36 TN

NUCLEAR ENGINEERING

MASSACHUSETTS INSTITUTE
OF TECHNOLOGY

MASTER

COLLECTION AND EVALUATION OF SALT MIXING
DATA WITH THE REAL TIME DATA ACQUISITION
SYSTEM

by

S. Glazer, C. Chiu,
N.E. Todreas

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

MN ONLY

C00-2245-36TR

NOTICE

**PORTIONS OF THIS REPORT ARE ILLEGIBLE. It
has been reproduced from the best available
copy to permit the broadest possible avail-
ability.**

DEPARTMENT OF NUCLEAR ENGINEERING
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
Cambridge, Massachusetts 02139

September 1977

NOTICE
This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the United States Department of Energy, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately owned rights.

COLLECTION AND EVALUATION OF SALT MIXING
DATA WITH THE REAL TIME DATA ACQUISITION
SYSTEM

by

S. Glazer, C. Chiu,
N.E. Todreas

ERDA Research and Development
Contract AT (11-1)-2245
U.S. Energy Research and Development Administration

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED *ED*

"This report was prepared as an account of Government-sponsored work. Neither the United States, or the Energy Research and Development Administration nor any person acting on behalf of the Commission

- A. Makes any warranty or representation, expressed or implied, with respect to the accuracy, completeness or usefulness of the information contained in this report, or that the use of any information, apparatus method, or process disclosed in this report may not infringe privately owned rights; or
- B. Assumes any liabilities with respect to the use of, or for damages resulting from the use of, any information, apparatus, method, or process disclosed in this report.

As used in the above, 'person acting on behalf of the Commission' includes any employee or contractor of the Administration or employee of such contractor, to the extent that such employee or contractor prepares, disseminates, or provides access to, any information pursuant to his employment or contract with the Administration or his employment with such contractor."

PATENT STATUS

This document, transmitted in advance of patent clearance, is made available in confidence solely for use in performance of work under contracts with the U. S. Energy Research and Development Administration (ERDA). It is not to be published nor its contents otherwise disseminated or used for purposes other than specified above before patent approval for such release or use has been secured from the cognizant assigned ERDA patent group.

Reports and Papers Published under
MIT Coolant Mixing in LMFBR Rod Bundles Project

A. Quarterly Progress Reports (Available from National Technical
Information Service, U.S. Department
of Commerce, Springfield, VA 22151)

COO-2245-1	Period June 1, 1972 - November 30, 1972
COO-2245-2	Period December 1, 1972 - February 28, 1973
COO-2245-3	Period March 1, 1973 - May 31, 1973
COO-2245-6	Period June 1, 1973 - August 31, 1973
COO-2245-7	Period September 1, 1973 - November 30, 1973
COO-2245-8	Period December 1, 1973 - February 28, 1974
COO-2245-10	Period March 1, 1974 - May 31, 1974
COO-2245-13	Period June 1, 1974 - August 31, 1974
COO-2245-14	Period September 1, 1974 - November 31, 1974
COO-2245-15	Period December 1, 1974 - February 28, 1975
COO-2245-23	Period March 1, 1975 - May 31, 1975
COO-2245-25	Period June 1, 1975 - August 31, 1975
COO-2245-26	Period September 1, 1975 - November 30, 1975
COO-2245-28	Period December 1, 1975 - February 29, 1976
COO-2245-30	Period March 1, 1976 - May 31, 1976
COO-2245-31	Period June 1, 1976 - August 31, 1976
COO-2245-34	Period September 1, 1976 - November 30, 1976
COO-2245-38	Period December 1, 1976 - February 28, 1977

Reports Issued Under This Contract

B. Topical Reports (Available from National Technical Information Service, U.S. Department of Commerce, Springfield, VA 22151)

E. Khan and N. Todreas, "A Review of Recent Analytical and Experimental Studies Applicable to LMFBR Fuel and Blanket Assembly Design," COO-2245-4TR, MIT, Sept. 1973.

E. Khan, W. Rohsenow, A. Sonin and N. Todreas, "A Simplified Approach for Predicting Temperature Distribution in Wire Wrapped Assemblies," COO-2245-5TR, MIT, Sept. 1973.

T. Eaton and N. Todreas, "Instrumentation Methods for Inter-channel Coolant Mixing Studies in Wire-Wrap Spaced Nuclear Fuel Assemblies," COO-2245-9TR, MIT, June 1974.

Y.B. Chen, K. Ip, N.E. Todreas, "Velocity Measurements in Edge Subchannels of Wire Wrapped LMFBR Fuel Assemblies," COO-2245-11TR, MIT, September 1974.

E. Khan, N. Todreas, W. Rohsenow, A.A. Sonin, "Analysis of Mixing Data Relevant to Wire-Wrapped Fuel Assembly Thermal-Hydraulic Design," COO-2245-12TR, MIT, September 1974.

E. Khan, W. Rohsenow, A. Sonin, N. Todreas, "A Porous Body Model for Predicting Temperature Distributions in Wire Wrapped Fuel and Blanket Assemblies of a LMFBR," COO-2245-16TR, MIT, March 1975.

E. Khan, W.M. Rohsenow, A. Sonin, N. Todreas, "Input Parameters to the ENERGY Code (To be used with the ENERGY Code Manual) COO-2245-17TR, MIT, May 1975.

E. Khan, W. Rohsenow, A. Sonin, N. Todreas, "Manual for ENERGY Codes I, II, III," COO-2245-18TR, MIT, May 1975.

E. Khan, W. Rohsenow, A. Sonin, N. Todreas, "Manual for ENERGY Codes I, II, III Computer Programs," COO-2245-18TR Revision 1, MIT, July 1976.

P. Carajilescov and N. Todreas, "Experimental and Analytical Study of Axial Turbulent Flows in an Interior Subchannel of a Bare Rod Bundle," COO-2245-19TR, MIT,

B. Chen and N. Todreas, "Prediction of Coolant Temperature Field in a Breeder Reactor Including Interassembly Heat Transfer," COO-2245-20TR, MIT, May 1975.

F. Carre and N. Todreas, "Development of Input Data to ENERGY Code for Analysis of Reactor Fuel Bundles," COO-2245-21TR, MIT, May 1975.

Reports Issued Under This Contract

B. Topical Reports, Continued

H. Ninokata and N.E. Todreas, "Turbulent Momentum Exchange Coefficients for Reactor Fuel Bundle Analysis," COO-2245-22TR, MIT, June 1975.

R. Anoba and N. Todreas, "Coolant Mixing in LMFBR Rod Bundles and Outlet Plenum Mixing Transients," COO-2245-24TR, August 1975.

B. Bosy, "Fabrication Details for Wire Wrapped Fuel Assembly Components," COO-2245-27TR, MIT, November 1975.

Ralph G. Bennett and Michael W. Golay, "Interferometric Investigation of Turbulently Fluctuating Temperature in an LMFBR Outlet Plenum Geometry," COO-2245-29TR, MIT, June 1976.

N.E. Todreas, "Analysis Methods for LMFBR Wire Wrapped Bundles," COO-2245-32TR, MIT, November 1976.

K.L. Basehore and N.E. Todreas, "Development of Stability Criteria and an Interassembly Conduction Model for the Thermal-Hydraulics Code SUPERENERGY," COO-2245-33TR, MIT December 1976.

Robert Masterson and Neil E. Todreas, "Analysis of the Feasibility of Implementing an Implicit Temporal Differencing Scheme in the SUPERENERGY Code," COO-2245-35TR, MIT, February 1977.

S. Glazer, C. Chiu and N. Todreas, "Collection and Evaluation of Salt Mixing Data with the Real Time Data Acquisition System," COO-2245-36TR, MIT, April 1977.

B. Mikic, E.U. Khan and N.E. Todreas "An Approximate Method for Predicting Temperature Distribution in Wire Wrapped Fuel Assemblies of a LMFBR," COO-2245-37TR, MIT, April 1977.

C. Chiu and N. Todreas, "Development of a Technique for Subchannel Flow Rate Measurements in LMFBR Wire Wrapped Assemblies," COO-2245-39TR, July 1977

C. Chiu and N. Todreas, "WARD Blanket Assembly Pre-Test Predictions by SUPERENERGY," COO-2245-40TR, July 1977

C. Chiu and N. Todreas, "Flow Split for a LMFBR 4" Wire Wrapped Blanket Assembly," COO-2245-41TR, July 1977

C. Chiu and N. Todreas, "Static Pressure and Pressure Drop for a LMFBR 4" Wire Wrapped Blanket Assembly," COO-2245-42TR, July 1977

Reports Issued Under This Contract

B. Topical Reports, Continued

C. Chiu and N. Todreas, "Mixing Experiments for a LMFBR 4" Wire Wrapped Blanket Assembly," COO-2245-43TR, July 1977

Yi Bin Chen and Michael W. Golay, "Coolant Mixing in the LMFBR Outlet Plenum," COO-2245-44TR, June 1977.

J. Kelly and N. Todreas, "Turbulent Interchange in Triangular Array Bare Rod Bundles," COO-2245-45TR, July 1977

K.L. Basehore and N.E. Todreas, "Assessment of the Need to Incorporate a Variable Swirl Flow Model into the ENERGY Code," COO-2245-46TR, July 1977.

K.L. Basehore and N. Todreas, "Analysis of the Thermal-Hydraulic Behavior in the CRBR Secondary Control Assembly, Including Interassembly Heat Transfer Effects," COO-2245-47TR, July 1977.

J.G. Bartzis and N.E. Todreas, "Hydrodynamic Behavior of a Bare Rod Bundle," COO-2245-48TR, June 1977.

Reports Issued under this Contract

C. Papers and Summaries

Yi Bin Chen, Ka-Lam Ip, Neil E. Todreas, "Velocity Measurements in Edge Channels of Wire-Wrapped LMFBR Fuel Assemblies," American Nuclear Society Transactions Vol. 19, 1974, pp. 323-324.

P. Carajilescov, N. Todreas, "Experimental and Analytical Study of Axial Turbulent Flows in an Interior Subchannel of a Bare Rod Bundle," J. of Heat Transfer, Vol. 98, No. 2, May 1976, pp. 262-268 (Included as Appendix to Quarterly Progress Report, COO-2245-15).

E. Khan, W. Rohsenow, A. Sonin, N. Todreas, "A Porous Body Model for Predicting Temperature Distribution in Wire-Wrapped Fuel Rod Assemblies," Nuclear Engineering and Design, 35 (1975) 1-12.

E. Khan, W. Rohsenow, A. Sonin, N. Todreas, "A Porous Body Model for Predicting Temperature Distribution in Wire-Wrapped Rod Assemblies Operating in Combined Forced and Free Convection," Nuclear Engineering and Design, 35 (1975) 199-211.

Ralph G. Bennett and Michael W. Golay, "Development of an Optical Method for Measurement of Temperature Fluctuation in Turbulent Flows," American Nuclear Society Transactions, Vol. 22, 1975, p. 581.

B. Chen and N. Todreas, "Prediction of the Coolant Temperature Field in a Breeder Reactor Including Interassembly Heat Transfer," Nuclear Engineering and Design 35, (1975) 423-440 (Included as Appendix to Quarterly Progress Report, COO-2245-23).

R. Bennett and M.W. Golay, "Interferometric Investigation of Turbulently Fluctuating Temperature in an LMFBR Outlet Plenum Geometry," Accepted for the ASME Winter Annual Meeting, Dec., 1976, (Included as Appendix in Quarterly Progress Report, COO-2245-30).

B.B. Mikic, E.U. Khan, N.E. Todreas, "An Approximate Method for Predicting Temperature Distribution in Wire Wrapped Fuel Assemblies of a Liquid Metal Fast Breeder Reactor," Mech. Res. Comm., Vol. 3, 353-360 (1976).

Reports Issued Under this Contract

C. Papers and Summaries (Continued)

L. Wolf, R. Karimi, I.Y. Kim, C.N. Wong, M.K. Yeung "2-D Thermoelastic Analysis of LMFBR Fuel Rod Claddings," Paper C4/d, 4th International Conf. Structural Mechanics in Reactor Technology, San Francisco, August 1977.

M. Yeung, L. Wolf, "Effective Conduction Mixing Lengths for Subchannel Analysis of Finite Hexagonal LMFBR Bundles," ANS Meeting, New York, June 1977.

C. Chiu and N. Todreas, "Flow Split Measurements In An LMFBR Radial Blanket Assembly," ANS Meeting, New York, June 1977.

ABSTRACT

A minicomputer based real time data acquisition system was designed and built to facilitate data collection during salt mixing tests in mock ups of LMFBR rod bundles. The system represents an expansion of data collection capabilities over previous equipment. It performs steady state and transient monitoring and recording of up to 512 individual electrical resistance probes. Extensive real time software was written to govern all phases of the data collection procedure, including probe definition, probe calibration, salt mixing test data acquisition and storage, and data editing. Offline software was also written to permit data examination and reduction to dimensionless salt concentration maps. Finally, the computer program SUPERENERGY was modified to permit rapid extraction of the parameters ϵ_1^* and C_1 from dimensionless salt concentration maps.

This document describes the computer system, and includes circuit diagrams of all custom built components. It also includes descriptions and listings of all software written, as well as extensive user instructions.

ACKNOWLEDGEMENTS

The authors wish to express their thanks to Professor R. Seidell, Mr. Ken Yeager, and Mr. Stan Knutson who designed and built the custom hardware for the Real Time Data Acquisition System. They also wish to thank Ms. Pei-ti Tung and Mr. Tim Gawne for assistance in the programming of the real time and off line portions of the system software, respectively. Ms. Wendy Wolfe also assisted in the development of the SUPERENERGY/DRV code.

The authors also wish to thank Mr. Alan Hanson. As principal design coordinator of the previous generation of data collection hardware, Mr. Hanson was able to make many valuable suggestions for the improvements of the system.

Thanks are also given to the US ERDA for financial support under contract number AT (11-1) -2245.

TABLE OF CONTENTS

	Page
TITLE PAGE	i
REPORTS AND PAPERS ISSUED UNDER THIS CONTRACT	iii
ABSTRACT	ix
ACKNOWLEDGEMENTS	x
TABLE OF CONTENTS	xi
LIST OF FIGURES	xiv
1.0 INTRODUCTION	1
2.0 HARDWARE DESCRIPTION	4
2.1 Overview	4
2.2 Signal Multiplexing, Software Commands and Electronic Circuit Diagrams	6
2.3 Hardware Setup and Initialization Pro- cedures	8
2.4 Troubleshooting	9
3.0 REAL TIME DATA ACQUISITION SYSTEM SOFTWARE-- PROGRAMMING ASPECTS	12
3.1 Overview	12
3.2 Disk Data File Structures	13
3.2.1 Disk Organization, RTDAS File Directory	14
3.2.2 Data Collection Parameters-SET Dataset	17
3.2.3 Probe Calibration Data - CAL Datasets	17
3.2.4 Salt Mixing Data -- MIX Datasets	18
3.3 RTDAS Subroutine Description	19
3.4 Disk Initialization Procedures	22
4.0 REAL TIME DATA ACQUISITION SYSTEM SOFTWARE-- USER INSTRUCTIONS	23
4.1 Overview	23
4.2 Software System Initialization Procedure	24

	Page
4.3 User Instructions	25
4.3.1 General Sequence of Events for Mixing Experiments	25
4.3.2 Interactive System-Logical Program Flow, User Responses, Input Formats, Consequences	31
4.3.2.1 Initial Input of Time, Date, and Bundle Geometric Data	32
4.3.2.2 Changes to Time, Date, Bundle Geometric Data	35
4.3.2.3 Setup of Data Collection Parameters--SET	36
4.3.2.4 Probe Calibration--CAL	44
4.3.2.5 Collection of Salt Mixing Data --MIX	50
4.3.2.6. Editing of Disk Files	68
4.3.2.6.1 Examination of RTDAS Disk Directory: Directory Listing, Remaining Room, Next Available File Names and Data Blocks	70
4.3.2.6.2 File Examination	71
4.3.2.6.3 Changing Data Files	72
4.3.2.6.4 Termination and Restart Procedures	73
5.0 INTERMEDIATE DATA PROCESSING SYSTEM	80
5.1 Overview	80
5.2 Interactive System-User Instructions	81
5.2.1 Hardcopy RTDAS Dataset Listings	81
5.2.2 Examination of Calibration Data: Resistance Versus Concentration Curves	82
5.2.3 Conversion of Mixing Data to Salt Concentration Maps, Salt Mass Balances, and Preparations of Data for ENERGY Code Parameter Determination	87

	Page
6.0 DETERMINATION OF ENERGY CODE PARAMETERS ϵ_1^* AND C	95
6.1 Overview	95
6.2 Code Structure and Determination of Optimum Values of ϵ_1^* and C_1	96
6.3 Input to the SUPERENERGY/DRV Code	97
6.4 Sample Problem	105
7.0 FUTURE WORK	106
FIGURES	109
REFERENCES	161
APPENDIX I - LISTING OF REAL TIME DATA ACQUISITION SYSTEM SOFTWARE	A-1
APPENDIX II - LISTING OF INTERMEDIATE DATA PROCESSING SYSTEM SOFTWARE	A-116
APPENDIX III - LISTING OF PROGRAMS TO DETERMINE ϵ_1^* AND C_1	A-140

LIST OF FIGURES

FIG. #	TITLE	PAGE
2-1(a)	Indirect Probe Resistance Measurement	110
2-1(b)	Input, Probe, and Sampled Voltage Signals	110
2-2	Operation of Blue Box Multiply/Divide Unit	111
2-3	Multiply/Divide Unit Timing Diagram	112
2-4	Multiply/Divide Unit Layout	113
2-5	Multiply/Divide Unit Control Circuitry Layout	114
2-6	Multiply/Divide Unit Control Circuitry	115
2-7	Summary of Software Commands for Multiplexor Hardware	116
2-8	Multiplexor Circuitry Timing Diagram	117
2-9	Multiplexor Support Hardware Circuitry (In Blue Box Annex)	118
2-10	Multiplexor Support Hardware Circuitry (In Blue Box Annex)	119
2-11	Multiplexor Support Hardware Circuitry (In Blue Box Annex)	120
2-12	Multiplexor Support Hardware (In Blue Box Annex)	121
2-13	Blue Box Annex Circuit Layout Diagram, Bottom View	122-124
2-14	Changes to Main Board in Blue Box Annex	125
2-15	Connectors to Main Board in Blue Box Annex	126
2-16	Signal Cross Reference List in Blue Box Annex	127-129
2-17	Multiplexor Circuitry Diagram (In White Box)	130
2-18	Multiplexor Circuitry (In White Box)	131
2-19	Multiplexor Circuitry (In White Box)	132
2-20	Multiplexor Circuitry (In White Box)	133
2-21	Multiplexor Circuitry Layout Diagram (In White Box, Bottom View)	134

FIG. #	TITLE	PAGE
2-22	White Box Channel Numbering Scheme (Top View)	136
3-1	Disk File Organization	137
3-2	Directory Information Contained in First Record of HEADER File	138
3-3	HEADER Record of a SET Dataset	139
3-4	HEADER Record of a CAL Dataset	140
3-5	HEADER Record of a MIX Dataset	141
4-1	Initial Input of Time, Date, and Bundle Geometric Data	142
4-2	Setup of Data Collection Parameters-SET	143-144
4-3	Probe Calibration-CAL	145-146
4-4	Collection of Mixing Data - MIX	147-151
4-5	Dataset Editing - EDT	152
5-1	Typical Calibration Curves for Probes #17-24	153
5-2	Calibration Curve Examination Program	154
5-3	Mixing Data Reduction Program	155
6-1	Data Calibration Procedure for ϵ_1^* and C_1 for SUPERENERGY/DRV	156
6-2	Flow Diagram of SUPERENERGY/DRV	157
6-3	Dimensionless Salt Concentration Map in the Uniform In-Phase Wire Wrap Bundle	158
6-4	Input to the Sample Problem	159
6-5	Salt Concentration Distribution Map Calibrated by SUPERENERGY/DRV	160

1.0 Introduction

The ENERGY model of fluid flow within the core region of Liquid Metal Fast Breeder Reactors (LMFBR's) proposed by E. Khan at MIT permits large scale numerical simplifications in temperature field prediction compared with other more traditional methods. Thus, for core geometries of the type encountered in LMFBR's, use of the ENERGY approach, and in particular, of the SUPERENERGY computer code to predict temperature fields yields important advantages in terms of computational time and core storage requirements. Knowledge of two empirical constants, ϵ^* and C , different for different bundle geometries, are prerequisite for use of the computer code. This document will discuss new hardware and software developed at MIT specifically for the purpose of data collection and reduction to obtain these empirical constants.

The ENERGY model developed by E. Khan is presented in ref. [1] and will not be discussed in detail here. The fundamental assumption of the ENERGY model is that the flow field within individual assemblies may be separated into 2 major regions. Coolant flow within the interior region is assumed to be purely axial, with no radial or circumferential component. Localized flow variation from axial resulting from diversion around rod spacer wires, which are spirally wrapped around each rod, is modeled as enhanced turbulent eddy diffusivity. The numerical value of this enhanced eddy diffusivity, ϵ^* , must be determined experimentally. Coolant flow in region II, immediately adjacent to the duct wall, is assumed to consist of 2 components: an

axial component, and a circumferential component. The circumferential component arises from the fact that the spirally wound spacer wire in the edge region can uniformly cause flow diversion in a preferred circumferential direction. In the interior region, flow diversion caused by the spacer wires on 3 adjacent rods prevent the radial flow from attaining a preferred direction. Immediately adjacent to the duct, however, the effect of the 3rd spacer wire is lost; hence, the presumption of flow in a preferred circumferential direction. The ratio of circumferential velocity to axial velocity in the edge region is called C_1 .

The parameters ϵ_1^* and C_1 , may be experimentally determined, for example, by performing salt mixing experiments. In this type of experiment, water flow is forced through a mock up of the precise bundle geometry under study. A tracer substance, typically salt, which changes the electrical resistivity of water is then injected at a known location within the bundle. Resistance measurements made at the outlet of the test section will then indicate the precise mixing patterns within the bundle. The analogy between heat and mass diffusion may be made, and the SUPERENERGY computer code may be used to analytically duplicate test results. The values of the constants ϵ_1^* and C_1 used in the code which most closely match test results are then taken as the correct values under the particular test conditions.

Previous hardware built for A. Hanson to record probe resistances was limited in size and speed of response. The equipment, built by Mr. Dave Gwynn, electronic technician at the MIT Nuclear Reactor, had capacity to sample 128 resistance probes. Each probe was serially scanned, and results were simultaneously printed and punched onto paper tape on a Model KSR teletype. Data recording therefore took in excess of 2 seconds per probe. Several pieces of associated equipment, such as a signal generator, resistance decade box, analog to digital voltmeter and conversion equipment, and multiplexor electronics, had to be assembled simultaneously. Equipment reliability and maintainability has at times posed difficulties.

Current ENERGY model expansion to include transients has led to the possible need for transient salt mixing experiments. Additionally, it was desired to attain the capability to perform salt mixing experiments on a newly acquired 217 pin fuel bundle, which would require 438 individual resistance type probes. The decision was therefore made in March of 1976 to modify a small minicomputer available for laboratory use, to permit extremely rapid and convenient data collection and storage for the largest bundle. This system, known as the Real Time Data Acquisition System (RTDAS), has the capability to perform steady state and transient salt mixing experiments for the 217 pin bundle, can greatly speed up the data collection, and requires fewer pieces of hardware, each of very high reliability. The data is collected under computer control, and stored on small, inexpensive, high capacity, easily transportable floppy disks. Data reduction and

permanent storage is simplified also, as a single floppy disk can store all data from an entire experiment.

Chapter 2 discusses specific hardware capabilities, initialization procedures, and presents electronic circuit diagrams. Chapter 3 discusses data structures and other aspects of the Real Time Data Acquisition System Software at the programmer level. Chapter 4 describes in detail the real time usage of the RTDAS system to acquire and store mixing data. Chapter 5 describes use of the Intermediate Data Processing System, which reduces the data from probe resistances to salt concentrations measured, and produces punched card output which forms the input to the final data reduction programs. Chapter 6 discusses the use of the programs which determine the parameters ϵ_1^* and C_1 , from salt concentration data. Chapter 7 briefly discusses directions for future system development.

2.0 Hardware Description

2.1 Overview

The hardware which makes up the Real Time Data Acquisition System includes an Interdata 716 minicomputer, a custom built white box containing multiplexors and an analog to digital converter, and a video terminal.

The minicomputer nominally has 32K words of core memory, hardware multiply and divide, a software accessible clock providing a resolution of .25 μ set, and a time of day counter. Core memory may be expanded to an additional 32K words if desired. The computer also contains 2 floppy disk drives, permitting 1 disk to hold system programs, and the other to accumulate data.

The custom built white box permits multiplexing the high level signals (in the range of -10 to +10 volts) from up to 512 resistor type salt conductivity probes, conversion of the analog signals via a 12 bit A to D convertor to digital form, and transfer of the data directly to the computer. Sensitivity of the A to D converter has been increased for use with the particular probes currently used. It has been modified so that voltages in the range 0-+5V occupy full scale readings. This appears optimum at this time, as the largest probe measurements occupy slightly more than 1/2 scale. The RTDAS system software has been set up to store the data directly on one of the floppy disks. The frequency of the square wave generator used to drive the probes may be set under program control. Practical limits on the oscillator frequency are approximately 500 to 80000 KHz. The number of channels sampled and the sequence of sampling is determined by means of lists stored in the computer memory. The data is returned in a user specified buffer.

The video terminal is a Lear Seigler ADM 3A. It can display 24 lines of alphanumeric text and data, consisting of up to 80 characters per line. It was originally purchased by the Mixing Project in February, 1977 to replace the old, fault prone, and extremely slow (100 baud) teletype used with Alan Hanson's equipment. It serves the function of high speed teletype replacement without hardcopy facility and graphics capabilities. The I/O transfer rate (baud) has been set at the maximum rate the computer can drive; 9600 baud. Changes to the I/O transfer rate will require a minor change to the system software. An additional I/O port exists at the rear of the terminal, to

permit interfacing with a high speed hardcopy device, such as a line printer.

All hardware modifications to the Interdata Computer, and the design and checkout of the additional white box were done by Ken Yeager, electronic engineer, Rm. 54-1811. Stanley Knutson, engineer, wrote the assembly language subroutines which initiate the resistance measurements and data transfer, set the square wave generator frequency, and access the clock. Originally having assisted in writing the existing operating system for the Interdata Computer, Mr. Knutson provided much valuable assistance during the writing of the Real Time Data Acquisition System Software.

2.2 Signal Multiplexing, Software Commands and Electronic Circuit Diagrams

The probes used in the mixing experiments to measure salt concentration are of the electrical resistance type. They record local changes in electrical conductivity of the water between the 2 probe leads due to changes in salt concentration. The passive probes are driven by the square wave signal generator, which produces a signal of +10 volts, through a reference resistor, as shown in Fig. 2.1a. Probe resistances are indirectly measured by the voltage read at the analog to digital converter, as shown in the figure. From the circuit shown, the probe resistance may be determined as:

$$V_{\text{measured}} = \frac{R_{\text{probe}}}{R_{\text{probe}} + R_{\text{ref}}} \cdot 10 \text{ volts} \quad (2-1)$$

Note that determination of probe resistance is not required as long as the calibration procedure, in which measured voltages may be equated with known salt concentrations, is carried out.

Fig. 2-1 (b) indicates the sampling procedure. At the beginning of the square wave cycle, measured voltage rises as shown. At the conclusion of $1/2$ the square wave period, the voltage signal is fed through a sample and hold circuit which maintains the voltage constant for the 8μ sec required for the analog to digital conversion. Note that the square wave period need not be extended to the point where the measured voltage reaches a steady state value. The probe calibration must, however, be performed at the same frequency as the actual mixing data is collected to preserve consistency.

To accommodate the "white box" containing the multiplexors, ADC, and associated hardware, additions and changes were required to the Interdata Computer known as the "Blue Box". In addition to interfacing equipment, a hardware multiply divide unit was added. Fig. 2-2 through 2-6, prepared by Ken Yeager, indicate the operations performed during multiplies and divides, and include timing diagrams, circuit diagrams and layout drawings for the multiply/divide unit in the Annex section of the Interdata Computer. Fig. 2-7 through 2-16 indicate programming commands for the multiplexor unit, timing diagrams, circuit diagrams, and layout drawings for the hardware supporting the multiplexor function, in the annex section of the computer. Also included are layout diagrams and signal cross reference lists applicable to both the multiply/divide unit and the multi-

plexor support hardware. Figures 2-17 through 2-21 are circuit diagrams and layout diagrams for the "white box", containing the multiplexors and ADC unit. Additional information on the hardware, including complete circuit diagrams for the Interdata Computer, may be obtained from Ken Yeager.

2.3 Hardware Setup and Initialization Procedures

Interconnections between the Interdata Computer, the Video terminal, and the multiplexors (White Box) is described in this section. The instructions listed need not be performed in the indicated sequence.

1. Plug in the computer, white box, and terminal to 110 V AC
2. Connect the 34 pin cable, permanently attached at the rear of the computer, to the white box, as indicated in Fig. 2-22. The cable should be oriented such that the edge of the cable with a red stripe is at the end of the connector in the white box with a corresponding strip of red tape.
3. Connect a 25 pin connector cable from the computer to the video terminal. The cable should be plugged into the receptacle marked "video terminal" at the computer rear panel. The other end must be connected to the receptacle marked "MODEM" on the bottom rear of the video terminal.
4. Plug the probes into the proper multiplexor positions in the white box. All probes should be connected in groups of 8 to 16 pin male connectors. A top view of the white box indicating channel numbers and corresponding multiplexor locations appears in Fig. 2-22. Care should be exercised, when connecting probes through extension cables to the white box, that the orientations of the male 16 pin plugs into the female sockets is consistent. Channel numbering within a group of 8 probes begins at the end of the connector with one corner cut off, and increments sequentially. This convention is shown in Fig. 2-22.

Before placing probes in the bundle, the user must ascertain the channel numbers of each probe within a group of 8. This may be accomplished, for example, by initiating the Real Time Data Acquisition System, defining a setup block to sample probes 1-8, and starting the Calibration subroutine. Then, groups of 8 probes may be plugged into the white box. Probes are then sampled with only one of the 8 probes in water, the others dry in air. The probe in the water will have a much lower value of resistance than those in air. Since probe resistances (voltages) will be printed out in the order of which the sampling took place (i.e. 1,2,3,4,5,6,7,8, if so defined in the SETUP block), it is possible to identify the number of the probe within the group of 8. The multiplexor marked "GROUP 1" on Fig. 2-22 will sample channels 1-8, that marked "GROUP 2" will sample channels 9-16, etc. For simplicity, it is suggested that probes be placed in the bundle corresponding to the subchannel numbering sequence defined in the computer code "SUPERENERGY" (see ref. 2).

5. Turn on the computer with the white rocker switch on the cross bar at the middle rear of the computer.
6. Turn on the video terminal with the white rocker switch at the bottom rear of the terminal.
7. Turn on the white box with the switch on the front panel. Note: The white box should be turned off when removing or replacing probes, resistors, multiplexors, etc.
8. Initiate the Real Time Data Acquisition System by following the procedure outlined in Section 4.2.

2.4 Troubleshooting

Most real time hardware errors should be corrected only by Ken Yeager or other electronic engineers familiar with the hardware. This section will describe 3 types of hardware errors the user can diagnose and correct.

Very rarely hardware errors may occur transferring data to or from the floppy disks. This may occur during the system initialization procedure. This type of error is indicated by

the message "ERR II". If the system will not respond to further commands from the terminal, then the restart procedure described in section 4.3.2.7 should be performed. It is important to remember that if this occurs, no information already stored on the floppy disk is affected.

Sampled probes with indicated resistances of 4095 are indications of very large resistances, and may be open circuits. This may be caused by faulty probes and/or bad connections. Individual channels displaying resistances of 4095 are probably failed probes or dry probes sitting in air. Entire groups of 8 probes with readings of 4095 are indicative of an improper connection and may result from plugs having pulled loose somewhere between the probes and the white box. Individual probes which fail during the course of the experiment cannot be replaced unless the entire calibration procedure is repeated.

Sampled probes with indicated resistances of 0 are indications of short circuits. This may be caused either by shorted probes, failed multiplexor units, or the white box not being turned on. Individual values of 0 resistance are most likely caused by shorted probe wires, or a failed multiplexor. Multiplexor failure is considered possible but unlikely, and may be checked by removing and replacing the affected multiplexor and observing the new resistance value. If all probes sampled indicate 0 resistance, the White Box is either turned off or the 34 wire cable between the computer and the white box is not properly connected. A resistance reading of 0 for several probes together or in groups of 8 are likely caused by failed

multiplexors. If this is the case, and if the failed multiplexors would normally monitor resistance in subchannels considered likely to amass large salt concentrations, then these "important" probes may be switched with another set of 8 unimportant probes at the white box. Then, the important probes will still be sampled, but not in the sequence originally specified in the SET block. For this reason, the SET subprogram must be entered, and the definition of sequential channel sampling should be made such that the switch probes are still sampled and stored in the same order as before, corresponding to the definition of sequential samplings appearing in the SET block on the floppy disk. For example, if the multiplexor monitoring channels (probes) #17-24 were faulty, and if probes 17-24 were considered important, while probes #9-16 were not considered important, then these 2 plugs may be switched at the white box; i.e. with the white box power turned off, probes 9-16 may be plugged into channel numbers 17-24 in the white box, while probes 17-24 are plugged into channel #'s 9-16. If the SET block initially defined the sampling sequence to be channel numbers 1-126, then after the switch, the probe sampling sequence would be 1-8, 17-24, 9-16, 25-126. All that remains is to use the SET subprogram to redefine the sequence of channel sampling as follows: 1-8, 17-24, 9-16, 25-126. When this is done, the data is collected and stored will once again refer to concentration measurements made in subchannels 1-126 sequentially. Note that no changes to the SET block existing on the disk should be made. This procedure should only be used if Ken Yeager or Stanley Knutson are not present.

Sufficient numbers of spare multiplexors were ordered, and are being stored by Ken Yeager, so that faulty multiplexors may be immediately replaced.

3.0 Real Time Data Acquisition System Software - Programming Aspects

3.1 Overview

The Real Time Data Acquisition System (known as the RTDAS) is a package of interactive user programs designed to run on the Interdata 716 laboratory computer, to facilitate the taking of data during salt mixing tests. The principal code author is Ms. Pei-ti Tung, currently an undergraduate in the Electrical Engineering & Computer Science Department at MIT. Technical programming assistance was provided by Mr. Stanley Knutson, engineer.

The package of programs was designed to function as a highly specialized operating System. Any of the major subprograms performing specialized tasks may be recalled from the disk and entered whenever the system is in the main program, or monitor, mode.

System operation and user responses are described in Chapter 4. This section provides descriptions of the data formats and file structures, and brief descriptions of all the subroutines and subprograms normally invisible to the user. Information contained in this section is intended primarily for users modifying the system programs, or writing additional software to expand the capability of the system. Program

listings appear in Appendix I.

3.2 Disk Data File Structures

At the current time, tasks performed during the course of the mixing experiment may be grouped into three major categories: 1) definition of probes to sampled, oscillator frequency, and setting up of the sampling sequence, 2) probe calibration, and 3) taking of actual mixing data. Major subprograms have been written to facilitate each set of tasks; SET, CAL, and MIX. Each procedure creates data which must be stored on the floppy disk for use during data collection and later during the intermediate data reduction (see Chapter 5). Sections 3.2.1-3.2.4 describe the precise format of individual types of files, and the location and meanings of all stored data.

The dataset naming convention is as follows: datasets created during the setup, calibration, or mixing tasks are prefaced by a three character mnemonic SET, CAL, or MIX, respectively. Datasets considered logically connected (for example, the datasets comprising a complete set of probe resistance versus concentration data), are assigned the same series number. Individual datasets within the series are assigned unique sequence numbers. Each dataset created during setup procedures is presumed not to be logically related to any other dataset; hence, no sequence numbers are assigned to descriptive Setup data, only series numbers. For calibration and mixing types of data, series numbers are automatically sequentially assigned by the RTD based on the last series number existing on the disk each

time the CAL and MIX subprograms are entered. Individual dataset sequence numbers are also automatically sequentially allocated by the system. For the CAL and MIX datasets, up to 99 series may be defined on the disk, with each series containing from 1 to 99 sets of data. It should not be necessary to enter setup information more than once, or at most twice during the course of an experiment. Hence, only series 1 and 2 may be defined for SET data.

Dataset names are stored in ASCII format, 1 character per 8 bit byte, in each record in the HEADER file. The specific format of the name of CAL and MIX datasets is:

XXXaa.bb , where XXX=CAL if calibration data
 =MIX if mixing data
 aa=series number,
 01< aa< 99
 bb=sequence number,
 01< bb< 99

The specific format of the SET datasets is:

SETcc , where cc=series number,
 01< cc <02

3.2.1 Disk Organization and RTDAS File Directory

Each floppy disk contains a total of 1024 segments of 256-8 bit bytes, or 128 (half) words. To be consistent with the regular operating system of the Interdata Computer in the Mechanical Engineering-Civil Engineering Joint Computer Facility (JCF), the disk contains a standard type of file directory. As it would have been very awkward and inefficient to use this

directory to store information about each of the data files taken with the RTDAS, the disk directory is defined to be very short, and serves to reserve large blocks of the disk for the system data. Its presence is necessary to permit the regular operating system to access the disk. Overall disk file structure is indicated in figure 3-1. Each of the major sections (i.e.; TEXT, HEADER, AND DATA), are defined and permanently reserved in the directory.

Also reserved on the disk is a 4 segment record, always occupying segments #6-10, which contains a text description of the experiment. Data in this record is stored as alphanumeric ASCII characters, 1 character per byte. This text is written and examined through the RTDAS editing subprogram (see Section 4.3.2.6.4).

Following the TEXT file is the HEADER file which always occupies physical segments #8-166. The HEADER file is subdivided into 840 logical records of 24 (half) words, or 48 bytes. The first record always contains information necessary for proper accessing of the datasets by the RTDAS. Fig. 3-2 indicates the types and locations of the pointers stored in the 1st record of the HEADER file. The 1st word is the record number of the next available (unused) record in the HEADER file. Sufficient room has been reserved for up to 840 records. The second word is the record number of the next available (unused) record in the data section. Words 3, 4, and 5 are respectively, the next available series numbers of SET, CAL, and MIX datasets.

Each of the 3 procedures; SET, CAL, and MIX, result in the definition of 2 types of information; header information, and primary data. Each logical dataset therefore consists of 2 parts; one record in the HEADER file, and from 1 to 4 records (of 128 half words) of data in the DATA file. The record in the HEADER file contains descriptive information about the data appearing in the DATA file. For example, the record of header information for a SET dataset (defined during the setup procedure) contains the number of probes to sampled, oscillator frequency, date, etc. The records in the data file contain a list of the channel numbers to be sampled and the order of sampling. Header blocks for CAL datasets contain information such as the salt concentration. The data section contains probe resistances. The header segment for a MIX file contains data such as bundle flowrate, Reynolds number, injection flowrate, etc. The data section contains the probe resistances. Each header segment points to the record number of the start of the data. Contents of the header sections of SET, CAL, and MIX datasets are more completely defined in sections 3.2.2-3.2.4.

The DATA file is subdivided into 840 logical records of 128 (half) words. It always occupies physical segments #167 to 1024 on the disk. The number of data records required for each dataset depends on the number of probes sampled. A 61 pin bundle has 126 subchannels, and therefore requires only 1 record for data storage. A 217 pin bundle has 438 subchannels, so 4 records are needed to store all the data. The number of records of data for each dataset is also stored in the corresponding header record,

3.2.2 Data Collection Parameters-SET Dataset

Information necessary for further data collection is defined in the setup procedure. Contents of the header record of a SET dataset is shown in Fig. 3-3. The format of the (half) words 1-8 are common to all 3 types of header blocks. Primary data which is required for subsequent data processing and intermediate data reduction are number of probes to be sampled, and the square wave period. The data type, probe type, and sampling duration were included for possible future system development, and are not currently used by the RTDAS or the intermediate data processing programs.

Half word #6 contains the record number of the start of the data in the DATA file. Although no actual probe resistance measurements are taken during the setup procedure, the data section contains the list of channel numbers to be sequentially sampled. This list is used in real time during data collection. The intermediate data processing programs access this information when making up resistance versus salt concentration curves, and when determining radial salt concentration distributions.

3.2.3 Probe Calibration Data - CAL Datasets

The type and formats of data contained in the header record associated with calibration data is shown in Fig. 3-4. Note that a pointer to the appropriate SET dataset, which defines oscillator frequency, number of probes, and sampling sequence, appears in half word #10. Water temperature (in °C), time of day, and date are included as a permanent record. The

known salt concentration which corresponds to the current resistance data is stored in units of (grams salt/lbm water) x 1000. By convention, this concentration is always measured relative to the current background concentration of the canal water, which is assigned a value of 0 concentration. This is important, as even clean canal water (without added salt), is relatively dirty compared with tap water, and resistance measurements made in canal water are always lower than those made in tap water. Probe resistance measurements made under the indicated conditions are stored in the data segments in the sequence specified in the SET dataset.

3.2.4 Salt Mixing Data - MIX Datasets

Figure 3-5 shows the types and locations of information contained in the header record associated with a set of actual mixing data. As may be seen from the figure, sufficient room exists in the header record for storage of all descriptive parameters required for data collection, reduction, and verification by a salt mass balance.

As before, data collection and sequence information is contained in the SET dataset specified in word #10. Salt concentration versus probe resistance curves required for data conversion to concentration form are made up from individual datasets with the CAL major series # identified in word 11. Corrections to the concentration versus resistance curves are made using information in the Calibration Correction Datasets, specified in word #24. Only 1 correction dataset (with the sequence #01) is assumed to exist within the major series

number. Individual probe resistances recorded in the correction block (which is always taken at background concentration levels), are subtracted from the corresponding probe resistance at 0 concentration, as measured in the regular calibration series, and is used to shift the entire resistance versus concentration curve.

Other parameters, such as bundle flowrate, concentration of injection solution, and flowrate of injection solution, are required for determination of the salt mass balance. Data such as axial height of injection and injection subchannel are used as inputs to the computer programs which determine the mixing parameters ϵ_1^* and C_1 for the ENERGY series of computer codes. Time, specified as seconds and milliseconds referenced from the current minute, are stored in anticipation of future system expansion to permit rapid data collection and storage during flow transients. All other data is retained for permanent bookkeeping purposes.

Probe resistance data is stored in the indicated segments of the data file.

3.3 RTDAS Subroutine Description

This section includes a brief description of all subroutines in alphabetical order existing in the RTDAS software package. Since most of these subroutines are normally invisible to the user, the descriptions are included merely for reference purposes. Additional comments appear within the body of the code, and may be seen in the listing in Appendix I.

CAL	Controls collection of parameters and probe sampling during calibration function
CALRE	Computes Reynolds Number (in terms of bundle cross-sectional area, equivalent diameter, bundle flow-rate, etc.)
CHANGE	Uses namelist option to provide user a chance to change initial information (date, time, area, bundle DE) under the CHA monitor mode
CHAC	Permits user to change information stored in CAL header block by use of namelist option
CHAM	Permits user to change information stored in MIX header block using namelist option
CHAS	Same as CHAC, CHAM; but for setup header block
COPY	Displays 1) NP, number of probes; 2) IDURA, sample duration; 3) Period, which are the pertinent setup information for the current CALIBRATION/MIXING series
DATAR	Writes the set of data currently collected (probe numbers for SETUP, probe resistances for CALIBRATION & MIXING) and kept in the core into the DATA FILE on the floppy disk, and updates the DATA FILE pointer
EDIT	Subprogram EDIT is the request processor (request: L,R,N...) under command EDT
EDTCHA	Permits the users to change information on the block specified under "C" (change) request in "EDT" command
GETNUM	Prints out a prompting message at the terminal and collects a numeric parameter value from the keyboard. This is the main subroutine used throughout the RTDAS to accept user input.
GVNAME	Gives the series number for the current block, e.g. SET01 (for a SETUP record) CAL11.01 (for a CALIBRATION record) MIX20.01 (for a MIXING record)
HEADER	Writes the current header information onto the records in Header File on the floppy disk. (always 24 (half) words per record). Updates the HEADER file pointer.

INAME	Accepts the SETUP, CALIBRATION, or CALIBRATION CORRECTION series number for the current CALIBRATION or MIXING block from the terminal.
INDATE	Collects date from the terminal (month, day, year).
INTEXT	Collects text to be stored in the TEXT FILE, and writes it on the floppy disk
LIST	Lists all the block names in the HEADER FILE, in sequential order.
MIX	Controls collection of parameters and probe sampling during MIXING experiment.
OUTDATE	Unpacks the data information stored as the 13th (half) word in each HEADER block into Month, Day, Year
PICKER	Collects a series of numbers (either integer or floating point) from the terminal, max of 10 numbers per line
PRINT	Allows user to input a block name, once "P" or "C" (print or change) is entered as the request under "EDT" command; and then prints (if "P") or lets user change information on (if "C") the block, as specified
PRINTC	Prints the 15th-24th (half) words on a CALIBRATION record specified by the user under "P" (print) request in "EDT" command
PRINTD	Prints one or more records of data corresponding to the header record specified under "P" request in "EDT" command at the terminal
PRINTH	Prints the contents of the first 14 (half) words common for every header record, at the console
PRINTM	Prints the 15th-24th (half) words of a MIXING header block
PRINTS	Prints the 15th-24th (half) words on a SETUP header block
PROBER	Collects probe resistances
RANGE	Checks if all the probe samplings fall within the range specified by the user as higher and lower bound

SAVER Write the 5 updated pointers onto the 1st record in HEADER file on the floppy disk
The 5 (half) words are:

- (2) HEADER FILE pointer, points to the next available record in HEADER file
- (2) DATA FILE pointer
- (3) Next SETUP block series number
- (4) Next CALIBRATION block series number
- (5) Next MIXING block series number

SET Controls collection of parameters and the sequence of channel sampling during SETUp procedure

SEARCH Searches through the HEADER FILE for a block name specified in array "NAME"

TEXT Prints out contents of the TEXT file

These subprograms are stored in 2 Libraries on the system disks in JCF.

- One library is called GLAS-LIB and contains the following subprograms:

CAL	CALRE	CHANGE	CHECK	COPY	DATAR
GETNUM	GUNAME	HEADER	INAME	INDATE	MIX
PICKER	PROBER	RANGE	SAVER	SET	

- One library, called TUNG-LIB, contains the following subprograms:

CHAC	CHAM	CHAS	EDIT	EDTCHA
INTEXT	LIST	OUTDATE	PRINT	PRINTC
PRINTD	PRINTH	PRINTM	SEARCH	TEXT

3.4 Disk Initialization Procedures

Prior to use of a disk for data collection, parameters defining file types and storage allocation must be initially input into the disk. A special program entitled INIT has been written by S. Knutson for this purpose. The program is kept in the drawer marked MIXING in the JCF program storage cabinets.

Specifically, the INIT program reserves the maximum space for each of the files TEXT, HEADER, and DATA, as defined in section 3.2.1 and in Fig. 3-1. Also stored on the disk are the directory information pointers used by the RTDAS, as shown in Fig. 3-2. All pointers, with the exception of the next available header record, are set to 1, indicating an unused disk. The number of the next available HEADER record is set to 2, as the 1st record is always reserved for directory information pointers. The listing of this program is included in Appendix I.

New disks may be purchased from Ed Newman, room 1-125. Currently, floppy disk numbers 4121-4125 have been purchased for exclusive use of the MIXING project (disk cartridge numbers are written on each disk jacket in the upper left corner, and are written onto the disk directory itself by programs which format the units). Data disks are color coded with blue tabs, and the system disk is color coded with a red tab.

To initialize a floppy disk, insert the disk into one of the disk drives in the JCF Interdata computer. Then, run the INIT program on either the M-70 or M-80 operating systems. At the appropriate console, the message "CARTRIDGE NUMBER" will be printed out. The user must then input the number of the floppy disk which has been loaded into the drive to be initiated (e.g. 4124). The program will automatically terminate when done.

4.0 Real Time Data Acquisition System Software User Instructions

4.1 Overview

The real time data acquisition system software permits

the user to interact with the computer to facilitate the taking and storing of salt mixing data. Specific software has been written to permit probe calibration, taking of actual mixing data in the bundle, storage of data on disk units, examination, and changing of data. This section describes the operation of the software written primarily by Ms. Peiti Tung, intended to be run on the Interdata 716 computer in the laboratory, during the course of an actual mixing experiment.

4.2 Software System Initialization Procedure

Once the computer, multiplexor box ("White Box"), and video terminal have been connected and turned on in preparation for an experiment as described in section 2.3, the system and data disks must be mounted in the disk drives. The system disk, #4116, containing both the operating system and the Real Time Data Acquisition System (RTDAS) software, which has been color coded with a red label, must be carefully loaded into disk drive B in the front panel of the Interdata Computer. The data disk, color coded green, must be similarly loaded into disk drive A. The data disk must at some time have been properly initialized, as discussed in Section 3.4. The disk may or may not contain previously stored data. Then, the following sequence of events must be enacted to begin execution of the software system:

1. Push the INI button on the front panel of the Interdata Computer

2. Push up the IPL switch on the disk drive section of the computer
3. Push the DTA button on the computer
4. Push the ADR button on the computer
5. Push the RUN button on the computer

At this point, the computer will print the following message on the terminal:

JOB HEAT TRANSFER LAB

DUP MAINLINE

D:

To begin execution of the software system, type:

```
// XEQ GLSINIT
```

Program execution will be signalled by the initial request for the current date (see Section 4.3.2.1)

4.3 User Instructions

4.3.1 General Sequence of Events for Mixing Experiments

Initial program development was begun in July 1976, and continued through February, 1977. Design of the real time software system was based originally on the description of the conduct of mixing experiments by Alan Hanson. During the course of system development, the mixing test procedure was modified by Chong Chu. Changes introduced by Mr. Chiu, and subsequent recommendations as to desirable features in a real time data acquisition system, were incorporated into specific portions of the code.

This section will assume a general knowledge as to the purpose behind, and general conduct of, a mixing experiment. A separate

report will be written in the near future to document in detail the purpose, preparations required for, and general experimental procedures for mixing experiments. This section will present the sequence of subprogram and subroutine usage during a typical salt mixing experiment.

After program initialization and input of the date, time, and bundle geometric parameters (discussed in section 4.3.2.1), the program will wait in a monitor mode. A ">" symbol signifies the monitor mode. At this time, any of the major subroutines written to facilitate specific experiment data taking needs, may be recalled from the system disk and executed.

The first program which should be executed in the course of a mixing experiment is the SETUP program, called by typing SET while in the monitor mode. The SETUP program permits the definition of the probes to be sampled, the oscillator frequency to be used, and other principally invariant data during the entire experiment. Such definitions are mandatory before probe calibration and mixing data can be taken and stored. Further instructions are given in section 4.3.2.3.

The next subprogram which should be run is the calibration subprogram. This program is executed during the entire course of the probe calibration phase of the experiment. Upon user commands, probe resistances with known salt concentration values are sampled, presented for display, and stored for permanent record on the disk. Additional descriptions of user inputs are discussed in section 4.3.2.4. After the entire set of calibration data has been taken, it is suggested to leave the system

THIS PAGE IS BLANK

temporarily in the calibration subprogram, and remove the data disk for immediate examination of the resistance versus concentration curves. These may be examined by taking the data disk to the ME-CE Joint Computer Facility, RML-11, and running the CAL program which is part of the intermediate data processing system, described in section 5.3.1. If probe resistance appears consistently inaccurate for many probes at a particular concentration, that particular calibration set may be retaken. Note that since it is not possible to delete, or alter file names, the new calibration set must be taken as an additional set of the initial calibration series; hence the reason for not leaving the calibration subprogram in the Real Time System. Effective deletion of the erroneous calibration block is accomplished at any time later by using the editing program to change the value of salt concentration to be larger than 10000. The intermediate data processing routines have been set up to ignore calibration blocks with concentrations larger than 10000, when making up resistance versus concentration curves. See Section 5.2.2 for further instructions as to the operation of the Intermediate Data Processing System's CALIB program. That section makes additional suggestions as to methods of retaking and correcting erroneous calibration data.

Upon completion of the calibration phase of the experiment, and subsequent return to the monitor, the experimenter should alternate between the Mixing subprogram and the Calibration subprogram. Although not originally planned during code development, Mr. Chiu has found that probe resistances will

vary slightly over time due to postulated instantaneous changes in water temperature, salt concentrations, and probe wire movements, and oxidation film on the platinized probe pins. To account for this, it is now the practice to take a reading of probe resistances with only background salinity in the bundle, i.e. with the salt injection disabled, between each set of mixing data. This background reading is used in the intermediate data reduction procedure as corrections to the initial calibration data. Localized correction data taken in this way have been found to substantially reduce errors and permit salt mass balances close to unity, i.e. the salt measured by the probes versus the known quantity of salt injected.

It is anticipated that the automated data acquisition system will greatly speed up the data taking process over the older equipment used by Alan Hanson. Also, based on Hanson's experimental techniques, the subprograms which allows the taking of actual mixing data has been set up to especially facilitate the taking of data for one value of Reynolds number, from one injection rod, throughout a set of user defined axial injection heights. For these reasons, it is suggested that the experimental procedure be to first take a calibration correction block by reentering the CAL subprogram and taking one set of data at 0 (background) concentration. Then, the mixing subroutine, initiated by typing MIX when in monitor mode, should be executed and used to take actual mixing data at 1 value of Reynolds number and with 1 injection rod, but at several axial heights. The mixing program may then be terminated, and the calibration

program reentered to take additional correction data in preparation for the next set of mixing data. The procedure may be continued until the experiment is concluded. As discussed in section 4.3.2.5, the mixing program gives great flexibility in the taking of data. The above procedure may be modified, for instance, by taking one set of correction data for every set of mixing data. If desired, no correction data need be taken at all. The difference between the calibration correction data and the initial calibration series is used to effectively bias the complete resistance versus concentration curves on a probe by probe basis. If, in the MIX program, the name of the calibration correction data block is the same as the initial calibration block taken at zero (background) salt concentration, no biasing of the calibration curves will take place during the intermediate data processing. It should be noted that the name of an existing calibration correction block should be entered at the appropriate point in the mixing program, as the data disk is searched to verify the validity of the block name.

At any time during the experiment, descriptive text may be written onto the data disk in the manner described in section 4.3.2.6.4. File examination, listing, and changes may be accomplished by using the Editing program, as discussed in section 4.3.2.6. Changes to basic bundle geometric data and the time and date, effective for all subsequent stored data, may be done by using the change subroutine, discussed in section 4.3.2.2. Any of the currently available 5 major sub-programs (SET, CAL, MIX, EDT, CHA) may be executed at any time

during the course of the experiment when the software system is in the Monitor mode.

4.3.2 Interactive System - Logical Program Flow, User Responses, Input Formats, Consequences

This section will describe the messages output by the computer in each of the various subprograms. Most messages are output to prompt the user for certain information required either by the real time system, or in subsequent data processing. For all such messages, the message meaning, input required, and input format will be precisely defined in this section. The method of presentation of this data consists of two parts: 1) a complete flowchart indicating the sequences and wording of all messages output by the computer during the course of each subprogram execution, and 2) word descriptions of the meaning, input, input format, and consequences of the input for all messages requiring user interaction. Small numbers appearing beside the message boxes in the flowcharts identify individual messages for later discussion in sections 4.3.2.1 to 4.3.2.7.

It should be recalled from Chapter 3 that each of the 5 major subprograms, i.e., SET, CAL, MIX, EDT, and CHA may call several subroutines each, and may consist of several sections not in core at any one time. The flowcharts presented in this section are intended to provide the user an overview of the logical operation of each major subprogram. They will not identify the individual functions performed by the various smaller subroutines in the course of the subprogram execution,

as the operation of these smaller subroutines are invisible at the user level. Brief descriptions of the operations performed by these subroutines are given in the programmers data section, section 3.3.

All subprograms have been written to permit maximum flexibility in experimental data collection. Although not shown in most cases, the ability to quit the subprogram and return to the monitor mode exists at nearly all points where user input is requested. This may be done by typing Q when the alphabetic input is expected, or -1 when numeric input is expected.

4.3.2.1 Initial Input of Time, Date and Bundle Geometric Data

Immediately after operating system initialization, described in section 4.2, the system will automatically request certain data which need only be input once during the course of the experiment. Fig. 4.1 indicates the sequence of system messages output after system initialization. Following, under individual message number, is a description of messages as listed in Fig. 4.1.

Message #: 1

Input Required: Current Date

Input Format: 3 integers, separated by commas. The date, input in the form: a,b,c, has the following restrictions: $1 \leq a \leq 12$, $1 \leq b \leq 31$, $76 \leq c \leq 100$

Sample Input: 2, 11, 77

Comments: The date is required for record keeping purposes only. The date each set of data is recorded is stored in the header record corresponding to the

data section. The date is automatically updated by the computer at midnight according to an internal clock.

Message #: 2

Input Required: Current time

Input Format: 2 integers, separated by commas. The time, input, in the form: a,b, has the following restrictions: $0 \leq a \leq 24$, $0 \leq b \leq 60$, where a is the hour, and b= the minute

Sample Input: 9, 30

Comments: The time is required for bookkeeping purposes only during steady state data collection. The time of day each set of data is taken is stored in the header block corresponding to the data block. The time is updated continuously during the experiment according to an internal clock. The storage of second and millisecond is only important during reduction of transient mixing data. This data is also currently stored.

Message #: 3

Input Required: Bundle equivalent (hydraulic) diameter, in inches

Input Format: 1 real (floating point or integer) number. The restrictions on the input is: $0.000 < DE \leq 10.000$

Comments: The bundle equivalent diameter is required by the real time system to permit calculation of instant-

aneous Reynolds numbers in the bundle. As the magnitude of the largest Reynolds number which may be calculated and stored on the data disk is limited to 65,535, without computer error, it is important to enter the correct value of equivalent diameter. The Reynolds number is stored on the disk in the header section of each MIX dataset, and is later used during intermediate and final data processing procedures.

Message #: 4

Input Required: The cross sectional area of the bundle, in square inches

Input Format: 1 real (floating point or integer) number. The restrictions on the input is $0.000 < \text{area} \leq 32.000$

Comments: The area is required in real time for the purpose of calculation of instantaneous Reynolds numbers in the bundle. For reasons discussed in the comments section for message 3, the correct value of bundle area should be input. The area is not stored on the data disk.

Message #: 5

Input Required: A number identifying the particular bundle being used

Input Format: 1 integer. The restrictions are: $0 < \text{ID} \leq +32767$

Comments: Although stored with the data on the disk, the bundle identification number is required for bookkeeping

purposes only, and is not currently used in any phase of the data collection or processing.

After the bundle identification number is input, the system enters the monitor mode, signified by the symbol ">".

4.3.2.2 Changes to Time, Date, Bundle Geometric Data

Changes to the date, time and bundle equivalent diameter, area and identification number may be accomplished with the CHANGE subprogram. This program may be entered by typing CHA when the system is in the monitor mode. When started, the values of aforementioned parameters are printed. Following is a list of variable names and their meanings:

<u>variable name</u>	<u>meaning</u>
NEWDAT	date
NEWTIM	time
AREA	bundle area
DE	bundle equivalent diameter
ID	bundle identification number

To effect a change, type the name of the variable to be changed followed by an "=", and followed by the new value the variable is to be changed to. Several changes may be made at once by separating the inputted changes by commas. Completion of the changes is signalled by typing an "&". To change either the time or date, which are printed out here as 0, enter NEWTIM=1, or NEWDAT=1, respectively. The time and/or date will then be separately requested, and input format will be as described in

messages #1 and 2 of section 4.3.2.1. This is necessary due to the bit packing performed for these two variables.

4.3.2.3 Setup of Data Collection Parameters - SET

Definition of the parameters necessary to sample the probes is done in the SETUP subprogram. The subprogram is executed by typing SET when the system is in the monitor mode. Among data specified in this subprogram are oscillator frequency, the number of probes to be sampled, the channel or probe numbers, and the sequence of sampling. This data must be entered into the computer each time it is turned on, and is mandatory before any other data can be taken and stored on the disk.

Fig. 4.2 shows the sequence of system messages output after execution of the Setup subprogram. As before, description of messages listed in Fig. 4.2 are described by message number.

Message #: 1

Input Required: None

Input Format: NA

Sample Input: NA

Comments: The operating system was originally designed to permit continuous data storage on several disk units. The implication is that the Setup blocks, defined at the outset of the experiment, may not be stored on any but the 1st disk. It is therefore necessary to store the SETUP information in core. Limitations on core storage require that no more than 2 sets of Setup data, including a list of the

probes to be sampled, be kept in core at once (Revisions to the data collection procedures indicate one disk to be sufficient for an entire set of mixing data for a 61 pin bundle).

The number of Setup blocks existing on the data disk, printed as part of message #1, is determined from a search of the disk. If this number is 2, then no new setup information may be specified and stored. However, the name of one of the setup blocks already existing on the disk may be used to reinform the computer of the data in the old block on the disk.

Message #: 2

Input Required: Whether a new or existing setup block is to be specified

Input Format: One of the following letters: N (new), O (old), Q (quit and return to monitor)

Sample Input: N

Comments: See discussion in the comments section of message #1. If the number of SET blocks existing on the data disk is 0 or 1, then a new setup block may be input. If N is specified, then the information taken during the current execution of the SET block will not only be stored in core memory, but

will be stored on the disk as well. If 0 is specified, then the information gathered will only be stored in core memory. At least one disk of any group taken during a complete experiment must have at least 1 setup block stored on it for later data reduction purposes. Therefore, at the start of a new experiment with an unused data disk, N should be specified so that the setup information will be stored in the disk. Neither the real time system nor the subsequent data processing programs have to date been tested with multiple data disks.

Data in setup blocks already on the disk will not be overwritten if 0 is specified, and the name of the setup block is the same as an existing one. The editing subprogram must be used for this purpose. Specification of either N or 0 will, however, overwrite data currently kept in core in setup blocks with the same name as previously specified. Care should be taken that setup information written into core only with the 0 option be the same as data currently existing on the disk in the setup block with the same name. Further, changes made to a setup block in core will only affect data taken subsequently. Data taken and existing on the disk, i.e., in calibration and mixing data blocks, must be changed with the editing program.

Message #: 3

Input Required: None

Input Format: NA

Sample Input: NA

Comments: If a new Setup block is to be entered, as defined in message #2, then the system will automatically inform the user of the next sequentially available Setup series number. This will be the number of the new SETUP block. If 2 SETUP blocks already exist on the disk, the user is told, and is not permitted to enter another one.

Message #: 4

Input Required: The number of the Setup block

Input Format: An integer number. The restriction on the number is $1 \leq \text{number of SET block} \leq 99$

Sample Input: 1 (this specifies the name SET01)

Comments: If an old Setup block was specified in message #2, then the Setup block herein defined must currently exist on the data disk.

Message #: 5

Input Required: Number of probes to be sampled

Input Format: Integer. NP is the variable name given to the number of of probes. The restrictions on NP are: $1 \leq NP \leq 512$

Comments: This parameter determines the number of 128 word records allocated on the disk for the storage of probe resistance data. A 61 pin bundle has 126

probes, which fits entirely within a single 128 word record. A 217 pin bundle, however, has 438 probes, which requires 4 128 word records in the data file. Allocation of the proper number of records for all subsequent recording of data is automatically accomplished by the system.

Message #: 6

Input Required: Oscillator frequency, in Hertz

Input Format: An integer number. Restrictions on the frequency is: $1 \leq \text{frequency} \leq 32767$

Sample Input: 10000

Comments: The oscillator is set under program control by this parameter. The oscillator emits a square wave at the specified frequency. Currently, the frequency should not be set below 500 Hz, nor above approximately 50000. All calibration and mixing data referring to the same Setup block will be taken at the same oscillator frequency. Care must be taken that mixing data must be recorded at the same frequency as corresponding calibration data to insure correct conversion of probe resistance data to salt concentrations. Frequencies of either 1000 Hz or 10000 Hz have been successfully used to gather data, and either of these values is recommended for steady state data recording.

Although frequency is input, the assembly language subroutines written by Stanley Knutson which sets the oscillator, was set up to use the period in units of .5 microseconds. This period is subsequently stored in the header record of a Setup dataset.

Message #: 7

Input Required: Sample Duration

Input Format: An integer number. IDURA is the variable name given to the duration. The restrictions are: $0 < \text{IDURA} \leq 32767$

Sample Input: 1

Comments: This parameter is currently unused in steady state mixing data. It was reserved for use in future transient mixing subprograms.

Message #: 8

Input Required: Decision as to changes desired in any data input up to this point

Input Format: One of the following three letters: Y (yes, make changes); N (no changes, all data input so far is correct), Q (quit, return to monitor mode).

Sample Input: Y

Comments: This is the only chance to change either the oscillator frequency, number of probes, or sample duration in the SET subprogram

Message#: 9

Input Required: Changes to either oscillator frequency, number of probes, or sample duration

Input Format: The name of the variable to be changed, followed by an "=", followed by the new data. Several changes may be separated by commas. Completion of all changes should be signified by a "&".

Sample Input: NP = 124, PERIOD = 200&

Comments: The opportunity to make changes is done through the NAMELIST option. The variables NP, PERIOD, and IDURA are printed out before changes are made.

Message #: 10

Input Required: Channel (Probe) numbers to be sampled, in sequence

Input Format: Either integer or floating point numbers, separated by commas. Floating point numbers will be truncated to the appropriate integer values. A maximum of 10 numbers per line may be input. Restrictions on input: $1 \leq \text{probe (channel number)} \leq 512$.

An alternate input format has been devised which permits several sequential probe or channel numbers to be input once. The input format in this case is 3 integers, the first being the first probe number of the series, the second number zero, and the third integer is the last

number of the series. The two input formats may be mixed as desired until all probe numbers have been defined.

Sample Input: 1,0, 126 - results in probes 1 through 126 being sequentially sampled. Additional sample inputs may be :

1, 2, 3, 126 or

126, 125, ..., 1 or

1, 3, 2,...89, 90, 0, 126

Comments: The input formats have been designed to permit maximum flexibility and ease of probe definition. Probes will be sampled in the order specified in the input list. Additionally, entering -2 during the probe collection sequence prints out the probe numbers entered so far, and -1 causes termination of the Setup procedure and a return to the monitor,

Message #: 11

Input Required: None

Input Format: NA

Sample Input: NA

Comments: Upon completion of probe (Channel) definition, the probe numbers input are printed out. On figure 4.2, output is shown for input of the type: C, 0, D. On the figure, A and B would be 1 and 2, signifying the 1st and 2nd numbers input, and C and D would be the 1st and last probe numbers in the sequence.

Message #: 12

Input Required: Decision to change any previously input
channel (probe) numbers

Input Format: One of the following letters: C (for change),
K (for kill), and O (for OK)

Sample Input: C

Comments: C should be typed when only minor changes to the
probe numbers are desired. K permits the user to
start over and redefine the entire set of probe
numbers.

Message #:13

Input Required: Changes to the probe number definitions

Input Format: The name of the particular array element(s) to
be changed, followed by "=", followed by the
corrected probe number. The last change should
be concluded with an "&"

Sample Input: IBUFB(28) = 72, IBUFB (32) = 148

Comments: The namelist option is used to effect changes.
During the change procedure, before "&" is entered,
all variables may be examined by typing "?". This
should be done sparingly when arrays are to be
changed, as most arrays are 512 elements long.

4.3.2.4 Probe Calibration - CAL

One subprogram is used during the entire probe calibration
procedure. This program facilitates the sampling of the probes
previously defined in a Setup sequence with several values of

known salt concentration. Information such as concentration, time, date and water temperature are stored in the header record associated with each set of probe resistances. Probe resistances may be sampled a number of times and averaged to reduce random errors.

The calibration subprogram may be initiated by typing CAL when the system is in the monitor mode. Termination and return to the monitor mode may be accomplished. points user input is requested, by typing either Q or -1. The sequence of system messages printed during execution of the Calibration subprogram is shown in Fig. 4.3. Nearly all messages will be discussed by reference to message number on Fig. 4.3.

Message #: 1

Input Required: None

Input Format: NA

Sample Input: NA

Comments: Each time the calibration subprogram is entered, the system reserves a new series name. For example, if series CAL01.-- and CAL02.-- already exist on the data disk, the system will automatically reserve the next sequential series number for the current calibration data. This message merely informs the user of the number of the current calibration series, which will be used to store data as long as the calibration subprogram is run.

Message #: 2

Input Required: The number of the SET block which defines pertinent setup information for this calibration series.

Input Format: An integer number. Due to the method of naming data blocks, the restriction on the input is: $1 \leq \text{number} \leq 99$.

Sample Input: 2

Comments: The Setup block must have been previously defined and must exist in core.

Message #: 3

Input Required: None

Input Format: NA

Sample Input: NA

Comments: The pertinent data from the Setup block specified is reviewed for the user

Message #: 4

Input Required: Water temperature, in degrees C.

Input Fromat: Integer or floating point. The restrictions are $0.0 \leq \text{temperature} \leq 37.8$

Sample Input: 28.3

Comments: The water temperature is not used in subsequent data processing, and is stored in the header record for record keeping purposes only.

Message #: 5

Input Required: The number of times to sample the probes to produce an averaged reading.

Input Format: Integer. Restrictions: $1 \leq \text{NAME} \leq 7$

Sample Input: 7

Comments: The extreme speed of the computer system makes multiple probe samplings feasible to reduce random noise.

Message #: 6

Input Required: Concentration of solution, in units of milligrams of salt per lbm of water

Input Format: Integer. The restrictions are: $1 \leq \text{ICCON} \leq 32767$

Sample Input: 10

Comments: This data is used in the intermediate data reduction procedure to develop calibration versus resistance curves for each probe and to calculate salt mass balances. The data is not used by the real time system, but is stored in the header record associated with each set of probe resistance data.

Message #: 7

Input Required: Decision as to desire to change water temperature, solution concentration, or the name of the Setup block prior to storage on the disk.

Input Format: One of the following letters: C (for change), N (for name change), G (for go), or Q (for quit)

Sample Input: G

Comments: C permits changes to the water temperature and salt concentration, N permits changing the Setup block, G

signals go which indicates all data is correct and then causes the probes to be sampled, and Q causes a return to monitor.

Message #: 8

Input Required: New values of water temperature and/or
solution concentration

Input FORMAT The name of the variable followed by an "=" followed by the new value, and concluded with an "&".
TEMPIC is the water temperature, and ICCON is the concentration.

Sample Input: ICCON=50&

Comments: Current Values of the parameters are first printed.
Then the namelist option is used to effect changes.

Message #: 9

Input Required: The number of the new Setup block which will
define data taking parameters.

Input Format: An integer. Due to the method of numbering of the Setup blocks, the restriction on the number is: $1 \leq \text{number} \leq 99$

Sample Input: 2

Comments: Changing the Setup block affects subsequent data only. The number of the old Setup block is first printed. The new Setup block must have been previously defined and exist in core.

Message #: 10

Input Required: None

Input Format: NA

Sample Input: NA

Comments: The period, sample duration, and number of probes
sampled from the new Setup block are printed

Message #: 11

Input Required: Decision to print averaged resistance data
just taken.

Input Format: One of the following letters: Y(for yes), N
(for No), and Q (for quit)

Sample Input: Y

Comments: None

Message #: 12

Input Required: None

Input Format: NA

Sample Input: NA

Comments: If Y is selected in message #11, the probe data
is printed out for user review

Message #: 13

Input Required: Decision to store the data just taken or not

Input Format: One of the following letters: Y(for yes), N (for
No)

Sample Input: Y

Comments: The system automatically updates the sequence number
if the data is to be stored. The data should only

be stored if the user is satisfied as to its correctness. This option is important, as a "No" specified at this point will cause a return to message #6. This loop provides a quick and easy method for rapid sampling and examination of probe resistances.

Message #: 14

Input Required: Decision to take additional calibration data

Input Format: One of the following letters: Y (yes) or N (no)

Sample Input: Y

Comments: None

4.3.2.5 Collection of Salt Mixing Data - MIX

One major subprogram controls the actual collection of mixing data. Currently, a subprogram only permitting probe data collection during steady state mixing tests has been written. An additional mainline program may be written in the future to control the recording of probe data during flow transients. This may be necessary should examination of ENERGY code parameters ϵ^* and C_1 during flow transients be desired.

To initiate the subprogram which takes mixing data, type "MIX" when the system is in the monitor mode. The subprogram will then prompt the user for such data as the names of the Setup, Calibration, and Calibration correction blocks, bundle flowrate, injection flowrate, etc. A flowchart of the MIX subprogram, indicating the sequence of computer system messages, is shown in Fig. 4.4. All parameters necessary for conversion of the

raw data to salt concentration maps, and determination of a salt mass balance (total salt measured by the probes divided by total salt injected) are automatically stored in the header record associated with each set of mixing data.

Subprogram termination and subsequent return to the monitor mode may be affected by typing either Q or -1 at nearly all points where user input is required. It should be noted also from Fig. 4.4 that a large number of parameters are requested each time the program is entered. Repetitive input of these variables may be eliminated by typing a carriage return at the following message numbers, as shown on Fig. 4.4: 7,8,9,10,11,12 and 14. Once entered during the course of the experiment, inputs requested at these points, plus the axial levels of salt injection (input at setp 15), remain in core. The MIX program may be terminated and reentered without reinputting these variables. Explanation of system messages and description of user responses during the MIX subprogram is now given. System message numbers refer to numbers beside the various blocks in Fig. 4.4

Message #: 1

Input Required: None

Input Format: NA

Sample Input: NA

Comments: Each time the MIX subprogram is entered, successive mixing data is assumed to belong to a new series.

The series number, representing the two digits to

the left of the decimal point is automatically determined by the system. It will always be the next sequential series number not used on the disk. Series numbers may be in range of 01 to 99.

Message #: 2

Input Required: The number of the Setup block defining the probes and oscillator frequency

Input Format: An integer number in the range $1 \leq \text{no.} \leq 99$.

Sample Input: 1

Comments: This Setup block must have been previously defined and exist in core.

Message #: 3

Input Required: The number of the calibration series containing sufficient data to construct resistance versus concentration curves

Input Format: An integer number, in the range: $1 \leq \text{no.} \leq 99$

Sample Input: 12

Comments: This calibration series must currently exist on the disk.

Message #: 4

Input Required: The series number of the calibration correction block recently taken with background concentration only.

Input Format: An integer number, in the range $1 \leq \text{no.} \leq 99$.

Sample Input: 34

Comments: Data from the correction block is used only during the intermediate data processing and reduction of probe resistances to salt concentration (see Section 5.3.3). The calibration correction block must previously have been defined, and must exist on the disk.

Message #: 5

Input Required: None

Input Format: NA

Sample Input: NA

Comments: Pertinent data from the Setup block specified by the user as input to message #2 is printed out to remind the user. The variable meanings are:
PERIOD = oscillator period (in units of $.5\mu$ sec)
IDURA = sample duration (not currently used)
NP = number of probes to be sampled

Message #: 6

Input Required: Number of times to sample each probe to obtain average reading

Input Format: An integer, in the range $1 \leq \text{no} \leq 7$

Sample Input: 5

Comments: Averaging several probe readings will help to eliminate random errors. It is recommended that 7 readings be taken.

Message #: 7

Input Required: The largest value of probe resistance expected during the experiment

Input Format: An integer number, in the range of $0 \leq \text{upper bound} \leq 32767$

Sample Input: 3000

Comments: See comments for message #8

Message #: 8

Input Required: The smallest value of probe resistance expected during the experiment

Input Format: An integer number, in the range of $0 \leq \text{lower bound} \leq 32767$

Sample Input: 100

Comments: All measured probe resistances will lie in the range of $0 \leq \text{resistance} \leq 4095$ because the output from the A-D converter is 12 bits in length. Sensitivity of the hardware has been adjusted so that the largest resistance of any probe (in clean water) is approximately 2000. The smallest values of probe resistance, taken in very high salt concentration solutions, seems to be approximately 400. Indicated readings less than 400 may result from short circuited probes or failed multiplexers. Values above 2000 (typically 4095) are indicative of open circuits, or exceedingly high resistance. To eliminate the need to examine all probe resistances each time a set of mixing data is taken, the computer system has been set up to

automatically scan all readings and report any probes with resistances out of the allowable range. If this type of error checking is not desired, or if consistent reporting of individual failed probes is annoying, the feature may be circumvented by setting the upper bound larger than 4096, and setting the lower bound at 0. Determination of appropriate upper and lower bounds for any probes under local conditions may be accomplished by examining resistance readings at 0 and extremely high values of salt concentrations, respectively.

Message #: 9

Input Required: Water temperature, in degrees Centigrade

Input Format: An integer or floating point number, in the range
of: $0 \leq \text{temperature} \leq 37.8$

Sample Input: 30.2

Comments: Water temperature is stored in the header record associated with the mixing data for bookkeeping purposes only.

Message #: 10

Input Required: Salt concentration of injection solution, in
units of milligrams of salt per lbm water

Input Format: Floating point or integer, in the range of $0 \leq$
concentration ≤ 32767 . The data is stored on the
disk as an integer

Sample Input: 150

Comments: This data is required during intermediate data reduction to determine salt mass balances.

Message #: 11

Input Required: Salt solution injection flowrate, in gpm x 100000

Input Format: A floating point or integer number, in the range of $0 \leq \text{inj. flowrate} \leq 32767$. The data is stored in floating point format on the disk.

Sample Input: 1562.2

Comments: This information is required during intermediate data processing to determine salt mass balances.

Message #: 12

Input Required: Bundle flowrate, in gpm

Input Format: Integer or floating point number, in the range of: $0 \leq \text{flowrate} \leq 300$. The number is stored in floating point format on the disk.

Sample Input: 139.73

Comments: Bundle flowrate is stored for bookkeeping purposes only. Also, see Message #13.

Message #: 13

Input Required: None

Input Format: NA

Sample Input: NA

Comments: The code uses the values of bundle cross-sectional area, equivalent diameter, and bundle flowrate to calculate the Reynolds number. The water temperature

input at step #9 is used to interpolate in a prestored table of water viscosity versus temperature in 10°F intervals to determine current viscosity. The Reynolds number is printed out at this step as a floating point quantity, but is stored on the disk as integer. If $Re < 0$ or $Re > 65535$, an error message is printed, and the value of the Reynolds number stored on the disk is in error.

Message #: 14

Input Required: Subchannel number (using the probe numbering scheme) where salt solution is injected

Input Format: An integer or floating point number, in the range $1 \leq \text{subchannel} \leq 512$. An integer is stored on the disk.

Sample Input: 6

Comments: This information is stored for bookkeeping purposes only.

Message #: 15

Input Required: Decision to enter a new set of axial injection levels.

Input Format: One of the letters: Y, N, or Q

Sample Input: Y

Comments: This statement gives the user the opportunity to enter for the first time, or change, the number of or values of axial injection positions which will

be stored in the header blocks associated with subsequent mixing data. The program is set up to permit the user to take many sets of mixing data, varying only the axial injection position. Data taken in this fashion shows axial mixing patterns. Individual values of axial levels, input as a group in steps 15-18, are stored sequentially with each new set of mixing data. This permits the user to take several sets of mixing data very rapidly. Y indicates a new set of axial levels will be defined, N indicates no new set will be input, and Q causes a return to the monitor mode.

Message #: 16

Input Required: The number of sets of mixing data to be taken with only axial injection position varied.

Input Format: An integer or floating point number, in the range: $1 \leq \text{no. of levels} \leq 99$. The data is converted to integer for further use.

Sample Input: 4

Comments: Current experimental procedure calls for taking 1 set of mixing data, followed by taking 1 set of calibration correction data. It is possible to define 1 axial injection position at a time by inputting 1 at this step.

Message #: 17

Input Required: Axial injection positions in inches

Input Format: One, or a series of integer or floating point numbers, in the range of $0.000 \leq \text{heights} \leq 60.00$

Sample Input: 12.34, 14.85, 32.73, 4.25

Comments: The method of inputting is exactly as discussed in section 4.3.2.3, under message #9. The data is stored in integer format on the disk, in units of inches x 100; i.e. a height input as 12.34 will be stored as 1234. This data is required for bookkeeping purposes only.

Message #: 18

Input Required: None

Input Format: NA

Sample Input: NA

Comments: The axial injection positions just input in step 17 are printed out to the user for verification.

Message #: 19

Input Required: Decision to change either the number of or values of axial injection positions just input.

Input Format: Any of the following letters: K,C,O

Sample Input: 0

Comments: K causes the computer to kill the set of axial levels input, and permits the user to reenter the entire series again. C indicates a change to one or several of the axial levels, and is intended to

be used when the number of changes is small. 0, for OK, indicates the list of axial levels was entered correctly.

Message #: 20

Input Required: Changes to the existing list of axial injection positions.

Input Format: Axial levels are stored sequentially in the order in which they were input, in the array BUFB. The NAMELIST option is used to effect changes. To change the *i*th axial level in the series, type BUFB(*i*) = followed by the new value. Several values may be changed by repeating the above procedure, with individual elements separated with commas. An & signifies completion.

Sample Input: BUFB(2)=23.52, BUFB(1)=7.293&.

Comments: None

Message #: 21

Input Required: Decision to change the number of axial injection levels defined, or the levels themselves

Input Format: One of the letters: A, C, O, Q.

Sample Input: 0

Comments: 0, for OK, signifies all axial injection data is correct, and continues. C indicates changes are desired to the values of the axial levels. A orders the system to start again at step 15 (see

fig. 4.4), and Q causes a return to the monitor mode.

Message #: 22

Input Required: Changes to the axial injection levels previously input.

Input Format: As described in message #20, except the array LEVELS holds the axial levels

Sample Input: LEVELS(3) = 52.188

Comments: See the description of message #20.

Message #: 23

Input Required: Decision to change any information input so far except the axial level data.

Input Format: One of the letters: C, N, O, Q

Sample Input: 0

Comments: "C" permits the user to change the water temperature salt concentration, flowrate, injection sub-channel, bundle flowrate, upper and lower probe bounds for valid data, and the number of times to sample the probes for averaged data. "N" permits the user to change the names of the Setup, Calibration, and Calibration Correction series blocks. "O", for OK, indicates all data is correct. Q causes the mixing subprogram to quit and return to the monitor mode.

Message #: 24

Input Required: Changes to previously input parameters

Input Format: Type in the variable name, followed by an "=", followed by the new value. This procedure may be repeated, with changes separated by commas. An "&" signifies an end to the changes.

Sample Input: SLFLOW=457.3, ISUB=12&

Comments: Current parameter values are printed out prior to permitting the user to make changes. Variable names and their meanings are given below.

<u>Variable Name</u>	<u>Parameter</u>
TEMPIM	water temperature
IMCON	injection salt solution concentration
BFLOW	bundle flowrate
SLFLOW	salt solution injection flowrate
ISUB	injection subchannel
NAVG	# probe samplings/ averaged reading
IPROH	Maximum valid probe resistance
IPROL	Minimum valid probe resistance

Message #: 25

Input Required: None

Input Format: NA

Sample Input: NA

Comments: The numbers of the Setup, Calibration, and Calibration Correction blocks are printed out for user review

Message #: 26

Input Required: The number of the SETUP block to be used in subsequent data collection.

Input Format: An integer number in the range: $1 \leq \text{Setup block number} \leq 99$

Sample Input: 2

Comments: The new Setup block must have been previously defined and exist in core. If it is not desired to change the number of the SETUP block, type a carriage return.

Message #: 27

Input Required: The new Calibration Series number

Input Format: An integer, in the range $1 \leq \text{no.} \leq 99$

Sample Input: 2

Comments: The new Calibration series must exist on the disk. If it is not desired to change the number of the Calibration block number, enter a carriage return.

Message #: 28

Input Required: The new Calibration Correction series number

Input Format: An integer, in the range $1 \leq \text{no} \leq 99$

Sample Input: 19

Comments: The new calibration correction block number must exist on the disk. If it is not desired to change the number of the Calibration correction block, type carriage return.

Message #: 29

Input Required: None

Input Format: NA

Sample Input: NA

Comments: See description of message # 5

Message #: 30

Input Required: Decision to use the teletype (video-terminal)
or a portable button box for further input

Input Format: One of the following letters: T,B, or Q

Sample Input: T

Comments: The original intent was to use a portable, hand held button box to signal the computer to take successive sets of data. This would permit a single experimenter to physically alter the axial injection location and then take data without dismounting from the ladder at the bundle. It has since been recognized that additional user input, in the forms of decisions to examine, change, and store the data, is necessary. The utility of the manual button box in this application is lost. Therefore, the teletype should always be specified at this point.

Message #: 31

Input Required: None

Input Format: NA

Sample Input: NA

Comments: At this time, assembly language routines to read the manual button box have not been written. Hence, if B is input at step 30, the computer indicates the button box is not available. The teletype is used as the default input device.

Message #: 32

Input Required: Authorization to take the mixing data

Input Format: One of the following letters: G, or Q

Sample Input: G

Comments: G(GO) input at this time signals the computer to sample the probes and average the results from N readings, where N was input at step 6. Q (QUIT) causes return to the monitor mode.

Message #: 33

Input Required: None

Input Format: NA

Sample Input: NA

Comments: No message is printed out here. This simply indicates an important logical step.

Message #: 34

Input Required: None

Input Format: NA

Sample INPUT: NA

Comments: For each of the N times the probes are sampled, probe resistances are checked to verify that they fall between the specified lower and upper limit.

The numbers of any probes exceeding these limits
are printed out at this point.

Message #: 35

Input Required: Decision to print averaged probe resistances
just taken.

Input Format: One of the following letters: Y or N

Sample Input: Y

Comments: The high speed video terminal makes it feasible
to examine all probe resistances. It is suggested
that "Y" always be specified at this point so
the user may verify reasonable concentration patterns
have been established

Message #: 36

Input Required: None

Input Format: NA

Sample Input: NA

Comments: Probe (channel) numbers and measured resistances
are printed out

Message #: 37

Input Required: Decision to change any of the mixing data
just taken prior to storage on the disk

Input Format: One of the following letters: Y(yes) or N (no)

Sample Input: N

Comments: Probe resistances may also be changed at any
later point in time through the change option
in the editing subprogram

Message #: 38

Input Required: Changes to any probe resistances just measured

Input Format: Changes are made in the same manner as described in message #20, except the array containing the probe resistances is named IBUFB .

Sample Input: IBUFB(24)=1328, IBUFB(9) = 7968&

Comments: None

Message #: 39

Input Required: Decision to store or retake the mixing data.

Input Format: One of the following letters: Y(yes), N(no)

Sample Input: Y

Comments: Y causes the data to be stored on the disk. The subprogram automatically keeps track of and increments the Mixing set number, so that each set of mixing data is uniquely numbered. If the decision is made not to store the data, the set number is not incremented, and as seen on fig. 4.4, the program attempts to retake the mixing data.

Message #: 40

Input Required: None

Input Format: NA

Sample Input: NA

Comments: No message is printed at this point. The intent is to indicate that the program will remain in a tight loop permitting rapid collection of all

mixing data previously specified. The program will continue in this loop until one set of mixing data has been stored for each axial level input at step 17.

Message #: 41

Input Required: Decision to continue taking mixing data

Input Format: One of the following letters: M (MIX) or
Q (QUIT)

Sample Input: Q

Comments: If more mixing data is to be taken, the program loops back to point 2 (see fig. 4.4) to permit new information to be input. At any point where information is requested from the user, if the information is not to be changed, then the return key alone will signify this. If Q is input at step 41, the system returns to the monitor mode.

4.3.2.6 Editing of Disk Files

Dataset descriptive information, stored in the header block, and probe resistance measurements stored on the disk are utilized directly to produce calibration curves, salt mass balances, and planar salt concentration maps. Slightly inaccurate measurements taken during the calibration procedure may adversely affect all subsequent mixing data. Faulty probes may yield resistance measurements grossly inconsistent with the rest of

the probes. Incorrect dataset descriptive information, entered by the user during the course of the experiment, may cause completely incorrect interpretation of the associated data. It is therefore necessary to be able to examine and alter all data and descriptive headerblock information stored on the disk. An editing subprogram has been written as part of the Real Time Data Acquisition System which facilitates dataset listing (by name), examination and alteration of all stored information, examination of file directory information, and examination and input of descriptive text. The editing subprogram is initiated by entering EDT when the system is in the monitor mode. As before, return to the monitor mode may be accomplished at nearly every point user input is requested, by typing either -1 or Q. A flow chart indicating the sequence of system messages and user interaction required is shown in fig. 4.5. This section will discuss user responses necessary during operation of the editing program.

Message #: 1

Input Required: Decision of type of editing to be done.

Input Format: One of the following letters: L (for List), R (for Remaining Space), N (for Next available block names), P (for Printing of file contents), C (for Changing of file contents), I (for Inputting descriptive text), T (for descriptive text printout), and Q (for Quit and return to monitor mode.)

Sample Input: L

Comments: The ? = symbol shown in message block #1 of fig.

4.5 indicates the editing subprogram is waiting to accept further user commands. After processing any of the above commands (except Q), the editing subprogram returns to this point.

4.3.2.6.1 Examination of RTDAS Disk Directory: Directory

Listing, Remaining Room, Next Available File Names
and Data Blocks

Knowledge of the contents of the disk may be desirable to check on the number and types of data currently stored, and to determine the amount of additional data which can be stored. To accomplish these tasks, 3 commands have been set up within the editor, and are discussed in the description of messages 2-4.

Message #2:

Input Required: None

Input Format: NA

Sample Input: NA

Comments: Entering an L at message point 1 in fig. 4.5 causes the subprogram to access the disk directory to determine the number of and types of the datasets stored on the disk. The subprogram then lists sequentially by name (i.e., CAL01.01, MIX02.01, etc.) all files stored.

Message #: 3

Input Required: None

Input Format: NA

Sample Input: NA

Comments: Entering an R at message point 1 causes the subprogram to examine the disk directory and print out the number of unused header blocks and data blocks remaining. The first header block is always used to store directory data and is never available. The number of data blocks of length 128 words used for each dataset depends on the number of probes read. For example, a 217 pin bundle with 438 subchannels requires 4 datablocks and only 1 header block, per dataset..

Message #: 4

Input Required: None

Input Format: NA

Sample Input: NA

Comments: Entering an N at message point 1 causes the subprogram to search the disk directory and print out the next available series numbers for SET, CAL, and MIX blocks. Also printed out are the locations, or block numbers of the next available header and data blocks.

4.3.2.6.2 File Examination

The procedures to examine any data or header information stored is explained in descriptions of messages 5-7. File data is printed by typing P at message point #1 on Fig. 4.5.

Message #: 5

Input Required: The name of the block to be examined

Input Format: The full name of one block exactly as it appears in the directory listing

Sample Input: MIX25.04

Comments: The block named must exist on the data disk.

Message #: 6

Input Required: Decision to examine the header information, information stored in the data section, or both.

Input Format: One of the following letters: H(for header record printout only), D (for data record printout only,) or B (for both)

Sample Input: B

Comments: This step was initially inserted when use of a standard teletype as system-user I/O device was assumed, so that much time would not be wasted printing all file data if only a portion is required. However, it is useful with the new and very fast video data terminal as well. The terminal produces no hardcopy output, and displays only the last 24 lines on the screen at once. Printing of data-blocks with large amounts of data may cause pushing

of the first points off the screen. Also, entering B at this point will almost always cause partial pushing off the screen of the header information, which always appears first.

Message #: 7a,7b,7c

Input Required: None

Input Format: NA

Sample Input: NA

Comments: Information stored in the header and/or data sections of a dataset are printed, depending on the response to question 6. All pertinent information in each of the 3 types of datasets (SET, CAL, and MIX) are printed. Information stored in the data section are printed in the sequence in which it was taken. Examination of the data section in a Setup block shows the sequence of the probes sampled.

4.3.2.6.3 Changing Data Files

The procedure to change any data contained in the header-block or datablock sections of a file is discussed in the description of messages 8-10. Changes are accomplished by typing C at message point #1.

Message #: 8

Input Required: The full name of the file to be changed

Input Format: The full name of the file exactly as it appears in the directory listing

Sample Input: CAL11.04

Comments: The block named must exist on the data disk.

Message #: 9

Input Required: Decision to change data in the headerblock
section, datablock section, or both

Input Format: One of the following letters: H(for headerblock
only), D (for data only), or B (for both)

Sample Input: B

Comments: None

Message #: 10a,10b,10c

Input Required: Changes to the information stored in the
dataset

Input Format: The namelist option was used to permit changes
to the data in the header and data sections.
When changing information in the header block
section, the variable names to which all pertinent
header block data has been assigned are first
printed out. The user may examine the current
values of the data by entering "?". The data
may be changed by typing the name of the variable,
followed by an "=", followed by the new value.
Several pieces of data may be changed by repeating
the above procedure, separating each sequence
with a comma. Completion of changes is indicated
by entering "&".

Information stored in the data section of a file is, for the purpose of changing, temporarily stored in the array IBUFB, which has 512 elements. Data appears in the IBUFB array in the same sequence it was taken. Array contents may be examined by first typing 'a "?'. The data may be changed by typing the element of array, followed by an "=", followed by the new value. Several pieces of data may be changed by repeating the above procedure, separating each sequence with a comma. Completion of changes is indicated by entering "&".

Sample Input: Changing data in header section: JDAY = 11,
TEMPIM = 12.88&

Changing data in data section:

IBUFB(12) = 1810, IBUFB(37) = 16, IBUFB(4) = 1798&

Comments: Different data appears in the header section of the Setup, Calibration, and Mixing files. Variable names used when changing header data and their meanings are listed below for each of the 3 types of blocks:

SET header records:

<u>Variable Name</u>	<u>Meaning</u>
JMON	month
JDAY	day
JYR	year
JHR	hour

THIS PAGE IS BLANK

<u>Variable Name</u>	<u>Meaning</u>
JMIN	minute
ID	bundle ID#
NP	number of probes

CAL header records:

<u>Variable Name</u>	<u>Meaning</u>
IPSET	# of Setup block that defines sequence of probe sampling
IPTCAL	# of Calibration series
ICRCRAL	# of calibration correction block
JMON	month
JDAY	day
JYR	year
JHR	hour
JMIN	minute
JSEC	second
JMSEC	millisecond
TEMPIM	water temperature
BFLOW	bundle flowrate
RE	bundle Reynolds number
IMCON	Concentration of injection solution (gms salt/lbm water) x 1000
SLFLOW	Injection solution flowrate (gpm x 100000)
ISUB	Injection subchannel
AXIAL	Axial height of injection

4.3.2.6.4 Examination and Changing of Descriptive Text

Up to 8 lines of alphanumeric descriptive text may be stored on the disk. The intent was to provide a basic description of experimenters, test goals, hardware used, and notes as to irregularities, as a permanent record. Text examination and changing procedures are discussed in the descriptions of messages 11 and 12.

Message #: 11

Input Required: Text

Input Format: Up to 8 lines of alphanumeric text, no more than 64 characters/line. The text may be ended by either entering 8 full lines, or entering a period as the first character on a new line.

Sample Input: 1:

THIS EXPERIMENT WAS PERFORMED ON

2:

FEBRUARY 12-14 BY C. CHIU AND

3:

S. GLAZER, WITH THE 61 PIN BLANKET BUNDLE

4:

,

Comments: The number of each line is printed out before the line is entered by the user. Text input is initiated by typing I as a response to message #1

Message #: 12

Input Required: None

Input Format: NA

Sample Input: NA

Comments: Text examination is initiated by typing T as a response to message #1. The text is printed out exactly as entered.

4.3.2.7 Termination and Restart Procedures

To conclude an experiment, the proper procedure is to type END when in the monitor mode. The system will acknowledge this, list all the files stored on the data disk, and type:

DUP MAINLINE

D:

At this point, the terminal and the computer may be shut off and disconnected. It should be noted that removal of either the system disk or the data disk does not require system termination and computer shutdown.

If the experiment is to be continued on another day all startup procedures described in Section 4.2 are applicable. In this case, the information in the Setup block must be re-entered, as discussed in section 4.3.2.3. If the RTDAS system crashes, which may occur as a result of hardware reading or writing errors, the system may be restarted without shutting down the computer. All startup procedures described in section 4.2 are again applicable. The information in the Setup block must be re-entered, as discussed in section 4.3.2.3.

5.0 Intermediate Data Processing System

5.1 Overview

The Intermediate Data Processing system is the vehicle for reduction of mixing data to the point suitable for input to the computer programs which evaluate ENERGY code calibration constants. The system consists of three interactive FORTRAN programs, written primarily by Tim Gawne, currently an undergraduate student at MIT. These programs are designed to run on the Interdata 70-Computer in the Mechanical Engineering Civil Engineering Joint Computer Facility (JCF), Rm. 1-111. In addition to extensive graphics, plotting, listing, and hardcopy facilities, the JCF computer has 2 floppy disk drives.

The intermediate data processing programs have been specifically designed to accept as input data stored on the floppy disk by the RTDAS. They may be used at any time to examine probe resistance versus concentration curves, evaluate mixing data, or to provide a hardcopy listing of all data on the disk. The proximity of the JCF to the EPL, and the speed of the turnaround makes feasible evaluation of test data during the course of the mixing experiment.

Use of the Intermediate Data Processing System requires physical removal of the floppy disk containing the data from disk drive B of the RTDAS laboratory computer, and insertion into either of the floppy disk drives of the Interdata Computer in the JCF. Card decks of the source programs, located in the card file cabinet drawer marked "TODREAS", are then

read into the computer by the user.

It should be pointed out that all the programs require specification of floppy disk number before program execution. This is accomplished by 3 punched cards which must immediately precede the //XEQ card in the deck. Required card formats are shown below:

	Column
Card #	1234567890111111111122 012345678901
1	FILE GLAS-TEXT 1 abcd
2	FILE GLAS-HDRS 2 abcd
3	FILE GLAS-DATA 3 abcd

where abcd is the 4 digit (integer) disk identification number. Each disk is labeled by ID number in the upper right hand corner.

Further instructions for use of specific programs are contained in sections 5.2.1-5.2.3.

5.2 Interactive System - User Instructions

Sections 5.2.1 - 5.2.3 describe the sequence of computer generated messages, and the required user responses, in the same format as Chapter 4. Listings of the 3 programs appear in Appendix B.

5.2.1 Hardcopy RTDAS Dataset Listings

This program, entitled "LIST", prints out on the line printer the information contained in the header record and the data record for all datasets on the floppy disk. The program automatically terminates when all datasets have been listed.

No user input is required for this program. It is suggested that the program be run under the M-70 operating system as rates charged are less than for the M-80.

5.2.2 Examination of Calibration Data: Resistance versus Concentration Curves

The program entitled CALIB permits the user to examine and manually change the calibration data in a given calibration series. It scans each dataset in the series, and compiles for each probe a list of the known solution concentrations and corresponding values of probe resistances. The data is presented graphically for probes singly or in groups of 8. A sample set of curves appear in figure 5-1. The number appended to each curve indicate the probe position within the group of 8.

The CALIB program must be run on the M-80 system because of its interactive plotting capabilities and because the M-70 system is typically reserved for very short jobs. It is suggested that this program be run immediately after completion of the calibration procedure, with the RTDAS left in the CAL subprogram, to verify validity of the data. Bad sets of calibration data are indicated by consistent spikes in the resistance versus concentration curves for many, or all, probes at the same value of salt concentration. This data may be retaken by returning the disk to the RTDAS in the laboratory, and repeating the calibration procedure for this concentration. The new data set must be part of the same calibration series, and hence, must be taken before the CAL subprogram is terminated.

The CALIB program ignores all but the last encountered dataset generated with the same salt concentration. The user may effectively invalidate any particular dataset within the series by assigning to it a salt concentration greater than 10000 with the EDT subprogram of the RTDAS.

A flowchart indicating the sequence of computer messages printed at the M-80 console during execution of the CALIB program appears in Figure 5-2. Explanations of message meanings and required user responses are given below.

Message #: 1

Input Required: The name and number of the calibration series.

Input Format: The three characters CAL followed by the series number.

Sample Input: CAL11

Comments: The series named must exist on the disk and must contain at least 2 datasets corresponding to probe resistances at different concentrations.

Message #: 2

Input Required: Decision to produce plots with resistance versus concentration curves for probes singly or in groups of 8.

Input Format: One of the letters: Y(yes), N(no), or Q (quit)

Sample Input: N

Comments: Rates charged by the JCF are based on the total

number of plotter vectors generated. Plots containing curves for 8 probes will save money and time, and permit comparison of curve shapes for several probes at once. Plots of resistance versus concentration curves for single probes is only advantageous if the option to manually change the data is used. Complete identification of individual curves among 8 may be impossible for probes with similar characteristics. It is suggested that 8 curves be displayed on each plot.

Message #: 3

Input Required: The number of the probe to be examined

Input Format: An integer number, in the range $1 \leq \text{probe number} \leq 512$.

Sample Input: 106

Comments: The resistance versus concentration curve for this probe will be displayed.

Message #: 4

Input Required: None

Input Format: NA

Sample Input: NA

Comments: This message is not output to the user. It was included in the flow chart to indicate that selecting all probes to be examined at message #2 causes the program to sequence through all probes

automatically, and display curves for 8
probes at once.

Message #: 5

Input Required: Decision to obtain a CALCOMP plot of the
resistance versus concentration curve

Input Format: One of the letters: Y(yes) or N(no)

Sample Input: N

Comments: This message is printed on the terminal after
the resistance versus concentration plots appear
on the Tektronix CRT. To obtain a hardcopy
of the plot on the screen, push the button marked
"REQUEST" on the small box above the CRT. The
hardcopy will emerge from the Tektronix Hard-
copy Unit. A more detailed plot of probe
resistance versus concentration curves on the
CALCOMP may be requested by typing "Y" at this
point.

Message #: 6

Input Required: Decision to manually modify any points
in resistance versus concentration curves

Input Format: One of the letters: Y(yes) or N(no)

Sample Input: N

Comments: None

Message #: 7

Input Required: Changes to calibration data to modify shape
of probe resistance vs. concentration curves

Input Format: The namelist option is used to change calibration data. To change data, enter the array element name, followed by "=", followed by the new value. This procedure may be repeated for multiple changes. Completion of changes is signalled by entering "&".

Sample Input: CALIB(3,47) = 1274, CALIB(5,16) = 1059

Comments: The CALIB array is a 2 dimensional array containing the probe resistances. The 1st subscript refers to the data set number of the calibration block (i.e., CALIB(1,-) corresponds to the first calibration set), and the second subscript corresponds to the probe number.

Message #: 8

Input Required: Decision to rewrite onto the floppy disk
any changes just made.

Input Format: One of the letters: Y(yes) or N(no)

Sample Input: Y

Comments: All changes are accumulated during the entire curve examination procedure. Changes, if made, need only be rewritten onto the disk once just before the program terminates. This is necessary for subsequent conversion of mixing data to salt concentrations by the MIX program (see Section 5.2.3)

Message #: 9

Input Required: None

Input Format: NA

Sample Input: NA

Comments: No message actually appears at this point. This block was inserted into the flow chart to indicate that the entire curve examination procedure will be repeated for either another user defined probe number, or the next sequential group of 8 probes.

5.2.3 Conversion of Mixing Data to Salt Concentration Maps, Salt Mass Balances, and Preparation of Data for ENERGY Code Parameter Determination

The third of the package of intermediate data reduction programs entitled MIX, performs the actual conversion of mixing data to salt concentrations, prints out maps of salt concentrations, provides salt balances (ratio of salt measured to salt injected) and punches cards in a format suitable for input to existing computer codes which determine calibration parameters for the ENERGY codes. This program may be run either on the M-70 or M-80 operating system. It is recommended that the M-70 system be used when short connection times, resulting when conversion of 10-15 datasets, are anticipated. Reduction of large numbers of mixing datasets should be done on the M-80 system due to the rate differential.

A flowchart indicating system messages and required user inputs is shown in Fig. 5-3. Explanation of messages follows below. The current version of this program will work properly only for a 61 pin bundle. See additional comments in Appendix II.

Message #: 1

Input Required: Cross sectional areas in square inches of
interior, edge, and corner subchannels

Input Format: 3 floating point numbers separated by commas

Sample Input: .0243, .0516, .0212

Comments: This information is necessary during calculation
of salt mass balances (salt detected/salt injected).
Currently, the program assumes only 3 types of
subchannels within the (hexagonal) bundle. Future
allowance for different geometries may be
accomplished through minor program modification.

Message #: 2

Input Required: The number of sets of flowsplit versus
Reynolds number data to be input by the user

Input Format: One integer number, in the range $1 \leq \text{number} \leq 10$

Sample Input: 2

Comments: The program permits user definition of up to 10 sets
of Reynolds number versus flowsplit data. This data
is used only during calculation of salt mass
balances. Currently, the program has been set up
to use flowsplits defined at the nearest Reynolds
number to the individual test data, in computing
salt mass balances. Note that the test Reynolds
number is stored in real time in the header section
of each MIX dataset, so that flowsplit selection
is automatic for all datasets.

Salt mass balances are computed as:

$$\text{Mass Balance} = \frac{\sum_{i=1}^{\# \text{ int. subch.}} A_1 X_1 V_{b \text{ avg}} C_1 + \sum_{j=1}^{\# \text{ edge subch.}} A_2 X_2 V_{b \text{ avg}} C_j + \sum_{k=1}^6 A_3 X_3 V_{b \text{ avg}} C_k}{Q_{\text{inj}} \cdot C_{\text{inj}}} \quad (5.1)$$

where A_1, A_2, A_3 = Areas of interior, edge, and corner subchannels, respectively

X_1, X_2, X_3 = Flow splits ($\frac{V \text{ subchannel type}}{V \text{ bundle average}}$)
for interior, edge, and corner subchannels, respectively.

$V_{b \text{ avg}}$ = average bundle velocity = $\frac{Q \text{ bundle (flowrate)}}{A \text{ bundle}}$

C_1 = concentration of salt measured in each subchannel

C_{inj} = concentration of injection solution

Q_{inj} = volumetric flowrate of injection solution.

Message #: 3

Input Required: Reynolds number, with 3 corresponding values of flowsplit for the interior, edge, and corner subchannels, respectively.

Input Format: Four floating point numbers separated by commas.
The sequence is repeated as many times as specified in the input to message #2.

Sample Input: 10000., 0.85, 1.14, 1.02

Comments: Flowsplits should ideally have been measured or calculated prior to mixing data evaluation. Mass balance calculations are very sensitive to the flow-

splits. Incorrect specifications of flowsplits will also affect salt concentration maps, printouts of dimensionless ratios, and further reduction to the form of ENERGY code parameters ϵ^* and C .

Message #: 4

Input Required: Decision to convert data in all mixing datasets to salt concentrations and dimensionless concentration ratios at once.

Input Format: One of the letters Y(yes) or N(no)

Sample Input: Y

Comments: Y should be specified if all datasets are to be converted automatically. This saves computer time, but also generates much paper if the disk contains many MIX datasets. The "NO" option should be selected if only 1 or a few datasets are to be converted (see Fig. 5-3).

Message #5:

Input Required: Decision to print salt concentration maps, dimensionless concentration ratio maps, both, or neither, for all datasets.

Input Format: One of the following letters

C (Concentration map printout only)

D (Dimensionless Concentration ratio map printout only)

B (Both)

N (Neither)

Sample Output: D

Comments: The salt concentration map is a printout of the salt concentration measured by each probe, after reference to the appropriate calibration curve and correction dataset. Conversion of resistances to concentration is a necessary intermediate step for final determination of ϵ_1^* and C_1 , so no additional computation is performed if the printout is required. It will, however, double the amount of paper generated if both type maps are printed. Data contained in the concentration map, while occasionally useful for checking individual probe validity, is generally not required for determination of ϵ_1^* and C_1 . The user must decide if the cost of the paper is warranted by the data.

Dimensionless ratio maps are printouts of the data in the following form:

$$C_1 = \frac{C_1}{C_o} \quad (5-2)$$

where

$$C_o = \frac{\sum_{i=1}^{\text{\# int. Subch.}} A_1 X_1 V_{b \text{ avg}} \cdot C_1 + \sum_{j=1}^{\text{\# edge Subch.}} A_2 X_2 V_{b \text{ avg}} C_j + \sum_{k=1}^6 A_3 X_3 V_{b \text{ avg}} C_k}{(Q_{inj} + A_{inj} X_{inj} V_{b \text{ avg}})} \quad (5-3)$$

and

A_{inj} = injection subchannel area
 X_{inj} = injection subchannel flowsplit
 $\bar{C}_1$ = dimensionless salt concentration in each subchannel

and A_{inj} = injection subchannel area

X_{inj} = injection subchannel flowsplit

C_1 = dimensionless salt concentration in each
subchannel.

This data is used as input to the programs which determine ϵ_1^* and C_1 . The printout is useful for showing salt mixing patterns. Actual input to the final data reduction programs is through punched cards (see explanation to message #6).

It is suggested that data validity be checked first by requesting printouts of the concentration maps or dimensionless ratios, without punched card output. Depending on certainty of knowledge of the flow splits, salt mass balances consistently close to 1.0 indicate valid data. The MIXING program may then be rerun, and punched cards obtained for input to the final data reduction program.

Message: #6

Input Required: Decision to obtain punched card output

Input Format: One of the letters: Y(yes) or N(no)

Sample Input: N

Comments: The final data reduction program, written previously by C. Chiu, accepts as input the values of dimensionless concentration ratio on cards. The dimensionless concentration ratios for each subchannel, starting with subchannel #1, and continuing sequentially, must appear in a specified integer format, on cards.

Specification of Y at this point causes the cards to be punched at the Information Processing Center under the user number P14411, account M12989. For more information as to final data reduction program input, see Chapter 6.

Message #: 7

Input Required: None

Input Format: NA

Sample Input: NA

Comments: Punched cards and/or concentration and dimensionless concentration ratio maps are produced according to instructions for all MIX datasets on the floppy disk. The program will automatically terminate when all datasets have been processed.

Message #: 8

Input Required: The name of the mixing dataset for which salt concentration maps, dimensionless concentration ratio maps, punched card output is to be produced.

Input Format: The complete name of the MIX dataset, as outlined in section 3.2.1.

Sample Input: MIX14.01

Comments: None

Message #: 9

Input Required: Decision to input either a salt concentration map, a dimensionless concentration ratio map, both , or neither

Input Format: One of the letters: C,D,B, or N

Sample Input: D

Comments: See comments for message #5.

Message #: 10

Input Required: Decision to punch cards containing dimensionless concentration ratios.

Input Format: One of the letters: Y(yes) or N(no)

Sample Input: N

Comments: See comments for message #6

Message #: 11

Input Required: None

Input Format: NA

Sample Input: NA

Comments: On completion of execution of user instructions, the program returns to request another MIX dataset as shown in Fig. 5-3. Note: The only way to terminate the program when in this branch is to manually push the red button marked "QUIT" on the switch box near the console.

6.0. Determination of ENERGY Code Parameters ϵ_1^* and C_1

6.1. Overview

The computer code "SUPERENERGY/DATA REDUCTION VERSION" hereafter called SUPERENERGY/DRV, has been developed to calibrate the mixing parameters ϵ_1^* and C_1 against the experimental mixing data. The experimental mixing data are presented in the form of dimensionless salt concentrations and punched onto computer cards in a format of (16F5.1) (See Section 5.2.2). These data cards are used directly as part of the input deck to the code "SUPER-ENERGY/DRV."

Computer core storage requirements of approximately 118K bytes require that the code be run at the computer facility of the Nuclear Science Laboratory (LNS) located in the basement of Bldg. 24. The normal execution time for a typical run with 30 trial values of ϵ_1^* or C_1 , calibrated against data is approximately 10 minutes. The card deck of this program is stored in in the drawer labeled "MIXING" in a cabinet at the Mechanical and Civil Engineering Joint Computer Facility.

The recommended procedure for calibration of ϵ_1^* and C_1 , is shown in figure 6-1. The enhanced eddy diffusivity, ϵ_1^* , should first be calibrated using the dimensionless salt concentration data corresponding to a case of salt solution injection from the center pin. Unless injection depth exceeds several wire wrap lead lengths, the salt should remain entirely in the central region. Salt data in this form permits decoupling the effects of the eddy diffusivity, ϵ_1^* from C_1 , and allows individual

parameter determination. In this case, ϵ_1^* is varied over a range while C_1 is set to any arbitrary (fixed) value. The technique for selection of the most correct value of ϵ_1^* and C_1 is outline in section 6.2. When a single value of ϵ_1^* is determined in this fashion, C_1 may be found by using a dimensionless salt concentration map resulting from edge region salt injection. The presence of salt in both the edge and interior regions at the outlet plane indicates that the effects of ϵ_1^* and C_1 , may not be decoupled. In this case, C_1 must be varied over a range, while ϵ_1^* is held fixed at the previously determined value. The methods of section 6.2 should then be used to select the most correct value of C_1 . Note that ϵ_1^* and C_1 may be uniquely determined for any bundle geometry at a particular Reynolds number by examination of only 2 datasets. Data collection at several axial locations may still be desirable to graphically illustrate mixing patterns and to provide the ability to check ϵ_1^* and C_1 against additional datasets if errors are suspected.

6.2. Code Structure and Determination of Optimum Values of ϵ_1^* and C_1

The mixing parameters are determined by using a least mean square technique to identify the closest match of SUPER-ENERGY/DRV generated temperatures at particular values of ϵ_1^* and C_1 , with actual dimensionless salt concentration data. The least mean square indicator corresponding to a specific set of mixing parameters ϵ_1^* is calculated on a subchannel by subchannel basis, as:

$$LMS = \sqrt{\sum_i (C_i^* - T_i^*)^2} \quad (6-1)$$

where T^* $\equiv$ dimensionless temperature

C_i^* $\equiv$ dimensionless salt concentration
(input into computer code)

The value of LMS gives an indication of the closeness of the match of the dimensionless temperature distribution to the experimental dimensionless salt distribution. The values of ϵ_1^* and C_1 corresponding to a minimum value of LMS are the recommended mixing parameters. According to the above philosophy, the SUPERENERGY/DRV determines the ϵ_1^* and C_1 corresponding to a minimum value of LMS over possible ranges of ϵ_1^* or C_1 , which are input to the code, and prints out the dimensionless temperature distribution corresponding to the recommended values of the mixing parameters. The flow chart of the SUPERENERGY/DRV code is shown in Figure 6-2. The listing of this code is contained in appendix 3.

6.3. Input to the SUPERENERGY/DRV Code

The users are required to input the following parameters in the SUPERENERGY/DRV code:

- 1) bundle geometric parameters - rod diameter, wire wrap diameter, pitch (including tolerance), flat to flat distance, number of rods.

- 2) Range of ϵ_1^* (or C_1) which is expected to cover the value ϵ_1^* (or C_1) corresponding to the minimum value of LMS and the number of ϵ_1^* (or C_1) values in the range to be examined against the experimental data ,
- 3) Flow split data - x_1 , x_2 and x_3 . If experimental data on x_1 , x_2 and x_3 are not available, the code can generate these parameters according to Norendstern's method.⁽²⁾
- 4) Injection subchannel identification number (The subchannel numbering scheme is established in the reference (3)) and, injection depth
- 5) Bundle operating conditions: bundle total flow rate and Reynolds number.

The formats and specifications of these input parameters are listed below.

Card Number 1

Required to be present: always

FORTTRAN READ list: JXQ, NCOE, NINJ, NROD, IW, IWC,
NOPRNT

FORTTRAN FORMAT: 7I5

<u>Variable</u>	<u>Columns</u>	<u>Format</u>	<u>Description</u>
JXQ	1-5	I5	Number of ϵ_1^* (or) C_1 Values in the range of E_1^* to be examined.
NCOE	6-10	I5	NCOE = 1 for ϵ_1^* calibration NCOE $\neq$ 1 for C_1 calibration
NINJ	11-15	I5	Identification number of salt injection subchannel
NROD	16-20	I5	Number of rods in the bundle
IW	21-25	I5	IW = 0, Read in x_1, x_2 and x_3 IW $\neq$ 0, x_1, x_2 and x_3 are calculated.
IWC	26-30	I5	IWC = 0, Calibration calculation IWC = 1, Prediction calculation
NOPRNT	31-35	I5	number of printouts of dimensionless temperature distribution

Card Number 2

Required to be present: IF IWC = 1

FORTTRAN READ list: (PRNT(I), I = 1, NOPRNT)

FORTTRAN FORMAT: 8E10.0

<u>Variables</u>	<u>Columns</u>	<u>Format</u>	<u>Description</u>
PRNT	1-80	E10.0	Axial heights at which the dimensionless temperature distribution is printed out.

NOTE:

IWC = 0, code determines optimum ϵ_1^* (or C_1) from
experimental dimensionless salt concentration data

IWC = 1, code predicts the dimensionless salt concentration map at heights specified by PRNT(I).

Card Number 3

Required to be present: always

FORTTRAN READ list: CHA,CHI,CHC,REE,FLOW

FORTTRAN FORMAT: 8E10.0

<u>Variable</u>	<u>Columns</u>	<u>Format</u>	<u>Description</u>
CHA	1-10	E10.1	Maximum expected value of ϵ_1^* for NCOE = 1 (or C_1 for NCOE $\neq$ 1)
CHI	11-20	E10.1	Minimum expected value of ϵ_1^* for NCOE = 1 (or C_1 for NCOE $\neq$ 1)
CHC	21-30	E10.1	C_1 , for NCOE = 1 ϵ_1^* for NCOE $\neq$ 1
REE	31-40	E10.1	Bundle Reynolds Number
FLOW	41-50	E10.1	Bundle flow rate (GPM)

NOTE: IF IWC=1, CHA = CHI = ϵ_1^* and CHC = C_1 , where ϵ_1^* and C_1 are known mixing parameters.

Card Number 4

Required to be present = always

FORTTRAN READ list = DR,DW,LEAD,P, D FLAT, LENGTH

FORTTRAN FORMAT: 8E10.1

Variable	Columns	Format	Description
DR	1-10	E10.1	Rod Diameter (inch)
DW	11-20	E10.1	Wire Diameter (inch)
LEAD	21-30	E10.1	Wire Wrap lead (inch)
P	31-40	E10.1	Pitch (inch)
DFLAT	41-50	E10.1	Flat to Flat distance (inch)
LENGTH	51-60	E10.1	IF IWC = 0 injection depth (inch) IF IWC = 1 bundle length (inch)
DWS	61-70	E10.1	Shaved wire diameter. If the wires are not shaved, input DW

Card Number 5

Required to be present: if IW = 0

FORTTRAN READ list: VIR, V2R, V3R

FORTTRAN FORMAT: 8E10.1

Variable	Columns	Format	Description
V1R	1-10	E10.0	x_1 ratio of average inter- ior subchannel velocity to average bundle velocity
V2R	11-20	E10.0	x_2 , ration of average edge subchannel velocity to aver- age bundle velocity.
V3R	21-30	E10.0	x_3 , ratio of average corner subchannel velocity to aver- age bundle velocity.

NOTE: IF shaved wire-wrapped assembly is under investigation,
V1R, V2R and V3R is recommended to be input.

Card Number 6

Required to be present: IWC=0

FORTTRAN READ list: (SALT (I), I = 1, M4)

FORTTRAN FORMAT: 16E5.0

Variable	Columns	Format	Description
SALT	1-80	E5.0	Dimensionless salt concentration data for each sub-channel, as produced by the intermediate data processing system.

NOTE: For 61 pin bundle, M4 = 126.

6.4 Sample Problem

Dimensionless salt concentration experimental data for the 4 inch lead blanket assembly mockup test ⁽⁴⁾ shown in figure 6-3, is calibrated to obtain the ϵ_1^* by SUPERENERGY/DRV. The input and output listings are presented in figures 6-4 and 6-5 respectively.

7.0 Future Work

The complete operation of the RTDAS, and its integration into the total procedure for determination of ENERGY code calibration constants has been discussed in this document. This section will briefly discuss additional applications for which the system may be adapted, and directions for future system modification to enhance its convenience and operating capabilities.

Expansion of the system to permit data taking during flow transients may be desirable in the future. Currently, to facilitate program development, data storage onto floppy disks is done under the high level language FORTRAN. Transfer of a complete dataset to the disk is relatively slow, requiring of the order of 1/2 second. This may be greatly speeded up by rewriting sections of the RTDAS software so that core to disk data transfers are directed at the assembly language level. Double buffering of rapidly collected data may also be used.

Most of the MIX subprogram may be used to gather prerequisite descriptive data about the transient dataset, but a small subroutine must be written to actually control transient data collection and storage. Note that an internal clock may be used to trigger data collection at precise intervals, and that space has already been reserved in the header record of MIX datasets to record the precise time of collection. Difficulties may arise however, in integration of main bundle flow coastdown with precisely taken probe resistance measurements.

An additional type of experiment related to transient

data collection is flow split determination by the "time of flight" technique. This involves injection of a slug of salt into the bundle, and measuring probe resistances at the outlet repetitively as quickly as possible. The time after injection that a sharp drop in probe resistance is recorded at the bundle outlet may be used to determine axial fluid velocity. Accuracy from tests performed in this way have not been good, and difficulties are anticipated with instantaneous salt injection at known positions. This technique is only suggested as an alternative to the isokinetic sampling method currently employed to determine flow splits. At this time, it is felt that it is not worthwhile to modify the RTDAS to perform the time of flight experiments.

Hardware already built expressly for the mixing project includes integer multiply/divide circuitry and a multiplexor and ADC equipment. Additionally, the project has purchased a high speed video terminal. The next logical system expansion, which could be utilized by all other users of the laboratory Interdata Computer, is an additional 32K of core memory. This would permit operation of much larger programs, and would allow certain data processing functions, such as real time conversion of resistances to concentration on examination of concentration maps, and determination of salt mass balances, to be performed by the Interdata 716 minicomputer in the laboratory.

Another technique may be investigated for determining flow split data from salt mixing experiments. As indicated in section 5.2.3, salt mass balances are highly sensitive to flowsplits. It may be desirable to investigate modification of the Inter-

mediate Data Reduction Mixing code to determine the values of flowsplit which would make the mass balance = 1.0. This technique may prove acceptable for flow split determination in turbulent flow for new bundle geometries, but caution must be exercised in the laminar and transition regions. Data reported by C. Chiu indicates salt mass balances in the range of 75-90% for tests run at the lower Reynolds numbers.

The Real Time Data Acquisition System currently performs all tasks necessary during the conduct of steady state salt mixing experiments. Only software modifications are needed to perform transient mixing tests and time of flight tests. Further expansion of the system to perform additional types of experiments not now considered possible should be investigated.

FIGURES

Square
Wave
Input

Voltage Measured at ADC

$$V = \frac{R_{\text{probe}}}{R_{\text{probe}} + R_{\text{ref}}} \cdot 10V$$

Fig. 2.1a Indirect Probe Resistance Measurement

Fig. 2.1b Input, Probe, and Sampled Voltage Signals

BLUE-BOX MULTIPLY-DIVIDE UNIT

KY 12 OCT 1976

Device

OC

WH

RH

(NOTE: Bit 12=0 signed, bit 12=1 unsigned arithmetic.)

Sets mode for multiply/divide (see below, bits 13-15)

Loads registers prior to operation, in order listed below.

Operation proceeds as soon as all data is available.

Reads back results following an operation.

- Reads Register L following an WH

- Reads Register H following a RH

(i.e. first L,
H second)

Three 16-bit shift regs. are used within the multiply/divide units: K, H, L

Multiply: $K * L \Rightarrow H, L$ (32-bit product, H = high order 16 bits
L = low order 16 bits)Divide: $H, L \div K \Rightarrow L \text{ rem } H$ (32-bit dividend H = high order 16 bits
L = low order 16 bits)

K = 16-bit divisor

Results:

L is 16-bit quotient

H is 16-bit remainder (same sign as dividend.)

MODE: 0 (0) WH $\rightarrow$ L(1) WH $\rightarrow$ K

(2) multiply

O $\rightarrow$ H

(shifted 16)

A
add
subtractALU
M 32 10K
SL SR RH
SL SR RL
SL SR R

Shifting:

0 0

0 0

1 1

1 1

X X 1

0 1

H00 $\rightarrow$ H $\rightarrow$ L $\rightarrow$ mpy.1 (1) WH $\rightarrow$ K

(2) multiply

O $\rightarrow$ H4 (4) WH $\rightarrow$ K(5) WH $\rightarrow$ L(6) WH $\rightarrow$ K

(7) divide

K $\rightarrow$ H

(shifted 16, add/subtract or shift)

1 1 0 1 0

0 0

1 1

1 1

1 1

0 0

0 0

0 0

0 0

1 0

1 0

1 0

E $\leftarrow$ H $\leftarrow$ L $\leftarrow$ Quotient5 (5) WH $\rightarrow$ L(6) WH $\rightarrow$ K

(7) divide

sign L $\rightarrow$ H

D00=0

1 0 0 1 1

0 0

1 1

1 1

D00=1

1 1 1 0 0

1 1

0 0

0 0

6 (6) WH $\rightarrow$ K

(7) divide

1 1

0 0

0 0

(8) if quotient < 0 ,

0 0

0 0

0 1

-L $\rightarrow$ L

ALU:

add

subtract

'8' (load K)

0

1

M S3 S2 S1 S0

0 1 0 0 1

0 0 1 1 0

1 1 0 1 0

1 0 0 1 1

1 1 1 0 0

ADD

SUB

LDK

LDZ

LDU

Fig. 2-2 Operation of Blue Box Multiply Divide Unit

Fig. 2-4 Multiply/Divide Unit Layout

BLUE-BOX ANNEX

ALU and Multiply/Divide Registers

KY 12 September 1976

A

Fig. 2-5 Multiply/Divide Unit Control Circuitry Layout

BLUE-BOX ANNEX

Multiply / Divide Control

KR 13 October 1976

Fig. 2-6 Multiply/Divide Unit Control Circuitry

BLUE-BOX ANNEX

Multiply / Divide Control

KY 14 October 1976

BLUE-Box ANNEX: 512 ANALOG CHANNEL A-D PROGRAMMING KY 12-3-76

Device x'50' (data buss)
x'40' Register 40 is list address of command words.
x'41' Register 41 is data buffer address for conversion data.

CMD x'50' bit 14=1 starts conversion routine using present values in registers 40 and 41 to execute a list of programmable commands.

SR x'50'

WH x'40' sets list address (may also be set by list command x'6000')
RH x'40' reads list address
WH x'41' sets data address (may also be set by list command x'7000')
RH x'41' reads data address
OC x'40' bits 8,9 set enable and arm flip-flops for interrupts (standard)

List Commands:

0xxx special bit 15=0 stop

bit 14=1 interrupt

1nnn set channel channel counter = nnn, and one conversion is done.

(After conversion, channel counter is incremented
data is stored in memory at data address.
data address is incremented by 2 bytes)

2xxx convert group

The next nn channels are converted and stored in memory
at the data address (register x'41'). Note that the channel
counter gets incremented by nn, the data address is incremented
by 2*nn bytes.

Note: conversions are done synchronously with the falling edge
of the square-wave timing signal.

3xxx delay

List execution is delayed for nn cycles of the square-wave signal.

4xxx set frequency

Sets the period of the square-wave timing signal to:

$(4096 - nnn) * 0.25 \mu\text{sec} * 2$

Note: ADC requires at least 10 μsec cycle.)

5xxx (no function)

6xxx
aaaa

'don't care' address - Causes following word to be loaded into register x'40'
This is effectively a branch instruction.

(May be used to implement continuous operation.)

7xxx
aaaa

load data address - causes following word to be loaded into register x'41'
this changes location of the data buffer.
(May be used to implement double buffering.)

Fig. 2-8 Multiplexor Circuitry Timing Diagram

BLUE-BOX ANNEX

Timing for analog multiplexor.

KY 12-3-76

KY 15 Sept 1976

[illegible]

Fig. 2-10 Multiplexor Support Hardware Circuitry
(in Blue Box Annex)

KY 12 September 1976

BLUE-BOX ANNEX

Fig. 2-11 Multiplexor Support Hardware
Circuitry (In Blue Box Annex)

BLUE-BOX ANNEX

KY 12 September 1976

Fig. 2-12 Multiplexor Support Hardware
(in Blue Box Annex)

KY 12 September 1976

BLUE-BOX ANNEX

[illegible][illegible][illegible]

Fig. 2-13
(sheet 3 of 3)

BLUE BOX ANNEX
BOTTOM VIEW 9-12-76

Fig. 2-14 Changes to Main Board in Blue Box Annex

BLUE-BOX ANNEX

KY 12-3-76

Changes to main board: (see also: CONNECTORS TO MAIN BOARD.)

Note: 11N is now 74S153

16P is now 74S00

Fig. 2-15 Connectors to Main Board of Blue Box Annex

BLUE BOX ANNEX — CONNECTORS TO MAIN BOARD

16 Sept 1976

CONN. A (50 pin SCOTCH FLEX)

Pin	Signal	Main Board	Pin	Signal	Main Board
200	D00	11D3	100	M00	12D3
201	D01	6	101	M01	13D3
202	D02	8	102	M02	14D3
203	D03	11	103	M03	15D3
204	D04	11E3	104	M04	12E3
205	D05	6	105	M05	13E3
206	D06	8	106	M06	14E3
207	D07	11	107	M07	15E3
208	D08	11F3	108	M08	12F3
209	D09	6	109	M09	13F3
210	D10	8	110	M10	14F3
211	D11	11	111	M11	15F3
212	D12	11H3	112	M12	12H3
213	D13	6	113	M13	13H3
214	D14	8	114	M14	14H3
215	D15	11	115	M15	15H3
216	.		116	ANX0	13L3
217	.		117	MC0	17J6
218	.		118	PSEL	10L13
219	.		119	RAD10	11L7
220	.		120	RAD11	11L9
221	.		121	RAD12	11M7
222	.		122	RAD13	17L13
223	.		123	RAD14	17L14
224	.		124	RAD15	17L15

CONN. B

Pin	Signal	Main Board	Pin	Signal	Main Board
200	GCMD	16M2	100	GDA	19N4
201	GSR	13C4	101	GDR	19N1
202	MAV	16N6	102	MADR	20N4
203	MDAT	22J3	103	MWR	21L6
204	CLRO	22L10	104	4MHz	18L3
205	SCO	17J8	105	.	
206	SEL410	76P12	106	SNRQ0	18J4
207	RAMLDO	16P9	107	SNSEL	21J3
208	BSYMD0	12L9	108	SNW0	14P4
209			109		
210			110		
211			111		
212			112		
213	.		113	+5	
214	.		114	+5	
215	.		115	+5	
216	.		116	+5	

Fig. 2-16 Signal Cross Reference List
in Blue Box Annex (Sheet 1 of 3)

127

BLUE - BOX ANNEX: SIGNAL CROSS REFERENCE

KY 15 Oct 1976

Signal	Origin	Use
ADMI	B-18F13	22E3 C-15E9
ANXI		
ALUM	C-18H6 B-17F11	>A116 H-13,14A,B,C,D 15 A-17,18A,B 8
ALUOS	C-18D8	" 6
ALUIS	C-17D11	" 5
ALU2S	C-18D6	" 4
ALU3S	C-17D3	" 3
AS	B-16F3 B-23F6	17F13 18F8,12 C-16E1 17F4 B-16F1
E20-315	E	A(00-15) A(00) B(12-15) C(000,001) D(14,15)
ALU00-ALUIS	A	A
BSYMD	B-21F8 B-19D2	B-23H1,10 C-21E1 B-19D1 >B208
CCAR	F-4B15	D-14E4 GCMD >B200 B-23F12 B-17E10 D-14F5
CLR		
CHA	>B204 D-15H14	D-10H15 D-14H5 D-13H5,11 D-12H14,4 D-11H4,14 F-4B,C1 B-20H1 16H1 D-13E4 D-10F9
CM8	D-15H13 D-12E12	16H2 12E13 16E10 D-16E5 D-11F4
CMC	D-15H12	16H3 16E11 D-16E4
CMP	B-21H11 B-22F2	22F1 A-22E10 A-23E2 22H6 21F11 C-20E10 20E5
CONT	D-15E4	D-14F9
CBUNT	D-13F12	11F5 15F12 F-4B,C7 F-3A,B,C7
* CAR		
	A-18A16	A-16D12 A-9F210
DIV	B-22F8 B-22H7	B-21F4 C-19F1,13 18H9 C-19H4 22F9 21F9 B-17F12 C-18E10
DLYF	D-13H9	
	10	15F11 D-15E6
DV4		
	B-22H11	C-21F2
DV5		
	B-22H10	C-18F2,5 19E1 C-18E13
DV6	B-22F10 B-22H9	C-19H13 C-19H1,2 22F11 C-21F13
EDC	D-12F3	
	2	13E4
FIT	A-19D4 A-21D8	B-11E1 19D3 C-18E9

BLUE-Box ANNEX

page 2

FREQ	E-5A15 E-12E6	11H2,3 5A,8,C9	12E5 D-13F3	D-13F13	11H12	15F5		
					GDA	>B100	C-17F2	
GET	D-13F6	12H3,11	2C4,5	17H1	GDR	>B101	C-17H4	1844
GETP					GCMD	(alpha under cmd)		
	D-17H3	F-1A9	1B9					
GØ	D-11H9 7	15F4	D-14E1		GSR	>B201		
GØT	D-12H5 6	13F5	D-14E2					
H00-H15	A	A(00-15)	A(15)	H(00-15)				
HØLD	D-13H7 6	13F1	D-14F3					
HSGN	C-21E9 8	19F4 19F10	A-15Ø9	A-16D13				
HSL	C-20E12	A-19A,B23						
HSR	C-19E6	A-19A,B1			MAV	>B202	D-17D10	
K00-K15	A	A(000,001)	C(000,001)		MSA	C-17H8	H-13,14A,B,C,D14	
KSLD	C-17D6	A-16A,B23			MSB	=MPDØ	H-13,14A,B,C,D2	
LCNT								
	D-13F8	13H3	D-14F13	F-4B,C9				
LDK	C-18F3,4			LIN	A-23E8	A-15A2		
LDU				LSL	C-19E8	A-15A,B23		
	C-18D3,4	17D12		LSR	C-20E6	A-15A,B1		
LDZ								
	C-18D11,10	17D2						
L00-L15	A	A(00,15)	B(151,15Ø,151,141)	H(00-15)				
MCH	C-9H3,4,5	A-22E13	B-16F11	B-17E9	B-22E1	C-19H9	D17H2	D-1ØH1
		D-11H13	D-12H1,13	E-11H1				
MCO	C-9H6	A-15A,B11	A-16A,B11	A-19A,B11	B-23H2	B-21H2	C-21E3	D-12F9 15H2
		D-13H4,12	D-14H4		E-5A,B,C2	E-14E9	F-4B,C2	F-3A,B,C2 C-14H12
MDA	B-2ØH15 14	21H5	C-18F6					
MDB	B-2ØH1Ø 11	21H4		LCØØ-11	F	H(ØØ-11)		
MDC	B-2ØH7 6	21H3		MØØ-15	H			
MPØ1				MPD	C-12E2	17F1	B-23F13	B-17E11 C-17H5,9 and 18H5
Ø	B-22H15	C-19E2			C-8H14	12E1		
MP11				MADR	>B1Ø2	D-11F9	13E2	
Ø	E-22H14	A-14E13	C-21F1					
MPY	B-22F12	B-22E4	A-23E13	B-23F1	B-2ØF4,1Ø	C-16E3	17H12	
	B-22H13	22F13	21F1Ø					
MDAT	>B2Ø3	D-1ØH2	D-16F4	MWR	>B1Ø3	C-18H3		

BLUE-Box ANNEX

Page 3

PDS	B-22E5 6	B-23F3 B-23F5			RAMLD	D-12E10 D-16E8, 10F10, 14E11	13E5 16H6	D-15D15
QNEG	C-19D6 C-19F8	15E8 19D5	18H10,11		A-16D2			
SC								
SGN	B-20H3 2	A-21D6 A-21D10	B-20F3,11 C-19H10	C-19F2,12	B-9F12	B-15D12		
SHIFT	C-20E8	B-23H7	B-17F10	C-19H11	C-19H5	B-17B5		
SNA	D-11F8	12H12	D-11F1	D-11E4				
SNAR	D-11F3	10H3	D-14H2		SH	B-17E6	C-18E11	
SNCH								
SNCHT	D-16H14	D-13F11	F-3A,B,C9					
SNCLY	D-16H13	D-13F10			SCN	D-15F10 9	11H11 F-2A1,15 2B1,15	
SNFRQ	D-16H12	D-13F9	13H13		SN	C-12E4 C-8H15	D-14H4 12E3	
SNLC	D-16H11	E-14E10			SEL41	D-13E6	>B206	
SNLD	D-16H9				SNRB	D-13E8	>B106	
SNLSP	D-16H7				SNSEL	D>10Z,12E9,11F10,13E1	C-18H1,2 D-17D9	
SQW	E-11H5 6	2C12,13	D-13F2		SNW	D-12E8 D-11E6	>B108	
STA	B-21H14	22H1						
STB	B-21H13	22H2						
STC	B-21H12	22H3,22F3						
SUB	B-22F4							
SUBM	C-18H8 B-18F10,11	17D1 22E2	18D5 C-18H12,13	A-17B7	A-20D13			
SXF	B-23H15 B-22F6	22F5 B-20F12	17F9	B-20F2	C-18E3,4			
SXS	B-17F8	B-21F3	23F9	B-11E2				
WHC	B-16F10 9	C-19H12 A-14E12	23E5					
WHM	C-17F3 C-15F13	19H3 C-18E1,2	B-16F12	C-19E13	C-17D4	C-14H14	C-15F15	
WRITE	D-12H9 7	D-11E5 D-11F2	13E3					
STR	D-15F6 7	2C1	D-12H2					

12 September 1976

ANALOG MULTIPLEXOR

The circuit diagram illustrates an Analog Multiplexer system. Key components and their connections include:

- Power Supply:** +15V and -15V rails with decoupling capacitors (4.7μF and 0.1μF).
- DAC (12-bit):** Includes a 6.3K Bipolar Offset, Gain Adjust, and Reference Out. It is connected to a 100kΩ resistor network.
- Comparator:** Receives a 10V input and a 20V input, with a 5KΩ feedback resistor.
- Sample and Hold Amplifier (SHM-10-1):** A 14-pin DIP connected for ±10V input. It includes a 0.001μF capacitor and a 100V Polystyrene capacitor.
- ADC (H212B):** A 32-pin DIP connected for ±10V input. It includes a 100kΩ resistor network and a 10V input.
- Logic and Timing:** Includes a 74367 decoder, a 74174 counter, and a 74265 timer. The circuit is controlled by digital signals (START, GETO, SQWAVE, HOLD) and a clock rate.
- Output:** The circuit outputs a digital signal (EPCO C115) and an analog signal (EPCO C114).

Notes:

- Above for ±10V range 2-7-77
- Change for 0-5V operation

Above for $\pm 10\text{v}$ range
2-7-77
Change for 0-5v operation

Fig. 2-18 Multiplexor Circuitry (In "White Box")

ANALOG MULTIPLEXOR

23 Sept 1976

Fig. 2-19 Multiplexor Circuitry (In "White Box")

ANALOG MULTIPLEXOR

6-8-76

Fig. 2-20 Multiplexor Circuitry (In "White Box")

6-8-76

ANALOG MULTIPLEXOR

is circuitry is repeated 16 times.

Fig. 2-21 Multiplexor Circuitry Layout Diagram (In White Box, Bottom View, Sheet 1 or 2)

ANALOG MULTIPLEXOR

KY 6-7-76

First of Each Group of 8 Channels
Begins at End of Socket with Cutoff Corner

2	4	10	12	18	20	26	28	34	36	42	44	50	52	58	60
---	---	----	----	----	----	----	----	----	----	----	----	----	----	----	----

Group 1*	3	9	11	17	19	25	27	33	35	41	43	49	51	57	59
-------------	---	---	----	----	----	----	----	----	----	----	----	----	----	----	----

6	8	14	16	22	24	30	32	38	40	46	48	54	56	62	64
---	---	----	----	----	----	----	----	----	----	----	----	----	----	----	----

5	7	13	15	21	23	29	31	37	39	45	47	53	55	61	63
---	---	----	----	----	----	----	----	----	----	----	----	----	----	----	----

16 Pin Connector to Computer

FRONT PANEL

* Channel numbers corresponding to group
number G are: From $(G-1)*8+1$ to $G*8$

On-Off
Switch

FIG. 2-22 "White Box" Channel Numbering Scheme (Top View)

* First Record of HEADER File Contains RTDAS Information

Fig. 3-1 Disk File Organization

(Half)

Word#

1	# of Next Available HEADER Record
2	# of Next Available DATA Record
3	# of Next SET Series
4	# of Next CAL Series
5	# of Next MIX Series
6	Unused
7	"
8	"
9	"
10	"
11	"
12	"
13	"
14	"
15	"
16	"
17	"
18	"
19	"
20	"
21	"
22	"
23	"
24	"

Fig. 3-2 Directory Information Contained
in First Record of HEADER File

(Half)
Word #

1	Dataset Name*
2	"
3	"
4	"
5	"
6	Record # of Start of Data
7	# of 128 (Half) Word DATA Records
8	Unused
9	Data Type = 1
10	Unused
11	"
12	Bundle ID #
13	Date (Mo/Day/Yr)**
14	Time of Day (Hr/Min)***
15	Unused
16	Probe Type = 1
17	# Probes to be Sampled
18	Square Wave Period****
19	Sample Duration
20	Unused
21	"
22	"
23	"
24	"

* Name of Dataset is stored in BYTES 1-10 in ASCII, 1 Character per Byte. Ex: SET01, or CAL01.01

** Date Stored in Following Format:

*** Time Stored in Following Format

**** Period Stored in Units of Half μ Sec.

Fig. 3-3 HEADER Record of a SET Dataset

(Half)

Word #

1	Dataset Name*
2	"
3	"
4	"
5	"
6	Record # of Start of Data
7	# of 128 (Half) Word DATA Records
8	Unused
9	Data Type = 2
10	Series # of SET Block**
11	Unused
12	Bundle ID #
13	Date (Mo/Day/Yr)*
14	Time of Day (Hr/Min)*
15	Unused
16	Water Temperature (°C) ***
17	Salt Concentration (GM/1000 lb Water)
18	Unused
19	"
20	"
21	"
22	"
23	"
24	"

* See Notes on Fig. 3-3

** # of Probes sampled, Oscillator Frequency, and Order of Channel Sampling are Defined in this SET Dataset

*** Stored as water temperature *10 (i.e. 26.5°C stored as 265)

Fig. 3-4 HEADER Record of a CAL Dataset

1	Dataset Name*
2	"
3	"
4	"
5	"
6	Record # of Start of Data
7	# of 128 (Half) Word DATA Records
8	Unused
9	Data Type = 3
10	Series # of SET Block
11	Series # of CAL Block ***
12	Bundle ID#
13	Date (Mo/Day/Yr)*
14	Time of Day (Hrs, Min) *
15	Time (Sec, Msec)****
16	Water Temperature (°C) **
17	Bundle Flowrate (GPM)*****
18	Unused
19	Bundle Reynolds Number
20	Injection Solution Concentration*****
21	Injection Solution Flowrate*****($\text{gpm} \times 10^6$)
22	Injection Subchannel
23	Axial Height of Injection (In.*10)
24	Series # of Calibration Correction Data *****

* See notes on Fig. 3-3

** See notes on Fig. 3-4

*** Required for Conversion of Probe Resistances to Concentrations

**** Stored in Format:

Required for Future	Bit#16	11	10	1
Transient Mixing Data Capability	Sec			Msec

***** Required for Salt Mass Balance

***** Used to shift R vs C curves according to real time correction

Fig. 3-5 HEADER Record of a MIX Dataset

* Numbers beside message blocks identify individual messages for descriptions in the text.

Fig. 4-1 Initial Input of Time, Date, and Bundle Geometric Data

Fig. 4-2 Setup of Data Collection Parameters - SET

Fig. 4-2 (sheet 2 of 2)

Fig. 4-3 Probe Calibration-CAL

Fig. 4-3 (Sheet 2 of 2)

Fig. 4-4 Collection of Mixing Data - MIX

Fig. 4-4 (Sheet 2 of 5)

Fig. 4-4 (Sheet 3 of 5)

Fig. 4-4 (Sheet 4 of 5)

Fig. 4-4 (Sheet 5 of 5)

Fig. 4-5 Data Set Editing-EDT

Fig. 5-1
Typical Calibration Curves for Probes #17-24

Fig. 5-2 Calibration Curve Examination Program

Fig. 5-3 Mixing Data Reduction Program

Figure 6-1. Data calibration procedure for ϵ_1^* and C_1 by SUPERENERGY/DRV.

Figure 6-2. Flow diagram of SUPERENERGY/DRV.

CENTER INJECTION AT CHANNEL 6

INJECTION POSITION = 16.00 INCHES FROM EXIT

REYNOLD NUMBER = 10650.2

SALT OBSERVED IN GRAM-SALT/HR = 93.18

SALT INJECTION IN GRAM-SALT/HR = 98.48

INITIAL SALT CONCENTRATION IN MG-SALT/LBM-WATER AT CHANNEL 6 = 194.8

SALT MASS BALANCE IN % = 94.6%

DATA IN THIS MAP IN $C(I)/C_0\%$, WHERE C_0 IS EFFECTIVE ORIGINAL SALT CONCENTRATION

Figure 6-3 Dimensionless Salt Concentration Map in the Uniform In-Phase Wire Wrap Bundle

159

Figure 6-4. Input to Sample Problem

DIMENSIONLESS RATIOS OF SALT CONCENTRATION * 0.01
 AXIAL LEVEL (IN.) = 16.0
 INJ. SUBCHANNEL = 6
 REYNOLDS NUMBER = 10650.0

0.0	0.0	0.0	0.0	0.0	0.0				
	0.0	0.1	0.1	0.1					
0.0	0.0	0.1	0.1	0.1	0.1	0.0			
	0.1	0.2	0.4	0.4	0.2				
0.0	0.0	0.2	0.5	0.7	0.5	0.2	0.0		
	0.1	0.4	1.1	1.6	1.1	0.4			
0.0	0.0	0.2	1.0	2.1	2.1	1.0	0.2	0.0	
	0.1	0.4	1.6	3.4	3.4	1.6	0.4		
0.0	0.0	0.2	1.0	3.0	4.5	3.0	1.0	0.2	0.0
0.0									0.0
0.0	0.0	0.2	1.1	3.4	5.1	3.4	1.1	0.2	0.1
	0.1	0.5	2.1	4.5	4.5	2.1	0.5		
0.0	0.1	0.4	1.6	3.4	3.4	1.6	0.4	0.1	
	0.1	0.7	2.1	3.0	2.1	0.7			
0.0	0.1	0.4	1.1	1.6	1.1	0.4	0.1		
	0.1	0.5	1.0	1.0	0.5				
0.0	0.1	0.2	0.4	0.4	0.2	0.0			
	0.1	0.2	0.2	0.2					
0.0	0.0	0.0	0.0	0.0	0.0	0.0			

LEAST MEAN SQUARE INDICATOR = 9.452

EFFECTIVE EDDY DIFFUSIVITIES AND SWIRL VELOCITY RATIO
 0.072 0.072 0.072 0.072 0.375

Figure 6-5

REFERENCES

1. Khan, E., Rohsenow, W., Sonin, A., Todreas, N., "A Porous Body Model for Predicting Temperature Distributions in Wire Wrapped Fuel and Blanket Assemblies of a LMFBR," Topical Report #COO-2245-16TR, MIT, June 1975.
2. Novendstern, E., "Turbulent Flow Pressure Drop Model for Fuel Rod Assemblies Utilizing a Helical Wire Wrap Spacer System," Nuclear Engineering and Design, Vol. 22, No.1, August 1972.
3. Chen, B., Todreas, N., "Prediction of Coolant Temperature Field in a Breeder REactor Including Interassembly Heat Transfer," Topical Report #COO-2245-20TR, MIT, May 1975.
4. Chiu, C., Todreas, N., "Mixing Experiment of a LMFBR Blanket Assembly, to be issued as a topical report #COO-2245-45TR, Nuclear Engineering Department, MIT.

APPENDIX I

LISTING OF REAL TIME DATA ACQUISITION SYSTEM SOFTWARE

MAINLINES

USER=TRAC 055 34170 JOINT COMPUTER FACILITY, MIT PAGE 1

```

C THE WHOLE PROGRAM IS SEPERATED INTO SEVERAL MAINLINES
C THIS IS THE "MAINLINE('GLSINIT') WHICH INITIALIZE THE EXECUTION
C OF THE WHOLE PROGRAM
C ***

```

```

I EXPLICIT INTEGER*2 (I-N)
INTEGER*2 PERIOD
REAL LBOUND, IT, INPUT(10)
LOGICAL EQUAL
COMMON      ITH, ITH, IA, IN, NS, IIS, NP, IERPOP, IDATE, ITIME, ID,
             ISEFER, IPTH, NR, NT, HBOUND, LBOUND, IPTSET, IPTCAL,
             JMON, JDAY, JYR, JHR, JMIN, JSEC, JMLSEC, JDATE,
             PERIOD, IDUPA, IOLD, IMON, IDAY, IYR, IHR, IMIN, ISEC, IMLSEC,
             INTEXP, IBFLOW, IBRE, BELOW, TEMPIH, ICTEMP, TEMPIC,
             AREA, DE, APL, IASK, IN, IT,
             NSET, NCAL, NYIN, NUM, ICUIT,
             IAXSIT,
             NSET(2, 24), CHAN(2, 512), INTID,
             I-UTR(512), IT(10, 2), LEVELS(100), IMP(5), ICAL(5),
             I-UTR(512), I-UTR(512),
             TRUE(24), I-UTR(512), NAME(5), ICOND(2),
             TON, TOR, IECIO,
             I-UTR, IPT,
             ICOND, ISUC, NAVG, ELEFLOW, IERCH, IPPOL, IJFLOW, NAVG,
             NL, NLEVEL, ICID, ICCON,
             NAINS(60), INTIME, ICECAL

```

C E N D O F F I C E

PIASTON 111531(60)

22 E. 17

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
84

```

C=0
DO I=1(180-TEMP),17,1,180)
  T=(TIR*(I+.5)-.5)/9.0

```

```

11  =
12  CALL 'DAS-ADRS',11,2,179)
13  (11,1,1,0) 10 TO 9,0

```

```

C 11 2-1('ULC-1'F) 1,12,3,1EF)
C 11 2-1('ULC-1'F) 1,12,3,1EF)

```

$$D = \frac{1}{10} \sum_{j=1,5}^5 \left(1(1,64,0,177.7), 2(100,12,0,177.7), 3(100,64,0,177.7) \right)$$
$$- \quad (I) = \quad \text{IN.}^-(J)$$

— — — — —

USER=TUNG 255 44170 JOINT COMPUTER FACILITY, MIT PAGE 2

```

01900      CALL GETCHN(MAINS,15)
01950      C GO TO ROUTINE GLSMIN WHICH CONTAINS THE COMMAND PROCESSOR
01960      ICND=1
01980      1000 DO 2 I=1,2
01990      DO 2 J=1,24
02000      2 MSET(I,J)=0
02010      DO 3 I=1,2
02020      DO 3 J=1,512
02030      3 MCHAN(I,J)=0
02420
02425      C SPECIFY THE INPUT, OUTPUT DEVICES LOGIC UNIT #S
02430      ICN=6
02440      IPR=6
02450      ICHD=6
02500
02505      C INDICATE TOTAL #S OF HEADER BLOCKS AND DATA BLOCKS
02510      HTA=140
02520      ITH=800
02530      MAXSET=2
02720
02725      1 FORMAT(I5)
02730      C READ HIS FROM HEADER FILE OR,IP FOR HEADER FILE, IA FOR DATA FILE
02740      READ(2:1) ISUPA
02750      IH=ISUPA(1)
02760      IA=ISUPA(2)
02770      MSET=ISUPA(3)
02780      NCAL=ISUPA(4)
02790      NTH=ISUPA(5)
02800
02805      C INITIALIZATION OF VARIABLES IN CAL AND MIX BLOCKS
02810      TEMPIC=0.
02820      ICDB=0
02830      IPRDH=0
02840      IPRDL=0
02850      TEMPIN=0.
02860      INCON=0
02870      SFLON=0.
02880      SFLIN=0.
02890      ISUB=0
02900      NLEVEL=0
02910      NAVE=1
02920      NAVE=1
02930
02935      C REQ. DATE AND TIME
02940      C REQ. DATE IS DONE BY CALLING SUB INDATE, 'IASK' IS SFT (=1) AND
02950      C PASSED TO SUB INDATE TO INDICATE THAT SUB INDATE IS CALLED TO GET A
02960      C NEW DATE IN
02970      IASK=1
02980      CALL INDATE
02990
03000      C *****
03010      F12=0.0, 'TIME(12,IN):'
03020      F12= (IT,2012)IPR,IPI
03030      F222=0.0, '(215)

```

USER=TUNG 255 84170 JOINT COMPUTER FACILITY, MIT PAGE 3

```

039A8      CALL SETTIM(IHR,IMIN)
03A48      ITIME=IHR*256+IMIN
03B48      C *****
03B48      CALL LINKER(MAINS,1)
03B88      900  WRITE(6 ,905)IER
03D48      WRITE(5 ,905)IER
03E68      905  FORMAT(5X,I5,'FILE ERROR')
04108      STOP
04168      END
1EC5[C]  //      0000[V]  ITA      0002[V]  ITH      0004[V]  IA      0006[V]  IH
0008[V]  NS      0008[V]  IIS      000C[V]  NP      000E[V]  IERROR  0010[V]  IDATE
0012[V]  ITIME    0014[V]  ID      0016[V]  IREFER  0018[V]  IPTH    001A[V]  NR
001C[V]  NT      001E[V]  HBOUND  0022[V]  LBOUND  0026[V]  IPTSET  0028[V]  IPTCAL
002A[V]  JMCN    002C[V]  JDAY    002E[V]  JYR      0030[V]  JHR      0032[V]  JMIN
0034[V]  JSEC    0036[V]  JALSEC  0038[V]  JDATE    003A[V]  PERIOD  003C[V]  IDUPA
003E[V]  IOLD    0040[V]  IACN    0042[V]  IDAY    0044[V]  IYR      0046[V]  IHS
0048[V]  IMIR    004A[V]  ISEC    004C[V]  IMLSEC  004E[V]  INTMP    0050[V]  IBFLOW
0052[V]  IERS    0054[V]  IFLOW   0056[V]  TTEMP    005C[V]  ICTEMP  005E[V]  TEMPIC
0062[V]  AREA    0064[V]  DE      006A[V]  AXEL    006E[V]  IASK    0070[V]  IV
0072[V]  IT      0076[V]  NSET    0078[V]  NCAL    007A[V]  NMIX    007C[V]  NUX
007E[V]  IOUTI   0080[V]  NYSSET  0082[V]  NSET    0084[V]  MCHAN  0086[V]  IHELP
0088[V]  IRUSD   008A[V]  IP      008C[V]  LEVELS  008E[V]  IMIX    0090[V]  ICAL
0092[V]  IBUPC   11E8[V]  BUFB    11E8[V]  IRUPA    1A12[V]  IRUPB    1E18[V]  NAME
1E22[V]  ICODD   1E26[V]  ICN     1E28[V]  IER     1E2A[V]  IECHO   1E2C[V]  IRMAIN
1E2E[V]  IENT    1E30[V]  IACONV  1E32[V]  ISUP    1E34[V]  NAVG    1E36[V]  SLELOW
1E38[V]  IPRCH   1E3C[V]  IPRCL   1E3E[V]  IUFLOW  1E40[V]  MAVG    1E42[V]  NL
1E44[V]  NLEVEL  1E46[V]  ICHD    1E48[V]  ICCON    1E4A[V]  MAINS   1E4C[V]  INTIME
1E4E[V]  ICRCAL  0000[S]  .U      0418[V]  INPUT    0442[V]  EQUILL  0002[V]  MAINS1
0444[V]  IE      0000[S]  DFI     044C[V]  IEP      03EE[L]  900    0000[S]  2G
0452[V]  ITXT    0454[V]  IRD     0456[V]  IDAT    0174[L]  10     0458[V]  J
0000[S]  SATCHN  11A8[L]  1000    01BE[L]  2      045C[V]  I      0218[L]  3
0272[L]  1      045E[S]  SETTIM  0000[S]  BK      038F[L]  2222   0000[S]  3I
0000[S]  SETTIM  0000[S]  LINKER  03F6[L]  905    0000[S]  .S    0000[S]  .V

```

THIS PROGRAM WILL OCCUPY X'046A' BYTES OF STORAGE SPACED

PROGRAM *MAIN* HAS NO ERRORS

// FOR

2

JELP=TUNG 255 3417 JOINT COMPUTER FACILITY, MIT PAGE 1

```

0004- C C MAIN PROGRAM CONTAINS THE COMMAND PROCESSOR,
0004- C DIFFERENT ROUTINES WILL BE CALLED FOR DIFFERENT COMMANDS
0004- C COMMANDS ARE:
0004- I SELECT INTEGER*2 (I-V)
0004- I INTEGER*2 FREQ
0004- K ALL LOCUS,IT,INVT(10)
0004- LOGICAL EQUAL
0004- DO ... TFL,ILH,IA,IN,NS,IIS,NP,ISPROP,IDATE,ITIME,IO,
0004- I IREFR,IPTH,VR,NT,ROUND,LBCUND,IPSET,IPCAL,
0004- I JION,JDAY,JYP,JHR,JMIN,JSEC,JMLSEC,JDATE,
0004- I PERIOD,IPUR4,IOLD,IMON,IDAY,IYR,IHR,IYIN,ISFC,IMLSEC,
0004- I ITEMP,IFLOW,ISSE,IFLOW,TEMPIN,ICTEMP,TEMPIC,
0004- I AREA,DE,AMPL,IASK,IV,IT,
0004- I VSET,CAL,NAT,NUL,IQUIT,
0004- I AXSET,
0004- I VSET(2,24),VCNAN(2,512),IHELP,
0004- I IREFR(512),IT(11,2),LEVELS(100),IYIN(5),ICAL(5),
0004- I IREFR(512),IREFR(512),
0004- I IREFR(24),IREFR(512),IREFR(5),ICORR(2),
0004- I IREFR,IREFR,
0004- I IREFR,IPNT,
0004- I IREFR,ISUP,NAVG,SLEFOW,TEROH,IPROL,IJELOH,NAVG,
0004- I IREFR,LEVEL,ICOR,ICOR,
0004- I IREFR(60),IREFR,ICOR
0004- C END OF COMMANDS FIELD
0004- READ CDS(1)
0004- C-IF CDS/'SET','CAL','IYIN','CNAN','EDT','END'/
0020- DO I=1,N
0022-
0024- CALL FIELD 1 (2,64,0,ITEXT)
0026- CALL FIELD 2 (100,12,0,IREF)
0040- CALL FIELD 3 (100,64,0,IPAT)
0050-
0052- CALL FIELD (100,50,65,75,45,95),ICOR
0070- 1000 CONTINUE
0072-
0074- C REF. NUMBER 2, WHICH IS USED TO COMPUTE BUNDLE PE LATER IS SUB-
0076- IREF = .
0078- IREF = .
0080- IREF = .
0082- IREF = .
0084- IREF = .
0086- IREF = .
0088- IREF = .
0090- IREF = .
0092- IREF = .
0094- IREF = .
0096- IREF = .
0098- IREF = .
0100- IREF = .
0102- IREF = .
0104- IREF = .
0106- IREF = .
0108- IREF = .
0110- IREF = .
0112- IREF = .
0114- IREF = .
0116- IREF = .
0118- IREF = .
0120- IREF = .
0122- IREF = .
0124- IREF = .
0126- IREF = .
0128- IREF = .
0130- IREF = .
0132- IREF = .
0134- IREF = .
0136- IREF = .
0138- IREF = .
0140- IREF = .
0142- IREF = .
0144- IREF = .
0146- IREF = .
0148- IREF = .
0150- IREF = .
0152- IREF = .
0154- IREF = .
0156- IREF = .
0158- IREF = .
0160- IREF = .
0162- IREF = .
0164- IREF = .
0166- IREF = .
0168- IREF = .
0170- IREF = .
0172- IREF = .
0174- IREF = .
0176- IREF = .
0178- IREF = .
0180- IREF = .
0182- IREF = .
0184- IREF = .
0186- IREF = .
0188- IREF = .
0190- IREF = .
0192- IREF = .
0194- IREF = .
0196- IREF = .
0198- IREF = .
0200- IREF = .

```

USFR=TJN, 255 14170 JOINT COMPUTER FACILITY, MIT PAGE 2

```

0130:      CALL RETURN('REQ. DU DLE ID:',APR,C.,32767.)
0160:      IF( .S.DL.C.) GO TO 20
0172:      IL=100
0180:
0180:      C COMMAND LEVEL:
0190:      35 I UNIT=1
0195:      PRT 5,5,' >'
01A0:      READ(ICV,40) ICVD
01A5:      40 FORMAT(2'D)
01B5:
01C0:      C PROCESS COMMAND, TAKE THE RIGHT ACTION
01C5:      35 43 I=1,7
01D0:      IF(EQUAL(CPOS(I),0,ICVD,0,3)) GO TO 431
01E5:      43 CONTINUE
020:
020:      C INV LINE NO
021:      IL=IL+1
022:      45 IF( .S.DL.C.) GO TO 431
023:      45 IF( .S.DL.C.) GO TO 431
024:
025:
026:
027:
027:      C GO TO LINE 'GLESST' FOR COMMAND 'SET'
027:      50 I UNIT=1
028:      CALL LINEAR(MINS,2)
029:      55 IF( .S.DL.C.) GO TO 431
02A:
02A:      C GO TO LINE 'GLESST' FOR COMMAND 'CAL'
02A:      60 I UNIT=1
02B:      CALL LINEAR(MINS,3)
02C:      65 IF( .S.DL.C.) GO TO 431
02D:
02D:      C GO TO LINE 'GLESST' FOR COMMAND 'CAL'
02D:      70 I UNIT=1
02E:      CALL LINEAR(MINS,4)
02F:      75 IF( .S.DL.C.) GO TO 431
030:
030:      C GO TO LINE 'GLESST' FOR COMMAND 'CAL'
030:      80 I UNIT=1
031:      CALL LINEAR(MINS,5)
032:      85 IF( .S.DL.C.) GO TO 431
033:
033:      C GO TO LINE 'GLESST' FOR COMMAND 'CAL'
033:      90 I UNIT=1
034:      CALL LINEAR(MINS,6)
035:
035:      C GO TO LINE 'GLESST' FOR COMMAND 'CAL'
035:      95 I UNIT=1
036:      CALL LINEAR(MINS,7)
037:
037:
038:
039:

```

11/11/71 // [1] 10 01 0111 1111 00 4111 1- 0001111 11
00/11/ [1] 11 00 0111 1111 0001111 1111 0010111 101111

UCLF=TO... 65 4170 JOINT COMPUTER FACILITY, MIT PAGE 3

0012 [V]	ITIME	0014 [V]	IT	0016 [V]	ITREFR	0018 [V]	IPTH	0019 [V]	VR
0013 [V]	IT	0015 [V]	ITCONF	0017 [V]	LROUND	0019 [V]	IPRSET	0020 [V]	IPFCAL
002 [V]	ITON	0017 [V]	JDR	0021 [V]	JYF	0020 [V]	JHF	0022 [V]	JMT
0034 [V]	JSLS	0023 [V]	JLCLD	0022 [V]	JD TL	0031 [V]	PERIOD	0030 [V]	IDUR
003 [V]	LOT	0024 [V]	I	0040 [V]	ID Y	0044 [V]	IYF	0036 [V]	IYF
004 [V]	LINE	0024 [V]	ISFC	0040 [V]	IMISEC	0045 [V]	IMTEMP	0050 [V]	IRFLOW
0052 [V]	ISBE	0054 [V]	ISFLD	0050 [V]	TEAMIN	0050 [V]	ICTEMP	0059 [V]	TECPIC
0062 [V]	ISFLD	0055 [V]	IS	006 [V]	AMEL	0060 [V]	IASV	0070 [V]	IV
0072 [V]	IT	0075 [V]	ISLT	007 [V]	NCAL	0071 [V]	MMIV	0070 [V]	NUM
0075 [V]	ISQUIT	0080 [V]	ISSET	0082 [V]	XSIT	0072 [V]	MSHAN	0072 [V]	INTLP
0084 [V]	ISLFD	0084 [V]	IS	0080 [V]	LEVELS	0084 [V]	IMIX	0085 [V]	ICAL
0085 [V]	ISLFC	1120 [V]	ISFB	1924 [V]	IBUFA	1A19 [V]	IBUFB	1E18 [V]	NAME
1020 [V]	ISOLD	1126 [V]	IS	1524 [V]	ITP	1E21 [V]	TECHO	1E20 [V]	IRMMV
1E21 [V]	ISLT	113 [V]	ISCON	1130 [V]	ISU	1E34 [V]	NAV	1E36 [V]	SLFLOW
1E3 [V]	ITFCH	1130 [V]	ISCOL	1E37 [V]	ISLLO	1E40 [V]	NAV	1E12 [V]	NL
1E4 [V]	ISLVEL	114 [V]	ISL	1E45 [V]	ISCON	1E42 [V]	MAINS	1E62 [V]	IMTIME
1E6 [V]	ISLVEL	114 [V]	ISL	0355 [V]	IMELT	0000 [S]	EQUAI	0008 [V]	CMFC
0020 [V]	ITIME	114 [V]	ISL	0356 [V]	IMAT	030 [V]	ITFCH	0304 [V]	ITIME
007 [L]	10	115 [L]	IS	0000 [L]	65	0308 [L]	75	0301 [L]	35
0080 [L]	95	0000 [L]	IS	0000 [L]	10	0302 [V]	150	0000 [S]	GETNUM
0080 [L]	10	010 [L]	2	0000 [L]	1	0000 [L]	1	0100 [L]	35
0000 [L]	10	010 [L]	4	0000 [L]	10	0100 [L]	43	0304 [V]	I
0251 [L]	431	020 [L]	45	027 [L]	50	0200 [L]	60	0256 [L]	70
0334 [L]	00	033 [L]	1	0260 [L]	110				

***** SLIP = 0000 T = 0000

```
0000[1] SINGLE CO [1] CHANGE 0000[1] .B      0000[1] .V  
    FILE PROGRAM WITH CONTROL POINTS OF STORAGE STACK  
//
```

JDE VI ORIGINATOR: AGENCY OF VISION 202 6/13/76 GENPATID 12/17/76 03:

DATE	DESCRIPTION	AMOUNT	CHECK NO.	BANK	BALANCE
JUL 1	SALES	1.35	1	1	1.35
JUL 2	SALES	1.35	2	1	2.70
JUL 3	SALES	1.50	3	1	4.20
JUL 4	SALES	1.25	4	1	5.45
JUL 5	SALES	1.25	5	1	6.70
JUL 6	SALES	1.25	6	1	7.95
JUL 7	SALES	1.25	7	1	9.20
JUL 8	SALES	1.25	8	1	10.45
JUL 9	SALES	1.25	9	1	11.70
JUL 10	SALES	1.25	10	1	12.95
JUL 11	SALES	1.25	11	1	14.20
JUL 12	SALES	1.25	12	1	15.45
JUL 13	SALES	1.25	13	1	16.70
JUL 14	SALES	1.25	14	1	17.95
JUL 15	SALES	1.25	15	1	19.20
JUL 16	SALES	1.25	16	1	20.45
JUL 17	SALES	1.25	17	1	21.70
JUL 18	SALES	1.25	18	1	22.95
JUL 19	SALES	1.25	19	1	24.20
JUL 20	SALES	1.25	20	1	25.45
JUL 21	SALES	1.25	21	1	26.70
JUL 22	SALES	1.25	22	1	27.95
JUL 23	SALES	1.25	23	1	29.20
JUL 24	SALES	1.25	24	1	30.45
JUL 25	SALES	1.25	25	1	31.70
JUL 26	SALES	1.25	26	1	32.95
JUL 27	SALES	1.25	27	1	34.20
JUL 28	SALES	1.25	28	1	35.45
JUL 29	SALES	1.25	29	1	36.70
JUL 30	SALES	1.25	30	1	37.95
JUL 31	SALES	1.25	31	1	39.20
JUL 32	SALES	1.25	32	1	40.45
JUL 33	SALES	1.25	33	1	41.70
JUL 34	SALES	1.25	34	1	42.95
JUL 35	SALES	1.25	35	1	44.20
JUL 36	SALES	1.25	36	1	45.45
JUL 37	SALES	1.25	37	1	46.70
JUL 38	SALES	1.25	38	1	47.95
JUL 39	SALES	1.25	39	1	49.20
JUL 40	SALES	1.25	40	1	50.45
JUL 41	SALES	1.25	41	1	51.70
JUL 42	SALES	1.25	42	1	52.95
JUL 43	SALES	1.25	43	1	54.20
JUL 44	SALES	1.25	44	1	55.45
JUL 45	SALES	1.25	45	1	56.70
JUL 46	SALES	1.25	46	1	57.95
JUL 47	SALES	1.25	47	1	59.20
JUL 48	SALES	1.25	48	1	60.45
JUL 49	SALES	1.25	49	1	61.70
JUL 50	SALES	1.25	50	1	62.95
JUL 51	SALES	1.25	51	1	64.20
JUL 52	SALES	1.25	52	1	65.45
JUL 53	SALES	1.25	53	1	66.70
JUL 54	SALES	1.25	54	1	67.95
JUL 55	SALES	1.25	55	1	69.20
JUL 56	SALES	1.25	56	1	70.45
JUL 57	SALES	1.25	57	1	71.70
JUL 58	SALES	1.25	58	1	72.95
JUL 59	SALES	1.25	59	1	74.20
JUL 60	SALES	1.25	60	1	75.45
JUL 61	SALES	1.25	61	1	76.70
JUL 62	SALES	1.25	62	1	77.95
JUL 63	SALES	1.25	63	1	79.20
JUL 64	SALES	1.25	64	1	80.45
JUL 65	SALES	1.25	65	1	81.70
JUL 66	SALES	1.25	66	1	82.95
JUL 67	SALES	1.25			

11. 205 417, 1957 11/22/57 5:47. 32.52 JEFF INTER 122 LOTS.

USER=TW 255 04170 JOINT COMPUTER FACILITY, MIT PAGE 1

```

0004 0 MAIN LINE 'GLOBAL' PROCESSES COMMAND 'SET'
0004 0 COMMAND
0004 1 'MIDIF INTEG.*2 (T-M)'
0004 1 'G.L.*2' 'FICD'
0004 1 'L L L OUL,IT,INPUT(10)'
0004 1 LOGICAL & UAL
0004 1 'MIDIF' ITA,ITH,IT,IF,NS,IIS,VP,ITPROP,IDATE,ITIMP,IO,
0004 1 IPREFI,IPTH,IP,NT,HBOUND,LBOUND, IPTSET,IPTCAL,
0004 1 JON,JDAY,JYP,JHP,JMIN,JSEC,JMLSEC, JDATE,
0004 1 PERIOD,IDUR,IOLE,IMON,IDAY,IYP,IHP,IYIN,ISEC,IMLSEC,
0004 1 INTIMP,IFLOW,IRPE,IFLOW,TEMPIM,ICTEMP,TEMPIC,
0004 1 REA,DE,AMPL,ASK,IN,IT,
0004 1 'SET,ICAL,MIX,MUX,IQUIT,
0004 1 'ACST,
0004 1 'MST(2,24),MCHN(2,512),IMFLP,
0004 1 LVEL(512),IF(10,2),LVLIS(100),IMF(5),ICL(5),
0004 1 LEL(512),MPP(512),
0004 1 LUF(24),LUF(512),MPL(5),ICMD(0),
0004 1 ICN,IEF,TECH,
0004 1 TCM,TM,TM,
0004 1 LCON,ISV,VS,CLEFOW,IFLOW,IRPOL,IUFLOW,IRVS,
0004 1 'L,LVVEL,ICMD,ICCON,
0004 1 WINS(60),INTIMP,ICPCNT
0004 0 END OF COMMAND FILE
0004 0 END OF
0004 0 END OF (MINS,1)

```

100 [V] //	100 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
000 [V] IT	000 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
001 [V] IT	001 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
001 [V] IT	001 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
002 [V] IT	002 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
003 [V] IT	003 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
004 [V] IT	004 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
005 [V] IT	005 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
006 [V] IT	006 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
007 [V] IT	007 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
008 [V] IT	008 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
009 [V] IT	009 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
010 [V] IT	010 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
011 [V] IT	011 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
012 [V] IT	012 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
013 [V] IT	013 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
014 [V] IT	014 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
015 [V] IT	015 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
016 [V] IT	016 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
017 [V] IT	017 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
018 [V] IT	018 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
019 [V] IT	019 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
020 [V] IT	020 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
021 [V] IT	021 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
022 [V] IT	022 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
023 [V] IT	023 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
024 [V] IT	024 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
025 [V] IT	025 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
026 [V] IT	026 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
027 [V] IT	027 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
028 [V] IT	028 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
029 [V] IT	029 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
030 [V] IT	030 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
031 [V] IT	031 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
032 [V] IT	032 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
033 [V] IT	033 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
034 [V] IT	034 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
035 [V] IT	035 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
036 [V] IT	036 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
037 [V] IT	037 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
038 [V] IT	038 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
039 [V] IT	039 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
040 [V] IT	040 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
041 [V] IT	041 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
042 [V] IT	042 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
043 [V] IT	043 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
044 [V] IT	044 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
045 [V] IT	045 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
046 [V] IT	046 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
047 [V] IT	047 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
048 [V] IT	048 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
049 [V] IT	049 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
050 [V] IT	050 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
051 [V] IT	051 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
052 [V] IT	052 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
053 [V] IT	053 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
054 [V] IT	054 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
055 [V] IT	055 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
056 [V] IT	056 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
057 [V] IT	057 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
058 [V] IT	058 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
059 [V] IT	059 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
060 [V] IT	060 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
061 [V] IT	061 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
062 [V] IT	062 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
063 [V] IT	063 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
064 [V] IT	064 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
065 [V] IT	065 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
066 [V] IT	066 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
067 [V] IT	067 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
068 [V] IT	068 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
069 [V] IT	069 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
070 [V] IT	070 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
071 [V] IT	071 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
072 [V] IT	072 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
073 [V] IT	073 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
074 [V] IT	074 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
075 [V] IT	075 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
076 [V] IT	076 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
077 [V] IT	077 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
078 [V] IT	078 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
079 [V] IT	079 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
080 [V] IT	080 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
081 [V] IT	081 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
082 [V] IT	082 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
083 [V] IT	083 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
084 [V] IT	084 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
085 [V] IT	085 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
086 [V] IT	086 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
087 [V] IT	087 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
088 [V] IT	088 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
089 [V] IT	089 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
090 [V] IT	090 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
091 [V] IT	091 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
092 [V] IT	092 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
093 [V] IT	093 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
094 [V] IT	094 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
095 [V] IT	095 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
096 [V] IT	096 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
097 [V] IT	097 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
098 [V] IT	098 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
099 [V] IT	099 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT
100 [V] IT	100 [V] IT	0004 [V] IT	0004 [V] IT	0004 [V] IT

NOTES OF STORAGE SPACES

2 USFR=TL 00 4170 CONT COMPUTER FACILITY, "IT PIG" 1

[illegible]

0000	0001	0002	0003	0004	0005	0006	0007	0008	0009	0010	0011	0012	0013	0014	0015	0016	0017	0018	0019	0020	0021	0022	0023	0024	0025	0026	0027	0028	0029	0030	0031	0032	0033	0034	0035	0036	0037	0038	0039	0040	0041	0042	0043	0044	0045	0046	0047	0048	0049	0050	0051	0052	0053	0054	0055	0056	0057	0058	0059	0060	0061	0062	0063	0064	0065	0066	0067	0068	0069	0070	0071	0072	0073	0074	0075	0076	0077	0078	0079	0080	0081	0082	0083	0084	0085	0086	0087	0088	0089	0090	0091	0092	0093	0094	0095	0096	0097	0098	0099
0000	0001	0002	0003	0004	0005	0006	0007	0008	0009	0010	0011	0012	0013	0014	0015	0016	0017	0018	0019	0020	0021	0022	0023	0024	0025	0026	0027	0028	0029	0030	0031	0032	0033	0034	0035	0036	0037	0038	0039	0040	0041	0042	0043	0044	0045	0046	0047	0048	0049	0050	0051	0052	0053	0054	0055	0056	0057	0058	0059	0060	0061	0062	0063	0064	0065	0066	0067	0068	0069	0070	0071	0072	0073	0074	0075	0076	0077	0078	0079	0080	0081	0082	0083	0084	0085	0086	0087	0088	0089	0090	0091	0092	0093	0094	0095	0096	0097	0098	0099

THE UNIVERSITY OF CHICAGO

11.

PROSPERITY *MAIN* MIS NO TERRORS
FOR

USFR=TL 8417 JOEL F COMPUTER FACILITY, WIT PAGE 1

C (AI IL) *OLS DS * CALLS OVER HEADER, DATAP, SAVIF

C COM

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

C *LIST I PRG *D (I-1)

1500 [V] //	[V]	0002 [V] ITH	0004 [V] I*	0005 [V] IV
0001 [V] VS	[V]	0007 [V] NP	0008 [V] IERROR	0010 [V] IDATE
0012 [V] ITIME	[V]	0016 [V] IREFLO	0013 [V] ITH	0014 [V] IP
0015 [V] .T	[V]	0020 [V] LBOUND	0026 [V] IPTST	0028 [V] IPTCAL
0021 [V] JCO	[V]	0025 [V] JYP	0030 [V] JMP	0032 [V] INT
0024 [V] JLOC	[V]	0027 [V] JDATE	0031 [V] PERIOD	0037 [V] ID
0028 [V] IOL	[V]	0038 [V] I*	0041 [V] IYD	0046 [V] IYD
0031 [V] ITH	[V]	0043 [V] IALCFC	0044 [V] INTIMP	0050 [V] IREFLO
0035 [V] I	[V]	0045 [V] TEMPI	0047 [V] ICTIMP	0057 [V] TEMPI
0038 [V] ITH	[V]	0046 [V] AMPL	0068 [V] IASV	0070 [V] IV
0071 [V] IT	[V]	0070 [V] IOL	0071 [V] INT	0070 [V] NUM
0071 [V] ITH	[V]	0072 [V] MSET	0072 [V] MCHN	0812 [V] ITHLP
0074 [V] T	[V]	0073 [V] LEVLS	0074 [V] IMI	0075 [V] IOL
0075 [V] IOL	[V]	0076 [V] ITH	0076 [V] ITH	0076 [V] ITH
1E22 [V] IOL	[V]	1E27 [V] ITH	1E28 [V] ITH	1E28 [V] ITH
1E21 [V] ITH	[V]	1E29 [V] ITH	1E30 [V] ITH	1E30 [V] ITH
1E31 [V] ITH	[V]	1E31 [V] ITH	1E32 [V] ITH	1E32 [V] ITH
1E32 [V] ITH	[V]	1E33 [V] ITH	1E34 [V] ITH	1E34 [V] ITH
1E33 [V] ITH	[V]	1E35 [V] ITH	1E36 [V] ITH	1E36 [V] ITH
1E34 [V] ITH	[V]	1E37 [V] ITH	1E38 [V] ITH	1E38 [V] ITH
1E35 [V] ITH	[V]	1E39 [V] ITH	1E40 [V] ITH	1E40 [V] ITH
1E36 [V] ITH	[V]	1E41 [V] ITH	1E42 [V] ITH	1E42 [V] ITH
1E37 [V] ITH	[V]	1E43 [V] ITH	1E44 [V] ITH	1E44 [V] ITH
1E38 [V] ITH	[V]	1E45 [V] ITH	1E46 [V] ITH	1E46 [V] ITH
1E39 [V] ITH	[V]	1E47 [V] ITH	1E48 [V] ITH	1E48 [V] ITH
1E40 [V] ITH	[V]	1E49 [V] ITH	1E50 [V] ITH	1E50 [V] ITH
1E41 [V] ITH	[V]	1E51 [V] ITH	1E52 [V] ITH	1E52 [V] ITH
1E42 [V] ITH	[V]	1E53 [V] ITH	1E54 [V] ITH	1E54 [V] ITH
1E43 [V] ITH	[V]	1E55 [V] ITH	1E56 [V] ITH	1E56 [V] ITH
1E44 [V] ITH	[V]	1E57 [V] ITH	1E58 [V] ITH	1E58 [V] ITH
1E45 [V] ITH	[V]	1E59 [V] ITH	1E60 [V] ITH	1E60 [V] ITH
1E46 [V] ITH	[V]	1E61 [V] ITH	1E62 [V] ITH	1E62 [V] ITH
1E47 [V] ITH	[V]	1E63 [V] ITH	1E64 [V] ITH	1E64 [V] ITH
1E48 [V] ITH	[V]	1E65 [V] ITH	1E66 [V] ITH	1E66 [V] ITH
1E49 [V] ITH	[V]	1E67 [V] ITH	1E68 [V] ITH	1E68 [V] ITH
1E50 [V] ITH	[V]	1E69 [V] ITH	1E70 [V] ITH	1E70 [V] ITH
1E51 [V] ITH	[V]	1E71 [V] ITH	1E72 [V] ITH	1E72 [V] ITH
1E52 [V] ITH	[V]	1E73 [V] ITH	1E74 [V] ITH	1E74 [V] ITH
1E53 [V] ITH	[V]	1E75 [V] ITH	1E76 [V] ITH	1E76 [V] ITH
1E54 [V] ITH	[V]	1E77 [V] ITH	1E78 [V] ITH	1E78 [V] ITH
1E55 [V] ITH	[V]	1E79 [V] ITH	1E80 [V] ITH	1E80 [V] ITH
1E56 [V] ITH	[V]	1E81 [V] ITH	1E82 [V] ITH	1E82 [V] ITH
1E57 [V] ITH	[V]	1E83 [V] ITH	1E84 [V] ITH	1E84 [V] ITH
1E58 [V] ITH	[V]	1E85 [V] ITH	1E86 [V] ITH	1E86 [V] ITH
1E59 [V] ITH	[V]	1E87 [V] ITH	1E88 [V] ITH	1E88 [V] ITH
1E60 [V] ITH	[V]	1E89 [V] ITH	1E90 [V] ITH	1E90 [V] ITH
1E61 [V] ITH	[V]	1E91 [V] ITH	1E92 [V] ITH	1E92 [V] ITH
1E62 [V] ITH	[V]	1E93 [V] ITH	1E94 [V] ITH	1E94 [V] ITH
1E63 [V] ITH	[V]	1E95 [V] ITH	1E96 [V] ITH	1E96 [V] ITH
1E64 [V] ITH	[V]	1E97 [V] ITH	1E98 [V] ITH	1E98 [V] ITH
1E65 [V] ITH	[V]	1E99 [V] ITH	1E00 [V] ITH	1E00 [V] ITH

PROGRAM **MAIN** + S NO ERRORS

FOR

CP


```

033      I=1
034      275 IF (L=LEVEL)
035      IF (SLFLOW.CT.C) .AND. SLFLOW.LE.32767.) GO TO 276
036      WRITE(IPR,155) SLFLOW
037      IPR=1
038      276 IF (ISUB.GT.1 .AND. ISUB.LE.512) GO TO 2773
039      IF (IPR,165) ISUB
040      IE=1
041      2773 IF (IPSO .GT. IPROL) GO TO 2774
042      WRITE(IPR,2777) IPR04,IPROL
043      IPR=1
044      2774 IF (AVG.CT.C) GO TO 277
045      WRITE(IPR,25) "AVG
046      IPRCT=1
047      277 IF (T=TDOT.EQ.C) GO TO 278
048      GO TO 27
049      278 IF (T=TDOT.EQ.C) .AND. (VF.EQ.SFLOW) GO TO 250
050      IF (T=TDOT.EQ.C) .AND. (VF.EQ.SFLOW) GO TO 250
051      CALL LVL (T,IS,I,VI)
052      250 IF (IC.GT.2)
053      252 'A LIST INPUT CHANGE FOR SERIAL LEVELS(LEVELS)',(LEVELS)
054      CALL LVL (T,IS,I,VI)
055      290 READ(IC,202)
056      292 'A LIST INPUT CHANGE(TIME)',(TIME)
057      CALL LVL (T,IS,I,VI)

```

[illegible]

2 USER=TIM 005 4417 JOINT COMPUTER FACILITY, MIT PAGE 3

0404[1] 282 012[1] 212 0000[S] .V
 7.10 17.68M ILL COPY 1105500 BYTES OF STORAGE SPACED
 //

JCF 710 OPERATING SYSTEM VERSION 2 REVISION 222 6/13/76 GENERATED 12/17/76 03:

JCF PRINTING CHARGE	\$.35 / JOB	.35
004 LINES PRINTED PER	\$ 1.25 / K LN	.76
181 CHARACTERS RECD	\$ 1.50 / K CD	.23
00 PLOTTER REVISIONS	\$.25 / 1000	.00
33 CBL 70 SECONDS	\$25.00 / HOUR	.23
00 CBL 50 SECONDS	\$35.00 / HOUR	.00
TOTAL CHARGE		1.57

TOTAL 285 4417 1.57M JOB 005/2/62 14:52. \$ 35.77 LEFT AFTER 120 LOGINS.

GLAS-LIB

(Library)

USC&F=TR 005 84171 JOINT COMPUTER FACILITY, MIT PAGE

```

00041      SUBROUTINE SET
00042      I=10 TO 1000 THAT COLLECTS DATA FOR A SETUP HEADER BLOCK, AND IT
00043      C COLLECTS DATA BLOCKS (CHS REL = 0)
00044      C COMMON FILE
00045      I=10 TO 1000 THAT COLLECTS DATA FOR A SETUP HEADER BLOCK, AND IT
00046      C COLLECTS DATA BLOCKS (CHS REL = 0)
00047      I=10 TO 1000 THAT COLLECTS DATA FOR A SETUP HEADER BLOCK, AND IT
00048      C COLLECTS DATA BLOCKS (CHS REL = 0)
00049      I=10 TO 1000 THAT COLLECTS DATA FOR A SETUP HEADER BLOCK, AND IT
00050      C COLLECTS DATA BLOCKS (CHS REL = 0)
00051      I=10 TO 1000 THAT COLLECTS DATA FOR A SETUP HEADER BLOCK, AND IT
00052      C COLLECTS DATA BLOCKS (CHS REL = 0)
00053      I=10 TO 1000 THAT COLLECTS DATA FOR A SETUP HEADER BLOCK, AND IT
00054      C COLLECTS DATA BLOCKS (CHS REL = 0)
00055      I=10 TO 1000 THAT COLLECTS DATA FOR A SETUP HEADER BLOCK, AND IT
00056      C COLLECTS DATA BLOCKS (CHS REL = 0)
00057      I=10 TO 1000 THAT COLLECTS DATA FOR A SETUP HEADER BLOCK, AND IT
00058      C COLLECTS DATA BLOCKS (CHS REL = 0)
00059      I=10 TO 1000 THAT COLLECTS DATA FOR A SETUP HEADER BLOCK, AND IT
00060      C COLLECTS DATA BLOCKS (CHS REL = 0)
00061      I=10 TO 1000 THAT COLLECTS DATA FOR A SETUP HEADER BLOCK, AND IT
00062      C COLLECTS DATA BLOCKS (CHS REL = 0)
00063      I=10 TO 1000 THAT COLLECTS DATA FOR A SETUP HEADER BLOCK, AND IT
00064      C COLLECTS DATA BLOCKS (CHS REL = 0)
00065      I=10 TO 1000 THAT COLLECTS DATA FOR A SETUP HEADER BLOCK, AND IT
00066      C COLLECTS DATA BLOCKS (CHS REL = 0)
00067      I=10 TO 1000 THAT COLLECTS DATA FOR A SETUP HEADER BLOCK, AND IT
00068      C COLLECTS DATA BLOCKS (CHS REL = 0)
00069      I=10 TO 1000 THAT COLLECTS DATA FOR A SETUP HEADER BLOCK, AND IT
00070      C COLLECTS DATA BLOCKS (CHS REL = 0)
00071      I=10 TO 1000 THAT COLLECTS DATA FOR A SETUP HEADER BLOCK, AND IT
00072      C COLLECTS DATA BLOCKS (CHS REL = 0)
00073      I=10 TO 1000 THAT COLLECTS DATA FOR A SETUP HEADER BLOCK, AND IT
00074      C COLLECTS DATA BLOCKS (CHS REL = 0)
00075      I=10 TO 1000 THAT COLLECTS DATA FOR A SETUP HEADER BLOCK, AND IT
00076      C COLLECTS DATA BLOCKS (CHS REL = 0)
00077      I=10 TO 1000 THAT COLLECTS DATA FOR A SETUP HEADER BLOCK, AND IT
00078      C COLLECTS DATA BLOCKS (CHS REL = 0)
00079      I=10 TO 1000 THAT COLLECTS DATA FOR A SETUP HEADER BLOCK, AND IT
00080      C COLLECTS DATA BLOCKS (CHS REL = 0)
00081      I=10 TO 1000 THAT COLLECTS DATA FOR A SETUP HEADER BLOCK, AND IT
00082      C COLLECTS DATA BLOCKS (CHS REL = 0)
00083      I=10 TO 1000 THAT COLLECTS DATA FOR A SETUP HEADER BLOCK, AND IT
00084      C COLLECTS DATA BLOCKS (CHS REL = 0)
00085      I=10 TO 1000 THAT COLLECTS DATA FOR A SETUP HEADER BLOCK, AND IT
00086      C COLLECTS DATA BLOCKS (CHS REL = 0)
00087      I=10 TO 1000 THAT COLLECTS DATA FOR A SETUP HEADER BLOCK, AND IT
00088      C COLLECTS DATA BLOCKS (CHS REL = 0)
00089      I=10 TO 1000 THAT COLLECTS DATA FOR A SETUP HEADER BLOCK, AND IT
00090      C COLLECTS DATA BLOCKS (CHS REL = 0)
00091      I=10 TO 1000 THAT COLLECTS DATA FOR A SETUP HEADER BLOCK, AND IT
00092      C COLLECTS DATA BLOCKS (CHS REL = 0)
00093      I=10 TO 1000 THAT COLLECTS DATA FOR A SETUP HEADER BLOCK, AND IT
00094      C COLLECTS DATA BLOCKS (CHS REL = 0)
00095      I=10 TO 1000 THAT COLLECTS DATA FOR A SETUP HEADER BLOCK, AND IT
00096      C COLLECTS DATA BLOCKS (CHS REL = 0)
00097      I=10 TO 1000 THAT COLLECTS DATA FOR A SETUP HEADER BLOCK, AND IT
00098      C COLLECTS DATA BLOCKS (CHS REL = 0)
00099      I=10 TO 1000 THAT COLLECTS DATA FOR A SETUP HEADER BLOCK, AND IT
00100      C COLLECTS DATA BLOCKS (CHS REL = 0)

```


[illegible]

NAME	DATE	TIME	LOCATION	STATUS	REMARKS
0002	1970	12:00	0002	0002	0002
0003	1970	12:00	0003	0003	0003
0004	1970	12:00	0004	0004	0004
0005	1970	12:00	0005	0005	0005
0006	1970	12:00	0006	0006	0006
0007	1970	12:00	0007	0007	0007
0008	1970	12:00	0008	0008	0008
0009	1970	12:00	0009	0009	0009
0010	1970	12:00	0010	0010	0010
0011	1970	12:00	0011	0011	0011
0012	1970	12:00	0012	0012	0012
0013	1970	12:00	0013	0013	0013
0014	1970	12:00	0014	0014	0014
0015	1970	12:00	0015	0015	0015
0016	1970	12:00	0016	0016	0016
0017	1970	12:00	0017	0017	0017
0018	1970	12:00	0018	0018	0018
0019	1970	12:00	0019	0019	0019
0020	1970	12:00	0020	0020	0020
0021	1970	12:00	0021	0021	0021
0022	1970	12:00	0022	0022	0022
0023	1970	12:00	0023	0023	0023
0024	1970	12:00	0024	0024	0024
0025	1970	12:00	0025	0025	0025
0026	1970	12:00	0026	0026	0026
0027	1970	12:00	0027	0027	0027
0028	1970	12:00	0028	0028	0028
0029	1970	12:00	0029	0029	0029
0030	1970	12:00	0030	0030	0030
0031	1970	12:00	0031	0031	0031
0032	1970	12:00	0032	0032	0032
0033	1970	12:00	0033	0033	0033
0034	1970	12:00	0034	0034	0034
0035	1970	12:00	0035	0035	0035
0036	1970	12:00	0036	0036	0036
0037	1970	12:00	0037	0037	0037
0038	1970	12:00	0038	0038	0038
0039	1970	12:00	0039	0039	0039
0040	1970	12:00	0040	0040	0040
0041	1970	12:00	0041	0041	0041
0042	1970	12:00	0042	0042	0042
0043	1970	12:00	0043	0043	0043
0044	1970	12:00	0044	0044	0044
0045	1970	12:00	0045	0045	0045
0046	1970	12:00	0046	0046	0046
0047	1970	12:00	0047	0047	0047
0048	1970	12:00	0048	0048	0048
0049	1970	12:00	0049	0049	0049
0050	1970	12:00	0050	0050	0050
0051	1970	12:00	0051	0051	0051
0052	1970	12:00	0052	0052	0052
0053	1970	12:00	0053	0053	0053
0054	1970	12:00	0054	0054	0054
0055	1970	12:00	0055	0055	0055
0056	1970	12:00	0056	0056	0056
0057	1970	12:00	0057	0057	0057
0058	1970	12:00	0058	0058	0058
0059	1970	12:00	0059	0059	0059
0060	1970	12:00	0060	0060	0060
0061	19				

0018=1000 45 470 480 490 500 510 520 530 540 550 560 570 580 590 600 610 620 630 640 650 660 670 680 690 700 710 720 730 740 750 760 770 780 790 800 810 820 830 840 850 860 870 880 890 900 910 920 930 940 950 960 970 980 990

JOINT COMBUSTION FACTORY, WIT PAGE 4

USER=TUNG 055 8417 JOINT COMPUTER FACILITY, WIT PAGE 3

```

0362- I=(100.0-1.) RETURN
0363- IPRBL=ARG
0364- IF(IPRBL.GT.IPRBL) GO TO 100
0365- WRITE(IPRBL,80) IPRBL,IPRBL
0410 35 PRINT(2,' ERROR: HIGHTS COUNT', I5, '<= LOWER BOUND', I5)
0440 GO TO 20

```

```

0441 C REQ. WATER TEMP
0442 100 ARG=TEMP
0452- ALOW=0.
0453- AHIGH=37.3
0462- CALL GETNUM(' REQ. WATER TEMP',ARG,ALOW,AHIGH)
0463- IF( ARG.LT.0.) RETURN
0464- TEMP=ARG
0465- ITEMP=TEMP*10.

```

```

0466 C R-2. INJECTION CONCENTR FIC :>
0467- ARG=IPRBL
0468- ALOW=0.
0469- AHIGH=32767.
0470- CALL GETNUM(' R-2. INJ CON (PPM)',ARG,ALOW,AHIGH)
0471- IF( ARG.LT.0.) RETURN
0472- IICCN=ARG

```

```

0473 C R-3. INJECTION FLOWRATE BFT 0 AND 2000
0474- ARG=CLFLO
0475- ALOW=0.
0476- AHIGH=32767.
0477- CALL GETNUM(' R-3. INJ FLO (L/HR)',ARG,ALOW,AHIGH)
0478- IF( ARG.LT.0.) RETURN
0479- CLFLO=ARG

```

```

0480 C R-4. INJ OIL FLOWRATE BFT 0 AND 300
0481- ARG=IFLO
0482- ALOW=0.
0483- AHIGH=300.
0484- CALL GETNUM(' R-4. INJ FLO (BFT)',ARG,ALOW,AHIGH)
0485- IF( ARG.LT.0.) RETURN
0486- IFLO=ARG

```

```

0487 C COMPUTE OIL BE A FUNCTION OF BUNDLE NO, CROSS-SEC AREA, WATER TEMP
0488 C AND OIL FLOW RATE
0489 140 CALL OILB

```

```

0490 C REQ. INJECTION SURFACE VEL, BFT 1 AND 512
0491- ARG=IPRBL
0492- ALOW=1.
0493- AHIGH=512.
0494- CALL GETNUM(' REQ. INJ SURF VEL (BFT)',ARG,ALOW,AHIGH)
0495- IF( ARG.LT.0.) RETURN
0496- ITEMP=ARG

```

```

0497 17 PRINT(1,' INJECTION SURFACE VEL (BFT) =', ITEMP)

```


USER=TUNG 255 34170 JOINT COMPUTER FACILITY, MIT PAGE 5

```

0A06R      IF(EQUAL(IANS,0,'C',0,1)) GO TO 270
0A26R      IF(EQUAL(IANS,0,'Q',0,1)) RETURN
0A41R      WRITE(IPR,185) IANS
0A56R      GO TO 252
0A6AR
0A6AR      255 PUT 6,0,' READY TO TAKE PROBES'
0A96R      PUT 6,0,' USE TT OR BUTN? T/B/Q'
0AC2R      ITT=0
0ACAR      READ(ICN,175) IANS
0AE6R      IF(EQUAL(IANS,0,'T',0,1)) GO TO 2570
0B06R      IF(EQUAL(IANS,0,'B',0,1)) GO TO 2575
0B26R      IF(EQUAL(IANS,0,'Q',0,1)) RETURN
0B4AR      WRITE(IPR,195) IANS
0B66R      GO TO 255
0B6AR
0B6AR      2570 ITT=1
0B72R      257 PUT 6,0,' ? G/Q'
0B86R      READ(ICN,175) IANS
0BAAR      IF(EQUAL(IANS,0,'G',0,1)) GO TO 258
0BCAR      IF(EQUAL(IANS,0,'Q',0,1)) RETURN
0BE6R      WRITE(IPR,185) IANS
0C0AR      GO TO 257
0C06R      2575 PUT ' BUTTON BOX IS NOT AVAILABLE,TEMP BUTTON BOX IS USED.'
0C54R      GO TO 257
0C54R
0C68R      258 NL=NL+1
0C74R      GO TO 284
0C78R
0C78R      C CHANGE CAL OR/AND SET NAME(S)
0C78R      C LET USER CHANGE SET OR/AND CAL REFERENCES NAMES UPON REQ.
0C78R      260 WRITE(IPR,265) IPTSET, IPTCAL , IORCAL
0CA4R      265 FORMAT(5X,'SET:',I2,' CAL:',I2,' CAL CORRECTION:',I2)
0CE0R      IF(IPR.NE.IECHO)WRITE(IECHO,265) IPTSET, IPTCAL , IORCAL
0D1AR      IH=0
0D22R      ICHAF=1
0D2AR      GO TO 30
0D22R
0D2AR      C LET USER CHANGE WATER TEMP,FLOWRATES,CONCENTRA,SUB CHANG UPON REQ.
0D2AR      270 IPST=1
0D36R      IMAIN=4
0D3E7      CALL LINKER(MAINS,9)
0D48R
0D48R      C NAMELIST CHANGE FOR THE AXIAL LEVELS
0D48R      280 IPST=4
0D50R      IMAIN=4
0D58R      CALL LINKER(MAINS,9)
0D62R
0D62R      C COLLECT PROBE MEASUREMENTS
0D62R      284 DO 134 I=1,NP
0D6AR      1204 IJF0(T)=0
0D6AR      C SET FREQUENCY
0D84R      CALL TT0(DEFICO)
0D84R      I=1, I=1, IJF0

```

USER=TUNG 255 84170 JOINT COMPUTER FACILITY, MIT PAGE 6

```

0091+ CALL PROBE(IBUFC,NP,IBUFB,IBUFF)
00A0+ C BUFB IS WORK AREA FOR PROBE
00A3+ CALL RANGE
00A6+ DO 286 J=1,NP
00B6+ 286 IBUFD(J)=IBUFD(J)+IBUFB(J)
00E0+ 288 CONTINUE
00E3+ 2283 DO 3284 J=1,NP
00E2+ 3284 IBUFB(J)=IBUFD(J)/I'AVG
00E3+
00E3+ C ASK USER IF HE WANTS DATA(PROBE RESIS ) PRINTED OR NOT
00E3+ 2881 PUT 6,0,' PRINT DATA? Y/N'
00E3+ READ(ICN,175) I'ANS
00E3+ IF(EQUAL(I'ANS,0,'Y',0,1)) GO TO 2882
00E3+ IF(EQUAL(I'ANS,0,'N',0,1)) GO TO 2884
00E3+ WRITE(IPR,185) I'ANS
00E3+ GO TO 2881
00E3+ 2882 IF(I'PR,2883)
00E3+ 2883 FOR I(5('CH# ','RESIS',' '))
00E3+ WRITE(IPR,2886)(I-UTC(N),IBUFB(N),'=1,NP)
00E3+ 2883 FOR I(5(13,' ',15,' '))
00E3+
00E3+ C ASK USER IF HE WANTS TO CHANGE ANY DATA OR NOT
00E3+ 2884 PUT 6,0,' CHANGE OR NOT? Y/N'
00E3+ READ(ICN,175) I'ANS
00E3+ IF(EQUAL(I'ANS,0,'Y',0,1)) GO TO 2885
00E3+ IF(EQUAL(I'ANS,0,'N',0,1)) GO TO 2887
00E3+ C USER LIST TO CHANGE DATA I. ARRAY IBUFB
00E3+ IF(I'PR,185) I'ANS
00E3+ GO TO 2884
00E3+ 2885 IF(I'PR,185) I'ANS
00E3+ IF(I'PR,185) I'ANS
00E3+ CALL LINKER(MAINS,0)
00E3+ C ASK USER IF HE WANTS TO STOP THIS SET OF DATA OR NOT
00E3+ 2887 PUT 6,0,' STOP DATA? Y/N'
00E3+ READ(ICN,175) I'ANS
00E3+ IF(EQUAL(I'ANS,0,'Y',0,1)) GO TO 2888
00E3+ IF(EQUAL(I'ANS,0,'N',0,1)) GO TO 2889
00E3+ IF(I'PR,185) I'ANS
00E3+ GO TO 2887
00E3+ 2888 IF(I-1)
00E3+ GO TO 2885
00E3+ 2889 IF(I-1)
00E3+ GO TO 2885
00E3+ C COMPUTE DATA AND TIME
00E3+ 2890 IF(I'PR,185) I'ANS
00E3+ IF(I'PR,185) I'ANS
00E3+ CALL LINKER(MAINS,7)
00E3+
00E3+ C ASK USER IF HE WANTS TO STOP THIS SET OF DATA OR NOT
00E3+ 2891 PUT 6,0,' STOP DATA? Y/N'
00E3+ READ(ICN,175) I'ANS
00E3+ IF(EQUAL(I'ANS,0,'Y',0,1)) GO TO 2892
00E3+ IF(EQUAL(I'ANS,0,'N',0,1)) GO TO 2893
00E3+ IF(I'PR,185) I'ANS
00E3+ GO TO 2891
00E3+ 2892 IF(I-1)
00E3+ GO TO 2891
00E3+ 2893 IF(I-1)
00E3+ GO TO 2891

```


USER=TUNG 255 *417 JOINT COMPUTER FACILITY, MIT PAGE 8

```

12843 IANS=255*(NR/10+48)+MOD(NR,10)+48
123C4 CALL MOVE(IANS,0,IMIX,6,2)
12CCF WRITE(IPR,325) IMIX
12E4F 325 FOR AT(5X,542)
12F5P IF(IPT)257,2575,257
130AF 330 PUT 6,0,' MIX OR QUIT? M/Q'
1332P READ(ICN,175) IANS
134EP IF(EQU'L(IANS,0,'M',0,1)) GO TO 340
136EP IF(EQUAL(IANS,0,'Q',0,1)) GO TO 400
138EP WRITE(IPR,135) IANS
13AAR GO TO 330
13AEP 340 IN=0
1336P ICHAN=C
135EP GO TO 20
13C2P 400 IQUIT=C
13C4P BLANK
13D2P END
1C5[C] // 000[V] ITA 0002[V] ITB 0004[V] IA 0006[V] IV
000[V] NS 000[V] IIC 000C[V] IF 000C[V] ITRROR 0010[V] IDATE
001L[V] ITTY 0014[V] ID 0016[V] IRTED 0018[V] IETP 001A[V] NR
001C[V] NT 001L[V] IBOUND 0020[V] LBOUND 0026[V] IFTSET 0028[V] IPTCAL
002A[V] JYCN 002C[V] JDAY 002F[V] JYR 0030[V] JAR 0032[V] JMIN
0034[V] JSFC 0036[V] JMLSEC 0038[V] JDATE 003A[V] PERIOD 003C[V] IDUPA
003E[V] IOLD 0040[V] IDY 0042[V] IDY 0044[V] IYR 0046[V] IHP
0048[V] IMIN 004[V] ISEC 004C[V] IMLSEC 004E[V] IMTWP 0050[V] IEFLOW
0052[V] IURE 0054[V] BFLOW 0058[V] TEMPM 005C[V] ICTEMP 005E[V] TEMPIC
0062[V] AREA 0066[V] DE 006A[V] A4PL 006E[V] IASK 0070[V] IN
0072[V] IT 0074[V] ASST 0078[V] NCAL 007A[V] NMTY 007C[V] NUA
007E[V] IOUT 0080[V] ASST 0082[V] NSEF 0084[V] NCHN 0086[V] I47LP
0090[V] IPOD 0094[V] IE 009C[V] LEVELS 009E[V] TMT 00A0[V] ICAT
00A2[V] IJAC 00A4[V] IJFB 00A6[V] IJUF 00A8[V] IDUPB 00AA[V] N47C
00AC[V] ICAD 00AE[V] ICN 00B0[V] IPR 00B2[V] IECHO 00B4[V] IPATV
00B6[V] IE T 00B8[V] I CON 00BA[V] IJF 00BC[V] N47V 00BE[V] SLTLOW
00C0[V] IPRCH 00C2[V] IJFOL 00C4[V] IJFLOW 00C6[V] N47V 00C8[V] NL
00CA[V] NLEVEL 00CC[V] ICAD 00CE[V] ICCON 00D0[V] MAINS 00D2[V] IMTME
00D4[V] ICICAL 00D6[V] LI 00D8[V] MI 00DA[V] .0 00DC[V] .0
00DE[V] IJUT 00E0[V] TVAL 00E2[V] IJLX 00E4[V] 1 00E6[V] 7
00E8[V] 175 00EA[V] 145 00EC[V] 1100 00EE[V] 1200 00F0[V] 1000
00F2[V] 2000 00F4[V] 250 00F6[V] 250 00F8[V] 2847 00FA[V] 5000
00FC[V] 6000 00FE[V] 10 0100[V] ICHAN 00A[V] 20 00C0[S] MOV
00C2[S] GVAME 0126[L] 30 00C4[S] IJARE 00C6[S] 37 00C8[S] COPY
014C[V] AEG 00C8[S] .V 0142[V] ALOW 0148[V] AITGH 00C0[S] GETUM
00D0[C] .Y 00D2[S] 80 0144[L] 100 0146[V] 85 00C0[S] 21
00D4[L] 14 00D6[S] CALPE 0148[L] 170 014C[V] IJVS 00C0[S] 200
00D8[L] 210 00DA[S] LIKER 0150[L] 220 014E[V] T 00D8[L] 230
00DA[L] 255 00DE[L] 260 0152[L] 270 014F[V] IJTT 00DA[L] 2570
00DC[L] 2575 00E0[L] 257 0158[L] 258 0152[V] 284 00DA[L] 265
00E2[L] 1284 00E4[L] ST42 00E6[L] 238 00C0[S] PROPER 00C0[S] RANTE
00E8[L] 0 00E8[V] 1 00E8[L] 220 00C0[S] 224 00E4[L] 2200
00F0[L] 2002 00F0[L] 204 00F0[L] 2843 00F0[L] 2300 00F0[L] 235
00F2[L] 235 00F2[L] 230 00F2[L] 230 00F2[L] 235
00F4[L] 330 00F4[L] 320 00F4[V] IJ 00C0[S] MOV 00F4[L] 325
00F6[L] 14 00F6[L] 40

```

I J 003477 JUL 0010 1141 700 0015 00 STORAGE SPACE

SECHS O TROCKS S H M 1964

USER=TUNG 255 24170 JOINT COMPUTER FACILITY, MIT PAGE 1

```

0004P      SUBROUTINE RANGE
002AP      C THIS SUBROUTINE WILL CHECK IF EACH ELEMENT IN IBUF8 (PROBE RESIS
002AR      C TAKEN BY SUBROUTINE PROLES) IS WITHIN THE REQUIRED RANGE, DISPLAY
002AP      C THOSE THAT ARE OUT OF RANGE AND IT'S CORRESP PROBE #. IF THERE'S ANY
002AR      C RESIS OUT OF RANGE, ASK USER IF HE WANTS TO KEEP THIS SET OF DATA.
002AP      C COMMON FILE
002AP      IMPLICIT INTEGER*2 (I-V)
002AR      INTEGER*2 PERIOD
002AR      REAL LBOUND, IT, INPUT(10)
002AR      LOGICAL EQUAL
002AR      COMMON   ITA, ITH, IA, IH, NS, IIS, NP, IERROR, IDATE, ITIME, ID,
002AP      1         IREFER, IPTW, NR, NT, HBOUND, LBOUND, IPTSET, IPTCAL,
002AR      1         JMON, JDAY, JYR, JHP, JMIN, JSEC, J*LSPEC, JDATE,
002AR      1         PERIOD, IDURA, IOLF, IYCN, IDAY, JYR, IHP, IMIN, ISFC, I*LSFC,
002AR      1         ITEMP, IIFLOW, IBSF, IFLOW, TEMPIV, ICTEMP, TEMPIC,
002AR      1         ARPA, DE, A*PL, IACK, IN, IT,
002AR      1         ISET, NCAL, NMIX, NUW, IQUIT,
002AR      1         MAXSET,
002AR      1         ISET(2,24), NCHAN(2,512), IHELP,
002AR      1         IBUF8(512), IE(10,2), LEVELS(100), I*IX(5), ICAL(5),
002AR      1         IBUF8(512), IBUF8(512),
002AR      1         IBUF8(24), IBUF8(512), NAME(5), ICCMD(2),
002AR      1         ICN, IPR, IECHO,
002AR      1         IPHAIN, IPWT,
002AR      1         IACON, ISUP, NAVG, SLFLOW, IPRON, IPROL, IJFLOW, NAVG,
002AR      1         NL, NLEVEL, ICCMD, ICCON,
002AR      1         MAINC(60), I*TIME, ICPCAL
002AP      C END OF COMMON FILE
002P      PUT 6,0, ' OUT OF RANGE PROBE RESIS CHECKING:'
0004      PUT 6,0, ' PROBE#:' RESIS:'
0005P      C CHECK EACH PROBE RESIS, IF OUT OF RANGE, DISPLAY ITS VALUE AND
0005P      C ITS CORRESP PROBE#
0005P      DO 10 J=1,NP
0005P      IF(IIBUF8(J).LE.IPTOL.AND.IBUF8(J).GE.IPROL) GO TO 20
0005P      I IIS(IIP,15) IIBUF8(J), IBUF8(J)
0005P      15 PRINT(10,15,5,15)
0005P      IIR-01=1
0005P      20 CONTINUE
0005P      C OUTPUT 'OK' IF ALL RESIS ARE OUT OF RANGE, THEN RETURN
0005P      IF (IERROR.EQ.1) RETURN
0005P      PUT 6,0, ' OK'
0005P      RETURN
0005P      END
1E06[C] // 0000[V] ITA 0002[V] ITH 0004[V] IY 0006[V] IH
0008[V] NS 0010[V] NP 0012[V] IREFER 0014[V] IDATE
0016[V] IPTW 0018[V] NR 0020[V] NT 0022[V] HBOUND 0024[V] IPTSET 0026[V] IPTCAL
0028[V] JMON 0030[V] JYR 0032[V] JHP 0034[V] JMIN 0036[V] J*LSPEC 0038[V] JDATE
0040[V] PERIOD 0042[V] IDURA 0044[V] IOLF 0046[V] IYCN 0048[V] IDAY 0050[V] IHP 0052[V] IMIN 0054[V] ISFC 0056[V] I*LSFC
0058[V] ITEMP 0060[V] IIFLOW 0062[V] IBSF 0064[V] IFLOW 0066[V] TEMPIV 0068[V] ICTEMP 0070[V] TEMPIC
0072[V] ARPA 0074[V] DE 0076[V] A*PL 0078[V] IACK 0080[V] IN 0082[V] IT
0084[V] ISET 0086[V] NCAL 0088[V] NMIX 0090[V] NUW 0092[V] IQUIT
0094[V] MAXSET
0096[V] ISET(2,24) 0098[V] NCHAN(2,512) 0100[V] IHELP
0102[V] IBUF8(512) 0104[V] IE(10,2) 0106[V] LEVELS(100) 0108[V] I*IX(5) 0110[V] ICAL(5)
0112[V] IBUF8(512) 0114[V] IBUF8(512)
0116[V] IBUF8(24) 0118[V] IBUF8(512) 0120[V] NAME(5) 0122[V] ICCMD(2)
0124[V] ICN 0126[V] IPR 0128[V] IECHO
0130[V] IPHAIN 0132[V] IPWT
0134[V] IACON 0136[V] ISUP 0138[V] NAVG 0140[V] SLFLOW 0142[V] IPRON 0144[V] IPROL 0146[V] IJFLOW 0148[V] NAVG
0150[V] NL 0152[V] NLEVEL 0154[V] ICCMD 0156[V] ICCON
0158[V] MAINC(60) 0160[V] I*TIME 0162[V] ICPCAL

```


0 JSEP=EU - 255 4417 TOTAL COMPUTER FACILITY, 11" PAGE 2

002 [V] JFON	0020 [V] J V	0008 [V] J/P	0030 [V] J/P	0002 [V] JMI
003 [V] JFAC	003 [V] J ISEC	003 [V] JDATE	0034 [V] PERIOD	0030 [V] ITH
004 [V] JFID	004 [V] J/K	004 [V] J/T	0044 [V] TVT	0046 [V] IIE
005 [V] JFIB	005 [V] J/C	0040 [V] IWISEC	0048 [V] INTIME	0050 [V] IIFLOW
006 [V] JFIB	006 [V] J/TIC	0050 [V] TRIPIN	0050 [V] ICTIME	0058 [V] TEATIC
007 [V] JFIB	007 [V] J/T	006 [V] AXIL	0060 [V] IASK	0070 [V] IV
008 [V] JFIB	008 [V] J/T	0070 [V] NCIL	0071 [V] MAY	0070 [V] NGV
009 [V] JFIB	009 [V] J/T	0080 [V] MSFT	0082 [V] VCHAN	0082 [V] ITHLP
010 [V] JFIB	010 [V] J/T	0090 [V] LFVILS	0094 [V] ITH	0098 [V] ICIL
011 [V] JFIB	011 [V] J/T	0100 [V] IEUFA	0108 [V] ITHFB	0118 [V] NAME
012 [V] JFIB	012 [V] J/T	0120 [V] IPR	0122 [V] ITHO	0120 [V] ITHIN
013 [V] JFIB	013 [V] J/T	0132 [V] ISUP	0134 [V] NAVG	0136 [V] SIFLOW
014 [V] JFIB	014 [V] J/T	0132 [V] IJELON	0140 [V] NAVG	0142 [V] NL
015 [V] JFIB	015 [V] J/T	0140 [V] ICCON	0142 [V] MAINS	0152 [V] INTIME
016 [V] JFIB	016 [V] J/T	0170 [L] 200	0000 [S] .0	0000 [S] .0
017 [V] JFIB	017 [V] J/T	0170 [L] 200	0000 [S] .0	0000 [S] .0
018 [V] JFIB	018 [V] J/T	0170 [L] 200	0000 [S] .0	0000 [S] .0
019 [V] JFIB	019 [V] J/T	0170 [L] 200	0000 [S] .0	0000 [S] .0

ALL 2000 4 4 2151 BYTES OF STORAGE SPACE

USER=TUVC 255 84170 JOINT COMPUTER FACILITY, MIT PAGE 1

```

00045      SUBROUTINE CHANGE
002AR      C THIS IS A SUB ENTERED WHEN 'CHANGE' IS ENTERED AS A COMMAND AT
002AR      C THE COMMAND LEVEL IN THE MAIN PROG
002AR      C IT'S PURPOSE IS TO LET USER CHANGE DATA INPUT IN THE MAIN PROG
002AR      C COMMON FILE
002AR      IMPLICIT INTEGER*2 (I-N)
002AR      INTEGER*2 PERIOD
002AR      REAL LBOUND,IT,INPUT(10)
002AR      LOGICAL EQUAL
002AR      COMMON ITA,ITH,IA,IH,NS,IIS,NP,IERRO,DATE,ITIME,ID,
002AR      1 IREFEP,IPTH,NR,NT,FBOUND,LBOUND,IPTSET,IPTCAL,
002AR      1 JMON,JDAY,JYR,JHR,JMIN,JSEC,JMLSEC,JDATE,
002AR      1 PERIOD,IDURA,IOLD,IYON,IDAY,IYF,IHP,IMIN,ISEC,IWLSEC,
002AR      1 IMTFP,IPFLOW,IBRE,BFLOW,TEMPIM,ICTEMP,TEMPIC,
002AR      1 AREA,DE,AMPL,IASK,IN,IT,
002AR      1 NSET,NCAL,NMIX,MUV,IQUIT,
002AR      1 MAXST,
002AR      1 NSET(2,24),*CHAN(2,512),IHELP,
002AR      1 IPUF(512),IP(10,2),LEVELS(100),IMIX(5),ICAL(5),
002AR      1 IPUFC(512),BUFB(512),
002AR      1 IPUF(24),IPUFB(512),NAME(5),ICMD(2),
002AR      1 ICN,IPR,IECHO,
002AR      1 IRMAIN,IPNT,
002AR      1 IYCON,ISUR,MAVG,SIFLOW,IPROH,IPROL,IJFLOW,MAVG,
002AR      1 NL,NLEVEL,ICMD,ICCON,
002AR      1 MAINS(60),IMTIME,ICPCAL
002AR      C END OF COMMON FILE
002AR      13 FORMAT(5X,F6.2,' RESTRICTION: <DE<=10.0IN')
0056      18 FORMAT(5X,F6.2,' RESTRICTION: <AREA<=31.IN**2')
0057      15 FORMAT(5X,'ERROR: ',/, 'ID=',I5,' RESTRICTION: ID>0')
0058      80 NEWDAT=0
0059      NEWTIM=
0060      I=ITR(IPR&35)
0061      85 PRINTLIST ' CHANGE LIST:',(DE,AREA,ID,NEWDAT,NEWTIM)
0062      IERR=0
0063      86 PRINTLIST ' INPUT CHANGE:',(DE,AREA,ID,NEWDAT,NEWTIM)
0064      IF(LE.GT.0..AND. DE.LE.10.) GO TO 88
0065      WRITE(IPR,13)DE
0066      IERR=1
0067      88 IF(AREA.GT.0..AND. AREA.LE.31.) GO TO 89
0068      WRITE(IPR,13) AREA
0069      IERR=1
0070      89 IF(ID.GT.0) GO TO 90
0071      WRITE(IPR,15)ID
0072      IERR=1
0073      90 IF(NEWDAT.LE.0) GO TO 100
0074      IERR=1
0075      91 IF(NEWTIM.LE.0) GO TO 110
0076      PRINTLIST ' TIME(YY,MM):'

```


USER=TUNG 255 84170 JOINT COMPUTER FACILITY, MIT PAGE 1

```

00045      SUBROUTINE INDATE
002A5      C COMMON FILE
002A5      IMPLICIT INTEGER*2 (I-N)
002A5      INTEGER*2 PERIOD
002A5      REAL LBOUND,IT,INPUT(10)
002A5      LOGICAL EQUAL
002A5      COMMON ITA,ITH,IA,IH,NS,IIS,NP,IERRO,IDATE,ITIME,ID,
002A5      1 IREFER,IPTH,NR,NT,HBOUND,LBOUND,IPTSET,IPTCAL,
002A5      1 JMON,JDAY,JYR,JHR,JMIN,JSEC,JMLSEC,JDATE,
002A5      1 PERIOD,IDURA,IOLD,IMON,IDAY,IYR,IHR,IMIN,ISEC,IMLSEC,
002A5      1 IMTEMP,IBFLOW,IBRE,BFLOW,TEMPIM,ICTEMP,TEMPIC,
002A5      1 AREA,DE,AMPL,IASK,IV,IT,
002A5      1 NSET,NCAL,NMIX,NUM,IQUIT,
002A5      1 MAXSET,
002A5      1 MSET(2,24),MCHAN(2,512),IHELP,
002A5      1 IBUED(512),ID(10,2),LEVELS(100),IMIV(5),ICAL(5),
002A5      1 IBUFF(512),IUFF(512),
002A5      1 IBUFF(24),IBUFF(512),NAME(5),ICOMD(2),
002A5      1 ION,IPP,IECHO,
002A5      1 TEMIN,IPNT,
002A5      1 IMCON,ICUE,AVG,CLFLOW,IPRO,IPRO,IJFLOW,MAVE,
002A5      1 NL,NLEVEL,ICMD,ICCON,
002A5      1 MAINS(60),IMTIME,ICRCAL
002A5      C END OF COMMON FILE
002A5      CALL GETTIM(IHR,IMIN,ISEC,IMS)
003A5      IMTIME=ITIME
00405      ITIME=IHR*256+IMIN
00505      ITIME=ISEC*1024+IMS
00505      IIP(125,55) IMTIME
00705      55 FORMAT(24,'IMTIME=',I7)
00805      55 WRITE(IIP,65) IHR,IMIN,ISEC,IMS
00805      65 FORMAT(24,'IHR,IMIN,ISEC,IMS:',4I7)
00905      C CHECK IF THIS SUB IS CALLED TO HAVE DATE INPUT:
00905      IF(ITY.0) GO TO 100
00905      PUT IT IN SCREEN IT UP
00905      RETURN
01305
01305      C *****
01305      10 N=)
01405      IF(IDEFLA.GT.ITIME) N=1
01505      IF( .E.0) PUT 6,0,' JUST PAST MIDNITE'
01605      IF(N.E.0) RETURN
01705      C *****
01805      110 IYR=IYR+1000
01805      LEAP=MOD(IYR,4)
01905      IF(IDEFLA.EQ.31) GO TO 120
01905      IF(IDEFLA.EQ.29 .AND. IMON.EQ.2) GO TO 120
01905      IF(IDEFLA.EQ.28 .AND. IMON.EQ.2 .AND. LEAP.EQ.0) GO TO 120
02005      IF(IDEFLA.EQ.30 .AND.
02105      1 (IMON.EQ.20.4.0 .I .EQ.5 .O .IMON.EQ.9 .OR. IMON.EQ.11 ))
02105      1 GO TO 120
02205      C NOT T
02305      IL =IYR+1

```


USER=TUNG 255 84177 JOINT COMPUTER FACILITY, MIT PAGE 1

```

00042      SUBROUTINE SETDAT
002A2      C COMMON FILE
002A2      IMPLICIT INTEGER*2 (I-N)
002A2      INTEGER*2 PERIOD
002A2      REAL LBOUND,IT,INPUT(10)
002A2      LOGICAL EQUAL
002A2      COMMON      ITA,ITH,IA,IH,NS,IIS,NP,IERROR,IDATE,ITIME,ID,
002A2      1      IREFER,IPTH,NR,NT,HBOUND,LBOUND,IPTSFT,IPTCAL,
002A2      1      JMON,JDAY,JYR,JHP,JMIN,JSEC,JMLSEC,JDATE,
002A2      1      PERIOD,IDURA,IOLD,IYR,IMON,IDAY,IYR,IHR,IMIN,ISEC,IYMLSEC,
002A2      1      INTMP,IBFLOW,IBRE,BFLOW,TEMPIM,ICTEMP,TEMPIC,
002A2      1      AREA,DE,AMPL,IASK,IN,IT,
002A2      1      NSET,NCAL,NMIX,NUM,IQUIT,
002A2      1      MAXSET,
002A2      1      MSET(2,24),*CHAN(2,512),IHELP,
002A2      1      IPUFD(512),IP(10,2),LEVELS(100),I*IX(5),ICAL(5),
002A2      1      IPUFC(512),I*UFB(512),
002A2      1      I*UFA(24),I*UFB(512),NAME(5),ICOMD(2),
002A2      1      ICM,IPR,I*CHC,
002A2      1      I*MAIV,I*PT,
002A2      1      ICCON,ISUS,NAVG,SLFLOW,IPROH,IPROL,IJFLOW,MAVG,
002A2      1      NL,NLEVEL,ICMD,ICCON,
002A2      1      *PINS(60),I*TIME,ICPCAL
002A2      C END OF COMMON FILE
012A2      2 PUT 5,0,' DATE(MON,DAY,YR):'
00522      READ(ICM,15) IYMON,IDAY,IYR
007A2      15 FORMAT(3I5)
008A2
008A2      IF(IYMON.GE.1 .AND. IYMON.LE.12 .AND. IDAY.GE.1 .AND.
008A2      1IDAY.LE.31 .AND. IYR.GE.76 .AND. IYR.LE.91) GO TO 150
008A2
008A2      WRITE(IPR,25) IYMON,IDAY,IYR
010A2      25 FORMAT(5X,2(I3,' '),I3,' ' INVALID INPUT DATE)
013A2      GO TO 1
014A2      150 IDATE=IYMON*512+(IDAY*16)+(IYR-76)
016A2      RETURN
017A2      END
1E08[C] //      0000[V] IT      0002[V] ITH      0004[V] IA      0006[V] IH
0008[V] NS      0010[V] IIS      0012[V] NP      0014[V] IERROR      0016[V] IDATE
0018[V] ITIME      0020[V] ID      0022[V] IREFER      0024[V] IPTH      0026[V] NR
0028[V] NT      0030[V] HBOUND      0032[V] LBOUND      0034[V] IPTSFT      0036[V] IPTCAL
0038[V] JMON      0040[V] JDAY      0042[V] JYR      0044[V] JHP      0046[V] JMIN
0048[V] JSEC      0050[V] JMLSEC      0052[V] JDATE      0054[V] PERIOD      0056[V] IDURA
0058[V] IOLD      0060[V] IYR      0062[V] IMON      0064[V] IDAY      0066[V] IYR
0068[V] IHR      0070[V] IMIN      0072[V] ISEC      0074[V] IYMLSEC      0076[V] INTMP
0078[V] IBRE      0080[V] BFLOW      0082[V] TEMPIM      0084[V] ICTEMP      0086[V] TEMPIC
0088[V] AREA      0090[V] DE      0092[V] AMPL      0094[V] IASK      0096[V] IN
0098[V] IT      0100[V] NSET      0102[V] NCAL      0104[V] NMIX      0106[V] NUM
0108[V] IQUIT      0110[V] MAXSET      0112[V] MSET      0114[V] *CHAN      0116[V] IHELP
0118[V] IPUFD      0120[V] IP      0122[V] LEVELS      0124[V] I*IX      0126[V] ICAL
0128[V] IPUFC      0130[V] I*UFB      0132[V] I*UFA      0134[V] NAME      0136[V] ICOMD
0138[V] ICM      0140[V] IPR      0142[V] I*CHC      0144[V] I*MAIV      0146[V] I*PT
0148[V] ICCON      0150[V] ISUS      0152[V] NAVG      0154[V] SLFLOW      0156[V] IPROH
0158[V] IROL      0160[V] IJFLOW      0162[V] MAVG      0164[V] NL      0166[V] NLEVEL
0168[V] ICMD      0170[V] ICCON      0172[V] *PINS      0174[V] I*TIME      0176[V] ICPCAL

```

2 USER=ICG 255 0417 JOINT COMPUTER FACILITY, MIT PAGE 2

1E3 [V] IPRDH	1 30 [V] IPRDL	1E3E [V] IJFLOW	1E40 [V] MAVG	1E42 [V] NL
1E44 [V] NLVEL	1 40 [V] IPRD	1E4E [V] ICCON	1E50 [V] AINS	1E52 [V] IXTIME
1E54 [V] ICRAL	1 50 [V] IPRDAT	1E5E [V] SETINT	1E60 [S] .0	1E62 [S] .P
1E64 [V] IPRUT	1 60 [V] IPRAL	1E6E [L] 2	1E70 [S] 0"	1E72 [L] 15
1E74 [S] LI	1 70 [L] 100	1E7E [L] 25		

THIS PROGRAM WILL OCCUPY 10100 BYTES OF STORAGE SPACE
 PROBLEM SETTING HAS NO PROBLEMS

USER=TUNG 255 84170 JOINT COMPUTER FACILITY, MIT PAGE 1

```

00048 SUBROUTINE PICKER
002AR C SUB PICKER IS TO PICK UP #S FROM THE CONSOLE
002AR C COMMON FILE
002AR IMPLICIT INTEGER*2 (I-N)
002AR INTEGER*2 PERIOD
002AR REAL LBOUND,IT,INPUT(10)
002AR LOGICAL EQUAL
002AR COMMON ITA,ITH,IA,IH,NS,IIS,NP,IERROR,IDATE,ITIME,ID,
002AR 1 IREFER,IPTH, NR,NT,HBOUND,LBOUND, IPTSET,IPTCAL,
002AR 1 JMON,JDAY,JYR,JHR,JMIN,JSEC,JMLSEC, JDATE,
002AR 1 PERIOD,IDURA,IOLD, IMON,IDAY,IYR,IHR,IMIN,ISEC,IMLSEC,
002AR 1 IMTEMP,IBFLOW,IBRE, BFLOW,TEMPIM,ICTEMP,TEMPIC,
002AR 1 AREA,DE, AMPL,IASK,IN,IT,
002AR 1 NSET,NCAL,NMIX,NUM,IQUIT,
002AR 1 MAXSET,
002AR 1 MSET(2,24),MCHAN(2,512),IHELP,
002AR 1 IUFED(512),IP(10,2),LEVELS(100),I*IV(5),ICAL(5),
002AR 1 IBUFC(512), IUFEB(512),
002AR 1 IBUFA(24),IBUFB(512),NAME(5), ICOND(2),
002AR 1 ICN,IPR,IECHO,
002AR 1 IRMAIN,IPNT,
002AR 1 IMCON,ISUP,NAVG,SLFLOW,IPROH,IPROL,IJFLOW,MAVG,
002AR 1 NL,NLEVEL,ICMD,ICCON,
002AR 1 MAINS(60),IMTIME,ICRCAL
002AR C END OF COMMON FILE
002AR C SPECIFICATION DATA BLOCK FOR THIS SUBROUTINE
002AR C START TO COLLECT INPUT #S, STORE IN ARRAY IBUFB
002AR C CLEAR BUFFER B (ARRAY IUFEB)
002AR 1 DO 3 I=1,512
0032- 3 IUFEB(I)=0
0056-
0056- L=1
0060- IHELP=
0062- IP(L,1)=0
0082- IP(L,2)=0
0090- IPR=1
0094- ICN(1)=1
0096-
0096- 4 PUT 5,0,' INPUT #S,<=10 PER LINE,-1 FOR QUIT,-2 FOR #S DISPLAY'
0096- 5 PUT 6,0,' #S>'
0112- READ(ICN,15) INPUT
0130- 15 FORMAT(10F3.0)
0132- 16 FORMAT(5X,10F3.2)
0150- IERROR=0
0158-
0158- IT=INPUT(1)
0164- IF(IT.EQ.-1.) RETURN
0172-
0172- IF(IT.EQ.-2.) GO TO 17
0174- L=L+1
0174- IPR=IPR+1
0174- GO TO 10
0174-

```


USER=TUNG 255 84170 JOINT COMPUTER FACILITY, MIT PAGE 3

```

06245      70 CONTINUE
06345
06346      GO TO 5
06347      80 PUT 6,0,' ERROR:# OF INPUTS EXCEEDS EXPECTED INPUT #S'
06400      IF(ICONSE.EQ.1) IP(L-1,2)=NT
06401      WRITE(IPR,85) BUFB(NT)
06402      85 FORMAT(5X,'LAST INPUT #S:',F6.2)
06403      GO TO 100
06404
06405      C THE 10 INPUT ITEMS HAS BEEN PROCESSED:
06406      C INPUT MORE #S IF NOT ENOUGH #S HAVE BEEN INPUT
06407      90 IF(INP.LE.NT) GO TO 5
06408      C
06409      95 PUT 6,0,' #S COLLECTING IS DONE'
06410      WRITE(IPR,85) BUFB(NT)
06411
06412      100 IE=0
06413      IF(BUFB(1).NE.0.)GO TO 101
06414      PUT 6,0,' NO INPUT #S YET'
06415      GO TO 5
06416      101 DO 150 I=1,L
06417          IF(I.NE.L) GO TO 102
06418          IF(IP(I,1).EQ.0) GO TO 150
06419          N=NT-IP(I,1)+1
06420          IF(IE.EQ.1) N=LESS -IP(I,1)+1
06421          GO TO 112
06422      102 IF(IP(I,2).EQ.0) GO TO 110
06423      C CONSECUTIVE INPUT DISPLAY
06424      IB=IP(I,1)
06425      IE=IP(I,2)
06426      103 IF(MPL.EQ.1.)
06427          1 WRITE(IPR,104) IB,IE, BUFB(IB), BUFB(IE)
06428          IF(MPL.EQ.100.)
06429              1 WRITE(IPR,105) IB,IE, BUFB(IB), BUFB(IE)
06430      104 FORMAT(5X,2I5,':',F5.0,' TO',F5.0)
06431      105 FORMAT(5X,2I5,':',F5.0,' TO',F5.2)
06432      109 GO TO 150
06433      C NON-CONSECUTIVE ENTRY #S DISPLAY
06434      110 N=IP(I+1,1)-IP(I,1)
06435      112 LLEN=N/10
06436      IPST="00(N,10)
06437      IF (LTEN.EQ.0) GO TO 125
06438      C DISPLAY 10 NON-PLAY #S
06439      DO 120 K=1,LTEN
06440          IB=IE+1
06441          IE=IE+9
06442      113 IF(MPL.EQ.1.)
06443          1 WRITE(IPR,114) IB,IE,( BUFB("),N=IB,IE)
06444          IF(MPL.EQ.100.)
06445              1 WRITE(IPR,115) IB,IE,( BUFB("),N=IB,IE)
06446      114 FORMAT(5X,2I5,':',10F5.0)
06447      115 FORMAT(5X,2I5,':',10F5.2)
06448      120 CONTINUE

```

```

0001 125 IF(IEBOT.EQ.0) GO TO 150
0002      I=IE+1
0003      IL=1+(I-1)*11
0004      IF( IL.GT.100.)
0005      *ITF(IE,114 ) I ,TF,( 3 FB(J),J=IE,IT)
0006      IF( IL.EQ.100.)
0007      *ITF(IE,115 ) I ,TF,( 8UFB(J),J=IE,IF)
0008 150      IF(IEBOT.EQ.1) GO TO 5
0009 200 PUT 6, , ? K/C/O (WILL/CHANGE/OK)'
0010      REEL(IE,205) INMS
0011 205 REPEAT( 2)
0012 220 IF( .NOT.(INMS ,0,'Y',0,1)) GO TO 1
0013      I ( .NOT.(INMS ,0,'Y',0,1)) GO TO 26
0014 230 I-( .NOT.(INMS ,0,'Y',0,1)) GO TO 300
0015      *ITF(IE,250) INVT
0016 240 IF( .NOT.(INVT ,0,'Y',0,1)) GO TO 250
0017      *ITF(IE,251) INVT
0018 250 IF( .NOT.(INVT ,0,'Y',0,1)) GO TO 250
0019 260 IF( .NOT.(INVT ,0,'Y',0,1)) GO TO 260
0020 300 I=1,*T
0021 400 I (I)= UFB(I)*-1 L
0022      R 11
0023      I 0
1000 [C] //
0001 [V] AS
0011 [V] ITI I
0012 [V] T
002 [V] J
003 [V] JC C
004 [V] IOL
005 [V] IOL
006 [V] IOL
007 [V] IT
008 [V] IOL
009 [V] IOL
010 [V] IOL
011 [V] IOL
012 [V] IOL
013 [V] IOL
014 [V] IOL
015 [V] IOL
016 [V] IOL
017 [V] IOL
018 [V] IOL
019 [V] IOL
020 [V] IOL
021 [V] IOL
022 [V] IOL
023 [V] IOL
024 [V] IOL
025 [V] IOL
026 [V] IOL
027 [V] IOL
028 [V] IOL
029 [V] IOL
030 [V] IOL
031 [V] IOL
032 [V] IOL
033 [V] IOL
034 [V] IOL
035 [V] IOL
036 [V] IOL
037 [V] IOL
038 [V] IOL
039 [V] IOL
040 [V] IOL
041 [V] IOL
042 [V] IOL
043 [V] IOL
044 [V] IOL
045 [V] IOL
046 [V] IOL
047 [V] IOL
048 [V] IOL
049 [V] IOL
050 [V] IOL
051 [V] IOL
052 [V] IOL
053 [V] IOL
054 [V] IOL
055 [V] IOL
056 [V] IOL
057 [V] IOL
058 [V] IOL
059 [V] IOL
060 [V] IOL
061 [V] IOL
062 [V] IOL
063 [V] IOL
064 [V] IOL
065 [V] IOL
066 [V] IOL
067 [V] IOL
068 [V] IOL
069 [V] IOL
070 [V] IOL
071 [V] IOL
072 [V] IOL
073 [V] IOL
074 [V] IOL
075 [V] IOL
076 [V] IOL
077 [V] IOL
078 [V] IOL
079 [V] IOL
080 [V] IOL
081 [V] IOL
082 [V] IOL
083 [V] IOL
084 [V] IOL
085 [V] IOL
086 [V] IOL
087 [V] IOL
088 [V] IOL
089 [V] IOL
090 [V] IOL
091 [V] IOL
092 [V] IOL
093 [V] IOL
094 [V] IOL
095 [V] IOL
096 [V] IOL
097 [V] IOL
098 [V] IOL
099 [V] IOL
1000 [V] IOL

```


USER=TUNG 255 84170 JOINT COMPUTER FACILITY, MIT PAGE 5

0344[V] LTV	0E46[V] IRFST	0000[S] MOD	0B50[L] 125	0B2E[L] 120
0A36[L] 113	0A1E[L] 114	0B14[L] 115	0E4A[V] J	0CF2[L] 200
0C9E[L] 205	0E4C[V] IAMS	0CA2[L] 220	0D5A[L] 260	0CE8[L] 230
0D3E[L] 300	0D24[L] 250	0D6C[L] 265	0000[S] 24	0D40[L] 400

THIS PROGRAM WILL OCCUPY X'0E4E' BYTES OF STORAGE SPACE@
 PROGRAM PICKER HAS NO ERRORS

USER=TUNG 255 84170 JOINT COMPUTER FACILITY, MIT PAGE 2

1EC4[V] ICRCAL 0024[B] GVNAME 0122[R] GVNAME 0000[S] .Q 0000[S] .P
012[V] INPJT 0000[S] EQUAL 0152[V] IAVUM 0000[S] MOD 0000[S] MOVE
00F6[L] 15 0000[S] xI 0112[L] 20

THIS PROGRAM WILL OCCUPY X'016C' BYTES OF STORAGE SPACE@
PROGRAM GVNAME HAS NO ERRORS

1 JCRC=JCV. 255 4170 JOINT COMPUTER FACILITY, MIT PAGE 2

[illegible]

FORAS RECORDS - 21 -

USER=TUNG 255 84170 JOINT COMPUTER FACILITY, MIT PAGE 1

```

0004*      SUBROUTINE INAME
002A*
002A*      C SUBR INAME IS CALLED BY SUBR CAL OR SUBR MIX TO ASK THE USER FOR
002A*      C A SET REFERENCE NAME, OR A CAL REFERENCE NAME.
002A*      C IF AN IMPROPER NAME IS INPUT, AN ERROR MESSAGE WILL BE PRINTED AND
002A*      C THE USER WILL BE ASKED TO RE-INPUT A NAME.
002A*      C COMMON FILE
002A*      IMPLICIT INTEGER*2 (I-N)
002A*      INTEGER*2 PERIOD
002A*      REAL LBOUND,IT,INPUT(10)
002A*      LOGICAL EQUAL
002A*      COMMON ITA,ITH,IA,IH,NS,IIS,NP,IERROR,IDATE,ITIME,ID,
002A*      1 IREFER,IPTH,VR,NT,WBOUND,LBOUND,IPTSET,IPTCAL,
002A*      1 J1ON,JDAY,JYR,JHR,JMIN,JSEC,JMLSEC, JDATE,
002A*      1 PERIOD,IDURA,IOLD,IMON,IDAY,IYP,IHP,IMIN,ISEC,I*MLSEC,
002A*      1 INTMP,IBFLOW,IB*FLOW,TEMPIM,ICTEMP,TEMPIC,
002A*      1 AREA,DE,AMPL,IASK,IN,IT,
002A*      1 NSET,NCAL,NMIX,NUM,IQUIT,
002A*      1 MAXSET,
002A*      1 NSET(2,24),NCHAN(2,512),IHELP,
002A*      1 IBUFD(512),IP(10,2),LEVELS(100),IMIV(5),ICAL(5),
002A*      1 IBUFC(512),IBUFB(512),
002A*      1 IBUFA(24),IBUFB(512),NAME(5),ICOMD(2),
002A*      1 ICN,IPR,IEMO,
002A*      1 IRMAIN,IPVT,
002A*      1 IACON,ISUB,NAVG,SIFLOW,IPROH,IPROL,INFLOW,M*VG,
002A*      1 NL,M*VEL,IC*V,ICCON,
002A*      1 IAINS(60),INTIME,ICRCAL
002A*      C END OF COMMON FILE
002A*      5 WRITE(IPR,15) (NAME(I),I=1,2)
0062*      15 FORMAT(2X,'ENTER ',2A2,' NAME#:')
0090*      8 AD(ICN,25) IPTH
004CE      25 FORMAT(I3)
00B6*      35 FORMAT('+',20X,I2)
00C6*      IF(IPTH.EQ.-1) RETURN
00B2*      IF(IPTH.EQ.0 .AND. IN.VT.1) RETURN
0100*      IF(IPTH.GT.0 .AND. IPTH.LT.100) GO TO 5
0120*      WRITE(IPR,45) IPTH
0142*      45 FORMAT(2X,'ERROR: IPTH=',I2,' RESTRICTION: 0<=100')
0174*      GO TO 5
0178*      50 IF(IPTH.LT.NUM) RETURN
0182*      WRITE(IPR,55) IPTH,(NAME(I),I=1,2)
01DA*      55 FORMAT(2X,'+',I2,' ',2A2,' BLOCK IS NOT CREATED')
020C*      GO TO 5
0210*      END
12C5[C] // 0000[V] ITA 0002[V] ITH 0004[V] IA 0006[V] IH
0008[V] NS 0010[V] IIS 0012[V] NP 0014[V] IERROR 0016[V] IDATE
0018[V] ITIME 0020[V] ID 0022[V] IREFER 0024[V] IPTH 0026[V] VR
0028[V] NT 0030[V] WBOUND 0032[V] LBOUND 0034[V] IPTSET 0036[V] IPTCAL
0038[V] J1ON 0040[V] JDAY 0042[V] JYR 0044[V] JHR 0046[V] JMIN
0048[V] JSEC 0050[V] JMLSEC 0052[V] JDATE 0054[V] PERIOD 0056[V] IDURA
0058[V] IOLD 0060[V] IMON 0062[V] IDAY 0064[V] IYP 0066[V] IHP
0068[V] IMIN 0070[V] ISEC 0072[V] I*MLSEC 0074[V] INTMP 0076[V] IBFLOW
0078[V] IB*FLOW 0080[V] TEMPIM 0082[V] ICTEMP 0084[V] TEMPIC
0086[V] AREA 0088[V] DE 0090[V] AMPL 0092[V] IASK 0094[V] IN
0096[V] IT 0098[V] NSET 0100[V] NCAL 0102[V] NMIX 0104[V] NUM
0106[V] IQUIT 0108[V] MAXSET 0110[V] NSET(2,24) 0112[V] NCHAN(2,512)
0114[V] IHELP 0116[V] IBUFD(512) 0118[V] IP(10,2) 0120[V] LEVELS(100)
0122[V] IMIV(5) 0124[V] ICAL(5) 0126[V] IBUFC(512) 0128[V] IBUFB(512)
0130[V] IBUFA(24) 0132[V] IBUFB(512) 0134[V] NAME(5) 0136[V] ICOMD(2)
0138[V] ICN 0140[V] IPR 0142[V] IEMO 0144[V] IRMAIN 0146[V] IPVT
0148[V] IACON 0150[V] ISUB 0152[V] NAVG 0154[V] SIFLOW 0156[V] IPROH
0158[V] IPROL 0160[V] INFLOW 0162[V] M*VG 0164[V] NL 0166[V] M*VEL
0168[V] IC*V 0170[V] ICCON 0172[V] IAINS(60) 0174[V] INTIME
0176[V] ICRCAL

```

USAR=TUN, 255 54177 JOINT COMPUTER FACILITY, MIT PAGE 2

0052[V] I-FC	0054[V] BELOW	0056[V] TEMPIV	0058[V] TCTEMP	005E[V] TEMPIC
0052[V] APFA	0056[V] DI	0058[V] A-FL	005F[V] IASK	0070[V] IN
0072[V] IT	0074[V] ACET	0076[V] WCAL	0078[V] WMIN	007C[V] NUN
0077[V] IOUT	0081[V] W-SET	0082[V] WST	0082[V] WCHAN	0872[V] IPFLP
0084[V] I-VID	0084[V] IP	008C[V] LEWELT	0084[V] IMIX	008E[V] ICAT
0088[V] I-UFEC	1100[V] LHEA	10E2[V] IBWFA	1A12[V] IPUEB	1E18[V] NAME
1E22[V] ICCAD	1126[V] ICN	1E22[V] IPR	1E2A[V] IECHO	1E2C[V] IRMAIN
1E2C[V] IPVT	1131[V] T-CON	1E22[V] ISUP	1E34[V] W-ANG	1E36[V] SLEFLOW
1E34[V] IP-ON	113C[V] IPROL	1E37[V] IJFLOW	1E40[V] W-ANG	1E42[V] NL
1E44[V] NL-VEI	1E46[V] ICAD	1E44[V] ICCON	1E4A[V] W-AINS	1E02[V] INTIME
1E04[V] ICRCAL	0024[L] I-NAME	0213[P] I-NAME	0000[S] .Q	0000[S] .P
0220[V] INPUT	0240[V] EQUAL	002A[L] 5	006F[L] 15	0000[S] DI
024C[V] I	00A0[L] 25	0006[L] 35	0178[L] 50	0142[L] 45
010A[L] 55				

THIS PROGRAM WILL OCCUPY 40250 BYTES OF STORAGE SPACE
 FROM 100000 TO 100000 + 40250 = 140250

USER=TUNG 255 84170 JOINT COMPUTER FACILITY, MIT PAGE 1

```

0004P      SUBROUTINE CHECK
0024P      C COMMON FILE
002A1      IMPLICIT INTEGER*2 (I-N)
0021R      INTEGER*2 PERIOD
002AR      REAL LBOUND,IT,INPUT(10)
002AF      LOGICAL EQUAL
002AF      COMMON   ITA,ITH,IA,IH,NS,IIS,NP,IEPROR,IDATE,ITIME,ID,
002AR      1         IREFER,IPTH,NP,NT,HBOUND,LBOUND,IPTSET,IPTCAL,
002AR      1         JMON,JDAY,JYR,JHR,JMIN,JSEC,JMLSEC,JDATE,
002AR      1         PERIOD,IDURA,IOLD,IMON,IDAY,IYR,IHP,IMIN,ISEC,IMLSEC,
002AR      1         IMTEMP,IBFLOW,IBFE,BFLOW,TEMPIM,ICTEMP,TEMPIC,
002AP      1         AREA,DE,AMPL,ASK,IN,IT,
002AR      1         NSET,NCAL,NMIX,NUM,IQUIT,
002AP      1         MAXSET,
002AP      1         MSET(2,24),*CHAN(2,512),IHELP,
002AP      1         IPUFB(512),IP(10,2),LEVELS(100),IMIV(5),ICAL(5),
002AP      1         IPUF(512),PUFB(512),
002AP      1         IPUFA(24),IPUFB(512),NAME(5),ICOMD(2),
002AP      1         ICN,IPR,IECHO,
002AP      1         IPMATN,IPNT,
002AP      1         IMCON,ISUB,AVG,SFLOW,IPROH,IPROL,IJFLOW,AVG,
002AP      1         NL,NLEVEL,ICMD,ICCON,
002AP      1         MAINS(60),ITIME,ICRCAL
002AR      C END OF COMMON FILE
002AR      I=IA+NS-ITA-1
002AR      J=IA-ITH
004AP      IF(I.LE.0 .AND. J.LE.0) GO TO 100
006AP      10 PRINT 'DISKIDINGOLD DISK IS FILLED, MOUNT A NEW DISC, INPUT **F
006AP      11 FOR **READY** IF NEW DISC IS MOUNTED'
008AP      12 PRINT '>READY?(Y)'
011AP      READ(ICN,15)IANS
013AP      15 FORMAT(A2)
013AP      IF(-JUL(IANS,0,'F',0,1)) GO TO 40
015AP      WRITE(IPR,25) IANS
017AP      25 FORMAT(2X,A2,' IMPROPER RESPONSE')
019AP      GO TO 12
021AP      C =====
021AP      40 DO 1000 I=1,24
021AP      1000 I UFA(I)=0
021AP      PRINT ' INPUT IH,IA:(0,0)'
021AP      READ(ICN,2000)J,K
021AP      2000 FORMAT(2I5)
022AP      WRITE(IPR,2500)J,K
024AP      2500 FORMAT(' IH=',I5,' IA=',I5)
026AP      I=UFA(1)=J
027AP      I=UFA(2)=K
028AP      WRITE(2*1)IPUFB
028AP      PRINT(2*1)IPUFA
028AP      IF(I=UFA(1).EQ.0 .AND. IPUFA(2).EQ.0) GO TO 44
028AP      PRINT ' FLOPPY DISK HAS NOT BEEN MOUNTED ON, PLEASE DO IT'
028AP      GO TO 12
028AP      C =====
028AP      44 I UFA(1)=

```


USER=TUNG 255 84170 JOINT COMPUTER FACILITY, MIT PAGE 1

```

0004P      SUBROUTINE HEADER
002AP      C THIS SUB ROUTE 24-WORD HEADER BLOCK ON TO HEADER FILE
002AP      C NOTIFY THE USER WHEN THE LAST RECORD IS USED
002AP      C IH: PTR TO THE HRD FILE, IT POINTS TO THE NEXT BLANK RECORD
002AP      C ITH: TOTAL # OF RECORDS IN THE HRD FILE
002AP      C NH: TOTAL # OF BLANK RECORDS LEFT IN THE HRD FILE
002AP      C COMMON FILE
002AP      IMPLICIT INTEGER*2 (I-N)
002AP      INTEGER*2 PERIOD
002AP      REAL LBOUND, IT, INPUT(10)
002AP      LOGICAL EQUAL
002AP      COMMON ITA, ITH, IA, IH, NS, IIS, NP, IERROR, IDATE, ITIME, ID,
002AP      1 IREFER, IPTH, NP, NT, HROUND, LBOUND, IPTSET, IPTCAL,
002AP      1 JMON, JDAY, JYR, JHP, JMIN, JSEC, JMLSEC, JDATE,
002AP      1 PERIOD, IDURA, IOLD, IMON, IDAY, IYR, IHP, IMIN, ISEC, IMLSEC,
002AP      1 ITEMP, IREFLOW, IBRE, BFLOW, TEMPI, ICTEMP, TEMPIC,
002AP      1 REA, DE, AMPL, IACK, IN, IT,
002AP      1 VSET, VCAL, VMIX, NUM, IQUIT,
002AP      1 MAXSET,
002AP      1 VSET(2,24), VCHAN(2,512), IHELP,
002AP      1 IBUFD(512), IF(10,2), LEVELS(100), IMIX(5), ICAL(5),
002AP      1 IBUFC(512), BUFR(512),
002AP      1 IBUFA(24), IBUFB(512), NAME(5), ICMD(2),
002AP      1 ICN, IPR, IECHO,
002AP      1 IRMAIN, IPNT,
002AP      1 ICON, ISUB, NAVG, SLFLOW, IPROH, IPROL, IJFLOW, NAVG,
002AP      1 VL, NLEVEL, ICMD, ICCON,
002AP      1 VAINS(60), IMTIME, ICPCAL
002AP      C END OF COMMON FILE
002AP      WRITE(2,'IN') IBUFA
004AP
004AP      C UPDATE IH,
004AP      1 I=IH+1
005AP      C RECOMPUTE N
005AP      N=ITH-IH+1
006AP      IF(IH.EQ.1.AND. IECHO.NE.IPR) WRITE(IPR,15)
006AP      15 FORMAT(2,'ONLY 1 HDR BLOCK LEFT')
007AP      RETURN
007AP
1505[1] //      0000[V] IT      0002[V] ITF      0004[V] IA      0006[V] IH
0008[V] IC      0008[V] IIF      0008[V] NP      0008[V] IERROR      0010[V] IDATE
0012[V] ITIME      0014[V] ID      0016[V] IREFER      0018[V] IPTH      001A[V] NR
001C[V] NT      001E[V] HROUND      0022[V] LBOUND      0026[V] IPTSET      0028[V] IPTCAL
002A[V] JMON      002C[V] JDAY      002E[V] JYR      0030[V] JHP      0032[V] JMIN
0034[V] JSEC      0036[V] JMLSEC      0038[V] JDATE      003A[V] PERIOD      003C[V] IDURA
003E[V] IOLD      0040[V] IMON      0042[V] IDAY      0044[V] IYR      0046[V] IHP
0048[V] IMIN      004A[V] ISEC      004C[V] IMLSEC      004E[V] ITEMP      0050[V] IREFLOW
0052[V] IBRE      0054[V] BFLOW      0056[V] TEMPI      0058[V] ICTEMP      005A[V] TEMPIC
005C[V] REA      005E[V] DE      0060[V] AMPL      0062[V] IACK      0064[V] IN
0066[V] IT      0068[V] VSET      006A[V] VMIX      006C[V] NUM      006E[V] IQUIT
0070[V] MAXSET      0072[V] VSET      0074[V] VCHAN      0076[V] IHELP
0078[V] IBUFD      007A[V] IF      007C[V] LEVELS      007E[V] IMIX
0080[V] IBUFC      0082[V] BUFR      0084[V] IBUFA      0086[V] IBUFB
0088[V] NAME      008A[V] ICMD      008C[V] ICN      008E[V] IPR
0090[V] IECHO      0092[V] IRMAIN      0094[V] IPNT      0096[V] ICON
0098[V] ISUB      009A[V] NAVG      009C[V] SLFLOW      009E[V] IPROH
00A0[V] IPROL      00A2[V] IJFLOW      00A4[V] NAVG      00A6[V] VL
00A8[V] NLEVEL      00AA[V] ICMD      00AC[V] ICCON      00AE[V] VAINS
00B0[V] IMTIME      00B2[V] ICPCAL

```

2 USER=TUNG 255 24170 JOINT COMPUTER FACILITY, MIT PAGE 2

1E22[V] ICCAD	1E26[V] IC	1E28[V] IEP	1E2A[V] IECHO	1ECC[V] IPMAIN
1E2L[V] IPRT	1E30[V] IACON	1E32[V] IST-	1E34[V] NAVG	1E36[V] SLFLOW
1E38[V] IRTON	1E3C[V] IAPOL	1E3E[V] IJELC	1E40[V] MARG	1E42[V] NL
1E44[V] ILEVEL	1E46[V] IC D	1E48[V] ICCM	1E4A[V] MAINS	1E4C[V] INTIME
1E4E[V] IOTCAL	0024[V] HJ DER	0026[V] HJ DER	0028[V] .C	002A[V] .P
002C[V] INEUT	002E[V] LOTAL	0030[V] EG	0100[V] NW	0036[V] 15
0000[S] DI				

THIS PROGRAM WILL OCCUPY X'0102' BYTES OF STORAGE SPACES

PROGRAM HEADER HAS NO ERRORS

USER=TUNG 255 04170 JOINT COMPUTER FACILITY, MIT PAGE 2

0062[V] AREA	0066[V] DE	006A[V] AMPL	006E[V] IASK	0070[V] IN
0072[V] IT	0076[V] NCET	0078[V] NCAL	007A[V] NMIV	007C[V] NUM
007E[V] IQUIT	0080[V] MAXSET	0082[V] MSTT	008E[V] MCHAN	0092[V] IMFLP
0094[V] IBUFD	0098[V] IP	009C[V] LEVELS	009E[V] IMIX	00DE[V] ICAL
009F[V] IBUFC	1122[V] BUFB	19E8[V] IBUFA	1A18[V] IBUFB	1E18[V] NAME
1E22[V] ICOMD	1E26[V] ICV	1E28[V] IPR	1E2A[V] IECHO	1E2C[V] IMYAIN
1E2E[V] IPNT	1E30[V] IMCON	1E32[V] ISUB	1E34[V] NAVG	1E36[V] SLFLOW
1E3A[V] IPROH	1E3C[V] IPPOL	1E3E[V] IJFLOW	1E40[V] NAVG	1E42[V] NL
1E44[V] VLEVEL	1E46[V] ICOMD	1E48[V] ICCON	1E4A[V] MAINS	1E4C[V] IMTMR
1E4E[V] ICRCAL	0024[E] DATAR	01C2[R] DATAR	0000[S] .Q	0000[S] .P
01CA[V] INPUT	01E2[V] EQUAL	01E6[V] IREST	0000[S] MOD	008C[L] 10
01FC[V] J	0200[V] K	006C[L] S	0202[V] I	0106[L] 20
00B4[L] 8	0000[S] 2G	0206[V] N	0208[V] NA	0150[L] 15
0000[S] 9I				

THIS PROGRAM WILL OCCUPY X'020C' BYTES OF STORAGE SPACE
 PROGRAM DATAR PLS NO ERRORS

2 USER=TV13 255 84170 JOINT COMPUTER FACILITY, MIT PAGE 2

1E22[V] ICMD	1E24[V] ICN	1E28[V] IPP	1E2A[V] IECHO	1E2C[V] IRMLN
1E2E[V] IPST	1E30[V] INCON	1E32[V] ISU	1E34[V] NAVE	1E36[V] SLPLOW
1E38[V] IPPON	1E3C[V] IPOL	1E3E[V] INELOW	1E40[V] NAVE	1E42[V] NL
1E44[V] NLEVEL	1E46[V] ICMD	1E48[V] ICCON	1E4A[V] MAINS	1E4C[V] IMTMT
1E4E[V] ICECAL	0024[E] SAVER	002E[E] SAVER	00C0[S] .C	00C0[S] .P
00C6[V] INPUT	003E[V] EQUAL	0042[I] 130	00F2[V] I	00C0[S] 0G

THIS PROGRAM WILL OCCUPY X'00F0' BYTES OF STORAGE SPACE

PROGRAM SAVED WAS NO ERRORS

// FOR

TUNG-LIB
(Library)

USER=TUNG 255 34170 JOINT COMPUTER FACILITY, MIT PAGE 2

```

00FCR
00FCR      13 GO TO (90,100,200,300,400,500,550,600),I
011CR
011CR      C C(CHANGE):
011CR      90 IN=1
0124R      CALL PRINT
012AR      IF(EQUAL(NAME,0,'Q',0,1)) RETURN
014ER      GO TO 2
0152R
0152R      C P(PRINT):
0152R      100 IN=0
015AR      CALL PRINT
0160R      IF(EQUAL(NAME,0,'Q',0,1)) RETURN
0184R      GO TO 2
0188R
0188R      C L(LIST):
0188R      200 CALL LIST
0188R      GO TO 2
0192R
0192R      C T(TEXT DISPLAY):
0192R      300 CALL TEXT
0192R      GO TO 2
019CR
019CR      C I(INPUT TEXT):
019CR      400 CALL INTEXT
01A2R      GO TO 2
01A6R
01A6R      C R(REMAIN):
01A6R      500 I-LEFT=ITR-IN+1
01A6R      I-L-RT=ITA-IA+1
01C6R      WRITE(IPR,605)I-LEFT,INLEFT
01E4R      505 FORMAT(5X,'EMPTY DATABLOCKS LEFT:',I5,/,
01E4R      14X,'EMPTY HEADERBLOCKS LEFT:',I5)
0234R      GO TO 2
0238R
0238R      C Q(QUIT)
0238R      550 RETURN
0240R
0240R      C I(TEXT):
0240R      600 WRITE(IPR,605)NSET,NCAL,NMIX,IN,IA
027CR      605 FORMAT(2X,'NEXT SET:',I2,/,
027CR      1      1X,'NEXT CAL:',I2,/,
027CR      1      1X,'NEXT MIX:',I2,/,
027CR      1      1X,'NEXT HDS:',I5,/,
027CR      1      1X,'NEXT DATER:',I5)
02E4R      602 S,0,' INDT:NSET,NCAL,NMIX,IN,IA'
031CR      READ(ICV,606)NSET,NCAL,NMIX,IN,IA
0358R      606 FORMAT(3I5)
0364R      GO TO 2
0368R
1-2-70 [C] // 0001 [V] ITR 0002 [V] ITR 0004 [V] IA 0006 [V] IN
0001 [V] NR 0001 [V] ITR 0002 [V] NR 0003 [V] ITRPOP 0004 [V] ITRTE
0001 [V] ITRTE 0001 [V] ITR 0002 [V] ITRTE 0003 [V] ITRTE 0004 [V] NR

```

UCL-17 JOINT COMPUTER FACILITY, MIT PAGE 3

[illegible]

FILE PROGRAM WILL OCCUR. A/CBAC' BYTES OF STORAGE SPACE.

7-25-53 10:11 AM

USER=TUNG 255 84170 JOINT COMPUTER FACILITY, MIT PAGE 1

SUBROUTINE LIST

```

00048      C SUBR LIST IS CALLED WHEN 'L' IS ENTERED AS THE REQUEST UNDER COMMAND
002A7      C 'EDT' IT LISTS THE NAMES OF ALL THE HEADER BLOCKS IN THE CURRENT
002A8      C FLOPPY DISC.
002A9      C COMMON FILE
002AB      IMPLICIT INTEGER*2 (I-N)
002AC      INTEGER*2 PERIOD
002AD      REAL LBOUND,IT,INPUT(10)
002AE      LOGICAL EQUAL
002AF      COMMON   ITA,ITH,IA,IN,NS,IIS,NP,IERROR,IDATE,ITIME,ID,
002B0      1         IREFER,IPTH,WR,NT,HBOUND,LBOUND,IPTSET,IPTCAL,
002B1      1         JMON,JDAY,JYR,JHR,JMIN,JSEC,JMLSEC,JDATE,
002B2      1         PERIOD,IDURA,IOLD,IMON,IDAY,IYR,IHP,IMIN,ISEC,IWLSEC,
002B3      1         INTMP,INFLOW,IBRE,BFLOW,TEMPIN,ICTEMP,TEMPIC,
002B4      1         AREA,DE,AMPL,IASK,IN,IT,
002B5      1         NSET,NCAL,NHIX,NUM,IQUIT,
002B6      1         MAXSET,
002B7      1         NSET(2,24),MCHAN(2,512),INELP,
002B8      1         TRUED(512),IP(10,2),LEVELS(100),IMIN(5),ICAL(5),
002B9      1         IPUFC(512),PUFR(512),
002BA      1         IPUFA(24),IPUFB(512),NAME(5),ICMD(2),
002BB      1         ICN,IPR,IECHO,
002BC      1         IRKIN,IPNT,
002BD      1         IACON,ISUB,NAVG,SLFLOW,IPROH,IPROL,IJFLOW,NAVG,
002BE      1         NL,NLEVEL,ICMD,ICCON,
002BF      1         IAINS(60),INTIME,ICRCAL
002C0      C END OF COMMON FILE
002C1      DIMENSION NAMEA(5),NAMEB(5)
002C2      IF(IP.NE.2) GO TO 5
002C3      PUT 5,0,' THERE EXISTS NO BLOCKS IN THE FILES'
002C4      RETURN
002C5      5  IFILL=0
002C6      IFILL=0
002C7      NH=IN-1
002C8      DO 100 I=2,NH
002C9      READ(2,I)IBUFA
002CA      IF(IFILL.NE.0) GO TO 10
002CB      IFILL=1
002CC      CALL COPY(IBUFA,0,NAMEA,0,10)
002CD      GO TO 100
002CE      10  IF(EQUAL(IBUFA,0,NAMEA,0,6)) GO TO 30
002CF      IF(IFILL.NE.0) GO TO 20
002D0      WRITE(IPR,15)NAMEA
002D1      15  FORMAT(5X,5A2)
002D2      GO TO 40
002D3      20  WRITE(IPR,25)NAMEA,NAMEB
002D4      25  FORMAT(5X,5A2,' TO ',5A2)
002D5      IFILL=0
002D6      GO TO 40
002D7      30  CALL COPY(IREFER,0,NAMEB,0,10)
002D8      IFILL=1
002D9      GO TO 100
002DA      40  CALL COPY(IBUFA,0,NAMEA,0,10)

```


0 USER=TOJIS 235 04170 JOINT COMPUTER FACILITY, VIT PAGE 2

1E35[C] //	0000[V] IP	0002[V] ITH	0004[V] IA	0006[V] IH
0008[V] IS	0010[V] ITB	0012[V] IP	0014[V] IERROR	0016[V] IDATE
0018[V] ITD	0020[V] IP	0022[V] IREPER	0024[V] IDTH	0026[V] IP
0028[V] IT	0030[V] IREPER	0032[V] LCHND	0034[V] IDTSET	0036[V] IPTCLL
0038[V] JMON	0040[V] IDT	0042[V] JVB	0044[V] JHR	0046[V] JMI
0048[V] JSEC	0050[V] JMLSEC	0052[V] JDATE	0054[V] PERIOD	0056[V] IDUR
0058[V] IOLD	0060[V] IDAY	0062[V] IDAY	0064[V] IYR	0066[V] IHR
0068[V] IML	0070[V] ITG	0072[V] IMLSEC	0074[V] IMTMB	0076[V] IBFLOW
0078[V] ISCT	0080[V] IFLGV	0082[V] TEMPL	0084[V] ICTMB	0086[V] TEMPLIC
0088[V] AREA	0090[V] OR	0092[V] AMPL	0094[V] TASK	0096[V] IN
0098[V] IT	0100[V] ISET	0102[V] NCAL	0104[V] NMIV	0106[V] NUM
0108[V] IQUIT	0110[V] ISET	0112[V] MSIT	0114[V] YCHAN	0116[V] IHELP
0118[V] IBUF	0120[V] ID	0122[V] LEVELS	0124[V] IMIV	0126[V] ICAL
0128[V] IBUEC	0130[V] IPR	0132[V] IBUEA	0134[V] IFUER	0136[V] NAME
0138[V] ICCID	0140[V] IPR	0142[V] IPR	0144[V] ICHO	0146[V] IRMTH
0148[V] IPR	0150[V] IPR	0152[V] ISUE	0154[V] NAVG	0156[V] SLFLOW
0158[V] IPR	0160[V] IPR	0162[V] IPR	0164[V] NAVG	0166[V] NL
0168[V] NLLEVEL	0170[V] IPR	0172[V] ICCID	0174[V] MAINS	0176[V] INTIME
0178[V] ICCAL	0180[V] IPR	0182[V] INTXT	0184[V] .C	0186[V] .P
0188[V] IPR	0190[V] IPR	0192[V] ISLAN	0194[V] 10	0196[V] I
0198[V] IPR	0200[V] IPR	0202[V] IPR	0204[V] 15	0206[V] BI
0208[V] IPR	0210[V] IPR	0212[V] IPR	0214[V] 70	0216[V] SG

1.10 PROGRAM WILL OCCUPY 210 BYTES OF STORAGE SPACE
 2.10 INTXT AND 10 BYTES

2 UCIR=TUNG 255 34170 JOINT COMPUTER FACILITY, MIT PAGE 2

0122[V] INPUT 014 [V] L151L 0000[S] 05 0150[V] N 0056[L] 80
0056[L] 85 0000[S] 07

THIS PROGRAM WILL OCCUPY 1154 BYTES OF STORAGE SPACE
PROGRAM TEXT HAS NO ERRORS

USER=TIME 055 44170 JOINT COMPUTER FACILITY, MIT PAGE 2

```

001      GO TO 2
002
003      C USER IS TO DEAL WITH THE WRD INFOR. OF THE DATA OF
004      C THE PREVIOUS LOG (OF BOTH)
005      C READ THIS LOG IN TO COPY INFO
006      40 READ(7,100)INFO
007
008      C ASK USER IF HE WANTS TO DEAL WITH THE WRD INFOR. OF THE DATA OF
009      C THE PREVIOUS LOG (OF BOTH)
010      42 PRINT(100,45)
011      45 PRINT(20,'Y/N/D/L: ',/)
012      READ(100,46) INFO
013      46 PRINT(42)
014
015      I (-,1,1,1,1,1,1,1,1,1) GO TO 100
016      I (-,1,1,1,1,1,1,1,1,1) GO TO 200
017      I (-,1,1,1,1,1,1,1,1,1) GO TO 300
018      PRINT(47,7) INFO
019      47 PRINT(7,7) INFO OF DISPOSE
020
021      C USER IS TO DEAL WITH WRD INFOR ONLY
022      100 INFO=
023      105 IF(100) 110,110,120
024
025      110 CALL INFO (INFO,10)
026      120 CALL INFO (INFO,11)
027
028      200 INFO=1
029      GO TO 100
030
031      C SUBROUTINE INFO WILL DEAL WITH INFO (C INFO), OR (PRINT) SERVICE
032      300 INFO=PRINT
033      305 INFO=
034
035
036
037
038
039
040
041
042
043
044
045
046
047
048
049
050
051
052
053
054
055
056
057
058
059
060
061
062
063
064
065
066
067
068
069
070
071
072
073
074
075
076
077
078
079
080
081
082
083
084
085
086
087
088
089
090
091
092
093
094
095
096
097
098
099
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

```

USER=THAI 155 24176 JOINT COMPUTER FACILITY, MIT PAGE 3

1E44[V] NLEVEL	1E44[V] ICMD	1E44[V] ICCOM	1E44[V] MAINS	1E02[V] INTIME
1E04[V] ICPCAL	0024[T] PRINT	0302[S] PRINT	0000[S] .Q	0000[S] .P
0304[V] INPUT	0000[S] MANUAL	002E[V] IBIANY	0030[L] 2	0044[L] 4
0000[S] DI	0064[L] 3	0302[V] I	00A8[L] 5	0177[L] 30
0134[L] 20	0134[L] 8	0158[L] 15	0000[S] MOVE	0000[S] SEARCH
021E[L] 40	01F4[L] 35	0000[S] 26	023F[L] 42	0252[L] 45
0284[L] 46	0300[V] IANS	0332[L] 100	0360[L] 200	0360[L] 300
0304[L] 47	033 [L] 105	034C[L] 110	0356[L] 120	0000[S] LINKED
0000[S] PRINTD				

THIS PROGRAM WILL OCCUPY X' 3C6' BYTES OF STORAGE SPACE
PROGRAM PRINT HAS NO ERRORS

USER=TUN 235 84170 JOINT COMPUTER FACILITY, MIT PAGE 2

08E4[V] IEUED	0CE4[V] IP	0D0C[V] LEVELS	0DD4[V] IMIV	0DDE[V] ICAL
0DE8[V] IBUEC	11E8[V] IJFR	1078[V] IBUEA	1118[V] IFUEB	1E18[V] NAME
1E22[V] ICCMD	1E20[V] IC	1E28[V] IPR	1E2A[V] IECHO	1E2C[V] IRMIN
1E2E[V] IPNT	1 30[V] IMCON	1E32[V] ISU	1E34[V] MAVG	1E36[V] SLEFLOW
1E3A[V] IPPOH	1 37[V] IEFOL	1E3E[V] IJLOW	1E40[V] YAVG	1E42[V] NL
1E44[V] NLEVEL	1E46[V] ICMD	1E48[V] ICCON	1E4A[V] MAINS	1E4C[V] INTIME
1E4E[V] ICRCAL	0024[L] PRINTS	01C2[R] PRINTS	0000[S] .0	0000[S] .P
01C4[V] INPUT	01E2[V] EQUAL	0032[L] 4	00D2[L] 16	0000[S] 2I
010C[L] 17	012E[L] 1	015C[L] 19	0152[L] 40	

THIS PROGRAM WILL OCCUPY X'01E8' BYTES OF STORAGE SPACE
 PROGRAM PRINTS HAS NO ERRORS

USER=TUNG 055 04170 JOINT COMPUTER FACILITY, MIT PAGE 2

1E44[V] NLEVEL	1044[V] ICND	1E4B[V] ICCDN	1E4A[V] MAINS	1E02[V] IMTIME
1E04[V] ICPCAL	0024[L] PRINTC	0104[E] PRINTC	0000[S] .Q	0000[S] .P
0100[V] INPUT	0134[V] EQUAL	0000[E] .W	0050[L] 4	0004[L] 16
0000[S] PI	0004[L] 17	0004[L] 40		

THIS PROGRAM WILL OCCUPY X'0138' BYTES OF STORAGE SPACE
 PROGRAM PRINTC HAS NO PEROPS

USER=TYNT 285 84170 JOINT COMPUTER FACILITY, MIT PAGE 1

```

0004-      CALLOWING PRINT*
002-      C SUBROUTINE PRINTM IS CALLED BY SUBR PRINT TO COMPLETE THE PRINTING OF
0027-      C MIN. FOLLOWING INFORMATION
002-      C COMMON FILE
002-      IMPLICIT INTEGER*2 (I-N)
002-      INT CLR*2 PERIOD
002-      REAL LBUND,IT,INPUT(10)
002-      LOGICAL EQUAL
002-      COMMON   ITA,ITH,IA,IH,NS,IIS,VP,TERPOR,IDATE,ITIME,ID,
002-      1        IP,FET,IPTH,VP,AT,HBOUND,LBOUND,ITSET,IPICAL,
002-      1        JMON,JDAY,JYR,JHR,JMIN,JSEC,JMLSEC,JDATE,
002-      1        PERIOD,IDURA,IOLD,IYON,IDAY,IYR,IHR,IYIN,ISEC,IMLSEC,
002-      1        INTMP,IBFLOW,IBFE,BELOW,TEMPIN,ICTEMP,TEMPIC,
002-      1        AREA,DE,AMPL,PACK,IN,IT,
002-      1        NSET,NCAL,NMIN,NUT,IOBIT,
002-      1        NRESET,
002-      1        GET(2,24),NCHAN(2,512),INFILE,
002-      1        BUEND(512),TE(10,2),LEVFLS(100),IXIN(5),ICPL(5),
002-      1        IUFEC(512),IUFER(512),
002-      1        IURAR(24),IUPTR(512),NAME(5),ICOND(2),
002-      1        ION,IPR,IECHO,
002-      1        IRRMIN,IPRT,
002-      1        IMCON,ISUB,NAVG,CIFLOW,IPROH,IPROL,IUFLOW,MAVG,
002-      1        AL,NLEVEL,ICAD,ICCON,
002-      1        RING(60),ITIME,ICPCAL
002-      C END OF COMMON FILE
002-      I=1
002-      C UNPACK THE INFO-ILLISFC WORD
002-      C *****
002-      TIME=IUFEC(15)
002-      II(PI*1.5708) TIME=TIME+65536.
002-      ICOND=II/1024.
002-      ICOND=1.0(IIF(1,1024.))
002-      C *****
002-      4  WHILE(IE,15) ISEC,IMLSEC
002-      1  TIME=IUFER(15)
002-      1  TIME=TIME/10.0
002-      1  WHILE(IE,16) IUFEC(17)
002-      1  TIME=TIME(17)
002-      1  IL(1.5708) I=PD+2.**16
002-      1  WHILE(IE,18) IE
002-      1  WHILE(IE,20) IUFER(20)
002-      1  WHILE(IE,21) IUFER(21)
002-      1  WHILE(IE,22) IUFER(22)
002-      1  WHILE(IE,23) IUFER(23)
002-      1  WHILE(IE,24) IUFER(24)
002-      1  WHILE
002-      C 8 - IF " " IF:
002-      1  IF (IE,1) 17  IF (SEC,ILLISFC) 17,74,1,74)
002-      1  IF (IE,1) 17  IF (IF(C) 17,55.0)
002-      1  IF (IE,1) 17  IF (IF(STATE(00)) 17,75)

```

JOINT COMPUTER FACILITY, MIT PAGE 2

02BFF	19	FORMAT(5X,'	19	BUNDLE RE	:',F8.0)
02BLE	20	FORMAT(5X,'	20	INJ SALT CON(PPM)	:',I5)
031CI	21	FORMAT(5X,'	21	SALT FLR'TE(ML/MIN)	:',I5)
034A	22	FORMAT(5X,'	22	INJ SU CHANN	:',I5)
037CI	23	FORMAT(5X,'	23	AXIAL LOC(IN/100)	:',I5)
03A6L	24	FORMAT(5X,'	24	CAL CORRECT BLOCK	:',I5,/))
03D6F		END			

0006[V] NS	0007[V] IIS	0008[V] ITH	0009[V] NP	0010[V] IERROR	0011[V] IDATE
0012[V] ITIME	0013[V] ID	0014[V] IREFEP	0015[V] IPTH	0016[V] IPTSET	0017[V] IPTCAL
0018[V] JMON	0019[V] JBOUND	0020[V] JBOUND	0021[V] JYR	0022[V] JHP	0023[V] JMIN
0024[V] JSEC	0025[V] JDATE	0026[V] JDATE	0027[V] JPERIOD	0028[V] JIDUFA	0029[V] JIDUFA
0030[V] JIDU	0031[V] JIDU	0032[V] JIDU	0033[V] JIDU	0034[V] JIDU	0035[V] JIDU
0036[V] JIDU	0037[V] JIDU	0038[V] JIDU	0039[V] JIDU	0040[V] JIDU	0041[V] JIDU
0042[V] JIDU	0043[V] JIDU	0044[V] JIDU	0045[V] JIDU	0046[V] JIDU	0047[V] JIDU
0048[V] JIDU	0049[V] JIDU	0050[V] JIDU	0051[V] JIDU	0052[V] JIDU	0053[V] JIDU
0054[V] JIDU	0055[V] JIDU	0056[V] JIDU	0057[V] JIDU	0058[V] JIDU	0059[V] JIDU
0060[V] JIDU	0061[V] JIDU	0062[V] JIDU	0063[V] JIDU	0064[V] JIDU	0065[V] JIDU
0066[V] JIDU	0067[V] JIDU	0068[V] JIDU	0069[V] JIDU	0070[V] JIDU	0071[V] JIDU
0072[V] JIDU	0073[V] JIDU	0074[V] JIDU	0075[V] JIDU	0076[V] JIDU	0077[V] JIDU
0078[V] JIDU	0079[V] JIDU	0080[V] JIDU	0081[V] JIDU	0082[V] JIDU	0083[V] JIDU
0084[V] JIDU	0085[V] JIDU	0086[V] JIDU	0087[V] JIDU	0088[V] JIDU	0089[V] JIDU
0090[V] JIDU	0091[V] JIDU	0092[V] JIDU	0093[V] JIDU	0094[V] JIDU	0095[V] JIDU
0096[V] JIDU	0097[V] JIDU	0098[V] JIDU	0099[V] JIDU	0100[V] JIDU	0101[V] JIDU
0102[V] JIDU	0103[V] JIDU	0104[V] JIDU	0105[V] JIDU	0106[V] JIDU	0107[V] JIDU
0108[V] JIDU	0109[V] JIDU	0110[V] JIDU	0111[V] JIDU	0112[V] JIDU	0113[V] JIDU
0114[V] JIDU	0115[V] JIDU	0116[V] JIDU	0117[V] JIDU	0118[V] JIDU	0119[V] JIDU
0120[V] JIDU	0121[V] JIDU	0122[V] JIDU	0123[V] JIDU	0124[V] JIDU	0125[V] JIDU
0126[V] JIDU	0127[V] JIDU	0128[V] JIDU	0129[V] JIDU	0130[V] JIDU	0131[V] JIDU
0132[V] JIDU	0133[V] JIDU	0134[V] JIDU	0135[V] JIDU	0136[V] JIDU	0137[V] JIDU
0138[V] JIDU	0139[V] JIDU	0140[V] JIDU	0141[V] JIDU	0142[V] JIDU	0143[V] JIDU
0144[V] JIDU	0145[V] JIDU	0146[V] JIDU	0147[V] JIDU	0148[V] JIDU	0149[V] JIDU
0150[V] JIDU	0151[V] JIDU	0152[V] JIDU	0153[V] JIDU	0154[V] JIDU	0155[V] JIDU
0156[V] JIDU	0157[V] JIDU	0158[V] JIDU	0159[V] JIDU	0160[V] JIDU	0161[V] JIDU
0162[V] JIDU	0163[V] JIDU	0164[V] JIDU	0165[V] JIDU	0166[V] JIDU	0167[V] JIDU
0168[V] JIDU	0169[V] JIDU	0170[V] JIDU	0171[V] JIDU	0172[V] JIDU	0173[V] JIDU
0174[V] JIDU	0175[V] JIDU	0176[V] JIDU	0177[V] JIDU	0178[V] JIDU	0179[V] JIDU
0180[V] JIDU	0181[V] JIDU	0182[V] JIDU	0183[V] JIDU	0184[V] JIDU	0185[V] JIDU
0186[V] JIDU	0187[V] JIDU	0188[V] JIDU	0189[V] JIDU	0190[V] JIDU	0191[V] JIDU
0192[V] JIDU	0193[V] JIDU	0194[V] JIDU	0195[V] JIDU	0196[V] JIDU	0197[V] JIDU
0198[V] JIDU	0199[V] JIDU	0200[V] JIDU	0201[V] JIDU	0202[V] JIDU	0203[V] JIDU
0204[V] JIDU	0205[V] JIDU	0206[V] JIDU	0207[V] JIDU	0208[V] JIDU	0209[V] JIDU
0210[V] JIDU	0211[V] JIDU	0212[V] JIDU	0213[V] JIDU	0214[V] JIDU	0215[V] JIDU
0216[V] JIDU	0217[V] JIDU	0218[V] JIDU	0219[V] JIDU	0220[V] JIDU	0221[V] JIDU
0222[V] JIDU	0223[V] JIDU	0224[V] JIDU	0225[V] JIDU	0226[V] JIDU	0227[V] JIDU
0228[V] JIDU	0229[V] JIDU	0230[V] JIDU	0231[V] JIDU	0232[V] JIDU	0233[V] JIDU
0234[V] JIDU	0235[V] JIDU	0236[V] JIDU	0237[V] JIDU	0238[V] JIDU	0239[V] JIDU
0240[V] JIDU	0241[V] JIDU	0242[V] JIDU	0243[V] JIDU	0244[V] JIDU	0245[V] JIDU
0246[V] JIDU	0247[V] JIDU	0248[V] JIDU	0249[V] JIDU	0250[V] JIDU	0251[V] JIDU
0252[V] JIDU	0253[V] JIDU	0254[V] JIDU	0255[V] JIDU	0256[V] JIDU	0257[V] JIDU
0258[V] JIDU	0259[V] JIDU	0260[V] JIDU	0261[V] JIDU	0262[V] JIDU	0263[V] JIDU
0264[V] JIDU	0265[V] JIDU	0266[V] JIDU	0267[V] JIDU	0268[V] JIDU	0269[V] JIDU
0270[V] JIDU	0271[V] JIDU	0272[V] JIDU	0273[V] JIDU	0274[V] JIDU	0275[V] JIDU
0276[V] JIDU	0277[V] JIDU	0278[V] JIDU	0279[V] JIDU		

COPIES

USER=TUNG 255 04170 JOINT COMPUTER FACILITY, MIT PAGE 2

```

02005      IB=0
02006      DO 300 I=ITS,IES
02007      ID=I +1
02008      IS=I+127
02009      IF(I.E..IES) IS=NP
02010      READ(3'I')(IPUFA(N),N=IB,IE)
03040      200 CONTINUE
03041      WRITE(IPR,305)(IPUFA(N),N=1,5)
03042      305 FORMAT(5X,'DATABLOCK:',5X,5A2)
03061      IF(EQUAL(IPUFA,0,'S',0,1)) WRITE(IPR,315)
03062      IF(.NOT.EQUAL(IPUFA,0,'S',0,1)) WRITE(IPR,325)
04061      315 FORMAT(5X,'PROBE CHANNEL #S:')
04062      325 FORMAT(5X,'PROBE RESISTANCES:')
04063      IF(I) 330,330,350
04064      330 WRITE(IPR,335)(IPUFA(N),N=1,NP)
04065      335 FORMAT(5X,5I7)
04066      340 IF(I) 340,340,350
04067      340 IF(I) 340,340,350
04068      350 WRITE(IPR,355)('LIST TO CHANGE DAT IN BUFFER I=IB,IE)
04069      IB=0
04070      DO 300 I=ITS,IES
04071      ID=I+1
04072      IS=I+127
04073      IF(I.E..IES) IS=NP
04074      WRITE(3'I')(IPUFA(N),N=IB,IE)
04075      400 CONTINUE
04076      RETURN

```

1000 [V] //	000 [V] IT	0000 [V] ITL	0004 [V] IT	0006 [V] IT
000 [V] IT	000 [V] IT	0000 [V] IP	0002 [V] IP	0010 [V] IP
0012 [V] IT	0014 [V] IT	0016 [V] IP	0018 [V] IP	0020 [V] IP
0012 [V] IT	0014 [V] IT	0016 [V] IP	0018 [V] IP	0020 [V] IP
0022 [V] IT	0024 [V] IT	0026 [V] IP	0028 [V] IP	0030 [V] IP
0032 [V] IT	0034 [V] IT	0036 [V] IP	0038 [V] IP	0040 [V] IP
0042 [V] IT	0044 [V] IT	0046 [V] IP	0048 [V] IP	0050 [V] IP
0052 [V] IT	0054 [V] IT	0056 [V] IP	0058 [V] IP	0060 [V] IP
0062 [V] IT	0064 [V] IT	0066 [V] IP	0068 [V] IP	0070 [V] IP
0072 [V] IT	0074 [V] IT	0076 [V] IP	0078 [V] IP	0080 [V] IP
0082 [V] IT	0084 [V] IT	0086 [V] IP	0088 [V] IP	0090 [V] IP
0092 [V] IT	0094 [V] IT	0096 [V] IP	0098 [V] IP	0100 [V] IP
0102 [V] IT	0104 [V] IT	0106 [V] IP	0108 [V] IP	0110 [V] IP
0112 [V] IT	0114 [V] IT	0116 [V] IP	0118 [V] IP	0120 [V] IP
0122 [V] IT	0124 [V] IT	0126 [V] IP	0128 [V] IP	0130 [V] IP
0132 [V] IT	0134 [V] IT	0136 [V] IP	0138 [V] IP	0140 [V] IP
0142 [V] IT	0144 [V] IT	0146 [V] IP	0148 [V] IP	0150 [V] IP
0152 [V] IT	0154 [V] IT	0156 [V] IP	0158 [V] IP	0160 [V] IP
0162 [V] IT	0164 [V] IT	0166 [V] IP	0168 [V] IP	0170 [V] IP
0172 [V] IT	0174 [V] IT	0176 [V] IP	0178 [V] IP	0180 [V] IP
0182 [V] IT	0184 [V] IT	0186 [V] IP	0188 [V] IP	0190 [V] IP
0192 [V] IT	0194 [V] IT	0196 [V] IP	0198 [V] IP	0200 [V] IP
0202 [V] IT	0204 [V] IT	0206 [V] IP	0208 [V] IP	0210 [V] IP
0212 [V] IT	0214 [V] IT	0216 [V] IP	0218 [V] IP	0220 [V] IP
0222 [V] IT	0224 [V] IT	0226 [V] IP	0228 [V] IP	0230 [V] IP
0232 [V] IT	0234 [V] IT	0236 [V] IP	0238 [V] IP	0240 [V] IP
0242 [V] IT	0244 [V] IT	0246 [V] IP	0248 [V] IP	0250 [V] IP
0252 [V] IT	0254 [V] IT	0256 [V] IP	0258 [V] IP	0260 [V] IP
0262 [V] IT	0264 [V] IT	0266 [V] IP	0268 [V] IP	0270 [V] IP
0272 [V] IT	0274 [V] IT	0276 [V] IP	0278 [V] IP	0280 [V] IP
0282 [V] IT	0284 [V] IT	0286 [V] IP	0288 [V] IP	0290 [V] IP
0292 [V] IT	0294 [V] IT	0296 [V] IP	0298 [V] IP	0300 [V] IP
0302 [V] IT	0304 [V] IT	0306 [V] IP	0308 [V] IP	0310 [V] IP
0312 [V] IT	0314 [V] IT	0316 [V] IP	0318 [V] IP	0320 [V] IP
0322 [V] IT	0324 [V] IT	0326 [V] IP	0328 [V] IP	0330 [V] IP
0332 [V] IT	0334 [V] IT	0336 [V] IP	0338 [V] IP	0340 [V] IP
0342 [V] IT	0344 [V] IT	0346 [V] IP	0348 [V] IP	0350 [V] IP
0352 [V] IT	0354 [V] IT	0356 [V] IP	0358 [V] IP	0360 [V] IP
0362 [V] IT	0364 [V] IT	0366 [V] IP	0368 [V] IP	0370 [V] IP
0372 [V] IT	0374 [V] IT	0376 [V] IP	0378 [V] IP	0380 [V] IP
0382 [V] IT	0384 [V] IT	0386 [V] IP	0388 [V] IP	0390 [V] IP
0392 [V] IT	0394 [V] IT	0396 [V] IP	0398 [V] IP	0400 [V] IP
0402 [V] IT	0404 [V] IT	0406 [V] IP	0408 [V] IP	0410 [V] IP
0412 [V] IT	0414 [V] IT	0416 [V] IP	0418 [V] IP	0420 [V] IP
0422 [V] IT	0424 [V] IT	0426 [V] IP	0428 [V] IP	0430 [V] IP
0432 [V] IT	0434 [V] IT	0436 [V] IP	0438 [V] IP	0440 [V] IP
0442 [V] IT	0444 [V] IT	0446 [V] IP	0448 [V] IP	0450 [V] IP
0452 [V] IT	0454 [V] IT	0456 [V] IP	0458 [V] IP	0460 [V] IP
0462 [V] IT	0464 [V] IT	0466 [V] IP	0468 [V] IP	0470 [V] IP
0472 [V] IT	0474 [V] IT	0476 [V] IP	0478 [V] IP	0480 [V] IP
0482 [V] IT	0484 [V] IT	0486 [V] IP	0488 [V] IP	0490 [V] IP
0492 [V] IT	0494 [V] IT	0496 [V] IP	0498 [V] IP	0500 [V] IP

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100

PROG. PRINT, AND TO EUROPE

2 UCLP=TU16 255 64170 JOINT COMPUTER FACILITY, MIT PAGE 2

0071(V) IQUIT	0072(V) ISET	0073(V) VSTT	0074(V) MCHAN	0075(V) IHEIP
0076(V) ISSFD	0077(V) LE	0078(V) LEVLE	0079(V) IMIX	0080(V) ICAT
0081(V) IASFC	0082(V) IJEP	0083(V) IBURA	0084(V) IPHEE	0085(V) WAC
0086(V) ICCAD	0087(V) IIN	0088(V) IPR	0089(V) ICHQ	0090(V) IRYIN
0091(V) IPPT	0092(V) ICDN	0093(V) ISUR	0094(V) WAGS	0095(V) SLFLOX
0096(V) IPBOH	0097(V) IPOL	0098(V) IJFLO	0099(V) WAGS	0100(V) NL
0091(V) NLEVEL	0101(V) ICID	0102(V) ICCON	0103(V) MAINS	0104(V) IMTIME
0105(V) ICPCAL	0106(V) EDTCHA	0107(V) EDTCHA	0108(S) .C	0109(S) .P
0110(V) INPUT	0111(S) EQUAL	0112(S) QUDATT	0113(V) JTIVE	0114(S) .I
0115(S) MOD	0116(S) CHS	0117(S) CHAC	0118(S) CHAY	0119(S) RG

THIS PROGRAM WILL OCCUPY X'01F4' BYTES OF STORAGE SPACE

PROGRAM EDTCHA HAS NO ERRORS

USER=TONG 255 24170 JOINT COMPUTER FACILITY, MIT PAGE 2

0025[V] IBUFC	1122[V] BUEB	1250[V] IBUEA	1A13[V] ISUFB	1E18[V] NAME
1E22[V] ICCID	1225[V] ICN	1E25[V] IPB	1E2A[V] IFCNO	1E2C[V] IRVIV
1E25[V] IPBT	113[V] IACON	1E2C[V] ISUP	1E24[V] XAVG	1E26[V] SLFLOW
1E35[V] IEPCH	1130[V] IPECL	1E27[V] IJFLOW	1E40[V] XAVG	1E42[V] VL
1E44[V] ILEVEL	1E46[V] ICND	1E48[V] ICCON	1E4A[V] MAINS	1E42[V] IATIME
1E44[V] ICPAL	0024[S] CHS	0154[S] CHAT	0000[S] .0	0000[S] .P
0163[V] INPUT	0124[V] EQUAL	0125[V] IPTYPE	006C[L] 15	0000[S] 24

THIS PROGRAM WILL OCCUPY X'019A' BYTES OF STORAGE SPACE

PROGRAM CHAS HAS NO ERRORS

2 JSTR=TIME 055-4170 JOINT COMPUTER FACILITY, MIT PAGE 2

1504[7] ICPAL 0024[7] CHAC 0104[7] CHAC 0000[S] .0 0000[S] .P
 0120[V] INPUT 015-1[7] CHAL 0000[S] FLOAT 0004[L] 15 0000[S] 2H
 0000[S] .Y

THIS PROGRAM WILL OCCUPY 11150 BYTES OF STORAGE SPACE
 PROGRAM CHAC 11150 BYTES NO ERRORS

2 USER=TUNT 255 84170 JOINT COMPUTER FACILITY, MIT PAGE 2

1E3A[VI] IPROH	1E37[VI] IPOL	1E3F[VI] IJELON	1E40[VI] MAVG	1E42[VI] NL
1E44[VI] MLIVEL	1E41[VI] ICND	1E4C[VI] ICCDN	1E4A[VI] MAINS	1E02[VI] INTIMP
1E04[VI] ICECAL	0024[VI] OUDATE	0004[VI] OUDATE	0000[SI] .Q	0000[SI] .P
0002[VI] IPST	0004[VI] EURL	000E[VI] IDIV	0000[SI] .I	0004[VI] JPST
0000[SI] NO				

THIS PROGRAM WILL OCCUPY 20000 BYTES OF STORAGE SPACES

PROGRAM OUDATE HAS NO ERRORS

```

SUBROUTINE SEARCH
002A  C COMMON FILE
002A  I EXPLICIT INTEGER*2 (I-N)
002A  INTEGER*2 PERIOD
002A  REAL LBOUND,IT,INPUT(10)
002A  LOGICAL EQUAL
002A  COMMON ITA,ITH,ITA,IPH,NS,IIS,MP,IEPROP,IDATE,ITIME,ID,
002A  1 IREFR,IPTH,NR,NT,HBOUND,LBOUND,IPTSFT,IPTCAL,
002A  1 JMON,JDAY,JYP,JHR,JMIN,JSEC,JMISEC,JDATE,
002A  1 PERIOD,IDURA,IOLD,JMON,IDAY,IYP,IHR,IMIN,ISEC,IMLSEC,
002A  1 INTMP,IBFLOW,IBDF,BFLOW,TEMPIN,ICTEMP,TEMPIC,
002A  1 APER,DF,AMPL,IASK,IN,IT,
002A  1 ISET,NCAL,NMAY,NUM,ICUIT,
002A  1 MAXSET,
002A  1 IUT(2,24),NCHAN(2,512),THFLP,
002A  1 IUPR(512),IT(10,2),LEVELS(100),IMIN(5),ICHI(5),
002A  1 IUFER(512),IUF=512,
002A  1 IUTR(24),IUFER(512),NANT(5),ICORD(2),
002A  1 TOP,IOP,ICHO,
002A  1 IFRIN,IRIT,
002A  1 IMCON,ISUT,NAVG,SLSLOW,IFROW,IPROL,IJFLOW,NAVG,
002A  1 AL,NLEVEL,ICHI,ICORD,
002A  1 NAINS(60),IMTIN,ICPCAL
002A  C END OF COMMON FILE
002A  IREFR=0
002A  N=NT-1
002A  DO 10 I=1,N
002A  10 I=2*I (IUF=0),I=1,24)
002A  IF (JLL(NAIN,0,IUF=0,100)) GO TO 20
002A  10 CONTINUE
002A  IOP=1
002A  20 I=1
002A  30 I=1
002A  40 I=1
002A  50 I=1
002A  60 I=1
002A  70 I=1
002A  80 I=1
002A  90 I=1
002A  100 I=1
002A  110 I=1
002A  120 I=1
002A  130 I=1
002A  140 I=1
002A  150 I=1
002A  160 I=1
002A  170 I=1
002A  180 I=1
002A  190 I=1
002A  200 I=1
002A  210 I=1
002A  220 I=1
002A  230 I=1
002A  240 I=1
002A  250 I=1
002A  260 I=1
002A  270 I=1
002A  280 I=1
002A  290 I=1
002A  300 I=1
002A  310 I=1
002A  320 I=1
002A  330 I=1
002A  340 I=1
002A  350 I=1
002A  360 I=1
002A  370 I=1
002A  380 I=1
002A  390 I=1
002A  400 I=1
002A  410 I=1
002A  420 I=1
002A  430 I=1
002A  440 I=1
002A  450 I=1
002A  460 I=1
002A  470 I=1
002A  480 I=1
002A  490 I=1
002A  500 I=1
002A  510 I=1
002A  520 I=1
002A  530 I=1
002A  540 I=1
002A  550 I=1
002A  560 I=1
002A  570 I=1
002A  580 I=1
002A  590 I=1
002A  600 I=1
002A  610 I=1
002A  620 I=1
002A  630 I=1
002A  640 I=1
002A  650 I=1
002A  660 I=1
002A  670 I=1
002A  680 I=1
002A  690 I=1
002A  700 I=1
002A  710 I=1
002A  720 I=1
002A  730 I=1
002A  740 I=1
002A  750 I=1
002A  760 I=1
002A  770 I=1
002A  780 I=1
002A  790 I=1
002A  800 I=1
002A  810 I=1
002A  820 I=1
002A  830 I=1
002A  840 I=1
002A  850 I=1
002A  860 I=1
002A  870 I=1
002A  880 I=1
002A  890 I=1
002A  900 I=1
002A  910 I=1
002A  920 I=1
002A  930 I=1
002A  940 I=1
002A  950 I=1
002A  960 I=1
002A  970 I=1
002A  980 I=1
002A  990 I=1
002A  1000 I=1
002A  1010 I=1
002A  1020 I=1
002A  1030 I=1
002A  1040 I=1
002A  1050 I=1
002A  1060 I=1
002A  1070 I=1
002A  1080 I=1
002A  1090 I=1
002A  1100 I=1
002A  1110 I=1
002A  1120 I=1
002A  1130 I=1
002A  1140 I=1
002A  1150 I=1
002A  1160 I=1
002A  1170 I=1
002A  1180 I=1
002A  1190 I=1
002A  1200 I=1
002A  1210 I=1
002A  1220 I=1
002A  1230 I=1
002A  1240 I=1
002A  1250 I=1
002A  1260 I=1
002A  1270 I=1
002A  1280 I=1
002A  1290 I=1
002A  1300 I=1
002A  1310 I=1
002A  1320 I=1
002A  1330 I=1
002A  1340 I=1
002A  1350 I=1
002A  1360 I=1
002A  1370 I=1
002A  1380 I=1
002A  1390 I=1
002A  1400 I=1
002A  1410 I=1
002A  1420 I=1
002A  1430 I=1
002A  1440 I=1
002A  1450 I=1
002A  1460 I=1
002A  1470 I=1
002A  1480 I=1
002A  1490 I=1
002A  1500 I=1
002A  1510 I=1
002A  1520 I=1
002A  1530 I=1
002A  1540 I=1
002A  1550 I=1
002A  1560 I=1
002A  1570 I=1
002A  1580 I=1
002A  1590 I=1
002A  1600 I=1
002A  1610 I=1
002A  1620 I=1
002A  1630 I=1
002A  1640 I=1
002A  1650 I=1
002A  1660 I=1
002A  1670 I=1
002A  1680 I=1
002A  1690 I=1
002A  1700 I=1
002A  1710 I=1
002A  1720 I=1
002A  1730 I=1
002A  1740 I=1
002A  1750 I=1
002A  1760 I=1
002A  1770 I=1
002A  1780 I=1
002A  1790 I=1
002A  1800 I=1
002A  1810 I=1
002A  1820 I=1
002A  1830 I=1
002A  1840 I=1
002A  1850 I=1
002A  1860 I=1
002A  1870 I=1
002A  1880 I=1
002A  1890 I=1
002A  1900 I=1
002A  1910 I=1
002A  1920 I=1
002A  1930 I=1
002A  1940 I=1
002A  1950 I=1
002A  1960 I=1
002A  1970 I=1
002A  1980 I=1
002A  1990 I=1
002A  2000 I=1
002A  2010 I=1
002A  2020 I=1
002A  2030 I=1
002A  2040 I=1
002A  2050 I=1
002A  2060 I=1
002A  2070 I=1
002A  2080 I=1
002A  2090 I=1
002A  2100 I=1
002A  2110 I=1
002A  2120 I=1
002A  2130 I=1
002A  2140 I=1
002A  2150 I=1
002A  2160 I=1
002A  2170 I=1
002A  2180 I=1
002A  2190 I=1
002A  2200 I=1
002A  2210 I=1
002A  2220 I=1
002A  2230 I=1
002A  2240 I=1
002A  2250 I=1
002A  2260 I=1
002A  2270 I=1
002A  2280 I=1
002A  2290 I=1
002A  2300 I=1
002A  2310 I=1
002A  2320 I=1
002A  2330 I=1
002A  2340 I=1
002A  2350 I=1
002A  2360 I=1
002A  2370 I=1
002A  2380 I=1
002A  2390 I=1
002A  2400 I=1
002A  2410 I=1
002A  2420 I=1
002A  2430 I=1
002A  2440 I=1
002A  2450 I=1
002A  2460 I=1
002A  2470 I=1
002A  2480 I=1
002A  2490 I=1
002A  2500 I=1
002A  2510 I=1
002A  2520 I=1
002A  2530 I=1
002A  2540 I=1
002A  2550 I=1
002A  2560 I=1
002A  2570 I=
```

2 USER=TON 15 417 JOINT COMPUTER FACILITY, MIT PAGE 2

122L[V] IPAT	1230[V] TCCN	1232[V] ISU	1234[V] N'VG	1236[V] SLFLOW
1231[V] IPAL	1238[V] I'POL	1239[V] IJLO	1240[V] N'VG	1242[V] NL
1244[V] NL VLL	1245[V] I'	1246[V] TCCN	1247[V] N'IS	1248[V] INTIME
1234[V] I'GAL	1244[L] I'OC	1245[V] SEARCH	0000[S] .Q	0000[S] .F
0000[V] INPUT	0000[L] I'AL	1247[V] NW	0004[L] 12	0112[V] I
0000[S] LG	0114[V] V	0004[L] 20		

1. IS PROGRAM WILL OCCUPY 113 BYTES OF STORAGE SPACES

2. FOR ALL SEARCH AND NO ERROR

// 02

USER=TVUJ 255 34170 JOINT COMPUTER FACILITY, MIT PAGE 3

```

0394 CALL SLETT(IHP,IMIN)
0394 IHP=IHP*156+IMIN
0394 C *****
0394 CALL LINKR(MAINS,1)
0394 900 WRITE(6,205)IHP
0394 4-ITS(5,205)IHP
0394 905 FORIAT(6X,15,'FILE ERROR')
0410 STOP
0410 END
1205[C] // 0000[V] ITH 0002[V] ITH 0004[V] IA 0006[V] IH
000 [V] NS 0001[V] IIS 0003[V] NP 0005[V] IEPROR 0010[V] IDATE
0012[V] ITIME 0014[V] ID 0016[V] IREFEP 0018[V] IPTH 0019[V] NR
0012[V] IT 0017[V] IBOUND 0022[V] LBOUND 0026[V] IPTSET 0029[V] IPTCAL
0021[V] JMS 0020[V] JDAY 0027[V] JYR 0030[V] JHR 0032[V] JMIN
0034[V] JS-C 0031[V] JHSEC 0039[V] JDATE 0034[V] PERIOD 0033[V] IDURA
003 [V] IOL 0040[V] I 0042[V] IDAY 0044[V] IYS 0046[V] IUP
004 [V] IAI 0041[V] ISEC 0043[V] IMLSEC 0047[V] INTIME 0050[V] IFFLOW
0052[V] ILE 0054[V] IFLOW 0059[V] TEXPIM 0050[V] ICTFYP 0058[V] TEMPIC
0062[V] AREA 0066[V] LE 0069[V] AMCL 0068[V] TASK 0070[V] IN
0072[V] IT 0074[V] NTR 0078[V] NCAL 0073[V] NMTN 0070[V] NM
007 [V] IOUT 0071[V] M-CAT 0080[V] MCHN 0072[V] IHPED
0074[V] IJFD 0074[V] IV 0080[V] LTVLS 0084[V] IAI 0082[V] ICL
0082[V] IJFC 1112[V] IJFB 1118[V] IJFA 1119[V] IPUFE 1118[V] NAME
1122[V] ICHD 112 [V] ICH 1123[V] IPR 1124[V] IECHO 1120[V] IRMAIN
112 [V] IPR 1130[V] IACON 1132[V] ISUP 1134[V] NAYG 1126[V] SLEFLOW
113 [V] IPIOM 1131[V] ICHOL 1135[V] IJLOW 1140[V] NAYG 1142[V] NL
114 [V] NLLVTL 1146[V] ICHD 114 [V] ICCON 1142[V] MINS 1142[V] INTIME
114 [V] ICHOL 1140[V] I 1141[V] INBT 0442[V] FOUNL 0009[V] MAINS1
044 [V] I 114 [V] I 1140[V] I 0342[L] 900 0000[S] 23
0450[V] ITH 0454[V] I 0450[V] I 0450[V] I 0474[L] 10 0450[V] J
000 [V] SLIP 114 [V] I 0474[L] 2 0450[V] I 0208[L] 3
0272[V] 1 000 [V] ITH 0000[C] 2X 0228[L] 2222 0000[S] 21
0000[L] SETTIA 0000[S] ITH 0386[L] 205 0000[S] .S 0000[S] .V
C PROGRAM ALL 0000 X 46 BYTES OF STORAGE SPACE
//

```


USER=TUN 255 8417 JOINT COMPUTER FACILITY, MIT PAGE 2

```

01300 CALL GETSUB(' 332. BUNDLE ID:',ARG,0.,32767.)
01301 IF(ARG.LT.0.) GO TO 20
01302 ID=ARG
01303
01304 C COMMAND LEVEL:
01305 35 INIT=1
01306 PUT 6,0,' >'
01307 READ(ICM,40) ICOMD
01308 40 FOR 'AT(2A2)
01309
01310 C PROCESS COMMAND, TAKE THE RIGHT ACTION
01311 DO 43 I=1,7
90F4 I/O ERROR

```

30	PACKING CHARGE	0 .35 / 100	.35
3013	LINE PRINTER	0 1.25 / 100	1.25
4169	CHARGE PER D	0 1.50 / 100	1.50
30	PLOTTER VOLUME	0 .25 / 1000	.25
417	CHARGE 70 SLIP	0 25.00 / 100	25.00
30	MODEL 40 S C	0 15.00 / 100	15.00
	TOTAL CHARGE		32.92

LOGS 255 84173 LOGGED OUT ??/?/67 04:55. 32.92 LEFT AFTER 112 LOGINS.

APPENDIX II

LISTING OF INTERMEDIATE DATA REDUCTION PROGRAMS

PROGRAM THAT LISTS FLOPPY DISK DATA FILES

TUNG

34170

```

C VARIABLE ARG IS A GENERAL PURPOSE VARIABLE:
  REAL ARG
C IBUFA CONTAINS HEADER BLOCK DATA, MYDATA CONTAINS DATA BLOCK
C DATA, AND DTAPOS IS A POINTER TO THE DATA BLOCK:
  INTEGER*2 IDUFA(24), MYDATA(512), DTAPOS
C HDRNUM CONTAINS THE NUMBER OF HEADER BLOCKS ON THE FLOPPY:
  INTEGER*2 HDRNUM
C EQUAL IS A CHARACTER FUNCTION THAT RETURNS A LOGIC VALUE:
  LOGICAL EQUAL
C RECORD IS THE NUMBER OF RECORDS IN A DATA BLOCK, PRONUM IS THE
C NUMBER OF PRORES BEING SAMPLED AT A TIME, AND NAME CONTAINS
C THE ASCII TITLE OF A HEADER BLOCK:
  INTEGER*2 RECORD, PRONUM, Q, NAME(5)
  IMPLICIT INTEGER*2 (I-N)
  DEFINE FILE 1(2,64,U,ITEXT)
  DEFINE FILE 2(800,12,U,IHTD)
  DEFINE FILE 3(1000,64,U,IDAT)
C IXI IS THE LOGICAL NUMBER OF THE PRINTER:
  IXI=5
  ICI=5
C IOLDST IS THE POINTER TO THE LAST SETUP BLOCK...
  IOLDST=-1
  READ(2*1)IBUFA
C SETTING HDRNUM EQUAL TO THE NUMBER OF HEADER BLOCKS:
  HDRNUM=IBUFA(1)-1
487  FORMAT(' NO SUCH BLOCK EXISTS')
487  FORMAT(5A2)
3011  FORMAT(A2)
C THE FOLLOWING CODE PRINTS OUT ALL HEADER INFORMATION FROM DTSC
1000  READ(2*1) IBUFA
      IR=IBUFA(1)
      J=IR-1
      DO 2000 I=2,J
      READ(2*I) IBUFA
      WRITE(IXI,3000)
C PRINTING OUT THE TITLE OF THE BLOCK:
1008  WRITE(IXI,2100)(IBUFA(N),N=1,5)
2100  FORMAT(5X,5I2)
      WRITE(IXI,1984) IBUFA(12)
1984  FORMAT(' BUNDLE IDENT.CODE  :',I5)
C DECODING DATE (DAYS, MONTHS, YEARS)
      JDATE=IBUFA(13)
      IDIV=2**3
      JMON=JDATE/IDIV
      JREST=MOD(JDATE,IDIV)
      IDIV=16
      JDAY=JREST/IDIV
      JYR=MOD(JREST,IDIV)+76
      WRITE(IXI,2002) JMON,JDAY,JYR
2002  FORMAT(' DATE (MO.,DAY,YE.) :',I2,'/',I2,'/',I2)

```

US TUNG 255 84170 JOINT COMPUTER FACILITY, MIT PAGE 2

SUNDAY

```

C DECODING TIME OF DAY (HRS,MINC)
  ITIME=IBUFA(14)
  IHOUE=ITIME/256
  IMIN=MOD(ITIME,256)
  WRITE(IXI,2003) IHOUE,IMIN
2003  FORMAT(' TIME(HRS,MINC)      :',I2,',',I2)
C DECODING TIME OF DAY (SECS,MSECS)
  ITIME=IBUFA(15)
  ISEC=ITIME/1000
  IMSEC=MOD(ITIME,1000)
  IF(IBUFA(9).EQ.3) WRITE(IXI,2004) ISEC,IMSEC
2004  FORMAT(' TIME(SECS,MSECS)    :',I2,',',I3)
C THE FOLLOWING CODE APPLIES TO SETUP BLOCKS ONLY:
  IF(IBUFA(9).NE.1) GOTO 1776
  WRITE(IXI,2001) IBUFA(16)
2001  FORMAT(' PROBE TYPE          :',I5)
  WRITE(IXI,2005) IBUFA(17)
2005  FORMAT(' NUMBER OF PROBES    :',I5)
  WRITE(IXI,2006) IBUFA(18)
2006  FORMAT(' OSC.FREQ.(HZ.)      :',I5)
  WRITE(IXI,2007) IBUFA(19)
2007  FORMAT(' SAMPLE TIME(10*SEC):',I5)
C THE FOLLOWING CODE APPLIES TO CALIBRATION BLOCKS ONLY:
1776  IF(IBUFA(9).NE.2) GOTO 1976
  WRITE(IXI,1234) IBUFA(10)
  WRITE(IXI,2008) IBUFA(17)
2008  FORMAT(' INJ.SALT,GPM*1000: ',I5)
C THE FOLLOWING CODE APPLIES TO MIXING BLOCKS ONLY:
1876  IF(IBUFA(9).NE.3) GOTO 1976
  WRITE(IXI,1234) IBUFA(10)
  WRITE(IXI,1235) IBUFA(11)
  WRITE(IXI,1235) IBUFA(24)
  WRITE(IXI,2009) IBUFA(17)
2009  FORMAT(' FLOWRATE (GPM)      :',I5)
  WRITE(IXI,2010) IBUFA(18)
2010  FORMAT(' BUNDLE D.E.         :',I5)
C IBUFA(19) NEGATIVE SIGNIFIES LARGE NUMBERS PACKED IN.
  IF(IBUFA(19).LT.0) IBUFA(19)=IBUFA(19)+2**16
  WRITE(IXI,2011) IBUFA(19)
2011  FORMAT(' BUNDLE R.E.         :',I5)
  WRITE(IXI,2009) IBUFA(20)
  WRITE(IXI,2012) IBUFA(21)
2012  FORMAT(' INJ.SALT,GPM*10000: ',I5)
  WRITE(IXI,2013) IBUFA(22)
2013  FORMAT(' INJ.SUPCHANNEL       :',I5)
C DECODING AXIAL HEIGHT DATA TO GET THE DECIMAL POINT RIGHT
  ARG=IBUFA(23)/100
  WRITE(IXI,2014) ARG
2014  FORMAT(' AXIAL LEVEL(IN.)    :',F6.2)
1976  IF(IBUFA(9).EQ.1) GOTO 2076
C DECODING WATER TEMP. DATA TO GET THE DECIMAL POINT CORRECT
  ARG=IBUFA(16)/10
  WRITE(IXI,2015) ARG

```

JGER=TUNG 255 84170 JOINT COMPUTER FACILITY, MIT PAGE 3

SHUBS

```

2015  FORMAT(' WATER TEMP(DEG.C.) :',F5.1)
2076  WRITE(IXI,3000)
2000  FORMAT(' ')
C DTAPOS IS THE POSITION OF THE NUMERICAL DATA:
  DTAPOS=IBUFA(6)
C RECORD IS THE NUMBER OF RECORDS IN THIS DATA BLOCK:
  RECORD=IBUFA(7)
C ISETPS IS THE NUMBER OF SETUP BLOCK SERIES:
  ISETPS=IBUFA(10)
C PERHAPS WE HAVE ALREADY ACCESSED THE SETUP BLOCK...
  IF(ISETPS.EQ.IOLDST) GO TO 140
  IOLDST=ISETPS
C IF THE PTR TO SETUP IS ZERO, IT IS A SETUP BLOCK:
  IF(ISETPS.EQ.0) PRONUM=IBUFA(17)
  IF(ISETPS.EQ.0) GO TO 149
C OTHERWISE WE MUST SEARCH FOR THE PROPER SETUP BLOCK:
  INUM=256*(ISETPS/10+4)+MOD(ISETPS,10)+48
  CALL MOVE('SET',0,NAME,0,3)
  CALL MOVE(INUM,0,NAME,3,2)
  DO 143 N=2,PRONUM
    READ(2'N')IBUFA
    IF(EQUAL(IBUFA,0,NAME,0,5)) GO TO 147
143  CONTINUE
C PRONUM CONTAINS THE NUMBER OF PROBES IN THIS DATA SET:
147  PRONUM=IBUFA(17)
149  I/=1
2077  IH=IX+127
      IF(DTAPOS.LE.0) GO TO 2000
      READ(3'DTAPOS')(MXDATA(N),N=IH,IH)
      DTAPOS=DTAPOS+1
      RECORD=RECORD-1
      IF(RECORD.LE.0) GO TO 2079
      IX=IX+128
      GO TO 2077
2079  DO 2079 N=1,PRONUM,8
      IX=N+7
      IF((PRONUM-IX).LT.8) IX=PRONUM
      WRITE(IXI,2000)(MXDATA(N),N=IX,IX)
2000  FORMAT(5,'8I5)
      IF(PRONUM.EQ.IX) GO TO 2070
2070  CONTINUE
      WRITE(IXI,3000)
2000  CONTINUE
C
1234  FORMAT(' SETUP BLOCK :',I5)
1235  FORMAT(' CALIB.CORRECTION :',I5)
1236  FORMAT(' CALIBRATION BLOCK :',I5)
500  PUT 'HEADER BLOCK EXAMINATION ROUTINE ENDS NOW.'
END

```

J	0014[V] ARG	022[V] IUF	052[V] MXDATA	1052[V] DTAPOS
FROM	0000[S] EQUAL	1056[V] RECORD	1058[V] PRONUM	1059[V] 0
NAME	0000[D] 03	1066[V] ITEMP	1067[V] ISETPS	1068[V] IOLDST
IXI	1070[V] ICI	1074[V] IOLDST	1075[V] 0000	1076[V] 0000

PROGRAM WHICH GRAPHS CALIBRATION DATA

450

TUNE

94170

```

C SALT MIXING DATA ANALYSIS PROGRAM
C THIS PROGRAM GRAPHS OUT THE CALIBRATION DATA AND ALLOWS THE
C USER TO EDIT BAD DATA POINTS. BE SURE THAT THE FILE DEFINITIONS
C AT THE END OF THE PROGRAM MATCH THE CARTRIDGE I.D. FOR THE
C FLOPPY DISK YOU ARE USING.
C
C REAL M(512) IS A DATA BUFFER:
    REAL M(512)
C VARIABLE XLAB IS A CHARACTER ARRAY USED TO TITLE GRAPHS:
    INTEGER*2 XLAB(40)
C ARRAY IS USED FOR SORTING DATA BEFORE GRAPHING:
    REAL ARRAY(9,10)
C EQUAL IS A SYSTEM LIBRARY CHARACTER FUNCTION:
    LOGICAL EQUAL
C SUM IS A GENERAL PURPOSE VARIABLE:
    REAL SUM
C IPPOS AND IREC DEAL WITH THE POSITION OF DATA BLOCKS, SO THAT
C ADDED DATA MAY BE WRITTEN INTO THE CORRECT POSITION:
    INTEGER*2 IPPOS(10),IREC(10)
C ARRAY NAME IS USED FOR STORING CHARACTER FIELDS
    INTEGER*2 NAME(5)
C RATIO IS A REAL ARRAY IN WHICH ARE STORED THE DIMENSIONLESS RATIOS.
    REAL RATIO(512)
C CALNUM IS THE NUMBER OF CALIBRATION BLOCKS IN THE SERIES,
C HDRNUM IS THE NUMBER OF HEADER BLOCKS ON THE FLOPPY, PRONUM IS
C THE NUMBER OF PROBES SAMPLED AT A TIME, SALT(10) CONTAINS THE
C SALT CONCENTRATIONS AT WHICH THE CALIB. BLOCKS WERE TAKEN,
C AND RCORD IS THE NUMBER OF RECORDS PER SAMPLE:
    INTEGER*2 CALNUM,HDRNUM,2,PRONUM,SALT(10),RECORD
C DATA BLOCK BUFFERS
    INTEGER*2 MNDATA(512),PROSD(512),CALCOR(512),CALIB(10,512)
C DATA BLOCK POINTERS
    INTEGER*2 SETPOS,CALPOS,DTAPCS,CORPOS
C HEADER BLOCK BUFFER:
    INTEGER*2 IBUFA(24)
    IMPLICIT INTEGER*2 (I-N)
C RG IS A GENERAL PURPOSE VARIABLE. (FREQUENCY).
    REAL RG
    DEFINE FILE 1(2,64,U,ITXT)
    DEFINE FILE 2(800,12,J,IHPD)
    DEFINE FILE 3(1000,64,U,IDAT)
C SETTING UP THE GRAPH TITLE:
    CALL MOVE('SALT CONC.(GR'S SALT*1000/LBS.WATER) ',
    2 0,XLAB,0,40)
    CALL MOVE('PROBE RESISTANCE,FIRST CURVE=PROBE ',
    2 0,XLAB,40,40)
    ICHANG=0
    ICHLT=0
    ICHAY=0
C HDRNUM SIGNIFIED THE NUMBER OF HEADER BLOCKS IN FILE

```


US-B-TUNG 255 94170 JOINT COMPUTER FACILITY, MIT PAGE 2

SUNDAY

```

      READ(2*1)(IBUFA(N),N=1,24)
      HDRNUM=IBUFA(1)-1
C OUTPUT DEVICE NUMBER IS 5
      IYI=5
C INTERACTIVE DEVICE NUMBER = 6
      ICI=6
487  FORMAT(' NO SUCH BLOCK EXISTS')
48C  FORMAT(5A2)
3011 FORMAT(A2)
C
C THE FOLLOWING READS IN THE DESIRED BLOCKS:
C
69  CALNUM=0
      WRITE(ICI,71)
71  FORMAT(' WHICH CALIPRATION SET?')
      READ(ICI,48C)NAME
C SEARCHING FOR THE SELECTED SERIES:
      DO 73 I=2,HDRNUM
        READ(2*I)IBUFA
        IF(.NOT.(EQUAL(IBUFA,C,NAME,C,5))) GO TO 70
C WE IGNORE BLOCKS WITH SALT CONC.>10000 :
        IF(IBUFA(17).GE.10000 ) GO TO 70
C DTAPOS IS THE POSITION OF THE DATA BLOCK:
        DTAPOS=IBUFA(6)
C RECORD IS THE NUMBER OF RECORDS IN THE DATA BLOCK:
        RECORD=IBUFA(7)
        IF(CALNUM.EQ.0) GO TO 7770
C WHEN TWO BLOCKS WERE TAKEN AT THE SAME CONCENTRATION,THE
C FIRST ONE IS REPLACED WITH THE LAST ONE:
        DO 7771 N=1,CALNUM
          IF(SALT(N).EQ.IBUFA(17)) GO TO 7775
7771  CONTINUE
        GO TO 7770
7775  READ(3*DTAPOS)(CALIB(N,IX),IX=1,128)
C IPOS AND IREC STORE THE POSITION ON THE FLOPPY FOR IITER:
        IPOS(N)=DTAPOS
        IREC(N)=RECORD
        GO TO 70
7770  CALNUM=CALNUM+1
C SALT IS THE CORRESPONDING SALT CONCENTRATION:
        SALT(CALNUM)=IBUFA(17)
        IY=1
        IPOS(CALNUM)=DTAPOS
        IREC(CALNUM)=RECORD
2001  IY=IY+127
        READ(3*DTAPOS)(CALIB(CALNUM,J),J=IX,IX)
        RECORD=RECORD-1
        DTAPOS=DTAPOS+1
        IY=IY+128
        IF(RECORD.GT.0) GO TO 2001
        CONTINUE
        IF(CALNUM.LE.0) WRITE(ICI,487)
        IF(CALNUM.LE.0) GO TO 6

```

USER=TUNG 255 84170 JOINT COMPUTER FACILITY, MIT PAGE 3

SUNDAY

C THE PROPER CALIB SET SHOULD NOW BE IN CORE.

```

C
  ICHANG=
  WRITE(ICI,132)
132  FORMAT(' DO YOU WANT TO LOOK AT ALL THE PROBS?')
  READ(ICI,3011)IANS
  IF(EQUAL(IANS,0,'Y',0,1)) GO TO 133
  ICHANG=1
137  WRITE(ICI,134)
134  FORMAT(' ENTER THE NUMBER OF THE FIRST PROBE:')
  READ(ICI,135)J
135  FORMAT(I3)
  GO TO 66
133  DO 76 J=1,512,8
C ARRAY IS WHAT WE ACTUALLY USE TO GRAPH THE CALIBRATION DATA:
66  DO 78 N=1,CALNUM
  APRAY(9,N)=SALT(N)
  DO 77 IX=1,8
  IX=J-1+IX
  ARRAY(IX,N)=CALIB(N,IX)
77  CONTINUE
78  CONTINUE
C NOW WE HAVE TO ORGANIZE THE DATA IN ORDER OF SALT CONC.:
64  IX=CALNUM-1
  DO 65 N=1,IX
  IZ=N+1
  IF(ARRAY(9,N).LE.ARRAY(9,IZ)) GO TO 65
  DO 121 Q=1,8
  ARG=APRAY(Q,N)
  ARRAY(Q,N)=ARRAY(Q,IZ)
  ARRAY(Q,IZ)=ARG
121  CONTINUE
  ARG=APRAY(9,N)
  ARRAY(9,N)=ARRAY(9,IZ)
  ARRAY(9,IZ)=ARG
  GO TO 64
65  CONTINUE
C THE NEXT TWO LINES SET THE FRONT NUMBER ON THE TITLE:
  IANUM=256*(J/10 +43)+MOD(J,10)+43
  CALL MOVE(IANUM,0,XLAB,78,2)
C BY NOW THE DATA SHOULD BE SET FOR GRAPHING:
  IGRAPH=1
  GO TO 994
993  IGRAPH=2
994  IF(ICHANG.EQ.1) GO TO 997
C THIS IS THE CALL TO THE SYSTEM LIBRARY PROGRAM THAT DOES THE
C ACTUAL GRAPHING:
  CALL QPICTR(APRAY,9,CALNUM,QY(1,2,3,4,5,6,7,8),
  2 QX(9),QXLAB(XLAB),CLABEL(4),CLOOK(IGRAPH))
  GO TO 993
997  CALL QPICTR(ARRAY,9,CALNUM,QY(1),QX(9),
  2 QXLAB(XLAB),CLABEL(4),CLOOK(IGRAPH))
998  IF(IGRAPH.EQ.2) GO TO 995

```

USER=TUNG 255 84170 JOINT COMPUTER FACILITY, MIT PAGE 4

SUNDAY

```

WRITE(ICI,992)
992  FORMAT(' DO YOU WANT CALCOMP OUTPUT?')
    READ(ICI,3011)IANS
    IF(EQUAL(IANS,0,'Y',0,1)) GO TO 993
995  IGRAPH=1
    IF(IDELTA.EQ.1) GO TO 129
    GO TO 987
128  WRITE(ICI,129)
129  FORMAT(' DO YOU WANT TO RE-WRITE OVER THE CALIBRATION',
2 //, ' SERIES ON THE FLOPPY DISK?')
    READ(ICI,3011)IANS
    IF(EQUAL(IANS,0,'N',0,1)) GO TO 987
C THE FOLLOWING CODE RE-WRITES DATA ON FLOPPY:
    DO 131 I=1,CALNUM
    DTAPOS=IPOS(I)
    RECORD=IREC(I)
    IX=1
    DO 144 N=1,RECORD
    IH=IX+127
    WRITE(3'DTAPOS)(CALIB(I,I7),IE=IX,IV)
    DTAPOS=DTAPOS+1
    IV=IX+128
144  CONTINUE
131  CONTINUE
C END OF CODE TO RE-WRITE FLOPPY.
C
987  WRITE(ICI,126)
    IDELTA=0
126  FORMAT(' DO YOU WANT TO CHANGE ANY DATA?')
    READ(ICI,3011)IANS
    IF((EQUAL(IANS,0,'N',0,1)).AND.(ICHANG.EQ.1)) GO TO 137
    IF(EQUAL(IANS,0,'N',0,1)) GO TO 75
    IF(EQUAL(IANS,0,'Q',0,1)) GO TO 123
    IDELTA=1
    WRITE(ICI,122)
    WRITE(ICI,127)J
127  FORMAT(' THE FIRST POINT IN THIS SERIES IS:',I5)
122  FORMAT(' THESE ARE THE NUMERICAL VALUES FOR CALIB:')
    DO 124 IZ=1,CALNUM
    IJ=J
    IV=J+7
    WRITE(ICI,125)SALT(IZ),(CALIB(IZ,Q),Q=IJ,IV)
125  FORMAT(9I5)
124  CONTINUE
    READ(ICI&111)
111  NAMELIST ' ENTER CALIB(SET,PROBE):' CALIB
    GO TO 66
76  CONTINUE
123  PUT ' CALIBRATION EXAMINATION ROUTINE ENDS '
C
END
CDDC[V] A      15CC[V] VLAB      1610[V] PRB.V      1600[SI] PRB.V
SU.  177C[V] IPOS  1790[V] IREC      17A4[V] NAME      17B5[V] RATIO

```

PROGRAM WHICH ANALYZES MIXING DATA,
OUTPUTS DIMENSIONAL AND/OR DIMENSIONLESS
CONCENTRATION MAPS, AND PUNCHES INPUT
DECKS FOR SUPERENERGY/DRV

(NOTE: Current version of this program can only produce output maps and salt mass balances for 61 pin bundles, where the probe numbering scheme devised in [3] is used. Addition of a more general mapping subroutine and salt mass balance routine is required for proper display of data from different size bundles.)

TUNG

84170

```

C THIS PROGRAM CONVERTS THE DATA ON THE FLOPPY DISK FROM
C RESISTANCES TO SALT CONCENTRATION TO DIMENSIONLESS RATIO AND
C COMPUTES MASSBALANCE. IN ADDITION IT ALLOWS A VARIETY OF
C PRINT OUT ROUTINES, INCLUDING PUNCHED CARDS.
C
C ARRAY F(30) STORES THE CHANNEL NUMBERS FOR EDGE AND
C CORNER CHANNELS, RESPECTIVELY:
C   INTEGER*2 F(30)
C AVGGPM IS THE AVERAGE FLOW THROUGH THE SUBCHANNELS:
C   REAL AVGGPM
C VAVG IS THE AVERAGE VELOCITY THROUGH THE SUBCHANNELS:
C   REAL VAVG
C NETSLT IS THE AMOUNT OF SALT INJECTED INTO THE BUNDLE,
C OUTSLT IS THE AMOUNT OF SALT DETECTED:
C   REAL NETSLT, OUTSLT
C ISALTF IS THE INJECTED SALT FLOW, ISALTC IS THE INJECTED
C SALT CONCENTRATION:
C   REAL ISALTF, ISALTC
C ARRAY M(512) IS A BUFFER FOR THE DATA:
C   REAL M(512)
C EQUAL IS A LIBRARY PROGRAM THAT RETURNS A LOGIC VALUE:
C   LOGICAL EQUAL
C SUM IS A GENERAL PURPOSE VARIABLE:
C   REAL SUM
C NAME AND MXNAME ARE ASCII CHARACTER DATA:
C   INTEGER*2 NAME(5), MXNAME(5)
C ARRAY FLOW CONTAINS FLOWSPLIT FOR INTERIOR, EDGE AND CORNER
C CHANNELS, PLUS REYNOLDS NUMBERS, RESPECTIVELY. AREA CONTAINS
C THE AREAS OF THE BUNDLE SUBCHANNEL:
C   REAL FLOW(4,20), AREA(3), TOTALS
C FLOWM CONTAINS THE NUMBER OF DIFFERENT REYNOLDS
C NUMBERS FOR WHICH FLOWSPLIT HAS BEEN INPUT:
C   INTEGER*2 IFLOWM
C RATIO IS A DATA BUFFER:
C   REAL RATIO(512)
C CALNUM IS THE NUMBER OF CALIBRATION BLOCKS IN A PARTICULAR
C SERIES, HDNUM IS THE NUMBER OF HEADER BLOCKS OF THE FLOWM,
C PRONUM IS THE NUMBER OF PROBES SAMPLED, AREA SALT CONTAINS CONC.
C VALUES FOR THE CALIBRATION SERIES, RECORD IS THE NUMBER OF
C RECORDS STORED IN A PARTICULAR DATA SET.:
C   INTEGER*2 CALNUM, HDNUM, C, PRONUM, SALT(10), RECORD
C THE FOLLOWING ARE DATA BLOCK BUFFERS:
C   INTEGER*2 MCDATA(512), PROSEQ(512), CALCOR(512), CLINT(10,512)
C THE FOLLOWING ARE POINTERS:
C   INTEGER*2 SETPOS, CALPOS, DTAPOS, CORPOS
C ARRAY IBUFA IS A HEADER BLOCK BUFFER:
C   INTEGER*2 IBUFA(24)
C
C   IMPLICIT INTEGER*2 (I-N)
C RG IS A GENERAL PURPOSE VARIABLE:

```

USER=TUNG 255 84170 JOINT COMPUTER FACILITY, MIT PAGE 2

CV DIV

```

      REAL ARG
C GENERAL PURPOSE REAL VARIABLES:
      REAL AREAL,BREAL,CREAL,DREAL
C MASEAL IS THE MASS-BALANCE:
      REAL MASEAL
      DEFINE FILE 1 (2,64,7,ITXT)
      DEFINE FILE 2 (800,12,U,IWRD)
      DEFINE FILE 3 (1000,64,U,IDAT)
      LU=7
      ICHAN=0
C IONE IS 0 WHEN EXAMINING ALL 41Y BLOCKS AT ONCE, ELSE 1
      IONE=0
      IFLOM=0
      IBEGIN=0
C HEADING IN CHANNEL NUMBERS FOR EDGE AND CORNER CHANNELS:
      READ (8,519) F
519   FORMAT(15I5)
      READ(2*1)IBUFA
C IORNUM CONTAINS THE NUMBER OF HEADER BLOCKS OF THE FLOW:
      IORNUM=IBUFA(1)-1
C IPUNCH DICTATES WHETHER OR NOT CARDS ARE TO BE PUNCHED:
      IPUNCH=0
C IXI IS THE PRINTER DEVICE:
      IXI=5
C ICI IS THE INTERACTIVE DEVICE:
      ICI=6
487   FORMAT(' NO SUCH BLOCK EXISTS.')
488   FORMAT(5A2)
3011  FORMAT(A2)
3000  FORMAT(' ')
2011  FORMAT(' BUNDLE P.E.           :',I5)
2013  FORMAT(' INJ. SUBCHANNEL       :',I5)
2014  FORMAT(' AXIAL LEVEL (IN.)     :',F5.2)
C
C THIS CODE INPUTS FLOW SPLIT DATA:
      WRITE(ICI,490)
490   FORMAT(' ENTER SUBCHANNEL AREAS, (IN SQUARE INCHES), //
2 * FOR INTERIOR, EDGE, AND CORNER SUBCHANNELS:')
      READ(ICI,491) AREA(1),AREA(2),AREA(3)
491   FORMAT(3F7.4)
      WRITE(ICI,493)
493   FORMAT(' INPUT NUMBER OF REYNOLDS NUMBERS:')
      READ(ICI,494) IFLOM
494   FORMAT(I5)
      DO 499 N=1,IFLOM
      WRITE(ICI,495)
495   FORMAT(' ENTER R.E., AND VELOCITY/VELOCITY VARIOUS //
2 * FOR INTERIOR, EDGE, AND CORNER CHANNELS:')
      READ(ICI,496)FLOW(4,N),FLOW(1,N),FLOW(2,N),FLOW(3,N)
496   FORMAT(4F10.4)
499   CONTINUE
C END OF CODE FOR INPUTTING FLOW SPLIT DATA.
C

```

US=TUNG 255 84170 JOINT COMPUTER FACILITY, MIT PAGE 3

SUNDAY

```

      IY=2
      WRITE(ICI,100)
100  FORMAT(' DO YOU WANT ALL MIXING DATA CONVERTED AT ONCE?')
      READ(ICI,3011) IANS
      IF(EQUAL(IANS,0,'Y',0,1)) GO TO 180
      IF(EQUAL(IANS,0,'Q',0,1)) GO TO 9000
      IONE=1
106  WRITE(ICI,102)
102  FORMAT(' INPUT TITLE OF DESIRED BLOCK:')
      READ(ICI,480)NAME
      IF(EQUAL(NAME,0,'Q',0,1)) GO TO 9000
      DO 103 I=2,HDRNUM
      READ(2*I)IBUFA
      IF(EQUAL(IBUFA,0,NAME,0,10)) GO TO 104
103  CONTINUE
      WRITE(ICI,487)
      GO TO 106
104  GO TO 700
      GO TO 100
180  WRITE(ICI,115)
115  FORMAT(' DO YOU WANT CARDS PUNCHED ALSO?')
      READ(ICI,3011)IANS
      IF(EQUAL(IANS,0,'N',0,1)) IPUNCH=1
      IF(EQUAL(IANS,0,'Y',0,1)) CALL RJPUV(LU,'STANDARD')
      WRITE(ICI,9090)
9090  FORMAT(' ENTER NUMBER OF FIRST BLOCK TO BE ANALYZED:')
      READ(ICI,494) ISTART
      ICONCF=0
      IDIMEN=0
      WRITE(ICI,4321)
4321  FORMAT(' DO YOU WANT CONCENTRATIONS AND OR DIMENSIONALITY',
2 //, ' RATIOS?(C/D/B/N):')
      READ(ICI,3011)IANS
      IF(EQUAL(IANS,0,'D',0,1)) IDIMEN=1
      IF(EQUAL(IANS,0,'C',0,1)) ICONCF=1
      IF(EQUAL(IANS,0,'B',0,1)) IDIMEN=1
      IF(EQUAL(IANS,0,'N',0,1)) ICONCF=1
200  READ(2*IY)(IBUFA(N),N=1,24)
      IF(IBUFA(9).EQ.3) IBEGIN=IBEGIN+1
      IF(IBEGIN.LT.ISTART) GO TO 950
      IF(IBUFA(9).EQ.3) GO TO 700
      IF(IY.GE.HDRNUM) GO TO 9000
      GO TO 950
700  WRITE(ICI,3000)
C
C
C ASSUME THAT BY NOW THE APPROPRIATE MIXING HEADER BLOCK
C HAS BEEN STORED IN ARRAY IBUFA(24),-WE FIRST HAVE TO STORE
C DATA OF INTEREST FROM THIS BLOCK:
C THE NEXT TWO LINES STORE THE TITLE:
      DO 3014 N=1,5
3014  IYNAME(N)=IBUFA(N)
C INJECTED SALT CONCENTRATION (1000 GRMS.SALT/10. WATER):

```

USER=TUNG 255 84170 JOINT COMPUTER FACILITY, MIT PAGE 4

SUBJECT

```

      ISALPC=IBUFA(20)
C SALT FLOW (GPM*1000000):
      ISALTF=IBUFA(21)
C BUNDLE FLOW IN G.P.M.:
      IGPM=IBUFA(17)
C DTAPOS CONTAINS THE POSITION OF THE FIRST RECORD OF DATA:
      DTAPOS=IBUFA(6)
C RECORD CONTAINS THE NUMBER OF RECORDS INVOLVED:
      RECORD=IBUFA(7)
C SETPOS IS A PTR TO THE SETUP BLOCK:
      SETPOS=IBUFA(10)
C CALPOS IS THE PTR TO THE CALIBRATION SERIES:
      CALPOS=IBUFA(11)
C CORPOS IS THE PTR TO THE CALIBRATION CORRECTION BLOCK:
      CORPOS=IBUFA(24)
C THE AXIAL POSITION OF INJECTED SALT:
      IAXIAL=IBUFA(23)
C REYNOLDS NUMBER OF THE BUNDLE:
      IREYN=IBUFA(19)
C INJECTION SUBCHANNEL:
      ISUBCS=IBUFA(22)
C
C ALL DATA FROM HEADER BLOCK HAS BEEN STORED.
C NOW WE HAVE TO ACCESS THE MIXING DATA BLOCK:
      IX=1
3015  Ix=IX+127
      READ(3'DTAPOS)(MADATA(N),N=IX,IX)
      RECORD=RECORD-1
      DTAPOS=DTAPOS+1
      IF(RECORD.EQ.0) GO TO 3013
      Ix=IX+128
      GO TO 3015
C NOW WE HAVE THE MIXING DATA READ INTO MADATA(512).
C
C NOW WE HAVE TO MAKE SURE THAT WE HAVE THE CORRECT SETUP
C BLOCK PROPERLY ACCESSED:
3013  IF(SETPOS.EQ.IOLDST) GO TO 3210
      IOLDST=SETPOS
C SWIP OVER THIS MIX BLOCK IF PTR TAKES NO SENSE:
      IF((SETPOS.LE.0).OR.(SETPOS.GT.HDRNUM)) GO TO 350
C WE NEED TO EXECUTE A CHARACTER SEARCH ROUTINE:
      IANUM=256*(SETPOS/10 +48)+101(SETPOS,10)+48
      CALL MOVE('SET',0,NAME,0,3)
      CALL MOVE(IANUM,0,NAME,3,2)
      DO 30 I=2,HDRNUM
      READ(2'I)IBUFA
      IF(EQUAL(IBUFA,0,NAME,0,5)) GO TO 44
30  CONTINUE
C IF WE GET HERE, SETUP BLOCK COULD NOT BE FOUND.
  PUT ' THIS SETUP BLOCK COULD NOT BE ACCESSED.'
  GO TO 350
44  SETPOS=I
      RECORD=IBUFA(17)

```


USNS=TUNG 255 84170 JOINT COMPUTER FACILITY, MIT PAGE 5

SUNDAY

```

DTAPOS=IBUFA(6)
RECORD=IBUFA(7)
C READING SAMPLING SEQUENCE INTO PROSEQ(512):
IX=1
3100 IH=IX+127
READ(3'DTAPOS)(PROSEQ(N),N=IV,IH)
DTAPOS=DTAPOS+1
RECORD=RECORD-1
IF(RECORD.LE.0) GO TO 3200
IX=IX+128
GO TO 3100

C
C NOW WE HAVE TO SET UP THE CALIBRATION BLOCKS:
3200 IF(CALPOS.EQ.IOLDCL) GO TO 3300
IOLDCL=CALPOS
C CHECKING TO SEE IF POINTER MAKES SENSE:
IF((CALPOS.LE.0).OR.(CALPOS.GT.HORNUM)) GO TO 3500
C IF IT DOES, EXECUTE CHARACTER SWAPCH ROUTINE:
IANUM=256*(CALPOS/10+42)+MOD(CALPOS,10)+42
CALL MOVE('CAL',0,NAME,0,3)
CALL MOVE(IANUM,0,NAME,3,2)
DO 10 I=2,HORNUM
READ(2'I)IBUFA
IF(EQUAL(IBUFA,0,NAME,0,5)) GO TO 33
10 CONTINUE
PUT ' THIS CALIBRATION SET COULD NOT BE FOUND.'
GO TO 950
33 CALPOS=I
CALNUM=0
J=1
3210 IZ=CALPOS+J-1
READ(2'IZ)IBUFA
IF(.NOT.(EQUAL(IBUFA,0,NAME,0,5)))GO TO 3300
C WE IGNORE ALL BLOCKS WITH SALT>=10000.
IF(IBUFA(17).GE.10000) GO TO 3400
IF(CALNUM.LE.0) GO TO 3240
C IF TWO CALIBRATION BLOCKS WERE TAKEN AT THE SAME CONCENTRATION,
C WE REPLACE THE OLD ONE WITH THE NEW ONE:
DO 3212 N=1,CALNUM
IF(SALT(N).EQ.IBUFA(17)) GO TO 3230
3212 CONTINUE
GO TO 3240
3230 IX=1
DTAPOS=IBUFA(6)
RECORD=IBUFA(7)
3231 IH=IX+127
READ(3'DTAPOS)(CALIB(N,IZ),IZ=IX,IH)
DTAPOS=DTAPOS+1
RECORD=RECORD-1
IF(RECORD.LE.0) GO TO 3400
IX=IX+128
GO TO 3231
3240 SALT(J)=IBUFA(17)

```

USER=TUNG 255 84170 JOINT COMPUTER FACILITY, MIT PAGE 6

SUNDAY

```

      CALNUM=CALNUM+1
      DTAPOS=IBUFA(5)
      RECORD=IBJFA(7)
      IV=1
3250  IV=IX+127
      READ(3*DTAPOS)(CALIB(J,N),N=IV,IH)
      DTAPOS=DTAPOS+1
      RECORD=RECORD-1
      IF(RECORD.LE.0) GO TO 3400
      IX=IX+128
      GO TO 3250
3400  J=J+1
      GO TO 3210
C END OF ROUTINE FOR PENDING IN CALIBRATION SETS.
C
C NOW WE HAVE TO SET UP THE CALIBRATION CORRECTION BLOCK:
3300  IF(CORPOS.LE.0) GO TO 3305
      IANUM=256*(CORPOS/10 +48)+100*(CORPOS,10)+48
      CALL MOVE('CAL',,NAME,0,3)
      CALL MOVE(IANUM,0,NAME,3,2)
      DO 3301 I=2,NDRNUM
      READ(2*I)IBUFA
      IF(EQUAL(IBUFA,0,NAME,0,5)) GO TO 3302
3301  CONTINUE
      PUT ' CALIBRATION CORRECTION BLOCK CANNOT BE FOUND.'
      GO TO 950
3302  IX=1
      DTAPOS=IBUFA(5)
      RECORD=IBUFA(7)
3303  IV=IX+127
      READ(3*DTAPOS)(CALCOR(N),I=IV,IH)
      RECORD=RECORD-1
      DTAPOS=DTAPOS+1
      IF(RECORD.LE.0) GO TO 3304
      IX=IX+128
      GO TO 3303
C WE MUST SEARCH FOR THE CALIBRATION BLOCK TAKEN AT ZERO SALT
C CONCENTRATION, AS THE CALIBRATION CORRECTION BLOCK WAS TAKEN
C UNDER THE SAME CONDITIONS:
3304  DO 1234 IZERO=1,CALNUM
      IF(SALT(IZERO).EQ.0) I=IZERO
1234  CONTINUE
      DO 3309 N=1,PRNUM
C NOW WE SHIFT THE MIXING DATA TO CORRECT FOR THE EFFECTS OF
C TEMPERATURE AND PROBE OXIDATION:
      MYDATA(N)=MYDATA(N)+CALIB(0,N)-CALCOR(N)
3309  CONTINUE
C END OF CALIBRATION CORRECTION PROCEDURE.
C
C THIS PART OF THE PROGRAM CONVERTS THE R.D. RESISTANCE VALUES
C INTO SALT CONCENTRATIONS BY LINEARLY INTERPOLATING WITH THE
C CALIBRATION SERIES:
3310  IF(CALNUM.LE.0) PUT(ICI,3300)

```

US=TUNG 255 84170 JOINT COMPUTER FACILITY, MIT PAGE 7

SUNDAY

```

3306  FORMAT(' THERE ARE NOT ENOUGH CALIBRATION BLOCKS. ')
      IF(CALNUM.LE.0) GO TO 950
      DO 3800 J=1,PRONUM
      ILO=0
      IHI=20000
      DO 3700 N=1,CALNUM
      IF(CALIB(N,J).LT.*XDATA(J)) GO TO 3701
      IF(CALIB(N,J).GT.IHI) GO TO 3700
      IHI=CALIB(N,J)
      IZ=SALT(N)
      GO TO 3700
3701  IF(CALIB(N,J).LT.ILO) GO TO 3700
      ILO=CALIB(N,J)
      IH=SALT(N)
3710  CONTINUE
C BY NOW WE SHOULD HAVE ILO JUST BELOW THE VALUE OF THE MINING
C DATA AT THIS POINT, AND IHI JUST ABOVE. IF AND IF ARE CORRESPONDING
C SALT CONCENTRATIONS:
C CHECKING TO SEE IF IT IS POSSIBLE TO INTERPOLATE:
3710  IF((ILO.LE.*XDATA(J)).AND.(IHI.GE.*XDATA(J))) GO TO 3720
C IF NOT, WE SET THE VALUE TO ZERO AND CONTINUE:
      RATIO(J)=0
      GO TO 3800
C THE NEXT SEVERAL LINES OF CODE DOES THE ACTUAL INTERPOLATION:
3720  IF(ILO.EQ.IHI) RATIO(J)=IY
      IF(ILO.EQ.IHI) GO TO 3800
      AREAL=*XDATA(J)-ILO
      BREAL=IHI-ILO
      AREAL=AREAL/BREAL
      BREAL=IY
      ARG=IZ-IH
      RATIO(J)=BREAL+ARG*AREAL
      IF(IHI.LE.20000) RATIO(J)=0.0
3800  CONTINUE
C THIS ENDS THE BASIC COMPUTATIONAL CODE.
C
C 39 WE STORE IN X(N) THE SALT VALUES IN PROPER ORDER
3900  DO 500 J=1,PRONUM
      IX=PROSEQ(N)
      X(IX)=RATIO(N)
500  CONTINUE
C
      VAVG=IGPY
      VAVG=VAVG/3.746
C NOW WE HAVE TO SEARCH FOR THE FLOWSPLIT DATA:
      SUM=50000.0
      ARG=IREYNO
      DO 250 N=1,IFLOW4
      IF(ABS(ARG-FLOW(4,N)).GT.SUM) GO TO 250
      SUM=ABS(ARG-FLOW(4,N))
      ISTORE=N
250  CONTINUE
C END OF SEARCH FOR FLOWSPLIT DATA..

```

USER=TUNG 255 94170 JOINT COMPUTER FACILITY, MIT PAGE 8

SUNDAY

C INITIALIZING DETECTED SALT AT ZERO.

OUTSLT=0.0

C ADDING UP THE DETECTED SALT FOR THE INTERIOR CHANNELS:

DO 3940 N=1,96

3940 OUTSLT=OUTSLT+FLOW(1,ISTORE)*AREA(1)*Y(N)*VAVG

C ADDING UP THE DETECTED SALT FOR THE EDGE CHANNELS:

DO 3950 N=1,24

IEDG=F(N)

3950 OUTSLT=OUTSLT+FLOW(2,ISTORE)*AREA(2)*Y(IEDG)*VAVG

C ADDING UP THE DETECTED SALT FOR THE CORNER CHANNELS:

DO 3960 N=25,30

ICOR=F(N)

3960 OUTSLT=OUTSLT+FLOW(3,ISTORE)*AREA(3)*Y(ICOR)*VAVG

C NET SALT INJECTED IS THE PRODUCT OF FLOW AND CONC.:

NETSLT=ISALTF*ISALTC

C THE MASS-BALANCE RATIO IS THE RATIO OF SALT DETECTED TO SALT

C INJECTED, AND IDEALLY =1.

MASBAL=OUTSLT*100000.0/NETSLT

C THAT ENDS THE BASIC COMPUTATIONAL ROUTINES.

C THIS IS THE PRINTOUT ROUTINE:

IF((IDNE.EQ.0).AND.(ICONCE.EQ.0)) GO TO 119

IF(IDNE.EQ.0) GO TO 433

WRITE(ICI,95)

95 FORMAT(' DO YOU WANT TRUE SALT CONC. PRINTED OUT?')

READ(ICI,3011) IANS

IF(EQUAL(IANS,0,'N',0,1)) GO TO 96

433 IYSALT=1

GO TO 97

96 WRITE(ICI,96)

96 FORMAT(' DO YOU WANT A PRINTOUT OF DIMENSIONLESS S.BAL?')

READ(ICI,3011) IANS

IF(EQUAL(IANS,0,'N',0,1)) GO TO 950

119 IYSALT=

IF((IDNE.EQ.0).AND.(IDIMEN.EQ.0)) GO TO 397

IF(ISUBCH.EQ.6) CMI=ISALTF* 0.00001+VAVG*FLOW(1,ISTORE)*AREA(1)

IF(ISUBCH.EQ.120) CMI=ISALTF* 0.00001+VAVG*FLOW(2,ISTORE)*AREA(2)

DO 392 N=1,PRONUM

392 Y(N)=Y(N)/(OUTSLT/CMI)*100.0

GO TO 97

C CODE FOR PLOTTING A MAP OF MIXING DATA.

C THE FOLLOWING CODE IS FOR A 126 CHANNEL RUN ONLY.

90 FORMAT('1 SALT CONCENTRATION DISTRIBUTION END: '//)

91 FORMAT(1H,12X,6F6.1)

92 FORMAT(1H,12X,6F6.1)

93 FORMAT(1H,15X,5F6.1)

94 FORMAT(1H,9X,7F6.1)

95 FORMAT(1H,12X,6F6.1)

96 FORMAT(1H,6X,8F6.1)

97 FORMAT(1H,3X,7F6.1)

98 FORMAT(1H,3X,2F6.1)

99 FORMAT(1H,6X,3F6.1)

90 FORMAT(1H,6F6.1,4X,6F6.1)

92 FORMAT('1')

```

97  IXI=5
    DO 4444 JPRIN=1,2
    IF(JPRIN.EQ.2)IXI=6
    WRITE(IXI,80)
    IF(IYSALT.EQ.1) WRITE(IXI,395)
395  FORMAT(' SALT CONCENTRATION(GM.SALT/LBWATER)*1000.')
```

IF(IYSALT.EQ.0) WRITE(IXI,396)

```
396  FORMAT(' DIMENSIONLESS RATIOS OF SALT CONCENTRATION*0.01')
    WRITE(IXI,481)MXNAME
481  FORMAT(1X,5A2)
    WRITE(IXI,2011) IREYNO
    ARG=IAXIAL
    ARG=ARG/100.
    WRITE(IXI,2014) ARG
    WRITE(IXI,2013) ISUBCH
    WRITE(IXI,1111) FLOW(1,ISTORE),FLOW(2,ISTORE),FLOW(3,ISTORE)
1111 FORMAT(' FLOW SPLITS      :',/, ' Y1=',F7.4,
2  ' X2=',F7.4, ' X3=',F7.4)
    WRITE(IXI,111) MASBAL
111  FORMAT(' MASSBALANCE RATIO  :',F6.3)
    WRITE(IXI,3000)
    WRITE(IXI,81) M(107),M(106),M(105),M(104),M(103),M(102)
    IF(JPRIN.EQ.1) WRITE(IXI,3000)
    WRITE(IXI,82) M(100),M(68),M(66),M(64),M(62),M(61)
    IF(JPRIN.EQ.1) WRITE(IXI,3000)
    WRITE(IXI,83) M(69),M(67),M(65),M(63),M(61)
    IF(JPRIN.EQ.1) WRITE(IXI,3000)
    WRITE(IXI,84) M(109),M(70),M(34),M(32),M(30),M(60),M(100)
    IF(JPRIN.EQ.1) WRITE(IXI,3000)
    WRITE(IXI,85) M(71),M(35),M(33),M(31),M(29),M(59)
    IF(JPRIN.EQ.1) WRITE(IXI,3000)
    WRITE(IXI,86) M(110),M(72),M(35),M(12),M(10),M(20),M(53),M(94)
    IF(JPRIN.EQ.1) WRITE(IXI,3000)
    WRITE(IXI,87) M(73),M(37),M(13),M(11),M(9),M(27),M(57)
    IF(JPRIN.EQ.1) WRITE(IXI,3000)
    WRITE(IXI,88) M(111),M(74),M(38),M(14),M(2),M(8),M(20),M(56),M(9)
    IF(JPRIN.EQ.1) WRITE(IXI,3000)
    WRITE(IXI,89) M(75),M(30),M(15),M(3),M(1),M(7),M(25),M(57)
    IF(JPRIN.EQ.1) WRITE(IXI,3000)
    WRITE(IXI,90) M(112),M(97)
    IF(JPRIN.EQ.1) WRITE(IXI,3000)
    WRITE(IXI,89) M(76),M(40),M(16),M(4),M(6),M(24),M(54),M(9)
    IF(JPRIN.EQ.1) WRITE(IXI,3000)
    WRITE(IXI,83) M(113),M(77),M(41),M(17),M(5),M(23),
2  M(53),M(35),M(126)
    IF(JPRIN.EQ.1) WRITE(IXI,3000)
    WRITE(IXI,87) M(78),M(42),M(18),M(20),M(22),M(72),M(4)
    IF(JPRIN.EQ.1) WRITE(IXI,3000)
    WRITE(IXI,86) M(114),M(79),M(43),M(14),M(21),M(51),M(33),M(12)
    IF(JPRIN.EQ.1) WRITE(IXI,3000)
    WRITE(IXI,85) M(90),M(44),M(40),M(42),M(50),M(9)
    IF(JPRIN.EQ.1) WRITE(IXI,3000)
    WRITE(IXI,84) M(115),M(81),M(45),M(47),M(49),M(91),M(94)
```

USER=TUNG 255 34170 JOINT COMPUTER FACILITY, MIT PAGE 10

SUM

```

IF(JPRIN.EQ.1) WRITE(IXI,3000)
WRITE(IXI,83) M(82),M(84),M(86),M(88),M(90)
IF(JPRIN.EQ.1) WRITE(IXI,3000)
WRITE(IXI,82) M(116),M(83),M(85),M(87),M(89),M(123)
IF(JPRIN.EQ.1) WRITE(IXI,3000)
WRITE(IXI,81) M(117),M(118),M(119),M(120),M(121),M(122)
IF(JPRIN.EQ.1) WRITE(IXI,3000)

```

4444 CONTINUE

IXI=5

IF((IYSALT.EQ.1).AND.(IONE.EQ.0)) GO TO 119

IF(IYSALT.EQ.1) GO TO 96

C THIS ENDS THE PRINTING FOR THE 126 CHANNEL DATA.

C PUNCHED CARD OUTPUT ROUTINE

390 IF(IPUNCH.EQ.1) GO TO 950

IF(IONE.EQ.0) GO TO 432

WRITE(ICI,101)

101 FORMAT(' DO YOU WANT PUNCHED CARD OUTPUT?')

READ(TCI,3011) IANS

IF(IANS.EQ.0) GO TO 950

432 WRITE(LU,145) ('(Q),Q=1,PRONUM')

145 FORMAT(16F5.1)

950 IY=IY+1

IF(IONE.EQ.1) GO TO 106

GO TO 210

9000 PUT ' PROGRAM ENDS NOW'

END

.U	2254[VI]	F	2D90[IV]	AVGGE*	2D94[VI]	V*VS	2D97[VI]	SETSL*
OUTSLT	2D90[VI]	ISALTE	2D94[IV]	ISALTC	2D97[VI]	"	2D97[VI]	SETSL*
CU*	35AC[VI]	NAME	35B6[VI]	NAME*	35C0[VI]	FLO*	3700[VI]	NAME*
TOTALF	3710[VI]	IFLOX*	3712[VI]	PTIC	3712[VI]	CLINUM	3714[VI]	NAME*
	3718[VI]	PRONUM	371A[VI]	SALT	372E[VI]	37C0F0	3730[VI]	NAME*
PCOSID	4730[VI]	CALCOR	4830[VI]	CALIP	7330[VI]	SETP00	7332[VI]	CLP00*
UTAFCS	7336[VI]	CORPOS	7338[VI]	TRUFA	7363[VI]	120	7360[VI]	120*
REAL	7374[VI]	CREAL	7378[VI]	DEPAL	737C[VI]	NAME*	000[VI]	0
ITEXT	7382[VI]	IHRD	7384[VI]	IDAT	7386[VI]	LU	7387[VI]	120*
IONE	7390[VI]	IREGIN	0096[LI]	519	0000[LI]	-I	7394[VI]	120*
IXI	7397[VI]	ICI	00A[LI]	497	00F8[LI]	490	0100[LI]	3011
3000	0110[LI]	2011	0130[LI]	2013	0160[LI]	2014	0190[LI]	100
491	0260[LI]	493	0284[LI]	494	0404[LI]	490	0500[VI]	100
495	03F8[LI]	496	0000[LI]	97	73A2[VI]	IX	0430[LI]	100
IANS	0530[LI]	180	0512[LI]	9000	040E[LI]	100	0432[LI]	100
103	73A6[VI]	I	05B0[LI]	104	039C[LI]	700	0500[LI]	115
BJPUN	0672[LI]	9090	73AC[VI]	ISTART	73A7[VI]	ICOR07	7370[VI]	120*
4321	07F2[LI]	200	00F0[LI]	950	0378[LI]	3014	000[LI]	0
13PM	73B0[VI]	IAXIAL	73B4[VI]	IPRYNO	73B7[VI]	ISURCH	7320[VI]	IX
3015	73BE[VI]	I	0A10[LI]	3013	73C4[VI]	ICL007	0015[LI]	3200
TANUM	0000[LI]	MOD	0000[LI]	MOVE	0AF2[LI]	00	004E[LI]	04
00	0382[LI]	3100	73D4[VI]	ICL007	0F08[LI]	3300	0070[LI]	1
73	73D0[VI]	J	0D62[LI]	3210	73E3[VI]	IT	0070[LI]	3400
3240	0000[LI]	3210	0D14[LI]	3230	07B4[LI]	3024	0010[LI]	000
3305	1076[LI]	3301	10D8[LI]	3302	10F0[LI]	3303	1100[LI]	3304
1234	73E0[VI]	IZERO	1220[LI]	3309	1250[LI]	3310	1006[LI]	000
120	73E7[VI]	I I	1210[LI]	3700	1360[LI]	3701	1300[LI]	371

FLOPPY DISK INITIALIZATION PROGRAM

USER=TUNG 255 84170 JOINT COMPUTER FACILITY, MIT PAGE 2

SUNDAY

```

      DO 10 J=1,60
      10 MAINS(J)=MAINS1(J)
C INITIALIZE ARRAY MAINS
      CALL SETCHN(MAINS,16)
C GO TO MAINLINE GLSMATN WHICH CONTAINS THE COMMAND PROCESSOR
      ICMD=1
1000 DO 2 I=1,2
      DO 2 J=1,24
      2 MSET(I,J)=0
      DO 3 I=1,2
      DO 3 J=1,512
      3 MCHAN(I,J)=0

C SPECIFY THE INPUT, OUTPUT DEVICES LOGIC UNIT #S
      ICN=6
      IPR=6
      ICMD=5

C INDIC-TE TOTAL #S OF HEADER BLOCKS AND DATA BLOCKS
      ITR=140
      ITH=840
      MAXSET=3

      1 FORMAT(I5)
C READ PTS FROM HEADER FILE OR, IF FOR HEADER FILE, IS FOR DATA FILE
      READ(2,1) IPUFA
      IP=IBUFA(1)
      IQ=IBUFA(2)
      NSET=IBUFA(3)
      NCAL=IBUFA(4)
      NMIA=IBUFA(5)

C INITIALIZATION OF VARIABLES IN CIL AND MIX BLOCKS
      TEMPIC=0.
      ICCON=0
      IPROH=0
      IPOL=0
      TEMPIN=0.
      IACON=0
      SIFLOW=0.
      BFLOW=0.
      ICUE=0
      NLEVEL=1
      NAVG=1
      MAYG=1

C SET DATE AND TIME
C SET DATE IS DONE BY CALLING SUB INDATE, 'IASK' IS SET (-1) AND
C PASSED TO SUB INDATE TO INDICATE THAT SUB INDATE IS CALLED TO GET A
C NEW DATE IN
      IASK=-1
      CALL SETDAT
C *****

```


US TUNG 255 84170 JOINT COMPUTER FACILITY, "IT" PAGE 3

SUNDAY

PUT 6,0,' TIME(HR,MIN):'

READ(ICN,2222)IHR,IMIN

2222 FORMAT(2I5)

CALL SETTIM(IHR,IMIN)

ITIME=IHR*256+IMIN

C *****

CALL LINKER(MAINS,1)

900 WRITE(6 ,905)IEP

WRITE(5 ,905)IEP

905 FORMAT(5X,I5,'FILE EPROR')

STOP

END

//	0000[V]	ITA	0002[V]	ITH	0004[V]	IA	0006[V]	IH
IS	0008[V]	IIS	0010[V]	IF	0012[V]	IERPOR	0014[V]	IDATE
ITIME	0016[V]	ID	0018[V]	IREFER	0020[V]	IPTH	0022[V]	42
E	0024[V]	HSOUND	0026[V]	LSOUND	0028[V]	IETSFT	0030[V]	IPTT
JMON	0032[V]	JDAY	0034[V]	JYF	0036[V]	JHR	0038[V]	JH
JSEC	0040[V]	JMLSEC	0042[V]	JDATE	0044[V]	PERIOD	0046[V]	TDURA
IOLF	0048[V]	IMON	0050[V]	IDAY	0052[V]	IYR	0054[V]	IHP
IMIN	0056[V]	ISEC	0058[V]	IMLSEC	0060[V]	IMTTPR	0062[V]	TABLE
IERI	0064[V]	BFLOW	0066[V]	TEMPIN	0068[V]	ICTF	0070[V]	TEMPIN
AREA	0072[V]	DE	0074[V]	AMPL	0076[V]	IASK	0078[V]	IN
IT	0080[V]	NSFT	0082[V]	NCAL	0084[V]	VMIX	0086[V]	MM
EQUIT	0088[V]	MAXSET	0090[V]	MSET	0092[V]	MCHAN	0094[V]	INELD
IBUFE	0096[V]	IP	0098[V]	LEVELS	0100[V]	IMIX	0102[V]	ICAL
IBUFC	0104[V]	BUFB	0106[V]	IBUFA	0108[V]	IBUFR	0110[V]	NAME
ICOMD	0112[V]	ICN	0114[V]	IPR	0116[V]	IECHO	0118[V]	IRMAIN
IPNT	0120[V]	IMCON	0122[V]	ICUP	0124[V]	WAVG	0126[V]	SUFLOW
IPROH	0128[V]	IPROL	0130[V]	IUFLOW	0132[V]	WAVG	0134[V]	W
LEVEL	0136[V]	ICMD	0138[V]	ICCON	0140[V]	MAINS	0142[V]	ITIME
ICRCAL	0144[V]	.U	0146[V]	INPUT	0148[V]	EQUAL	0150[V]	MAINS
IZ	0152[V]	DEFI	0154[V]	IER	0156[V]	900	0158[V]	90
ITEXT	0160[V]	IPRD	0162[V]	IDAT	0164[V]	10	0166[V]	1
ETCHA	0168[V]	1000	0170[V]	2	0172[V]	7	0174[V]	2
1	0176[V]	SETDAT	0178[V]	OX	0180[V]	2222	0182[V]	22
SETTIM	0184[V]	LINKER	0186[V]	905	0188[V]	.C	0190[V]	.V

PROGRAM WILL OCCUPY X'0463' BYTES OF STORAGE SPACE

MAIN HAS NO EPRORS

IT >

3

3

3

S-LIB,HTLTIMER

IUP DOBS CHAIN CLASS-LIB HTLTIMER PCL

APPENDIX III

LISTING OF SUPERENERGY/DRY

```

REAL LENGTH, LEAD, LXX
INTEGER T                                00000030
COMMON/PROP1/RHO, CP, COND, CONDW, ALPHA 00000040
COMMON/GEO /DR(1), DW(1), LEAD(1), P(1), DFLAT(1), POD(1), GAP(1), A1(1), 00000050
1A2(1), A3(1), A4(1), A5(3), A6(1), A7(1), ATOTAL(1), ATOTLG(1), DE1(1), 00000060
2DE2(1), DE3(1), DE6(1), DE7(1), HP1(1), HP2(1), HP3(1), HPTOTL(1), 00000070
3ETA1(1), ETA2(1), ETA3(1), ETA4(1), ETA6(1), ETA8(1), X1(1), X2(1), X3(3), 00000080
4S1(1), S2(1), LENGTH(1), V1R(1), V2R(1), V3R(1), V6R(1), V7R(1) 00000090
COMMON/NUM /NROD(1), NRING(1), NCHANL(1), N1(1), N2(1), N3(1), N4(1),
1N5(1), N6(1), N7(1), N8(1), LC(1, 550, 4), MROD(1, 440, 3)
COMMON/TEMP/T(550)
COMMON/WEND/Y11(500), NINJ, LXX, REE
COMMON FACTOR(3, 3)                                00000130
DIMENSION SALT(500), CINDI(50)
DIMENSION FLOW(01), FLOWGP(01), G(01), GGAP(01), VAVG(01), VAVGG(01), 00000140
1V1(01), V2(01), V3(01), V6(01), V7(01), VSWIRL(01)
DIMENSION ITYPE(01), NOOFAX(01), AXIAL(50, 01), FAX(50, 01), NA(01, 6) 00000160
DIMENSION EH1(01), EH2(01), EH3(01), EH4(01), Y(500)
DIMENSION PE2(01), PE3(01), PE6(01), PE7(01), NU2(01), NU3(01), NU6(01), 00000180
1NU7( 1), RES2(01), RES3(01), RES6(01), RES7(01)
DIMENSION QTOTAL(01), DIFFH(01), DIFFHG(01), QSA(01), QSB(01), QSC(01) 00000200
DIMENSION QTPRIM(550, 01), QTP(550, 01), SUMM(440, 01), POWER(220) 00000 10
DIMENSION T1(550, 01), T2(550, 01), T3(550, 01), QAVG(01), TAVG(01) 00000220
DIMENSION PRNT(30)
DO 1234 IJK=1, 3
IWC=C
READ(5, 10) JXQ, NCDE, NINJ, NROD(1), IW, IWC, NOPRNT
IF(NOPRNT.EQ.0) NOPRNT=1
IF(IWC.EQ.1) READ(5, 9978) (PRNT(I), I=1, NOPRNT)
9978 FORMAT(8E10.3)
READ(5, 50) CH1UA, CH1UI, CH1UC, REE, FLOW(1)
FACTOR(1, 1)=1.                                00000270
FACTOR(1, 2)=0.5                                00000280
FACTOR(2, 1)=2.                                00000290
FACTOR(2, 2)=1.                                00000300
CONTINUE                                         00000310
TT=600.0
FLOW(1)=FLOW(1)*500.57
REE=REE*1000.0
TIN= .
S1(1)=0
S2(1)=0
CALL PROP(TT)
NOASSM=1
KOND= ?
INDIC=0
NOINCR=500
ISOLAT=2
10 FORMAT(7I5)
K=1
WRITE(6, 16)                                00000510
16 FORMAT(1X, 8CHROD DIA. WIRE DIA. LEAD PITCH FLATS LENGTH) 00000520
1GTH DUCT TH. GAP TH. ) 00000530
READ(5, 50) DR(K), DW(K), LEAD(K), P(K), DFLAT(K), LENGTH(K), DDWL
IF(IWC.EQ.1) GO TO 9977
PRNT(1)=LENGTH(1)
NOPRNT=1
9977 CONTINUE

```

```
40 FORMAT(8F10.3)
   WRITE(6,40)DR(K),DW(K),LEAD(K),P(K),DFLAT(K),LENGTH(K)
50 FORMAT(8E10.3)
   FLOWGP(1)=0.0
   QAVG(1)=0.0
C   AXIAL FLUX DISTRIBUTION AS A FUNCTION OF AXIAL HEIGHT
   CALL NUM0(K)
   CALL GEOM(K)
   IF(IW.EQ.0) GO TO 5000
   GO TO 5001
5000 READ(5,50) V1R(1),V2R(1),V3R(1)
5001 CONTINUE
   M2=N2(K)
   M3=N3(K)
   M4=N4(K)
   M6=N6(1)
   MK2=M2+1
   MK3=M3+1
   IF(IWC.EQ.1) GO TO 9974
   READ(5,9997) (SALT(I),I=1,M4)
9997 FORMAT(10E5.0)
   SALT(121)=SALT(97)
   SALT(122)=SALT(102)
   SALT(123)=SALT(107)
   SALT(124)=SALT(112)
   SALT(125)=SALT(117)
   SALT(126)=SALT(122)
   DO 9993 I1=1,4
   SALT(96+I1)=SALT(97+I1)
   SALT(100+I1)=SALT(102+I1)
   SALT(108+I1)=SALT(112+I1)
   SALT(104+I1)=SALT(107+I1)
   SALT(112+I1)=SALT(117+I1)
   SALT(116+I1)=SALT(122+I1)
9993 CONTINUE
9974 CONTINUE
   DO 7776 JXS=1,JXQ
   DO 30 L=1,NOASSM
   K=1
   IF(INCOE.EQ.1) GO TO 9999
   EHSTA1=CHIUC
   EHSTA2=CHIUC
   EHSTA3=CHIUC
   EHSTA4=CHIUC
   IF(IWC.EQ.1) C1=CHIUUA
   IF(IWC.NE.1) C1=(CHIUUA-CHIUI)/FLOAT(JXQ-1)*FLOAT(JXS-1)+CHIUI
   GO TO 9998
9999 CONTINUE
   C1=CHIUC
   IF(IWC.EQ.1) CHIUK=CHIUUA
   IF(IWC.NE.1) CHIUK=(CHIUUA-CHIUI)/FLOAT(JXQ-1)*FLOAT(JXS-1)+CHIUI
   EHSTA1=CHIUK
   EHSTA2=CHIUK
   EHSTA3=CHIUK
   EHSTA4=CHIUK
9998 CONTINUE
   G(L)=(FLOW(L)/ATOTAL(K))*144.0
   VAVG(L)=G(L)/RHO
```

00000810
0000082000000830
00000840
0000085000000870
00000880

00000440

00001120
00001140

LEVEL 21

MAIN

DATE = 77237

16/16/36

```

V1(L)=VAVG(L)*V1R(K)                                00001160
V2(L)=VAVG(L)*V2R(K)                                00001170
V3(L)=VAVG(L)*V3R(K)                                00001180
CONVER=VAVG(L)*DR(K)/12.                             00001210
EH1(L)=EHSTA1*CONVER                                00001220
EH2(L)=EHSTA2*CONVER                                00001230
EH3(L)=EHSTA3*CONVER                                00001240
EH4(L)=EHSTA4*CONVER                                00001250
VSWIRL(L)=C1*VAVG(L)                                00001260
RES2(L)=10000.0
RES3(L)=10000.0
30 CONTINUE
  IPRINT=1
  PRINT=PRNT(IPRINT)
  LXX=PRINT
  DZ=LENGTH(1)/NOINCR
  Z=DZ/2.                                             00001510
  DO 115 L=1,NOASSM                                00001520
    M6=N6(L)
    DO 110 I=1,M4
      T2(I,L)=TIN                                     00001590
110 CONTINUE                                         00001600
      T2(NINJ,L)=700.0
115 CONTINUE                                         00001610
1000 CONTINUE                                       00001620
120 CONTINUE                                       00001710
    DO 140 L=1,NOASSM                                00001720
      K=1
      M1=N1(K)                                       00001740
      M2=N2(K)                                       00001750
      M3=N3(K)                                       00001760
      M4=N4(K)                                       00001770
      M5=N5(K)                                       00001780
      M6=N6(K)                                       00001790
      MK1=N1(K)+1                                    00001820
      MK2=N2(K)+1                                    00001830
      MK3=N3(K)+1                                    00001840
      MK4=N4(K)+1                                    00001850
      MK5=N5(K)+1                                    00001860
      MK6=N6(K)+1                                    00001870
      DO 150 I=1,M1                                  00001890
        MM1=LC(K,I,1)                                00001900
        MM2=LC(K,I,2)                                00001910
        MM3=LC(K,I,3)                                00001920
        T3(I,L)=T2(I,L)+((P(K)-DR(K))*DZ/
1A1(K))*((ALPHA+EH1(L))/V1(L))*(T2(MM1,L)+T2(MM2,L)+T2(MM3,L)-
23*T2(I,L))/ETA1(K)*12                                00001940
        00001950
150 CONTINUE                                         00001970
      DO 160 I=MK1,M2                                00001980
        MM1=LC(K,I,1)                                00001990
        MM2=LC(K,I,2)                                00002000
        MM3=LC(K,I,3)                                00002010
        R=(I-1)/6.                                    00002020
        N=SQRT(R)+1.0001
        IQ=(I-6*(N-1)*(N-1)-1)/(2*N-1)+0.0001      00002040
        IR=I-6*(N-1)*(N-1)-IQ*(2*N-1)+0.0001      00002050
        J=IR/2                                         00002060
        IF(IR.EQ.2*J) GO TO 1160                     00002070

```

G LEVEL 21

MAIN

DATE = 77237

16/16/36

```

      EHB=EH2(L)                                00002090
      ETAA=ETA1(K)                              00002100
      ETAB=ETA2(K)                              00002110
      GO TO 2160                                00002120
1160  CONTINUE                                00002130
      EHA=EH1(L)                                00002140
      EHB=EH1(L)                                00002150
      ETAA=ETA1(K)                              00002160
      ETAB=ETA1(K)                              00002170
2160  CONTINUE                                00002180
      T3(I,L)=T2(I,L)+((P(K)-DR(K))*DZ)/
      1(A1(K)*V1(L))*((ALPHA+EHA)*(T2(MM1,L)+T2(MM2,L)-2*T2(I,L))/ETAA+
      2(ALPHA+EHB)*(T2(MM3,L)-T2(I,L))/ETAB)*12 00002200
160  CONTINUE                                00002210
      DO 170 I=MK2,M3                         00002230
      MM1=LC(K,I,1)                             00002240
      MM2=LC(K,I,2)                             00002250
      MM3=LC(K,I,3)                             00002260
      IF(MM2.GT.M3) GO TO 1170                  00002270
      ETAA=ETA3(K)                             00002290
      EHA=EH3(L)                               00002300
      GO TO 2170                                00002310
1170  CONTINUE                                00002320
      ETAA=ETA4(K)                             00002330
      EHA=EH4(L)                               00002340
2170  CONTINUE                                00002350
      IF(MM3.GT.M3) GO TO 3170                  00002360
      ETAB=ETA3(K)                             00002370
      EHB=EH3(L)                               00002380
      GO TO 4170                                00002390
3170  CONTINUE                                00002400
      ETAB=ETA4(K)                             00002410
      EHB=EH4(L)                               00002420
4170  CONTINUE                                00002430
      MM4=LC(K,I,4)                             00002440
      RESI=RES2(L)                             00002280
      TTT=T2(I,L)
      T3(I,L)=T2(I,L)+( DDW1*DZ/A2(K))*
      1(VSWIRL(L)/V2(L))*(T2(MM2,L)-T2(I,L))+12*(P(K)-DR(K))*DZ/A2(K)* 00002460
      2((ALPHA+EH2(L))/V2(L))*(T2(MM1,L)-T2(I,L))/ETA2(K)+12*( DDW1*DZ/ 00002470
      3(V2(L)*A2(K)))*((ALPHA+EHA)*(T2(MM2,L)-T2(I,L))/ETAA+(ALPHA+EHB)* 00002480
      4(T2(MM3,L)-T2(I,L))/ETAB)
170  CONTINUE                                00002520
      DO 180 I=MK3,M4                         00002530
      MM1=LC(K,I,1)                             00002540
      MM2=LC(K,I,2)                             00002550
      MM3=LC(K,I,3)                             00002560
      TTT=T2(I,L)
      RESI=RES3(L)
      T3(I,L)=T2(I,L)+(GAP(K)*DZ/A3(K))*
      1(VSWIRL(L)/V3(L))*(T2(MM1,L)-T2(I,L))+ (GAP(K)*DZ/A3(K))*12* 00002580
      2((ALPHA+EH4(L))/V3(L))*(T2(MM1,L)+T2(MM2,L)-2*T2(I,L))/ETA4(K)
180  CONTINUE                                00002620
      DO 190 I=MK4,M6                         00003260
      MM1=LC(K,I,1)                             00003
      T3(I,L)=T3(MM1,L)
190  CONTINUE                                00003450
140  CONTINUE                                00003460

```

LEVEL 21

MAIN

DATE = 77237

16/16/36

```

DO 230 L=1,NOASSM                                00003470
K=1
CONTINUE
DO 240 I=1,M4
T2(I,L)=T3(I,L)                                00003510
240 CONTINUE                                00003520
230 CONTINUE                                00003530
Z=Z+DZ                                00003540
DIFF=PRINT-Z                                00003550
IF(DIFF.GT.DZ) GO TO 1000                    00003560
DO 250 L=1,NOASSM                                00003580
K=1
M2=N2(K)                                00003600
M3=N3(K)                                00003610
M4=N4(K)                                00003620
M5=N5(K)                                00003630
M6=N6(K)                                00003640
MK2=M2+1                                00003670
MK3=M3+1                                00003680
MK6=M6+1                                00003690
MK7=M7+1                                00003700
DO 260 I=1,M4
Y(I)=((T2(I,L)-TIN)/(7.0-0-TIN))*100.0
T(I)=Y(I)*10.0
260 CONTINUE                                00003740
T22=0                                00003760
T11=0                                00003750
T33=0                                00003770
T66=0                                00003780
T77=0                                00003790
IF(IWC.EQ.1) GO TO 9976
CINDI(JXS)=0.0
DO 9995 I=1,M4
CINDI(JXS)=CINDI(JXS)+(Y(I)-SALT(I))*2
9995 CONTINUE
CINDI(JXS)=CINDI(JXS)*0.5
IF(JXS.EQ.1) GO TO 9990
IF(CINDI(JXS).GT.CINDI(JXS-1)) INDIC=1+INDIC
DIFFHG(L)=0.0
9976 CONTINUE
QTOTAL(L)=0.0
DO 500 J=1,M2                                00003800
500 T11=T11+T2(J,L)                                00003810
DO 501 J=MK2,M3                                00003820
501 T22=T22+T2(J,L)                                00003830
DO 502 J=MK3,M4                                00003840
502 T33=T33+T2(J,L)                                00003850
TAVG(L)=(T11*V1(L)*A1(K)+T22*V2(L)*A2(K)+T33*V3(L)*A3(K))/
1(M2*V1(L)*A1(K)+(M3-M2)*V2(L)*A2(K)+(M4-M3)*V3(L)*A3(K)) 00003900
HIN=RHO*CP*(TIN*(M2*A1(K)*V1(L)+(M3-M2)*A2(K)*V2(L)+(M4-M3)*A3(K)*
1V3(L))+212.0*A1(K)*V1(L))/144.0/3412.0 00003910
HTOTAL=RHO*CP*(T11*A1(K)*V1(L)+T22*A2(K)*V2(L)+T33*A3(K)*V3(L))/
1144/3412 00003960
DIFFH(L)=HTOTAL-HIN 00003970
WRITE(6,503) TAVG(L),QTOTAL(L),DIFFH(L),DIFFHG(L) 00003990
503 FORMAT(1H1,1X,5HTAVG=,1X,F7.1/,1X,11HQTOTAL(KW)=,1X,F7.1/,1X,10HDI00004020
1FFH(KW)=,1X,F7.1/,1X,11HDIFFHG(KW)=,1X,F7.1/) 00004030
WRITE(6,506) L 00004040

```

3 LEVEL 21

MAIN

DATE = 77237

16/16/36

```
506 FORMAT(1X,37HTEMPERATURE DISTRIBUTION FOR ASSEMBLY,I5) 00004050
WRITE(6,508) PRINT
508 FORMAT(1X,'AXIAL HEIGHT=',F6.2,' INCHES')
IF(IWC.EQ.1) INDIC=1
IF(IWC.NE.1) GO TO 9967
DO 5556 I=1,M4
Y11(I)=Y(I)
5556 CONTINUE
9967 CONTINUE
IF(INDIC.EQ.1) CALL PRIN(K)
IF(IWC.EQ.1) GO TO 9975
PRINT 9994,CINDI(JXS-1)
9994 FORMAT(3X,' LEAST MEAN SQUARE INDICATOR=',F10.3)
WRITE(6,18) 00000610
18 FORMAT(///,4X,'FOUR EFFECTIVE EDDY DIFFUSIVITIES AND SWIRL VELOCIT
LY RATIO')
WRITE(6,40) EHWC1,EHWC2,EHWC3,EHWC4,CCC1
9975 CONTINUE
9990 CONTINUE
250 CONTINUE 00004070
DO 9979 I=1,M4
Y11(I)=Y(I)
9979 CONTINUE
EHWC1=EHSTA1
EHWC2=EHSTA2
EHWC3=EHSTA3
EHWC4=EHSTA4
CCC1=C1
IPRINT=IPRINT+1
IF(IPRINT.GT.NOPRNT) GO TO 2450
PRINT=PRNT(IPRINT)
LXX=PRINT
GO TO 2550
2450 PRINT=LENGTH(1)+10.0
2550 CONTINUE
IF(PRINT.GT.LENGTH(1)) GO TO 2000
GO TO 1000 00004090
2000 CONTINUE 00004100
3000 CONTINUE 00004110
7776 CONTINUE
1234 CONTINUE
STOP 00004120
END 00004130
```


16/16/36

LEVEL 21

PROP

DATE = 77237

```
SUBROUTINE PROP(TT)
COMMON/PROP1/RHO,CP,COND,CONDW,ALPHA
RHO=62.4
CP=1.000
COND=0.0001
ALPHA=COND/(RHO*CP)
CONDW=12.1
RETURN
END
```

00004140
00004150

00004190
00004200
00004210
00004220

LEVEL 21

CURVE

DATE = 77237

16/16/36

SUBROUTINE CURVE(FX,X,F,Y,N,J)	00004230
DIMENSION F(50,7),Y(50,7),N(7)	00004240
M=N(J)	00004250
10 DO 20 KK=1,M	00004260
IF(X-Y(KK,J))30,15,20	00004270
15 IF(KK.EQ.M) GO TO 40	00004280
20 CONTINUE	00004290
GO TO 60	00004300
30 IF(KK.EQ.1) GO TO 60	00004310
40 B=(X-Y(KK-1,J))/(Y(KK,J)-Y(KK-1,J))	00004320
50 FX=F(KK-1,J)+B*(F(KK,J)-F(KK-1,J))	00004330
RETURN	00004340
60 PRINT 61	00004350
61 FORMAT(1X,'PROG.STOPPED.IN CURVE')	00004360
RETURN	00004370
END	00004380

LEVEL 21

PRIN

DATE = 77237

16/16/36

```

SUBROUTINE PRIN(K)
REAL LXX
COMMON/TEMP/T(550)
COMMON/NUM /NROD(1),NRING(1),NCHANL(1),N1(1),N2(1),N3(1),N4(1),
IN5(1),N6(1),N7(1),N8(1),LC(1,550,4),MROD(1,440,3)
COMMON/WEND/S(500),NINJ,LXX,REE
IF(NRING(K).EQ.4) GO TO 600
IF(NRING(K).EQ.8) GO TO 700
GO TO 800
600 CONTINUE
PRINT 601
PRINT 602
PRINT 601
PRINT 603,LXX,NINJ,REE
PRINT 601
PRINT 604,S(123),S(104),S(103),S(102),S(101),S(122)
PRINT 605,S(68),S(66),S(64),S(62)
PRINT 606,S(105),S(69),S(67),S(65),S(63),S(61),S(100)
PRINT 607,S(70),S(34),S(32),S(30),S(60)
PRINT 608,S(106),S(71),S(35),S(33),S(31),S(29),S(59),S(99)
PRINT 604,S(72),S(36),S(12),S(10),S(28),S(58)
PRINT 609,S(107),S(73),S(37),S(13),S(11),S(9),S(27),S(57),S(98)
PRINT 606,S(74),S(38),S(14),S(2),S(8),S(26),S(56)
PRINT 610,S(108),S(75),S(39),S(15),S(3),S(1),S(7),S(25),S(55),
1S(97)
PRINT 611,S(124),S(121)
PRINT 610,S(109),S(76),S(40),S(16),S(4),S(6),S(24),S(54),S(96),
1S(120)
PRINT 606,S(77),S(41),S(17),S(5),S(23),S(53),S(95)
PRINT 609,S(110),S(78),S(42),S(18),S(20),S(22),S(52),S(94),S(119)
PRINT 604,S(79),S(43),S(19),S(21),S(51),S(93)
PRINT 608,S(111),S(80),S(44),S(46),S(48),S(50),S(92),S(118)
PRINT 607,S(81),S(45),S(47),S(49),S(91)
PRINT 606,S(112),S(82),S(84),S(86),S(88),S(90),S(117)
PRINT 605,S(83),S(85),S(87),S(89)
PRINT 604,S(125),S(113),S(114),S(115),S(116),S(126)
601 FORMAT(///)
602 FORMAT(1H1,10X,'SALT CONCENTRATION DISTRIBUTION MAP CALIBRATED BY
1SUPERENERGY')
603 FORMAT(1H,4X,'DIMENSIONLESS RATIOS OF SALT CONCENTRATION * 0.01',
1/,5X,'AXIAL LEVEL (IN.) =',F10.1,/,5X,'INJ. SUBCHANNEL =',I5,
2/,5X,'REYNOLDS NUMBER =',F8.1)
604 FORMAT(1H,15X,6F6.1/)
605 FORMAT(1H,21X,4F6.1/)
606 FORMAT(1H,12X,7F6.1/)
607 FORMAT(1H,18X,5F6.1/)
608 FORMAT(1H,9X,8F6.1/)
609 FORMAT(1H,6X,9F6.1/)
610 FORMAT(1H,3X,10F6.1/)
611 FORMAT(1H,F6.1,55X,F6.1/)
GO TO 249
700 CONTINUE
WRITE(6,721) T(543),T(508),T(507),T(506),T(505),T(504),T(503),
1T(502),T(501),T(542)
WRITE(6,722) T(489),T(454),T(453),T(452),T(451),T(450),T(449),
1T(448),T(447),T(488)
WRITE(6,719) T(435),T(400),T(399),T(398),T(397),T(396),T(395),
1T(394),T(393),T(434)

```

WRITE(6,702) T(324),T(322),T(320),T(318),T(316),T(314),T(312),
1T(310) 00005010
00005020
WRITE(6,703) T(509),T(455),T(401),T(325),T(323),T(321),T(319),
1T(317),T(315),T(313),T(311),T(309),T(392),T(446),T(500) 00005030
00005040
WRITE(6,704) T(326),T(242),T(240),T(238),T(236),T(234),T(232),
1T(230),T(308) 00005050
00005060
WRITE(6,705) T(510),T(456),T(402),T(327),T(243),T(241),T(239),
1T(237),T(235),T(233),T(231),T(229),T(307),T(391),T(445),T(499) 00005070
00005080
WRITE(6,706) T(328),T(244),T(172),T(170),T(168),T(166),T(164),
1T(162),T(228),T(306) 00005090
00005100
WRITE(6,707) T(511),T(457),T(403),T(329),T(245),T(173),T(171),
1T(169),T(167),T(165),T(163),T(161),T(227),T(305),T(390),T(444),
1T(498) 00005110
00005120
WRITE(6,708) T(330),T(246),T(174),T(114),T(112),T(110),T(108),
1T(106),T(160),T(226),T(304) 00005140
00005150
WRITE(6,709) T(512),T(458),T(404),T(331),T(247),T(175),T(115),
1T(113),T(111),T(109),T(107),T(105),T(159),T(225),T(303),T(389),
2T(443),T(497) 00005160
00005170
WRITE(6,710) T(332),T(248),T(176),T(116),T(68),T(66),T(64),T(62),
1T(104),T(158),T(224),T(302) 00005180
00005190
WRITE(6,711) T(513),T(459),T(405),T(333),T(249),T(177),T(117),
1T(69),T(67),T(65),T(63),T(61),T(103),T(157),T(223),T(301),T(388),
2T(442),T(496) 00005200
00005210
WRITE(6,712) T(334),T(250),T(178),T(118),T(70),T(34),T(32),T(30),
1T(60),T(102),T(156),T(222),T(300) 00005220
00005230
WRITE(6,713) T(514),T(460),T(406),T(335),T(251),T(179),T(119),
1T(71),T(35),T(33),T(31),T(29),T(59),T(101),T(155),T(221),T(299),
2T(387),T(441),T(495) 00005240
00005250
WRITE(6,714) T(336),T(252),T(180),T(120),T(72),T(36),T(12),T(10),
1T(28),T(58),T(100),T(154),T(220),T(298) 00005260
00005270
WRITE(6,715) T(515),T(461),T(407),T(337),T(253),T(181),T(121),
1T(73),T(37),T(13),T(11),T(9),T(27),T(57),T(99),T(153),T(219),
2T(297),T(386),T(440),T(494) 00005280
00005290
WRITE(6,716) T(338),T(254),T(182),T(122),T(74),T(38),T(14),T(2),
1T(8),T(26),T(56),T(98),T(152),T(218),T(296) 00005300
00005310
WRITE(6,720) T(516),T(462),T(408),T(339),T(255),T(183),T(123),
1T(75),T(39),T(15),T(3),T(1),T(7),T(25),T(55),T(97),T(151),T(217),
2T(295),T(385),T(439),T(493) 00005320
00005330
WRITE(6,718) T(544),T(490),T(436),T(433),T(487),T(541) 00005340
00005350
WRITE(6,717) T(517),T(463),T(409),T(340),T(256),T(184),T(124),
1T(76),T(40),T(16),T(4),T(6),T(24),T(54),T(96),T(150),T(216),
2T(294),T(384),T(432),T(486),T(540) 00005360
00005370
WRITE(6,716) T(341),T(257),T(185),T(125),T(77),T(41),T(17),T(5),
1T(23),T(53),T(95),T(149),T(215),T(293),T(383) 00005380
00005390
WRITE(6,715) T(518),T(464),T(410),T(342),T(258),T(186),T(126),
1T(78),T(42),T(18),T(20),T(22),T(52),T(94),T(148),T(214),T(292),
2T(382),T(431),T(485),T(539) 00005400
00005410
WRITE(6,714) T(343),T(259),T(187),T(127),T(79),T(43),T(19),T(21),
1T(51),T(93),T(147),T(213),T(291),T(381) 00005420
00005430
WRITE(6,713) T(519),T(465),T(411),T(344),T(260),T(188),T(128),
1T(80),T(44),T(46),T(48),T(50),T(92),T(146),T(212),T(290),T(380),
2T(430),T(484),T(538) 00005440
00005450
WRITE(6,712) T(345),T(261),T(189),T(129),T(81),T(45),T(47),T(49),
1T(91),T(145),T(211),T(289),T(379) 00005460
00005470
WRITE(6,711) T(520),T(466),T(412),T(346),T(262),T(190),T(130),
1T(82),T(84),T(86),T(88),T(90),T(144),T(210),T(288),T(378),T(429),
2T(480),T(537) 00005480
00005490
WRITE(6,710) T(347),T(263),T(191),T(131),T(83),T(85),T(87),T(89), 00005500
00005510
00005520
00005530
00005540
00005550
00005560
00005570
00005580

1T(143),T(209),T(287),T(377)	00005590
WRITE(6,709) T(521),T(467),T(413),T(348),T(264),T(192),T(132),	00005600
1T(134),T(136),T(138),T(140),T(142),T(208),T(286),T(376),T(428),	00005610
2T(482),T(536)	00005620
WRITE(6,708) T(349),T(265),T(193),T(133),T(135),T(137),T(139),	00005630
1T(141),T(207),T(285),T(375)	00005640
WRITE(6,707) T(522),T(468),T(414),T(350),T(266),T(194),T(196),	00005650
1T(198),T(200),T(202),T(204),T(206),T(284),T(374),T(427),T(481),	00005660
2T(535)	00005670
WRITE(6,706) T(351),T(267),T(195),T(197),T(199),T(201),T(203),	00005680
1T(205),T(283),T(373)	00005690
WRITE(6,705) T(523),T(469),T(415),T(352),T(268),T(270),T(272),	00005700
1T(274),T(276),T(278),T(280),T(282),T(372),T(426),T(480),T(534)	00005710
WRITE(6,704) T(353),T(269),T(271),T(273),T(275),T(277),T(279),	00005720
1T(281),T(371)	00005730
WRITE(6,703) T(524),T(470),T(416),T(354),T(356),T(358),T(360),	00005740
1T(362),T(364),T(366),T(368),T(370),T(425),T(479),T(533)	00005750
WRITE(6,702) T(355),T(357),T(359),T(361),T(363),T(365),T(367),	00005760
1T(369)	00005770
WRITE(6,701) T(437),T(417),T(418),T(419),T(420),T(421),T(422),	00005780
1T(423),T(424),T(438)	00005790
WRITE(6,722) T(491),T(471),T(472),T(473),T(474),T(475),T(476),	00005800
1T(477),T(478),T(492)	00005810
WRITE(6,721) T(545),T(525),T(526),T(527),T(528),T(529),T(530),	00005820
1T(531),T(532),T(546)	00005830
701 FORMAT(32X,10I5)	00005840
702 FORMAT(37X,8I5/)	00005850
703 FORMAT(19X,15I5/)	00005860
704 FORMAT(34X,9I5/)	00005870
705 FORMAT(17X,16I5/)	00005880
706 FORMAT(32X,10I5/)	00005890
707 FORMAT(14X,17I5/)	00005900
708 FORMAT(29X,11I5/)	00005910
709 FORMAT(12X,18I5/)	00005920
710 FORMAT(27X,12I5/)	00005930
711 FORMAT(9X,19I5/)	00005940
712 FORMAT(24X,13I5/)	00005950
713 FORMAT(7X,20I5/)	00005960
714 FORMAT(22X,14I5/)	00005970
715 FORMAT(4X,21I5/)	00005980
716 FORMAT(19X,15I5/)	00005990
717 FORMAT(2X,22I5/)	00006000
718 FORMAT(1X,3I5,82X,3I5)	00006010
719 FORMAT(32X,10I5/)	00006020
720 FORMAT(2X,22I5)	00006030
721 FORMAT(22X,15,10X,8I5,10X,15)	00006040
722 FORMAT(27X,15,5X,8I5,5X,15)	00006050
GO TO 249	00006060
800 CONTINUE	00006070
NN=N8(K)	00006080
WRITE(6,801) (T(I),I=1,NN)	00006090
801 FORMAT(10I10/)	00006100
CONTINUE	00006110
RETURN	00006120
END	00006130

S LEVEL 21

NUMB

DATE = 77237

16/16/36

```

SUBROUTINE NUMB(L)                                00006140
COMMON/NUM /NR0D(1),NRING(1),NCHANL(1),N1(1),N2(1),N3(1),N4(1),
IN5(1),N6(1),N7(1),N8(1),LC(1,550,4),MROD(1,440,3)
NWALLS=1
NR0D=NR0D(L)                                       00006180
XNRIN=(NR0D-1)/3.                                  00006190
NRING(L)=SQRT(XNRIN)                               00006200
NORING=NRING(L)                                    00006210
NOTRG=6*NORING*NORING                             00006220
NOTRG1=NOTRG+1                                     00006230
NOTG1=6*(NORING-1)*(NORING-1)                     00006240
NOG01=3*(NORING-1)*NORING                          00006250
NOINCR=6*(NORING+1)                                00006260
LSTCH=NOTRG+NOINCR                                 00006270
LSTCH1=LSTCH+1                                     00006280
LSTW=LSTCH-6                                        00006290
LSTCHG=LSTCH+NOINCR                               00006300
LSTWG=LSTCHG-6                                     00006310
N1(L)=NOTG1                                         00006320
N2(L)=NOTRG                                          00006330
N3(L)=LSTW                                           00006340
N4(L)=LSTCH                                          00006350
N5(L)=LSTCH+6*NORING                               00006360
N6(L)=N5(L)+6                                       00006370
N7(L)=N6(L)+6*NORING                               00006380
N8(L)=N7(L)+6                                       00006390
NCHANL(L)=N4(L)                                    00006400
DO 7 I=1,NOTRG                                     00006410
Z=(I-1)/6.                                          00006420
N=SQRT(Z)+1.0001                                   00006430
NOTRG=6*N*N                                         00006440
NOTR1=6*(N-1)*(N-1)                                00006450
NOTR2=6*(N-2)*(N-2)                                00006460
NOT01=3*N*(N-1)                                     00006470
NOT12=3*(N-1)*(N-2)                                00006480
IQ=(I-NOTR1-1)/(2*N-1)+0.0001                     00006490
IR=I-NOTR1-IQ*(2*N-1)+0.0001                      00006500
IY=1.6-1/N                                          00006510
LC(L,I,4)=I                                         00006520
IF(I-NOTR1-1)901,902,901                           00006530
901 IF(I.EQ.NOTRG)GO TO 903                          00006540
J=IR/2                                              00006550
IF(IR-J*2)905,905,904                              00006560
905 IZ=1.0001/(NOTRG-I)                            00006570
LC(L,I,1)=NOTR2+IQ*(2*N-3)+IR-1                    00006580
LC(L,I,2)=I-1                                       00006590
LC(L,I,3)=I+1                                       00006600
MROD(L,I,1)=1+NOT12+IQ*(N-1)+IR/2                 00006610
MROD(L,I,2)=MROD(L,I,1)+1-6*IZ*(N-1)              00006620
MROD(L,I,3)=NOT01+IQ*N+IR/2+2                     00006630
GO TO 6                                              00006640
904 LC(L,I,1)=I-1                                    00006650
LC(L,I,2)=I+1                                       00006660
MROD(L,I,1)= NOT12+IQ*(N-1)+(IR+1)/2+IY           00006670
MROD(L,I,2)=1+NOT01+IQ*N+(IR+1)/2                 00006680
MROD(L,I,3)=MROD(L,I,2)+1                          00006690
908 IF(N.EQ.NORING)GO TO 906                       00006700
LC(L,I,3)=NOTR1+IQ*(2*N+1)+IR+1                   00006710

```

LEVEL 21

NUMB

DATE = 77237

16/16/36

	GO TO 6	00006720
906	LC(L,I,3)=NOTRG+IQ*N+(IR+1)/2	00006730
	GO TO 6	00006740
	LC(L,I,1)=NOTR1+1	00006750
	LC(L,I,2)=NOTRG-1	00006760
	MROD(L,I,1)=1+IY+NOT12	00006770
	MROD(L,I,2)=2+NOTC1	00006780
	MROD(L,I,3)=1+3*N*(N+1)	00006790
	GO TO 908	00006800
902	LC(L,I,1)=NOTR1+2	00006810
	LC(L,I,2)=NOTRG	00006820
	MROD(L,I,1)=1+IY+NOT12	00006830
	MROD(L,I,2)=2+NOTC1	00006840
	MROD(L,I,3)=MROD(L,I,2)+1	00006850
	GO TO 908	00006860
6	CONTINUE	00006870
7	CONTINUE	00006880
	DO 9 I=NOTRG1,LSTCH	00006890
	MROD(L,I,3)=NOROD+1	00006900
	IF(I.LE.LSTW) GO TO 801	00006910
	IQ=I-LSTCH+5	00006920
	LC(L,I,2)=NOTRG+IQ*NORING+1	00006930
	LC(L,I,3)=LSTCHG-5+IQ	00006940
	LC(L,I,4)=I	00006950
	MROD(L,I,1)=2+NOGC1+IQ*NORING	00006960
	MROD(L,I,2)=NOROD+1	00006970
	IF(IQ.EQ.0) GO TO 802	00006980
	LC(L,I,1)=NOTRG+IQ*NORING	00006990
	GO TO 8	00007000
802	LC(L,I,1)=I-1	00007010
	GO TO 8	00007020
801	IZ=1.0001/(LSTCH-5-I)	00007030
	IQ=(I-NOTRG-1)/NORING+0.0001	00007040
	IR=I-NOTRG-IQ*NORING+0.0001	00007050
	LC(L,I,1)=NOTG1+IQ*(2*NORING-1)+2*IR-1	00007060
	LC(L,I,4)=LSTCH+IQ*NORING+IR	00007070
	MROD(L,I,1)=1+NOGC1+IQ*NORING+IR	00007080
	MROD(L,I,2)=MROD(L,I,1)+1-6*IZ*NORING	00007090
	IF(IR.EQ.1) GO TO 803	00007100
	IF(IR.EQ.NORING) GO TO 804	00007110
	LC(L,I,2)=I-1	00007120
	LC(L,I,3)=I+1	00007130
	GO TO 8	00007140
803	LC(L,I,2)=LSTW+IQ+1	00007150
	LC(L,I,3)=I+1	00007160
	GO TO 8	00007170
804	LC(L,I,2)=I-1	00007180
	LC(L,I,3)=(1-IZ)*(LSTW+IQ+2)+IZ*(I+1)	00007190
	GO TO 8	00007200
8	CONTINUE	00007210
9	CONTINUE	00007220
	DO 13 J=1,NWALLS	00007230
	DO 11 I=LSTCH1,LSTCHG	00007240
	IF(I.LE.LSTWG) GO TO 701	00007250
	IQ=I-LSTWG-1	00007260
	LC(L,I,1)=LSTW+1+IQ	00007270
	LC(L,I,3)=LSTCH+IQ*NORING+1	00007280
	IF(J.EQ.NWALLS) GO TO 12	00007290

LEVEL 21

NO. 18

DATE = 77237

16/16/36

```

      LC(L,I,4)=LSTWG+NORING+IQ+1
      GO TO 14
12  CONTINUE
      LC(L,I,4)=I
14  CONTINUE
      IF(I..EQ.. ) GO TO 702
      LC(L,I,2)=LSTCH+IQ*NORING
      GO TO 1
702 LC(L,I,2)=I-1
      GO TO 10
701 IX=1. - 1/(LSTWG+1-I)
      ID=(1-LSTCH-1)/NORING+.001
      IX=I-LSTCH-IQ*NORING+.001
      LC(L,I,1)=NOTRG+IQ*NORING+IX
      IF(J..EQ..WALLS) GO TO 16
      LC(L,I,4)=LSTCHG+IQ*NORING+IX
      GO TO 18
16  CONTINUE
      LC(L,I,4)=I
18  CONTINUE
      IF(IX..EQ..1) GO TO 703
      IF(I..EQ..NORING) GO TO 704
      LC(L,I,2)=I-1
      LC(L,I,3)=I+1
      GO TO 1
704 LC(L,I,2)=I-1
      LC(L,I,3)=(1-IX)*(LSTWG+IQ+2)+IX*(I+1)
      GO TO 1
703 LC(L,I,2)=LSTWG+IQ+1
      LC(L,I,3)=I+1
      GO TO 1
1  CONTINUE
11  CONTINUE
      NOTRG=NOTRG+NORING
      LSTCH=LSTCH+NORING
      LSTCH1=LSTCH1+NORING
      LSTW=LSTW+NORING
      LSTWG=LSTWG+NORING
      LSTCHG=LSTCHG+NORING
13  CONTINUE
      RETURN
      END

```

```

00007310
00007320
00007330
00007340
00007350
00007360
00007370
00007380
00007390
00007400
00007410
00007420
00007430
00007440
00007450
00007460
00007470
00007480
00007490
00007500
00007510
00007520
00007530
00007540
00007550
00007560
00007570
00007580
00007590
00007600
00007610
00007620
00007630
00007640
00007650
00007660
00007670
00007680
00007690
00007700
00007710

```


LEVEL 21

GEOM

DATE = 77237

16/16/36

```

SUBROUTINE GEOM(L)                                00007720
COMMON/GEOM /DR(1),DW(1),LEAD(1),P(1),DFLAT(1),PCD(1),GAP(1),A1(1),00000050
1A2(1),A3(1),A4(1),A5(3),A6(1),A7(1),ATOTAL(1),ATOTLG(1),DE1(1), 00000060
2DE2(1),DE3(1),DE6(1),DE7(1),HP1(1),HP2(1),HP3(1),HPTOTL(1), 00000070
3ETA1(1),ETA2(1),ETA3(1),ETA4(1),ETA6(1),ETA8(1),X1(1),X2(1),X3(3),00000080
4S1(1),S2(1),LENGTH(1),V1R(1),V2R(1),V3R(1),V6R(1),V7R(1) 00000090
COMMON/NUM /NROD(1),NRING(1),NCHANL(1),N1(1),N2(1),N3(1),N4(1),
1N5(1),N6(1),N7(1),N8(1),LC(1,550,4),MROD(1,440,3)
N11=N2(L)                                00007800
N22=N3(L)-N2(L)                          00007810
N33=N4(L)-N3(L)                          00007820
GAP(L)=(DFLAT(L)- 1.732* NRING(L)*P(L)-DR(L))/2 00007830
DFACE=DFLAT(L)/1.732                    00007840
PI=3.141592                             00007850
X1(L)=(DR(L)+2*GAP(L))/1.732            00007860
X2(L)=X1(L)+2*S1(L)/1.732              00007870
X3(L)=X2(L)+S2(L)/1.732                00007880
ETA1(L)=P(L)/1.732                     00007890
ETA2(L)=(ETA1(L)+GAP(L)+DR(L)/2)/2.    00007900
ETA3(L)=P(L)                           00007910
ETA4(L)=ETA3(L)/2.+X1(L)/4.            00007920
ETA6(L)=ETA3(L)/2.+(X1(L)+X2(L))/4.    00007930
ETA8(L)=ETA3(L)/2.+(X2(L)+X3(L))/4.    00007940
A1(L)=1.732*P(L)**2/4.-(DR(L)**2+DW(L)**2)*PI/8. 00007950
A2(L)=P(L)*(DR(L)/2+GAP(L))-(DR(L)**2+DW(L)**2)*PI/8. 00007960
A3(L)=(DR(L)/4+GAP(L)/2)*(DFACE-P(L)* NRING(L))-(DR(L)**2+DW(L)**200007970
1)*PI/24.                                00007980
ATOTAL(L)=N11*A1(L)+N22*A2(L)+N33*A3(L) 00008030
WP1=(DR(L)+DW(L))*PI/2.                00008050
WP2=P(L)+WP1                           00008060
WP3=(DR(L)+DW(L))*PI/6+DFACE-P(L)* NRING(L) 00008070
WP6=ETA3(L)                            00008080
WP7=X2(L)                              00008090
DE1(L)=4.0*A1(L)/WP1                   00008100
DE2(L)=4.0*A2(L)/WP2                   00008110
DE3(L)=4.0*A3(L)/WP3                   00008120
HP1(L)=DR(L)*PI/2                      00008150
HP2(L)=DR(L)*PI/2                      00008160
HP3(L)=DR(L)*PI/6                      00008170
HPTOTL(L)=N11*HP1(L)+N22*HP2(L)+N33*HP3(L) 00008180
V1R(L)=ATOTAL(L)/(N11*A1(L)+N22*A2(L))*((DE2(L)/DE1(L))**0.714)+ 00008190
1N33*A3(L))*((DE3(L)/DE1(L))**0.714)) 00008200
V2R(L)=ATOTAL(L)/(N11*A1(L))*((DE1(L)/DE2(L))**0.714)+N22*A2(L)+ 00008210
1N33*A3(L))*((DE3(L)/DE2(L))**0.714)) 00008220
V3R(L)=ATOTAL(L)/(N11*A1(L))*((DE1(L)/DE3(L))**0.714)+N22*A2(L)* 00008230
1((DE2(L)/DE3(L))**0.714)+N33*A3(L)) 00008240
RETURN
END                                00008280

```