PDP-8
FLOATING-POINT SYSTEM
PROGRAMMER'S REFERENCE MANUAL

DIGITAL   EQUIPMENT   CORPORATION • MAYNARD, MASSACHUSETTS

PREFACE

This manual supplies a detailed description of floating-point software for the PDP-8 family of computers.

The software consists of individual routines which perform basic arithmetic and trigonometric functions on data stored in floating-point format and facilitate data readin, decimal-binary conversion, conversion to floating-point format, and writeout. Four programs obtainable from Digital Equipment Corporation provide various combinations of these functions; listings, entry points, and order numbers for the programs are supplied in this manual.

Automatic calling of user subroutines can be incorporated in the existing routines with minor coding modifications.

CONTENTS

CONTENTS (continued)

CHAPTER 1

INTRODUCTION

The Floating-Point System enables the programmer to concentrate on the logic of his computation rather than on decimal points. The system maintains a constant number of significant digits throughout the computation, thereby enhancing the accuracy of the result.

Floating-point notation is particularly useful for computations involving numerous multiplications and divisions where magnitudes are likely to vary widely and where only crude predictions can be made as to the amount of variation involved. The main advantage of the Floating-Point System is derived from the ability to store very large or very small numbers by storing only the significant digits together with the exponent for that number. In an integral or fractional fixed-point machine, the programmer must either include all of the 0's between the significant digits and the decimal point (and in so doing loose considerable accuracy), or account for the point in each step of his programming.

The Floating-Point System is constructed as a self-contained package which includes its own input, arithmetic, and output routines. It allows the programmer to use floating-point arithmetic without having to construct his own arithmetic subroutines. For example, the polynomial:

$$Y = AX^3 + BX^2 + CX + D$$

may be programmed as follows (the variables X and Y, and the parameters A, B, C, and D are stored in registers of the same name):

```
......
JMS   I  7      /ENTER FLOATING PACKAGE
FGET  A         /LOAD PSEUDO AC
FMPY  X         /MULTIPLY
FADD  B         /ADD
FMPY  X         /MULTIPLY
FADD  C         /ADD
FMPY  X         /MULTIPLY
FADD  D         /ADD
FPUT  Y         /STORE
FEXT            /EXIT FLOATING PACKAGE
......
```

CHAPTER 2

FLOATING-POINT REPRESENTATION

A floating-point number consists of a mantissa and an exponent. For example, the decimal number 12 may be represented as:

$$1200.0 \cdot 10^{-2}$$
$$120.00 \cdot 10^{-1}$$
$$12.000 \cdot 10^{0}$$
$$1.2000 \cdot 10^{1}$$
$$.12000 \cdot 10^{2}$$
. . . .
etc.

where the exponents are the numbers -2, -1, 0, etc.

Since the PDP-8 is a binary machine, floating-point numbers are stored as floating-point binary internally. For example, the binary number 110 (6 decimal) may be represented as:

. . . .
$$11000.0 \cdot 2^{-2} \qquad (24 \cdot 1/4 = 6)$$
$$1100.00 \cdot 2^{-1} \qquad (12 \cdot 1/2 = 6)$$
$$110.000 \cdot 2^{0} \qquad (6 \cdot 1 = 6)$$
$$11.0000 \cdot 2^{1} \qquad (3 \cdot 2 = 6)$$
$$1.10000 \cdot 2^{2} \qquad (3/2 \cdot 4 = 6)$$
$$.110000 \cdot 2^{3} \qquad (3/4 \cdot 8 = 6)$$
. . . .
etc.

Notice that the binary exponent is always a signed integer. The PDP-8 Floating-Point System uses the convention:

$$1/2 \leq |\text{MANTISSA}| < 1.$$

When this is true, the word is said to be normalized.

The value of the number is then:

$$\text{MANTISSA} \cdot 2^{\text{EXPONENT}}$$

where the MANTISSA is a signed quantity. The result of this convention is that more significant bits are retained. For example, the number .10 (decimal) is equal to:

$|0.00\ 011\ 001\ 100\ |\ 110\ 011\ 001\ 100\ |\ 110\ 011\ 001\ 100\ 110\ ....$

in binary.

When this word is stored in two 12-bit words (24 bits), the leading 0's are nonsignificant and only 20 bits of significance are maintained. If the number is rewritten as:

$2^{-3}.|0.11\ 001\ 100\ 110\ |\ 011\ 001\ 100\ 110\ |\ 011\ 001\ 100\ 110\ ....$

and the mantissa stored in two 12-bit words, 23 bits of significance are maintained.

The exponent is stored in a third word making a total of three words for storage.

sign of exponent   binary exponent 2's complement
signed quantity

sign of mantissa

high order mantissa word

low order mantissa word
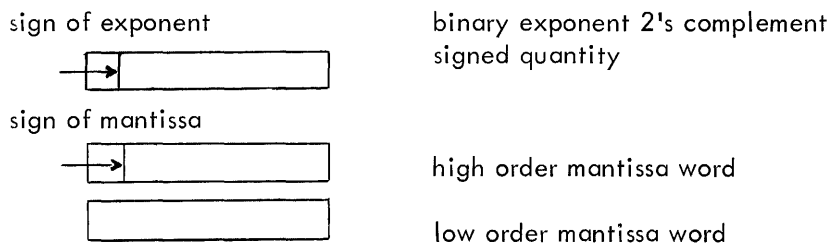
The number .1 (decimal) would be stored as:

$111\ \ 111\ \ 111\ \ 101\ |\ 011001100110\ |\ 011001100110$

or written in octal:

7775 3146 3146

The address of this number would be considered 2146 if it were stored as follows:

2146/7775
2147/3146
2150/3146

The number $-.1$ (decimal) would be $-.063146314$ (octal) which is $0.000\ 110\ 011\ 001\ 100\ 110\ 011\ 011\ 100$ (binary).

Normalized, this becomes: $(2^{-3})_8 \cdot (0.11\ 001\ 100\ 110\ 011\ 001\ 100\ 110)_2$

Complemented and incremented to become the negative 2's complement
this becomes: $(2^{-3})_8 \cdot (1.00\ 110\ 011\ 001\ 100\ 110\ 011\ 010)_2$
which is: $(2^{-3})_8 \cdot (4\ \ \ \ \ 6\ \ \ \ 3\ \ \ \ 1\ \ \ \ 4\ \ \ \ 6\ \ \ \ 3\ \ \ \ 2\ \ \ )_8$

This number would be stored (within octal) as:

7775	exponent of -3
4631	high order mantissa word
4632	low order mantissa word

Since the mantissa is greater (in magnitude) than 1/2 and less than 1, its binary point is considered to be between bit 0 and bit 1 of the high order mantissa word.

Further examples:

12 (decimal)

$1100. = (2^4) (0.11000000000000000000000)$

would be stored as (written in octal):

0004

3000

0000

Octal	Decimal	Floating Binary		
-1	-1	0001	6000	0000
-10	-8	0004	6000	0000
.4	.5	0000	2000	0000
3	3	0002	3000	0000
14	12	0004	3000	0000
.2	.25	7777	2000	0000
14.26074475	12.3456	0004	3054	1711
-14.26074475	12.3456	0004	4723	6067

Convert the decimal number 1.6 to floating-binary form for storage.

An easy method to convert a decimal fraction to an octal fraction is illustrated below:

$$1.6_{10} = 1.46314631_8$$

$$= 1.100110011001100110011001$$

$$= (2^1) (0.11001100110011001100110)$$

$$= 0001$$

$\times\,8$

1) .6　　　　3146

8

4) .8　　　　3146

8

6) .4

8

3) .2

8

1) .6

ARITHMETIC

Since floating-point numbers are stored in a three-register format, the floating-point system uses a "psuedo" floating accumulator (FAC) which consists of three registers in the floating-point package: 44, 45, and 46. Register 44 contains the exponent; 45 and 46 contain the high and low order parts of the mantissa, respectively.

BASIC FLOATING-POINT COMMANDS

The basic floating-point commands include the following:

    load floating accumulator

    store floating accumulator

    add to floating accumulator

    subtract from floating accumulator

    multiply by floating accumulator

    divide into floating accumulator

    normalize floating accumulator

All arithmetic operations are called through an interpreter. The command codes have a format that is almost identical to the format of the memory reference instruction, namely:

| OP | I | Z | ADDRESS |
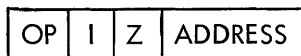
where the op code is from 000 to 111 or $0_8 - 7_8$.

INTERPRETER

The interpreter contains, at all times, the address of the memory location containing the next pseudo instruction to be executed. This is initially stored when the program enters the interpreter using a JMS

instruction. When the interpreter encounters an instruction with an op code of 0 and with bits 8-11 of the pseudo instruction equal to 0, it exits to the next memory location.

FLOATING-POINT INSTRUCTIONS

The floating-point instructions are:

Op Code	Mnemonic	Effect
1	FADD	Floating Addition Add the contents of the effective address to the floating accumulator.
2	FSUB	Floating Subtract Subtract the contents of the effective address from the floating accumulator.
3	FMPY	Floating Multiply Multiply the floating accumulator by the contents of the effective address.
4	FDIV	Floating Divide Divide the floating accumulator by the contents of the effecitve address.
5	FGET	Floating Get Load the floating accumulator with the contents of the effective address.
6	FPUT	Floating Put Store the contents of the floating accumulator at the locations specified by the effective address. The contents of the floating accumulator are unchanged.
7	FNOR	Floating Normalize Normalize the contents of the floating accumulator.
0	is decoded as follows:	
Bits 8-11 = 0000		Floating Exit Return control to following instruction
= 0001		Floating Square Square the contents of the floating accumulator

2-5

= 0010   Floating Square Root
Take the root of the absolute value of the floating accumulator

= 0011-1111   Expandable commands

This may be summarized:

FADD   Y; 1000; $C(FAC) + C(Y) \rightarrow C(FAC)$

FSUB   Y; 2000; $C(FAC) - C(Y) \rightarrow C(FAC)$

FMPY   Y; 3000; $C(FAC) \times C(Y) \rightarrow C(FAC)$

FDIV   Y; 4000; $C(FAC) \div C(Y) \rightarrow C(FAC)$

FGET   Y; 5000; $C(Y) \rightarrow C(FAC)$

FPUT   Y; 6000; $C(FAC) \rightarrow C(Y)$

FNOR   ; 7000; $C(FAC)$ normalized $\rightarrow C(FAC)$

FEXT   ; 0000; exit from interpreter to instruction following this command

SQUARE ; 0001; $C(FAC)^2 \rightarrow C(FAC)$

SQROOT; 0002; $C(FAC)^{1/2} \rightarrow C(FAC)$

FADD–FDIV: Result is normalized

FADD–FPUT: C(Y) unchanged

The assembler recognizes all of these mnemonics except for SQUARE and SQROOT which may be defined as:

SQUARE=0001

SQROOT=0002

The floating-point interpreter is entered with an effective JMS 5600.

For example:

```
SQROOT=0002
*7
 5600
*200
        JMS I 7    200/    4407
        FGET A     201/    5206
        SQROOT     202/    0002
        FPUT I B   203/    6611
        FEXT       204/    0000
        HLT        205/    7402
A,      0003       206/    0003
        2000       207/    2000
        0000       210/    0000
B,      300        211/    0300
$
```

When this program is started at 0200, it will halt at location 205. The state of the machine will be:

44/	0002	
45/	2000	floating accumulator contains
46/	0000	2.0, i.e., $(4.0)^{1/2}$
206/	0003	
207/	2000	register A contains 4.0
210/	0000	
300/	0002	
301/	2000	answer stored here
302/	0000	

CHAPTER 3

FLOATING-POINT INPUT/OUTPUT

FLOATING-POINT INPUT

The basic floating-point package contains an input routine to read characters from the 33 ASR keyboard. Numbers are read in in decimal and converted to the internal floating-point binary format. Input format is floating decimal. The number 726.7 may be read in in any of the following forms:

```
726.7
.7267E3
.7267E+03
+7267E-1
etc.
```

Input is terminated when a character is typed that is not a part of this format. The conversion of "12.0." would be terminated on the second "." and the binary number:

000000000100	(octal 0004)
011000000000	(octal 3000)
000000000000	(octal 0000)

would be in the floating-point accumulator upon completion of the conversion.

Flags

There are several flags associated with the input routine that are useful to the programmer.

0056   This register is a switch that has the following meaning:

If C(56) = 0, do not type a line-feed after a carriage-return was read.
If C(56) = 7777, type a line-feed when a carriage-return is inputted.
This switch is initially set to 7777.

0057   This register contains the character that terminated the input conversion.

0060   This register contains 0000 if no input conversion was made; i.e., a space or other terminator was initially typed.

The input routine is entered with an effective JMS 7400. It returns control to the instruction following the calling JMS upon receipt of a terminator. The floating accumulator contains the input number in normalized floating-binary. Register 0057 contains the terminating character in ASCII, and C(0060) indicates whether or not there was a valid input.

For example:

```
        *5
         7400
        *7
         5600
        *200
         JMS I 5        /INPUT ROUTINE
         JMS I 7        /CALL FLOATING POINT
         FPUT A
         FEXT
         JMS I 5
         JMS I 7
         FPUT B
         FEXT
         HLT
        A,  0
            0
            0
        B,  0
            0
            0
        $
```

When this program is started at 0200 and the following is typed:

    X2.0Y

The program will halt at location 0210, and A and B will contain

```
        A,  0
            0
            0          and C(57) = 0331, the second TERMINATOR
        B,  0002
            2000
            0000
```

The first input was considered a 0 because a terminator "X" was inputted.

This program could be written to ignore the non-numeric information as follows:

```
*5
    7400
*7
    5600
*200
        JMS I 5        /CALL INPUT ROUTINE
        TAD 60         /ANY VALID INPUT?
        SNA  CLA
        JMP .-3        /NO - IGNORE
        JMS I 7        /YES
        FPUT A         /STORE IT
        FEXT
        JMS I 5        /GET NEXT
        TAD 60
        SNA CLA        /VALID?
        JMP .-3        /NO - IGNORE
        JMS I 7        /YES
        FPUT B         /STORE IT
        FEXT
        HLT       /HALT
A,  0
    0
    0
B,  0
    0
    0
$
```

Register 57 may be used for integrating control characters into the input.

<div align="center">

Rubout
</div>

There is one special input character, the rubout. If it is struck before an input delimiter, the input routine is restarted and previous numbers deleted.

For example, assume the input routine has been entered.

276 Rubout 1T

The input routine would exit with 1 (decimal) in the floating accumulator or:

```
44/        0001
           2000
           0000
```

FLOATING-POINT OUTPUT

The basic floating-point package contains an output routine. Upon entry the contents of the floating accumulator are converted to floating-point decimal and typed out on the ASR 33 in the following format:

$$\pm 0.XXXXXXXE\pm XX$$

For example, if the floating accumulator contained:

```
44/        0002
           2000
           0000
```

and the output routine were entered,

+0.2000000E+01

would be typed.

Entry

The output routine is entered by an effective JMS 7200. After outputting the contents of the floating accumulator, it returns control to the instruction following the calling JMS instruction. The contents of the floating accumulator are lost. The floating-output routine has a flag or program switch in location 0055 on page 0.

If C(0055) does not equal 0, type a carriage-return/line-feed following each output. This location initially contains 7777.

Program Example

```
/SOLVE THE QUADRATIC EQUATION
/AX²+BX+C=0 FOR VALUES
/OF A, B, C.          TYPE OUT ROOTS
SQROOT=0002
SQUARE=0001
*5        7400        /POINTER TO INPUT ROUTINE
          7200        /POINTER TO OUTPUT ROUTINE
          5600        /POINTER TO INTERPRETER
*200
          KCC
          TLS
BEGIN,    JMS I 5      /INPUT A
          JMS I 7      /ENTER INTERPRETER
```

```
            FMPY TWO      /2.0
            FPUT A        /STORE 2.0·A
            FEXT
            JMS I 5       /INPUT B
            JMS I 7       /ENTER INTERPRETER
            FPUT B        /STORE B
            FEXT
            JMS I 5       /INPUT C
            JMS I 7       /ENTER INTERPRETER
            FMPY TWO      /MULTIPLY BY 2.
            FMPY A        /MULTIPLY BY 2A
            FPUT TEMP     /4AC
            FGET B
            SQUARE        /B SQUARED
            FSUB TEMP     /B SQUARED -4AC
            SQROOT        /TAKE SQUARE ROOT
            FPUT TEMP     /STORE IT
            FGET B
            FMPY MIN1     /-1·B = — B
            FPUT B        /STORE
            FSUB TEMP     /-B—SQUARE ROOT
            FDIV A
            FEXT          /ANSWER IN FAC
            JMS I 6       /OUTPUT IT
            JMS I 7       /ENTER INTERPRETER
            FGET B        /-B
            FADD TEMP     /-B + SQUARE ROOT
            FDIV A
            FEXT          /ANSWER IN FAC
            JMS I 6       /OUTPUT IT
            JMP BEGIN     /CONTINUE
TWO,        0002          /CONSTANT 2.0
            2000
            0000
MIN1,       0001          /CONSTANT -1.0
            6000
            0000
TEMP,       0
            0
            0
A,          0
            0
            0
B,          0
            0
            0
C,          0
            0
            0
$
```

Both the input-conversion routine and the output-conversion routine use a typeout subroutine that contains the instructions:

```
...
TSF
JMP .-1
TLS
CLA
...
```

This means that the teleprinter flag must be set when these routines are entered or it must be in the process of typing. A TLS instruction can be put into the initialization section of a program.

THE BASIC PACKAGE

Subroutines

When the floating-point interpreter encounters a 0 op code, it further decodes bits 8-11. If these bits equal 0, the interpreter exits. If the bits are nonzero, they are used in a table look-up to specify the address of a subroutine. The called subroutine may use the floating-point interpreter, the input, or the output, but it may not use bits 8-11 to call another subroutine. The look-up table is on page 16 of the interpreter listing. For example, if the user wished to define a routine that negated the contents of the floating accumulator, the following steps would be taken:

On page 6 of the interpreter listing, there is a routine to negate the floating accumulator. It is a subroutine and has, as entry point, the address 6000. If the negate subroutine were to be called by 0010 in the interpreter mode, the word 6000 would be placed in address 6554 of the calling table. Input and output could be called in the interpreter mode if 7400 were placed in 6555 (0011) and 7200 were placed in 6556 (0012).

```
NEGATE = 0010
INPUT  = 0011
OUTPUT = 0012
SQUARE = 0001
*7
 5600
 *200
  KCC
  TLS
  JMS I 7    /ENTER INTERPRETER
  INPUT      /CALL INPUT ROUTINE
  SQUARE     /SQUARE IT
  NEGATE     /CALL OUTPUT ROUTINE
  OUTPUT     /CALL OUTPUT ROUTINE
  FEXT       /EXIT
  HLT        /HALT
$
```

DESCRIPTION OF BASIC FUNCTION

Addition

Floating-point addition is carried out by first aligning the binary points of the two numbers. This is accomplished by scaling the smaller number to the right. Then the mantissas are both scaled right once so that overflow will not occur into the sign bit. A 2's complement addition of the mantissas is then made. The result is normalized and control returns to the interpreter. This may be represented as:

Can operands be aligned?

Yes   No

Align Exponents
Add Mantissas   Put larger number in FAC
Normalize
Return to Interpreter
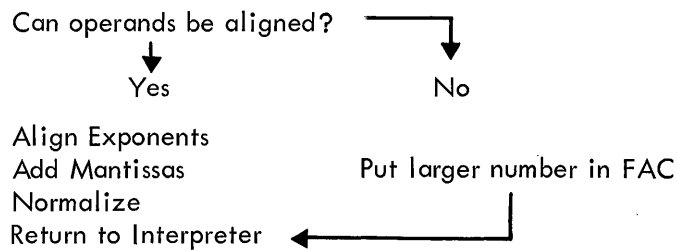
Subtraction

Floating-point subtraction is accomplished by negating the operand, and then calling the addition subprogram.

Negate Operand
Call Addition

Multiplication

Floating-point multiplication is accomplished by adding the exponents together and then performing a double-precision multiplication. The result is normalized and control returns to the interpreter.

Add Exponents
Multiply Mantissas carrying result to 35 bits
Normalize
Return to Interpreter

Division

Floating-point division is accomplished by subtracting the exponent of the divisor from the exponent of the dividend. The mantissa is divided and the result is normalized. Control returns to the interpreter.

3-7

```
Is divisor = 0? ─────────────┐
      ↓                       ↓
     No                      Yes

Subtract Exponents
Divide Mantissas         +1 to error flag
Normalize                Set result to highest possible
Return to interpreter ◄──────┘
```
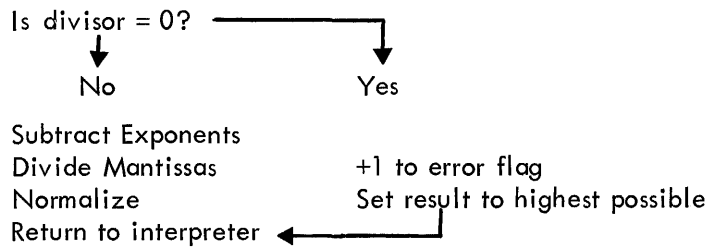
Square

The square routine calls the multiplication routine internally.

Square Root

The square root is calculated using Newton's method in which an initial approximation is made and then each succeeding approximation is calculated. The routine exits when two successive approximations are equal to within the least significant bit of the mantissas.

Form first approximation, X_1, of $\sqrt{N}$:

then:

$$X_i + 1 = 1/2 \left(X_i + \frac{N}{X_i} \right)$$

is $\left| X_i - X_{i+1} \right| <$ least significant bit?

```
       No        Yes
```

Return to interpreter
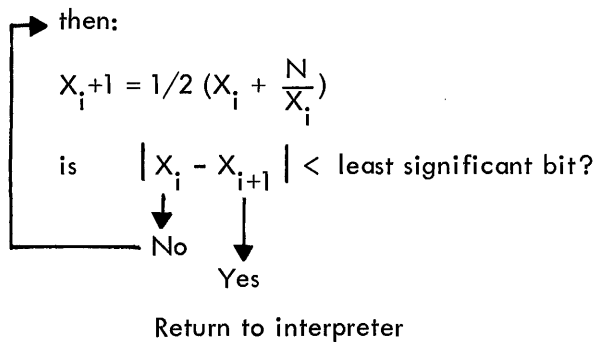
Error Flag

Division by 0 causes $C(61)$ to be incremented by 1. The quotient is set to the highest positive number.

Attempting to extract the square root of a negative number causes $C(61)$ to be incremented by 1. The root of the absolute value is taken.

The contents of 61 are set to 0 at the beginning of each divide and square root operation.

SUMMARY OF BASIC PACKAGE

Entry Points

5600	Arithmetic Interpreter
7400	Floating Input
7200	Floating Output

NOTE: Both the input and the output routines require that register 7 contains the interpreter entry point: 5600.

Flags

55	If $\neq 0$, type carriage-return/line-feed after output.
56	If $\neq 0$, follow each input carriage-return by a line-feed.
57	Contains the input terminating character.
60	Equals 0 if no valid input.
61	Is nonzero if (a) divide by 0 or (b) square root of a negative number.

Commands

FADD	1000	Floating Add
FSUB	2000	Floating Subtract
FMPY	3000	Floating Multiply
FDIV	4000	Floating Divide
FGET	5000	Floating Get
FPUT	6000	Floating Put
FNOR	7000	Floating Normalize
FEXT	0000	Floating Exit
SQUARE	0001	Square
SQROOT	0002	Square Root
	0003	
	....	Expandable
	0017	

Storage

The floating accumulator is in registers:

> 44 - Exponent
>
> 45 - High Order Mantissa
>
> 46 - Low Order Mantissa

The basic floating-point package (DEC-08-YQ1A-PB) uses memory locations:

> 5-7; 40-61; 5600-7577

FIXED TO FLOATING/FLOATING TO FIXED

Since the floating-point package stores numbers in the following format:

Signed Exponent   2's Complement Signed Mantisa

Implied Binary Point
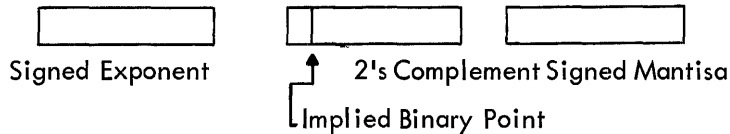
and since the exponent indicates where the real exponent is, this information may be used to either convert a floating-point number to fixed point or to convert a fixed-point number to floating point.

For example, assume that there is an integer in the accumulator that is less in magnitude than 2047. To float this number, the following sequence of steps may be employed:

```
        . . .
        DCA 45      /PUT INTO HIGH-ORDER MANTISSA
        DCA 46      /PUT 0 INTO LOW-ORDER MANTISSA
        TAD C13     /11 (10) INTO
C,      DCA 44      /EXPONENT
        JMS I 7     /CALL INTERPRETER
        FNOR        /NORMALIZE
        FEXT        /LEAVE INTERPRETER
        . . .
C13,    0013        /11 (DECIMAL)
```

At point C, we have set the binary point of the integer to the right end of the high order mantissa word, or eleven (decimal) locations to the right of the implicit binary point.

To float this number, the floating accumulator is scaled right until the exponent contains the location of the desired binary point. To fix the floating accumulator as an 11-bit signed integer, the following sequence of coding may be employed:


```
          ....
          CLA
          TAD 44        /FETCH EXPONENT
          SZA SMA       /IS THE NUMBER <1?
          JMP .+3       /NO:
          CLA           /YES:  FIX IT TO 0
          JMP DONE+1
          TAD M13       /NO:  SET BINARY POINT AT
          SNA           /11 (10) PLACES TO RIGHT OF CURRENT POINT
          JMP DONE      /IT IS ALREADY THERE:  ALL DONE
          SMA           /TEST TO SEE IF IT IS TOO LARGE
          JMP ERROR     /YES:  NUMBER >2**11
          DCA 44        /NO:  SET SCALE COUNT
GO,       CLL           /0 TO C(L)
          TAD 45        /FETCH MANTISSA
          SPA           /IS IT <0?
          CML           /YES:  PUT A 1 IN LEFT BIT
          RAR           /SCALE RIGHT
          DCA 45        /RESTORE IT
          ISZ 44        /TEST IF SHIFTED ENOUGH
          JMP 00        /NO:  CONTINUE
DONE,     TAD 45        /ANSWER IN C(AC)
          ....
          ....
M13,      -13           /-11 (DECIMAL)
```

This may be coded as a subroutine.

EXTENDED FLOATING-POINT PACKAGE

The extended floating-point package contains the following additional commands:

0003   Floating Sine
Take the sine of the angle in the floating accumulator (considered to be in radians) and leave the result in the floating accumulator.

0004   Floating Cosine
Take the cosine of the angle in the floating accumulator (considered to be in radians) and leave the result in the floating accumulator.

0005   Floating Arc Tangent
Take the arc tangent of the number in the floating accumulator and leave the result (in radians) in the floating accumulator.

0006   Floating Exponent
Raise e to the C(FAC) power. Leave the result in the floating accumulator.

0007   Floating Logarithm
Take the natural logarithm of the absolute value of the number in the floating accumulator.

3-11

The sine routine uses the following identities:

$$SIN(-X)=-SIN(X)$$
$$SIN(X)=SIN(2\pi N+F)=SIN(F)$$

where $0<F<2\pi$ and N is an integer:

$$SIN(\pi-F)=-SIN(F)$$

Using these identities, F is reduced to the range $-\pi/2<F<\pi/2$.

If $Y = 2F/\pi$ so that $-1<F<1$, then:

$$SIN(X)=(C_1 F+C_3 F^3+C_5 F^5+C_7 F^7+C_9 F^9)$$

where:

$$C_1 = 1.57079631847$$

$$C_3 = -0.64596371106$$

$$C_5 = +0.07968967928$$

$$C_7 = -0.00467376557$$

$$C_9 = +0.00015148419$$

The sine subroutine is valid over the range:

$$-2\pi.2047<X<2\pi.2047$$

Cosine

The cosine routine uses the identity:

$$COS(X)=SIN(\pi/2-X)$$

It then uses the SINE routine.

Arc Tangent

The arc tangent routine uses the following identities:

$$ARCTAN(-X)=-ARCTAN(X)$$
IF $X>, 1$; then $ARCTAN(X)=\pi/2-ARCTAN(1/X)$
FOR $0<X<1$,

$$ARCTAN(X) = \frac{X(A_0 + A_1 X^2 + A_2 X^4)}{B_0 + B_1, X^2 + B_2 X^4}$$

where:

$A_0 = +0.6402481953$

$A_1 = +0.4229908144$

$A_2 = +0.0264694361$

$B_0 = +0.6402487022$

$B_1 = +0.6363779373$

$B_2 = +0.1108328778$

The result is in the range:

$-\pi/2 < ARCTAN(X) < \pi/2$

$-\infty < X < \infty$

Logarithm

The logarithm routine uses the identities:

For $X<1$;

$LOG(X) = LOG(2^N \cdot F) \ (1 < F < 2)$

$= NLOG(2) + LOG(F)$

$LOG(1+Y) = \sum\limits_{C=1}^{8} A_i Y^i$

where:

$A_1 = +0.9999964239$

$A_2 = -0.4998741238$

$A_3 = +0.3317990258$

$A_4 = +0.2407338084$

$A_5 = +0.1676540711$

$A_6 = -0.0953293897$

$$A_7 = +0.0360884937$$

$$A_8 = -0.0064553442$$

for:

$0 < X < 1$, the identity:

$LOG(X) = -LOG(1/X)$ is used.

Floating Exponent

The exponent routine uses the identity for $X > 0$,

$$e^X = 2^{X \cdot \log_2 e} = 2^{N+F} = 2^N \cdot 2^F$$

where N is an integer and $< F < 1$

$$2^F = 1 + \cfrac{2F}{A - F + BF^2 - \cfrac{C}{D + F^2}}$$

where:

$A = +9.95459578$

$B = +0.03465735903$

$C = +617.97226053$

$D = +87.417497202$

$\log_2 e = 1.4426950409$

If $X < 0$

$$e^x = \frac{1}{e^{-x}}$$

Example, input A, B, and output:

$$Y = LOG(COS(A/B) + \sqrt{A \cdot B})$$

```
SQROOT=0002              /DEFINITIONS TO ASSEMBLER
COS=0004
LOG=0007

*5
    7400
    7200
    5600
/USES EXTENDED INTERPRETER
```

3-14

```
*200
KCC
TLS
BEGIN,    JMS I 5         /INPUT A
          JMS I 7         /ENTER INTERPRETER
          FPUT A          /STORE A
          FEXT            /EXIT
          JMS I 5         /INPUT B
          JMS I 7         /ENTER INTERPRETER
          FPUT B          /STORE B
          FMPY A          /A·B
          SQROOT          /EXTRACT ROOT
          FPUT TEMP       /STORE IT
          FGET A          /LOAD FAC WITH A
          FDIV B          /DIVIDE BY B
          COS             /TAKE COSINE
          FADD TEMP       /ADD
          LOG             /TAKE LOG.
          FEXT
          JMS I 6         /OUTPUT ANSWER
          JMP BEGIN
A,        0
          0
          0
B,        0
          0
          0
TEMP,     0
          0
          0
$
```

Output Controller

There is an additional routine available that formats floating-point output. It is available with both the basic floating-point package and the extended package.

The controller routine requires two parameters for output formatting. It is called by an effective JMS 7200. At this point:

C(62) = total number of digits to be outputted. If C(62)=0, output in E format.

C(AC) = number of digits to the right of the decimal point. If it equals 0, do not type a ".".

If the number in the floating accumulator is larger than the field width allows, "X's" will be typed. The sign is typed and leading 0's are suppressed.

Example: If the contents of the floating accumulator are 678.234 (decimal).

C(62)	C(AC)	Output
3	0	+ 678
4	0	+ 678
4	1	+ 678.2
5	2	+ 678.23
6	2	+ 678.23
3	2	+ 678.
2	0	+ XX
0	1	+ 0.678234E+03

FLOATING POINT PACKAGE VERSIONS

Four versions of the floating-point package are available:

DEC-08-YQ1A-PB

This is the basic floating-point package. It consists of the input/output package and the basic arithmetic instructions. Its core limits are:

    5-7; 40-61; 5600-7577

DEC-08-YQ2A-PB

This is the basic floating-point package with the output controller. Its limits are:

    7; 40-62; 5400-7577

The output controller does not type a carriage-return/line-feed after output and, hence, the programmer must provide his own routine to do so. Because of the way in which the typeout routine in the floating-point packages are constructed, the user must construct similar typeout routines in order to avoid timing errors in the teleprinter. In other words, the floating-point package uses a routine similar to:

    TSF
    JMP .-1
    TLS

to do its typing. The user must use a similar routine, i.e., one that waits for the teleprinter flag to be set before executing the TLS instruction.

DEC-08-YQ3A-PB

This is the basic floating-point package with the extended functions. In addition, two interpretive commands are provided for input and output. The additional interpretive commands are:

SINE	3
COSINE	4
ARCTANGENT	5
EXPONENTIAL	6
LOG	7
INPUT	13
OUTPUT	14

Its core limits are:

7; 40-61; 4757-7577

Using this package, the routine on page 3-14 and 3-15 for calculating:

$Y=LOG(COS(A/B)+ \sqrt{A \cdot B})$ may be rewritten as:

```
SQROOT=2
COS=4
LOG=7
INPUT=13
OUTPUT=14

*7
 5600

*200
KCC
TLS
BEGIN,    JMS I 7     /CALL INTERPRETER
          INPUT       /READ A
          FPUT A      /STORE IT
          INPUT       /READ B
          FPUT B
          FMPY A      /A·B
          SQROOT
          FPUT TEMP  /(A·B)** .5
          FGET A
          FDIV B
          FCOS
          FADD TEMP
          FLOG
          OUTPUT
          FEXT
          JMP BEGIN
A,        0
          0
          0
B,        0
          0
          0
TEMP,     0
          0
          0
$
```

DEC-08-YQ4A-PB

This is the basic package, the output controller, and the extended functions. This package does not have the interpretive I/O commands of input and output. Its core limits are:

7; 40-62; 4557-7577

PROGRAM LISTINGS

```
                    /FLOATING POINT ARITHMETIC INTERPRETER
                    *40
0040    0000    EX1,        0                   /OPERAND STORAGE
0041    0000    AC1H,       0
0042    0000    AC1L,       0
0043    0000    OVER1,      0
0044    0000    EXP,        0                   /F.A.
0045    0000    HORD,       0
0046    0000    LORD,       0
0047    0000    OVER2,      0
0050    0000    EXP1,       0
0051    0000    QUOL,       0
0052    0000    FPAC1,      0
0053    0000                0
0054    0000                0
                    *61
0061    0000    FLAG,       0                   /ARITHMETIC ERROR FLAG
                    *5600
5600    0000    FPNT,       0
5601    7300                CLA CLL
5602    3043                DCA OVER1
5603    3047                DCA OVER2
5604    1600                TAD I FPNT          /GET NEXT INSTRUCTION
5605    3253                DCA JUMP
5606    1253                TAD JUMP
5607    0263                AND PAGENO          /GET PAGE BIT
5610    7650                SNA CLA             /PAGE ZERO?
5611    5214                JMP .+3             /YES
5612    1261                TAD MASK5           /NO
5613    0200                AND FPNT            /C(FPNT)0-4 CONTAINS PAGE BITS
5614    3256                DCA ADDR
5615    1262                TAD MASK7           /GET 7 BIT ADDRESS
5616    0253                AND JUMP
5617    1256                TAD ADDR
5620    3256                DCA ADDR
5621    1264                TAD INDRCT          /INDIRECT BIT=1?
5622    0253                AND JUMP
5623    7650                SNA CLA
5624    5227                JMP LOOP01          /NO-GO ON
5625    1656                TAD I ADDR          /YES DEFER
5626    3256                DCA ADDR
5627    2200    LOOP01,     ISZ FPNT
5630    1656                TAD I ADDR
5631    3040                DAC EX1             /EXPONENT
5632    1256                TAD ADDR
```

5633	3257		DCA SAVE	
5634	2257		ISZ SAVE	
5635	1657		TAD I SAVE	
5636	3041		DCA AC1H	/HIGH ORDER MANTISSA
5637	2257		ISZ SAVE	
5640	1657		TAD I SAVE	
5641	3042		DAC AC1L	/LOW ORDER MANTISSA
5642	1253		TAD JUMP	
5643	7106		CLL RTL	
5644	7006		RTL	
5645	0260		AND MASK3	/GET BITS 0-2, IE OPCODE
5646	1265		TAD TABLE	/LOOKUP IN TABLE
5647	3254		DCA JUMP2	
5650	1654		TAD I JUMP2	
5651	3254		DCA JUMP2	
5652	5654		JMP I JUMP2	/GO THERE
5653	0000	JUMP,	0	
5654	0000	JUMP2,	0	
5655	0000	GO2,	0	
5656	0000	ADDR,	0	
5657	0000	SAVE,	0	
5660	0017	MASK3,	0017	
5661	7600	MASK5,	7600	
5662	0177	MASK7,	0177	
5663	0200	PAGENO,	0200	
5664	0400	INDRCT,	0400	
5665	5666	TABLE,	.+1	
5666	5742		EXIT	/TABLE USED IN INTERPRETING
5667	5716		FLAD	/BITS 0-2 OF PSEUDO
5670	5737		FLSU	/INSTRUCTION
5671	5761		FLMY	
5672	6305		FLDV	/IF OPCODE=0, GO TO EXIT
5673	5676		FLGT	/AND INTERPRET BITS 8-11
5674	5705		FLPT	
5675	5773		NORF	
5676	1040	FLGT,	TAD EX1	/FGET=5
5677	3044		DCA EXP	
5700	1041		TAD AC1H	
5701	3045		DCA HORD	
5702	1042		TAD AC1L	
5703	3046		DCA LORD	
5704	5201		JMP FPNT+1	
5705	1044	FLPT,	TAD EXP	/FPUT=6
5706	3656		DCA I ADDR	
5707	2256		ISZ ADDR	
5710	1045		TAD HORD	
5711	3656		DCA I ADDR	
5712	2256		ISZ ADDR	
5713	1046		TAD LORD	
5714	3656		DCA I ADDR	
5715	5201		JMP FPNT+1	
5716	4771	FLAD,	JMS I ALGN	/FLAD=1 - FIRST ALIGN EXPONENTS
5717	5201		JMP FPNT+1	/RETURN IF NO ALIGNMENT IS POSSIBLE
5720	4772		JMS I UNORM	/LARGER OF THE TWO IS IN F.A.

```
5721    7100               CLL
5722    1043               TAD OVER1           /TRIPLE PRECISION ADDITION
5723    1047               TAD OVER2           /SINCE BITS ARE SHIFTED
5724    3047               DCA OVER2           /RIGHT
5725    7004               RAL
5726    1042               TAD AC1L
5727    1046               TAD LORD
5730    3046               DCA LORD
5731    7004               RAL
5732    1041               TAD AC1H
5733    1045               TAD HORD
5734    3045               DCA HORD
5735    4770               JMS I NORM          /NORMALIZE THE RESULT
5736    5201               JMP FPNT+1

5737    4741    FLSU,      JMS I OPMINS        /FSUB=2 - NEGATE THE OPERAND
5740    5316               JMP FLAD            /ADD
5741    6400    OPMINS,    MINUS2
5742    1253    EXIT,      TAD JUMP            /OPCODE=0
5743    0260               AND MASK3           /ARE BITS 8-11=0?
5744    7450               SNA
5745    5600               JMP I FPNT          /YES=FEXT
5746    1360               TAD ACON6           /LOOKUP ON TABLE
5747    3254               DCA JUMP2
5750    1654               TAD I JUMP2
5751    3254               DCA JUMP2
5752    1200               TAD FPNT
5753    3255               DCA GO2
5754    4654               JMS I JUMP2         /CALL AS A SUBROUTINE
5755    1255               TAD GO2             /RESTORE F.P. POINTER
5756    3200               DCA FPNT
5757    5201               JMP FPNT+1          /GET NEXT PSEUDO INSTRUCTION
5760    6544    ACON6,     TABLE6-1            /CALLING CAN BE TO A DEPTH ONE
5761    7201    FLMY,      CLA IAC             /FMPY=3
5762    1040               TAD EX1
5763    1044               TAD EXP             /ADD EXPONENTS TOGETHER
5764    3044               DCA EXP
5765    4767               JMS I MULT          /MULTIPLY
5766    5201               JMP FPNT+1
5767    6221    MULT,      DMULT
5770    6600    NORM,      DNORM
5771    6020    ALGN,      ALIGN
5772    6564    UNORM,     DUNORM
5773    4770    NORF,      JMS I NORM
5774    5201               JMP FPNT+1
                *6000
6000    0000    ACMINS,    0                   /ROUTINE TO PERFORM
6001    7300               CLL CLA
6002    1047               TAD OVER2           /TRIPLE PRECISION NEGATION
6003    7041               CMA IAC             /OF FLOATING AC
6004    3047               DCA OVER2
6005    1046               TAD LORD
6006    7040               CMA
6007    7430               SZL
```

6010	7101		CLL IAC	
6011	3046		DCA LORD	
6012	1045		TAD HORD	
6013	7040		CMA	
6014	7430		SZL	
6015	7101		CLL IAC	
6016	3045		DCA HORD	
6017	5600		JMP I ACMINS	
6020	0000	ALIGN,	0	/SUBROUTINE TO ALIGN
6021	1045		TAD HORD	/BINARY POINTS FOR
6022	7640		SZA CLA	/ADD-SUBTRACT
6023	5227		JMP .+4	
6024	1046		TAD LORD	/IS MANTISSA ZERO?
6025	7650		SNA CLA	
6026	5337		JMP NOHERE	/YES
6027	1041		TAD AC1H	/IS OPERAND ZERO?
6030	7640		SZA CLA	
6031	5235		JMP .+4	
6032	1042		TAD AC1L	
6033	7650		SNA CLA	
6034	5620		JMP I ALIGN	/BOTH ARE ZERO-EXIT
6035	1040		TAD EX1	
6036	7041		CMA IAC	
6037	1044		TAD EXP	
6040	7450		SNA	/ARE EXPONENTS EQUAL?
6041	5314		JMP DONE	/YES
6042	7500		SMA	
6043	7041		CMA IAC	
6044	3342		DCA AMOUNT	
6045	1342		TAD AMOUNT	
6046	1343		TAD TEST2	
6047	7700		SMA CLA	/CAN EXPONENTS BE ALIGNED?
6050	5256		JMP .+6	/YES
6051	4316		JMS OUTGO	/NO
6052	7430		SZL	
6053	1350		TAD TAG2	
6054	1347		TAD TAG1	
6055	5325		JMP NOGO	
6056	4316		JMS OUTGO	
6057	7420		SNL	/SET UP ADDRESSES
6060	1350		TAD TAG2	
6061	1347		TAD TAG1	
6062	3344		DCA TEST3	
6063	1342		TAD AMOUNT	
6064	7041		CMA IAC	
6065	1744		TAD I TEST3	
6066	3744		DCA I TEST3	
6067	2344		ISZ TEST3	
6070	1344		TAD TEST3	
6071	3345		DCA TEST4	
6072	2345		ISZ TEST4	
6073	1345		TAD TEST4	
6074	3346		DCA TEST5	
6075	2346		ISZ TEST5	

6076	7100	SHIFT,	CLL	/THIS ROUTINE DOES
6077	1744		TAD I TEST3	/THE ACUTAL SHIFTING
6100	7510		SPA	
6101	7020		CML	
6102	7010		RAR	
6103	3744		DCA I TEST3	
6104	1745		TAD I TEST4	
6105	7010		RAR	
6106	3745		DCA I TEST4	
6107	1746		TAD I TEST5	
6110	7010		RAR	
6111	3746		DCA I TEST5	
6112	2342		ISZ AMOUNT	
6113	5276		JMP SHIFT	
6114	2220	DONE,	ISZ ALIGN	
6115	5620		JMP I ALIGN	
6116	0000	OUTGO,	0	/DETERMINE WHICH TO SHIFT
6117	1040		TAD EX1	
6120	7041		CMA IAC	
6121	1044		TAD EXP	
6122	7004		RAL	
6123	7200		CLA	
6124	5716		JMP I OUTGO	
6125	3344	NOGO,	DCA TEST3	/CAN'T BE ALIGNED
6126	1744		TAD I TEST3	/LARGEST GOES INTO FAC
6127	3044		DCA EXP	
6130	2344		ISZ TEST3	
6131	1744		TAD I TEST3	
6132	3045		DCA HORD	
6133	2344		ISZ TEST3	
6134	1744		TAD I TEST3	
6135	3046		DCA LORD	
6136	5620		JMP I ALIGN	
6137	1040	NOHERE,	TAD EX1	/MANTISSA=0
6140	3044		DCA EXP	
6141	5314		JMP DONE	
6142	0000	AMOUNT,	0	
6143	0030	TEST2,	0030	
6144	0000	TEST3,	0	
6145	0000	TEST4,	0	
6146	0000	TEST5,	0	
6147	0044	TAG1,	EXP	
6150	7774	TAG2,	EX1-EXP	
6151	5601	RETN2,	FPNT+1	
6152	1362	ERROR1,	TAD GOOF	/DIVISION BY ZERO
6153	3044		DCA EXP	/SET TO LARGEST + VALUE
6154	1362		TAD GOOF	
6155	3045		DCA HORD	
6156	7040		CMA	
6157	3046		DCA LORD	
6160	2061		ISZ FLAG	/SET FLAG
6161	5751		JMP I RETN2	
6162	3777	GOOF,	3777	

```
6163    0000    SQUARE,    0
6164    4407               JMS I 0007
6165    6052               FPUT FPAC1
6166    3052               FMPY FPAC1
6167    0000               FEXT
6170    5763               JMP I SQUARE
6171    0000    EXIT6,     0              /DUMMY SUBROUTINE
6172    5771               JMP I EXIT6
                *6200
6200    0000    DIV1,      0              /SHIFT FAC RIGHT
6201    7300               CLA CLL
6202    1045               TAD HORD
6203    7510               SPA
6204    7120               CLL CML
6205    7010               RAR
6206    3045               DCA HORD
6207    1046               TAD LORD
6210    7010               RAR
6211    3046               DCA LORD
6212    1047               TAD OVER2
6213    7010               RAR
6214    3047               DCA OVER2
6215    7100               CLL
6216    2044               ISZ EXP
6217    7000               NOP
6220    5600               JMP I DIV1
6221    0000    DMULT,     0              /DOUBLE PRECISION MULTIPLY
6222    7300               CLA CLL        /SAVE PRODUCT TRIPLE PRECISION
6223    1363               TAD SMACLA
6224    3347               DCA SNSWIT     /CALLS A SINGLE PRECISION
6225    4336               JMS SIGN       /MULTIPLY 3 TIMES
6226    1042               TAD AC1L
6227    3756               DCA I MP2PT
6230    1046               TAD LORD
6231    4755               JMS I MP4PT
6232    7200               CLA
6233    1757               TAD I MP5PT
6234    3047               DCA OVER2
6235    1045               TAD HORD
6236    3756               DCA I MP2PT
6237    1042               TAD AC1L
6240    4755               JMS I MP4PT
6241    1047               TAD OVER2
6242    3047               DCA OVER2
6243    7004               RAL
6244    1757               TAD I MP5PT
6245    3367               DCA D
6246    7004               RAL
6247    3370               DCA KEEP
6250    1041               TAD AC1H
6251    3756               DCA I MP2PT
6252    1046               TAD LORD
6253    4755               JMS I MP4PT
```

4-6

```
6254    1047                TAD OVER2
6255    3047                DCA OVER2
6256    7004                RAL
6257    1757                TAD I MP5PT
6260    1367                TAD D
6261    3367                DCA D
6262    7004                RAL
6263    1370                TAD KEEP
6264    3370                DCA KEEP
6265    1045                TAD HORD
6266    3756                DCA I MP2PT
6267    1041                TAD AC1H
6270    4755                JMS I MP4PT
6271    1367                TAD D
6272    3046                DCA LORD
6273    7004                RAL
6274    1757                TAD I MP5PT
6275    1370                TAD KEEP
6276    3045                DCA HORD
6277    4760                JMS I NORMF
6300    3047                DCA OVER2
6301    2365                ISZ SGN
6302    5621                JMP I DMULT
6303    4773                JMS I MINS
6304    5621                JMP I DMULT
6305    1041    FLDV,       TAD AC1H
6306    7640                SZA CLA
6307    5313                JMP .+4
6310    1042                TAD AC1L
6311    7650                SNA CLA
6312    5774                JMP I ERROR         /DIVISION BY ZERO
6313    1040                TAD EX1
6314    7041                CMA IAC
6315    1044                TAD EXP
6316    7001                IAC
6317    3044                DCA EXP             /SUBTRACT EXPONENTS
6320    1362                TAD SPACLA
6321    3347                DCA SNSWIT
6322    4336                JMS SIGN            /SET UP SIGNS
6323    4761                JMS I DIVIDE        /DIVIDE
6324    1757                TAD I MP5PT
6325    1041                TAD AC1H            /ROUND OFF IN 23RD BIT
6326    7630                SZL CLA
6327    7001                IAC
6330    3042                DCA AC1L
6331    3041                DCA AC1H
6332    2365                ISZ SGN             /TEST SIGN
6333    4773                JMS I MINS          /NEGATE
6334    5735                JMP I .+1           /ADD ROUNDING
6335    5720                FLAD+2
6336    0000    SIGN,       0                   /TEST SIGN OF RESULT
6337    1366                TAD REST            /SET UP BY MULTIPLY AND
6340    3365                DCA SGN             /DIVIDE
```

6341	1045		TAD HORD	
6342	7700		SMA CLA	
6343	5346		JMP .+3	
6344	4773		JMS I MINS	
6345	2365		ISZ SGN	
6346	1041		TAD AC1H	
6347	7700	SNSWIT,	SMA CLA	/OR SPA CLA
6350	5736		JMP I SIGN	
6351	4771		JMS I MINS2	
6352	2365		ISZ SGN	
6353	7000		NOP	
6354	5736		JMP I SIGN	
6355	6437	MP4PT,	MP4	
6356	6471	MP2PT,	MP2	
6357	6465	MP5PT,	MP5	
6360	6600	NORMF,	DNORM	
6361	6472	DIVIDE,	DUBDIV	
6362	7710	SPACLA,	SPA CLA	
6363	7700	SMACLA,	SMA CLA	
6364	5601	RETURN,	FPNT+1	
6365	0000	SGN,	0	
6366	7776	REST,	7776	
6367	0000	D,	0	
6370	0000	KEEP,	0	
6371	6400	MINS2,	MINUS2	/-AC1H,AC1L
6372	6420	RAR2,	DIV2	/AC1H,AC1L/2
6373	6000	MINS,	ACMINS	
6374	6152	ERROR,	ERROR1	
		*6400		
6400	0000	MINUS2,	0	/NEGATE OPERAND
6401	7300		CLA CLL	/TRIPLE PRECISION
6402	1043		TAD OVER1	
6403	7041		CMA IAC	
6404	3043		DCA OVER1	
6405	1042		TAD AC1L	
6406	7040		CMA	
6407	7430		SZL	
6410	7101		CLL IAC	
6411	3042		DCA AC1L	
6412	1041		TAD AC1H	
6413	7040		CMA	
6414	7430		SZL	
6415	7101		CLL IAC	
6416	3041		DCA AC1H	
6417	5600		JMP I MINUS2	
6420	0000	DIV2,	0	/SHIFT OPERAND RIGHT
6421	7300		CLA CLL	/TRIPLE PRECISION
6422	1041		TAD AC1H	
6423	7510		SPA	
6424	7120		CLL CML	
6425	7010		RAR	
6426	3041		DCA AC1H	
6427	1042		TAD AC1L	

```
6430    7010                RAR
6431    3042                DCA AC1L
6432    1043                TAD OVER1
6433    7010                RAR
6434    3043                DCA OVER1
6435    7100                CLL
6436    5620                JMP I DIV2
6437    0000    MP4,        0                   /SINGLE PRECISION MULTIPLY
6440    3266                DCA MP1             /12 BITS BY 12 BITS
6441    3265                DCA MP5
6442    1270                TAD THIR
6443    3267                DCA MP3
6444    7100                CLL
6445    1266                TAD MP1
6446    7010                RAR
6447    3266                DCA MP1
6450    1265                TAD MP5
6451    7420                SNL
6452    5255                JMP .+3
6453    7100                CLL
6454    1271                TAD MP2
6455    7010                RAR
6456    3265                DCA MP5
6457    2267                ISZ MP3
6460    5245                JMP MP4+6
6461    1266                TAD MP1
6462    7010                RAR
6463    7100                CLL
6464    5637                JMP I MP4
6465    0000    MP5,        0
6466    0000    MP1,        0
6467    0000    MP3,        0
6470    7764    THIR,       7764
6471    0000    MP2,        0
6472    0000    DUBDIV,     0                   /DOUBLE PRECISION DIVIDE
6473    7300                CLA CLL
6474    3051                DCA QUOL
6475    1344                TAD MIF
6476    3267                DCA MP3
6477    5306                JMP DVX
6500    1046    DV3,        TAD LORD
6501    7004                RAL
6502    3046                DCA LORD
6503    1045                TAD HORD
6504    7004                RAL
6505    3045                DCA HORD
6506    1042    DVX,        TAD AC1L
6507    1046                TAD LORD
6510    3271                DCA MP2
6511    7004                RAL
6512    1045                TAD HORD
6513    1041                TAD AC1H
6514    7420                SNL
```

6515	5321		JMP DV2-1	
6516	3045		DCA HORD	
6517	1271		TAD MP2	
6520	3046		DCA LORD	
6521	7200		CLA	
6522	1051	DV2,	TAD QUOL	
6523	7004		RAL	
6524	3051		DCA QUOL	
6525	1047		TAD OVER2	
6526	7004		RAL	
6527	3047		DCA OVER2	
6530	2267		ISZ MP3	
6531	5300		JMP DV3	
6532	1051		TAD QUOL	
6533	3046		DCA LORD	
6534	1045		TAD HORD	
6535	7106		CLL RTL	
6536	3265		DCA MP5	
6537	1047		TAD OVER2	
6540	3045		DCA HORD	
6541	3047		DCA OVER2	
6542	1265		TAD MP5	
6543	5672		JMP I DUBDIV	
6544	7751	MIF,	7751	
6545	6163	TABLE6,	SQUARE	/TABLE FOR INTERPRETATION
6546	6656		SQROOT	/OF BITS 8-11
6547	6171		EXIT6	/CONTAINS ABSOLUTE ADDRESSES
6550	6171		EXIT6	/OF PROGRAMS CALLED AS
6551	6171		EXIT6	/SUBROUTINES
6552	6171		EXIT6	/EXIT6=A DUMMY OR NOP
6553	6171		EXIT6	
6554	6171		EXIT6	
6555	6171		EXIT6	
6556	6171		EXIT6	
6557	6171		EXIT6	
6560	6171		EXIT6	
6561	6171		EXIT6	
6562	6171		EXIT6	
6563	6171		EXIT6	
6564	0000	DUNORM,	0	
6565	4220		JMS DIV2	/SHIFT OPERAND RIGHT
6566	4772		JMS I RAR1	
6567	2040		ISZ EX1	
6570	7000		NOP	
6571	5764		JMP I DUNORM	
6572	6200	RAR1,	DIV1	
		*6600		
6600	0000	DNORM,	0	/SUBROUTINE TO NORMALIZE
6601	7300		CLA CLL	/FLOATING ACCUMULATOR
6602	3255		DCA AMT1	
6603	3254		DCA SIGN1	
6604	1045		TAD HORD	
6605	7510		SPA	/IS MANTISSA NEGATIVE?

```
6606    2254            ISZ SIGN1       /YES
6607    7640            SZA CLA         /IS MANTISSA=0?
6610    5217            JMP GO6         /NO
6611    1046            TAD LORD
6612    7640            SZA CLA
6613    5217            JMP GO6
6614    1047            TAD OVER2
6615    7650            SNA CLA
6616    5251            JMP EXIT2       /YES
6617    1254    GO6,    TAD SIGN1       /NO
6620    7640            SZA CLA         /NEGATIVE?
6621    4653            JMS I NEG       /YES
6622    1045    LOP,    TAD HORD        /WILL SHIFT BE TOO FAR?
6623    7104            RAL CLL
6624    7710            SPA CLA
6625    5241            JMP EXIT1       /YES
6626    1047            TAD OVER2
6627    7104            CLL RAL
6630    3047            DCA OVER2
6631    1046            TAD LORD        /NO SHIFT MANTISSA LEFT
6632    7004            RAL
6633    3046            DCA LORD
6634    1045            TAD HORD
6635    7004            RAL
6636    3045            DCA HORD
6637    2255            ISZ AMT1        /COUNT NO. OF TIMES SHIFTED
6640    5222            JMP LOP
6641    1255    EXIT1,  TAD AMT1        /CORRECT EXPONENT
6642    7041            CMA IAC
6643    1044            TAD EXP
6644    3044            DCA EXP
6645    1254            TAD SIGN1       /NEGATIVE?
6646    7640            SZA CLA
6647    4653            JMS I NEG       /YES
6650    5600            JMP I DNORM
6651    3044    EXIT2,  DCA EXP         /SET TO ZERO
6652    5600            JMP I DNORM
6653    6000    NEG,    ACMINS
6654    0000    SIGN1,  0
6655    0000    AMT1,   0
6656    0000    SQROOT, 0               /FLOATING SQUARE ROOT
6657    3362            DCA FLAG1       /TAKES ROOT OF ABSOLUTE
6660    4407            JMS I 007
6661    6052            FPUT FPAC1      /VALUE
6662    0000            FEXT            /NEWTON'S METHOD IS USED
6663    1045            TAD HORD
6664    7710            SPA CLA
6665    5351            JMP SQEND1      /NUMBER IS NEGATIVE
6666    1044            TAD EXP
6667    7100            CLL
6670    7510            SPA
6671    7020            CML
6672    7010            RAR
6673    3356            DCA ITER1       /MAKE FIRST APPROXIMATION
```

```
6674    7430              SZL
6675    2356              ISZ ITER1
6676    7000              NOP
6677    1361              TAD SQCON1
6700    3357              DCA ITER1+1
6701    3360              DCA ITER1+2
6702    1053              TAD FPAC1+1
6703    7640              SZA CLA
6704    5310              JMP CLCU
6705    1054              TAD FPAC1+2
6706    7650              SNA CLA
6707    5354              JMP SQEND          /NUMBER=0
6710    4407    CLCU,     JMS I 0007
6711    5052              FGET FPAC1
6712    4356              FDIV ITER1
6713    1356              FADD ITER1
6714    0000              FEXT
6715    7240              CLA CMA
6716    1044              TAD EXP
6717    3044              DCA EXP
6720    1044              TAD EXP
6721    7041              CMA IAC
6722    1356              TAD ITER1
6723    7640              SZA CLA            /ARE EXPONENTS EQUAL?
6724    5345              JMP ROOTGO         /NO
6725    1045              TAD HORD           /ARE HIGH-ORDER MANTISSAS EQUAL?
6726    7041              CMA IAC
6727    1357              TAD ITER1+1
6730    7640              SZA CLA
6731    5345              JMP ROOTGO         /NO
6732    1046              TAD LORD
6733    7041              CMA IAC
6734    1360              TAD ITER1+2        /DO LOW-ORDER MANTISSAS AGREE?
6735    7500              SMA
6736    7041              CMA IAC            /WITHIN ONE BIT?
6737    7001              IAC
6740    7710              SPA CLA
6741    5345              JMP ROOTGO         /NO
6742    1362              TAD FLAG1
6743    3061              DCA FLAG
6744    5656              JMP I SQROOT
6745    4407    ROOTGO,   JMS I 0007
6746    6356              FPUT ITER1
6747    0000              FEXT
6750    5310              JMP CLCU
6751    4653    SQEND1,   JMS I NEG          /NEGATE FAC
6752    2362              ISZ FLAG1
6753    5260              JMP SQROOT+2
6754    3044    SQEND,    DCA EXP
6755    5656              JMP I SQROOT
6756    0000    ITER1,    0
6757    0000              0
6760    0000              0
6761    3015    SQCON1,   3015
6762    0000    FLAG1,    0
```

4-12

ACMINS	6000	FLPT	5705	NORMF	6360
ACON6	5760	FLSU	5737	OPMINS	5741
AC1H	0041	FPAC1	0052	OUTGO	6116
AC1L	0042	FPNT	5600	OVER1	0043
ADDR	5656	GOOF	6162	OVER2	0047
ALGN	5771	GO2	5655	PAGENO	5663
ALIGN	6020	GO6	6617	QUOL	0051
AMOUNT	6142	HORD	0045	RAR1	6572
AMT1	6655	INDRCT	5664	RAR2	6372
CLCU	6710	ITER1	6756	REST	6366
D	6367	JUMP	5653	RETN2	6151
DIVIDE	6361	JUMP2	5654	RETURN	6364
DIV1	6200	KEEP	6370	ROOTGO	6745
DIV2	6420	LOOP01	5627	SAVE	5657
DMULT	6221	LOP	6622	SGN	6365
DNORM	6600	LORD	0046	SHIFT	6076
DONE	6114	MASK3	5660	SIGN	6336
DUBDIV	6472	MASK5	5661	SIGN1	6654
DUNORM	6564	MASK7	5662	SMACLA	6363
DVX	6506	MIF	6544	SNSWIT	6347
DV2	6522	MINS	6373	SPACLA	6362
DV3	6500	MINS2	6371	SQCON1	6761
ERROR	6374	MINUS2	6400	SQEND	6754
ERROR1	6152	MP1	6466	SQEND1	6751
EXIT	5742	MP2	6471	SQROOT	6656
EXIT1	6641	MP2PT	6356	SQUARE	6163
EXIT2	6651	MP3	6467	TABLE	5665
EXIT6	6171	MP4	6437	TABLE6	6545
EXP	0044	MP4PT	6355	TAG1	6147
EXP1	0050	MP5	6465	TAG2	6150
EX1	0040	MP5PT	6357	TEST2	6143
FLAD	5716	MULT	5767	TEST3	6144
FLAG	0061	NEG	6653	TEST4	6145
FLAG1	6762	NOGO	6125	TEST5	6146
FLDV	6305	NOHERE	6137	THIR	6470
FLGT	5676	NORF	5773	UNORM	5772
FLMY	5761	NORM	5770		

```
                    /FLOATING POINT I/O ROUTINES
                    /REQUIRES FLOATING POINT INTERPRETER
                    /ENTRY AT 0007

                    *7
0007  5600          FPNT,     5600

                    *44
0044  0000          EXPONT,   0
0045  0000          HORDER,   0
0046  0000          LORDER,   0

                    *52
0052  0000          FPAC1,    0
0053  0000                    0
0054  0000                    0
0055  7777          SWIT1,    7777    /IF = 0, NO CR-LF AFTER OUTPUT
0056  7777          SWIT2,    7777    /IF = 0, NO LF AFTER CR IN INPUT
0057  0000          CHAR,     0       /CONTAINS LAST CHARACTER READ
0060  0000          DSWIT,    0       /= 0 IF NO CONVERSION TOOK PLACE

                    *6767
6767  0000          PRCHAR,   0
6770  1056                    TAD SWIT2
6771  7650                    SNA CLA
6772  5767                    JMP I PRCHAR
6773  1377                    TAD LFED
6774  4776                    JMS I OPUT
6775  5767                    JMP I PRCHAR
6776  7344          OPUT,     OUT
6777  0212          LFED,     0212
```

```
/DOUBLE PRECISION DECIMAL-BINARY
/INPUT AND CONVERSION
*7000

7000    0000    DECONV,     0
7001    7200                CLA                     /INITIALIZE MANTISSA
7002    3045                DCA HORDER
7003    3046                DCA LORDER
7004    3265                DCA SIGN
7005    3266                DCA DNUMBR
7006    4342                JMS INPUT
7007    1336                TAD PLUS                /TEST FOR SIGN
7010    7450                SNA
7011    5217                JMP DECON
7012    1335                TAD MINUS
7013    7440                SZA
7014    5220                JMP .+4
7015    7240                CLA CMA
7016    3265                DCA SIGN                /IF-, SET SWITCH
7017    4342    DECON,      JMS INPUT
7020    7200                CLA
7021    1057                TAD CHAR                /IS IT A DIGIT
7022    1337                TAD MIN9
7023    7500                SMA
7024    5600                JMP I DECONV            /NO
7025    1340                TAD PLUS12
7026    7510                SPA
7027    5600                JMP I DECONV            /NO
7030    3263                DCA DIGIT               /YES
7031    1045                TAD HORDER
7032    0341                AND MASK                /OVERFLOW?
7033    7440                SZA
7034    5217                JMP DECON               /YES-IGNORE
7035    2060                ISZ DSWIT
7036    2266                ISZ DNUMBR              /INDEX NUMBER OF DIGITS
7037    4241                JMS MULT10
7040    5217                JMP DECON               /CONTINUE
7041    0000    MULT10,     0                       /ROUTINE TO MULTIPLY
7042    1046                TAD LORDER              /DOUBLE PRECISION WORD
7043    3261                DCA LTEMP               /BY TEN (DECIMAL)
7044    1045                TAD HORDER              /REMAIN=REMAINDER
7045    3262                DCA HTEMP
7046    3264                DCA REMAIN
7047    4267                JMS MULT2               /CALL SUBROUTINE TO
7050    4267                JMS MULT2               /MULTIPLY BY TWO
7051    4303                JMS DUBLAD              /CALL DOUBLE ADD
7052    4267                JMS MULT2
7053    1263                TAD DIGIT               /ADD LAST DIGIT RECEIVED
7054    3261                DCA LTEMP
7055    3262                DCA HTEMP
7056    4303                JMS DUBLAD
7057    1264                TAD REMAIN              /EXIT WITH REMAINDER
7060    5641                JMP I MULT10            /IN AC

7061    0000    LTEMP,      0                       /DOUBLE PRECISION WORD
7062    0000    HTEMP,      0
7063    0000    DIGIT,      0                       /STORAGE FOR DIGIT
7064    0000    REMAIN,     0
7065    0000    SIGN,       0                       /=0 IF PLUS: =7777 IF MINUS
7066    0000    DNUMBR,     0                       /=NUMBER OF DIGITS
7067    0000    MULT2,      0                       /MULTIPLY LORDER, HORDER BY 2
```

```
7070    7300            CLA CLL
7071    1046            TAD LORDER
7072    7004            RAL
7073    3046            DCA LORDER
7074    1045            TAD HORDER
7075    7004            RAL
7076    3045            DCA HORDER
7077    1264            TAD REMAIN
7100    7004            RAL
7101    3264            DCA REMAIN
7102    5667            JMP I MULT2
7103    0000    DUBLAD,     0                   /DOUBLE PRECISION ADDITION
7104    7300            CLA CLL
7105    1046            TAD LORDER
7106    1261            TAD LTEMP
7107    3046            DCA LORDER
7110    7004            RAL
7111    1045            TAD HORDER
7112    1262            TAD HTEMP
7113    3045            DCA HORDER
7114    7004            RAL
7115    1264            TAD REMAIN
7116    3264            DCA REMAIN
7117    5703            JMP I DUBLAD
7120    0000    MSIGN,      0                   /ROUTINE TO FORM
7121    7300            CLA CLL                  /2'S COMPLEMENT IF
7122    2265            ISZ SIGN                 /MINUS
7123    5720            JMP I MSIGN              /SIGN=0000: EXIT
7124    1046            TAD LORDER
7125    7041            CMA IAC
7126    3046            DCA LORDER
7127    1045            TAD HORDER
7130    7040            CMA
7131    7430            SZL
7132    7001            IAC
7133    3045            DCA HORDER
7134    5720            JMP I MSIGN

7135    7776    MINUS,      253-255              /TEST FOR SIGN
7136    7525    PLUS,       -253
7137    7506    MIN9,       -272                 /TEST FOR DIGIT
7140    0012    PLUS12,     272-260
7141    7600    MASK,       7600                 /TEST FOR OVERFLOW
                /INPUT A CHARACTER, IF CR, TEST
                /INPUT SWITCH TO SEE IF LF SHOULD
                /BE TYPED. IF RUBOUT, RESTART INPUT
7142    0000    INPUT,      0                   /INPUT A CHARACTER
7143    7200            CLA
7144    6031            KSF
7145    5344            JMP .-1
7146    6036            KRB
7147    3057            DCA CHAR
7150    1057            TAD CHAR
7151    4766            JMS I OUTPUT
7152    1057            TAD CHAR
7153    7450            SNA
7154    5343            JMP INPUT+1              /IGNORE BLANKS
7155    1370            TAD MRBOUT
7156    7450            SNA
7157    5767            JMP I RESTRT             /RUBOUT-RESTART INPUT
7160    1371            TAD MINCR
```

```
7161    7650            SNA CLA
7162    4765            JMS I PRINT         /CR - SEE IF TO BE FOLLOWED
7163    1057            TAD CHAR            /BY LF
7164    5742            JMP I INPUT         /EXIT ROUTINE

7165    6767    PRINT,  PRCHAR
7166    7344    OUTPUT, OUT
7167    7401    RESTRT, FLINTP+1
7170    7401    MRBOUT, -377
7171    0162    MINCR,  377-215


                /FLOATING OUTPUT "E" FORMAT
                /USES:   TSF
                /        JMP .-1
                /        TLS
                *7200
7200    0000    FLOUTP, 0
7201    4217            JMS FOUTCN          /CONVERT MANTISSA AND OUTPUT
7202    1324            TAD BEXP
7203    3044            DCA EXPONT
7204    1343            TAD CHE
7205    4344            JMS OUT
7206    4737            JMS I FEXPPT        /CONVERT EXPONENT AND OUTPUT
7207    1055            TAD SWIT1           /PRINT CR-LF?
7210    7650            SNA CLA
7211    5600            JMP I FLOUTP        /NO-EXIT
7212    1341            TAD CARRTN /YES
7213    4344            JMS OUT
7214    1342            TAD LNFEED
7215    4344            JMS OUT
7216    5600            JMP I FLOUTP        /EXIT
                /THIS WHOLE SUBROUTINE MAY BE ALTERED TO BUFFER
                /THE OUTPUT DIGITS : CHANGE JMS OUTDG TO DCA I 10, ETC.
7217    0000    FOUTCN, 0
7220    7300            CLA CLL
7221    1045            TAD HORDER          /NUMBER>0??
7222    7710            SPA CLA
7223    7220            CLA CML             /NO SET LINK
7224    1327            TAD SPLUS           /YES
7225    7430            SZL
7226    1330            TAD SMINUS /NO
7227    4344            JMS OUT
7230    4352            JMS OUTDG           /OUTPUT "0"
7231    1331            TAD PERIOD
7232    4344            JMS OUT             /OUTPUT "."
7233    7300            CLA CLL
7234    1045            TAD HORDER
7235    7700            SMA CLA
7236    5242            JMP FG01
7237    7040            CMA                 /NUMBER IS NEGATIVE
7240    3733            DCA I SNPT          /NEGATE
7241    4732            JMS I MSNPT
7242    7240    FG01,   CLA CMA             /SUBTRACT 1 FROM BINARY EXPONENT
7243    1044            TAD EXPONT          /COMPENSATE AT FG04
7244    3044            DCA EXPONT
7245    3324            DCA BEXP            /INITIALIZE DECIMAL EXPONENT
7246    1044    FG02,   TAD EXPONT          /IS -4<EXPONENT<-1
7247    7500            SMA
7250    5263            JMP FG03            /TOO LARGE: MULTIPLY BY 1/10
7251    1326            TAD FOUR
```

```
7252   7700           SMA CLA
7253   5270           JMP FG04
7254   4407           JMS I FPNT          /TOO SMALL-TIMES TEN
7255   3740           FMPY I TENPT        /TEN
7256   0000           FEXT
7257   7240           CLA CMA
7260   1324           TAD BEXP
7261   3324           DCA BEXP
7262   5246           JMP FG02
7263   4407   FG03,   JMS I FPNT
7264   3372           FMPY C.10           /ONE TENTH
7265   0000           FEXT
7266   2324           ISZ BEXP
7267   5246           JMP FG02
7270   3734   FG04,   DCA I DPT           /MULTIPLY BY TWO
7271   4736           JMS I M2PT          /IE.SHIFT LEFT
7272   4735           JMS I M10PT         /MULTIPLY BY TEN
7273   7410           SKP
7274   4357   FG05A,  JMS DIVTWO          /COMPENSATE FOR
7275   2044           ISZ EXPONT          /BINARY EXPONENT
7276   5274           JMP FG05A
7277   7450           SNA                 /IS FIRST DIGIT A ZERO
7300   5311           JMP FG07            /YES, IGNORE
7301   4352   FG06,   JMS OUTDG           /MULTIPLICATIONS YIELD
7302   1325           TAD MINUS7          /DECIMAL DIGITS AS HIGH
7303   3044           DCA EXPONT          /ORDER REMAINDERS
7304   4735   FG06A,  JMS I M10PT         /IE. .672X10=6+.72.. ETC
7305   4352           JMS OUTDG
7306   2044           ISZ EXPONT          /7 DIGITS OUTPUT??
7307   5304           JMP FG06A           /NO: CONTINUE
7310   5617           JMP I FOUTCN        /YES:EXIT

7311   7240   FG07,   CLA CMA             /IGNORE FIRST DIGIT
7312   1324           TAD BEXP            /SUBTRACT 1 FROM
7313   3324           DCA BEXP            /DECIMAL EXPONENT
7314   1045           TAD HORDER
7315   7640           SZA CLA
7316   5322           JMP .+4             /IS MANTISSA ZERO?
7317   1046           TAD LORDER
7320   7650           SNA CLA
7321   3324           DCA BEXP            /YES: EXP=0
7322   7240           CLA CMA
7323   5302           JMP FG06+1

7324   0000   BEXP,   0                   /CONTAINS DECIMAL EXPONENT
7325   7772   MINUS7, 7772                /NUMBER OF DIGTS OUTPUT
7326   0004   FOUR,   0004
7327   0253   SPLUS,  253
7330   0002   SMINUS, 255-253
7331   0256   PERIOD, 256
7332   7120   MSNPT,  MSIGN
7333   7065   SNPT,   SIGN                /POINTERS
7334   7063   DPT,    DIGIT
7335   7041   M10PT,  MULT10
7336   7067   M2PT,   MULT2
7337   7522   FEXPPT, FEXC
7340   7504   TENPT,  TEN
7341   0215   CARRTN, 0215
7342   0212   LNFEED, 0212
7343   0305   CHE,    305
```

```
7344    0000    OUT,        0                   /OUTPUT ONE ASCII CHARACTER
7345    6041                TSF
7346    5345                JMP .-1
7347    6046                TLS
7350    7200                CLA
7351    5744                JMP I OUT

7352    0000    OUTDG,      0                   /OUTPUT ONE DIGIT
7353    1356                TAD C260
7354    4344                JMS OUT
7355    5752                JMP I OUTDG

7356    0260    C260,       0260

7357    0000    DIVTWO,     0                   /DIVIDE BY TWO IE.
7360    7110                CLL RAR             /ROTATE RIGHT
7361    3344                DCA OUT             /TEMPORARY STORAGE
7362    1045                TAD HORDER
7363    7010                RAR
7364    3045                DCA HORDER
7365    1046                TAD LORDER
7366    7010                RAR
7367    3046                DCA LORDER
7370    1344                TAD OUT
7371    5757                JMP I DIVTWO
7372    7775    C.10,       7775                /CONSTANT .10 USED IN
7373    3146                3146                /FLOATING OUTPUT-PROVIDES
7374    3147                3147
                /FLOATING POINT INPUT
                *7400
7400    0000    FLINTP,     0
7401    7240                CLA CMA             /INITIALIZE "PERIOD SWITCH"
7402    3313                DCA PRSW
7403    3060                DCA DSWIT
7404    4716                JMS I DPCVPT        /7777 = NO PERIOD
7405    7200                CLA
7406    1057                TAD CHAR
7407    1312                TAD PER
7410    7640                SZA CLA
7411    5220                JMP FIG01
7412    1313                TAD PRSW            /PERIOD FOUND
7413    7650                SNA CLA             /SECOND PERIOD
7414    5222                JMP FIG02           /YES,TERMINATE
7415    3721                DCA I DPN           /NO - SET NUMBER OF DIGITS TO 0
7416    3313                DCA PRSW            /SET PERIOD SWITCH TO 0
7417    5717                JMP I DPCSPT        /CONVERT REST OF STRING

7420    1313    FIG01,      TAD PRSW            /PERIOD READ IN PREVIOUSLY?
7421    7650                SNA CLA
7422    1721    FIG02,      TAD I DPN           /YES:-NUMBER OF DIGITS IN SEXP
7423    7041                CMA IAC             /NO
7424    3314                DCA SEXP
7425    4720                JMS I MSGNPT        /TEST SIGN
7426    1311    FIG03,      TAD C27
7427    3044                DCA EXPONT
7430    4407                JMS I FPNT          /NORMALIZE F.P. NUMBER
7431    7000                FNOR
7432    6052                FPUT FPAC1          /SAVE NUMBER
7433    0000                FEXT
7434    1057                TAD CHAR
7435    1310                TAD MINUSE
```

4-19

```
7436   7640            SZA CLA               /"E" READ IN?
7437   5252            JMP ENDFI             /NO
7440   4716            JMS I DPCVPT          /YES - CONVERT DECIMAL EXPONENT
7441   4720            JMS I MSGNPT          /TEST SIGN
7442   1045            TAD HORDER            /EXPONENT TOO LARGE??
7443   7510            SPA
7444   7001            IAC
7445   7640            SZA CLA
7446   5277            JMP EXCESS            /YES
7447   1046            TAD LORDER           /NO:DECIMAL POINT IS
7450   1314            TAD SEXP             /C(SEXP)PLACES TO RIGHT
7451   3314            DCA SEXP             /OF LAST DIGIT
              /END OF FLOATING POINT INPUT
              /COMPENSATE FOR DECIMAL EXPONENTS

7452   4407   ENDFI,   JMS I FPNT           /RESTORE MANTISSA
7453   5052            FGET FPAC1
7454   0000            FEXT
7455   1314            TAD SEXP
7456   7450            SNA
7457   5600            JMP I FLINTP
7460   7700            SMA CLA
7461   5270            JMP FIGO4
7462   4407            JMS I FPNT           /. IS TO THE LEFT:
7463   3707            FMPY I PC.10         /TIMES .1000
7464   0000            FEXT
7465   2314            ISZ SEXP
7466   5255            JMP ENDFI+3
7467   5600            JMP I FLINTP

7470   4407   FIGO4,   JMS I FPNT           /. IS TO THE RIGHT:
7471   3304            FMPY TEN             /MULTIPLY BY 10
7472   0000            FEXT
7473   7240            CLA CMA
7474   1314            TAD SEXP
7475   3314            DCA SEXP
7476   5255            JMP ENDFI+3
7477   1315   EXCESS,  TAD C3777
7500   3044            DCA EXPONT
7501   1315            TAD C3777
7502   3045            DCA HORDER
7503   5600            JMP I FLINTP
7504   0004   TEN,     0004
7505   2400            2400
7506   0000            0000
7507   7372   PC.10,   C.10                 /.10
7510   7473   MINUSE,  -305
7511   0027   C27,     0027
7512   7522   PER,     -256
7513   0000   PRSW,    0
7514   0000   SEXP,    0                    /CONTAINS DECIMAL EXPONENT
7515   3777   C3777,   3777

7516   7000   DPCVPT,  DECONV
7517   7017   DPCSPT,  DECON
7520   7120   MSGNPT,  MSIGN
7521   7066   DPN,     DNUMBR
              /OUTPUT THE EXPONENT

7522   0000   FEXC,    0
7523   7300            CLA CLL
```

```
7524    1044                TAD EXPONT
7525    7510                SPA
7526    7061                CMA IAC CML
7527    3044                DCA EXPONT
7530    1366                TAD C253
7531    7430                SZL
7532    1367                TAD C255
7533    4774                JMS I DGPT
7534    3045                DCA HORDER
7535    1044                TAD EXPONT
7536    2045                ISZ HORDER
7537    1370                TAD M144
7540    7500                SMA
7541    5336                JMP .-3
7542    1371                TAD C144
7543    3044                DCA EXPONT
7544    7040                CMA
7545    1045                TAD HORDER
7546    7440                SZA
7547    4774                JMS I DGPT
7550    3045                DCA HORDER
7551    1044                TAD EXPONT
7552    2045                ISZ HORDER
7553    1372                TAD M12
7554    7500                SMA
7555    5352                JMP .-3
7556    1373                TAD C12
7557    3046                DCA LORDER
7560    7240                CLA CMA
7561    1045                TAD HORDER
7562    4774                JMS I DGPT
7563    1046                TAD LORDER
7564    4774                JMS I DGPT
7565    5722                JMP I FEXC

7566    7773    C253,       0253-260
7567    0002    C255,       255-253
7570    7634    M144,       7634
7571    0144    C144,       0144
7572    7766    M12,        7766
7573    0012    C12,        0012
7574    7352    DGPT,       OUTDG
```

BEXP	7324		HORDER	0045
CARRTN	7341		HTEMP	7062
CHAR	0057		INPUT	7142
CHE	7343		LFED	6777
C.10	7372		LNFEED	7342
C12	7573		LORDER	0046
C144	7571		LTEMP	7061
C253	7566		MASK	7141
C255	7567		MINCR	7171
C260	7356		MINUS	7135
C27	7511		MINUSE	7510
C3777	7515		MINUS7	7325
DECON	7017		MIN9	7137
DECONV	7000		MRBOUT	7170
DGPT	7574		MSGNPT	7520
DIGIT	7063		MSIGN	7120
DIVTWO	7357		MSNPT	7332
DNUMBR	7066		MULT10	7041
DPCSPT	7517		MULT2	7067
DPCVPT	7516		M10PT	7335
DPN	7521		M12	7572
DPT	7334		M144	7570
DSWIT	0060		M2PT	7336
DUBLAD	7103		OPUT	6776
ENDFI	7452		OUT	7344
EXCESS	7477		OUTDG	7352
EXPONT	0044		OUTPUT	7166
FEXC	7522		PC.10	7507
FEXPPT	7337		PER	7512
FGO1	7242		PERIOD	7331
FGO2	7246		PLUS	7136
FGO3	7263		PLUS12	7140
FGO4	7270		PRCHAR	6767
FGO5A	7274		PRINT	7165
FGO6	7301		PRSW	7513
FGO6A	7304		REMAIN	7064
FGO7	7311		RESTRT	7167
FIGO1	7420		SEXP	7514
FIGO2	7422		SIGN	7065
FIGO3	7426		SMINUS	7330
FIGO4	7470		SNPT	7333
FLINTP	7400		SPLUS	7327
FLOUTP	7200		SWIT1	0055
FOUR	7326		SWIT2	0056
FOUTCN	7217		TEN	7504
FPAC1	0052		TENPT	7340
FPNT	0007			

/FLOATING POINT PACKAGE

GETSGN=TAD 45
CLFPNT=JMS I 7

/FLOATING POINT EXPONENTIAL

```
                *5000
5000    0000    FEXP,       0
5001    1045                GETSGN
5002    7710                SPA CLA
5003    4310                JMS FNEG
5004    3304                DCA SIGN        /C(SIGN)=-1 IF X<0
5005    4407                CLFPNT
5006    3273                FMPY LG2E
5007    6315                FPUT X
5010    0000                FEXT
5011    4706                JMS I FIXIT
5012    1045                GETSGN
5013    3305                DCA FLAG2
5014    4707                JMS I FLOAT
5015    4407                CLFPNT
5016    6320                FPUT XSQR
5017    5315                FGET X
5020    2320                FSUB XSQR
5021    6315                FPUT X
5022    3315                FMPY X
5023    6320                FPUT XSQR
5024    1270                FADD D
5025    6323                FPUT TEMP
5026    5265                FGET C
5027    4323                FDIV TEMP
5030    2315                FSUB X
5031    1257                FADD A
5032    6323                FPUT TEMP
5033    5262                FGET B
5034    3320                FMPY XSQR
5035    1323                FADD TEMP
5036    6323                FPUT TEMP
5037    5315                FGET X
5040    4323                FDIV TEMP
5041    3301                FMPY TWO
5042    1276                FADD ONE
5043    0000                FEXT
5044    1305                TAD FLAG2
5045    1044                TAD 44
5046    3044                DCA 44
5047    2304                ISZ SIGN
5050    5600                JMP I FEXP
5051    4407                CLFPNT
5052    6315                FPUT X
5053    5276                FGET ONE
5054    4315                FDIV X
5055    0000                FEXT
5056    5600                JMP I FEXP
                /CONSTANTS FOR FEXP

5057    0004    A,          0004
5060    2372                2372
5061    1402                1402
```

```
5062   7774   B,        7774
5063   2157             2157
5064   5157             5157
5065   0012   C,        0012
5066   5454             5454
5067   0343             0343
5070   0007   D,        0007
5071   2566             2566
5072   5341             5341
5073   0001   LG2E,     0001
5074   2705             2705
5075   2435             2435
5076   0001   ONE,      0001
5077   2000             2000
5100   0000             0000
5101   0002   TWO,      0002
5102   2000             2000
5103   0000             0000
5104   0000   SIGN,     0
5105   0000   FLAG2,    0
5106   4757   FIXIT,    FIX
5107   5563   FLOAT,    FLOA
              /NEGATION SUBROUTINE
5110   0000   FNEG,     0
5111   4714             JMS I ACMINS      /CALL SUBROUTINE IN
5112   7240             CLA CMA           /INTERPRETER
5113   5710             JMP I FNEG
5114   6000   ACMINS,   6000              /POINTER TO INTERPRETER

              /TEMPORARY STORAGE
5115   0000   X,        0
5116   0000             0
5117   0000             0
5120   0000   XSQR,     0
5121   0000             0
5122   0000             0
5123   0000   TEMP,     0
5124   0000             0
5125   0000             0
5126   0000   ZER,      0
5127   0000             0
5130   0000             0
              /MAIN ALGORITHM FOR ARCTANGENT
5131   4407   ARCALG,   CLFPNT
5132   5315             FGET X
5133   3315             FMPY X
5134   6320             FPUT XSQR
5135   3373             FMPY BET2
5136   1370             FADD BET1
5137   3320             FMPY XSQR
5140   1365             FADD BETZ
5141   6323             FPUT TEMP
5142   5362             FGET ALF2
5143   3320             FMPY XSQR
5144   1357             FADD ALF1
5145   3320             FMPY XSQR
5146   1354             FADD ALFZ
5147   3315             FMPY X
5150   4323             FDIV TEMP
5151   0000             FEXT
5152   5753             JMP I .+1
5153   5226             ARCRTN
```

```
                    /CONSTANTS - FLOATING ARC TANGENT
5154    0000    ALFZ,       0000
5155    2437                2437
5156    1643                1643
5157    7777    ALF1,       7777
5160    3304                3304
5161    4434                4434
5162    7773    ALF2,       7773
5163    3306                3306
5164    5454                5454
5165    0000    BETZ,       0000
5166    2437                2437
5167    1646                1646
5170    0000    BET1,       0000
5171    2427                2427
5172    2323                2323
5173    7775    BET2,       7775
5174    3427                3427
5175    7052                7052
                    /FLOATING POINT ARC TANGENT
                    *5200
5200    0000    ARTN,       0
5201    1045                GETSGN
5202    7710                SPA CLA
5203    4642                JMS I NEGIT
5204    3243                DCA FLAG1
5205    4407                CLFPNT
5206    6644                FPUT I X1
5207    5644                FGET I X1
5210    2645                FSUB I CON1
5211    0000                FEXT
5212    1045                GETSGN
5213    7710                SPA CLA
5214    5223                JMP GO
5215    4407                CLFPNT
5216    5645                FGET I CON1
5217    4644                FDIV I X1
5220    6644                FPUT I X1
5221    0000                FEXT
5222    7240                CLA CMA
5223    3251    GO,         DCA FLOG         /FLAG
5224    5625                JMP I .+1        /CALL ALGORITHM
5225    5131                ARCALG
5226    2251    ARCRTN,     ISZ FLOG         /RETURN HERE
5227    5235                JMP EXIT
5230    4407                CLFPNT
5231    6644                FPUT I X1
5232    5246                FGET PIOT
5233    2644                FSUB I X1
5234    0000                FEXT
5235    2243    EXIT,       ISZ FLAG1
5236    5600                JMP I ARTN
5237    4642                JMS I NEGIT
5240    7200                CLA
5241    5600                JMP I ARTN
                    /CONSTANTS FOR ARCTANGENT

5242    5110    NEGIT,      FNEG
5243    0000    FLAG1,      0
5244    5115    X1,         X
5245    5076    CON1,       ONE
```

```
5246   0001   PIOT,      0001
5247   3110              3110
5250   3755              3755
              /FLOATING LOGARITHM

5251   0000   FLOG,      0
5252   1045              GETSGN
5253   7440              SZA
5254   5261              JMP OK
5255   4407              CLFPNT              /INDICATE ERROR
5256   4775              FDIV I ZERO         /DIVIDE BY ZERO
5257   0000              FEXT
5260   5651              JMP I FLOG
5261   7710   OK,        SPA CLA
5262   4642              JMS I NEGIT
5263   4407              CLFPNT
5264   6774              FPUT I TEM
5265   2645              FSUB I CON1
5266   0000              FEXT
5267   1045              GETSGN
5270   7450              SNA
5271   5350              JMP ZERGO
5272   7710              SPA CLA
5273   5341              JMP INVERT
5274   3243   START,     DCA FLAG1
5275   7040              CMA
5276   1774              TAD I TEM
5277   3045              DCA 45
5300   4777              JMS I FLOATP
5301   4407              CLFPNT
5302   3776              FMPY I LOG2
5303   6644              FPUT I X1
5304   0000              FEXT
5305   7001              IAC
5306   3774              DCA I TEM
5307   4407              CLFPNT
5310   5774              FGET I TEM
5311   2645              FSUB I CON1
5312   6774              FPUT I TEM
5313   3773              FMPY I L8
5314   1772              FADD I L7
5315   3774              FMPY I TEM
5316   1771              FADD I L6
5317   3774              FMPY I TEM
5320   1770              FADD I L5
5321   3774              FMPY I TEM
5322   1365              FADD L4
5323   3774              FMPY I TEM
5324   1362              FADD L3
5325   3774              FMPY I TEM
5326   1357              FADD L2
5327   3774              FMPY I TEM
5330   1354              FADD L1
5331   3774              FMPY I TEM
5332   1644              FADD I X1
5333   0000              FEXT
5334   2243              ISZ FLAG1
5335   5651              JMP I FLOG
5336   4642              JMS I NEGIT
5337   7200              CLA
5340   5651              JMP I FLOG
```

```
5341   4407   INVERT,   CLFPNT
5342   5645             FGET I CON1
5343   4774             FDIV I TEM
5344   6774             FPUT I TEM
5345   0000             FEXT
5346   7240             CLA CMA
5347   5274             JMP START
5350   4407   ZERGO,    CLFPNT
5351   5775             FGET I ZERO
5352   0000             FEXT
5353   5651             JMP I FLOG
5354   0000   L1,       0000
5355   3777             3777
5356   7742             7742
5357   7777   L2,       7777
5360   4000             4000
5361   4100             4100
5362   7777   L3,       7777
5363   2517             2517
5364   0307             0307
5365   7776   L4,       7776
5366   4113             4113
5367   7211             7211
5370   5547   L5,       LOG5
5371   5552   L6,       LOG6
5372   5555   L7,       LOG7
5373   5560   L8,       LOG8
5374   5123   TEM,      TEMP
5375   5126   ZERO,     ZER
5376   5544   LOG2,     LOGE2
5377   5563   FLOATP,   FLOA
                        /FLOATING POINT SINE
                        *5400
5400   0000   FSIN,     0
5401   1045             GETSGN              /X>0?
5402   7740             SMA SZA CLA
5403   5210             JMP MOD             /YES
5404   1045             GETSGN
5405   7700             SMA CLA             /NO   X=0?
5406   5600             JMP I FSIN          /YES SIN(0)=0
5407   4741             JMS I NEGT          /NO:SIN(-X)=-SIN(X)
5410   3343   MOD,      DCA PNTR            /REDUCE X MODULO 2 PI
5411   4407             CLFPNT
5412   4315             FDIV TWOPI
5413   6724             FPUT I XSQ2
5414   0000             FEXT
5415   4742             JMS I FIXR
5416   4363             JMS FLOA
5417   4407             CLFPNT
5420   6723             FPUT I X2
5421   5724             FGET I XSQ2
5422   2723             FSUB I X2
5423   3315             FMPY TWOPI
5424   6723             FPUT I X2
5425   2320             FSUB PI             /X<PI?
5426   0000             FEXT
5427   1045             GETSGN
5430   7710             SPA CLA
5431   5241             JMP PCHECK          /YES
5432   4407             CLFPNT              /NO SIN(X-PI)=-SIN(X)
5433   6723             FPUT I X2
5434   0000             FEXT
```

```
5435   1343              TAD PNTR
5436   7650              SNA CLA
5437   7040              CMA
5440   3343              DCA PNTR
5441   4407     PCHECK,  CLFPNT              /X<PI/2?
5442   5723              FGET I X2
5443   2714              FSUB I PI2
5444   0000              FEXT
5445   1045              GETSGN
5446   7710              SPA CLA
5447   5255              JMP PALG            /YES
5450   4407              CLFPNT              /NO
5451   5320              FGET PI             /SIN(X)=SIN(PI-X)
5452   2723              FSUB I X2
5453   6723              FPUT I X2
5454   0000              FEXT

5455   4407     PALG,    CLFPNT
5456   5723              FGET I X2
5457   4714              FDIV I PI2
5460   6723              FPUT I X2
5461   3723              FMPY I X2
5462   6724              FPUT I XSQ2
5463   5325              FGET C9
5464   3724              FMPY I XSQ2
5465   1330              FADD C7
5466   3724              FMPY I XSQ2
5467   1333              FADD C5
5470   3724              FMPY I XSQ2
5471   1336              FADD C3
5472   3724              FMPY I XSQ2
5473   1714              FADD I PI2
5474   3723              FMPY I X2
5475   0000              FEXT
5476   2343              ISZ PNTR
5477   5600              JMP I FSIN
5500   4741              JMS I NEGT
5501   7200              CLA
5502   5600              JMP I FSIN
                /FLOATING POINT COSINE

5503   0000     FCOS,    0
5504   4407              CLFPNT              /COS(X)=SIN(PI/2-X)
5505   6723              FPUT I X2
5506   5714              FGET I PI2
5507   2723              FSUB I X2
5510   0000              FEXT
5511   1303              TAD FCOS
5512   3200              DCA FSIN
5513   5201              JMP FSIN+1

                /CONSTANTS AND POINTERS

5514   5246     PI2,     PIOT                /PI/2
5515   0003     TWOPI,   0003
5516   3110              3110
5517   3755              3755
5520   0002     PI,      0002
5521   3110              3110
5522   3755              3755
5523   5115     X2,      X
5524   5120     XSQ2,    XSQR
```

4-28

```
                    /SINE CONSTANTS
5525    7764    C9,         7764
5526    2366                2366
5527    5735                5735
5530    7771    C7,         7771
5531    5466                5466
5532    6317                6317
5533    7775    C5,         7775
5534    2431                2431
5535    5053                5053
5536    0000    C3,         0000
5537    5325                5325
5540    0420                0420
5541    5110    NEGT,       FNEG
5542    4757    FIXR,       FIX
5543    0000    PNTR,       0

                    /LOGARITHM CONSTANTS

5544    0000    LOGE2,      0000
5545    2613                2613
5546    4414                4414
5547    7776    LOG5,       7776
5550    2535                2535
5551    3301                3301
5552    7775    LOG6,       7775
5553    4746                4746
5554    0771                0771
5555    7774    LOG7,       7774
5556    2236                2236
5557    4304                4304
5560    7771    LOG8,       7771
5561    4544                4544
5562    1735                1735
                    /FLOAT C(45)
5563    0000    FLOA,       0
5564    7300                CLA CLL
5565    3046                DCA 46
5566    1374                TAD C13
5567    3044                DCA 44
5570    4407                CLFPNT
5571    7000                FNOR
5572    0000                FEXT
5573    5763                JMP I FLOA
5574    0013    C13,        0013
                *4757
                    /FIX C(FAC)
4757    0000    FIX,        0
4760    1044                TAD 44
4761    7540                SMA SZA
4762    5365                JMP .+3
4763    7200                CLA
4764    5375                JMP FIXEND
4765    1377                TAD M13
4766    3044                DCA 44
4767    1044    LOOP,       TAD 44
4770    7700                SMA CLA
4771    5757                JMP I FIX
4772    4774                JMS I .+2
4773    5367                JMP LOOP
4774    6200                6200            /DIV1 IN INTERPRETER
```

```
4775   3045   FIXEND,     DCA 45
4776   5757               JMP I FIX
4777   7765   M13,        -13
```

A	5057
ACMINS	5114
ALFZ	5154
ALF1	5157
ALF2	5162
ARCALG	5131
ARCRTN	5226
ARTN	5200
B	5062
BETZ	5165
BET1	5170
BET2	5173
C	5065
CLFPNT	4407
CON1	5245
C13	5574
C3	5536
C5	5533
C7	5530
C9	5525
D	5070
EXIT	5235
FCOS	5503
FEXP	5000
FIX	4757
FIXEND	4775
FIXIT	5106
FIXR	5542
FLAG1	5243
FLAG2	5105
FLOA	5563
FLOAT	5107
FLOATP	5377
FLOG	5251
FNEG	5110
FSIN	5400
GETSGN	1045
GO	5223
INVERT	5341
LG2E	5073
LOGE2	5544

LOG2	5376
LOG5	5547
LOG6	5552
LOG7	5555
LOG8	5560
LOOP	4767
L1	5354
L2	5357
L3	5362
L4	5365
L5	5370
L6	5371
L7	5372
L8	5373
MOD	5410
M13	4777
NEGIT	5242
NEGT	5541
OK	5261
ONE	5076
PALG	5455
PCHECK	5441
PI	5520
PIOT	5246
PI2	5514
PNTR	5543
SIGN	5104
START	5274
TEM	5374
TEMP	5123
TWO	5101
TWOPI	5515
X	5115
XSQR	5120
XSQ2	5524
X1	5244
X2	5523
ZER	5126
ZERGO	5350
ZERO	5375

```
/FLOATING OUTPUT PROGRAM
/IF(62)=0, THEN OUTPUT IN E FORMAT
/ELSE, C(62)=NUMBER OF DIGITS
/C(AC)=NUMBER OF PLACES TO RIGHT OF .
/IF C(AC)=0, THEN DON'T OUTPUT POINT
/SIGN AND . NOT COUNTED IN OUTPUT
/CONTENTS OF 15 LOST DURING OPERATION

                    *7201
7201    4777        JMS I 7377 /ALTERATIONS FLOATING OUTPUT
7202    4217        JMS 7217
7203    1324        TAD 7324
7204    4776        JMS I 7376

                    *7207
7207    7200        CLA


                    *7227
7227    3415        DCA I 15
7230    7000        NOP
7231    7000        NOP
7232    7000        NOP

                    *7301
7301    3415        DCA I 15

                    *7305
7305    3415        DCA I 15

                    *7376
7376    5412        TGO
7377    5400        EDIT
                    *5400
5400    0000  EDIT,     0
5401    3352            DCA SAC
5402    1062            TAD 62
5403    7041            CMA IAC
5404    3347            DCA COUNT1
5405    1357            TAD M8
5406    3350            DCA COUNT2
5407    1363            TAD SAD1
5410    3015            DCA 15
5411    5600            JMP I EDIT
5412    0000  TGO,      0
5413    3044            DCA 44
5414    1363            TAD SAD1
5415    3015            DCA 15
5416    1415            TAD I 15
5417    4761            JMS I OUT1
5420    1062            TAD 62
5421    7650            SNA CLA
5422    5327            JMP EFORM
5423    2212            ISZ TGO
5424    2212            ISZ TGO
5425    1044  TRYAGN,   TAD 44
5426    7510            SPA
5427    5300            JMP MINS
5430    1352            TAD SAC
5431    7041            CMA IAC
5432    1062            TAD 62
```

```
5433    7510            SPA
5434    5266            JMP ERR
5435    7450            SNA
5436    5245            JMP GO1
5437    7041            CMA IAC
5440    3351            DCA CNTR
5441    1353            TAD SPCE
5442    4321            JMS OUT
5443    2351            ISZ CNTR
5444    5241            JMP .-3

5445    1044    GO1,    TAD 44
5446    7041            CMA IAC
5447    7450            SNA
5450    5256            JMP GO2
5451    3351            DCA CNTR
5452    4337            JMS GET
5453    4321            JMS OUT
5454    2351            ISZ CNTR
5455    5252            JMP .-3

5456    1352    GO2,    TAD SAC
5457    7650            SNA CLA
5460    5263            JMP .+3
5461    1354            TAD PERIOD
5462    4761            JMS I OUT1
5463    4337            JMS GET
5464    4321            JMS OUT
5465    5263            JMP .-2
5466    1352    ERR,    TAD SAC
5467    7700            SMA CLA
5470    5274            JMP ERGO
5471    1356            TAD CHX
5472    4321            JMS OUT
5473    5271            JMP .-2

5474    7240    ERGO,   CLA CMA
5475    1352            TAD SAC
5476    3352            DCA SAC
5477    5225            JMP TRYAGN

5500    7200    MINS,   CLA
5501    1062            TAD 62
5502    7041            CMA IAC
5503    1352            TAD SAC
5504    7450            SNA
5505    5313            JMP GO3
5506    3351            DCA CNTR
5507    1353            TAD SPCE
5510    4321            JMS OUT
5511    2351            ISZ CNTR
5512    5307            JMP .-3

5513    1354    GO3,    TAD PERIOD
5514    4761            JMS I OUT1
5515    4321            JMS OUT
5516    2044            ISZ 44
5517    5315            JMP .-2
5520    5263            JMP GO2+5
5521    0000    OUT,    0
5522    4762            JMS I OUT2
```

```
5523    2347                ISZ COUNT1
5524    5721                JMP I OUT
5525    1355                TAD CHE
5526    5612                JMP I TGO

5527    4762    EFORM,      JMS I OUT2
5530    1354                TAD PERIOD
5531    4761                JMS I OUT1
5532    1360                TAD M7
5533    3347                DCA COUNT1
5534    4337                JMS GET
5535    4321                JMS OUT
5536    5334                JMP .-2

5537    0000    GET,        0
5540    2350                ISZ COUNT2
5541    5345                JMP .+4
5542    7240                CLA CMA
5543    3350                DCA COUNT2
5544    5737                JMP I GET
5545    1415                TAD I 15
5546    5737                JMP I GET
5547    0000    COUNT1,     0
5550    0000    COUNT2,     0
5551    0000    CNTR,       0
5552    0000    SAC,        0
5553    7760    SPCE,       240-260
5554    0256    PERIOD,     256
5555    0305    CHE,        305
5556    0050    CHX,        330-260
5557    7770    M8,         -10
5560    7771    M7,         -7
5561    7344    OUT1,       7344
5562    7352    OUT2,       7352
5563    5563    SAD1,       BUFFER-1
                BUFFER,
BUFFER    5564
CHE       5555
CHX       5556
CNTR      5551
COUNT1    5547
COUNT2    5550
EDIT      5400
EFORM     5527
ERGO      5474
ERR       5466
GET       5537
GO1       5445
GO2       5456
GO3       5513
MINS      5500
M7        5560
M8        5557
OUT       5521
OUT1      5561
OUT2      5562
PERIOD    5554
SAC       5552
SAD1      5563
SPCE      5553
TGO       5412
TRYAGN    5425
```

INDEX

Digital Equipment Corporation maintains a continuous effort to improve the quality and usefulness of its publications. To do this effectively we need user feedback — your critical evaluation of this manual.

Please comment on this manual's completeness, accuracy, organization, usability, and readability.

_____

_____

_____

_____

_____

Did you find errors in this manual? _____

_____

_____

_____

_____

How can this manual be improved? _____

_____

_____

_____

_____

_____

DEC also strives to keep its customers informed of current DEC software and publications. Thus, the following periodically distributed publications are available upon request. Please check the appropriate boxes for a current issue of the publication(s) desired.

☐ Software Manual Update, a quarterly collection of revisions to current software manuals.

☐ User's Bookshelf, a bibliography of current software manuals.

☐ Program Library Price List, a list of currently available software programs and manuals.

Please describe your position. _____

Name _____   Organization _____

Street _____   Department _____

City_____ State _____ Zip or Country _____

- - - - - - - - - - - - - - - - Fold Here -

- - - - - - - - - - - - - Do Not Tear - Fold Here and Staple - - - - - - - - - - -
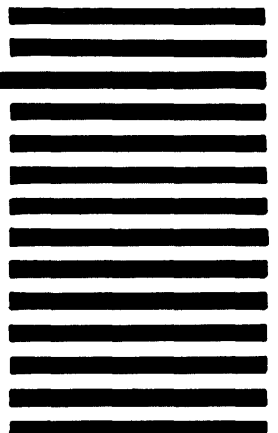
HOW TO OBTAIN SOFTWARE INFORMATION

Announcements for new and revised software, as well as programming notes, software problems, and documentation corrections are published by Software Information Service in the following newsletters.

Digital Software News for the PDP-8 Family
Digital Software News for the PDP-9/15 Family
PDP-6/PDP-10 Software Bulletin

These newsletters contain information applicable to software available from Digital's Program Library.

Please complete the card below to place your name on the newsletter mailing list.

Questions or problems concerning DEC Software should be reported to the Software Specialist at your nearest DEC regional or district sales office. In cases where no Software Specialist is available, please send a Software Trouble Report form with details of the problem to:

Software Information Service
Digital Equipment Corporation
146 Main Street, Bldg. 3-5
Maynard, Massachusetts 01754

These forms, which are available without charge from the Program Library, should be fully filled out and accompanied by teletype output as well as listings or tapes of the user program to facilitate a complete investigation. An answer will be sent to the individual and appropriate topics of general interest will be printed in the newsletter.

New and revised software and manuals, Software Trouble Report forms, and cumulative Software Manual Updates are available from the Program Library. When ordering, include the document number and a brief description of the program or manual requested. Revisions of programs and documents will be announced in the newsletters and a price list will be included twice yearly. Direct all inquiries and requests to:

Program Library
Digital Equipment Corporation
146 Main Street, Bldg. 3-5
Maynard, Massachusetts 01754

Digital Equipment Computer Users Society (DECUS) maintains a user Library and publishes a catalog of programs as well as the DECUSCOPE magazine for its members and non-members who request it. For further information please write to:

DECUS
Digital Equipment Corporation
146 Main Street
Maynard, Massachusetts 01754

— — — — — — — — — — — — — — — — —

Send Digital's software newsletters to:

Name_____

Company Name_____

Address_____

_____

(zip code)

My computer is a

PDP-8/I ☐   PDP-8/L ☐
LINC-8 ☐   PDP-12 ☐
PDP-9 ☐   PDP-15 ☐   Please specify
PDP-10 ☐   OTHER ☐ _____

My system serial number is_____ (if known)

— — — — — — — — — — — — — — — — — Fold Here —

— — — — — — — — — — — — — Do Not Tear - Fold Here and Staple — — — — — — — — — — — — — — —