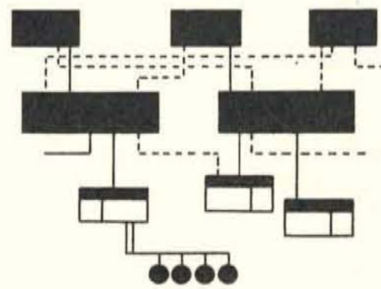
GENERAL ELECTRIC
COMPUTERS

WRITE-UPS OF MATH. PROGRAMMING SUB-ROUTINES

GE-625/635 FORTRAN IV Math Library

SYSTEM
SUPPORT
INFORMATION



See also CPB-1185 for other subroutine documentation

ABSTRACT

This manual describes FORTRAN IV Math Routines available for use with all configurations of the GE-625/635 computer systems.

GENERAL ELECTRIC

CPB-1083B

GE-625/635
FORTRAN IV MATH LIBRARY

September 1964

Rev. June 1966

GENERAL ELECTRIC

INFORMATION SYSTEMS DIVISION

(c) 1964, 1966 by General Electric Company

PREFACE

The FORTRAN IV Math Routines described in this manual are part of an integrated programming system available for the GE-625/635 computer system. The numbers assigned to the writeups are the same as those assigned to the actual programs which they explain. The numbering system is described on the following page.

As is true of all programs for the GE-625/635, The FORTRAN IV Math Library Routines are upward compatible. Any program described in this manual can be executed by any central processor in the GE-625/635 Series of computer systems.

The FORTRAN IV Math Library Manual is distributed in loose leaf form to facilitate the incorporation of additions and changes. As soon as new programs are completed, corresponding writeups will be made available to users. When changes become necessary, change pages will be distributed. Revised pages will be identified by the date at the top of the page, and revisions within pages will be identified by a bar in the margin beside the sentence or sentences changed.

Suggestions and criticisms relative to form, content, purpose, or use of this manual are invited. Comments may be sent on the Document Review Sheet in the back of this manual or may be addressed directly to Engineering Publications Standards, B-90, Computer Equipment Department, General Electric Company, 13430 North Black Canyon Highway, Phoenix, Arizona 85029.

NUMBERING SYSTEM

The FORTRAN IV Math Routines included in this publication are each assigned a number in accordance with a numbering system used for all 600-Series programming routines. For example, XP1--Exponential--Integer Base and Exponent is assigned the number CD600D2.001. This number is described to illustrate the numbering system.

CD600D2.001

The last three digits, which always follow a decimal point, make a sequential listing of the routines in the order they are made available to the Program Library. The sequence is within the classification of the number and letter to the left of the decimal point.

The digit before the decimal point makes a grouping of routine types within the alphabetic classification described in the following paragraph. The Math Routines are classified in eight categories:

1. Programmed Arithmetic
2. Elementary Functions
3. Statistical Routines
4. Operations on Matrices, Vectors and Simultaneous Equations
5. Polynomial and Special Functions
6. Curve Fitting and Other Approximations
7. Operations Research
8. Numerical Integration and Differentiation and Solutions of Differential Equations

The alphabetic letter in the center of the number classifies the routines according to the following list:

A. Diagnostic Routines
B. Service Routines
C. Internal Data Manipulation
D. Math Routines
E. Input/Output Routines
F. Assembly Systems
G. Generators
H. Compilers/Translators
I. Simulators
J. Service Systems
K. Special Systems

The 600 means that the programs are programmed for use on the GE-600 Series Computer Systems.

The CD means that the program was originated by the General Electric Computer Equipment Department.

GE-600 MATH LIBRARY PROGRAMMING ROUTINES

CD600-	NAME	DESCRIPTION
D1.001	FDMD	Double Precision Flt. Pt. Modulus
.002	FDXP	" " " " Exponential
.003	FDSQ	" " " " Square Root
.004	FDSC	" " " " Sine/Cosine
.005	FDAT	" " " " Arc Tangent
.006	FDLG	" " " " Logarithm
.007	FPRO	"False" Subroutines
.008	FCMP	Complex Multiplication & Division
.009	FCAB	" Absolute value
D2.001	FXP1	
.002	FXP2	
.003	FXP3	
.004	FDX1	
.005	FDX2	
.006	ALGT	
.007	ASIN	
.008	TANH	
.009	SINH	
.010	FDEX	
.011	FDSQ	
.012	FDSN	
.013	FDAT	
.014	FDLG	
.015	FEXP	
.016	FALG	
.017	FATN	
.018	FSIN	
.019	FTNH	
.020	FSQR	
.021	FCEX	
.022	FCLG	
.023	FCSQ	
.024	FCSN	

see listing book (margin note beside .007)

	CD600	NAME	DESCRIPTION
	D3.001	FXPF	Single Precision Floating Point natural Exponential
	.002	FLOG	" " " " Logarithm
	.003	FATN	" " " " Arc tangent
	.004	FSCN	" " " " Sine/Cosine
	.005	FTNH	" " " " Hyperbolic Tangent
	.006	FSQR	" " " " Square Root
See CPB - 1183	.007	BMD	Biomedical Statistical programs
	.008	RNGN	
	D4.001	FCAS	Complex Multiplication & Division
	.002	FCAB	" Absolute value
	.003	FCXP	" Exponential
	.004	FCLG	" Logarithm
	.005	FCSQ	" Square root
	.006	FCSC	" Sine/Cosine
	.007	MINV	Matrix Inversion
See CPB - 1167 (SIMEQ)	.008	SMEQ	Simultaneous Equations Solution
See CPB - 1166 (EIGEN)	.009	EIGP	Eigen vectors/values of a square matrix
	.010	MADD	Matrix Addition
	.011	MSUB	" Subtraction
	.012	MMPY	" Multiplication
	.013	MTRN	" Transposition
	.014	MMOV	" Moving
See CPB - 1152 (POLRTS)	D5.001	POLY	Roots of a Polynomial
	.002	BESSL	
See CPB - 1165 (LSPF)	D6.001	LSPF	LEAST SQUARES POLYNOMIAL CURVE FITTING
	.002	INTP	INTERPOLATION
See CPB - 1141	D7.001	LP/600	LINEAR PROGRAMMING
See CPB - 1168 (DIFFE)	D8.001	DIFE	DIFFERENTIAL EQUASION SOLUTION

GE 600 Mathematical Routines and Scientific Languages

1. SIMEQ - Solves system of linear equations in either real single or double precision. (CPB - 1167)

2. MINV - Matrix inversion routine (MATH LIBRARY: CD600D4.007)

3. POLY - Finds roots of a polynomial in either real single or double precision. (CPB - 1152)

4. LSPF - Least squares polynomial curve fit (CPB - 1165)

5. DIFFER - Solves first order differential equations (CPB - 1168)

6. EIGNP - Handles Eigenvalues & Eigenvectors (CPB - 1166)

7. LP - Linear Programming System which includes a comprehensive FORTRAN-like control language, matrix generator, report writer, transportation and decomposition algorithens, etc. (CPB - 1141; 1222; 1262; 1263; 1264; 1267)

8. SIMSCRIPT- Event oriented. Simulation Language. ~~Available first quarter of 1966~~. (CPB - 1218)

9. BMD - UCLA Statistics Package (CPB - 1183)

   a. Description & Tabulation - Data Editing Routines (11)

   b. Multi-Variate Analysis

      1) Factor analysis

      2) Discriminant Analysis for Two Groups

      3) " " " Several "

      4) Canonical "

      5) Principal Component Analysis

      6) Regression on Principal Components

c. Regression Analysis

1) Simple Linear Regression

2) Step-Wise "

3) Multiple Regression with Case Combinations

4) Periodic " and Harmonic Analysis

5) Polynomial "

6) Asymptotic "

d. Time Series Analysis

1) Amplitude & Phase Analysis

2) Autocovariance and Power Spectral Analysis

E. Variance Analysis

1) Analysis of Variance for One-Way Design

2) " " " " Factorial "

3) " " Covariance for " "

4) " " " with Multiple Covariance

5) General Linear Hypotheses

6) " " " with Contrasts

7) Multiple Range Tests

10. ALGOL - "Full" implementation of Algol-60 (CPB-1087)

11. PL/1 - New Programming Language - Version 1

(designed for both scientific and commercial problems). ~~Available first part of 1966.~~ NOT SCHEDULED FOR IMPLEMENTATION

12. JOVIAL - Jules Own Version of an International Algebraic Language. ~~Available first quarter 1966.~~ (CPB - 1187)

13. FORTRAN IV - Standard ASA FORTRAN IV and is compatible with IBM's FORTRAN IV. Mathematical routines include:

   a. FDMD - Double Precision Modulus

   b. FDXP - Double Precision Exponential

   c. FDSQ - Double Precision Square Root

   d. FDSC - Double Precision Sine and Cosine

   e. FDAT - Double Precision Arctangent

   f. FDLG - Double Precision Logarithm

   g. XP1 - Exponential Integer Base and Exponent

   h. XP2 - Exponential Floating Point Base, Integer Exponent

   i. XP3 - Exponential Real Base and Exponent

   j. FDX1 - Exponential Complex Base, Integer Exponent

   k. FDX2 - Exponential Double Precision Base and Exponent

   l. FXPF - Real Natural Exponential

   m. FLOG - Real Logarithm

   n. FATN - Real Arctangent

   o. FSCN - Real Sine and Cosine

p. FTNH - Real Hyperbolic Tangent

q. FSQR - Real Square Root

r. FCAS - Complex Multiplication and Division

s. FCAB - Complex Absolute Value

t. FCXP - Complex Exponential

u. FCLG - Complex Logarithm

v. FCSQ - Complex Square Root

w. FCSC - Complex Sine and Cosine

CONTENTS

FDMD--DOUBLE-PRECISION MODULUS

PURPOSE

To compute A = X (mod Y) for DMOD(X,Y) in an expression.

METHOD

1. If Y = 0, then A = X. Otherwise, compute
   Z = the greatest integer $\leq \frac{|X|}{|Y|}$ and give
   Z the same sign as that of $\frac{X}{Y}$. Then A = X - Y * Z.

2. A, X, and Y are double-precision numbers, with values from -2^{127} to $2^{127}-2^{64}$ inclusive.

3. A is accurate to 63 binary positions.

USAGE

1. Calling Sequence--CALL DMOD(X,Y)

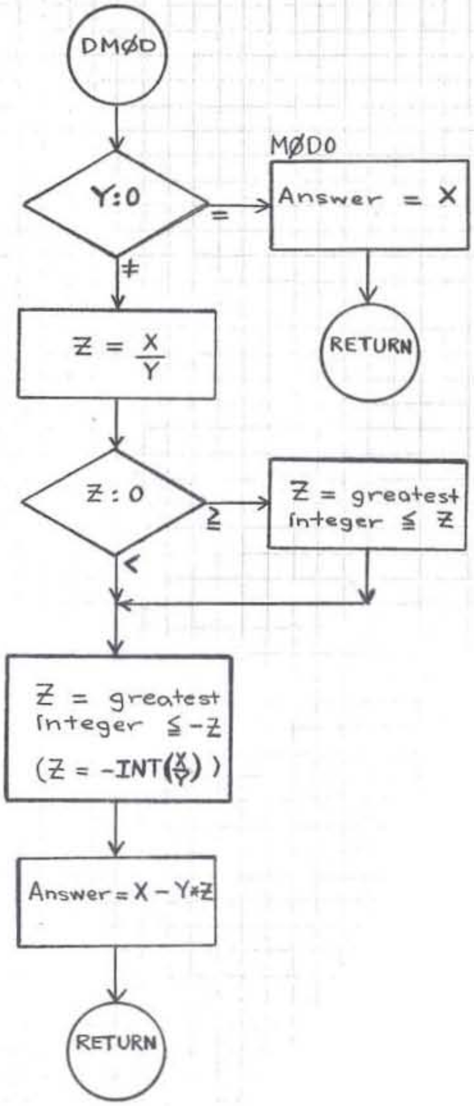
2. FDMD uses 16 words.

3. No error conditions.

RESTRICTIONS

None.

COMPUTE X (MOD Y) FOR DOUBLE PRECISION X AND Y



FDXP--DOUBLE-PRECISION EXPONENTIAL

PURPOSE

To compute e^X for EXP(X) in an expression.

METHOD

1. Use the same method as in FXPF--Real Natural Exponential, CD600D3.001, except that $2^F = 1 + F \log_e 2 + \frac{(F \log_e 2)^2}{\lfloor 2} + \ldots + \frac{(F \log_e 2)^{13}}{\lfloor 13}$

2. X and e^X are double-precision numbers, with $|X| \leq 88.028$

3. e^X is accurate to 16 decimal positions.

USAGE

1. Calling Sequence--CALL DEXP(X)

2. FDXP uses 68 words.

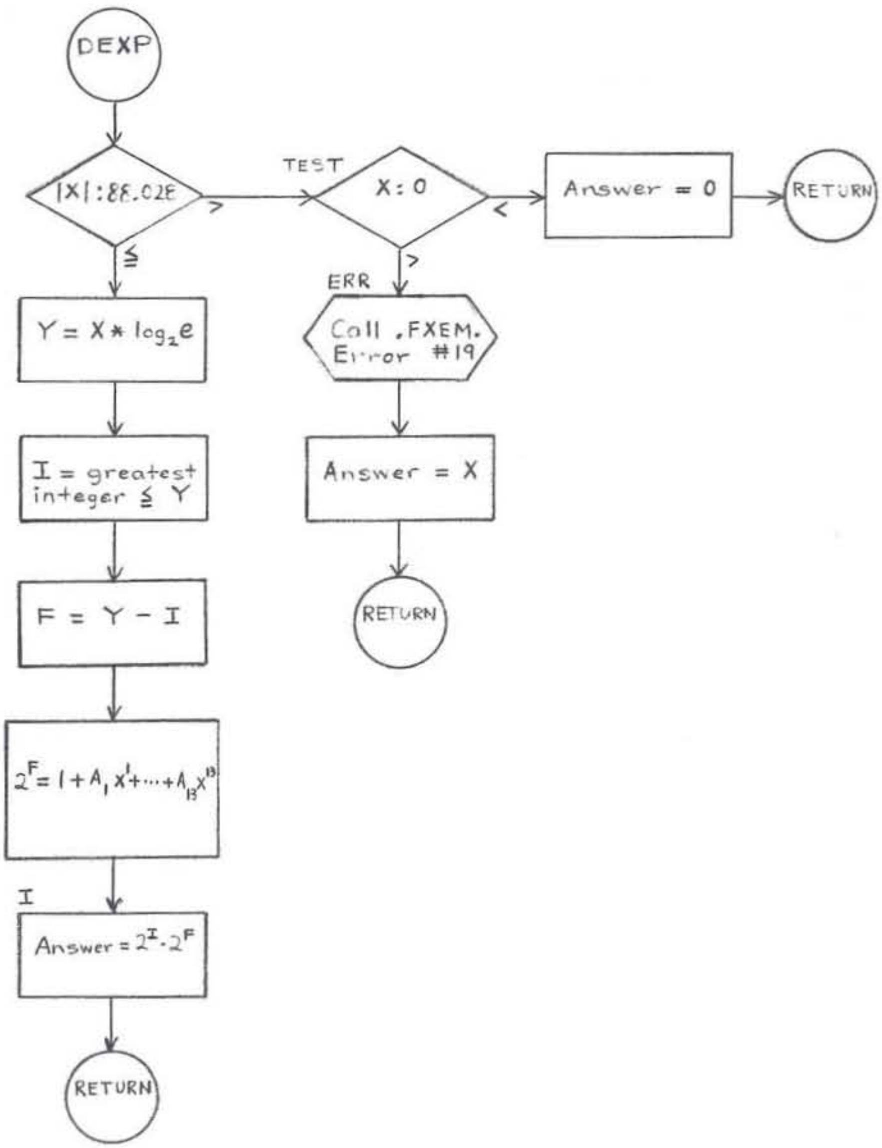
3. The error condition is:

   FXEM Error #19 if $|X| > 88.028$. Then $e^X = X$.

RESTRICTIONS

The subprogram FXEM must be in memory.

COMPUTE e^X FOR DOUBLE PRECISION X



FDSQ--DOUBLE-PRECISION SQUARE ROOT

PURPOSE

To compute $\sqrt{X}$ for DSQRT(X) in an expression.

METHOD

1. Use the same method as in FSQR--Real Square Root, CD600D3.006 except that $P_3 = \frac{1}{2} * (P_2 + \frac{F}{P_2})$ and $\sqrt{X} = 2^{A-1} * (P_3 + \frac{F}{P_3})$.
2. X and $\sqrt{X}$ are double-precision numbers, with values of X from 0 to $2^{127} - 2^{64}$ inclusive.
3. $\sqrt{X}$ is accurate to 18 decimal positions.

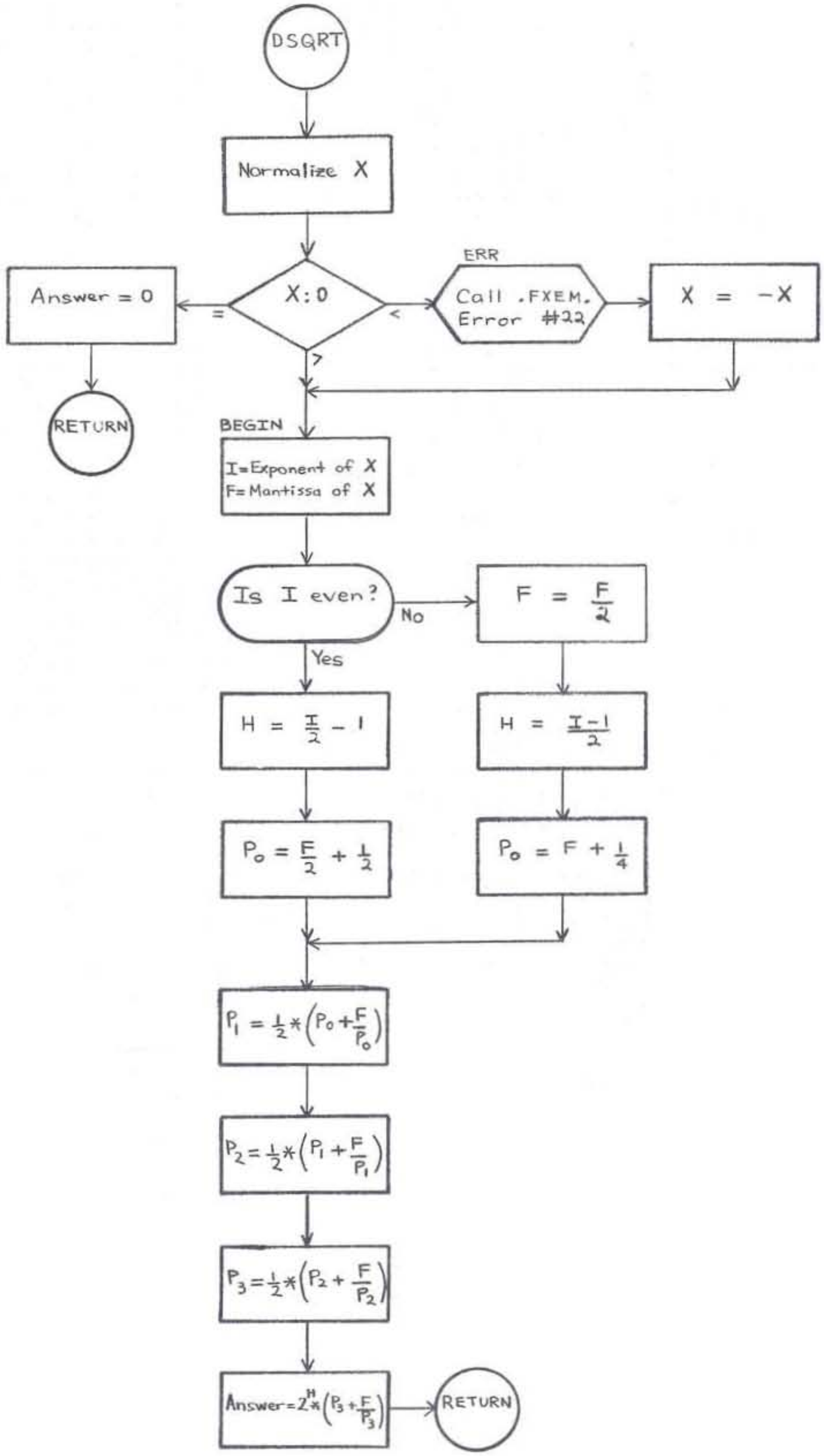
USAGE

1. Calling Sequence--CALL DSQRT(X)
2. FDSQ uses 50 words.
3. The error condition is:
   FXEM Error #22 if $X < 0$. Then $\sqrt{X} = \sqrt{|X|}$.

RESTRICTIONS

The subprogram FXEM must be in memory.

COMPUTE $\sqrt{X}$ FOR DOUBLE PRECISION X



FDSC--DOUBLE-PRECISION SINE AND COSINE

PURPOSE

To compute sin X or cos X for DSIN(X) or DCOS(X) in an expression, where X is in radians.

METHOD

1. Use the same method as in FSCN--Real Sine and Cosine, CD600D3.004, with the following exceptions:

   a. Do not make $X < \frac{1}{256}$ a special case. Use $\frac{\pi}{2}$ instead of 0.3 as the breakpoint.

   b. Use a Taylor Series approximation instead of a Continued Fraction:

   $\sin X = X - \frac{X^3}{\underline{|3}} + \frac{X^5}{\underline{|5}} - \ldots$ or $\cos X = 1 - \frac{X^2}{\underline{|2}} + \frac{X^4}{\underline{|4}} - \ldots$

   Include enough terms in the series until $\frac{X^n}{\underline{|n}} < \frac{\text{first term}}{10^{18}}$.

   (When $\frac{\text{first term}}{10^{18}} = 0$, include only the first term in the series.)

2. X, sin X, and cos X are double-precision numbers with $|X| < 2^{54}$.

3. The answer is accurate to 18 decimal positions.

USAGE

1. Calling Sequence--CALL DSIN(X) for sin X
   CALL DCOS(X) for cos X

2. DSCN uses 98 words.

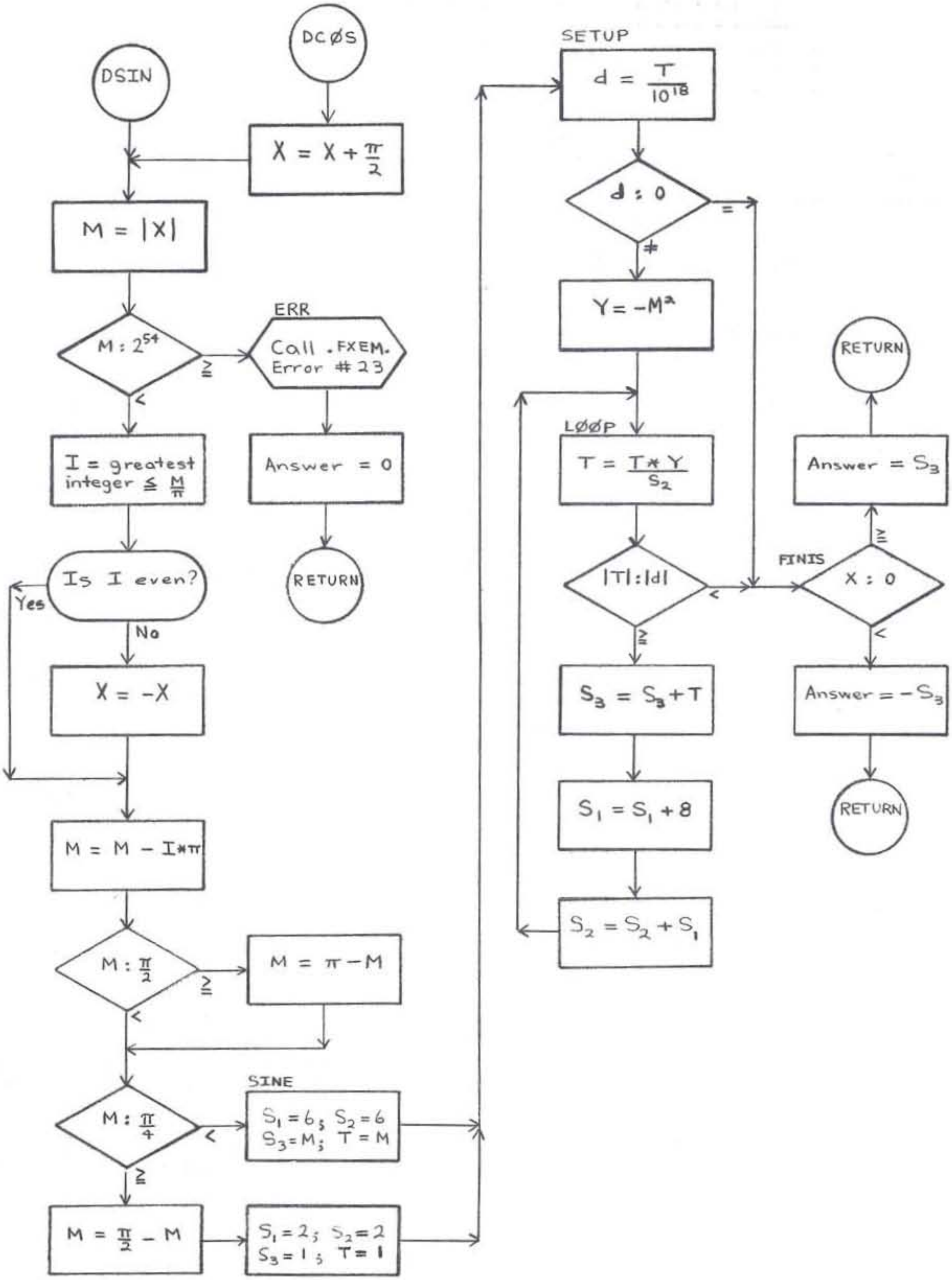
3. The error condition is:

   FXEM Error #23 if $|X| \geqq 2^{54}$. Then the answer is 0.

RESTRICTIONS

The subprogram FXEM must be in memory.

COMPUTE SIN X OR COS X FOR DOUBLE PRECISION X



FDAT--DOUBLE-PRECISION ARCTANGENT

PURPOSE

To compute the principal value of arctan X or arctan $\frac{Y}{Z}$ (in radians) for DATAN(X) or DATAN2 (Y,Z) in an expression.

METHOD

1. Use the same method as in FATN--Real Arctangent, CD600D3.003 with the following exceptions:

   a. The intervals are 0° - 7.5°, 7.5° - 22.5°, 22.5° - 37.5°, 37.5° -52.5°, 52.5° - 67.5°, and 67.5° - 82.5°. For 82.5° - 90°, compute $\frac{\pi}{2}$ - arctan $\frac{1}{X}$, where arctan $\frac{1}{X}$ is in the first interval.

   b. For 0° - 7.5°, $T = AL_6 * X$. Otherwise, $T = AL_I - \frac{BETA_I}{G_I + X}$.

   c. $\arctan X = N_I + \frac{C_{12} * T}{C_{12} * C_{14} - C_8}$, where $C_{14} = B + T^2$,

      $C = B_2 + T^2$, $C_2 = B_4 + T^2$, $C_4 = B_6 + T^2$, $C_6 = C_2 * C_4 - A_4$, $C_8 = A * C_6$, $C_{10} = C * C_6$, $C_{12} = C_{10} - A_2 * C_4$.

2. X, Y, and Z are double-precision numbers, with values from $-\left(2^{127}\right)$ to $\left(2^{127} - 2^{64}\right)$ inclusive. The answer is a double-precision number.

3. The answer is accurate to 16 decimal positions.

USAGE

1. Calling Sequence--CALL DATAN(X) for arctan X
   CALL DATAN2(Y,Z) for arctan $\frac{Y}{Z}$

2. FDTN uses 204 words.

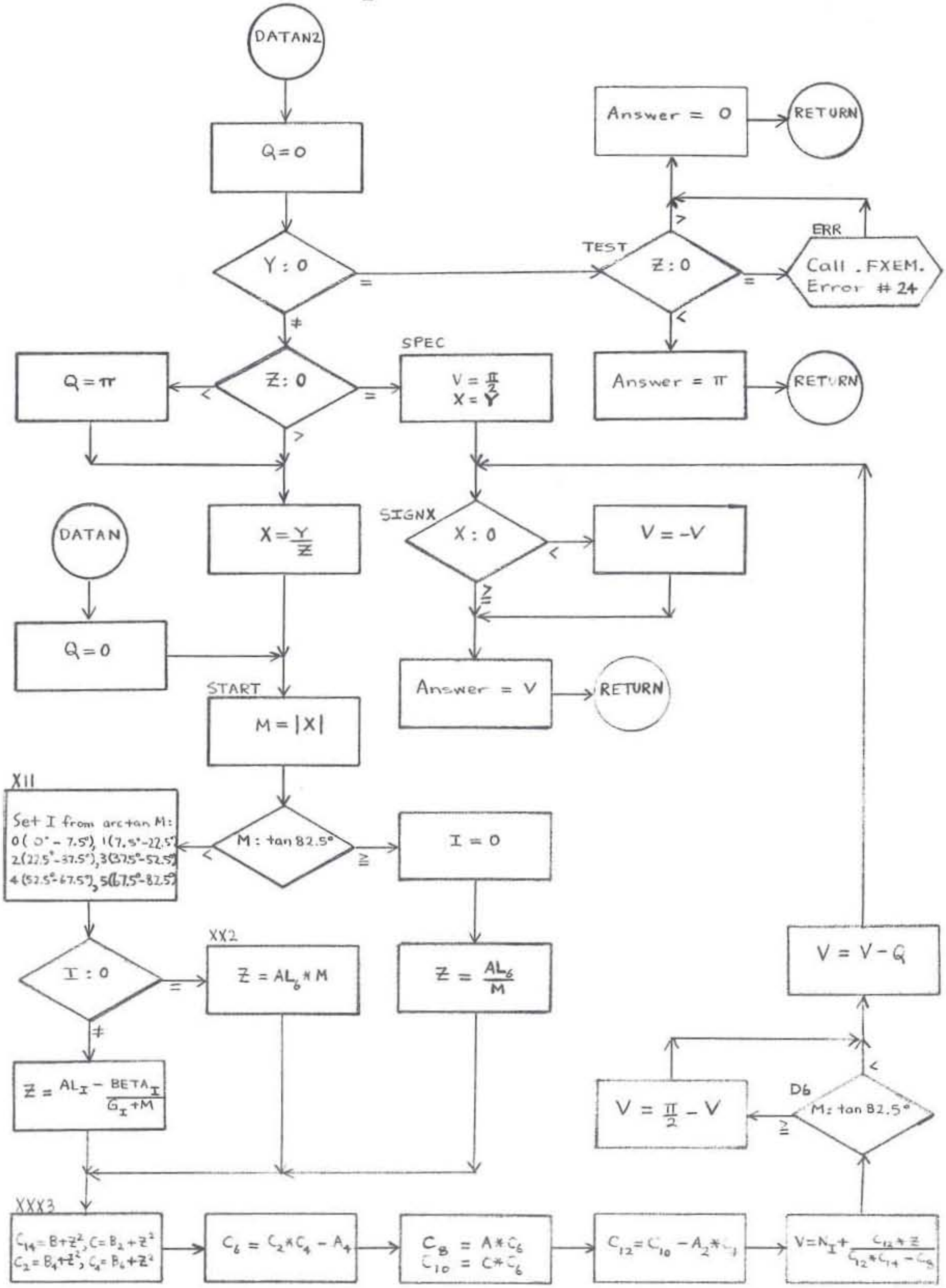
3. The error condition is:

   FXEM Error #24 if Y = 0 and Z = 0. Then arctan $\frac{Y}{Z} = 0$.

RESTRICTIONS

The subprogram FXEM must be in memory.

COMPUTE ARCTAN X OR ARCTAN $\frac{Y}{Z}$ FOR DOUBLE PRECISION X, Y, AND Z



FDLG--DOUBLE-PRECISION LOGARITHM

PURPOSE

To compute $\log_e X$ for DLOG(X) or $\log_{10} X$ for DLOG10(X) in an expression.

METHOD

1. $\log_2 X = \log_2 (2^I * F) = I + \log_2 F$, where $X = 2^I * F$.

2. $\log_e X = \log_e 2^{(\log_2 X)} = (\log_2 X) * (\log_e 2)$

$= I * \log_e 2 + (\log_2 F) * (\log_e 2)$

$= I * \log_e 2 + \log_e 2^{(\log_2 F)}$

$= I * \log_e 2 + \log_e F$

3. Let A = most significant 5 bits of F and let $Z = \frac{F - A}{F + A}$

Then $\log_e F = \log_e A + 2* \left(Z + \frac{Z^3}{3} + \ldots + \frac{Z^{11}}{11}\right)$

4. $\log_{10} X = (\log_e X) * (\log_{10} e)$

5. X and log X are double-precision numbers; values of X range from 2^{-129} to $2^{127}-2^{64}$ inclusive.

6. log X is accurate to 16 places.

USAGE

1. Calling Sequence--CALL DLOG(X) for $\log_e X$
   CALL DLOG10(X) for $\log_{10} X$

2. FDLG uses 120 words.

3. The error conditions are:

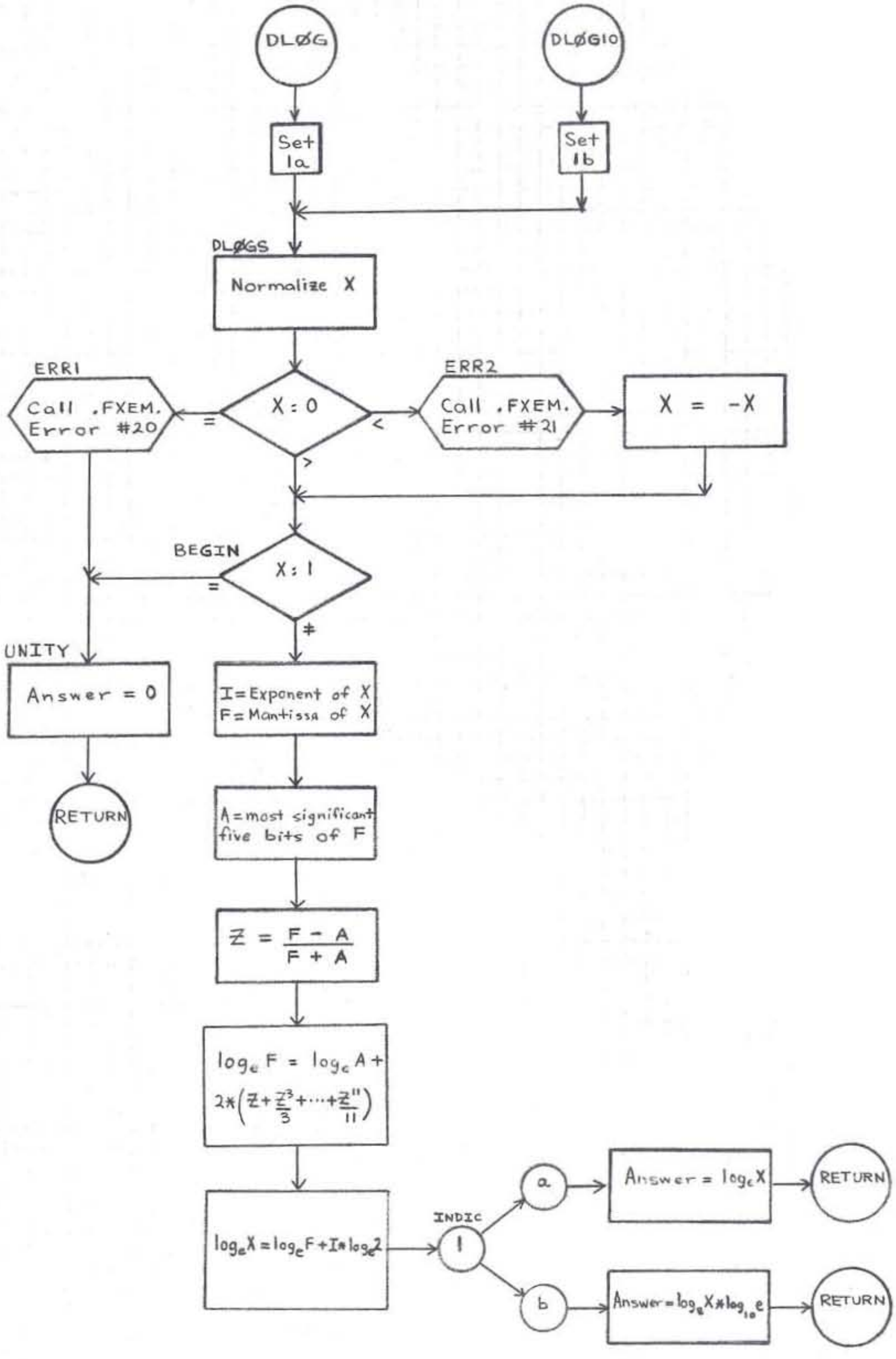
   a. FXEM Error #20 if X = 0. Then log X = 0.

   b. FXEM Error #21 if X < 0. Then log X = log |X| .

RESTRICTIONS

The subprogram FXEM must be in memory.

COMPUTE $\log_e X$ OR $\log_{10} X$ FOR DOUBLE PRECISION X



FCMP

FCAS--COMPLEX MULTIPLICATION AND DIVISION

I. PURPOSE

To compute (A, B) * (C, D) or $\frac{(A,\ B)}{(C,\ D)}$ in an expression.

II. METHOD

1. $(A,\ B) * (C,\ D) = (A * C - B * D,\ A * D + B * C)$

2. $\frac{(A,\ B)}{(C,\ D)} = \frac{(A,\ B) * (C,\ -D)}{C^2 + D^2} = \frac{(A * C + B * D,\ B * C - A * D)}{C^2 + D^2}$

3. If (A, B) = (0, 0), then the quotient = (0, 0). Otherwise, divide the numerator by A * C and the denominator by C^2: $\frac{(A,\ B)}{(C,\ D)} = \frac{\frac{A}{C}}{1 + \left(\frac{D}{C}\right)^2} * \left(1 + \frac{B}{A} * \frac{D}{C},\ \frac{B}{A} - \frac{D}{C}\right)$, where $\frac{A}{C} = \frac{A * C}{C^2}$.

4. Before computing $\frac{(A,\ B)}{(C,\ D)}$, replace the numerator by (-B, A) if $|A| \leq |B|$, and the denominator by (-D, C) if $|C| \leq |D|$. Adjust the quotient (X, Y) accordingly:

   a. If $|A| > |B|$ and $|C| > |D|$, then the result = (X, Y).
   b. If $|A| > |B|$ and $|C| \leq |D|$, then the result = (-Y, X).
   c. If $|A| \leq |B|$ and $|C| > |D|$, then the result = (Y, -X).
   d. If $|A| \leq |B|$ and $|C| \leq |D|$, then the result = (X, Y).

5. A, B, C, D, X, and Y are real numbers, with values from -2^{127} to $2^{127} - 2^{100}$ inclusive.

6. The answer is accurate to 8 decimal positions.

III. USAGE

1. Calling Sequence--CALL FCFMP. (R, S) for R * S
   CALL FCFDP. (R, S) for R/S
   where R = (A, B) and S = (C, D)

2. FCAS uses 94 words.

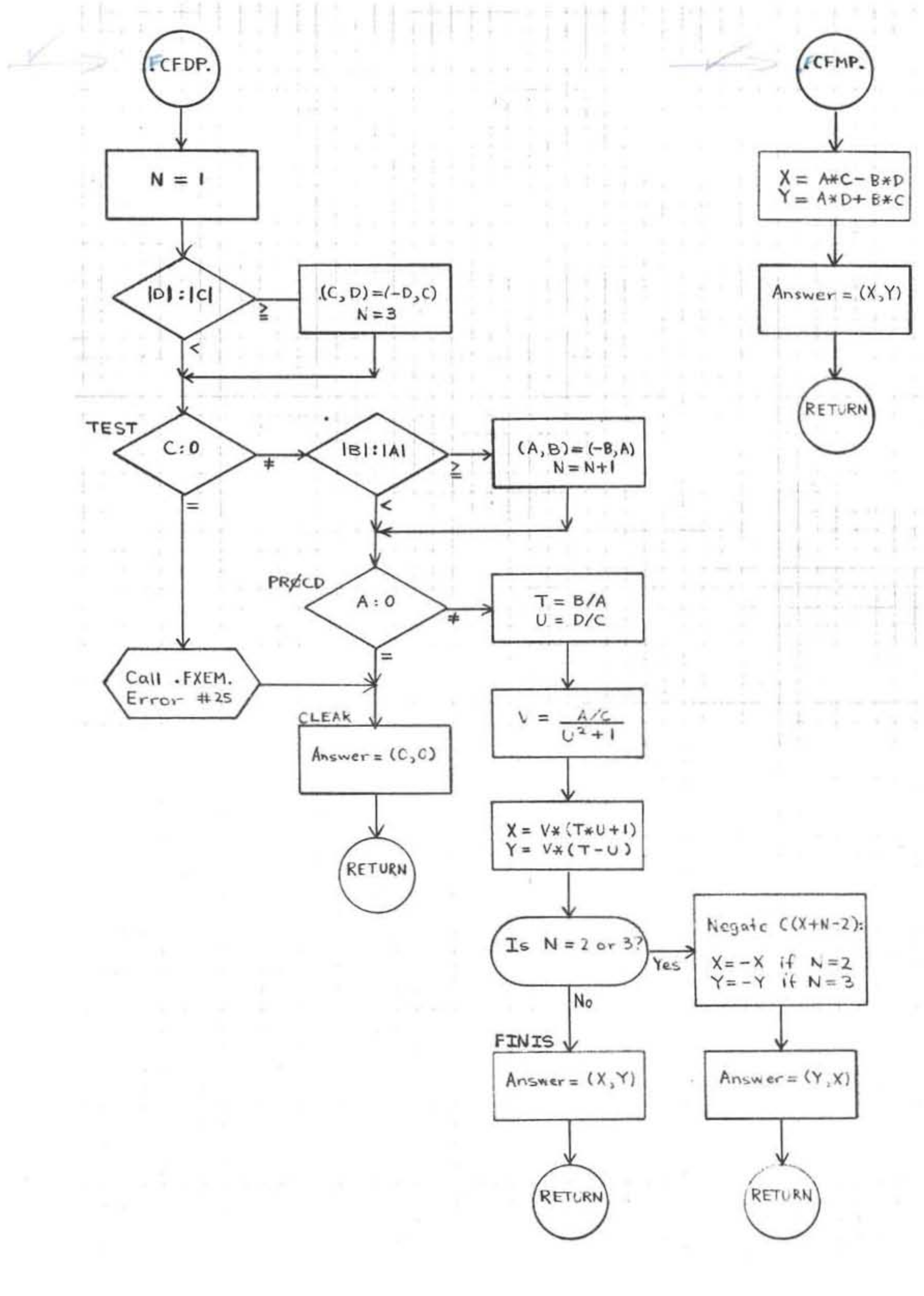
3. The error condition is only in division:

   FXEM Error #25 if (C, D) = (0, 0). Then $\frac{(A, B)}{(C, D)} = (0.0)$.

IV. RESTRICTIONS

The subprogram FXEM must be in memory.

COMPUTE (A, B) * (C, D) OR (A, B) / (C, D) FOR COMPLEX (A, B) AND (C, D)



FCAB--COMPLEX ABSOLUTE VALUE

I. PURPOSE

To compute $|Z|$ for CABS(Z) in an expression.

II. METHOD

1. Compute $W = \sqrt{X^2 + Y^2}$ (where $Z = (X, Y)$) as follows:
   a. If $Y = 0$, then $W = |X|$
   b. If $Y \neq 0$ and $|X| \leqq |Y|$, then $W = |Y| * \sqrt{1 + \left(\frac{X}{Y}\right)^2}$
   c. If $Y \neq 0$ and $|X| > |Y|$, then $W = |X| * \sqrt{1 + \left(\frac{Y}{X}\right)^2}$
2. ~~$|Z| = (W, 0)$~~ $|Z| = W$ (returns real value in EAQ)
3. W, X, and Y are real numbers; values of X and Y range from -2^{127} to $2^{127}-2^{100}$ inclusive. ~~Hence, both Z and $|Z|$ are complex numbers.~~ Therefore Z is a complex number but $|Z|$ is a real number (See Change Letter #6 dated 4/7/66)
4. W is accurate to 8 decimal positions.

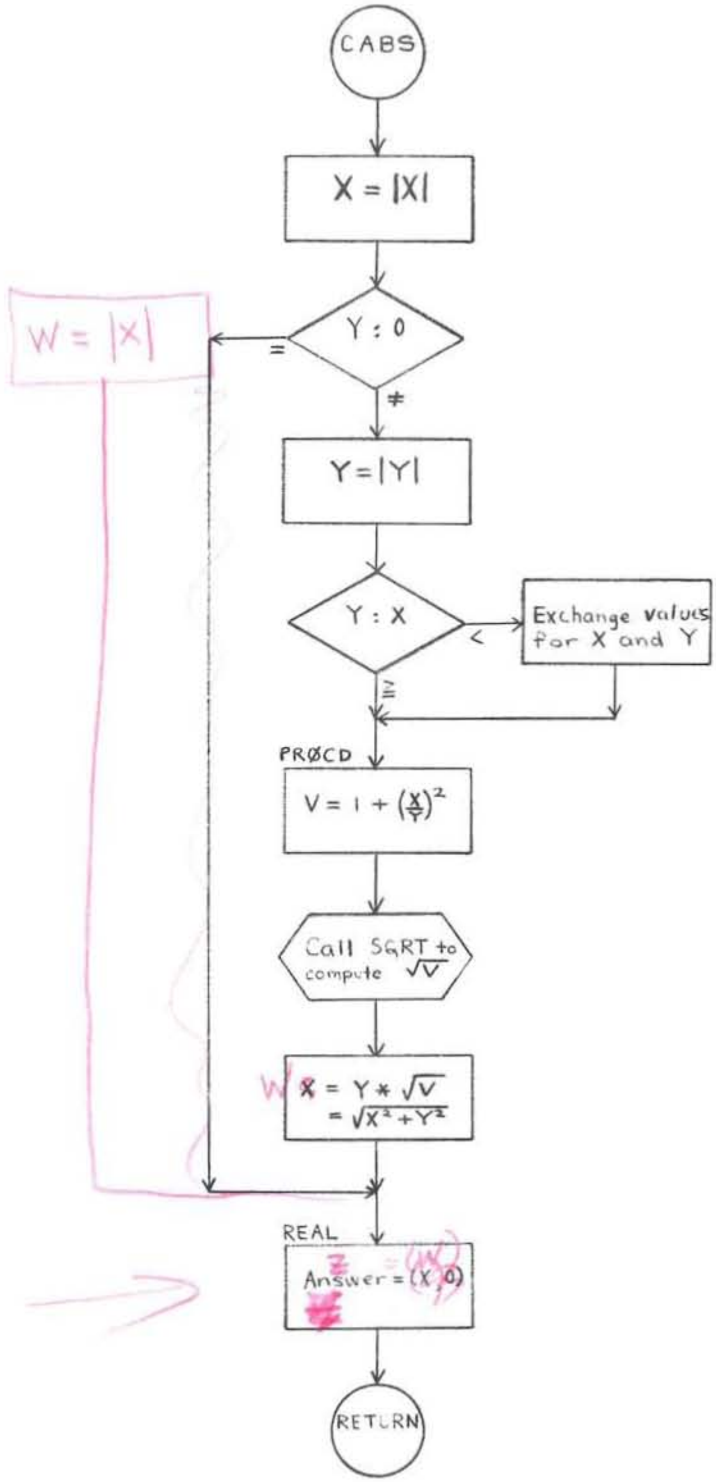
III. USAGE

1. Calling Sequence--CALL CABS (Z)
2. CABS uses ~~36~~ 40_{10} words. 46_8
3. No error conditions.

IV. RESTRICTIONS

The subprogram FSQR must be in memory.

COMPUTE $\sqrt{X^2+Y^2}$ FOR COMPLEX (X, Y)



FXP1--EXPONENTIAL--INTEGER BASE AND EXPONENT

PURPOSE

To compute I^J for I**J in an expression.

METHOD

1. For positive values of J, let $k_m \ldots k_2 k_1 k_0$ be the binary representation of J, where $0 \leqq m \leqq 34$.

   Then $I^J = I^{(k_0 + 2*k_1 + 4*k_2 + \ldots + 2^m*k_m)}$

   $= (I^1)^{k_0}*(I^2)^{k_1}*(I^4)^{k_2}* \ldots *(I^{(2^m)})^{k_m}$

   = the product of those powers of I above for which $k_n = 1$, where $0 \leqq n \leqq m$.

2. For negative values of J, $I^J = 0$ if $|I| \neq 1$. Use the method above with J (mod 2) if $|I| = 1$.

3. I, J, and I^J are integers with values from -2^{35} to $2^{35}-1$ inclusive.

4. The algorithm uses integer multiplication (MPY) exclusively.

5. I^J is accurate to 35 binary positions.

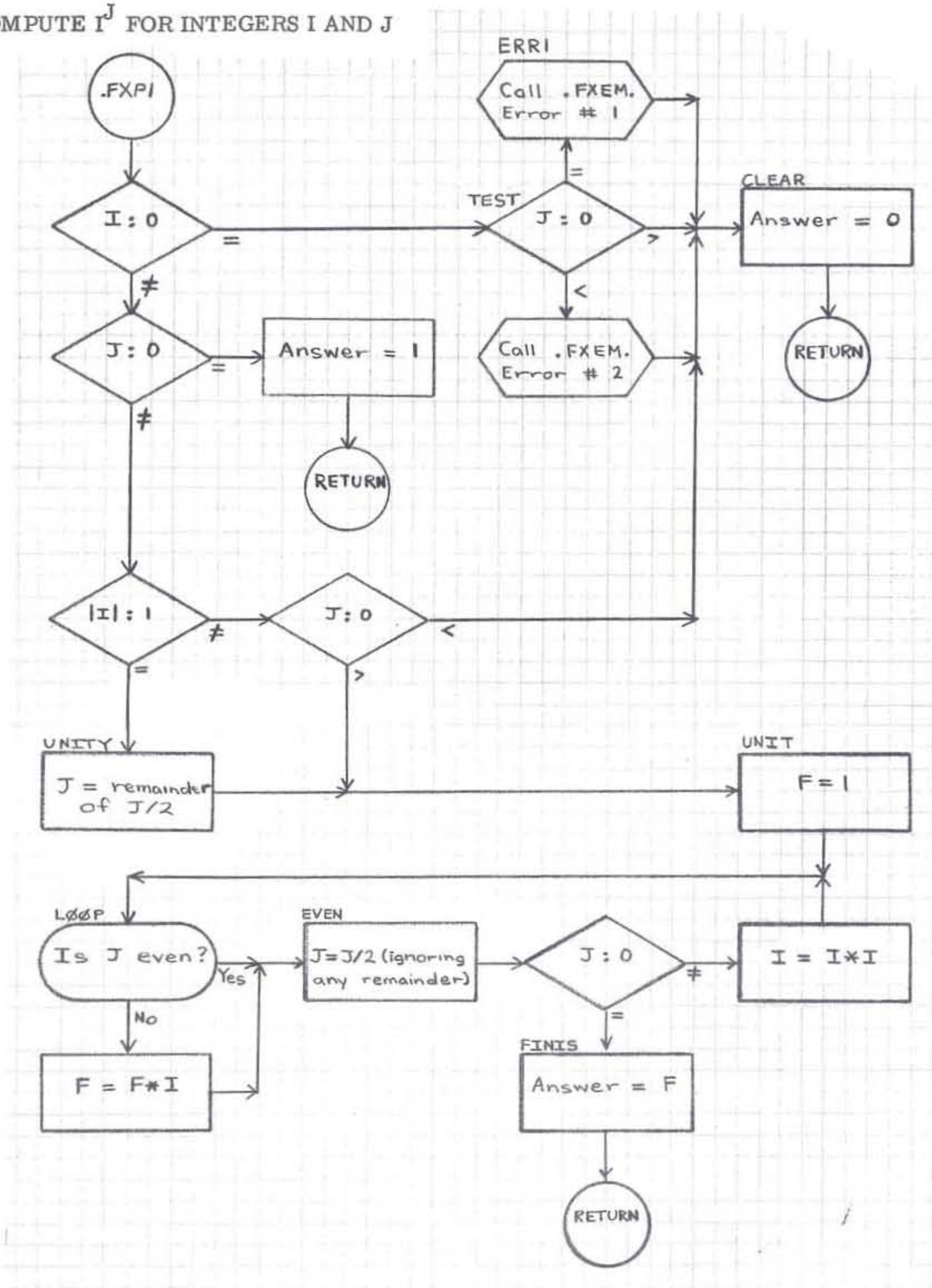
USAGE

1. Calling Sequence--CALL .FXP1 (I, J)
2. FXP1 uses 52 words.
3. The error conditions are:
   a. FXEM Error #1 if I = 0 and J = 0. Then $I^J = 0$.
   b. FXEM Error #2 if I = 0 and J < 0. Then $I^J = 0$.

RESTRICTIONS

The subprogram FXEM must be in memory.

COMPUTE I^J FOR INTEGERS I AND J



FXP2--EXPONENTIAL--FLOATING-POINT BASE, INTEGER EXPONENT

PURPOSE

To compute A^K for A**K in an expression.

METHOD

1. For positive values of K, use the same method as in FXP1--Exponential--Integer Base and Exponent, CD600D2.001.

2. For negative values of K, proceed with $|K|$ as above, and then take the reciprocal of the result.

3. K is an integer with values from -2^{35} to $2^{35}-1$ inclusive; A and A^K are floating-point numbers with values from -2^{127} to $2^{127}-2^{64}$ inclusive.

4. A^K is accurate to 8 decimal positions for .XP2.
   or 16 decimal positions for .DXP1.

USAGE

1. Calling Sequence--CALL .FXP2 (A, K) for Real A
   CALL .FDXP1 (A, K) for Double-Precision A

2. FXP2 uses 60 words.

3. The error conditions are:

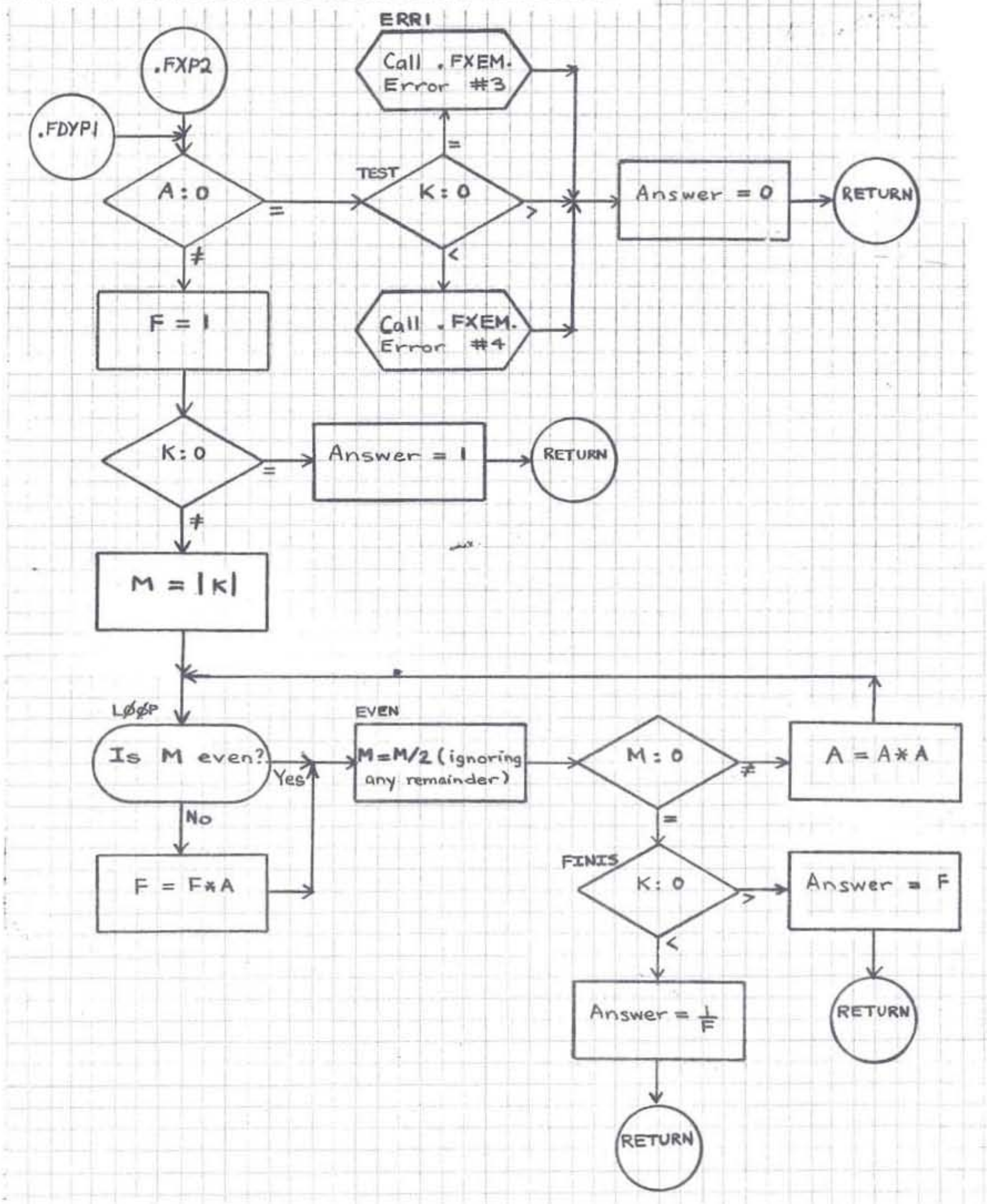
   a. FXEM Error #3 if A = 0 and K = 0. Then $A^K = 0$.

   b. FXEM Error #4 if A = 0 and K < 0. Then $A^K = 0$.

RESTRICTIONS

The subprogram FXEM must be in memory.

COMPUTE A^K FOR FLOATING POINT A AND INTEGER K



FXP3--EXPONENTIAL--REAL BASE AND EXPONENT

PURPOSE

To compute A^B for A**B in an expression.

METHOD

1. $A^B = (e^{\log_e A})^B = e^{(B*\log_e A)}$

2. A, B, and A^B are real numbers with values from -2^{127} to $2^{127}-2^{100}$ inclusive.

3. A^B is accurate to 7 decimal positions.

USAGE

1. Calling Sequence--CALL .FXP3 (A, B)

2. FXP3 uses 50 words.

3. The error conditions are:

   a. FXEM Error #5 if $A < 0$ and $B \neq 0$. Then $A^B = |A|^B$.

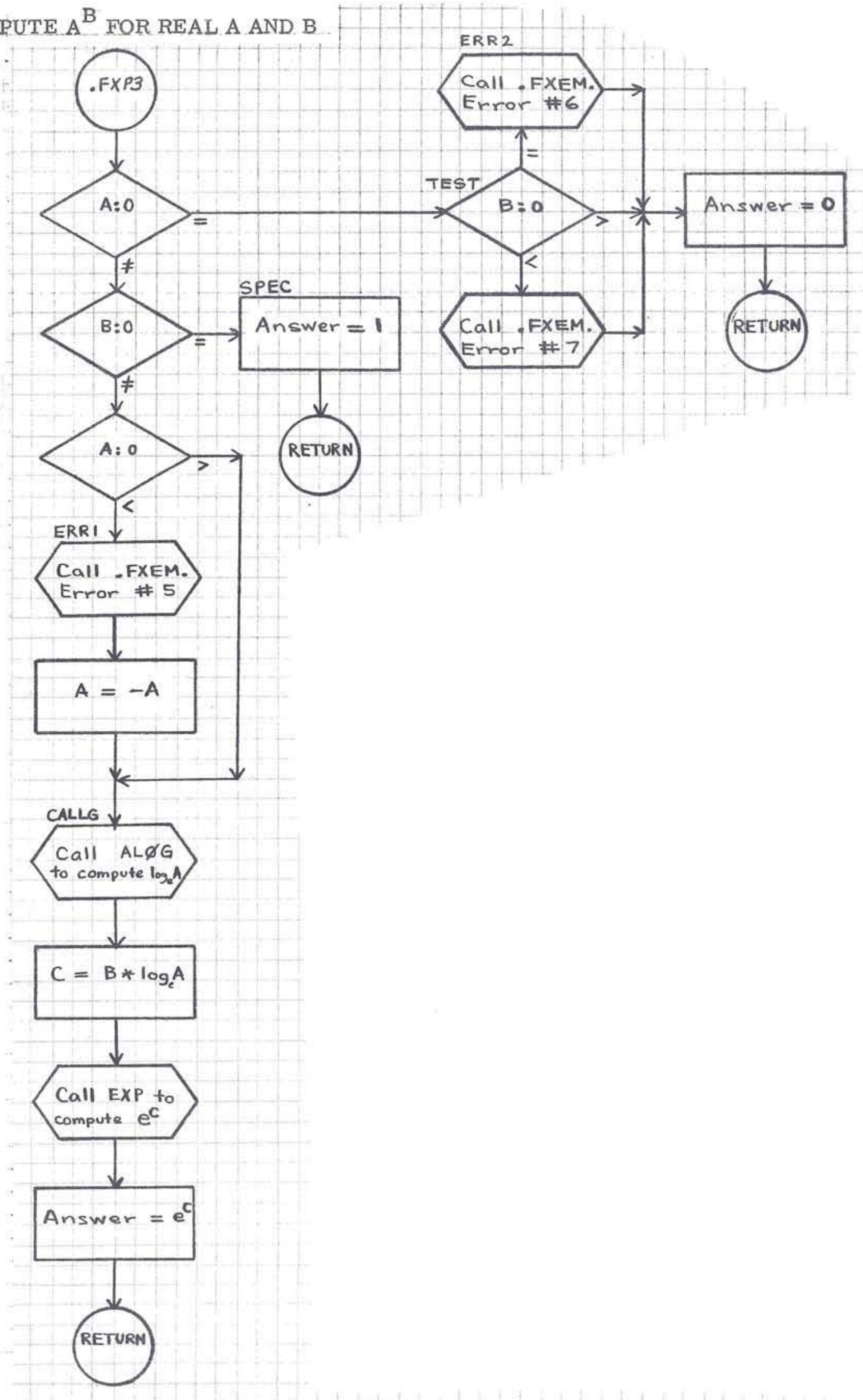
   b. FXEM Error #6 if $A = 0$ and $B = 0$. Then $A^B = 0$.

   c. FXEM Error #7 if $A = 0$ and $B < 0$. Then $A^B = 0$.

RESTRICTIONS

The subprograms FLOG, FXPF, and FXEM must be in memory.

COMPUTE A^B FOR REAL A AND B



FDX1--EXPONENTIAL--COMPLEX BASE, INTEGER EXPONENT

PURPOSE

To compute A^K for A**K in an expression.

METHOD

1. Use the same method as in FXP2--Exponential--Floating Point Base, Integer Exponent, CD600D2.002.

2. A is a complex number (X, Y), with values of X and Y from -2^{127} to $2^{127}-2^{100}$ inclusive. K is an integer, with values from -2^{35} to $2^{35}-1$ inclusive.

3. A^K is accurate to 8 decimal positions.

USAGE

1. Calling Sequence--CALL .FCXP1 (A, K)

2. FDX1 uses 68 words.

3. The error conditions are:

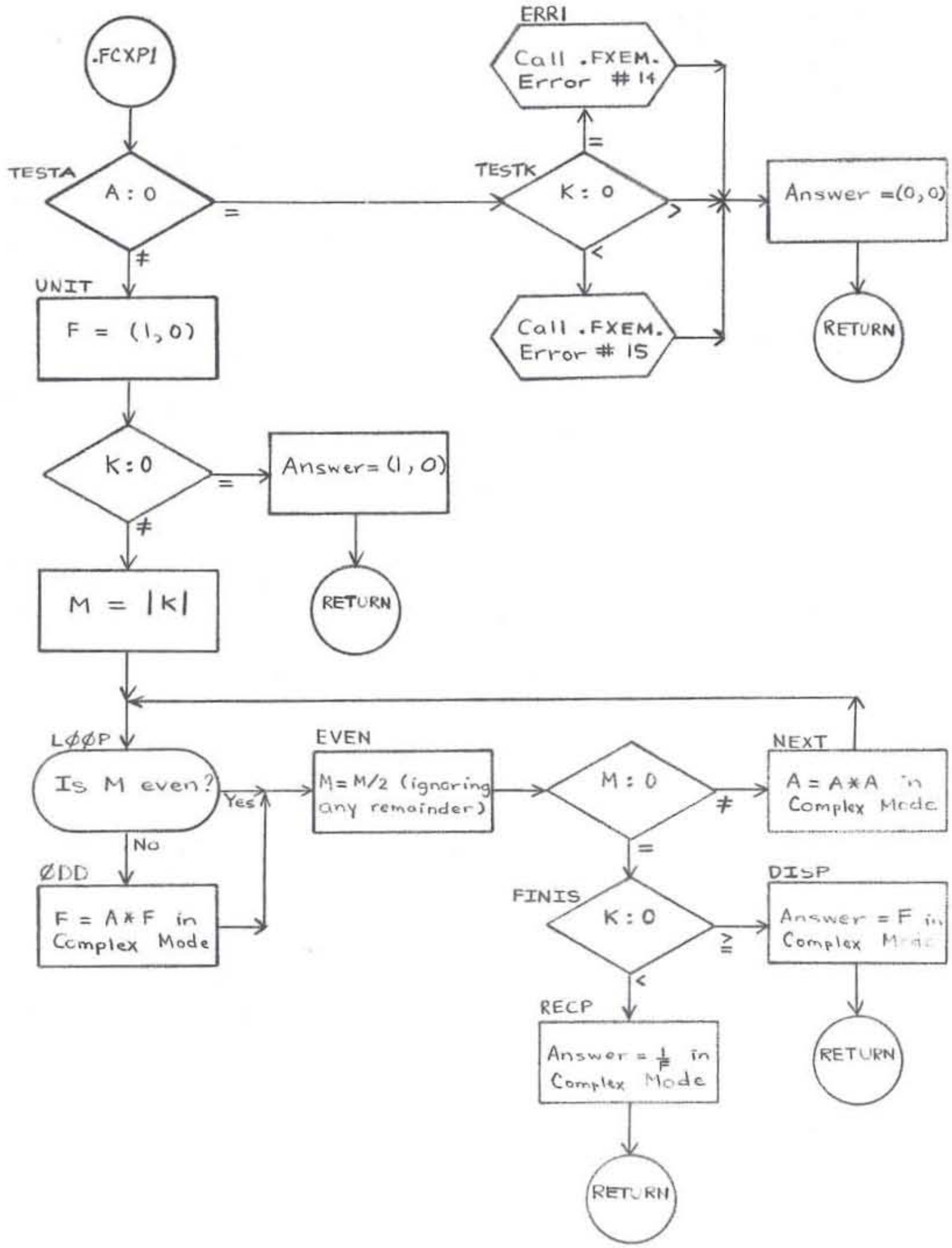
   a. FXEM Error #14 if A = (0,0) and K = 0.
      Then $A^K = (0,0)$.

   b. FXEM Error #15 if A = (0,0) and K < 0.
      Then $A^K = (0,0)$.

RESTRICTIONS

The subprograms FCAS and FXEM must be in memory.

COMPUTE A^K FOR COMPLEX A AND INTEGER K



FDX2--EXPONENTIAL--DOUBLE-PRECISION BASE AND EXPONENT

PURPOSE

To compute A^B for A**B in an expression.

METHOD

1. Use the same method as in FXP3--Exponential--Real Base and Exponent, CD600D2.003.

2. A, B, and A^B are double-precision numbers, with values from -2^{127} to $2^{127}-2^{64}$ inclusive.

3. A^B is accurate to 16 decimal digits.

USAGE

1. Calling Sequence--CALL .FDXP2 (A, B)

2. FDX2 uses 52 words.

3. The error conditions are:

   a. FXEM Error #16 if $A < 0$ and $B \neq 0$. Then $A^B = |A|^B$.

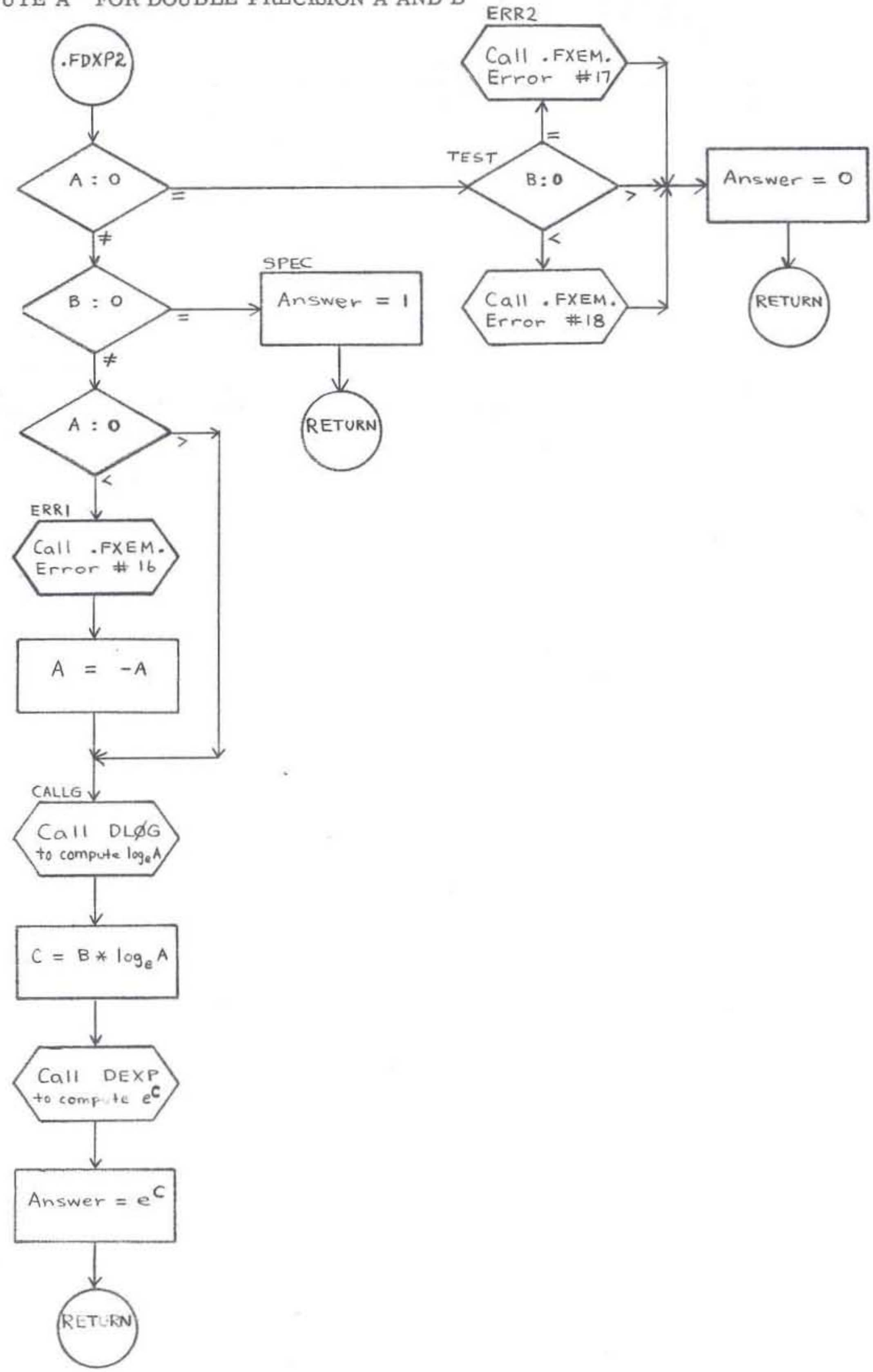
   b. FXEM Error #17 if $A = 0$ and $B = 0$. Then $A^B = 0$.

   c. FXEM Error #18 if $A = 0$ and $B < 0$. Then $A^B = 0$.

RESTRICTIONS

The subprograms FDLG, FDXP, and FXEM must be in memory.

COMPUTE A^B FOR DOUBLE PRECISION A AND B



REAL AND DOUBLE-PRECISION LOGARITHM, BASE 2 (ALGT)

PURPOSE

Real and Double-Precision Logarithm (ALGT) computes $A = \log_2 X$ in a FORTRAN expression.

METHOD

The double-precision $\log_e X$ is divided by $\log_e 2$ and the result returned. Because of the hardware representation, this result is valid for both double and single precision.

USAGE

ALGT is designed to be used as a FORTRAN IV function:

A = ALOG2(X) for single precision;

A = DLOG2(X) for double precision.

RESTRICTIONS

The argument for ALGT must be ≥ 0.

If $X = 0$, FXEM Error #20 is returned and $\log_2 X = 0$.

If $X < 0$, FXEM Error #21 is returned and $\log_2 X = \log_2 |X|$.

ARCSINE AND ARCCOSINE (ASIN)

PURPOSE

Arcsine and Arccosine routine (ASIN) computes $\sin^{-1} X$ or $\cos^{-1} X$ in a FORTRAN IV expression.

METHOD

The arcsine or arccosine is calculated by computing the complementary function (sine or cosine), and calling ATAN2 (sin,cos) to get the resulting angle in radians. The computation is done entirely in double precision.

USAGE

ASIN is used as a FORTRAN IV function in the following ways:

- A = ASIN(X) for real arcsine;
- A = ACOS(X) for real arccosine;
- A = DASIN(X) for double-precision arcsine;
- A = DACOS(X) for double-precision arccosine.

TANGENT & COTANGENT (TANG)

PURPOSE

The Tangent and Cotangent routine (TANG) computes tan X or cot X in a FORTRAN IV expression.

METHOD

Using double-precision arithmetic, tan X and cot X are computed from the trigonometric identities:

- Tan X = sin X/cos X
- Cot X = cos X/sin X

If the divisor is zero, the largest possible floating-point number is returned.

USAGE

TANG is used as a FORTRAN IV function in the following ways:

- A = TAN(X) for real tangent;
- A = COT(X) for real cotangent;
- A = DTAN(X) for double-precision tangent;
- A = DCOT(X) for double-precision cotangent.

RESTRICTIONS

TANG produces FXEM Error #23 if $|X| > 2^{54}$.

HYPERBOLIC SINE AND COSINE (SINH)

PURPOSE

The Hyperbolic Sine and Cosine routine (SINH) computes sinh X or cosh X in a FORTRAN IV expression.

METHOD

Sinh X and cosh X are computed, using double-precision arithmetic, from the definitions:

- Sinh X = $0.5(e^{x} - e^{-x})$
- Cosh X = $0.5(e^{x} + e^{-x})$

USAGE

SINH is used as a FORTRAN IV function in the following ways:

- A = SINH(X) for real hyperbolic sine;
- A = COSH(X) for real hyperbolic cosine;
- A = DSINH(X) for double-precision hyperbolic sine;
- A = DCOSH(X) for double-precision hyperbolic cosine.

RESTRICTIONS

SINH produces FXEM Error #19 if $|X| > 88.028$.

FLXP

FXPF--REAL NATURAL EXPONENTIAL

I. PURPOSE

To compute e^X for EXP(X) in an expression.

II. METHOD

1. $e^X = 2^{\log_2 e^X} = 2^{X*\log_2 e} = 2^{I+F} = 2^I * 2^F$, where
   I = greatest integer $\leq X*\log_2 e$ and $I+F = X*\log_2 e$

2. Then $$2^F = 1 + \cfrac{2*F}{D + C*F^2 - F + \cfrac{B}{F^2 + A}}$$

   where A = 87.417497202
   B = -617.9722695
   C = 0.03465735903
   and D = 9.9545957821

3. X and e^X are real numbers; $X \leq 88.028$

4. e^X is accurate to 8 decimal positions.

III. USAGE

1. Calling Sequence--CALL EXP(X)

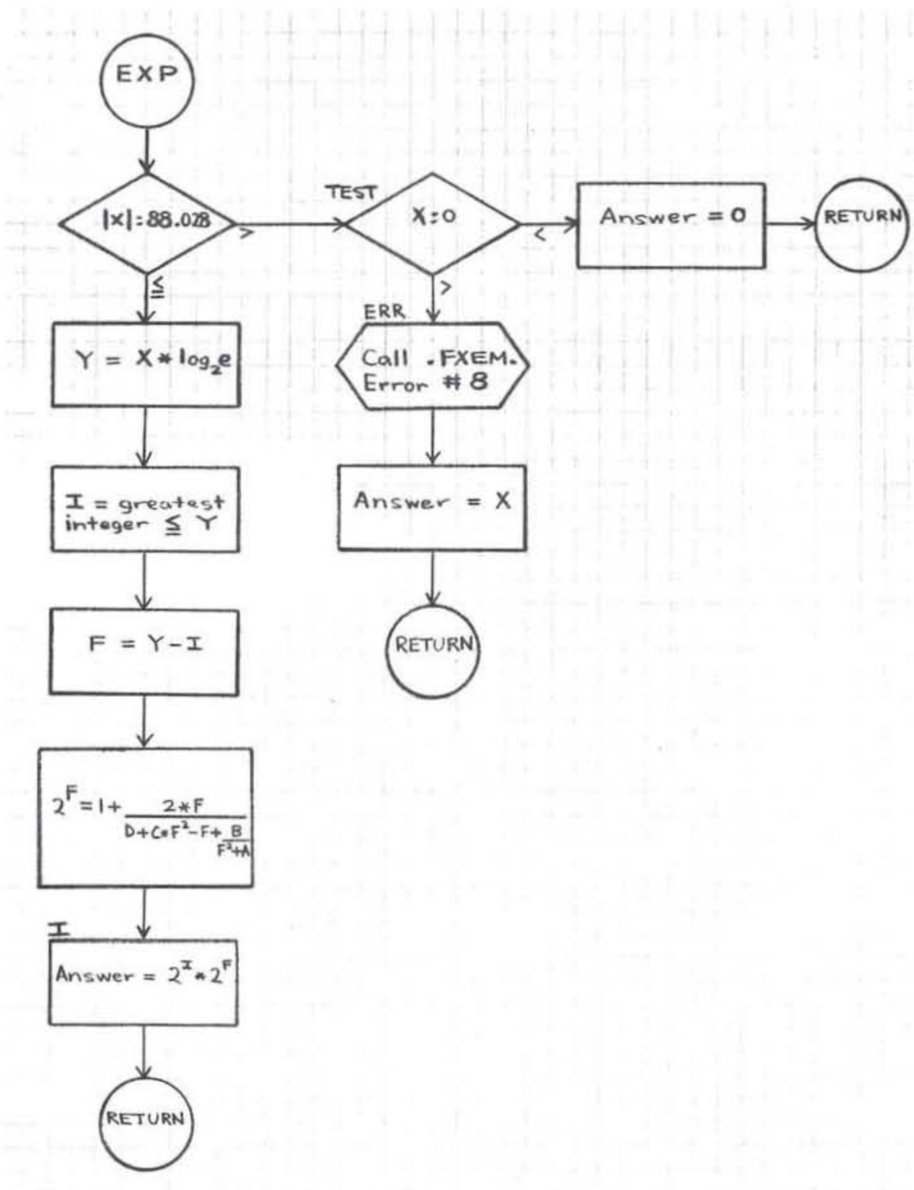
2. FXPF uses 56 words. 72₁₀ 110(8)

3. The error condition is:
   FXEM Error #8 if X > 88.028. Then e^X = X.

IV. RESTRICTIONS

The subprogram FXEM must be in memory.

COMPUTE e^X FOR REAL X



FLOG--REAL LOGARITHM

I. PURPOSE

To compute log X for ALOG(X) or ALOG10(X) in an expression.

II. METHOD

1. $\log_2 X = \log_2 (2^I * F) = I + \log_2 F$, where $X = 2^I * F$.

2. $\log_e X = \log_e 2^{(\log_2 X)} = (\log_2 X) * (\log_e 2)$, and similarly $\log_{10} X = (\log_2 X) * (\log_{10} 2)$.

3. $\log_2 X = Z * \left(A + \frac{B}{Z^2 - C}\right) - 1/2$, where $Z = \frac{F - \frac{\sqrt{2}}{2}}{F + \frac{\sqrt{2}}{2}}$ and

   A = 1.2920070987
   B = -2.6398577031
   C = 1.6567626301

4. X and log X are real numbers; values of X range from 2^{-129} to $2^{127}-2^{100}$ inclusive.

5. log X is accurate to 8 decimal places.

III. USAGE

1. Calling Sequence--CALL ALOG(X) for $\log_e X$
   CALL ALOG10(X) for $\log_{10} X$

2. FLOG uses 64 words.

3. The error conditions are:

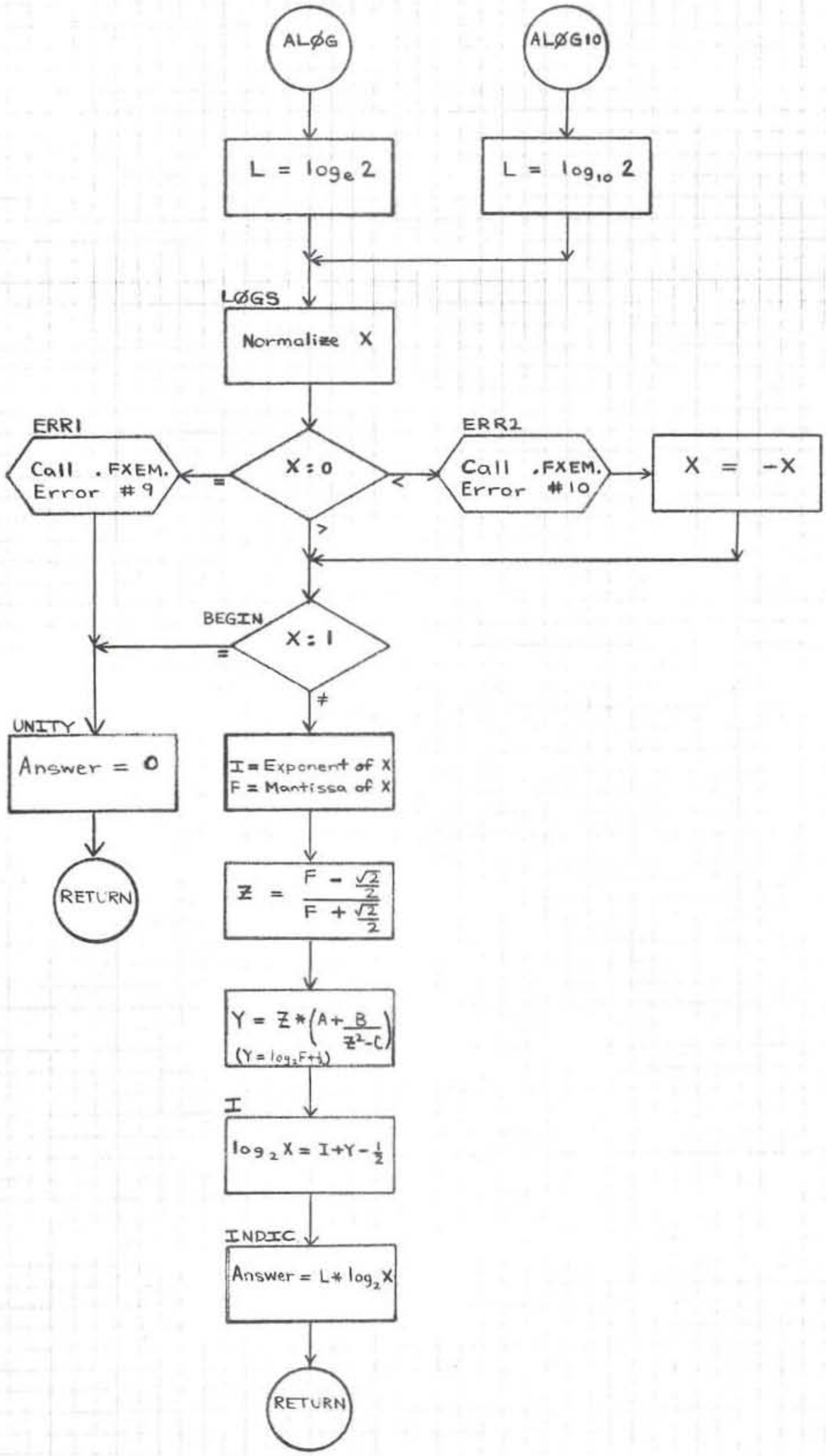
   a. FXEM Error #9 if X = 0. Then log X = 0.

   b. FXEM Error #10 if $X < 0$. Then log X = log $|X|$.

IV. RESTRICTIONS

The subprogram FXEM must be in memory.

COMPUTE $\log_e X$ OR $\log_{10} X$ FOR REAL X



FATN--REAL ARCTANGENT

I. PURPOSE

To compute the principal value of arctan X or arctan $\frac{Y}{Z}$ (in radians) for ATAN(X) or ATAN2(Y,Z) in an expression.

II. METHOD

1. Compute arctan X as follows:

   a. For positive values of X:

      (1) If $X \geqq 2^{27}$, then arctan X = $\frac{\pi}{2}$.

      (2) If $X < 2^{-27}$, then arctan X = X.

      (3) If $2^{-27} \leqq X < 2^{27}$, then set I from the interval containing arctan X:

      0 for 0°-10°, 1 for 10°-30°, 2 for 30°-50°, 3 for 50°-70°, and 4 for 70°-90°.

      Then $$\arctan X = N_I + \cfrac{T}{T^2 + K_3 + \cfrac{K_2}{T^2 + K_1}}$$

      where $T = \frac{9}{55} * X$ for I = 0

      $T = A_I + \frac{B_I}{X + C_I}$ for I = 1,2,3,4

      and the constants $A_I, B_I, C_I, N_I, K_1, K_2, K_3$ assume appropriate values.

   b. For negative values of X, arctan X = -arctan |X| .

2. Compute arctan $\frac{Y}{Z}$ as follows:

   a. If Z = 0, then arctan $\frac{Y}{Z} = \frac{\pi}{2}$ for Y> 0,

      and arctan $\frac{Y}{Z} = -\frac{\pi}{2}$ for Y < 0.

   b. If Z≠ 0, then X = $\frac{Y}{Z}$; compute arctan X as above.

   c. If Z> 0, then arctan $\frac{Y}{Z}$ = arctan X.

   d. If Z< 0, then arctan $\frac{Y}{Z}$ = arctan X+π for Y>0,

      and arctan $\frac{Y}{Z}$ = arctan X-π for Y<0.

3. X, Y, and Z are real numbers with values from -2^{127} to $2^{127}-2^{100}$ inclusive. The result is a real number.

4. The answer is accurate to 8 decimal positions.

III. USAGE

1. Calling Sequence--CALL ATAN(X) for arctan X
   CALL ATAN2(Y, Z) for arctan $\frac{Y}{Z}$

2. FATN uses 98 words.

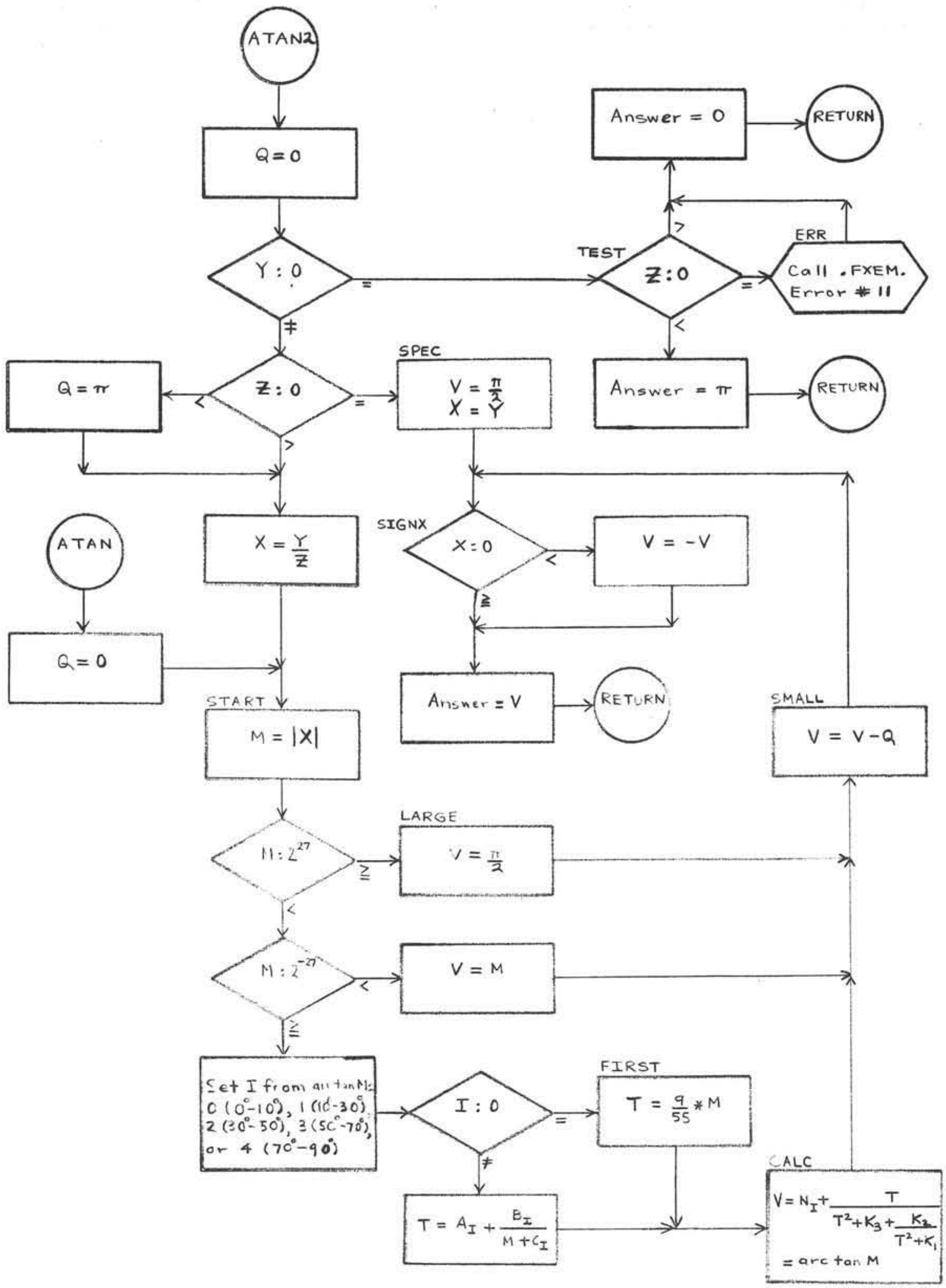
3. The error condition is:

   FXEM Error #11 if Y = 0 and Z = 0.
   Then arctan $\frac{Y}{Z} = 0$.

IV. RESTRICTIONS

The subprogram FXEM must be in memory.

COMPUTE ARCTAN X OR ARCTAN $\frac{Y}{Z}$ FOR REAL X, Y, AND Z



FSIN

FSCN--REAL SINE AND CONSINE

I. PURPOSE

To compute sin X or cos X for SIN(X) or COS(X) in an expression, where X is in radians.

II. METHOD

1. Compute sin X as follows:

   a. For positive values of X:

      (1) Replace the value of X by the value of $X - I*\pi$, where I is the greatest integer $\leq \frac{X}{\pi}$, noting that $\sin(X + n\pi) = (-1)^n * \sin X$. Now if $X \geq \frac{\pi}{2}$, then replace the value of X by the value of $\pi - X$ and proceed, noting that $\sin(\pi - X) = \sin X$.

      (2) If $X \leq 2^{-8}$, then $\sin X = X$. If $2^{-8} < X \leq 0.3$, then $\sin X = X * \left(A_1 + \frac{B_1}{C_1 + X^2} + \frac{X^2}{4}\right)$.
      If $X \geq 0.3$, then replace the value of X by the value of $\frac{\pi}{2} - X$, noting that $\cos\left(\frac{\pi}{2} - X\right) = \sin(X)$. Then compute
      $$\cos(X) = D - 2 * X^2 + \frac{E - 320 * X^2}{A_2 + \frac{B_2}{C_2 + X^2} + X^2}$$

   b. For negative values of X, $\sin X = -\sin |X|$.

2. Compute $\cos X = \sin\left(X + \frac{\pi}{2}\right)$.

3. X, sin X, and cos X are real numbers, with $|X| < 2^{27}$.

4. The answer is accurate to 8 decimal positions.

III. USAGE

1. Calling Sequence--CALL SIN(X) for sin X
   CALL COS(X) for cos X

2. FSCN uses 100 words.

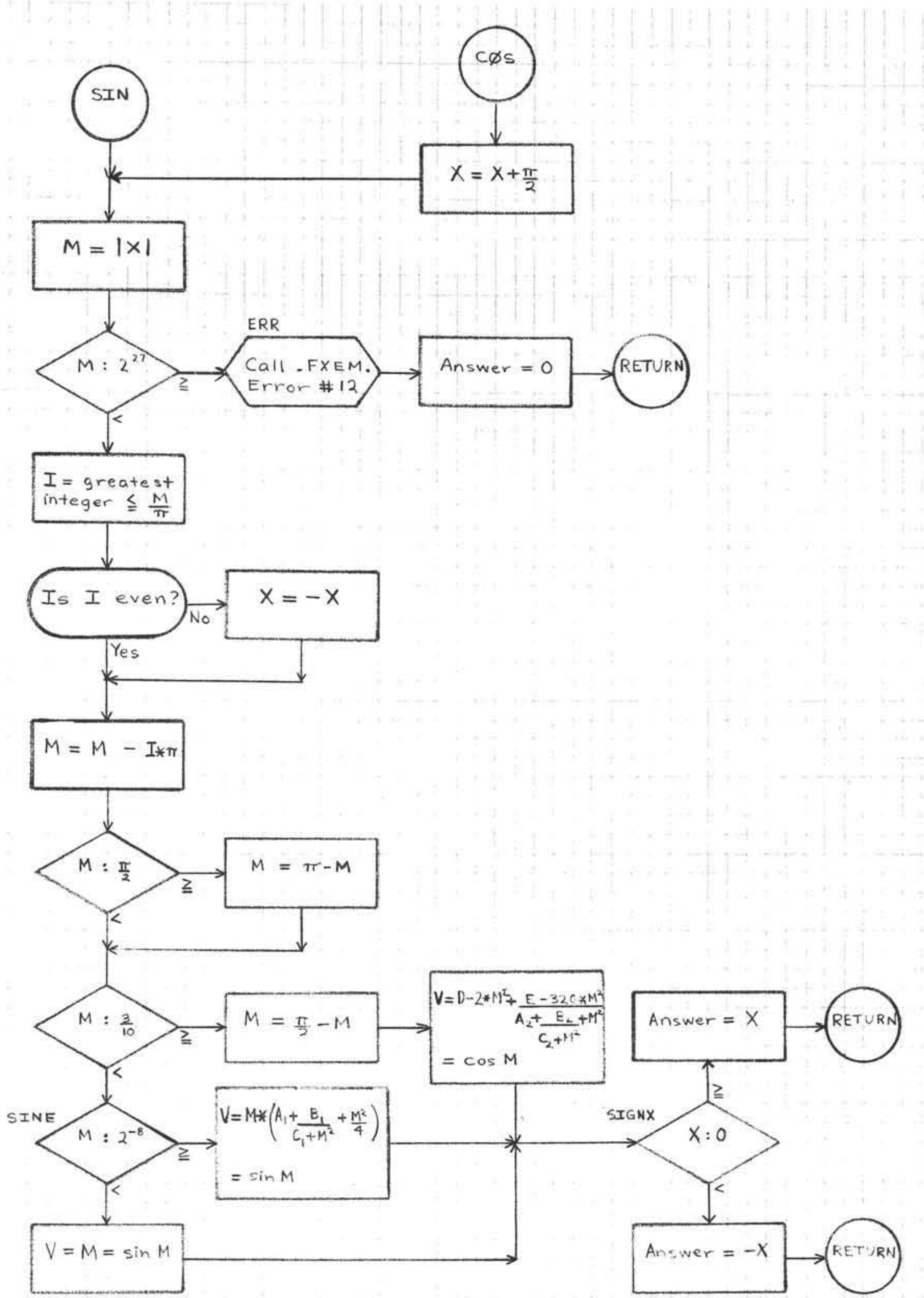
3. The error condition is:

   FXEM Error # 12 if $|X| \geqq 2^{27}$.
   Then the answer is 0.

IV. RESTRICTIONS

The subprogram FXEM must be in memory.

COMPUTE SIN X OR COS X FOR REAL X



FTNH--REAL HYPERBOLIC TANGENT

I. PURPOSE

To compute tanh X for TANH(X) in an expression.

II. METHOD

1. For positive values of X:

   a. If $X > 12$, then $\tanh X = 1$

   b. If $0.17 \leqq X < 12$, then $\tanh X = \dfrac{e^{2*X}-1}{e^{2*X}+1}$

   c. If $0.00034 \leqq X < 0.17$, then $\tanh X = \dfrac{F}{A+F^2*\left(B+\dfrac{C}{D+F^2}\right)}$

   where F = A * X
   A = 5.77078016
   B = 0.0173286795
   C = 14.1384114
   and D = 349.669989

   d. If $X < 0.00034$, then $\tanh X = X$

2. For negative values of X, $\tanh X = -\tanh |X|$.

3. X and tanh X are real numbers; values of X range from -2^{127} to $2^{127}-2^{100}$ inclusive.

4. tanh X is accurate to 8 decimal positions.

III. USAGE

1. Calling Sequence--CALL TANH(X)

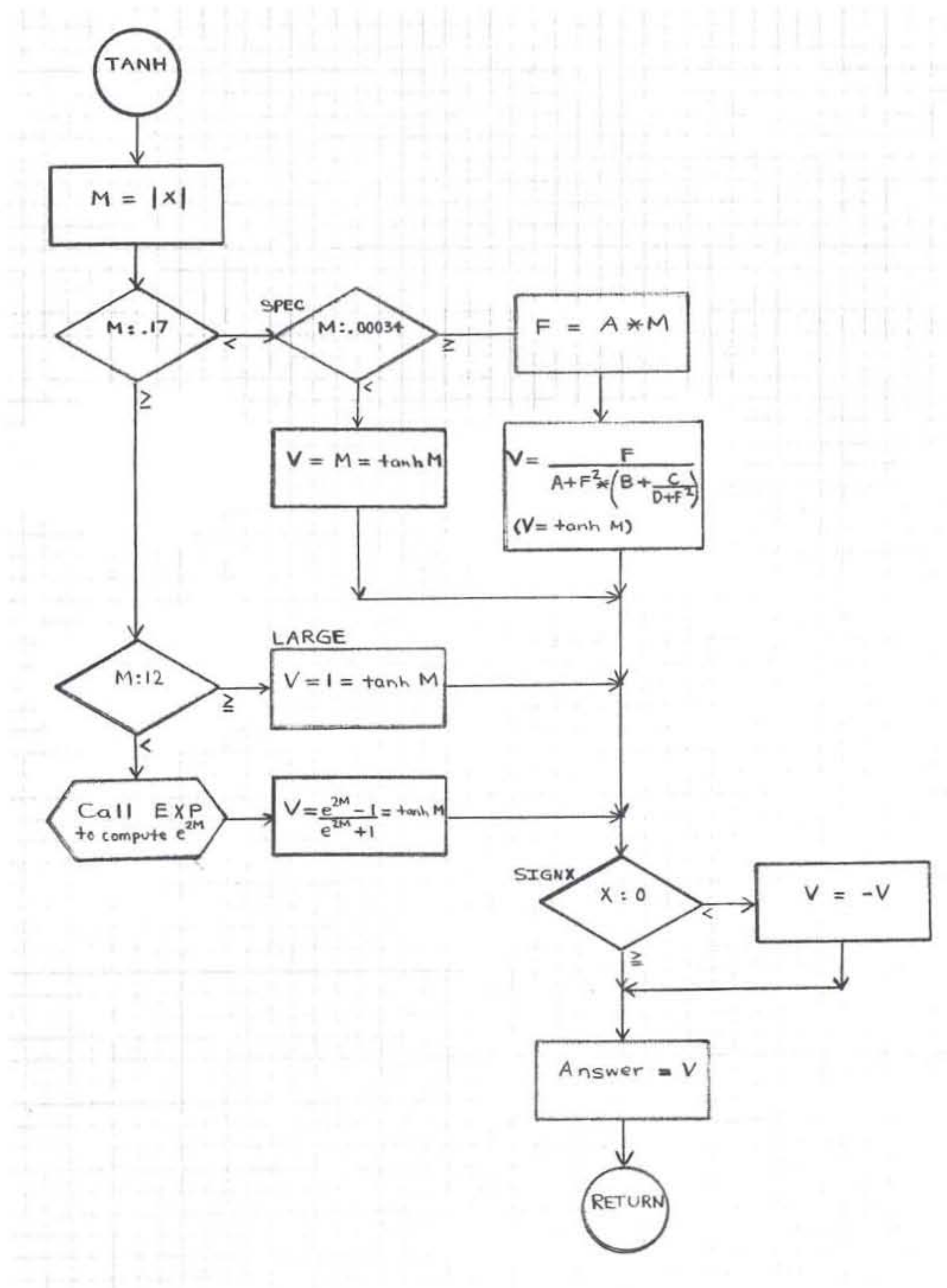
2. FTNH uses 48 words.

3. No error conditions.

IV. RESTRICTIONS

The subprogram FXPF must be in memory.

COMPUTE TANH X FOR REAL X



FSQR--REAL SQUARE ROOT

I. PURPOSE

To compute $\sqrt{X}$ for SQRT(X) in an expression.

II. METHOD

1. If $X = 0$, then $\sqrt{X} = 0$. Otherwise, let $X = 2^{2*A} * B$, where $1/4 \leqq B < 1$. Then $\sqrt{X} = 2^{A} * \sqrt{B} = 2^{A-1} * (2 * \sqrt{B})$.

   First Approximation: $P_o = 1/4 + B$ if $1/4 \leqq B < 1/2$,

   or $P_o = 1/2 + B/2$ if $1/2 \leqq B < 1$.

   Then $P_1 = \frac{1}{2} * (P_o + \frac{B}{P_o})$, $P_2 = \frac{1}{2} * (P_1 + \frac{B}{P_1})$, and finally

   $\sqrt{X} = 2^{A-1} * (P_2 + \frac{B}{P_2})$.

2. X and $\sqrt{X}$ are real numbers; values of X range from -2^{127} to $2^{127}-2^{100}$ inclusive.

3. $\sqrt{X}$ is accurate to 8 decimal positions. The last iteration is in double precision.

III. USAGE

1. Calling Sequence--CALL SQRT(X)

2. FSQR uses ~~46~~ 58₁₀ words. 72(8)

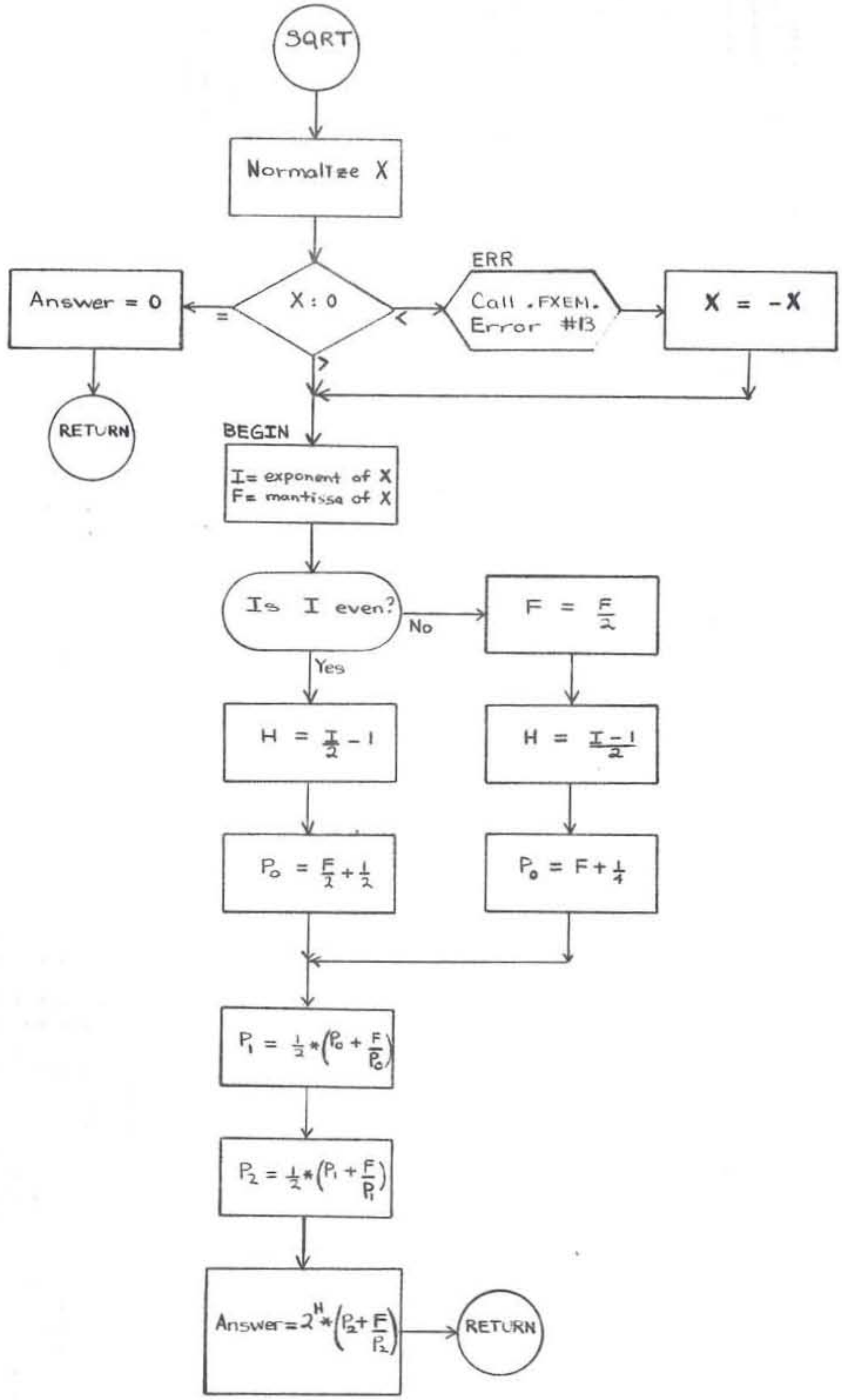
3. The error condition is:

   FXEM Error #13 if $X < 0$. Then $\sqrt{X} = \sqrt{|X|}$.

IV. RESTRICTIONS

The subprogram FXEM must be in memory.

COMPUTE $\sqrt{X}$ FOR REAL X



FCEX

FCXP--COMPLEX EXPONENTIAL

I. PURPOSE

To compute e^Z for CEXP(Z) in an expression.

II. METHOD

1. $e^Z = e^{(X, Y)}$ (where Z = (X, Y))

   $= e^X * e^{(0, Y)}$

   $= e^X * (\cos Y, \sin Y)$

   $= (e^X * \sin (Y + \frac{\pi}{2}), e^X * \sin Y)$

2. Z and e^Z are complex numbers, with

   $X \leqq 88.028$, $|Y| < 2^{27}$, and $|Y + \frac{\pi}{2}| < 2^{27}$.

3. e^Z is accurate to 7 decimal positions.

III. USAGE

1. Calling Sequence--CALL CEXP (Z)

2. CEXP uses ~~54~~ 78₁₀ words. 116₍₈₎

3. The error conditions are:

   a. FXEM Error #26 if $X > 88.028$. Then $e^Z = Z$.

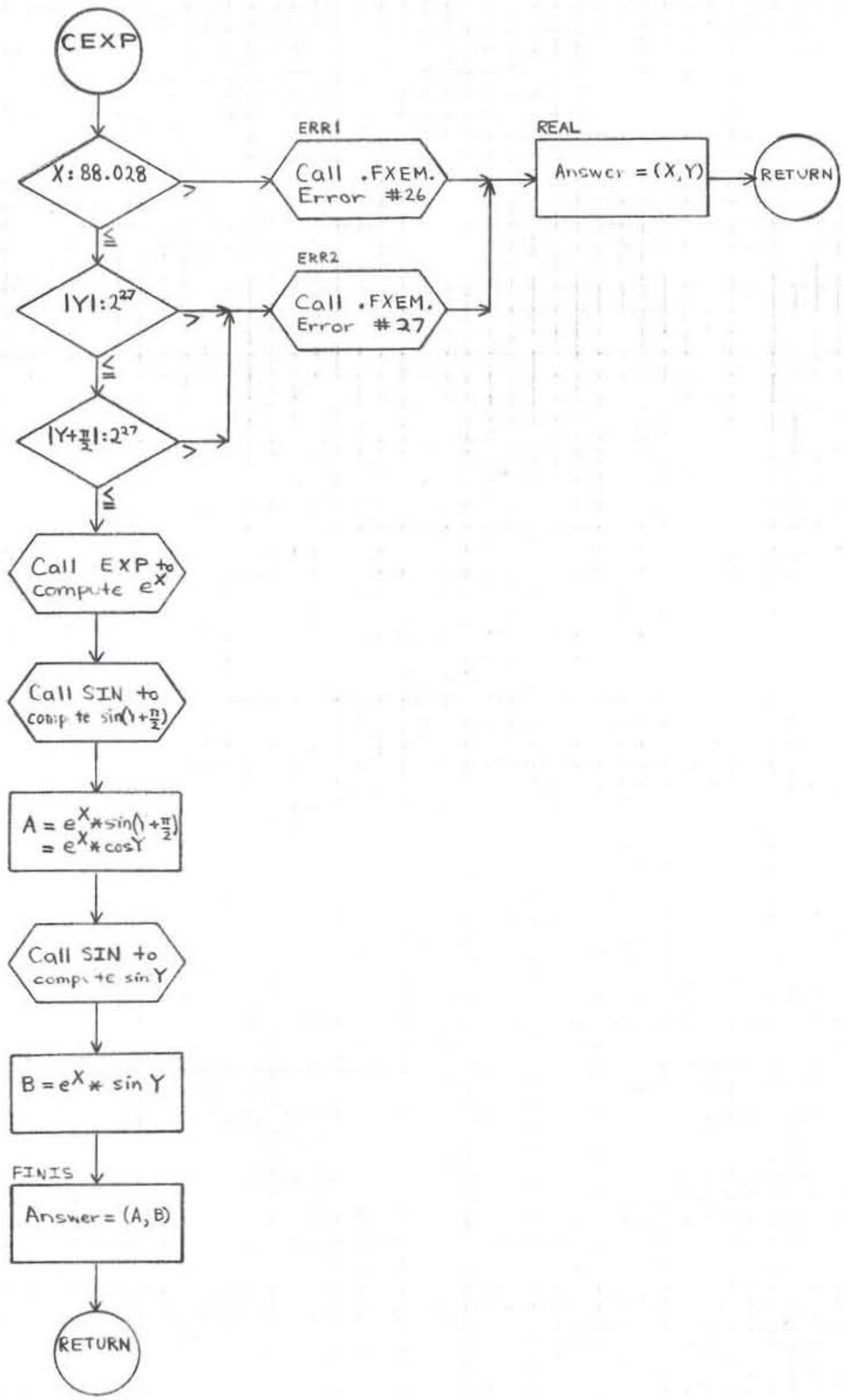
   b. FXEM Error #27 if $|Y| \geqq 2^{27}$

      or if $|Y + \frac{\pi}{2}| \geqq 2^{27}$. Then $e^Z = Z$.

IV. RESTRICTIONS

The subprograms FXPF, FSCN, and FXEM must be in memory.

COMPUTE e^Z FOR COMPLEX Z = (X, Y)



FCLG--COMPLEX LOGARITHM

I. PURPOSE

To compute $\log_e Z$ for CLOG(Z) in an expression

II. METHOD

1. $\log_e Z = \log_e (X, Y)$ (where $Z = (X, Y)$)

$= (\log |Z| , \arc \tan \frac{Y}{X})$

2. Z and $\log_e Z$ are complex numbers; values of X and Y range from -2^{127} to $2^{127}-2^{100}$ inclusive.

3. $\log_e Z$ is accurate to 7 decimal positions.

III. USAGE

1. Calling Sequence--CALL CLOG(Z)

2. FCLG uses ~~40~~ 50_{10} words. $62_{(8)}$

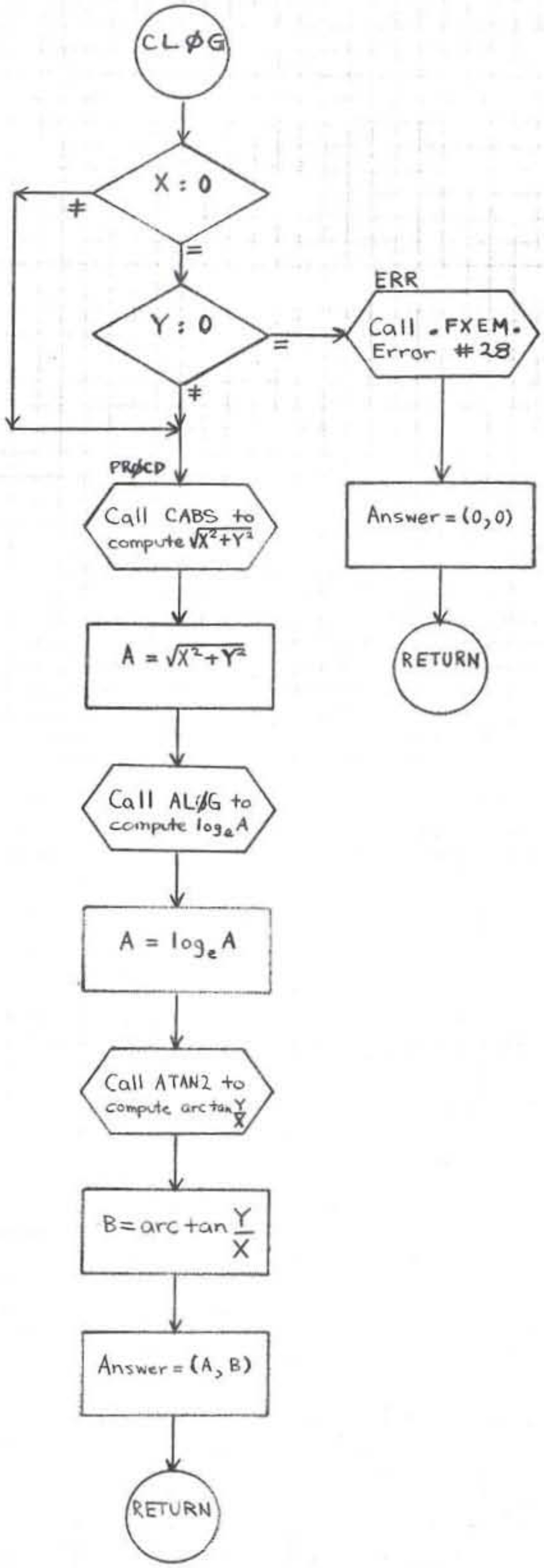
3. The error condition is:

   FXEM Error #28 if $Z = (0, 0)$. Then $\log_e Z = (0, 0)$.

IV. RESTRICTIONS

The subprograms FATN, FCAB, FLOG, and FXEM must be in memory.

COMPUTE $\log_e Z$ FOR COMPLEX Z = (X, Y)



FCSQ--COMPLEX SQUARE ROOT

I. PURPOSE

To compute $\sqrt{Z}$ for CSQRT(Z) in an expression.

II. METHOD

1. Let Z = (X, Y). If Y = 0, then set $A = \sqrt{|X|}$ and B = 0. Otherwise, compute $R = \sqrt{\frac{|X| + |Z|}{2}}$

   and set $A = +R$ if either $X \geq 0$ or $Y \geq 0$,
   or $A = -R$ if both $X < 0$ and $Y < 0$.

   Compute $B = \frac{Y}{2*A}$. Then $\sqrt{Z} = (A, B)$ if $X \geq 0$,
   or $\sqrt{Z} = (B, A)$ if $X < 0$.

2. Z and $\sqrt{Z}$ are complex numbers; values of X and Y range from -2^{127} to $2^{127}-2^{100}$ inclusive.

3. $\sqrt{Z}$ is accurate to 8 decimal positions.

III. USAGE

1. Calling Sequence--CALL CSQRT (Z)

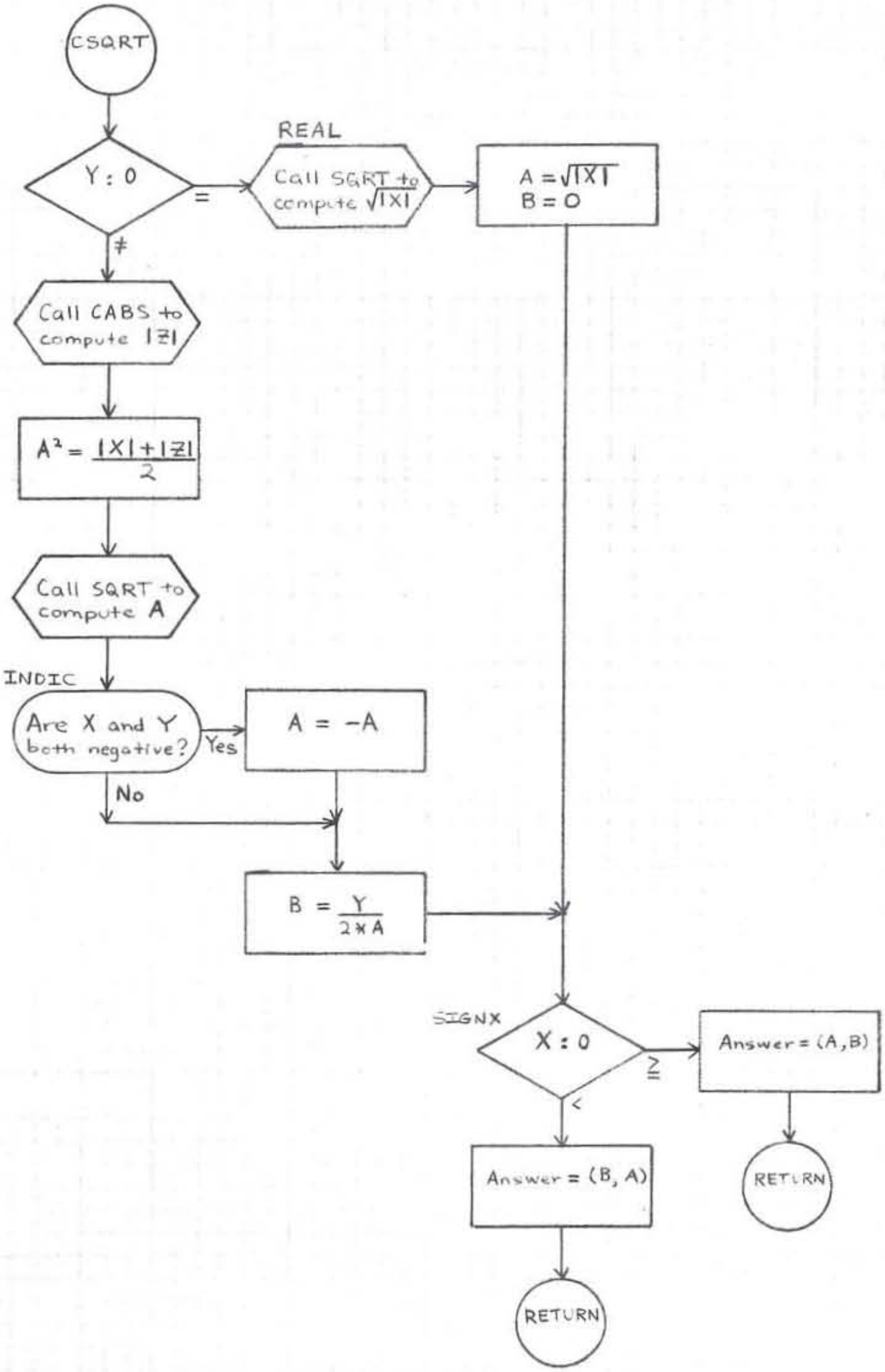
2. CSQRT uses ~~50~~ 52_{10} words. 64_8

3. No error conditions.

IV. RESTRICTIONS

The subprograms FCAB and FSQR must be in memory.

COMPUTE $\sqrt{Z}$ FOR COMPLEX Z = (X, Y)



FCSC--COMPLEX SINE AND COSINE

I. PURPOSE

To compute sin Z or cos Z for CSIN(Z) or CCOS(Z) in an expression, where Z is in radians.

II. METHOD

1. Sin Z = sin (X, Y)   (where Z = (X, Y))
   = sin X * cos (0, Y) + cos X * sin (0, Y)
   = (sin X * cosh Y, 0) + (0, cos X * sinh Y)
   = (sin X * cosh Y, cos X * sinh Y)

2. $\cos Z = \sin (Z + \frac{\pi}{2})$

3. Z, sin Z, and cos Z are complex numbers, with $|X| < 2^{27}$, $|X + \frac{\pi}{2}| < 2^{27}$, and $|Y| < 88.028$.

4. The answer is accurate to 7 decimal positions.

III. USAGE

1. Calling Sequence--CALL CSIN(Z) for sin Z
   CALL CCOS(Z) for cos Z

2. FCSC uses 72 words.

3. The error conditions are:

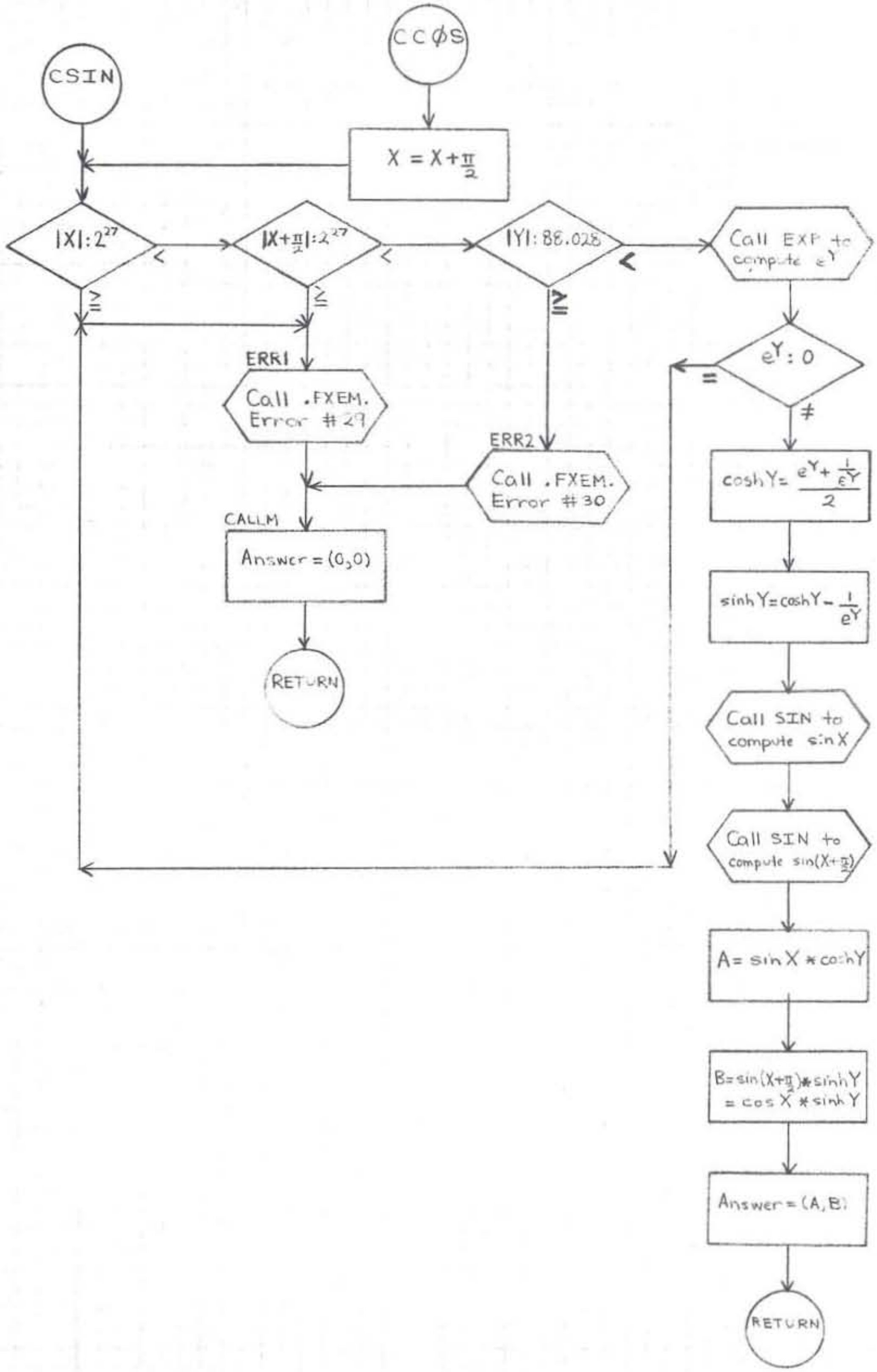
   a. FXEM Error #29 if $|X| \geqq 2^{27}$, $\left|X + \frac{\pi}{2}\right| \geqq 2^{27}$, or $e^{Y} = 0$. Then the answer is (0, 0).

   b. FXEM Error #30 if $|Y| > 88.028$. Then the answer is (0, 0).

IV. RESTRICTIONS

The subprograms FXPF, FSCN, and FXEM must be in memory.

COMPUTE SIN Z OR COS Z FOR COMPLEX Z = (X, Y)



FXPF--REAL NATURAL EXPONTIAL

PURPOSE

To compute e^X for EXP(X) in an expression.

METHOD

1. $e^X = 2^{\log_2 e^X} = 2^{X*\log_2 e} = 2^{I+F} = 2^I * 2^F$, where
   I = greatest integer $\leq X*\log_2 e$ and $I+F = X*\log_2 e$

2. Then $2^F = 1 + \cfrac{2*F}{D + C*F^2 - F + \cfrac{B}{F^2 + A}}$

   where A = 87.417497202
   B = -617.9722695
   C = 0.03465735903
   and D = 9.9545957821

3. X and e^X are real numbers; $|X| \leq 88.028$

4. e^X is accurate to 8 decimal positions.

USAGE

1. Calling Sequence--CALL EXP(X)

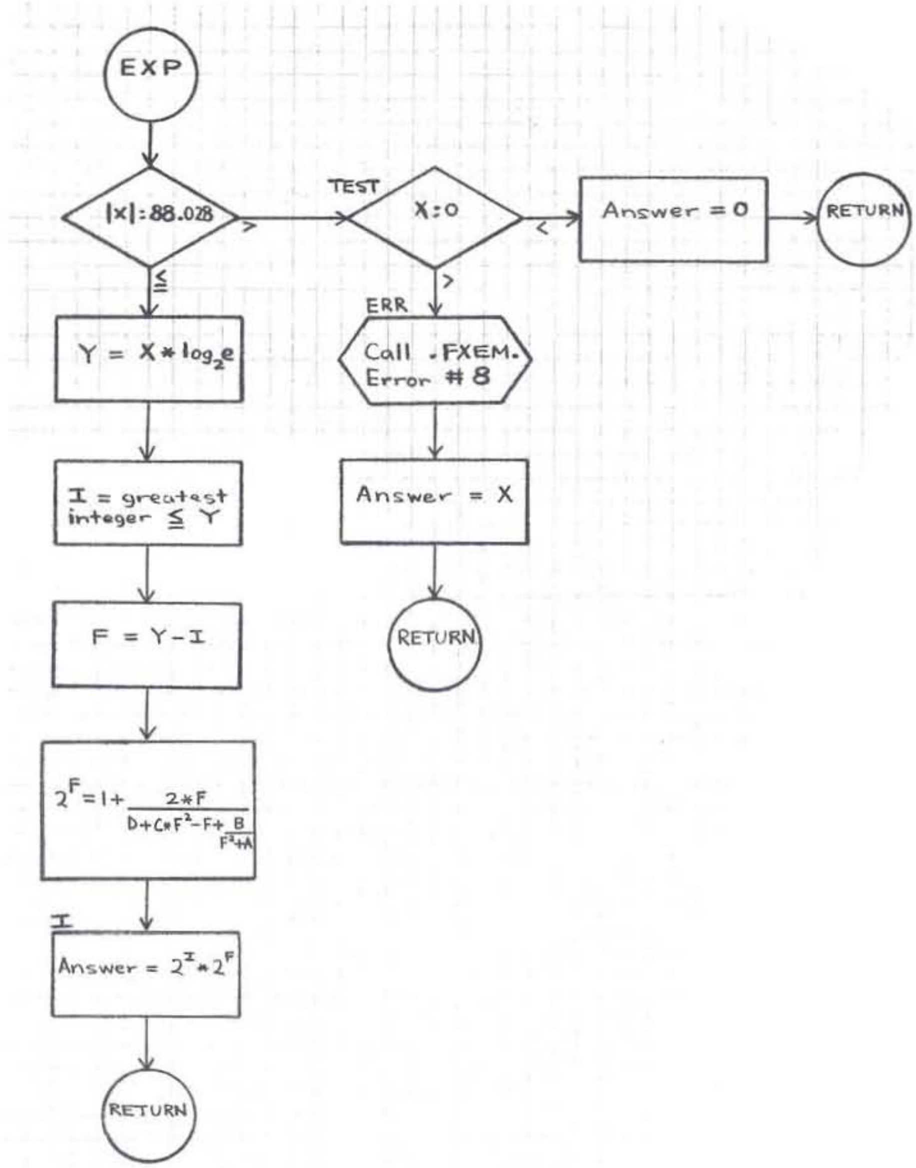
2. FXPF uses 56 words.

3. The error condition is:
   FXEM Error #8 if $|X| > 88.028$. Then if $X > 88.028$, $e^X = X$
   if $X < -88.028$, $e^X = 0$

RESTRICTIONS

The subprogram FXEM must be in memory.

COMPUTE e^X FOR REAL X



FLOG--REAL LOGARITHM

PURPOSE

To compute $\log_e X$ for ALOG(X) and $\log_{10} X$ for ALOG10(X)^e.

METHOD

1. $\log_2 X = \log_2 (2^I * F) = I + \log_2 F$, where $X = 2^I * F$.

2. $\log_e X = \log_e 2^{(\log_2 X)} = (\log_2 X) * (\log_e 2)$, and similarly $\log_{10} X = (\log_2 X) * (\log_{10} 2)$.

3. $\log_2 X = Z * \left(A + \frac{B}{Z^2 - C} \right) - 1/2$, where $Z = \frac{F - \frac{\sqrt{2}}{2}}{F + \frac{\sqrt{2}}{2}}$ and

   A = 1.2920070987
   B = -2.6398577031
   C = 1.6567626301

4. X and log X are real numbers; values of X range from 2^{-129} to $2^{127} - 2^{100}$ inclusive.

5. log X is accurate to 8 decimal places.

USAGE

1. Calling Sequence--CALL ALOG(X) for $\log_e X$
   CALL ALOG10(X) for $\log_{10} X$

2. FLOG uses 64 words.

3. The error conditions are:

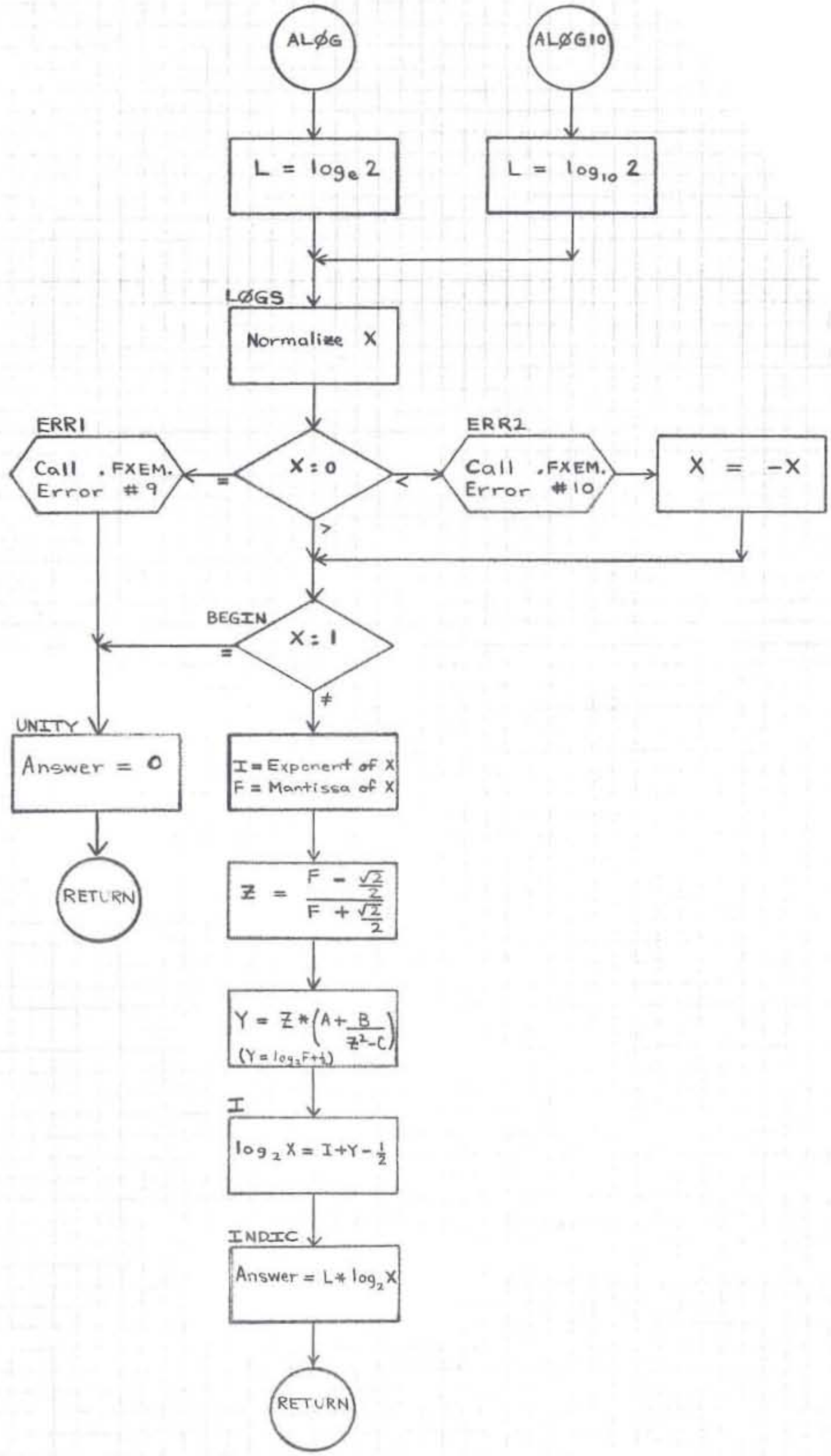
   a. FXEM Error #9 if X = 0. Then log X = 0.

   b. FXEM Error #10 if X < 0. Then log X = log |X| .

RESTRICTIONS

The subprogram FXEM must be in memory.

COMPUTE $\log_e X$ OR $\log_{10} X$ FOR REAL X



FATN--REAL ARCTANGENT

PURPOSE

To compute the principal value of arctan X or arctan $\frac{Y}{Z}$ (in radians) for ATAN(X) or ATAN2(Y,Z) in an expression.

METHOD

1. Compute arctan X as follows:

   a. For positive values of X:

      (1) If $X \geqq 2^{27}$, then arctan $X = \frac{\pi}{2}$.

      (2) If $X < 2^{-27}$, then arctan X = X.

      (3) If $2^{-27} \leqq X < 2^{27}$, then set I from the interval containing arctan X:

      0 for 0°-10°, 1 for 10°-30°, 2 for 30°-50°, 3 for 50°-70°, and 4 for 70°-90°.

      Then $\arctan X = N_I + \cfrac{T}{T^2 + K_3 + \cfrac{K_2}{T^2 + K_1}}$

      where $T = \frac{9}{55} * X$ for I = 0

      $T = A_I + \frac{B_I}{X + C_I}$ for I = 1,2,3,4

      and the constants $A_I, B_I, C_I, N_I, K_1, K_2, K_3$ assume appropriate values.

   b. For negative values of X, arctan X = -arctan |X| .

2. Compute arctan $\frac{Y}{Z}$ as follows:

   a. If Z = 0, then arctan $\frac{Y}{Z} = \frac{\pi}{2}$ for Y> 0,

      and arctan $\frac{Y}{Z} = -\frac{\pi}{2}$ for Y < 0.

   b. If $Z \neq 0$, then $X = \frac{Y}{Z}$; compute arctan X as above.

   c. If Z> 0, then arctan $\frac{Y}{Z}$ = arctan X.

   d. If Z< 0, then arctan $\frac{Y}{Z}$ = arctan X+π for Y>0,

      and arctan $\frac{Y}{Z}$ = arctan X-π for Y<0.

3. X, Y, and Z are real numbers with values from -2^{127} to $2^{127}-2^{100}$ inclusive. The result is a real number.

4. The answer is accurate to 8 decimal positions.

USAGE

1. Calling Sequence--CALL ATAN(X) for arctan X
CALL ATAN2(Y, Z) for arctan $\frac{Y}{Z}$

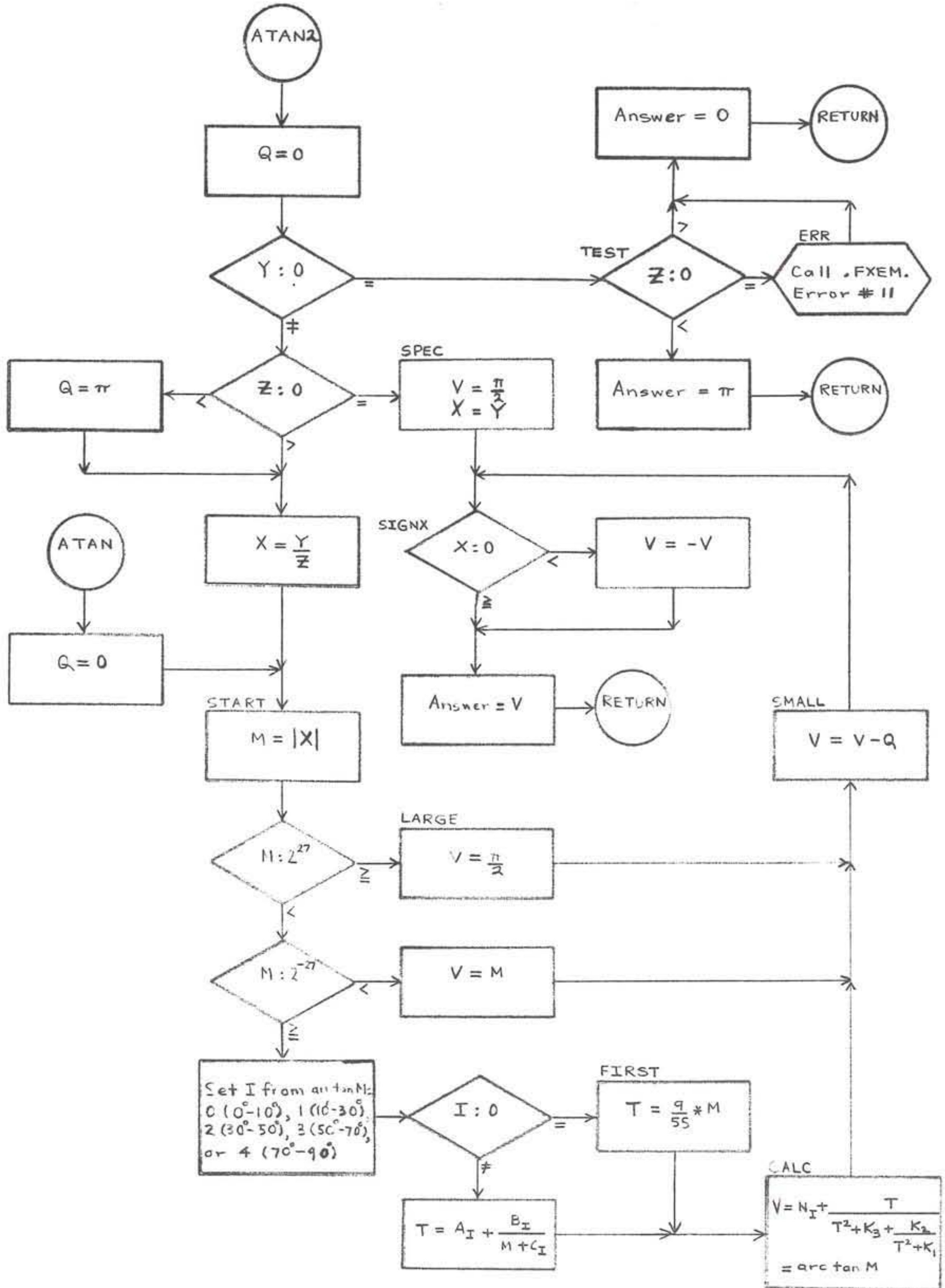
2. FATN uses 98 words.

3. The error condition is:

   FXEM Error #11 if Y = 0 and Z = 0.
   Then arctan $\frac{Y}{Z} = 0$.

RESTRICTIONS

The subprogram FXEM must be in memory.


ATAN2
Q=0
Y : 0
=
≠
TEST
Z:0
>
=
<
ERR
Call .FXEM. Error # 11
Answer = 0
RETURN
Answer = π
RETURN
Q = π
<
Z : 0
=
>
SPEC
V = π/2
X = Y
X = Y/Z
ATAN
Q = 0
SIGNX
X : 0
<
≧
V = -V
Answer = V
RETURN
START
M = |X|
M : 2^27
≧
<
LARGE
V = π/2
M : 2^-27
<
≧
V = M
SMALL
V = V-Q
Set I from arctan M: 0 (0°-10°), 1 (10°-30°), 2 (30°-50°), 3 (50°-70°), or 4 (70°-90°)
I : 0
=
≠
FIRST
T = 9/55 * M
T = A_I + B_I / (M + C_I)
CALC
V = N_I + T / (T²+K_3 + K_2/(T²+K_1))
= arc tan M

FSCN--REAL SINE AND COSINE

PURPOSE

To compute sin X or cos X for SIN(X) or COS(X) in an expression, where X is in radians.

METHOD

1. Compute sin X as follows:

   a. For positive values of X:

      (1) Replace the value of X by the value of X - I*π, where I is the greatest integer $\leq \frac{X}{\pi}$, noting that $\sin(X + n\pi) = (-1)^n * \sin X$. Now if $X \geq \frac{\pi}{2}$, then replace the value of X by the value of π - X and proceed, noting that $\sin(\pi - X) = \sin X$.

      (2) If $X \leq 2^{-8}$, then sin X = X. If $2^{-8} < X \leq 0.3$, then $\sin X = X * (A_1 + \frac{B_1}{C_1 + X^2} + \frac{X^2}{4})$.
      If $X \geq 0.3$, then replace the value of X by the value of $\frac{\pi}{2}$ - X, noting that $\cos(\frac{\pi}{2} - X) = \sin(X)$. Then compute
      $$\cos(X) = D - 2 * X^2 + \frac{E - 320 * X^2}{A_2 + \frac{B_2}{C_2 + X^2} + X^2}$$

   b. For negative values of X, sin X = - sin |X| .

2. Compute $\cos X = \sin(X + \frac{\pi}{2})$.

3. X, sin X, and cos X are real numbers, with $|X| < 2^{27}$.

4. The answer is accurate to 8 decimal positions.

USAGE

1. Calling Sequence--CALL SIN(X) for sin X
   CALL COS(X) for cos X

2. FSCN uses 100 words.

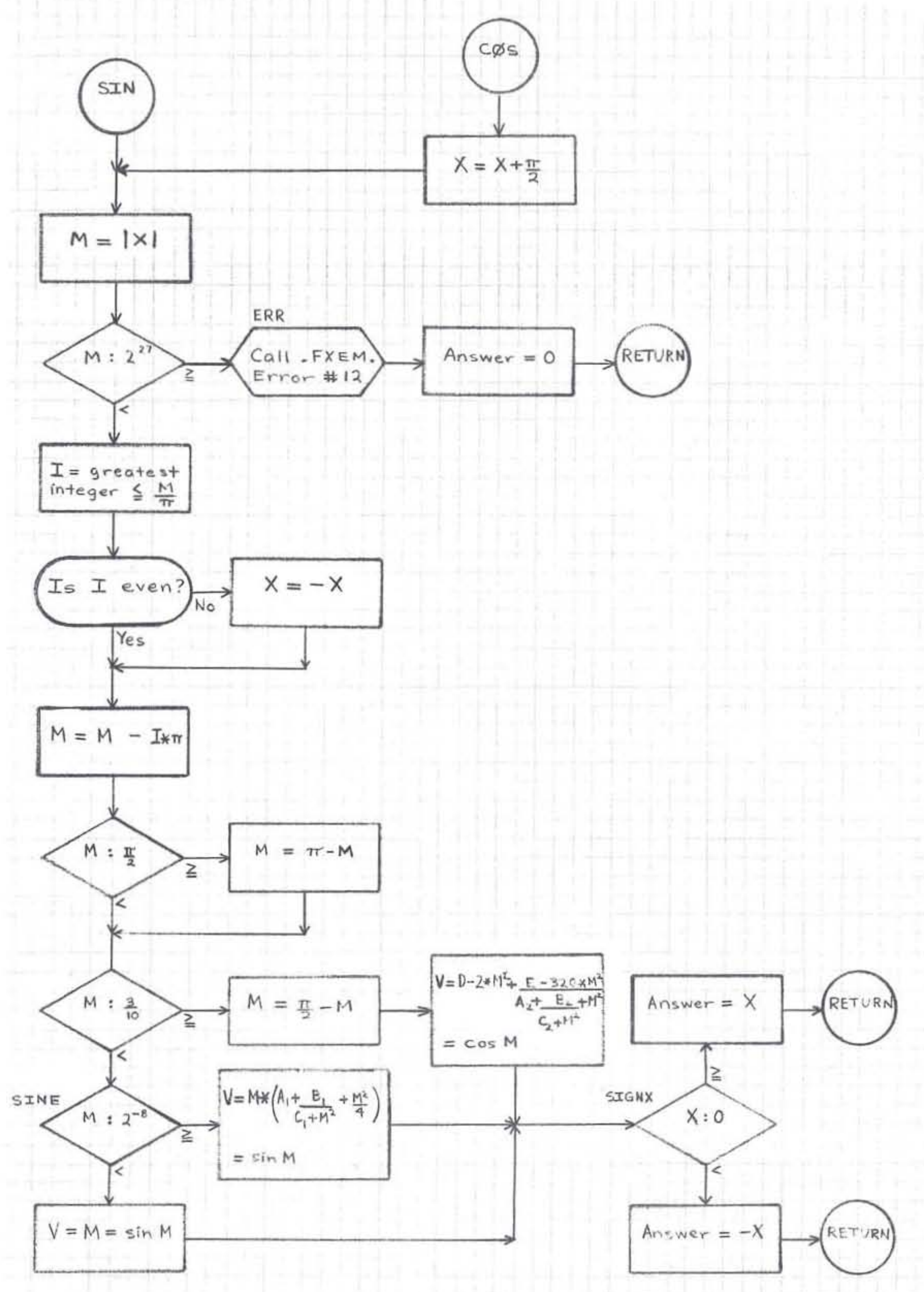
3. The error condition is:

   FXEM Error # 12 if $|X| \geqq 2^{27}$.
   Then the answer is 0.

RESTRICTIONS

The subprogram FXEM must be in memory.

COMPUTE SIN X OR COS X FOR REAL X



FTNH--REAL HYPERBOLIC TANGENT

PURPOSE

To compute tanh X for TANH(X) in an expression.

METHOD

1. For positive values of X:

   a. If $X > 12$, then $\tanh X = 1$

   b. If $0.17 \leqq X < 12$, then $\tanh X = \frac{e^{2*X}-1}{e^{2*X}+1}$

   c. If $0.00034 \leqq X < 0.17$, then $\tanh X = \frac{F}{A+F^2*\left(B+\frac{C}{D+F^2}\right)}$

   where F = A * X
   A = 5.77078016
   B = 0.0173286795
   C = 14.1384114
   and D = 349.669989

   d. If $X < 0.00034$, then $\tanh X = X$

2. For negative values of X, $\tanh X = -\tanh |X|$.

3. X and tanh X are real numbers; values of X range from -2^{127} to $2^{127}-2^{100}$ inclusive.

4. tanh X is accurate to 8 decimal positions.

USAGE

1. Calling Sequence--CALL TANH(X)

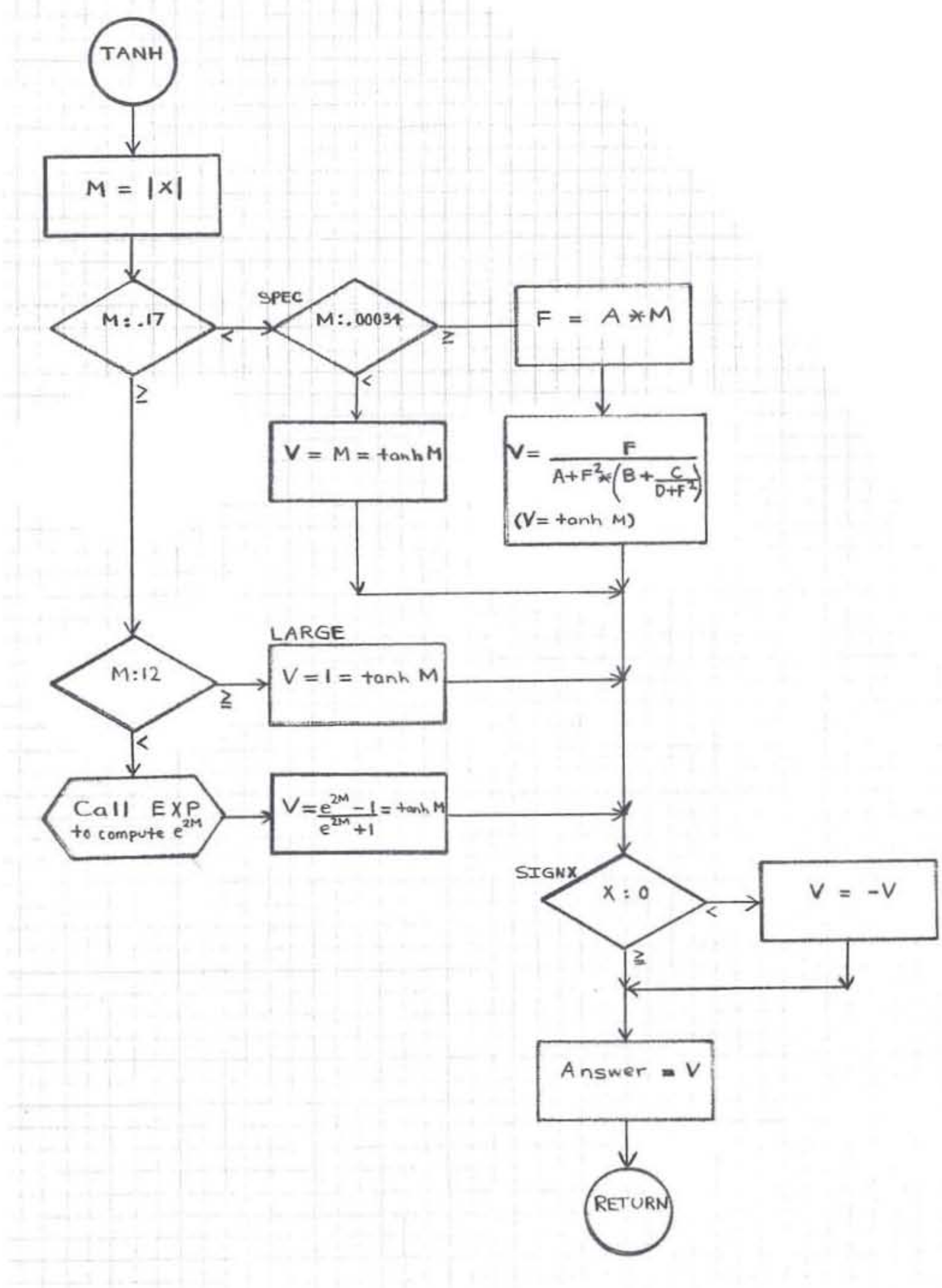
2. FTNH uses 48 words.

3. No error conditions.

RESTRICTIONS

The subprogram FXPF must be in memory.

COMPUTE TANH X FOR REAL X



FSQR--REAL SQUARE ROOT

PURPOSE

To compute $\sqrt{X}$ for SQRT(X) in an expression.

METHOD

1. If $X = 0$, then $\sqrt{X} = 0$. Otherwise, let $X = 2^{2*A} * B$,

   where $1/4 \leq B < 1$. Then $\sqrt{X} = 2^{A} * \sqrt{B} = 2^{A-1} * (2 * \sqrt{B})$.

   First Approximation: $P_o = 1/4 + B$ if $1/4 \leq B < 1/2$,

   or $P_o = 1/2 + B/2$ if $1/2 \leq B < 1$.

   Then $P_1 = \frac{1}{2} * (P_o + \frac{B}{P_o})$, $P_2 = \frac{1}{2} * (P_1 + \frac{B}{P_1})$, and finally

   $\sqrt{X} = 2^{A-1} * (P_2 + \frac{B}{P_2})$.

2. X and $\sqrt{X}$ are real numbers; values of X range from -2^{127} to $2^{127} - 2^{100}$ inclusive.

3. $\sqrt{X}$ is accurate to 8 decimal positions. The last iteration is in double precision.

USAGE

1. Calling Sequence--CALL SQRT(X)

2. FSQR uses 46 words.

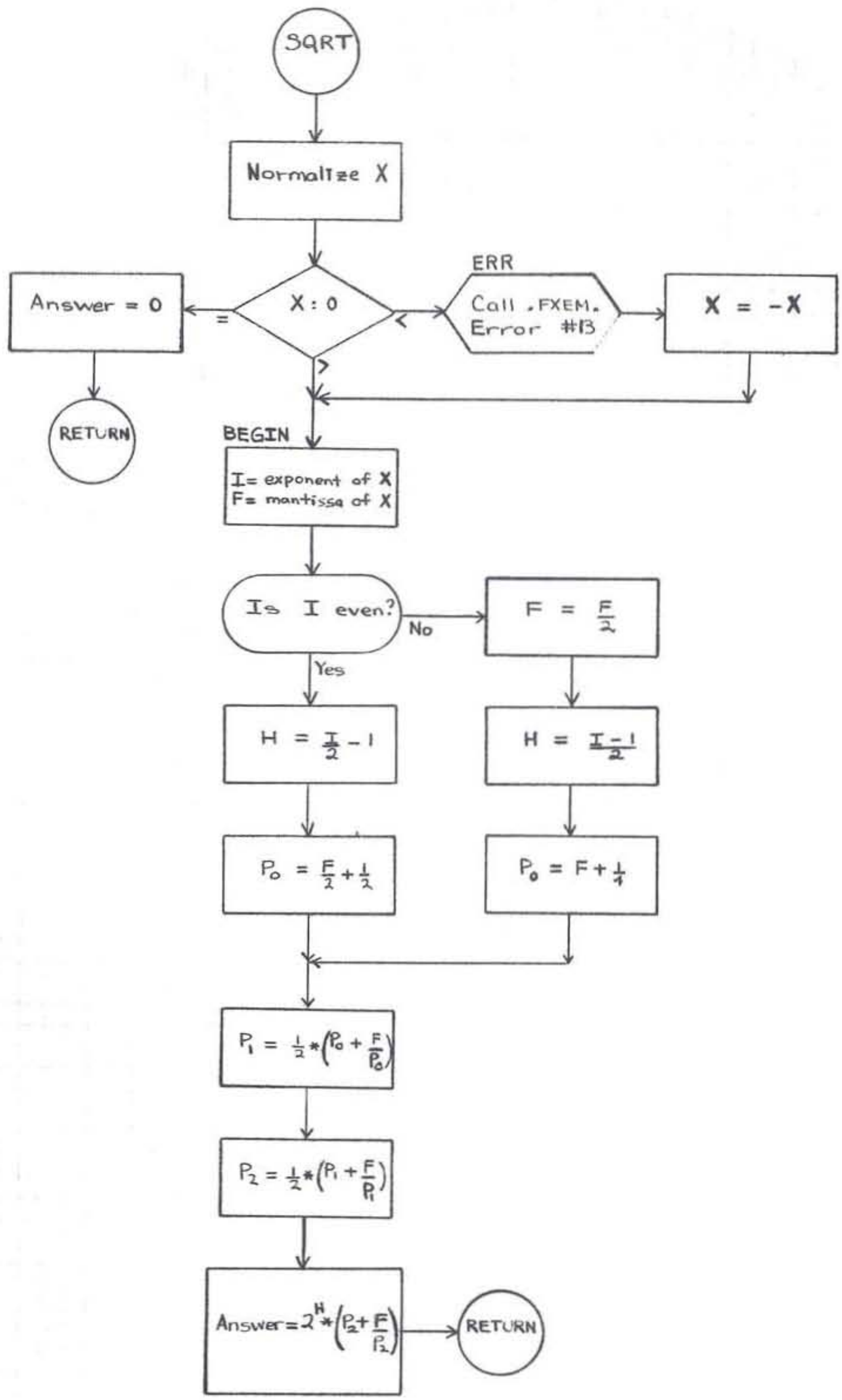
3. The error condition is:

   FXEM Error #13 if $X < 0$. Then $\sqrt{X} = \sqrt{|X|}$.

RESTRICTIONS

The subprogram FXEM must be in memory.

COMPUTE $\sqrt{X}$ FOR REAL X



UNIFORM RANDOM NUMBER GENERATOR (URAN)

PURPOSE

Uniform Random Number Generator (URAN) creates a list of random numbers between 0 and 1.0 with a period of 2^{33}. 2^{33} (was 2^{35})

DESCRIPTION

URAN uses a mixed congruential method:

$$R_{n+1} = (\alpha + 1)R_n + \beta$$

The values of $\alpha = 2^9$ and $\beta = 262035034724_8$ chosen have been shown experimentally to yield good results.

CALLING SEQUENCE

The following calling sequence is used:

CALL URAN(START, N, RANDOM)

where: START is a starting value (changed by URAN to prevent repetition on subsequent calls).

N is the number of values to be generated.

RANDOM is the name of a vector in which random numbers will be stored.

requires 44_8 words (36_{10})

FCAS--COMPLEX MULTIPLICATION AND DIVISION

PURPOSE

To compute (A, B) * (C, D) or $\frac{(A, B)}{(C, D)}$ in an expression.

METHOD

1. (A, B) * (C, D) = (A * C - B * D, A * D+B * C)

2. $\frac{(A, B)}{(C, D)} = \frac{(A, B) * (C, -D)}{C^2 + D^2} = \frac{(A * C + B * D,\ B * C - A * D)}{C^2 + D^2}$

3. If (A, B) = (0, 0), then the quotient = (0, 0). Otherwise, divide the numerator by A * C and the denominator by C^2: $\frac{(A, B)}{(C, D)} = \frac{\frac{A}{C}}{1 + \left(\frac{D}{C}\right)^2} * \left(1 + \frac{B}{A} * \frac{D}{C}, \frac{B}{A} - \frac{D}{C}\right)$, where $\frac{A}{C} = \frac{A * C}{C^2}$.

4. Before computing $\frac{(A, B)}{(C, D)}$, replace the numerator by (-B, A) if $|A| \leq |B|$, and the denominator by (-D, C) if $|C| \leq |D|$. Adjust the quotient (X, Y) accordingly:

   a. If $|A| > |B|$ and $|C| > |D|$, then the result = (X, Y).
   b. If $|A| > |B|$ and $|C| \leq |D|$, then the result = (-Y, X).
   c. If $|A| \leq |B|$ and $|C| > |D|$, then the result = (Y, -X).
   d. If $|A| \leq |B|$ and $|C| \leq |D|$, then the result = (X, Y).

5. A, B, C, D, X, and Y are real numbers, with values from -2^{127} to $2^{127}-2^{100}$ inclusive.

6. The answer is accurate to 8 decimal positions.

USAGE

1. Calling Sequence--CALL .FCFMP (R, S) for R * S
   CALL .FCFDP (R, S) for R/S
   where R = (A, B) and S = (C, D)

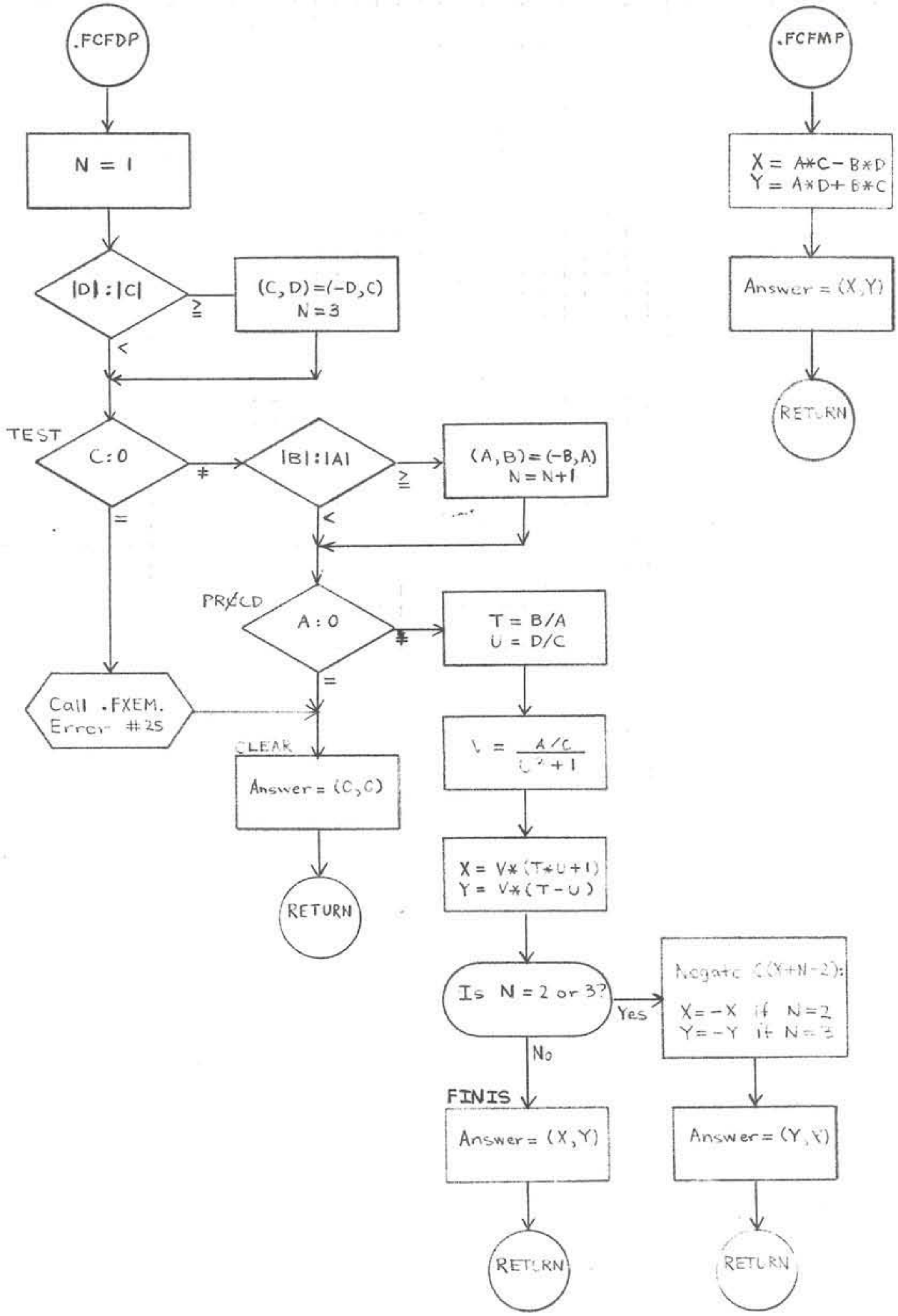
2. FCAS uses 94 words.

3. The error condition is only in division:

   FXEM Error #25 if (C, D) = (0, 0). Then $\frac{(A, B)}{(C, D)} = (0, 0)$.

RESTRICTIONS

The subprogram FXEM must be in memory.


.FCFDP
N = 1
|D| : |C|
≧
(C, D) = (−D, C)
N = 3
<
TEST
C : 0
≠
=
|B| : |A|
≧
(A, B) = (−B, A)
N = N+1
<
PRØCD
A : 0
≠
=
T = B/A
U = D/C
Call .FXEM.
Error #25
CLEAR
Answer = (C, C)
RETURN
A/C
X = V*(T*U+1)
Y = V*(T−U)
Is N = 2 or 3?
Yes
No
Negate
X = −X if N = 2
Y = −Y if N = 3
FINIS
Answer = (X, Y)
Answer = (Y, X)
RETURN
RETURN
.FCFMP
X = A*C − B*D
Y = A*D + B*C
Answer = (X, Y)
RETURN

FCAB--COMPLEX ABSOLUTE VALUE

PURPOSE

To compute $|Z|$ for CABS(Z) in an expression.

METHOD

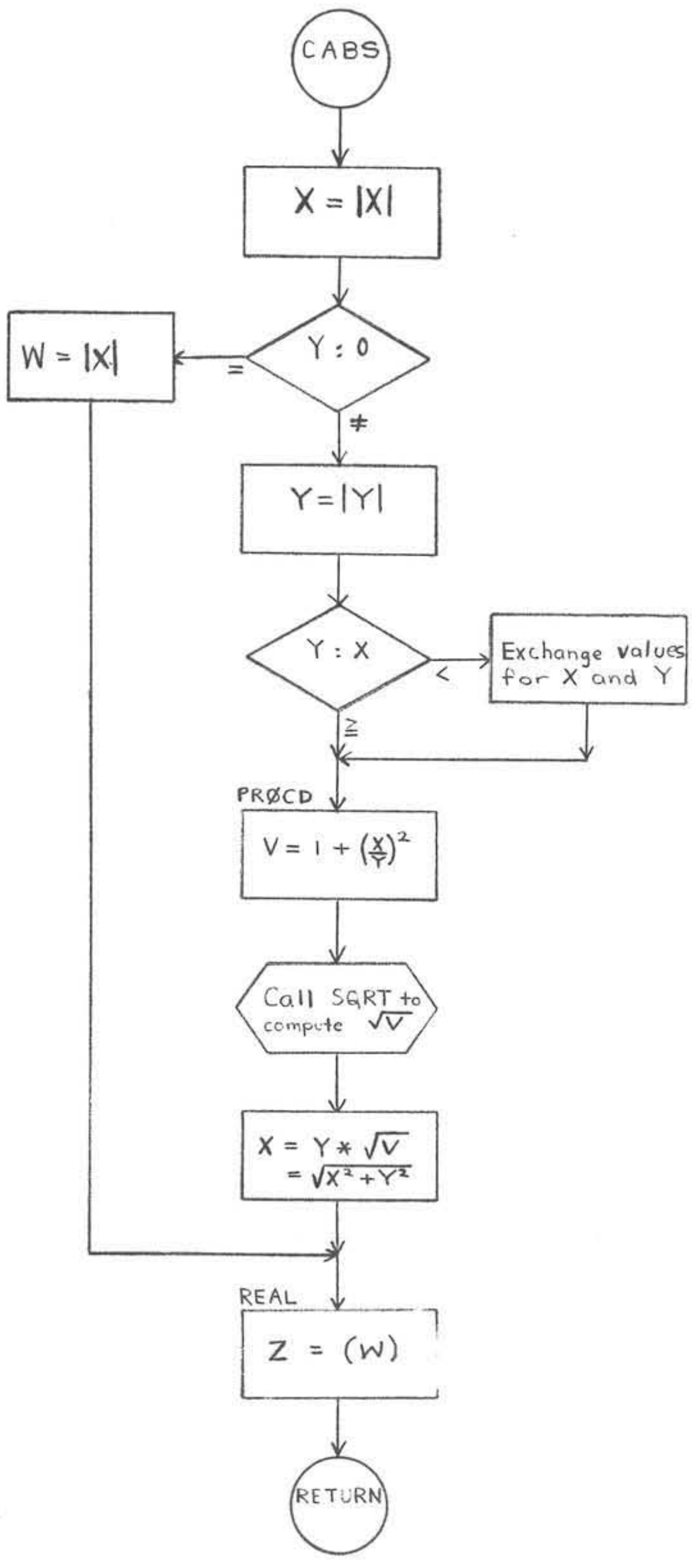
1. Compute $W = \sqrt{X^2 + Y^2}$ (where Z = (X, Y)) as follows:
   a. If Y = 0, then W = |X|
   b. If $Y \neq 0$ and $|X| \leqq |Y|$, then $W = |Y| * \sqrt{1 + \left(\frac{X}{Y}\right)^2}$
   c. If $Y \neq 0$ and $|X| > |Y|$, then $W = |X| * \sqrt{1 + \left(\frac{Y}{X}\right)^2}$
2. |Z| = (W) returns a real value in EAQ
3. W, X, and Y are real numbers; values of X and Y range from -2^{127} to $2^{127}-2^{100}$ inclusive. Therefore Z is a complex number, but |Z| is a real number.
4. W is accurate to 8 decimal positions.

USAGE

1. Calling Sequence--CALL CABS (Z)
2. CABS uses 36 words.
3. No error conditions.

RESTRICTIONS

The subprogram FSQR must be in memory.


CABS
X = |X|
Y : 0
=
≠
W = |X|
Y = |Y|
Y : X
<
≥
Exchange values for X and Y
PRØCD
V = 1 + (X/Y)²
Call SQRT to compute √V
X = Y * √V
= √(X² + Y²)
REAL
Z = (W)
RETURN

FCXP--COMPLEX EXPONENTIAL

PURPOSE

To compute e^Z for CEXP(Z) in an expression.

METHOD

1. $e^Z = e^{(X, Y)}$, where Z = (X, Y)

   $= e^X * e^{(0, Y)}$

   $= e^X * (\cos Y, \sin Y)$

   $= \left[e^X * \sin (Y + \frac{\pi}{2}), e^X * \sin Y\right]$

2. Z and e^Z are complex numbers, with

   $X \leq 88.028$, $|Y| < 2^{27}$, and $|Y + \frac{\pi}{2}| < 2^{27}$.

3. e^Z is accurate to 7 decimal positions.

USAGE

1. Calling Sequence--CALL CEXP (Z)

2. CEXP uses 54 words.

3. The error conditions are:

   a. FXEM Error #26 if $X > 88.028$. Then $e^Z = Z$.

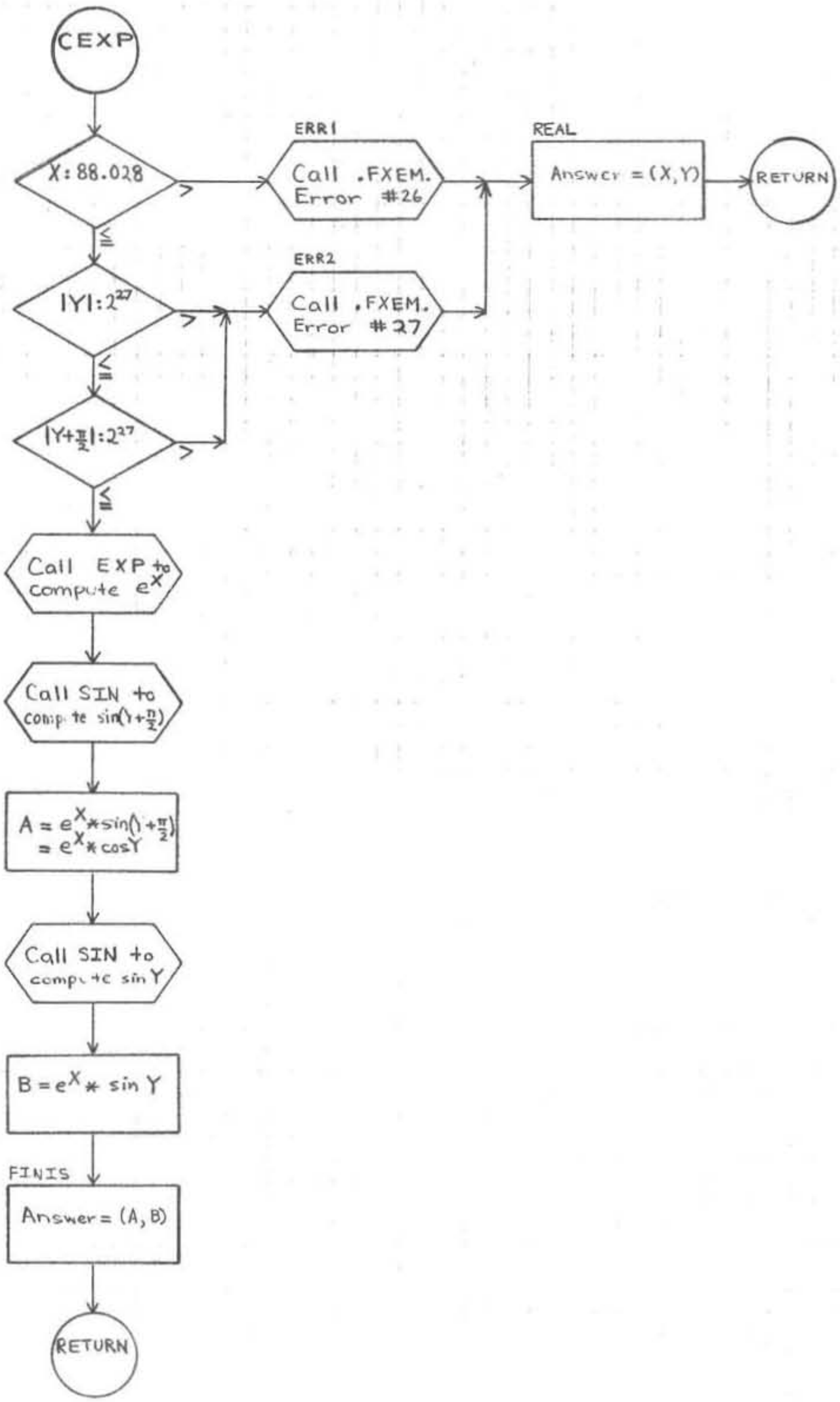
   b. FXEM Error #27 if $|Y| \geq 2^{27}$

      or if $|Y + \frac{\pi}{2}| \geq 2^{27}$. Then $e^Z = Z$.

RESTRICTIONS

The subprograms FXPF, FSCN, and FXEM must be in memory.

COMPUTE e^Z FOR COMPLEX Z = (X, Y)



FCLG--COMPLEX LOGARITHM

PURPOSE

To compute $\log_e Z$ for CLOG(Z) in an expression

METHOD

1. $\log_e Z = \log_e (X, Y)$ (where Z = (X, Y))

   $= (\log |Z| \ , \ \arctan \frac{Y}{X})$

2. Z and $\log_e Z$ are complex numbers; values of X and Y range from -2^{127} to $2^{127}-2^{100}$ inclusive.

3. $\log_e Z$ is accurate to 7 decimal positions.

USAGE

1. Calling Sequence--CALL CLOG(Z)

2. FCLG uses 40 words.

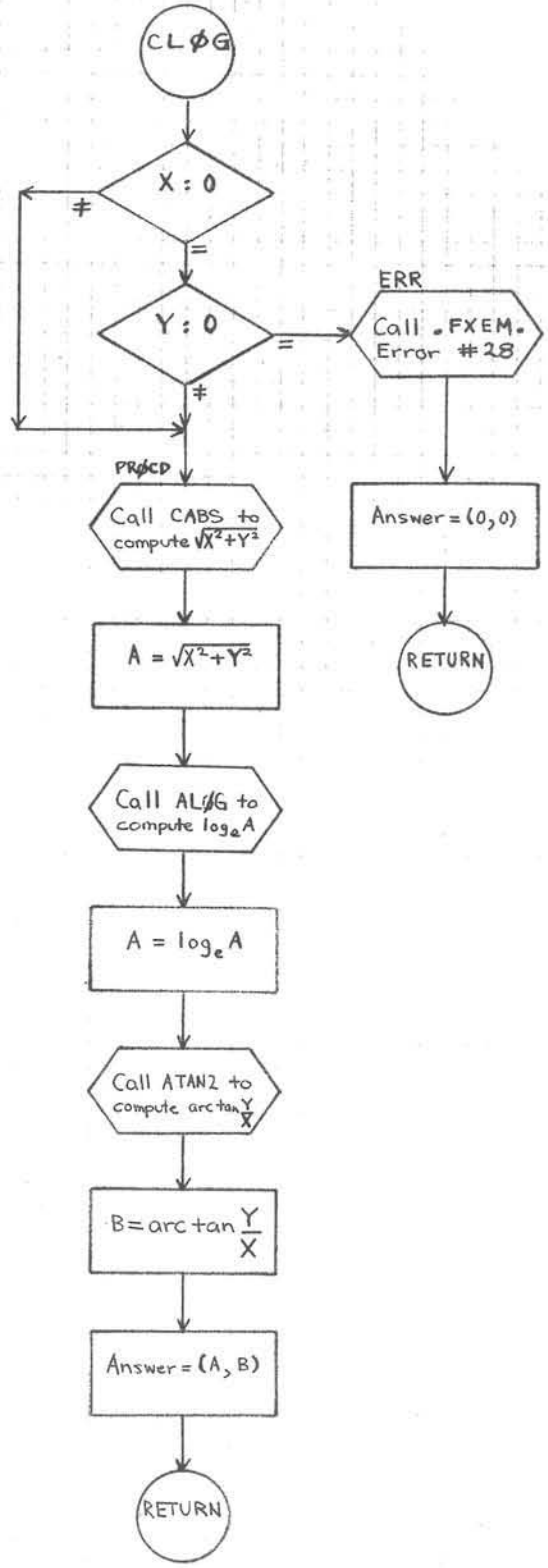
3. The error condition is:

   FXEM Error #28 if Z = (0, 0). Then $\log_e Z = (0, 0)$.

RESTRICTIONS

The subprograms FATN, FCAB, FLOG, and FXEM must be in memory.

COMPUTE $\log_e Z$ FOR COMPLEX Z = (X, Y)



FCSQ--COMPLEX SQUARE ROOT

PURPOSE

To compute $\sqrt{Z}$ for CSQRT(Z) in an expression.

METHOD

1. Let Z = (X, Y). If Y = 0, then set $A = \sqrt{|X|}$ and B = 0. Otherwise, compute $R = \sqrt{\dfrac{|X| + |Z|}{2}}$

   and set $A = +R$ if either $X \geq 0$ or $Y \geq 0$,
   or $A = -R$ if both $X < 0$ and $Y < 0$.

   Compute $B = \dfrac{Y}{2*A}$. Then $\sqrt{Z} = (A, B)$ if $X \geq 0$,
   or $\sqrt{Z} = (B, A)$ if $X < 0$.

2. Z and $\sqrt{Z}$ are complex numbers; values of X and Y range from -2^{127} to $2^{127}-2^{100}$ inclusive.

3. $\sqrt{Z}$ is accurate to 8 decimal positions.

USAGE

1. Calling Sequence--CALL CSQRT (Z)

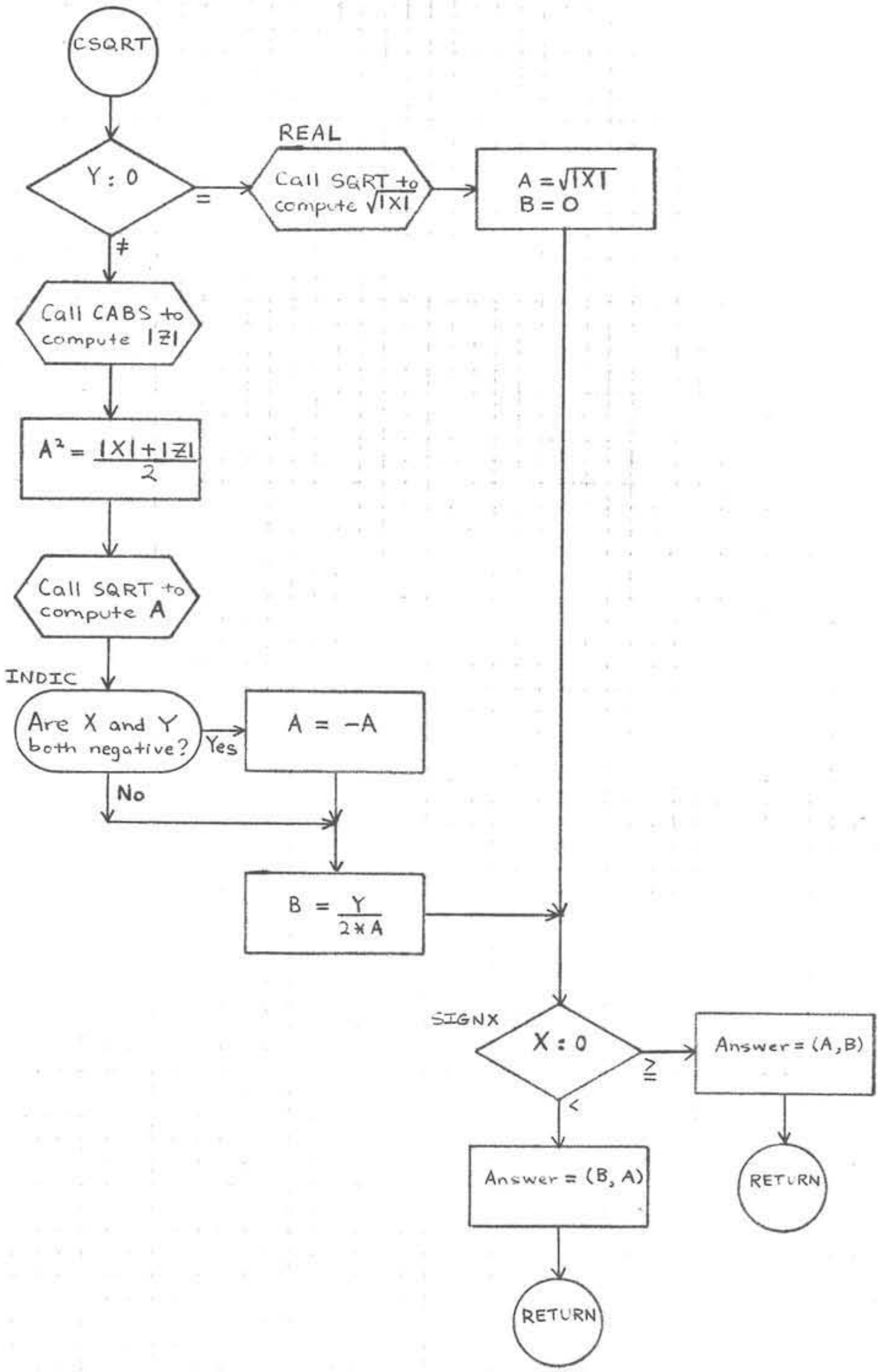
2. CSQRT uses 50 words.

3. No error conditions.

RESTRICTIONS

The subprograms FCAB and FSQR must be in memory.

COMPUTE $\sqrt{Z}$ FOR COMPLEX Z = (X, Y)



FCSC--COMPLEX SINE AND COSINE

PURPOSE

To compute sin Z or cos Z for CSIN(Z) or CCOS(Z) in an expression, where Z is in radians.

METHOD

1. Sin Z = sin (X, Y)   (where Z = (X, Y))
   = sin X * cos (0, Y) + cos X * sin (0, Y)
   = (sin X * cosh Y, 0) + (0, cos X * sinh Y)
   = (sin X * cosh Y, cos X * sinh Y)

2. $\cos Z = \sin (Z + \frac{\pi}{2})$

3. Z, sin Z, and cos Z are complex numbers, with $|X| < 2^{27}$, $|X + \frac{\pi}{2}| < 2^{27}$, and $|Y| < 88.028$.

4. The answer is accurate to 7 decimal positions.

USAGE

1. Calling Sequence--CALL CSIN(Z) for sin Z
   CALL CCOS(Z) for cos Z

2. FCSC uses 72 words.

3. The error conditions are:

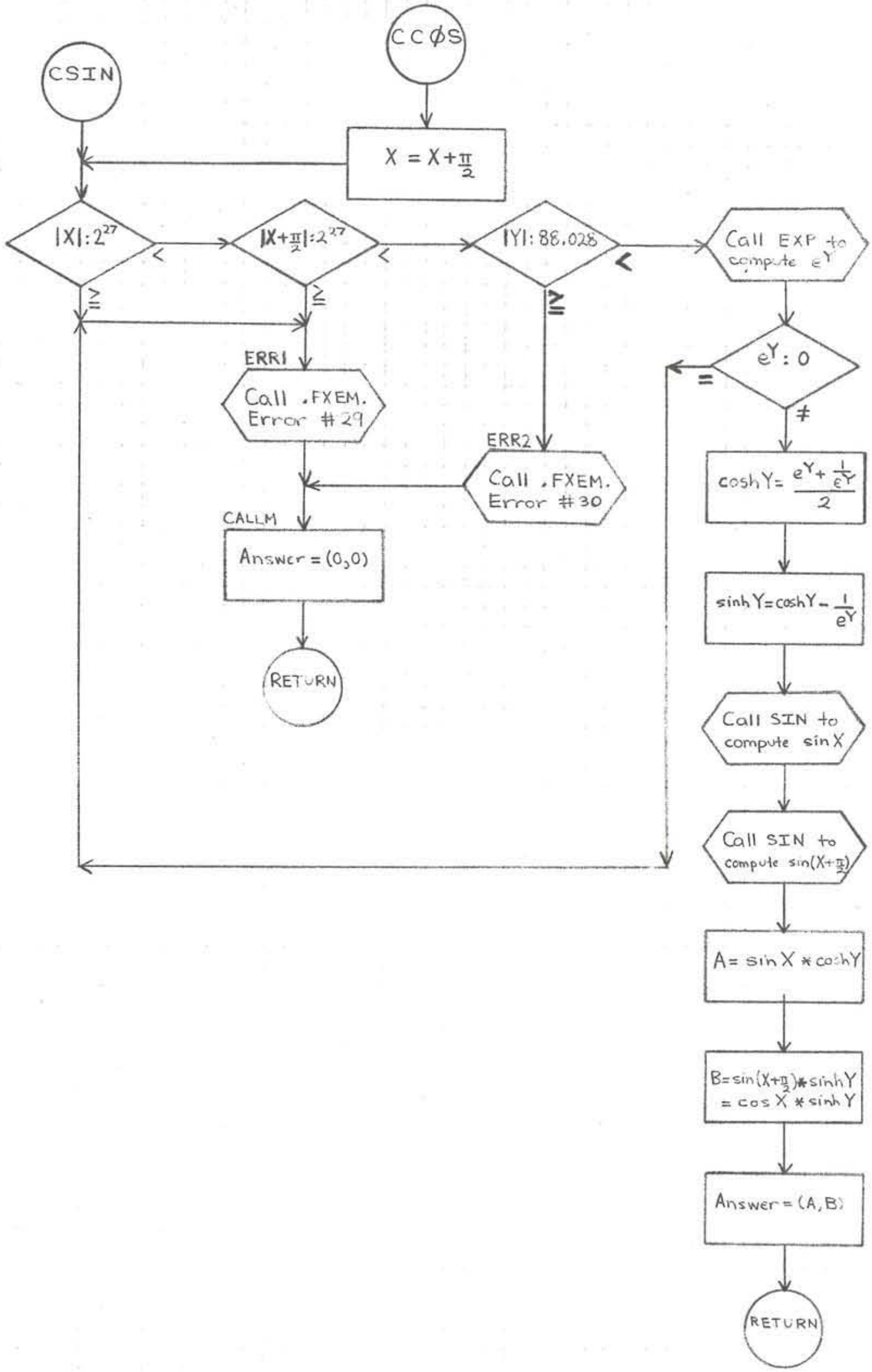
   a. FXEM Error #29 if $|X| \geqq 2^{27}$, $|X + \frac{\pi}{2}| \geqq 2^{27}$, or $e^Y = 0$. Then the answer is (0, 0).

   b. FXEM Error #30 if $|Y| > 88.028$.
      Then the answer is (0, 0).

RESTRICTIONS

The subprograms FXPF, FSCN, and FXEM must be in memory.

COMPUTE SIN Z OR COS Z FOR COMPLEX Z = (X, Y)



not in "A" version

MATRIX INVERSE ROUTINE (MINV)

PURPOSE

The Matrix Inverse Routine (MINV) obtains the explicit inverse of a matrix whose elements are FORTRAN REAL numbers.

DESCRIPTION

The inverse is computed by the method of decomposition and back-substitution outlined in SIMEQ: A Set of FORTRAN Subroutines for Solving Linear Algebraic Systems, CPB-1167. MINV calls three routines of SIMEQ (DECOM, INVRS, and SOLV).

CALLING SEQUENCE

The following calling sequence is used:

CALL MINV (A, IMAX, N, INTR, DET)

where: A is the matrix to be inverted; A is dimensioned (IMAX, IMAX) in a DIMENSION statement.

N is the actual number of rows and columns stored in A.

INTR is a working vector used by MINV; its dimension must be at least N.

DET upon return contains the fractional portion of the determinant, with an integer power of 10 in INTR(N), so that determinant = DET*10.0**(INTR(N)).

RESTRICTIONS

The inverse replaces the original contents of matrix A.

MATRIX ADDITION ROUTINE (MADD)

PURPOSE

The Matrix Addition routine (MADD) performs addition of two real matrices.

USAGE

The matrices A and B and the result C are assumed to be stored as i by j matrices in m by n arrays. Associated with each matrix is a dimension vector of four integers (i, j, m, n).

The calling sequence to MADD is:

CALL MADD (A,IA,B,IB,C,IC,IND) to calculate [C] = [A] + [B]

with IA, IB, and IC being the dimension vectors of A, B, and C, respectively.

IND is an error indicator set as follows:

- IND = 0 for correct results;
- IND = 1 if results would have been larger than m_c by n_c;
- IND = 2 if the dimensions are not consistent.

Consistent dimensions are $i_A = i_B = i_C$ and $j_A = j_B = j_C$.

MATRIX SUBTRACTION ROUTINE (MSUB)

PURPOSE

The Matrix Subtraction routine (MSUB) performs subtraction of two real matrices.

USAGE

The matrices A and B and the result C are assumed to be stored as i by j matrices in m by n arrays. Associated with each matrix is a dimension vector of four integers (i, j, m, n).

The calling sequence to MSUB is:

CALL MSUB (A,IA,B,IB,C,IC,IND) to calculate [C] = [A] - [B]

with IA, IB, and IC being the dimension vectors of A, B, and C, respectively.

IND is an error indicator set as follows:

- IND = 0 for correct results;
- IND = 1 if results would have been larger than m_c by n_c;
- IND = 2 if the dimensions are not consistent.

Consistent dimensions are $i_A = i_B = i_C$ and $j_A = j_B = j_C$.

MATRIX MULTIPLY ROUTINE (MMPY)

PURPOSE

The Matrix Multiply routine (MMPY) calculates the product of two real matrices.

USAGE

The matrices A and B and the result C are assumed to be stored as i by j matrices in m by n arrays. Associated with each matrix is a dimension vector of four integers (i, j, m, n).

The calling sequence to MMPY is:

CALL MMPY (A,IA,B,IB,C,IC,IND) to calculate $[C] = [A] \times [B]$

with IA, IB, and IC being the dimension vectors of A, B, and C, respectively.

IND is an error indicator set as follows:

- IND = 0 for correct results;
- IND = 1 if results would have been larger than m_c by n_c;
- IND = 2 if the dimensions are not consistent.

Consistent dimensions are $j_A = i_B$, $i_C = i_A$, and $j_C = j_B$.

B.J.

MATRIX TRANSPOSE ROUTINE (MTRN)

PURPOSE

The Matrix Transpose routine (MTRN) transposes a matrix.

USAGE

The matrix A and its transpose C are stored as i by j matrices in m by n arrays. Associated with each matrix is a dimension vector of four integers (i, j, m, n).

The calling sequence to MTRN is

CALL MTRANS (A, IA, C, IC, IND) to form $[C] = [A]^T$

with IA and IC being the dimension vectors of A and C, respectively.

IND is an error indicator set as follows:

- IND = 0 for correct results;
- IND = 1 if results would not have fit within an m_c by n_c array.
- IND = 2 if the dimensions are not consistent.

Consistent dimensions are $i_A = j_c$ and $j_c = i_c$.

MATRIX MOVE ROUTINE (MMOV)

PURPOSE

The Matrix Move routine (MMOV) moves a submatrix to another matrix.

USAGE

The sending matrix A and the receiving matrix C are stored as i by j matrices in m by n arrays. Associated with each matrix is a dimension vector of four integers (i, j, m, n).

The calling sequence to MMOV is:

CALL MMOV (A, IA, C, IC, IS, JS, IR, JR, I, J, IND) to move an I by J submatrix whose upper left element is $A_{IS, JS}$ into an area whose upper left element is $C_{IR, JR}$.

IND is an error indicator whose value is 0 for normal execution and 1 if this call would have stored any elements beyond $C_{m, n}$.

BESSEL FUNCTIONS SUBROUTINE (BESSL)†

PURPOSE

The Bessel Functions subroutine (BESSL) calculates the two Bessel Functions of the first kind quickly and accurately. This method is particularly adapted to the problem of calculating more than one order for a given X. For p = NMIN, NMIN+1,...,NMAX, where NMIN and NMAX are zero or positive integers, and for $X > 0$, either Jp(X) or Ip(X) is calculated depending on a parameter ITYPE.

METHOD

The method used by BESSL is discussed in an article by Irene Stegun and Milton Abramowitz.* It consists of three steps for Jp which are:

A. k is chosen the larger of 1.5X and NMAX, then k = k+10 for sufficient accuracy,

then $\bar{J}_{k+2} = 0$

$J_{k+1} = \alpha$, an arbitrarily small constant which is 0.1000E-10 in this program.

B. The recursion formula

$$\bar{J}_p = \frac{2(p+1)}{X} \bar{J}_{p+1} - \bar{J}_{p+2}$$

is used to generate $\bar{J}_p$ for p = k down to p = 0.

C. The results can be normalized, since

$$J_o + 2 \sum_{m=1}^{\infty} J_{2m} = 1,$$

therefore a constant $c = \bar{J}_o + 2 \sum_{m=1}^{k/2} \bar{J}_{2m}$

is determined and then

$J_p = \bar{J}_p/c$ for p = NMIN, NMIN+1,...,NMAX

The procedure for Ip is essentially the same except that in step B the recursion formula is:

$$\bar{I}_p = \frac{2(p+1)}{X} \bar{I}_{p+1} + \bar{I}_{p+2}$$

†Portions of this routine have been reprinted from C. B. Chandler's "BESSL - Bessel Functions Subroutine," TIS No. 64TIP5, issued by the Telecommunications & Information Processing Department of the General Electric Company at Schenectady, New York.

*Stegun, Irene A., and Abramowitz, Milton, "Generation of Bessel Functions on High Speed Computers," Mathematics and Other Aids to Computation, 1957, 11:255-257.

and in step C the normalization is due to:

$$I_o + 2 \sum_{m=1}^{\infty} I_m = e^x$$

and therefore the constant $c = (\bar{I}_o + 2 \sum_{m=1}^{k} \bar{I}_m)/e^x$.

USAGE

The calling sequence for this routine is:

```
CALL BESSL (ITYPE,X,NMIN,NMAX,BESJI)
```

where ITYPE = 1 for Bessel Function Jp(X);
ITYPE = 2 for Modified Bessel Function Ip(X);
X = independent variable > 0;
NMIN, NMAX = 0, 1, 2,... giving range of orders of Jp(X) or Ip(X) desired;
BESJI( ) = a vector where answers are stored in increasing order. The maximum size of this vector is determined by the user.

RESTRICTIONS

The generated $\bar{J}p$ (or $\bar{I}p$) must fall within the limits 10^{-36} and 10^{+36}. If either $\geq 10^{+38}$ then ITYPE is set equal to zero and control is transferred to RETURN. If $\bar{J}p$ (or $\bar{I}p$) $\leq 10^{-36}$ then that term is set equal to zero and the program continues. The user can check on overflow by branching on ITYPE although normally overflow will not occur.

INTERPOLATION ROUTINE (INTP)

PURPOSE

Using a vector of X values in ascending order and a corresponding vector of Y values, the Interpolation routine (INTP) finds, by three-point interpolation, the value of Y for a given value of X.

METHOD

The routine first finds the first X equal to or greater than the given value (X_2). If there is no X meeting this requirement, exit is made with a dummy Y value of plus bits. If X_2 is one of the end points of the X vector, the next value is choosen as X_2. The preceding X is chosen as X_1 and the following as X_3. Assuming a curve of the form

$$Y = a + bx + cx^2$$

to pass through these three points, the unknown constants a, b, and c can be expressed in terms of X_1, Y_1, X_2, Y_2, X_3, Y_3.

Substituting these known values and the given value of X yields a value of Y.

USAGE

INTP is called by a sequence of the form:

```
CALL INTPL (X,Y,N,XVAL,YVAL)
```

where X and Y are the vectors required, each of dimension N.

XVAL is the given X value, and the interpolated value of Y will be stored by INTP into YVAL.

RESTRICTIONS

The vector of X values must be stored in ascending order.

DOCUMENT REVIEW SHEET

TITLE: GE-625/635 FORTRAN IV Math Library (SSI)

CPB #: 1083B

Name: ______________________

Position: ______________________

Address: ______________________

____

CHECK ONE:

☐ Additional information would be helpful on following subjects.

☐ Errors indicated and pages where errors occur.

☐ Usefulness of manual could be improved as noted.

My comments are: ______________________

Please cut along this line

STAPLE STAPLE

FOLD

FIRST CLASS
PERMIT, No. 4332
PHOENIX, ARIZONA

BUSINESS REPLY MAIL

NO POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

POSTAGE WILL BE PAID BY

GENERAL ELECTRIC COMPUTER DEPARTMENT
13430 NORTH BLACK CANYON HIGHWAY
PHOENIX, ARIZONA - 85001

ATTENTION: Program Documentation
Systems and Processors Operation

FOLD

Progress Is Our Most Important Product

GENERAL ELECTRIC

INFORMATION SYSTEMS DIVISION

LITHO U.S.A.