010022		MOV	#340, 3DOWNVEC+2
001146	012737	011006	000020		MOV	#.IOT, 3#20
001154	012700	000030			MOV	#30, RD
001160	012720	007542			MOV	#.TRAP, (0)+
001164	012720	000340			MOV	#340, (0)+
001170	012720	006140			MOV	#.EMT, (0)+
001174	012710	000340			MOV	#340, (0)
001200	012777	007210	007754		MOV	#FLTRR, 3FPVECT
001206	012777	000340	007750		MOV	#340, 3FPVECT+2
001214	005067	177560			CLR	ICNT
001220	005067	007756			CLR	LAD
						:SET TRACE TRAP VECTOR
						:SET UP VECTOR 20
						:SET RD TO VECTOR 30
						:SET EMT VECTOR
						:SET TRAP VECTOR
						:LOAD INTERRUPT VECTOR
						:LOCK UP PROCESSOR

```
*****
:TEST 1:      EXERCISE DIVF (DIVIDE FLOATING)
:              ALL INTERRUPTS ON
:              ROUNDING MODE
*****
```

001224	104400			SCOPE		
001226	012767	007400	177614	MOV	#007400,\$FPS	;SET IE BITS IN FORTRAN ANSWER
001234	005067	177612		CLR	\$FEC	;CLR FORTRAN FEC
001240	005067	177610		CLR	\$FPC	;CLR FORTRAN FPC
001244	005067	177572		CLR	FPS	;CLR FPU FPS BUFFER
001250	005067	177570		CLR	FEC	;CLR FPU FEC BUFFER
001254	005067	177566		CLR	FPC	;CLR FPU FPC BUFFER
001260	004767	005762		JSR	PC, RANDM2	;GET RANDOM INPUT DATA
001264	004467	005010		JSR	R4, \$POLSH	;ENTER POLISH MODE
001270	006312			\$P.2A		;PUSH 2 WORDS ON STACK (LONUM)
001272	006334			\$P.2B		;PUSH 2 WORDS ON STACK (HINUM)
001274	000000G			\$DVR		;ADDRESS OF FORTRAN DIVIDE
001276	006402			\$TST		;DETERMINE THE CONDITION CODES
001300	006346			\$POP2X		;POP 2 WORDS AND EXIT POLISH MODE
001302	016700	177542		MOV	\$FPS, RD	;DISPLAY FLOATING POINT STATUS
001306	170127	040000		LDFPS	#040000	;SET INTERRUPT DISABLE
001312	172467	177464		LDF	LONUM, ACO	;LOAD ACO WITH A RANDOM NUMBER
001316	172567	177470		LDF	HINUM, AC1	;LOAD AC1 WITH A RANDOM NUMBER
001322	172767	177504		LDF	ANS2, AC3	;LOAD AC3 WITH THE SUM
001326	170127	007400		LDFPS	#007400	;TURN INTERRUPTS ON
001332	012767	001340	007642	MOV	#+6, LAD	;RESET LOOP ADDRESS

001340	172600			LDF	ACO, AC2	;LOAD ACO INTO AC2
001342	174601			DIVF	AC1, AC2	;DIVIDE AC1 INTO AC2
001344	005767	177500		TST	\$FPS	;CHECK FOR FORTRAN FPS ERROR FLAG
001350	100412			BMI	ERR1	;BRANCH IF ERROR FLAG SET
001352	170267	177464		STFPS	FPS	;STORE FLOATING POINT STATUS
001356	026767	177460	177464	CMP	FPS, \$FPS	;CHECK FPS
001364	001427			BEQ	TST1	;BRANCH IF OK
001366	174267	177430		STF	AC2, ANS1	;SAVE FPU ANSWER
001372	104001			HLT+1		;FPS ERROR
001374	000444			BR	END1	;SKIP COMPARE
001376	170000			CFCC		;WAIT FOR FPU TO FINISH
001400	026767	177436	177442	CMP	FPS,\$FPS	;CHECK THE FLOATING POINT STATUS
001406	001402			BEQ	#+6	;BRANCH IF OK
001410	104377			HLT+377		;FPS ERROR
001412	000435			BR	END1	;SKIP TO END
001414	026767	177424	177430	CMP	FEC,\$FEC	;CHECK THE FLOATING EXCEPTION CODES
001422	001402			BEQ	#+6	;BRANCH IF OK
001424	104377			HLT+377		;FEC IS WRONG
001426	000427			BR	END1	;SKIP TO END
001430	026767	177412	177416	CMP	FPC,\$FPC	;CHECK FLOATING PC
001436	001402			BEQ	TST1	;BRANCH IF OK


```

001440 104377      HLT+377      ;WRONG ADDRESS IN FPC
001442 000421      BR          END1 ;SKIP TO END

001444 173702      TST1: CMPF     AC2,   AC3      ;COMPARE FPU ANSWER TO FORTRAN ANSWER
001446 170000      CFCC          ;COPY FLOATING CONDITION CODES
001450 001416      BEQ          END1      ;ANSWERS CHECK
;COMPENSATE FOR FORTRAN INACCURACIES.
001452 174267 177344 STF        AC2,   ANS1      ;SAVE FPU ANSWER
001456 162767 000001 177340 SUB      #1,   ANS1+2 ;DECREMENT FPU ANSWER
001464 005667 177332 SBC      ANS1
001470 173767 177326 CMPF     ANS1,   AC3      ;CHECK ANSWERS AGAIN
001474 170000      CFCC          ;COPY FLOATING CONDITION CODES
001476 001403      BEQ          END1      ;BRANCH IF OK
001500 174267 177316 STF        AC2,   ANS1      ;SAVE FPU ANSWER
001504 104000      HLT          ;FPU AND FORTRAN DISAGREE

001506 005067 177330 END1: CLR      FPS          ;CLR FPU FPS BUFFER
001512 104400      SCOPE

```

```

;*****
;TEST 2:      EXERCISE DIVD (DIVIDE DOUBLE PRECISION)
;              ALL INTERRUPTS ON
;              ROUNDING MODE
;*****

```

```

001514 012767 007600 177326 MOV      #007600,$FPS ;SET IE BITS IN FORTRAN ANSWER
001522 005067 177324 CLR      $FEC        ;CLR FORTRAN FEC
001526 005067 177322 CLR      $FPC        ;CLR FORTRAN FPC
001532 005067 177304 CLR      FPS          ;CLR FPU FPS BUFFER
001536 005067 177302 CLR      FEC          ;CLR FPU FEC BUFFER
001542 005067 177300 CLR      FPC          ;CLR FPU FPC BUFFER
001546 004767 005632 JSR      PC,      RANDM4 ;GET RANDOM INPUT DATA
001552 004467 004522 JSR      R4,      $POLSH ;ENTER POLISH MODE
001556 006302      $P.4A      ;PUSH 4 WORDS ON STACK (LONUM)
001560 006324      $P.4B      ;PUSH 4 WORDS ON STACK (HINUM)
001562 000000G      $DVD      ;ADDRESS OF FORTRAN DIVIDE
001564 006402      $TST      ;DETERMINE THE CONDITION CODES
001566 006360      $POP4X     ;POP 4 WORDS AND EXIT POLISH MODE

001570 016700 177254 MOV      $FPS,   R0      ;DISPLAY FLOATING POINT STATUS
001574 170127 040200 LDFPS   #040200 ;SET FID AND FD
001600 172467 177176 LDD      LONUM,   ACO      ;LOAD ACO WITH A RANDOM NUMBER
001604 172567 177202 LDD      HINUM,   AC1      ;LOAD AC1 WITH A RANDOM NUMBER
001610 172767 177216 LDD      ANS2,   AC3      ;LOAD AC3 WITH THE SUM
001614 170127 007600 LDFPS   #007600 ;TURN INTERRUPTS ON
001620 012767 001626 007354 MOV      #.+6,   LAD      ;RESET LOOP ADDRESS

```

```

;*****

```

```

001626 172600      RET2: LDD      ACO,   AC2      ;LOAD ACO INTO AC2
001630 174601      DIVD     AC1,   AC2      ;DIVIDE AC1 INTO AC2
001632 005767 177212 TST      $FPS      ;CHECK FOR FORTRAN FPS ERROR FLAG
001636 100412      BMI      ERR2      ;BRANCH IF ERROR FLAG SET
001640 170267 177176 STFPS   FPS          ;STORE FLOATING POINT STATUS
001644 026767 177172 177176 CMP      FPS,   $FPS      ;CHECK FPS
001652 001427      BEQ      TST2      ;BRANCH IF OK

```

001654	174267	177142	STD	AC2,	ANS1	:SAVE FPU ANSWER
001656	104001		HLT+1			:FPS ERROR
001662	000450		BR	END2		:SKIP TO END
001664	170000		ERR2:	CFCC		:WAIT FOR FPU TO FINISH
001666	026767	177150 177154	CMP	FPS,\$FPS		:CHECK THE FLOATING POINT STATUS
001674	001402		BEQ	.+6		:BRANCH IF OK
001676	104377		HLT+377			:FPS ERROR
001700	000441		BR	END2		:SKIP TO END
001702	026767	177136 177142	CMP	FEC,\$FEC		:CHECK THE FLOATING EXCEPTION CODES
001710	001402		BEQ	.+6		:BRANCH IF OK
001712	104377		HLT+377			:FEC IS WRONG
001714	000433		BR	END2		:SKIP TO END
001716	026767	177124 177130	CMP	FPC,\$FPC		:CHECK FLOATING PC
001724	001402		BEQ	TST2		:BRANCH IF OK
001726	104377		HLT+377			:WRONG ADDRESS IN FPC
001730	000425		BR	END2		:SKIP TO END
001732	173702		TST2:	CMPO	AC2, AC3	:COMPARE FPU ANSWER TO FORTRAN ANSWER
001734	170000		CFCC			:COPY FLOATING CONDITION CODES
001736	001422		BEQ	END2		:ANSWERS CHECK
				:COMPENSATE FOR	FORTRAN	INACCURACIES.
001740	174267	177056	STD	AC2,	ANS1	:SAVE FPU ANSWER
001744	162767	000001 177056	SUB	#1	ANS1+6	:DECREMENT FPU ANSWER
001752	005667	177050	SBC	ANS1+4		
001756	005667	177042	SBC	ANS1+2		
001762	005667	177034	SBC	ANS1		
001766	173767	177030	CMPO	ANS1,	AC3	:CHECK ANSWERS AGAIN
001772	170000		CFCC			:COPY FLOATING CONDITION CODES
001774	001403		BEQ	END2		:BRANCH IF OK
001776	174267	177020	STD	AC2,	ANS1	:SAVE FPU ANSWER
002002	104000		HLT			:FPU AND FORTRAN DISAGREE
002004	005067	177032	END2:	CLR	FPS	:CLR FPU FPS BUFFER
002010	104400		SCOPE			

```

:*****
:TEST 3:      EXERCISE DIVF (DIVIDE FLOATING)
:              OVERFLOW AND UNDERFLOW INTERRUPTS OFF.
:              ROUNDING MODE
:*****

```

002012	012767	004400 177030	MOV	#004400,\$FPS		:SET IE BITS IN FORTRAN ANSWER
002020	005067	177026	CLR	\$FEC		:CLR FORTRAN FEC
002024	005067	177024	CLR	\$FPC		:CLR FORTRAN FPC
002030	005067	177006	CLR	FPS		:CLR FPU FPS BUFFER
002034	005067	177004	CLR	FEC		:CLR FPU FEC BUFFER
002040	005067	177002	CLR	FPC		:CLR FPU FPC BUFFER
002044	004767	005176	JSR	PC,	RANDM2	:GET RANDOM INPUT DATA
002050	004467	004224	JSR	R4,	\$POLSH	:ENTER POLISH MODE
002054	006312		\$P, 2A			:PUSH 2 WORDS ON STACK (LONUM)
002056	006334		\$P, 2B			:PUSH 2 WORDS ON STACK (HINUM)
002060	000000		\$DVR			:ADDRESS OF FORTRAN DIVIDE
002062	006402		\$TST			:DETERMINE THE CONDITION CODES

002064	006346			\$POP2X		:POP 2 WORDS AND EXIT POLISH MODE	
002066	016700	176756		MOV	\$FPS	RD	:DISPLAY FLOATING POINT STATUS
002070	170127	040000		LDFPS	#040000		:SET FID
002076	172467	176700		LDF	LONUM,	AC0	:LOAD AC0 WITH A RANDOM NUMBER
002102	172567	176704		LDF	HINUM,	AC1	:LOAD AC1 WITH A RANDOM NUMBER
002106	172767	176720		LDF	ANS2,	AC3	:LOAD AC3 WITH THE SUM
002110	170127	004400		LDFPS	#004400		:TURN INTERRUPTS ON, EXCEPT OVERFLOW AND UNDERFLOW
002116	012767	002124	007056	MOV	#.+6,	LAD	:RESET LOOP ADDRESS

:*****

002124	172600			LDF	AC0,	AC2	:LOAD AC0 INTO AC2
002126	174601			DIVF	AC1,	AC2	:DIVIDE AC1 INTO AC2
002130	005767	176714		TST	\$FPS		:CHECK FOR FORTRAN FPS ERROR FLAG
002134	100412			BMI	ERR3		:BRANCH IF ERROR FLAG SET
002136	170267	176700		STFPS	FPS		:STORE FLOATING POINT STATUS
002142	026767	176674	176700	CMP	FPS,	\$FPS	:CHECK FPS
002150	001427			BEQ	TST3		:BRANCH IF OK
002152	174267	176644		STF	AC2,	ANS1	:SAVE FPU ANSWER
002156	104001			HLT+1			:FPS ERROR
002160	000450			BR	END3		:SKIP COMPARE
002162	170000			CFCC			:WAIT FOR FPU TO FINISH
002164	026767	176652	176656	CMP	FPS,\$FPS		:CHECK THE FLOATING POINT STATUS
002172	001402			BEQ	#+6		:BRANCH IF OK
002174	104377			HLT+377			:FPS ERROR
002176	000441			BR	END3		:SKIP TO END
002200	026767	176640	176644	CMP	FEC,\$FEC		:CHECK THE FLOATING EXCEPTION CODES
002206	001402			BEQ	#+6		:BRANCH IF OK
002210	104377			HLT+377			:FEC IS WRONG
002212	000433			BR	END3		:SKIP TO END
002214	026767	176626	176632	CMP	FPC,\$FPC		:CHECK FLOATING PC
002222	001402			BEQ	TST3		:BRANCH IF OK
002224	104377			HLT+377			:WRONG ADDRESS IN FPC
002226	000425			BR	END3		:SKIP TO END
002230	032767	000002	176612	TST3:	BIT	#2,	\$FPS
002236	001021			BNE	END3		:CHECK FOR OVERFLOW
002240	173702			CMPF	AC2,	AC3	:BRANCH IF OVERFLOW
002242	170000			CFCC			:COMPARE FPU ANSWER TO FORTRAN ANSWER
002244	001416			BEQ	END3		:COPY FLOATING CONDITION CODES
							:ANSWERS CHECK
							:INACCURACIES.
002246	174267	176550		STF	AC2,	ANS1	:SAVE FPU ANSWER
002252	162767	000001	176544	SUB	#1,	ANS1+2	:DECREMENT FPU ANSWER
002260	005667	176536		SBC	ANS1		
002264	173767	176532		CMPF	ANS1,	AC3	:CHECK ANSWERS AGAIN
002270	170000			CFCC			:COPY FLOATING CONDITION CODES
002272	001403			BEQ	END3		:BRANCH IF OK
002274	174267	176522		STF	AC2,	ANS1	:SAVE FPU ANSWER
002300	104000			HLT			:FPU AND FORTRAN DISAGREE
002302	005067	176534		END3:	CLR	FPS	:CLR FPU FPS BUFFER
002306	104400			SCOPE			

```

*****
:TEST 4:      EXERCISE DIVD (DIVIDE DOUBLE PRECISION)
:              OVERFLOW AND UNDERFLOW INTERRUPTS OFF
:              ROUNDING MODE
*****

002310 012767 004600 176532      MOV      #004600,$FPS      :SET IE BITS IN FORTRAN ANSWER
002316 005067 176530      CLR      $FEC      :CLR FORTRAN FEC
002322 005067 176526      CLR      $FPC      :CLR FORTRAN FPC
002326 005067 176510      CLR      FPS      :CLR FPU FPS BUFFER
002332 005067 176506      CLR      FEC      :CLR FPU FEC BUFFER
002336 005067 176504      CLR      FPC      :CLR FPU FPC BUFFER
002342 004767 005036      JSR      PC,      RANDM4      :GET RANDOM INPUT DATA
002346 004467 003726      JSR      R4,      $POLSH      :ENTER POLISH MODE
002352 006302      $P.4A      :PUSH 4 WORDS ON STACK (LONUM)
002354 006324      $P.4B      :PUSH 4 WORDS ON STACK (HINUM)
002356 000000G      $DVD      :ADDRESS OF FORTRAN DIVIDE
002360 006402      $TST      :DETERMINE THE CONDITION CODES
002362 006360      $POP4X      :POP 4 WORDS AND EXIT POLISH MODE

002364 016700 176460      MOV      $FPS,      R0      :DISPLAY FLOATING POINT STATUS
002370 170127 040200      LD FPS      #040200      :SET FID AND FD
002374 172467 176402      LDD      LONUM,      ACO      :LOAD ACO WITH A RANDOM NUMBER
002400 172567 176406      LDD      HINUM,      AC1      :LOAD AC1 WITH A RANDOM NUMBER
002404 172767 176422      LDD      ANS2,      AC3      :LOAD AC3 WITH THE SUM
002410 170127 004600      LD FPS      #004600      :TURN INTERRUPTS ON, EXCEPT OVER AND UNDERFLOW
002414 012767 002422 006560      MOV      #.+6,      LAD      :RESET LOOP ADDRESS

*****

002422 172600      LDD      ACO,      AC2      :LOAD ACO INTO AC2
002424 174601      DIVD      AC1,      AC2      :DIVIDE AC1 INTO AC2
002426 005767 176416      TST      $FPS      :CHECK FOR FORTRAN FPS ERROR FLAG
002432 100412      BMI      ERR4      :BRANCH IF ERROR FLAG SET
002434 170267 176402      ST FPS      :STORE FLOATING POINT STATUS
002440 026767 176376 176402      CMP      FPS,      $FPS      :CHECK FPS
002446 001427      BEQ      TST4      :BRANCH IF OK
002450 174267 176346      STD      AC2,      ANS1      :SAVE FPU ANSWER
002454 104001      HLT+1      :FPS ERROR
002456 000454      BR      END4      :SKIP COMPARE

002460 170000      CFCC      :WAIT FOR FPU TO FINISH
002462 026767 176354 176360      CMP      FPS,$FPS      :CHECK THE FLOATING POINT STATUS
002470 001402      BEQ      .+6      :BRANCH IF OK
002472 104377      HLT+377      :FPS ERROR
002474 000445      BR      END4      :SKIP TO END

002476 026767 176342 176346      CMP      FEC,$FEC      :CHECK THE FLOATING EXCEPTION CODES
002504 001402      BEQ      .+6      :BRANCH IF OK
002506 104377      HLT+377      :FEC IS WRONG
002510 000437      BR      END4      :SKIP TO END

002512 026767 176330 176334      CMP      FPC,$FPC      :CHECK FLOATING PC
002520 001402      BEQ      TST4      :BRANCH IF OK
002522 104377      HLT+377      :WRONG ADDRESS IN FPC

```



```

002524 000431          BR      END4          :SKIP TO END
002526 032767 000002 176314 TST4:  BIT      #2,      $FPS      :CHECK FOR OVERFLOW
002528 001025          BNE      END4          :BRANCH IF OVERFLOW
002530 173702          CMPD     AC2,      AC3      :COMPARE FPU ANSWER TO FORTRAN ANSWER
002532 170000          CFCC          :COPY FLOATING CONDITION CODES
002534 001422          BEQ      END4          :ANSWERS CHECK
                                :COMPENSATE FOR FORTRAN INACCURACIES.
002536 174267 176252  STD      AC2,      ANS1      :SAVE FPU ANSWER
002538 162767 000001 176252  SUB      #1,      ANS1+6    :DECREMENT FPU ANSWER
002540 005667 176244  SBC      ANS1+4
002542 005667 176236  SBC      ANS1+2
002544 005667 176230  SBC      ANS1
002546 173767 176224  CMPD     ANS1,      AC3      :CHECK ANSWERS AGAIN
002548 170000          CFCC          :COPY FLOATING CONDITION CODES
002550 001403          BEQ      END4          :BRANCH IF OK
002552 174267 176214  STD      AC2,      ANS1      :SAVE FPU ANSWER
002554 104000          HLT          :FPU AND FORTRAN DISAGREE

002610 005067 176226          END4:  CLR      FPS          :CLR FPU FPS BUFFER
002612 104400          SCOPE

```

```

:*****
:TEST 5:      EXERCISE DIVF (DIVIDE FLOATING)
:              ALL INTERRUPTS ON
:              TRUNCATE MODE
:*****

```

```

002616 012767 007440 176224  MOV      #007440,$FPS      :SET IE BITS IN FORTRAN ANSWER
002618 005067 176222  CLR      $FEC          :CLR FORTRAN FEC
002620 005067 176220  CLR      $FPC          :CLR FORTRAN FPC
002622 005067 176202  CLR      FPS          :CLR FPU FPS BUFFER
002624 005067 176200  CLR      FEC          :CLR FPU FEC BUFFER
002626 005067 176176  CLR      FPC          :CLR FPU FPC BUFFER
002628 004767 004372  JSR      PC,      RANDM2    :GET RANDOM INPUT DATA
002630 004467 003420  JSR      R4,      $POLSH    :ENTER POLISH MODE
002632 006312  SP.2A    :PUSH 2 WORDS ON STACK (LONUM)
002634 006334  SP.2B    :PUSH 2 WORDS ON STACK (HINUM)
002636 000000G  $DVR      :ADDRESS OF FORTRAN DIVIDE
002638 006402  $TST      :DETERMINE THE CONDITION CODES
002640 006346  $POP2X    :POP 2 WORDS AND EXIT POLISH MODE

002672 016700 176152  MOV      $FPS,      RD      :DISPLAY FLOATING POINT STATUS
002674 170127 040000  LDFPS     #040000    :SET FID
002676 172467 176074  LDF      LONUM,      AC0      :LOAD AC0 WITH A RANDOM NUMBER
002678 172567 176100  LDF      HINUM,      AC1      :LOAD AC1 WITH A RANDOM NUMBER
002680 172767 176114  LDF      ANS2,      AC3      :LOAD AC3 WITH THE SUM
002682 170127 007440  LDFPS     #007440    :TURN INTERRUPTS ON
002684 012767 002730 006252  MOV      #.+6,      LAD      :RESET LOOP ADDRESS

```

```

:*****

```

```

002730 172600          LDF      ACC,      AC2      :LOAD ACC INTO AC2
002732 174601          DIVF     AC1,      AC2      :DIVIDE AC1 INTO AC2
002734 003767 176110  TST      $FPS          :CHECK FOR FORTRAN FPS ERROR FLAG
002736 100412          BMI      ERR5          :BRANCH IF ERROR FLAG SET

```

002742	170267	176074		STFPS	FPS		:STORE FLOATING POINT STATUS
002746	026767	176070	176074	CMP	FPS,	\$FPS	:CHECK FPS
002754	001427			BEQ	TST5		:BRANCH IF OK
002756	174267	176040		STF	AC2,	ANS1	:SAVE FPU ANSWER
002762	104001			HLT+1			:FPS ERROR
002764	000455			BR	ENDS		:SKIP COMPARE
002766	170000						
002770	026767	176046	176052	ERRS: CFCC	FPS,\$FPS		:WAIT FOR FPU TO FINISH
002776	001402			CMP	.+6		:CHECK THE FLOATING POINT STATUS
003000	104377			BEQ			:BRANCH IF OK
003002	000446			HLT+377			:FPS ERROR
				BR	ENDS		:SKIP TO END
003004	026767	176034	176040	CMP	FEC,\$FEC		:CHECK THE FLOATING EXCEPTION CODES
003012	001402			BEQ	.+6		:BRANCH IF OK
003014	104377			HLT+377			:FEC IS WRONG
003016	000440			BR	ENDS		:SKIP TO END
003020	026767	176022	176026	CMP	FPC,\$FPC		:CHECK FLOATING PC
003026	001402			BEQ	TST5		:BRANCH IF OK
003030	104377			HLT+377			:WRONG ADDRESS IN FPC
003032	000432			BR	ENDS		:SKIP TO END
003034	173702			TST5: CMPF	AC2,	AC3	:COMPARE FPU ANSWER TO FORTRAN ANSWER
003036	170000			CFCC			:COPY FLOATING CONDITION CODES
003040	001427			BEQ	ENDS		:ANSWERS CHECK
				:COMPENSATE FOR FORTRAN			:INACCURACIES.
003042	174267	175754		STF	AC2,	ANS1	:SAVE FPU ANSWER
003046	062767	000001	175750	ADD	#1,	ANS1+2	:INCREMENT FPU ANSWER
003054	005567	175742		ADC	ANS1,		
003060	173767	175736		CMPF	ANS1,	AC3	:CHECK ANSWERS AGAIN
003064	170000			CFCC			:COPY FLOATING CONDITION CODES
003066	001414			BEQ	ENDS		:BRANCH IF OK
003070	162767	000002	175726	SUB	#2,	ANS1+2	:DECREMENT FPU ANSWER
003076	005667	175720		SBC	ANS1,		
003102	173767	175714		CMPF	ANS1,	AC3	:CHECK ANSWERS AGAIN
003106	170000			CFCC			:COPY FLOATING CONDITION CODES
003110	001403			BEQ	ENDS		:BRANCH IF OK
003112	174267	175704		STF	AC2,	ANS1	:SAVE FPU ANSWER
003116	104000			HLT			:FPU AND FORTRAN DISAGREE
003120	005067	175716		ENDS: CLR	FPS		:CLR FPU FPS BUFFER
003124	104400			SCOPE			

```

*****
:TEST 6:      EXERCISE DIVD (DIVIDE DOUBLE PRECISION)
:              ALL INTERRUPTS ON
:              TRUNCATE MODE
*****

```

003126	012767	007640	175714	MOV	#007640,\$FPS		:SET IE BITS IN FORTRAN ANSWER
003134	005067	175712		CLR	\$FEC		:CLR FORTRAN FEC
003140	005067	175710		CLR	\$FPC		:CLR FORTRAN FPC
003144	005067	175672		CLR	FPS		:CLR FPU FPS BUFFER
003150	005067	175670		CLR	FEC		:CLR FPU FEC BUFFER
003154	005067	175666		CLR	FPC		:CLR FPU FPC BUFFER

003160	004767	004220		JSR	PC.	RANDM4	:GET RANDOM INPUT DATA
003164	004467	003110		JSR	R4.	\$POLSH	:ENTER POLISH MODE
003170	006302			\$P.4A			:PUSH 4 WORDS ON STACK (LONUM)
003172	006324			\$P.4B			:PUSH 4 WORDS ON STACK (HINUM)
003174	000000G			\$DVD			:ADDRESS OF FORTRAN DIVIDE
003176	006402			\$TST			:DETERMINE THE CONDITION CODES
003200	006360			\$POP4X			:POP 4 WORDS AND EXIT POLISH MODE
003202	016700	175642		MOV	\$FPS.	RD	:DISPLAY FLOATING POINT STATUS
003206	170127	040200		LDFPS	#040200		:SET FID AND FD
003212	172467	175564		LDD	LONUM.	ACD	:LOAD ACD WITH A RANDOM NUMBER
003216	172567	175570		LDD	HINUM.	AC1	:LOAD AC1 WITH A RANDOM NUMBER
003222	172767	175604		LDD	ANS2.	AC3	:LOAD AC3 WITH THE SUM
003226	170127	007640		LDFPS	#007640		:TURN INTERRUPTS ON
003232	012767	003240	005742	MOV	#+6.	LAD	:RESET LOOP ADDRESS

;*****

003240	172600			LDD	ACD.	AC2	:LOAD ACD INTO AC2
003242	174601			DIVD	AC1.	AC2	:DIVIDE AC1 INTO AC2
003244	005767	175600		TST	\$FPS		:CHECK FOR FORTRAN FPS ERROR FLAG
003250	100412			BMI	ERR6		:BRANCH IF ERROR FLAG SET
003252	170267	175564		STFPS	FPS		:STORE FLOATING POINT STATUS
003256	026767	175560	175564	CMP	FPS.	\$FPS	:CHECK FPS
003264	001427			BEG	TST6		:BRANCH IF OK
003266	174267	175530		STD	AC2.	ANS1	:SAVE FPU ANSWER
003272	104001			HLT+1			:FPS ERROR
003274	000465			BR	END6		:SKIP TO END
003276	170000			CFCC			:WAIT FOR FPU TO FINISH
003300	026767	175536	175542	CMP	FPS.\$FPS		:CHECK THE FLOATING POINT STATUS
003306	001402			BEG	#+6		:BRANCH IF OK
003310	104377			HLT+377			:FPS ERROR
003312	000456			BR	END6		:SKIP TO END
003314	026767	175524	175530	CMP	FEC.\$FEC		:CHECK THE FLOATING EXCEPTION CODES
003322	001402			BEG	#+6		:BRANCH IF OK
003324	104377			HLT+377			:FEC IS WRONG
003326	000450			BR	END6		:SKIP TO END
003330	026767	175512	175516	CMP	FPC.\$FPC		:CHECK FLOATING PC
003336	001402			BEG	TST6		:BRANCH IF OK
003340	104377			HLT+377			:WRONG ADDRESS IN FPC
003342	000442			BR	END6		:SKIP TO END
003344	173702			TST6:	AC2.	AC3	:COMPARE FPU ANSWER TO FORTRAN ANSWER
003346	170000			CFCC			:COPY FLOATING CONDITION CODES
003350	001437			BEG	END6		:ANSWERS CHECK
					:COMPENSATE FOR FORTRAN		:INACCURACIES.
003352	174267	175444		STD	AC2.	ANS1	:SAVE FPU ANSWER
003356	062767	000001	175444	ADD	#1.	ANS1+6	:INCREMENT FPU ANSWER
003364	005567	175436		ADC	ANS1+4		
003370	005567	175430		ADC	ANS1+2		
003374	005567	175422		ADC	ANS1		
003400	173767	175416		CMPD	ANS1.	AC3	:CHECK ANSWERS AGAIN
003404	170000			CFCC			:COPY FLOATING CONDITION CODES

```

003406 001420      BEQ      END6      ;BRANCH IF OK
003410 162767 000002 175412      SUB      #2      ;DECREMENT FPU ANSWER
003416 005667 175404      SBC      ANS1+4
003422 005667 175376      SBC      ANS1+2
003426 005667 175370      SBC      ANS1
003432 173767 175364      CMPD     ANS1, AC3      ;CHECK ANSWERS AGAIN
003436 170000      CFCC      ;COPY FLOATING CONDITION CODES
003440 001403      BEQ      END6      ;BRANCH IF OK
003444 174267 175354      STD      AC2, ANS1      ;SAVE FPU ANSWER
003446 104000      HLT              ;FPU AND FORTRAN DISAGREE

003450 005067 175366      END6:   CLR      FPS      ;CLR FPU FPS BUFFER
003454 104400      SCOPE

```

```

;*****
;TEST 7:      EXERCISE DIVF (DIVIDE FLOATING)
;              OVERFLOW AND UNDERFLOW INTERRUPTS OFF.
;              TRUNCATE MODE
;*****

```

```

003456 012767 004440 175364      MOV      #004440,$FPS      ;SET IE BITS IN FORTRAN ANSWER
003464 005067 175362      CLR      $FEC      ;CLR FORTRAN FEC
003470 005067 175360      CLR      $FPC      ;CLR FORTRAN FPC
003474 005067 175342      CLR      FPS      ;CLR FPU FPS BUFFER
003500 005067 175340      CLR      FEC      ;CLR FPU FEC BUFFER
003504 005067 175336      CLR      FPC      ;CLR FPU FPC BUFFER
003510 004767 003532      JSR      PC, RANDM2      ;GET RANDOM INPUT DATA
003514 004467 002560      JSR      R4, $POLSH      ;ENTER POLISH MODE
003520 006312      $P.2A      ;PUSH 2 WORDS ON STACK (LONUM)
003522 006334      $P.2B      ;PUSH 2 WORDS ON STACK (HINUM)
003524 000000G      $DVR      ;ADDRESS OF FORTRAN DIVIDE
003526 006402      $TST      ;DETERMINE THE CONDITION CODES
003530 006346      $POP2X      ;POP 2 WORDS AND EXIT POLISH MODE

003532 016700 175312      MOV      $FPS, R0      ;DISPLAY FLOATING POINT STATUS
003536 170127 040000      LDFPS     #040000      ;SET FID
003542 172467 175234      LDF      LONUM, AC0      ;LOAD AC0 WITH A RANDOM NUMBER
003546 172567 175240      LDF      HINUM, AC1      ;LOAD AC1 WITH A RANDOM NUMBER
003552 172767 175254      LDF      ANS2, AC3      ;LOAD AC3 WITH THE SUM
003556 170127 004440      LDFPS     #004440      ;TURN INTERRUPTS ON, EXCEPT OVERFLOW AND UNDERFLOW
003562 012767 003570 005412      MOV      #.+6, LAD      ;RESET LOOP ADDRESS

```

```

;*****

```

```

003570 172600      LDF      AC0, AC2      ;LOAD AC0 INTO AC2
003572 174601      DIVF     AC1, AC2      ;DIVIDE AC1 INTO AC2
003574 005767 175250      TST      $FPS      ;CHECK FOR FORTRAN FPS ERROR FLAG
003600 100412      BMI      ERR7      ;BRANCH IF ERROR FLAG SET
003602 170267 175234      STFPS     FPS      ;STORE FLOATING POINT STATUS
003606 026767 175230 175234      CMP      FPS, $FPS      ;CHECK FPS
003614 001427      BEQ      TST7      ;BRANCH IF OK
003616 174267 175200      STF      AC2, ANS1      ;SAVE FPU ANSWER
003622 104001      HLT+1      ;FPS ERROR
003624 000461      BR        END7      ;SKIP COMPARE

003626 170000      ERR7:   CFCC      ;WAIT FOR FPU TO FINISH

```



```

003630 026767 175206 175212      CMP      FPS,$FPS      :CHECK THE FLOATING POINT STATUS
003636 001402      BEQ      .+6      :BRANCH IF OK
003640 104377      HLT+377      :FPS ERROR
003642 000452      BR      END7      :SKIP TO END

003644 026767 175174 175200      CMP      FEC,$FEC      :CHECK THE FLOATING EXCEPTION CODES
003652 001402      BEQ      .+6      :BRANCH IF OK
003654 104377      HLT+377      :FEC IS WRONG
003656 000444      BR      END7      :SKIP TO END

003660 026767 175162 175166      CMP      FPC,$FPC      :CHECK FLOATING PC
003666 001402      BEQ      TST7      :BRANCH IF OK
003670 104377      HLT+377      :WRONG ADDRESS IN FPC
003672 000436      BR      END7      :SKIP TO END

003674 032767 000002 175146 TST7: BIT      #2,$FPS      :CHECK FOR OVERFLOW
003702 001032      BNE      END7      :BRANCH IF OVERFLOW
003704 173702      CMPF      AC2, AC3      :COMPARE FPU ANSWER TO FORTRAN ANSWER
003706 170000      CFCC      :COPY FLOATING CONDITION CODES
003710 001427      BEQ      END7      :ANSWERS CHECK
:COMPENSATE FOR FORTRAN INACCURACIES.
003712 174267 175104      STF      AC2, ANS1      :SAVE FPU ANSWER
003716 062767 000001 175100      ADD      #1, ANS1+2      :INCREMENT FPU ANSWER
003724 005567 175072      ADC      ANS1, ANS1+2
003730 173767 175066      CMPF      ANS1, AC3      :CHECK ANSWERS AGAIN
003734 170000      CFCC      :COPY FLOATING CONDITION CODES
003736 001414      BEQ      END7      :BRANCH IF OK
003740 162767 000002 175056      SUB      #2, ANS1+2      :DECREMENT FPU ANSWER
003746 005667 175050      SBC      ANS1, ANS1+2
003752 173767 175044      CMPF      ANS1, AC3      :CHECK ANSWERS AGAIN
003756 170000      CFCC      :COPY FLOATING CONDITION CODES
003760 001403      BEQ      END7      :BRANCH IF OK
003762 174267 175034      STF      AC2, ANS1      :SAVE FPU ANSWER
003766 104000      HLT      :FPU AND FORTRAN DISAGREE

003770 005067 175046      END7: CLR      FPS      :CLR FPU FPS BUFFER
003774 104400      SCOPE

```

```

:*****
:TEST 10:      EXERCISE DIVD (DIVIDE DOUBLE PRECISION)
:              OVERFLOW AND UNDERFLOW INTERRUPTS OFF
:              TRUNCATE MODE
:*****

```

```

003776 012767 004640 175044      MOV      #004640,$FPS      :SET IE BITS IN FORTRAN ANSWER
004004 005067 175042      CLR      $FEC      :CLR FORTRAN FEC
004010 005067 175040      CLR      $FPC      :CLR FORTRAN FPC
004014 005067 175022      CLR      FPS      :CLR FPU FPS BUFFER
004020 005067 175020      CLR      FEC      :CLR FPU FEC BUFFER
004024 005067 175016      CLR      FPC      :CLR FPU FPC BUFFER
004030 004767 003350      JSR      PC, RANDM4      :GET RANDOM INPUT DATA
004034 004467 002240      JSR      R4, $POLSH      :ENTER POLISH MODE
004040 006302      $P.4A      :PUSH 4 WORDS ON STACK (LONUM)
004042 006324      $P.4B      :PUSH 4 WORDS ON STACK (HINUM)
004044 000000G      $DIVD      :ADDRESS OF FORTRAN DIVIDE
004046 006402      $TST      :DETERMINE THE CONDITION CODES

```

004050	006360			\$POP4X		;POP 4 WORDS AND EXIT POLISH MODE
004052	016700	174772		MOV	\$FPS,	RD
004056	170127	040200		LDFPS	#040200	
004062	172467	174714		LDD	LONUM,	AC0
004066	172567	174720		LDD	HINUM,	AC1
004072	172767	174734		LDD	ANS2,	AC3
004076	170127	004640		LDFPS	#004640	
004102	012767	004110	005072	MOV	#+6,	LAD

;*****

004110	172600				LDD	AC0,	AC2	;LOAD AC0 INTO AC2
004112	174601			RET10:	DIVD	AC1,	AC2	;DIVIDE AC1 INTO AC2
004114	005767	174730			TST	\$FPS		;CHECK FOR FORTRAN FPS ERROR FLAG
004120	100412				BMI	ERR10		;BRANCH IF ERROR FLAG SET
004122	170267	174714			STFPS	FPS		;STORE FLOATING POINT STATUS
004126	026767	174710	174714		CMP	FPS,	\$FPS	;CHECK FPS
004134	001427				BEQ	TST10		;BRANCH IF OK
004136	174267	174660			STD	AC2,	ANS1	;SAVE FPU ANSWER
004142	104001				HLT+1			;FPS ERROR
004144	000471				BR	END10		;SKIP COMPARE
004146	170000				CFCC			;WAIT FOR FPU TO FINISH
004150	026767	174666	174672	ERR10:	CMP	FPS,\$FPS		;CHECK THE FLOATING POINT STATUS
004156	001402				BEQ	#+6		;BRANCH IF OK
004160	104377				HLT+377			;FPS ERROR
004162	000462				BR	END10		;SKIP TO END
004164	026767	174654	174660		CMP	FEC,\$FEC		;CHECK THE FLOATING EXCEPTION CODES
004172	001402				BEQ	#+6		;BRANCH IF OK
004174	104377				HLT+377			;FEC IS WRONG
004176	000454				BR	END10		;SKIP TO END
004200	026767	174642	174646		CMP	FPC,\$FPC		;CHECK FLOATING PC
004206	001450				BEQ	END10		;BRANCH IF OK
004210	104377				HLT+377			;WRONG ADDRESS IN FPC
004212	000446				BR	END10		;SKIP TO END
004214	032767	000002	174626	TST10:	BIT	#2,	\$FPS	;CHECK FOR OVERFLOW
004222	001042				BNE	END10		;BRANCH IF OVERFLOW
004224	173702				CMPD	AC2,	AC3	;COMPARE FPU ANSWER TO FORTRAN ANSWER
004226	170000				CFCC			;COPY FLOATING CONDITION CODES
004230	001437				BEQ	END10		;ANSWERS CHECK
004232	174267	174564			STD	AC2,	ANS1	;COMPENSATE FOR FORTRAN INACCURACIES.
004236	062767	000001	174564		ADD	#1,	ANS1+6	;SAVE FPU ANSWER
004244	005567	174556			ADC	ANS1+4		;INCREMENT FPU ANSWER
004250	005567	174550			ADC	ANS1+2		
004254	005567	174542			ADC	ANS1		
004260	173767	174536			CMPD	ANS1,	AC3	;CHECK ANSWERS AGAIN
004264	170000				CFCC			;COPY FLOATING CONDITION CODES
004266	001420				BEQ	END10		;BRANCH IF OK
004270	162767	000002	174532		SUB	#2,	ANS1+6	;DECREMENT FPU ANSWER
004276	005667	174524			SBC	ANS1+4		
004302	005667	174516			SBC	ANS1+2		

004306	005667	174510	SBC	ANS1		
004312	173767	174504	CMPO	ANS1,	AC3	;CHECK ANSWERS AGAIN
004316	170000		CFCC			;COPY FLOATING CONDITION CODES
004320	001403		BEO	END10		;BRANCH IF OK
004322	174267	174474	STD	AC2,	ANS1	;SAVE FPU ANSWER
004326	104000		HLT			;FPU AND FORTRAN DISAGREE
004330	005067	174506	END10:	CLR	FPS	;CLR FPU FPS BUFFER
004334	104400			SCOPE		

;TEST 11: EXERCISE DIVF (DIVIDE FLOATING)
; INTERUPT DISABLE SET
; ROUNDING MODE

004336	012767	047400	174504	MOV	#047400,\$FPS	;SET IE BITS IN FORTRAN ANSWER
004344	005067	174502		CLR	\$FEC	;CLR FORTRAN FEC
004350	005067	174500		CLR	\$FPC	;CLR FORTRAN FPC
004354	005067	174462		CLR	FPS	;CLR FPU FPS BUFFER
004360	005067	174460		CLR	FEC	;CLR FPU FEC BUFFER
004364	005067	174456		CLR	FPC	;CLR FPU FPC BUFFER
004370	004767	002652		JSR	PC,	RANDM2
004374	004467	001700		JSR	R4,	\$POLSH
004400	006312			\$P.2A		;PUSH 2 WORDS ON STACK (LONUM)
004402	006334			\$P.2B		;PUSH 2 WORDS ON STACK (HINUM)
004404	000000G			\$DVR		;ADDRESS OF FORTRAN DIVIDE
004406	006402			\$TST		;DETERMINE THE CONDITION CODES
004410	006346			\$POP2X		;POP 2 WORDS AND EXIT POLISH MODE
004412	016700	174432		MOV	\$FPS,	RD
004416	170127	040000		LDFPS	#040000	;DISPLAY FLOATING POINT STATUS
004422	172467	174354		LDF	LONUM,	AC0
004426	172567	174360		LDF	HINUM,	AC1
004432	172767	174374		LDF	ANS2,	AC3
004436	170127	047400		LDFPS	#047400	;SET INTERUPT DISABLE AND INTERUPT BITS
004442	012767	004450	004532	MOV	#+6,	LAD
						;RESET LOOP ADDRESS

004450	172600			LDF	AC0,	AC2	;LOAD AC0 INTO AC2
004452	174601			DIVF	AC1,	AC2	;DIVIDE AC1 INTO AC2
004454	170267	174362		STFPS	FPS		;STORE FLOATING POINT STATUS
004460	005767	174364		TST	\$FPS		;CHECK FOR ERROR FLAG
004464	100410			BMI	ERR11		;BRANCH IF ERROR FLAG SET
004466	026767	174350	174354	CMF	FPS,	\$FPS	;CHECK FPS
004474	001430			BEO	TST11		;BRANCH IF OK
004476	174267	174320		STF	AC2,	ANS1	;SAVE FPU ANSWER
004502	104001			HLT+1			;FPS ERROR
004504	000451			BR	END11		;SKIP COMPARE
004506	170367	174332		STST	FEC		;STORE FEC AND FPC
004512	026767	174324	174330	CMF	FPS,	\$FPS	;CHECK THE FLOATING POINT STATUS
004520	001402			BEO	#+6		;BRANCH IF OK
004522	104377			HLT+377			;FPS ERROR
004524	000441			BR	END11		;SKIP TO END

004526	026767	174312	174316	CMP	FEC,\$FEC		;CHECK THE FLOATING EXCEPTION CODES
004534	001402			BEQ	+.6		;BRANCH IF OK
004536	104377			HLT+377			;FEC IS WRONG
004540	000433			BR	END11		;SKIP TO END
004542	026767	174300	174304	CMP	FPC,\$FPC		;CHECK FLOATING PC
004550	001402			BEQ	TST11		;BRANCH IF OK
004552	104377			HLT+377			;WRONG ADDRESS IN FPC
004554	000425			BR	END11		;SKIP TO END
004556	032767	000002	174264	TST11:	BIT	#2,\$FPS	;CHECK FOR OVERFLOW
004564	001021				BNE	END11	;BRANCH IF OVERFLOW
004566	173702				CMPF	AC2, AC3	;COMPARE FPU ANSWER TO FORTRAN ANSWER
004570	170000				CFCC		;COPY FLOATING CONDITION CODES
004572	001416				BEQ	END11	;ANSWERS CHECK
					;COMPENSATE FOR FORTRAN INACCURACIES.		
004574	174267	174222			STF	AC2, ANS1	;SAVE FPU ANSWER
004600	162767	000001	174216		SUB	#1, ANS1+2	;DECREMENT FPU ANSWER
004606	005667	174210			SBC	ANS1	
004612	173767	174204			CMPF	ANS1, AC3	;CHECK ANSWERS AGAIN
004616	170000				CFCC		;COPY FLOATING CONDITION CODES
004620	001403				BEQ	END11	;BRANCH IF OK
004622	174267	174174			STF	AC2, ANS1	;SAVE FPU ANSWER
004626	104000				HLT		;FPU AND FORTRAN DISAGREE
004630	005067	174206		END11:	CLR	FPS	;CLR FPU FPS BUFFER
004634	104400				SCOPE		

:TEST 12: EXERCISE DIVD (DIVIDE DOUBLE PRECISION)
: INTERUPT DISABLE SET
: ROUNDING MODE
:*****

004636	012767	047600	174204	MOV	#047600,\$FPS	:SET FID AND IE BITS IN FORTRAN ANSWER
004644	005067	174202		CLR	\$FEC	:CLR FORTRAN FEC
004650	005067	174200		CLR	\$FPC	:CLR FORTRAN FPC
004654	005067	174162		CLR	FPS	:CLR FPU FPS BUFFER
004660	005067	174160		CLR	FEC	:CLR FPU FEC BUFFER
004664	005067	174156		CLR	FPC	:CLR FPU FPC BUFFER
004670	004767	002510		JSR	PC, RANDM4	:GET RANDOM INPUT DATA
004674	004467	001400		JSR	R4, \$POLSH	:ENTER POLISH MODE
004700	006302			\$P.4A		:PUSH 4 WORDS ON STACK (LONUM)
004702	006324			\$P.4B		:PUSH 4 WORDS ON STACK (MINUM)
004704	000000G			\$DVD		:ADDRESS OF FORTRAN DIVIDE
004706	006402			\$TST		:DETERMINE THE CONDITION CODES
004710	006360			\$POP4X		:POP 4 WORDS AND EXIT POLISH MODE
004712	016700	174132		MOV	\$FPS, R0	:DISPLAY FLOATING POINT STATUS
004716	170127	040200		LDFPS	#040200	:SET FID AND FD
004722	172467	174054		LDD	LONUM, AC0	:LOAD AC0 WITH A RANDOM NUMBER
004726	172567	174060		LDD	HINUM, AC1	:LOAD AC1 WITH A RANDOM NUMBER
004732	172767	174074		LDD	ANS2, AC3	:LOAD AC3 WITH THE SUM
004736	170127	047600		LDFPS	#047600	:SET INTERUPT DISABLE AND INTERUPT BITS
004742	012767	004750	004232	MOV	#+6, LAD	:RESET LOOP ADDRESS

004750	172600			LDD	AC0, AC2	:LOAD AC0 INTO AC2
004752	174601			DIVD	AC1, AC2	:DIVIDE AC1 INTO AC2
004754	170267	174062		STFPS	FPS	:STORE FLOATING POINT STATUS
004760	005767	174064		TST	\$FPS	:CHECK FOR ERROR FLAG
004764	100410			BMI	ERR12	:BRANCH IF ERROR FLAG SET
004766	026767	174050	174054	CMP	FPS, \$FPS	:CHECK FPS
004774	001430			BEQ	TST12	:BRANCH IF OK
004776	174267	174020		STD	AC2, ANS1	:SAVE FPU ANSWER
005002	104001			HLT+1		:FPS ERROR
005004	000455			BR	END12	:SKIP COMPARE
005006	170367	174032		STST	FEC	:STORE FEC AND FPC
005012	026767	174024	174030	CMP	FPS, \$FPS	:CHECK THE FLOATING POINT STATUS
005020	001402			BEQ	#+6	:BRANCH IF OK
005022	104377			HLT+377		:FPS ERROR
005024	000445			BR	END12	:SKIP TO END
005026	026767	174012	174016	CMP	FEC,\$FEC	:CHECK THE FLOATING EXCEPTION CODES
005034	001402			BEQ	#+6	:BRANCH IF OK
005036	104377			HLT+377		:FEC IS WRONG
005040	000437			BR	END12	:SKIP TO END
005042	026767	174000	174004	CMP	FPC,\$FPC	:CHECK FLOATING PC
005050	001402			BEQ	TST12	:BRANCH IF OK
005052	104377			HLT+377		:WRONG ADDRESS IN FPC

```

005054 000431          BR      END12          ;SKIP TO END
005056 032767 000002 173764 TST12: BIT      #2,      $FPS      ;CHECK FOR OVERFLOW
005064 001025          BNE      END12          ;BRANCH IF OVERFLOW
005066 173702          CMPD     AC2,      AC3      ;COMPARE FPU ANSWER TO FORTRAN ANSWER
005070 170000          CFCC          ;COPY FLOATING CONDITION CODES
005072 001422          BEQ      END12          ;ANSWERS CHECK
          ;COMPENSATE FOR FORTRAN INACCURACIES.
005074 174267 173722          STD      AC2,      ANS1      ;SAVE FPU ANSWER
005100 162767 000001 173722          SUB      #1,      ANS1+6  ;DECREMENT FPU ANSWER
005106 005667 173714          SBC      ANS1+4
005112 005667 173706          SBC      ANS1+2
005116 005667 173700          SBC      ANS1
005122 173767 173674          CMPD     ANS1,      AC3      ;CHECK ANSWERS AGAIN
005126 170000          CFCC          ;COPY FLOATING CONDITION CODES
005130 001403          BEQ      END12          ;BRANCH IF OK
005132 174267 173664          STD      AC2,      ANS1      ;SAVE FPU ANSWER
005136 104000          HLT          ;FPU AND FORTRAN DISAGREE

005140 005067 173676          END12: CLR      FPS          ;CLR FPU FPS BUFFER
005144 104400          SCOPE

```

```

;*****
;TEST 13:      EXERCISE DIVF (DIVIDE FLOATING)
;              INTERRUPT DISABLE SET
;              TRUNCATE MODE
;*****

```

```

005146 012767 047440 173674          MOV      #047440,$FPS      ;SET FID AND IE BITS IN FORTRAN ANSWER
005154 005067 173672          CLR      $FEC          ;CLR FORTRAN FEC
005160 005067 173670          CLR      $FPC          ;CLR FORTRAN FPC
005164 005067 173652          CLR      FPS          ;CLR FPU FPS BUFFER
005170 005067 173650          CLR      FEC          ;CLR FPU FEC BUFFER
005174 005067 173646          CLR      FPC          ;CLR FPU FPC BUFFER
005200 004767 002042          JSR      PC,      RANDM2      ;GET RANDOM INPUT DATA
005204 004467 001070          JSR      R4,      $POLSH      ;ENTER POLISH MODE
005210 006312          $P.2A          ;PUSH 2 WORDS ON STACK (LONUM)
005212 006334          $P.2B          ;PUSH 2 WORDS ON STACK (HINUM)
005214 000000G          $DVR          ;ADDRESS OF FORTRAN DIVIDE
005216 006402          $TST          ;DETERMINE THE CONDITION CODES
005220 006346          $POP2X         ;POP 2 WORDS AND EXIT POLISH MODE

005222 016700 173622          MOV      $FPS,      R0          ;DISPLAY FLOATING POINT STATUS
005226 170127 040000          LDFPS     #040000          ;SET FID
005232 172467 173544          LDF      LONUM,      ACO      ;LOAD ACO WITH A RANDOM NUMBER
005236 172567 173550          LDF      HINUM,      AC1      ;LOAD AC1 WITH A RANDOM NUMBER
005242 172767 173564          LDF      ANS2,      AC3      ;LOAD AC3 WITH THE SUM
005246 170127 047440          LDFPS     #047440          ;SET INTERRUPT DISABLE AND INTERRUPT BITS
005252 012767 005260 003722          MOV      #.+6,      LAD      ;RESET LOOP ADDRESS

```


[illegible]

Address	Instruction	Op1	Op2	Op3	Op4	Comment
005260	LDI	AC0.	AC2			:LOAD AC0 INTO AC2
005262	DIVF	AC1.	AC2			:DIVIDE AC1 INTO AC2
005264	STFPS	FPS				:STORE FLOATING POINT STATUS
005270	TST	\$FPS				:CHECK FOR ERROR FLAG
005274	BMI	ERR13				:BRANCH IF ERROR FLAG SET
005276	CMP	FPS	\$FPS			:CHECK FPS
005304	BEQ	TST13				:BRANCH IF OK
005306	STF	AC2.	ANS1			:SAVE FPU ANSWER
005312	HLT+1					:FPS ERROR
005314	BR	END13				:SKIP COMPARE
005316	STST	FEC				:STORE FEC AND FPC
005322	CMP	FPS.	\$FPS			:CHECK THE FLOATING POINT STATUS
005330	BEQ	+.6				:BRANCH IF OK
005332	HLT+377					:FPS ERROR
005334	BR	END13				:SKIP TO END
005336	CMP	FEC.\$FEC				:CHECK THE FLOATING EXCEPTION CODES
005344	BEQ	+.6				:BRANCH IF OK
005346	HLT+377					:FEC IS WRONG
005350	BR	END13				:SKIP TO END
005352	CMP	FPC.\$FPC				:CHECK FLOATING PC
005360	BEQ	TST13				:BRANCH IF OK
005362	HLT+377					:WRONG ADDRESS IN FPC
005364	BR	END13				:SKIP TO END
005366	BIT	#2	\$FPS			:CHECK FOR OVERFLOW
005374	BNE	END13				:BRANCH IF OVERFLOW
005376	CMPF	AC2.	AC3			:COMPARE FPU ANSWER TO FORTRAN ANSWER
005400	CFCC					:COPY FLOATING CONDITION CODES
005402	BEQ	END13				:ANSWERS CHECK
005404	STF	AC2.	ANS1			:COMPENSATE FOR FORTRAN INACCURACIES.
005410	ADD	#1	ANS1+2			:SAVE FPU ANSWER
005416	ADC	ANS1				:INCREMENT FPU ANSWER
005422	CMPF	ANS1.	AC3			:CHECK ANSWERS AGAIN
005426	CFCC					:COPY FLOATING CONDITION CODES
005430	BEQ	END13				:BRANCH IF OK
005432	SUB	#2	ANS1+2			:DECREMENT FPU ANSWER
005440	SBC	ANS1				
005444	CMPF	ANS1.	AC3			:CHECK ANSWERS AGAIN
005450	CFCC					:COPY FLOATING CONDITION CODES
005452	BEQ	END13				:BRANCH IF OK
005454	STF	AC2.	ANS1			:SAVE FPU ANSWER
005460	HLT					:FPU AND FORTRAN DISAGREE
005462	CLR	FPS				:CLR FPU FPS BUFFER
005466	SCOPE					

```

*****
:TEST 14:      EXERCISE DIVD (DIVIDE DOUBLE PRECISION)
:              INTERRUPT DISABLE SET
:              TRUNCATE MODE
*****

```

005470	012767	047640	173352	MOV	\$047640,\$FPS	:SET FID AND IE BITS IN FORTRAN ANSWER
005476	005067	173350		CLR	\$FEC	:CLR FORTRAN FEC
005502	005067	173346		CLR	\$FPC	:CLR FORTRAN FPC
005506	005067	173330		CLR	FPS	:CLR FPU FPS BUFFER
005512	005067	173326		CLR	FEC	:CLR FPU FEC BUFFER
005516	005067	173324		CLR	FPC	:CLR FPU FPC BUFFER
005522	004767	001656		JSR	PC, RANDM4	:GET RANDOM INPUT DATA
005526	004467	000546		JSR	R4, \$POLSH	:ENTER POLISH MODE
005532	006302			\$P.4A		:PUSH 4 WORDS ON STACK (LONUM)
005534	006324			\$P.4B		:PUSH 4 WORDS ON STACK (HINUM)
005536	000000G			\$DIVD		:ADDRESS OF FORTRAN DIVIDE
005540	006402			\$TST		:DETERMINE THE CONDITION CODES
005542	006360			\$POP4X		:POP 4 WORDS AND EXIT POLISH MODE
005544	016700	173300		MOV	\$FPS, RO	:DISPLAY FLOATING POINT STATUS
005550	170127	040200		LDFPS	\$040200	:SET FID AND FD
005554	172467	173222		LDD	LONUM, AC0	:LOAD AC0 WITH A RANDOM NUMBER
005560	172567	173226		LDD	HINUM, AC1	:LOAD AC1 WITH A RANDOM NUMBER
005564	172767	173242		LDD	ANS2, AC3	:LOAD AC3 WITH THE SUM
005570	170127	047640		LDFPS	\$047640	:SET INTERRUPT DISABLE AND INTERRUPT BITS
005574	012767	005602	003400	MOV	\$.+6, LAD	:RESET LOOP ADDRESS

005602	172600			LDD	AC0, AC2	:LOAD AC0 INTO AC2
005604	174601			DIVD	AC1, AC2	:DIVIDE AC1 INTO AC2
005606	170267	173230		STFPS	FPS	:STORE FLOATING POINT STATUS
005612	005767	173232		TST	\$FPS	:CHECK FOR ERROR FLAG SET
005616	100410			BMI	ERR14	:BRANCH IF ERROR FLAG SET
005620	026767	173216	173222	CMP	FPS, \$FPS	:CHECK FPS
005626	001430			BEG	TST14	:BRANCH IF OK
005630	174267	173166		STD	AC2, ANS1	:SAVE FPU ANSWER
005634	104001			HLT+1		:FPS ERROR
005636	000472			BR	END14	:SKIP COMPARE
005640	170367	173200		STST	FEC	:STORE FEC AND FPC
005644	026767	173172	173176	CMP	FPS, \$FPS	:CHECK THE FLOATING POINT STATUS
005652	001402			BEG	+.6	:BRANCH IF OK
005654	104377			HLT+377		:FPS ERROR
005656	000462			BR	END14	:SKIP TO END
005660	026767	173160	173164	CMP	FEC,\$FEC	:CHECK THE FLOATING EXCEPTION CODES
005666	001402			BEG	+.6	:BRANCH IF OK
005670	104377			HLT+377		:FEC IS WRONG
005672	000454			BR	END14	:SKIP TO END
005674	026767	173146	173152	CMP	FPC,\$FPC	:CHECK FLOATING PC
005702	001402			BEG	TST14	:BRANCH IF OK
005704	104377			HLT+377		:WRONG ADDRESS IN FPC

005706	000446			BR	END14		;SKIP TO END
005710	032767	000002	173132	TST14:	BIT	#2	\$FPS
005716	001042				BNE	END14	:CHECK FOR OVERFLOW
005720	173702				CMPD	AC2,	AC3
005722	170000				CFCC		:COMPARE FPU ANSWER TO FORTRAN ANSWER
005724	001437				BEG	END14	:COPY FLOATING CONDITION CODES
							:ANSWERS CHECK
							:COMPENSATE FOR FORTRAN INACCURACIES.
005726	174267	173070			STD	AC2,	ANS1
005732	062767	000001	173070		ADD	#1	ANS1+6
005740	005567	173062			ADC	ANS1+4	:SAVE FPU ANSWER
005744	005567	173054			ADC	ANS1+2	:INCREMENT FPU ANSWER
005750	005567	173046			ADC	ANS1	
005754	173767	173042			CMPD	ANS1,	AC3
005760	170000				CFCC		:CHECK ANSWERS AGAIN
005762	001420				BEG	END14	:COPY FLOATING CONDITION CODES
005764	162767	000002	173036		SUB	#2	ANS1+6
005772	005667	173030			SBC	ANS1+4	:BRANCH IF OK
005776	005667	173022			SBC	ANS1+2	:DECREMENT FPU ANSWER
006002	005667	173014			SBC	ANS1	
006006	173767	173010			CMPD	ANS1,	AC3
006012	170000				CFCC		:CHECK ANSWERS AGAIN
006014	001403				BEG	END14	:COPY FLOATING CONDITION CODES
006016	174267	173000			STD	AC2,	ANS1
006022	104000				HLT		:BRANCH IF OK
							:SAVE FPU ANSWER
							:FPU AND FORTRAN DISAGREE
006024	005067	173012		END14:	CLR	FPS	:CLR FPU FPS BUFFER
006030	104400				SCOPE		

```

006032 032737 002000 177570 DONE: BIT #SW10,2#SWR ;RING THE BELL?
006040 001005 1$ BNE 1$ ;NO!
006042 012767 000207 003126 MOV #BELL,.TYPE ;TYPE A BELL
006050 000004 011176 TYPE. ;TYPE
006054 005046 1$: CLR -(6) ;CLEAR TRACE TRAP
006056 032737 010000 177570 BIT #SW12,2#SWR ;RUN WITH TRT?
006064 001010 2$ BNE 2$
006066 005167 003106 COM TRPB
006072 100005 2$ BPL 2$
006074 052716 000020 BIS #20,(6) ;SET TRACE TRAP
006100 012746 006132 MOV #3$, -(6) ;JUMP TO START OF TEST
006104 000002 RTI
006106 012746 006114 2$: MOV #4$, -(6) ;JUMP TO START OF TEST
006112 000002 RTI
006114 013700 000042 4$: MOV @#42,PO ;GET MONITOR ADDRESS
006120 001404 BEQ 3$ ;IF NONE
006122 004710 JSR 7,(0) ;GO TO MONITOR
006124 000240 NOP
006126 000240 NOP
006130 000240 NOP
006132 000137 000200 3$: JMP @#200 ;JUMP TO START OF TEST
006136 000002 YESRT: RTI ;RETURN TO PROGRAM FROM TRAP

006140 032737 000400 177570 .EMT: BIT #SW08,2#SWR ;KILL LDUB OR LOOP ON SPEC. TEST
006146 001404 BEQ 1$
006150 123767 177570 172622 CMPB @#SWR,ICNT ;ON RIGHT TEST? *SW7-0*
006156 001437 BEQ OVER
006160 113703 177570 1$: MOVB @#SWR,R3 ;GET US BITS
006164 170003 LDUB
006166 032737 040000 177570 BIT #SW14,2#SWR ;LOOP ON TEST
006174 001026 BNE KIT
006176 032737 004000 177570 BIT #SW11,2#SWR ;KILL ITERATIONS
006204 001012 BNE SAVLAD
006206 105767 172567 TSTB ICNT+1
006212 001404 BEQ 2$ ;BRANCH IF FIRST
006214 126767 002770 172557 CMPB TIMES,ICNT+1 ;DONE?
006222 001013 BNE KIT ;BRANCH IF NOT
006224 112767 000001 172547 2$: MOVB #1,ICNT+1 ;FIRST ITERATION
006232 105267 172542 SAVLAD: INCB ICNT ;COUNT TEST NUMBERS
006236 011667 002740 MOV (6),LAD ;SAVE LOOP ADDRESS
006242 016737 172532 177570 MOV ICNT,@#DISPLAY ;DISPLAY TEST NO. AND ITERATION COUNT
006250 000002 RTI ;RETURN

006252 105267 172523 KIT: INCB ICNT+1
006256 016737 172516 177570 OVER: MOV ICNT,@#DISPLAY ;SET UP DISPLAY
006264 005767 002712 TST LAD ;FIRST ONE?
006270 001760 BEQ SAVLAD
006272 016716 002704 MOV LAD,(6) ;FUDGE RETURN ADDRESS
006276 000002 RTI ;FIXES PS

```


006300	000134			\$FOLSH: JMP	2(R4)+	
006302	016746	172502		\$P.4A: MOV	LONUM+6, -(SP)	
006306	016746	172474		MOV	LONUM+4, -(SP)	
006312	016746	172466		\$P.2A: MOV	LONUM+2, -(SP)	
006316	016746	172460		MOV	LONUM, -(SP)	
006322	000134			JMP	2(R4)+	
006324	016746	172470		\$P.4B: MOV	HINUM+6, -(SP)	
006330	016746	172462		MOV	HINUM+4, -(SP)	
006334	016746	172454		\$P.2B: MOV	HINUM+2, -(SP)	
006340	016746	172446		MOV	HINUM, -(SP)	
006344	000134			JMP	2(R4)+	
006346	012667	172460		\$POP2X: MOV	(SP)+, ANS2	
006352	012667	172456		MOV	(SP)+, ANS2+2	
006356	000204			RTS	R4	:EXIT POLISH MODE
006360	012667	172446		\$POP4X: MOV	(SP)+, ANS2	
006364	012667	172444		MOV	(SP)+, ANS2+2	
006370	012667	172442		MOV	(SP)+, ANS2+4	
006374	012667	172440		MOV	(SP)+, ANS2+6	
006400	000204			RTS	R4	:EXIT POLISH MODE
006402	032767	100002	172440	\$TST: BIT	#100002, \$FPS	:CHECK FOR ERROR FLAG AND OVERFLOW
006410	100453			\$MT	\$TER	:BRANCH IF FLAG SET
006412	001061			\$BNE	\$TOV	:BRANCH IF OVERFLOW
006414	032716	077600		BIT	#077600, (6)	:TEST THE EXPONENT
006420	001041			\$BNE	\$TST1	:BRANCH IF NOT ZERO
006422	052767	000004	172420	\$BIS	#04, \$FPS	:SET Z BIT
006430	032767	077600	172354	BIT	#077600, HINUM	:CHECK DIVISOR FOR ZERO
006436	001032			\$BNE	\$TST1	:BRANCH IF NON-ZERO
006440	052767	100000	172402	\$BIS	#100000, \$FPS	:SET THE ERROR FLAG
006446	012767	000004	172376	MOV	#04, \$FEC	:DIVIDE BY ZERO EXCEPTION CODE
006454	116701	172320		MOVB	ICNT, R1	:GET THE TEST NUMBER
006460	006301			ASL	R1	: *2
006462	016167	006572	172364	MOV	RETAD-2(R1), \$FPC	:GET EXPECTED PC
006470	016716	172306		MOV	LONUM, (SP)	:RESTORE DIVIDEND
006474	016766	172304	000002	MOV	LONUM+2, 2(SP)	
006502	105767	172342		\$TSTB	\$FPS	:CHECK FOR DOUBLE PRECISION
006506	100006			\$BPL	\$TST1	:BRANCH IF NOT
006510	016766	172272	000004	MOV	LONUM+4, 4(SP)	
006516	016766	172266	000006	MOV	LONUM+6, 6(SP)	
006524	005716			\$TST1: TST	(SP)	:FIND THE SIGN
006526	100003			\$BPL	\$TST2	:BRANCH IF PLUS
006530	052767	000010	172312	\$BIS	#10, \$FPS	:SET N BIT
006536	000134			\$TST2: JMP	2(R4)+	
006540	116701	172234		\$TER: MOVB	ICNT, R1	:GET TEST NUMBER
006544	005301			DEC	R1	:SET POINTER
006546	006301			ASL	R1	:TEST * * 2
006550	016167	006574	172276	MOV	RETAD(1), \$FPC	:STORE FORTRAN FPC
006556	022626			\$TOV: CMP	(SP)+, (SP)+	: "POP" 2 WORDS
006560	105767	172264		\$TSTB	\$FPS	:CHECK FD BIT
006564	100001			\$BPL	\$TOV1	:BRANCH IF FLOATING MODE

Address	Hex	Label	Op	Op2	Op3	Op4	Op5	Op6	Op7	Op8	Op9	Op10	Op11	Op12	Op13	Op14	Op15	Op16	Op17	Op18	Op19	Op20	Op21	Op22	Op23	Op24	Op25	Op26	Op27	Op28	Op29	Op30	Op31	Op32	Op33	Op34	Op35	Op36	Op37	Op38	Op39	Op40	Op41	Op42	Op43	Op44	Op45	Op46	Op47	Op48	Op49	Op50	Op51	Op52	Op53	Op54	Op55	Op56	Op57	Op58	Op59	Op60	Op61	Op62	Op63	Op64	Op65	Op66	Op67	Op68	Op69	Op70	Op71	Op72	Op73	Op74	Op75	Op76	Op77	Op78	Op79	Op80	Op81	Op82	Op83	Op84	Op85	Op86	Op87	Op88	Op89	Op90	Op91	Op92	Op93	Op94	Op95	Op96	Op97	Op98	Op99	Op100	Op101	Op102	Op103	Op104	Op105	Op106	Op107	Op108	Op109	Op110	Op111	Op112	Op113	Op114	Op115	Op116	Op117	Op118	Op119	Op120	Op121	Op122	Op123	Op124	Op125	Op126	Op127	Op128	Op129	Op130	Op131	Op132	Op133	Op134	Op135	Op136	Op137	Op138	Op139	Op140	Op141	Op142	Op143	Op144	Op145	Op146	Op147	Op148	Op149	Op150	Op151	Op152	Op153	Op154	Op155	Op156	Op157	Op158	Op159	Op160	Op161	Op162	Op163	Op164	Op165	Op166	Op167	Op168	Op169	Op170	Op171	Op172	Op173	Op174	Op175	Op176	Op177	Op178	Op179	Op180	Op181	Op182	Op183	Op184	Op185	Op186	Op187	Op188	Op189	Op190	Op191	Op192	Op193	Op194	Op195	Op196	Op197	Op198	Op199	Op200	Op201	Op202	Op203	Op204	Op205	Op206	Op207	Op208	Op209	Op210	Op211	Op212	Op213	Op214	Op215	Op216	Op217	Op218	Op219	Op220	Op221	Op222	Op223	Op224	Op225	Op226	Op227	Op228	Op229	Op230	Op231	Op232	Op233	Op234	Op235	Op236	Op237	Op238	Op239	Op240	Op241	Op242	Op243	Op244	Op245	Op246	Op247	Op248	Op249	Op250	Op251	Op252	Op253	Op254	Op255	Op256	Op257	Op258	Op259	Op260	Op261	Op262	Op263	Op264	Op265	Op266	Op267	Op268	Op269	Op270	Op271	Op272	Op273	Op274	Op275	Op276	Op277	Op278	Op279	Op280	Op281	Op282	Op283	Op284	Op285	Op286	Op287	Op288	Op289	Op290	Op291	Op292	Op293	Op294	Op295	Op296	Op297	Op298	Op299	Op300	Op301	Op302	Op303	Op304	Op305	Op306	Op307	Op308	Op309	Op310	Op311	Op312	Op313	Op314	Op315	Op316	Op317	Op318	Op319	Op320	Op321	Op322	Op323	Op324	Op325	Op326	Op327	Op328	Op329	Op330	Op331	Op332	Op333	Op334	Op335	Op336	Op337	Op338	Op339	Op340	Op341	Op342	Op343	Op344	Op345	Op346	Op347	Op348	Op349	Op350	Op351	Op352	Op353	Op354	Op355	Op356	Op357	Op358	Op359	Op360	Op361	Op362	Op363	Op364	Op365	Op366	Op367	Op368	Op369	Op370	Op371	Op372	Op373	Op374	Op375	Op376	Op377	Op378	Op379	Op380	Op381	Op382	Op383	Op384	Op385	Op386	Op387	Op388	Op389	Op390	Op391	Op392	Op393	Op394	Op395	Op396	Op397	Op398	Op399	Op400	Op401	Op402	Op403	Op404	Op405	Op406	Op407	Op408	Op409	Op410	Op411	Op412	Op413	Op414	Op415	Op416	Op417	
---------	-----	-------	----	-----	-----	-----	-----	-----	-----	-----	-----	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	--

006776	004467	177276	\$OVERD:	JSR	R4,	\$POLSH	:ENTER POLISH MODE
007002	006312			\$P.2A			:PUSH 2 WORDS ON STACK (LONUM)
007004	006334			\$P.2B			:PUSH 2 WORDS ON STACK (HINUM)
007006	007114			\$AD200			:ADD 200 TO THE EXPONENT OF THE DIVISOR
007010	000000G			\$DVR			:ADDRESS OF FORTRAN DIVIDE
007012	007106			\$200AD			:ADD 200 TO THE EXPONENT OF ANS
007014	006402			\$TST			:DETERMINE CONDITION CODES
007016	006346			\$POP2X			:POP 2 WORDS AND EXIT POLISH MODE
007020	000411			BR	\$OVERA		
007022	004467	177252	\$OVER1:	JSR	R4,	\$POLSH	:ENTER POLISH MODE
007026	006302			\$P.4A			:PUSH 2 WORDS ON STACK (LONUM)
007030	006324			\$P.4B			:PUSH 2 WORDS ON STACK (HINUM)
007032	007114			\$AD200			:ADD 200 TO THE EXPONENT OF THE DIVISOR
007034	000000G			\$DVD			:ADDRESS OF FORTRAN DIVIDE
007036	007106			\$200AD			:ADD 200 TO THE EXPONENT OF ANS
007040	006402			\$TST			:DETERMINE CONDITION CODES
007042	006360			\$POP4X			:POP 4 WORDS AND EXIT POLISH MODE
007044	032767	003000	\$OVERA:	BIT	#003000,\$FPS		:CHECK FOR INTERRUPTS ON OR OFF
007052	001406			BEG	\$OVR1		:BRANCH IF OFF
007054	052767	100000		BIS	#100000,\$FPS		:SET FPS ERROR FLAG
007062	012767	000010		MOV	#10,\$FEC		:SET OVERFLOW EXCEPTION CODE
007070	052767	000002	\$OVR1:	BIS	#02,\$FPS		:SET OVERFLOW BIT IN FPS
007076	000205		\$ERTS:	RTS	%5		
007100	162716	040000	\$SB200:	SUB	#040000,(SP)		
007104	000134			JMP	2(R4)+		
007106	062716	140000	\$200AD:	ADD	#140000,(SP)		
007112	000134			JMP	2(R4)+		
007114	062716	040000	\$AD200:	ADD	#040000,(SP)		
007120	000134			JMP	2(R4)+		
007122	162716	040000	\$200SB:	SUB	#040000,(SP)		
007126	032767	000003		BIT	#3,PS		:TEST FOR C OR V BITS
007134	001402			BEG	\$201SB		
007136	062716	100000		ADD	#100000,(SP)		
007142	000134		\$201SB:	JMP	2(R4)+		
007144	004467	177130	\$DVBZ0:	JSR	R4,	\$POLSH	:ENTER POLISH MODE
007150	006312			\$P.2A			:PUSH DIVIDEND
007152	006402			\$TST			:CALCULATE CONDITION CODES
007154	006346			\$POP2X			:POP DIVIDEND INTO ANSWER
007156	000405			BR	\$DVBZA		
007160	004467	177114	\$DVBZ1:	JSR	R4,	\$POLSH	:ENTER POLISH MODE
007164	006302			\$P.4A			:PUSH DIVIDEND
007166	006402			\$TST			:CALCULATE CONDITION CODES
007170	006360			\$POP4X			:POP DIVIDEND INTO ANSWER
007172	052767	100000	\$DVBZA:	BIS	#100000,\$FPS		:SET FPS ERROR FLAG
007200	012767	000004		MOV	#04,\$FEC		:DIVIDE BY ZERO EXCEPTION CODE
007206	000205			RTS	%5		:RETURN TO FORTRAN

:*****

;FPP INTERRUPT SERVICE ROUTINE

;*****

007210	170267	171626	FLTERR:	STFPS	FPS	;STORE FLOATING POINT STATUS
007214	170367	171624		STST	FEC	;STORE FLOATING EXCEPTION CODES
007220	032767	040000	171622	BIT	#40000, \$FPS	;CHECK INTERRUPT DISABLE
007226	001402			BEQ	.+6	;BRANCH IF OFF
007230	104377			HLT+377		;INTERUPT NOT SUPPOSED TO BE ENABLED
007232	000404			BR	ERRTI	
007234	005767	171610		TST	\$FPS	;CHECK FORTRAN FPS FOR ERROR FLAG
007240	100401			BMI	ERRTI	;BRANCH IF FLAG SET
007242	104377			HLT+377		;FLOATING POINT STATUS ERROR
007244	000002		ERRTI:	RTI		

007246	010046		RANDM2: MOV	%0, -(6)	:SAVE R0
007250	010146		MOV	%1, -(6)	:SAVE R1
007252	010246		MOV	%2, -(6)	:SAVE R2
007254	010346		MOV	%3, -(6)	:SAVE R3
007256	010446		MOV	%4, -(6)	:SAVE R4
007260	010546		MOV	%5, -(6)	:SAVE R5
007262	012704	001002	MOV	#LONUM, %4	:SET UP LONUM POINTER
007266	012705	001012	MOV	#HINUM, %5	:SET UP HINUM POINTER
007272	011400		MOV	(4), %0	:SET R0 WITH LOW
007274	011501		MOV	(5), %1	:SET R1 WITH HIGH
007276	012703	177771	REPET2: MOV	#-7, %3	:SET SHIFT COUNT
007302	005002		CLR	%2	
007304	006300		SHIFT2: ASL	%0	:SHIFT R0 LEFT AND
007306	006101		ROL	%1	:ROTATE CARRY INTO R1 AND
007310	006102		ROL	%2	:ROTATE CARRY INTO R2
007312	005203		INC	%3	:CHECK FOR DONE
007314	001373		BNE	SHIFT2	:CONTINUE SHIFT LOOP
007316	061402		ADD	(4), %2	:ADD NUMBER TO MAKE X 129
007320	005501		ADC	%1	:PROPOGATE CARRY
007322	061501		ADD	(5), %1	:ADD NUMBER TO MAKE X 129
007324	005502		ADC	%2	:PROPOGATE CARRY
007326	062700	001057	ADD	#1057, %0	:ADD LOW CONSTANT
007332	005501		ADC	%1	:PROPOGATE CARRY
007334	005502		ADC	%2	:PROPOGATE CARRY
007336	062701	047401	ADD	#47401, %1	:ADD HIGH CONSTANT
007342	005502		ADC	%2	:PROPOGATE CARRY
007344	062702	000006	ADD	#6, %2	:ADD HIGHEST CONSTART
007350	060200		ADD	%2, %0	:REPRIME R0 WITH HIGHEST DIGIT
007352	005501		ADC	%1	:PROPOGATE CARRY
007354	010024		MOV	%0, (4)+	:SAVE R0
007356	010125		MOV	%1, (5)+	:SAVE R1
007360	020427	001006	CMP	%4, #LONUM+4	:CHECK FOR DONE ENOUGH
007364	001344		BNE	REPET2	:BRANCH IF NOT DONE
007366	012605		MOV	(6)+, %5	:RESTORE R5
007370	012604		MOV	(6)+, %4	:RESTORE R4
007372	012603		MOV	(6)+, %3	:RESTORE R3
007374	012602		MOV	(6)+, %2	:RESTORE R2
007376	012601		MOV	(6)+, %1	:RESTORE R1
007400	012600		MOV	(6)+, %0	:RESTORE R0
007402	000207		RTS	%7	:RETURN

007404	010046		RANDM4: MOV	%0,-(6)	:SAVE R0
007406	010146		MOV	%1,-(6)	:SAVE R1
007410	010246		MOV	%2,-(6)	:SAVE R2
007412	010346		MOV	%3,-(6)	:SAVE R3
007414	010446		MOV	%4,-(6)	:SAVE R4
007416	010546		MOV	%5,-(6)	:SAVE R5
007420	012704	001002	MOV	#LONUM,%4	:SET UP LONUM POINTER
007424	012705	001012	MOV	#HINUM,%5	:SET UP HINUM POINTER
007430	011400		MOV	(4),%0	:SET R0 WITH LOW
007432	011501		MOV	(5),%1	:SET R1 WITH HIGH
007434	012703	177771	REPET4: MOV	#-7,%3	:SET SHIFT COUNT
007440	005002		CLR	%2	
007442	006300		SHIFT4: ASL	%0	:SHIFT R0 LEFT AND
007444	006101		ROL	%1	:ROTATE CARRY INTO R1 AND
007446	006102		ROL	%2	:ROTATE CARRY INTO R2
007450	005203		INC	%3	:CHECK FOR DONE
007452	001373		BNE	SHIFT4	:CONTINUE SHIFT LOOP
007454	061402		ADD	(4),%2	:ADD NUMBER TO MAKE X 129
007456	005501		ADC	%1	:PROPOGATE CARRY
007460	061501		ADD	(5),%1	:ADD NUMBER TO MAKE X 129
007462	005502		ADC	%2	:PROPOGATE CARRY
007464	062700	001057	ADD	#1057,%0	:ADD LOW CONSTANT
007470	005501		ADC	%1	:PROPOGATE CARRY
007472	005502		ADC	%2	:PROPOGATE CARRY
007474	062701	047401	ADD	#47401,%1	:ADD HIGH CONSTANT
007500	005502		ADC	%2	:PROPOGATE CARRY
007502	062702	000006	ADD	#6,%2	:ADD HIGHEST CONSTART
007506	060200		ADD	%2,%0	:REPRIME R0 WITH HIGHEST DIGIT
007510	005501		ADC	%1	:PROPOGATE CARRY
007512	010024		MOV	%0,(4)+	:SAVE R0
007514	010125		MOV	%1,(5)+	:SAVE R1
007516	020427	001012	CMP	%4,#LONUM+10	:CHECK FOR DONE ENOUGH
007522	001344		BNE	REPET4	:BRANCH IF NOT DONE
007524	012605		MOV	(6)+,%5	:RESTORE R5
007526	012604		MOV	(6)+,%4	:RESTORE R4
007530	012603		MOV	(6)+,%3	:RESTORE R3
007532	012602		MOV	(6)+,%2	:RESTORE R2
007534	012601		MOV	(6)+,%1	:RESTORE R1
007536	012600		MOV	(6)+,%0	:RESTORE R0
007540	000207		RTS	%7	:RETURN

007542	032767	002000	170020	.TRP:	BIT	#2000,SWR	
007550	001405				BEG	.ET	
007552	012767	000207	001416		MOV	#BELL,.TYPE	;TYPE A BELL
007560	000004	011176			TYPE,	.TYPE	
007564	005267	001414		.ET:	INC	ERRORS	;COUNT THE NUMBER OF ERRORS
007570	032767	020000	167772		BIT	#20000,SWR	;SKIP TYPEOUT IF SET
007576	001116				BNE	NHEAD	
007600	174267	171216			STF	AC2, ANS1	;SAVE THE ANSWER TO BE SURE
007604	000004	011102			TYPE,	RET	
007610	000004	011102			TYPE,	RET	
007614	011646				MOV	(6),-(6)	;PUT ADDRESS OF INSTRUCTION ON STACK
007616	162716	000002			SUB	#2,(6)	
007622	117667	000000	001356		MOVB	2(6), WORDS	
007630	012605				MOV	(6)+,TTY	;TYPE (6)+ IN OCTAL
007632	004767	000546			JSR	%7,PRINTR	;TYPE LEADING ZERO'S
007636	000004	011077			TYPE,	SPACE+3	
007642	012703	001002			MOV	#LONUM,%3	;SET UP POINTER
007646	004767	000174			JSR	7, \$TYPE	;TYPE A FLOATING POINT NUMBER
007652	000004	011105			TYPE,	\$SIGN	
007656	004767	000164			JSR	7, \$TYPE	;TYPE A FLOATING POINT NUMBER
007662	000004	011130			TYPE,	FPUAN	
007666	004767	000154			JSR	7, \$TYPE	;TYPE A FLOATING POINT NUMBER
007672	000004	011076			TYPE,	SPACE+2	
007676	016705	171140			MOV	FPS,TTY	;TYPE FPS IN OCTAL
007702	004767	000476			JSR	%7,PRINTR	;TYPE LEADING ZERO'S
007706	105767	001274			TSTB	WORDS	;CHECK FOR STATUS ERROR
007712	100014				BPL	.STAT	;BRANCH IF NOT
007714	000004	011076			TYPE,	SPACE+2	
007720	016705	171120			MOV	FEC,TTY	;TYPE FEC IN OCTAL
007724	004767	000454			JSR	%7,PRINTR	;TYPE LEADING ZERO'S
007730	000004	011076			TYPE,	SPACE+2	
007734	016705	171106			MOV	FPC,TTY	;TYPE FPC IN OCTAL
007740	004767	000440			JSR	%7,PRINTR	;TYPE LEADING ZERO'S
007744	000004	011111		.STAT:	TYPE,	FORTAN	
007750	004767	000072			JSR	7, \$TYPE	;TYPE A FLOATING POINT NUMBER
007754	105767	001226		NOHEAD:	TSTB	WORDS	
007760	001425				BEG	NHEAD	
007762	000004	011076			TYPE,	SPACE+2	
007766	016705	171056			MOV	\$FPS,TTY	;TYPE \$FPS IN OCTAL
007772	004767	000406			JSR	%7,PRINTR	;TYPE LEADING ZERO'S
007776	105767	001204			TSTB	WORDS	
010002	100014				BPL	NHEAD	
010004	000004	011076			TYPE,	SPACE+2	
010010	016705	171036			MOV	\$FEC,TTY	;TYPE \$FEC IN OCTAL
010014	004767	000364			JSR	%7,PRINTR	;TYPE LEADING ZERO'S
010020	000004	011076			TYPE,	SPACE+2	
010024	016705	171024			MOV	\$FPC,TTY	;TYPE \$FPC IN OCTAL
010030	004767	000350			JSR	%7,PRINTR	;TYPE LEADING ZERO'S
010034	005737	177570		NHEAD:	TST	2#SWR	
010040	100001				BPL	.+4	
010042	000000				HALT		
010044	000002				RTI		

```

010046 032767 001000 167514 $TYPE: BIT #1000, SWR ;CHECK TTY FORMAT
010054 001007 BNE TYPEI
010056 105767 170766 TSTB $FPS
010062 100432 BMI TYPED
010064 004767 000076 JSR 7, TYPEF
010070 022323 TYPEA: CMP (3)+, (3)+ ;UP DATE THE TYPEOUT POINTER
010072 000207 RTS 7

010074 TYPEI:
010074 012305 MOV (3)+, TTY ;TYPE (3)+ IN OCTAL
010076 004767 000302 JSR %7, PRINTR ;TYPE LEADING ZERO'S
010102 000004 011100 TYPE, SPACE+4
010106 012305 MOV (3)+, TTY ;TYPE (3)+ IN OCTAL
010110 004767 000270 JSR %7, PRINTR ;TYPE LEADING ZERO'S
010114 105767 170730 TSTB $FPS
010120 100363 BPL TYPEA
010122 000004 011100 TYPE, SPACE+4
010126 012305 MOV (3)+, TTY ;TYPE (3)+ IN OCTAL
010130 004767 000250 JSR %7, PRINTR ;TYPE LEADING ZERO'S
010134 000004 011100 TYPE, SPACE+4
010140 012305 MOV (3)+, TTY ;TYPE (3)+ IN OCTAL
010142 004767 000236 JSR %7, PRINTR ;TYPE LEADING ZERO'S
010146 000207 RTS 7

010150 012346 TYPED: MOV (3)+, -(6) ;GET WORD 1
010152 012346 MOV (3)+, -(6) ;GET WORD 2
010154 012346 MOV (3)+, -(6) ;GET WORD 3
010156 012346 MOV (3)+, -(6) ;GET WORD 4
010160 012746 000022 MOV #18, -(6) ;CHAR COUNT
010164 000406 BR TYPEI
010166 012346 TYPEF: MOV (3)+, -(6) ;GET WORD 1
010170 012346 MOV (3)+, -(6) ;GET WORD 2
010172 005046 CLR -(6) ;CLEAR WORD 3
010174 005046 CLR -(6)
010176 012746 000010 MOV #8, -(6) ;CHAR COUNT
010202 004767 000104 JSR 7, TY1 ;TYPE 1 BIT
010206 105766 000011 TSTB 9, (6) ;CHECK EXPONENT
010212 001001 BNE .+4 ;BRANCH ON NON-ZERO EXPONENT
010214 005116 COM (6) ;FLAG ZERO EXPONENT
010216 000004 011100 TYPE, SPACE+4
010222 004767 000072 JSR 7, TY2 ;TYPE 2 BITS
010226 004767 000074 JSR 7, TY3 ;TYPE 3 BITS
010232 004767 000070 JSR 7, TY3 ;TYPE 3 BITS
010236 000004 011100 TYPE, SPACE+4
010242 004767 000020 JSR 7, TYH ;TYPE 2 BITS
010246 004767 000054 TYPE2: JSR 7, TY3 ;TYPE 3 BITS
010252 005316 DEC (6) ;DONE?
010254 001374 BNE TYPE2
010256 022626 CMP (6)+, (6)+ ;RESTORE
010260 022626 CMP (6)+, (6)+ ;THE
010262 005726 TST (6)+ ;STACK
010264 000207 RTS 7

```


010266	005766	000002	TYH:	TST	2(6)	:CHECK FOR ZERO EXPONENT FLAG
010272	100405			BMI	TYZ	:BRANCH ON ZERO EXPONENT
010274	012746	000001		MOV	#1,-(6)	:TYPE HIDDEN BIT AND ONE
010300	011667	000672		MOV	(6),TYPE	:FUDGE HIDDEN BIT
010304	000414			BR	TY+4	
010306	005166	000002	TYZ:	COM	2(6)	:GET RID OF ZERO EXPONENT FLAG
010312	012746	000001	TY1:	MOV	#1,-(6)	:TYPE 1 BIT
010316	000405			BR	TY	
010320	012746	000002	TY2:	MOV	#2,-(6)	:TYPE 2 BITS
010324	000402			BR	TY	
010326	012746	000003	TY3:	MOV	#3,-(6)	:TYPE 3 BITS
010332	005067	000640	TY:	CLR	.TYPE	
010336	006166	000006		ROL	6(6)	:SHIFT WORD 4
010342	006166	000010		ROL	8(6)	:SHIFT WORD 3
010346	006166	000012		ROL	10(6)	:SHIFT WORD 2
010352	006166	000014		ROL	12(6)	:SHIFT WORD 1
010356	006167	000614		ROL	.TYPE	:GET IT
010362	005316			DEC	(6)	:DONE?
010364	001364			BNE	TY+4	
010366	052767	000060 000602		BIS	#0,TYPE	:MAKE IT ASCII
010374	000004	011176		TYPE.	.TYPE	:TYPE IT
010400	005726			TST	(6)+	:RESTORE THE STACK
010402	000207			RTS	7	

```

010404 112767 000001 000130 PRINTR: MOVB #1, .PR ;SET ZERO FILL SWITCH
010412 000402 BR .+6
010414 005067 000122 PRINTS: CLR .PR ;SUPPRESS LEADING ZERO'S
010420 112767 177772 000115 MOVB #5, .PR+1 ;SET COUNT
010426 010446 MOV R4, -(6) ;SAVE R4
010430 012704 010532 MOV #3, R4 ;SET POINTER TO FIRST ASCII CHAR.
010436 105014 CLRB (4) ;CLEAR FIRST BYTE
010440 000405 BR 2$ ;ROTATE FIRST BIT
010446 105014 1$: CLRB (4) ;CLEAR BYTE OF CHARACTER
010450 006105 ROL TTY ;ROTATE BIT INTO C
010454 106114 ROLB (4) ;PACK IT
010456 006105 ROL TTY ;ROTATE BIT INTO C
010460 106114 2$: ROLB (4) ;PACK IT
010466 105714 TSTB (4) ;ROTATE BIT INTO C
010470 001402 BEQ .+6 ;PACK IT
010474 105267 000054 INCB .+6
010476 105767 000050 TSTB .PR ;CHECK FILL SWITCH
010480 001402 BEQ .+6
010484 152724 000060 B1SB #0, (4)+ ;MAKE INTO ASCII CHAR
010500 105267 000037 INCB .PR+1
010504 001355 BNE 1$ ;REPEAT
010506 022704 010532 CMP #3, R4
010512 001002 BNE .+6
010514 112724 000060 MOVB #0, (4)+
010520 105014 CLRB (4)
010522 000004 010532 TYPE TTY ;TYPE IT
010526 012604 MOV (6)+, R4 ;RESTORE R4
010530 000207 RTS PC

010532 000004 3$: .BLKW 4
010542 000000 .PR: 0

010544 005267 000434 ERROR: INC ERRORS ;COUNT ERRORS
010550 132737 000001 000041 BITB #1, 2$41 ;AUTO MODE?
010556 001412 BEQ 1$ ;NO!
010560 022767 000010 000416 CMP #10, ERRORS ;TOO MANY?
010566 001006 BNE 1$ ;NOT YET
010570 013700 000042 MOV 2$42, R0 ;GET ADDRESS
010574 001403 BEQ 1$ ;FORGET IT IF ZERO
010576 005037 000042 CLRB 2$42 ;ZAP 42
010602 004710 JSR PC, (0) ;CALL THE MONITOR
010604 000207 1$: RTS PC ;RETURN

```


MAINDEC-11-DCFPU-3
DCFPUD.P11

FLOATING POINT DIVIDE EXERCISER
POWER DOWN AND UP ROUTINES

010606	012777	011002	000356	POWDN:	MOV	#ILLUP, SUPVEC	:SET FOR FAST UP
010614	012777	000340	000352		MOV	#340, SUPVEC+2	:PRIO:7
010622	170246				STFPS	-(6)	:GET THE FPS
010630	170011				SETD		
010638	174046				STD	AC0, -(6)	:SAVE AC'S
010646	174146				STD	AC1, -(6)	
010654	174246				STD	AC2, -(6)	
010662	174346				STD	AC3, -(6)	
010670	172404				LDD	AC4, AC0	
010678	174046				STD	AC0, -(6)	
010686	172405				LDD	AC05, AC0	
010694	174046				STD	AC00, -(6)	
010702	010046				MOV	RO, -(6)	:SAVE REGISTERS
010710	010146				MOV	RI, -(6)	
010718	010246				MOV	RI, -(6)	
010726	010346				MOV	RI, -(6)	
010734	010446				MOV	RI, -(6)	
010742	010546				MOV	RI, -(6)	
010750	010667	000264			MOV	SP, SAVE6	:SAVE SP
010758	012777	010676	000276		MOV	#POWUP, SUPVEC	:SET UP VECTOR
010766	000000				HALT		
010774							
010782	016706	000250		POWUP:	MOV	SAVE6, SP	:GET SP
010790	005001				CLR	RI	:WAIT LOOP FOR THE TTY
010798	005201			18:	BNE	RI	
010806	001376				MOV	(6)+, RI	:GET THE REGISTERS
010814	012605				MOV	(6)+, RI	
010822	012604				MOV	(6)+, RI	
010830	012603				MOV	(6)+, RI	
010838	012602				MOV	(6)+, RI	
010846	012601				MOV	(6)+, RI	
010854	012600				MOV	(6)+, RI	
010862	170011				SETD		
010870	172426				LDD	(6)+, AC0	:RESTORE THE AC'S
010878	174005				STD	AC0, AC5	
010886	172426				LDD	(6)+, AC0	
010894	174004				STD	AC0, AC4	
010902	172726				LDD	(6)+, AC3	
010910	172626				LDD	(6)+, AC2	
010918	172526				LDD	(6)+, AC1	
010926	172426				LDD	(6)+, AC0	
010934	170126				LDFPS	(6)+, AC0	:RESTORE FPS
010942	012777	010606	000210		MOV	#POWDN, SDWVEC	:SET UP THE POWER DOWN VECTOR
010950	012777	000340	000204		MOV	#340, SDWVEC+2	
010958	000004	010770			TYPE,	..+2	:ASCIZ <15><12>"POWER"
010966	000002				RTI		
010974							
010982	000000			ILLUP:	HALT		:THE POWER UP SEQUENCE WAS STARTED
010990	000776				BR	..-2	:BEFORE THE POWER DOWN WAS COMPLETE

```

011006 010546 .IOT: MOV TTY -(6) :SAVE TTY
011010 017605 000002 MOV 22(6),TTY :GET ADDRESS TO BE TYPED
011014 105715 1S: TSTB (TTY) :TERMINATOR?
011016 001406 BEQ 2S
011020 112537 177566 MOVB (TTY)+,2#177566 :LOAD AND TYPE THE CHARACTER
011024 105737 177564 TSTB 2#177564 :IS THE PRINTER READY
011030 100375 BPL 1S-4
011032 000770 BR 1S :GET THE NEXT CHARACTER
011034 017646 000002 2S: MOV 22(6),-(6) :GET ADDRESS TO BE TYPED
011040 062766 000002 000004 ADD 2,4(6) :ADD 2 TO THE ADDRESS
011046 022666 000002 CMP (6)+,2(6) :IS IT .+2?
011052 001006 BNE 3S :NO
011054 062705 000002 ADD 2,TTY :ADD 2 TO THE ADDRESS
011060 042705 000001 BIC 2,TTY :BACK UP TO AN EVEN BYTE
011064 010566 000002 MOV TTY,2(6) :RESTORE ADDRESS
011070 012605 3S: MOV (6)+,TTY :RESTORE TTY
011072 000002 RTI :RETURN

011074 005015 020040 000040 SPACE: .ASCIZ <15><12>" "
011102 005015 000 RET: .ASCIZ <15><12>
011105 040 020057 000 SSIGN: .ASCIZ "/"
011111 015 020012 043040 FORTAN: .ASCIZ <15><12>" FORTRAN: "
011116 051117 051124 047101
011124 020072 000040
011130 036440 005015 020040 FPUAN: .ASCIZ " = "<15><12>" FPU: "
011136 050106 035125 020040
011144 020040 020040 000
011152 011152 .EVEN
011152 000000 SAVES: 0
011154 177564 TPS: 177564 :TELEPRINTER STATUS REGISTER
011156 177566 TPB: 177566 :TELEPRINTER DATA BUFFER
011160 172160 FPTADR: 172160 :FLOATING POINT ADDRESS ON THE 11 20
011162 000244 000246 FPVECT: 244,246 :FLOATING POINT VECTOR ADDRESS
011166 000024 000026 DWNVEC: 244,26 :POWER DOWN VECTOR ADDRESS
011172 000024 000026 UPVEC: 244,26 :POWER UP VECTOR ADDRESS
011176 000000 .TYPE:
011200 000000 TRPB:
011202 000000 LAD: :LOOP ADDRESS
011204 000000 ERRORS: :ERROR COUNT
011206 000000 WORDS: :CONTAINS TYPEOUT INFO
011210 000277 TIMES: 277 :ITERATION COUNT
011212 000001 .END

```


MAINDEC-11-DCFPD-D FLOATING POINT DIVIDE EXERCISER
DCFPD.F11 CROSS REFERENCE TABLE -- USER SYMBOLS

0005316	1266	1273#													
0005640	1346	1353#													
0001664	1570#	1570#													
0002162	1637	1645#													
0002460	7112	720#													
0002766	789	797#													
0003276	867	875#													
0003676	949	957#													
0010444	466#	466#	502	539*	576	614*	651	689*	726	766*	803	844*	881		
	962	962	1006*	1043	1090*	1121*	1127	1165*	1196*	1202	1242*	1273*	1313		
	1353*	1353*	1359	1617*	1739										
007210	1616#	1616#													
0111111	1957#	1957#													
001046	467*	467*	508	540*	581	615*	656	690*	731	767*	800	846*	886		
	968	968	1007*	1048	1091*	1122*	1166*	1207*	1242*	1284*	1320*	1364*	1406		
001042	465*	465*	490*	491	498	528*	568*	607*	644*	684*	721*	761*	799*		
	846*	846*	869*	880*	898	912*	930*	951*	968*	990*	1000*	1020*	1040*		
	1077*	1077*	1089*	1112*	1115	1122*	1145*	1164*	1187*	1190	1200*	1220*	1240*		
	1274	1274	1309*	1309*	1321*	1344*	1347	1364*	1387*	1390	1400*	1420*	1440*		
0111160	1967#	1967#													
0111130	1960#	1960#													
0111620	451*	451*	1968												
001012	450#	450#	552#												
	479	479	552#												
	494	494	500												
	658	658	674												
	883	883	883												
	1075	1075	1118												
	1281	1281	1286												
	1421	1421	1421												
	1429	1429	1429												
	1431	1431	1431												
	1433*	1433*	1433*												
	1434*	1434*	1434*												
	1436	1436	1436												
	1439*	1439*	1439*												
	1440	1440	1440												
	1441	1441	1441												
	1443	1443	1443												
	1444	1444	1444												
	1445	1445	1445												
	1446	1446	1446												
	1447	1447	1447												
	1448	1448	1448												
	1449	1449	1449												
	1450	1450	1450												
	1451	1451	1451												
	1452	1452	1452												
	1453	1453	1453												
	1454	1454	1454												
	1455	1455	1455												
	1456	1456	1456												
	1457	1457	1457												
	1458	1458	1458												
	1459	1459	1459												
	1460	1460	1460												
	1461	1461	1461												
	1462	1462	1462												
	1463	1463	1463												
	1464	1464	1464												
	1465	1465	1465												
	1466	1466	1466												
	1467	1467	1467												
	1468	1468	1468												
	1469	1469	1469												
	1470	1470	1470												
	1471	1471	1471												
	1472	1472	1472												
	1473	1473	1473												
	1474	1474	1474												
	1475	1475	1475												
	1476	1476	1476												
	1477	1477	1477												
	1478	1478	1478												
	1479	1479	1479												
	1480	1480	1480												
	1481	1481	1481												
	1482	1482	1482												
	1483	1483	1483												
	1484	1484	1484												
	1485	1485	1485												
	1486	1486	1486												
	1487	1487	1487												
	1488	1488	1488												
	1489	1489	1489												
	1490	1490	1490												
	1491	1491	1491												
	1492	1492	1492												
	1493	1493	1493												
	1494	1494	1494												
	1495	1495	1495												
	1496	1496	1496												
	1497	1497	1497												
	1498	1498	1498												
	1499	1499	1499												
	1500	1500	1500												
	1501	1501	1501												
	1502	1502	1502												
	1503	1503	1503												
	1504	1504	1504												
	1505	1505	1505												
	1506	1506	1506												
	1507	1507	1507												
	1508	1508	1508												
	1509	1509	1509												
	1510	1510	1510												
	1511	1511	1511												
	1512	1512	1512												
	1513	1513	1513												
	1514	1514	1514												
	1515	1515	1515												
	1516	1516	1516												
	1517	1517	1517												
	1518	1518	1518												
	1519	1519	1519												
	1520	1520	1520												
	1521	1521	1521												
	1522	1522	1522												
	1523	1523	1523												
	1524	1524	1524												
	1525	1525	1525												
	1526	1526	1526												
	1527	1527	1527												
	1528	1528	1528												
	1529	1529	1529												
	1530	1530	1530												
	1531	1531	1531												
	1532	1532	1532												
	1533	1533	1533												
	1534	1534	1534												
	1535	1535	1535												
	1536	1536	1536												
	1537	1537	1537												
	1538	1538	1538												
	1539	1539	1539												
	1540	1540	1540												
	1541	1541	1541												
	1542	1542	1542												
	1543	1543	1543												

MACY11 27(732) 16-SEP-76 16:30 PAGE 46

[illegible]

```

00000001
00000002
00000003
00000004

```

```

RE          =%000005
SAVE6      011152
SAVEAD     006232
SCOPE      = 104400

```

```
SHIFT2 007304
SHIFT4 007442
SP      =%000006
```

[illegible]

FLOATING POINT DIVIDE EXERCISER
REFERENCE TABLE -- USER SYMBOLS

1027	1512
1111	1513
1186	1514
1263	1515
1343	1516
560	1506
635	1507
710	1508
787	1509
865	1510
947	1511
362	445
1332	1410
362	1480
364	1900
365	1423
366	469
1446	1452
1587	1590
367	1903
1904	1908
1428	1434
390	461
1394	
1641	1645
1682	1686
369	430
1464	1465
1589	1592
1726	1733
385	387
377	1419
378	
373	1396
380	1427
381	1401
382	
383	1425
384	
1431	1976
1966	
1966	
1503	1972
1592	509
1032	1053
1116	1133
1191	1208
1268	1285
1348	1365
1	582
665	657
710	732
790	808
790	827
930	969
960	1724

[illegible][illegible]

17404

1749

1754

1757

[illegible][illegible][illegible][illegible][illegible]

100 +
100
100

100 +
100
100

100 +
100
100

H04

MAINDEC-11-DOFPU-D
DOFPU.D.F11FLOATING POINT DIVIDE EXERCISER
CROSS REFERENCE TABLE -- USER SYMBOLS

MACY11 27(732) 16-SEP-76 16:30 PAGE 47

		1852*	1854*	1936	1937*	1938	1940	1948*	1949*	1950	1951*			
TY	0103332	1821*	1824*	1826	1828*	1835								
TYE	0102666	1800*	1817*											
TYPE	0000004	1799*	1799*	1714	1719	1720	1726	1729	1731	1733	1738	1741	1744	1748
		1775	1780	1775	1780	1783	1803	1807	1837	1868	1930			
TY	0100070	1775*	1779*											
TY	0100150	1775*	1788*											
TY	0100166	1775*	1794*											
TY	0100077	1775*	1772*											
TY	0100200	1775*	1799*											
TY	0100246	1800*	1811*											
TY	0100300	1800*	1822*											
TY	0100320	1800*	1823*											
TY	0103226	1800*	1825*											
UPVEC	0111172	1806*	1897*	1809	1827*									
WORDSC	011206	1723*	1736	1905*	1970*									
YESRT	006136	433*	441	1746	1751	1975*								
AD200	007114	1561	1571	1417*										
DVBZA	007172	1502	1608*	1589*										
DVBZO	007144	1527	1598*											
DVBZ1	007160	1529	1604*											
DVD	0000000	1514	545	695	850	1012	1171	1328	1550	1572				
DVD	0000000	1514	472	620	772	932	1096	1248	1538	1562				
MC	006624	1514	1518*	1581*										
MC	001052	1503	462*	503	536*	576	611*	651	686*	726	762*	802	838*	874*
		1503	962*	1003*	1042*	1087*	1127*	1162*	1202*	1239*	1278*	1319*	1361*	1403*
STPC	001054	1503	464*	508	537*	581	612*	656	687*	731	764*	808	844*	880*
		1497*	968	1004*	1048	1088*	1132*	1162*	1207*	1240*	1284*	1320*	1364*	1408*
STPS	001050	1497*	1757											
		426*	462*	476	488	491	498	535*	549	561	564	571	577	583
		762	639*	646	661	685*	699	711	714	721	726	731	736	741
		991	798	840*	854	866	869	876	882*	886	891	896	901	906
		1000*	1016	1028*	1031	1038	1053	1066*	1080*	1093	1106	1119	1132	1145
		1117	1188	1190	1197	1212	1238*	1252*	1267*	1281	1294	1308	1321	1335
		1314	1347	1354	1369	1470	1475*	1478*	1485*	1491*	1499	1506	1513	1520
		1576*	1578*	1580*	1608*	1618	1623	1749	1766	1778				
OVERA	007044	1576*	1576*											
OVERO	006776	1576*	1576*											
OVER1	007022	1576*	1576*											
OVER1	007070	1577	1580*											
SPOLSH	006300	468	542	617	692	769	847	929	1009	1093	1168	1245	1325	1406*
		1534	1546	1558	1568	1598	1604							
		474	622*	774	934	1098	1250	1460*	1541	1565	1601			
		477	637	832	1014	1172	1320	1460*	1541	1565	1601			
		470	630	770	930	1094	1246	1456*	1537	1561	1597			
		471	631	771	931	1095	1247	1457*	1538	1562	1598			
		472	632	772	932	1096	1248	1458*	1539	1563	1600			
		473	633	773	933	1097	1249	1459*	1540	1564	1601			
		474	634	774	934	1098	1250	1460*	1541	1565	1601			
		475	635	775	935	1099	1251	1461*	1542	1566	1602			
		476	636	776	936	1100	1252	1462*	1543	1567	1603			
		477	637	777	937	1101	1253	1463*	1544	1568	1604			
		478	638	778	938	1102	1254	1464*	1545	1569	1605			
		479	639	779	939	1103	1255	1465*	1546	1570	1606			
		480	640	780	940	1104	1256	1466*	1547	1571	1607			
		481	641	781	941	1105	1257	1467*	1548	1572	1608			
		482	642	782	942	1106	1258	1468*	1549	1573	1609			
		483	643	783	943	1107	1259	1469*	1550	1574	1610			
		484	644	784	944	1108	1260	1470*	1551	1575	1611			
		485	645	785	945	1109	1261	1471*	1552	1576	1612			
		486	646	786	946	1110	1262	1472*	1553	1577	1613			
		487	647	787	947	1111	1263	1473*	1554	1578	1614			
		488	648	788	948	1112	1264	1474*	1555	1579	1615			
		489	649	789	949	1113	1265	1475*	1556	1580	1616			
		490	650	790	950	1114	1266	1476*	1557	1581	1617			
		491	651	791	951	1115	1267	1477*	1558	1582	1618			
		492	652	792	952	1116	1268	1478*	1559	1583	1619			
		493	653	793	953	1117	1269	1479*	1560	1584	1620			
		494	654	794	954	1118	1270	1480*	1561	1585	1621			
		495	655	795	955	1119	1271	1481*	1562	1586	1622			
		496	656	796	956	1120	1272	1482*	1563	1587	1623			
		497	657	797	957	1121	1273	1483*	1564	1588	1624			
		498	658	798	958	1122	1274	1484*	1565	1589	1625			
		499	659	799	959	1123	1275	1485*	1566	1590	1626			
		500	660	800	960	1124	1276	1486*	1567	1591	1627			
		501	661	801	961	1125	1277	1487*	1568	1592	1628			
		502	662	802	962	1126	1278	1488*	1569	1593	1629			
		503	663	803	963	1127	1279	1489*	1570	1594	1630			
		504	664	804	964	1128	1280	1490*	1571	1595	1631			
		505	665	805	965	1129	1281	1491*	1572	1596	1632			
		506	666	806	966	1130	1282	1492*	1573	1597	1633			
		507	667	807	967	1131	1283	1493*	1574	1598	1634			
		508	668	808	968	1132	1284	1494*	1575	1599	1635			
		509	669	809	969	1133	1285	1495*	1576	1600	1636			
		510	670	810	970	1134	1286	1496*	1577	1601	1637			
		511	671	811	971	1135	1287	1497*	1578	1602	1638			
		512	672	812	972	1136	1288	1498*	1579	1603	1639			
		513	673	813	973	1137	1289	1499*	1580	1604	1640			
		514	674	814	974	1138	1290	1500*	1581	1605	1641			
		515	675	815	975	1139	1291	1501*	1582	1606	1642			
		516	676	816	976	1140	1292	1502*	1583	1607	1643			
		517	677	817	977	1141	1293	1503*	1584	1608	1644			
		518	678	818	978	1142	1294	1504*	1585	1609	1645			
		519	679	819	979	1143	1295	1505*	1586	1610	1646			
		520	680	820	980	1144	1296	1506*	1587	1611	1647			
		521	681	821	981	1145	1297	1507*	1588	1612	1648			
		522	682	822	982	1146	1298	1508*	1589	1613	1649			
		523	683	823	983	1147	1299	1509*	1590	1614	1650			
		524	684	824	984	1148	1300	1510*	1591	1615	1651			
		525	685	825	985	1149	1301	1511*	1592	1616	1652			
		526	686	826	986	1150	1302	1512*	1593	1617	1653			
		527	687	827	987	1151	1303	1513*	1594	1618	1654			
		528	688	828	988	1152	1304	1514*	1595	1619	1655			
		529	689	829	989	1153	1305	1515*	1596	1620	1656			
		530	690	830	990	1154	1306	1516*	1597	1621	1657			
		531	691	831	991	1155	1307	1517*	1598	1622	1658			
		532	692	832	992	1156	1308	1518*	1599	1623	1659			
		533	693	833	993	1157	1309	1519*	1600	1624	1660			
		534	694	834	994	1158	1310	1520*	1601	1625	1661			
		535	695	835	995	1159	1311	1521*	1602	1626	1662			
		536	696	836	996	1160	1312	1522*	1603	1627	1663			
		537	697	837	997	1161	1313	1523*	1604	1628	1664			

MAINDEC-11-DCFPD-D FLOATING POINT DIVIDE EXERCISER
DCFPD.P11 CROSS REFERENCE TABLE -- USER SYMBOLS

STST	006402	473	546	621	696	773	851	933	1013	1097	1172	1249	1329	1470*
STST1	006524	1540	1552	1564	1574	1600	1606							
STST2	006536	1474	1477	1486	1489*									
STST3	010046	1490	1492*											
STST4	006760	1728	1730	1732	1745	1764*								
STST5	006760	1542	1554*											
STST6	006760	1519	1532*											
STST7	006760	1521	1544*											
STST8	006760	1530*												
STST9	007106	1562	1573	1586*										
STST10	007106	1562	1551	1592*										
STST11	007106	1591	1596*											
STST12	011212	705*	396	397*	401*	482	499	504	555	572	577	600	647	660*
STST13		1039	722	727*	782	799	804	860	877	880	894	900	900	900*
STST14		1360	1044	1106	1123	1128	1181	1198	1203	1206	1206	1206	1206	1206*
STST15		1419	1619	1760	1801	1842	1857	1860	1866	1872*	1877	1884	1884	1884*
STST16		1712	1715*											
STST17		1936	1936*											
STST18		1841*	1843*	1844*	1858*	1859	1862*	1873*						
STST19		1737	1744*											
STST20		446	1711*											
STST21		1398*	1399	1713*	1714	1820*	1826*	1833*	1836*	1837	1871*			

K04

MAINDEC-11-DOFPU-D
DOFPU.D.P11FLOATING POINT DIVIDE EXERCISER
CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

MACY11 27(732) 16-SEP-76 16:30 PAGE 52

ADC	819	897	898	899	981	1061	1062	1063	1297	1377	1378	1379	1647	1649	1651
ADD	1652	1654	1657	1688	1690	1692	1693	1695	1698						
ASL	816	896	980	1060	1296	1376	1586	1589	1595	1646	1648	1650	1653	1655	1656
BEG	1687	1689	1691	1694	1696	1697	1945	1948							
	1481	1496	1641	1682											
	492	499	504	509	515	522	565	572	577	582	588	597	640	647	652
	657	665	672	715	722	727	732	740	749	792	799	804	809	815	822
	827	870	877	882	887	893	902	909	952	959	964	969	977	984	989
	1032	1039	1044	1049	1057	1066	1073	1116	1123	1128	1133	1141	1148	1191	1198
	1203	1208	1216	1225	1268	1275	1280	1285	1293	1300	1305	1348	1355	1360	1365
	1373	1382	1389	1411	1420	1422	1430	1442	1519	1521	1523	1525	1527	1529	1533
	1545	1577	1594	1619	1712	1747	1857	1860	1877	1881	1939				
	1949														
BIC	1405	1475	1478	1491	1554	1578	1580	1608	1836						
BISB	1861														
BIT	661	736	973	1053	1137	1212	1289	1369	1396	1401	1419	1425	1427	1470	1473
	1476	1532	1544	1576	1593	1618	1711	1716	1764						
BITB	1876														
BMI	489	562	637	712	789	867	949	1029	1114	1189	1266	1346	1471	1624	1767
	1818														
BNE	662	737	974	1054	1138	1213	1290	1370	1397	1402	1426	1428	1432	1472	1474
	1477	1645	1661	1686	1702	1717	1765	1801	1811	1835	1863	1865	1879	1911	1947
BPL	1404	1486	1490	1500	1737	1752	1760	1779	1942						
BR	434	495	501	506	511	568	574	579	584	643	649	654	659	718	724
	729	734	795	801	806	811	873	879	884	889	955	961	966	971	1035
	1041	1046	1051	1119	1125	1130	1135	1194	1200	1205	1210	1271	1277	1282	1287
	1351	1357	1362	1367	1542	1566	1602	1621	1793	1821	1824	1826	1842	1848	1934
	1943														
DFCC	497	514	521	570	587	596	645	664	671	720	739	748	797	814	821
	826	875	892	901	908	957	976	983	988	1037	1056	1065	1072	1140	1147
CLR	1215	1224	1292	1299	1304	1372	1381	1388							
	452	453	463	464	465	466	467	526	536	537	538	539	540	601	611
	612	613	614	615	676	686	687	688	689	690	753	763	764	765	766
	767	831	841	842	843	844	845	913	923	924	925	926	927	993	1003
	1004	1005	1006	1007	1077	1087	1088	1089	1090	1091	1152	1162	1163	1164	1165
	1166	1229	1239	1240	1241	1242	1243	1309	1319	1320	1321	1322	1323	1393	1400
	1640	1681	1796	1797	1828	1843	1882	1909							
CLRE	1847	1849	1867												
CMP	491	498	503	508	564	571	576	581	639	646	651	656	714	721	726
	731	791	798	803	808	869	876	881	886	951	958	963	968	1031	1038
	1043	1048	1115	1122	1127	1132	1190	1197	1202	1207	1267	1274	1279	1284	1347
	1354	1359	1364	1498	1501	1518	1520	1522	1524	1526	1528	1660	1701	1769	1812
	1813	1864	1878	1946											
CMPB	1421	1431													
CMPD	586	595	738	747	891	900	907	1055	1064	1071	1214	1223	1371	1380	1387
CMPF	513	520	663	670	813	820	825	975	982	987	1139	1146	1291	1298	1303
COM	1403	1802	1822												
DEC	1495	1810	1834												
DIVD	560	710	865	1027	1186	1343									
DIVF	487	635	787	947	1111	1263									
EMT	391														
HALT	388	396	1530	1761	1906	1933									
INC	1644	1685	1715	1875	1910										
INCB	1434	1439	1858	1862											
IOI	392														
JMP	399	1416	1446	1452	1458	1492	1584	1597	1590	1596					

JSR	468	469	541	542	616	617	691	692	768	769	846	847	928	929	1008
	1009	1092	1093	1167	1168	1244	1245	1324	1325	1412	1534	1546	1558	1562	1598
	1604	1725	1728	1730	1732	1735	1740	1743	1745	1750	1755	1758	1768	1774	1777
	1782	1785	1799	1804	1805	1806	1808	1809	1883						
LDD	551	552	553	559	701	702	703	709	856	857	858	864	1018	1019	1020
	1026	1177	1178	1179	1185	1334	1335	1336	1342	1894	1896	1919	1921	1923	1924
	1925	1926													
LDF	478	479	480	486	626	627	628	634	778	779	780	786	938	939	940
	946	1102	1103	1104	1110	1254	1255	1256	1262						
LDFPS	477	481	550	554	625	629	700	704	777	781	855	859	937	941	1017
	1021	1101	1105	1176	1180	1253	1257	1333	1337	1927					
LDUB	1424														
MOV	430	431	433	436	439	440	441	442	443	444	445	446	447	448	449
	450	451	462	476	482	535	549	555	610	624	630	685	699	705	762
	776	782	840	854	860	922	936	942	1002	1016	1022	1086	1100	1106	1161
	1175	1181	1238	1252	1258	1318	1332	1338	1398	1406	1408	1410	1435	1436	1440
	1443	1448	1449	1450	1451	1454	1455	1456	1457	1460	1461	1464	1465	1466	1467
	1479	1482	1483	1484	1487	1488	1497	1555	1579	1609	1629	1630	1631	1632	1633
	1634	1635	1636	1637	1638	1639	1658	1659	1662	1663	1664	1665	1666	1667	1670
	1671	1672	1673	1674	1675	1676	1677	1678	1679	1680	1699	1700	1703	1704	1705
	1706	1707	1708	1713	1721	1724	1727	1734	1739	1742	1749	1754	1757	1773	1776
	1781	1784	1788	1789	1790	1791	1792	1794	1795	1798	1819	1820	1823	1825	1827
	1845	1846	1869	1880	1886	1887	1898	1899	1900	1901	1902	1903	1904	1905	1908
	1912	1913	1914	1915	1916	1917	1928	1929	1936	1937	1944	1950	1951		
MOVW	1423	1433	1480	1494	1723	1841	1844	1866	1940						
NOP	1413	1414	1415												
ROL	1642	1643	1683	1684	1829	1830	1831	1832	1833	1850	1852	1854			
ROLB	1851	1853	1855												
RTI	1407	1409	1417	1437	1444	1627	1762	1931	1952						
RTS	1462	1468	1503	1556	1581	1610	1668	1709	1770	1786	1815	1839	1870	1884	
SBC	519	592	593	594	669	744	745	746	824	904	905	906	986	1068	1069
	1070	1145	1220	1221	1222	1302	1384	1385	1386						
SETD	1889	1918													
STD	566	590	598	716	742	750	871	895	910	1033	1059	1074	1192	1218	1226
	1349	1375	1390	1890	1891	1892	1893	1895	1897	1920	1922				
STF	493	517	523	641	667	673	793	817	828	953	979	990	1117	1143	1149
	1269	1295	1306	1718											
STFPS	490	563	638	713	790	868	950	1030	1112	1187	1264	1344	1616	1898	
STST	1121	1196	1273	1353	1617										
SUB	518	591	668	743	823	903	985	1067	1144	1219	1301	1383	1583	1592	1722
TRAP	390														
TST	432	488	561	636	711	788	866	948	1028	1113	1198	1265	1345	1441	1489
	1502	1623	1759	1814	1817	1838									
TSTB	1429	1485	1499	1736	1746	1751	1766	1778	1800	1856	1859	1938	1941		
.ASCIZ	1931	1954	1955	1956	1957	1960									
.ASECT	313														
.BLKW	1872														
.END	1977														
.EVEN	1931	1963													
.GLOBL	314														
.LIST	311	361	396	400	454	1395	1445	1628	1710	1763	1840	1885	1931	1935	
.MACR	361														
.MACRO	361														
.NLIST	311	361	396	400	454	1395	1445	1628	1710	1763	1840	1885	1931	1935	
.PAGE	1154	1259	1311	1395											
.REPT	2	396													

MO4

MAINDEC-11-DCFPD-D FLOATING POINT DIVIDE EXERCISER
DCFPD.P11 CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

MACY11 27(732) 16-SEP-76 16:30 PAGE 54

.SSTL	311	361	400	454	1395	1445	1628	1710	1763	1840	1885	1935
.TITLE	312											

.ABS.	011212	000
	000000	001

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

*.DCFPD.SEG/SOL/CRF/PAGNUM+DCFPD
RUN-TIME: 6 12 2 SECONDS
RUN-TIME RATIO: 130/22=5.9
CORE USED: 9K (17 PAGES)

Spooler runtime 7 Seconds, 33 KCS, 189 disk reads, 3 disk writes, 51 pages.

Approved for Release by NSA on 09-08-2013 pursuant to E.O. 13526

[illegible]