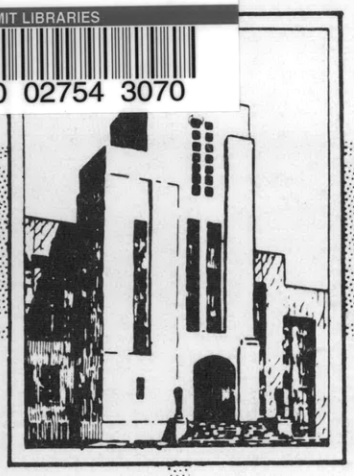

DEPARTMENT OF THE NAVY

DAVID TAYLOR MODEL BASIN

HYDROMECHANICS

AERODYNAMICS

STRUCTURAL MECHANICS

APPLIED MATHEMATICS

LISA

(LARC INSTRUCTION ASSEMBLY)

by

Joseph P. Johnson

APPLIED MATHEMATICS LABORATORY

RESEARCH AND DEVELOPMENT REPORT

January 1959

Report 1283

LISA

(LARC INSTRUCTION ASSEMBLY)

by

Joseph P. Johnson

January 1959

Report 1283

ACKNOWLEDGMENT

The author gratefully acknowledges the assistance of Naomi Millet in the preparation of this assembly.

TABLE OF CONTENTS

ABSTRACT

This report presents a description of and operating instructions for LISA (LARC Instruction Assembly). An assembly system for the UNIVAC, LISA uses the standard LARC mnemonic instruction code as input and produces LARC instructions and an analyzer as output. The output from LISA may also be used as input into LIS (LARC Instruction Simulator). The analyzer lists, side by side, the absolute line number, absolute LARC orders, source code as written by the programmer, and cross references to the absolute line number.

INTRODUCTION

A computer such as the LARC, which has a large set of numeric commands, requires a pseudocode of instructions and an assembly system to aid the programmer in writing codes. If the assembly system produces an analyzer, this device is an additional aid in locating programming errors.

The LISA (LARC Instruction Assembly) operates on the UNIVAC computer and uses a pseudocode to produce LARC instructions and an analyzer. The LARC instructions produced can be used as input for LIS (LARC Instruction Simulator), and the analyzer lists the generated LARC numeric code, the pseudocode, and cross references. Thus, prior to the LARC's arrival, we have an operational assembly system which has a pseudocode as input and facilitates code checking by providing an analyzer and by detecting certain types of errors during the assembly.

The pseudocode which the programmer writes as input to the assembly is called the Source Code, and the set of LARC instructions produced is called the Object Code. The Source Code is written in LISA language, which makes use of the standard mnemonic instruction code adopted for the LARC Computing Unit, plus certain other conventions which will be explained here.

The assembly language has provisions for:

a) the use of symbolic as well as relative addresses

b) the setting of symbolic as well as relative addresses

c) absolute memory assignment by the programmer.

The Object Code contains LARC instructions and LIS parameters. Because actual LARC instructions are used, the routine which produces them is self-contained and modifications can be made without altering the entire assembly. Hence, the production of input programs for the LARC itself can be accomplished with ease.

GENERAL SPECIFICATIONS

HEADER ITEM

It is usually desirable to arrange input, instructions and instruction constants, and output into sections, and to identify each section distinctly. One device which enables the programmer to do this is the header item. The header item gives a unique name to a section and allows the programmer to place this section in the computer memory at any location. * In LISA the header item consists of two UNIVAC words, the first locates the section in the memory, and the second names it. Twenty-six header items may be used with the assembly. The number of lines of instructions within a section must be less than or equal to 2500.

---

* An item in this assembly system will always consist of 24 characters.

The header item for a section has the following format:

w	x	x	x	x	x	/	y	y	y	y	y	z	z	z	z	z	z	z	z	z	z	z	z

where

1. W is one of five symbols.

A. W ≡ I means this is the instruction section where simulation begins.

B. W ≡ / means this is a section of instructions.

C. W ≡ Δ means this is a general storage area.

Example:

Δ	0	0	1	0	0	/	1	0	0	0	0		O	U	T	P	U	T					
D	E	L																					

Upon detection of this header, the assembly generates 100 lines of zeros beginning at line DEL; that is, at line 10,000. (See 2B and 3.)

D. W ≡ # means this is a section of constants.

E. W ≡ ; means this is an origin for the A and B registers.

Example:

;						0	0	0	2	0													

Except for register 00, all registers in instructions following this header begin at location 21; i.e., register 01 becomes 21.

2. XXXXX is one of the following:

A. /////, if W is not a space; that is, if the header item does not denote a general storage area.

B. An integer, representing the number of words of zeros if the header is a general storage area.

3. YYYYY is either the starting line number for the contents of the section or a register origin.

4. ZZZZZZZZZZZ, the second UNIVAC word, is the name assigned to the section.

SENTINEL ITEM

An item of periods is the intermediate sentinel indicating the end of a section. A new section can either immediately follow this sentinel or begin in the next UNIVAC block, the next LISA coding sheet. The final sentinel consists of two items of periods.

SEQUENCE OF GENERATED LARC CODE

The sections during simulation with LIS appear in the UNIVAC memory in the same order as in the assembly Source Code. However, memory locations in the analyzer appear in numerical order. The time required to assemble a program is shortened if the memory locations referred to in the header items are in an ascending sequence in the Source Code.

INSTRUCTION FORMAT

A programmer often needs to refer to addresses without knowing their exact line number. To resolve this difficulty, he may leave the address fields blank until the correct line number

has been determined or he may place mnemonics in them. In either case, he must return later to replace these addresses with the correct numeric line number. This assembly provides for interpreting symbolic line numbers and mnemonic orders in the Source Code and thus saves the programmer time in coding and helps to avoid bookkeeping errors.

The pseudoinstruction order has the form

I A M B T

where I is the mnemonic order, A and B are the registers, M is the address location, and T is the tracing mode digit. (The mnemonic orders included in LISA I are listed in Appendix I.) The A and B registers are numeric; however, when used in the M part of an order they are written as:

99900NN or

;00NN

where NN is the register number.

To identify the location of the instruction, the line number found to the left of the instruction on the coding sheet can be used. It has the following form:

C C C

Up to 120 CCC tags are handled by the assembly.

The M address has the following form:

(1) C C C X N N N

where:

1) CCC is a set of three alphanumeric characters. (The first thousand lines of absolute coding can be referred to by setting CCC = 000.)

2) XNNN is the line number relative to the alphanumeric origin. (X is one of four symbols +, Δ, 0, - and NNN is a number.)

An M address of the form (1) is determined as follows:

A. When X ≡ +: The memory address referred to is NNN lines succeeding CCC.

B. When X ≡ Δ: The memory address is CCC.

C. When X ≡ 0: The memory address is CCC.

D. When X ≡ -: The memory address referred to is NNN lines preceding CCC.

Note that Δ and 0 are equivalent symbols. Example:

Line No.				I			A			M						B		T	
				F			1	0		A	C	E	-	0	0	1	0	0		1
				M			0	1		A	1	0	+	0	4	0	0	3		
A	C	E				S	1	5		D	O	G					0	0		4

(ACE-001 refers to the line immediately preceding line ACE)

The legitimate symbols for tracing mode are a period, a number greater than zero, or an Ignore.

A. T = 0 means that the assembly is to insert a period.

B. T = . means that the assembly is to insert a period.

C. T = Δ means that the assembly is to insert a period.

D. T equals a number greater than zero means that the assembly is to insert that number into the tracing mode.

E. T = # means that a set of twelve numerics, such as a data constant, is understood to occupy digit positions 5 through 16 of the item and must be written exactly as it will appear in the LARC. It cannot be symbolic.

F. T = ɣ means that the assembly is to insert an Ignore. If any other symbol is found in the tracing mode position, and when any other part of the order is not legitimate, an indication is made in the analyzer by the use of an asterisk.

ANALYZER LISTINGS

The analyzer lists, side by side, the corresponding absolute line number, absolute LARC orders, the Source Code as written by the programmer, and cross references to the absolute number. On separate pages, registers referred to and undefined tags are listed. The page number, routine name, and date are listed at the top of each analyzer page. Routine name and date are typed during assembly.

The analyzer calls attention to errors by placing an asterisk beside the error, whether it appears in the tracing mode, instruction, address, or register field. See example in Appendix II.

Some errors which are located and identified are: the use of undefined addresses, alphabetics in the tracing mode column, alphabetics as a register address, and nonexistent orders.

FUTURE IMPROVEMENTS

The present assembly provides for no automatic call-in of routines from a library tape. The programmer must now merge these routines himself. Because of the possibility of overlapping sections, the programmer may have to count lines in order to place the correct origins in the header items. A library tape and automatic assignment of sections in the memory should be the next extension to LISA. Such an extension should have LISA assembly instructions and instruction constants into different LARC boxes, and should allow the programmer to specify the location of some sections, if he wishes, and let the assembly determine all others.

APPENDIX I

SAMPLE PROBLEM AND LARC MNEMONIC CODE

SAMPLE CODE

Compute $\sum_{i=0}^{4} a_j \; x_j^i = \left\{\left[\left[a_4 \; x_j + a_3 \right] x_j \; + a_2 \right] \; x_j \; + a_1 \right\} \; x_j \; + a_0$

where

x_j has 10 values beginning in XXJ,

a_j has 5 values beginning in AAJ,

B boxes are located in BBX and BBA

instructions begin in 3000 at line ACE, and

output is in 12500 - 12509 with DEL equal to 12500.

The source code and analyzer with explanations follow. Errors have been purposely inserted into the source code in order to illustrate the manner in which they are indicated in the analyzer.

DTMB
LARC C.U. ASSEMBLY
CODING SHEET

Page
Date:
Programmer:

Problem

Line No.		I		A		M		B		t	
I ///	///	///	0	3 0	0	0 INSTRUCTIONS				N	S
ACE		F		0 1		BBX				1	
		F		0 2		BBA					
		F		0 3		AA1		0 2			
MUL		M		0 3		XXJ		0 1		A	
		A		Z Z		AAI - 1		0 2			
		BOT				MUL					
		S				DEL		0 1			
		BIT				ACE + 1					
END		H						AA			
BBX		0 1 0 0	0 0	1						#	
BBA		0 0 4 0	0 0	1						#	
XXJ		0 5 2 3	7 2	1	0 9					#	
		0 5 2 3	7 2	1	1 4					#	
		0 5 2 3	7 2	1	1 9					#	
		0 5 2 3	7 2	1	2 4					#	
		0 5 2 3	7 2	1	2 9					#	
		0 5 2 3	7 2	1	3 4					#	
		0 5 2 3	7 2	1	3 9					#	
		0 5 2 3	7 2	1	4 4					#	
		0 5 2 3	7 2	1	4 9					#	
		0 5 2 3	7 2	1	5 4					#	
AAI		0 5 2 4	2 7	1						#	
		- 4 6 2	5 9	2	2 9 1					#	
		- 4 6 3	1 4	1	7					#	
		0 5 0 1								#	
		0 5 7	4 6	1	3 9 2					#	
. . . .		. . . .	. .	.	. . . .	. . . . .	. .	. . .	.	-	.
△ 0 0 0 DEL		1 0 /	0 2	5	0 0	/ OUTPUT					

DTMB
LARC C. U. ASSEMBLY
CODING SHEET

Page:
Date:
Programmer:

Problem

Line No.		I		A		M		B		t	

P. 1 NAME..... DATE.....

00080 03008 H ,

P. 2 NAME..... DATE.....
OUTPUT

LINE		TAG		
02500	00C000000000	DEL	03006 S	'
02501	00C000000000			
02502	00C000000000			
02503	00C000000000			
02504	00C000000000			
02505	00C000000000			
02506	00C000000000			
02507	00C000000000			
02508	00C000000000			
02509	000000000000			

P. 3 NAME..... DATE.....

INSTRUCTIONS

```
LINE   LARC ORDER  TAG  I  A     M     B  T
03000 1430100030O9 ACE  F  01 BBX     00 1
03001 .43020003010      F  02 BBA     00      03007 BIT ,
03002 .430302*****.     F  03 AAI     02
03003 A23030103011 MUL  M  03 XXJ     01 A*.  03005 BOT ,
03004 .02ZZ0203020      A  ZZ*AAI-001 02
03005 .**020003003      BOT*02 MUL    00
03006 .4C030102500      S  03 DEL     01
03007 .8C010003001      BIT 01 ACE*001 00
03008 .9900AA00000 END  H  00         AA*
03009 01C000100000 BBX  010000100000    #  03000 F  ,
03010 004000100000 BBA  004000100000    #  03001 F  ,
03011 052372109000 XXJ  052372109000    #  03003 M  ,
03012 052372114000      052372114000    #
03013 052372119000      052372119000    #
03014 052372124000      052372124000    #
03015 052372129000      052372129000    #
03016 052372134000      052372134000    #
03017 052372139000      052372139000    #
03018 052372144000      052372144000    #
03019 052372149000      052372149000    #
03020 052372154000      052372154000    #  03004 A  ,
03021 052427100000 AAI  052427100000    #
03022 -46259229100      -46259229100    #
03023 -46314170000      -46314170000    #
03024 05C100000000      050100000000    #
03025 057461392000      057461392000    #
```

UNDEFINED ADDRESS

ALPHABETIC IN TRACING MODE

ALPHABETIC IN A REGISTER

NON EXISTENT ORDER

ALPHABETIC IN B REGISTER

P. # NAME..... DATE.....

REGISTERS

99900	03008	A	;									
99901	03000	A	;	03003	B	;	03006	B	;	03007	A	;
99902	03001	A	;	03002	B	;	03004	B	;	03005	A	;
99903	03002	A	;	03003	A	;	03006	A	;			
999AA	03008	B	;									
999ZZ	03004	A	;									

03002 AA1 ,

00 SK SKIP	10	20 8 ~~MX~~ MXR $m_m \times a_a \rightarrow A_a$	30 32 DX $a_a \div m_m \rightarrow A_a$	40 4 S $a_a \rightarrow M_m$	50 4 CX $FL \rightarrow Fx$ convert $a_a \rightarrow A_a$ Scale factor M	60 4 EOP $m_{mI} \rightarrow A_{aI}$	70 12/4 TE M→C if $a_a = a_{a+1}$	80 12/8 BIT N-1 → N M→C if N-1 ≠ 0 Δ+D→Δ if N-1 ≠ 0	90 8 T M→C
01 4 AX $m_m + a_a \rightarrow A_a$	11 4 NX $a_a - m_m \rightarrow A_a$	21 12 MXE $m_m \times a_a \rightarrow A'_a$	31 36 DXE $a_a \div m_m \rightarrow A_a$ Remainder → A_{a+1}	41 4 SN $-a_a \rightarrow M_m$	51 4 C $Fx \rightarrow FL$ convert $a_a \rightarrow A_a$ Scale factor M	61 4 EA $m_{mA} \rightarrow A_{aA}$	71 12/4 TG M→C if $a_a > a_{a+1}$	81 12/8 BDT N-1 → N M→C if N-1 ≠ 0 Δ-D→Δ if N-1 ≠ 0	91 12 TR M+1 → C T, C+1 → M [Return control]
02 4 A $m_m \oplus a_a \rightarrow A_a$	12 4 N $a_a \ominus m_m \rightarrow A_a$	22 12 MR $m_m \otimes a_a \rightarrow A_a$ rounded	32 28 DR $a_a \oplus m_m \rightarrow A_a$ rounded	42 4 SM $\lvert a_a \rvert \rightarrow M_m$	52 4 PR $a_a \cdot 10^{-M} \rightarrow A_a$	62 4 EB $m_{mB} \rightarrow A_{aB}$	72 12/4 TZ M→C if $a_a = 0$	82 12/8 BIC N-1 → N M→C if N-1 = 0 Δ+D→Δ if N-1 ≠ 0	92 8 TB C → A_{aM} M → C [Set B for return control]
03 4 AM $\lvert m_m \rvert \oplus a_a \rightarrow A_a$	13	23 8 M $m_m \otimes a_a \rightarrow A_a$	33	43 4 F $m_m \rightarrow A_a$	53 4 PL $a_a \cdot 10^{M} \rightarrow A_a$	63 4 EAB $m_{mAB} \rightarrow A_{aAB}$	73 12/4 TGE M→C if $a_a > 0$	83 12/8 BDC N-1 → N M→C if N-1 = 0 Δ-D→Δ if N-1 ≠ 0	93
04 4 AU $m_m \oplus a_a \rightarrow A_{a+1}$	14 4 NU $a_a \ominus m_m \rightarrow A_{a+1}$	24 8 MU $m_m \otimes a_a \rightarrow A_{a+1}$	34 28 ~~DD~~ DVR $a_a \oslash m_m \rightarrow A_{a+1}$	44	54	64 4 EM $m_{mM} \rightarrow A_{aM}$	74 12/4 TLZ M→C if $a_a < 0$	84	94
05 12 AAX $m'_m + a'_a \rightarrow A'_a$	15 12 NNX $a'_a - m'_m \rightarrow A'_a$	25 12 ME $m_m \otimes a_a \rightarrow A'_a$	35 192 DDX $a'_a \div m'_m \rightarrow A'_a$	45 8 SS $a'_a \rightarrow M'_m$	55 12 CCX $FL' \rightarrow Fx'$ a'_a convert $\rightarrow A_a$ Scale factor M	65 8 EL $a_{a-1} \xrightarrow{(m)} A_a$	75 12/8 ~~TGE~~ TTE M→C if $a'_a = a'_{a+2}$	85 4 BI Δ+D→Δ	95 12/4 TF M→C if Flip flop A_a set
06 16 AA $m'_m \oplus a'_a \rightarrow A'_a$	16 16 NN $a'_a \ominus m'_m \rightarrow A'_a$	26 36 MMX $m'_m \times a'_a \rightarrow A'_a$	36 176 DD $a'_a \oslash m'_m \rightarrow A'_a$	46 8 SSN $-a'_a \rightarrow M'_m$	56 12 CC $Fx' \rightarrow FL'$ a'_a convert $\rightarrow A'_a$ Scale factor M	66 8 EU $a_{a+1} \xrightarrow{(m)} A_a$	76 12/8 ~~TGE~~ TTG M→C if $a'_a > a'_{a+2}$	86 4 BD Δ-D→Δ	96 4 RF Reset f.f. A_a
07	17	27 36 MH $m'_m \otimes a'_a \rightarrow A'_a$	37 60 ~~DSE~~ $a'_a \oslash m_m \rightarrow A'_a$	47 8 SSM $\lvert a'_a \rvert \rightarrow M'_m$	57 8 PPR $a'_a \cdot 10^{-M} \rightarrow A'_a$	67	77	87	97 4 SF Set f.f. A_a.
08	18	28	38	48 8 FF $m'_m \rightarrow A'_a$	58 8 PPL $a'_a \cdot 10^{M} \rightarrow A'_a$	68	78	88	98 SAP
09 FV $5d.d \rightarrow A_a$	19 FVK $12d.d \rightarrow A_a$	29 SV $a_a \rightarrow 5dd.$	39 SVK $a_a \rightarrow 12dd.$	49	59 12 PPC L Circular Shift a'_a M places [Includes sign]	69	79	89	99 H STOP

LARC COMPUTING UNIT INSTRUCTION CODE

APPENDIX II

OPERATING INSTRUCTIONS

Servo Allocations (Initially)

Servo 1 LISA Assembly Instructions

Servo 2 Source Code Input

Servo 3-9 Blanks

The LISA Assembly consists of three passes, two edits and a sort routine. The Pratt-Goetz 2-word 3-way sort, with minor modifications, is used to sort references.

Normally, after the first pass the Source Code is removed from servo 2 and a blank mounted. Block subdivider 2 is set and the analyzer is written on servo 2.

If it is desired to obtain the Object Code for direct use on LARC, set breakpoint 1 initially and force transfer. Remove the Object Code from servo 9 and mount a blank. Depress the start bar to continue the routine and obtain the analyzer on servo 2.

RERUN PROCEDURE

Restart is possible at three places: the sort, pass three, and the analyzer edit. These are identified by type-outs on the SCP. See SCP output on page 20 for the example given in Appendix I.

To rerun from the sort, save tapes 2 and 9, mount new blanks, and force transfer of breakpoint 2.

To rerun from pass three, save tapes 6 and 9, and force transfer on breakpoint 3.

To rerun from the analyzer edit, save tapes 4 and 6 and force transfer on breakpoint 4.

UNITYPING INSTRUCTIONS

All uncoded positions should be filled with spaces. The typist types straight across the coding sheet, two 12-digit words to the line. Each section of the coding paper, divided by dark horizontal lines, represents one blockette (10 words) of information. There are six such divisions on one coding page; thus each page represents a UNIVAC block.

```
LISA PASS=1

PASS 2

BLANK ON SERVO 2

LIS


NAME    NAME.....................

DATE    DATE.........

LIS ON SVO 9

BLANK ON SVO 9


SORT


PASS 3


EDIT


END
```

INITIAL DISTRIBUTION

Copies

8 CHBUSHIPS, library (Code 312)
- 2 Tech Library
- 1 Tech Asst to Chief (Code 106)
- 3 Electronic Computer Div (Code 280)
- 1 Asst Chief for Field Activities (Code 700)
- 1 Asst Chief for Nuclear Propulsion (Code 1500)

1 Chief, Bureau of Aeronautics
Washington 25, D. C.

1 Chief, Bureau of Census
Suitland, Maryland

1 Chief, Bureau of Ordnance
Washington 25, D. C.

1 Chief, Bureau of Supplies and Accounts
Washington 25, D. C.

1 Chief of Naval Research
Washington 25, D. C. (USNONR)

1 CDR, Boston Naval Shipyard
Boston, Massachusetts

1 CDR, Charleston Naval Shipyard
Charleston, South Carolina

1 CDR. Long Beach Naval Shipyard
Long Beach, California

2 CDR, New York Naval Shipyard
Naval Base, Brooklyn 1, N. Y.
- 1 Material Laboratory

1 CDR, Mare Island Naval Shipyard
Vallejo, California

1 CDR, Norfolk Naval Shipyard
Portsmouth, Virginia

1 CDR, San Francisco Naval Shipyard
San Francisco, California

1 CDR, Philadelphia Naval Shipyard
Philadelphia, Pennsylvania

1 CDR, Portsmouth Naval Shipyard
Portsmouth, New Hampshire

1 CDR, Puget Sound Naval Shipyard
Bremerton, Washington

1 CDR, Pearl Harbor Naval Shipyard
Navy No. 128, FPO
San Francsico, California

1 CO & DIR, U. S. Naval Boiler
and Turbine Laboratory
Naval Base
Philadelphia, Pennsylvania

1 CO & DIR, U. S. Navy
Electronics Laboratory
San Diego 52, California

1 CO & DIR, U. S. Naval
Radiological Defense Laboratory
San Francisco, California

1 CO & DIR, U. S. Naval
Trg Device Ctr, Computer Br
Port Washington, N. Y.

Copies

1 CO & DIR
U. S. Navy Mine Defense Laboratory
Panama City, Florida

1 CO & DIR
U. S. Navy Underwater Sound Laboratory
Fort Trumbull
New London, Connecticut

1 CO, U. S. Naval Computing
Machine Laboratory
St. Paul 4, Minnesota

1 CDR, U. S. Naval Proving Ground
Dahlgren, Virginia

3 CDR, USNOTS, China Lake Calif.
1 Michelson Lab (Code 5038)
1 Attn: Library

1 CDR, U. S. Naval Ordnance Laboratory
White Oak, Silver Spring, Maryland

1 Director, U. S. Naval Engineering
Experiment Station
Annapolis, Maryland

1 Director, U. S. Naval Research Lab
Washington 25, D. C.

1 Superintendent, U. S. Naval
Postgraduate School
Monterey, California
1 Attn: Library, Tech Reports
Section

1 SUPT, U. S. Naval Academy
Dept of Math
Annapolis, Maryland

1 Commanding General
Aberdeen Proving Ground
Aberdeen, Maryland

1 Director, National Bureau of Standards
Washington 25, D. C.

Copies

1 Director, Lewis Research Cente
NASA
Cleveland 11, Ohio

2 New York University
New York, N. Y.
1 Inst of Math Sciences
1 AEC Computing Facility

1 University of California
Berkeley, California

2 University of Illinois
Urbana, Illinois
1 Dept of Math
1 Electronic Digital
Computer Project

2 Harvard University
Cambridge, Massachusetts
1 Dept of Math
Attn: Prof. J. L. Walsh
1 Computation Laboratory

2 MIT Computation Center
Cambridge, Massachusetts
1 Attn: Dr. F. M. Verzuh
Room 26-142
1 Attn: Document Room
26-270

1 Midwest Research Institute
425 Volker Boulevard
Kansas City, Missouri
1 Attn: Mr. Yudell L. Luke

1 Princeton University
Princeton, New Jersey
1 Attn: Prof. H. J. Maehly
Chief of Computer
Project

1 Head of Nuclear Physics
Combustion Engineering, Inc.
Windsor, Connecticut

Copies

1 Westinghouse Electric Corporation
Bettis Atomic Power Division
P. O. Box 1468
Pittsburgh 30, Pennsylvania

2 Argonne National Laboratory
P. O. Box 299
Lemont, Illinois

1 Illinois Institute of Technology
Armour Research Foundation
Chicago 16, Illinois

1 Battelle Memorial Institute
Columbus, Ohio

1 Brookhaven National Laboratory
Upton, L. I., New York

1 Curtiss-Wright Corp. Res Div
Clifton, New Jersey

1 Douglas Aircraft Co., Inc.
Santa Monica Div
Santa Monica, California

1 Lockheed Aircraft Corp
Van Nuys, California

1 Remington Rand Univac
Div of Sperry Rand
Electronic Computer Dept
New York 10, New York

1 Ramo-Wooldridge Corp
Los Angeles, California

1 Rand Corporation
1700 Main Street
Santa Monica, California

1 Sandia Corporation, Library
Albuquerque, New Mexico

1 Hudson Laboratories
Columbia University
Dobbs Ferry, New York

Copies

1 United Aircraft Corporation
400 Main Street
East Hartford, Connecticut

1 Vitro Corporation of America
120 Wall Street
New York, N. Y.

1 Remington Rand Univac
Div of Sperry Rand Corp
Engineering Res Associates Div.
St. Paul 4, Minnesota

1 IBM Corporation
New York, N. Y.

1 Knolls Atomic Power Laboratory
General Electric Co.
Math Analysis Unit
Schenectady, N. Y.

1 Lincoln Lab, B-125
Lexington, Massachusetts

1 Dr. Franz Alt
Mathematical Computation Lab
National Bureau of Standards
Washington, D. C.

1 IBM Corporation
Prof. Dr. E. G. Kogbetliantz
425 Park Avenue - 10th floor
New York 22, N. Y.

1 George Washington University
Logistics Research Project
707 22nd Street, N. W.
Washington 7, D. C.

1 Johns Hopkins University
Applied Physics Laboratory
Silver Spring, Maryland

1 University of California
Librarian, Numerical Analysis
Los Angeles 24, California

David Taylor Model Basin. Report 1283.
LISA (LARC INSTRUCTION ASSEMBLY) by Joseph P. Johnson. January 1959. iv, 23p. UNCLASSIFIED

This report presents a description of and operating instructions for LISA (LARC Instruction Assembly). An assembly system for the UNIVAC, LISA uses the standard LARC mnemonic instruction code as input and produces LARC instructions and an analyzer as output. The output from LISA may also be used as input into LIS (LARC Instruction Simulator). The analyzer lists, side by side, the absolute line number, absolute LARC orders, source code as written by the programmer, and cross references to the absolute line number.

1. Digital computers – UNIVAC-LARC – Automatic coding
2. Automatic programming systems – LISA

I. Johnson, Joseph P.

David Taylor Model Basin. Report 1283.
LISA (LARC INSTRUCTION ASSEMBLY) by Joseph P. Johnson. January 1959. iv, 23p. UNCLASSIFIED

This report presents a description of and operating instructions for LISA (LARC Instruction Assembly). An assembly system for the UNIVAC, LISA uses the standard LARC mnemonic instruction code as input and produces LARC instructions and an analyzer as output. The output from LISA may also be used as input into LIS (LARC Instruction Simulator). The analyzer lists, side by side, the absolute line number, absolute LARC orders, source code as written by the programmer, and cross references to the absolute line number.

1. Digital computers – UNIVAC-LARC – Automatic coding
2. Automatic programming systems – LISA

I. Johnson, Joseph P.

David Taylor Model Basin. Report 1283.
LISA (LARC INSTRUCTION ASSEMBLY) by Joseph P. Johnson. January 1959. iv, 23p. UNCLASSIFIED

This report presents a description of and operating instructions for LISA (LARC Instruction Assembly). An assembly system for the UNIVAC, LISA uses the standard LARC mnemonic instruction code as input and produces LARC instructions and an analyzer as output. The output from LISA may also be used as input into LIS (LARC Instruction Simulator). The analyzer lists, side by side, the absolute line number, absolute LARC orders, source code as written by the programmer, and cross references to the absolute line number.

1. Digital computers – UNIVAC-LARC – Automatic coding
2. Automatic programming systems – LISA

I. Johnson, Joseph P.

David Taylor Model Basin. Report 1283.
LISA (LARC INSTRUCTION ASSEMBLY) by Joseph P. Johnson. January 1959. iv, 23p. UNCLASSIFIED

This report presents a description of and operating instructions for LISA (LARC Instruction Assembly). An assembly system for the UNIVAC, LISA uses the standard LARC mnemonic instruction code as input and produces LARC instructions and an analyzer as output. The output from LISA may also be used as input into LIS (LARC Instruction Simulator). The analyzer lists, side by side, the absolute line number, absolute LARC orders, source code as written by the programmer, and cross references to the absolute line number.

1. Digital computers – UNIVAC-LARC – Automatic coding
2. Automatic programming systems – LISA

I. Johnson, Joseph P.

David Taylor Model Basin. Report 1283.
LISA (LARC INSTRUCTION ASSEMBLY) by Joseph P. Johnson. January 1959. iv, 23p. UNCLASSIFIED

This report presents a description of and operating instructions for LISA (LARC Instruction Assembly). An assembly system for the UNIVAC, LISA uses the standard LARC mnemonic instruction code as input and produces LARC instructions and an analyzer as output. The output from LISA may also be used as input into LIS (LARC Instruction Simulator). The analyzer lists, side by side, the absolute line number, absolute LARC orders, source code as written by the programmer, and cross references to the absolute line number.

1. Digital computers – UNIVAC-LARC – Automatic coding
2. Automatic programming systems – LISA

I. Johnson, Joseph P.

David Taylor Model Basin. Report 1283.
LISA (LARC INSTRUCTION ASSEMBLY) by Joseph P. Johnson. January 1959. iv, 23p. UNCLASSIFIED

This report presents a description of and operating instructions for LISA (LARC Instruction Assembly). An assembly system for the UNIVAC, LISA uses the standard LARC mnemonic instruction code as input and produces LARC instructions and an analyzer as output. The output from LISA may also be used as input into LIS (LARC Instruction Simulator). The analyzer lists, side by side, the absolute line number, absolute LARC orders, source code as written by the programmer, and cross references to the absolute line number.

1. Digital computers – UNIVAC-LARC – Automatic coding
2. Automatic programming systems – LISA

I. Johnson, Joseph P.

David Taylor Model Basin. Report 1283.
LISA (LARC INSTRUCTION ASSEMBLY) by Joseph P. Johnson. January 1959. iv, 23p. UNCLASSIFIED

This report presents a description of and operating instructions for LISA (LARC Instruction Assembly). An assembly system for the UNIVAC, LISA uses the standard LARC mnemonic instruction code as input and produces LARC instructions and an analyzer as output. The output from LISA may also be used as input into LIS (LARC Instruction Simulator). The analyzer lists, side by side, the absolute line number, absolute LARC orders, source code as written by the programmer, and cross references to the absolute line number.

1. Digital computers – UNIVAC-LARC – Automatic coding
2. Automatic programming systems – LISA

I. Johnson, Joseph P.

David Taylor Model Basin. Report 1283.
LISA (LARC INSTRUCTION ASSEMBLY) by Joseph P. Johnson. January 1959. iv, 23p. UNCLASSIFIED

This report presents a description of and operating instructions for LISA (LARC Instruction Assembly). An assembly system for the UNIVAC, LISA uses the standard LARC mnemonic instruction code as input and produces LARC instructions and an analyzer as output. The output from LISA may also be used as input into LIS (LARC Instruction Simulator). The analyzer lists, side by side, the absolute line number, absolute LARC orders, source code as written by the programmer, and cross references to the absolute line number.

1. Digital computers – UNIVAC-LARC – Automatic coding
2. Automatic programming systems – LISA

I. Johnson, Joseph P.