

UNCLASSIFIED

AD 274 130

*Reproduced
by the*

**ARMED SERVICES TECHNICAL INFORMATION AGENCY
ARLINGTON HALL STATION
ARLINGTON 12, VIRGINIA**



UNCLASSIFIED

NOTICE: When government or other drawings, specifications or other data are used for any purpose other than in connection with a definitely related government procurement operation, the U. S. Government thereby incurs no responsibility, nor any obligation whatsoever; and the fact that the Government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data is not to be regarded by implication or otherwise as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use or sell any patented invention that may in any way be related thereto.

CATALOGED BY ASTIA
AS AD 130

274 130

274 130



DEPARTMENT OF THE NAVY
DAVID TAYLOR MODEL BASIN

HYDROMECHANICS

○

AERODYNAMICS

○

STRUCTURAL
MECHANICS

○

APPLIED
MATHEMATICS

A BASIC SET OF MATHEMATICAL SUBROUTINES
FOR LARC

by

Naomi C. Millet



APPLIED MATHEMATICS LABORATORY
RESEARCH AND DEVELOPMENT REPORT

March 1961

Report 1303

A BASIC SET OF MATHEMATICAL SUBROUTINES FOR LARC

by

Naomi C. Millet

March 1961

Report 1303

TABLE OF CONTENTS

	<u>Page</u>
ABSTRACT.	1
INTRODUCTION	1
SUBROUTINES.	3
Square Root.	3
Cube Root.	8
Arc Tangent I.	14
Arc Tangent II	20
Natural Logarithm	24
Sine-Cosine.	30
Exponential.	36
Complex Addition-Subtraction	42
Complex Multiplication.	44
Complex Division	46
Reciprocal of a Complex Number	48
Absolute Value of a Complex Number	50
Argument of a Complex Number.	52
 COMMENTS ON THE USE OF THE ASSEMBLY AND SIMULATOR.	 55
 APPENDIX A. Optimum Linear Approximation Used in the Square Root Routine.	 57
 APPENDIX B. Derivation of the Iterative Procedure Used in the Cube Root Subroutine.	 60
 BIBLIOGRAPHY.	 62

LIST OF TABLES

Table 1 - Comparison of Computed Square Root Values with True Values.	5
 Table 2 - Comparison of Computed Cube Root Values with True Values.	 10

LIST OF TABLES

	<u>Page</u>
Table 3 - Comparison of Computed Arc Tangent Values (Livermore) with True Values.	16
Table 4 - Comparison of Computed Arc Tangent Values (Rem Rand) with True Values.	21
Table 5 - Comparison of Computed Natural Logarithm Values with True Values.	25
Table 6 - Comparison of Computed Sine Values with True Values.	32
Table 7 - Comparison of Computed Cosine Values with True Values.	32
Table 8 - Comparison of Computed Exponential Values with True Values.	38

ACKNOWLEDGEMENT

The author wishes to thank Dr. John W. Wrench, Jr. for the computations of the true values which are in the tables following each routine. His comments were helpful in successfully completing this report.

ABSTRACT

This report presents a basic set of mathematical subroutines for LARC selected from those programmed by personnel of the University of California Radiation Laboratory, Remington Rand, and the Applied Mathematics Laboratory of the David Taylor Model Basin. Each subroutine was analyzed, then code-checked on UNIVAC I using LARC assembly and simulation routines (LISA and LIS). Information is given for each subroutine in a form for easy application. Derivations of numerical methods used, tables of computed values obtained from running the subroutines, and the true values are included.

INTRODUCTION

Since a great deal of programming effort will be involved in code-checking large scale problems for the LARC, the availability of a reliable library of mathematical subroutines will be of considerable help.

One way of code-checking the subroutines in advance has been developed by Applied Mathematics Laboratory personnel who have programmed a routine, LIS,¹ which will simulate LARC computing unit operations on UNIVAC I. In addition, Applied Mathematics Laboratory personnel have developed an assembly language and programmed an assembly routine, LISA,² which will produce an input program for the simulator, LIS, and for the LARC Computing Unit.

¹References are listed on page 62.

A basic set of subroutines was selected from those programmed for LARC by the University of California Radiation Laboratory, Remington Rand, and Applied Mathematics Laboratory personnel. They were analyzed and prepared for code-checking on UNIVAC I, using LISA and LIS. The accuracy of each routine was checked and compared with existing tables. The result is a set of subroutines which are presented here in a form for easy reference.

For each routine the following information is included: name, purpose, restrictions, accuracy, use, timing, programmer, method, cases used in code-checking, and coding in the form required as input for LISA. The address of the memory location from which a subroutine is entered is stored in register 01. The exit from each subroutine is a return to the address following that stored in register 01.

SUBROUTINES

SQUARE ROOT

File Name: SQR

Purpose: To calculate the square root of a nonnegative, single-precision floating-point number x .

Accuracy: $\sqrt{x}$ is obtained to 9 significant figures.

Usage:

(a) Entrance

1. Starting location: SQR.
2. Argument: contents of register 02.

(b) Storage Requirements

1. Five consecutive registers: 01 through 05.
2. Instructions: 22 lines beginning at location SQR.
3. Constants: 24 lines beginning at location CON.
4. Error return: location SQN.

(c) Exit

Results stored in register 02.

(d) Error Detected

If $x < 0$ control is sent to memory location SQN.

Timing: $160\mu s$

Based On Program By: L. Barr, Livermore

Method: The determination of the square root of x proceeds as follows:

(a) If $x = 0$, the result is zero.

(b) If $x < 0$, there is an error return.

(c) If $x > 0$, then

(1) $x = M \cdot 10^N$ where N is an integer and $1 < M < 10$, and

(2) $\sqrt{M}$ is determined as follows.

The initial approximation to $\sqrt{M}$ is obtained using a set of straight line segments to approximate $y = \sqrt{M}$ for $n \leq M < n+1$ where $n = 1, \dots, 9$. The equation for the line segment for the n th interval is considered to be written as follows:

$$y^* = a_n M + b_n$$

The a_n and b_n are obtained from stored tables. The determination of a_n and b_n is such that $|\sqrt{M} - (a_n M + b_n)|$ is minimized for the interval $n \leq M < n+1$. * This approximation is accurate to three significant figures. Two Newton iterations are then used to determine the square root to nine significant figures.³

The square roots of the numbers shown in Table 1 have been used in checking out SQR. In each case nine significant figures were obtained.

*See APPENDIX A.

x	Computed $\sqrt{x}$	True Value of $\sqrt{x}$
0.00000000	0.00000000	0.00000000
1.00000000	1.00000000	1.00000000
2.00000000	1.41421356	1.41421356
3.00000000	1.73205080	1.73205081
4.00000000	2.00000000	2.00000000
5.00000000	2.23606797	2.23606798
6.00000000	2.44948974	2.44948974
7.00000000	2.64575131	2.64575131
8.00000000	2.82842711	2.82842712
9.00000000	3.00000000	3.00000000
26.5938654	5.15692402	5.15692403
209.000000	14.4568322	14.4568323
0.00265938684	0.0515692402	0.0515692403
0.000265938654	0.0163076256	0.0163076256
0.999999999x10 ⁴⁹	0.316227765x10 ²⁵	0.316227766x10 ²⁵

TABLE 1

Comparison of Computed Square Root Values with True Values

DTMB

LARC C.U. ASSEMBLER CODING SHEET

Page: 1

Date:

Programmer: L. Barr

Problem SQR

Line No.	I	A	M	B	t
I / / / / /	X X X X X	Δ Δ S Q R Δ Δ	Δ Δ Δ Δ Δ		
S Q R	T Z	0 2	0 0 0 + 0 0 1	0 1	
	T L Z	0 2	S Q N		
	S	0 2	9 9 9 + 0 0 5		
	F F	0 3	C C N + 0 1 9		
	E A	0 3	9 9 9 + 0 0 2		
	P R	0 3	0 0 0 + 0 0 8		
	E O P	0 5	9 9 9 + 0 0 4		
	E O P	0 4	9 9 9 + 0 0 2		
	M	0 5	C O N	0 3	
	A	0 5	C O N + 0 0 9	0 3	
	M X R	0 4	C O N + 0 2 0		
	A X	0 4	C O N + 0 2 1		
	E O P	0 5	9 9 9 + 0 0 4		
	E O P	0 3	9 9 9 + 0 0 4		
	T G	0 3	S Q R + 0 1 6		
	M	0 5	C O N + 0 2 2		
	D U R	0 2	9 9 9 + 0 0 5		
	A	0 3	9 9 9 + 0 0 5		
	D R	0 2	9 9 9 + 0 0 3		
	M	0 3	C O N + 0 2 3		
	A	0 2	9 9 9 + 0 0 3		
	T		0 0 0 + 0 0 1	0 1	
# / / / / /	X X X X X	Δ Δ C O N S T A N T S Δ			
C O N	0 0 0 0 0 0 0	0 0 0 0 0 0			
	0 5 0 4 1 3 1	0 0 0 0 0			
	0 5 0 3 1 7 5	0 0 0 0 0			
	0 5 0 2 6 7 8	0 0 0 0 0			
	0 5 0 2 3 5 8	0 0 0 0 0			

Page: 2
Date:
Programmer:

Problem SQR

[illegible]

CUBE ROOT

File Name: CUR

Purpose: To calculate the cube root of a single-precision,
floating-point number, X.

Accuracy: $\sqrt[3]{X}$ is obtained to 9 significant figures.

Usage:

(a) Entrance

1. Starting location: CUR.
2. Argument: contents of register 04.

(b) Storage requirements

1. Four consecutive registers: 01 through 04.
2. Instructions: 48 lines beginning at location CUR.
3. Constants: 10 lines beginning at location CON.
4. Working storage: 5 lines beginning at location COM.

(c) Exit

1. Results stored in 03.
2. Argument stored in 02.

(d) Errors detected

None.

Timing: 300 μ s

Based on Program By: Robert Shafer, Livermore

Method: The determination of the cube root of X proceeds as follows:

(a) If $X = 0$, then $X^{1/3}$ is zero.

(b) If $X \neq 0$, then $X = M \cdot 10^N$ where N is an integer and $.1 < M < 1.0$.

Therefore,

$$X^{1/3} = M^{1/3} \cdot 10^{N/3}$$

which is computed as follows.

(1) The division of $\frac{N}{3}$ is performed. Then we have

$$\frac{N}{3} = I + J$$

where I is an integer and $J = 0, 1/3$ or $2/3$, and

$$10^{N/3} = 10^I \cdot 10^J$$

is computed. The numbers 10^0 , $10^{1/3}$, and $10^{2/3}$ are stored as constants in the subroutine.

(2) To compute $M^{1/3}$, three iterations are performed, using the formula

$$m_{(n+1)} = \left(\frac{2M + m_n^3}{M + 2m_n^3} \right) m_n.$$

The initial approximation $m_0 = .7$ is used to obtain m_1 . The third iteration yields an approximation to $M^{1/3}$ which is accurate to the prescribed number of significant figures. A derivation of this iteration procedure is given in Appendix B. The results of code-checking may be found in Table 2.

X	Computed $\sqrt[3]{X}$	True Value of $\sqrt[3]{X}$
0.000000000000	0.00000000	0.00000000
1.000000000000	1.00000000	1.00000000
1.500000000000	1.14471424	1.14471424
2.000000000000	1.25992104	1.25992105
2.500000000000	1.35720880	1.35720880
3.000000000000	1.44224957	1.44224957
3.500000000000	1.51829448	1.51829448
4.000000000000	1.58740105	1.58740105
5.000000000000	1.70997594	1.70997595
6.000000000000	1.81712059	1.81712059
7.000000000000	1.91293118	1.91293118
-0.000367905300	-0.0716547973	-0.0716548099
791.000000000000	9.24821859	9.24823438
0.999999999x10 ⁴⁹	0.215442679x10 ¹⁷	0.215443469x10 ¹⁷

TABLE 2
Comparison of Computed Cube Root Values with True Values

DTMB

LARC C.U. ASSEMBLER CODING SHEET

Page: 1

Date:

Programmer: R. Shafer

Problem CUR

Line No.	I	A	M	B	t
I / / / / /	XXXXX	ΔΔCURΔ	ΔΔΔΔΔΔ		
CUR	TZ	04	CUR+046		
	S	04	COM+004		
	SM	04	999+004		
	F	03	CON+008		
	PPL	03	000+002		
	F	02	CON+002		
	MXE	02	999+003		
	AX	02	CON+001		
	PR	03	000+010		
	PR	04	000+001		
	SS	03	COM+001		
	AX	04	CON+004		
	F	03	COM+002		
	AX	03	999+003		
	S	03	COM+003		
	MXR	03	CON+007		
	AX	03	CON+005		
	DX	03	999+004		
	S	03	COM		
	MXR	03	999+003		
	MXR	03	COM		
	PR	03	000+001		
	S	03	999+004		
	AX	04	999+003		
	AX	03	COM+003		
	AX	04	COM+002		
	MXR	03	COM		
	DX	03	999+004		
	S	03	COM		

DTMB

LARC C.U. ASSEMBLER CODING SHEET

Page: 2

Date:

Programmer:

Problem

Line No.	I	A	M	B	t
	MXR	0 3	9 9 9 + 0 0 3		
	MXR	0 3	COM		
	PR	0 3	0 0 0 + 0 0 1		
	S	0 3	9 9 9 + 0 0 4		
	AX	0 4	9 9 9 + 0 0 3		
	AX	0 4	COM + 0 0 2		
	AX	0 3	COM + 0 0 3		
	MXR	0 3	COM		
	DX	0 3	9 9 9 + 0 0 4		
	F	0 4	COM + 0 0 1		
	MXR	0 3	CON	0 4	
	PPR	0 2	0 0 0 + 0 0 2		
	F	0 4	COM + 0 0 4		
	TLZ	0 4	CUR + 0 4 4		
	T		0 0 0 + 0 0 1	0 1	
	SN	0 3	9 9 9 + 0 0 3		
	T		0 0 0 + 0 0 1	0 1	
	S	0 4	9 9 9 + 0 0 3		
	T		0 0 0 + 0 0 1	0 1	
.....					
# / / / / / /	XXXXX	Δ Δ	CONSTANTS	Δ	
CON	0 2 1 5 4 4 3	4 6 9 0 1			
	0 0 0 0 0 0 0	0 0 0 3 4			
	0 3 3 3 3 3 3	3 3 3 3 4			
	0 4 6 4 1 5 8	8 8 3 3 6			
	0 0 6 8 6 0 0	0 0 0 0 0			
	0 0 2 4 0 1 0	0 0 0 0 0			
	0 9 9 9 9 9 9	9 9 9 9 9			
	0 7 0 0 0 0 0	0 0 0 0 0			
	0 0 0 0 0 0 0	0 0 0 0 0			

LARC C.U. ASSEMBLER CODING SHEET

Date:

Problem

[illegible]

ARC TANGENT I

File Name: ART

Purpose: To calculate the arc tangent of a single-precision,
floating-point number, X.

Restrictions:

$$10^{-25} < |X| < 10^{25}$$

Accuracy: Arc Tan X is obtained to 9 significant figures.

Usage:

(a) Entrance

1. Starting Location: ART.
2. Argument: contents of register 04.

(b) Storage requirements

1. Four consecutive registers: 01 through 04.
2. Instructions: 44 lines beginning at location ART.
3. Constants: 20 lines beginning at location CON.
4. Working storage: 4 lines beginning at location COM.

(c) Exit

Results stored in register 02.

(d) Errors detected

None.

Timing: 288 μ s

Based on Program By: Robert Shafer, Livermore

Method:

Case I: $X > 1$

In this case

$$\tan^{-1} X = \tan^{-1} (1) + \tan^{-1} \alpha + \tan^{-1} \nu$$

is used where $\alpha = \frac{N}{10}$ for some integer N , $0 < N < 10$ and $\nu < .1$.

The details follow.

First

$$\tan^{-1} X = \tan^{-1} (1) + \tan^{-1} \left(\frac{X-1}{X+1} \right)$$

is used and then

$$\tan^{-1} \left(\frac{X-1}{X+1} \right) = \tan^{-1} \alpha + \tan^{-1} \frac{\beta}{\alpha^{2+\alpha}\beta+1}$$

where

$$\alpha = \frac{1}{10} \left[10 \left\{ \left(\frac{X-1}{X+1} \right) + .05 \right\} \right]. \quad (\text{Here } [\zeta] \text{ designates}$$

the largest integer $\leq \zeta$.)

$$[\zeta] = \left[10 \left\{ \left(\frac{X-1}{X+1} \right) + .05 \right\} \right]$$

$$\beta = \left(\frac{X-1}{X+1} \right) - \alpha$$

Finally letting $\frac{\beta}{\alpha^{2+\alpha}\beta+1} = \nu$, then $\nu < .1$.

Case II:

$$X = 1$$

$\tan^{-1} (1)$ is available directly.

Case III: $0 \leq X < 1$

In this case

$$\tan^{-1} X = \tan^{-1} \alpha + \tan^{-1} \nu$$

is used where $\alpha = \frac{N}{10}$ for some integer N ,

$0 \leq N < 10$ and $\nu < .1$.

Case IV: $X < 0$.

In this case

$$\tan^{-1} X = -\tan^{-1} y, \text{ where } y = -X, y > 0 \text{ is}$$

used. Then Case I, II, or III, whichever is appropriate, is used to determine $\tan^{-1} y$.

The arguments used in code-checking are listed in Table 3.

X	Arc Tan X	True Value of Arc Tan X
2.50000000	1.19028994	1.19028995
-3.50000000	-1.29249666	-1.29249667
4.50000000	1.35212738	1.35212738
5.50000000	1.39094282	1.39094283
6.00000000	1.40564764	1.40564765
7.00000000	1.42889927	1.42889927
-0.11000000	-0.1095595267	-0.1095595268
0.577×10^{14}	1.57079632	1.57079633

TABLE 3

**Comparison of Computed Arc Tangent Values (Livermore)
with True Values**

DTMB

LARC C.U. ASSEMBLER
CODING SHEET

Page: 1

Date:

Programmer: R. E. Shafer

Problem ARC TANGENT I

Line No.	I	A	M	B	t
I / / / / /	X X X X X	Δ A R T Δ Δ Δ Δ Δ Δ Δ Δ			
ART	F	0 2	CON + 0 1 6		
	S	0 2	COM + 0 0 3		
	S M	0 4	9 9 9 + 0 0 2		
	N	0 2	CON + 0 1 7		
	T Z	0 2	ART + 0 4 3		
	T G Z	0 2	ART + 0 3 7		
	A X	0 2	CON + 0 2 0		
	F	0 3	CON + 0 1 6		
	A	0 2	CON		
	P R	0 2	0 0 0 + 0 0 1		
	E M	0 3	9 9 9 + 0 0 2		
	P L	0 2	0 0 0 + 0 0 1		
	N	0 2	CON + 0 0 1		
	S	0 2	COM		
	M R	0 2	9 9 9 + 0 0 4		
	S M	0 2	9 9 9 + 0 0 2		
	A	0 2	CON + 0 1 7		
	S	0 2	COM + 0 0 1		
	S M	0 4	9 9 9 + 0 0 2		
	N	0 2	COM		
	D R	0 2	COM + 0 0 1		
	S	0 2	COM		
	M R	0 2	9 9 9 + 0 0 2		
	S	0 2	COM + 0 0 2		
	M	0 2	CON + 0 0 2		
	A	0 2	CON + 0 0 3		
	M	0 2	COM + 0 0 2		
	A	0 2	CON + 0 0 4		
	M	0 2	COM + 0 0 2		

DTMB

LARC C.U. ASSEMBLER
CODING SHEET

Page: 2

Date:

Programmer:

Problem

Line No.	I	A	M	B	t
	A	0 2	CON+ 0 1 7		
	MR	0 2	COM		
	A	0 2	CON+ 0 0 5	0 3	
	A	0 2	COM+ 0 0 3		
	TLZ	0 4	ART+ 0 3 5		
	T		0 0 0 + 0 0 1	0 1	
	SN	0 2	9 9 9 + 0 0 2		
	T		0 0 0 + 0 0 1	0 1	
	F	0 3	CON+ 0 1 5		
	S	0 3	COM+ 0 0 3		
	AU	0 2	CON+ 0 1 8		
	DR	0 2	9 9 9 + 0 0 3		
	S	0 2	9 9 9 + 0 0 5		
	EU	0 4	CON+ 0 1 9		
	T		ART+ 0 0 8		
	F	0 2	CON+ 0 1 5		
	T		ART+ 0 3 3		
.					
# / / /	/ / /	XXXXX	X Δ	CONSTANTS	Δ Δ
CON	0 5 7 1	0 0 0 0	0 0 0 5		
	0 5 7 1				
	- 5 0 1	4 2 8	5 7 1 4 3		
	0 5 0 2				
	- 5 0 3	3 3 3	3 3 3 3 3		
	0 5 0				
	0 4 9	9 9 6 6	8 6 5 2 5		
	0 5 0	1 9 7 3	9 5 5 6 0		
	0 5 0	2 9 1 4	5 6 7 9 4		
	0 5 0	3 8 0 5	0 6 3 7 7		
	0 5 0	4 6 3 6	4 7 6 0 9		

Programmer:

Problem

[illegible]

ARC TANGENT II (Remington Rand)

File Name: ARC

Purpose: To calculate the arc tangent of a single-precision, floating-point number, X.

Accuracy: Arc tan X is obtained to 9 significant figures.

Usage:

(a) Entrance

1. Starting location: ARC.
2. Argument: contents of register 06.

(b) Storage requirements

1. Six consecutive registers: 01 through 06.
2. Instructions: 37 lines beginning at location ARC.
3. Constants: 10 lines beginning at location CON.

(c) Exit

Results stored in 06.

(d) Errors detected

None.

Timing: 144 μ s

Based on Program By: A. B. Tonik, Remington Rand

Method:

Case I: $0 \leq X \leq 1$

In this case

$$\begin{aligned} \text{arc tan } X &= \frac{\pi}{8} + \text{arc tan } Z \\ \text{where } Z &= \frac{X - (\sqrt{2} - 1)}{1 + (\sqrt{2} - 1) X} \end{aligned}$$

Case II: $|X| > 1$

In this case

$$\arctan X = \frac{3\pi}{8} + \arctan Z,$$

$$\text{where } Z = \frac{X + (\sqrt{2} - 1)}{X(\sqrt{2} - 1) - 1}$$

The arguments used in code-checking are listed in Table 4.

X	Arc Tan X	True Value of Arc Tan X
2.5	1.19028994	1.19028995
-3.5	-1.29249666	-1.29249667
4.5	1.35212738	1.35212738
5.5	1.39094282	1.39094283
6.0	1.40564129	1.40564765
7.0	1.42888707	1.42889927
-0.1	-0.0996686525	-0.0996686525
0.577×10^{14}	1.50920769	1.57079633
-0.577×10^{14}	-1.50920769	-1.57079633

TABLE 4

Comparison of Computed Arc Tangent Values (Rem. Rand)
with True Values

DTMB

LARC C.U. ASSEMBLER
CODING SHEET

Page: 1

Date:

Programmer: Remington Rand

Problem ARC

Line No.	I	A	M	B	t
I / / /	/ / /	X X X X	X Δ Δ A R C Δ Δ Δ Δ Δ Δ		
ARC	SM	0 6	9 9 9 + 0 0 1		
	F	0 3	CON		
	TG	0 2	ARC + 0 2 0		
	MU	0 2	CON + 0 0 1		
	A	0 3	CON		
	N	0 2	CON + 0 0 1		
	DUR	0 2	9 9 9 + 0 0 3		
	MU	0 3	9 9 9 + 0 0 3		
	MU	0 4	CON + 0 0 2		
	A	0 5	CON + 0 0 3		
	M	0 5	9 9 9 + 0 0 3		
	A	0 5	CON + 0 0 4		
	M	0 5	9 9 9 + 0 0 4		
	A	0 5	CON + 0 0 5		
	M	0 5	9 9 9 + 0 0 4		
	A	0 5	CON + 0 0 6		
	M	0 5	9 9 9 + 0 0 3		
	A	0 5	CON + 0 0 7		
	EU	0 5	CON + 0 0 8		
	T		0 0 0 + 0 0 1	0 1	
	MU	0 2	CON + 0 0 1		
	N	0 3	CON		
	A	0 2	CON + 0 0 1		
	DR	0 3	9 9 9 + 0 0 2		
	MU	0 3	9 9 9 + 0 0 3		
	MU	0 4	CON + 0 0 2		
	A	0 5	CON + 0 0 3		
	M	0 5	9 9 9 + 0 0 4		
	A	0 5	CON + 0 0 4		

NATURAL LOGARITHM

File Name: LOG

Purpose: To compute the natural logarithm of a floating-point,
single-precision number, X.

Restrictions: $X > 0$

Accuracy: $\ln X$ is obtained to 9 significant figures.

Usage:

(a) Entrance

1. Starting location: LOG.
2. Argument: contents of register 02.

(b) Storage Requirements

1. Five consecutive registers: 01 through 05.
2. Instructions: 25 lines beginning at location LOG.
3. Constants: 76 lines beginning at location LGC.

(c) Exit

Results stored in register 02.

(d) Errors Detected

If $x \leq 0$, control is sent to memory location NLA.

Timing: 148 μ s

Based On Program By: Leota Barr, Livermore

Method: The determination of $\ln x$ proceeds as follows:

- (a) If $x \leq 0$, there is an error return
- (b) If $x > 0$,

then

$$\ln X = I \ln 10 + \ln F$$

where I is an integer and $.1 \leq F < 1.0$

Let

$$F = \frac{F'}{d}$$

where d is a set of stored constants whose use is determined

by the first digit of F such that $.5 \leq Fd < 1.0$.

Let

$$F' = \frac{G}{H + 1}$$

where G is the two most significant digits of $F' + 0.009999999999$.

Then $H = \frac{G - F'}{F'} < .02$ and

$$\ln (H+1) \doteq C_1 H + C_2 H^2 + C_3 H^3$$

The C_i 's are derived using Chebyshev polynomials.

The arguments used in code-checking this subroutine are shown in

Table 5.

x	LN X	True Value of LN X
0.001	-6.90775526	-6.90775528
1.5	0.405465108	0.405465108
2.0	0.693147189	0.693147181
3.0	1.09861230	1.09861229
4.0	1.38629437	1.38629436
5.0	1.60943793	1.60943791
6.0	1.79175947	1.79175947
7.0	1.94591015	1.94591015
8.0	2.07944156	2.07944154
9.0	2.19722459	2.19722458
112.0	4.71849888	4.71849887
40000.0	10.5966348	10.5966347

TABLE 5

Comparison of Computed Natural Logarithm Values with True Values

DTMB

LARC C.U. ASSEMBLER CODING SHEET

Page: 1

Date:

Programmer: L. Barr

Problem LOG

Line No.	I	A	M	B	t
I / / /	/ / /	X X X X	X Δ L O G Δ Δ Δ Δ Δ Δ Δ		
LOG	T E Z	0 2	N L A		
	P P R	0 2	0 0 0 + 0 0 9		
	N X	0 2	C O N + 0 6 9		
	C	0 2	0 0 0 + 0 6 1		
	F	0 4	9 9 9 + 0 0 3		
	P R	0 4	0 0 0 + 0 1 0		
	M	0 2	C O N + 0 7 0		
	M X R	0 3	C O N	0 4	
	N	0 2	C O N + 0 0 9	0 4	
	F	0 4	C O N + 0 7 1		
	P L	0 3	0 0 0 + 0 0 1		
	E O P	0 4	9 9 9 + 0 0 3		
	N X	0 4	9 9 9 + 0 0 3		
	D X	0 4	9 9 9 + 0 0 3		
	P R	0 3	0 0 0 + 0 0 9		
	C	0 4	0 0 0 + 0 5 0		
	M U	0 4	C O N + 0 7 2		
	A	0 5	C O N + 0 7 3		
	M	0 5	9 9 9 + 0 0 4		
	A	0 5	C O N + 0 7 4		
	M	0 5	9 9 9 + 0 0 4		
	N X	0 3	C O N		
	A	0 2	C O N + 0 1 9	0 3	
	N	0 2	9 9 9 + 0 0 5		
	T		0 0 0 + 0 0 1	0 1	
.....					
# / / /	/ / /	X X X X	X Δ C O N S T A N T S Δ Δ		
C O N	0 0 0 0	0 0 0 0	0 0 0 4 9		
	0 5 0 0	0 0 0 0	0 0 0 0 0		

DTMB

LARC C.U. ASSEMBLER CODING SHEET

Page: 2

Date:

Programmer:

Problem

Line No.	I	A	M	B	t
	0 3 3 3 3 3 3 3 3 3 3				
	0 2 5 0 0 0 0 0 0 0 0				
	0 2 0 0 0 0 0 0 0 0 0				
	0 1 6 6 6 6 6 6 6 6 6				
	0 1 4 2 8 5 0 0 0 0 0				
	0 1 2 5 0 0 0 0 0 0 0				
	0 1 1 1 1 1 1 1 1 1 1				
	0 1 0 0 0 0 0 0 0 0 0				
	0 5 1 1 6 0 9 4 3 7 9 1				
	0 5 1 1 2 0 3 9 7 2 8 0				
	0 5 0 9 1 6 2 9 0 7 3 2				
	0 5 0 6 9 3 1 4 7 1 8 0				
	0 5 0 5 1 0 8 2 5 6 2 0				
	0 5 0 3 5 6 6 2 4 9 4 3				
	0 5 0 2 2 2 3 1 4 5 5 1				
	0 5 0 1 0 5 3 6 0 5 1 5				
	0 5 0 0 0 0 0 0 0 0 0				
	- 5 0 6 9 3 1 4 7 1 8 1				
	- 5 0 6 7 3 3 4 4 5 5 3				
	- 5 0 6 5 3 9 2 6 4 6 7				
	- 5 0 6 3 4 8 7 8 2 7 2				
	- 5 0 6 1 6 1 8 6 1 3 9				
	- 5 0 5 9 7 8 3 7 0 0 1				
	- 5 0 5 7 9 8 1 8 4 9 5				
	- 5 0 5 6 2 1 1 8 9 1 8				
	- 5 0 5 4 4 7 2 7 1 7 5				
	- 5 0 5 2 7 6 3 2 7 4 2				
	- 5 0 5 1 0 8 2 5 6 2 4				
	- 5 0 4 9 4 2 9 6 3 2 2				
	- 5 0 4 7 8 0 3 5 8 0 1				

DTMB

LARC C.U. ASSEMBLER CODING SHEET

Page: 3

Date:

Programmer:

Problem

Line No.	I	A	M	B	t
	- 5 0 4	6 2 0	3 5 4 6 0		
	- 5 0 4	4 6 2	8 7 1 0 3		
	- 5 0 4	3 0 7	8 2 9 1 6		
	- 5 0 4	1 5 5	1 5 4 4 4		
	- 5 0 4	0 0 4	7 7 5 6 7		
	- 5 0 3	8 5 6	6 2 4 8 1		
	- 5 0 3	7 1 0	6 3 6 8 1		
	- 5 0 3	5 6 6	7 4 9 4 4		
	- 5 0 3	4 2 4	9 0 3 0 9		
	- 5 0 3	2 8 5	0 4 0 6 7		
	- 5 0 3	1 4 7	1 0 7 4 5		
	- 5 0 3	0 1 1	0 5 0 9 3		
	- 5 0 2	8 7 6	8 2 0 7 2		
	- 5 0 2	7 4 4	3 6 8 4 6		
	- 5 0 2	6 1 3	6 4 7 6 4		
	- 5 0 2	4 8 4	6 1 3 5 9		
	- 5 0 2	3 5 7	2 2 3 3 4		
	- 5 0 2	2 3 1	4 3 5 5 1		
	- 5 0 2	1 0 7	2 1 0 3 1		
	- 5 0 1	9 8 4	5 0 9 3 9		
	- 5 0 1	8 6 3	2 9 5 7 8		
	- 5 0 1	7 4 3	5 3 3 8 7		
	- 5 0 1	6 2 5	1 8 9 2 9		
	- 5 0 1	5 0 8	2 2 8 9 0		
	- 5 0 1	3 9 2	6 2 0 6 7		
	- 5 0 1	2 7 8	3 3 3 7 2		
	- 5 0 1	1 6 5	3 3 8 1 6		
	- 5 0 1	0 5 3	6 0 5 1 6		
	- 4 9 9	4 3 1	0 6 7 9 0		
	- 4 9 8	3 3 8	1 6 0 9 0		

LARC C.U. ASSEMBLER CODING SHEET

Date:

Problem

29

SINE-COSINE

File Name: SINCOS

Purpose: To compute the sine or cosine of a single-precision, floating-point number, x , in radians.

Restrictions: $|x| < 10^{10}$

Accuracy: Sine or cosine is obtained to 9 significant figures.

Usage:

(a) Entrance

1. Starting location: SIN (or COS).
2. Argument: contents of register 02.

(b) Storage requirements

1. Three consecutive registers: 01 through 03.
2. Instructions: 69 lines beginning at location COS or SIN.
3. Constants: 22 lines beginning at location CON.
4. Working storage: 1 line at location COM.

(c) Exit

Results of SIN or COS stored in 03.

(d) Errors detected

None.

Timing: 348 μ s average

Based On Program By: K. B. Williams, Livermore

Method:

Case A: Cosine Routine

1 $\rightarrow$ quadrant storage location, QSL

Case B: Sine Routine

0 $\rightarrow$ quadrant storage location, QSL

Case I: $0 \leq x \leq \frac{\pi}{2}$

In Cases I and IV if $x \leq 0.001$, then $x = \sin x$.

If $x > 0.001$ then sine or cosine series is used, depending on the contents of location QSL. The coefficients in the series are obtained from a Los Alamos Scientific Laboratory report.⁴

Case II: $x > 0$; $x > \frac{\pi}{2}$

Then

$x \cdot \frac{2}{\pi} = Y$, where Y consists of an integral part, N , and a fractional part.

Then

$$N + (\text{QSL}) \rightarrow \text{QSL},$$

$$(Y - N) \left(\frac{\pi}{2} \right) \rightarrow x,$$

and Case I is used.

Case III: $x < 0$; $x \leq -\frac{\pi}{2}$

$$2 + \text{QSL} \rightarrow \text{QSL}$$

then Case I is used.

Case IV: $x < 0$; $|x| > \frac{\pi}{2}$

Case III and Case II are used.

Arguments used for checking out the routine are shown in

Table 6 and Table 7.

x	Sine x	True Values of Sine x
0.7	0.644217693	0.644217687
-0.51196	-0.489886890	-0.489886887
1.7482	0.984305147	0.984305196
0.0009	0.000900000	0.000900000
4.18879	-0.866025309	-0.866025301
5.75959	-0.499997010	-0.499996996
-1.05883	-0.871782915	-0.871782905
-2.01586	-0.902583260	-0.902583255
-4.71239	1.000000000	1.000000000
-5.58505	0.642790380	0.642790372

TABLE 6

Comparison of Computed Sine Values with True Values

x	Cosine x	True Values of Cosine x
0.7	0.764842190	0.764842187
-0.51196	0.871786010	0.871786004
1.7482	-0.176474592	-0.176474593
0.0009	0.999999600	0.999999595
4.18879	-0.500000190	-0.500000177
5.75959	0.866027140	0.866027138
-1.05883	0.489892410	0.489892403
-2.01586	-0.430515351	-0.430515352
-4.71239	$0.101685500 \times 10^{-5}$	$0.101961531 \times 10^{-5}$
-5.58505	0.766042130	0.766042125

TABLE 7

Comparison of Computed Cosine Values with True Values

DTMB

LARC C. U. SIMULATOR
CODING SHEET

Page: 1

Date:

Programmer: K. B. Williams

Problem SINCOS

Line No.	I	A	M	B	t
I / / / /	/ / / /	X X X X	X Δ Δ S I N C	Φ S Δ Δ Δ Δ	
C O S	F	0 7	C O N + 0 1 1		
	S M	0 2	9 9 9 + 0 0 2		
	F	0 3	9 9 9 + 0 0 0		
	T		S I N + 0 0 2		
S I N	F	0 7	9 9 9 + 0 0 0		
	F	0 3	C O N + 0 1 2		
	F	0 4	9 9 9 + 0 0 0		
	S	0 3	C O M		
	F	0 3	C O N + 0 1 3		
	S	0 2	C O M + 0 0 1		
	T L Z	0 2	S I N + 0 4 2		
	T G	0 2	S I N + 0 4 5		
	E O P	0 4	9 9 9 + 0 0 2		
	P R	0 4	0 0 0 + 0 0 9		
	F	0 3	C O M		
	T G	0 3	S I N + 0 5 3		
	E M	0 3	9 9 9 + 0 0 7		
	P L	0 3	0 0 0 + 0 0 5		
	M X R	0 3	C O N + 0 1 5		
	E M	0 4	9 9 9 + 0 0 3		
	M X R	0 4	C O N + 0 1 6		
	M	0 2	9 9 9 + 0 0 2		
	T		S I N + 0 1 9	0 4	
	F	0 7	C O N + 0 1 7		
	T		S I N + 0 2 6		
	P R	0 4	0 0 0 + 0 0 2		
	T		S I N + 0 2 5		
	F	0 7	C O N + 0 1 7		
	T		S I N + 0 2 6		

DTMB

LARC C. U. SIMULATOR
CODING SHEET

Page: 2

Date:

Programmer:

Problem

Line No.	I	A	M	B	t
	F	0 7	9 9 9 + 0 0 0		
	M U	0 2	C O N +	0 7	
	A	0 3	C O N + 0 0 1	0 7	
	M	0 3	9 9 9 + 0 0 2		
	A	0 3	C O N + 0 0 2	0 7	
	M	0 3	9 9 9 + 0 0 2		
	A	0 3	C O N + 0 0 3	0 7	
	M	0 3	9 9 9 + 0 0 2		
	A	0 3	C O N + 0 0 4	0 7	
	M	0 3	9 9 9 + 0 0 2		
	A	0 3	C O N + 0 0 5	0 7	
	T Z	0 7	S I N + 0 8 8		
	M	0 3	C O M + 0 0 1		
	T Z	0 4	9 9 9 + 0 0 1	0 1	
	S M	0 3	9 9 9 + 0 0 3		
	S N	0 3	9 9 9 + 0 0 3		
	T		0 0 0 + 0 0 1	0 1	
	A X	0 7	C O N + 0 1 8		
	S M	0 2	9 9 9 + 0 0 2		
	T		S I N + 0 0 7		
	M E	0 2	C O N + 0 1 9		
	C C X	0 2	0 0 0 + 0 6 1		
	A X	0 7	9 9 9 + 0 0 2		
	C	0 3	0 0 0 + 0 5 0		
	M R	0 3	C O N + 0 1 3		
	S	0 3	9 9 9 + 0 0 2		
	S	0 3	C O M + 0 0 1		
	T		S I N + 0 1 2		
	F	0 3	C O M + 0 0 1		
	T		0 0 0 + 0 0 1	0 1	

Page: 3
Date:
Programmer:

Problem

Line No.	I	A	M	B	t
• • • • •	• • • • •	• • • • •	• • • • •	• • • • •	• • • • •
# / / / / /	X X X X X	Δ C O N S T A N T S	Δ Δ		
C O N	- 4 4	2 6 0 5	0 0 0 0 0		
	0 4 6	2 4 7 6	0 9 0 0 0		
	- 4 8	1 3 8 8	8 3 9 7 0		
	0 4 9	4 1 6 6	6 6 4 1 8		
	- 5 0	4 9 9 9	9 9 9 9 6		
	0 5 1	1 0 0 0	0 0 0 0 0		
	- 4 3	2 3 9 0	0 0 0 0 0		
	0 4 5	2 7 5 2	6 0 0 0 0		
	- 4 7	1 9 8 4	0 9 0 0 0		
	0 4 8	8 3 3 3	3 3 1 5 0		
	- 5 0	1 6 6 6	6 6 6 6 6		
	0 0 0	0 0 0 0	0 0 0 0 1		
	0 0 0	0 0 0 0	0 0 0 4 8		
	0 5 1	1 5 7 0	7 9 6 3 3		
	0 0 0	0 0 0 0	0 0 0 0 0		
	0 2 5	0 0 0 0	0 0 0 0 0		
	0 0 0	0 0 8 0	0 0 0 0 0		
	0 0 0	0 0 0 0	0 0 0 0 6		
	0 0 0	0 0 0 0	0 0 0 0 2		
	0 5 0	6 3 6 6	1 9 7 7 2		
• • • • •	• • • • •	• • • • •	• • • • •	• • • • •	• • • • •
• • • • •	• • • • •	• • • • •	• • • • •	• • • • •	• • • • •

EXPONENTIAL

File Name: EXP

Purpose: To calculate the exponential e^x of a floating-point, single-precision number, x .

Restrictions: $-111 < x < 112$

Accuracy: e^x is obtained to 9 significant figures.

Usage:

(a) Entrance

1. Starting location: EXP.
2. Argument: contents of register 02.

(b) Storage Requirements

1. Eight consecutive registers: 01 through 08.
2. Instructions: 28 lines beginning at location EXP.
3. Constants: 38 lines beginning at location CON.

(c) Exit

Results stored in 02.

(d) Errors Detected

1. If $x < -111.0$, then an absolute zero will be stored in 02.
2. If $x > 112.0$, then a transfer to memory location ETL will be made.

Timing: $208\mu s$

Based On Program By: Robert Shafer, Livermore

Method: The determination of the exponential function proceeds as follows:

(a) If $x \leq -111.0$, then $e^x = 0$.

(b) If $x \geq 112.0$, there is an error return.

(c) If $-111.0 < x < 112.0$, the following two cases must be considered:

(1) If $x < 0$

$$e^x = e^I \cdot e^F$$

where I is an integer and

$$.1 < F < 1.0$$

Now

$$x = (I \ln 10 + F \ln 10) \frac{1}{\ln 10}$$

Let

$$F = \frac{F'}{\ln 10}$$

$$F' = F \ln 10$$

$$e^{F'} = e^G \cdot e^{F'-G}$$

where G = first two digits of F' .

Let $H = F' - G$

Then

$$e^H = c_0 + c_1 H + c_2 H^2 + c_3 H^3 + c_4 H^4$$

where the c_i 's are derived using Chebyshev polynomials.⁵

(2) If $x \geq 0$

then

$$e^x = e^{I+1} e^{F-1}$$

and

$$x = (I + 1) \ln 10 + (F-1) \ln 10 \quad \ln 10$$

Again let $F = \frac{\bar{F}'}{\ln 10}$, and the procedure is followed as

in (1) above.

The arguments used in code-checking are shown in Table 8.

x	e ^x	True Value of e ^x
1.0	2.71828181	2.71828183
2.0	7.38905612	7.38905610
3.0	20.0855369	20.0855369
4.0	54.5981495	54.5981500
5.0	148.413160	148.413159
6.0	403.428795	403.428793
7.0	1096.63316	1096.63316
8.0	2980.95795	2980.95799
9.0	8103.08379	8103.08393
111.0	0.160948712x10 ⁴⁹	0.160948707x10 ⁴⁹
-110.0	0.168891188x10 ⁻⁴⁷	0.168891188x10 ⁻⁴⁷

TABLE 8

Comparison of Computed Exponential Values with True Values

DTMB

LARC C.U. ASSEMBLER CODING SHEET

Page: 1

Date:

Programmer: L. Barr

Problem EXP

Line No.	I	A	M	B	t
I / / /	/ / / /	X X X X	X Δ E X P Δ Δ Δ	Δ Δ Δ Δ Δ	
EXP	T G Z	0 2	E X P + 0 0 5		
	F	0 3	C O N + 0 2 4		
	T G	0 2	E X P + 0 0 7		
	F	0 2	C O N + 0 2 5		
	T		0 0 0 + 0 0 1	0 5	
	F	0 3	C O N + 0 2 6		
	T G	0 2	E T L		
	M	0 2	C O N + 0 2 7		
	T B	0 9	E X P + 0 2 8		
	M X R	0 3	C O N + 0 2 8		
	P L	0 2	0 0 0 + 0 0 9		
	T L Z	0 3	E X P + 0 1 4		
	A X	0 2	C O N + 0 2 9		
	N X	0 3	C O N + 0 2 8		
	S	0 3	9 9 9 + 0 0 4		
	P P R	0 3	0 0 0 + 0 0 9		
	C	0 4	0 0 0 + 0 4 9		
	M U	0 4	C O N + 0 3 0		
	A	0 5	C O N + 0 3 1		
	M	0 5	9 9 9 + 0 0 4		
	A	0 5	C O N + 0 3 2		
	M	0 5	9 9 9 + 0 0 4		
	M U	0 4	C O N + 0 3 0		
	A	0 5	C O N + 0 3 1		
	M	0 5	9 9 9 + 0 0 4		
	A	0 5	C O N + 0 3 2		
	M	0 5	9 9 9 + 0 0 4		
	A	0 5	C O N + 0 3 3		
	M	0 5	9 9 9 + 0 0 4		

DTMB

LARC C.U. ASSEMBLER CODING SHEET

Page: 2

Date:

Programmer:

Problem

Line No.	I	A	M	B	t
	A	0 5	C O N + 0 3 4		
	A X	0 2	C O N	0 3	
	M	0 2	9 9 9 + 0 0 5		
	T		0 0 0 + 0 0 1	0 6	
	F F	0 7	C O N + 0 2 5		
	E O P	0 8	C O N + 0 3 5		
	E O P	0 7	9 9 9 + 0 0 2		
	E U	0 7	C O N + 0 3 6		
	N X	0 8	9 9 9 + 0 0 7		
	P R	0 8	0 0 0 + 0 0 9		
	F	0 7	E X P + 0 4 3		
	E M	0 7	9 9 9 + 0 0 8		
	S	0 7	E X P + 0 4 3		
	P L	0 2	0 0 0 + 0 0 2		
	P P R	0 2	0 0 0 + 0 0 0		
	T		0 0 0 + 0 0 1	0 9	
.					
# / / / / / / X X X X X Δ Δ C O N S T A N T S Δ					
C O N	0 5 1 1 0		0		
	0 5 0 9 0 4 8	3 7 4 1 8			
	0 5 0 8 1 8 7	3 0 7 5 3			
	0 5 0 7 4 0 8	1 8 2 2 0			
	0 5 0 6 7 0 3	2 0 0 5 0			
	0 5 0 6 0 6 5	3 0 6 6 0			
	0 5 0 5 4 8 8	1 1 6 4 0			
	0 5 0 4 9 6 5	8 5 3 0 0			
	0 5 0 4 4 9 3	2 8 9 6 0			
	0 5 0 4 0 6 5	6 9 6 6 0			
	0 5 0 3 6 7 8	7 9 4 4 0			
	0 5 0 3 3 2 8	7 1 0 8 0			

DTMB

LARC C.U. SIMULATOR
CODING SHEET

Page: 3

Date:

Programmer:

Problem

Line No.	I	A	M	B	t
	0 5 0 3 0 1 1 9 4 2 1 0				
	0 5 0 2 7 2 5 3 1 7 9 0				
	0 5 0 2 4 6 5 9 6 9 6 0				
	0 5 0 2 2 3 1 3 0 1 6 0				
	0 5 0 2 0 1 8 9 6 5 2 0				
	0 5 0 1 8 2 6 8 3 5 2 0				
	0 5 0 1 6 5 2 9 8 8 9 0				
	0 5 0 1 4 9 5 6 8 8 2 0				
	0 5 0 1 3 5 3 3 5 2 8 0				
	0 5 0 1 2 2 4 5 6 4 3 0				
	0 5 0 1 1 0 8 0 3 1 6 0				
	0 5 0 1 0 0 2 5 8 8 4 0				
	- 5 3 1 1 1 0 0 0 0 0 0				
	0 0 0 0 0 0 0 0 0 0 0				
	0 5 3 1 1 2 0 0 0 0 0 0				
	0 5 0 4 3 4 2 9 4 4 8 2				
	0 2 3 0 2 5 8 5 0 9 2 9				
	0 0 1 0 0 0 0 0 0 0 0				
	0 4 9 3 9 6 8 0 0 0 0 0				
	0 5 0 1 6 6 4 9 9 2 0 0				
	0 5 0 4 9 9 9 9 4 3 2 0				
	0 5 0 9 9 9 9 9 9 3 8				
	0 5 1 1 0 0 0 0 0 0 0 0				
	0 6 1 0 0 0 0 0 0 0 0 0				
	1 0 0 0 0 0 0 0 0 0 0 0				
...	...	...	...	...	...
...	...	...	...	...	...

COMPLEX ADDITION-SUBTRACTION

File Name: ADC SBC

Purpose: To add or subtract two floating-point, single-precision, complex numbers $Z_1 = x_1 + ix_2$ and $Z_2 = y_1 + iy_2$.

Accuracy: The result of the addition or subtraction is obtained to 9 significant figures.

Usage:

(a) Entrance

1. Starting location: ADC (or SBC).
2. Each complex number is represented in two adjacent registers, 02 and 03, 04 and 05 containing the real part and the imaginary part, respectively.

(b) Storage Requirements

1. Five consecutive registers: 01 through 05.
2. Instructions: 6 lines beginning at location ADC (or SBC).

(c) Exit

1. Real part stored in register 02.
2. Imaginary part stored in register 03.

(d) Errors Detected

None.

Timing: 16 μ s

Programmed by: Naomi Millet, David Taylor Model Basin

Method:

$$(X_1 + iX_2) \pm (y_1 + iy_2) = (X_1 \pm y_1) + i(X_2 \pm y_2)$$

COMPLEX MULTIPLICATION

File Name: CPM

Purpose: To multiply two floating-point, single-precision, complex numbers, $Z_1 = X_1 + iX_2$ and $Z_2 = y_1 + iy_2$.

Accuracy: The result of the multiplication is obtained to 9 significant figures.

Usage:

(a) Entrance

1. Starting Location: CPM.
2. Each complex number is represented in two adjacent registers 02 and 03 containing the real part and the imaginary part, respectively.

(b) Storage Requirements

1. Five consecutive registers: 01 through 05.
2. Instructions: 8 lines beginning at location CPM.

(c) Exit

1. Real part stored in register 02.
2. Imaginary part stored in register 03.

(d) Errors Detected

None.

Timing: 56 μ s

Programmed By: Naomi Millet, David Taylor Model Basin

Method:

$$(X_1 + iX_2)(y_1 + iy_2) = (X_1 y_1 - X_2 y_2) + i(X_2 y_1 + X_1 y_2)$$

LARC C.U. ASSEMBLER CODING SHEET

Date:

Problem CPM

45

COMPLEX DIVISION

File Name: CPD

Purpose: To divide two floating-point, single-precision, complex

numbers $Z_1 = X_1 + iX_2$ and $Z_2 = y_1 + iy_2$.

Accuracy: The result of the division is obtained to 9 significant figures.

Usage:

(a) Entrance

1. Starting location: CPD.
2. Each complex number is represented in two adjacent registers, 02 and 03, 04 and 05, containing the real part and the imaginary part, respectively.

(b) Storage Requirements

1. Five consecutive registers: 01 through 05.
2. Instructions: 16 lines beginning at location CPM.
3. Working storage: 4 lines beginning at location COM.

(c) Exit

1. Real part stored in 02.
2. Imaginary part stored in 03.

(d) Errors Detected

None.

Timing: 148 μ s

Programmed By: Naomi Millet, David Taylor Model Basin

Method:

$$\frac{(X_1 + iX_2)}{(y_1 + iy_2)} = \frac{X_1 y_1 + X_2 y_2}{y_1^2 + y_2^2} + i \left(\frac{X_2 y_1 - X_1 y_2}{y_1^2 + y_2^2} \right)$$

DTMB

LARC C.U. ASSEMBLER CODING SHEET

Page: 1

Date:

Programmer: N. Millet

Problem CPD

Line No.	I	A	M	B	t
I / / / / /	X X X X X	Δ C P D	Δ Δ Δ	Δ Δ Δ	Δ Δ Δ
CPD	SS	0 2	COM		
	SS	0 4	COM + 0 0 2		
	M	0 4	9 9 9 + 0 0 4		
	M	0 5	9 9 9 + 0 0 5		
	A	0 4	9 9 9 + 0 0 5		
	M	0 2	COM + 0 0 2		
	M	0 3	COM + 0 0 3		
	A	0 2	9 9 9 + 0 0 3		
	DR	0 2	9 9 9 + 0 0 4		
	F	0 5	COM		
	M	0 5	COM + 0 0 3		
	F	0 3	COM + 0 0 1		
	M	0 3	COM + 0 0 2		
	N	0 3	9 9 9 + 0 0 5		
	DR	0 3	9 9 9 + 0 0 4		
	T		0 0 0 + 0 0 1	0 1	
...	...	...	...	...	...
...	...	...	...	...	...

RECIPROCAL OF A COMPLEX NUMBER

File Name: RCC

Purpose: To calculate the reciprocal of a floating-point, single-precision, complex number, $Z = X_1 + iX_2$.

Accuracy: The reciprocal is obtained to 9 significant figures.

Usage:

(a) Entrance

1. Starting location: RCC.
2. The complex number is represented in two adjacent registers, 02 and 03 containing the real part and the imaginary part, respectively.

(b) Storage Requirements:

1. Four consecutive registers: 01 through 04.
2. Instructions: 10 lines beginning at location RCC.
3. Working storage: 2 lines beginning at location COM.

(c) Exit

1. Real part stored in 02.
2. Imaginary part stored in 03.

(d) Errors Detected

None.

Timing: 104 μ s

Programmed by: Naomi Millet, David Taylor Model Basin

Method:

$$\frac{1}{(X_1 + iX_2)} = \frac{X_1}{X_1^2 + X_2^2} + i \left(\frac{-X_2}{X_1^2 + X_2^2} \right)$$

LARC C.U. ASSEMBLER CODING SHEET

Date:

Problem RCC

49

ABSOLUTE VALUE OF A COMPLEX NUMBER

File Name: MGC

Purpose: To calculate the modulus of a floating-point, single-precision, complex number $Z = X_1 + iX_2$.

Accuracy: $|X_1 + iX_2|$ is obtained to 9 significant figures.

Usage:

(a) Entrance

1. Starting location: MGC.
2. The complex number is represented in two registers, 02 and 03, containing the real part and the imaginary part, respectively.

(b) Storage Requirements

1. Five consecutive registers: 01 through 05.
2. Instructions: 28 lines beginning at location MGC.
3. Constants: 24 lines beginning at location SQC.
4. Working storage: 1 line at STO.

(c) Exit

Results stored in register 02.

Timing: 192 μ s

Programmed By: Naomi Millet, David Taylor Model Basin

Method:

$$|X_1 + iX_2| = \sqrt{X_1^2 + X_2^2}$$

ARGUMENT OF A COMPLEX NUMBER

File Name: PHA

Purpose: To calculate the phase angle of a floating-point, single-precision, complex number, $Z = X_1 + iX_2$.

Restrictions: $10^{-25} < \left| \frac{X_2}{X_1} \right| < 10^{25}$

Accuracy: The phase angle in radians is obtained to 9 significant figures.

Usage:

(a) Entrance

1. Starting location: PHA.
2. Each complex number is represented in two adjacent registers, X_1 in 02 and X_2 in 03 containing the real part and the imaginary part, respectively.

(b) Storage Requirements

1. Five consecutive registers: 01 through 05.
2. Instructions: 47 lines beginning at location PHA.
3. Constants: 20 lines beginning at location ARC.
4. Working storage: 1 line at STO.

(c) Exit

Result stored in register 02.

(d) Errors Detected

None.

Timing: $336 \mu s$

Programmed By: Naomi Millet, David Taylor Model Basin

Method:

$$\Theta = \text{arc tan } \frac{X_2}{X_1}$$

Use Livermore arc tangent routine.

LARC C.U. ASSEMBLER CODING SHEET

Date:

Problem PHA

54

COMMENTS ON THE USE OF THE ASSEMBLY AND SIMULATOR

For the types of routines included in this report, the LARC Input Simulator Assembly (LISA) assembled input for the LARC Instruction Simulator (LIS) and provided an analyzer which listed programming errors in approximately fifteen minutes of UNIVAC I running time. Multiply-defined symbols were not indicated, and any new definitions of a symbol previously defined were ignored. Memory space was reserved for all registers from register zero up to and including the highest-numbered register mentioned in the specific program. Register zero always contains zeros. Other registers cannot be assumed to contain zeros initially.

LIS consists of four parts. Part 1 selects the order-subroutines which are required to simulate the given routine. These order-subroutines are in x-1 assembly language. Part 2 converts these chosen order-subroutines into UNIVAC machine language. Part 3 performs the actual simulation of the LARC instructions. Part 4 edits the results and prepares the output. The running of Parts 3 and 4 of LIS took approximately five minutes for each of the routines included in this report. The accuracy of the computed answers, permitting up to nine significant figures in single-precision floating-point and eleven significant figures in single-precision fixed-point, may be judged by the table following each routine.

LIS provides the programmer with a tape containing the input to Part 3 of LIS. Therefore, most corrections necessary during code-checking may be made on this tape, eliminating the need to assemble and perform Parts 1 and 2 of LIS more than once. However, care should be taken in correcting and adding to this tape since it will contain only the order-subroutines required for simulating those instructions which are in the original input to Part 1. A chart is contained in the report on LIS listing the set of instructions which each order-subroutine can simulate. This will allow the programmer to determine which order-subroutines have been included in the tape containing the input to Part 3.

At the writing of this report, the CCX instruction could not yet be handled by LIS, but it is expected that this instruction will be added in the near future.

APPENDIX A

Optimum Linear Approximation Used in the Square Root Routine

The problem is to approximate the curve $y = f(x)$ by a straight line $y^* = ax + b$ in an interval $x_0 \leq x \leq x_1$, $x \geq 0$, over which $f'(x) > 0$ and $f''(x) < 0$.

(1) To determine the line $y^{**} = a^*x + b^*$ through the points $(x_0, f(x_0))$, $(x_1, f(x_1))$, we have

$$y^{**} = \frac{x}{f(x_0) + f(x_1)} + \frac{x_0 f(x_1) - x_1 f(x_0)}{x_0 - x_1}$$

where

$$a^* = \frac{1}{f(x_0) + f(x_1)}$$

$$b^* = \frac{x_0 f(x_1) - x_1 f(x_0)}{x_0 - x_1}$$

(2) To determine the maximum value of $(y - y^{**})$ in the interval $x_0 \leq x \leq x_1$, $x \geq 0$, let

$$\frac{d(y - y^{**})}{dx} = 0$$

Since

$$y - y^{**} = f(x) - a^*x - b^*$$

then

$$\frac{d(y - y^{**})}{dx} = f'(x) - a^*.$$

The maximum value is attained at the point x^* for which

$$f'(x^*) = a^*$$

or

$$x^* = f'^{-1}(a^*)$$

and

$$\max (y - y^{**}) = f(x^*) - (a^* x^* + b^*) = d^*$$

(3) By choosing the straight line parallel to y^{**}

$$y^* = ax + b$$

for which

$$\max (y - y^*) = \frac{1}{2} d^*$$

where

$$a = a^*$$

$$\begin{aligned} b &= b^* + \frac{1}{2} d^* \\ &= \frac{b^*}{2} - \frac{a^* x^*}{2} + \frac{f(x^*)}{2} \\ &= \frac{1}{2} (b^* - a^* x^* + f(x^*)). \end{aligned}$$

Under these circumstances

$$(Y - y^*) = \frac{1}{2} d^* \text{ at } x = x_0, x_1; (y - y^*) = \frac{1}{2} d^* \text{ at } x = x^*,$$

and we have the optimum straight line to approximate $f(x)$ in the specified interval.

Now

$$y = \sqrt{x}$$

$$a = \frac{1}{\sqrt{x_0} + \sqrt{x_1}}$$

$$b = \frac{1}{2} \left(\frac{x_0 \sqrt{x_1} - x_1 \sqrt{x_0}}{x_0 - x_1} - a^* x^* + f(x^*) \right)$$

$$x^* = \frac{1}{4a^{*2}}; f(x^*) = \frac{1}{2a^*}; a^* = a$$

$$b = \frac{1}{2} \left(\frac{x_0 \sqrt{x_1} - x_1 \sqrt{x_0}}{x_0 - x_1} - \frac{1}{4a^*} + \frac{1}{2a^*} \right)$$

$$= \frac{1}{2} \left(\frac{4 x_0 \sqrt{x_1} - x_1 \sqrt{x_0} + (x_0 - x_1) (\sqrt{x_0} + \sqrt{x_1})}{4 (x_0 - x_1)} \right)$$

$$= \frac{1}{8 (x_0 - x_1)} \left[(5 x_0 - x_1) \sqrt{x_1} - (5 x_1 - x_0) \sqrt{x_0} \right]$$

OPTIMUM LINEAR APPROXIMATION

UCRL

n	a	b	max y-y*	n	a	b	max y-y*
1	.4142	.5947	.009	1	.4131	.596	.009
2	.3178	.7826	.004	2	.3175	.784	.004
3	.2679	.9306	.003	3	.2678	.931	.003
4	.2361	1.0574	.002	4	.2358	1.059	.002
5	.2134	1.1702	.001	5	.2133	1.170	.002
6	.1963	1.2729	.001	6	.1962	1.273	.001
7	.1827	1.3678	.001	7	.1827	1.368	.001
8	.1716	1.4565	.001	8	.1716	1.457	.001
9	.1623	1.5400	.001	9	.1623	1.540	.001

APPENDIX B

Derivation of the Iterative Procedure Used in the Cube Root Subroutine ⁶

Given a real number, N , the problem is to find a number, x , such that $x^3 = N$. Consider the function

$$(1) f(x) = x^3 - N.$$

Expanding $f(x)$ in a Taylor series about the point x_0 , we can write

$$(2) f(x) = f(x_0) + (x-x_0) f'(x_0) + (x-x_0)^2 \frac{f''(x_0)}{2!} + \dots$$

Truncating after the third term and using the fact that $f(x) = 0$,

$$(3) 0 = f(x_0) + (x-x_0) f'(x_0) + (x-x_0)^2 \frac{f''(x_0)}{2!}$$

or

$$(4) -f(x_0) = (x-x_0) \left[f'(x_0) + (x-x_0) \frac{f''(x_0)}{2!} \right].$$

Hence,

$$(5) x - x_0 = - \frac{f(x_0)}{f'(x_0) + (x-x_0) \frac{f''(x_0)}{2!}}$$

and

$$(6) x = x_0 - \frac{f(x_0)}{f'(x_0) + (x-x_0) \frac{f''(x_0)}{2}}$$

To eliminate x from the right-hand side of this equation, we use the first-order approximation obtained by terminating the Taylor expansion of $f(x)$ after the second term, namely

$$(7) (x - x_0) = \frac{-f(x_0)}{f'(x_0)}.$$

Substituting this expression in Equation (6) we have

$$(8) \ x = x_0 - \frac{f(x_0)}{f'(x_0) - \frac{f(x_0) f''(x_0)}{2f'(x_0)}}$$

or

$$(9) \ x = x_0 - \frac{2f(x_0) f'(x_0)}{2[f'(x_0)]^2 - f(x_0) f''(x_0)}$$

Note that if $f''(x_0) = 0$, Equation (9) reduces to the Newton-Raphson formula

$$x = x_0 - \frac{f(x_0)}{f'(x_0)}.$$

Since from Equation (1) we have $f'(x) = 3x^2$ and $f''(x) = 6x$, we can substitute these expressions in Equation (9), obtaining

$$x = x_0 - \frac{2(x_0^3 - N)(3x_0^2)}{2(9x_0^4) - (x_0^3 - N)(6x_0)}$$

which reduces to

$$x = \frac{2N + x_0^3}{2x_0^3 + N} \cdot x_0$$

This is the procedure used in the Cube Root Routine, with $x_0 = .7$ in the initial approximation.

REFERENCES

1. Dawson, C., Johnson, J. P., and Mejia, R., "LARC Instruction Simulator," Applied Mathematics Laboratory Report 77 (Dec 1958).
2. Johnson, J. P., "LISA (LARC Instruction Simulator Assembly)," David Taylor Model Basin Report 1283 (Jan 1959).
3. Scarborough, J. B., "Numerical Mathematical Analysis," The Johns Hopkins Press, Baltimore (1950), p. 199.
4. A Los Alamos Scientific Laboratory Publication, LA 1943 (Aug 1955).
5. U.S. Department of Commerce, National Bureau of Standards, "Tables of Chebyshev Polynomials," (Dec 1952).
6. Householder, A. S., "Principles of Numerical Analysis," McGraw-Hill Book Company, Inc., New York (1953), p. 125.

INITIAL DISTRIBUTION

Copies

- 9 CHBUSHIPS
 - 3 Tech Info Br (Code 335)
 - 3 Computer Sys & Appli Br (Code 732)
 - 1 L 5 Mgt (Code 320)
 - 1 Field Activities (Code 700)
 - 1 Nuclear Propulsion (Code 1500)
- 1 CHBUWEPs
- 1 CHBUSANDA
- 1 CHBUCEN
- 1 CHONR
- 1 NAVSHIP BSN
- 1 NAVSHIP CHASN
- 1 NAVSHIP LBEACH
- 2 NAVSHIP NYK
 - 1 Matl Lab
- 1 NAVSHIP MARE
- 1 NAVSHIP PTSMH
- 1 NAVSHIP SFRAN
- 1 NAVSHIP PHILA
- 1 NAVSHIP PUG
- 1 NAVSHIP PEARL
- 1 DTMB (UERG Code 780)
- 1 CO & DIR, USNEL
- 1 CO & DIR, USNRDL
- 1 CO & DIR, USNAVTRADEVCE
- 1 CO & DIR, USNMDL
- 1 CO & DIR, USNUSL
- 1 CO, USNCML
- 1 USNWEPLAB, Dahlgren
- 3 CDR, USNOTS, China Lake
 - 1 Michelson Lab (Code 5038)
 - 1 Attn: Library
- 1 CDR, USNOL

Copies

- 1 CO & DIR, USNEES
- 1 DIR, USNRL
- 1 SUPT, USNAVPGSCOL
 - 1 Attn: Lib, Tech Rpt Sec
- 1 SUPT, USNA
- 2 CG, Aberdeen PG
- 1 DIR, Natl BuStand
- 2 New York Univ, New York
 - 1 Inst of Math Sciences
 - 1 AEC Computing Facility
- 2 Univ of Illinois, Urbana
 - 1 Dept of Math
 - 1 Digital Computer Lab
- 1 Hd, Nuclear Physics Combustion Engineering, Inc., Windsor, Conn.
- 1 DIR, Westinghouse Elec Corp, Bettis Atomic Power Div, P.O. Box 1468 Pittsburgh
- 6 Lawrence Radiation Lab Livermore, Calif
- 1 Remington Rand UNIVAC Div of Sperry Rand Electronic Computer Dept New York
- 1 Knolls Atomic Power Lab Genl Elec Co Math Analysis Unit Schenectady
- 10 CDR, ASTIA

David Taylor Model Basin. Report 1303. A BASIC SET OF MATHEMATICAL SUBROUTINES FOR LARC, by Naomi C. Millet. Mar 1961. iv, 63p. illus., tables, UNCLASSIFIED This report presents a basic set of mathematical subroutines for LARC selected from those programmed by personnel of the University of California Radiation Laboratory, Remington Rand and the Applied Mathematics Laboratory of the David Taylor Model Basin. Each subroutine was analyzed, then code-checked on UNIVAC I using LARC assembly and simulation routines (LISA and LIS). Information is given for each subroutine in a form for easy application. Derivations of numerical methods used, tables of computed values obtained from running the subroutines, and the true values are included.	1. Digital computers-- LARC--Programming I. Millet, Naomi C.
David Taylor Model Basin. Report 1303. A BASIC SET OF MATHEMATICAL SUBROUTINES FOR LARC, by Naomi C. Millet. Mar 1961. iv, 63p. illus., tables, UNCLASSIFIED This report presents a basic set of mathematical subroutines for LARC selected from those programmed by personnel of the University of California Radiation Laboratory, Remington Rand and the Applied Mathematics Laboratory of the David Taylor Model Basin. Each subroutine was analyzed, then code-checked on UNIVAC I using LARC assembly and simulation routines (LISA and LIS). Information is given for each subroutine in a form for easy application. Derivations of numerical methods used, tables of computed values obtained from running the subroutines, and the true values are included.	1. Digital computers-- LARC--Programming I. Millet, Naomi C.