

AD-A080 175

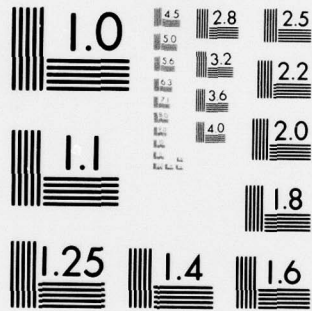
AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOO--ETC F/8 9/2
AUTOMATED HALL EFFECT EXPERIMENT DATA ACQUISITION SYSTEM (AHEAD--ETC(U))
DEC 79 E A VERCHOT
AFIT/0EO/EE/79D-5

NL

1 OF 3

AD
A080175





MICROCOPY RESOLUTION TEST CHART
 NATIONAL BUREAU OF STANDARDS-1963-A



AFIT/GEO/EE/79D-5

AUTOMATED HALL EFFECT EXPERIMENT
DATA ACQUISITION SYSTEM
(AHEEDAS)

THESIS

AFIT/GEO/EE/79D-5 ✓ Edgar A. Verchot, Jr.
Captain USAF

DDC
RECEIVED
FEB 5 1980
RECEIVED
A

DISTRIBUTION STATEMENT A
Approved for public release,
Distribution Unlimited

14

D

AFIT/GEO/EE/79-5

6

**AUTOMATED HALL EFFECT EXPERIMENT
DATA ACQUISITION SYSTEM
(AHEEDAS).**

9

Master's THESIS

Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology
in Partial Fulfillment of the
Requirements for the Degree of
Master of Science in Electrical Engineering

Accession For	
NTIS GRA&I	<input checked="" type="checkbox"/>
DDC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By _____	
Distribution/ _____	
Availability Codes	
Dist	Avail and/or special
A	

10 Edgar ^{Alphonse} Verchot, Jr., B.S., B.S.E.E.
Captain USAF

Graduate Electro Optics

12 208

11 December 1979

Approved for public release; distribution unlimited.

012225

LB

Preface

Dr. Patrick M. Heminger of the Air Force Materials Laboratory, Wright-Patterson Air Force Base, Ohio, developed and built a Hall effect experimental apparatus to gather data on the impurity levels and electrical properties of silicon samples. While successful, the experiment was long and awkward to perform. He decided to automate it and towards this end purchased an LSI-11 microcomputer system and automated instrumentation compatible with it. Starting with the system, I designed and implemented the software system necessary to make this system function. This report details the process of designing and implementing this system.

I want to thank Dr. Thomas C. Hartrum, my thesis advisor, for his help and guidance. He provided the basic direction from which I was able to proceed to begin the design of AHEEDAS, and as the design progressed, was always available to discuss problem areas. In addition, he went over multiple drafts of this report in detail to help shape it into a useable product. The other members of the committee, Dr. Gary B. Lamont and Capt. John M. Borky also provided valuable guidance during the course of this project.

I also want to thank Dr. Heminger, the thesis sponsor who provided the thesis topic and Mr. Steve Smith of the University of Dayton Research Institute (on site contractor). Both of these men provided valuable guidance on the desired results of the experiment and suggestions as to how the

calculations necessary to implement the various functions might best be done.

Special mention must go to Mr. Frank E. Beital and Mr. Dane Hanby of the University of Dayton Research Institute (on site contractors). Mr. Beital wrote several of the device drivers and provided continual and invaluable guidance on how the LSI-11 could be made to efficiently perform the desired functions. Whenever I reached a "dead end," he was usually able to show me the error(s) that I had made and put me back "on track." Mr. Hanby did all of the hardware interfacing for the automated equipment. He designed and built several of the interfaces from "scratch" and without his expertise, the AHEEDAS would have had no system to control.

I would also like to thank Capt. William Walker for allowing me the use of the resources of the Computer Activities Division, Air Force Materials Laboratory, without which successful completion of this thesis would not have been possible and to Ms. Carla Smith who typed the final copy of this report.

Finally and most importantly, I would like to thank my wife, Deborah, and son, Edgar. Without their love, understanding and support through the long nights and weekends, this thesis would not ever have been finished.

Contents

Preface	ii
List of Figures	vi
List of Tables	viii
Abstract	ix
I. Introduction	1
Background	1
Statement of Problem	2
Constraints/Assumptions	2
Scope of Thesis	3
II. Experimental Environment	4
Theory	4
System Hardware	11
Original Environment	14
Summary	15
III. Requirements Definition	16
Method for Analysis and Design	16
Activity Model	17
Node A-0, Conduct Experiment and Process Data	18
Node A0, Conduct Experiment and Process Data	18
Node A1, Initialize Parameters	21
Node A2, Determine Parameters	23
Node A3, Set Parameters	24
Node A4, Acquire Data	28
Node A5, Reduce Data	29
Summary	31
IV. System Design and Implementation	33
Heirarchical Design	33
Implementation	39
Method of Implementation	39
Summary	42
V. System Testing and Validation	43
Incremental Testing	43
Validation	44
Hardware Testing	44
Final Validation	45
Summary	46

Contents

VI. Recommendations and Conclusions	47
Recommendations	47
Conclusions	50
Bibliography	51
Appendix A: Structured Analysis and Design Tool (SADT)	53
Appendix B: Operator's Manual	58
Appendix C: AHEEDAS Algorithm	63
Vita	194

List of Figures

<u>Figure</u>		<u>Page</u>
1	van der Pauw Configuration Sample Geometry	5
2	The Three Possible Cases for V_e	7
3	Hall Bar Configuration Sample Geometry	9
4	Block Diagram of the AHEEDAS	12
5	Node A-0 Conduct Experiment and Process Data	19
6	Node A0 Conduct Experiment and Process Data	20
7	Node A1 Initialize Parameters	22
8	Node A2 Determine Settings	24
9	Node A3 Set Parameters	26
10	Node A4 Acquire Data	27
11	Node A5 Reduce Data	30
12	Heirarchical Chart of AHEEDAS	34
13	Heirarchical Chart of A3 and A4	35
14	Overlaid Program Structure	41
15	Box/Arrow Convention	54
16	OR Branch and Join Structure	56
17	ICOM Numbering Convention	56
18	Flow Chart for Module A0	64
19	Flow Chart for Module A1	69
20	Flow Chart for Module A11	73
21	Flow Chart for Module A12	75
22	Flow Chart for Module A13	81
23	Flow Chart for Module A14	85

List of Figures

<u>Figure</u>		<u>Page</u>
24	Flow Chart for Module A15	92
25	Flow Chart for Module A2	97
26	Flow Chart for Module A21	101
27	Flow Chart for Module A22	104
28	Flow Chart for Module A23	107
29	Flow Chart for Module A24	113
30	Flow Chart for Module A3	116
31	Flow Chart for Module A31	119
32	Flow Chart for Module A32	122
33	Flow Chart for Module A33	125
34	Flow Chart for Module A34	128
35	Flow Chart for Module A35	131
36	Flow Chart for Module A36	133
37	Flow Chart for Module A4	135
38	Flow Chart for Module A41	138
39	Flow Chart for Module A42	141
40	Flow Chart for Module A43	144
41	Flow Chart for Module A45	147
42	Flow Chart for Module A5	150
43	Flow Chart for Module A51	154
44	Flow Chart for Module A52	157
45	Flow Chart for Module A53	162
46	Flow Chart for Module A55	166
47	Flow Chart for Module A56	169
48	Flow Chart for Module A57	172

List of Tables

<u>Table</u>		<u>Page</u>
I	Node Index	17

Abstract

The Air Force Materials Laboratory conducts experiments using the Hall effect to characterize the electrical properties and impurity levels of silicon samples. Both the van der Pauw and the classical Hall bar methods are used.

The purpose of this study was the development of an Automated Hall Effect Experiment Data Acquisition System (AHEEDAS) to control the conduct of the experiment and to reduce all of the necessary data. The design system is capable of controlling all aspects of the experiment except the temperature. The AHEEDAS produces as output, sample resistivity, Hall mobility, carrier concentration and the Hall coefficient as a function of temperature. These are stored in data files on floppy disk storage along with all of the raw data from the experiment. The output is also printed on the computer terminal as the experiment is done.

An LSI-11 microcomputer with 28K words of memory is used to control the experiment. Software was developed to allow this system to handle the acquisition and processing of data. The AHEEDAS was successfully implemented and tested. All functions other than the temperature control are fully operational.

I. Introduction

The Air Force Materials Laboratory (AFML), Wright-Patterson Air Force Base, Ohio, uses a Hall effect experiment to analyze the dopant and impurity content of silicon samples. They wanted to develop an Automated Hall Effect Experimental Data Acquisition System (AHEEDAS) in order to increase both the speed and accuracy of their experiment. This report describes the design, development and implementation of this system.

Background

The concentrations and activation energies of electrically active impurities are important quantities for the complete characterization of semiconductors. Measurements of sample resistivity and Hall voltage versus temperature enable the experimenter to determine the electrical transport properties of a semiconductor sample from which the net impurity concentrations and carrier mobility can be determined (Ref 4).

The experiment was originally done manually; it took from six to eight hours to make all the necessary measurements on one sample. Using the van der Pauw method, twenty voltage versus current measurements are made for each temperature point. Thirty to sixty temperature points are needed for a complete curve. Therefore, 600 to 1200 data points are needed for each sample. Clearly automation was needed to assist in this experiment. In addition to freeing the experimenter to do other tasks, the experiment could be

greatly enhanced by the increase in accuracy as well as speed. The increased speed allows many more data points to be taken per sample.

Statement of Problem

The purpose of this study was to design and implement the hardware and software to control the Hall Effect experiment and to collect and reduce the data from it with an LSI-11 data acquisition system. The system was to be designed to control all aspects of the experiment and to take all required data. These data were to be written onto floppy disc storage for later use. A real-time monitoring of the experimental data was required. Future expansion needs were also to be defined.

Constraints/Assumptions

Prior to requesting assistance with developing the system, AFML purchased an LSI-11 microcomputer system (described in Chapter II) and automated instrumentation. This constrained the freedom of hardware selected. The only hardware which was not specified was that for the implementation of the temperature controller and that future expansion.

The user requested that for the first stage of development the temperature control be left manual and that the remainder of the system be implemented. This was to ensure that a minimum system would be made operational by the end of this investigation.

Scope of Thesis

The structured analysis and design technique (SADT) (Ref 11) was used to develop the activity model of AHEEDAS (Ref Chapter III). The LSI-11 microprocessor was used as the processor. It and its associated hardware system are discussed along with the experimental environment in Chapter II. The software modules were designed and coded (Chapter IV). The hardware and the software were integrated and functionally tested (Chapter V). All portions of the first stage system are functional at this time. Recommendations for further study have been included in Chapter VI.

II. Experimental Environment

Before the requirements of the system are defined, the original environment, including the design constraints, will be described and discussed. Once all of the constraints are identified, the next step of the analysis can be performed. In this chapter, the background theory of the Hall Effect experiment will be discussed, followed by a discussion of the originally purchased hardware system.

Theory

Two types of sample configurations are used in the Hall Effect experiment: the van der Pauw and the Hall bar (Ref 22). The geometry of the two configurations and the calculations necessary to use them are presented below.

The van der Pauw technique uses a sample having only four electrical contacts (see Figure 1). In each of the six configurations shown, the potential difference between V_1 and V_2 is recorded. Additionally, these measurements are repeated with the battery polarity reversed. In sample configurations (e) and (f), the voltage difference between V_1 and V_2 with a magnetic field applied perpendicularly to the sample is also measured. Both polarities of the field must be used for these measurements (Ref 4). From these data, four parameters are calculated: resistivity, Hall mobility, carrier concentration, and the Hall coefficient.

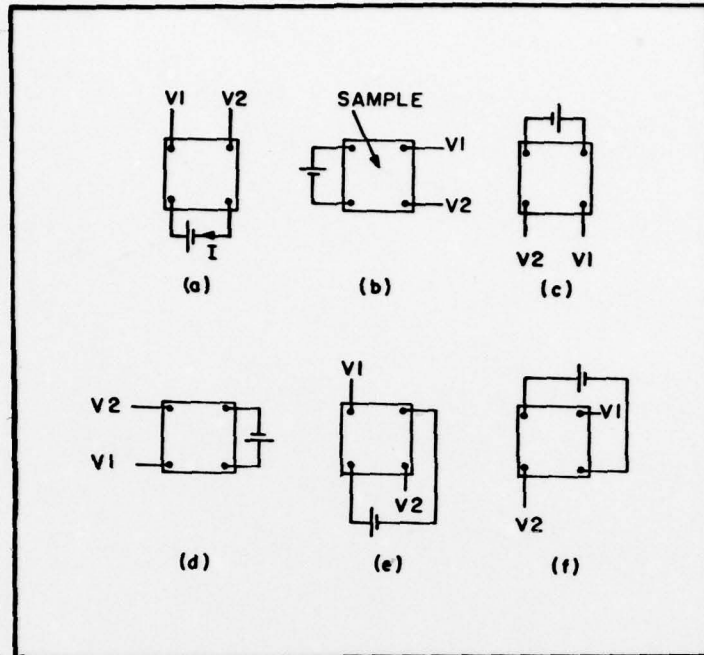


Figure 1. van der Pauw Configuration Sample Geometry.

The derivations of the necessary formulae for both the van der Pauw and the Hall bar configurations can be found in the van der Pauw report on the Hall Effect experiment (Ref 22).

Resistivity, ρ , in ohms-centimeters, is given by,

$$\rho = \frac{\pi t(R_a + R_b)}{2 \ln 2} f\left(\frac{R_a}{R_b}\right) \quad (1)$$

where t is the thickness of the sample in centimeters, and R_a is found, using sample configuration (a) from

$$R_a = \frac{V_a}{I_a} \quad (2)$$

where V_a is the voltage difference between V1 and V2, and I_a is the current between the battery contacts. Likewise from sample configuration (b)

$$R_b = \frac{V_b}{I_b} \quad (3)$$

Sample configuration (c) is equivalent to (a), and (d) is equivalent to (b). Both polarities of these configurations are measured and the results averaged. The parameter $f\left(\frac{R_a}{R_b}\right)$ is the "van der Pauw f," which is approximated by Reference 22.

$$f = 1 - \left(\frac{R_a - R_b}{R_a + R_b}\right)^2 \left(\frac{\ln 2}{2}\right) - \left(\frac{R_a - R_b}{R_a + R_b}\right)^4 \left[\left(\frac{\ln 2}{4}\right)^2 - \left(\frac{\ln 2}{12}\right)^3\right] \quad (4)$$

Hall mobility, μ_h , in square centimeters/volt-second is given by

$$\mu_h = \frac{\Delta R_e t}{\rho B} \cdot 10^8 \quad (5)$$

where t is sample thickness, ρ is resistivity, B is the applied magnetic field in gauss, and ΔR_e is the change in resistance computed from position (e) of Figure 1 when the magnetic field is applied perpendicularly to the sample.

The value of ΔR_e can be found by analyzing the three possible cases (See Figure 2.).

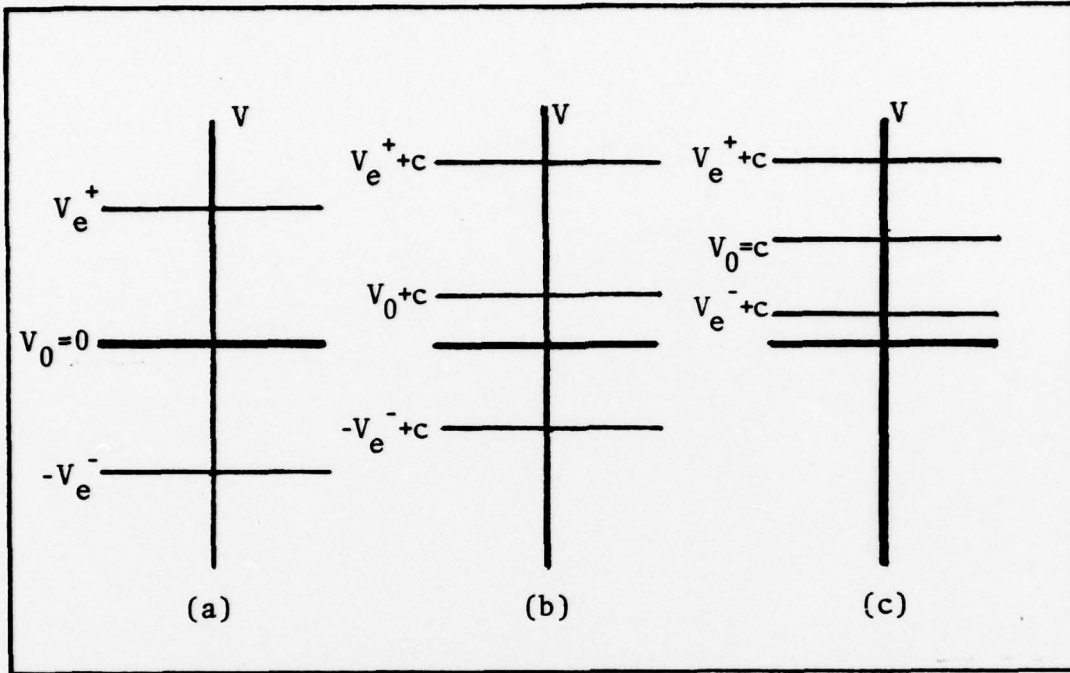


Figure 2. The Three Possible Cases for V_e .

V_e^+ is the voltage measured in sample position (e) with an applied magnetic field; V_e^- is the same measurement with the magnetic field reversed, and V_0 is an offset voltage generated by non-uniform sample composition or geometry or both. Reversal of the current polarity and proper averaging of the measured voltages give a correct value for the true Hall voltage, V_e . Now,

$$R_e^+ = \frac{V_e^+ + c}{I_e^+} \quad (6)$$

and

$$R_e^- = \frac{V_e^- + c}{I_e^-} \quad (7)$$

therefore,

$$\Delta R_e = \frac{|R_e^+ - R_e^-|}{2} \quad (8)$$

or

$$\Delta R_e = \frac{\left| \frac{V_e^+}{I_e^+} - \frac{V_e^-}{I_e^-} \right|}{2} \quad (9)$$

where V_e^+ and V_e^- are as defined above and I_e^+ and I_e^- are the corresponding currents. The above measurements are repeated with the current reversed and the results are averaged. Position (f) yields the same results and these are averaged with the results from (e).

Carrier concentration, p , in inverse centimeters cubed, can be determined from

$$p = \frac{1}{\mu_c e \rho} \cong \frac{1}{\mu_h e \rho} \quad (10)$$

where μ_c is the conductivity mobility, μ_h is the Hall mobility, e is the electronic charge in coulombs and ρ is the resistivity in ohms-centimeters. Conductivity mobility is approximated by the Hall mobility because precise information about $\frac{\mu_c}{\mu_h}$ is not available. It can be assumed that

$$\frac{\mu_c}{\mu_h} \cong 1 \quad (11)$$

The Hall coefficient, R_H , in centimeters cubed coulomb,

is given by

$$R_H = \frac{1}{pe} = \mu_c \rho = \mu_h \rho \quad (12)$$

where ρ is the resistivity in ohm-centimeters, μ_c is the conductivity mobility in squared centimeters/volt-second, p is the carrier concentration in inverse centimeters cubed, e is the electronic charge in coulombs, μ_h is the Hall mobility in squared centimeters/volt-second.

The Hall bar geometry is shown in Figure 3.

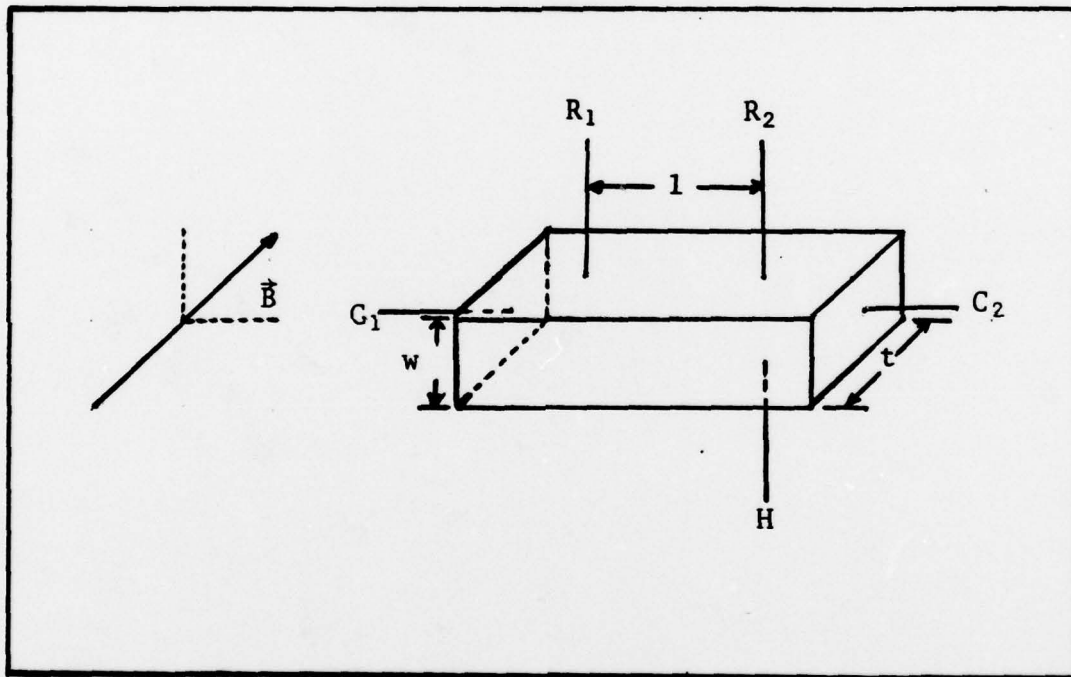


Figure 3. Hall Bar Configuration Sample Geometry.

As Figure 3 shows, the Hall bar sample has five contacts rather than the four of the van der Pauw. The equations used for this configuration are similar to those used for the van der Pauw sample. However, with this sample all three dimensions must be known instead of just the

thickness of the sample. The method is discussed completely in the Reference 22. The formulae needed to reduce the sample data for the Hall bar are given below.

Resistivity, ρ , in ohm centimeters, is given by

$$\rho = \frac{wtV}{lI} \quad (13)$$

where w is the sample width in centimeters, t is the thickness in centimeters, l is the distance in centimeters between contacts R1 and R2 in Figure 3, V is the voltage drop between the contacts and I is the current between contacts C1 and C2, in amperes.

Hall mobility, μ , in centimeters squared/volt-second, is given by

$$\mu_h = \frac{1}{wB} \left(\frac{I}{I'} \right) \left(\frac{V_H}{V} \right) (10^8) \quad (14)$$

where B is the magnetic field, in gauss, V_H is the voltage measured between R2 and H with the magnetic field applied, I' is the corresponding current.

Carrier concentration, p , in inverse centimeters cubed is given by

$$p = \frac{B}{et} \left(\frac{I'}{V_H} \right) \quad (15)$$

where B , I' , V_H , and t are defined above, and e is the electronic charge in coulombs.

The Hall coefficient, R_H , is as defined in equation (12). These are the formulae that are used in the experiment to produce the output data.

System Hardware

Figure 4 shows the block diagram for the hardware system which had been purchased. The components are as follows.

--- Digital Equipment Corporation (DEC) LSI-11 micro-computer with 4K words of memory.

--- DEC RXV 11 dual floppy disk option with dual RX01 floppy disk drive.

--- 24K words of add-on memory.

--- Anderson Jacobson A242A acoustic coupler and DLV-11 serial interface card.

--- Hewlett-Packard Model 7221A, Remote terminal Four Color X-Y plotter (not yet received) with DLV-11 serial interface card.

--- Computer Devices Terminal Model Miniterm 1202, with DLV-11 serial interface card.

--- Hewlett-Packard Digital Voltage Source, Model 6130C, with two DRV-11 parallel interface cards.

--- Walker Magnometrics Gaussmeter, Model MG-3D with DRV-11 parallel interface card.

--- Magnet controller (designed and built in-house by University of Dayton Research Institute contractor) with DRV-11 parallel interface card.

--- Test Controller (designed and built by University of Dayton Research Institute contractor) with DRV-11 parallel interface card.

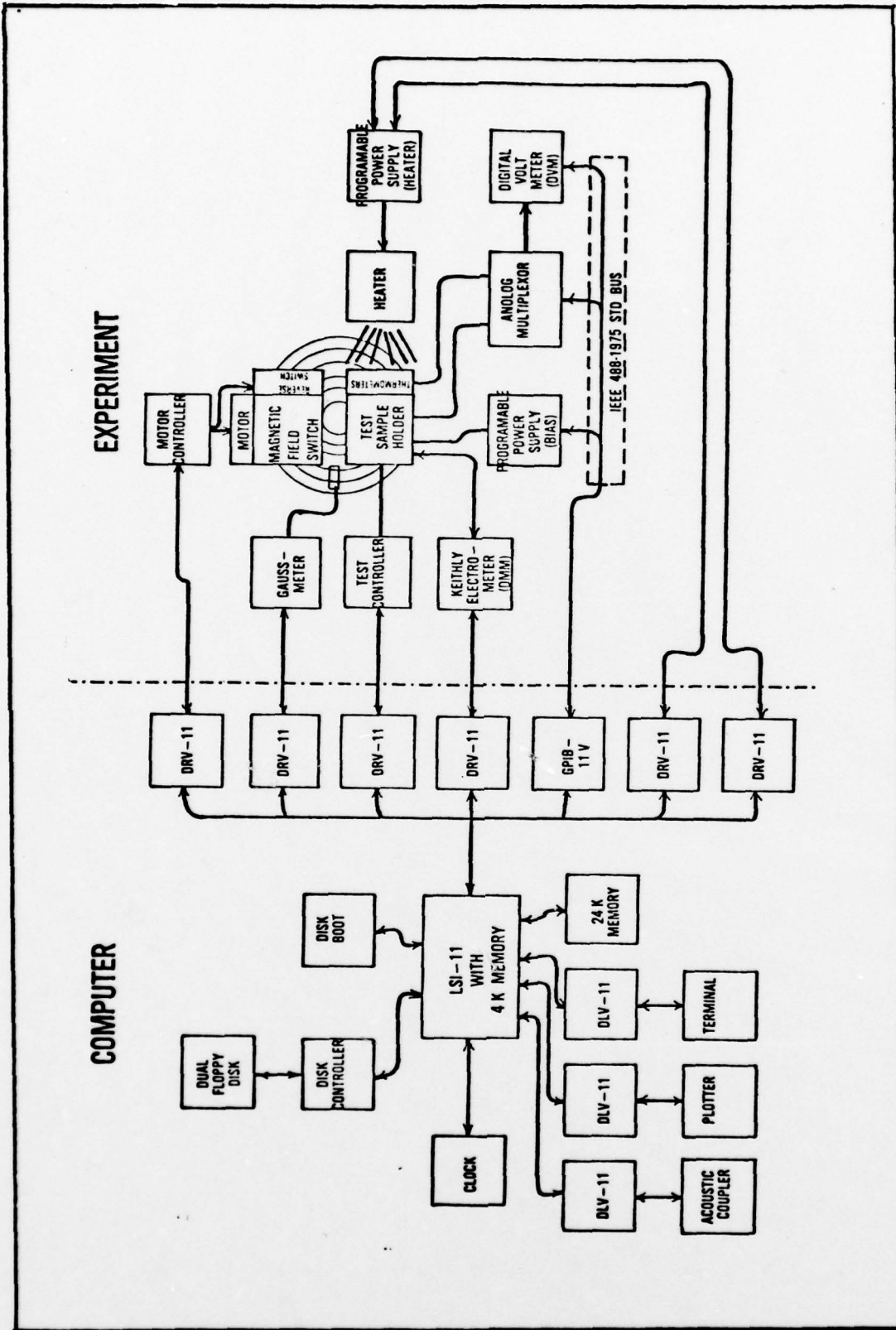


Figure 4. Block Diagram of the AHEEDAS

--- Hewlett-Packard, IEEE 488-1975 compatible, HP-IB isolated Digital/Analog/Power Supply programmer, Model 59501A.

--- Keithley Digital Electrometer, Model 616 with Model 6162 isolated output control with DRV-11 parallel interface card.

--- Hewlett-Packard, IEEE 488-1975 compatible, Digital Voltmeter, Model 3455A.

--- Hewlett-Packard, IEEE 488-1975 compatible, Multiplex Scanner, Model 3485A.

--- Hewlett-Packard, IEEE 488-1975 compatible, GPIB-11 instrumentation bus controller.

Each piece of equipment was interfaced to the LSI-11 with a small MACRO assembly language driver subprogram supplied by AFML/DOC (reference Appendix C). All but two pieces of the equipment were adequate for the implementation of the system. It was not determined whether the Hewlett-Packard Model 6130C, Digital Voltage Source would be adequate for the temperature control. See the Recommendations in Chapter VI for more discussion of this problem. The Walker gaussmeter is a problem. It intermittently locks up in use. The digital interface appears to have been added by the manufacturer as an afterthought to the design and is very awkward to use. At best, its use will be limited by the lack of any auto ranging ability. The system was designed to work automatically with or without the gaussmeter in the circuit. The GPIB-11 controller and software were a continuous source of

problems. However, these problems were solved after much consultation with National Instruments. They ultimately replaced both the original interface card and the software. Future expansion of the system may uncover more problems with this interface.

Original Environment

Originally, the experiment was executed manually. The temperature was controlled with an analog, closed-loop controller. All other instrumentation was manually operated. The operator had to set the voltage across the sample, switch between sample positions, and set the field as well as set the temperature controller. Data were read by the operator and hand logged. An abbreviated set of data points was used for van der Pauw samples in order to keep the execution time of the experiment down to about six hours. On the first and each succeeding alternate temperature, sample configurations, plus and minus a, plus and minus b, and plus and minus e were done (reference Figure 1). On the second and each succeeding alternate temperature, sample configurations plus and minus c, plus and minus d, and plus and minus f were executed. In this manner, 20 to 30 temperature points could be taken in the range from about 20 degrees Kelvin to 300 degrees Kelvin. These data were then entered manually into a computer so that they could be reduced and analyzed. Hall bar samples were run in a similar manner except that it was not necessary to abbreviate the set of data points used.

Summary

In this chapter, the pre-existing system constraints were briefly listed and discussed. The original manual method of executing the experiment was also discussed to give some background on what will be required of AHEEDAS. In the next chapter, the requirement for AHEEDAS will be defined using SADT.

III. Requirements Definition

It is difficult to translate the conceptual idea for any system larger than trivial size, into the solid basis for system design. The first step of the design was to break the complete system into functional subunits, or modules. The modules are further divided until their function is simple enough to be easily defined. This process is called the requirements definition phase. A carefully done requirements definition ensures that the system requirements are fully understood by both the system designer and the system user. Ambiguities in the design and implementation are eliminated.

Method for Analysis and Design

The Structured Analysis and Design Technique (SADT) is a design methodology for performing the functional analysis and design of complex systems (Ref 11). A functional model (usually referred to as an activity model) is created first to aid in understanding what activities the system is expected to perform. This activity model is then validated with the system user to ensure that all requirements have been met. Any necessary changes are easily and quickly made at this stage. Following validation, the functional model is broken down further to create the actual design model which will show how the system actually performs its functions. The SADT methodology is defined in Appendix A.

The SADT was chosen as the analysis and design tool

because it is a graphical method of analyzing the functional requirements of the system, and it is readily understood by people with a variety of backgrounds. This facilitates design review. The activity model -- which shows all of the functions, or activities, which the system must perform -- is independent of how a function is implemented. Any changes in the model can be made at this stage without the major changes to the system that would be required later. The completed activity model is easily reviewed by all concerned system users and any errors corrected.

Activity Model

Before drawing the activity model, the system is divided into its component functions. These are assigned node numbers and organized together to produce the initial breakdown of the system. Table I shows the node heirarchy. Explanations of each node follow.

Table I. Node Index.

A-0 Conduct Experiment and Process Data

- A1 Initialize Parameters
- A2 Determine Settings
- A3 Set Parameters
- A4 Acquire Data
- A5 Reduce Data

Node A-0, Conduct Experiment and Process Data (Figure 5)

This node is the top level model of the complete AHEEDAS. The system requirements are shown. User requirements for the Experimental Data Parameters are input (I1) from the terminal. When equipment is inoperative, the required data is manually read and entered from the terminal by an operator to simulate data readings from the automated equipment (I2). Actual data are input from the automated instrumentation (I3). Terminal entries (C1, C2 and C3) control the execution of the system. The dialogue is sent to the terminal (O1); the experimental data parameters are then set by the hardware (O2); the data are recorded and output to the terminal (O3, O4, and O7), the disk (O6), and the plotter (O5). If manual input is required, a prompting message is displayed on the terminal (O1).

The AHEEDAS will be able to complete a sample run unattended once all initialization is complete. The run time should be limited only by the experimental system's capacity for coolants (liquid helium). The operator need only assure that all systems are serviced prior to the start of execution.

Node A0, Conduct Experiment and Process Data (Figure 6)

Node A0 shows the decomposition of the node A-0 into five subordinate nodes.

A1 --- Initialize Parameters

A2 --- Determine Settings

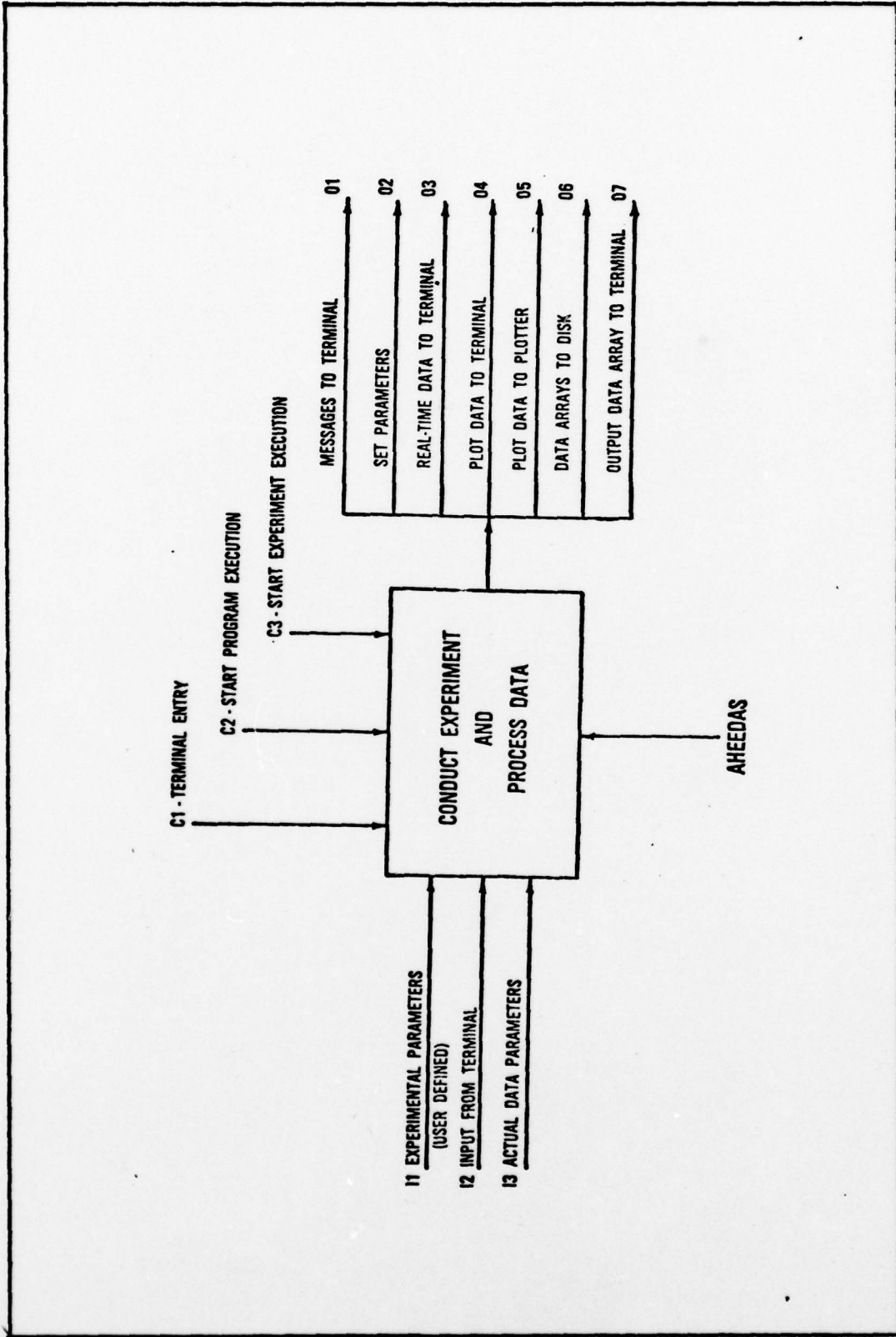


Figure 5. Node A-0 Conduce Experiment and Process Data

A3 --- Set Parameters

A4 --- Acquire Data

A5 --- Reduce Data

When the user starts execution, the activity Initialize Parameters interactively interrogates him to determine the required experimental parameters. Execution will then halt, allowing the user to make certain that the experimental equipment is ready to run, prior to starting the experiment.

The activity Determine Settings calculates the data parameters settings for each data point and passes them to the activity Set Parameters. Set Parameters makes sure that all parameters are properly set and generates the Start Data Acquisition signal. The activity Acquire Data is activated by this signal and reads all of the necessary data variables. Control is then passed back to Determine Settings which repeats the cycle. All of the data points at one temperature constitute a block of data. When a block of data is complete, the activity Reduce Data is called. Reduce Data calculates all of the necessary results and outputs these results and the data. Control is then passed back to Determine Settings to begin the next block of data.

Node A1, Initialize Parameters (Figure 7)

Node A1 is subdivided into five modules:

A11 --- Select Function

A12 --- Create Desired Data Points Files

A13 --- Create Thermometer Calibration Files

A14 --- Operator Initialization Dialogue

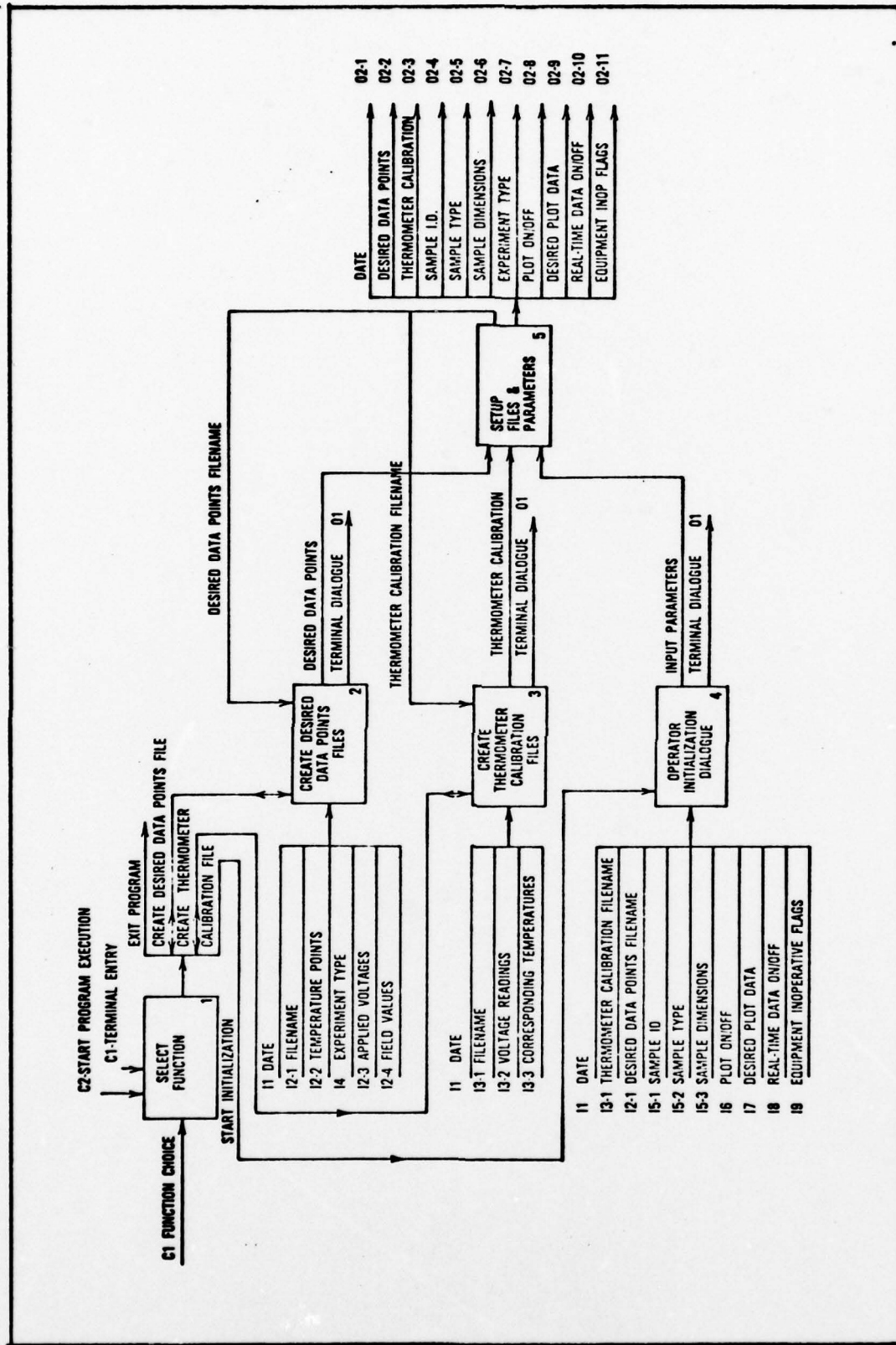


Figure 7. Node A1 Initialize Parameters

A15 --- Setup Files and Parameters

When Start Program Execution is received, the activity Select Function, asks the operator if he wants to

- create a desired data points file
- create a thermometer calibration file
- run an experiment
- exit the program

If "create a desired data points file" or "create a thermometer calibration file" is selected then the appropriate activity (A12 Create Desired Data Points Files or A13 Create Thermometer Calibration Files) done and control is returned to Select Function for a new selection. If "Run an Experiment" is selected, then the activity Operator Initialization Dialogue is executed. It interrogates the operator to determine all necessary experimental parameters. Control then passes to the activity Setup Files and Parameters, which initializes all needed experimental parameters and arrays and opens the necessary output files. The activity then pauses and informs the operator that initialization is complete and requests that he input a Start Execution command.

Node A2, Determine Parameters (Figure 8)

Node A2 is subdivided into five activities:

- A21 --- Determine Temperature Settings
- A22 --- Determine Field Settings
- A23 --- Determine Sample Configuration Settings
- A24 --- Determine Applied Voltage Settings

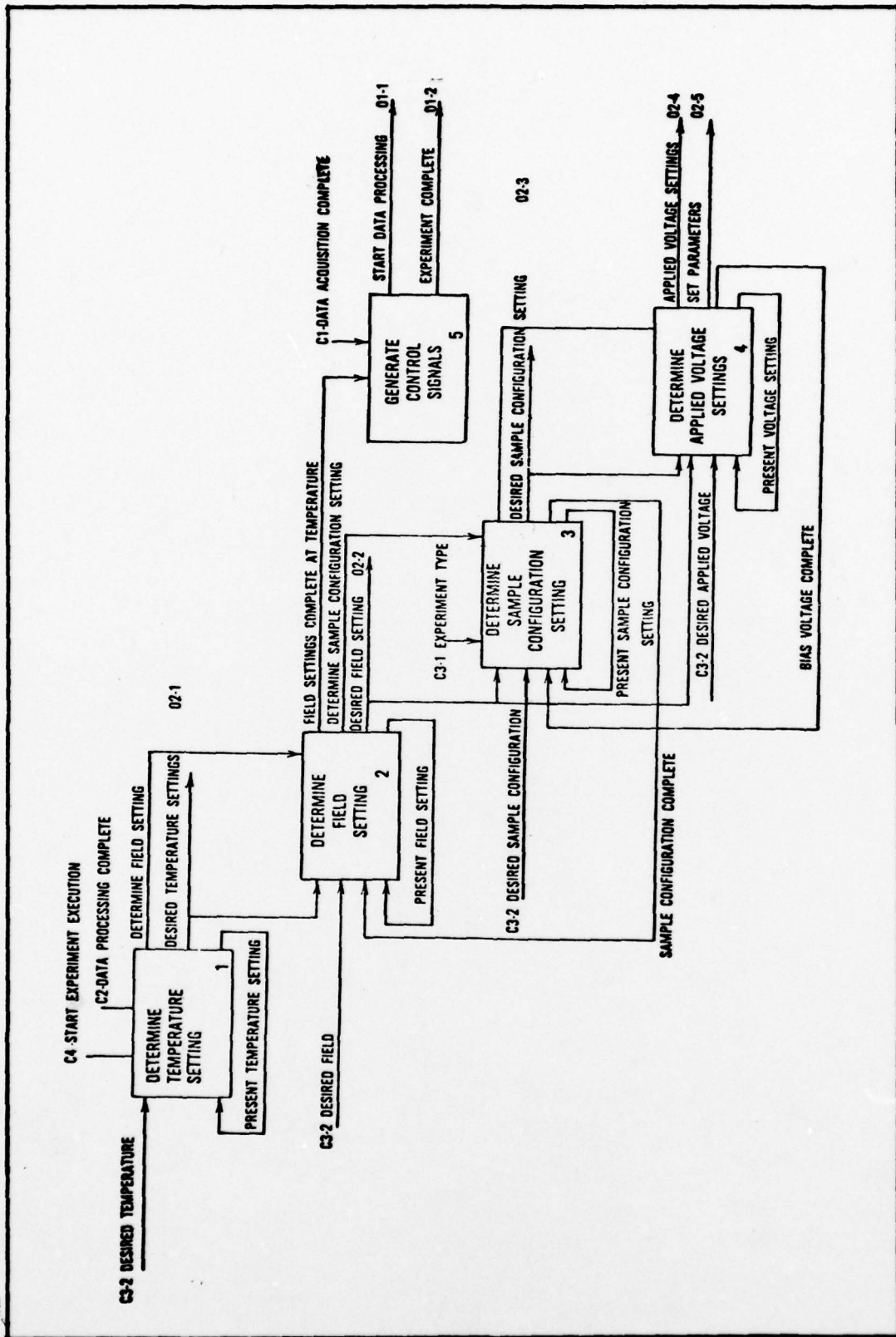


Figure 8. Node A2 Determine Settings

A25 --- Generate Control Signals

When experiment execution is begun, the activity Determine Temperature Settings receives control, performs its function and passes control to the activities Determine Field Settings, Determine Sample Configuration Settings, and Determine Applied Voltage Settings, in turn. This order is important. Each block of data is defined by a single temperature. Therefore, temperature must be determined first. The field magnitude is determined next, then sample configuration. The voltage setting changes at every data point. It, therefore, is determined last. This order allows the parameters that reach steady state the most rapidly to be reset the most often. Temperature and field, which take the longest to settle to their steady state value, are allowed to remain constant for as long as possible. When Determine Applied Voltage Settings completes its calculations, it then generates the Set Parameters signal. When all of the points at one temperature are complete, control passes to the activity Generate Control Signals, which sends out the Start Data Processing or the Experiment Complete signals, as appropriate.

Node A3, Set Parameters (Figure 9)

Node A3 is subdivided into six activities:

A31 --- Check/Set Field

A32 --- Check/Set Applied Voltage

A33 --- Check/Set Sample Configuration

A34 --- Check/Set Temperature

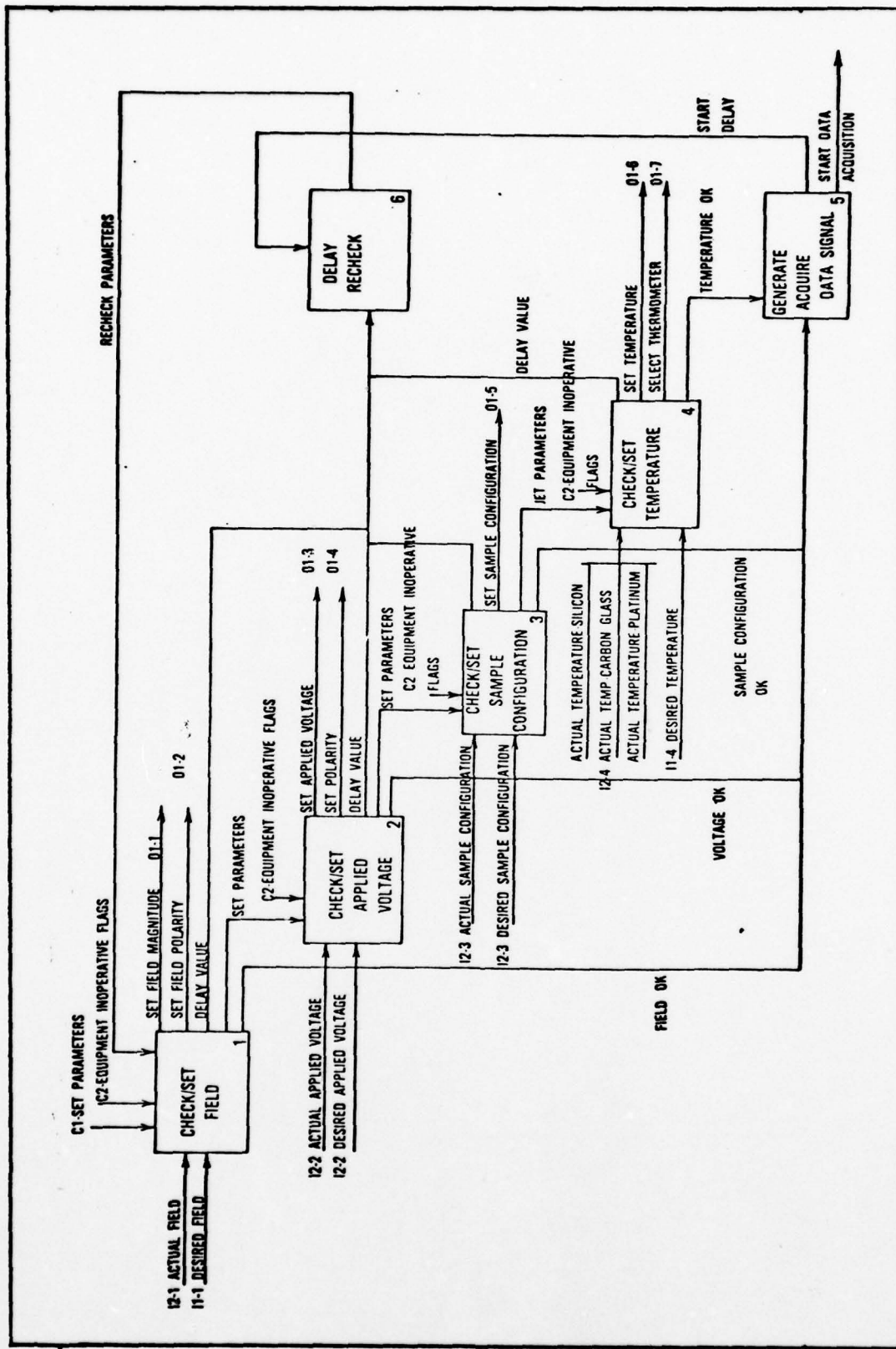


Figure 9. Node A3 Set Parameters

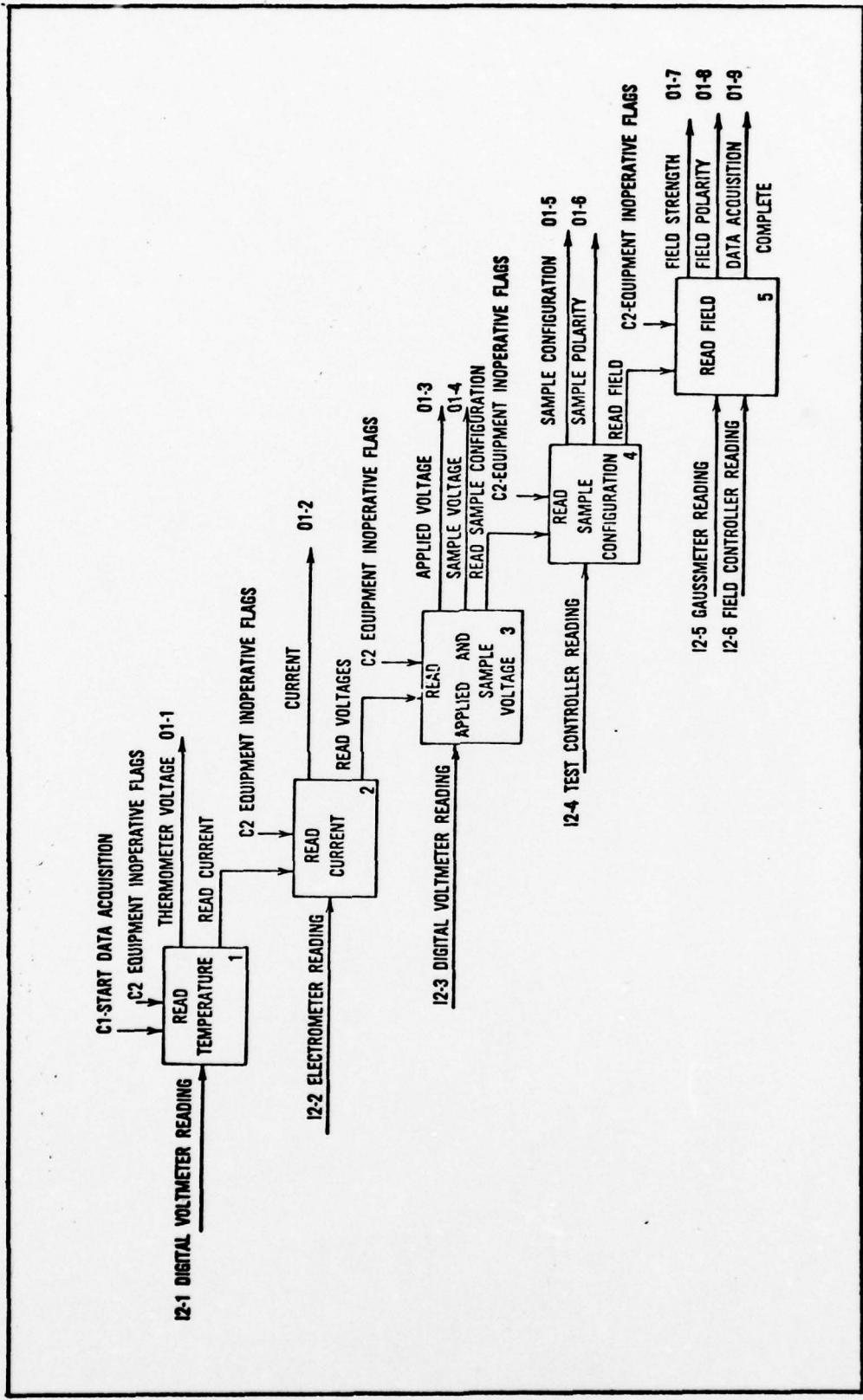


Figure 10. Node A4 Acquire Data

A35 --- Generate Acquire Data Signal

A36 --- Delay Recheck

When the Set Parameters signal is received, the activities Check/Set Field, Check/Set Applied Voltage, Check/Set Sample Configuration, and Check/Set Temperature each check the value of the parameters under their control. They each generate a setting if needed and generate either a value correct or an anticipated settling delay time signal. These modules execute serially in this order based on which are the most stable parameters. The most stable parameters are set first and left to settle to steady state. Temperature is the most unstable of the variables. As it varies, it affects the sample current and voltage readings. Therefore, it is set last so that a minimum of time will elapse after a parameter OK signal is generated and the Start Data Acquisition signal is sent. After Check/Set Temperature is through, it passes control to the activity Generate Acquire Data Signal, which checks whether all Value Correct signals have been received. If they have, the Start Data Acquisition signal is generated. If they have not, control is passed to the activity, Delay Recheck, which delays for the longest of the received delay time signals before passing control back to Check/Set Field to repeat the cycle until all values are correct. The Start Data Acquisition signal passes control to Node A4.

Node A4, Acquire Data (Figure 10)

Node A4 is subdivided into the following five functions:

- A41 --- Read Temperature
- A42 --- Read Current
- A43 --- Read Applied Voltage and Sample Voltage
- A44 --- Read Sample Configuration
- A45 --- Read Field

When the Start Data Acquisition signal is received, the five activities read their parameters or, if their Equipment Inoperative Flag is set, request that the operator input the data reading from the terminal. While the order in which the data is taken is not extremely critical, the most unstable parameters, temperature and current, are read first whenever possible. When the field is on, temperature cannot be accurately read. Therefore, for these cases, the temperature is saved to last. All parameters except temperature are read, the field is turned to zero and temperature is then recorded. The data is stored in the data array and the Data Acquisition Complete signal is generated. This passes control back to node A2, Set Parameters, which repeats its function as previously discussed.

Node A5, Reduce Data (Figure 11)

Node A5 is subdivided into the following seven activities:

- A51 --- Compute Temperature
- A52 --- Compute Data Output
- A53 --- Print Plot Data
- A54 --- Plot Data
- A55 --- Print Real-Time Data

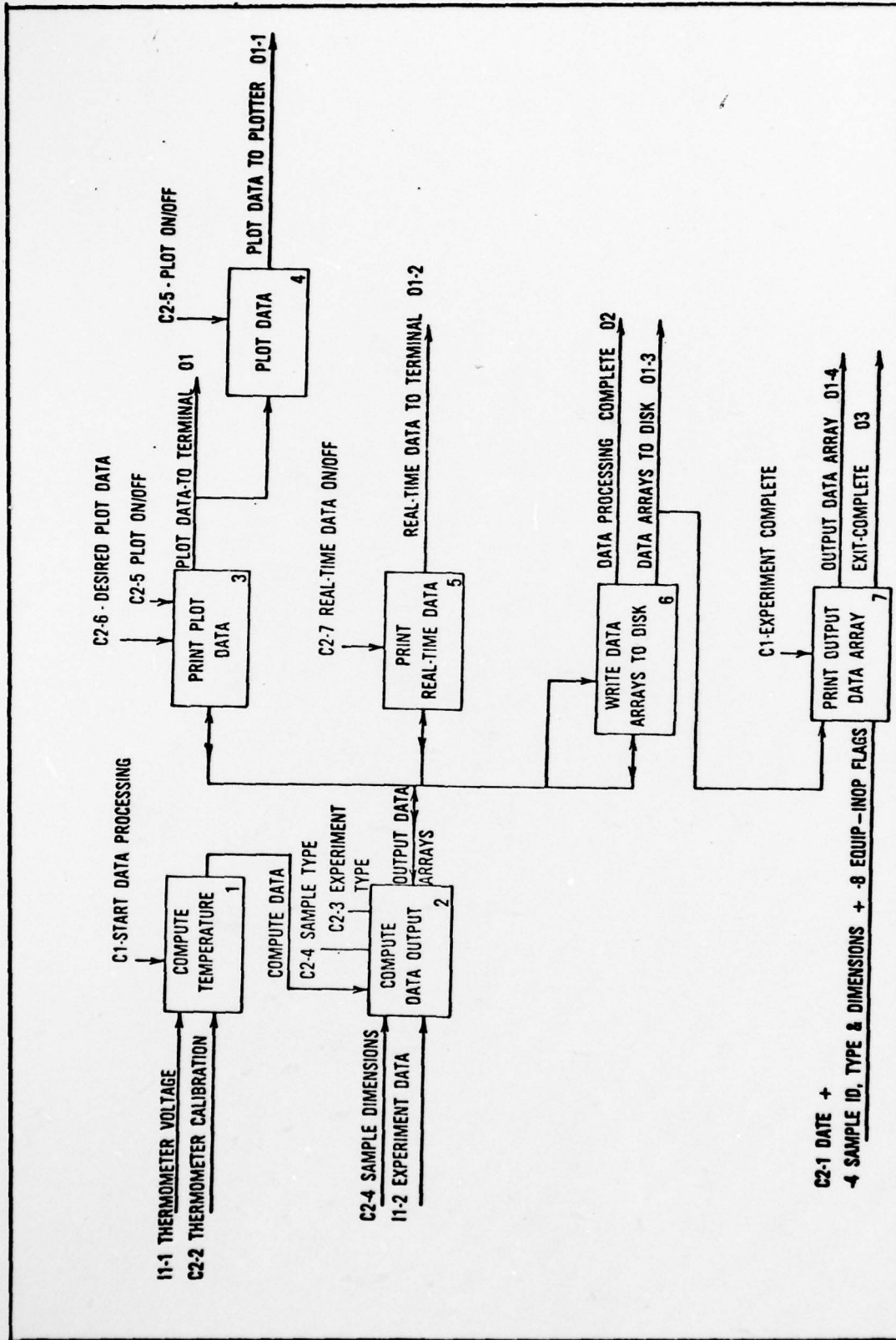


Figure 11. Node A5 Reduce Data

A56 --- Write Data Arrays to Disk

A57 --- Print Output Data Array

When the Start Data Processing signal is received from Node A2 (after the completing of each block of data), the activity Compute Temperature converts the voltage readings from the thermometer into the calibrated temperature values. Then, the activity Compute Data Output calculates all of the required output data (the composition of this output data was detailed in Chapter II, Theory). The appropriate data streams are then passed, as required, to the activities Print Plot Data and Plot Data, Print Real-Time Data, and Write Data Arrays to Disk, in turn. When A56 Write Data Arrays to the Disk is complete, the Data Processing Complete signal is generated. This passes control back to A2 Set Parameters. When the Experiment Complete signal is generated, control is passed directly to the activity Print Output Data Array. This module reads the complete Output Data Array from disk and outputs it to the terminal. It then terminates and control returns to the computer operating system.

Summary

In this chapter, the overall system requirements were defined. All activities that are to be accomplished by AHEEDAS have been documented. This design was presented to the sponsor, Dr. Patrick M. Hemenger, AFML/LPO. He concurred with the design. No engineering decisions that were not dictated by the system constraints, had been made at

this point. The next step is the design stage where detailed breakdowns of the activities may be used, if necessary, to facilitate design. Additional structure charts and flow diagrams will be used to document the algorithm design in Chapter IV.

IV. System Design and Implementation

After the requirements of the system were defined (see Chapter III), this definition had to be transformed into the actual design of the system. This design was guided by, but not completely bound by, the SADT definition. Several factors caused departures from the SADT design. Some modules were found to be trivial and were combined with other related modules. Some control signals were needed that were implemented differently from those on the SADT definition. The overall design is shown in heirarchical charts in Figures 12 and 13. In this chapter, the heirarchical design will be discussed along with the implementation of the algorithms.

Heirarchical Design

Figure 12 shows the overall AHEEDAS system. All system modules are shown excluding the implementation of the device drivers. This implementation is shown in Figure 13. These software modules correspond to the nodes of the SADT definition except for A44 Read Sample Configuration which was combined with A4 Acquire Data. Module A0 Conduct Experiment and Process Data is the executive for the AHEEDAS system. It contains all of the data specifications and it calls each of the five main submodules, A1 Initialize Parameters - A5 Reduce Data, as they are needed. All required intermodule data and control signals pass through A0 Conduct Experiment and Process Data.

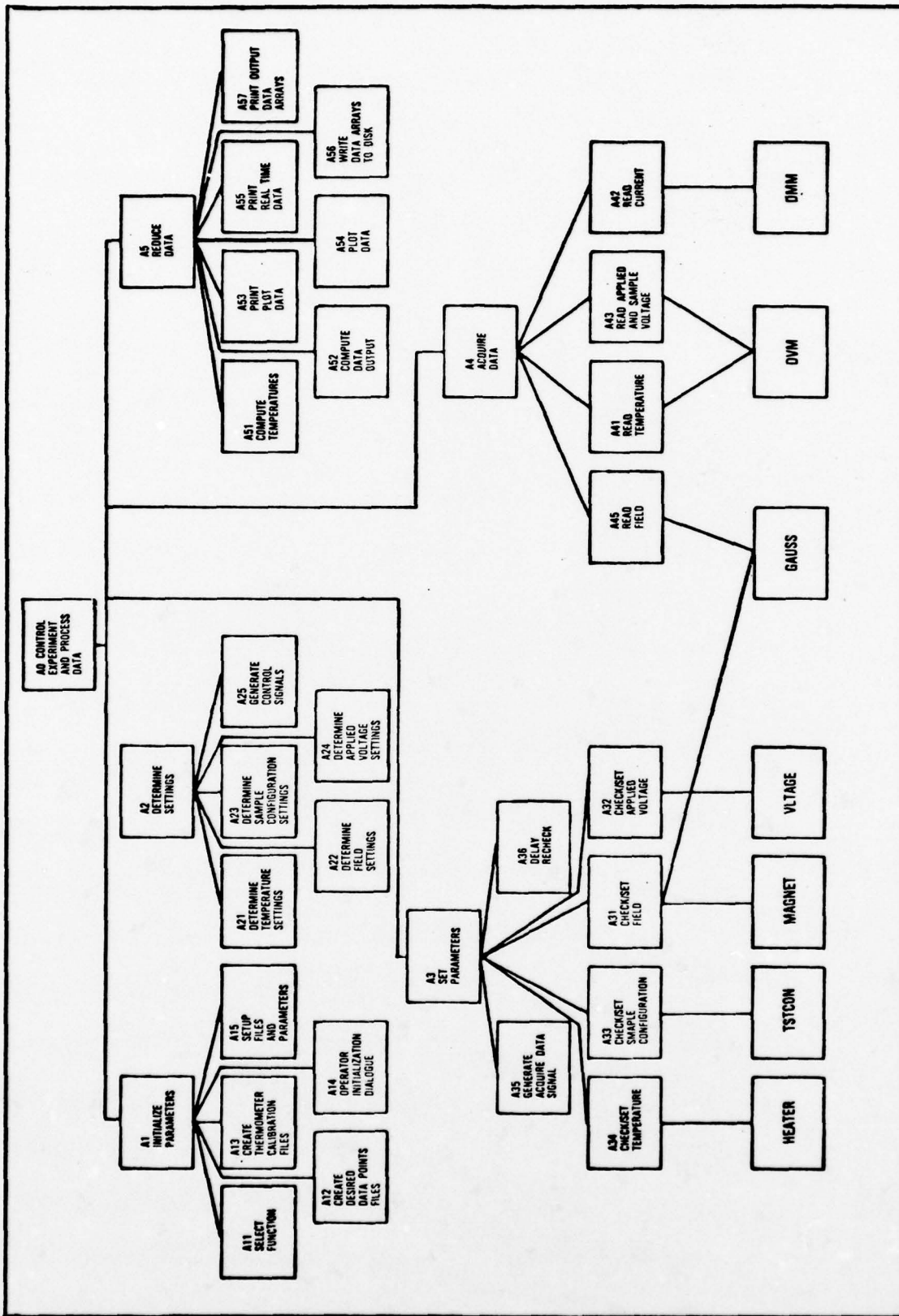


Figure 12. Hierarchical Chart of AHEEDAS

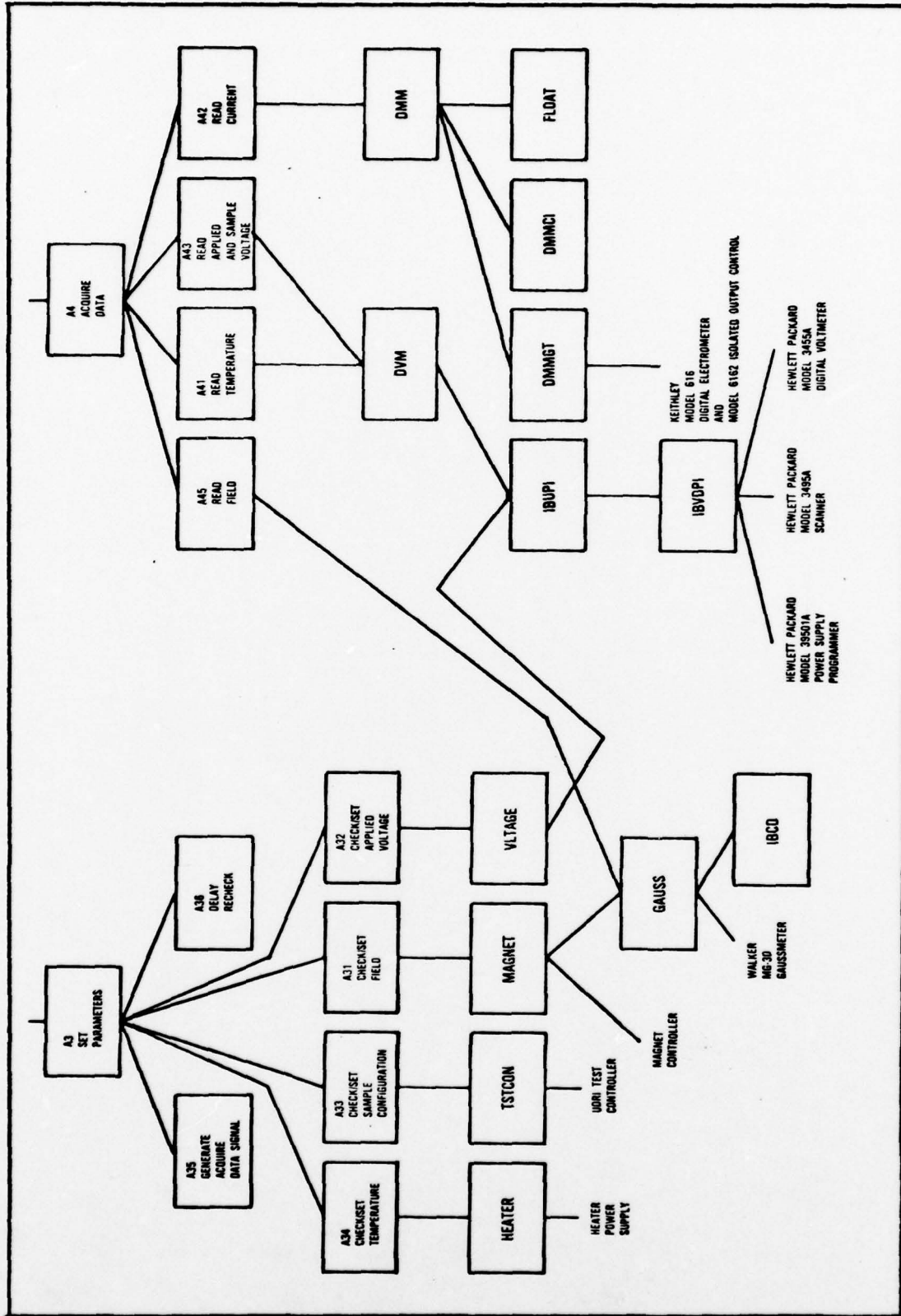


Figure 13. Hierarchical Chart of A3 and A4

Modules A1 Initialize Parameters - A5 Reduce Data are the second level executive modules for the system. They coordinate the flow of data and control signals between their subordinate modules.

Module A1 Initialize Parameters is the first module called by A0 Conduct Experiment and Process Data. It initially calls A11 Select Function which requests the user to enter his choice of what to do next. This choice is passed back to A1 Initialize Parameters which calls the appropriate function. Once initialization is complete (module A15 Setup Files and Parameters has finished), A1 Initialize Parameters returns to A0 Conduct Experiment and Process Data which calls A2 Determine Settings.

Module A2 Determine Settings calls each of its subordinate modules in turn to determine the initial data parameter settings. The data settings are also organized in an heirarchical manner. Therefore, each module will not need to be called each time that A2 Determine Settings is called to provide the parameter settings. Control signals are provided so that A2 Determine Settings can determine which modules need to execute for each run. When a block of data is complete A2 Determine Settings passes the Start Data Processing signal back to A0 Conduct Experiment and Process Data. A0 Conduct Experiment and Process Data will then call A5 Reduce Data. When A2 Determine Settings returns to A0 Conduct Experiment and Process Data without setting the Start Data Processing signal A0 Conduct Experiment and Process Data calls A3 Set Parameters.

A3 Set Parameters takes the parameter settings from A0 Conduct Experiment and Process Data and calls each of its subordinate modules. The modules will then check to see if the value is already set and if it is not, will set the proper value by calling the appropriate device driver. The order in which these parameters are set is very important. A3 Set Parameters will not allow A31 Set Field to set the field if the field has other than a zero value, before the temperature is set. This is necessary because the field will affect the reading of the silicon thermometer which is used to set the temperature. As each module sets its parameter, it will pass a parameter OK signal to A3 Set Parameters. If the parameter needs time to settle to a steady state value, the module will not set the parameter OK true but will instead pass a settling delay time back. If all parameter OK signals are not received, A35 Generate Acquire Data Signal sets the delay value to the longest delay time requested by the other modules and returns to A3 Set Parameters. A3 Set Parameters then calls A36 Delay Recheck which waits the required amount of time and then returns. A3 Set Parameters then causes the Check/Set modules to execute again. When all four have executed A35 Generate Acquire Data Signal again checks the parameter OK signals and, if they are true, generates a Start Data Acquisition signal. The A34 Check/Set Temperature module was implemented manually. As discussed in Chapter I, Constraints/Assumptions, the automated temperature controller could not be implemented within the scope of this study. Therefore, this module requests that

the temperature be set by the operator. He must then enter a signal that the temperature is set and at steady state. Because of the delay that this requires, the other settling delays are reset to zero whenever manual input is required. The operator's input signal is then treated like a parameter OK signal. When A3 Set Parameters returns to A0 Control Experiment and Process Data, A4 Acquire Data is called to read the data.

Module A4 Acquire Data reads all of the required data parameters by calling A41 Read Temperature - A45 Read Field, in turn. A44 Read Sample Configuration was combined with A4 Acquire Data and does not execute separately. They read their parameters by calling the appropriate device interface drivers. As soon as A45 Read Field returns, A4 Acquire Data returns to A0 Control Experiment and Process Data calls A2 Determine Settings which determines whether or not the data block is complete. If it is, A2 Determine Settings returns to A0 Control Experiment and Process Data with the command to start data processing. A0 Control Experiment and Process Data then calls A5 Reduce Data.

Module A5 Reduce Data first calls A51 Compute Temperatures to convert all of the temperature voltage data into temperatures. It does this by using a calibration table placed into memory by A1 Initialize Parameters. A55 Print Real-Time Data is called next, if the user has requested real-time data printout, to print out the raw data. Next A52 Compute Data Output is called to compute the remaining output data.

The calculations required for these data were discussed in Chapter II, Theory. A53 Print Plot Data is called next if the user has requested a plot. It prints the data pairs to be plotted. A54 Plot Data is then called to plot the data. A54 Plot Data is implemented as a non-functioning module because no printer has been purchased. When one is, the user will be able to implement it easily by putting the necessary software into this module. Up to four data pairs can be specified to be plotted. Then A56 Write Data Arrays to Disk is called to write all of the data to the disk files. If the experiment is complete (the Experiment Complete flag set true by A2 Determine Settings), A57 Print Output Data Array is called to print out all of the processed output data in tabular form. Otherwise, A5 Reduce Data returns to A0 Control Experiment and Process Data after executing A56 Write Data Arrays to Disk. A0 Control Experiment and Process Data then resets all of the control signals and settings for the next run.

Implementation

The software used to implement the above system are presented as listings in Appendix C. Listings of all modules are shown except for the National Instruments software which controls the IEEE 488 Instrumentation bus. This is available from National Instruments (Ref 1). Where needed, flow charts are included with each module to clarify its design. The main body of the algorithm (Node A0) was interfaced to the necessary device driver programs. In

writing the code, the author attempted to use structured code wherever possible. Named common was used for passing most of the parameters between subroutines. This allowed the parameters to be grouped into functional groupings and labeled. Additionally, the RT-11 operating system lists all named common blocks on the Link Map. This aided in debugging the programs and should aid in system maintenance. All of the subroutines pass control heirarchically. No direct calls are allowed between lower level modules. This eliminated confusion about control paths.

Method of Implementation

The language Fortran IV was used throughout the system design at the request of the user. The use of this higher order language will enable the user to more easily maintain the system. This implementation required a lot of memory space and, therefore, an overlaid program structure was required (Ref 20 and 21 describe the use of overlays in RT11 Fortran programming). A diagram of the overlay structure for AHEEDAS is shown in Figure 14.

To overlay a program the programmer must divide his program into several segments which do not have to be resident in memory simultaneously. A control or root segment must contain the main program, system library routines and any other program segments which must be present in memory at all times. The remainder of the program is divided into which need not be co-resident in core memory. The operating system swaps these segments in and out of memory from the desk

writing the code, the author attempted to use structured code wherever possible. Named common was used for passing most of the parameters between subroutines. This allowed the parameters to be grouped into functional groupings and labeled. Additionally, the RT-11 operating system lists all named common blocks on the Link Map. This aided in debugging the programs and should aid in system maintenance. All of the subroutines pass control heirarchically. No direct calls are allowed between lower level modules. This eliminated confusion about control paths.

Method of Implementation

The language FORTRAN IV was used throughout the system design at the request of the user. The use of this higher order language will enable the user to more easily maintain the system. This implementation required a lot of memory space and, therefore, an overlaid program structure was required (Ref 20 and 21 describe the use of overlays in RT11 FORTRAN programming). A diagram of the overlay structure for AHEEDAS is shown in Figure 14.

To overlay a program, the programmer must divide his program into several segments which do not have to be resident in memory simultaneously. A control or root segment must contain the main program, system library routines and any other program segments which must be present in memory at all times. The remainder of the program is divided into segments which need not be co-resident in core memory. The operating system swaps these segments in and out of memory

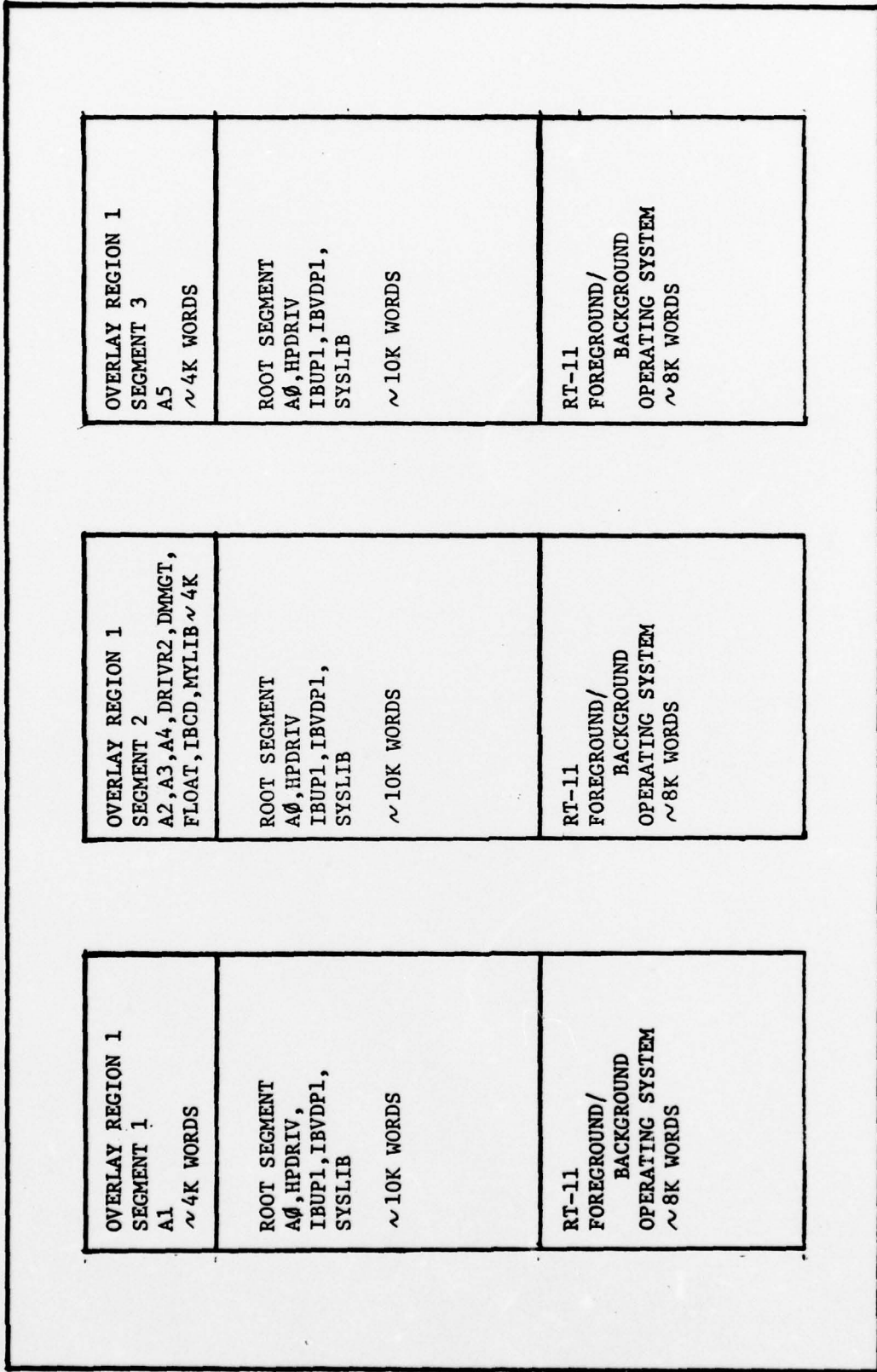


Figure 14. Overlaid Program Structure

from the disk storage unit as they are needed. Care must be taken to divide the program segments so that the segments are swapped as infrequently as possible. Otherwise, system performance will be seriously degraded by the large amount of system time spent swapping the overlays in and out of memory. This structure is very important. AO, any driver routines utilizing interrupts and the system library must be in the root segment. The division of the three segments of the overlay region is optimum for minimizing the overlay time. Any other structure resulted in an unacceptably high amount of time spent shifting the overlay segments in and out of memory.

V. System Testing and Validation

Any large system, such as AHEEDAS, must be tested in stages. The first stage was incremental testing to ensure that all levels of AHEEDAS would function. Next, the basic system was validated. The object of validation was to show that, if validation data were normally entered, AHEEDAS would produce the correct outputs. Then comes hardware testing in which all of the device drivers were functionally checked. Last, final validation was performed. To accomplish this, the entire AHEEDAS was integrated. The system was then used to run experiments of each type (see Chapter II for types). In this chapter, these stages of testing and validating the AHEEDAS will be discussed.

Incremental Testing

As the AHEEDAS was written, each module and submodule was tested individually. All modules were thus functionally tested prior to being integrated into the system. AHEEDAS was written and tested from the top down. This required AO to be functionally tested with substitute submodules first. Then, the next level of modules was written and functionally tested with substitute modules for the third level. Next, each major module (examples: A1, A2, ...) was tested in its final form with all submodules, except the device drivers, included. Finally, all of the system modules were combined with small printing programs substituting for the actual equipment driver programs and the entire system was functionally tested. During functional testing, the primary

concern was to see that each activity executed as planned in the design and met the system requirements (reference Chapter III). Towards this end, extra output statements were used in the code to ensure that the modules were executing in the proper order and passing the proper parameters to each other. Functional testing alone, however, is not sufficient to validate the system operation.

Validation

To validate the system, it had to be confirmed that for all normal data inputs, the system produced correct results. To assure that the AHEEDAS was a valid system, the small substitute equipment drivers were again used in the system. These drivers allowed the operator to manually input the required data to the system. Past manual runs of the experiment were used to provide valid raw data inputs. This raw data had been reduced on the CDC 6600 computer and this output was used for comparison. The experimental raw data were entered through the substitute driver programs. The reduced data output was then compared with that from these past manual runs. (Since this was an algebraic validation, precise agreement was necessary. Because the degree of accuracy used was the same, there was no problem with the CDC 6600's greater accuracy capability.) Any discrepancies were traced and the problems corrected. The process was repeated until the data outputs of AHEEDAS agreed with the manual data outputs. Runs of all three experiment types were used: van der Pauw sample -- all sample configurations

used; van der Pauw sample -- abbreviated sample configurations (reference Chapter II, Original Environment); and Hall bar sample. The AHEEDAS provided valid outputs for all three types. The next step in the testing was then to connect the system to the actual hardware.

Hardware Testing

Before final implementation of the AHEEDAS, it was necessary to validate the interface of the hardware to the device driver programs. This was done by using a substitute test program which merely called the instrumentation to perform all of the functions that the AHEEDAS would require. Major problems were discovered in the IEEE 488 Standard Bus (National Instruments GPIB-11). National Instruments replaced both the interface card and the software drivers for this interface. This solved the problems with the bus. However, the interface has not been extensively used in the field and the documentation on its use is not complete. Future expansion of the system could uncover more problems. The magnet driver was an in-house produced interface. Numerous trials were necessary to get this interface to work properly. The Walker gaussmeter did not work properly. It functions erratically and should not be used in AHEEDAS until it is repaired. The AHEEDAS was designed to work with or without a functioning gaussmeter. The other interfaces worked properly with only minor problems which are corrected. When the proper functioning of these interfaces was confirmed, all that was left was to

test the full AHEEDAS in its final configuration.

Final Validation

Final validation was relatively simple. The LSI-11 and all associated hardware systems were connected into the final experimental configuration and experiments of each type were run. Data was then checked for reasonableness, consistency and, as far as possible, for accuracy. This was done by comparison with past manual experiments on similar samples. No point by point comparison was made. It was shown in Validation that for valid input, AHEEDAS produced valid output. The instrumentation used had been checked out in use for months. Therefore, it was assumed that the data input from the instruments was valid. The primary concern was the ability of the user to perform a complete experiment with consistent results. The users performed most of these tests under the guidance of the author. Thus, user familiarization was accomplished concurrently. No major problems were uncovered in this stage of testing. Minor corrections were all that were necessary to bring AHEEDAS to full functional status.

Summary

In this chapter, the process of validating the AHEEDAS was discussed. The AHEEDAS met all of the user's specifications for the first stage system. In the next chapter, the conclusions of this investigation will be discussed and recommendations for further study presented.

VI. Recommendations and Conclusions

During the course of this investigation, many problems were uncovered. New questions were raised. Not all of them were within the scope of this study to investigate. Recommendations for further investigation will be presented in this chapter. Then, the conclusions drawn from this study will be presented.

Recommendations

The design and implementation of the automated temperature control function of AHEEDAS remains to be accomplished. The temperature is currently controlled by an analog, closed-loop controller which supplies current to a heater element in the sample dewar. It gets feedback from its own uncalibrated thermometer elements and holds the voltage read from this thermometer constant by varying the heater current. The tolerance of this controller is approximately 0.01 degrees Kelvin. The user must use this controller and a separate voltmeter which reads a calibrated silicon thermometer to set a specific temperature. Two approaches appear to be the most promising. The first approach would be to use an existing computer controlled power supply to do the job. This would require the use of the same LSI-11 processor for temperature control that is used to run the rest of the experiment. It is not clear at the present time whether the temperature is stable enough to permit the use of this scheme. The differential between the temperature inside the sample dewar and the laboratory room temperature

can be as high as 290 degrees Kelvin at the low end of the temperature range. The temperature is very sensitive to the outside environment (for example, air currents caused by opening and closing doors or people walking by the apparatus affect the temperature). Since the activity of the analog controller is transparent to the user, how often the heater current must be adjusted to hold a constant temperature is not known. Experiments need to be conducted with a fully manual controller to determine the stability of the temperature. Another approach would be to use another processor to build a closed-loop temperature controller. A separate dedicated processor would be able to continuously adjust the heater settings to prevent the temperature from drifting. This controller would take the temperature setting from the LSI-11 and set the required value. This would free the LSI-11 to do other processing as required until interrupted by the controller when the temperature was at the proper steady state value. This appears to be the most desirable of the two options. However, it is not known at this time whether or not it is necessary to invest the extra money into the implementation of the second approach. Space was left in Module A3 Set Parameters to implement this function.

The user would like to add a plotting capability to the AHEEDAS. He desires that a four-color plotter be utilized. This device has not yet been purchased. The primary device being considered was listed in Chapter II in the list of the system hardware. A blank module was inserted into the

system for the software that will be needed for this interface. The plotter which has been tentatively specified, the HP7221A, will require a small driver to be written to control the system software that will be provided with the plotter.

AHEEDAS is a large system and required the use of an overlaid program structure to implement. The physical size of the algorithm could be considerably reduced if some of the driver modules had possibly some of the smaller system modules were written in MACRO-11, the system assembly language. The present system requires almost 16K words of memory. Over 4K words of useable memory remain. However, future system growth could mandate that some optimization of this type be done. If such optimization is done, the overlaid program structure discussed in detail in Chapter IV should be used for all future expansions. Any other structure will result in a considerable loss of time simply reading the overlay segments in and out of memory. Numerous options for optimizing program size are discussed in Reference 20. These compiler options can be used to reduce the program size with only small run time increases.

Once an experiment is complete, the output data from AHEEDAS still must be entered manually into the CDC 175 computer so that it may be analyzed in depth. AFML is presently installing a Prime Model 550, medium frame computer which is linked into the CDC computer. One of the purposes of this computer is to act as a buffer between the AFML minicomputer network and the CDC computer. The user

desires to interface to this computer via an acoustic coupler. The interface package to support this function needs to be developed so that the experimental data can be read directly into storage files in the CDC computer. The data could then be read directly into the analysis programs.

Conclusions

The AHEEDAS implemented is the first stage of a fully automated system. The initial design and implementation of this system were successfully completed. The LSI-11 micro-computer proved to be very satisfactory for the job. The initial software fulfilled the specifications for the first stage system (Ref Chapter 1). Time did not permit further investigation into the later steps in system development. Possible development for implementation in later stages of AHEEDAS were discussed in the recommendations in this chapter.

Bibliography

1. GP1B11V-1, Operating and Service Manual. Austin, Texas: National Instruments, 1977.
2. GP1B11V-1, Software Reference Manual. Austin, Texas: National Instruments, 1977.
3. Hall/van der Pauw Interface. Dayton, Ohio: University of Dayton Research Institute, 1979.
4. Hemenger, Patrick M. "Measurement of High Resistivity Semiconductors Using the van der Pauw Method," Review of Scientific Instrumentation, 44:689-700 (June 1973).
5. HP-IB Isolated D/A Power Supply Programmer, Model 59501A, Operating and Service Manual. Loveland, Co.: Hewlett-Packard Company, 1977.
6. IEEE Standard Digital Interface for Programmable Instrumentation. New York: The Institute of Electrical and Electronics Engineers, Inc., 1975.
7. IDEF, Architect's Manual, ICAM Definition Method, (Version 0). Waltham, Mass.: Softec, Inc., 1978.
8. Iftekar, Saleem. Microprocessor Based Data Acquisition and Processing Thesis, AFIT/GE/EE/79-29, ADAO 64728. Dayton, Ohio: Air Force Institute of Technology, 1978.
9. Instruction Manual, Model 616, Digital Electrometer. Cleveland, Ohio: Keithley Instruments, Inc., 1975.
10. Instruction Manual, Walker/Magnametrics Gaussmeter, Model MG-3D. Worcester, Mass.: Walker Scientific, Inc., 1974.
11. An Introduction to SADT, Structured Analysis and Design Technique. Softech, Inc., November 1976 (9022-78R).
12. Microcomputer Handbook Series, Memories and Peripherals. Maynard, Mass.: Digital Equipment Corporation, 1978.
13. Model 3455A Digital Voltmeter, Operating and Service Manual. Loveland, Co.: Hewlett-Packard Co., July 1977.
14. Model 3495A Scanner, Programming and Service Manual. Loveland, Co.: Hewlett-Packard Co., November 1977.

15. Operating Information, Model 3455A Digital Voltmeter. Loveland, Co.: Hewlett-Packard Co., July 1977.
16. PDP-11 Fortran Language Reference Manual, RT-11 V3. Maynard, Mass.: Digital Equipment Corporation, June 1977.
17. PDP-11 Macro-11 Language Reference Manual. Maynard, Mass.: Digital Equipment Corporation, November 1978.
18. Reeve, William H. and Stinson, Jerry L. Software Design for a Visually Coupled Airborne System Simulator (VCASS). Thesis, AFIT/GCS/EE/78-6. AD A055226. Dayton, Ohio: Air Force Institute of Technology, March 1978.
19. RT-11 Advanced Programmers Guide, RT-11 V03B. Maynard, Mass.: Digital Equipment Corporation, November 1978.
20. RT-11/RSTS/E Fortran IV User's Guide, RT-11 V03B. Fortran IV V02. Maynard, Mass.: Digital Equipment Corporation, June 1977.
21. RT-11 System User's Guide, RT-11 V03B. Maynard, Mass.: Digital Equipment Corporation, March 1978.
22. van der Pauw, L., Jr. "A Method of Measuring Specific Resistivity and Hall Effect of Discs of Arbitrary Shape," Philips Research Report 13, 1958.

Appendix A

Structured Analysis and Design Tool

Introduction

This appendix contains a brief explanation of the functional analysis phase of the Structured Analysis and Design Tool (SADT) to aid the reader in understanding the design development. A more complete discussion can be found in References 11 and 7. The method used here is a subset of the full tool. The explanation used below is condensed from Reference 18.

Structured Analysis

SADT is a comprehensive methodology for performing functional analysis and design. In the functional analysis phase, the emphasis is on analyzing and documenting the requirements on the system. A set of diagrams results which are called activity diagrams. They describe the system in terms of the activities it must perform. The diagrams are created by decomposing the system into smaller and smaller pieces. The completed set of diagrams provide a model of the system.

Diagram Syntax

SADT diagrams consist of labelled boxes and arrows for expressing the system activity and data models. Figure 15 illustrates the basic syntax of the model. Inside a box is the name of the activity model. This name expresses the action taking place.

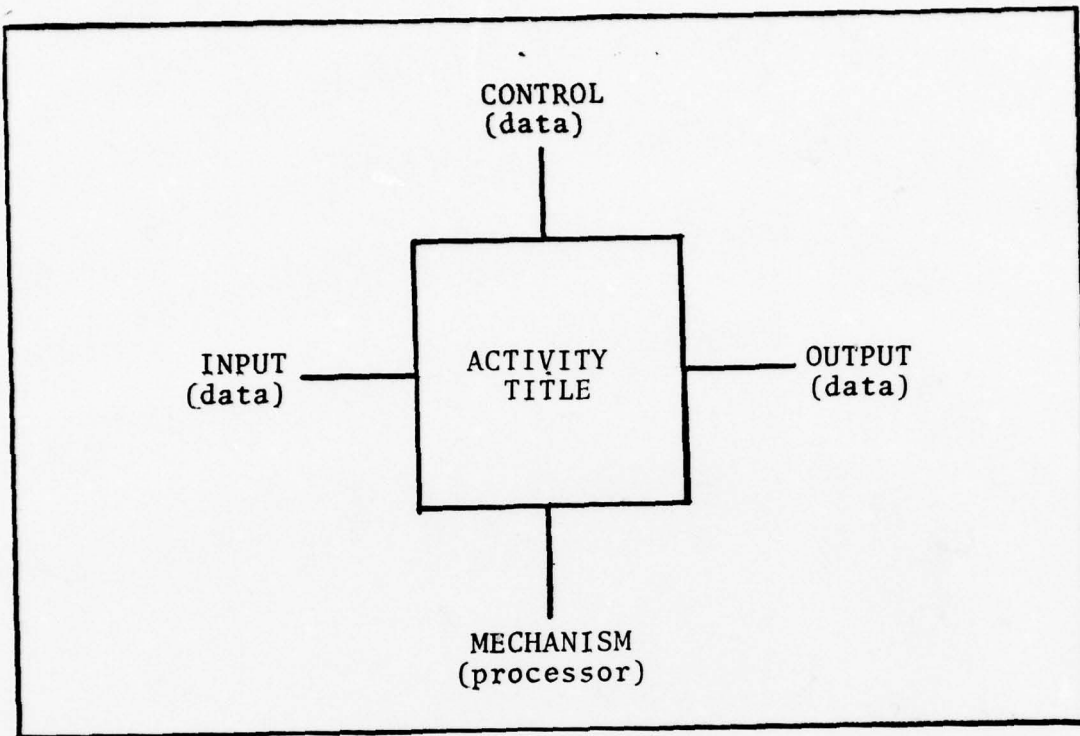


Figure 15. Box/Arrow Conventions

The boxes of the parent diagram are decomposed into more detailed diagrams called children. Each diagram is numbered in a Dewey-decimal manner (Ref 11:2-3) which represents the parent-child relationship. For example, diagrams 31, 32, 33 and 34 would be children of diagram 3. Each diagram is referenced as a node.

The boxes of a diagram are connected by arrows which represent the interface between the boxes. The sides of the box define the kinds of arrows which may enter or leave that side of the box.

Four types of arrows represent the kinds of interface. As in Reference 11:3-6, for activity diagrams these are:

Input: Data transformed by the activity into the output.

Output: Data created by the activity.

Control: Data used to control the process of converting the input into the output.

Mechanism: The processor which performs the activity.

It should be noted that the "mechanism" arrow represents the tool necessary to "realize the box" (Ref 11:3-4); since it is usually evident from the title of the box, the mechanism arrow is not always shown.

The "multiple branch" (EXCLUSIVE OR) is used to indicate multiple, but not simultaneous, outputs. The "multiple join" indicates, multiple but not simultaneous, inputs. Both conventions are shown in Figure 16. SADT also permits the use of simultaneous joining of signals into a pipeline of data.

An "ICOM" code is used to connect arrows across the parent/child boundaries. The name ICOM is derived from the arrow names: Input, Control, Output, and Mechanism. Each boundary crossing arrow (ones which do not have both ends connected to a box is labeled with its parent-context ICOM code, in addition to its normal label. This aids the reader in locating the matching parent arrow. The "ICOM" code is written near the unconnected end of the arrow and consists of the letter I, C, O, or M followed by a number. This number gives the relative position that the arrow enters or leaves the side of the parent box. Numbering is

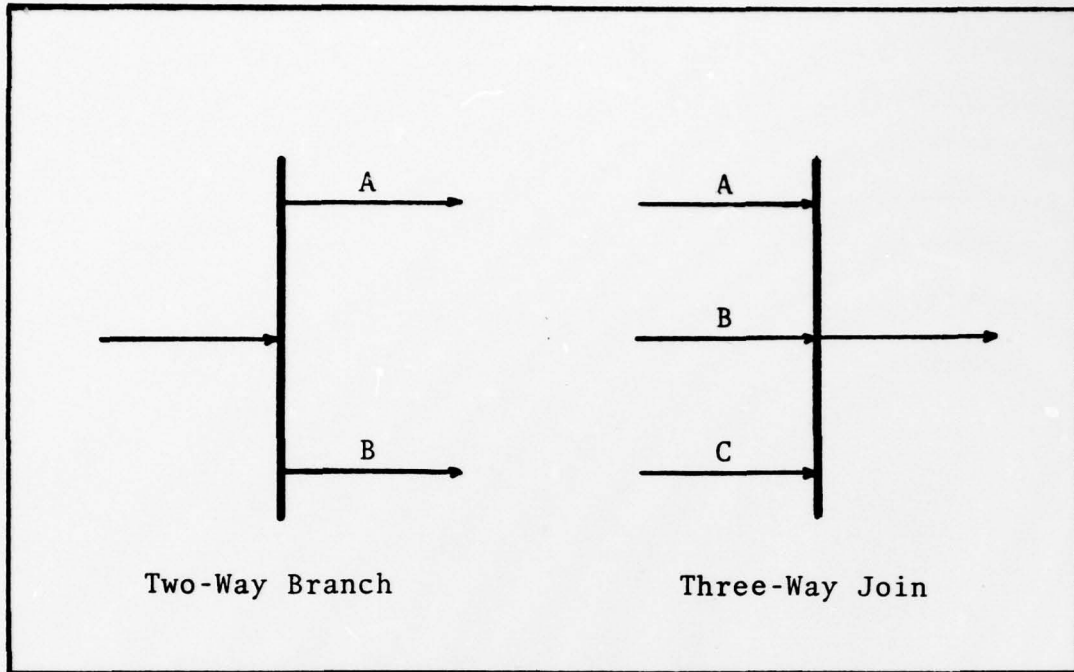


Figure 16. OR Branch and Join Structure

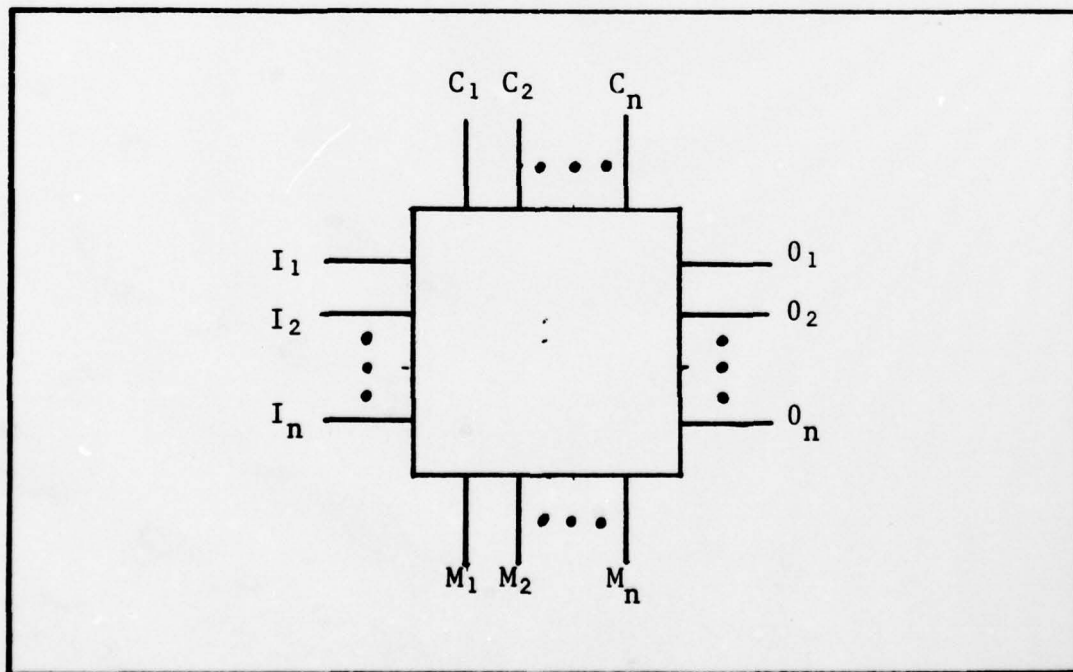


Figure 17. ICOM Numbering Convention

done from left to right and top-to-bottom as illustrated in Figure 17. For example, "C2" on an arrow in a child diagram indicated the arrow is the second control arrow entering the parent box.

In the text associated with each diagram, the arrows are identified with an "ICOM" code consisting of a letter (I, O, C, or M), a suffix number, and, where necessary for clarity, a dashed number referring to the position of the signal in a pipeline of data. The code refers to the box within the diagram and the suffix number refers to the top-down or left-right order of the arrow on the box. For example, "C2-6" refers to the sixth data item in a pipeline of data which entered the parent box at C2.

Appendix B

Operator's Manual for AHEEDAS

Every attempt has been made to make the AHEEDAS simple to use. Only a rudimentary knowledge of computer operation is needed to operate the system. A checklist of instructions for use of AHEEDAS follows.

I. Initialization

A. Turn on power to all LSI-11 systems.

1. Turn on power to the LSI-11 computer cabinet.
2. Turn RUN/HALT switch on LSI-11 front panel to the RUN position.
3. Turn ON/RTC switch to the ON position.
4. Turn ON/POWER switch to the ON position.

B. Put floppy disks into the system.

1. Insert system disk into Drive 0 of the RX01 drive.
2. Insert data disk into Drive 1.

CAUTION: THE DATA DISK MUST HAVE AT LEAST 130 CONTINUOUS FREE BLOCKS BEFORE STARTING AHEEDAS.

CAUTION: IF THE THREE DATA FILES RAWOUT.DAT, INTRMD.DAT, AND OUTPUT.DAT ARE PRESENT ON THE DISK, THEY SHOULD BE RENAMED BEFORE RUNNING AHEEDAS OR THEY WILL BE DESTROYED. RENAME FILES USING THE COMMAND RENAM DX1:RAWOUT.DAT DX1:Filename. THE DEVICE NAME MUST BE THE SAME FOR BOTH FILE SPECIFICATIONS.

C. Boot the system.

1. Type DX<cr> on the terminal.
2. When the system startup messages finish printing, type: DATE day-month-year (example: DATE 10-OCT-79) to enter the current date.

D. Run AHEEDAS.

1. Type: RUN DX):AHEDAS<cr> to start the system running.

2. The system will respond:

```
D(ATAFILE,T(EMCALFILE,I(NITIALIZE,Q(UIT
```

Enter the letter corresponding to your choice.

- a. For choice D(ATAFILE and choice T(EMCALFILE respond to all system prompts to create the desired files. If disk space permits all such input files should be stored on the system disk in Drive 0 by entering the file specification:

```
DX0:Filename
```

when prompted for the filename.

- b. Choice I(NITIALIZE should not be selected until after equipment turn on is complete.

E. Equipment Turn On and Initialization

1. Mount the sample and begin cool down procedure.
2. Turn on all necessary instruments.
 - a. HP3445A -- Digital Voltmeter, ensure that the proper input selected, front or back terminals.
 - b. HP3495A -- Scanner.

- c. Keithley 616 Electrometer and 6162 Isolated Output Control -- Set sensitivity AUTO. FAST/NORMAL=FAST, RANGE= $10^{**} - 11$ Amperes.
 - d. Walker Gaussmeter -- set to 10,000 Gauss scale, ensure probe is installed to read proper field polarity.
 - e. Magnet and controller -- AUTO/MANUAL switch to AUTO, Field Selector=0.
 - f. UDRI Test Controller -- Set to AUTO.
 - g. Heater power supply and Artronix Controller.
 - h. Power supply to silicon thermometer.
 - i. Any other required equipment needed for this experiment.
3. If any of the automatic instrumentation is inoperative, the experiment can still be performed manually if a substitute piece of equipment is available. (Note: No substitute is needed for the gaussmeter as it is redundant.) The user must simply enter which piece of apparatus is inoperative and stand by to enter the readings manually.

F. Initialize Experiment

1. To begin initializing parameters for an experiment, respond to the prompt in D.2 above with choice 3. Respond to all system prompts for information and enter the appropriate entry for any inoperative system components when prompted. When

finished, the system will prompt:

INITIALIZATION IS COMPLETE

ENTER A 1 WHEN READY TO START EXPERIMENT

Before responding, recheck all experimental systems to assure that they are all turned on, warmed up, initialized and at steady state values. When finished, enter a 1<cr> to start the experiment.

II. Conduct Experiment

- A. The AHEEDAS will execute the experiment using the the information you have given it. Each time that it needs a reading or an equipment setting done manually, it will ask for it by a prompt on the terminal. Make sure all data is entered carefully just as the system requests so that accuracy will be maintained.

CAUTION: IMPROPER ENTRY OF DATA CAN RESULT IN PROGRAM ABORT. TO RESTART, RE-EDIT DESIRED DATA POINTS FILE TO BEGIN AT FAILURE POINT, REINITIALIZE AND BEGIN AGAIN. BE SURE TO RENAME THE DATA FILES FROM THE FIRST ABORTED RUN OR THE SECOND RUN WILL DESTROY THEM.

III. Experiment Completion

- A. AHEEDAS will output the data files and stop execution when finished.
- B. Accomplish normal shutdown of all equipment when finished.

C. Shutdown LSI-11 computer.

1. Remove both floppy disk and store safely.
2. Move RUN/HALT switch to HALT.
3. Move ON/RTC switch to RTC.
4. Move ON/POWER switch to the down position
(Note: All three switches should now be in
the down position.)
5. Turn off power to the computer cabinet.

IV. To Interrupt Experiment

- A. Turn 6162 Isolated Output off -- experiment will
abort.

CAUTION: DO NOT ABORT EXPERIMENT USING CONTROL C (^C).
THE DISK FILES WILL NOT BE SAVED. PROCEDURE IN IV.A
WILL SAVE OUTPUT FILES.

- B. Print output files, INTRMD.DAT, OUTPUT.DAT,
and/or RAWOUT.DAT with utility program LISTER.SAV.
(If Lister is used with a parallel printer, you
must enter ASSIGN LP: 7: before running program.)
- C. Execute normal shutdown (III).

Appendix C
AHEEDAS Program

This appendix contains the computer listings of the computer code used to implement the AHEEDAS. These modules correspond to those shown in Figures 12 and 13 in Chapter IV. Where needed for clarity, each module is preceded by a flow chart which outlines its functions. Modules A0 - A5 correspond to the Nodes A0 - A5 described in Chapter IV. Also shown are the device driver programs. The software modules IBUP1 and IBVDP1 (see Figure 13) are not shown. These programs were purchased from National Instruments (Ref 1 and 2). Several extensions to the RP-11 system library were also used. These were written by Mr. Frank E. Beital, University of Dayton Research Institute and are included in the AHEEDAS archives which are kept by the Air Force Materials Laboratory. Complete system listings and linking instructions can be obtained from Dr. Patrick M. Hemenger, AFML/LPO, Wright-Patterson AFB, Ohio 45433.

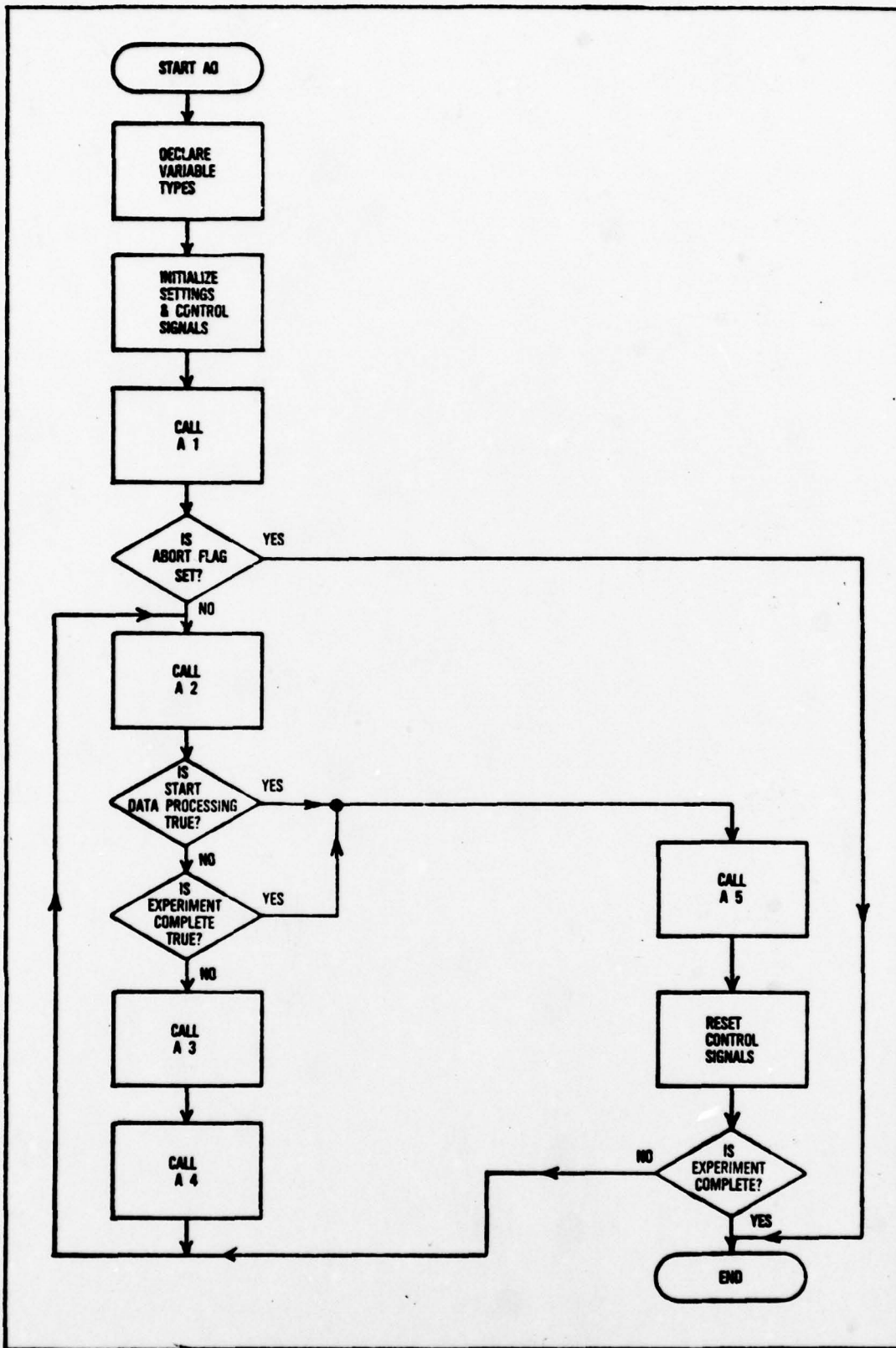


Figure 18. Flow Chart for Module A0


```

0022     REAL TEMSET,FLDSET,ULTSET,X0
C THESE ARE THE DATA SPECIFICATIONS FOR A3
C   A3COM
0023     INTEGER FDELAY,SAM,FLDOK,TEMOK,ULTOK,SAMOK,
        2UDELAY,SDELAY,TDELAY,DELAY
0024     REAL FLD,ULT,TEM
C   DMMCOM
0025     REAL CRNTRD
C   GAUSSM
0026     REAL FLDRD
C   DUMCOM
0027     INTEGER FUNC
0028     REAL TEMRD,ULTRD
C   ULTPWR
0029     REAL ULTAGE
C   TSTCOM
0030     INTEGER SIGN
C THESE ARE THE DATA SPECIFICATIONS FOR A4
C   RAWDAT
0031     REAL TEMDAT(20),ULTDAT(20),SUDATA(20),FLDATA(20),IDATA(20)
0032     INTEGER SCDATA(20)
C THESE ARE THE DATA SPECIFICATIONS FOR A5
C   DATOUT
0033     REAL RHO,P,MU,RH,F
0034     REAL TEMOUT(20),AVGTEM,DELTA,IATEM
0035     REAL R(4),R1R2,R3R4,R7,R8,UHALL,RMAG
0036     REAL R1,R2,R12,R5,R6,DELTAR,DELR5,DELR6
0037     REAL E,LN2,AFIELD,PI
C   PLOTER
0038     REAL PLOTS(4,2)
C THESE ARE THE COMMON BLOCKS FROM MODULE A1
C
0039     COMMON /HEADER/TITLE,ITEMP,TYPTM,TODAY
0040     COMMON /DATAIN/NTEMPT,TEMP,ETYPE,FIELD,
        2NAVOLT,AULT,NDATPT
0041     COMMON /TCALIB/THRMD,NTEMP,TEMCAL
0042     COMMON /SAMPLE/SAMID,SAMTYP,SAMT,SAMW,SAML
0043     COMMON /PLTOUT/PLOT,POPTS,NP,PLOTAB,PLOTOR
0044     COMMON /FILEIN/TNAME,DPNAME
0045     COMMON /EQPOUT/EOFLAG,EQUIPF,IEIOF
0046     COMMON /RELTIM/RTDATA
0047     COMMON /CONTRL/ABORT
C THE NAMED COMMON BLOCKS FOR A2CMN FOLLOW.
0048     COMMON /CONSIG/SDP,EXPC,FSCOM,SCSCOM,ULTCOM
0049     COMMON /A2COM/TEMSET,FLDSET,ULTSET,X0,NTEM,SCSET,SCOUNT,
        2UCOUNT,RUN
C THE COMMON BLOCKS FOR A3 FOLLOW
0050     COMMON /A3COM/FDELAY,UDELAY,SDELAY,TDELAY,DELAY,FLD,ULT,
        2TEM,FLDOK,TEMOK,ULTOK,SAMOK,SAM
0051     COMMON /DMMCOM/CRNTRD
0052     COMMON /GAUSSM/FLDRD
0053     COMMON /DUMCOM/FUNC,TEMRD,ULTRD
0054     COMMON /ULTPWR/ULTAGE

```

```

C THE COMMON BLOCKS FOR A4 FOLLOW
0055 COMMON /RANDAT/TEMDAT,ULTDAT,SUDATA,FLDATA,IDATA,SCDATA
C THE COMMON BLOCKS FOLLOW FOR A5
0056 COMMON /DATOUT/RHO,P,MU,RH,F,TEMOUT,AUGTEM,DELTA,IATEM,R,
2R1R2,R3R4,R7,R8,UHALL,RMAG,R1,R2,R12,R5,R6,DELTAR,DELR5,DELR6,
3E,LN2,AFIELD,PI
0057 COMMON /PLOTTER/PLOTS
C INITIALIZE ALL REQUIRED CONTROL SIGNALS AND DATA SETTINGS
C
0058 DATA TEMSET/0.0/
0059 DATA NTEM,SCSET,FLDSET,ULTSET,SCOUNT,UCOUNT/1.0,0.0,0.0,0.0/
0060 DATA FLD,ULT,SAM,TEM,RUN/0.0,0.0,0.0,0.0/
C NOW GO TO THE INITIALIZATION MODULE
C
0061 CALL A1
C
C
C CHECK IF ABORT FLAG IS SET
C
0062 IF<ABORT.EQ.1>GO TO 900
C
C NOW BEGIN THE EXPERIMENT---CALL DETERMINE SETTINGS
C
0064 20 CONTINUE
0065 CALL A2
C
C TEST TO SEE IF START DATA PROCESSING IS TRUE
C
0066 IF<SDP.EQ.1>GO TO 800
C
C
C TEST TO SEE IF EXPERIMENT IS COMPLETE
C
0068 IF<EXPC.EQ.1>GO TO 800
C
C CALL SET PARAMETERS
C
0070 CALL A3
C
C START DATA ACQUISITION
C
0071 CALL A4
C
C GO BACK TO A2
C
0072 GO TO 20
C
C CALL REDUCE DATA
0073 800 CONTINUE
0074 CALL A5
C RESET ALL CONTROL SIGNALS AND SETTINGS FOR THE NEXT RUN
0075 ULTSET=0.0
0076 FLDSET=0.0

```

```
0077          SCSET=0
0078          SDP=0
0079          RUN=0
0080          ULTCOM=0
0081          SCSCOM=0
0082          UCOUNT=0
0083          IF(ETYPE.EQ.1)SCOUNT=0
```

```
      C
      C IF EXPERIMENT IS COMPLETE EXIT
0085          IF(EXPC.EQ.1)GO TO 900
0087          GO TO 20
0088  900     CONTINUE
0089          STOP
0090          END
```

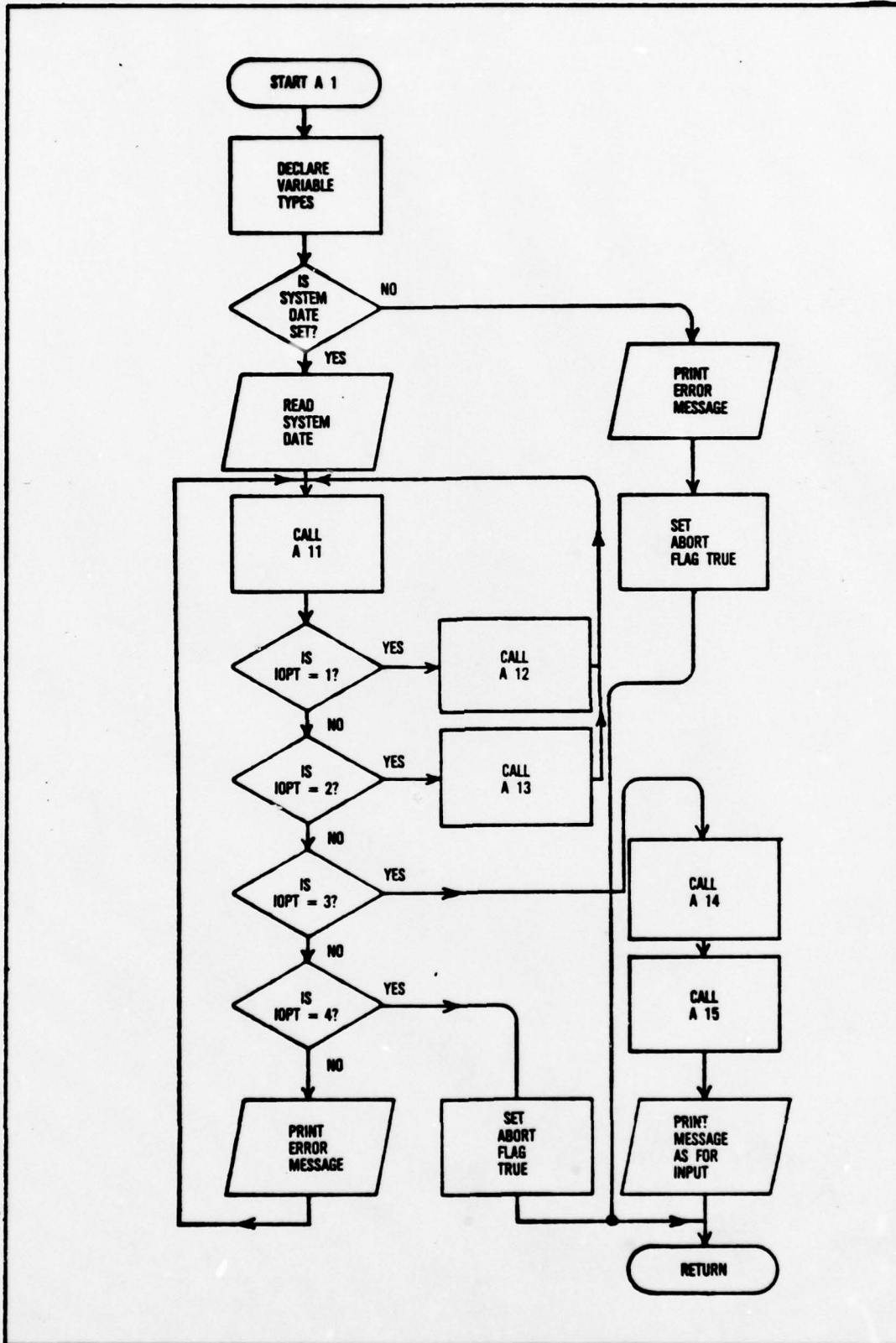


Figure 19. Flow Chart for Module A.

```

0001      SUBROUTINE A1
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C
          C   MODULE A1--INITIALIZE PARAMETERS
          C   A1---CONTROLS MODULES A11-A15
          C
          C   THIS MODULE CONTROLS THE INITIALIZATION MODULE.
          C   A USER RUNS THE PROGRAM AND IS REQUESTED BY A11 TO SUPPLY A
          C   NUMBER WHICH DIRECTS THE PROGRAM TO EXECUTE ONE OF FOUR
          C   OPTIONS:
          C
          C       1.  CREATE A DESIRED DATA POINTS FILE.
          C
          C       2.  CREATE A THERMOMETER CALIBRATION FILE.
          C
          C       3.  INITIALIZE PARAMETERS FOR AN EXPERIMENT.
          C
          C       4.  STOP EXECUTION OF THE PROGRAM.
          C
          C   AFTER CREATION OF EITHER A DESIRED DATA POINTS FILE OR A THERMOMETER
          C   CALIBRATION FILE THE PROGRAM RETURNS TO THE USER AND REQUESTS HIS
          C   NEXT OPTION.  AT THIS TIME HE MAY CREATE ANOTHER FILE
          C   OR BEGIN INITIALIZATION OF PARAMETERS FOR THE EXPERIMENT OR STOP.
          C   IF HE STOPS HIS FILES WILL REMAIN ON THE DISK HE SPECIFIED FOR USE
          C   AT A FUTURE TIME.  IF HE ELECTS TO INITIALIZE THE EXPERIMENTAL
          C   PARAMETERS THEN MODULES A14 AND A15 EXECUTE.  UPON COMPLETION
          C   THE PROGRAM HALTS UNTIL THE USER GIVES THE COMMAND TO BEGIN THE
          C   EXPERIMENT EXECUTION.  PRIOR TO ISSUING THIS SIGNAL THE USER
          C   SHOULD ENSURE THAT ALL OF THE EQUIPMENT IS TURNED ON AND
          C   READY TO GO.  IF NOT INFORMED VIA THE EQUIPMENT OUT FLAGS THE
          C   COMPUTER CANNOT TELL WHETHER THE INSTRUMENTS ARE ON LINE OR
          C   NOT.  ONCE TOLD TO PROCEED WITH THE EXPERIMENT
          C   A1 WILL PASS CONTROL TO A2--DETERMINE DATA PARAMETERS VIA THE
          C   COMMAND MODULE A0.  EXECUTION WILL THEN PROCEED TO EXPERIMENT
          C   COMPLETION
          C
          C               AUTHOR:  CAPTAIN EDGAR A. VERCHOT, JR., USAF
          C
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C
          C   THESE ARE THE DATA SPECIFICATIONS FROM MODULE A1
          C
          C   HEADER
          0002      BYTE TITLE(20), TODAY(9)
          0003      REAL ITEM(100)
          0004      INTEGER TYPTM
          C   DATAIN
          0005      REAL TEMP(100), FIELD, AULT(2,6)
          0006      INTEGER NTEMP, ETYPE, NAVOLT, NDATPT
          C   TCALIB

```



```

0033      CALL IDATE(MONTH, DAY, YEAR)
0034      IF (MONTH.EQ.0)GO TO 810
0036      CALL DATE(TODAY)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
0037  10   CONTINUE
0038      CALL A11
0039      IF(IOPT.NE.1)GO TO 20
0041      CALL A12
0042      GO TO 10
0043  20   CONTINUE
0044      IF(IOPT.NE.2)GO TO 30
0046      CALL A13
0047      GO TO 10
0048  30   CONTINUE
0049      IF(IOPT.NE.3)GO TO 40
0051      CALL A14
0052      CALL A15
0053      GO TO 50
0054  40   CONTINUE
0055      IF(IOPT.NE.4) GO TO 800
0057      GO TO 900
0058  50   CONTINUE
0059      WRITE(7,1010)
0060  1010  FORMAT( // 'INITIALIZATION IS COMPLETE'//
                2   ' ENTER A 1 WHEN READY TO START EXPERIMENT')
0061      READ(5,*) IGO
0062      GO TO 910
C ERROR MESSAGES
0063  800   CONTINUE
0064      WRITE(7,8000)
0065  8000  FORMAT( // 'ILLEGAL ENTRY, TRY AGAIN')
0066      GO TO 10
C
C ERROR MESSAGE FOR NO SYSTEM DATE SET
C
0067  810  CONTINUE
0068      WRITE(7,8010)
0069  8010  FORMAT( // ' YOU DID NOT SET THE SYSTEM DATE.'//
                2   ' THE PROGRAM WILL ABORT.'//
                3   ' SET THE DATE AND RESTART.')
0070      GO TO 900
0071  900  CONTINUE
C
C SET THE ABORT FLAG TO END THE PROGRAM
C
0072      ABORT=1
0073  910  CONTINUE
0074      RETURN
0075      END

```

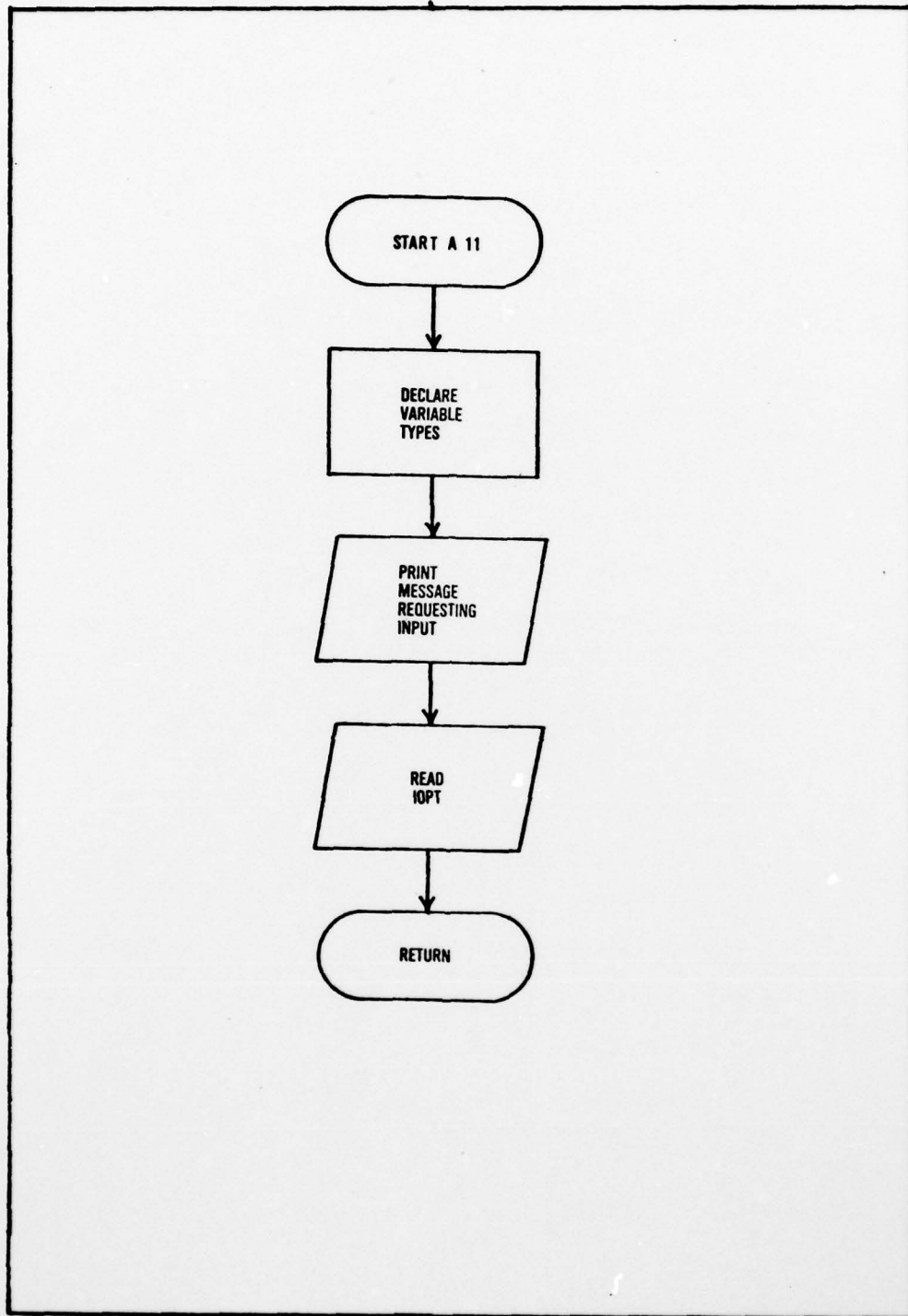


Figure 20. Flow Chart for Module A11

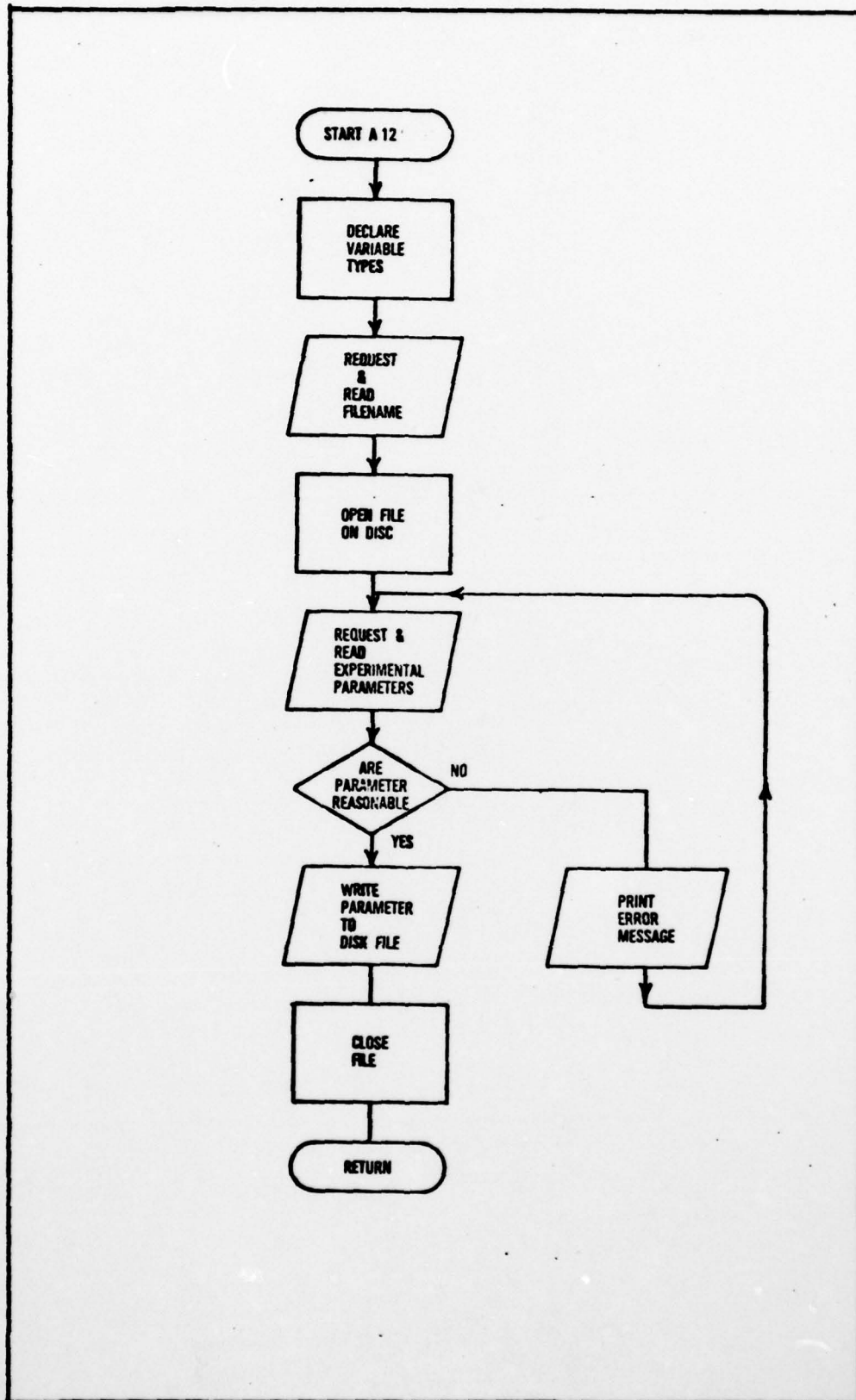


Figure 21. Flow Chart for Module A12

0001

SUBROUTINE A12

```
C
C THIS PROGRAM IS TO CREATE THE DESIRED INPUT DATA FILE
C
C CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C MODULE A12—CREATE DESIRED DATA POINTS FILES
C
C
C THIS MODULE IS AN INTERACTIVE ROUTINE WHICH GUIDES THE USER
C TO CREATE A FILE ON THE DISK TO CONTAIN THE DESIRED DATA
C PARAMETERS FOR AN EXPERIMENTAL RUN. THE USER MUST FIRST
C SUPPLY A FILENAME TO USE ON THE APPROPRIATE DISK. SINCE THESE
C FILES CAN BE RETAINED FOR REUSE FOR AN EXTENDED PERIOD OF TIME,
C DX0: IS THE PREFERRED DISK IF THERE IS STORAGE SPACE. OTHERWISE
C DX1: SHOULD BE USED. THE FORMAT TO BE ENTERED IS,
C DX1:UDPDAT.ONE.
C ANY APPROPRIATE NAME IN THIS FORMAT CAN BE USED. THE PROGRAM
C THEN REQUESTS THE USER TO ENTER A ONE LINE(80 CHARACTER) EXPERIMENT
C TITLE. THE EXPERIMENTOR IS THEN ASKED TO SPECIFY WHETHER HE
C WISHES TO USE TEMPERATURE POINTS OR INVERSE TEMPERATURE POINTS.
C FOLLOWING THIS THE NUMBER OF DESIRED TEMPERATURE DATA POINTS
C IS REQUESTED. THE DATA POINTS ARE ENTERED NEXT. THEY CAN BE
C ENTERED IN ONE OR MORE LINES BUT EACH ENTRY SHOULD BE FOLLOWED
C BY A COMMA EXCEPT THE LAST.
C UP TO 100 POINTS CAN BE SPECIFIED.
C THE EXPERIMENT TYPE IS REQUESTED NEXT. THIS PARAMETER SPECIFIES
C WHAT SAMPLE CONFIGURATIONS WILL BE USED:
C TYPE "1" DOES ALL OF THE VAN DER PAUW SAMPLE POSITIONS
C AT EACH TEMPERATURE(POSITIONS:1,2,3,4,5,6).
C TYPE "2" DOES SAMPLE POSITIONS(1,2,5)
C AT THE ODD TEMPERATURE POINTS AND THE REMAINING
C SAMPLE POSTIONS(3,4,6) AT THE
C EVEN TEMPERATURES.
C TYPE "3" DOES THE EXPERIMENT WITH THE HALL BAR SAMPLE.
C NEXT, THE APPLIED VOLTAGES ARE REQUESTED. UP TO SIX CHANGES OF APPLIED
C VOLTAGE CAN BE SPECIFIED. THE USER MUST SPECIFY THE TEMPERATURE AT
C WHICH THE APPLIED VOLTAGE IS TO BE CHANGED AND THE VALUE TO WHICH
C IT WILL BE CHANGED. FIRST THE NUMBER OF PAIRS OF VALUES WILL BE REQUESTED
C THEN THE PAIRS WILL BE REQUESTED INDIVIDUALLY.
C LAST THE NECESSARY FIELD VALUE IS NEEDED.
C ALL OF THESE VALUES ARE WRITTEN TO THE DISK FILE IN THE
C SAME FORMAT THAT IS ENTERED. ALL OF THESE INPUT/OUTPUT
C OPERATIONS ARE DONE IN FREE FORMAT.
C
C
C AUTHOR: CAPTAIN EDGAR A. VERCHOT, JR., USAF
C
C CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C THESE ARE THE DATA SPECIFICATIONS FROM MODULE A1
C
```

```

C HEADER
0002     BYTE TITLE(20),TODAY(9)
0003     REAL ITEMP(100)
0004     INTEGER TYPTM
C DATAIN
0005     REAL TEMP(100),FIELD,AULT(2,6)
0006     INTEGER NTEMPT,ETYPE,NAVOLT
C FILEIN
0007     BYTE TNAME(20),DPNAME(20)
C
C
C
C THESE ARE THE COMMON BLOCKS FROM MODULE A1
C
0008     COMMON /HEADER/TITLE,ITEMP,TYPTM,TODAY
0009     COMMON /DATAIN/NTEMPT,TEMP,ETYPE,FIELD,
        2NAVOLT,AULT
C
0010     BYTE ERRFLG
0011     INTEGER ERRET
C
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C
0012 10  WRITE(7,1000)
0013 1000 FORMAT( /,' ENTER FILE NAME IN FORMAT EX: DX1:UDPDAT.ONE  ', $)
0014     CALL GETSTR(5,DPNAME,19,ERRFLG)
C
0015     OPEN(UNIT=1,NAME=DPNAME,TYPE='NEW')
C
0016     WRITE(1,1210) TODAY
0017 1210 FORMAT( 9A1)
0018     WRITE(7,1002)
0019 1002 FORMAT( /,' ENTER EXPERIMENT TITLE IN 19 CHARACTERS'/' *', $)
0020     CALL GETSTR(5,TITLE,19,ERRFLG)
0021     WRITE(1,1003) (TITLE(I),I=1,LEN(TITLE))
0022 1003 FORMAT(80A1)
C
C CHECK TO SEE IF TEMPERATURE OR INVERSE TEMPERATURE IS USED
C
0023     ASSIGN 500 TO ERRET
0024 500  CONTINUE
0025     WRITE(7,1011)
0026 1011 FORMAT( /,' INPUT A "0" IF TEMPERATURE WILL BE USED'/' ,
        1' OR A "1" IF INVERSE TEMPERATURE WILL BE USED  ', $)
0027     READ(5,*) TYPTM
0028     WRITE(1,*) TYPTM
C CHECK FOR ERROR
0029     IF(TYPTM.NE.0.AND.TYPTM.NE.1)GO TO 900
C     NOW GET THE TEMPERATURE POINTS TO BE TAKEN
C

```

```

C FIRST GET THE NUMBER OF TEMPERATURES TO BE USED
C
0031 ASSIGN 510 TO ERRET
0032 510 CONTINUE
0033 WRITE(7,1005)
0034 1005 FORMAT( // ' ENTER NUMBER OF TEMPERATURE POINTS TO BE TAKEN '//
1' BETWEEN ONE AND ONE HUNDRED ' $)
0035 READ(5,*,END=20)NTEMP
0036 WRITE(1,*)NTEMP
0037 IF(TYPTEM.NE.0) GOTO 40
C CHECK FOR ERROR
0039 IF(NTEMP.LT.1.OR.NTEMP.GT.100)GO TO 910
C
C NOW GET THE VALUES OF TEMPERATURE THAT WILL BE USED
C
C
C
0041 WRITE(7,1020)
0042 1020 FORMAT( // ' ENTER THE DESIRED TEMPERATURES ' )
0043 READ(5,*,END=20)(TEMP(J),J=1,NTEMP)
C
0044 WRITE(1,*)(TEMP(J),J=1,NTEMP)
0045 GO TO 45
C?
C GET THE DESIRED INVERSE TEMPERATURE POINTS
C
C
C
0046 40 CONTINUE
0047 WRITE(7,1012)
0048 1012 FORMAT( // ' ENTER THE DESIRED INVERSE TEMPERATURES '//
1' SEPARATED BY COMMAS. USE UNITS OF 1000/TEMPERATURE ' )
0049 READ(5,*) (ITEMP(J),J=1,NTEMP)
C
0050 WRITE(1,*) (ITEMP(J),J=1,NTEMP)
0051 45 CONTINUE
C
C
C
C NOW GET THE EXPERIMENT TYPE
C
C
0052 ASSIGN 520 TO ERRET
0053 520 CONTINUE
0054 WRITE(7,1035)
0055 1035 FORMAT( // ' ENTER ZERO IF YOU NEED OPTIONS ON EXPERIMENT TYPE,
1 OTHERWISE ENTER 1 ',$)
0056 READ(5,*) K
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0057 IF(K.NE.0)GOTO 50
0059 WRITE(7,1040)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0060 1040 FORMAT(

```

```

1      / THE TYPE SPECIFIES THE SAMPLE CONFIGURATIONS',
1      / TO BE DONE.' //
1      / "1" CAUSES PLUS OR MINUS ONE THROUGH SIX TO BE' //
1      / DONE FOR EACH TEMPERATURE POINT.' //
1      / "2" CAUSES PLUS OR MINUS ONE, TWO, AND FIVE TO' //
1      / BE DONE AT ODD TEMPERATURE POINTS AND PLUS OR' //
1      / MINUS THREE, FOUR, AND SIX TO BE DONE AT EVEN' //
1      / TEMPERATURE POINTS.' //
1      / "3" CAUSES SAMPLE CONFIGURATIONS SEVEN AND EIGHT' //
1      / TO BE DONE.' //
0061  50  CONTINUE
0062      WRITE(7,1130)
0063  1130  FORMAT( /, ' ENTER THE EXPERIMENT TYPE      ', $)
0064      READ(5,*) ETYPE
0065      WRITE(1,*) ETYPE
C CHECK FOR ERROR
0066      IF(ETYPE.NE.1.AND.ETYPE.NE.2.AND.ETYPE.NE.3)GO TO 920
C
C NOW READ THE APPLIED VOLTAGE VALUES.
C
0068      ASSIGN 530 TO ERRET
0069  530  CONTINUE
0070      WRITE(7,1135)
0071  1135  FORMAT( /, ' THE APPLIED VOLTAGES ARE IN VOLTS' )
0072      WRITE(7,1140)
0073  1140  FORMAT( /, ' YOU MAY ENTER UP TO 6 PAIRS OF TEMPERATURES,APPLIED
1 VOLTAGES', /, ' THESE PAIRS WILL BE THE TEMPERATURES
1AT WHICH', /, ' THE APPLIED VOLTAGES WILL CHANGE TO
1 THE SPECIFIED VALUE' )
0074      WRITE(7,1150)
0075  1150  FORMAT( /, ' ENTER THE NUMBER OF TEMPERATURES,APPLIED VOLTAGES
1 PAIRS      ', $)
0076      READ(5,*) NAVOLT
0077      WRITE(1,*) NAVOLT
0078      DO 100 I=1,NAVOLT
0079          WRITE(7,1160)
0080  1160  1  FORMAT( /, ' ENTER THE TEMPERATURE,APPLIED VOLTAGE PAIRS', /, '
/ IN ASCENDING SEQUENTIAL ORDER BY TEMPERATURE VALUES
READ(5,*) AVLT(1,NAVOLT-I+1),AVLT(2,NAVOLT-I+1)
0081          WRITE(1,*) AVLT(1,NAVOLT-I+1),AVLT(2,NAVOLT-I+1)
0082
0083  100  CONTINUE
C
C
C
C
C NOW GET FIELD VALUES
C
C
0084      WRITE(7,1220)
0085  1220  FORMAT( /, ' ENTER DESIRED FIELD MAGNITUDE IN KGAUSS  ')
0086      READ(5,*) FIELD
0087      WRITE(1,*) FIELD
0088  20  CONTINUE

```

```

0089      CLOSE(UNIT=1)
      C
      C
0090      GO TO 999
      C ERROR MESSAGES TO FOLLOW
      C
      C ERROR FOR TYPE OF TEMPERATURE INCORRECT
      C
0091 900  CONTINUE
0092      WRITE(7,9000)
0093 9000 FORMAT(// VALUE MUST BE "0" OR "1"')
0094      GO TO ERRET
      C
      C ERROR FOR INCORRECT NUMBER OF TEMPERATURES
      C
0095 910  CONTINUE
0096      WRITE(7,9010)
0097 9010 FORMAT(// VALUE MUST BE GREATER THAN ZERO AND LESS
      1      THAN 100')
0098      GO TO ERRET
      C ERROR FOR EXPERIMENT TYPE
0099 920  CONTINUE
0100      WRITE(7,9020)
0101 9020 FORMAT(// ALLOWED VALUES ARE "1","2",OR "3"')
0102      GO TO ERRET
      C ERROR FOR NUMBER OF APPLIED VOLTAGE PAIRS
0103 930  CONTINUE
0104      WRITE(7,9030)
0105 9030 FORMAT(// VALUE MUST BE AN INTEGER BETWEEN "1" AND "6"')
0106      GO TO ERRET
0107 999  CONTINUE
0108      RETURN
0109      END

```

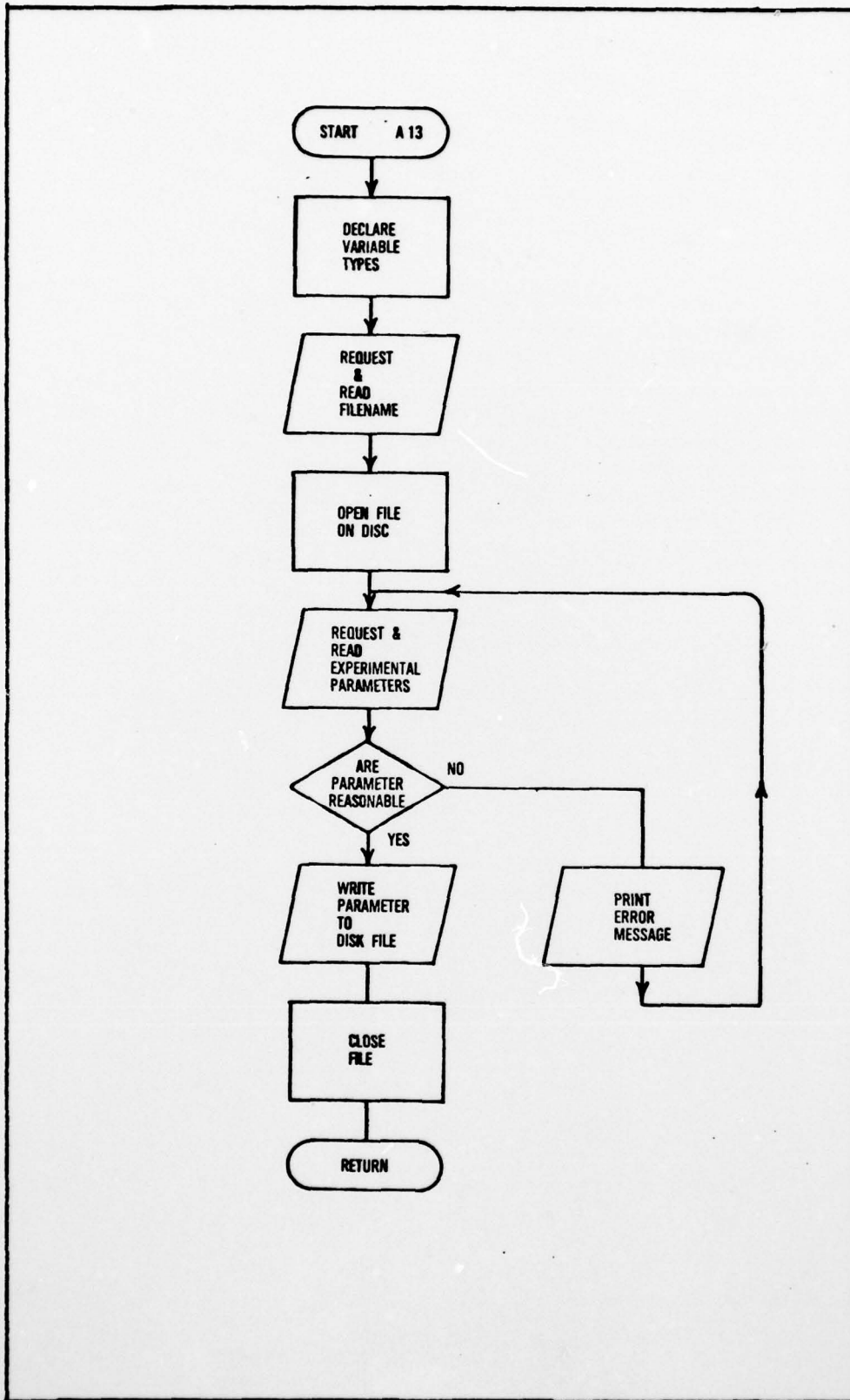


Figure 22. Flow Chart for Module A13

```

0001      SUBROUTINE A13
C
C THIS PROGRAM IS TO CREATE THE TEMPERATURE CALIBRATION FILES
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C A13-CREATE THERMOMETER CALIBRATION FILES
C
C THIS MODULE WILL INTERACTIVELY GUIDE THE EXPERIMENTOR
C TO CREATE A THERMOMETER CALIBRATION FILE ON THE DISK.
C HE WILL FIRST BE REQUESTED TO ENTER THE FILE NAME. THE FILE
C CAN BE STORED ON EITHER DX0: OR DX1:. DX0 IS PREFERRED IF THERE
C IS ROOM SO THAT DX1: IS LEFT FREE FOR ONLY THE OUTPUT DATA AND THUS
C CAN BE READILY REPLACED WITHOUT COPYING FILES. A THERMOMETER
C CALIBRATION FILE WOULD NORMALLY BE USED FOR THE LIFE OF THE
C THERMOMETER. THE NAME FORMAT IS DX0:THERMO.RED. THE THERMOMETER
C IDENTIFIER CAN BE ENTERED NEXT(UP TO 19 CHARACTERS). THEN THE
C NUMBER OF CALIBRATION POINTS TO BE USED(UP TO 100) IS ENTERED.
C THE CALIBRATION POINTS ARE ENTERED NEXT IN THE FORMAT,
C TEMPERATURE, VOLTAGE.
C EACH PAIR IS ENTERED INDIVIDUALLY TO ASSURE MINIMUM ERRORS.
C THE FILE IS WRITTEN ON DISK IN THE SAME FORMAT AS IT WAS ENTERED
C ON THE TERMINAL. ALL INPUT/OUTPUT IS FORMAT FREE.
C
C AUTHOR: CAPTAIN EDGAR A. VERCHOT, JR., USAF
C
C
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C THESE ARE THE DATA SPECIFICATIONS FROM MODULE A1
C
C HEADER
0002      BYTE TITLE(20),TODAY(9)
0003      REAL ITEMP(100)
0004      INTEGER TYPTEN
0005      BYTE ERRFLG
C TCALIB
0006      BYTE THRMID(20)
0007      INTEGER NTEMP
0008      REAL TEMCAL(2,100)
C FILEIN
0009      BYTE TNAME(20),DPNAME(20)
C
C RELTIM
0010      INTEGER RTDATA
C
C
C
C THESE ARE THE COMMON BLOCKS FROM MODULE A1
C
0011      COMMON /HEADER/TITLE,ITEMP,TYPTEN,TODAY
0012      COMMON /TCALIB/THRMID,NTEMP,TEMCAL

```

```

CC
C
C
C
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
0013      WRITE(7,1000)
0014 1000  FORMAT( ' ENTER THE FILE NAME IN FORMAT  DX1:THERMO.RED  ', $)
0015      CALL GETSTR(5,TNAME,19,ERRFLG)
C
0016      OPEN(UNIT=1,NAME=TNAME,TYPE='NEW')
C
C WRITE THE DATE TO THE TEMPEUTRE FILE
C
0017      WRITE(1,1900) TODAY
0018 1900  FORMAT( 9A1)
C
C
C
C NOW GET THE THERMOMETER IDENTIFIER
C
0019      WRITE(7,1010)
0020 1010  FORMAT( // ' ENTER THE THERMOMETER ID IN 19 CHARACTERS  ', $)
0021      CALL GETSTR(5,THRMID,19,ERRFLG)
0022      WRITE(1,2000) THRMID
0023 2000  FORMAT( 80A1)
C
C NOW GET THE NUMBER OF CALIBRATION POINTS TO BE USED
C
0024      WRITE(7,1020)
0025 1020  FORMAT( // ' ENTER THE NUMBER OF CALIBRATION POINTS
1 TO BE USED // ' MUST BE LESS THAN 100  ', $)
0026      READ(5,*) NTEMP
0027      WRITE(1,*) NTEMP
C
C GET THE CALIBRATION TABLE
C
C
0028      WRITE(7,1040)
0029 1040  FORMAT( // ' TEMPERATURES ARE TO BE ENTERED SEQUENTIALLY
1 // ' BEGINNING WITH THE LOWEST  ')
0030      DO 10 I=1,NTEMP
0031          WRITE(7,1030)
0032 1030  FORMAT( // ' ENTER THE PAIRS OF TEMPERATURES,
1 VOLTAGES  ', $)
0033          READ(5,*) TEMCAL(1,I),TEMCAL(2,I)
0034          WRITE(1,*) TEMCAL(1,I),TEMCAL(2,I)
0035 10    CONTINUE
0036      GOTO 20
0037 20    CONTINUE
0038      CLOSE(UNIT=1)
0039      RETURN

```

0040

END

AD-A080 175

AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOO--ETC F/8 9/2
AUTOMATED HALL EFFECT EXPERIMENT DATA ACQUISITION SYSTEM (AHEAD--ETC(U)
DEC 79 E A VERCHOT
AFIT/8EO/EE/790-5

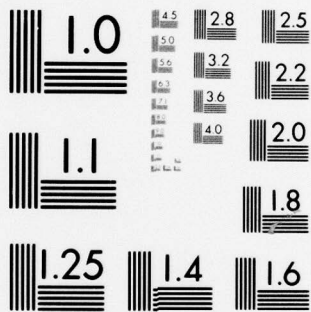
NL

2 OF 3

AD
A080175



The table consists of 72 individual microfilm frames arranged in a 6x12 grid. Each frame contains technical content, including circuit diagrams, flowcharts, and text. The diagrams are complex, showing various components and their interconnections. The text is dense and appears to be technical specifications or data related to the experiment. The frames are separated by thin white lines, and the overall appearance is that of a microfilm strip with a grid overlay.



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

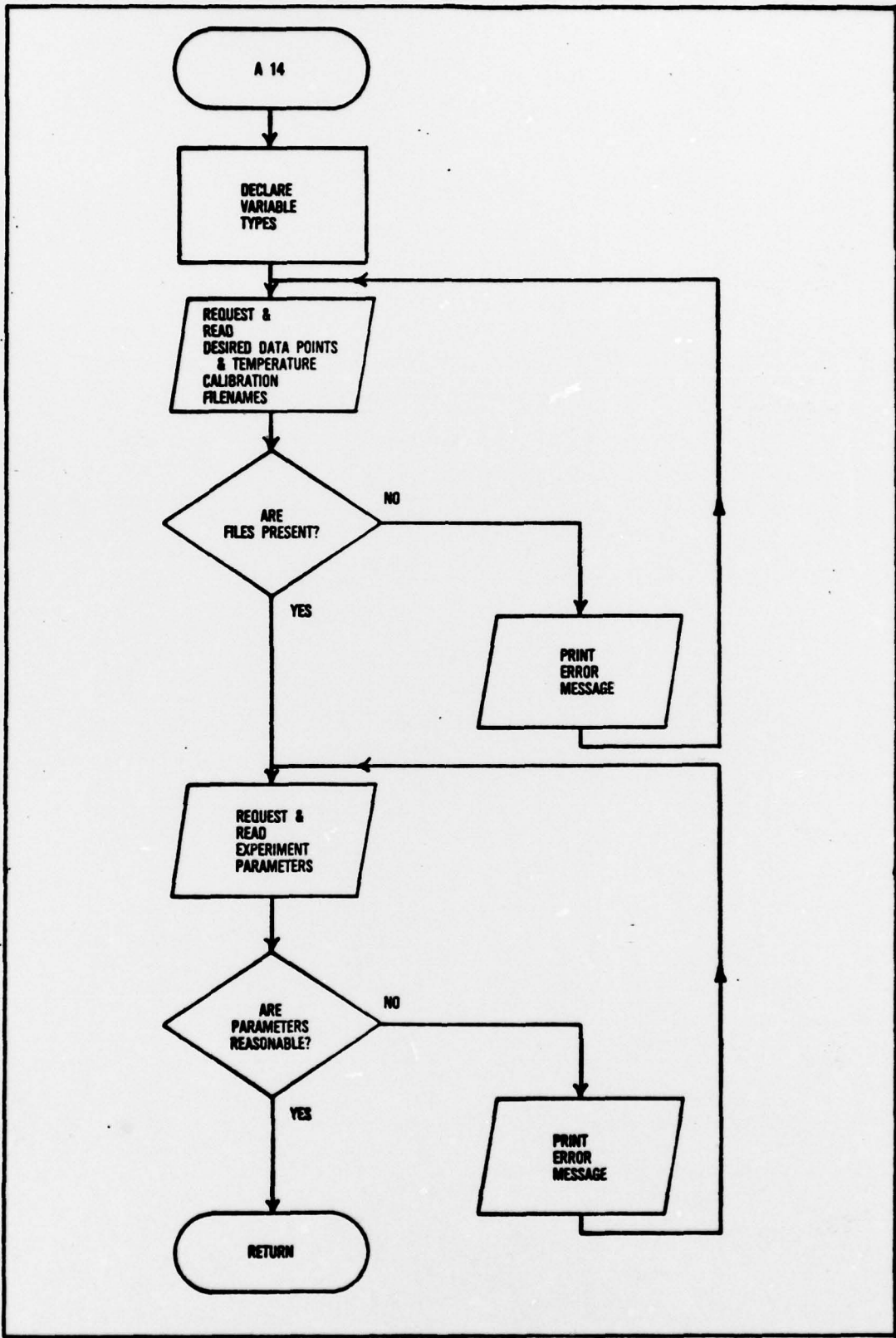


Figure 23. Flow Chart for Module A14


```

C
C
0031      ASSIGN 20 TO ERRET
0032  20   CONTINUE
0033      WRITE(7,1010)
0034  1010 FORMAT( ' ENTER DESIRED DATA POINTS FILENAME      ', $)
0035      CALL GETSTR(5,DPNAME,19,ERRFLG)

C
C
C   NOW CHECK TO SEE IF FILE EXISTS
C
C
0036      OPEN(UNIT=2,NAME=DPNAME,TYPE='OLD',READONLY,ERR=810)
0037      CLOSE(UNIT=2)

C
C
C   NOW GET THE SAMPLE ID
C
C
0038      WRITE(7,1020)
0039  1020 FORMAT( //, ' ENTER THE SAMPLE IDENTIFIER      ', $)
0040      CALL GETSTR(5,SAMID,19,ERRFLG)

C
C
C   GET SAMPLE TYPE
CC
C
0041      ASSIGN 30 TO ERRET
0042  30   CONTINUE
0043      WRITE(7,1030)
0044  1030 FORMAT( //, ' ENTER A "0" IF THE SAMPLE IS VAN DER PAUW' /
1          ' ENTER A "1" IF THE SAMPLE IS A HALL BAR' /
2          ' ENTER THE SAMPLE TYPE      ', $)
0045      READ(5,*) SAMTYP
0046      IF(SAMTYP.NE.0.AND.SAMTYP.NE.1) GOTO 830

C
C   GET THE SAMPLE DIMENSIONS
C
C
0048      WRITE(7,1040)
0049  1040 FORMAT( //, ' ENTER THE SAMPLE THICKNESS IN CENTIMETERS  ', $)
0050      READ(5,*) SAMT

C
C
0051      IF(SAMTYP.EQ.0)GOTO 40

C
C
0053      WRITE(7,1050)
0054  1050 FORMAT( //, ' ENTER THE SAMPLE LENGTH IN CENTIMETERS  ', $)
0055      READ(5,*) SAML

C
C
0056      WRITE(7,1060)

```

```

0057 1060 FORMAT( /, ' ENTER THE SAMPLE WIDTH IN CENTIMETERS ', $)
0058      READ(5,*) SAMW
      C
0059 40   CONTINUE
      C
      C
      C SET THE PLOT FLAG
      C
      C
0060      WRITE(7,1070)
0061 1070 FORMAT( /, ' DO YOU WANT ANY DATA PLOTTED IN REALTIME?' /
1         ' ENTER A "0" FOR NO OR A "1" FOR YES ', $)
0062      READ(5,*) PLOT
      C
0063      IF(PLOT.EQ.0)GOTO 50
      C
      C GET THE PARAMETERS TO BE PLOTTED
      C
      C
0065      ASSIGN 45 TO ERRET
0066 45   CONTINUE
0067      WRITE(7,1080)
0068 1080 FORMAT( /, ' YOU CAN PLOT UP TO FOUR DATA PAIRS IN REAL TIME.' /
1         ' THE OPTIONS ARE:' /
1         ' TEMPERATURE = TEMP' /
1         ' R1/R2 = R1/R2' /
1         ' R3/R4 = R3/R4' /
1         ' AVERAGE R1/R2= AR1/R2' /
1         ' INVERSE TEMPERATURE = 1/T' /
1         ' DELTA TEMPERATURE = DELTA' /
1         ' RESISTIVITY = RHO' /
1         ' MOBILITY = MU' /
1         ' CARRIER CONCENTRATION = P' /
1         ' HALL COEFFICIENT = RH' //
1         ' ENTER THE NUMBER OF DATA PAIRS DESIRED ', $)
      C
0069      READ(5,*) NP
      C
0070      IF(NP.LT.0.OR.NP.GT.4) GOTO 840
      C
      C
      C NOW GET THE PLOT PARAMETERS
      C
      C
0072      DO 60 K=1,NP
0073 80   CONTINUE
0074      WRITE(7,1100)
0075 1100 FORMAT( /, ' ENTER THE DATA PAIR IN FORMAT ABCISSA.ORDINATE' )
0076      CALL GETSTR(5,LINE,19,ERRFLG)
0077      I=INDEX(LINE,COMMA)
0078      IF(I.LE.1) GO TO 850
0080      LINE(I)=0

```

```

0081          J=I+1
0082          CALL STRPAD(LINE(J),6,ERRFLG)
0083          IORD1=INDEX(POPTS,LINE(J))
0084          CALL STRPAD(LINE(1),6,ERRFLG)
0085          IABC1=INDEX(POPTS,LINE(1))
C
0086          IF(IABC1.LT.1) GO TO 850
0088          IF(IORD1.LT.1) GO TO 850
C
0090          PLOTAB(K)=(IABC1+5)/6
0091          PLOTOR(K)=(IORD1+5)/6
0092          GOTO 60
0093 850      CONTINUE
0094          WRITE(7,8050)
0095 8050     FORMAT( //, ' INVALID DATA PAIR ENTERED')
0096          GO TO 80
C
0097 60      CONTINUE
0098 50      CONTINUE
C
C
C SET THE REAL TIME DATA ON/OFF FLAG
C
C
0099          WRITE(7,1130)
0100 1130    FORMAT( //, ' ENTER A "1" IF YOU WANT REAL TIME DATA PRINTOUT'
1, //, ' ENTER A "0" IF NOT ', $)
0101          READ(5,*) RTDATA
C
C
C SET THE EQUIPMENT INOPERATIVE FLAGS
C
C
0102          WRITE(7,1150)
0103 1150    FORMAT( //, ' DO YOU NEED TO SET ANY EQUIPMENT INOPERATIVE
1 FLAGS?', //, ' ENTER "1" FOR YES OR A "0" FOR NO ', $)
0104          READ(5,*) IEIOF
C
0105          IF(IEIOF.NE.1) GOTO 70
C
0107          WRITE(7,1160)
0108 1160    FORMAT( //, ' HOW MANY EQUIPMENT OUT FLAGS WILL YOU SET? ', $)
0109          READ(5,*) NFLAG
C
C
0110          WRITE(7,1170)
0111 1170    FORMAT( //, ' DMM OUT = "1" ' /
2          ' DUM OUT = "2" ' /
3          ' HEATER POWER SUPPLY OUT = "3" ' /
4          ' BIAS VOLTAGE POWER SUPPLY = "4" ' /
5          ' MAGNET CONTROLLER OUT = "5" ' /
6          ' TEST CONTROLLER OUT = "6" ' /
7          ' GAUSSMETER OUT = "7" ')

```

```

0112      WRITE(7,1180)
0113 1180  FORMAT( /, ' ENTER THE FLAGS SEPARATED BY COMMAS ', $)
0114      READ(5,*) (EFLAG(I), I=1, NFLAG)
      C
      C INITIALIZE FLAGS
      C
0115      DO 100 I=1,7
0116          EOFLAG(I)=0
0117 100   CONTINUE
      C
      C SET FLAGS
      C
0118      DO 90 I=1, NFLAG
0119          EOFLAG(EFLAG(I))=1
0120 90   CONTINUE
0121 70   CONTINUE
      C
0122      GO TO 1
      C
      C ERROR MESSAGES
      C
      C
      C
      C ERROR FOR INPUT FILE DOES NOT EXIST
      C
      C
0123 810  CONTINUE
0124      WRITE(7,8010)
0125 8010  FORMAT( /, ' FILENAME DOES NOT EXIST ON SYSTEM' )
0126      GO TO ERRET
      C
      C
      C
      C ERROR FOR SAMPLE TYPE ILLEGAL
      C
      C
0127 830  CONTINUE
0128      WRITE(7,8030)
0129 8030  FORMAT( /, ' SAMPLE TYPE MUST BE EITHER A "0" OR A "1"' )
0130      GO TO ERRET
      C
      C
      C ERROR FOR ILLEGAL NUMBER OF PLOT PAIRS ENTERED
      C
      C
0131 840  CONTINUE
0132      WRITE(7,8040)
0133 8040  FORMAT( /, ' NUMBER OF PAIRS CAN ONLY BE 1,2,3,OR 4' )
0134      GO TO ERRET
      C
0135 1    CONTINUE
0136      RETURN
0137      END

```

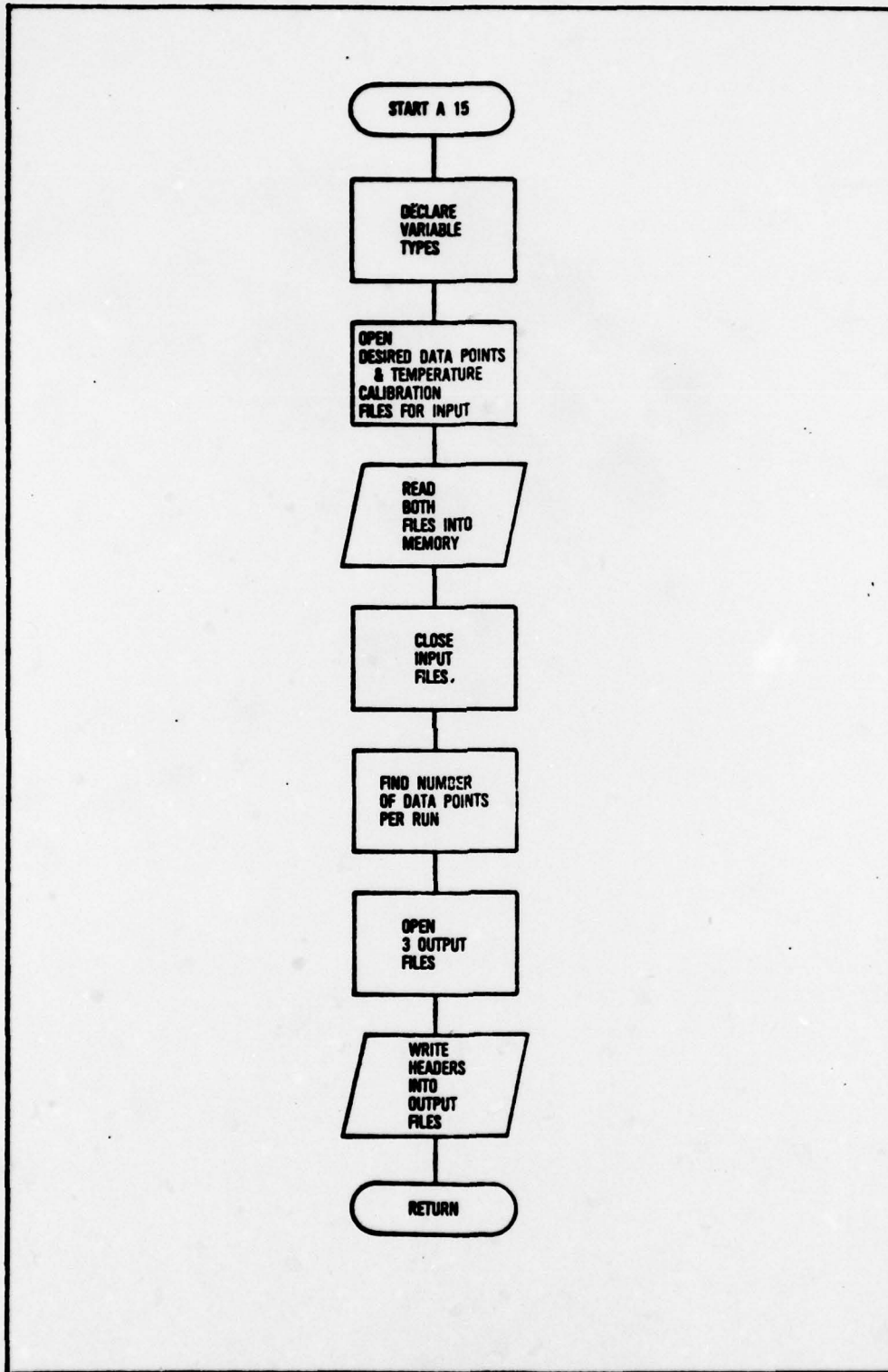


Figure 24. Flow Chart for Module A15

```

0001      SUBROUTINE A15
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C MODULE A15--SETUP FILES AND PARAMETERS
          C
          C
          C THIS MODULE OPENS THE DESIRED DATA
          C POINTS FILE AND THE THERMOMETER CALIBRATION FILE AND ENTERS THE
          C NEEDED DATA FROM THESE INTO THE APPROPRIATE ARRAYS IN THE PROGRAM.
          C IT THEN OPENS THE DISK FILES NEEDED FOR DATA OUTPUT AND INITIALIZES THEM.
          C PRIOR TO PROGRAM EXECUTION ANY FILES FROM PREVIOUS EXPERIMENTS
          C SHOULD BE RENAMED USING THE MONITOR COMMAND.
          C
          C RENAM FILESPEC: FILESPEC:
          C BECAUSE THE PROGRAM WILL DESTROY THEM WHEN THE NEW OUTPUT FILES ARE
          C CREATED. THE PROGRAM PAUSES WHEN A14 IS FINISHED UNTIL THE USER
          C GIVES THE COMMAND TO BEGIN THE EXPERIMENT. HE SHOULD DO THIS
          C ONLY WHEN HE IS CERTAIN THAT HE HAS TURNED ON AND INITIALIZED ALL
          C REQUIRED EQUIPMENT FOR THE EXPERIMENT.
          C
          C AUTHOR: CAPTAIN EDGAR A. VERCHOT, JR., USAF
          C
          C
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002      BYTE ERRFLG
          C THESE ARE THE DATA SPECIFICATIONS FROM MODULE A1
          C
          C HEADER
0003      BYTE TITLE(20),TODAY(9)
0004      REAL ITEM(100)
0005      INTEGER TYITEM
          C DATAIN
0006      REAL TEMP(100),FIELD,AULT(2,6)
0007      INTEGER NTEMP,ETYPE,NAVOLT,NDATPT
          C TCALIB
0008      BYTE THRMID(20)
0009      INTEGER NTEMP
0010      REAL TEMCAL(2,100)
          C SAMPLE
0011      BYTE SAMID(20)
0012      INTEGER SANTYP
0013      REAL SAMT,SAMW,SAML
          C PLTOUT
0014      BYTE POPTS(6,10)
0015      INTEGER PLOTAB(4),PLOTOR(4),PLOT,NP
          C FILEIN
0016      BYTE TNAME(20),DPNAME(20)
          C EQPOUT
0017      BYTE EQUIPF(10,8)
0018      INTEGER IEIOF,EQFLAG(7)
          C CONTRL
0019      INTEGER ABORT
          C
          C RELTIM

```

```

0020      INTEGER RTDATA
C
C
C
C THESE ARE THE COMMON BLOCKS FROM MODULE A1
C
0021      COMMON /HEADER/TITLE,ITEMP,TYPTM,TODAY
0022      COMMON /DATAIN/NTEMP,TEMP,ETYPE,FIELD,
2NAULT,AULT,NDATPT
0023      COMMON /TCALIB/THRMID,NTEMP,TEMCAL
CC
C
0024      COMMON /SAMPLE/SAMID,SANTYP,SAMT,SAMW,SAML
0025      COMMON /PLTOUT/PLOT,POPTS,NP,PLOTAB,PLOTOR
0026      COMMON /FILEIN/TNAME,DPNAME
0027      COMMON /EQPOUT/EOFLAG,EQUIFF,IEIOF
0028      COMMON /RELTIM/RTDATA
0029      COMMON /CONTRL/ABORT
C
C
C NOW OPEN THE INPUT FILES
C
0030      OPEN(UNIT=1,NAME=TNAME,TYPE='OLD',READONLY)
0031      OPEN(UNIT=2,NAME=DPNAME,TYPE='OLD',READONLY)
C
C NOW READ THE THERMOMETER CALIBRATION FILE INTO MEMORY
C
0032      CALL GETSTR(1,DATE,19,ERRFLG)
0033      CALL GETSTR(1,THRMID,19,ERRFLG)
C
C NOW READ THE CALIBRATION TABLE IN
C
C TEMCAL(1,I) CONTAINS THE TEMPERATURES AND
0034      READ(1,*) NTEMP
0035      DO 10 I=1,NTEMP
0036          READ(1,*) TEMCAL(1,I),TEMCAL(2,I)
0037  10  CONTINUE
C
0038      CLOSE(UNIT=1)
C
C READ THE DESIRED DATA POINTS FILE INTO MEMORY
C
0039      CALL GETSTR(2,DATE,19,ERRFLG)
0040      CALL GETSTR(2,TITLE,19,ERRFLG)
0041      READ(2,*) TYPTM
0042      READ(2,*) NTEMP
C
C TEST TO DETERMINE WHETHER TEMPERATURE OR INVERSE
C TEMPERATURE IS TO BE USED IN THE EXPERIMENT.
C THEN GET THE TEMPERATURE POINTS.
C
C
0043      IF(TYPTM.NE.0) GO TO 20

```

```

C
C READ TEMPERATURE POINTS AND CALCULATE INVERSE TEMPERATURES
C
0045 READ(2,*) (TEMP(I),I=1,NTEMP)
0046 DO 30 I=1,NTEMP
0047 ITEMP(I)=1000/TEMP(I)
0048 30 CONTINUE
0049 GO TO 100
C
C READ THE INVERSE TEMPERATURES AND CALCULATE THE TEMPERATURES.
C
0050 20 CONTINUE
0051 READ(2,*) (ITEMP(I),I=1,NTEMP)
0052 DO 40 I=1,NTEMP
0053 TEMP(I)=1000/ITEMP(I)
0054 40 CONTINUE
0055 100 CONTINUE
C
C GET THE EXPERIMENT TYPE
C
0056 READ(2,*) ETYPE
C
C NOW FIND OUT HOW MANY DATA POINTS THERE ARE
0057 IF(ETYPE.EQ.1)NDATPT=20
0059 IF(ETYPE.EQ.2)NDATPT=10
0061 IF(ETYPE.EQ.3)NDATPT=8
C
C GET THE NUMBER OF APPLIED VOLTAGE CHANGE POINTS
C
0063 READ(2,*) NAVOLT
C
C
C READ THE TEMPERATURE,APPLIED VOLTAGE PAIRS
C
C
0064 DO 50 I=1,NAVOLT
0065 READ(2,*) AVLT(1,I),AVLT(2,I)
0066 50 CONTINUE
C
C GET THE FIELD VALUES
C
C
0067 READ(2,*) FIELD
C
0068 CLOSE(UNIT=2)
C
C
C
C NOW SET UP THE OUTPUT FILE FOR THE DATA
0069 850 CONTINUE
C
C
0070 OPEN(UNIT=3,NAME='DX1:OUTPUT.DAT',TYPE='NEW',

```

```

0071      2FORM='UNFORMATTED', INITIALSIZE=8, ERR=800)
          OPEN(UNIT=1, NAME='DX1:RAWOUT.DAT', TYPE='NEW',
0072      2FORM='UNFORMATTED', INITIALSIZE=100, ERR=800)
          OPEN(UNIT=2, NAME='DX1:INTRND.DAT', TYPE='NEW',
          2FORM='UNFORMATTED', INITIALSIZE=8, ERR=800)
0073      GO TO 310
0074      800  CONTINUE
0075          WRITE(7,8000)
0076      8000  FORMAT(' ', ' THERE IS NOT ENOUGH ROOM ON DX1: FOR THE
2          OUTPUT FILES',/, ' PLACE A FRESH DISK IN THE DRIVE THEN',/,
3          ' INPUT AN INTEGER')
0077      ERRFLG=0
0078      READ(5,*) INTGER
0079      GO TO 850
0080      810  CONTINUE
          C      WRITE THE HEADERS ON THE OUTPUT FILES
0081          DO 5000 N=1,3
0082              WRITE(N) (TODAY(N),M=1,9)
0083              WRITE(N) (TITLE(J),J=1,19)
0084              WRITE(N) (THRNID(K),K=1,19)
0085              WRITE(N) (SAMID(L),L=1,20)
0086              WRITE(N) SANTYP
0087              IF(SANTYP.NE.0) GO TO 5010
0089              WRITE(N) SANT
0090              GO TO 5000
0091      5010  CONTINUE
0092              WRITE(N) SANT,SAMW,SAML
0093      5000  CONTINUE
          C
0094      RETURN
0095      END

```

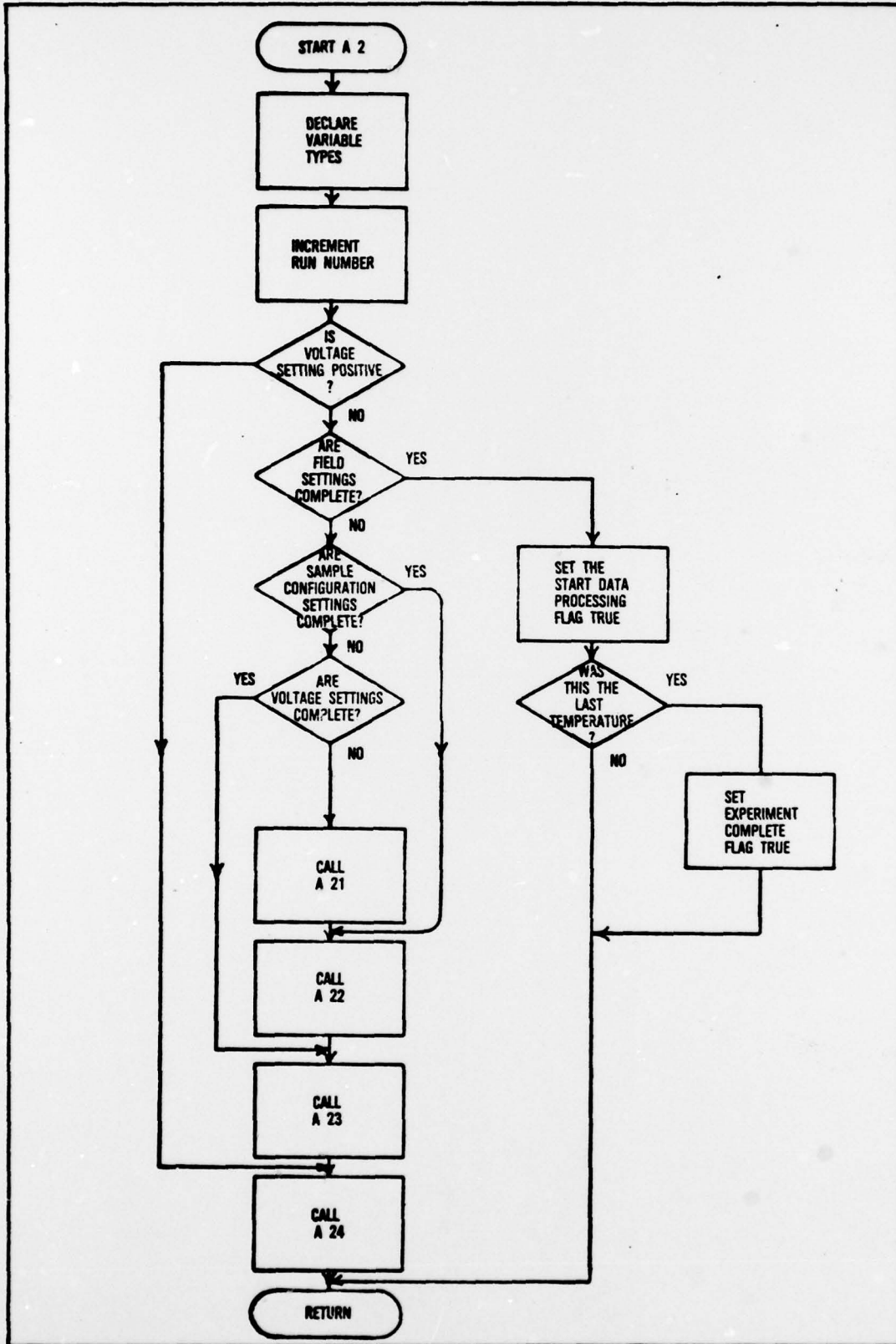


Figure 25. Flow Chart for Module A2

```

0001      SUBROUTINE A2
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C  MODULE A2---DETERMINE SETTING
C  THIS MODULE CONTROLS A21---A24.
C
C  THIS MODULE WILL DETERMINE THE APPROPRIATE
C  SETTINGS FOR ALL OF THE DATA PARAMETERS.
C  A2 IS THE EXECUTIVE MODULE. IT CHECKS ALL OF THE CONTROL FLAGS AND
C  CALLS THE APPROPRIATE PARAMETER SETTING MODULES. IT ALSO RESETS
C  ALL OF THE CONTROL FLAGS AS APPROPRIATE.
C    FOUR MODULES ARE UNDER THE CONTROL OF A2:
C      A21---DETERMINE TEMPERATURE SETTING
C      A22---DETERMINE FIELD SETTING
C      A23---DETERMINE SAMPLE CONFIGURATION SETTINGS
C      A24---DETERMINE APPLIED VOLTAGE SETTING.
C    AFTER EACH BLOCK OF DATA IS COMPLETE A2 GENERATES A START
C  DATA PROCESSING SIGNAL WHICH WILL SIGNAL MODULE A5,
C  REDUCE DATA, TO EXECUTE AS SOON AS DATA ACQUISITION COMPLETE IS RECEIVED.
C  AT THE APPROPRIATE TIME, A2 GENERATES THE EXPERIMENT COMPLETE SIGNAL
C  TO END THE EXPERIMENTAL RUN.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C  DATA SPECIFICATIONS
C
C  THESE ARE THE DATA SPECIFICATIONS FROM MODULE A1
C  DATAIN
0002      REAL TEMP(100),FIELD,AULT(2,6)
0003      INTEGER NTEMP,ETYPE,NAVOLT,NDATPT
C  TCALIB
0004      BYTE THRMID(20)
0005      INTEGER NTEMP
0006      REAL TEMCAL(2,100)
C  EQPOUT
0007      BYTE EQUIPF(10,8)
0008      INTEGER IEIOF,EOFLAG(7)
C  THESE ARE THE DATA SPECIFICATIONS FOR A2
C  CONSIG
0009      INTEGER SDP,EXPC,FSCOM,SCSCOM,ULTCOM
C
C  A2COM
0010      INTEGER NTEM,SCSET,SCOUNT,UCOUNT,RUN
0011      REAL TEMSET,FLDSET,ULTSET,X0
C  THESE ARE THE COMMON BLOCKS FROM MODULE A1
0012      COMMON /DATAIN/NTEMP,TEMP,ETYPE,FIELD,
        2NAVOLT,AULT,NDATPT
0013      COMMON /TCALIB/THRMID,NTEMP,TEMCAL
0014      COMMON /EQPOUT/EOFLAG,EQUIPF,IEIOF
C  THE NAMED COMMON BLOCKS FOR A2CMN FOLLOW.
0015      COMMON /CONSIG/SDP,EXPC,FSCOM,SCSCOM,ULTCOM
0016      COMMON /A2COM/TEMSET,FLDSET,ULTSET,X0,NTEM,SCSET,SCOUNT,
        2UCOUNT,RUN
C

```

```

C RESET SCOUNT
C
C
CC TEST THE CONTROL SIGNALS TO DETERMINE WHERE TO GO NEXT
0017 10 CONTINUE
0018 RUN=RUN+1
C IF THE OLD VOLTAGE SETTING IS POSITIVE GO DO THE NEGATIVE VOLTAGE
C WITH ALL THE OTHER PARAMETERS THE SAME.
0019 IF<ULTSET.GT.0.0>GO TO 24
C IF ALL FIELD SETTINGS ARE COMPLETE AT THIS TEMPERATURE SET THE CONTROL
C SIGNALS AND GO TO A5:REDUCE DATA.
0021 IF<FSCOM.EQ.1.AND.SCSET.EQ.5.AND.ETYPE.EQ.1>GO TO 11
0023 IF<FSCOM.EQ.1>GO TO 100
0025 11 CONTINUE
C IF ALL THE SAMPLE CONFIGURATIONS ARE COMPLETE AT THIS FIELD SETTING
C GO TO A NEW FIELD SETTING.
0026 IF<SCSCOM.EQ.1>GO TO 22
C IF ALL THE VOLTAGE SETTINGS ARE COMPLETE AT THIS SAMPLE
C CONFIGURATION SET A NEW CONFIGURATION.
0028 IF<ULTCOM.EQ.1>GO TO 23
0030 IF<VCOUNT.EQ.1>GO TO 24
C
C CALL THE TEMPERATURE SETTING MODULE
C
0032 21 CONTINUE
0033 CALL A21
C
C CALL THE FIELD SETTING MODULE
C
0034 22 CONTINUE
0035 SCSCOM=0
0036 CALL A22
C
C CALL THE SAMPLE CONFIGURATIONS SETTING MODULE
C
0037 23 CONTINUE
0038 ULTCOM=0
0039 CALL A23
C
C CALL THE APPLIED VOLTAGE SETTING MODULE
C
0040 24 CONTINUE
0041 CALL A24
0042 GO TO 900
C NOW GO TO THE REDUCE DATA MODULE TO REDUCE A BLOCK OF DATA
C
0043 100 CONTINUE
0044 RUN=RUN - 1
0045 FSCOM=0
0046 SDP=1
0047 IF<<NTEM-1>.EQ.NTEMPT>EXPC=1
0049 GO TO 900
C

```

0050
0051
0052

C
C
900

CONTINUE
RETURN
END

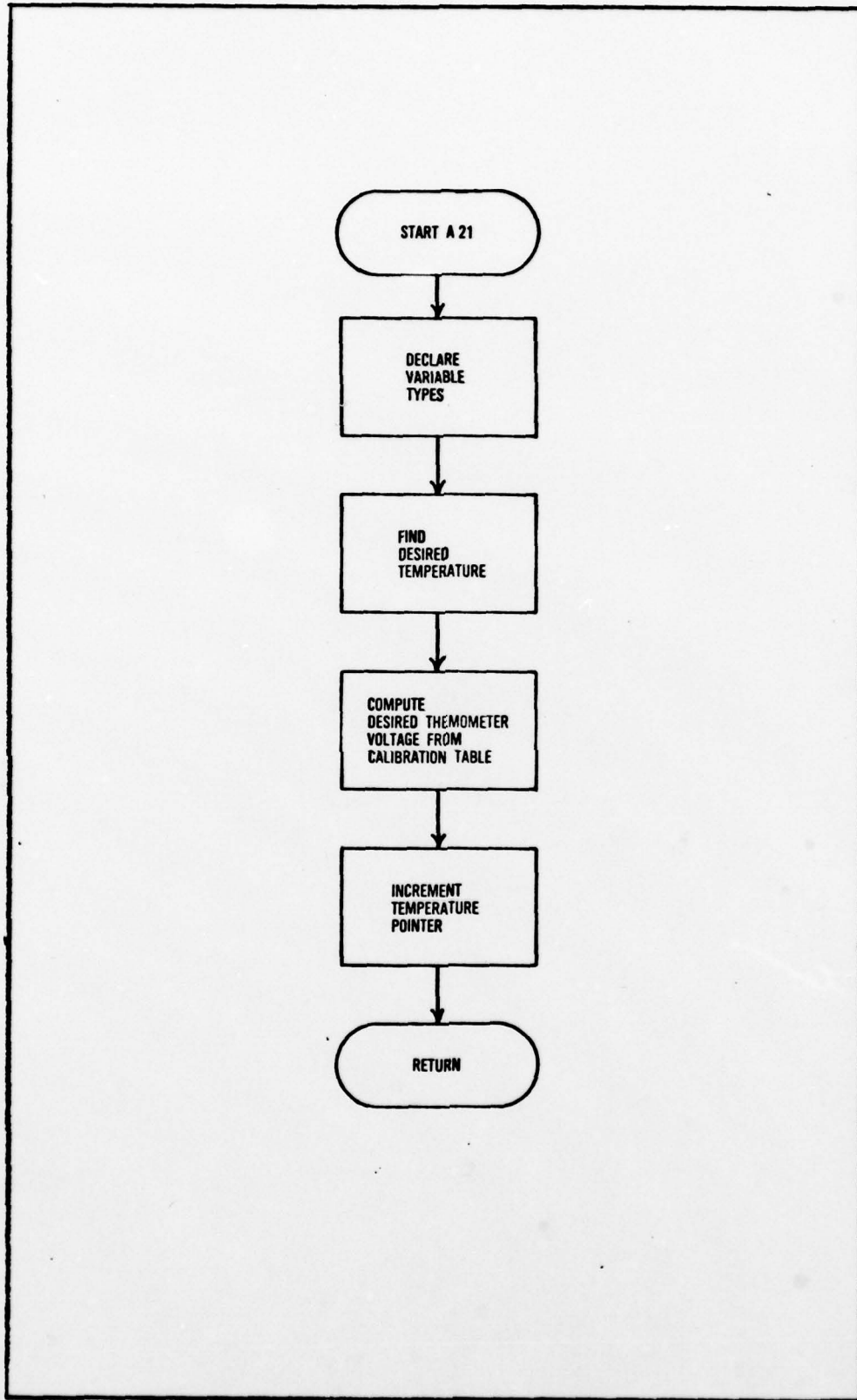


Figure 26. Flow Chart for Module A21


```

0021      IF(NS.LT.1)NS=1
0023      NF=I+3
0024      IF(NF.GT.NTEMP)NF=NTEMP

      C
      C
0026      Y0=0.0
0027      DO 30 I=NS,NF
0028          D(I)=1.0
0029          P(I)=1.0
0030          DO 40 J=NS,NF
0031              IF(I.EQ.J)GO TO 40
0033              D(I)=D(I)*(TEMCAL(1,I)-TEMCAL(1,J))
0034              P(I)=P(I)*(X0-TEMCAL(1,J))
0035 40          CONTINUE
0036          Y0=Y0+(P(I)/D(I))*TEMCAL(2,I)
0037 30      CONTINUE
0038      NTEM=NTEM+1
0039      TEMSET=Y0
0040      RETURN
0041      END

```

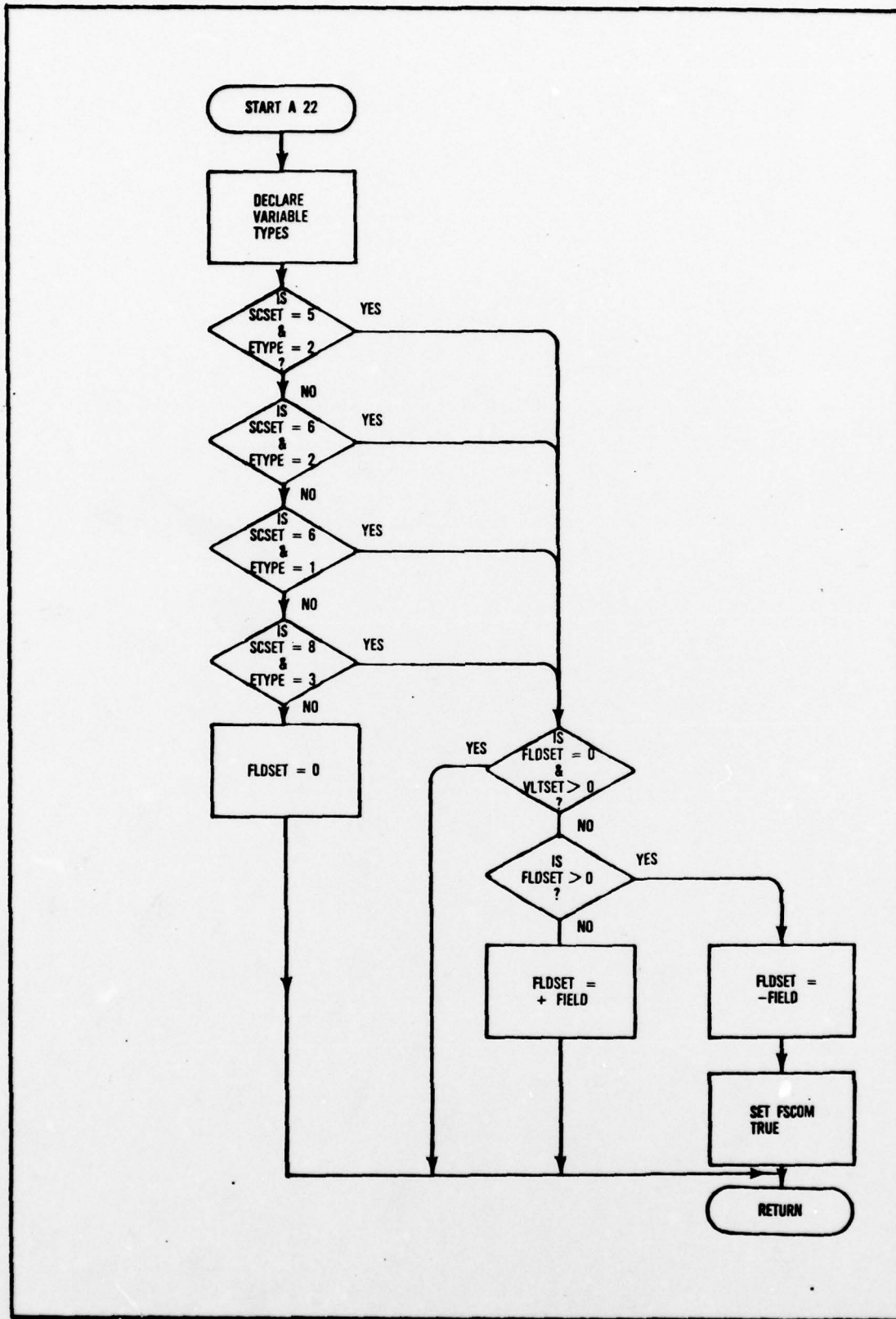


Figure 27. Flow Chart for Module A22
104


```
0023          IF<FLDSET.GT.0.0) GO TO 100
0025          FLDSET=FIELD
0026          GO TO 900
0027 100      CONTINUE
0028          FLDSET=-FIELD
0029          FSCOM=1
0030 900      CONTINUE
0031          RETURN
0032          END
```

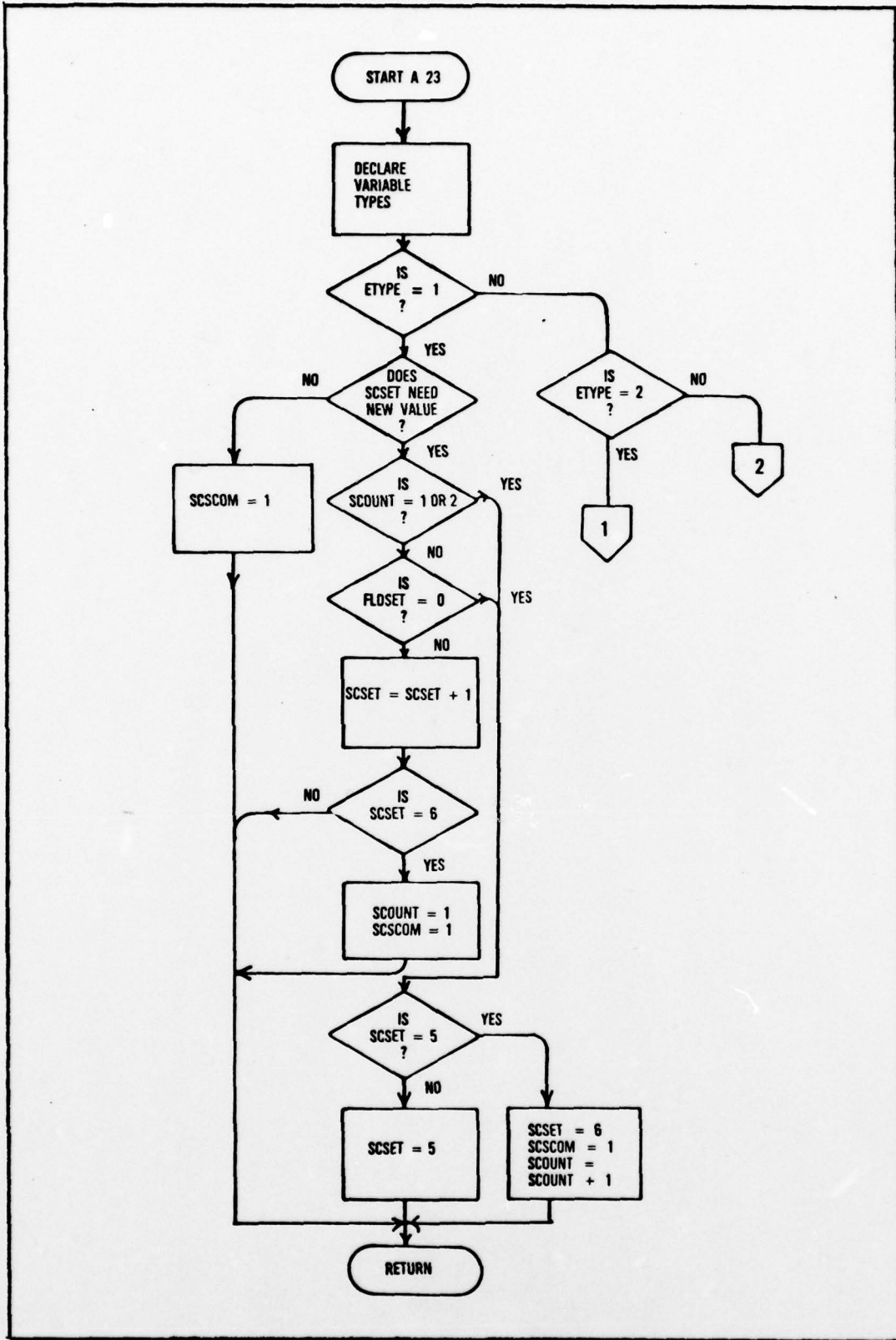


Figure 28. Flow Chart for Module A23

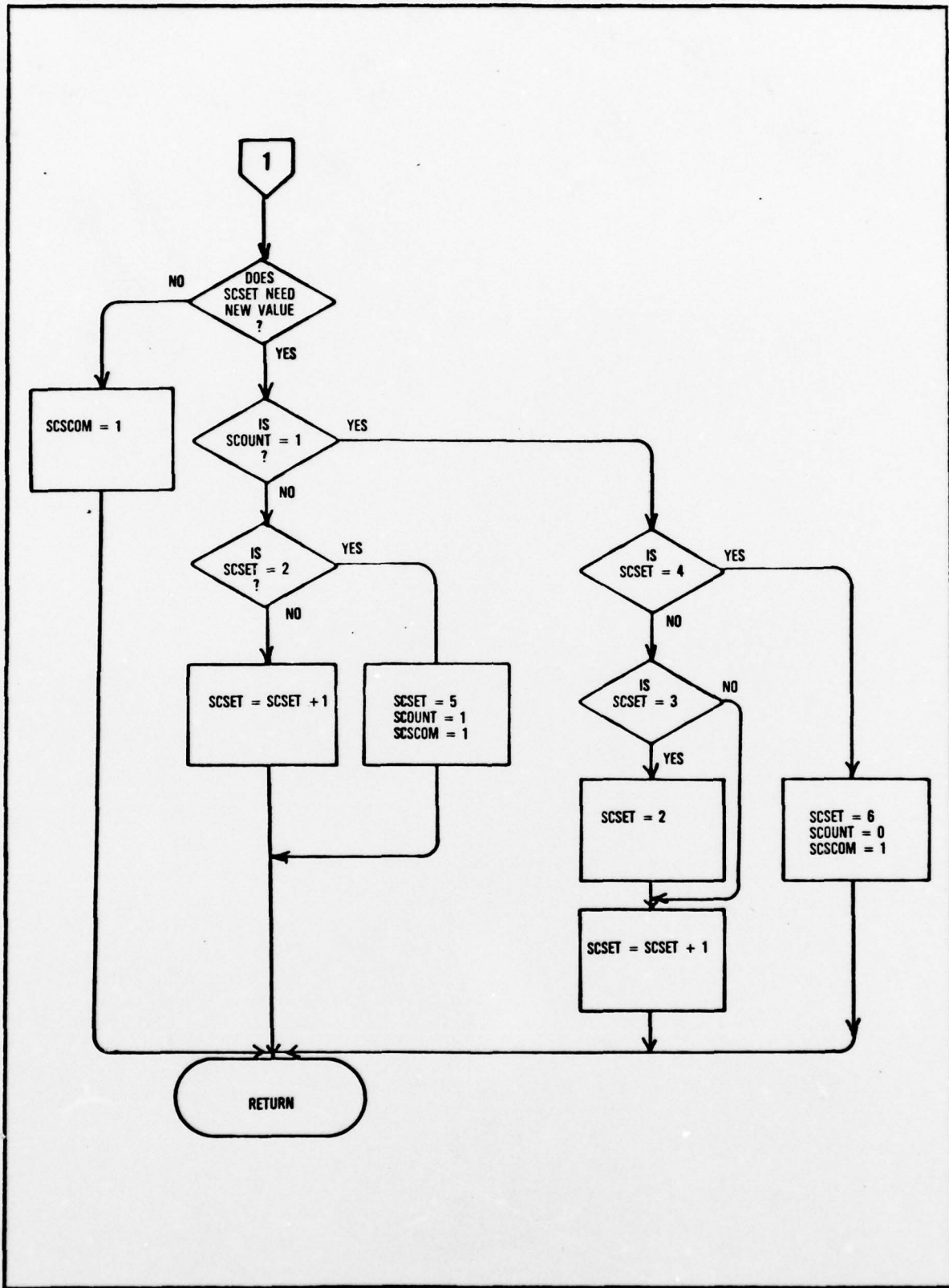


Figure 28-1. Flow Chart for Module A23

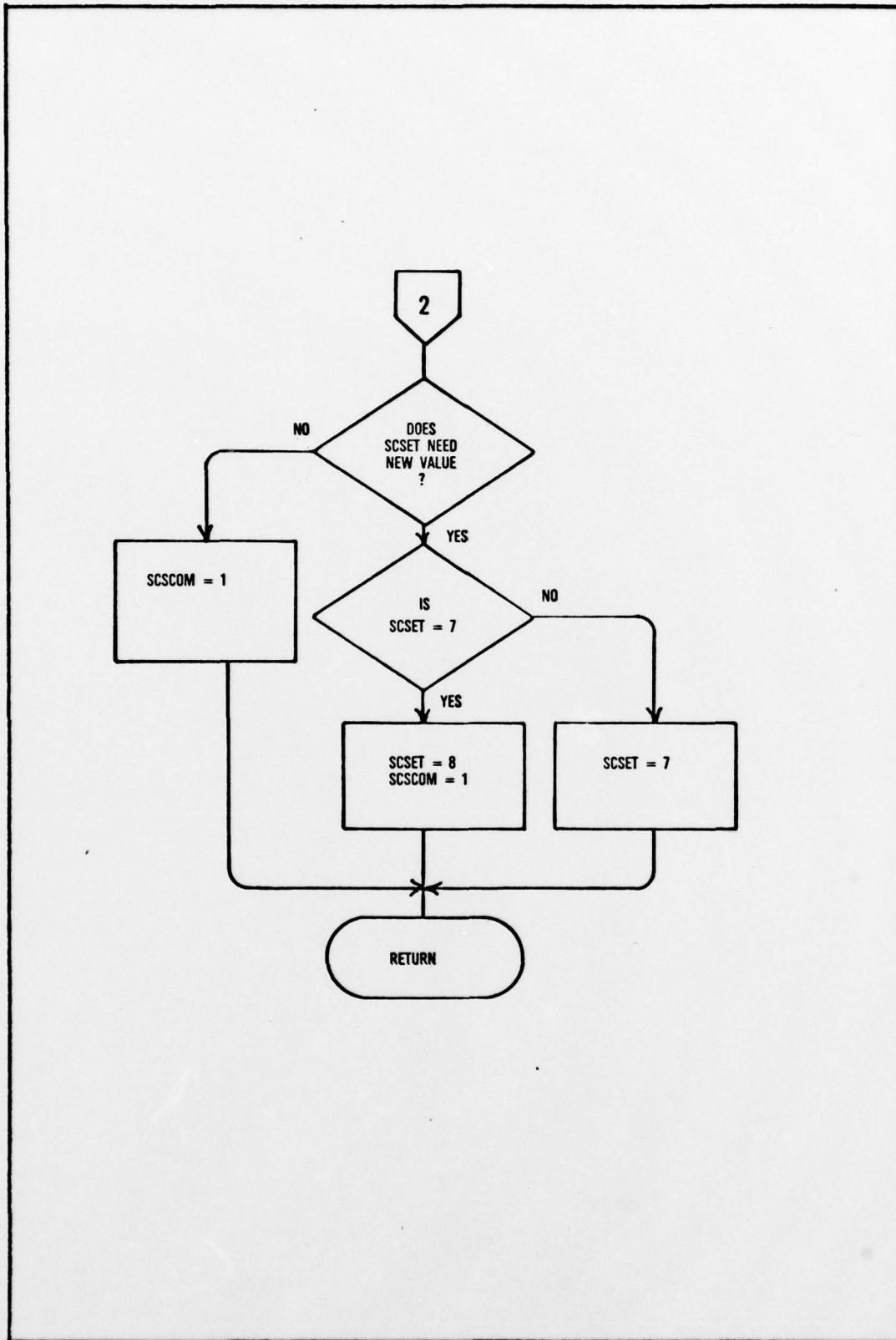


Figure 28-2. Flow Chart for Module A23

```

0001      SUBROUTINE A23
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C MODULE A23---DETERMINE SAMPLE CONFIGURATION SETTING
          C
          C     THIS MODULE WILL DETERMINE THE SAMPLE CONFIGURATION
          C SETTING.  THE EXPERIMENT PARAMETER DETERMINES HOW THE SAMPLE
          C CONFIGURATIONS WILL BE SET.  THE THREE CHOICES ARE,
          C
          C     ETYPE=1---SAMPLE CONFIGURATIONS 1,2,3,4,5,6 ARE SET IN TURN.
          C
          C     ETYPE=2---SAMPLE CONFIGURATIONS 1,2,5 ARE SET FOR THE FIRST AND
          C EACH SUCCEEDING ODD TEMPERATURE.
          C     ---SAMPLE CONFIGURATIONS 3,4,6 ARE SET FOR THE SECOND AND
          C EACH SUCCEEDING EVEN TEMPERATURE.
          C
          C     ETYPE=3---SAMPLE CONFIGURATIONS 7 AND 8 ARE SET.
          C
          C WHEN ALL SAMPLE CONFIGURATIONS ARE COMPLETE THE SCSCOM(SAMPLE
          C CONFIGURATIONS COMPLETE) SIGNAL IS SET TRUE.
          C
          C
          C     AUTHOR:  CAPTAIN EDGAR A. VERCHOT, JR., USAF
          C
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C DATA SPECIFICATIONS
          C
          C THESE ARE THE DATA SPECIFICATIONS FROM MODULE A1
          C DATAIN
0002      INTEGER ETYPE
0003      REAL TEMP(100),FIELD,AULT(2,6)
          C     THESE ARE THE DATA SPECIFICATIONS FOR A2
          C
          C     CONSIG
0004      INTEGER SDP,EXPC,FSCOM,SCSCOM,ULTCOM
          C
          C     A2COM
0005      INTEGER NTEM,SCSET,SCOUNT,UCOUNT,RUN
0006      REAL TEMSET,FLDSET,ULTSET,X0
          C THE NAMED COMMON BLOCKS FOR A2CMN FOLLOW.
0007      COMMON /CONSIG/SDP,EXPC,FSCOM,SCSCOM,ULTCOM
0008      COMMON /A2COM/TEMSET,FLDSET,ULTSET,X0,NTEM,SCSET,SCOUNT,
          C     ZUCOUNT,RUN
          C THESE ARE THE COMMON BLOCKS FROM MODULE A1
0009      COMMON /DATAIN/NTEMPT,TEMP,ETYPE,FIELD,
          C     ZNAVOLT,AULT,NDATPT
          C
          C
          C     RESET APPLIED VOLTAGES COMPLETE SIGNAL(ULTCOM)
          C
0010      DATA ULTCOM/0/

```

```

C
C TEST TO FIND THE EXPERIMENT TYPE
C
0011      IF(ETYPE.NE.1)GO TO 20
C          IF EXPERIMENT TYPE EQUALS 1.
0013      IF<FLDSET.EQ.0.0.AND.VLTSET.GT.0.0.AND.(SCSET.EQ.5.
20R.SCSET.EQ.6)>> GO TO 35
0015      IF<FLDSET.NE.0.0>GO TO 10
0017          IF<SCOUNT.EQ.1.OR.SCOUNT.EQ.2>GO TO 10
0019          SCSET=SCSET+1
0020          IF<SCSET.EQ.6>SCOUNT=1
0022          IF<SCSET.EQ.6>SCSCOM=1
0024          GO TO 900
C          NOW DO THE 5 AND 6 CONFIGURATIONS WHEN THE FIELD IS SET
0025 10      CONTINUE
0026          IF<SCSET.EQ.5>GO TO 11
0028          SCSET=5
0029          GO TO 900
0030 11      CONTINUE
0031          SCSET=6
0032          SCSCOM=1
0033          SCOUNT=SCOUNT+1
0034          IF<SCOUNT.GT.2>SCOUNT=2
0036          GO TO 900
C          IF ETYPE EQUALS 2
0037 20      CONTINUE
0038          IF(ETYPE.NE.2)GO TO 30
0040          IF<FLDSET.EQ.0.0.AND.VLTSET.GT.0.0.AND.(SCSET.EQ.5.
20R.SCSET.EQ.6)>> GO TO 35
0042          IF<FLDSET.NE.0.0>GO TO 35
0044          IF<SCOUNT.EQ.1>GO TO 32
C          DO THE ODD TEMPERATURE SAMPLE CONFIGURATIONS
0046          IF<SCSET.EQ.2>GO TO 31
0048          SCSET=SCSET+1
0049          GO TO 900
0050 31      CONTINUE
0051          SCSET=5
0052          SCOUNT=1
0053          SCSCOM=1
0054          GO TO 900
C          DO THE EVEN TEMPERATURE SAMPLE CONFIGURATIONS
0055 32      CONTINUE
0056          IF<SCSET.EQ.4>GO TO 33
0058          IF<SCSET.NE.3>SCSET=2
0060          SCSET=SCSET+1
0061          GO TO 900
0062 33      CONTINUE
0063          SCSET=6
0064          SCOUNT=0
0065          SCSCOM=1
0066          GO TO 900
0067 35      CONTINUE
C          SET THE SCSCOM SIGNAL TRUE .

```

```

C THIS WILL ALLOW THE EXECUTION OF THE CONFIGURATION WITH FIELD APPLIED
0068      SCSCOM=1
0069      GO TO 900
C
0070 30   IF ETYPE EQUALS 3
0071      CONTINUE
0072      IF<FLDSET.EQ.0.0.AND.VLTSET.GT.0.0.AND.SCSET.EQ.8> GO TO 35
0073      IF<FLDSET.NE.0.0>GO TO 40
0074      IF<SCSET.NE.7>GO TO 47
0075      SCSET=8
0076      SCSCOM=1
0077      GO TO 900
0078 47   CONTINUE
0079      SCSET=7
0080      GO TO 900
0081 40   CONTINUE
0082      SCSCOM=1
0083      GO TO 900
C
C
0086 900  CONTINUE
0087      RETURN
0088      END

```

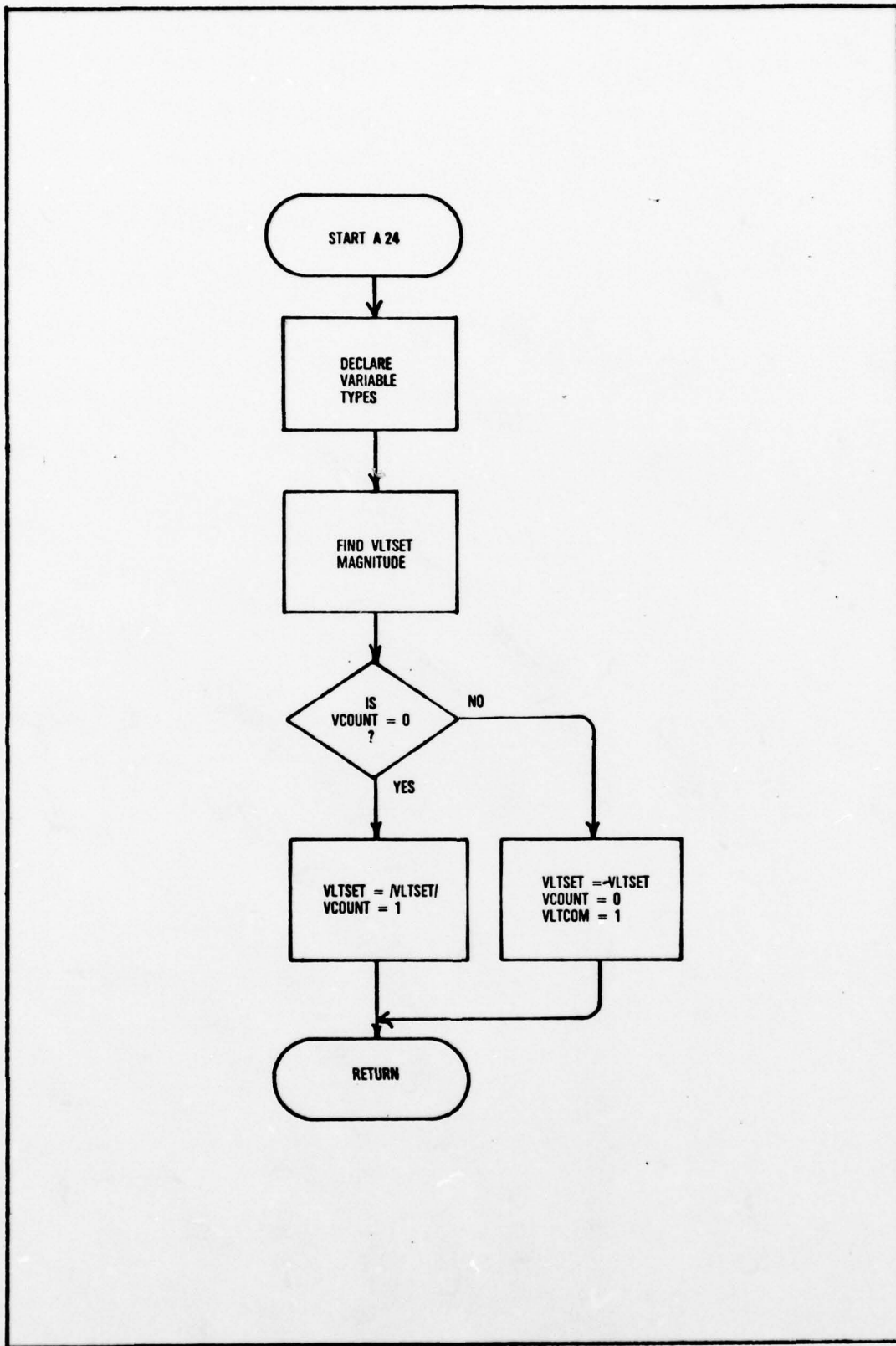


Figure 29. Flow Chart for Module A24

```

0001      SUBROUTINE A24
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C MODULE A24—DETERMINE APPLIED VOLTAGE SETTINGS
          C
          C      THIS MODULE WILL DETERMINE WHAT VOLTAGE WILL BE
          C APPLIED TO THE SAMPLE.  NORMALLY THIS WILL BE ONLY A POLARITY
          C CHANGE.  THE MAGNITUDE WILL ONLY BE CHANGED AT USER DESIGNATED
          C TEMPERATURE POINTS.
          C
          C
          C      AUTHOR:  CAPTAIN EDGAR A. VERCHOT, JR., USAF
          C
          C
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C DATA SPECIFICATIONS
          C THESE ARE THE DATA SPECIFICATIONS FROM MODULE A1
          C DATAIN
0002      INTEGER ETYPE
0003      REAL TEMP(100),FIELD,AULT(2,6)
          C
          C      THESE ARE THE DATA SPECIFICATIONS FOR A2
          C
          C      CONSIG
0004      INTEGER SDP,EXPC,FSCOM,SCSCOM,ULTCOM
          C
          C      A2COM
0005      INTEGER NTEM,SCSET,SCOUNT,UCOUNT,RUN
0006      REAL TEMSET,FLDSET,ULTSET,X0
          C THE NAMED COMMON BLOCKS FOR A2CMN FOLLOW.
0007      COMMON /CONSIG/SDP,EXPC,FSCOM,SCSCOM,ULTCOM
0008      COMMON /A2COM/TEMSET,FLDSET,ULTSET,X0,NTEM,SCSET,SCOUNT,
          C 2UCOUNT,RUN
          C THESE ARE THE COMMON BLOCKS FROM MODULE A1
0009      COMMON /DATAIN/NTEMP,TEMP,ETYPE,FIELD,
          C 2NAULT,AULT,NDATPT
          C
          C
          C
          C CHECK IF THE VOLTAGE MAGNITUDE NEEDS TO BE CHANGED
          C
0010      DO 10 I=1,NAULT
0011          IF(TEMP(NTEM-1).GE.AULT(1,I))ULTSET=AULT(2,I)
0013  10  CONTINUE
          C UCOUNT IS THE FLAG THAT TELLS THE PROGRAM WHETHER THE VOLTAGE SHOULD
          C BE POSITIVE OR NEGATIVE EACH PASS THROUGH THE PROGRAM.
          C
0014          IF(UCOUNT.NE.0) GO TO 20
0016          ULTSET=ABS(ULTSET)
0017          UCOUNT=1
0018          GO TO 900
0019  20  CONTINUE
0020          ULTSET=-ULTSET

```

```
0021          UCOUNT=0
0022          ULTCOM=1
0023 900      CONTINUE
0024          RETURN
0025          END
```

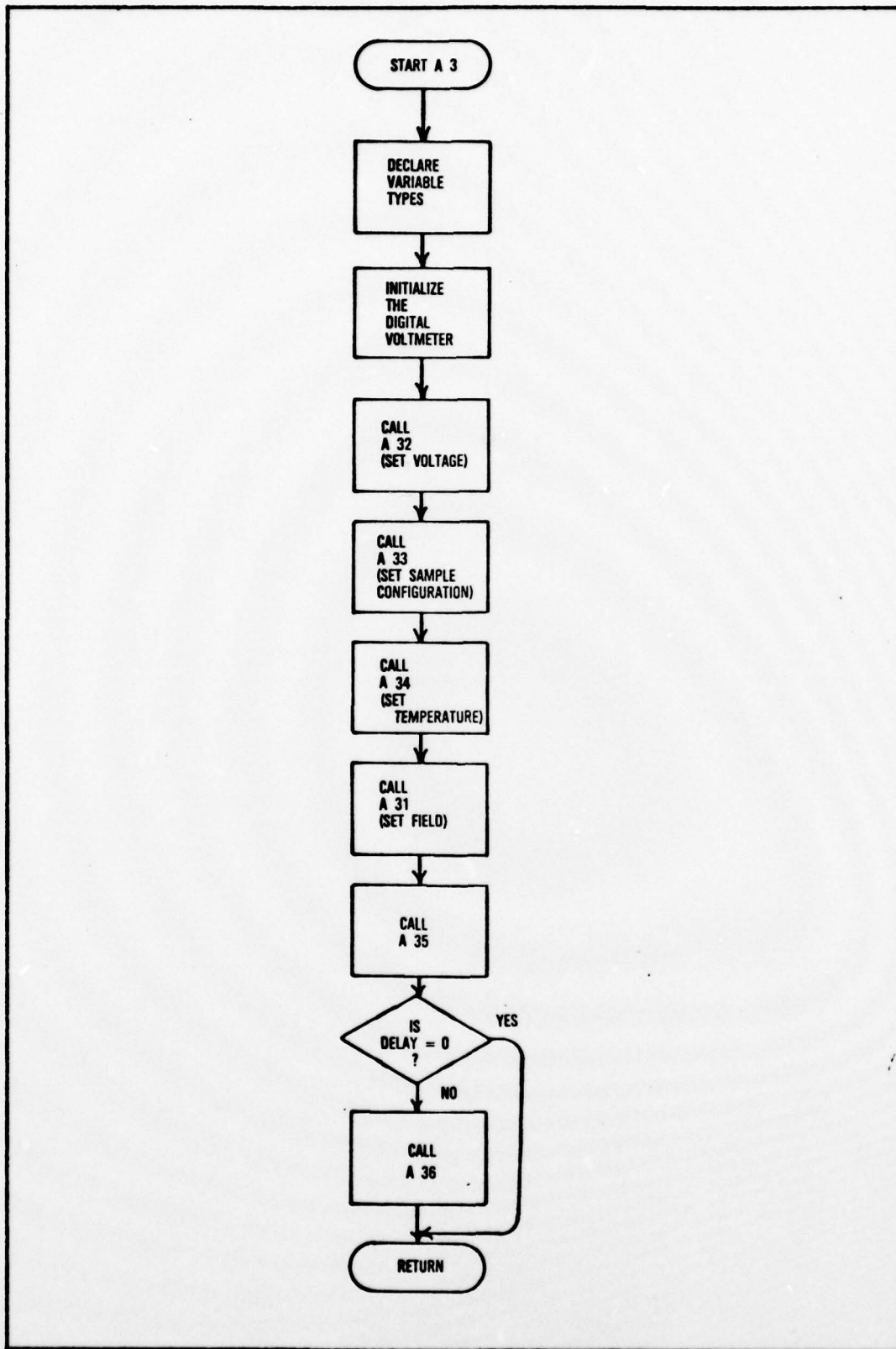


Figure 30. Flow Chart for Module A3

```

0001      SUBROUTINE A3
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C MODULE A3---SET PARAMETERS
          C
          C THIS MODULE IS THE EXECUTIVE FOR SET PARAMETERS. IT CONTROLS
          C WHICH MODULES OPERATE AND WHEN THEY DO SO. SIX MODULES ARE
          C SUBORDINATE TO A3:
          C A31---CHECK/SET FIELD
          C A32---CHECK/SET APPLIED VOLTAGE
          C A33---CHECK/SET SAMPLE CONFIGURATION
          C A34---CHECK/SET TEMPERATURE
          C A35---GENERATE ACQUIRE DATA SIGNAL
          C A36---DELAY RECHECK
          C AS THE MODULE EXECUTES, A31 THROUGH A34 SET THEIR PARAMETER
          C AND GENERATE EITHER A SETTling TIME DELAY OR A PARAMETER OK SIGNAL.
          C A31, CHECK/SET FIELD IS THE LAST TO SET ITS PARAMETER BECAUSE
          C IT WILL AFFECT THE OTHER PARAMETERS IF THE FIELD IS ON.
          C THE GENERATE ACQUIRE DATA SIGNAL MODULE THEN CHECKS ALL OF THE
          C PARAMETER OK SIGNAL IS GENERATED. IF NOT CONTROL PASSES BACK
          C TO DELAY RECHECK, WHICH PASSES CONTROL BACK TO THE TOP OF THE
          C CHECK/SET LOOP AFTER DELAYING FOR THE LONGEST OF THE ESTIMATED
          C SETTling TIMES THAT WERE GENERATED.
          C
          C
          C AUTHOR: CAPTAIN EDGAR A. VERCHOT, JR. USAF
          C
          C
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C DATA SPECIFICATIONS
          C
          C THESE ARE THE DATA SPECIFICATIONS FROM MODULE A1
          C EQPOUT
0002      BYTE EQUIPF(10,8)
0003      INTEGER IEIOF,EFLAG(7)
          C THESE ARE THE DATA SPECIFICATIONS FOR A2
          C A2COM
0004      INTEGER NTEM,SCSET,SCOUNT,UCOUNT,RUN
0005      REAL TEMSET,FLDSET,ULTSET,X0
          C THESE ARE THE DATA SPECIFICATIONS FOR A3
0006      INTEGER ULTAGE,TSTCON,MAGNET
          C A3COM
0007      INTEGER FDELAY,SAM,FLDOK,TEMOK,ULTOK,SAMOK,
          2UDELAY,SDELAY,TDELAY,DELAY
          C GAUSSM
0008      REAL FLDRD
          C DUMCOM
0009      INTEGER FUNC
0010      REAL TEMRD,ULTRD
          C TSTCOM
0011      INTEGER SIGN
          C THESE ARE THE COMMON BLOCKS FROM MODULE A1

```

```

0012      COMMON /EQPOUT/EOFLAG,EQUIPF,IEI0F
C THE NAMED COMMON BLOCKS FOR A2CMN FOLLOW.
0013      COMMON /A2COM/TEMSET,FLDSET,ULTSET,X0,NTEM,SCSET,SCOUNT,
          2UCOUNT,RUN
C THE COMMON BLOCKS FOR A3 FOLLOW
0014      COMMON /A3COM/FDELAY,UDELAY,SDELAY,TDELAY,DELAY,FLD,ULT,
          2TEM,FLDOK,TEMOK,ULTOK,SAMOK,SAM
0015      COMMON /DMMCOM/CRNTRD
0016      COMMON /GAUSSM/FLDRD
0017      COMMON /DUNCOM/FUNC,TEMRD,ULTRD
C
C
0018      DATA FLDOK,TEMOK,ULTOK,SAMOK/0,0,0,0/
0019      IF(EOFLAG(2).NE.0)GO TO 10
0021      A=DUM(0)
0022      10  CONTINUE
C START CHECKING THE PARAMETERS
C VOLTAGE
0023      CALL A32
C SAMPLE CONFIGURATION
0024      CALL A33
C TEMPERATURE
0025      40  CONTINUE
0026      CALL A34
C FIELD
0027      12  CONTINUE
0028      CALL A31
C NOW CHECK TO SEE OF ALL PARAMETERS ARE SET OK
C
0029      50  CONTINUE
0030      CALL A35
0031      IF(DELAY.EQ.0)GO TO 900
C DELAY RECHECK
0033      CALL A36(DELAY)
0034      900  CONTINUE
0035      RETURN
0036      END

```

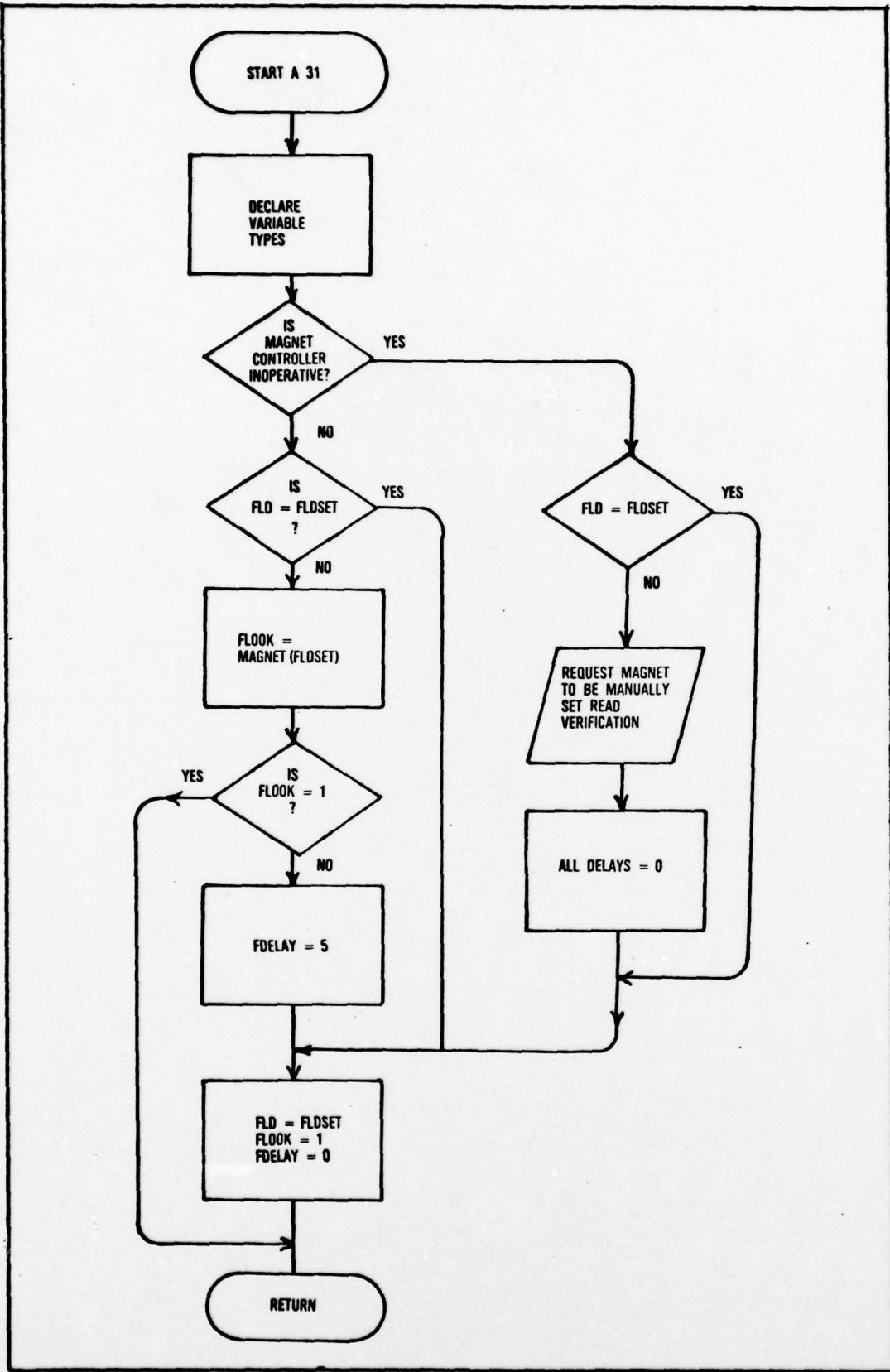


Figure 31. Flow Chart for Module A31

```

0001      SUBROUTINE A31
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C MODULE A31---CHECK/SET FIELD
          C
          C THIS MODULE PASSES THE FIELD SETTING TO THE STEPPER
          C MOTOR DRIVER PROGRAM WHICH CONTROLS THE SETTING OF
          C THE MAGNET. IT RECEIVES BACK AN ANTICIPATED DELAY AND A
          C FIELD OK SIGNAL. IF THE GAUSSMETER OR THE MAGNET CONTROL
          C EQUIPMENT OUT FLAG IS SET INPUT FROM THE TERMINAL IS
          C REQUIRED TO GET THE PROPER SETTING.
          C
          C
          C      AUTHOR: CAPTAIN EDGAR A. VERCHOT, JR. USAF
          C
          C
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C DATA SPECIFICATIONS
          C
          C THESE ARE THE DATA SPECIFICATIONS FROM MODULE A1
          C EQPOUT
0002      BYTE EQUIPF(10,8)
0003      INTEGER IEIOF,EOFLAG(7)
          C THESE ARE THE DATA SPECIFICATIONS FOR A2
          C A2COM
0004      INTEGER NTEM,SCSET,SCOUNT,UCOUNT,RUN
0005      REAL TEMSET,FLDSET,ULTSET,%0
          C THESE ARE THE DATA SPECIFICATIONS FOR A3
0006      INTEGER ULTAGE,TSTCON,MAGNET
          C A3COM
0007      INTEGER FDELAY,SAM,FLDOK,TEMOK,ULTOK,SAMOK,
          2UDELAY,SDELAY,TDELAY,DELAY
          C GAUSSM
0008      REAL FLDRD
          C THESE ARE THE COMMON BLOCKS FROM MODULE A1
0009      COMMON /EQPOUT/EOFLAG,EQUIPF,IEIOF
          C THE NAMED COMMON BLOCKS FOR A2CMN FOLLOW.
0010      COMMON /A2COM/TEMSET,FLDSET,ULTSET,%0,NTEM,SCSET,SCOUNT,
          2UCOUNT,RUN
          C THE COMMON BLOCKS FOR A3 FOLLOW
0011      COMMON /A3COM/FDELAY,UDELAY,SDELAY,TDELAY,DELAY,FLD,ULT,
          2TEM,FLDOK,TEMOK,ULTOK,SAMOK,SAM
0012      COMMON /GAUSSM/FLDRD
          C
          C CHECK EQUIPMENT OUT FLAGS
0013      IF(EOFLAG(5),EQ.1) GO TO 10
          C SET THE FIELD AUTOMATICALLY BY CALLING MAGNET
          C IF THE PREVIOUS SETTING AND THE NEW SETTING ARE THE SAME DON'T
          C BOTHER TO CALL MAGNET.
0015      IF(FLD,EQ.FLDSET)GO TO 20
0017      FLDOK=MAGNET(FLDSET,FLD)
          C DELAY IS EQUAL TO 2 SECONDS PER KGAUSS SETTING

```

```

0018      IF<FLDOK.NE.1>FDELAY=5
0020      GO TO 21
0021 10     CONTINUE
0022      IF<FLD.EQ.FLDSET> GO TO 20
0024      WRITE<7,1000> FLDSET
0025 1000  1     FORMAT<' ', ' SET THE MAGNET TO ',G14.7,' KGAUSS '/
           1     ' CARRIAGE RETURN WHEN SET ', $)
0026      PAUSE
C IF FIELD IS MANUAL ZERO ALL OF THE DELAYS. THEY WILL NOT
C BE NEEDED
0027      FDELAY=0
0028      SDELAY=0
0029      TDELAY=0
0030      UDELAY=0
0031 20     CONTINUE
0032      FLDOK=1
0033      FDELAY=0
0034 21     CONTINUE
0035      FLD=FLDSET
0036 900    CONTINUE
0037      RETURN
0038      END

```

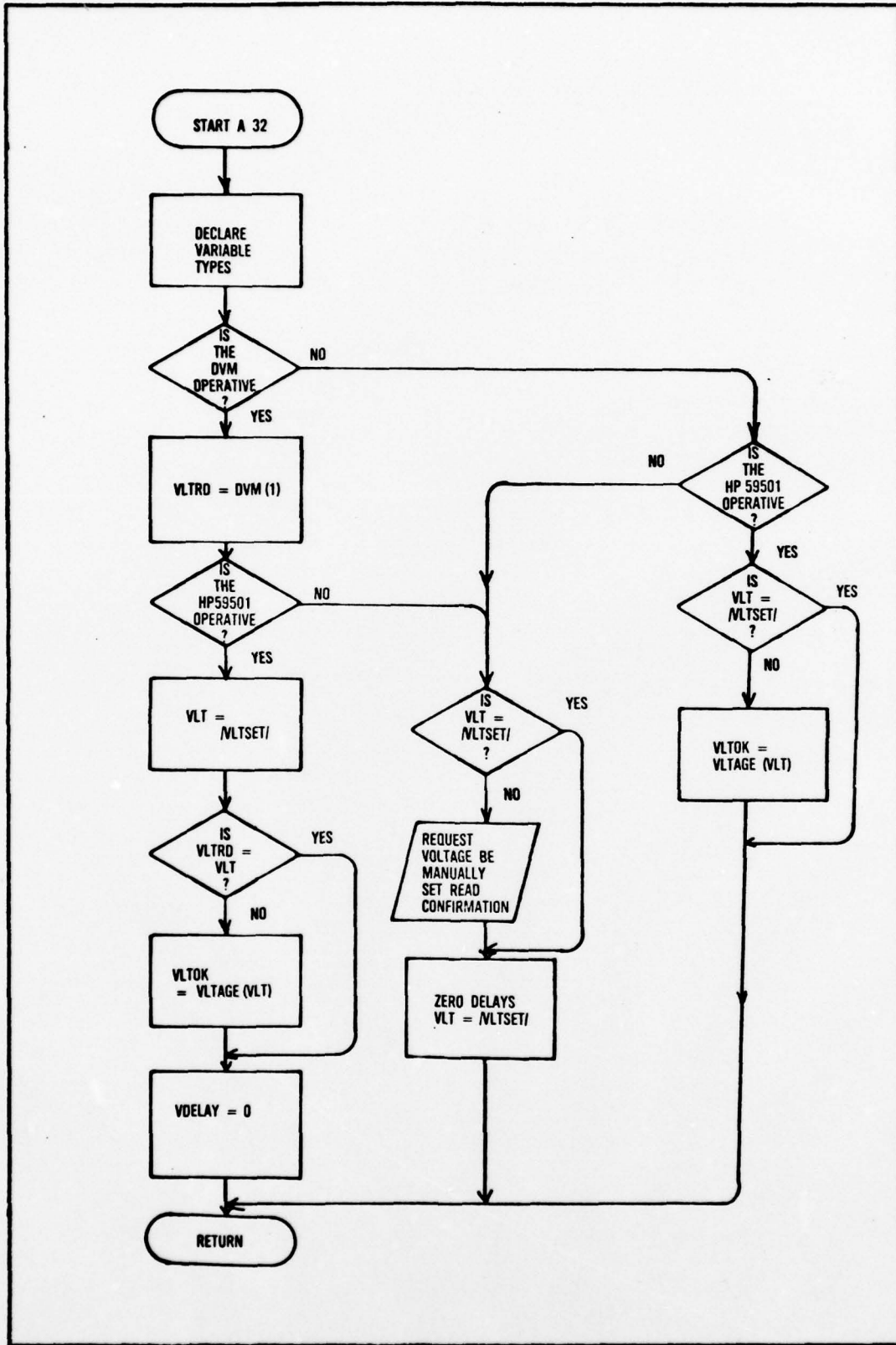


Figure 32. Flow Chart for Module A32

```

0001      SUBROUTINE A32
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C  MODULE A32---CHECK/SET APPLIED VOLTAGE
C
C    THIS MODULE CALLS THE DRIVER PROGRAM FOR THE POWER SUPPLY
C  WHICH SETS THE APPLIED VOLTAGE TO THE SAMPLE. THIS VALUE IS
C  CROSS CHECKED FOR ACCURACY ON THE DIGITAL VOLTMETER. IF
C  THE DUM EQUIPMENT OUT FLAG IS SET THE APPLIED VOLTAGE READING IS
C  ASSUMED TO BE ULTSET. THIS IS A GOOD ASSUMPTION BECAUSE THIS
C  VALUE RELATIVELY STABLE. IF THE POWER SUPPLY IS INOPERATIVE
C  THE OPERATOR IS REQUIRED TO MAKE THE SETTING BY A MESSAGE TO THE
C  TERMINAL.
C
C
C      AUTHOR:  CAPTAIN EDGAR A. VERCHOT, JR., USAF
C
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C  DATA SPECIFICATIONS
C
C  THESE ARE THE DATA SPECIFICATIONS FROM MODULE A1
C  EQPOUT
0002      BYTE EQUIPF(10,8)
0003      INTEGER IEIOF,EOFLAG(7)
C      THESE ARE THE DATA SPECIFICATIONS FOR A2
C  A2COM
0004      INTEGER NTEM,SCSET,SCOUNT,UCOUNT,RUN
0005      REAL TEMSET,FLDSET,ULTSET,X0
C  THESE ARE THE DATA SPECIFICATIONS FOR A3
0006      INTEGER ULTAGE,TSTCON,MAGNET
C  A3COM
0007      INTEGER FDELAY,SAM,FLDOK,TEMOK,ULTOK,SAMOK,
          ZUDELAY,SDELAY,TDELAY,DELAY
C  DUMCOM
0008      INTEGER FUNC
0009      REAL TEMRD,ULTRD
C  THESE ARE THE COMMON BLOCKS FROM MODULE A1
0010      COMMON /EQPOUT/EOFLAG,EQUIPF,IEIOF
C  THE NAMED COMMON BLOCKS FOR A2CMN FOLLOW.
0011      COMMON /A2COM/TEMSET,FLDSET,ULTSET,X0,NTEM,SCSET,SCOUNT,
          ZUCOUNT,RUN
C  THE COMMON BLOCKS FOR A3 FOLLOW
0012      COMMON /A3COM/FDELAY,UDELAY,SDELAY,TDELAY,DELAY,FLD,ULT,
          2TEM,FLDOK,TEMOK,ULTOK,SAMOK,SAM
0013      COMMON /DUMCOM/FUNC,TEMRD,ULTRD
C
C
C
0014  10  CONTINUE
C  CHECK THE EQUIPMENT OUT FLAG FOR THE DIGITAL VOLTMETER
0015      IF(EOFLAG(2).EQ.1) GO TO 500

```

```

C CHECK THE VOLTAGE AGAINST THE VOLTAGE SETTING
0017      ULTRD=DUM(1)
C CHECK TO SEE IF THE POWER SUPPLY IS OPERATIVE
0018      IF(EOFLAG(4).EQ.1) GO TO 600
0020      ULT=ABS(ULTSET)
0021      IF(ULTRD.NE.ULT) ULTOK=VLTAGE(ULT)
0023      UDELAY=0
0024      GO TO 900
0025  500  CONTINUE
0026      IF(EOFLAG(4).EQ.1) GO TO 600
0028      IF(ULT.NE.ABS(ULTSET)) ULTOK=VLTAGE(ULT)
0030      GO TO 900
0031  600  CONTINUE
0032      IF(ULT.EQ.ABS(ULTSET)) GO TO 20
0034      WRITE(7,1000) ULTSET
0035  1000 1      FORMAT(' ', ' PLEASE SET ',G14.7,' VOLTS' /
                ' CARRIAGE RETURN WHEN SET', '$)

0036      PAUSE
0037  20   CONTINUE
0038      UDELAY=0
0039      FDELAY=0
0040      ULTOK=1
0041      ULT=ABS(ULTSET)
0042  900  CONTINUE
0043      RETURN
0044      END

```

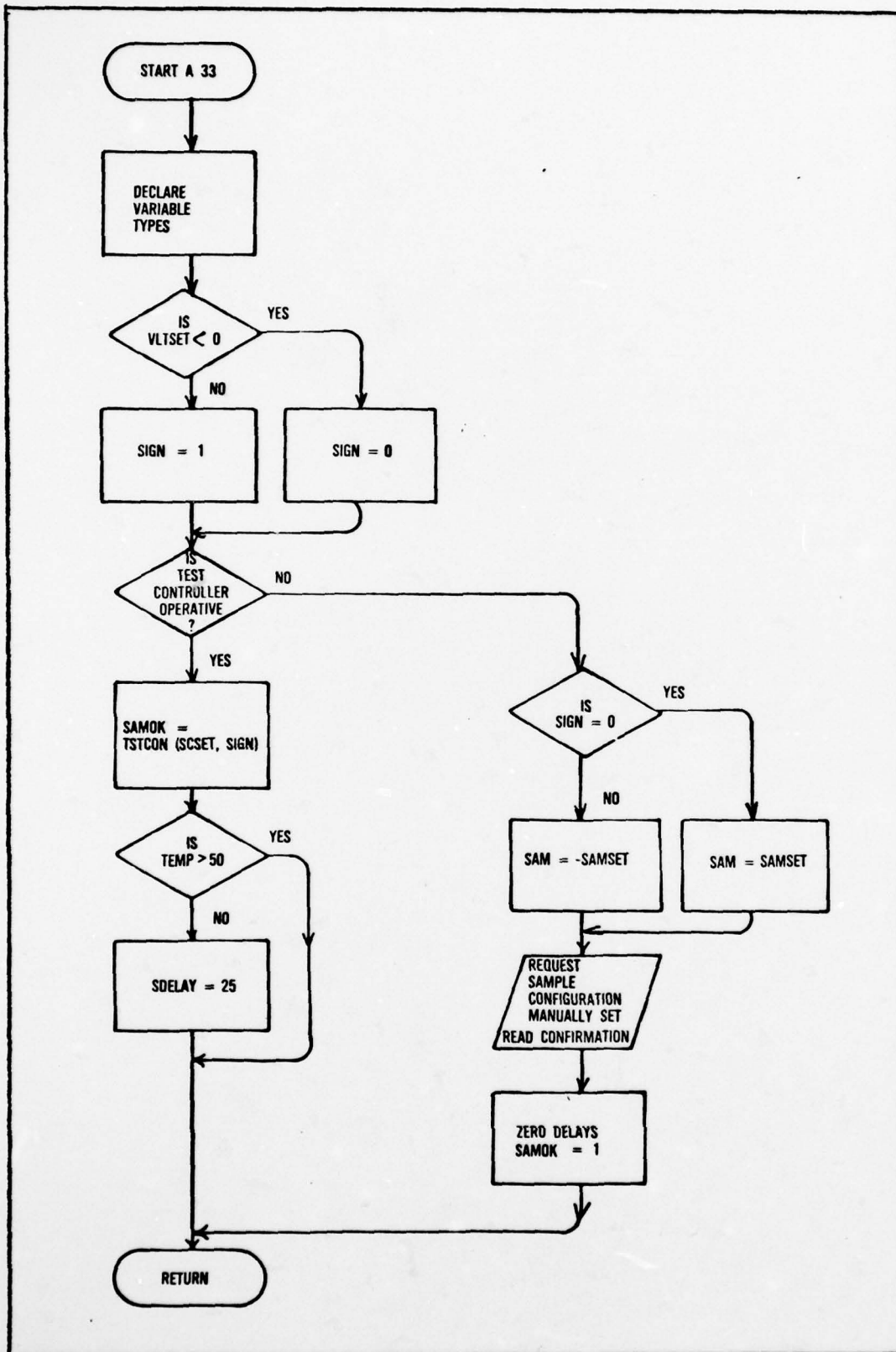


Figure 33. Flow Chart for Module A33

```

0001      SUBROUTINE A33
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C MODULE A33—CHECK/SET SAMPLE CONFIGURATION
          C
          C THIS MODULE CALLS THE DRIVER SUBPROGRAM FOR THE TEST CONTROLLER
          C WHICH SETS THE SAMPLE CONFIGURATION. THE POLARITY OF THE APPLIED
          C VOLTAGE IS ALSO SET HERE ON THE TEST CONTROLLER. IF THE TEST
          C CONTROLLER IS INOPERATIVE, A MANUAL SETTING IS REQUESTED ON THE
          C TERMINAL. THE PROGRAM, THEN SETS THE SETTLING DELAY TIME.
          C
          C
          C      AUTHOR: CAPTAIN EDGAR A. VERCHOT, JR., USAF
          C
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C DATA SPECIFICATION
          C
          C
          C THESE ARE THE DATA SPECIFICATIONS FROM MODULE A1
          C EQPOUT
0002      BYTE EQUIPF(10,8)
0003      INTEGER IEIOF,EOFLAG(7)
          C THESE ARE THE DATA SPECIFICATIONS FOR A2
          C A2COM
0004      INTEGER NTEM,SCSET,SCOUNT,UVCOUNT,RUN
0005      REAL TEMSET,FLDSET,ULTSET,X0
          C THESE ARE THE DATA SPECIFICATIONS FOR A3
0006      INTEGER ULTAGE,TSTCON,MAGNET
          C A3COM
0007      INTEGER FDELAY,SAM,FLDOK,TEMOK,ULTOK,SANOK,
          2UDELAY,SDELAY,TDELAY,DELAY
          C TSTCOM
0008      INTEGER SIGN
          C RAWDAT
0009      REAL TEMDAT(20),ULTDAT(20),SUDATA(20),FLDATA(20),IDATA(20)
0010      INTEGER SCDATA(20)
0011      COMMON /RAWDAT/TEMDAT,ULTDAT,SUDATA,FLDATA,IDATA,SCDATA
          C THESE ARE THE COMMON BLOCKS FROM MODULE A1
0012      COMMON /EQPOUT/EOFLAG,EQUIPF,IEIOF
          C THE NAMED COMMON BLOCKS FOR A2CMM FOLLOW.
0013      COMMON /A2COM/TEMSET,FLDSET,ULTSET,X0,NTEM,SCSET,SCOUNT,
          2UCOUNT,RUN
          C THE COMMON BLOCKS FOR A3 FOLLOW
0014      COMMON /A3COM/FDELAY,UDELAY,SDELAY,TDELAY,DELAY,FLD,ULT,
          2TEM,FLDOK,TEMOK,ULTOK,SANOK,SAM
          C
          C
          C
          C CHECK POLARITY
0015      IF<ULTSET.LT.0.0>GO TO 20

```

```

0017          SIGN=0
0018          GO TO 25
0019 20      CONTINUE
0020          SIGN=1
0021 25      CONTINUE
C          CHECK EQUIPMENT OUT FLAG
0022          IF<EOFLAG<6>.EQ.1> GO TO 500
C SET THE TEST CONTROLLER
0024          SANOK=TSTCON<SCSET,SIGN>
C SET THE DELAY VALUE
0025          IF<IDATA<RUN-1>.GT.5.0E-8>GO TO 10
0027          SDELAY=25
0028          GO TO 900
0029 10      CONTINUE
0030          SDELAY=5
0031          GO TO 900
0032 500    CONTINUE
0033          IF<SIGN.LT.1> GO TO 30
0035          GO TO 35
0036 30      CONTINUE
0037          SAM=-SCSET
0038 35      CONTINUE
0039          SAM=SCSET
0040          WRITE<7,5000> SAM
0041 5000   FORMAT<' ', ' SET THE TEST CONTROLLER TO
1           SAMPLE CONFIGURATION ',I3, /
2           ' CARRIAGE RETURN WHEN SET', $)
0042          PAUSE
0043          SANOK=1
0044          SDELAY=0
0045          UDELAY=0
0046          FDELAY=0
0047 900    CONTINUE
0048          RETURN
0049          END

```

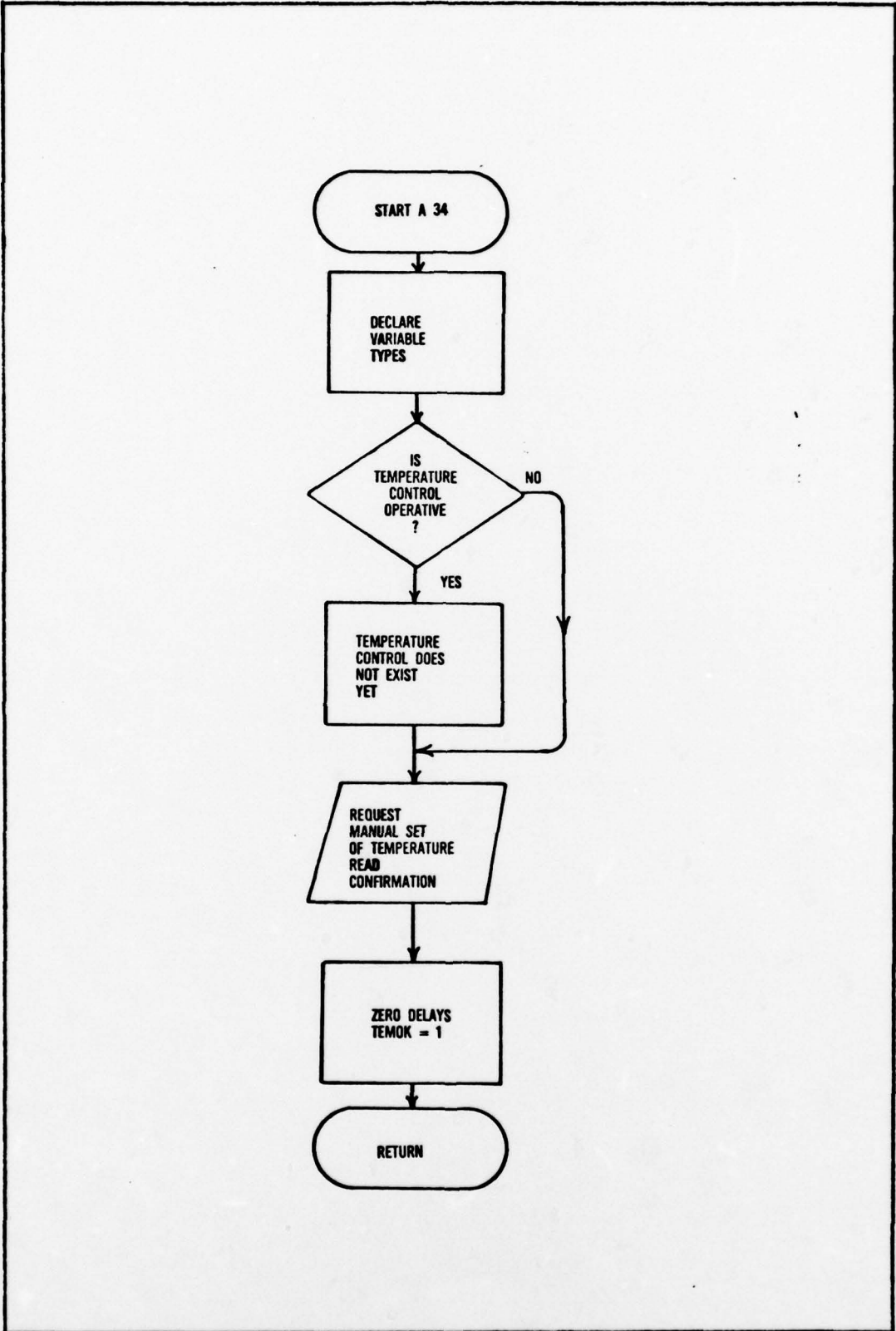


Figure 34. Flow Chart for Module A34

```

0001      SUBROUTINE A34
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C MODULE A34---CHECK/SET TEMPERATURE
          C
          C THIS MODULE WILL SET THE HEATER VOLTAGE ON THE HEATER
          C POWER SUPPLY TO THE PROPER VALUE TO STABILIZE THE SAMPLE
          C AT THE DESIRED STEADY STATE TEMPERATURE. IF THE EQUIPMENT
          C OUT FLAG IS SET, A MESSAGE WILL DIRECT THE OPERATOR TO SET THE
          C TEMPERATURE TO THE PROPER SETTING.
          C
          C AUTHOR: CAPTAIN EDGAR A. VERCHOT, JR., USAF
          C
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C DATA SPECIFICATIONS
          C THESE ARE THE DATA SPECIFICATIONS FROM MODULE A1
          C EQPOUT
0002      BYTE EQUIPF(10,8)
0003      INTEGER IEIOF,EOFLAG(7)
          C THESE ARE THE DATA SPECIFICATIONS FOR A2
          C A2COM
0004      INTEGER NTEM,SCSET,SCOUNT,UOUNT,RUN
0005      REAL TEMSET,FLDSET,ULTSET,X0
          C THESE ARE THE DATA SPECIFICATIONS FOR A3
0006      INTEGER ULTAGE,TSTCON,MAGNET
          C A3COM
0007      INTEGER FDELAY,SAM,FLDOK,TEMOK,ULTOK,SAMOK,
          2UDELAY,SDELAY,TDELAY,DELAY
          C DUMCOM
0008      INTEGER FUNC
0009      REAL TEMRD,ULTRD
          C THESE ARE THE COMMON BLOCKS FROM MODULE A1
0010      COMMON /EQPOUT/EOFLAG,EQUIPF,IEIOF
          C THE NAMED COMMON BLOCKS FOR A2CMN FOLLOW.
0011      COMMON /A2COM/TEMSET,FLDSET,ULTSET,X0,NTEM,SCSET,SCOUNT,
          2UCOUNT,RUN
          C THE COMMON BLOCKS FOR A3 FOLLOW
0012      COMMON /A3COM/FDELAY,UDELAY,SDELAY,TDELAY,DELAY,FLD,ULT,
          2TEM,FLDOK,TEMOK,ULTOK,SAMOK,SAM
0013      COMMON /DUMCOM/FUNC,TEMRD,ULTRD
          C
          C
          C
          C CHECK EQUIPMENT OUT FLAG
0014      IF(EOFLAG(3).EQ.1) GO TO 500
          C
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C THE HEATER CONTROL DOES NOT EXIST YET. WILL BE PLACED HERE
          C IN PROGRAM WHEN IMPLEMENTED.
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0016      500 CONTINUE
0017      IF(RUN.NE.1.AND.RUN.NE.6.AND.RUN.NE.11.AND.RUN.NE.16)GO TO 10

```

```

0019      WRITE(7,5000) TEMSET
0020 5000  FORMAT(' ', ' SET THE TEMPERATURE AS READ BY THE SILICON' /
           1      ' THERMOMETER TO APPROXIMATELY',G14.7,' VOLTS' /
           2      ' CARRIAGE RETURN WHEN SET', $)
0021      PAUSE
           C SET THE DELAYS TO ZERO IF TEMPERATURE IS RESET MANUALLY
0022      TDELAY=0
0023      FDELAY=0
0024      SDELAY=0
0025      UDELAY=0
0026 10   CONTINUE
0027      TEMOK=1
0028 900  CONTINUE
0029      RETURN
0030      END

```

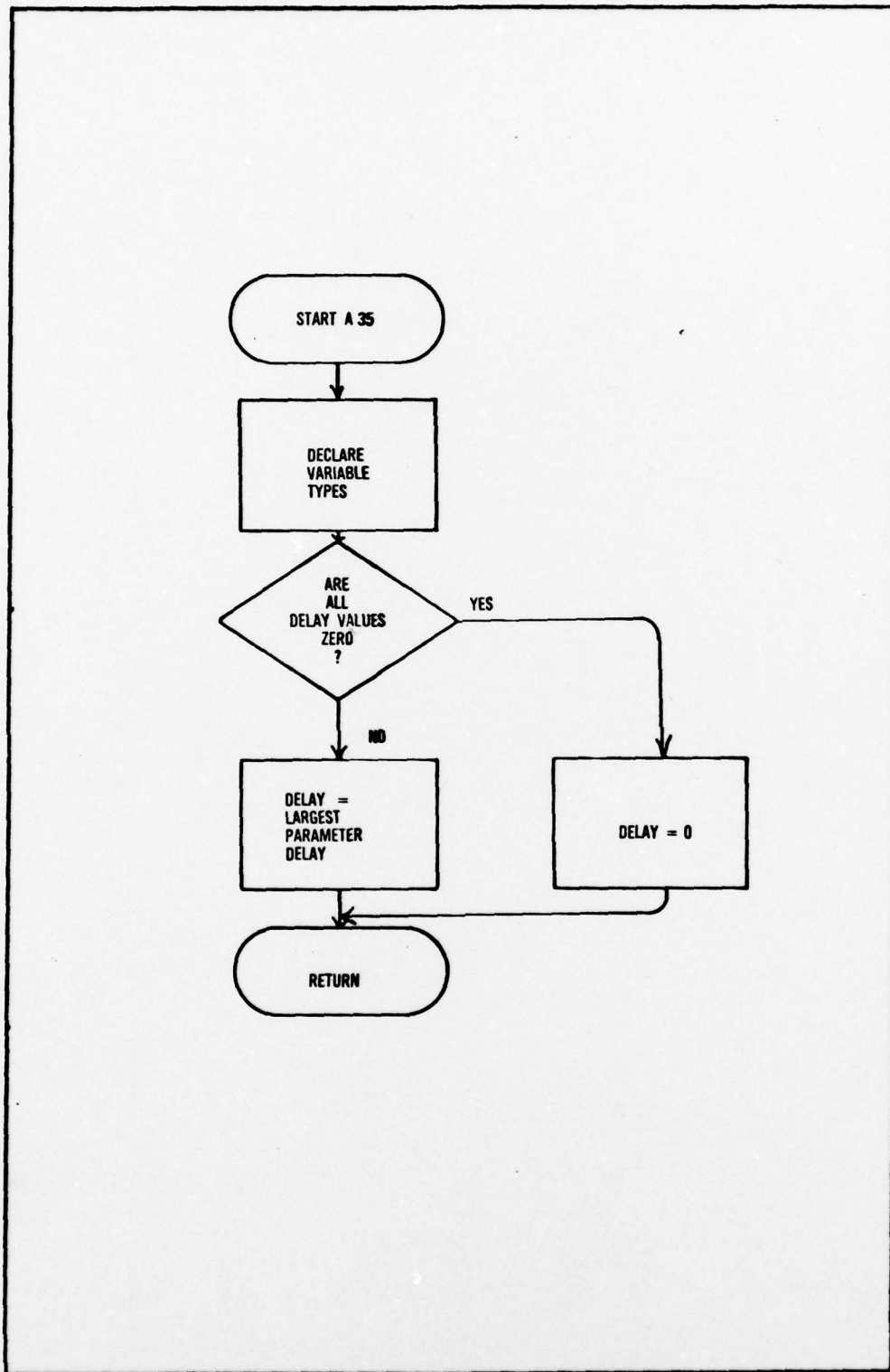


Figure 35. Flow Chart for Module A35

```

0001      SUBROUTINE A35
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C MODULE A35---GENERATE AQUIRE DATA SIGNAL
          C
          C THIS MODULE CHECKS ALL OF THE PARAMETER OK SIGNALS
          C AND IF ALL ARE TRUE, GENERATES THE START DATA ACQUISITION
          C SIGNAL. IF ALL ARE NOT TRUE, IT CALLS THE DELAY RECHECK
          C MODULE.
          C
          C AUTHOR: CAPTAIN EDGAR A. VERCHOT, JR. USAF
          C
          C
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C DATA SPECCIFICATIONS
          C THESE ARE THE DATA SPECIFICATIONS FOR A3
          C A3COM
0002      INTEGER FDELAY, SAM, FLDOK, TEMOK, ULTOK, SAMOK,
          2UDELAY, SDELAY, TDELAY, DELAY
          C THE COMMON BLOCKS FOR A3 FOLLOW
0003      COMMON /A3COM/ FDELAY, UDELAY, SDELAY, TDELAY, DELAY, FLD, ULT,
          2TEM, FLDOK, TEMOK, ULTOK, SAMOK, SAM
          C
          C USE A3COM
          C CHECK ALL OF THE PARAMETER OK SIGNALS
0004      IF (FDELAY.NE.0.AND.UDELAY.NE.0.AND.SDELAY.NE.0.AND.TDELAY.NE.0)
          1 GO TO 500
          C SET THE DELAY
0006      IF (TDELAY.GT.FDELAY) GO TO 10
0008      DELAY=FDELAY
0009      GO TO 20
0010 10 CONTINUE
0011      DELAY=TDELAY
0012 20 CONTINUE
0013      IF (SDELAY.GT.DELAY) DELAY=SDELAY
0015      IF (UDELAY.GT.DELAY) DELAY=UDELAY
0017      GO TO 900
0018 500 CONTINUE
0019      DELAY=5
0020 900 CONTINUE
          C RESET THE DELAY VALUES TO ZERO
0021      SDELAY=0
0022      TDELAY=0
0023      UDELAY=0
0024      FDELAY=0
0025      RETURN
0026      END

```

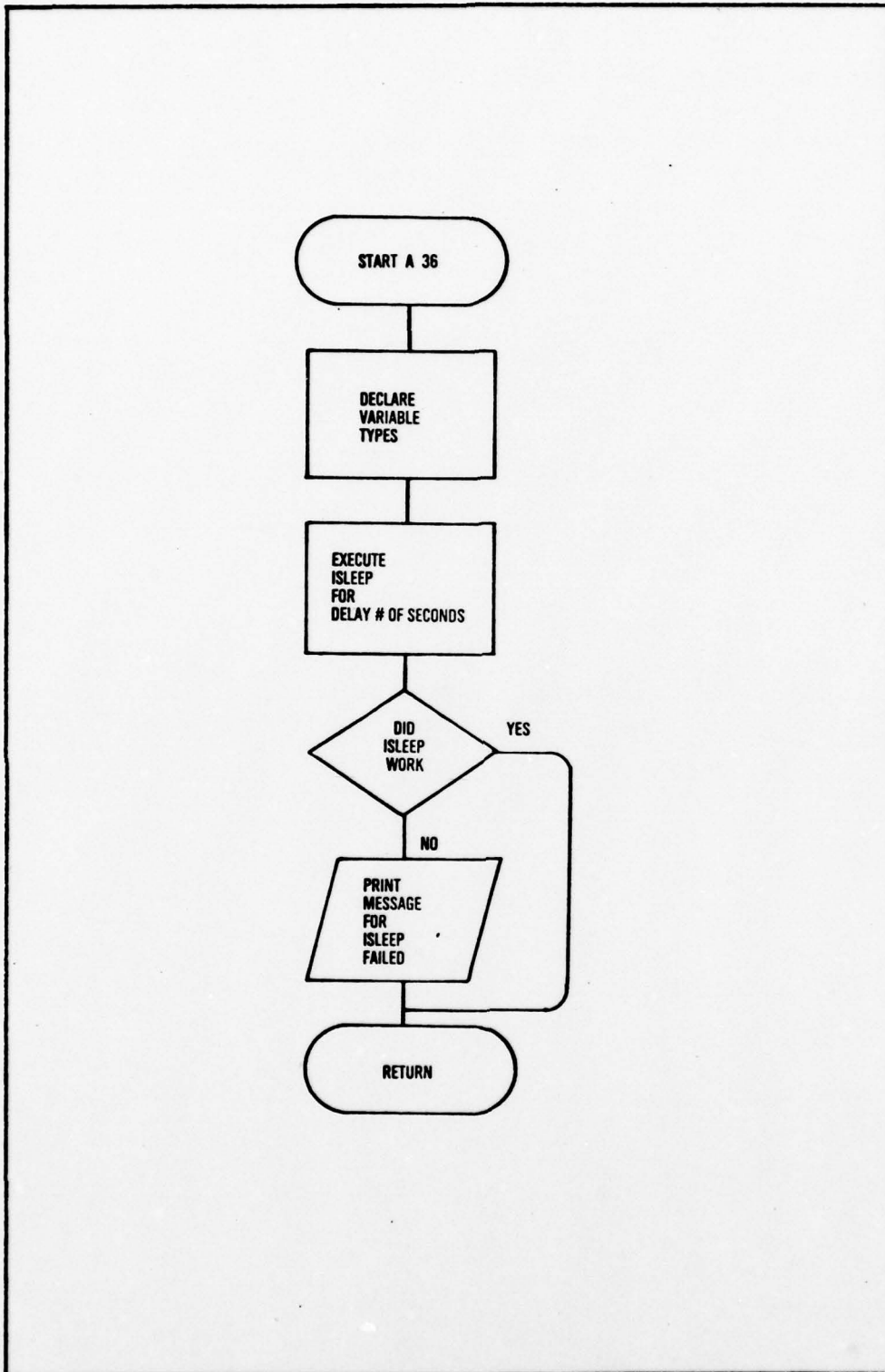


Figure 36. Flow Chart for Module A36

```

0001      SUBROUTINE A36(DELAY)
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C MODULE A36---DELAY RECHECK
          C
          C THIS MODULE IS CALLED TO DELAY THE RECHECKING OF PARAMETERS.
          C IF SIMPLY WAITS FOR THENUMBER OF SECONDS SPECIFIED BY THE DELAY
          C PARAMETER BEFORE RETURNING.
          C
          C
          C AUTHOR: CAPTAIN EDGAR A. VERCHOT, JR., USAF
          C
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
0002      INTEGER DELAY
          C CONVERT DELAY TO MINUTES AND SECONDS
0003      M=DELAY/60
0004      IS=DELAY-60*M
          C NOW DELAY
0005      I=ISLEEP(0,M,IS,0)
0006      IF(I.NE.0)GO TO 800
0008      GO TO 900
0009 800 CONTINUE
0010      WRITE(7,8000)
0011 8000 FORMAT(' ',' ISLEEP IN THE A36 MODULE FAILED')
0012 900 CONTINUE
0013      RETURN
0014      END

```

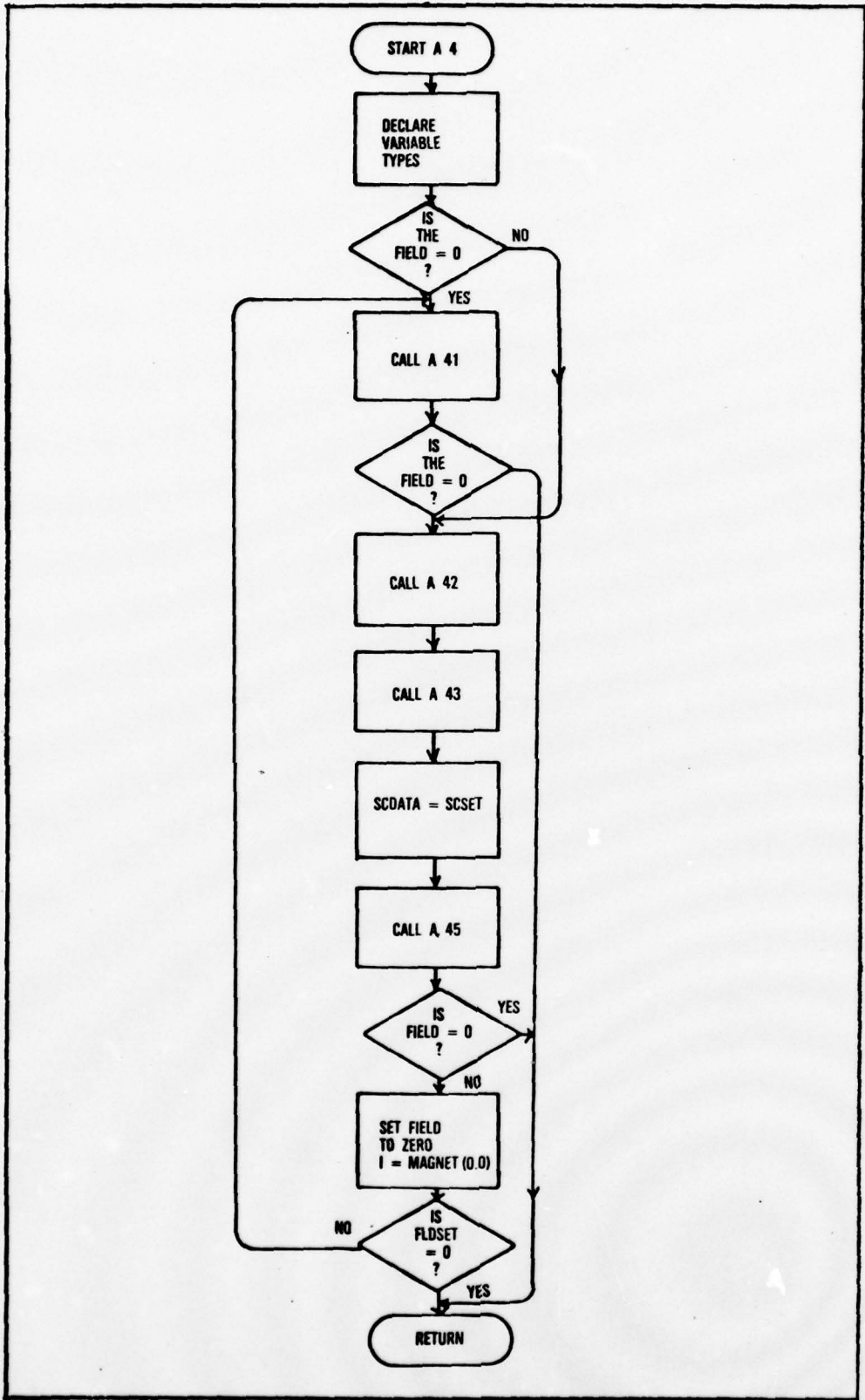


Figure 37. Flow Chart for Module A4

```

0001      SUBROUTINE A4
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C  MODULE A4---ACQUIRE DATA
C
C      MODULE A4 IS THE EXECUTIVE FOR ACQUIRE DATA.  IT CALLS
C  FOUR SUBORDINATE MODULES:
C    A41---READ TEMPERATURE
C    A42---READ CURRENT
C    A43---READ APPLIED AND SAMPLE VOLTAGE
C    A45---READ FIELD
C (A44 WAS DELETED INTENTIONALLY BECAUSE IT WAS TRIVIAL)
C EACH OF THESE MODULES READS THE DESIRED PARAMETER(S) FROM THE
C APPROPRIATE DRIVER SUBROUTINE(S) AND RETURNS THE DATA TO A4.
C A4 THEN PASSES IT BACK AND GENERATES THE DATA ACQUISITION
C COMPLETE SIGNAL.  IF THE FIELD IS ON, THE MODULE SKIPS A41 UNTIL
C AFTER A45.  IT THEN TURNS THE FIELD OFF AND READS THE
C TEMPERATURE.
C
C      AUTHOR:  CAPTAIN EDGAR A. VERCHOT, JR., USAF
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C DATA SPECIFICATIONS
C THESE ARE THE DATA SPECIFICATIONS FROM MODULE A1
C EQPOUT
0002      BYTE EQUIPF(10,8)
0003      INTEGER IEIOF,EOFLAG(7)
C      THESE ARE THE DATA SPECIFICATIONS FOR A2
C
C    A2COM
0004      INTEGER NTEM,SCSET,SCOUNT,VCOUNT,RUN
0005      REAL TEMSET,FLDSET,ULTSET,X0
C THESE ARE THE DATA SPECIFICATIONS FOR A3
0006      INTEGER ULTAGE,TSTCON,MAGNET
C A3COM
0007      INTEGER FDELAY,SAM,FLDOK,TEMOK,ULTOK,SAMOK,
          ZUDELAY,SDELAY,TDELAY,DELAY
C DUMCOM
0008      INTEGER FUNC
0009      REAL TEMRD,ULTRD
C THESE ARE THE DATA SPECIFICATIONS FOR A4
C  RAUDAT
0010      REAL TEMDAT(20),ULTDAT(20),SUDATA(20),FLDATA(20),IDATA(20)
0011      INTEGER SCDATA(20)
C THESE ARE THE COMMON BLOCKS FROM MODULE A1
0012      COMMON /EQPOUT/EOFLAG,EQUIPF,IEIOF
C THE NAMED COMMON BLOCKS FOR A2CMN FOLLOW.
0013      COMMON /A2COM/TEMSET,FLDSET,ULTSET,X0,NTEM,SCSET,SCOUNT,
          ZUCOUNT,RUN
C THE COMMON BLOCKS FOR A3 FOLLOW
0014      COMMON /A3COM/FDELAY,UDELAY,SDELAY,TDELAY,DELAY,FLD,ULT,
          ZTEM,FLDOK,TEMOK,ULTOK,SAMOK,SAM
0015      COMMON /DUMCOM/FUNC,TEMRD,ULTRD

```

```

0016 C THE COMMON BLOCKS FOR A4 FOLLOW
      COMMON /RAIDAT/TEMDAT,ULTDAT,SVDATA,FLDATA,IDATA,SCDATA
      C
      C
      C
      C NOW READ THE TEMPERATURE
0017 IF<FLDSET.NE.0.0>GO TO 20
0019 10 CONTINUE
0020 CALL A41
0021 IF<FLDSET.NE.0.0> GO TO 900
      C READ THE CURRENT
0023 20 CONTINUE
0024 CALL A42
      C READ THE APPLIED AND SAMPLE VOLTAGES
0025 CALL A43
      C READ THE SAMPLE CONFIGURATION
0026 SCDATA<RUN>=SCSET
0027 IF<ULTSET.LT.0.0>SCDATA<RUN>=-SCDATA<RUN>
      C READ THE FIELD
0029 CALL A45
      C SET THE FIELD TO ZERO IF NECESSARY TO READ THERMOMETER
0030 IF<FLDSET.EQ.0.0>GO TO 900
0032 I=MAGNET<0.0,FLD>
0033 IF<FLDSET.NE.0.0>GO TO 10
0035 900 CONTINUE
0036 RETURN
0037 END

```

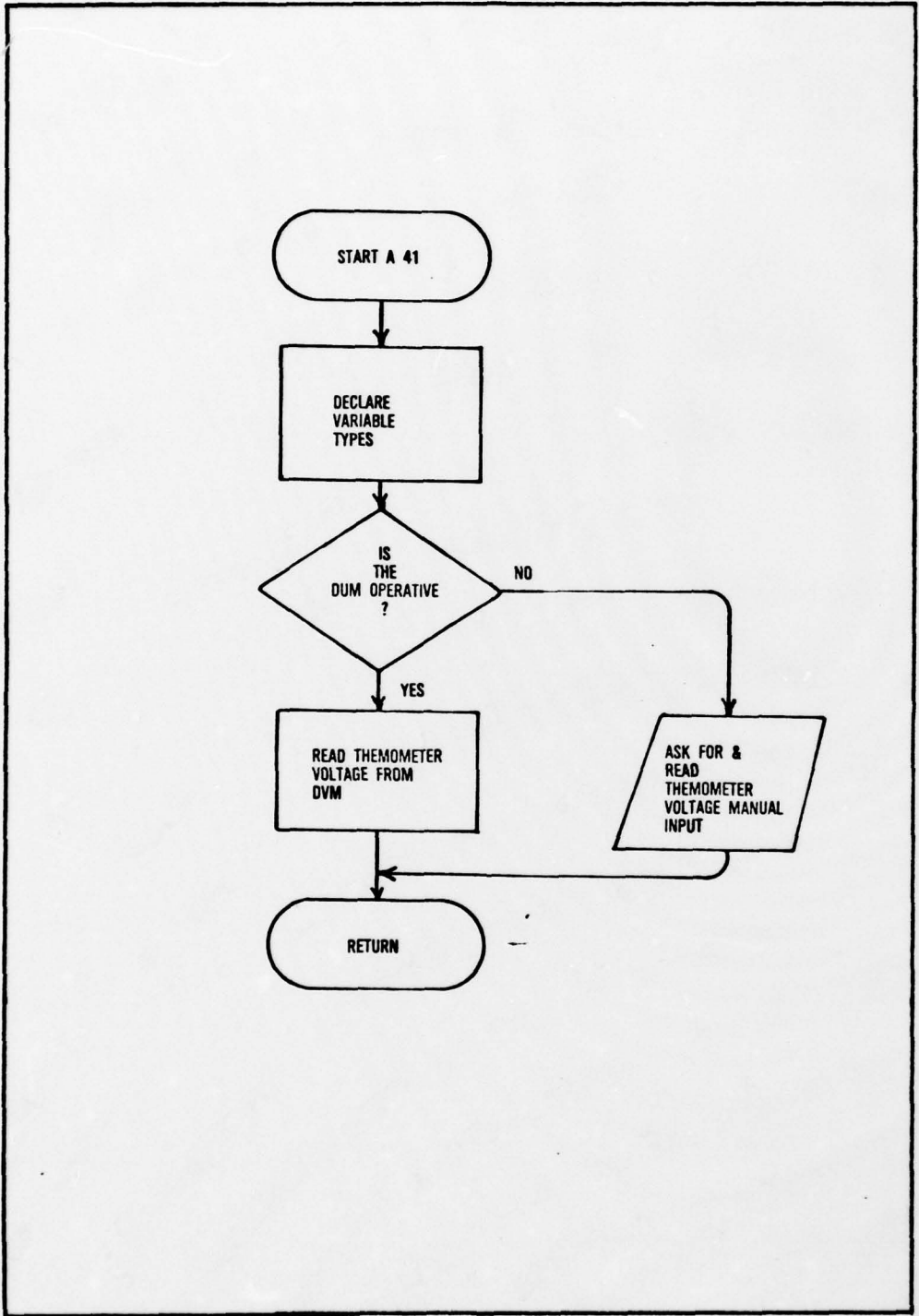


Figure 38. Flow Chart for Module A41

```

0001      SUBROUTINE A41
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C MODULE A41---READ TEMPERATURE
          C
          C     THIS MODULE READS THE VOLTAGE OF THE SILICON THERMOMETER FROM
          C THE DIGITAL VOLTMETER.
          C IF THE DUM IS INOPERATIVE IT REQUESTS THE VALUE FROM THE
          C OPERATOR ON THE TERMINAL.
          C
          C
          C     AUTHOR:  CAPTAIN EDGAR A. VERCHOT, JR., USAF
          C
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C DATA SPECIFICATIONS
          C THESE ARE THE DATA SPECIFICATIONS FROM MODULE A1
          C EQPOUT
0002      BYTE EQUIPF(10,8)
0003      INTEGER IEIOF,EOFLAG(7)
          C     THESE ARE THE DATA SPECIFICATIONS FOR A2
          C
          C     A2COM
0004      INTEGER NTEM,SCSET,SCOUNT,UCOUNT,RUN
0005      REAL TEMSET,FLDSET,ULTSET,X0
          C THESE ARE THE DATA SPECIFICATIONS FOR A3
0006      INTEGER ULTAGE,TSTCON,MAGNET
          C DUMCOM
0007      INTEGER FUNC
0008      REAL TEMRD,ULTRD
          C THESE ARE THE DATA SPECIFICATIONS FOR A4
          C     RANDAT
0009      REAL TEMDAT(20),ULTDAT(20),SVDATA(20),FLDATA(20),IDATA(20)
0010      INTEGER SCDATA(20)
          C THESE ARE THE COMMON BLOCKS FROM MODULE A1
0011      COMMON /EQPOUT/EOFLAG,EQUIPF,IEIOF
          C THE NAMED COMMON BLOCKS FOR A2CNV FOLLOW.
0012      COMMON /A2COM/TEMSET,FLDSET,ULTSET,X0,NTEM,SCSET,SCOUNT,
          2UCOUNT,RUN
          C THE COMMON BLOCKS FOR A3 FOLLOW
0013      COMMON /DUMCOM/FUNC,TEMRD,ULTRD
          C THE COMMON BLOCKS FOR A4 FOLLOW
0014      COMMON /RANDAT/TEMDAT,ULTDAT,SVDATA,FLDATA,IDATA,SCDATA
          C
          C CHECK IF DUM IS OPERATIVE
0015      IF(EOFLAG(2).EQ.2) GO TO 20
          C INTERROGATE THE DUM TO DETERMINE TEMPERATURE
0017      TEMRD=DUM(3)
0018      TEMDAT(RUN)=TEMRD
0019      GO TO 900
0020  20    CONTINUE
0021      WRITE(7,2000)
0022  2000  FORMAT(' ', ' ENTER THE CURRENT VOLTAGE READING ON THE
          2SILICON THERMOMETER ')

```

0023 READ(5,*) TEMPAT(RUN)
0024 900 CONTINUE
0025 RETURN
0026 END

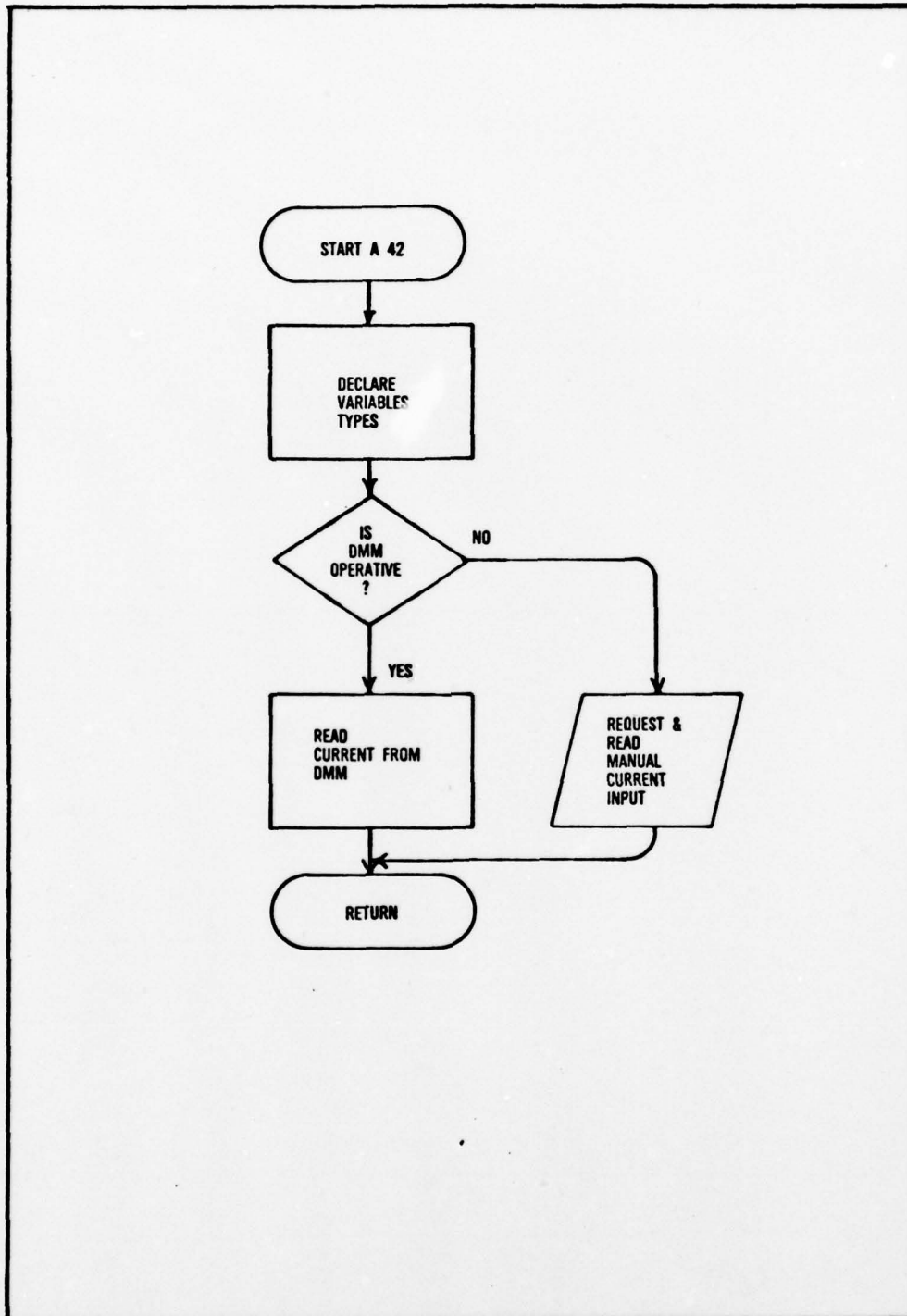


Figure 39. Flow Chart for Module A42

```

0001      SUBROUTINE A42
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C MODULE A42---READ CURRENT
          C
          C THIS MODULE READS THE SAMPLE CURRENT FROM THE ELECTROMETER.
          C IF THE DMM(ELECTROMETER) IS INOPERATIVE, IT REQUESTS THE
          C OPERATOR TO ENTER THE VALUE ON THE TERMINAL.
          C
          C
          C AUTHOR: CAPTAIN EDGAR A. VERCHOT, JR., USAF
          C
          C
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C THESE ARE THE DATA SPECIFICATIONS FROM MODULE A1
          C EQPOUT
0002      BYTE EQUIPF(10,8)
0003      INTEGER IEIOF,EOFLAG(7)
          C THESE ARE THE DATA SPECIFICATIONS FOR A2
          C
          C A2COM
0004      INTEGER NTEM,SCSET,SCOUNT,UCOUNT,RUN
0005      REAL TEMSET,FLDSET,ULTSET,X0
          C THESE ARE THE DATA SPECIFICATIONS FOR A3
0006      INTEGER ULTAGE,TSTCON,MAGNET
          C DMMCOM
0007      REAL CRNTRD
          C THESE ARE THE DATA SPECIFICATIONS FOR A4
          C
          C RANDAT
0008      REAL TEMDAT(20),ULTDAT(20),SUDATA(20),FLDATA(20),IDATA(20)
0009      INTEGER SCDATA(20)
          C THESE ARE THE COMMON BLOCKS FROM MODULE A1
0010      COMMON /EQPOUT/EOFLAG,EQUIPF,IEIOF
          C THE NAMED COMMON BLOCKS FOR A2CNV FOLLOW.
0011      COMMON /A2COM/TEMSET,FLDSET,ULTSET,X0,NTEM,SCSET,SCOUNT,
          2UCOUNT,RUN
          C THE COMMON BLOCKS FOR A3 FOLLOW
0012      COMMON /DMMCOM/CRNTRD
          C THE COMMON BLOCKS FOR A4 FOLLOW
0013      COMMON /RANDAT/TEMDAT,ULTDAT,SUDATA,FLDATA,IDATA,SCDATA
          C
          C CHECK IF THE ELECTROMETER IS INOPERATIVE
0014      IF(EOFLAG(1).EQ.1)GO TO 10
0016      CRNTRD=DMM(C)
0017      IDATA(RUN)=CRNTRD
0018      GO TO 900
0019  10  CONTINUE
          C GET THE VALUE OF CURRENT IF THE DMM IS INOPERATIVE
0020      WRITE(7,1000)
0021  1000  FORMAT(' ',' ENTER THE SAMPLE CURRENT ')
0022      READ(5,*) IDATA(RUN)
0023  900  CONTINUE
0024      RETURN

```

0025

END

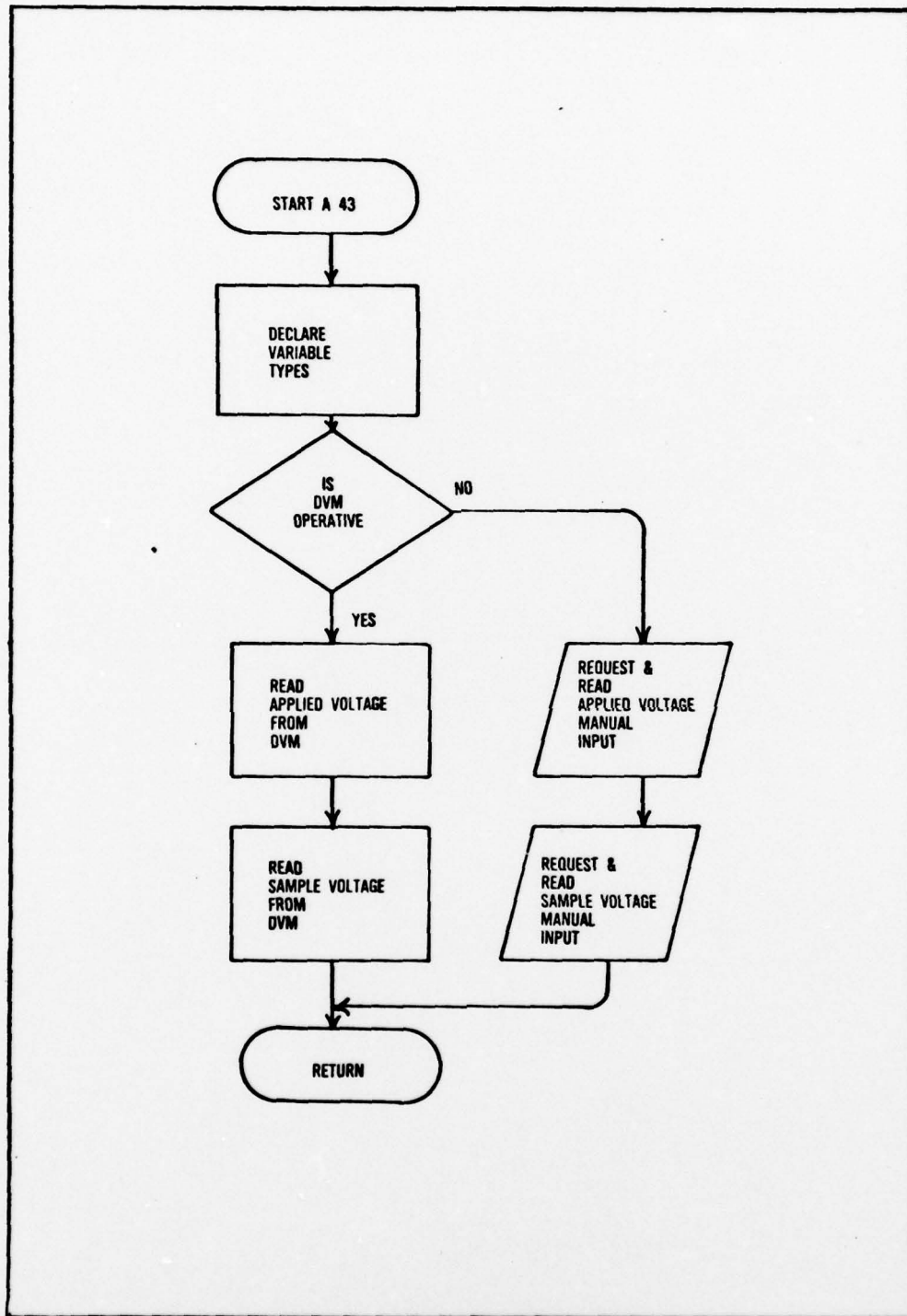


Figure 40. Flow Chart for Module A43

```

0001      SUBROUTINE A43
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C MODULE A43---READ APPLIED AND SAMPLE VOLTAGE
          C
          C THIS MODULE READS THE APPLIED VOLTAGE AND THE SAMPLE
          C VOLTAGE FROM THE DIGITAL VOLTMETER. IF THE DUM IS INOPERATIVE
          C THE MODULE REQUESTS THE OPERATOR TO INPUT THE VOLTAGE READINGS
          C FROM THE TERMINAL
          C
          C
          C AUTHOR: CAPTAIN EDGAR A. VERCHOT, JR., USAF
          C
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C THESE ARE THE DATA SPECIFICATIONS FROM MODULE A1
          C EQPOUT
0002      BYTE EQUIPF(10,8)
0003      INTEGER IEIOF,EOFLAG(7)
          C THESE ARE THE DATA SPECIFICATIONS FOR A2
          C
          C A2COM
0004      INTEGER NTEM,SCSET,SCOUNT,UCOUNT,RUN
0005      REAL TEMSET,FLDSET,ULTSET,X0
          C THESE ARE THE DATA SPECIFICATIONS FOR A3
          C DUMCOM
0006      INTEGER FUNC
0007      REAL TEMRD,ULTRD
          C THESE ARE THE DATA SPECIFICATIONS FOR A4
          C
          C RAWDAT
0008      REAL TEMDAT(20),ULTDAT(20),SUDATA(20),FLDATA(20),IDATA(20)
0009      INTEGER SCDATA(20)
          C THESE ARE THE COMMON BLOCKS FROM MODULE A1
0010      COMMON /EQPOUT/EOFLAG,EQUIPF,IEIOF
          C THE NAMED COMMON BLOCKS FOR A2CMM FOLLOW.
0011      COMMON /A2COM/TEMSET,FLDSET,ULTSET,X0,NTEM,SCSET,SCOUNT,
          2UCOUNT,RUN
          C THE COMMON BLOCKS FOR A3 FOLLOW
0012      COMMON /DUMCOM/FUNC,TEMRD,ULTRD
          C THE COMMON BLOCKS FOR A4 FOLLOW
0013      COMMON /RAWDAT/TEMDAT,ULTDAT,SUDATA,FLDATA,IDATA,SCDATA
          C
          C CHECK TO SEE IF THE DUM IS OPERATIVE
0014      IF(EOFLAG(2).EQ.1)GO TO 10
          C READ THE APPLIED VOLTAGE
0016      ULTRD=DUM(1)
0017      ULTDAT(RUN)=ULTRD
          C NOW READ THE SAMPLE VOLTAGE
0018      ULTRD=DUM(2)
0019      SUDATA(RUN)=ULTRD
0020      GO TO 900
0021      10 CONTINUE
          C HAVE THE OPERATOR ENTER THE REQUIRED VALUES
0022      WRITE(7,1000)

```

```
0023 1000   FORMAT(' ', ' ENTER THE APPLIED VOLTAGE READING ')
0024       READ(5,*) ULTDAT(RUN)
0025       WRITE(7,2000)
0026 2000   FORMAT(' ', ' ENTER THE SAMPLE VOLTAGE READING ')
0027       READ(5,*) SUDATA(RUN)
0028 900    CONTINUE
0029       RETURN
0030       END
```

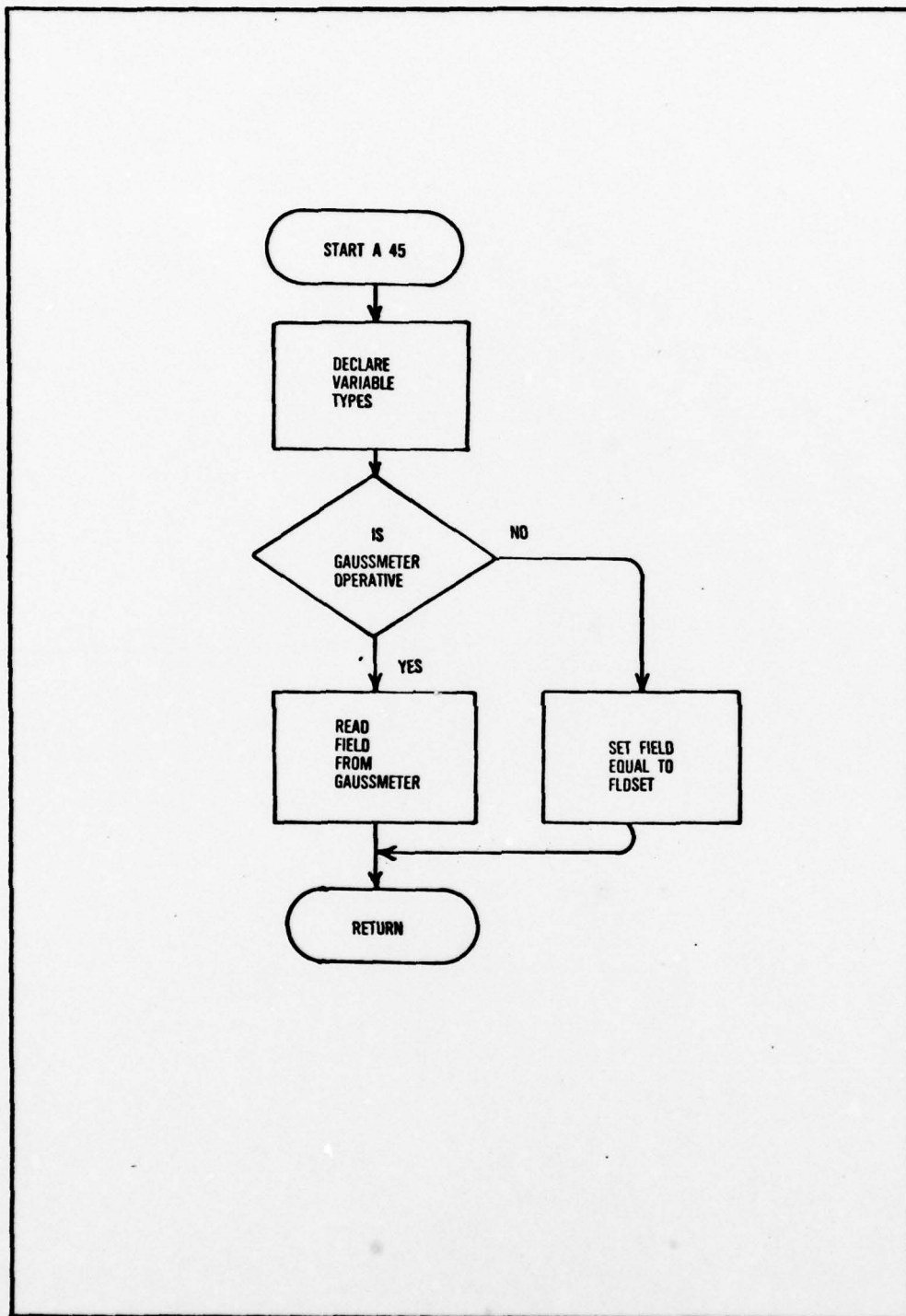


Figure 41. Flow Chart for Module A45

```

0001      SUBROUTINE A45
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C MODULE A45---READ FIELD
          C
          C     THIS MODULE CALLS GAUSS TO DETERMINE THE FIELD VALUE.
          C GAUSS IS A DRIVER ROUTINE FOR READING THE GAUSSMETER. IF
          C THE GAUSS METER IS INOPERATIVE THE MODULE ASSUMES THAT
          C THE SETTING IS THE CORRECT VALUE. THIS IS A LOW RISK
          C ASSUMPTION BECAUSE WITHOUT THE GAUSSMETER THE FIELD WILL
          C HAVE BEEN MANUALLY SET TO THE PROPER VALUE AND THE MAGNET
          C IS VERY STABLE.
          C
          C
          C     AUTHOR: CAPTAIN EDGAR A. VERCHOT, JR., USAF
          C
          C
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C DATA SPECIFICATIONS
          C THESE ARE THE DATA SPECIFICATIONS FROM MODULE A1
          C EQPOUT
0002      BYTE EQUIPF(10,8)
0003      INTEGER IEIOF,EFLAG(7)
          C     THESE ARE THE DATA SPECIFICATIONS FOR A2
          C
          C     A2COM
0004      INTEGER NTEM,SCSET,SCOUNT,UCOUNT,RUN
0005      REAL TEMSET,FLDSET,ULTSET,X0
          C THESE ARE THE DATA SPECIFICATIONS FOR A3
0006      INTEGER ULTAGE,MAGNET,TSTCON
          C GAUSSM
0007      REAL FLDRD
          C THESE ARE THE DATA SPECIFICATIONS FOR A4
          C     RAWDAT
0008      REAL TEMDAT(20),ULTDAT(20),SUDATA(20),FLDATA(20),IDATA(20)
0009      INTEGER SCDATA(20)
          C THESE ARE THE COMMON BLOCKS FROM MODULE A1
0010      COMMON /EQPOUT/EFLAG,EQUIPF,IEIOF
          C THE NAMED COMMON BLOCKS FOR A2CMN FOLLOW.
0011      COMMON /A2COM/TEMSET,FLDSET,ULTSET,X0,NTEM,SCSET,SCOUNT,
          2UCOUNT,RUN
          C THE COMMON BLOCKS FOR A3 FOLLOW
0012      COMMON /GAUSSM/FLDRD
          C THE COMMON BLOCKS FOR A4 FOLLOW
0013      COMMON /RAWDAT/TEMDAT,ULTDAT,SUDATA,FLDATA,IDATA,SCDATA
          C
          C
          C CHECK TO SEE IF THE GAUSS METER IS OPERATIVE
          C
          C
0014      IF(EFLAG(7).EQ.1) GO TO 100
0016      FLDRD=GAUSS(F)
0017      FLDATA(RUN)=FLDRD

```

```
0018          GO TO 900
0019 100      CONTINUE
          C    SET THE FLDATA VALUE TO EQUAL THE FIELD SETTING IF THE METER
          C    IS INOPERATIVE.
          C
0020          FLDATA(RUN)=FLDSET
0021 900      CONTINUE
0022          RETURN
0023          END
```

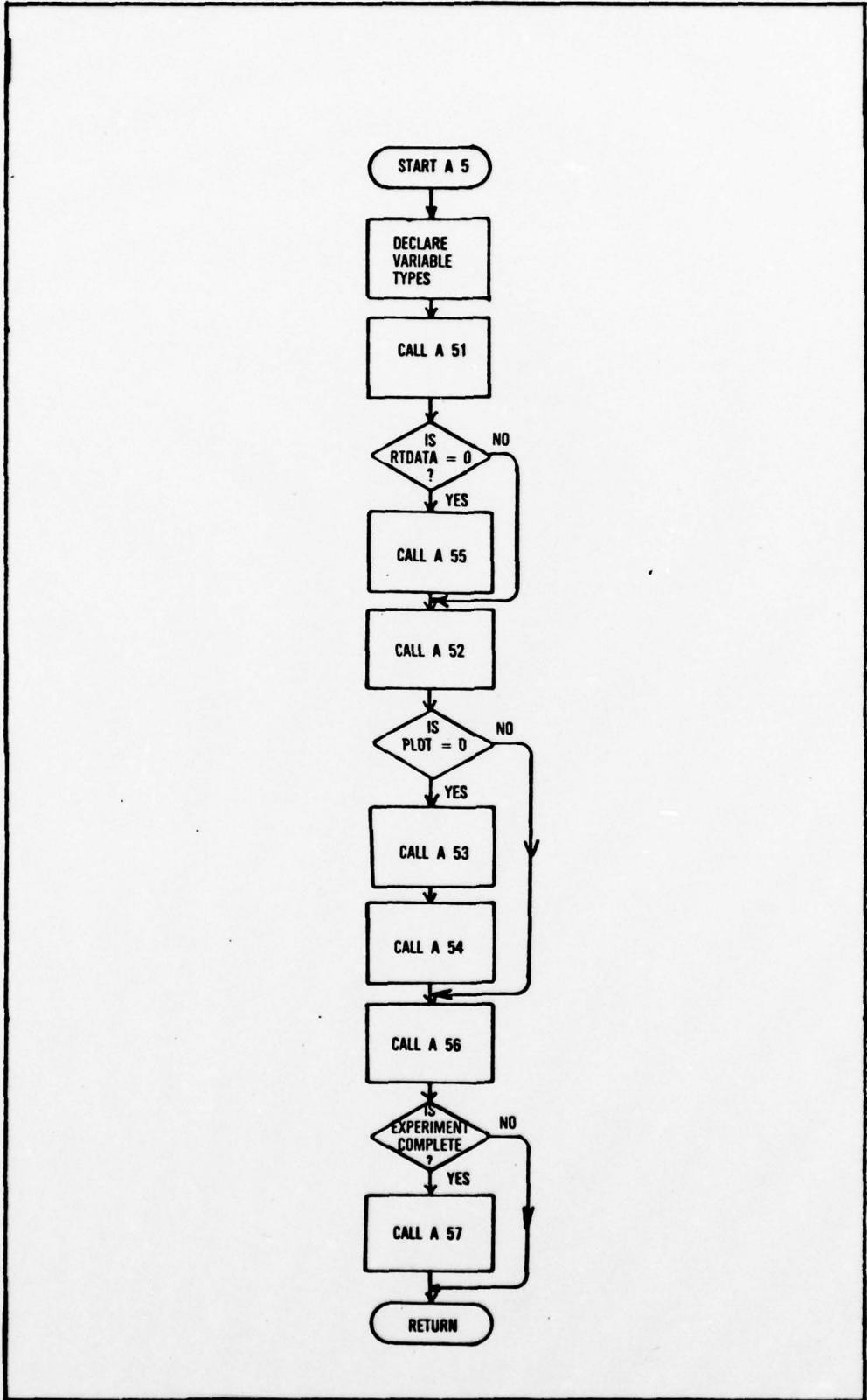


Figure 42. Flow Chart for Module A5

```

0001      SUBROUTINE A5
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C MODULE A5---REDUCE DATA
          C
          C   THIS MODULE IS RESPONSIBLE FOR REDUCING ALL THE DATA
          C GATHERED AND FOR WRITING IT, AND THE RESULTING PROCESSED
          C DATA, TO THE DISK STORAGE AND TO THE PRINTER. A5 IS THE EXECUTIVE
          C FOR THE MODULE AND CONTROLS SEVEN OTHER MODULES.
          C
          C   A51---COMPUTE TEMPERATURES
          C
          C   A52---COMPUTE DATA OUTPUT
          C
          C   A53---PRINT PLOT DATA
          C
          C   A54---PLOT DATA
          C
          C   A55---PRINT REAL TIME DATA
          C
          C   A56---WRITE DATA ARRAYS TO DISK
          C
          C   A57---PRINT OUTPUT DATA ARRAY
          C
          C   WHEN FINISHED, THE DATA PROCESSING COMPLETE SIGNAL ALLOW THE NEXT
          C BLOCK OF DATA TO BE GATHERED.
          C
          C
          C   AUTHOR: CAPTAIN EDGAR A. VERCHOT, JR., USAF
          C
          C
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C DATA SPECIFICATIONS
          C THESE ARE THE DATA SPECIFICATIONS FROM MODULE A1
          C
          C HEADER
0002      BYTE TITLE(20),TODAY(9)
0003      REAL ITEMP(100)
0004      INTEGER TYPTM
          C DATAIN
0005      REAL TEMP(100),FIELD,AULT(2,6)
0006      INTEGER NTEMP,ETYPE,NAVOLT,NDATPT
          C TCALIB
0007      BYTE THRMID(20)
0008      INTEGER NTEMP
0009      REAL TEMCAL(2,100)
          C SAMPLE
0010      BYTE SAMID(20)
0011      INTEGER SAMTYP
0012      REAL SAMT,SAMW,SAML
          C PLTOUT
0013      BYTE POPTS(6,11)

```

```

0014      INTEGER PLOTAB(4),PLOTOR(4),PLOT,NP
C EQPOUT
0015      BYTE EQUIPF(10,8)
0016      INTEGER IEIOF,EOFLAG(7)
C
C RELTIM
0017      INTEGER RTDATA
C
C
C
C
C THESE ARE THE DATA SPECIFICATIONS FOR A2
C
C CONSIG
0018      INTEGER SDP,EXPC,FSCOM,SCSCOM,ULTCOM
C
C A2COM
0019      INTEGER NTEM,SCSET,SCOUNT,UCCOUNT,RUN
0020      REAL TEMSET,FLDSET,ULTSET,X0
C THESE ARE THE DATA SPECIFICATIONS FOR A3
C THESE ARE THE DATA SPECIFICATIONS FOR A4
C RANDAT
0021      REAL TEMDAT(20),ULTDAT(20),SUDATA(20),FLDATA(20),IDATA(20)
0022      INTEGER SCDATA(20)
C THESE ARE THE DATA SPECIFICATIONS FOR A5
C DATOUT
0023      REAL RHO,P,MU,RH,F
0024      REAL TEMOUT(20),AUGTEM,DELTA,IATEM
0025      REAL R(4),R1R2,R3R4,R7,R8,UHALL,RMAG
0026      REAL R1,R2,R12,R5,R6,DELTAR,DELR5,DELR6
0027      REAL E,LN2,AFIELD,PI
C PLOTOR
0028      REAL PLOTS(4,2)
C THESE ARE THE COMMON BLOCKS FROM MODULE A1
C
0029      COMMON /HEADER/TITLE,ITEMP,TYPTEN,TODAY
0030      COMMON /DATAIN/NTEMPT,TEMP,ETYPE,FIELD,
2NAUOLT,AULT,NDATPT
0031      COMMON /TCALIB/THRMID,NTEMP,TEMCAL
0032      COMMON /SAMPLE/SAMID,SAMTYP,SAMT,SAMW,SAML
0033      COMMON /PLTOUT/PLOT,POPTS,NP,PLOTAB,PLOTOR
0034      COMMON /EQPOUT/EOFLAG,EQUIPF,IEIOF
0035      COMMON /RELTIM/RTDATA
C THE NAMED COMMON BLOCKS FOR A2CMN FOLLOW.
0036      COMMON /CONSIG/SDP,EXPC,FSCOM,SCSCOM,ULTCOM
0037      COMMON /A2COM/TEMSET,FLDSET,ULTSET,X0,NTEM,SCSET,SCOUNT,
2UCCOUNT,RUN
C THE COMMON BLOCKS FOR A3 FOLLOW
C THE COMMON BLOCKS FOR A4 FOLLOW
0038      COMMON /RANDAT/TEMDAT,ULTDAT,SUDATA,FLDATA,IDATA,SCDATA
C THE COMMON BLOCKS FOLLOW FOR A5
0039      COMMON /DATOUT/RHO,P,MU,RH,F,TEMOUT,AUGTEM,DELTA,IATEM,R,
2R1R2,R3R4,R7,R8,UHALL,RMAG,R1,R2,R12,R5,R6,DELTAR,DELR5,DELR6,

```

```

3E, LN2, AFIELD, PI
0040 COMMON /PLOTTER/PLOTS
C
C FIRST COMPUTE THE TEMPERATURES FROM THE DATA VOLTAGES.
C
C
0041 CALL A51
C
C CHECK IF REAL-TIME DATA PRINTOUT IS DESIRED
C
0042 IF<RTDATA.EQ.0> GO TO 52
0044 CALL A55<TEMOUT>
C
0045 52 CONTINUE
C THEN COMPUTE ALL OF THE OTHER NEEDED PARAMETERS
C
0046 CALL A52
C
C CHECK IF PLOTS ARE DESIRED
C
0047 IF<PLOT.EQ.0> GO TO 55
0049 CALL A53
0050 CALL A54
0051 55 CONTINUE
C
C WRITE THE DATA BLOCK TO DISK
C
0052 CALL A56
C
C CHECK IF EXPERIMENT IS COMPLETE
C
C IF IT IS PRINT THE OUTPUT DATA
C
0053 IF<EXPC.EQ.0> GO TO 900
0055 CALL A57
0056 900 CONTINUE
0057 RETURN
0058 END

```

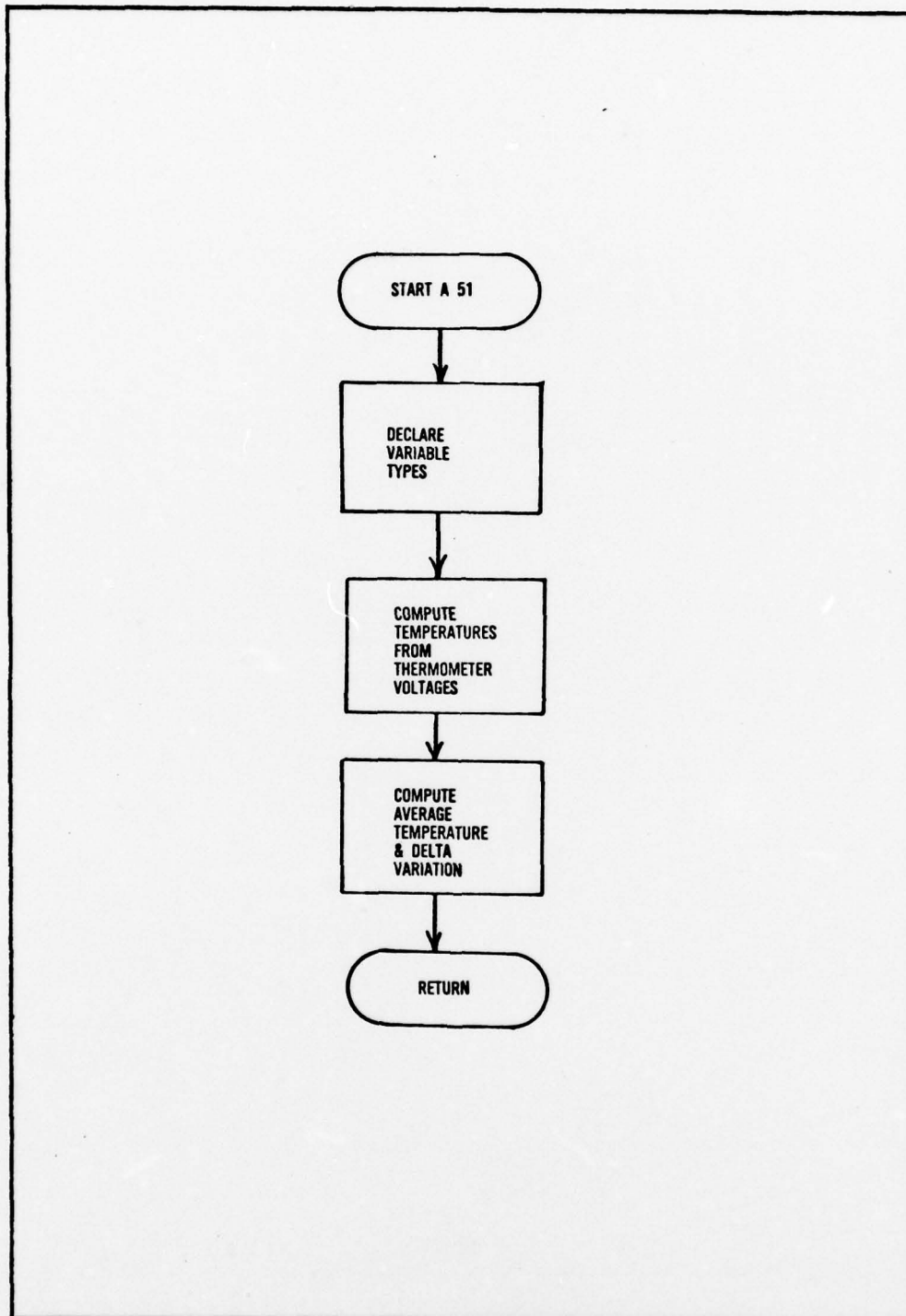


Figure 43. Flow Chart for Module A51

```

0001      SUBROUTINE A51
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C MODULE A51---COMPUTE TEMPERATURE
          C
          C THIS MODULE TAKES THE VOLTAGE RECORDED BY THE DATA ACQUISITION
          C MODULE FROM THE THERMOMETER AND CONVERTS IT TO THE CORRESPONDING
          C TEMPERATURES. THE MEAN AND THE DELTA FOR THESE DATA ARE THEN COMPUTED.
          C
          C
          C AUTHOR: CAPTAIN EDGAR A. VERCHOT, JR., USAF
          C
          C
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C DATA SPECIFICATIONS
          C
          C THESE ARE THE DATA SPECIFICATIONS FROM MODULE A1
          C DATAIN
0002      REAL TEMP(100),FIELD,AULT(2,6)
0003      INTEGER NTEMPT,ETYPE,NAVOLT,NDATPT
          C TCALIB
0004      BYTE THRMID(20)
0005      INTEGER NTEMP
0006      REAL TEMCAL(2,100)
          C THESE ARE THE DATA SPECIFICATIONS FOR A4
          C RANDAT
0007      REAL TEMDAT(20),ULTDAT(20),SUDATA(20),FLDATA(20),IDATA(20)
0008      INTEGER SCDATA(20)
          C THESE ARE THE DATA SPECIFICATIONS FOR A5
          C DATOUT
0009      REAL RHO,P,MU,RH,F
0010      REAL TEMOUT(20),AVGTEM,DELTA,IATEM
0011      REAL R(4),R1R2,R3R4,R7,R8,UHALL,RMAG
0012      REAL R1,R2,R12,R5,R6,DELTA,R,DELR5,DELR6
0013      REAL E,LN2,AFIELD,PI
          C THESE ARE THE COMMON BLOCKS FROM MODULE A1
0014      COMMON /DATAIN/NTEMPT,TEMP,ETYPE,FIELD,
          2NAVOLT,AULT,NDATPT
0015      COMMON /TCALIB/THRMID,NTEMP,TEMCAL
          C THE COMMON BLOCKS FOR A4 FOLLOW
0016      COMMON /RANDAT/TEMDAT,ULTDAT,SUDATA,FLDATA,IDATA,SCDATA
          C THE COMMON BLOCKS FOLLOW FOR A5
0017      COMMON /DATOUT/RHO,P,MU,RH,F,TEMOUT,AVGTEM,DELTA,IATEM,R,
          2R1R2,R3R4,R7,R8,UHALL,RMAG,R1,R2,R12,R5,R6,DELTA,R,DELR5,DELR6,
          3E,LN2,AFIELD,PI
          C
          C LOCAL VARIABLES
0018      REAL D(100),P1(100),X0,Y0,WDELTA,TEMSUM
0019      INTEGER NS,NF,NPT
0020      TEMSUM=0.0
0021      DELTA=0.0
          C COMPUTE THE TEMPERATURE EQUIVALENT TO THE VOLTAGE READINGS FOR

```

```

C THE THERMOMETER.
0022      DO 10 K=1,NDATPT
0023          X0=TEMOUT(K)
0024          DO 20 N=1,NTEMP
0025              IF(X0.GE.TEMCAL(2,N))GO TO 30
0027  20      CONTINUE
0028  30      CONTINUE
0029          NS=N-3
0030          IF(NS.LT.1)NS=1
0032          NF=N+3
0033          IF(NF.GT.NTEMP)NF=NTEMP
0035          Y0=0.0
0036          DO 40 I=NS,NF
0037              D(I)=1.0
0038              P1(I)=1.0
0039              DO 50 J=NS,NF
0040                  IF(I.EQ.J)GO TO 50
0042                      D(I)=D(I)*(TEMCAL(2,I)-TEMCAL(2,J))
0043                      P1(I)=P1(I)*(X0-TEMCAL(2,J))
0044  50      CONTINUE
0045          Y0=Y0+(P1(I)/D(I))*TEMCAL(1,I)
0046  40      CONTINUE
0047          TEMOUT(K)=Y0
C THIS PORTION OF THE PROGRAM COMPUTES THE MEAN TEMPERATURE
C FOR THE DATA BLOCK AND THE DELTA.
0048          TEMSUM=TEMSUM+TEMOUT(K)
0049  10      CONTINUE
0050          AVGTEM=TEMSUM/NDATPT
0051          IATEM=1000/AVGTEM
0052          DO 60 I=1,NDATPT
0053              WDELTA=AVGTEM-TEMOUT(I)
0054              IF(WDELTA.GT.DELTA)DELTA=WDELTA
0056  60      CONTINUE
0057          RETURN
0058          END

```

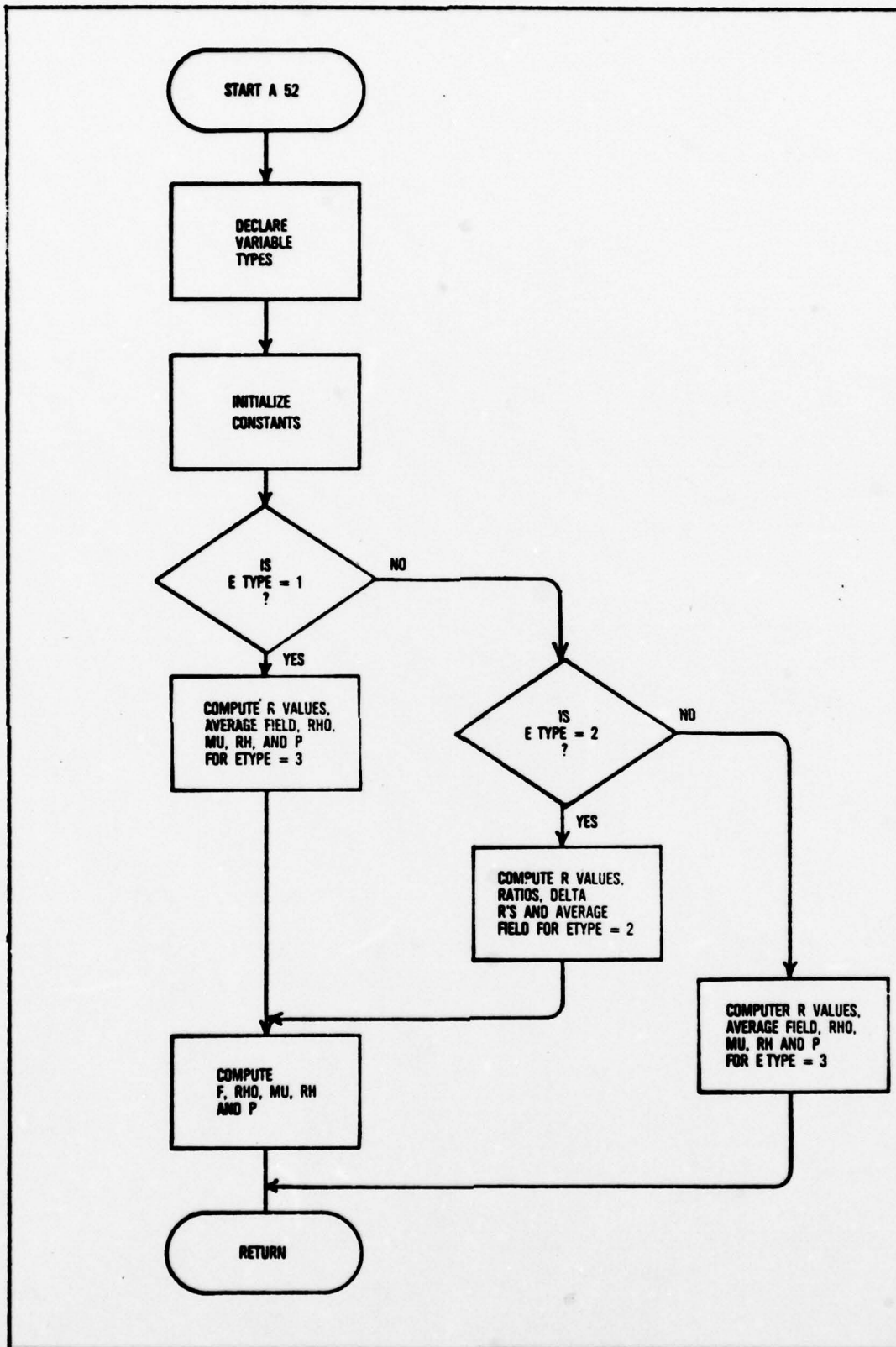


Figure 44. Flow Chart for Module A52

```

0001      SUBROUTINE A52
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C MODULE A52---COMPUTE DATA OUTPUT
          C
          C THIS MODULE COMPUTES ALL OF THE NEEDED OUTPUT DATA. THESE
          C ARE R(1),R(2),R(3),R(4),R1/R2,R3/R4,RHO,MU,P, AND RH.
          C DEFINITIONS FOLLOW:
          C R(1)=SU1/I1
          C R(2)=SU2/I2
          C R(3)=SU3/I3
          C R(4)=SU4/I4
          C R1/R2 AND R3/R4 ARE SELF EXPLANATORY.
          C RHO=RESISTIVITY
          C MU=HALL MOBILITY
          C P=CARRIER CONCENTRATION
          C RH=HALL COEFFICIENT
          C THESE DATA, COUPLED WITH THE TEMPERATURES FROM A51 ARE THE END RESULT
          C OF THE EXPERIMENT.
          C
          C
          C AUTHOR: CAPTAIN EDGAR A. VERCHOT, JR., USAF
          C
          C
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C DATA SPECIFICATIONS
          C THESE ARE THE DATA SPECIFICATIONS FOR A1
          C DATAIN
0002      REAL TEMP(100),FIELD,AULT(2,6)
0003      INTEGER NTEMPT,ETYPE,NAVOLT,NDATPT
          C SAMPLE
0004      BYTE SAMID(20)
0005      INTEGER SAMTYP
0006      REAL SANT,SAMW,SAML
          C THESE ARE THE DATA SPECIFICATIONS FOR A2
          C A2COM
0007      INTEGER NTEM,SCSET,SCOUNT,VCOUNT,RUN
0008      REAL TEMSET,FLDSET,ULTSET,X0
          C THESE ARE THE DATA SPECIFICATIONS FOR A5
          C DATOUT
0009      REAL RHO,P,MU,RH,F
0010      REAL TEMOUT(20),AUGTEM,DELTA,IATEM
0011      REAL R(4),R1R2,R3R4,R7,R8,VHALL,RMAG
0012      REAL R1,R2,R12,R5,R6,DELTA,R,DELR5,DELR6
0013      REAL E,LN2,AFIELD,PI
          C THESE ARE THE DATA SPECIFICATIONS FOR A4
          C
          C RANDAT
0014      REAL TEMDAT(20),ULTDAT(20),SUDATA(20),FLDATA(20),IDATA(20)
0015      INTEGER SCDATA(20)
          C THE COMMON BLOCKS FOR A1 FOLLOW.
0016      COMMON /DATAIN/NTEMPT,TEMP,ETYPE,FIELD,
          2NAVOLT,AULT,NDATPT
0017      COMMON /SAMPLE/SAMID,SAMTYP,SANT,SAMW,SAML

```

```

0018 C THE COMMON BLOCKS FOR A2 FOLLOW
COMMON /A2COM/TEMSET,FLDSET,ULTSET,X0,NTEM,SCSET,SCOUNT,
ZUCOUNT,RUN
0019 C THE COMMON BLOCKS FOR A4 FOLLOW
COMMON /RAWDAT/TEMDAT,ULTDAT,SUDATA,FLDATA,IDATA,SCDATA
0020 C THE COMMON BLOCKS FOLLOW FOR A5
COMMON /DATOUT/RHO,P,MU,RH,F,TEMOUT,AUGTEM,DELTA,IATEM,R,
2R1R2,R3R4,R7,R8,UHALL,RMAG,R1,R2,R12,R5,R6,DELTAR,DELR5,DELR6,
3E,LN2,AFIELD,PI
0021 C INITIALIZE NECESSARY PARAMETERS
RMAG=0.0
0022 AFIELD=0.0
0023 LN2=ALOG(2.)
0024 E=1.6021917E-19
0025 PI=3.1415827
0026 C FIRST FIND OUT THE EXPERIMENT TYPE
IF(ETYPE.NE.1) GO TO 20
C DO THE COMPUTATIONS FOR THE VAN DER PAUW SAMPLE TYPE
C COMPUTE THE RESISTANCE VALUES
0028 DO 10 I=1,4
0029 R(I)=(ABS(SUDATA(2*I-1)/IDATA(2*I-1))+ABS(SUDATA(2*I)/
2IDATA(2*I)))/2
0030 10 CONTINUE
C COMPUTE THE R RATIOS
0031 R1R2=R(1)/R(2)
0032 R3R4=R(3)/R(4)
C SET THE WORKING PARAMETERS NEEDED FOR LATER CALCULATION
C R1 IS THE AVERAGE OF R(1) AND R(3)
C R2 IS THE AVERAGE OF R(2) AND R(4)
C R12 IS THE RATIO OF R1/R2
0033 R1=(R(1)+R(3))/2
0034 R2=(R(2)+R(4))/2
0035 R12=R1/R2
C COMPUTE DELTA R
C DELTAR IS THE AVERAGE OF DELTA R5 AND DELTA R6
0036 DELR5=ABS((SUDATA(13)/IDATA(13)-SUDATA(14)/IDATA(14))/4)
2+ABS((SUDATA(17)/IDATA(17)-SUDATA(18)/IDATA(18))/4)
0037 DELR6=ABS((SUDATA(15)/IDATA(15)-SUDATA(16)/IDATA(16))/4)
2+ABS((SUDATA(19)/IDATA(19)-SUDATA(20)/IDATA(20))/4)
0038 DELTAR=(DELR5+DELR6)/2
C COMPUTE THE AVERAGE FIELD
0039 DO 11 I=13,20
0040 AFIELD=AFIELD+ABS(FLDATA(I))/8
0041 11 CONTINUE
0042 GO TO 40
0043 20 CONTINUE
0044 IF(ETYPE.NE.2) GO TO 30
C COMPUTE THE RESISTANCE VALUES FOR ETYPE 2.
0046 DO 21 I=1,2
0047 R(I)=(ABS(SUDATA(2*I-1)/IDATA(2*I-1))+ABS(SUDATA(2*I)/
2IDATA(2*I)))/2
0048 21 CONTINUE
C COMPUTE THE R RATIOS

```

```

0049      R1=R(1)
0050      R2=R(2)
0051      IF(SCOUNT.EQ.0)GO TO 25
0053      R1R2=R(1)/R(2)
0054      R3R4=0.0
0055      R12=R1R2
0056      GO TO 26
0057 25    CONTINUE
0058      R3R4=R(1)/R(2)
0059      R1R2=0.0
0060      R12=R3R4
0061 26    CONTINUE
          C      COMPUTE THE DELTA R
0062      DELTAR=ABS((SUDATA(7)/IDATA(7)-SUDATA(8)/IDATA(8))/4)
          2+ABS((SUDATA(9)/IDATA(9)-SUDATA(10)/IDATA(10))/4)
          C      COMPUTE THE AVERAGE FIELD
0063      DO 27 I=7,10
0064      AFIELD=AFIELD+ABS(FLDATA(I))/4
0065 27    CONTINUE
0066      GO TO 40
0067 40    CONTINUE
          C      COMPUTE THE VAN DER PAUW F VALUE
0068      F=1-(((R1-R2)/(R1+R2))**2)*(LN2/2.))-(((R1-R2)/(R1+R2))**4)
          2*((LN2**2/4.)-(LN2**3/12.))
          C      NOW FIND RHO, THE RESISTIVITY
0069      RHO=(PI*SAMT/LN2)*((R1+R2)/2.)*F
          C      NOW FIND MU, THE HALL MOBILITY
0070      MU=((DELTAR*SAMT)/(RHO*AFIELD))*1.0E+5
          C      NOW FIND P, THE CARRIER CONCENTRATION
0071      P=1./(MU*E*RHO)
          C      NOW FIND RH, THE HALL COEFFICIENT.
0072      RH=MU*RHO
0073      GO TO 900
0074 30    CONTINUE
          C NOW DO THE CALCULATIONS FOR THE HALL BAR, EXPERIMENT TYPE 3
          C      FIRST FIND THE RESISTANCE
0075      R7=(ABS(SUDATA(1)/IDATA(1))+ABS(SUDATA(2)/IDATA(2)))/2
          C      NOW FIND RHO, THE RESISTIVITY
0076      RHO=SAMW**SANT*R7/SANL
          C      FIND THE AVERAGE FIELD
0077      DO 31 I=5,8
0078      AFIELD=AFIELD+ABS(FLDATA(I))/4
0079 31    CONTINUE
          C      FIND U'/I'
0080      DO 32 I=5,8,2
0081      RMAG=RMAG+(SUDATA(I)/IDATA(I)-SUDATA(I+1)/IDATA(I+1))/2
0082 32    CONTINUE
          C      NOW FIND THE HALL MOBILITY, MU
0083      MU=(SANL/(AFIELD*SAMW))*((RMAG/R7)*1.0E+5)
          C      NEXT FIND P, THE CARRIER CONCENTRATION
0084      P=AFIELD/(E*SAMT*RMAG)
          C      NOW FIND THE HALL COEFFICIENT, RH
0085      RH=MU*RHO

```

0086 900 CONTINUE
0087 RETURN
0088 END

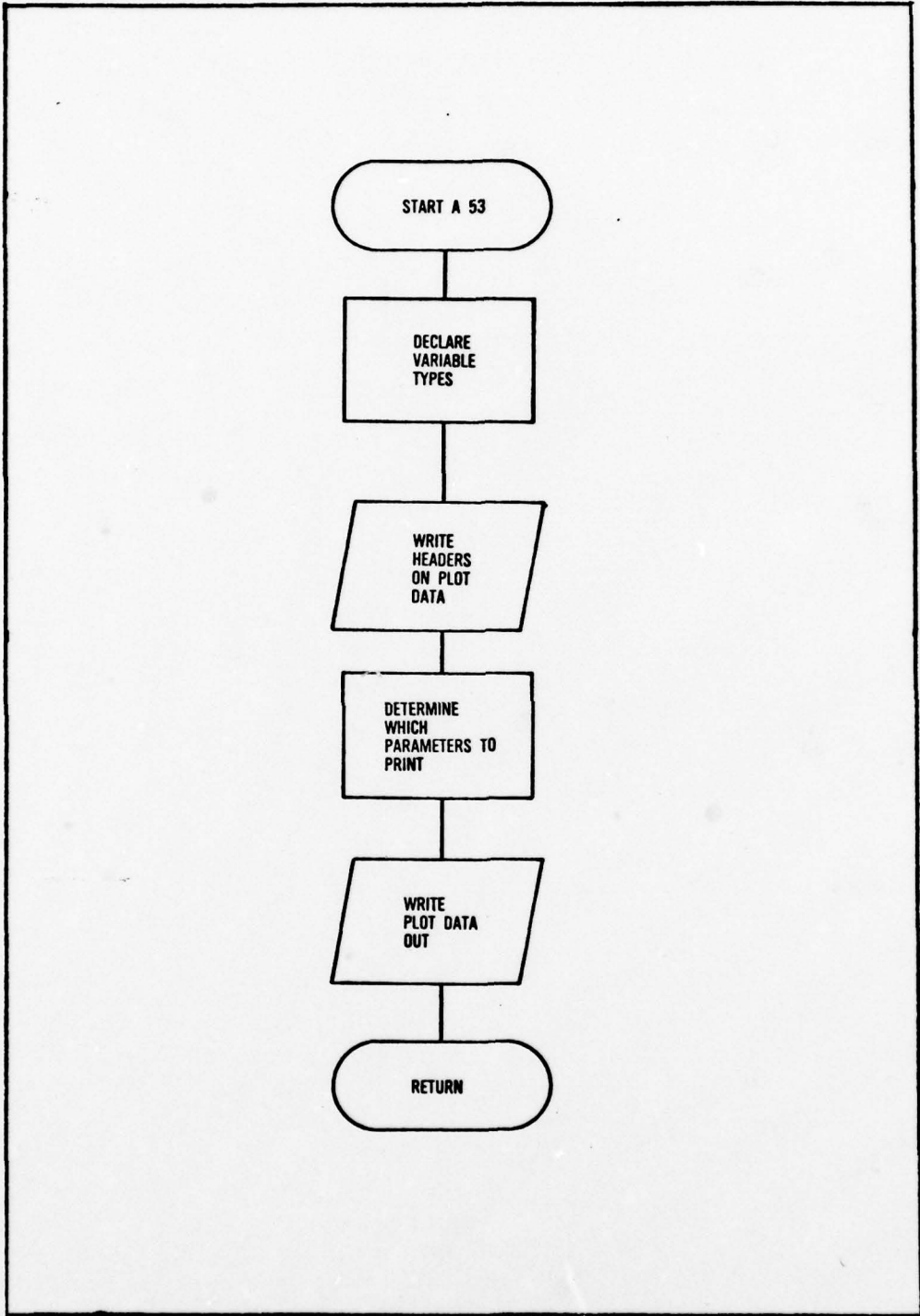


Figure 45. Flow Chart for Module A53

```

0001      SUBROUTINE A53
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C MODULE A53—PRINT PLOT DATA
          C
          C THIS MODULE PRINTS THE PLOT DATA ON THE TERMINAL
          C WHENEVER THE PLOT FLAG IS SET TRUE. IT THEN PASSES THE DATA TO
          C THE PLOT MODULE. IF THE PLOT FLAG IS FALSE THIS MODULE DOES
          C NOT EXECUTE.
          C
          C AUTHOR: CAPTAIN EDGAR A. VERCHOT, JR., USAF
          C
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C DATA SPECIFICATIONS
          C THESE ARE THE DATA SPECIFICATIONS FROM MODULE A1
          C PLTOUT
0002      BYTE POPTS(6,11)
0003      INTEGER PLOTAB(4),PLOTOR(4),PLOT,NP
          C THESE ARE THE DATA SPECIFICATIONS FOR A5
          C PLOTOR
0004      REAL PLOTS(4,2)
          C THESE ARE THE COMMON BLOCKS FROM MODULE A1
0005      COMMON /PLTOUT/PLOT,POPTS,NP,PLOTAB,PLOTOR
          C THESE ARE THE COMMON BLOCKS FOR A5
          C THESE ARE THE DATA SPECIFICATIONS FOR A5
          C DATOUT
0006      REAL RHO,P,MU,RH,F
0007      REAL TEMOUT(20),AVGTEM,DELTA,IATEM
0008      REAL R(4),R1R2,R3R4,R7,R8,UHALL,RMAG
0009      REAL R1,R2,R12,R5,R6,DELTAR,DELR5,DELR6
0010      REAL E,LN2,AFIELD,PI
          C THESE ARE THE DATA SPECIFICATIONS FOR A4
          C RAWDAT
0011      REAL TEMDAT(20),ULTDAT(20),SUDATA(20),FLDATA(20),IDATA(20)
0012      INTEGER SCDATA(20)
          C THE COMMON BLOCKS FOR A4 FOLLOW
0013      COMMON /RAWDAT/TEMDAT,ULTDAT,SUDATA,FLDATA,IDATA,SCDATA
          C THE COMMON BLOCKS FOLLOW FOR A5
0014      COMMON /DATOUT/RHO,P,MU,RH,F,TEMOUT,AVGTEM,DELTA,IATEM,R,
          2R1R2,R3R4,R7,R8,UHALL,RMAG,R1,R2,R12,R5,R6,DELTAR,DELR5,DELR6,
          3E,LN2,AFIELD,PI
0015      COMMON /PLOTOR/PLOTS
          C LOCAL VARIABLES
0016      INTEGER HOLD(2)
          C NOW PRINT THE DESIRED DATA
0017      DO 1 I=1,NP
0018          WRITE(7,1000) (POPTS(L,PLOTAB(I)),L=1,6),
          2(POPTS(K,PLOTOR(I)),K=1,6)
0019 1000      FORMAT(1X,6A1,7X,6A1)
0020          HOLD(1)=PLOTAB(I)
0021          HOLD(2)=PLOTOR(I)
0022          DO 200 J=1,2
0023              K=HOLD(J)

```

```

0024          GO TO (10,20,30,40,50,60,70,80,90,100),K
0025          WRITE(7,800)
0026 800      FORMAT(/' ERROR IN THE PLOT MODULE')
0027          GO TO 900
0028 10      CONTINUE
0029          PLOTS(I,J)=RUGTEM
0030          GO TO 200
0031 20      CONTINUE
0032          PLOTS(I,J)=IATEM
0033          GO TO 200
0034 30      CONTINUE
0035          PLOTS(I,J)=RHO
0036          GO TO 200
0037 40      CONTINUE
0038          PLOTS(I,J)=MU
0039          GO TO 200
0040 50      CONTINUE
0041          PLOTS(I,J)=P
0042          GO TO 200
0043 60      CONTINUE
0044          PLOTS(I,J)=DELTA
0045          GO TO 200
0046 70      CONTINUE
0047          PLOTS(I,J)=R1R2
0048          GO TO 200
0049 80      CONTINUE
0050          PLOTS(I,J)=R3R4
0051          GO TO 200
0052 90      CONTINUE
0053          PLOTS(I,J)=RH
0054          GO TO 200
0055 100     CONTINUE
0056          PLOTS(I,J)=R12
0057 200     CONTINUE
0058          WRITE(7,2000)(PLOTS(I,J),J=1,2)
0059 2000    FORMAT(1X,G15.8,3X,G15.8)
0060 1        CONTINUE
          C CALL THE PLOTTER
0061          CALL A54
0062 900     CONTINUE
0063          RETURN
0064          END

```

```
0001      SUBROUTINE A54
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C MODULE A54---PLOT DATA
          C
          C NO PLOTTER HAS YET BEEN PURCHASED.  THE APPROPRIATE DRIVER FOR IT
          C WILL BE ENTERED HERE WHEN IT IS.
          C
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002      RETURN
0003      END
```

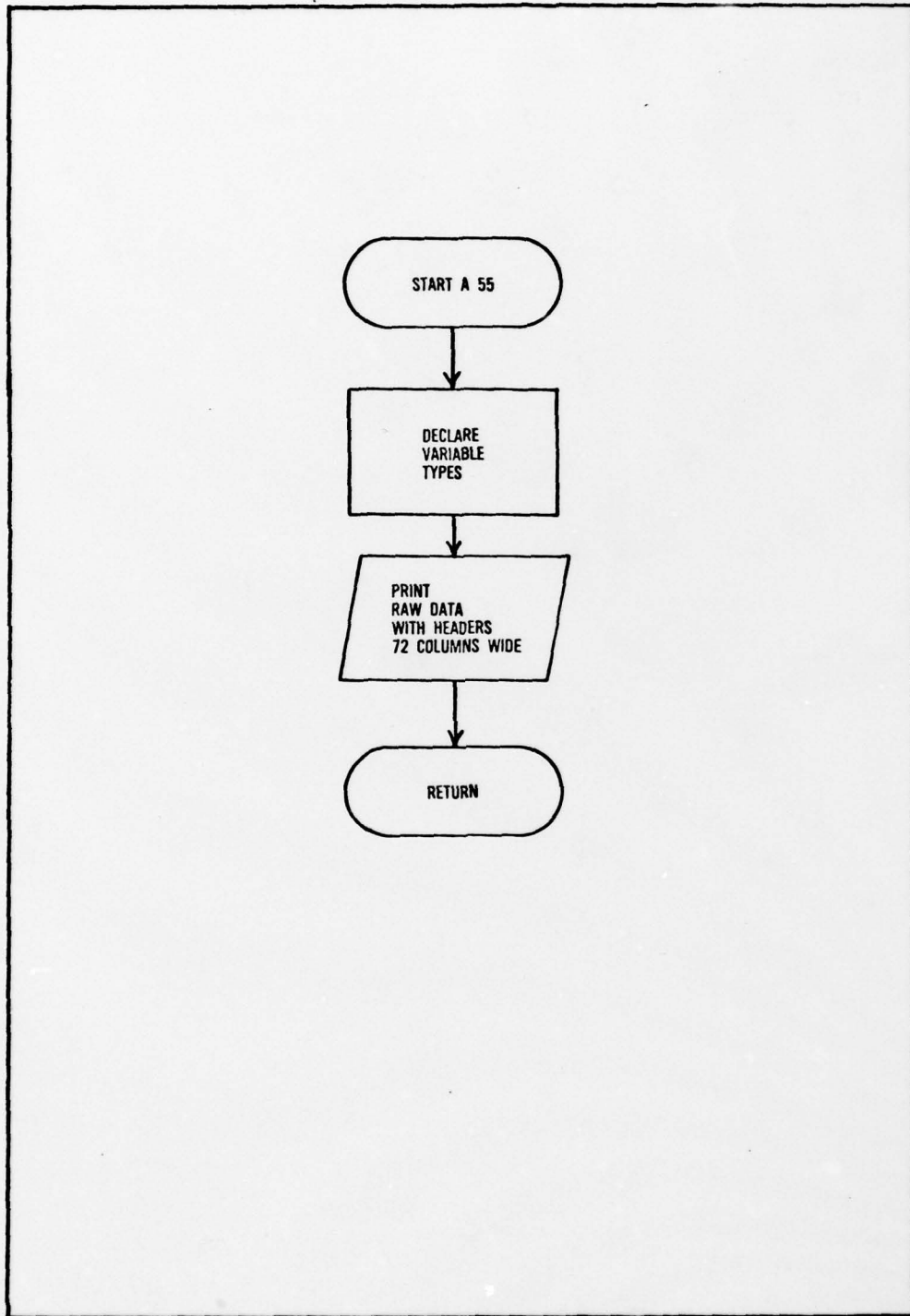


Figure 46. Flow Chart for Module A55

```

0001      SUBROUTINE A55(TEMOUT)
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C MODULE A55---PRINT REAL-TIME DATA
          C
          C     THIS MODULE TAKES THE RAW DATA AND IF THE RTDATA FLAG IS SET
          C TRUE, PRINTS IT OUT TO THE TEEMINAL AFTER EACH BLOCK OF DATA
          C IS GATHERED.
          C
          C     AUTHOR:  CAPTAIN EDGAR A. VERCHOT, JR., USAF
          C
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C DATA SPECIFICATIONS
          C
          C
          C THESE ARE THE DATA SPECIFICATIONS FROM MODULE A1
          C
          C HEADER
0002      BYTE TITLE(20),TODAY(9)
0003      REAL ITEMP(100)
0004      INTEGER TYPTM
          C TCALIB
0005      BYTE THRMID(20)
0006      INTEGER NTEMP
0007      REAL TEMCAL(2,100)
          C SAMPLE
0008      BYTE SAMID(20)
0009      INTEGER SAMTYP
0010      REAL SAMT,SAMW,SANL
          C     THESE ARE THE DATA SPECIFICATIONS FOR A2
          C A2COM
0011      INTEGER NTEM,SCSET,SCOUNT,UCOUNT,RUN
0012      REAL TEMSET,FLDSET,ULTSET,X0
          C THESE ARE THE COMMON BLOCKS FROM MODULE A1
          C
0013      COMMON /HEADER/TITLE,ITEMP,TYPTM,TODAY
0014      COMMON /TCALIB/THRMID,NTEMP,TEMCAL
0015      COMMON /SAMPLE/SAMID,SAMTYP,SAMT,SAMW,SANL
          C THE NAMED COMMON BLOCKS FOR A2COM FOLLOW.
0016      COMMON /A2COM/TEMSET,FLDSET,ULTSET,X0,NTEM,SCSET,SCOUNT,
          2UCOUNT,RUN
          C THESE ARE THE DATA SPECIFICATIONS FOR A4 COMMON
          C RAUDAT
0017      REAL TEMDAT(20),ULTDAT(20),SUDATA(20),FLDATA(20),IDATA(20)
0018      INTEGER SCDATA(20)
          C THE COMMON BLOCKS FOR A4 FOLLOW
0019      COMMON /RAUDAT/TEMDAT,ULTDAT,SUDATA,FLDATA,IDATA,SCDATA
          C
0020      REAL TEMOUT(20)
          C
          C
          C PRINT THE DATA 72 COLUMNS WIDE
          C (IF NTEM.NE.2) GO TO 100

```

```

0023      WRITE(7,1000)(TODAY(I),I=1,9),(TITLE(J),J=1,19)
0024 1000  FORMAT(' ',DATE: ',9A1,/,',TITLE: ',19A1)
0025      WRITE(7,1010)(THRMID(K),K=1,19),(SAMID(L),L=1,19)
0026 1010  FORMAT(' ',THERMOMETER IDENTIFIER: ',/,20A1,/,
2         ' SAMPLE IDENTIFIER: ',20A1)
0027      IF(SAMTYP.NE.0)GO TO 110
0029      WRITE(7,1100) SAMT
0030 1100  FORMAT(' ',SAMPLE DIMENSIONS ARE: ',/,
2         ' THICKNESS= ',G13.7,' CENTIMETERS')
0031      GO TO 100
0032 110   CONTINUE
0033      WRITE(7,1110) SAMT,SAMW,SAML
0034 1110  FORMAT(' ',SAMPLE DIMENSIONS ARE: ',/,
2         ' SAMPLE THICKNESS= ',G13.7,' CENTIMETERS',/,
3         ' SAMPLE WIDTH= ',G13.7,' CENTIMETERS',/,
4         ' SAMPLE LENGTH= ',G13.7,' CENTIMETERS')
0035 100   CONTINUE
0036      WRITE(7,2000)
0037 2000  FORMAT(' ',TEMPERATURE ',',APPLIED ',',SAMPLE ',
2' FIELD ',',SAMPLE ',',CURRENT ',/,
315X,' VOLTAGE ',',VOLTAGE ',12X,' CONFIGURATION',/,
4' (DEGREE K) ',', (VOLTS) ',', (VOLTS) ',', (KGAUSS) ',
512X,' (AMPERES) ',/)
0038      DO 200 I=1,RUN
0039      WRITE(7,2010) TEMOUT(I),ULTDAT(I),SUDATA(I),FLDATA(I),
2SCDATA(I),IDATA(I)
0040 2010  FORMAT(' ',1X,G12.5,3G12.5,5X,12.5X,G14.7)
0041 200   CONTINUE
0042      RETURN
0043      END

```

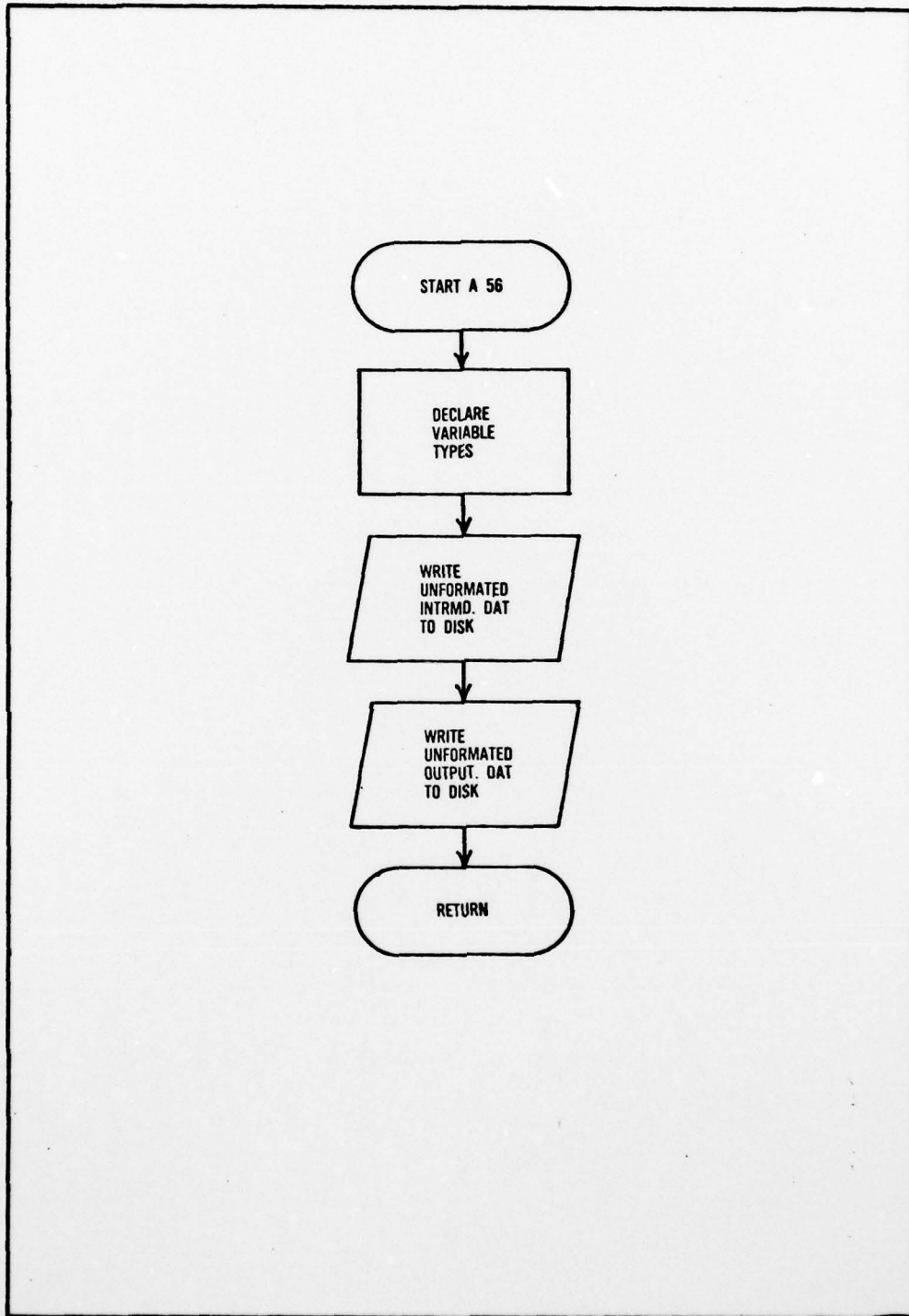


Figure 47. Flow Chart for Module A56

```

0001      SUBROUTINE A56
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C MODULE A56—WRITE DATA ARRAYS TO THE DISK FILES
          C
          C THIS MODULE TAKES THE RAW DATA AND THE PROCESSED DATA AND
          C WRITES IT TO THE DISK STORAGE USING UNFORMATTED WRITE STATEMENTS.
          C IT CAN BE RECALLED USING UNFORMATTED READ STATEMENTS. THREE FILES
          C ARE USED:
          C 1. RAWOUT.DAT
          C 2. INTRND.DAT
          C 3. OUTPUT.DAT
          C THESE FILES MUST NOT EXIST ON THE STORAGE DISK PRIOR TO BEING
          C CREATED BY THIS PROGRAM. IF THEY EXIST THEY MUST BE DELETED OR
          C A FRESH DISK SUBSTITUTED. WHEN FINISHED THE MODULE RETURNS
          C UNLESS THE EXPERIMENT COMPLETE FLAG IS SET TRUE. IF IT IS
          C CONTROL PASSED TO A57.
          C
          C AUTHOR: CAPTAIN EDGAR A. VERCHOT, JR., USAF
          C
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C DATA SPECIFICATIONS
          C THESE ARE THE DATA SPECIFICATIONS FOR A2
          C A2COM
0002      INTEGER NTEM, SCSET, SCOUNT, UCOUNT, RUN
0003      REAL TEMSET, FLDSET, ULTSET, X0
          C THESE ARE THE DATA SPECIFICATIONS FOR A4 COMMON
          C RANDAT
0004      REAL TEMDAT(20), ULTDAT(20), SUDATA(20), FLDATA(20), IDATA(20)
0005      INTEGER SCDATA(20)
          C THESE ARE THE DATA SPECIFICATIONS FOR A5
          C DATOUT
0006      REAL RHO, P, MU, RH, F
0007      REAL TEMOUT(20), AUGTEM, DELTA, IATEM
0008      REAL R(4), R1R2, R3R4, R7, R8, UHALL, RMAG
0009      REAL R1, R2, R12, R5, R6, DELTAR, DELR5, DELR6
0010      REAL E, LN2, AFIELD, PI
          C THE COMMON BLOCKS FOR A4 FOLLOW
0011      COMMON /RANDAT/TEMDAT, ULTDAT, SUDATA, FLDATA, IDATA, SCDATA
          C THE COMMON BLOCKS FOLLOW FOR A5
0012      COMMON /DATOUT/RHO, P, MU, RH, F, TEMOUT, AUGTEM, DELTA, IATEM, R,
          2R1R2, R3R4, R7, R8, UHALL, RMAG, R1, R2, R12, R5, R6, DELTAR, DELR5, DELR6,
          3E, LN2, AFIELD, PI
          C THE NAMED COMMON BLOCKS FOR A2CMN FOLLOW.
0013      COMMON /A2COM/TEMSET, FLDSET, ULTSET, X0, NTEM, SCSET, SCOUNT,
          2UCOUNT, RUN
          C
          C
          C WRITE RAWOUT.DAT
0014      DO 10 I=1, RUN
0015      WRITE(1) TEMOUT(I), ULTDAT(I), SUDATA(I), FLDATA(I),
          2 SCDATA(I), IDATA(I)

```

```
0016 10  CONTINUE
      C  WRITE INTRND.DAT
0017      WRITE(2) AUGTEM,DELTA,R1R2,R3R4,R12
0018      IUNIT=ILUN(3)
      C WRITE OUTPUT DATA
0019      WRITE(3) AUGTEM,IATEM,RHO,MU,P,RH
0020      I=IUNIT(IUNIT)
0021      RETURN
0022      END
```

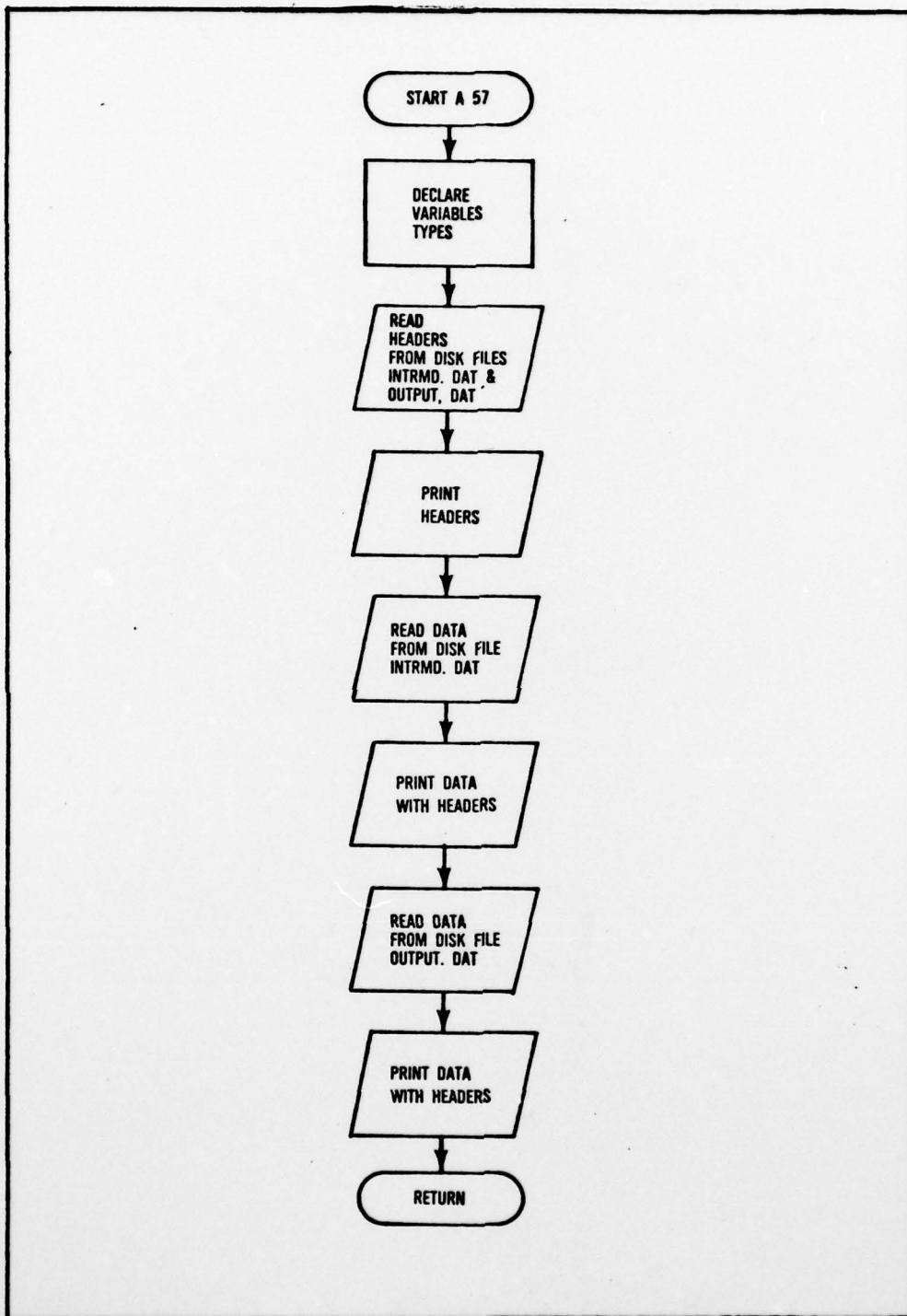


Figure 48. Flow Chart for Module A57

```

0001      SUBROUTINE A57
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C MODULE A57—PRINT OUTPUT DATA ARRAYS
          C
          C   THIS MODULE OUTPUTS THE CONTENTS OF THE TWO PROCESSED
          C DATA ARRAY FILES:
          C   INTRMD.DAT AND
          C   OUTPUT.DAT
          C WHEN THE EXPERIMENT IS COMPLETE.
          C
          C
          C   AUTHOR:  CAPTAIN EDGAR A. VERCHOT, JR., USAF
          C
          C
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C DATA SPECIFICATIONS
          C LOCAL VARIABLES
          C
0002      BYTE TODAY(9),TITLE(20),THRMID(20),SAMID(20)
0003      INTEGER SAMTYP
0004      REAL SAMT,SAMW,SAML
          C THESE ARE THE DATA SPECIFICATIONS FROM MODULE A1
          C EQPOUT
0005      BYTE EQUIPF(10,8)
0006      INTEGER IEIOF,EOFLAG(7)
          C THESE ARE THE DATA SPECIFICATIONS FOR A2
          C A2CMN
0007      INTEGER NTEM,SCSET,SCOUNT,UCOUNT,RUN
0008      REAL TEMSET,FLDSET,ULTSET,X0
          C THESE ARE THE COMMON BLOCKS FROM MODULE A1
0009      COMMON /EQPOUT/EOFLAG,EQUIPF,IEIOF
          C THE NAMED COMMON BLOCKS FOR A2CMN FOLLOW.
0010      COMMON /A2CMN/TEMSET,FLDSET,ULTSET,X0,NTEM,SCSET,SCOUNT,
          2UCOUNT,RUN
          C
          C
          C REWIND THE FILES
0011      REWIND 2
0012      REWIND 3
          C READ THE HEADER INFORMATION AND PRINT IT OUT
0013      DO 10 I=2,3
0014          READ(I) (TODAY(J),J=1,9)
0015          READ(I) (TITLE(K),K=1,19)
0016          READ(I) (THRMID(L),L=1,19)
0017          READ(I) (SAMID(M),M=1,20)
0018          READ(I) SAMTYP
0019          IF (SAMTYP.NE.0) GO TO 100
0021          READ(I) SAMT
0022          GO TO 11
0023      100  CONTINUE
0024          READ(I) SAMT,SAMW,SAML

```

```

0025 11 CONTINUE
      C THROW AWAY THE REDUNDANT INFORMATION
0026 IF(I.NE.2) GO TO 10
0028 WRITE(7,1000)(TODAY(J),J=1,9),(TITLE(K),K=1,19),
      2 (THRMID(L),L=1,19),(SAMID(M),M=1,19)
0029 1000 FORMAT(' ',1X,9A1,/,1X,19A1,/,1X,19A1,/,1X,19A1,/)
0030 IF(SAMTYP.NE.0) GO TO 200
0032 WRITE(7,1100) SAMT
0033 1100 FORMAT(' ', ' SAMPLE THICKNESS = ',G14.7,
      2 ' CENTIMETERS')
0034 GO TO 10
0035 200 CONTINUE
0036 WRITE(7,2000) SAMT,SAMW,SAML
0037 2000 FORMAT(' ', ' SAMPLE THICKNESS = ',G14.7,
      2 ' CENTIMETERS',/, ' SAMPLE WIDTH = ',G14.7,
      3 ' CENTIMETERS',/, ' SAMPLE LENGTH = ', G14.7,
      4 ' CENTIMETERS')

0038 10 CONTINUE
0039 IF(IEIOF.EQ.0) GO TO 21
0041 WRITE(7,1900)
0042 1900 FORMAT(' ', ' THE FOLLOWING AUTOMATIC EQUIPMENT WAS
      2 INOPERATIVE:')
0043 DO 20 I=1,7
0044 IF(EOFLAG(I).NE.1) GO TO 20
0046 WRITE(7,2020) (EQUIPF(J,I),J=1,10)
0047 2020 FORMAT(' ',10A1,///)
0048 20 CONTINUE
0049 21 CONTINUE
      C
      C PRINT THE INTERMEDIATE ARRAY
0050 WRITE(7,3000)
0051 3000 FORMAT(' ', ' TEMPERATURE ', ' DELTA ',
      2' R1/R2 ', ' R3/R4 ', ' AVG R1/R2 ')
0052 DO 30 I=1,100
0053 READ(2,END=35) A,D,R1,R3,R12
0054 WRITE(7,3030) A,D,R1,R3,R12
0055 3030 FORMAT(' ',1X,5G12.5)
0056 30 CONTINUE
0057 35 CONTINUE
      C NOW PRINT THE OUTPUT DATA ARRAY
0058 WRITE(7,4000)
0059 4000 FORMAT(' ', ' TEMPERATURE ', ' 1/T ',
      2' RESISTIVITY ', ' HALL ', ' CARRIER ',
      3' HALL ',/,36X, ' MOBILITY ', ' CONCENTRATION',
      4' COEFFICIENT')
0060 DO 40 I=1,100
0061 READ(3,END=45) A,T,R,U,P,RH
0062 WRITE(7,4040) A,T,R,U,P,RH
0063 4040 FORMAT(' ',1X,6G12.5)
0064 40 CONTINUE
0065 45 CONTINUE
0066 CLOSE(UNIT=1)
0067 CLOSE(UNIT=2)

```

0068
0069
0070

CLOSE(UNIT=3)
RETURN
END

```

0001      FUNCTION GAUSS(IOULD)
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C THIS IS THE GAUSS METER DRIVER PROVIDED BY MR. DANE HANBY, UDRI. THE
          C GAUSSMETER MUST BE SET TO THE 10K GAUSS SCALE FOR THE VALUE TO BE
          C CORRECT. THERE IS NO AUTOMATED CONTROL OF THE METER. IT SIMPLY PASSES
          C THE DATA OUT. FURTHERMORE IT DOES NOT PASS OUT THE SCALE THAT IT
          C IS SET TO. THEREFORE IT MUST BE MANUALLY SET TO THE 10K GAUSS SCALE)
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002      INTEGER CRS1,OUT1,INI
0003      DATA CRS1/"167770"/,OUT1/"167772"/,INI/"167774"/
0004      IOULD=0
0005      10 IWORD=IPEEK(INI)
0006      IF((IWORD .AND. "20000) .EQ. 0) GOTO 10
0008      IF((IWORD .AND. "40000) .NE. 0) IOULD=1
0010      IVAL=IBCD(IWORD .AND. "17777)
0011      IF((IWORD .AND. "100000) .NE. 0) IVAL=-IVAL
0013      GAUSS=0.1*IVAL
0014      RETURN
0015      END

```

```

0001      INTEGER FUNCTION TSTCON(ISAM,IPOL)
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          CC
          C THIS MODULE CONTROLS THE SAMPLE CONFIGURATION SETTING VIA THE UDRI
          C TEST CONTROLLER. THIS DRIVER MODULE WAS WRITTEN BY MR. DANE HANBY, UDRI.
          C IT SIMPLY SETS THE SAMPLE CONFIGURATION AND POLARITY TO THOSE SPECIFIED
          C AND PASSES A TST SAMPLE OK SIGNAL BACK TO THE CALLING PROGRAM
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
0002      INTEGER CRS1,OUT1,IN1
0003      DATA CRS1/"167770"/,OUT1/"167772"/,IN1/"167774"/
0004      ICONF=ISAM-1
0005      IREG=IPEEK(OUT1) .AND. "170377
0006      IOUT=IREG .OR. (IPOL * 2048 + ICONF * 256)
0007      CALL IPOKE(OUT1,IOUT)
0008      TSTCON=1
0009      RETURN
0010      END

```

```

0001      FUNCTION DMM(IDATA)
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C THIS MODULE CALLS THE KEITHLEY MODEL 616 DIGITAL ELECTROMETER
          C TO READ THE CURRENT DATA. THIS IS DONE BY CALLING THE DRIVER
          C SUBPROGRAM SUPPLIED BY UDRI, IDMMC1, IDMMGT, AND FLOAT. IDMMGT
          C READS THE CURRENT VALUE FORM THE DMM INTO THE IDATA ARRAY. THIS
          C VALUE IS THEN TRANSFORMED INTO A FLOATING POINT NUMBER USING THE
          C IDMMC1, AND FLOAT.
          C
          C
          C      AUTHOR:  CAPTAIN EDGAR A. VERCHOT, JR., USAF
          C
          C
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C DATA SPECIFICATIONS
0002      INTEGER IDATA(9)
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C THIS PROGRAM WAS ORIGINALLY WRITTEN BY FRANK BEITAL OF UDRI. IT WAS
          C MODIFIED BY CAPTAIN VERCHOT FOR USE WITH HIS AHEDAS SYSTEM.
          C
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C      READ THE KEITHLEY ELECTROMETER
0003      I      CONTINUE
0004      L=IDMMGT(IDATA)
          C      NOW DECODE THE IDATA ARRAY PASSED FROM THE KEITHLEY
0005      MANTIS = IDMMC1 (IDATA(1),4)
0006      IFUNC = IDATA(5)/4
0007      IEXP=IDMMC1(IDATA(5),2)
0008      IDP=IDATA(1)/8 .OR. 2*IDATA(8)
0009      IF(IDP.EQ.-2)GO TO 10
0011      ISCALE = -128
0012      IF (IDP.EQ. 1) ISCALE = -2
0014      IF (IDP.EQ. 2) ISCALE = -1
0016      IF (IDP.EQ. 4) ISCALE = 0
0018      IF (IDP.EQ. 8) ISCALE = 1
0020      IF (IDP.EQ.16) ISCALE = 2
          D      WRITE(5,1000) MANTIS,IEXP,ISCALE,IFUNC,IDP
          D1000 FORMAT(' ',I5,'*10**',I3,' SCL',I4,' F',I4,' DP',08)
0022      IEXP1 = IEXP + ISCALE - 3
0023      DMM = FLOAT(MANTIS)*10.0**IEXP1
0024      GO TO 900
          C CHECK TO SEE IF THE DMM NEEDS TO HAVE THE SCALE CHANGED
0025      10      CONTINUE
0026      WRITE(7,100) IEXP+2
0027      100      FORMAT(' SET THE KEITHLEY ELECTROMETER TO THE 10 TO THE ',I3,
          C      'SCALE'// ' CARRIAGE RETURN WHEN SET', $)
          C      PAUSE
0028      PAUSE
0029      GO TO 1
0030      900      CONTINUE
0031      RETURN
0032      END

```

```

0001      FUNCTION IDMMC1 (IDIG,LEN)
0002      INTEGER IDIG(LEN)
          C
          C
0003      ISIGN = IDIG(1).AND.2
0004      IDMMC1 = IDIG(1).AND.1
          C
0005      DO 10 I = 2,LEN
0006          IDMMC1 = 10*IDMMC1 + IDIG(I)
0007      10  CONTINUE
          C
0008      IF (ISIGN.EQ.0) IDMMC1=-IDMMC1
          C
0010      RETURN
0011      END

```

```

0001      INTEGER FUNCTION MAGNET(FLDSET,FLD)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C THIS MODULE WILL SET THE MAGNET TO THE DESIRED FIELD VALUE.
C IT TAKES THE FIELD SETTING AND COMPUTES THE REQUIRED NUMBER
C OF STEPS AND SETS THEM. IT THEN CHECKS THE GAUSS METER AND MAKES ANY
C MORE REQUIRED ADJUSTMENTS. THIS DRIVER INCORPORATED THE DRIVER
C SUBROUTINES WRITTEN BY DANE HANBY OF UDRI.
C
C
C      AUTHOR:  CAPTAIN EDGAR A. VERCHOT, JR., USAF
C
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C DATA SPECIFICATIONS
C EQPOUT
0002      BYTE EQUIPF(10,8)
0003      INTEGER IEIOF,EQFLAG(7)
0004      COMMON /EQPOUT/EQFLAG,EQUIPF,IEIOF
0005      INTEGER CRS1,OUT1,IN1,IN2
0006      DATA CRS1/"167770"/,OUT1/"167772"/,IN1/"167774"/,IN2/"167764"/
C CHECK TO SEE IF THE MAGNET CONTROLLER IS INOPERATIVE
0007      IF(EQFLAG(5).NE.0)GO TO 300
0009      FLDMAG=ABS(FLDSET)-ABS(FLD)
C DETERMINE THE POLARITY TO WHICH THE MAGNET MUST BE SET.
0010      IF(FLDSET.EQ.0.0) GO TO 3
0012      IF(FLDSET.GT.0.0) GO TO 2
0014          IREV=1
0015          GO TO 3
0016  2      CONTINUE
0017          IREV=0
0018  3      CONTINUE
C SET THE POLARITY OF THE MAGNETIC FIELD.
0019      ISIGN=IPEEK(OUT1).AND.16384
0020      IF(ISIGN.EQ.IREV) GO TO 49
0022      IREG=IPEEK(OUT1).AND."13777
0023      IOUT=IREG.OR.(IREV*16384)
0024      CALL IPOKE(OUT1,IOUT)
0025  49      CONTINUE
C NOW CALCULATE THE NUMBER OF STEPS REQUIRED TO SET THE FIELD TO THE
C DESIRED SETTING. USING THE FOUR TO ONE RATIO PULLEY, THE PARAMETERS ARE:
C 180 STEPS/SHAFT REVOLUTION, AT 4:1---720 STEPS/KGAUSS, 100 STEPS/SEC RATE.
C EXPERIMENTAL TESTING REVEALED THAT 720 STEPS WAS INCORRECT; THE
C CORRECT VALUE IS 738.46 STEPS/KGAUSS.
0026      NSTEPS=ABS(FLDMAG)*738.46
C      FIND THE DIRECTION OF TRAVEL TO THE STEPPER MOTOR
0027          IDIR=2
0028          IF(FLDMAG.GT.0.0)IDIR=1
0030          IOUT=IASH(IDIR,12,IDUM)
0031          J=IQSET(2)
C      MOVE THE STEPPER MOTOR AS REQUIRED
0032  50      CONTINUE

```

AD-A080 175 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOO--ETC F/8 9/2
AUTOMATED HALL EFFECT EXPERIMENT DATA ACQUISITION SYSTEM (AHEAD--ETC(U)

UNCLASSIFIED AFIT/0E0/EE/790-5

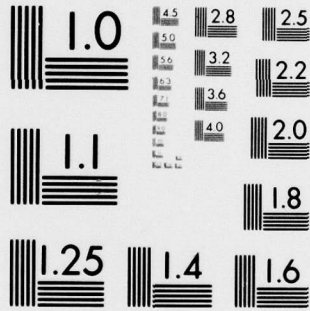
NL

3 OF 3

AD
A080175



END
DATE
FILMED
3 - 80
DDC



MICROCOPY RESOLUTION TEST CHART
 NATIONAL BUREAU OF STANDARDS-1963-A

```

0033      DO 10 I=1,NSTEPS
0034          CALL IBISA(IOUT,OUT1)
0035          CALL IBICA(IOUT,OUT1)
0036          K=ISLEEP(0.0,0.1)
0037  10    CONTINUE
      C    NOW CHECK THE MAGNET OUTPUT FOR STABILITY
0038  20    CONTINUE
0039          IF((IPEEK(IN2).AND."2").EQ.0)GO TO 30
0041          IF((IPEEK(IN2).AND."1").EQ.0)GO TO 30
0043          GO TO 40
0044  30    CONTINUE
0045          M=ISLEEP(0.0,1.0)
0046          GO TO 20
0047  40    CONTINUE
      C    NOW CHECK THE GAUSSMETER
0048          IF(EOFLAG(7).NE.0)GO TO 60
0050          DFIELD=ABS(FLDSET)-ABS(GAUSS(IOULD))
0051          IF(ABS(DFIELD).LE.0.01)GO TO 60
0053          NSTEPS=ABS(DFIELD)*738.46
0054          IDIR=2
0055          IF(DFIELD.GT.0.0)IDIR=1
0057          GO TO 50
0058  60    CONTINUE
0059          MAGNET=1
0060          GO TO 900
      C    GET OPERATOR TO SET MAGNET MANUALLY BECAUSE MAGNET CONTROLLER INOP
0061  800   CONTINUE
0062          WRITE(7,8000) FLDSET
0063  8000  FORMAT(' PLEASE SET THE MAGNET TO',G14.7,' KGAUSS'//
      2    ' ENTER AN INTEGER WHEN VALUE IS SET AND STABLE',$.)
0064          READ(5,*) INT
0065          MAGNET=1
0066  900   CONTINUE
0067          FLD=FLDSET
0068          RETURN
0069          END

```

```

0001      FUNCTION DUM(IFUNC)
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C THIS MODULE CALLS THE SCANNER AND THE DIGITAL VOLTMETER BY
          C CALLING THE GPIB, IEEE 488-1975 INTERFACE PROGRAM. THE NECESSARY
          C COMMAND STRINGS ARE LOADED INTO BYTE ARRAYS AND PASSED TO
          C THE INSTRUMENT VIA THE GPIBUIMAC SUBPROGRAM. THE DETAILS
          C OF THE CONTROL OPTIONS ARE SET OUT IN THE MANUALS OF THESE
          C THREE INSTRUMENTS: HP SCANNER, HP DIGITAL VOLTMETER, AND THE
          C NATIONAL INSTRUMENTS GPIB INTERFACE. THE SEQUENCE OF EVENTS
          C IS AS FOLLOWS:
          C   THE PROGRAM CALLS THE GPIB AND PASSES TO IT THE COMMAND
          C   ARRAY FOR THE SCANNER.
          C   THE SCANNER IS DIRECTED TO CLOSE ONE PARTICULAR CHANNEL.
          C   THE GPIB IS THEN CALLED AGAIN TO PASS A COMMAND ARRAY TO
          C   THE DIGITAL VOLTMETER WHICH CAUSES THE DATA TO BE READ FROM
          C   IT INTO THE PROGRAM. THERE IT IS DECODED FROM THE ASCII CODE
          C   AND PUT INTO A REAL FLOATING POINT VARIABLE.
          C
          C THIS IS DONE EACH TIME THE VOLT METER IS NEEDED TO MAKE A READING.
          C THE DUM IS THEN RESET TO READ THE THERMOMETER VOLTAGE CONTINUOUSLY.
          C
          C
          C   AUTHOR: CAPTAIN EDGAR A. VERCHOT, JR., USAF
          C
          C
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C DATA SPECIFICATIONS
          C
0002      BYTE MPX0(3),MPX1(3),MPX2(3),DUMCOM(14),VALUE(15),DUMON(4)
0003      INTEGER ERRET
0004      DATA MPX0/'0','0','E'/
0005      DATA MPX1/'0','1','E'/
0006      DATA MPX2/'0','2','E'/
0007      DATA DUMCOM/'F','1','R','7','T','1','M','3','A',
          2'0','H','0','D','0'/
0008      DATA DUMON/'T','3','T','3'/
0009      ICOUNT=0
0010  1    CONTINUE
0011      ASSIGN 1 TO ERRET
          C INITIALIZE THE DIGITAL VOLTMETER
0012      IF(IFUNC.NE.0) GO TO 10
0014          I=IBUP(0,3,DUMCOM,14)
0015          IF(I.NE.0)GO TO 800
0017          GO TO 900
0018  10   CONTINUE
0019      ASSIGN 10 TO ERRET
          C READ THE APPLIED VOLTAGE
0020  11   CONTINUE
0021      ASSIGN 11 TO ERRET
          IF(IFUNC.NE.1)GO TO 20
          C FIRST SET THE MULTIPLEX SCANNER(HP3495A) TO SELECT THE PROPER CHANNEL

```

```

0024      J=IBUP(0,2,MPX0,3)
0025      IF(J.NE.0)GO TO 810
C        NOW READ THE DUM DATA
0027      K1=IBUP(0,3,DUMON,4)
0028      K=IBUP(1,3,VALUE,15)
0029      K2=IBUP(2,3)
C        DECODE THE DUM OUTPUT
0030      DECODE(13,100,VALUE) DUM
0031 100   FORMAT(E13.7)
0032      GO TO 900
0033 20   CONTINUE
0034      ASSIGN 20 TO ERRET
C        READ THE SAMPLE VOLTAGE
0035      IF(IFUNC.NE.2)GO TO 30
C        FIRST SET UP THE HP3495A TO THE PROPER CHANNEL
0037      L=IBUP(0,2,MPX1,3)
0038      IF(L.NE.0)GO TO 810
C        NOW READ THE DUM
0040      L1=IBUP(0,3,DUMON,4)
0041      L=IBUP(1,3,VALUE,15)
0042      L2=IBUP(2,3)
C        DECODE THE DUM DATA OUTPUT
0043      DECODE(13,100,VALUE) DUM
0044      GO TO 900
0045 30   CONTINUE
0046      ASSIGN 30 TO ERRET
C        READ THE TEMPERATURE FROM THE SILICON THERMOMETER
0047      IF(IFUNC.NE.3)GO TO 820
C        FIRST SET UP THE HP 3495A TO THE PROPER CHANNEL
0049      M=IBUP(0,2,MPX2,3)
0050      IF(M.NE.0)GO TO 810
C        NEW READ THE DUM
0052      M1=IBUP(0,3,DUMON,4)
0053      M=IBUP(1,3,VALUE,15)
0054      M2=IBUP(2,3)
C        DECODE THE DUM DATA OUTPUT
0055      DECODE(13,100,VALUE) DUM
0056      GO TO 900
0057 800  CONTINUE
C ERROR MESSAGE FOR ERROR IN INITIALIZATION OF HP 3455A
0058      WRITE(7,8000)
0059 8000  FORMAT(' GPIB INITIALIZATION OF HP3455A FAILED')
0060      IF(ICOUNT.NE.0)STOP ' 3455A FAILURE'
0062      ICOUNT=1
0063      GO TO ERRET
0064 810  CONTINUE
C ERROR IN PASSING COMMAND ARRAY TO MPX SCANNER.
0065      WRITE(7,8100)
0066 8100  FORMAT(' GPIB FAILED TO SET HP3495 SCANNER PROPERLY')
0067      IF(ICOUNT.NE.0)STOP ' 3495A FAILURE'
0069      ICOUNT=1
0070      GO TO ERRET
0071 820  CONTINUE

```

```
C FUNC HAS AN ILLEGAL VALUE
0072      WRITE(7,8200)
0073 8200      FORMAT(' FUNC IN THE DUM ROUTINE HAS AN ILLEGAL VALUE')
0074 900      CONTINUE
C RESET THE DUM TO READ TEMPERATURE VOLTAGE CONTINUOUSLY
0075      N=IBUP(0.2,MPX2,3)
0076      RETURN
0077      END
```

```

0001      INTEGER FUNCTION ULTAGE(ULTSET)
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C THIS PROGRAM SETS THE HP39501A POWER SUPPLY PROGRAMMER TO THE
          C THE DESIRED OUTPUT VOLTAGE. THE POWER SUPPLY MUST BE SET TO THE
          C UNIPOLAR MODE. THE ALLOWABLE RANGE OF VALUES IS 0.00 TO 9.99 VOLTS.
          C
          C
          C          AUTHOR: CAPTAIN EDGAR A. VERCHOT, JR., USAF
          C
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C DATA SPECIFICATIONS
0002      BYTE SETCOM(4)
          C FIRST ENCODE THE VOLTAGE SETTING TO THE PROPER VALUE IN ASCII CODE
0003      IULT=(ULTSET*100)+2000
0004      ENCODE(4,100,SETCOM) IULT
0005      100 FORMAT(I4)
          C NOW CALL THE HP39501
0006      I=IBUP(0,1,SETCOM,4)
0007      ULTAGE=1
0008      RETURN
0009      END

```

.TITLE DMM -- KEITHLEY DMM DRIVER
.SBTTL ----- INTRODUCTION

PURPOSE: THESE ROUTINES SUPPORT THE KEITHLEY INSTRUMENTS, INC.
MODEL 616 DIGITAL ELECTROMETER EQUIPED WITH THE MODEL
6162 ISOLATED OUTPUT/CONTROL AS INTERFACED BY UDRI.

AUTHOR: FRANK E. BEITEL (UDRI) 8-JUN-79

.PAGE
.SBTTL ----- INTERFACE DEFINITION

INTERFACE DEFINITION:

THE INTERFACE CONSISTS OF:

1. CABLE FROM DMM TO LOGIC BOX
2. CABLE FROM LOGIC BOX TO LSI-11
3. ONE DRU-11 INTERFACE BOARD TO PROVIDE 16 BITS OF DATA
IN EACH DIRECTION
4. LOGIC BOX TO PERFORM DIGITAL SIGNAL CONDITIONING.

CONTROL SIGNALS FROM DMM TO COMPUTER

SIGNAL	CONN	PIN	PIN	DESCRIPTION
REQA	J1 LL	10	FLAG (LATCHED IN LOGIC BOX)	

.PAGE
.SBTTL ----- DEFINITIONS: DATA SIGNALS (DMM TO LSI-11)

DATA SIGNALS FROM DMM TO COMPUTER

SIGNAL	CONN	PIN	PIN	DESCRIPTION
IN00	J2 TT	21	DISPLAY 1 X 10**3 (STROBE 8)	
IN01	J2 LL	43	POLARITY	
IN02	J2 H	13	ZERO CHECK	

```

; IN03      J2  BB      7      DP1 (DECIMAL POSITION)
;
; IN00      J2  TT      30     DISPLAY 1 X 10**2 (STROBE 5)
; IN01      J2  LL      25     DISPLAY 2 X 10**2
; IN02      J2  H       26     DISPLAY 4 X 10**2
; IN03      J2  BB      24     DISPLAY 8 X 10**2
;
; IN00      J2  TT      47     DISPLAY 1 X 10**1 (STROBE 4)
; IN01      J2  LL      29     DISPLAY 2 X 10**1
; IN02      J2  H       28     DISPLAY 4 X 10**1
; IN03      J2  BB      46     DISPLAY 8 X 10**1
;
; IN00      J2  TT      17     DISPLAY 1 X 10**0 (STROBE 3)
; IN01      J2  LL      12     DISPLAY 2 X 10**0
; IN02      J2  H       11     DISPLAY 4 X 10**0
; IN03      J2  BB      16     DISPLAY 8 X 10**0
;
; IN00      J2  TT      42     EXP 1 X 10**1
; IN01      J2  LL      41     EXP POLARITY
; IN02      J2  H       44     F1 (STROBE 1)
; IN03      J2  LBB     45     F2
;
; IN00      J2  TT      37     EXP 1 X 10**0 (STROBE 2)
; IN01      J2  LL      38     EXP 2 X 10**0
; IN02      J2  H       35     EXP 4 X 10**0
; IN03      J2  BB      34     EXP 8 X 10**0
;
; IN00      J2  TT      8      DR (DOWN RANGE) (STROBE 6)
; IN01      J2  LL      9      UR (UP RANGE)
; IN02      J2  H       -      ***UNASSIGNED***
; IN03      J2  BB      -      ***UNASSIGNED***
;
; IN00      J2  TT      22     DP2 (STROBE 9)
; IN01      J2  LL      5      DP3
; IN02      J2  H       6      DP4
; IN03      J2  BB      23     DP5
;
; IN00      J2  TT      31     MANUAL RANGE (STROBE 5)
; IN01      J2  LL      32     R1
; IN02      J2  H       14     R2
; IN03      J2  BB      33     R3
;

```

```

.PAGE
.SBTTL ----- DEFINITIONS: DATA SIGNALS (LSI-11 TO DMM)
;
;

```

```

DATA SIGNALS FROM COMPUTER TO DMM
-----
;
;

```

```

; SIGNAL  CONN PIN      PIN      DESCRIPTION
;

```

```

; -----
;
; OUT00      J1  C      36  STROBE 5
; OUT01      J1  K       1  STROBE 4
; OUT02      J1  NN     18  STROBE 3
; OUT03      J1  U      19  STROBE 1
; OUT04      J1  L      49  STROBE 2
; OUT05      J1  N      48  STROBE 6
; OUT06      J1  R      20  STROBE 8
; OUT07      J1  T      39  STROBE 9
;
; OUT08      J1  W       2  OUTPUT HOLD
; OUT09      J1  X      50  DISPLAY HOLD
; OUT10      J1  Z      31  MANUAL RANGE
; OUT11      J1  AA     32  R1
; OUT12      J1  BB     14  R2
; OUT13      J1  FF     33  R3
; OUT14      J1  HH     15  ZERO CHECK
; OUT15      J1  JJ     --  STROBE 5 (LOGIC BOX CONTROL)
;

```

```

;*****

```

```

.PAGE
.SBTTL ----- DEFINITIONS: TRUTH TABLES

```

```

;
;
;*****

```

FUNCTION	F2	F1	*	*	SENSITIVITY	R4	R2	R1	*
-----	--	--	*	*	-----	--	--	--	*
			*	*					*
OHMS	0	0	*	*	.01	0	0	0	*
COULOMBS	0	1	*	*	.01	0	0	1	*
AMPERES	1	0	*	*	.01	0	1	0	*
VOLTS	1	1	*	*	.01	0	1	1	*
			*	*	.1	1	0	0	*
			*	*	1.	1	0	1	*
			*	*	10.	1	1	0	*
			*	*	100.	1	1	1	*
			*	*					*
			*	*					*

```

;*****

```

```

.PAGE
.SBTTL ----- DEFINITIONS: CONSTANTS AND GLOBAL SYMBOLS

```

```

;
;
; .MCALL .REGDEF
; .REGDEF
;
; ***** DEFINE ENTRY POINTS
;
; .GLOBL IDMMGT
;
; ***** DEFINE INTERRUPT VECTOR ADDRESSES
;

```

```

.GLOBAL DMMRDY
DMMRDY=330
;
; ***** DEFINE DEVICE ADDRESSES
;
; .GLOBAL DMMCSR, DMMIN, DMMOUT
DMMCSR=167750
DMMIN=DMMCSR+4
DMMOUT=DMMCSR+2
;
; .PAGE
; .SBTTL IDMMGT -- GET DATA FROM DMM
;
; PURPOSE: FORTRAN CALLABLE FUNCTION WHICH CHECKS TO SEE IF THE
; KEITHLEY DMM HAS DATA READY AND, IF SO, GETS IT. THIS
; FUNCTION RETURNS ZERO (.FALSE.) IF THE DMM IS DISABLED.
;
; CALLING SEQUENCE:
;
;     L = IDMMGT (IDATA)
;
; OUTPUT PARAMETERS:
;
;     L      -- TEST RESULT
;             L = .FALSE. (OR INTEGER 0) -- NO DATA READY OR
;                                     DMM DISABLED
;             .TRUE. (OR INTEGER 1) -- DATA READY
;
;     IDATA  -- 9-ELEMENT INTEGER VECTOR CONTAINING THE
;             DATA READ FROM THE DMM.
;
; *****
;
IDMMGT:      CLR      R0                ;R0,R1 <- FORTRAN LOGICAL .FALSE.
            CLR      R1
            BIT      #200,DMMCSR      ;IF DMM HAS DATA READY, THEN
            BEQ     2$,
            MOU     2(R5),R2          ; R2 <- ADDR(IDATA(1))
            MOU     #1,R3             ; R3 <- MASK FOR STROBE
            MOU     #9.,R1            ; R1 <- NUMBER OF CHANNELS TO STROBE
;
1$:         MOU     R3,DMMOUT          ; FOR I = 1 TO 9.
            MOU     DMMIN,(R2)        ; IDATA(I) <- VALUE FROM CHAN I
            BIC     #177760,(R2)+    ; KEEP ONLY LOW ORDER 4 BITS
            ASL     R3
            SOB     R1,1$            ; END FOR
;
            CLR     DMMOUT            ; CLEAR DMM CONTROL REGISTER
            INC     R0                ; R0,R1 <- FORTRAN LOGICAL .TRUE.
;
2$:         RTS     PC                ;RETURN
;
.END

```

```

      .GLOBL  FLOAT
;
FLOAT: CLR      R0
      MOV      &2(R5),-(SP) ;GET THE INTEGER TO BE CONVERTED
      BGE      1$ ;IF NEGATIVE THEN
      MOV      #100000,R0 ; R0 <- SIGN BIT ON
      NEG      (SP) ; MAKE IT POSITIVE
1$:   MOV      BIAS,-(SP) ;MAKE IT FLOATING POINT
      CLR      -(SP)
      MOV      BIAS,-(SP)
      FSUB     SP
      BIS      R0,(SP)
      MOV      (SP)+,R0
      MOV      (SP)+,R1
      RTS      PC
BIAS: .WORD    046000 ;THIS IS THE HIGH WORD OF A FLOATING POINT
;                                     ;2**24
      .END

```

```

.TITLE IBCD
.PSECT $CODE,RW,I,LCL,REL,CON
.GLOBL IBCD
IBCD:    MOV     &2(R5),R1      ;GET FIRST ARG
        CLR     R0
        CLR     R3
        MOV     #4,R2
1$:     ASHC    #4,R0          ;SHIFT R0,R1 FOUR BITS
        MUL     #10.,R3
        ADD     R0,R3
        SOB     R2,1$
        MOV     R3,R0
        RTS     PC
.END

```

THE FILE MYLIB.OBJ IS A LIBRARY OF ROUTINES DEVELOPED BY FRANK BEITAL,
OF UDRI, AFML/DOC. ANY QUESTIONS OR PROBLEMS WITH IT CAN BE DIRECTED TO
AFML/DOC, MR. FRANK BEITAL. THE AHEADAS SYSTEM USES THE ROUTINES IASH, IBICA,
AND IBISA. IASH, IBICA AND IBISA ARE USED IN THE MAGNET DRIVER.

THE AHEDAS SYSTEM IS AN OVERLAYED SYSTEM. IT WAS LINKED
IN THE FOLLOWING MANNER. IF A LINK MAP IS DESIRED ASS LP: MAP:, OTHERWISE

ASS NL: MAP:

R LINK

AHEDAS.SAU,MAP:=A0,HPDRIU,DX0:SYSLIB//

A1/0:1

A2,A3,A4,DRIUR2,FLOAT,DMMGT/0:1

IBUP1,IBUDP1,IBCD,MYLIB

A5/0:1//

THIS STRUCTURE SHOULD BE USED FOR ANY FUTURE VERSIONS OF AHEDAS IF THE OVERLAYS
ARE TO WORK PROPERLY.

Vita

Edgar Alphonse Verchot, Jr. was born 15 January 1951 in Auburn, Alabama. He graduated from Decatur High School, Decatur, Alabama, in 1968. He then attended the University of Alabama in Tuscaloosa, Alabama, where he received the Bachelor of Science degree with a major in Physics in 1972. He continued study and received the Bachelor of Science in Electrical Engineering degree in 1974. He was commissioned at this time in the United States Air Force.

After graduation, he was employed by the Alabama Power Company in Tuscaloosa until he reported for active duty 15 August 1974. He was assigned as a Wing Flight Test Director with the 4950th Test Wing at Wright-Patterson Air Force Base until he reported to the School of Engineering, Air Force Institute of Technology in August 1978.

He is married to the former Deborah Jean Mobley and has one son, Edgar, III.

Permanent Address: 1302 Morningside Court, S. E.
Decatur, Alabama 35601

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFIT/CEO/EE/79D-5 ✓	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Automated Hall Effect Experiment Data Acquisition System(AHEEDAS)		5. TYPE OF REPORT & PERIOD COVERED M.S. Thesis
7. AUTHOR(s) Edgar A. Verchot, Jr. Captain USAF		6. PERFORMING ORG. REPORT NUMBER
9. PERFORMING ORGANIZATION NAME AND ADDRESS Air Force Institute of Technology(AFIT/EN) Wright-Patterson AFB, Ohio 45433		8. CONTRACT OR GRANT NUMBER(s)
11. CONTROLLING OFFICE NAME AND ADDRESS Lasers and Materials Branch(AFML/LPO) Air Force Materials Laboratory Wright-Patterson AFB, Ohio 45433		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		12. REPORT DATE December 1979
		13. NUMBER OF PAGES 205
		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/ DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) Approved for Public Release, Distribution Unlimited.		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES Approved for Public Release; IAW AFR 190-17 Joseph P. Hipps, Major, USAF Director of Public Affairs		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Automation LSI-11 Microcomputer van der Pauw Microcomputer Hall Effect silicon Structured Analysis Automated Instrumentation		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) The Air Force Materials Laboratory conducts experiments using the Hall effect to characterize the electrical properties and impurity levels of silicon samples. Both the van der Pauw and the classical Hall bar methods are used. The purpose of this study was the development of an Automated Hall Effect Experiment Data Acquisition System(AHEEDAS) to control the conduct of the experiment and to reduce all of the necessary data. The designed system is capable of controlling all aspects of the experiment except the temperature. →		

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

The AHEEDAS produces as output, sample resistivity, Hall mobility, carrier concentration, and the Hall coefficient as a function of temperature. These are stored in data files on floppy disk storage along with all of the raw data from the experiment. The output is also printed on the computer terminal as the experiment is done. An LSI-11 microcomputer with 28K words of memory is used to control the experiment. Software was developed to allow this system to handle the acquisition and processing of data. The AHEEDAS was successfully implemented and tested. All functions other than the temperature control are fully operational.

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)