

# FP11

ADD and SUBTRACT EXERCISER

## MD-11-DCFPS-D

EP DCFPS-D DL A

OCT 1976

COPYRIGHT ©1976

**digital**

FICHE 1 OF 1

Made In U.S.A.

The microfiche card features a grid of frames on the left side, containing various data and exercises. The frames are arranged in a 10x5 grid. The right side of the card is a large, blank area.

| Frame | Content                    |
|-------|----------------------------|
| 1     | ADD and SUBTRACT EXERCISER |
| 2     | MD-11-DCFPS-D              |
| 3     | EP DCFPS-D DL A            |
| 4     | OCT 1976                   |
| 5     | COPYRIGHT ©1976            |
| 6     | digital                    |
| 7     | FICHE 1 OF 1               |
| 8     | Made In U.S.A.             |
| 9     | ADD and SUBTRACT EXERCISER |
| 10    | MD-11-DCFPS-D              |
| 11    | EP DCFPS-D DL A            |
| 12    | OCT 1976                   |
| 13    | COPYRIGHT ©1976            |
| 14    | digital                    |
| 15    | FICHE 1 OF 1               |
| 16    | Made In U.S.A.             |
| 17    | ADD and SUBTRACT EXERCISER |
| 18    | MD-11-DCFPS-D              |
| 19    | EP DCFPS-D DL A            |
| 20    | OCT 1976                   |
| 21    | COPYRIGHT ©1976            |
| 22    | digital                    |
| 23    | FICHE 1 OF 1               |
| 24    | Made In U.S.A.             |
| 25    | ADD and SUBTRACT EXERCISER |
| 26    | MD-11-DCFPS-D              |
| 27    | EP DCFPS-D DL A            |
| 28    | OCT 1976                   |
| 29    | COPYRIGHT ©1976            |
| 30    | digital                    |
| 31    | FICHE 1 OF 1               |
| 32    | Made In U.S.A.             |
| 33    | ADD and SUBTRACT EXERCISER |
| 34    | MD-11-DCFPS-D              |
| 35    | EP DCFPS-D DL A            |
| 36    | OCT 1976                   |
| 37    | COPYRIGHT ©1976            |
| 38    | digital                    |
| 39    | FICHE 1 OF 1               |
| 40    | Made In U.S.A.             |
| 41    | ADD and SUBTRACT EXERCISER |
| 42    | MD-11-DCFPS-D              |
| 43    | EP DCFPS-D DL A            |
| 44    | OCT 1976                   |
| 45    | COPYRIGHT ©1976            |
| 46    | digital                    |
| 47    | FICHE 1 OF 1               |
| 48    | Made In U.S.A.             |
| 49    | ADD and SUBTRACT EXERCISER |
| 50    | MD-11-DCFPS-D              |
| 51    | EP DCFPS-D DL A            |
| 52    | OCT 1976                   |
| 53    | COPYRIGHT ©1976            |
| 54    | digital                    |
| 55    | FICHE 1 OF 1               |
| 56    | Made In U.S.A.             |
| 57    | ADD and SUBTRACT EXERCISER |
| 58    | MD-11-DCFPS-D              |
| 59    | EP DCFPS-D DL A            |
| 60    | OCT 1976                   |
| 61    | COPYRIGHT ©1976            |
| 62    | digital                    |
| 63    | FICHE 1 OF 1               |
| 64    | Made In U.S.A.             |
| 65    | ADD and SUBTRACT EXERCISER |
| 66    | MD-11-DCFPS-D              |
| 67    | EP DCFPS-D DL A            |
| 68    | OCT 1976                   |
| 69    | COPYRIGHT ©1976            |
| 70    | digital                    |
| 71    | FICHE 1 OF 1               |
| 72    | Made In U.S.A.             |
| 73    | ADD and SUBTRACT EXERCISER |
| 74    | MD-11-DCFPS-D              |
| 75    | EP DCFPS-D DL A            |
| 76    | OCT 1976                   |
| 77    | COPYRIGHT ©1976            |
| 78    | digital                    |
| 79    | FICHE 1 OF 1               |
| 80    | Made In U.S.A.             |
| 81    | ADD and SUBTRACT EXERCISER |
| 82    | MD-11-DCFPS-D              |
| 83    | EP DCFPS-D DL A            |
| 84    | OCT 1976                   |
| 85    | COPYRIGHT ©1976            |
| 86    | digital                    |
| 87    | FICHE 1 OF 1               |
| 88    | Made In U.S.A.             |
| 89    | ADD and SUBTRACT EXERCISER |
| 90    | MD-11-DCFPS-D              |
| 91    | EP DCFPS-D DL A            |
| 92    | OCT 1976                   |
| 93    | COPYRIGHT ©1976            |
| 94    | digital                    |
| 95    | FICHE 1 OF 1               |
| 96    | Made In U.S.A.             |
| 97    | ADD and SUBTRACT EXERCISER |
| 98    | MD-11-DCFPS-D              |
| 99    | EP DCFPS-D DL A            |
| 100   | OCT 1976                   |
| 101   | COPYRIGHT ©1976            |
| 102   | digital                    |
| 103   | FICHE 1 OF 1               |
| 104   | Made In U.S.A.             |
| 105   | ADD and SUBTRACT EXERCISER |
| 106   | MD-11-DCFPS-D              |
| 107   | EP DCFPS-D DL A            |
| 108   | OCT 1976                   |
| 109   | COPYRIGHT ©1976            |
| 110   | digital                    |
| 111   | FICHE 1 OF 1               |
| 112   | Made In U.S.A.             |
| 113   | ADD and SUBTRACT EXERCISER |
| 114   | MD-11-DCFPS-D              |
| 115   | EP DCFPS-D DL A            |
| 116   | OCT 1976                   |
| 117   | COPYRIGHT ©1976            |
| 118   | digital                    |
| 119   | FICHE 1 OF 1               |
| 120   | Made In U.S.A.             |
| 121   | ADD and SUBTRACT EXERCISER |
| 122   | MD-11-DCFPS-D              |
| 123   | EP DCFPS-D DL A            |
| 124   | OCT 1976                   |
| 125   | COPYRIGHT ©1976            |
| 126   | digital                    |
| 127   | FICHE 1 OF 1               |
| 128   | Made In U.S.A.             |
| 129   | ADD and SUBTRACT EXERCISER |
| 130   | MD-11-DCFPS-D              |
| 131   | EP DCFPS-D DL A            |
| 132   | OCT 1976                   |
| 133   | COPYRIGHT ©1976            |
| 134   | digital                    |
| 135   | FICHE 1 OF 1               |
| 136   | Made In U.S.A.             |
| 137   | ADD and SUBTRACT EXERCISER |
| 138   | MD-11-DCFPS-D              |
| 139   | EP DCFPS-D DL A            |
| 140   | OCT 1976                   |
| 141   | COPYRIGHT ©1976            |
| 142   | digital                    |
| 143   | FICHE 1 OF 1               |
| 144   | Made In U.S.A.             |
| 145   | ADD and SUBTRACT EXERCISER |
| 146   | MD-11-DCFPS-D              |
| 147   | EP DCFPS-D DL A            |
| 148   | OCT 1976                   |
| 149   | COPYRIGHT ©1976            |
| 150   | digital                    |
| 151   | FICHE 1 OF 1               |
| 152   | Made In U.S.A.             |
| 153   | ADD and SUBTRACT EXERCISER |
| 154   | MD-11-DCFPS-D              |
| 155   | EP DCFPS-D DL A            |
| 156   | OCT 1976                   |
| 157   | COPYRIGHT ©1976            |
| 158   | digital                    |
| 159   | FICHE 1 OF 1               |
| 160   | Made In U.S.A.             |
| 161   | ADD and SUBTRACT EXERCISER |
| 162   | MD-11-DCFPS-D              |
| 163   | EP DCFPS-D DL A            |
| 164   | OCT 1976                   |
| 165   | COPYRIGHT ©1976            |
| 166   | digital                    |
| 167   | FICHE 1 OF 1               |
| 168   | Made In U.S.A.             |
| 169   | ADD and SUBTRACT EXERCISER |
| 170   | MD-11-DCFPS-D              |
| 171   | EP DCFPS-D DL A            |
| 172   | OCT 1976                   |
| 173   | COPYRIGHT ©1976            |
| 174   | digital                    |
| 175   | FICHE 1 OF 1               |
| 176   | Made In U.S.A.             |
| 177   | ADD and SUBTRACT EXERCISER |
| 178   | MD-11-DCFPS-D              |
| 179   | EP DCFPS-D DL A            |
| 180   | OCT 1976                   |
| 181   | COPYRIGHT ©1976            |
| 182   | digital                    |
| 183   | FICHE 1 OF 1               |
| 184   | Made In U.S.A.             |
| 185   | ADD and SUBTRACT EXERCISER |
| 186   | MD-11-DCFPS-D              |
| 187   | EP DCFPS-D DL A            |
| 188   | OCT 1976                   |
| 189   | COPYRIGHT ©1976            |
| 190   | digital                    |
| 191   | FICHE 1 OF 1               |
| 192   | Made In U.S.A.             |
| 193   | ADD and SUBTRACT EXERCISER |
| 194   | MD-11-DCFPS-D              |
| 195   | EP DCFPS-D DL A            |
| 196   | OCT 1976                   |
| 197   | COPYRIGHT ©1976            |
| 198   | digital                    |
| 199   | FICHE 1 OF 1               |
| 200   | Made In U.S.A.             |



IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DCFPS-D-D  
PRODUCT NAME: P11 ADD AND SUBTRACT 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-DCFPS-D-D  
TABLE OF CONTENTS

## FP11 ADD AND SUBTRACT 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-DCFPS-D-D  
DESCRIPTION

FP11 ADD AND SUBTRACT EXERCISER

PAGE 3

## 1. ABSTRACT

THIS PROGRAM EXERCISES THE FP11 FLOATING POINT ADD AND SUBTRACT INSTRUCTIONS (ADDF, ADD, SUBF, SUBD) 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 SEPERATELY 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.



**E01**

D FLOATING POINT ADD AND SUBTRACT EXERCISER

MACY11 27(732) 17-SEP-76 09:31 PAGE 4

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



MAINDEC-11-DCFPS-D-D  
DESCRIPTION

FP11 ADD AND SUBTRACT EXERCISER

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 UB 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 LOOPED 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



GO1

-0

FLOATING POINT ADD AND SUBTRACT EXERCISER

MACY11 27(732) 17-SEP-76 09:31 PAGE 6

REACHED.

MAINDEC-11-DCFPS-D-D  
DESCRIPTION

FP11 ADD AND SUBTRACT EXERCISER

PAGE 5

#### 5.2.4 TRAPCATCHER

A ".+2" - "HALT" SEQUENCE IS REPEATED FROM 0 - 776 TO CATCH ANY UNEXPECTED TRAPS. THUS ANY UNEXPECTED TRAPS OR INTERRUPTS WILL HALT AT THE VECTOR + 2.

#### 5.2.5 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 5.2.2). IF AN INTERRUPT WAS ANTICIPATED, BUT DIDN'T OCCUR, THE SUBTEST WILL DETECT THE ERROR.

### 6. ERRORS

#### 6.1 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-DCFPS-D-D  
DESCRIPTION

FP11 ADD AND SUBTRACT 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 ADD AND SUBTRACT 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 ADD AND SUBTRACT INSTRUCTIONS COULD CAUSE ERRORS.

.ENDR

00000

.TITLE MAINDEC-11-DCFPS-D FLOATING POINT ADD AND SUBTRACT EXERCISER  
 .ASECT  
 .GLOBL \$ADR,\$ADD,\$SBR,\$SBD,\$ERR  
 :COPYRIGHT 1972, DIGITAL EQUIPMENT CORP., MAYNARD, MASS  
 :PROGRAM BY KEN CHAPMAN

| SWITCH | USE                                                                                 |
|--------|-------------------------------------------------------------------------------------|
| 8      | 0 - LOAD UB REGISTER WITH SW<7:0><br>1 - LOOP ON TEST IN SW<7:0>                    |
| 9      | TTY OUTPUT FORMAT:<br>0 - SIGN, EXPONENT, MANTISSA<br>1 - CORE IMAGE (16 BIT WORDS) |
| 10     | 0 - BELL ON PASS COMPLETE<br>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                     |      |     |                         |



K01

# FLOATING POINT ADD AND SUBTRACT EXERCISER ASSIGNMENTS AND VECTOR AREA

MACY11 27(732) 17-SEP-76 09:31 PAGE 10

```

0000 R0= %0
0001 R1= %1
0002 R2= %2
0003 R3= %3
0004 R4= %4
0005 R5= %5
0005 TTY= %5
0006 SP= %6
0007 PC= %7
0000 AC0= %0
0001 AC1= %1
0002 AC2= %2
0003 AC3= %3
0004 AC4= %4
0005 AC5= %5
0400 SW08= 000400
0000 SW09= 001000
0000 SW10= 002000
0000 SW11= 004000
0000 SW12= 010000
0000 SW13= 020000
0000 SW14= 040000
0000 SW15= 100000
0570 SWR= 177570
0776 PS= 177776
0570 DISPLAY=SWR
0000 DUMMY= HALT
0240 NOP= 240
0400 SCOPE= TRAP
0000 HLT= EMT
0004 TYPE= IOT
0207 BELL= 207

000 . = 0
0200 . = 200

1167 000652 JMP BEG

000 . = 1000
000 ICNT: 0
000 LONUM: DUMMY
000 DUMMY
000 DUMMY
000 DUMMY

000 HINUM: DUMMY
000 DUMMY
000 DUMMY
000 DUMMY

```

;TRAP CATCHER FROM 0 - 776

```

000      ANS1:  DUMMY
000      DUMMY
000      DUMMY
000      DUMMY

000      ANS2:  DUMMY
000      DUMMY
000      DUMMY
000      DUMMY

000      FPS:   0
000      FEC:   0
000      FPC:   0
000      $FPS:  0
000      $FEC:  0
000      $FPC:  0
                                ;FLOATING POINT STATUS
                                ;FLOATING EXCEPTION CODES
                                ;FLOATING PC
                                ;FORTRAN FLOATING POINT STATUS
                                ;FORTRAN FLOATING EXCEPTION CODES
                                ;FORTRAN FLOATING PC

006 000600      BEG:  MOV      #600,SP      ;** STACK AT 600 **
007 001104      000004  MOV      #M1120,2#4  ;FIND OUT WHICH MACHINE THIS IS
007 177772      TST      2#177772          ;IS PIRQ THERE?
007 000006      006004  MOV      #6,YESRT    ;FUDGE IN RTT IF 11/45
003      BR      BEGIN

007 011012      000010  M1120: MOV      FPTADR,2#10 ;LOAD THE ILLEGAL INSTRUCTION VECTOR
                                ; WITH THE ADDRESS OF THE FPU.
                                ; THE FPU WILL HANDLE THE BAD OPCODES

007 000006      000004  BEGIN: MOV      #6,2#4      ;RESET 4
006 000600      MOV      #600,SP
007 007106      000014  MOV      #YESRT,2#14      ;SET TRACE TRAP VECTOR
007 011546      010770  MOV      #POWDWN,2DOWNVEC
007 000340      010764  MOV      #340,2DOWNVEC+2
007 011746      000020  MOV      #.IOT,2#20      ;SET UP VECTOR 20
000 000030      MOV      #30,R0                ;SET R0 TO VECTOR 30
000 010502      MOV      #.TRAP,(0)+          ;SET EMT VECTOR
000 000340      MOV      #340,(0)+
000 007110      MOV      #.EMT,(0)+          ;SET TRAP VECTOR
000 000340      MOV      #340,(0)
007 010150      010716  MOV      #FLTERR,2FPVECT ;LOAD INTERRUPT VECTOR
007 000340      010712  MOV      #340,2FPVECT+2 ;LOCK UP PROCESSOR
007 177560      CLR      ICNT
007 010720      CLR      LAD

```



```

*****
:TEST 1:      EXERCISE ADDF (ADD FLOATING)
:              ALL INTERRUPTS ON
:              ROUNDING MODE
*****

```

```

3
7 010617 010612      SCOPE
7 000000G 006376      MOV      $PLUS, $SIGN+1 ;PUT "+" SIGN IN TYPE-OUT ROUTINE
7 000000G 006424      MOV      #$ADR, .ERR1 ;SELECT PROPER FORTRAN ROUTINES
7 000000G 006530      MOV      #$ADD, .ERR2 ;SELECT PROPER FORTRAN ROUTINES
7 000000G 006546      MOV      #$ADR, .ERR3 ;SELECT PROPER FORTRAN ROUTINES
7 000000G 006546      MOV      #$ADD, .ERR4 ;SELECT PROPER FORTRAN ROUTINES
7 007400 177556      MOV      #007400, $FPS ;SET IE BITS IN FORTRAN ANSWER
7 177554          CLR      $FEC ;CLR FORTRAN FEC
7 177552          CLR      $FPC ;CLR FORTRAN FPC
7 177534          CLR      FPS ;CLR FPU FPS BUFFER
7 177532          CLR      FEC ;CLR FPU FEC BUFFER
7 177530          CLR      FPC ;CLR FPU FPC BUFFER
7 006664          JSR      PC, RANDM2 ;GET RANDOM INPUT DATA
7 005722          JSR      R4, $POLSH ;ENTER POLISH MODE
7          $P.2A      ;PUSH 2 WORDS ON STACK (LONUM)
7          $P.2B      ;PUSH 2 WORDS ON STACK (HINUM)
7          $ADR      ;ADDRESS OF FORTRAN ADD
7          $TST      ;DETERMINE THE CONDITION CODES
7          $POP2X     ;POP 2 WORDS AND EXIT POLISH MODE
7
7 177504          MOV      $FPS, R0 ;DISPLAY FLOATING POINT STATUS
7 040000          LDFPS   #040000 ;LOAD FPS, INTERRUPT DISABLE
7 177426          LDF     LONUM, ACO ;LOAD ACO WITH A RANDOM NUMBER
7 177432          LDF     HINUM, AC1 ;LOAD AC1 WITH A RANDOM NUMBER
7 177446          LDF     ANS2, AC3 ;LOAD AC3 WITH THE SUM
7 007400          LDFPS   #007400 ;TURN INTERRUPTS ON
7 001376 010546      MOV      #.+6, LAD ;RESET LOOP ADDRESS

```

```

*****

```

```

7 177442          RET1:  LDF     ACO, AC2 ;LOAD ACO INTO AC2
7          ADDF     AC1, AC2 ;ADD AC1 BY AC2
7 177426          TST     $FPS ;CHECK FOR FORTRAN FPS ERROR FLAG
7 177422 177426      BMI     ERR1 ;BRANCH IF ERROR FLAG SET
7          STFPS    FPS ;STORE FLOATING POINT STATUS
7 177372          CMP     FPS, $FPS ;CHECK FPS
7          BEQ      TST1 ;BRANCH IF OK
7          STF      AC2, ANS1 ;SAVE FPU ANSWER
7          HLT+1     ;FPS ERROR
7          BR       END1 ;SKIP COMPARE
7
7 177400 177404      ERR1: CFCC ;WAIT FOR FPU TO FINISH
7          CMP      FPS, $FPS ;CHECK THE FLOATING POINT STATUS
7          BEQ      .+6 ;BRANCH IF OK
7          HLT+377   ;FPS ERROR
7          BR       END1 ;SKIP TO END
7
7 177366 177372      CMP     FEC, $FEC ;CHECK THE FLOATING EXCEPTION CODES
7          BEQ      .+6 ;BRANCH IF OK

```

```

                                HLT+377      ;FEC IS WRONG
                                BR          END1 ;SKIP TO END

177354 177360      CMP      FPC,$FPC      ;CHECK FLOATING PC
                   BEQ      TST1         ;BRANCH IF OK
                   HLT+377      ;WRONG ADDRESS IN FPC
                   BR          END1         ;SKIP TO END

                                TST1:      CMPF      AC2,      AC3      ;COMPARE FPU ANSWER TO FORTRAN ANSWER
                                CFCC                     ;COPY FLOATING CONDITION CODES
                                BEQ      END1         ;ANSWERS CHECK
                                ;COMPENSATE FOR FORTRAN INACCURACIES.
177306 177302      STF      AC2,      ANS1      ;SAVE FPU ANSWER
000001             SUB      #1,      ANS1+2      ;DECREMENT FPU ANSWER
177274             SBC      ANS1
177270             CMPF      ANS1,      AC3      ;CHECK ANSWERS AGAIN
                                CFCC                     ;COPY FLOATING CONDITION CODES
                                BEQ      END1         ;BRANCH IF OK
177260             STF      AC2,      ANS1      ;SAVE FPU ANSWER
                                HLT                     ;FPU AND FORTRAN DISAGREE

177272             END1:      CLR      FPS      ;CLR FPU FPS BUFFER
                                SCOPE

;*****
;TEST 2:      EXERCISE ADDD (ADD DOUBLE PRECISION)
;              ALL INTERRUPTS ON
;              ROUNDING MODE
;*****

007600 177270      MOV      #007600,$FPS      ;SET IE BITS IN FORTRAN ANSWER
177266             CLR      $FEC              ;CLR FORTRAN FEC
177264             CLR      $FPC              ;CLR FORTRAN FPC
177246             CLR      FPS              ;CLR FPU FPS BUFFER
177244             CLR      FEC              ;CLR FPU FEC BUFFER
177242             CLR      FPC              ;CLR FPU FPC BUFFER
006534             JSR      PC,      RANDM4    ;GET RANDOM INPUT DATA
005434             JSR      R4,      $POLSH    ;ENTER POLISH MODE
                                $P.4A          ;PUSH 4 WORDS ON STACK (LONUM)
                                $P.4B          ;PUSH 4 WORDS ON STACK (HINUM)
                                $ADD           ;ADDRESS OF FORTRAN ADD
                                $TST           ;DETERMINE THE CONDITION CODES
                                $POP4X         ;POP 4 WORDS AND EXIT POLISH MODE

177216             MOV      $FPS,      R0      ;DISPLAY FLOATING POINT STATUS
040200             LDFPS      #040200        ;LOAD FPS, INTERRUPT DISABLE AND FD
177140             LDD      LONUM,      ACO    ;LOAD ACO WITH A RANDOM NUMBER
177144             LDD      HINUM,      AC1    ;LOAD AC1 WITH A RANDOM NUMBER
177160             LDD      ANS2,      AC3    ;LOAD AC3 WITH THE SUM
007600             LDFPS      #007600        ;TURN INTERRUPTS ON
001664 010260      MOV      #.+6,      LAD    ;RESET LOOP ADDRESS

;*****
RET2:      LDD      ACO,      AC2      ;LOAD ACO INTO AC2
           ADDD      AC1,      AC2      ;ADD AC1 BY AC2

```



|        |        |        |        |         |           |          |                                       |
|--------|--------|--------|--------|---------|-----------|----------|---------------------------------------|
| 001674 | 100412 |        |        | TST     | \$FPS     |          | :CHECK FOR FORTRAN FPS ERROR FLAG     |
| 001676 | 170000 | 177140 |        | BR      |           |          | :BRANCH IF ERROR FLAG SET             |
| 001700 | 026767 | 177134 | 177140 | STP     |           |          | :STORE FLOATING POINT STATUS          |
| 001710 | 001402 |        |        | CMF     | \$FPS     |          | :CHECK FPS                            |
| 001712 | 174267 | 177104 |        | BEQ     |           |          | :BRANCH IF OK                         |
| 001716 | 104001 |        |        | STD     | ANS1      |          | :SAVE FPU ANSWER                      |
| 001720 | 000450 |        |        | HLT+1   |           |          | :FPS ERROR                            |
|        |        |        |        | BR      | END2      |          | :SKIP TO END                          |
| 001722 | 170000 |        |        | ERR2:   | CFCC      |          | :WAIT FOR FPU TO FINISH               |
| 001724 | 026767 | 177112 | 177116 | CMF     | FPS,      | \$FPS    | :CHECK THE FLOATING POINT STATUS      |
| 001732 | 001402 |        |        | BEQ     | +.6       |          | :BRANCH IF OK                         |
| 001734 | 104377 |        |        | HLT+377 |           |          | :FPS ERROR                            |
| 001736 | 000441 |        |        | BR      | END2      |          | :SKIP TO END                          |
| 001740 | 026767 | 177100 | 177104 | CMF     | FEC,\$FEC |          | :CHECK THE FLOATING EXCEPTION CODES   |
| 001746 | 001402 |        |        | BEQ     | +.6       |          | :BRANCH IF OK                         |
| 001750 | 104377 |        |        | HLT+377 |           |          | :FEC IS WRONG                         |
| 001752 | 000433 |        |        | BR      | END2      |          | :SKIP TO END                          |
| 001754 | 026767 | 177066 | 177072 | CMF     | FPC,\$FPC |          | :CHECK FLOATING PC                    |
| 001762 | 001402 |        |        | BEQ     | TST2      |          | :BRANCH IF OK                         |
| 001764 | 104377 |        |        | HLT+377 |           |          | :WRONG ADDRESS IN FPC                 |
| 001766 | 000425 |        |        | BR      | END2      |          | :SKIP TO END                          |
| 001770 | 173702 |        |        | TST2:   | CMFD      | AC2, AC3 | :COMPARE FPU ANSWER TO FORTRAN ANSWER |
| 001772 | 170000 |        |        | CFCC    |           |          | :COPY FLOATING CONDITION CODES        |
| 001774 | 001422 |        |        | BEQ     | END2      |          | :ANSWERS CHECK                        |
| 001776 | 174267 | 177020 |        |         |           |          | :COMPENSATE FOR FORTRAN INACCURACIES. |
| 002002 | 162767 | 000001 | 177020 | STD     | AC2,      | ANS1     | :SAVE FPU ANSWER                      |
| 002010 | 005667 | 177012 |        | SUB     | #1,       | ANS1+.6  | :DECREMENT FPU ANSWER                 |
| 002014 | 005667 | 177004 |        | SBC     | ANS1+.4   |          |                                       |
| 002020 | 005667 | 176776 |        | SBC     | ANS1+.2   |          |                                       |
| 002024 | 173767 | 176772 |        | SBC     | ANS1      |          |                                       |
| 002030 | 170000 |        |        | CMFD    | ANS1,     | AC3      | :CHECK ANSWERS AGAIN                  |
| 002032 | 001403 |        |        | CFCC    |           |          | :COPY FLOATING CONDITION CODES        |
| 002034 | 174267 | 176762 |        | BEQ     | END2      |          | :BRANCH IF OK                         |
| 002040 | 104000 |        |        | STD     | AC2,      | ANS1     | :SAVE FPU ANSWER                      |
|        |        |        |        | HLT     |           |          | :FPU AND FORTRAN DISAGREE             |
| 002042 | 005067 | 176774 |        | END2:   | CLR       | FPS      | :CLR FPU FPS BUFFER                   |
| 002046 | 104400 |        |        | SCOPE   |           |          |                                       |

```

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

```

|        |        |        |        |     |               |        |                                |
|--------|--------|--------|--------|-----|---------------|--------|--------------------------------|
| 002050 | 012767 | 004400 | 176772 | MOV | #004400,\$FPS |        | :SET IE BITS IN FORTRAN ANSWER |
| 002056 | 005067 | 176770 |        | CLR | \$FEC         |        | :CLR FORTRAN FEC               |
| 002062 | 005067 | 176766 |        | CLR | \$FPC         |        | :CLR FORTRAN FPC               |
| 002066 | 005067 | 176750 |        | CLR | FPS           |        | :CLR FPU FPS BUFFER            |
| 002072 | 005067 | 176746 |        | CLR | FEC           |        | :CLR FPU FEC BUFFER            |
| 002076 | 005067 | 176744 |        | CLR | FPC           |        | :CLR FPU FPC BUFFER            |
| 002102 | 004767 | 006100 |        | JSR | PC,           | RANDM2 | :GET RANDOM INPUT DATA         |

```

002106 004467 005135      JSR      R4,      $POLSH ;ENTER POLISH MODE
002112 007262              $P.2A ;PUSH 2 WORDS ON STACK (LONUM)
002114 007304              $P.2B ;PUSH 2 WORDS ON STACK (HINUM)
002116 000000G            $ADR      ;ADDRESS OF FORTRAN ADD
002120 007352              $TST      ;DETERMINE THE CONDITION CODES
002122 007316              $POP2X   ;POP 2 WORDS AND EXIT POLISH MODE

002124 016700 176720      MOV      $FPS,   R0 ;DISPLAY FLOATING POINT STATUS
002130 170127 040000      LDFPS    #040000 ;LOAD FPS, INTERRUPT DISABLE
002134 172467 176642      LDF      LONUM,  AC0 ;LOAD AC0 WITH A RANDOM NUMBER
002140 172567 176646      LDF      HINUM,  AC1 ;LOAD AC1 WITH A RANDOM NUMBER
002144 172767 176662      LDF      ANS2,   AC3 ;LOAD AC3 WITH THE SUM
002150 170127 004400      LDFPS    #004400 ;TURN INTERRUPTS ON, EXCEPT OVERFLOW AND UNDERFLOW
002154 012767 002162 007762 MOV      #. +6,  LAD ;RESET LOOP ADDRESS

```

\*\*\*\*\*

```

002162 172600              LDF      AC0,   AC2 ;LOAD AC0 INTO AC2
002164 172201              ADDF     AC1,   AC2 ;ADD AC1 BY AC2
002166 170267 176650      STFPS    FPS,   $FPS ;STORE FLOATING POINT STATUS
002172 026767 176644 176650 CMP      FPS,   $FPS ;CHECK FPS
002200 001404              BEQ      TST3,   ;BRANCH IF OK
002202 174267 176614      STF      AC2,   ANS1 ;SAVE FPU ANSWER
002206 104001              HLT+1    ;FPS ERROR
002210 000425              BR       END3 ;SKIP COMPARE

002212 032767 000002 176630 TST3:  BIT      #2,   $FPS ;CHECK FOR OVERFLOW
002220 001021              BNE      END3 ;BRANCH IF OVERFLOW
002222 173702              CMPF     AC2,   AC3 ;COMPARE FPU ANSWER TO FORTRAN ANSWER
002224 170000              CFCC      ;COPY FLOATING CONDITION CODES
002226 001416              BEQ      END3 ;ANSWERS CHECK
;COMPENSATE FOR FORTRAN INACCURACIES.
002230 174267 176566      STF      AC2,   ANS1 ;SAVE FPU ANSWER
002234 162767 000001 176562 SUB      #1,   ANS1+2 ;DECREMENT FPU ANSWER
002242 005667 176554      SBC      ANS1,  AC3 ;CHECK ANSWERS AGAIN
002246 173767 176550      CMPF     ANS1,  AC3 ;COPY FLOATING CONDITION CODES
002252 170000              CFCC      ;BRANCH IF OK
002254 001403              BEQ      END3 ;SAVE FPU ANSWER
002256 174267 176540      STF      AC2,   ANS1 ;FPU AND FORTRAN DISAGREE
002262 104000              HLT

002264 005067 176552      END3:  CLR      FPS ;CLR FPU FPS BUFFER
002270 104400              SCOPE

```

\*\*\*\*\*  
 ;TEST 4: EXERCISE ADD (ADD DOUBLE PRECISION)  
 ; OVERFLOW AND UNDERFLOW INTERRUPTS OFF  
 ; ROUNDING MODE  
 \*\*\*\*\*

```

002272 012767 004600 176550      MOV      #004600, $FPS ;SET IE BITS IN FORTRAN ANSWER
002300 005067 176546      CLR      $FEC ;CLR FORTRAN FEC
002304 005067 176544      CLR      $FPC ;CLR FORTRAN FPC
002310 005067 176526      CLR      FPS ;CLR FPU FPS BUFFER
002314 005067 176524      CLR      FEC ;CLR FPU FEC BUFFER
002320 005067 176522      CLR      FPC ;CLR FPU FPC BUFFER

```



|        |         |        |        |         |         |         |                                                |
|--------|---------|--------|--------|---------|---------|---------|------------------------------------------------|
| 002324 | 004767  | 006014 |        | JSR     | PC,     | RANDOM4 | :GET RANDOM INPUT DATA                         |
| 002330 | 004467  | 004714 |        | JSR     | R4,     | \$POLSH | :ENTER POLISH MODE                             |
| 002334 | 007252  |        |        | \$P.4A  |         |         | :PUSH 4 WORDS ON STACK (LONUM)                 |
| 002336 | 007274  |        |        | \$P.4B  |         |         | :PUSH 4 WORDS ON STACK (HINUM)                 |
| 002340 | 000000G |        |        | \$ADD   |         |         | :ADDRESS OF FORTRAN ADD                        |
| 002342 | 007352  |        |        | \$TST   |         |         | :DETERMINE THE CONDITION CODES                 |
| 002344 | 007330  |        |        | \$POP4X |         |         | :POP 4 WORDS AND EXIT POLISH MODE              |
| 002346 | 016700  | 176476 |        | MOV     | \$FPS,  | RO      | :DISPLAY FLOATING POINT STATUS                 |
| 002352 | 170127  | 040200 |        | LDFPS   | #040200 |         | :LOAD FPS, INTERRUPT DISABLE AND FD            |
| 002356 | 172467  | 176420 |        | LDD     | LONUM,  | AC0     | :LOAD AC0 WITH A RANDOM NUMBER                 |
| 002362 | 172567  | 176424 |        | LDD     | HINUM,  | AC1     | :LOAD AC1 WITH A RANDOM NUMBER                 |
| 002366 | 172767  | 176440 |        | LDD     | ANS2,   | AC3     | :LOAD AC3 WITH THE SUM                         |
| 002372 | 170127  | 004600 |        | LDFPS   | #004600 |         | :TURN INTERRUPTS ON, EXCEPT OVER AND UNDERFLOW |
| 002376 | 012767  | 002404 | 007540 | MOV     | #.+6,   | LAD     | :RESET LOOP ADDRESS                            |

:\*\*\*\*\*

|        |        |        |        |       |        |        |                                       |
|--------|--------|--------|--------|-------|--------|--------|---------------------------------------|
| 002404 | 172600 |        |        | LDD   | AC0,   | AC2    | :LOAD AC0 INTO AC2                    |
| 002406 | 172201 |        |        | ADDD  | AC1,   | AC2    | :ADD AC1 BY AC2                       |
| 002410 | 170267 | 176426 |        | STFPS | FPS    |        | :STORE FLOATING POINT STATUS          |
| 002414 | 026767 | 176422 | 176426 | CMP   | FPS,   | \$FPS  | :CHECK FPS                            |
| 002422 | 001404 |        |        | BEQ   | TST4   |        | :BRANCH IF OK                         |
| 002424 | 174267 | 176372 |        | STD   | AC2,   | ANS1   | :SAVE FPU ANSWER                      |
| 002430 | 104001 |        |        | HLT+1 |        |        | :FPS ERROR                            |
| 002432 | 000431 |        |        | BR    | END4   |        | :SKIP COMPARE                         |
| 002434 | 032767 | 000002 | 176406 | TST4: | BIT    | #2,    | \$FPS                                 |
| 002442 | 001025 |        |        | BNE   | END4   |        | :CHECK FOR OVERFLOW                   |
| 002444 | 173702 |        |        | CMPD  | AC2,   | AC3    | :BRANCH IF OVERFLOW                   |
| 002446 | 170000 |        |        | CFCC  |        |        | :COMPARE FPU ANSWER TO FORTRAN ANSWER |
| 002450 | 001422 |        |        | BEQ   | END4   |        | :COPY FLOATING CONDITION CODES        |
| 002452 | 174267 | 176344 |        | STD   | AC2,   | ANS1   | :ANSWERS CHECK                        |
| 002456 | 162767 | 000001 | 176344 | SUB   | #1,    | ANS1+6 | :COMPENSATE FOR FORTRAN INACCURACIES. |
| 002464 | 005667 | 176336 |        | SBC   | ANS1+4 |        | :SAVE FPU ANSWER                      |
| 002470 | 005667 | 176330 |        | SBC   | ANS1+2 |        | :DECREMENT FPU ANSWER                 |
| 002474 | 005667 | 176322 |        | SBC   | ANS1   |        |                                       |
| 002500 | 173767 | 176316 |        | CMPD  | ANS1,  | AC3    | :CHECK ANSWERS AGAIN                  |
| 002504 | 170000 |        |        | CFCC  |        |        | :COPY FLOATING CONDITION CODES        |
| 002506 | 001403 |        |        | BEQ   | END4   |        | :BRANCH IF OK                         |
| 002510 | 174267 | 176306 |        | STD   | AC2,   | ANS1   | :SAVE FPU ANSWER                      |
| 002514 | 104000 |        |        | HLT   |        |        | :FPU AND FORTRAN DISAGREE             |
| 002516 | 005067 | 176320 |        | END4: | CLR    | FPS    | :CLR FPU FPS BUFFER                   |
| 002522 | 104400 |        |        | SCOPE |        |        |                                       |

:\*\*\*\*\*  
:TEST 5: EXERCISE ADDF (ADD FLOATING)  
: ALL INTERRUPTS ON  
: TRUNCATE MODE  
:\*\*\*\*\*

|        |        |        |        |     |               |  |                                |
|--------|--------|--------|--------|-----|---------------|--|--------------------------------|
| 002524 | 012767 | 007440 | 176316 | MOV | #007440,\$FPS |  | :SET IE BITS IN FORTRAN ANSWER |
| 002532 | 005067 | 176314 |        | CLR | \$FEC         |  | :CLR FORTRAN FEC               |
| 002536 | 005067 | 176312 |        | CLR | \$FPC         |  | :CLR FORTRAN FPC               |

|        |         |        |        |         |          |         |                                   |
|--------|---------|--------|--------|---------|----------|---------|-----------------------------------|
| 002542 | 005067  | 176274 |        | CLR     | FPS      |         | :CLR FPU FPS BUFFER               |
| 002546 | 005067  | 176272 |        | CLR     | FEC      |         | :CLR FPU FEC BUFFER               |
| 002552 | 005067  | 176270 |        | CLR     | FPC      |         | :CLR FPU FPC BUFFER               |
| 002556 | 004767  | 005424 |        | JSR     | PC,      | RANDM2  | :GET RANDOM INPUT DATA            |
| 002562 | 004467  | 004462 |        | JSR     | R4,      | \$POLSH | :ENTER POLISH MODE                |
| 002566 | 007262  |        |        | \$P.2A  |          |         | :PUSH 2 WORDS ON STACK (LONUM)    |
| 002570 | 007304  |        |        | \$P.2B  |          |         | :PUSH 2 WORDS ON STACK (HINUM)    |
| 002572 | 000000G |        |        | \$ADR   |          |         | :ADDRESS OF FORTRAN ADD           |
| 002574 | 007352  |        |        | \$TST   |          |         | :DETERMINE THE CONDITION CODES    |
| 002576 | 007316  |        |        | \$POP2X |          |         | :POP 2 WORDS AND EXIT POLISH MODE |
| 002600 | 016700  | 176244 |        | MOV     | \$FPS,   | RO      | :DISPLAY FLOATING POINT STATUS    |
| 002604 | 170127  | 040000 |        | LDFPS   | \$040000 |         | :LOAD FPS, INTERRUPT DISABLE      |
| 002610 | 172467  | 176166 |        | LDF     | LONUM,   | ACC     | :LOAD ACC WITH A RANDOM NUMBER    |
| 002614 | 172567  | 176172 |        | LDF     | HINUM,   | AC1     | :LOAD AC1 WITH A RANDOM NUMBER    |
| 002620 | 172767  | 176206 |        | LDF     | ANS2,    | AC3     | :LOAD AC3 WITH THE SUM            |
| 002624 | 170127  | 007440 |        | LDFPS   | \$007440 |         | :TURN INTERRUPTS ON               |
| 002630 | 012767  | 002636 | 007306 | MOV     | #+6,     | LAD     | :RESET LOOP ADDRESS               |

:\*\*\*\*\*

|        |        |        |        |                         |           |        |                                       |
|--------|--------|--------|--------|-------------------------|-----------|--------|---------------------------------------|
| 002636 | 172600 |        |        | LDF                     | ACC,      | AC2    | :LOAD ACC INTO AC2                    |
| 002640 | 172201 |        |        | ADDF                    | AC1,      | AC2    | :ADD AC1 BY AC2                       |
| 002642 | 005767 | 176202 |        | TST                     | \$FPS     |        | :CHECK FOR FORTRAN FPS ERROR FLAG     |
| 002646 | 100412 |        |        | BMI                     | ERR5      |        | :BRANCH IF ERROR FLAG SET             |
| 002650 | 170267 | 176166 |        | STFPS                   | FPS       |        | :STORE FLOATING POINT STATUS          |
| 002654 | 026767 | 176162 | 176166 | CMP                     | FPS,      | \$FPS  | :CHECK FPS                            |
| 002662 | 001427 |        |        | BEQ                     | TST5      |        | :BRANCH IF OK                         |
| 002664 | 174267 | 176132 |        | STF                     | AC2,      | ANS1   | :SAVE FPU ANSWER                      |
| 002670 | 104001 |        |        | HLT+1                   |           |        | :FPS ERROR                            |
| 002672 | 000455 |        |        | BR                      | END5      |        | :SKIP COMPARE                         |
| 002674 | 170000 |        |        | CFCC                    |           |        | :WAIT FOR FPU TO FINISH               |
| 002676 | 026767 | 176140 | 176144 | CMP                     | FPS,      | \$FPS  | :CHECK THE FLOATING POINT STATUS      |
| 002704 | 001402 |        |        | BEQ                     | #+6       |        | :BRANCH IF OK                         |
| 002706 | 104377 |        |        | HLT+377                 |           |        | :FPS ERROR                            |
| 002710 | 000446 |        |        | BR                      | END5      |        | :SKIP TO END                          |
| 002712 | 026767 | 176126 | 176132 | CMP                     | FEC,\$FEC |        | :CHECK THE FLOATING EXCEPTION CODES   |
| 002720 | 001402 |        |        | BEQ                     | #+6       |        | :BRANCH IF OK                         |
| 002722 | 104377 |        |        | HLT+377                 |           |        | :FEC IS WRONG                         |
| 002724 | 000440 |        |        | BR                      | END5      |        | :SKIP TO END                          |
| 002726 | 026767 | 176114 | 176120 | CMP                     | FPC,\$FPC |        | :CHECK FLOATING PC                    |
| 002734 | 001402 |        |        | BEQ                     | TST5      |        | :BRANCH IF OK                         |
| 002736 | 104377 |        |        | HLT+377                 |           |        | :WRONG ADDRESS IN FPC                 |
| 002740 | 000432 |        |        | BR                      | END5      |        | :SKIP TO END                          |
| 002742 | 173702 |        |        | CMPE                    | AC2,      | AC3    | :COMPARE FPU ANSWER TO FORTRAN ANSWER |
| 002744 | 170000 |        |        | CFCC                    |           |        | :COPY FLOATING CONDITION CODES        |
| 002746 | 001427 |        |        | BEQ                     | END5      |        | :ANSWERS CHECK                        |
| 002750 | 174267 | 176046 |        | :COMPENSATE FOR FORTRAN |           |        | :INACCURACIES.                        |
| 002754 | 062767 | 000001 | 176042 | STF                     | AC2,      | ANS1   | :SAVE FPU ANSWER                      |
| 002762 | 005567 | 176034 |        | ADD                     | #1,       | ANS1+2 | :INCREMENT FPU ANSWER                 |
| 002766 | 173767 | 176030 |        | ADC                     | ANS1      |        |                                       |
|        |        |        |        | CMPE                    | ANS1,     | AC3    | :CHECK ANSWERS AGAIN                  |



|        |        |        |        |       |       |        |                                |
|--------|--------|--------|--------|-------|-------|--------|--------------------------------|
| 002772 | 170000 |        |        | CFCC  |       |        | ;COPY FLOATING CONDITION CODES |
| 002774 | 001414 |        |        | BEQ   | END5  |        | ;BRANCH IF OK                  |
| 002776 | 162767 | 000002 | 176020 | SUB   | #2,   | ANS1+2 | ;DECREMENT FPU ANSWER          |
| 003004 | 005667 | 176012 |        | SBC   | ANS1  |        |                                |
| 003010 | 173767 | 176006 |        | CMPF  | ANS1, | AC3    | ;CHECK ANSWERS AGAIN           |
| 003014 | 170000 |        |        | CFCC  |       |        | ;COPY FLOATING CONDITION CODES |
| 003016 | 001403 |        |        | BEQ   | END5  |        | ;BRANCH IF OK                  |
| 003020 | 174267 | 175776 |        | STF   | AC2,  | ANS1   | ;SAVE FPU ANSWER               |
| 003024 | 104000 |        |        | HLT   |       |        | ;FPU AND FORTRAN DISAGREE      |
| 003026 | 005067 | 176010 | END5:  | CLR   | FPS   |        | ;CLR FPU FPS BUFFER            |
| 003032 | 104400 |        |        | SCOPE |       |        |                                |

```

:*****
:TEST 6:      EXERCISE ADDD (ADD DOUBLE PRECISION)
:              ALL INTERRUPTS ON
:              TRUNCATE MODE
:*****

```

|        |         |        |        |         |               |                                     |
|--------|---------|--------|--------|---------|---------------|-------------------------------------|
| 003034 | 012767  | 007640 | 175006 | MOV     | #007640,\$FPS | :SET IE BITS IN FORTRAN ANSWER      |
| 003042 | 005067  | 176004 |        | CLR     | \$FEC         | :CLR FORTRAN FEC                    |
| 003046 | 005067  | 176002 |        | CLR     | \$FPC         | :CLR FORTRAN FPC                    |
| 003052 | 005067  | 175764 |        | CLR     | FPS           | :CLR FPU FPS BUFFER                 |
| 003056 | 005067  | 175762 |        | CLR     | FEC           | :CLR FPU FEC BUFFER                 |
| 003062 | 005067  | 175760 |        | CLR     | FPC           | :CLR FPU FPC BUFFER                 |
| 003066 | 004767  | 005252 |        | JSR     | PC,           | :GET RANDOM INPUT DATA              |
| 003072 | 004467  | 004152 |        | JSR     | R4,           | :ENTER POLISH MODE                  |
| 003076 | 007252  |        |        | \$P.4A  |               | :PUSH 4 WORDS ON STACK (LONUM)      |
| 003100 | 007274  |        |        | \$P.4B  |               | :PUSH 4 WORDS ON STACK (HINUM)      |
| 003102 | 000000G |        |        | \$ADD   |               | :ADDRESS OF FORTRAN ADD             |
| 003104 | 007352  |        |        | \$TST   |               | :DETERMINE THE CONDITION CODES      |
| 003106 | 007330  |        |        | \$POP4X |               | :POP 4 WORDS AND EXIT POLISH MODE   |
| 003110 | 016700  | 175734 |        | MOV     | \$FPS, R0     | :DISPLAY FLOATING POINT STATUS      |
| 003114 | 170127  | 040200 |        | LDFPS   | #040200       | :LOAD FPS, INTERRUPT DISABLE AND FD |
| 003120 | 172467  | 175656 |        | LDD     | LONUM, AC0    | :LOAD AC0 WITH A RANDOM NUMBER      |
| 003124 | 172567  | 175662 |        | LDD     | HINUM, AC1    | :LOAD AC1 WITH A RANDOM NUMBER      |
| 003130 | 172767  | 175676 |        | LDD     | ANS2, AC3     | :LOAD AC3 WITH THE SUM              |
| 003134 | 170127  | 007640 |        | LDFPS   | #007640       | :TURN INTERRUPTS ON                 |
| 003140 | 012767  | 003146 | 006776 | MOV     | \$.+6, LAD    | :RESET LOOP ADDRESS                 |

|        |        |        |       |       |       |                                   |
|--------|--------|--------|-------|-------|-------|-----------------------------------|
| 003146 | 172600 |        | LDD   | AC0,  | AC2   | :LOAD AC0 INTO AC2                |
| 003150 | 172201 |        | ADD   | AC1,  | AC2   | :ADD AC1 BY AC2                   |
| 003152 | 005767 | 175672 | TST   | \$FPS |       | :CHECK FOR FORTRAN FPS ERROR FLAG |
| 003156 | 100412 |        | BMI   | ERR6  |       | :BRANCH IF ERROR FLAG SET         |
| 003160 | 170267 | 175656 | STFPS | FPS   |       | :STORE FLOATING POINT STATUS      |
| 003164 | 026767 | 175652 | CMP   | FPS,  | \$FPS | :CHECK FPS                        |
| 003172 | 001427 |        | BEQ   | TST6  |       | :BRANCH IF OK                     |
| 003174 | 174267 | 175622 | STD   | AC2,  | ANS1  | :SAVE FPU ANSWER                  |
| 003200 | 104001 |        | HLT+1 |       |       | :FPS ERROR                        |
| 003202 | 000465 |        | BR    | END6  |       | :SKIP TO END                      |
| 003204 | 170000 |        | ERR6: | CF00  |       | :WAIT FOR FPU TO FINISH           |

```

003206 026767 175630 175634      CMP      FPS,      $FPS      ;CHECK THE FLOATING POINT STATUS
003214 001402                      BEQ      .+6          ;BRANCH IF OK
003216 104377                      HLT+377       ;FPS ERROR
003220 000456                      BR      END6        ;SKIP TO END

003222 026767 175616 175622      CMP      FEC,$FEC      ;CHECK THE FLOATING EXCEPTION CODES
003230 001402                      BEQ      .+6          ;BRANCH IF OK
003232 104377                      HLT+377       ;FEC IS WRONG
003234 000450                      BR      END6        ;SKIP TO END

003236 026767 175604 175610      CMP      FPC,$FPC      ;CHECK FLOATING PC
003244 001402                      BEQ      TST6       ;BRANCH IF OK
003246 104377                      HLT+377       ;WRONG ADDRESS IN FPC
003250 000442                      BR      END6        ;SKIP TO END

003252 173702                      TST6:  CMPD     AC2,      AC3      ;COMPARE FPU ANSWER TO FORTRAN ANSWER
003254 170000                      CFCC                      ;COPY FLOATING CONDITION CODES
003256 001437                      BEQ      END6        ;ANSWERS CHECK
                                ;COMPENSATE FOR FORTRAN INACCURACIES.
003260 174267 175536              STD      AC2,      ANS1      ;SAVE FPU ANSWER
003264 062767 000001 175536      ADD      #1,      ANS1+6    ;INCREMENT FPU ANSWER
003272 005567 175530              ADC      ANS1+4
003276 005567 175522              ADC      ANS1+2
003302 005567 175514              ADC      ANS1
003306 173767 175510              CMPD     ANS1,      AC3      ;CHECK ANSWERS AGAIN
003312 170000                      CFCC                      ;COPY FLOATING CONDITION CODES
003314 001420                      BEQ      END6        ;BRANCH IF OK
003316 162767 000002 175504      SUB      #2,      ANS1+6    ;DECREMENT FPU ANSWER
003324 005667 175476              SBC      ANS1+4
003330 005667 175470              SBC      ANS1+2
003334 005667 175462              SBC      ANS1
003340 173767 175456              CMPD     ANS1,      AC3      ;CHECK ANSWERS AGAIN
003344 170000                      CFCC                      ;COPY FLOATING CONDITION CODES
003346 001403                      BEQ      END6        ;BRANCH IF OK
003350 174267 175446              STD      AC2,      ANS1      ;SAVE FPU ANSWER
003354 104000                      HLT                      ;FPU AND FORTRAN DISAGREE

003356 005067 175460              ENDB:  CLR      FPS          ;CLR FPU FPS BUFFER
003362 104400                      SCOPE

```

```

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

```

```

003364 012767 004440 175456      MOV      #004440,$FPS      ;SET IE BITS IN FORTRAN ANSWER
003372 005067 175454              CLR      $FEC          ;CLR FORTRAN FEC
003376 005067 175452              CLR      $FPC          ;CLR FORTRAN FPC
003402 005067 175434              CLR      FPS          ;CLR FPU FPS BUFFER
003406 005067 175432              CLR      FEC          ;CLR FPU FEC BUFFER
003412 005067 175430              CLR      FPC          ;CLR FPU FPC BUFFER
003416 004767 004564              JSR      PC,      RANDM2    ;GET RANDOM INPUT DATA
003422 004467 003622              JSR      R4,      $POLSH    ;ENTER POLISH MODE
003426 007262                      $P.2A          ;PUSH 2 WORDS ON STACK (LONUM)

```



• \*\*\*\*\*

```

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

```

```
003630 012767 004640 175212      MOV      #004640,$FPS      ;SET IE BITS IN FORTRAN ANSWER
003636 005067 175210      CLR      $FEC          ;CLR FORTRAN FEC
003642 005067 175206      CLR      $FPC          ;CLR FORTRAN FPC
```

|        |         |        |        |         |         |         |                                                |
|--------|---------|--------|--------|---------|---------|---------|------------------------------------------------|
| 003646 | 005067  | 175170 |        | CLR     | FPS     |         | :CLR FPU FPS BUFFER                            |
| 003652 | 005067  | 175166 |        | CLR     | FEC     |         | :CLR FPU FEC BUFFER                            |
| 003656 | 005067  | 175164 |        | CLR     | FPC     |         | :CLR FPU FPC BUFFER                            |
| 003662 | 004767  | 004456 |        | JSR     | PC,     | RANDOM4 | :GET RANDOM INPUT DATA                         |
| 003666 | 004467  | 003356 |        | JSR     | R4,     | \$POLSH | :ENTER POLISH MODE                             |
| 003672 | 007252  |        |        | \$P.4A  |         |         | :PUSH 4 WORDS ON STACK (LONUM)                 |
| 003674 | 007274  |        |        | \$P.4B  |         |         | :PUSH 4 WORDS ON STACK (HINUM)                 |
| 003676 | 000000G |        |        | \$ADD   |         |         | :ADDRESS OF FORTRAN ADD                        |
| 003700 | 007352  |        |        | \$TST   |         |         | :DETERMINE THE CONDITION CODES                 |
| 003702 | 007330  |        |        | \$POP4X |         |         | :POP 4 WORDS AND EXIT POLISH MODE              |
|        |         |        |        |         |         |         |                                                |
| 003704 | 016700  | 175140 |        | MOV     | \$FPS,  | RD      | :DISPLAY FLOATING POINT STATUS                 |
| 003710 | 170127  | 040200 |        | LDFPS   | #040200 |         | :LOAD FPS, INTERRUPT DISABLE AND FD            |
| 003714 | 172467  | 175062 |        | LDD     | LONUM,  | AC0     | :LOAD AC0 WITH A RANDOM NUMBER                 |
| 003720 | 172567  | 175066 |        | LDD     | HINUM,  | AC1     | :LOAD AC1 WITH A RANDOM NUMBER                 |
| 003724 | 172767  | 175102 |        | LDD     | ANS2,   | AC3     | :LOAD AC3 WITH THE SUM                         |
| 003730 | 170127  | 004640 |        | LDFPS   | #004640 |         | :TURN INTERRUPTS ON, EXCEPT OVER AND UNDERFLOW |
| 003734 | 012767  | 003742 | 006202 | MOV     | #.+6,   | LAD     | :RESET LOOP ADDRESS                            |

:\*\*\*\*\*

|        |        |        |        |        |                 |         |                                |
|--------|--------|--------|--------|--------|-----------------|---------|--------------------------------|
| 003742 | 172600 |        |        | LDD    | PC0,            | AC2     | :LOAD AC0 INTO AC2             |
| 003744 | 172201 |        |        | ADD    | AC1,            | AC2     | :ADD AC1 BY AC2                |
| 003746 | 170267 | 175070 |        | STFPS  | FPS             |         | :STORE FLOATING POINT STATUS   |
| 003752 | 026767 | 175064 | 175070 | CMP    | FPS,            | \$FPS   | :CHECK FPS                     |
| 003760 | 001404 |        |        | BEQ    | TST10           |         | :BRANCH IF OK                  |
| 003762 | 174267 | 175034 |        | STD    | AC2,            | ANS1    | :SAVE FPU ANSWER               |
| 003766 | 104001 |        |        | HLT+1  |                 |         | :FPS ERROR                     |
| 003770 | 000446 |        |        | BR     | END10           |         | :SKIP COMPARE                  |
|        |        |        |        |        |                 |         |                                |
| 003772 | 032767 | 000002 | 175050 | TST10: | BIT             | #2,     | \$FPS                          |
| 004000 | 001042 |        |        |        | BNE             | END10   |                                |
| 004002 | 173702 |        |        |        | CMPD            | AC2,    | AC3                            |
| 004004 | 170000 |        |        |        | CFCC            |         |                                |
| 004006 | 001437 |        |        |        | BEQ             | END10   |                                |
|        |        |        |        |        | :COMPENSATE FOR | FORTRAN | INACCURACIES.                  |
| 004010 | 174267 | 175006 |        | STD    | AC2,            | ANS1    | :SAVE FPU ANSWER               |
| 004014 | 062767 | 000001 | 175006 | ADD    | #1,             | ANS1+6  | :INCREMENT FPU ANSWER          |
| 004022 | 005567 | 175000 |        | ADC    | ANS1+4          |         |                                |
| 004026 | 005567 | 174772 |        | ADC    | ANS1+2          |         |                                |
| 004032 | 005567 | 174764 |        | ADC    | ANS1            |         |                                |
| 004036 | 173767 | 174760 |        | CMPD   | ANS1,           | AC3     | :CHECK ANSWERS AGAIN           |
| 004042 | 170000 |        |        | CFCC   |                 |         | :COPY FLOATING CONDITION CODES |
| 004044 | 001420 |        |        | BEQ    | END10           |         | :BRANCH IF OK                  |
| 004046 | 162767 | 000002 | 174754 | SUB    | #2,             | ANS1+6  | :DECREMENT FPU ANSWER          |
| 004054 | 005667 | 174746 |        | SBC    | ANS1+4          |         |                                |
| 004060 | 005667 | 174740 |        | SBC    | ANS1+2          |         |                                |
| 004064 | 005667 | 174732 |        | SBC    | ANS1            |         |                                |
| 004070 | 173767 | 174726 |        | CMPD   | ANS1,           | AC3     | :CHECK ANSWERS AGAIN           |
| 004074 | 170000 |        |        | CFCC   |                 |         | :COPY FLOATING CONDITION CODES |
| 004076 | 001403 |        |        | BEQ    | END10           |         | :BRANCH IF OK                  |
| 004100 | 174267 | 174716 |        | STD    | AC2,            | ANS1    | :SAVE FPU ANSWER               |
| 004104 | 104000 |        |        | HLT    |                 |         | :FPU AND FORTRAN DISAGREE      |
|        |        |        |        |        |                 |         |                                |
| 004106 | 005067 | 174730 |        | END10: | CLR             | FPS     | :CLR FPU FPS BUFFER            |
| 004112 | 104400 |        |        | SCOPE  |                 |         |                                |



```
*****
:TEST 11:      EXERCISE SUBF (SUBTRACT FLOATING)
:              ALL INTERRUPTS ON
:              ROUNDING MODE
:*****
```

|        |         |         |        |         |                   |                                   |
|--------|---------|---------|--------|---------|-------------------|-----------------------------------|
| 004114 | 116767  | 005732  | 005724 | MOVB    | \$MINUS, \$SIGN+1 | ;PUT "-" SIGN IN TYPE-OUT ROUTINE |
| 004122 | 012767  | 000000G | 003510 | MOV     | #\$SBR, .ERR1     | ;SELECT PROPER FORTRAN ROUTINES   |
| 004130 | 012767  | 000000G | 003536 | MOV     | #\$SBD, .ERR2     | ;SELECT PROPER FORTRAN ROUTINES   |
| 004136 | 012767  | 000000G | 003642 | MOV     | #\$SBR, .ERR3     | ;SELECT PROPER FORTRAN ROUTINES   |
| 004144 | 012767  | 000000G | 003660 | MOV     | #\$SBD, .ERR4     | ;SELECT PROPER FORTRAN ROUTINES   |
| 004152 | 012767  | 007400  | 174670 | MOV     | #007400, \$FPS    | ;SET IE BITS IN FORTRAN ANSWER    |
| 004160 | 005067  | 174666  |        | CLR     | \$FEC             | ;CLR FORTRAN FEC                  |
| 004164 | 005067  | 174664  |        | CLR     | \$FPC             | ;CLR FORTRAN FPC                  |
| 004170 | 005067  | 174646  |        | CLR     | FPS               | ;CLR FPU FPS BUFFER               |
| 004174 | 005067  | 174644  |        | CLR     | FEC               | ;CLR FPU FEC BUFFER               |
| 004200 | 005067  | 174642  |        | CLR     | FPC               | ;CLR FPU FPC BUFFER               |
| 004204 | 004767  | 003776  |        | JSR     | PC, RANDM2        | ;GET RANDOM INPUT DATA            |
| 004210 | 004467  | 003034  |        | JSR     | R4, \$POLSH       | ;ENTER POLISH MODE                |
| 004214 | 007262  |         |        | \$P.2A  |                   | ;PUSH 2 WORDS ON STACK (LONUM)    |
| 004216 | 007304  |         |        | \$P.2B  |                   | ;PUSH 2 WORDS ON STACK (HINUM)    |
| 004220 | 000000G |         |        | \$SBR   |                   | ;ADDRESS OF FORTRAN SUBTRACT      |
| 004222 | 007352  |         |        | \$TST   |                   | ;DETERMINE THE CONDITION CODES    |
| 004224 | 007316  |         |        | \$POP2X |                   | ;POP 2 WORDS AND EXIT POLISH MODE |
| 004226 | 016700  | 174616  |        | MOV     | \$FPS, RO         | ;DISPLAY FLOATING POINT STATUS    |
| 004232 | 170127  | 040000  |        | LDFPS   | #040000           | ;LOAD FPS, INTERRUPT DISABLE      |
| 004236 | 172467  | 174540  |        | LDF     | LONUM, ACO        | ;LOAD ACO WITH A RANDOM NUMBER    |
| 004242 | 172567  | 174544  |        | LDF     | HINUM, AC1        | ;LOAD AC1 WITH A RANDOM NUMBER    |
| 004246 | 172767  | 174560  |        | LDF     | ANS2, AC3         | ;LOAD AC3 WITH THE SUM            |
| 004252 | 170127  | 007400  |        | LDFPS   | #007400           | ;TURN INTERRUPTS ON               |
| 004256 | 012767  | 004264  | 005660 | MOV     | #+6, LAD          | ;RESET LOOP ADDRESS               |

```
*****
```

|        |        |        |        |         |            |                                     |
|--------|--------|--------|--------|---------|------------|-------------------------------------|
| 004264 | 172600 |        |        | LDF     | ACO, AC2   | ;LOAD ACO INTO AC2                  |
| 004266 | 173201 |        |        | SUBF    | AC1, AC2   | ;SUBTRACT AC1 BY AC2                |
| 004270 | 005767 | 174554 |        | TST     | \$FPS      | ;CHECK FOR FORTRAN FPS ERROR FLAG   |
| 004274 | 100412 |        |        | BMI     | ERR11      | ;BRANCH IF ERROR FLAG SET           |
| 004276 | 170267 | 174540 |        | STFPS   | FPS        | ;STORE FLOATING POINT STATUS        |
| 004302 | 026767 | 174534 | 174540 | CMP     | FPS, \$FPS | ;CHECK FPS                          |
| 004310 | 001427 |        |        | BEQ     | TST11      | ;BRANCH IF OK                       |
| 004312 | 174267 | 174504 |        | STF     | AC2, ANS1  | ;SAVE FPU ANSWER                    |
| 004316 | 104001 |        |        | HLT+1   |            | ;FPS ERROR                          |
| 004320 | 000444 |        |        | BR      | END11      | ;SKIP COMPARE                       |
| 004322 | 170000 |        |        | CFCC    |            | ;WAIT FOR FPU TO FINISH             |
| 004324 | 026767 | 174512 | 174516 | CMP     | FPS, \$FPS | ;CHECK THE FLOATING POINT STATUS    |
| 004332 | 001402 |        |        | BEQ     | #+6        | ;BRANCH IF OK                       |
| 004334 | 104377 |        |        | HLT+377 |            | ;FPS ERROR                          |
| 004336 | 000435 |        |        | BR      | END11      | ;SKIP TO END                        |
| 004340 | 026767 | 174500 | 174504 | CMP     | FEC, \$FEC | ;CHECK THE FLOATING EXCEPTION CODES |
| 004346 | 001402 |        |        | BEQ     | #+6        | ;BRANCH IF OK                       |
| 004350 | 104377 |        |        | HLT+377 |            | ;FEC IS WRONG                       |

```

004352 000427          BR      END11          ;SKIP TO END
004354 026767 174466 174472  CMP      FPC,$FPC      ;CHECK FLOATING PC
004362 001402          BEQ      TST11          ;BRANCH IF OK
004364 104377          HLT+377        ;WRONG ADDRESS IN FPC
004366 000421          BR      END11          ;SKIP TO END

004370 173702          TST11:  CMPF     AC2,    AC3      ;COMPARE FPU ANSWER TO FORTRAN ANSWER
004372 170000          CFCC          ;COPY FLOATING CONDITION CODES
004374 001416          BEQ      END11          ;ANSWERS CHECK
                                ;COMPENSATE FOR FORTRAN INACCURACIES.
004376 174267 174420  STF      AC2,    ANS1      ;SAVE FPU ANSWER
004402 162767 000001 174414  SUB      #1,    ANS1+2  ;DECREMENT FPU ANSWER
004410 005667 174406  SBC      ANS1
004414 173767 174402  CMPF     ANS1,    AC3      ;CHECK ANSWERS AGAIN
004420 170000          CFCC          ;COPY FLOATING CONDITION CODES
004422 001403          BEQ      END11          ;BRANCH IF OK
004424 174267 174372  STF      AC2,    ANS1      ;SAVE FPU ANSWER
004430 104000          HLT          ;FPU AND FORTRAN DISAGREE

004432 005067 174404  END11:  CLR      FPS          ;CLR FPU FPS BUFFER
004436 104400          SCOPE

```

```

;*****
;TEST 12:      EXERCISE SUBD (SUBTRACT DOUBLE PRECISION)
;              ALL INTERRUPTS ON
;              ROUNDING MODE
;*****

```

```

004440 012767 007600 174402  MOV      #007600,$FPS      ;SET IE BITS IN FORTRAN ANSWER
004446 005067 174400  CLR      $FEC          ;CLR FORTRAN FEC
004452 005067 174376  CLR      $FPC          ;CLR FORTRAN FPC
004456 005067 174360  CLR      FPS          ;CLR FPU FPS BUFFER
004462 005067 174356  CLR      FEC          ;CLR FPU FEC BUFFER
004466 005067 174354  CLR      FPC          ;CLR FPU FPC BUFFER
004472 004767 003646  JSR      PC,    R4,    RANDM4  ;GET RANDOM INPUT DATA
004476 004467 002546  JSR      R4,    $POLSH  ;ENTER POLISH MODE
004502 007252  $P.4A    PUSH     4 WORDS ON STACK (LONUM)
004504 007274  $P.4B    PUSH     4 WORDS ON STACK (HINUM)
004506 000000G  $SBD     ADDRESS OF FORTRAN SUBTRACT
004510 007352  $TST     DETERMINE THE CONDITION CODES
004512 007330  $POP4X   POP 4 WORDS AND EXIT POLISH MODE

004514 016700 174330  MOV      $FPS,    R0      ;DISPLAY FLOATING POINT STATUS
004520 170127 040200  LDFPS   #040200  ;LOAD FPS, INTERRUPT DISABLE AND FD
004524 172467 174252  LDD     LONUM,    AC0      ;LOAD AC0 WITH A RANDOM NUMBER
004530 172567 174256  LDD     HINUM,    AC1      ;LOAD AC1 WITH A RANDOM NUMBER
004534 172767 174272  LDD     ANS2,    AC3      ;LOAD AC3 WITH THE SUM
004540 170127 007600  LDFPS   #007600  ;TURN INTERRUPTS ON
004544 012767 004552 005372  MOV      #.+6,    LAD     ;RESET LOOP ADDRESS

```

```

;*****

```

```

004552 172600          RET12:  LDD     ACO,    AC2      ;LOAD ACO INTO AC2
004554 173201          SUBD     AC1,    AC2      ;SUBTRACT AC1 BY AC2
004556 005767 174266  TST      $FPS          ;CHECK FOR FORTRAN FPS ERROR FLAG

```



```

004562 100412          BMI      ERR12      ;BRANCH IF ERROR FLAG SET
004564 170267 174252   STFPS    FPS         ;STORE FLOATING POINT STATUS
004570 026767 174246 174252  CMP     FPS,    $FPS    ;CHECK FPS
004576 001427          BEQ      TST12      ;BRANCH IF OK
004600 174267 174216   STD      AC2,    ANS1    ;SAVE FPU ANSWER
004604 104001          HLT+1    ;FPS ERROR
004606 000450          BR       END12      ;SKIP TO END

004610 170000          ERR12: CFCC          ;WAIT FOR FPU TO FINISH
004612 026767 174224 174230  CMP     FPS,    $FPS    ;CHECK THE FLOATING POINT STATUS
004620 001402          BEQ      .+6        ;BRANCH IF OK
004622 104377          HLT+377   ;FPS ERROR
004624 000441          BR       END12      ;SKIP TO END

004626 026767 174212 174216  CMP     FEC, $FEC    ;CHECK THE FLOATING EXCEPTION CODES
004634 001402          BEQ      .+6        ;BRANCH IF OK
004636 104377          HLT+377   ;FEC IS WRONG
004640 000433          BR       END12      ;SKIP TO END

004642 026767 174200 174204  CMP     FPC, $FPC    ;CHECK FLOATING PC
004650 001402          BEQ      TST12      ;BRANCH IF OK
004652 104377          HLT+377   ;WRONG ADDRESS IN FPC
004654 000425          BR       END12      ;SKIP TO END

004656 173702          TST12: CMPD     AC2,    AC3    ;COMPARE FPU ANSWER TO FORTRAN ANSWER
004660 170000          CFCC          ;COPY FLOATING CONDITION CODES
004662 001422          BEQ      END12      ;ANSWERS CHECK
                                ;COMPENSATE FOR FORTRAN INACCURACIES.
004664 174267 174132   STD      AC2,    ANS1    ;SAVE FPU ANSWER
004670 162767 000001 174132  SUB     #1,    ANS1+6  ;DECREMENT FPU ANSWER
004676 005667 174124          SBC     ANS1+4
004702 005667 174116          SBC     ANS1+2
004706 005667 174110          SBC     ANS1
004712 173767 174104          CMPD     ANS1,    AC3    ;CHECK ANSWERS AGAIN
004716 170000          CFCC          ;COPY FLOATING CONDITION CODES
004720 001403          BEQ      END12      ;BRANCH IF OK
004722 174267 174074          STD      AC2,    ANS1    ;SAVE FPU ANSWER
004726 104000          HLT          ;FPU AND FORTRAN DISAGREE

004730 005067 174106          END12: CLR     FPS         ;CLR FPU FPS BUFFER
004734 104400          SCOPE

;*****
;TEST 13:      EXERCISE SUBF (SUBTRACT FLOATING)
;              OVERFLOW AND UNDERFLOW INTERRUPTS OFF.
;              ROUNDING MODE
;*****

004736 012767 004400 174104  MOV     #004400, $FPS  ;SET IE BITS IN FORTRAN ANSWER
004744 005067 174102          CLR     $FEC          ;CLR FORTRAN FEC
004750 005067 174100          CLR     $FPC          ;CLR FORTRAN FPC
004754 005067 174062          CLR     FPS          ;CLR FPU FPS BUFFER
004760 005067 174060          CLR     FEC          ;CLR FPU FEC BUFFER
004764 005067 174056          CLR     FPC          ;CLR FPU FPC BUFFER
004770 004767 003212          JSR     PC,    RANDM2  ;GET RANDOM INPUT DATA
004774 004467 002250          JSR     R4,    $POLSH  ;ENTER POLISH MODE

```

\*\*\*\*\*

```

*****
TEST 14:      EXERCISE SUBD (SUBTRACT DOUBLE PRECISION)
               OVERFLOW AND UNDERFLOW INTERRUPTS OFF
               ROUNDING MODE
*****

```

|        |        |        |        |     |               |                                |
|--------|--------|--------|--------|-----|---------------|--------------------------------|
| 005160 | 012767 | 004600 | 173662 | MOV | #004600,\$FPS | ;SET IE BITS IN FORTRAN ANSWER |
| 005166 | 005067 | 173660 |        | CLR | \$FEC         | ;CLR FORTRAN FEC               |
| 005172 | 005067 | 173656 |        | CLR | \$FPC         | ;CLR FORTRAN FPC               |
| 005176 | 005067 | 173640 |        | CLR | FPS           | ;CLR FPU FPS BUFFER            |
| 005202 | 005067 | 173636 |        | CLR | FEC           | ;CLR FPU FEC BUFFER            |
| 005206 | 005067 | 173634 |        | CLR | FPC           | ;CLR FPU FPC BUFFER            |
| 005212 | 004767 | 003126 |        | JSR | PC, RANDM4    | ;GET RANDOM INPUT DATA         |



|        |         |        |        |         |         |         |                                                |
|--------|---------|--------|--------|---------|---------|---------|------------------------------------------------|
| 005216 | 004467  | 002026 |        | JSR     | R4,     | \$POLSH | ;ENTER POLISH MODE                             |
| 005222 | 007252  |        |        | \$P.4A  |         |         | ;PUSH 4 WORDS ON STACK (LONUM)                 |
| 005224 | 007274  |        |        | \$P.4B  |         |         | ;PUSH 4 WORDS ON STACK (HINUM)                 |
| 005226 | 000000G |        |        | \$SBD   |         |         | ;ADDRESS OF FORTRAN SUBTRACT                   |
| 005230 | 007352  |        |        | \$TST   |         |         | ;DETERMINE THE CONDITION CODES                 |
| 005232 | 007330  |        |        | \$POP4X |         |         | ;POP 4 WORDS AND EXIT POLISH MODE              |
|        |         |        |        |         |         |         |                                                |
| 005234 | 016700  | 173610 |        | MOV     | \$FPS,  | RO      | ;DISPLAY FLOATING POINT STATUS                 |
| 005240 | 170127  | 040200 |        | LDFPS   | #040200 |         | ;LOAD FPS, INTERRUPT DISABLE AND FD            |
| 005244 | 172467  | 173532 |        | LDD     | LONUM,  | AC0     | ;LOAD AC0 WITH A RANDOM NUMBER                 |
| 005250 | 172567  | 173536 |        | LDD     | HINUM,  | AC1     | ;LOAD AC1 WITH A RANDOM NUMBER                 |
| 005254 | 172767  | 173552 |        | LDD     | ANS2,   | AC3     | ;LOAD AC3 WITH THE SUM                         |
| 005260 | 170127  | 004600 |        | LDFPS   | #004600 |         | ;TURN INTERRUPTS ON, EXCEPT OVER AND UNDERFLOW |
| 005264 | 012757  | 005272 | 004652 | MOV     | #+6,    | LAD     | ;RESET LOOP ADDRESS                            |

;\*\*\*\*\*

|        |        |        |        |        |        |        |                                       |
|--------|--------|--------|--------|--------|--------|--------|---------------------------------------|
| 005272 | 172600 |        |        | LDD    | AC0,   | AC2    | ;LOAD AC0 INTO AC2                    |
| 005274 | 173201 |        |        | SUBD   | AC1,   | AC2    | ;SUBTRACT AC1 BY AC2                  |
| 005276 | 170267 | 173540 |        | STFPS  | FPS,   |        | ;STORE FLOATING POINT STATUS          |
| 005302 | 026767 | 173534 | 173540 | CMP    | FPS,   | \$FPS  | ;CHECK FPS                            |
| 005310 | 001404 |        |        | BEQ    | TST14  |        | ;BRANCH IF OK                         |
| 005312 | 174267 | 173504 |        | STD    | AC2,   | ANS1   | ;SAVE FPU ANSWER                      |
| 005316 | 104001 |        |        | HLT+1  |        |        | ;FPS ERROR                            |
| 005320 | 000431 |        |        | BR     | END14  |        | ;SKIP COMPARE                         |
|        |        |        |        |        |        |        |                                       |
| 005322 | 032767 | 000002 | 173520 | TST14: | BIT    | #2,    | \$FPS                                 |
| 005330 | 001025 |        |        |        | BNE    | END14  | ;CHECK FOR OVERFLOW                   |
| 005332 | 173702 |        |        |        | CMPD   | AC2,   | AC3                                   |
| 005334 | 170000 |        |        |        | CFCC   |        | ;BRANCH IF OVERFLOW                   |
| 005336 | 001422 |        |        |        | BEQ    | END14  | ;COMPARE FPU ANSWER TO FORTRAN ANSWER |
|        |        |        |        |        |        |        | ;COPY FLOATING CONDITION CODES        |
|        |        |        |        |        |        |        | ;ANSWERS CHECK                        |
|        |        |        |        |        |        |        | ;INACCURACIES.                        |
| 005340 | 174267 | 173456 |        | STD    | AC2,   | ANS1   | ;SAVE FPU ANSWER                      |
| 005344 | 162767 | 000001 | 173456 | SUB    | #1,    | ANS1+6 | ;DECREMENT FPU ANSWER                 |
| 005352 | 005667 | 173450 |        | SBC    | ANS1+4 |        |                                       |
| 005356 | 005667 | 173442 |        | SBC    | ANS1+2 |        |                                       |
| 005362 | 005667 | 173434 |        | SBC    | ANS1   |        |                                       |
| 005366 | 173767 | 173430 |        | CMPD   | ANS1,  | AC3    | ;CHECK ANSWERS AGAIN                  |
| 005372 | 170000 |        |        | CFCC   |        |        | ;COPY FLOATING CONDITION CODES        |
| 005374 | 001403 |        |        | BEQ    | END14  |        | ;BRANCH IF OK                         |
| 005376 | 174267 | 173420 |        | STD    | AC2,   | ANS1   | ;SAVE FPU ANSWER                      |
| 005402 | 104000 |        |        | HLT    |        |        | ;FPU AND FORTRAN DISAGREE             |
|        |        |        |        |        |        |        |                                       |
| 005404 | 005067 | 173432 |        | END14: | CLR    | FPS    | ;CLR FPU FPS BUFFER                   |
| 005410 | 104400 |        |        |        | SCOPE  |        |                                       |

```

;*****
;TEST 15:      EXERCISE SUBF (SUBTRACT FLOATING)
;              ALL INTERRUPTS ON
;              TRUNCATE MODE
;*****

```

|        |        |        |        |     |          |       |                                |
|--------|--------|--------|--------|-----|----------|-------|--------------------------------|
| 005412 | 012767 | 007440 | 173430 | MOV | #007440, | \$FPS | ;SET IE BITS IN FORTRAN ANSWER |
| 005420 | 005067 | 173426 |        | CLR | \$FEC    |       | ;CLR FORTRAN FEC               |
| 005424 | 005067 | 173424 |        | CLR | \$FPC    |       | ;CLR FORTRAN FPC               |
| 005430 | 005067 | 173406 |        | CLR | FPS      |       | ;CLR FPU FPS BUFFER            |

|        |         |        |         |     |         |                                   |
|--------|---------|--------|---------|-----|---------|-----------------------------------|
| 005434 | 005067  | 173404 | CLR     | FEC |         | :CLR FPU FEC BUFFER               |
| 005440 | 005067  | 173404 | CLR     | FPC |         | :CLR FPU FPC BUFFER               |
| 005444 | 004767  | 002526 | JSR     | PC, | RANOM2  | :GET RANDOM INPUT DATA            |
| 005450 | 004467  | 001574 | JSR     | R4, | \$POLSH | :ENTER POLISH MODE                |
| 005454 | 007262  |        | SP,2A   |     |         | :PUSH 2 WORDS ON STACK (LONUM)    |
| 005456 | 007304  |        | SP,2B   |     |         | :PUSH 2 WORDS ON STACK (MINUM)    |
| 005460 | 000000G |        | \$SBR   |     |         | :ADDRESS OF FORTRAN SUBTRACT      |
| 005462 | 007352  |        | \$TST   |     |         | :DETERMINE THE CONDITION CODES    |
| 005464 | 007316  |        | \$POP2X |     |         | :POP 2 WORDS AND EXIT POLISH MODE |

|        |        |        |       |          |     |                                |
|--------|--------|--------|-------|----------|-----|--------------------------------|
| 005466 | 016700 | 173356 | MOV   | \$FPS,   | RO  | :DISPLAY FLOATING POINT STATUS |
| 005472 | 170127 | 040000 | LDFPS | \$040000 |     | :LOAD FPS, INTERRUPT DISABLE   |
| 005476 | 172467 | 173300 | LDF   | LONUM,   | AC0 | :LOAD AC0 WITH A RANDOM NUMBER |
| 005502 | 172567 | 173304 | LDF   | MINUM,   | AC1 | :LOAD AC1 WITH A RANDOM NUMBER |
| 005506 | 172767 | 173320 | LDF   | ANS2,    | AC3 | :LOAD AC3 WITH THE SUM         |
| 005512 | 170127 | 007440 | LDFPS | \$007440 |     | :TURN INTERRUPTS ON            |
| 005516 | 012767 | 005524 | MOV   | \$,+6,   | LAD | :RESET LOOP ADDRESS            |

:\*\*\*\*\*

|        |        |        |                         |           |        |                                     |                                       |
|--------|--------|--------|-------------------------|-----------|--------|-------------------------------------|---------------------------------------|
| 005524 | 172600 |        | LDF                     | AC0,      | AC2    | :LOAD AC0 INTO AC2                  |                                       |
| 005526 | 173201 |        | SUBF                    | AC1,      | AC2    | :SUBTRACT AC1 BY AC2                |                                       |
| 005530 | 005767 | 173314 | TST                     | \$FPS     |        | :CHECK FOR FORTRAN FPS ERROR FLAG   |                                       |
| 005534 | 100412 |        | BMI                     | ERR15     |        | :BRANCH IF ERROR FLAG SET           |                                       |
| 005536 | 170267 | 173300 | STFPS                   | FPS       |        | :STORE FLOATING POINT STATUS        |                                       |
| 005542 | 026767 | 173274 | CMP                     | FPS,      | \$FPS  | :CHECK FPS                          |                                       |
| 005550 | 001427 |        | BEQ                     | TST15     |        | :BRANCH IF OK                       |                                       |
| 005552 | 174267 | 173244 | STF                     | AC2,      | ANS1   | :SAVE FPU ANSWER                    |                                       |
| 005556 | 104001 |        | HLT+1                   |           |        | :FPS ERROR                          |                                       |
| 005560 | 000455 |        | BR                      | END15     |        | :SKIP COMPARE                       |                                       |
| 005562 | 170000 |        | CFCC                    |           |        | :WAIT FOR FPU TO FINISH             |                                       |
| 005564 | 026767 | 173252 | CMP                     | FPS,      | \$FPS  | :CHECK THE FLOATING POINT STATUS    |                                       |
| 005572 | 001402 |        | BEQ                     | .,+6      |        | :BRANCH IF OK                       |                                       |
| 005574 | 104377 |        | HLT+377                 |           |        | :FPS ERROR                          |                                       |
| 005576 | 000446 |        | BR                      | END15     |        | :SKIP TO END                        |                                       |
| 005600 | 026767 | 173240 | CMP                     | FEC,\$FEC |        | :CHECK THE FLOATING EXCEPTION CODES |                                       |
| 005606 | 001402 | 173244 | BEQ                     | .,+6      |        | :BRANCH IF OK                       |                                       |
| 005610 | 104377 |        | HLT+377                 |           |        | :FEC IS WRONG                       |                                       |
| 005612 | 000440 |        | BR                      | END15     |        | :SKIP TO END                        |                                       |
| 005614 | 026767 | 173226 | CMP                     | FPC,\$FPC |        | :CHECK FLOATING PC                  |                                       |
| 005622 | 001402 | 173232 | BEQ                     | TST15     |        | :BRANCH IF OK                       |                                       |
| 005624 | 104377 |        | HLT+377                 |           |        | :WRONG ADDRESS IN FPC               |                                       |
| 005626 | 000432 |        | BR                      | END15     |        | :SKIP TO END                        |                                       |
| 005630 | 173702 |        | TST15:                  | CMPF      | AC2,   | AC3                                 | :COMPARE FPU ANSWER TO FORTRAN ANSWER |
| 005632 | 170000 |        | CFCC                    |           |        | :COPY FLOATING CONDITION CODES      |                                       |
| 005634 | 001427 |        | BEQ                     | END15     |        | :ANSWERS CHECK                      |                                       |
| 005636 | 174267 | 173160 | :COMPENSATE FOR FORTRAN |           |        | :INACCURACIES.                      |                                       |
| 005642 | 062767 | 000001 | STF                     | AC2,      | ANS1   | :SAVE FPU ANSWER                    |                                       |
| 005650 | 005567 | 173146 | ADD                     | \$1,      | ANS1+2 | :INCREMENT FPU ANSWER               |                                       |
| 005654 | 173767 | 173142 | ADC                     | ANS1,     |        |                                     |                                       |
| 005660 | 170000 |        | CMPF                    | ANS1,     | AC3    | :CHECK ANSWERS AGAIN                |                                       |
|        |        |        | CFCC                    |           |        | :COPY FLOATING CONDITION CODES      |                                       |



```

005662 001414          BEQ      END15      ;BRANCH IF OK
005664 162767 000002 173132 SUB      #2,      ANS1+2 ;DECREMENT FPU ANSWER
005672 005667 173124          SBC      ANS1
005676 173767 173120          CMPF     ANS1,    AC3      ;CHECK ANSWERS AGAIN
005702 170000          CFCC          ;COPY FLOATING CONDITION CODES
005704 001403          BEQ      END15      ;BRANCH IF OK
005706 174267 173110          STF      AC2,    ANS1      ;SAVE FPU ANSWER
005712 104000          HLT              ;FPU AND FORTRAN DISAGREE

005714 005067 173122          END15: CLR      FPS          ;CLR FPU FPS BUFFER
005720 104400          SCOPE

```

```

:*****
:TEST 16:      EXERCISE SUBD (SUBTRACT DOUBLE PRECISION)
:              ALL INTERRUPTS ON
:              TRUNCATE MODE
:*****

```

```

005722 012767 007640 173120      MOV      #007640,$FPS ;SET IE BITS IN FORTRAN ANSWER
005730 005067 173116      CLR      $FEC      ;CLR FORTRAN FEC
005734 005067 173114      CLR      $FPC      ;CLR FORTRAN FPC
005740 005067 173076      CLR      FPS        ;CLR FPU FPS BUFFER
005744 005067 173074      CLR      FEC        ;CLR FPU FEC BUFFER
005750 005067 173072      CLR      FPC        ;CLR FPU FPC BUFFER
005754 004767 002364      JSR      PC,      RANDM4 ;GET RANDOM INPUT DATA
005760 004467 001264      JSR      R4,      $POLSH ;ENTER POLISH MODE
005764 007252      $P.4A      ;PUSH 4 WORDS ON STACK (LONUM)
005766 007274      $P.4B      ;PUSH 4 WORDS ON STACK (HINUM)
005770 000000G      $SBD      ;ADDRESS OF FORTRAN SUBTRACT
005772 007352      $TST      ;DETERMINE THE CONDITION CODES
005774 007330      $POP4X     ;POP 4 WORDS AND EXIT POLISH MODE

005776 016700 173046      MOV      $FPS,    R0      ;DISPLAY FLOATING POINT STATUS
006002 170127 040200      LDFPS   #040200 ;LOAD FPS, INTERRUPT DISABLE AND FD
006006 172467 172770      LDD      LONUM,    AC0      ;LOAD AC0 WITH A RANDOM NUMBER
006012 172567 172774      LDD      HINUM,    AC1      ;LOAD AC1 WITH A RANDOM NUMBER
006016 172767 173010      LDD      ANS2,    AC3      ;LOAD AC3 WITH THE SUM
006022 170127 007640      LDFPS   #007640 ;TURN INTERRUPTS ON
006026 012767 006034 004110      MOV      #.+6,    LAD ;RESET LOOP ADDRESS

```

```

:*****

```

```

006034 172600          LDD      AC0,      AC2      ;LOAD AC0 INTO AC2
006036 173201          SUBD     AC1,      AC2      ;SUBTRACT AC1 BY AC2
006040 005767 173004          TST      $FPS      ;CHECK FOR FORTRAN FPS ERROR FLAG
006044 100412          BMI      ERR16     ;BRANCH IF ERROR FLAG SET
006046 170267 172770          STFPS   FPS        ;STORE FLOATING POINT STATUS
006052 026767 172764 172770      CMP      FPS,    $FPS      ;CHECK FPS
006060 001427          BEQ      TST16     ;BRANCH IF OK
006062 174267 172734          STD      AC2,    ANS1      ;SAVE FPU ANSWER
006066 104001          HLT+1          ;FPS ERROR
006070 000465          BR        END16     ;SKIP TO END

006072 170000          ERR16: CFCC          ;WAIT FOR FPU TO FINISH
006074 026767 172742 172746      CMP      FPS,    $FPS      ;CHECK THE FLOATING POINT STATUS
006102 001402          BEQ      .+6        ;BRANCH IF OK

```

|        |        |        |        |         |                         |        |     |                                       |
|--------|--------|--------|--------|---------|-------------------------|--------|-----|---------------------------------------|
| 006104 | 104377 |        |        | HLT+377 |                         |        |     | :FPS ERROR                            |
| 006106 | 000456 |        |        | BR      | END16                   |        |     | :SKIP TO END                          |
| 006110 | 026767 | 172730 | 172734 | CMP     | FEC,\$FEC               |        |     | :CHECK THE FLOATING EXCEPTION CODES   |
| 006116 | 001402 |        |        | BEQ     | +.6                     |        |     | :BRANCH IF OK                         |
| 006120 | 104377 |        |        | HLT+377 |                         |        |     | :FEC IS WRONG                         |
| 006122 | 000450 |        |        | BR      | END16                   |        |     | :SKIP TO END                          |
| 006124 | 026767 | 172716 | 172722 | CMP     | FPC,\$FPC               |        |     | :CHECK FLOATING PC                    |
| 006132 | 001402 |        |        | BEQ     | TST16                   |        |     | :BRANCH IF OK                         |
| 006134 | 104377 |        |        | HLT+377 |                         |        |     | :WRONG ADDRESS IN FPC                 |
| 006136 | 000442 |        |        | BR      | END16                   |        |     | :SKIP TO END                          |
| 006140 | 173702 |        |        | TST16:  | CMPD                    | AC2,   | AC3 | :COMPARE FPU ANSWER TO FORTRAN ANSWER |
| 006142 | 170000 |        |        |         | CFCC                    |        |     | :COPY FLOATING CONDITION CODES        |
| 006144 | 001437 |        |        |         | BEQ                     | END16  |     | :ANSWERS CHECK                        |
|        |        |        |        |         | :COMPENSATE FOR FORTRAN |        |     | :INACCURACIES.                        |
| 006146 | 174267 | 172650 |        | STD     | AC2,                    | ANS1   |     | :SAVE FPU ANSWER                      |
| 006152 | 062767 | 000001 | 172650 | ADD     | #1,                     | ANS1+6 |     | :INCREMENT FPU ANSWER                 |
| 006160 | 005567 | 172642 |        | ADC     | ANS1+4                  |        |     |                                       |
| 006164 | 005567 | 172634 |        | ADC     | ANS1+2                  |        |     |                                       |
| 006170 | 005567 | 172626 |        | ADC     | ANS1                    |        |     |                                       |
| 006174 | 173767 | 172622 |        | CMPD    | ANS1,                   | AC3    |     | :CHECK ANSWERS AGAIN                  |
| 006200 | 170000 |        |        | CFCC    |                         |        |     | :COPY FLOATING CONDITION CODES        |
| 006202 | 001420 |        |        | BEQ     | END16                   |        |     | :BRANCH IF OK                         |
| 006204 | 162767 | 000002 | 172616 | SUB     | #2,                     | ANS1+6 |     | :DECREMENT FPU ANSWER                 |
| 006212 | 005667 | 172610 |        | SBC     | ANS1+4                  |        |     |                                       |
| 006216 | 005667 | 172602 |        | SBC     | ANS1+2                  |        |     |                                       |
| 006222 | 005667 | 172574 |        | SBC     | ANS1                    |        |     |                                       |
| 006226 | 173767 | 172570 |        | CMPD    | ANS1,                   | AC3    |     | :CHECK ANSWERS AGAIN                  |
| 006232 | 170000 |        |        | CFCC    |                         |        |     | :COPY FLOATING CONDITION CODES        |
| 006234 | 001403 |        |        | BEQ     | END16                   |        |     | :BRANCH IF OK                         |
| 006236 | 174267 | 172560 |        | STD     | AC2,                    | ANS1   |     | :SAVE FPU ANSWER                      |
| 006242 | 104000 |        |        | HLT     |                         |        |     | :FPU AND FORTRAN DISAGREE             |
| 006244 | 005067 | 172572 |        | END16:  | CLR                     | FPS    |     | :CLR FPU FPS BUFFER                   |
| 006250 | 104400 |        |        |         | SCOPE                   |        |     |                                       |

```

*****
:TEST 17:      EXERCISE SUBF (SUBTRACT FLOATING)
:              OVERFLOW AND UNDERFLOW INTERRUPTS OFF.
:              TRUNCATE MODE
*****

```



```

006252 012767 004440 172570      MOV      #004440,$FPS      ;SET IE BITS IN FORTRAN ANSWER
006260 005067 172566      CLR      $FEC          ;CLR FORTRAN FEC
006264 005067 172564      CLR      $FPC          ;CLR FORTRAN FPC
006270 005067 172546      CLR      FPS            ;CLR FPU FPS BUFFER
006274 005067 172544      CLR      FEC            ;CLR FPU FEC BUFFER
006300 005067 172542      CLR      FPC            ;CLR FPU FPC BUFFER
006304 004767 001676      JSR      PC,      RANDM2      ;GET RANDOM INPUT DATA
006310 004467 000734      JSR      R4,      $POLSH      ;ENTER POLISH MODE
006314 007262      $P.2A      ;PUSH 2 WORDS ON STACK (LONUM)
006316 007304      $P.2B      ;PUSH 2 WORDS ON STACK (MINUM)
006320 000000G      $SBR          ;ADDRESS OF FORTRAN SUBTRACT
006322 007352      $TST          ;DETERMINE THE CONDITION CODES
006324 007316      $POP2X      ;POP 2 WORDS AND EXIT POLISH MODE

006326 016700 172516      MOV      $FPS,      R0      ;DISPLAY FLOATING POINT STATUS
006332 170127 040000      LDFPS     #040000      ;LOAD FPS, INTERRUPT DISABLE
006336 172467 172440      LDF      LONUM,      ACO      ;LOAD ACO WITH A RANDOM NUMBER
006342 172567 172444      LDF      MINUM,      AC1      ;LOAD AC1 WITH A RANDOM NUMBER
006346 172767 172460      LDF      ANS2,      AC3      ;LOAD AC3 WITH THE SUM
006352 170127 004440      LDFPS     #004440      ;TURN INTERRUPTS ON, EXCEPT OVERFLOW AND UNDERFLOW
006356 012767 006364 003560      MOV      #.+6,      LAD      ;RESET LOOP ADDRESS

```

\*\*\*\*\*

```

006364 172600      LDF      ACO,      AC2      ;LOAD ACO INTO AC2
006366 173201      SUBF     AC1,      AC2      ;SUBTRACT AC1 BY AC2
006370 170267 172446      STFPS     FPS            ;STORE FLOATING POINT STATUS
006374 026767 172442 172446      CMP      FPS,      $FPS      ;CHECK FPS
006402 001404      BEQ      TST17      ;BRANCH IF OK
006404 174267 172412      STF      AC2,      ANS1      ;SAVE FPU ANSWER
006410 104001      HLT+1      ;FPS ERROR
006412 000436      BR       END17      ;SKIP COMPARE

006414 032767 000002 172426 TST17: BIT      #2,      $FPS      ;CHECK FOR OVERFLOW
006422 001032      BNE      END17      ;BRANCH IF OVERFLOW
006424 173702      CMPE     AC2,      AC3      ;COMPARE FPU ANSWER TO FORTRAN ANSWER
006426 170000      CFCC          ;COPY FLOATING CONDITION CODES
006430 001427      BEQ      END17      ;ANSWERS CHECK
                                ;COMPENSATE FOR FORTRAN INACCURACIES.
006432 174267 172364      STF      AC2,      ANS1      ;SAVE FPU ANSWER
006436 062767 000001 172360      ADD      #1,      ANS1+2      ;INCREMENT FPU ANSWER
006444 005567 172352      ADC      ANS1,      AC3      ;CHECK ANSWERS AGAIN
006450 173767 172346      CMPE     ANS1,      AC3      ;COPY FLOATING CONDITION CODES
006454 170000      BEQ      END17      ;BRANCH IF OK
006456 001414      SUB      #2,      ANS1+2      ;DECREMENT FPU ANSWER
006460 162767 000002 172336      SBC      ANS1,      AC3      ;CHECK ANSWERS AGAIN
006466 005667 172330      CMPE     ANS1,      AC3      ;COPY FLOATING CONDITION CODES
006472 173767 172324      BEQ      END17      ;BRANCH IF OK
006476 170000      STF      AC2,      ANS1      ;SAVE FPU ANSWER
006500 001403      HLT          ;FPU AND FORTRAN DISAGREE
006502 174267 172314      HLT          ;FPU AND FORTRAN DISAGREE
006506 104000      HLT          ;FPU AND FORTRAN DISAGREE

006510 005067 172326      END17: CLR      FPS            ;CLR FPU FPS BUFFER
006514 104400      SCOPE

```

\*\*\*\*\*  
:TEST 20: EXERCISE SUBD (SUBTRACT DOUBLE PRECISION)  
: OVERFLOW AND UNDERFLOW INTERRUPTS OFF  
: TRUNCATE MODE  
:\*\*\*\*\*

|        |         |        |        |         |               |                                                |
|--------|---------|--------|--------|---------|---------------|------------------------------------------------|
| 006516 | 012767  | 004640 | 172324 | MOV     | #004640,\$FPS | :SET IE BITS IN FORTRAN ANSWER                 |
| 006524 | 005067  | 172322 |        | CLR     | \$FEC         | :CLR FORTRAN FEC                               |
| 006530 | 005067  | 172320 |        | CLR     | \$FPC         | :CLR FORTRAN FPC                               |
| 006534 | 005067  | 172302 |        | CLR     | FPS           | :CLR FPU FPS BUFFER                            |
| 006540 | 005067  | 172300 |        | CLR     | FEC           | :CLR FPU FEC BUFFER                            |
| 006544 | 005067  | 172276 |        | CLR     | FPC           | :CLR FPU FPC BUFFER                            |
| 006550 | 004767  | 001570 |        | JSR     | PC, RANDM4    | :GET RANDOM INPUT DATA                         |
| 006554 | 004467  | 000470 |        | JSR     | R4, \$POLSH   | :ENTER POLISH MODE                             |
| 006560 | 007252  |        |        | \$P.4A  |               | :PUSH 4 WORDS ON STACK (LONUM)                 |
| 006562 | 007274  |        |        | \$P.4B  |               | :PUSH 4 WORDS ON STACK (HINUM)                 |
| 006564 | 000000G |        |        | \$SBD   |               | :ADDRESS OF FORTRAN SUBTRACT                   |
| 006566 | 007352  |        |        | \$TST   |               | :DETERMINE THE CONDITION CODES                 |
| 006570 | 007330  |        |        | \$POP4X |               | :POP 4 WORDS AND EXIT POLISH MODE              |
| 006572 | 016700  | 172252 |        | MOV     | \$FPS, R0     | :DISPLAY FLOATING POINT STATUS                 |
| 006576 | 170127  | 040200 |        | LDFPS   | #040200       | :LOAD FPS, INTERRUPT DISABLE AND FD            |
| 006602 | 172467  | 172174 |        | LDD     | LONUM, ACO    | :LOAD ACO WITH A RANDOM NUMBER                 |
| 006606 | 172567  | 172200 |        | LDD     | HINUM, AC1    | :LOAD AC1 WITH A RANDOM NUMBER                 |
| 006612 | 172767  | 172214 |        | LDD     | ANS2, AC3     | :LOAD AC3 WITH THE SUM                         |
| 006616 | 170127  | 004640 |        | LDFPS   | #004640       | :TURN INTERRUPTS ON, EXCEPT OVER AND UNDERFLOW |
| 006622 | 012767  | 006630 | 003314 | MOV     | #.+6, LAD     | :RESET LOOP ADDRESS                            |

\*\*\*\*\*

|        |        |        |        |        |               |                                       |
|--------|--------|--------|--------|--------|---------------|---------------------------------------|
| 006630 | 172600 |        |        | LDD    | ACO, AC2      | :LOAD ACO INTO AC2                    |
| 006632 | 173201 |        |        | SUBD   | AC1, AC2      | :SUBTRACT AC1 BY AC2                  |
| 006634 | 170267 | 172202 |        | STFPS  | FPS           | :STORE FLOATING POINT STATUS          |
| 006640 | 026767 | 172176 | 172202 | CMF    | FPS, \$FPS    | :CHECK FPS                            |
| 006646 | 001404 |        |        | BEQ    | TST20         | :BRANCH IF OK                         |
| 006650 | 174267 | 172146 |        | STD    | AC2, ANS1     | :SAVE FPU ANSWER                      |
| 006654 | 104001 |        |        | HLT+1  |               | :FPS ERROR                            |
| 006656 | 000446 |        |        | BR     | END20         | :SKIP COMPARE                         |
| 006660 | 032767 | 000002 | 172162 | TST20: | BIT #2, \$FPS | :CHECK FOR OVERFLOW                   |
| 006666 | 001042 |        |        | BNE    | END20         | :BRANCH IF OVERFLOW                   |
| 006670 | 173702 |        |        | CMFD   | AC2, AC3      | :COMPARE FPU ANSWER TO FORTRAN ANSWER |
| 006672 | 170000 |        |        | CFCC   |               | :COPY FLOATING CONDITION CODES        |
| 006674 | 001437 |        |        | BEQ    | END20         | :ANSWERS CHECK                        |
| 006676 | 174267 | 172120 |        | STD    | AC2, ANS1     | :SAVE FPU ANSWER                      |
| 006702 | 062767 | 000001 | 172120 | ADD    | #1, ANS1+6    | :INCREMENT FPU ANSWER                 |
| 006710 | 005567 | 172112 |        | ADC    | ANS1+4        |                                       |
| 006714 | 005567 | 172104 |        | ADC    | ANS1+2        |                                       |
| 006720 | 005567 | 172076 |        | ADC    | ANS1          |                                       |
| 006724 | 173767 | 172072 |        | CMFD   | ANS1, AC3     | :CHECK ANSWERS AGAIN                  |
| 006730 | 170000 |        |        | CFCC   |               | :COPY FLOATING CONDITION CODES        |
| 006732 | 001420 |        |        | BEQ    | END20         | :BRANCH IF OK                         |
| 006734 | 162767 | 000002 | 172066 | SUB    | #2, ANS1+6    | :DECREMENT FPU ANSWER                 |
| 006742 | 005667 | 172060 |        | SBC    | ANS1+4        |                                       |
| 006746 | 005667 | 172052 |        | SBC    | ANS1+2        |                                       |



G03

MAINDEC-11-DOFPS-D  
DOFPSD.F11FLOATING POINT ADD AND SUBTRACT EXERCISER  
TEST SECTION

MACY11 27(732) 17-SEP-76 09:31 PAGE 32

|        |        |        |        |       |      |                                |
|--------|--------|--------|--------|-------|------|--------------------------------|
| 006752 | 005667 | 172044 | SBC    | ANS1  |      |                                |
| 006756 | 173767 | 172040 | CMPD   | ANS1, | AC3  | :CHECK ANSWERS AGAIN           |
| 006762 | 170000 |        | CFCC   |       |      | :COPY FLOATING CONDITION CODES |
| 006764 | 001403 |        | BEQ    | END20 |      | :BRANCH IF OK                  |
| 006766 | 174267 | 172030 | STD    | AC2,  | ANS1 | :SAVE FPU ANSWER               |
| 006772 | 104000 |        | HLT    |       |      | :FPU AND FORTRAN DISAGREE      |
| 006774 | 005067 | 172042 | END20: | CLR   | FPS  | :CLR FPU FPS BUFFER            |
| 007000 | 104400 |        | SCOPE  |       |      |                                |

|        |        |        |        |         |       |                |                                       |
|--------|--------|--------|--------|---------|-------|----------------|---------------------------------------|
| 007002 | 032737 | 002000 | 177570 | DONE:   | BIT   | #SW10,2#SWR    | :RING THE BELL?                       |
| 007010 | 001005 |        |        |         | BNE   | 1\$            | :NO!                                  |
| 007012 | 012767 | 000207 | 003120 |         | MOV   | #BELL,.TYPE    | :TYPE A BELL                          |
| 007020 | 000004 | 012140 |        |         | TYPE, | .TYPE          |                                       |
| 007024 | 005046 |        |        | 1\$:    | CLR   | -(6)           | :CLEAR TRACE TRAP                     |
| 007026 | 032737 | 010000 | 177570 |         | BIT   | #SW12,2#SWR    | :RUN WITH TRT?                        |
| 007034 | 001010 |        |        |         | BNE   | 2\$            |                                       |
| 007036 | 005167 | 003100 |        |         | COM   | TRPB           |                                       |
| 007042 | 100005 |        |        |         | BPL   | 2\$            |                                       |
| 007044 | 052716 | 000020 |        |         | BIS   | #20,(6)        | :SET TRACE TRAP                       |
| 007050 | 012746 | 007102 |        |         | MOV   | #3\$, -(6)     | :JUMP TO START OF TEST                |
| 007054 | 000002 |        |        |         | RTI   |                |                                       |
| 007056 | 012746 | 007064 |        | 2\$:    | MOV   | #4\$, -(6)     | :JUMP TO START OF TEST                |
| 007062 | 000002 |        |        |         | RTI   |                |                                       |
| 007064 | 013700 | 000042 |        | 4\$:    | MOV   | 2#42,R0        | :GET MONITOR ADDRESS                  |
| 007070 | 001404 |        |        |         | BEQ   | 3\$            | :IF NONE                              |
| 007072 | 004710 |        |        |         | JSR   | 7,(0)          | :GO TO MONITOR                        |
| 007074 | 000240 |        |        |         | NOP   |                |                                       |
| 007076 | 000240 |        |        |         | NOP   |                |                                       |
| 007100 | 000240 |        |        |         | NOP   |                |                                       |
| 007102 | 000137 | 000200 |        | 3\$:    | JMP   | 2#200          | :JUMP TO START OF TEST                |
| 007106 | 000002 |        |        | YESRT:  | RTI   |                | :RETURN TO PROGRAM FROM TRAP          |
| 007110 | 032737 | 000400 | 177570 | .EMT:   | BIT   | #SW08,2#SWR    | :KILL LDUB OR LOOP ON SPEC. TEST      |
| 007116 | 001404 |        |        |         | BEQ   | 1\$            |                                       |
| 007120 | 123767 | 177570 | 171652 |         | CMPB  | 2#SWR,ICNT     | :ON RIGHT TEST? *SW7-0*               |
| 007126 | 001437 |        |        |         | BEQ   | OVER           |                                       |
| 007130 | 113703 | 177570 |        | 1\$:    | MOVB  | 2#SWR,R3       | :GET UB BITS                          |
| 007134 | 170003 |        |        |         | LDUB  |                |                                       |
| 007136 | 032737 | 040000 | 177570 |         | BIT   | #SW14,2#SWR    | :LOOP ON TEST                         |
| 007144 | 001026 |        |        |         | BNE   | KIT            |                                       |
| 007146 | 032737 | 004000 | 177570 |         | BIT   | #SW11,2#SWR    | :KILL ITERATIONS                      |
| 007154 | 001012 |        |        |         | BNE   | SAVLAD         |                                       |
| 007156 | 105767 | 171617 |        |         | TSTB  | ICNT+1         |                                       |
| 007162 | 001404 |        |        |         | BEQ   | 2\$            | :BRANCH IF FIRST                      |
| 007164 | 126767 | 002762 | 171607 |         | CMPB  | TIMES,ICNT+1   | :DONE?                                |
| 007172 | 001013 |        |        |         | BNE   | KIT            | :BRANCH IF NOT                        |
| 007174 | 112767 | 000001 | 171577 | 2\$:    | MOVB  | #1,ICNT+1      | :FIRST ITERATION                      |
| 007202 | 105267 | 171572 |        | SAVLAD: | INCB  | ICNT           | :COUNT TEST NUMBERS                   |
| 007206 | 011667 | 002732 |        |         | MOV   | (6) LAD        | :SAVE LOOP ADDRESS                    |
| 007212 | 016737 | 171562 | 177570 |         | MOV   | ICNT,2#DISPLAY | :DISPLAY TEST NO. AND ITERATION COUNT |
| 007220 | 000002 |        |        |         | RTI   |                | :RETURN                               |
| 007222 | 105267 | 171553 |        | KIT:    | INCB  | ICNT+1         |                                       |
| 007226 | 016737 | 171546 | 177570 | OVER:   | MOV   | ICNT,2#DISPLAY | :SET UP DISPLAY                       |
| 007234 | 005767 | 002704 |        |         | TST   | LAD            | :FIRST ONE?                           |
| 007240 | 001760 |        |        |         | BEQ   | SAVLAD         |                                       |
| 007242 | 016716 | 002676 |        |         | MOV   | LAD,(6)        | :FUDGE RETURN ADDRESS                 |
| 007246 | 000002 |        |        |         | RTI   |                | :FIXES PS                             |



|        |        |        |              |                |                                    |
|--------|--------|--------|--------------|----------------|------------------------------------|
| 007250 | 000134 |        | \$POLSH: JMP | 2(R4)+         |                                    |
| 007252 | 016746 | 171532 | \$P.4A: MOV  | LONUM+6, -(SP) |                                    |
| 007256 | 016746 | 171524 |              | MOV            | LONUM+4, -(SP)                     |
| 007262 | 016746 | 171516 | \$P.2A: MOV  | LONUM+2, -(SP) |                                    |
| 007266 | 016746 | 171510 |              | MOV            | LONUM, -(SP)                       |
| 007272 | 000134 |        |              | JMP            | 2(R4)+                             |
| 007274 | 016746 | 171520 | \$P.4B: MOV  | HINUM+6, -(SP) |                                    |
| 007300 | 016746 | 171512 |              | MOV            | HINUM+4, -(SP)                     |
| 007304 | 016746 | 171504 | \$P.2B: MOV  | HINUM+2, -(SP) |                                    |
| 007310 | 016746 | 171476 |              | MOV            | HINUM, -(SP)                       |
| 007314 | 000134 |        |              | JMP            | 2(R4)+                             |
| 007316 | 012667 | 171510 | \$POP2X: MOV | (SP)+, ANS2    |                                    |
| 007322 | 012667 | 171506 |              | MOV            | (SP)+, ANS2+2                      |
| 007326 | 000204 |        |              | RTS            | R4                                 |
|        |        |        |              |                | :EXIT POLISH MODE                  |
| 007330 | 012667 | 171476 | \$POP4X: MOV | (SP)+, ANS2    |                                    |
| 007334 | 012667 | 171474 |              | MOV            | (SP)+, ANS2+2                      |
| 007340 | 012667 | 171472 |              | MOV            | (SP)+, ANS2+4                      |
| 007344 | 012667 | 171470 |              | MOV            | (SP)+, ANS2+6                      |
| 007350 | 000204 |        |              | RTS            | R4                                 |
|        |        |        |              |                | :EXIT POLISH MODE                  |
| 007352 | 032767 | 100002 | \$TST: BIT   | #100002, \$FPS | :CHECK FOR ERROR FLAG AND OVERFLOW |
| 007360 | 100437 |        |              | \$TER          | :BRANCH IF FLAG SET                |
| 007362 | 001045 |        |              | \$TOV          | :BRANCH IF OVERFLOW                |
| 007364 | 032716 | 077600 |              | BIT            | #077600, (6)                       |
| 007370 | 001025 |        |              | \$TST1         | :TEST THE EXPONENT                 |
| 007372 | 052767 | 000004 |              | BIS            | #04, \$FPS                         |
| 007400 | 032767 | 077600 |              | BIT            | #077600, LONUM                     |
| 007406 | 001404 |        |              | \$TST3         | :BRANCH IF NOT ZERO                |
| 007410 | 032767 | 030000 |              | BIT            | #030000, \$FPS                     |
| 007416 | 001412 |        |              | \$TST1         | :SET Z BIT                         |
| 007420 | 005016 |        | \$TST3: CLR  | (6)            | :CHECK OPERAND FOR ZERO EXPONENT   |
| 007422 | 005066 | 000002 |              | CLR            | 2(6)                               |
| 007426 | 105767 | 171416 |              | TSTB           | \$FPS                              |
| 007432 | 100004 |        |              | \$TST1         | :BRANCH IF NOT                     |
| 007434 | 005066 | 000004 |              | CLR            | 4(6)                               |
| 007440 | 005066 | 000006 |              | CLR            | 6(6)                               |
| 007444 | 005716 |        | \$TST1: TST  | (6)            | :FIND THE SIGN                     |
| 007446 | 100003 |        |              | \$TST2         | :BRANCH IF PLUS                    |
| 007450 | 052767 | 000010 |              | BIS            | #10, \$FPS                         |
| 007456 | 000134 |        | \$TST2: JMP  | 2(R4)+         | :SET N BIT                         |
| 007460 | 116701 | 171314 | \$TER: MOV   | ICNT, R1       | :GET TEST NUMBER                   |
| 007464 | 005301 |        |              | DEC            | R1                                 |
| 007466 | 006301 |        |              | ASL            | R1                                 |
| 007470 | 016167 | 007514 |              | MOV            | RETAD(1), \$FPC                    |
| 007476 | 022626 |        | \$TOV: CMP   | (SP)+, (SP)+   | :STORE FORTRAN FPC                 |
| 007500 | 105767 | 171344 |              | TSTB           | \$FPS                              |
| 007504 | 100001 |        |              | \$TOV1         | :CHECK FD BIT                      |
| 007506 | 022626 |        |              | CMP            | (SP)+, (SP)+                       |
| 007510 | 005724 |        | \$TOV1: TST  | (R4)+          | :BRANCH IF FLOATING MODE           |
| 007512 | 000204 |        |              | RTS            | R4                                 |
|        |        |        |              |                | :POP 2 MORE WORDS                  |
|        |        |        |              |                | :SKIP "POPX" ROUTINE               |
|        |        |        |              |                | :EXIT POLISH MODE                  |

|        |        |        |        |          |         |          |         |                                          |
|--------|--------|--------|--------|----------|---------|----------|---------|------------------------------------------|
| 007514 | 001400 |        |        | RETAD:   | RET1    |          |         |                                          |
| 007516 | 001666 |        |        |          | RET2    |          |         |                                          |
| 007520 | 000000 |        |        |          | DUMMY   |          |         |                                          |
| 007522 | 000000 |        |        |          | DUMMY   |          |         |                                          |
| 007524 | 002640 |        |        |          | RET5    |          |         |                                          |
| 007526 | 003150 |        |        |          | RET6    |          |         |                                          |
| 007530 | 000000 |        |        |          | DUMMY   |          |         |                                          |
| 007532 | 000000 |        |        |          | DUMMY   |          |         |                                          |
| 007534 | 004266 |        |        |          | RET11   |          |         |                                          |
| 007536 | 004554 |        |        |          | RET12   |          |         |                                          |
| 007540 | 000000 |        |        |          | DUMMY   |          |         |                                          |
| 007542 | 000000 |        |        |          | DUMMY   |          |         |                                          |
| 007544 | 005526 |        |        |          | RET15   |          |         |                                          |
| 007546 | 006036 |        |        |          | RET16   |          |         |                                          |
| 007550 | 000000 |        |        |          | DUMMY   |          |         |                                          |
| 007552 | 000000 |        |        |          | DUMMY   |          |         |                                          |
| 007554 | 026527 | 000002 | 001005 | \$ERR:   | CMP     | 2(5)     | #1005   | :CHECK FOR UNDERFLOW, FD=0               |
| 007562 | 001415 |        |        |          | BEQ     | \$UNDR0  |         | :BRANCH IF UNDERFLOW                     |
| 007564 | 026527 | 000002 | 000405 |          | CMP     | 2(5)     | #0405   | :CHECK FOR UNDERFLOW, FD=1               |
| 007572 | 001427 |        |        |          | BEQ     | \$UNDR1  |         | :BRANCH IF UNDERFLOW                     |
| 007574 | 026527 | 000002 | 001003 |          | CMP     | 2(5)     | #1003   | :CHECK FOR OVERFLOW, FD=0                |
| 007602 | 001474 |        |        |          | BEQ     | \$OVER0  |         | :BRANCH IF OVERFLOW                      |
| 007604 | 026527 | 000002 | 000403 |          | CMP     | 2(5)     | #0403   | :CHECK FOR OVERFLOW, FD=1                |
| 007612 | 001502 |        |        |          | BEQ     | \$OVER1  |         | :BRANCH IF OVERFLOW                      |
| 007614 | 000000 |        |        | \$UNKER: | HALT    |          |         | :UNKNOWN ERROR!                          |
| 007616 | 032767 | 003000 | 171224 | \$UNDR0: | BIT     | #003000, | \$FPS   | :CHECK FOR INTERRUPTS ON OR OFF          |
| 007624 | 001523 |        |        |          | BEQ     | \$ERTS   |         | :BRANCH IF OFF                           |
| 007626 | 004467 | 177416 |        |          | JSR     | R4,      | \$POLSH | :ENTER POLISH MODE                       |
| 007632 | 007262 |        |        |          | \$P.2A  |          |         | :PUSH 2 WORDS ON STACK (LONUM)           |
| 007634 | 007304 |        |        |          | \$P.2B  |          |         | :PUSH 2 WORDS ON STACK (HINUM)           |
| 007636 | 007722 |        |        |          | \$200A0 |          |         | :ADD 200 TO THE EXPONENTS                |
| 007640 | 010142 |        |        | .ERR1:   | .ERR    |          |         | :ADDRESS OF FORTRAN ADD OR SUBTRACT      |
|        |        |        |        |          |         |          |         | :THIS ADDRESS IS MODIFIED BY THE PROGRAM |
| 007642 | 007752 |        |        |          | \$200SB |          |         | :SUBTRACT 200 FROM THE EXPONENT OF ANS   |
| 007644 | 007352 |        |        |          | \$TST   |          |         | :DETERMINE CONDITION CODES               |
| 007646 | 007316 |        |        |          | \$POP2X |          |         | :POP 2 WORDS AND EXIT POLISH MODE        |
| 007650 | 000415 |        |        |          | BR      | \$UNDR0  |         |                                          |
| 007652 | 032767 | 003000 | 171170 | \$UNDR1: | BIT     | #003000, | \$FPS   | :CHECK FOR INTERRUPTS ON OR OFF          |
| 007660 | 001505 |        |        |          | BEQ     | \$ERTS   |         | :BRANCH IF OFF                           |
| 007662 | 004467 | 177362 |        |          | JSR     | R4,      | \$POLSH | :ENTER POLISH MODE                       |
| 007666 | 007252 |        |        |          | \$P.4A  |          |         | :PUSH 2 WORDS ON STACK (LONUM)           |
| 007670 | 007274 |        |        |          | \$P.4B  |          |         | :PUSH 2 WORDS ON STACK (HINUM)           |
| 007672 | 007736 |        |        |          | \$200A1 |          |         | :ADD 200 TO THE EXPONENTS                |
| 007674 | 010142 |        |        | .ERR2:   | .ERR    |          |         | :ADDRESS OF FORTRAN ADD OR SUBTRACT      |
|        |        |        |        |          |         |          |         | :THIS ADDRESS IS MODIFIED BY THE PROGRAM |
| 007676 | 007752 |        |        |          | \$200SB |          |         | :SUBTRACT 200 FROM THE EXPONENT OF ANS   |
| 007700 | 007352 |        |        |          | \$TST   |          |         | :DETERMINE CONDITION CODES               |
| 007702 | 007330 |        |        |          | \$POP4X |          |         | :POP 4 WORDS AND EXIT POLISH MODE        |
| 007704 | 052767 | 100000 | 171136 | \$UNDR0: | BIS     | #100000, | \$FPS   | :SET FPS ERROR FLAG                      |
| 007712 | 012767 | 000012 | 171132 |          | MOV     | #12,     | \$FEC   | :SET UNDERFLOW EXCEPTION CODE            |
| 007720 | 000205 |        |        |          | RTS     | %5       |         | :RETURN TO FORTRAN ROUTINE               |
| 007722 | 062716 | 040000 |        | \$200A0: | ADD     | #040000, | (SP)    |                                          |



```

007726 062766 040000 000004      ADD      #040000,4(SP)
007734 000134                      JMP      @ (R4)+

007736 062716 040000          $200A1: ADD      #040000,(SP)
007742 062766 040000 000010      ADD      #040000,10(SP)
007750 000134                      JMP      @ (R4)+

007752 162716 040000          $200SB: SUB      #040000,(SP)
007756 032767 000003 170012      BIT      #3, PS      ;TEST FOR C OR V BITS
007764 001402                      BEQ      $2015B
007766 062716 100000          ADD      #100000,(SP)
007772 000134          $2015B: JMP      @ (R4)+

007774 004467 177250          $OVER0: JSR      R4, $POLSH ;ENTER POLISH MODE
010000 007262                      $P.2A      ;PUSH 2 WORDS ON STACK (LONUM)
010002 007304                      $P.2B      ;PUSH 2 WORDS ON STACK (HINUM)
010004 010104                      $200S0      ;SUBTRACT 200 FROM THE EXPONENTS
010006 010142          .ERR3: .ERR      ;ADDRESS OF FORTRAN ADD OR SUBTRACT
                                           ;THIS ADDRESS IS MODIFIED BY THE PROGRAM
                                           ;ADD 200 TO THE EXPONENT OF ANS
                                           ;DETERMINE CONDITION CODES
                                           ;POP 2 WORDS AND EXIT POLISH MODE

010010 010134                      $200AD
010012 007352                      $TST
010014 007316                      $POP2X
010016 000411                      BR      $OVERA

010020 004467 177224          $OVER1: JSR      R4, $POLSH ;ENTER POLISH MODE
010024 007252                      $P.4A      ;PUSH 2 WORDS ON STACK (LONUM)
010026 007274                      $P.4B      ;PUSH 2 WORDS ON STACK (HINUM)
010030 010120                      $200S1      ;SUBTRACT 200 FROM THE EXPONENTS
010032 010142          .ERR4: .ERR      ;ADDRESS OF FORTRAN ADD OR SUBTRACT
                                           ;THIS ADDRESS IS MODIFIED BY THE PROGRAM
                                           ;ADD 200 TO THE EXPONENT OF ANS
                                           ;DETERMINE CONDITION CODES
                                           ;POP 4 WORDS AND EXIT POLISH MODE

010034 010134                      $200AD
010036 007352                      $TST
010040 007330                      $POP4X
010042 032767 003000 171000 $OVERA: BIT      #003000,$FPS
010050 001406                      BEQ      $OVR1
010052 052767 100000 170770      BIS      #100000,$FPS
010060 012767 000010 170764      MOV      #10, $FEC
010066 052767 000002 170754 $OVR1: BIS      #02, $FPS
010074 052767 000004 170746 $ERTS: BIS      #04, $FPS
010102 000205                      RTS      %5
                                           ;SET FPS ERROR FLAG
                                           ;SET OVERFLOW EXCEPTION CODE
                                           ;SET OVERFLOW BIT IN FPS
                                           ;SET THE Z BIT IN FPS

010104 162716 040000          $200S0: SUB      #040000,(SP)
010110 162766 040000 000004      SUB      #040000,4(SP)
010116 000134                      JMP      @ (R4)+

010120 162716 040000          $200S1: SUB      #040000,(SP)
010124 162766 040000 000010      SUB      #040000,10(SP)
010132 000134                      JMP      @ (R4)+

010134 062716 140000          $200AD: ADD      #140000,(SP)
010140 000134                      JMP      @ (R4)+

010142 000000          .ERR:  HALT
010144 000137 000200          JMP      @#200      ;PROGRAM DID NOT START AT 200

```

;\*\*\*\*\*

## ;FPP INTERRUPT SERVICE ROUTINE

;\*\*\*\*\*

|        |        |        |         |         |               |                                      |
|--------|--------|--------|---------|---------|---------------|--------------------------------------|
| 010150 | 170267 | 170666 | FLTERR: | STFPS   | FPS           | ;STORE FLOATING POINT STATUS         |
| 010154 | 170367 | 170664 |         | STST    | FEC           | ;STORE FLOATING EXCEPTION CODES      |
| 010160 | 032767 | 040000 | 170662  | BIT     | #40000, \$FPS | ;CHECK INTERRUPT DISABLE             |
| 010166 | 001402 |        |         | BEQ     | +.6           | ;BRANCH IF OFF                       |
| 010170 | 104377 |        |         | HLT+377 |               | ;INTERUPT NOT SUPPOSED TO BE ENABLED |
| 010172 | 000404 |        |         | BR      | ERRTI         |                                      |
| 010174 | 005767 | 170650 |         | TST     | \$FPS         | ;CHECK FORTRAN FPS FOR ERROR FLAG    |
| 010200 | 100401 |        |         | BMI     | ERRTI         | ;BRANCH IF FLAG SET                  |
| 010202 | 104377 |        |         | HLT+377 |               | ;FLOATING POINT STATUS ERROR         |
| 010204 | 000002 |        | ERRTI:  | RTI     |               |                                      |



|        |        |        |             |             |                                |
|--------|--------|--------|-------------|-------------|--------------------------------|
| 010206 | 010046 |        | RANDM2: MOV | %0,-(6)     | ;SAVE R0                       |
| 010210 | 010146 |        | MOV         | %1,-(6)     | ;SAVE R1                       |
| 010212 | 010246 |        | MOV         | %2,-(6)     | ;SAVE R2                       |
| 010214 | 010346 |        | MOV         | %3,-(6)     | ;SAVE R3                       |
| 010216 | 010446 |        | MOV         | %4,-(6)     | ;SAVE R4                       |
| 010220 | 010546 |        | MOV         | %5,-(6)     | ;SAVE R5                       |
| 010222 | 012704 | 001002 | MOV         | #LONUM,%4   | ;SET UP LONUM POINTER          |
| 010226 | 012705 | 001012 | MOV         | #HINUM,%5   | ;SET UP HINUM POINTER          |
| 010232 | 011400 |        | MOV         | (4),%0      | ;SET R0 WITH LOW               |
| 010234 | 011501 |        | MOV         | (5),%1      | ;SET R1 WITH HIGH              |
| 010236 | 012703 | 177771 | REPET2: MOV | #-7,%3      | ;SET SHIFT COUNT               |
| 010242 | 005002 |        | CLR         | %2          |                                |
| 010244 | 006300 |        | SHIFT2: ASL | %0          | ;SHIFT R0 LEFT AND             |
| 010246 | 006101 |        | ROL         | %1          | ;ROTATE CARRY INTO R1 AND      |
| 010250 | 006102 |        | ROL         | %2          | ;ROTATE CARRY INTO R2          |
| 010252 | 005203 |        | INC         | %3          | ;CHECK FOR DONE                |
| 010254 | 001373 |        | BNE         | SHIFT2      | ;CONTINUE SHIFT LOOP           |
| 010256 | 061402 |        | ADD         | (4),%2      | ;ADD NUMBER TO MAKE X 129      |
| 010260 | 005501 |        | ADC         | %1          | ;PROPOGATE CARRY               |
| 010262 | 061501 |        | ADD         | (5),%1      | ;ADD NUMBER TO MAKE X 129      |
| 010264 | 005502 |        | ADC         | %2          | ;PROPOGATE CARRY               |
| 010266 | 062700 | 001057 | ADD         | #1057,%0    | ;ADD LOW CONSTANT              |
| 010272 | 005501 |        | ADC         | %1          | ;PROPOGATE CARRY               |
| 010274 | 005502 |        | ADC         | %2          | ;PROPOGATE CARRY               |
| 010276 | 062701 | 047401 | ADD         | #47401,%1   | ;ADD HIGH CONSTANT             |
| 010302 | 005502 |        | ADC         | %2          | ;PROPOGATE CARRY               |
| 010304 | 062702 | 000006 | ADD         | #6,%2       | ;ADD HIGHEST CONSTART          |
| 010310 | 060200 |        | ADD         | %2,%0       | ;REPRIME R0 WITH HIGHEST DIGIT |
| 010312 | 005501 |        | ADC         | %1          | ;PROPOGATE CARRY               |
| 010314 | 010024 |        | MOV         | %0,(4)+     | ;SAVE R0                       |
| 010316 | 010125 |        | MOV         | %1,(5)+     | ;SAVE R1                       |
| 010320 | 020427 | 001006 | CMP         | %4,#LONUM+4 | ;CHECK FOR DONE ENOUGH         |
| 010324 | 001344 |        | BNE         | REPET2      | ;BRANCH IF NOT DONE            |
| 010326 | 012605 |        | MOV         | (6)+,%5     | ;RESTORE R5                    |
| 010330 | 012604 |        | MOV         | (6)+,%4     | ;RESTORE R4                    |
| 010332 | 012603 |        | MOV         | (6)+,%3     | ;RESTORE R3                    |
| 010334 | 012602 |        | MOV         | (6)+,%2     | ;RESTORE R2                    |
| 010336 | 012601 |        | MOV         | (6)+,%1     | ;RESTORE R1                    |
| 010340 | 012600 |        | MOV         | (6)+,%0     | ;RESTORE R0                    |
| 010342 | 000207 |        | RTS         | %7          | ;RETURN                        |

|        |        |        |         |     |              |                                |
|--------|--------|--------|---------|-----|--------------|--------------------------------|
| 010344 | 010046 |        | RANDM4: | MOV | %0,-(6)      | ;SAVE R0                       |
| 010346 | 010146 |        |         | MOV | %1,-(6)      | ;SAVE R1                       |
| 010350 | 010246 |        |         | MOV | %2,-(6)      | ;SAVE R2                       |
| 010352 | 010346 |        |         | MOV | %3,-(6)      | ;SAVE R3                       |
| 010354 | 010446 |        |         | MOV | %4,-(6)      | ;SAVE R4                       |
| 010356 | 010546 |        |         | MOV | %5,-(6)      | ;SAVE R5                       |
| 010360 | 012704 | 001002 |         | MOV | #LONUM,%4    | ;SET UP LONUM POINTER          |
| 010364 | 012705 | 001012 |         | MOV | #HINUM,%5    | ;SET UP HINUM POINTER          |
| 010370 | 011400 |        |         | MOV | (4),%0       | ;SET R0 WITH LOW               |
| 010372 | 011501 |        |         | MOV | (5),%1       | ;SET R1 WITH HIGH              |
| 010374 | 012703 | 177771 | REFET4: | MOV | #-7,%3       | ;SET SHIFT COUNT               |
| 010400 | 005002 |        |         | CLR | %2           |                                |
| 010402 | 006300 |        | SHIFT4: | ASL | %0           | ;SHIFT R0 LEFT AND             |
| 010404 | 006101 |        |         | ROL | %1           | ;ROTATE CARRY INTO R1 AND      |
| 010406 | 006102 |        |         | ROL | %2           | ;ROTATE CARRY INTO R2          |
| 010410 | 005203 |        |         | INC | %3           | ;CHECK FOR DONE                |
| 010412 | 001373 |        |         | BNE | SHIFT4       | ;CONTINUE SHIFT LOOP           |
| 010414 | 061402 |        |         | ADD | (4),%2       | ;ADD NUMBER TO MAKE X 129      |
| 010416 | 005501 |        |         | ADC | %1           | ;PROPOGATE CARRY               |
| 010420 | 061501 |        |         | ADD | (5),%1       | ;ADD NUMBER TO MAKE X 129      |
| 010422 | 005502 |        |         | ADC | %2           | ;PROPOGATE CARRY               |
| 010424 | 062700 | 001057 |         | ADD | #1057,%0     | ;ADD LOW CONSTANT              |
| 010430 | 005501 |        |         | ADC | %1           | ;PROPOGATE CARRY               |
| 010432 | 005502 |        |         | ADC | %2           | ;PROPOGATE CARRY               |
| 010434 | 062701 | 047401 |         | ADD | #47401,%1    | ;ADD HIGH CONSTANT             |
| 010440 | 005502 |        |         | ADC | %2           | ;PROPOGATE CARRY               |
| 010442 | 062702 | 000006 |         | ADD | #6,%2        | ;ADD HIGHEST CONSTART          |
| 010446 | 060200 |        |         | ADD | %2,%0        | ;REPRIME R0 WITH HIGHEST DIGIT |
| 010450 | 005501 |        |         | ADC | %1           | ;PROPOGATE CARRY               |
| 010452 | 010024 |        |         | MOV | %0,(4)+      | ;SAVE R0                       |
| 010454 | 010125 |        |         | MOV | %1,(5)+      | ;SAVE R1                       |
| 010456 | 020427 | 001012 |         | CMP | %4,#LONUM+10 | ;CHECK FOR DONE ENOUGH         |
| 010462 | 001344 |        |         | BNE | REPET4       | ;BRANCH IF NOT DONE            |
| 010464 | 012605 |        |         | MOV | (6)+,%5      | ;RESTORE R5                    |
| 010466 | 012604 |        |         | MOV | (6)+,%4      | ;RESTORE R4                    |
| 010470 | 012603 |        |         | MOV | (6)+,%3      | ;RESTORE R3                    |
| 010472 | 012602 |        |         | MOV | (6)+,%2      | ;RESTORE R2                    |
| 010474 | 012601 |        |         | MOV | (6)+,%1      | ;RESTORE R1                    |
| 010476 | 012600 |        |         | MOV | (6)+,%0      | ;RESTORE R0                    |
| 010500 | 000207 |        |         | RTS | %7           | ;RETURN                        |



|        |        |        |        |         |       |             |                                      |
|--------|--------|--------|--------|---------|-------|-------------|--------------------------------------|
| 010502 | 032767 | 002000 | 167060 | .TRP:   | BIT   | #2000,SWR   |                                      |
| 010510 | 001405 |        |        |         | BEG   | .ET         |                                      |
| 010512 | 012767 | 000207 | 001420 |         | MOV   | #BELL,.TYPE | :TYPE A BELL                         |
| 010520 | 000004 | 012140 |        |         | TYPE, | .TYPE       |                                      |
| 010524 | 005267 | 001416 |        | .ET:    | INC   | ERRORS      | :COUNT THE NUMBER OF ERRORS          |
| 010530 | 032767 | 020000 | 167032 |         | BIT   | #20000,SWR  | :SKIP TYPEOUT IF SET                 |
| 010536 | 001116 |        |        |         | BNE   | NHEAD       |                                      |
| 010540 | 174257 | 170256 |        |         | STF   | AC2, ANS1   | :SAVE THE ANSWER, TO BE SURE         |
| 010544 | 000004 | 012042 |        |         | TYPE, | RET         |                                      |
| 010550 | 000004 | 012042 |        |         | TYPE, | RET         |                                      |
| 010554 | 011646 |        |        |         | MOV   | (6),-(6)    | :PUT ADDRESS OF INSTRUCTION ON STACK |
| 010556 | 162716 | 000002 |        |         | SUB   | #2,(6)      |                                      |
| 010562 | 117667 | 000000 | 001360 |         | MOVB  | 2(6), WORDS |                                      |
| 010570 | 012605 |        |        |         | MOV   | (6)+,TTY    | :TYPE (6)+ IN OCTAL                  |
| 010572 | 004767 | 000546 |        |         | JSR   | %7,PRINTR   | :TYPE LEADING ZERO'S                 |
| 010576 | 000004 | 012037 |        |         | TYPE, | SPACE+3     |                                      |
| 010602 | 012703 | 001002 |        |         | MOV   | #LONUM,%3   | :SET UP POINTER                      |
| 010606 | 004767 | 000174 |        |         | JSR   | 7,TYPE      | :TYPE A FLOATING POINT NUMBER        |
| 010612 | 000004 | 012045 |        |         | TYPE, | \$SIGN      |                                      |
| 010616 | 004767 | 000164 |        |         | JSR   | 7,TYPE      | :TYPE A FLOATING POINT NUMBER        |
| 010622 | 000004 | 012072 |        |         | TYPE, | FPUAN       |                                      |
| 010626 | 004767 | 000154 |        |         | JSR   | 7,TYPE      | :TYPE A FLOATING POINT NUMBER        |
| 010632 | 000004 | 012036 |        |         | TYPE, | SPACE+2     |                                      |
| 010636 | 016705 | 170200 |        |         | MOV   | FPS,TTY     | :TYPE FPS IN OCTAL                   |
| 010642 | 004767 | 000476 |        |         | JSR   | %7,PRINTR   | :TYPE LEADING ZERO'S                 |
| 010646 | 105767 | 001276 |        |         | TSTB  | WORDS       | :CHECK FOR STATUS ERROR              |
| 010652 | 100014 |        |        |         | BPL   | .STAT       | :BRANCH IF NOT                       |
| 010654 | 000004 | 012036 |        |         | TYPE, | SPACE+2     |                                      |
| 010660 | 016705 | 170160 |        |         | MOV   | FEC,TTY     | :TYPE FEC IN OCTAL                   |
| 010664 | 004767 | 000454 |        |         | JSR   | %7,PRINTR   | :TYPE LEADING ZERO'S                 |
| 010670 | 000004 | 012036 |        |         | TYPE, | SPACE+2     |                                      |
| 010674 | 016705 | 170146 |        |         | MOV   | FPC,TTY     | :TYPE FPC IN OCTAL                   |
| 010700 | 004767 | 000440 |        |         | JSR   | %7,PRINTR   | :TYPE LEADING ZERO'S                 |
| 010704 | 000004 | 012053 |        | .STAT:  | TYPE, | FORTAN      |                                      |
| 010710 | 004767 | 000072 |        |         | JSR   | 7,TYPE      | :TYPE A FLOATING POINT NUMBER        |
| 010714 | 105767 | 001230 |        | NOHEAD: | TSTB  | WORDS       |                                      |
| 010720 | 001425 |        |        |         | BEG   | NHEAD       |                                      |
| 010722 | 000004 | 012036 |        |         | TYPE, | SPACE+2     |                                      |
| 010726 | 016705 | 170116 |        |         | MOV   | \$FPS,TTY   | :TYPE \$FPS IN OCTAL                 |
| 010732 | 004767 | 000406 |        |         | JSR   | %7,PRINTR   | :TYPE LEADING ZERO'S                 |
| 010736 | 105767 | 001206 |        |         | TSTB  | WORDS       |                                      |
| 010742 | 100014 |        |        |         | BPL   | NHEAD       |                                      |
| 010744 | 000004 | 012036 |        |         | TYPE, | SPACE+2     |                                      |
| 010750 | 016705 | 170076 |        |         | MOV   | \$FEC,TTY   | :TYPE \$FEC IN OCTAL                 |
| 010754 | 004767 | 000364 |        |         | JSR   | %7,PRINTR   | :TYPE LEADING ZERO'S                 |
| 010760 | 000004 | 012036 |        |         | TYPE, | SPACE+2     |                                      |
| 010764 | 016705 | 170064 |        |         | MOV   | \$FPC,TTY   | :TYPE \$FPC IN OCTAL                 |
| 010770 | 004767 | 000350 |        |         | JSR   | %7,PRINTR   | :TYPE LEADING ZERO'S                 |
| 010774 | 005737 | 177570 |        | NHEAD:  | TST   | #SWR        |                                      |
| 011000 | 100001 |        |        |         | BPL   | .+4         |                                      |
| 011002 | 000000 |        |        |         | HALT  |             |                                      |
| 011004 | 000002 |        |        |         | RTI   |             |                                      |

```

011006 032767 001000 166554 $TYPE: BIT #1000, SWR ;CHECK TTY FORMAT
011014 001007 BNE TYPEI
011016 105767 170026 TSTB $FPS
011022 100432 BMI TYPED
011024 004767 000076 JSR 7, TYPEF
011030 022323 TYPEA: CMP (3)+, (3)+ ;UP DATE THE TYPEOUT POINTER
011032 000207 RTS 7

011034 TYPEI:
011034 012305 MOV (3)+, TTY ;TYPE (3)+ IN OCTAL
011036 004767 000302 JSR %7, PRINTR ;TYPE LEADING ZERO'S
011042 000004 012040 TYPE, SPACE+4
011046 012305 MOV (3)+, TTY ;TYPE (3)+ IN OCTAL
011050 004767 000270 JSR %7, PRINTR ;TYPE LEADING ZERO'S
011054 105767 167770 TSTB $FPS
011060 100363 BPL TYPEA
011062 000004 012040 TYPE, SPACE+4
011066 012305 MOV (3)+, TTY ;TYPE (3)+ IN OCTAL
011070 004767 000250 JSR %7, PRINTR ;TYPE LEADING ZERO'S
011074 000004 012040 TYPE, SPACE+4
011100 012305 MOV (3)+, TTY ;TYPE (3)+ IN OCTAL
011102 004767 000236 JSR %7, PRINTR ;TYPE LEADING ZERO'S
011106 000207 RTS 7

011110 012346 TYPED: MOV (3)+, -(6) ;GET WORD 1
011112 012346 MOV (3)+, -(6) ;GET WORD 2
011114 012346 MOV (3)+, -(6) ;GET WORD 3
011116 012346 MOV (3)+, -(6) ;GET WORD 4
011120 012746 000022 MOV #18, -(6) ;CHAR COUNT
011124 000406 BR TYPE1
011126 012346 TYPEF: MOV (3)+, -(6) ;GET WORD 1
011130 012346 MOV (3)+, -(6) ;GET WORD 2
011132 005046 CLR -(6) ;CLEAR WORD 3
011134 005046 CLR -(6)
011136 012746 000010 MOV #8, -(6) ;CHAR COUNT
011142 004767 000104 TYPE1: JSR 7, TY1 ;TYPE 1 BIT
011146 105766 000011 TSTB 9, (6) ;CHECK EXPONENT
011152 001001 BNE +4 ;BRANCH ON NON-ZERO EXPONENT
011154 005116 COM (6) ;FLAG ZERO EXPONENT
011156 000004 012040 TYPE, SPACE+4
011162 004767 000072 JSR 7, TY2 ;TYPE 2 BITS
011166 004767 000074 JSR 7, TY3 ;TYPE 3 BITS
011172 004767 000070 JSR 7, TY3 ;TYPE 3 BITS
011176 000004 012040 TYPE, SPACE+4
011202 004767 000020 TYPE2: JSR 7, TYH ;TYPE 2 BITS
011206 004767 000054 JSR 7, TY3 ;TYPE 3 BITS
011212 005316 DEC (6) ;DONE?
011214 001374 BNE TYPE2
011216 022626 CMP (6)+, (6)+ ;RESTORE
011220 022626 CMP (6)+, (6)+ ;THE
011222 005726 TST (6)+ ;STACK
011224 000207 RTS 7

```



|        |        |               |      |       |            |                                |
|--------|--------|---------------|------|-------|------------|--------------------------------|
| 011226 | 005766 | 000002        | TYH: | TST   | 2(6)       | :CHECK FOR ZERO EXPONENT FLAG  |
| 011232 | 100405 |               |      | BMI   | TYZ        | :BRANCH ON ZERO EXPONENT       |
| 011234 | 012746 | 000001        |      | MOV   | #1, -(6)   | :TYPE HIDDEN BIT AND ONE       |
| 011240 | 011667 | 000674        |      | MOV   | (6), .TYPE | :FUDGE HIDDEN BIT              |
| 011244 | 000414 |               |      | BR    | TY+4       |                                |
| 011246 | 005166 | 000002        | TYZ: | COM   | 2(6)       | :GET RID OF ZERO EXPONENT FLAG |
| 011252 | 012746 | 000001        | TY1: | MOV   | #1, -(6)   | :TYPE 1 BIT                    |
| 011256 | 000405 |               |      | BR    | TY         |                                |
| 011260 | 012746 | 000002        | TY2: | MOV   | #2, -(6)   | :TYPE 2 BITS                   |
| 011264 | 000402 |               |      | BR    | TY         |                                |
| 011266 | 012746 | 000003        | TY3: | MOV   | #3, -(6)   | :TYPE 3 BITS                   |
| 011272 | 005067 | 000642        | TY:  | CLR   | .TYPE      |                                |
| 011276 | 006166 | 000006        |      | ROL   | 6(6)       | :SHIFT WORD 4                  |
| 011302 | 006166 | 000010        |      | ROL   | 8(6)       | :SHIFT WORD 3                  |
| 011306 | 006166 | 000012        |      | ROL   | 10(6)      | :SHIFT WORD 2                  |
| 011312 | 006166 | 000014        |      | ROL   | 12(6)      | :SHIFT WORD 1                  |
| 011316 | 006167 | 000616        |      | ROL   | .TYPE      | :GET IT                        |
| 011322 | 005316 |               |      | DEC   | (6)        | :DONE?                         |
| 011324 | 001364 |               |      | BNE   | TY+4       |                                |
| 011326 | 052767 | 000060 000604 |      | BIS   | #0, .TYPE  | :MAKE IT ASCII                 |
| 011334 | 000004 | 012140        |      | TYPE. | .TYPE      | :TYPE IT                       |
| 011340 | 005726 |               |      | TST   | (6)+       | :RESTORE THE STACK             |
| 011342 | 000207 |               |      | RTS   | 7          |                                |

```

011344 112767 000001 000130 PRINTR: MOVB #1, .PR ;SET ZERO FILL SWITCH
011352 000402 BR .+6
011354 005067 000122 PRINTS: CLR .PR ;SUPPRESS LEADING ZERO'S
011360 112767 177772 000115 MOVB # -6, .PR+1 ;SET COUNT
011366 010446 MOV R4, -(6) ;SAVE R4
011370 012704 011472 MOV #3$, R4 ;SET POINTER TO FIRST ASCII CHAR.
011374 105014 CLRB (4) ;CLEAR FIRST BYTE
011376 000405 BR 2$ ;ROTATE FIRST BIT
011400 105014 1$: CLRB (4) ;CLEAR BYTE OF CHARACTER
011402 006105 ROL TTY ;ROTATE BIT INTO C
011404 106114 ROLB (4) ;PACK IT
011406 006105 ROL TTY ;ROTATE BIT INTO C
011410 106114 ROLB (4) ;PACK IT
011412 006105 2$: ROL TTY ;ROTATE BIT INTO C
011414 106114 ROLB (4) ;PACK IT
011416 105714 TSTB (4)
011420 001402 BEQ .+6
011422 105267 000054 INCB .PR
011426 105767 000050 TSTB .PR ;CHECK FILL SWITCH
011432 001402 BEQ .+6
011434 152724 000060 BISB #0, (4)+ ;MAKE INTO ASCII CHAR
011440 105267 000037 INCB .PR+1
011444 001355 BNE 1$ ;REPEAT
011446 022704 011472 CMP #3$, R4
011452 001002 BNE .+6
011454 112724 000060 MOVB #0, (4)+
011460 105014 CLRB (4)
011462 000004 011472 TYPE 3$ ;TYPE IT
011466 012604 MOV (6)+, R4 ;RESTORE R4
011470 000207 RTS PC

011472 000004 3$: .BLKW 4
011502 000000 .PR: 0

011504 005267 000436 ERROR: INC ERRORS ;COUNT ERRORS
011510 132737 000001 000041 BITB #1, @#41 ;AUTO MODE?
011516 001412 BEQ 1$ ;NO!
011520 022767 000010 000420 CMP #10, ERRORS ;TOO MANY?
011526 001006 BNE 1$ ;NOT YET
011530 013700 000042 MOV @#42, R0 ;GET ADDRESS
011534 001403 BEQ 1$ ;FORGET IT IF ZERO
011536 005037 000042 CLR @#42 ;ZAP 42
011542 004710 JSR PC, (0) ;CALL THE MONITOR
011544 000207 1$: RTS PC ;RETURN

```



```

011546 012777 011742 000360 POWDWN: MOV #ILLUP, @UPVEC ;SET FOR FAST UP
011554 012777 000340 000354 MOV #340, @UPVEC+2 ;PRIO:7
011562 170246 STFPS -(6) ;GET THE FPS
011564 170011 SETD ;
011566 174046 STD ACO, -(6) ;SAVE AC'S
011570 174146 STD AC1, -(6)
011572 174246 STD AC2, -(6)
011574 174346 STD AC3, -(6)
011576 172404 LDD AC4, ACO
011600 174046 STD ACO, -(6)
011602 172405 LDD AC5, ACO
011604 174046 STD ACO, -(6)
011606 010046 MOV RO, -(6) ;SAVE REGISTERS
011610 010146 MOV R1, -(6)
011612 010246 MOV R2, -(6)
011614 010346 MOV R3, -(6)
011616 010446 MOV R4, -(6)
011620 010546 MOV R5, -(6)
011622 010667 000266 MOV SP, SAVE6 ;SAVE SP
011626 012777 011636 000300 MOV #POWUP, @UPVEC ;SET UP VECTOR
011634 000000 HALT

011636 016706 000252 POWUP: MOV SAVE6, SP ;GET SP
011642 005001 CLR R1 ;WAIT LOOP FOR THE TTY
011644 005201 1$: INC R1
011646 001376 BNE 1$
011650 012605 MOV (6)+, R5 ;GET THE REGISTERS
011652 012604 MOV (6)+, R4
011654 012603 MOV (6)+, R3
011656 012602 MOV (6)+, R2
011660 012601 MOV (6)+, R1
011662 012600 MOV (6)+, RO
011664 170011 SETD ;
011666 172426 LDD (6)+, ACO ;RESTORE THE AC'S
011670 174005 STD ACO, AC5
011672 172426 LDD (6)+, ACO
011674 174004 STD ACO, AC4
011676 172726 LDD (6)+, AC3
011700 172626 LDD (6)+, AC2
011702 172526 LDD (6)+, AC1
011704 172426 LDD (6)+, ACO
011706 170126 LDFPS (6)+ ;RESTORE FPS
011710 012777 011546 000212 MOV #POWDWN, @DWNVEC ;SET UP THE POWER DOWN VECTOR
011716 012777 000340 000206 MOV #340, @DWNVEC+2
011724 000004 011730 TYPE, .+2 ;.ASCIZ <15><12>"POWER"
011740 000002 RTI

011742 000000 ILLUP: HALT ;THE POWER UP SEQUENCE WAS STARTED
011744 000776 BR .-2 ;BEFORE THE POWER DOWN WAS COMPLETE

```

|        |        |        |        |          |          |                       |                                      |
|--------|--------|--------|--------|----------|----------|-----------------------|--------------------------------------|
| 011746 | 010546 |        |        | .IOT:    | MOV      | TTY, -(6)             | :SAVE TTY                            |
| 011750 | 017605 | 000002 |        |          | MOV      | 22(6), TTY            | :GET ADDRESS TO BE TYPED             |
| 011754 | 105715 |        |        | 1\$:     | TSTB     | (TTY)                 | :TERMINATOR?                         |
| 011756 | 001406 |        |        |          | BEG      | 2\$                   |                                      |
| 011760 | 112577 | 000134 |        |          | MOVB     | (TTY)+, 2TPB          | :LOAD AND TYPE THE CHARACTER         |
| 011764 | 105777 | 000126 |        |          | TSTB     | 2TPB                  | :IS THE PRINTER READY                |
| 011770 | 100375 |        |        |          | BPL      | -4                    |                                      |
| 011772 | 000770 |        |        |          | BR       | 1\$                   | :GET THE NEXT CHARACTER              |
| 011774 | 017646 | 000002 |        | 2\$:     | MOV      | 22(6), -(6)           | :GET ADDRESS TO BE TYPED             |
| 012000 | 062766 | 000002 | 000004 |          | ADD      | 2, 4(6)               | :ADD 2 TO THE ADDRESS                |
| 012006 | 022666 | 000002 |        |          | CMP      | (6)+, 2(6)            | :IS IT .+2?                          |
| 012012 | 001006 |        |        |          | BNE      | 3\$                   | :NO                                  |
| 012014 | 062705 | 000002 |        |          | ADD      | 2, TTY                | :ADD 2 TO THE ADDRESS                |
| 012020 | 042705 | 000001 |        |          | BIC      | 1, TTY                | :BACK UP TO AN EVEN BYTE             |
| 012024 | 010566 | 000002 |        |          | MOV      | TTY, 2(6)             | :RESTORE ADDRESS                     |
| 012030 | 012605 |        |        | 3\$:     | MOV      | (6)+, TTY             | :RESTORE TTY                         |
| 012032 | 000002 |        |        |          | RTI      |                       | :RETURN                              |
| 012034 | 005015 | 020040 | 000040 | SPACE:   | .ASCIZ   | <15><12>" "           |                                      |
| 012042 | 005015 | 000    |        | RET:     | .ASCIZ   | <15><12>              |                                      |
| 012045 | 040    | 020077 | 000    | \$SIGN:  | .ASCIZ   | " ? "                 |                                      |
| 012051 | 053    |        |        | \$PLUS:  | .ASCIZ   | " + "                 |                                      |
| 012052 | 055    |        |        | \$MINUS: | .ASCIZ   | " - "                 |                                      |
| 012053 | 015    | 020012 | 043040 | FORTAN:  | .ASCIZ   | <15><12>" FORTRAN: "  |                                      |
| 012060 | 051117 | 051124 | 047101 |          |          |                       |                                      |
| 012066 | 020072 | 000040 |        |          |          |                       |                                      |
| 012072 | 036440 | 005015 | 020040 | FPUAN:   | .ASCIZ   | " = "<15><12>" FPU: " |                                      |
| 012100 | 050106 | 035125 | 020040 |          |          |                       |                                      |
| 012106 | 020040 | 020040 | 000    |          |          |                       |                                      |
| 012114 | 000000 |        |        | .EVEN    |          |                       |                                      |
| 012116 | 177564 |        |        | SAVE6:   | 0        |                       |                                      |
| 012120 | 177566 |        |        | TPS:     | 177564   |                       | :TELEPRINTER STATUS REGISTER         |
| 012122 | 172160 |        |        | TPB:     | 177566   |                       | :TELEPRINTER DATA BUFFER             |
| 012124 | 000244 | 000246 |        | FPTADR:  | 172160   |                       | :FLOATING POINT ADDRESS ON THE 11/20 |
| 012130 | 000024 | 000026 |        | FPVECT:  | 244, 246 |                       | :FLOATING POINT VECTOR ADDRESS       |
| 012134 | 000024 | 000026 |        | DWNVEC:  | 24, 26   |                       | :POWER DOWN VECTOR ADDRESS           |
| 012140 | 000000 |        |        | UPVEC:   | 24, 26   |                       | :POWER UP VECTOR ADDRESS             |
| 012142 | 000000 |        |        | .TYPE:   | 0        |                       |                                      |
| 012144 | 000000 |        |        | TRPB:    | 0        |                       |                                      |
| 012146 | 000000 |        |        | LAD:     | 0        |                       | :LOOP ADDRESS                        |
| 012150 | 000000 |        |        | ERRORS:  | 0        |                       | :ERROR COUNT                         |
| 012152 | 000377 |        |        | WORDS:   | 0        |                       | :CONTAINS TYPEOUT INFO               |
|        | 000001 |        |        | TIMES:   | 377      |                       | :ITERATION COUNT                     |
|        |        |        |        | .END     |          |                       |                                      |



|        |          |       |       |       |       |       |       |       |       |       |       |       |       |       |
|--------|----------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| ACC    | =%000000 | 371*  | 483*  | 491   | 556*  | 564   | 631*  | 639   | 688*  | 696   | 747*  | 755   | 826*  | 834   |
|        |          | 309*  | 917   | 971*  | 979   | 1042* | 1050  | 1115* | 1123  | 1190* | 1198  | 1247* | 1255  | 1306* |
|        |          | 1314  | 1384* | 1392  | 1466* | 1474  | 1528* | 1536  | 2065  | 2069* | 2070  | 2071* | 2072  | 2094* |
| AC1    | =%000001 | 2095  | 2096* | 2097  | 2101* |       |       |       |       |       |       |       |       |       |
|        |          | 372*  | 484*  | 492   | 557*  | 565   | 632*  | 640   | 689*  | 697   | 748*  | 756   | 827*  | 835   |
|        |          | 910*  | 918   | 972*  | 980   | 1043* | 1051  | 1116* | 1124  | 1191* | 1199  | 1248* | 1256  | 1307* |
| AC2    | =%000002 | 1315  | 1385* | 1393  | 1467* | 1475  | 1529* | 1537  | 2066  | 2100* |       |       |       |       |
|        |          | 373*  | 491*  | 492*  | 498   | 518   | 522   | 528   | 564*  | 565*  | 571   | 591   | 595   | 603   |
|        |          | 639*  | 640*  | 644   | 650   | 654   | 660   | 696*  | 697*  | 701   | 707   | 711   | 719   | 755*  |
|        |          | 756*  | 762   | 782   | 786   | 797   | 834*  | 835*  | 841   | 861   | 865   | 880   | 917*  | 918*  |
|        |          | 922   | 928   | 932   | 943   | 979*  | 980*  | 984   | 990   | 994   | 1009  | 1050* | 1051* | 1057  |
|        |          | 1077  | 1081  | 1087  | 1123* | 1124* | 1130  | 1150  | 1154  | 1162  | 1198* | 1199* | 1203  | 1209  |
|        |          | 1213  | 1219  | 1255* | 1256* | 1260  | 1266  | 1270  | 1278  | 1314* | 1315* | 1321  | 1341  | 1345  |
|        |          | 1356  | 1392* | 1393* | 1399  | 1419  | 1423  | 1438  | 1474* | 1475* | 1479  | 1485  | 1489  | 1500  |
| AC3    | =%000003 | 1536* | 1537* | 1541  | 1547  | 1551  | 1566  | 1893  | 2067  | 2099* |       |       |       |       |
|        |          | 374*  | 495*  | 518   | 525   | 558*  | 591   | 600   | 633*  | 650   | 657   | 690*  | 707   | 716   |
|        |          | 749*  | 782   | 789   | 794   | 828*  | 861   | 870   | 877   | 911*  | 928   | 935   | 940   | 973*  |
|        |          | 990   | 999   | 1006  | 1044* | 1077  | 1084  | 1117* | 1150  | 1159  | 1192* | 1209  | 1216  | 1249* |
|        |          | 1266  | 1275  | 1308* | 1341  | 1348  | 1353  | 1386* | 1419  | 1428  | 1435  | 1468* | 1485  | 1492  |
|        |          | 1497  | 1530* | 1547  | 1556  | 1563  | 2068  | 2098* |       |       |       |       |       |       |
| AC4    | =%000004 | 375*  | 2069  | 2097* |       |       |       |       |       |       |       |       |       |       |
| AC5    | =%000005 | 376*  | 2071  | 2095* |       |       |       |       |       |       |       |       |       |       |
| ANS1   | 001022   | 413*  | 498*  | 522*  | 523*  | 524*  | 525   | 528*  | 571*  | 595*  | 596*  | 597*  | 598*  | 599*  |
|        |          | 600   | 603*  | 644*  | 654*  | 655*  | 656*  | 657   | 660*  | 701*  | 711*  | 712*  | 713*  | 714*  |
|        |          | 715*  | 716   | 719*  | 762*  | 786*  | 787*  | 788*  | 789   | 792*  | 793*  | 794   | 797*  | 841*  |
|        |          | 865*  | 866*  | 867*  | 868*  | 869*  | 870   | 873*  | 874*  | 875*  | 876*  | 877   | 880*  | 922*  |
|        |          | 932*  | 933*  | 934*  | 935   | 938*  | 939*  | 940   | 943*  | 984*  | 994*  | 995*  | 996*  | 997*  |
|        |          | 998*  | 999   | 1002* | 1003* | 1004* | 1005* | 1006  | 1009* | 1057* | 1081* | 1082* | 1083* | 1084  |
|        |          | 1087* | 1130* | 1154* | 1155* | 1156* | 1157* | 1158* | 1159  | 1162* | 1203* | 1213* | 1214* | 1215* |
|        |          | 1216  | 1219* | 1260* | 1270* | 1271* | 1272* | 1273* | 1274* | 1275  | 1278* | 1321* | 1345* | 1346* |
|        |          | 1347* | 1348  | 1351* | 1352* | 1353  | 1356* | 1399* | 1423* | 1424* | 1425* | 1426* | 1427* | 1428  |
|        |          | 1431* | 1432* | 1433* | 1434* | 1435  | 1438* | 1479* | 1489* | 1490* | 1491* | 1492  | 1495* | 1496* |
|        |          | 1497  | 1500* | 1541* | 1551* | 1552* | 1553* | 1554* | 1555* | 1556  | 1559* | 1560* | 1561* | 1562* |
|        |          | 1563  | 1566* | 1893* |       |       |       |       |       |       |       |       |       |       |
| ANS2   | 001032   | 418*  | 485   | 558   | 633   | 690   | 749   | 828   | 911   | 973   | 1044  | 1117  | 1192  | 1249  |
|        |          | 1308  | 1386  | 1468  | 1530  | 1635* | 1636* | 1639* | 1640* | 1641* | 1642* |       |       |       |
| BEG    | 001056   | 399   | 430*  |       |       |       |       |       |       |       |       |       |       |       |
| BEGIN  | 001112   | 434   | 439*  |       |       |       |       |       |       |       |       |       |       |       |
| BELL   | = 000207 | 393*  | 1573  | 1888  |       |       |       |       |       |       |       |       |       |       |
| DISPLA | = 177570 | 387*  | 1611* | 1615* |       |       |       |       |       |       |       |       |       |       |
| DONE   | 007002   | 1571* |       |       |       |       |       |       |       |       |       |       |       |       |
| DUMMY  | = 000000 | 388*  | 403   | 404   | 405   | 406   | 408   | 409   | 410   | 411   | 413   | 414   | 415   | 416   |
|        |          | 418   | 419   | 420   | 421   | 1679  | 1680  | 1683  | 1684  | 1687  | 1688  | 1691  | 1692  |       |
| DWNVEC | 012130   | 442*  | 443*  | 2103* | 2104* | 2146* |       |       |       |       |       |       |       |       |
| END1   | 001544   | 500   | 506   | 511   | 516   | 520   | 527   | 531*  |       |       |       |       |       |       |
| END10  | 004106   | 986   | 989   | 992   | 1001  | 1008  | 1012* |       |       |       |       |       |       |       |
| END11  | 004432   | 1059  | 1065  | 1070  | 1075  | 1079  | 1086  | 1090* |       |       |       |       |       |       |
| END12  | 004730   | 1132  | 1138  | 1143  | 1148  | 1152  | 1161  | 1165* |       |       |       |       |       |       |
| END13  | 005152   | 1205  | 1208  | 1211  | 1218  | 1222* |       |       |       |       |       |       |       |       |
| END14  | 005404   | 1262  | 1265  | 1268  | 1277  | 1281* |       |       |       |       |       |       |       |       |
| END15  | 005714   | 1323  | 1329  | 1334  | 1339  | 1343  | 1350  | 1355  | 1359* |       |       |       |       |       |
| END16  | 006244   | 1401  | 1407  | 1412  | 1417  | 1421  | 1430  | 1437  | 1441* |       |       |       |       |       |
| END17  | 006510   | 1481  | 1484  | 1487  | 1494  | 1499  | 1503* |       |       |       |       |       |       |       |
| END2   | 002042   | 573   | 579   | 584   | 589   | 593   | 602   | 606*  |       |       |       |       |       |       |
| END20  | 006774   | 1543  | 1546  | 1549  | 1558  | 1565  | 1569* |       |       |       |       |       |       |       |
| END3   | 002264   | 646   | 649   | 652   | 659   | 663*  |       |       |       |       |       |       |       |       |

|        |          |       |       |       |       |       |       |       |       |       |       |       |       |       |  |  |  |
|--------|----------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|--|--|--|
| END4   | 002516   | 703   | 706   | 709   | 718   | 722*  |       |       |       |       |       |       |       |       |  |  |  |
| END5   | 003026   | 764   | 770   | 775   | 780   | 784   | 791   | 796   | 800*  |       |       |       |       |       |  |  |  |
| END6   | 003356   | 843   | 849   | 854   | 859   | 863   | 872   | 879   | 883*  |       |       |       |       |       |  |  |  |
| END7   | 003622   | 924   | 927   | 930   | 937   | 942   | 945*  |       |       |       |       |       |       |       |  |  |  |
| ERROR  | 011504   | 2050* |       |       |       |       |       |       |       |       |       |       |       |       |  |  |  |
| ERRORS | 012146   | 1990* | 2050* | 2053  | 2151* |       |       |       |       |       |       |       |       |       |  |  |  |
| ERRTI  | 010204   | 1796  | 1799  | 1802* |       |       |       |       |       |       |       |       |       |       |  |  |  |
| ERR1   | 001434   | 494   | 502*  |       |       |       |       |       |       |       |       |       |       |       |  |  |  |
| ERR11  | 004322   | 1053  | 1061* |       |       |       |       |       |       |       |       |       |       |       |  |  |  |
| ERR12  | 004610   | 1126  | 1134* |       |       |       |       |       |       |       |       |       |       |       |  |  |  |
| ERR15  | 005562   | 1317  | 1325* |       |       |       |       |       |       |       |       |       |       |       |  |  |  |
| ERR16  | 006072   | 1395  | 1403* |       |       |       |       |       |       |       |       |       |       |       |  |  |  |
| ERR2   | 001722   | 567   | 575*  |       |       |       |       |       |       |       |       |       |       |       |  |  |  |
| ERR5   | 002674   | 758   | 766*  |       |       |       |       |       |       |       |       |       |       |       |  |  |  |
| ERR6   | 003204   | 837   | 845*  |       |       |       |       |       |       |       |       |       |       |       |  |  |  |
| FEC    | 001044   | 424*  | 471*  | 508   | 544*  | 581   | 619*  | 676*  | 735*  | 772   | 814*  | 851   | 897*  | 959*  |  |  |  |
|        |          | 1030* | 1067  | 1103* | 1140  | 1178* | 1235* | 1294* | 1331  | 1372* | 1409  | 1454* | 1516* | 1792* |  |  |  |
|        |          | 1914  |       |       |       |       |       |       |       |       |       |       |       |       |  |  |  |
| FLTERR | 010150   | 450   | 1791* |       |       |       |       |       |       |       |       |       |       |       |  |  |  |
| FORTAN | 012053   | 1919  | 2134* |       |       |       |       |       |       |       |       |       |       |       |  |  |  |
| FPC    | 001046   | 425*  | 472*  | 513   | 545*  | 586   | 620*  | 677*  | 736*  | 777   | 815*  | 856   | 898*  | 960*  |  |  |  |
|        |          | 1031* | 1072  | 1104* | 1145  | 1179* | 1236* | 1295* | 1336  | 1373* | 1414  | 1455* | 1517* | 1917  |  |  |  |
| FPS    | 001042   | 423*  | 470*  | 495*  | 496   | 503   | 531*  | 543*  | 568*  | 569   | 576   | 606*  | 618*  | 641*  |  |  |  |
|        |          | 642   | 663*  | 675*  | 698*  | 699   | 722*  | 734*  | 759*  | 760   | 767   | 800*  | 813*  | 838*  |  |  |  |
|        |          | 839   | 846   | 883*  | 896*  | 919*  | 920   | 946*  | 958*  | 981*  | 982   | 1012* | 1029* | 1054* |  |  |  |
|        |          | 1055  | 1062  | 1090* | 1102* | 1127* | 1128  | 1135  | 1165* | 1177* | 1200* | 1201  | 1222* | 1234* |  |  |  |
|        |          | 1257* | 1258  | 1281* | 1293* | 1318* | 1319  | 1326  | 1359* | 1371* | 1396* | 1397  | 1404  | 1441* |  |  |  |
|        |          | 1453* | 1476* | 1477  | 1503* | 1515* | 1538* | 1539  | 1569* | 1791* | 1909  |       |       |       |  |  |  |
|        |          | 436   | 2144* |       |       |       |       |       |       |       |       |       |       |       |  |  |  |
| FPTADR | 012122   | 1906  | 2137* |       |       |       |       |       |       |       |       |       |       |       |  |  |  |
| FPUAN  | 012072   | 450*  | 451*  | 2145* |       |       |       |       |       |       |       |       |       |       |  |  |  |
| FPVECT | 012124   | 408*  | 484   | 557   | 632   | 689   | 748   | 827   | 910   | 972   | 1043  | 1116  | 1191  | 1248  |  |  |  |
| HINUM  | 001012   | 1307  | 1385  | 1467  | 1529  | 1629  | 1630  | 1631  | 1632  | 1811  | 1852  |       |       |       |  |  |  |
|        |          | 391*  | 499   | 505   | 510   | 515   | 529   | 572   | 578   | 583   | 588   | 604   | 645   | 661   |  |  |  |
|        |          | 702   | 720   | 763   | 769   | 774   | 779   | 798   | 842   | 848   | 853   | 858   | 881   | 923   |  |  |  |
|        |          | 944   | 985   | 1010  | 1058  | 1064  | 1069  | 1074  | 1088  | 1131  | 1137  | 1142  | 1147  | 1163  |  |  |  |
|        |          | 1204  | 1220  | 1261  | 1279  | 1322  | 1328  | 1333  | 1338  | 1357  | 1400  | 1406  | 1411  | 1416  |  |  |  |
|        |          | 1439  | 1480  | 1501  | 1542  | 1567  | 1795  | 1800  |       |       |       |       |       |       |  |  |  |
| ICNT   | 001000   | 402*  | 452*  | 1596  | 1604  | 1606  | 1608* | 1609* | 1611  | 1614* | 1615  | 1666  |       |       |  |  |  |
| ILLUP  | 011742   | 2061  | 2108* |       |       |       |       |       |       |       |       |       |       |       |  |  |  |
| KIT    | 007222   | 1601  | 1607  | 1614* |       |       |       |       |       |       |       |       |       |       |  |  |  |
| LAD    | 012144   | 453*  | 487*  | 560*  | 635*  | 692*  | 751*  | 830*  | 913*  | 975*  | 1046* | 1119* | 1194* | 1251* |  |  |  |
|        |          | 1310* | 1388* | 1470* | 1532* | 1610* | 1616  | 1618  | 2150* |       |       |       |       |       |  |  |  |
| LONUM  | 001002   | 403*  | 483   | 556   | 631   | 688   | 747   | 826   | 909   | 971   | 1042  | 1115  | 1190  | 1247  |  |  |  |
|        |          | 1306  | 1384  | 1466  | 1528  | 1623  | 1624  | 1625  | 1626  | 1651  | 1810  | 1835  | 1851  | 1876  |  |  |  |
|        |          | 1902  |       |       |       |       |       |       |       |       |       |       |       |       |  |  |  |
| M1120  | 001104   | 431   | 436*  |       |       |       |       |       |       |       |       |       |       |       |  |  |  |
| NHEAD  | 010774   | 1892  | 1922  | 1927  | 1934* |       |       |       |       |       |       |       |       |       |  |  |  |
| NOHEAD | 010714   | 1921* |       |       |       |       |       |       |       |       |       |       |       |       |  |  |  |
| NOP    | = 000240 | 389*  |       |       |       |       |       |       |       |       |       |       |       |       |  |  |  |
| OVER   | 007226   | 1597  | 1615* |       |       |       |       |       |       |       |       |       |       |       |  |  |  |
| PC     | =%000007 | 370*  | 473*  | 546*  | 621*  | 678*  | 737*  | 816*  | 899*  | 961*  | 1032* | 1105* | 1180* | 1237* |  |  |  |
|        |          | 1296* | 1374* | 1456* | 1518* | 2045* | 2058* | 2059* |       |       |       |       |       |       |  |  |  |
| POWDWN | 011546   | 442   | 2061* | 2103  |       |       |       |       |       |       |       |       |       |       |  |  |  |
| POWUP  | 011636   | 2080  | 2083* |       |       |       |       |       |       |       |       |       |       |       |  |  |  |
| PRINTR | 011344   | 1900  | 1910  | 1915  | 1918  | 1925  | 1930  | 1933  | 1949  | 1952  | 1957  | 1960  | 2016* |       |  |  |  |









|          |          |       |       |       |       |       |       |       |       |       |       |      |      |      |
|----------|----------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|------|------|------|
| \$OVER1  | 010020   | 1700  | 1756# |       |       |       |       |       |       |       |       |      |      |      |
| \$OVR1   | 010066   | 1766  | 1769# |       |       |       |       |       |       |       |       |      |      |      |
| \$PLUS   | 012051   | 462   | 2132# |       |       |       |       |       |       |       |       |      |      |      |
| \$POLSH  | 007250   | 474   | 547   | 622   | 679   | 738   | 817   | 900   | 962   | 1033  | 1106  | 1181 | 1238 | 1297 |
| \$POP2X  | 007316   | 1375  | 1457  | 1519  | 1621# | 1705  | 1718  | 1745  | 1756  |       |       |      |      |      |
| \$POP4X  | 007330   | 479   | 627   | 743   | 905   | 1038  | 1186  | 1302  | 1462  | 1635# | 1713  | 1753 |      |      |
| \$P.2A   | 007262   | 552   | 684   | 822   | 967   | 1111  | 1243  | 1380  | 1524  | 1639# | 1726  | 1764 |      |      |
| \$P.2B   | 007304   | 475   | 623   | 739   | 901   | 1034  | 1182  | 1298  | 1458  | 1625# | 1706  | 1746 |      |      |
| \$P.2B   | 007304   | 476   | 624   | 740   | 902   | 1035  | 1183  | 1299  | 1459  | 1631# | 1707  | 1747 |      |      |
| \$P.4A   | 007252   | 548   | 680   | 818   | 963   | 1107  | 1239  | 1376  | 1520  | 1623# | 1719  | 1757 |      |      |
| \$P.4B   | 007274   | 549   | 681   | 819   | 964   | 1108  | 1240  | 1377  | 1521  | 1629# | 1720  | 1758 |      |      |
| \$SBD =  | *****    | 314#  | 1023  | 1025  | 1109  | 1241  | 1378  | 1522  |       |       |       |      |      |      |
| \$SBR =  | *****    | 314#  | 1022  | 1024  | 1036  | 1184  | 1300  | 1460  |       |       |       |      |      |      |
| \$SIGN   | 012045   | 462*  | 1021* | 1904  | 2131# |       |       |       |       |       |       |      |      |      |
| \$TER    | 007460   | 1646  | 1666# |       |       |       |       |       |       |       |       |      |      |      |
| \$TOV    | 007476   | 1647  | 1670# |       |       |       |       |       |       |       |       |      |      |      |
| \$TOV1   | 007510   | 1672  | 1674# |       |       |       |       |       |       |       |       |      |      |      |
| \$TST    | 007352   | 478   | 551   | 626   | 683   | 742   | 821   | 904   | 966   | 1037  | 1110  | 1185 | 1242 | 1301 |
| \$TST1   | 007444   | 1379  | 1461  | 1523  | 1645# | 1712  | 1725  | 1752  | 1763  |       |       |      |      |      |
| \$TST2   | 007456   | 1649  | 1654  | 1658  | 1661# |       |       |       |       |       |       |      |      |      |
| \$TST3   | 007420   | 1662  | 1664# |       |       |       |       |       |       |       |       |      |      |      |
| \$TYPE   | 011006   | 1652  | 1655# |       |       |       |       |       |       |       |       |      |      |      |
| \$SUNDRA | 007704   | 1903  | 1905  | 1907  | 1920  | 1939# |       |       |       |       |       |      |      |      |
| \$SUNDRO | 007616   | 1714  | 1727# |       |       |       |       |       |       |       |       |      |      |      |
| \$SUNDR1 | 007652   | 1694  | 1703# |       |       |       |       |       |       |       |       |      |      |      |
| \$SUNKER | 007614   | 1696  | 1716# |       |       |       |       |       |       |       |       |      |      |      |
| \$200AD  | 010134   | 1701# |       |       |       |       |       |       |       |       |       |      |      |      |
| \$200AO  | 007722   | 1751  | 1762  | 1781# |       |       |       |       |       |       |       |      |      |      |
| \$200A1  | 007736   | 1708  | 1731# |       |       |       |       |       |       |       |       |      |      |      |
| \$200S8  | 007752   | 1721  | 1735# |       |       |       |       |       |       |       |       |      |      |      |
| \$200SO  | 010104   | 1711  | 1724  | 1739# |       |       |       |       |       |       |       |      |      |      |
| \$200S1  | 010120   | 1748  | 1773# |       |       |       |       |       |       |       |       |      |      |      |
| \$201SB  | 007772   | 1759  | 1777# |       |       |       |       |       |       |       |       |      |      |      |
| .        | = 012154 | 1741  | 1743# |       |       |       |       |       |       |       |       |      |      |      |
|          |          | 395#  | 396   | 397#  | 401#  | 487   | 504   | 509   | 560   | 577   | 582   | 635  | 692  | 751  |
|          |          | 768   | 773   | 830   | 847   | 852   | 913   | 975   | 1046  | 1063  | 1068  | 1119 | 1136 | 1141 |
|          |          | 1194  | 1251  | 1310  | 1327  | 1332  | 1388  | 1405  | 1410  | 1470  | 1532  | 1794 | 1935 | 1976 |
|          |          | 2017  | 2032  | 2035  | 2040  | 2047# | 2105  | 2109  | 2117  | 2140# |       |      |      |      |
| .EMT     | 007110   | 448   | 1594# |       |       |       |       |       |       |       |       |      |      |      |
| .ERR     | 010142   | 1709  | 1722  | 1749  | 1760  | 1784# |       |       |       |       |       |      |      |      |
| .ERR1    | 007640   | 463*  | 1022* | 1709# |       |       |       |       |       |       |       |      |      |      |
| .ERR2    | 007674   | 464*  | 1023* | 1722# |       |       |       |       |       |       |       |      |      |      |
| .ERR3    | 010006   | 465*  | 1024* | 1749# |       |       |       |       |       |       |       |      |      |      |
| .ERR4    | 010032   | 466*  | 1025* | 1760# |       |       |       |       |       |       |       |      |      |      |
| .ET      | 010524   | 1887  | 1890# |       |       |       |       |       |       |       |       |      |      |      |
| .IOT     | 011746   | 444   | 2111# |       |       |       |       |       |       |       |       |      |      |      |
| .PR      | 011502   | 2016* | 2018* | 2019* | 2033* | 2034  | 2037* | 2048# |       |       |       |      |      |      |
| .STAT    | 010704   | 1912  | 1919# |       |       |       |       |       |       |       |       |      |      |      |
| .TRP     | 010502   | 446   | 1886# |       |       |       |       |       |       |       |       |      |      |      |
| .TYPE    | 012140   | 1573* | 1574  | 1888* | 1889  | 1995* | 2003* | 2008* | 2011* | 2012  | 2148# |      |      |      |

|        |      |      |      |      |      |      |      |      |      |      |      |      |
|--------|------|------|------|------|------|------|------|------|------|------|------|------|
| DUMP   | 361# | 1899 | 1909 | 1914 | 1917 | 1924 | 1929 | 1932 | 1947 | 1951 | 1956 | 1959 |
| PRINT  | 361# | 2105 |      |      |      |      |      |      |      |      |      |      |
| SDUMP  | 361# |      |      |      |      |      |      |      |      |      |      |      |
| STATUS | 361# |      |      |      |      |      |      |      |      |      |      |      |
| TYPEM  | 361# | 1573 | 1888 |      |      |      |      |      |      |      |      |      |





|    |      |      |      |      |      |      |      |      |      |      |      |      |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |    |
|----|------|------|------|------|------|------|------|------|------|------|------|------|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|----|
| LD | 1591 | 1621 | 1627 | 1633 | 1664 | 1733 | 1737 | 1743 | 1775 | 1779 | 1782 | 1785 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | </ |
|----|------|------|------|------|------|------|------|------|------|------|------|------|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|----|

|        | 2106 | 2129 | 2130 | 2131 | 2134 | 2137 |      |      |      |      |      |      |      |      |  |
|--------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|--|
| .ASCIZ | 2106 |      |      |      |      |      |      |      |      |      |      |      |      |      |  |
| .ASECT | 3130 |      |      |      |      |      |      |      |      |      |      |      |      |      |  |
| .BLKW  | 2047 |      |      |      |      |      |      |      |      |      |      |      |      |      |  |
| .END   | 2154 |      |      |      |      |      |      |      |      |      |      |      |      |      |  |
| .EVEN  | 2106 | 2140 |      |      |      |      |      |      |      |      |      |      |      |      |  |
| .GLOBL | 314  |      |      |      |      |      |      |      |      |      |      |      |      |      |  |
| .LIS   | 311  | 361  | 396  | 400  | 454  | 1571 | 1620 | 1801 | 1885 | 1938 | 2015 | 2060 | 2106 | 2110 |  |
| .LIS   | 361  |      |      |      |      |      |      |      |      |      |      |      |      |      |  |
| .LIS   | 361  |      |      |      |      |      |      |      |      |      |      |      |      |      |  |
| .LIS   | 311  | 361  | 396  | 400  | 454  | 1571 | 1620 | 1801 | 1885 | 1938 | 2015 | 2060 | 2106 | 2110 |  |
| .LIS   | 2    | 396  |      |      |      |      |      |      |      |      |      |      |      |      |  |
| .LIS   | 311  | 361  | 400  | 454  | 1571 | 1620 | 1801 | 1885 | 1938 | 2015 | 2060 | 2110 |      |      |  |
| .LIS   | 312  |      |      |      |      |      |      |      |      |      |      |      |      |      |  |

.ABS. 012154 000  
000000 001

ERRORS DETECTED: 0  
DEFAULT GLOBALS GENERATED: 0

\*.DCFPSD.SEG/SOL/CRF/PAGNUM=DCFPSD  
RUN-TIME: 8 16 3 SECONDS  
RUN-TIME RATIO: 539/28=19.9  
CORE USED: 9K (17 PAGES)



Spooler runtime 6 Seconds, 36 KCS, 209 disk reads, 3 disk writes, 54 pages

100-443887-1000

[illegible]