NISTIR 88-4009

Evaluation of Thermal Probe Method for Estimating the Heat Loss From Underground Heat Distribution Systems

Jin B. Fang and Richard A. Grot

U.S. DEPARTMENT OF COMMERCE
Nationl Institute of Standards and Technology
(Formerly National Bureau of Standards)
National Engineering Laboratory
Center for Building Technology
Building Environment Division
Gaithersburg, MD 20899

December 1988

National Bureau of Standards became the National Institute of Standards and Technology on August 23, 1988, when the Omnibus Trade and Competitiveness Act was signed. NIST retains all NBS functions. Its new programs will encourage improved use of technology by U.S. industry.

Prepared for
U.S. Department of Energy
Buildings and Community Systems
Building Services Division
Washington, DC 20585

U.S. DEPARTMENT OF COMMERCE
C. William Verity, Secretary

NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY
Ernest Ambler, Director

NISTIR 88-4009

Evaluation of Thermal Probe Method for Estimating the Heat Loss From Underground Heat Distribution Systems

Jin B. Fang and Richard A. Grot

U.S. DEPARTMENT OF COMMERCE
Nationl Institute of Standards and Technology
(Formerly National Bureau of Standards)
National Engineering Laboratory
Center for Building Technology
Building Environment Division
Gaithersburg, MD 20899

December 1988

Prepared for
U.S. Department of Energy
Buildings and Community Systems
Building Services Division
Washington, DC 20585

ABSTRACT

An automated, microcomputer controlled instrumentation system for in situ measurements of the earth temperatures and soil thermal conductivities at different depths is described. The system can also be used on site for calculating the heat losses from the underground district heating pipes. Step-by-step use and operation procedures of the developed heat loss measuring system and computer software package are presented. The heat loss rates and locations of underground pipes are calculated from the measured values of soil thermal conductivity and the earth temperatures around the pipes using the non-linear least squares method. The thermal probe technique was used to estimate the heat loss rates and the depths of buried steam supply and condensate return pipes installed at James Madison University, Harrisonburg, Virginia.

Key Words: computer software; district heating and cooling; earth temperature; heat loss; instrumentation system; nonlinear least squares fitting; soil; temperature probe; thermal conductivity; thermal probe; underground heat distribution system.

TABLE OF CONTENTS

LIST OF FIGURES AND TABLE

1. INTRODUCTION

A centralized heating plant generates steam, hot water or chilled water and delivers these process fluids through a network of pipelines buried underground or installed above the ground. The supply and distribution of energy by means of a district heating and cooling system can be more efficient and economical than a number of smaller units. The system can also effectively utilize a variety of fuels including municipal refuse and industrial waste heat to provide energy at lower prices, with greater opportunities for associated urban development compared to other alternative energy systems. District heating and cooling is considered as one of the most viable means to help attain energy independence. However, these advantages are not realized unless the operating cost due to heat loss through the underground system is low.

To determine the performance of underground distribution system, there is interest to develop procedures for estimating heat losses and heat gains from the system. These procedures will provide necessary information for optimum design of insulated piping networks and indicate when significant deterioration of pipe insulation and metal conduits occur. The information can serve as a basis for determining the necessity and priority of pipeline repair or replacement. Several types of measurement techniques such as condensate production rate [1] and shallow earth temperature [2, 3] measurements have been used for estimating the heat loss from a section of buried pipeline. Presently, there is no easy-to-use in situ method to quantify pipe heat losses accurately without major disruptions of normal operation of the pipelines.

This report describes a thermal probe technique developed for in situ measurements of soil thermal conductivity and heat loss from a directly buried conduit heat distribution system in which two insulated pipes encased in one or two metal conduits are installed in direct contact with the earth. This technique uses heat transfer theory, nonlinear least squares method, and measured thermal conductivity of the surrounding soil to convert the earth temperature profile around the underground pipes into heat loss values. This report also describes the detailed construction and step-by-step operation of an automated instrumentation system. The system is controlled by a microcomputer to measure soil thermal conductivity and earth temperatures in the vicinity of the underground heat distribution system. An application of the thermal probe technique to estimate the heat loss rates and locations of buried steam and condensate pipes installed on the James Madison University campus in Virginia is described.

2. TEMPERATURE DISTRIBUTION NEAR BURIED PIPES

A heat conduction model is employed to describe the temperature distribution in the soil above and around a pair of underground pipes. Figure 1 shows a schematic of the two-pipe, direct buried conduit underground heat distribution system. The derivation of expressions describing the temperature field near the buried pipes is based on two assumptions. The thermal conductivity of soil is assumed to be independent of temperature and the depth of a pipe is large compared to its pipe radius. With these simplifying assumptions, the heat conduction equation derived under steady-state conditions, with negligible moisture migration effects, and subject to boundary conditions of constant temperature for the ground surface and outer pipe wall, can be solved using the method of images [4]. Thus, the earth temperature disturbance caused by the heating or cooling of an underground pipe buried at a finite depth from the ground surface can be expressed by

$$T - T_o = \frac{Q_i}{4\pi k} \ln \left[\frac{(X-b_i)^2 + (Y+a_i)^2}{(X-b_i)^2 + (Y-a_i)^2}\right] \tag{1}$$

where T is the temperature of the soil at a given location, T_o is the undisturbed soil temperature, Q_i is the heat loss or heat gain of the i-th pipe per unit length, k is the thermal conductivity of the soil, X and Y are the Cartesian coordinates of any arbitrary point in the temperature field, and b_i and a_i are the horizontal distance and vertical depth of the center of the i-th pipe.

The underground temperature field around a two-pipe system with each pipe encased in a metallic conduit can be obtained by superimposing the contribution of each pipe to give

$$T = \sum_{i=1}^{2} \frac{Q_i}{4\pi k} \ln \left[\frac{(X-b_i)^2 + (Y+a_i)^2}{(X-b_i)^2 + (Y-a_i)^2}\right] + T_o \tag{2}$$

This non-linear multivariable function can be solved to give the heat losses (Q_1, Q_2), locations (b_1, b_2) and depths (a_1, a_2) of the pipes using the method of non-linear least squares, provided the earth temperature and thermal conductivity data are available. In the field, the locations of the underground pipes may not be well known. In order to improve the estimate of pipe heat loss, one of the unknown parameters is removed by introducing a known separation distance, d, between the centers of the pipes (see figure 1). This separation distance can be obtained from either the pipeline layouts in the architectural drawings or by measurement where the pipes are accessible in the nearby manholes.

The temperature of the soil surrounding two underground pipes installed in a single metallic conduit (see figure 2) is given by

$$T = \frac{Qt}{4\pi k} \ln \left[\frac{(X-b)2 + (Y-a)2}{(X-b)^2 + (Y-a)^2}\right] + T_o \qquad (3)$$

where Q_t is the total heat loss per unit length of the pipes, and b and a are the horizontal distance and the depth of the center of the conduit, respectively. This equation is a simplified case of equation 1. With the use of non-linear least squares technique, this equation can be solved to yield the combined heat loss from the pipes, and the location and depth of the conduit based on the soil temperature and thermal conductivity data.

3. TRANSIENT NEEDLE METHOD FOR DETERMINATION OF SOIL THERMAL CONDUCTIVITY

The advantage of the transient needle method is both the soil thermal conductivity and diffusivity can be determined simultaneously from the test data without knowledge of the heat capacity of the soil. The instantaneous temperature rise at a point on the surface of a long heated cylinder or needle, which has smaller diameter compared to its length and is dissipating heat into an infinite homogeneous medium, can be approximated by [5]

$$T_s = \frac{Q}{4\pi kL} \left[\ln (t) + \ln \left(\frac{4\alpha}{r^2}\right) - \gamma\right] \qquad (4)$$

where T_s is the surface temperature of the cylinder or needle, Q/L is the power input per unit length, t is the elapsed time, r is radial distance from the line heat source, γ is Euler's constant (0.5772), and k and α are soil thermal conductivity and diffusivity, respectively.

From this equation, it is apparent that if temperature is plotted versus ln(t), the curve with t greater than some certain value becomes asymptotic to a straight line having the slope equal to $Q/(4\pi kL)$ and the intercept equal to $\ln(4\alpha/r^2) - \gamma$ on the ln(t), axis. The soil thermal conductivity and diffusivity can be determined from the slope and intercept values of the least squares method regression line, which best fits the experimental temperature-time data. If S is the slope and I is the intercept of the extrapolated straight line on the ln(t) axis, the thermal conductivity (k) and thermal diffusivity (α) of soil can be calculated from the following equations:

$$k = \frac{Q}{4\pi LS} \tag{5}$$

$$\alpha = \frac{r^2}{4} \exp\left(\frac{I}{S} + \gamma\right) \tag{6}$$

4. DESCRIPTION OF THE INSTRUMENTATION SYSTEM

A microcomputer based instrumentation system was developed for in situ measurements of the soil temperature and thermal conductivity, and the heat loss rates of the heat supply and the return pipes. The instrument can be operated in an interactive mode and contains computer routines for data storage on a floppy disk, and for on-line data analysis and plotting.

The thermal conductivity measuring system is a microcomputer-controlled unit providing programmable power to thermal conductivity probes, and measuring both the output voltages of the thermocouples attached on the probe wall surface and the electrical power consumed by the probe heater. The measuring system consists of a 200 W autoranging DC power supply, two terminal boxes each having eight analog input connections, a sixteen-channel analog interface card, a data acquisition board, and a portable microcomputer. This instrumentation system requires a constant 120 V/AC power source and is controlled entirely by the computer software. The computer is also used to calculate the soil thermal conductivity and diffusivity for each thermocouple input. The performance specifications of the major hardware are as follows:

4.1 Autoranging DC Power Supply

The programmable DC power supply (Hewlett Packard Model 6024A and option 002)[1] can provide output voltage ranging from 0 to 60 V, 0 to 10 A output current, and 200 W autoranging power output from 120 V/AC source. The power supply is equipped with a system interface board for remote monitoring and control of its output voltage and current. It can be remotely programmed to provide the output power varying over a wide and continuous range of voltage and current combinations. These output power characteristics are necessary for electrical heating of the thermal conductivity probe.

---

1. Certain trade names or company products are mentioned in the text here and in subsequent chapters to specify adequately the experimental procedure and equipment used. In no case does such identification imply recommendation or endorsement by the National Institute of Standards and Technology, nor does it imply that the products are necessarily the best available for the purpose.

The power supply has front panel controls, plus a voltage and a current meter for continuous display of the output levels. It is furnished with a built-in adjustable overvoltage protection circuit to safeguard against excess voltage and current output. The power supply will shut down if either condition is met. All connections between the interface board and external circuits are made with a 50-pin connector, which is modified to fit a 37-pin connector in the rear panel of the power supply.

4.2 Terminal Boxes and Analog Interface Card

The computer interface system for temperature measurement consists of two terminal boxes and an analog/digital interface board (Omega White Box Analog Interface and Control for IBM Personal Computer, Part No. WB-AIO-B16). The terminal box can accomodate a wide range of DC voltage and sensors, and has a cold junction compensation device for thermocouples. Its function in the system is to read the thermocouples of the temperature probes and thermal conductivity probes. It can accept up to 8 analog inputs with a peak continuous operating voltage of 50 V/DC and a maximum current of 1A. In order to determine the thermal conductivity of the soil and the heat loss of the underground pipes, thermocouples (type T) are used to measure the earth temperatures. The thermocouple wire leads are connected to the terminal blocks in the terminal boxes, and then to two 26-pin analog connectors for a 16-channel analog interface card by an analog ribbon cable. The terminal box also provides ports for input and output for each of the 8 digital lines for digital control.

The analog interface card contains an analog-to-digital converter with 14 bit resolution, 16 analog input channels, and 16 digital lines for digital inputs or outputs. This card was plugged into one of the empty expansion slots of the microcomputer. As suggested in the user's manual, the analog cable from the terminal box with the lower serial number was connected to the 26-pin connector for analog channels 1 to 8, while the other terminal box was connected to analog channels 9 to 16. The interface card accepts mV, V, mA and thermocouple inputs with an uncertainty of less than 0.04% of range, and has the filter delay time varying from 0.015 to 0.4 second per channel. It is capable of handling up to 16 thermocouple inputs and digitalizing into temperature values at a rate of 0.5 seconds per channel. The scan time for consecutive readings of the entire 16 thermocouple channels using this analog interface card was found to be approximately 13 seconds. The measurement errors due to type T thermocouples and the cold junction compensation device at 25°C are estimated to be $\pm$ 0.8°C and $\pm$ 0.02°C, respectively. To facilitate transporting and operating of the instruments, the programmable DC power supply and two terminal boxes were housed in a rugged metal case.

4.3 Data Acquisition System Board

A general purpose single board data acquisition system (Data Translation, Model DT 2801 board) was plugged directly into one of the expansion slots of an IBM PC compatible computer. This board served the purpose of controling the autoranging DC power supply using the digital-to-analog converter of the board to control the voltage and amperage levels of the power regulator, and the analog-to-digital data acquisition module to monitor the performance of the power supply. Certain special functions of the power supply are controlled and monitored using the digital input and output ports on the board. Prior to plugging the board into the backplane of the computer, the jumpers on the board were installed or removed according to functions of analog and digital conversion in selecting various board parameters as given in the user's manual. The board parameters selected are the analog input voltage ranging 0 to +5 V, bipolar input mode, single-ended inputs, and the board base address assigned at 2EC (HEX) port address. The board can be programmed from the IBM PC compatible to perform analog-to-digital (A/D) and digital-to-analog (D/A) conversions, and digital input and output transfers.

The board has an A/D converter for 16 single-ended or 8 differential analog input channels with 12-bit resolution or 0.024% of the analog input range, and two D/A converters with the same resolution. It contains a programmable gain amplifier to permit gains of 1, 2, 4 and 8 to be selected by software so that a wide range of input signal levels can be accommodated. The board also has two 8-line digital I/O ports, which can be used separately to read or write 8-bit data, or changed simultaneously for 12 or 16-bit data transfers, and an on-board programmable clock with a base frequency of 13.7 kHz to provide clock pulses to control the operations of A/D and D/A converters. The board for data acquisition has two connectors: a 62-pin PC I/O bus connector, which was made connections automatically to the PC bus when it was installed into the backplane of the PC, and a 50-pin connector. The 50-pin connector is accessible from the rear panel of the PC to connect all analog and digital inputs and outputs to the board. The system accuracy of the board is estimated to be within ± 0.05% of full scale input range, and with power off, the board can accept a maximum input voltage of ± 20 V. Details of the pin connections between the 37-pin rear panel connector J2 of the programmable DC power supply and the 50-pin connector J1 of the data acquisition DT2801 board are given in Table 1.

4.4 Microcomputer

A portable computer (XPC Compact model) is used as the base of automated operations for the instrumentation system. The major components of this IBM PC compatible are as follows:

(1) A 8088 CPU board with 640 KB RAM memory and six expansion board slots.

(2) A Hercules compatible video card and an Intel 8087 math-coprocessor.

(3) Two 360 KB double-density, double-sided, 5-1/4" disk drives.

(4) A parallel printer port, a serial I/O port and a 4.77 MHz clock with battery backup.

(5) A keyboard and a monochrome display monitor.

In addition to these basic components, the portable system contains a printer (Epson, model FX-86e). A nominal 110 V/AC power source is required for the entire automated instrumentation system.

5. SOFTWARE

A software package called the 'Underground Piping Heat Loss Diagnostics' has been developed to control all operations of the microcomputer-based automated thermal probe system. The software is written in such a way that all acquisition, storage and analysis of test data can be performed in an interactive manner.

The computer programs are written in FORTRAN and assembly language, and consists of a main program (HEAT) and thirty eight subroutines. In addition to coordinating all operations of the instrumentation and data acquisition system, the main program provides a main menu for selecting types of thermal measurements and calculations. Various functions of major subroutine subprograms are described briefly in Appendix A and a listing of the source code of the computer program is given in Appendix C.

6. MEASUREMENT EQUIPMENT AND PROCEDURES

In addition to the instrumentation and data acquisition system described previously, the major equipment employed for measurement of the heat loss from underground pipes include a mobile drilling rig, thermal conductivity and temperature probes, and a 120 V/AC generator to provide a constant power supply. The drilling rig is mounted on a two-wheel, single axle trailer (General Equipment Co. 550 Dig-R-Mobile) and equipped with a motorized auger powered by a 7 horse power gasoline engine. The drill bit used with the auger for boring into the ground is a 7/8 in. diameter drill attached to 3 ft and 6 ft long extension rods of 1 inch diameter.

The thermal conductivity probe is a hollow stainless steel sheath with both ends closed that contains an electric heater and thermocouples (type T) installed at the interior wall of the sheath. For field measurements, two types of the 1 in. (25 mm) diameter probes used are 4.3 ft (1.3 m) long with

2 thermocouples positioned at 1.3 ft (0.4 m) and 2.0 ft (0.62 m) from the lower end of the probe, and 6.6 ft (2 m) in length with 3 thermocouples separately installed at 1.4 ft (0.42 m), 2.8 ft (0.84 m) and 4.1 ft (1.25 m) from the lower end. In order to check the operations of the developed instrumentation system and to determine the thermal property of soil samples taken from the field, a laboratory probe (Geotherm, Inc.) 1/8 inch (3mm) diameter 4 inch (100 mm) long is used for measuring soil thermal conductivity. A typical commercial built laboratory probe containing a single thermocouple is shown in Figure 3. The thermal time constants, which is the time necessary to pass the startup transient, for the laboratory and field probes are typically 100 and 1000 seconds, respectively.

Figure 4 shows the construction details of a temperature probe. The temperature probe was fabricated from a nominal 3/4 in. (19 mm) steel pipe of 1 inch (27 mm) outside diameter by 7 ft (2.1 m) in length. Six thermocouples are installed on the outer wall surface at 1 ft (0.3 m) intervals starting at 3/4 in. (19 mm) from the lower end closed with a plug. The type T thermocouple junction was attached to the exposed surface of a 1/4 inch (6 mm) diameter by 1/4 in. (6 mm) thick teflon plug threaded into the temperature probe. This steel probe can eliminate probe deformation problems encountered at high temperatures in comparison to the temperature probe constructed from a flexible pvc pipe used in previous field measurements [2]. In field measurements, ground holes are drilled up to 6 feet in depth at 1 ft intervals along a line perpendicular to the buried pipes for an extension of at least 4 ft on both sides of the heat supply pipe. To minimize the oversizing of the hole caused by side-to-side movement of the drill, a special auger guide is utilized in ground boring. A ground hole having an outside diameter smaller than the thermal conductivity probe will minimize thermal contact resistance between the probe and the earth.

The thermal conductivity probe is pushed down manually one of the selected holes to assure good probe contact with the earth. The microcomputer-controlled instrumentation system is used in conjunction with the probe to provide programmable electric power to heat the probe; read thermocouples; probe electrical current and voltage, and calculate soil thermal conductivity and thermal diffusivity for each thermocouple location. The step-by-step use and operation procedures of the instrumentation system are given in Appendix B.

The instrumentation system can also be used to determine soil thermal properties in the laboratory. The thermal conductivity of a sand sample was determined using the developed instruments and a standard 1/8 in. (3 mm) diameter laboratory probe. The measured values were found to be comparable, with a deviation within 5%, with those obtained by a commercially available, microcomputer-controlled thermal property analyzer (Underground Systems, Inc.) on the same sample.

In field measurement of earth temperatures, the temperature probe is inserted carefully into the same hole to ensure again good probe-soil thermal contact, following the disconnections of probe power plug and thermocouple input and the manual removal of the thermal conductivity probe from the hole. The connections required are the thermocouple inputs from the probe to a terminal box. The instrumentation system can accomodate two temperature probes at different locations and can measure up to twelve earth temperatures of six separate depths simultaneously. The system is used in connection with the temperature probes to read thermocouple outputs; display and record the earth temperatures and relative locations of all thermocouples installed on the probe surfaces. The operation procedures of the instrumentation system for temperature measurement are presented in Appendix B.

The instrumentation system is then instructed to perform calculations based on the earth temperature data for determining the heat loss rates and the locations of two insulated pipes using the non-linear least squares method. The pipe heat loss rates are printed on the computer screen along with the horizontal distance and vertical depth of the underground pipes. The final display is also recorded on a floppy disk. The detailed operations of the instrumentation system for heat loss calculations are given in Appendix B.

For measurement of soil moisture content, a 3-in. (76 mm) diameter helicoid bore auger is used to drill and excavate soil samples at various depths. Each soil sample taken with a scoop is sealed in a plastic wrap and placed in a glass container. The moisture content of soil sample is determined by measuring the loss in mass of the sample after drying in an electric oven maintained at 100 ± 3^{o} C (212 ± 5^{o} F) for a week to a constant mass ($\pm$ 0.5%) in the laboratory.

7. SAMPLE CALCULATIONS

A set of test data obtained from a field measurement [2] performed on a directly buried conduit steam distribution system installed at the James Madison University campus, Harrisonburg, Virginia, was used to evaluate the heat loss calculation routines. This underground system consists of a nominal 6-in. (152 mm) steam pipe and a 3-in. (76 mm) condensate return pipe laid side by side with a separation distance of 13 in. (0.33 m) between the pipe centers, and buried approximately 4 ft (1.22 m) below the ground surface. The carrier pipes were installed in a 36 in. (0.91 m) wide by 30 in. (0.76 m) high rectangular trench filled with protexulate insulation (Protexulate Inc.), which is a mineral powder loose-fill insulating material,and covered with the earth. The earth temperatures were taken for 58 measuring locations in a plane normal to the pipes. These measuring points were distributed horizontally at 1 ft (0.31 m) intervals covering a total distance of 11 ft (3.35 m) on both sides of the steam pipe and vertically at 1 ft (0.31 m) intervals between depths 1 to 5 ft (0.31 to 1.52 m). The average value of measured soil thermal conductivities at the test site was found to be 0.524 Btu/h·ft·F (0.907 W/m·C).

Based on these earth temperature and thermal conductivity data obtained from the thermal probe method and the separation distance between the pipes as the input data, the heat loss rates and locations of the underground pipes are calculated using the computer code in option 3 of the main menu. The final results of the computer outputs for these sample calculations are given below:

	Pipe No. 1	Pipe No. 2
Heat Loss Rate, Btu/h·ft	183.9 (185.5)	-90.3 (-91.8)
Horizontal Distance, inch	72.4 (72.5)	59.4 (59.5)
Vertical Depth, inch	47.9 (47.9)	53.2 (53.3)

It is not possible to verify the calculated heat loss values of the steam and the condensate return pipes since the actual values are unknown. However, the estimated depths and locations of the underground pipes are in good agreement with the values found in the pipeline layouts of the architectural drawings. In order to check the validity of this calculation procedure, the numerical values in parenthese in the above table are the DATAPLOT software package implemented on an UNIVAC 1100/80 computer for statistical anlaysis and are also listed for comparison. It can be seen that the heat loss rates and locations of both buried pipes calculated from this computer code agree reasonably well with those obtained from the DATAPLOT software package.

8. CONCLUSIONS AND RECOMMENDATIONS

An automated instrumentation and real time data acquisition system controlled by an IBM PC compatible computer was constructed for in-situ measurements of soil thermal properties and earth temperatures at various depths. The heat loss from an underground insulated piping system in district heating and cooling can be measured using the developed instrumentation system. A step-by-step use and operation procedure of this instrument and the computer software package for field and laboratory thermal measurements are presented. The developed hardware and computer software were tested under actual use conditions and found generally to provide satisfactory performance.

The steady-state solutions describing the temperature distribution in the earth around two directly buried pipes installed in separate metallic conduits or a single conduit are given. The solution models the pipes as line heat sources and treats the ground surface and the outer pipe walls as isothermal surfaces. It is particularly applicable for the case when the pipe depth is large compared to the pipe radius. Using the method of nonlinear least squares, the equation describing the local earth temperature, which is a nonlinear multivariable function, can be solved to give the heat loss rates, the depths and locations of the buried pipes. The necessary input data include soil thermal conductivity and the earth temperatures obtained from the thermal probe technique, and the separation distance between pipes. The developed computer programs are implemented on the microcomputer and give proper computing speed and adequate accuracy on the calculated results. Sample calculations based on the test data obtained from field measurements conducted on a directly buried conduit steam distribution system installed at the James Madison University, Harrisonburg, Virginia are presented. The calculated pipe depths and locations are generally consistent with the actual values found in the pipeline layouts of architectural drawings, and the estimated heat loss rates agree reasonably well with those by the DATAPLOT software package installed on a mainframe computer for statistical analysis.

The use of thermal probe technique exhibits a considerable promise for estimating the heat loss from an underground heat distribution system. Continuous measurements of earth temperatures and thermal conductivities, and processing and analysis of test data can be carried out rapidly and effectively in the field. Further work is recommended to validate and improve this measurement technique by using tests involving a pair of long insulated pipes with known heat losses and pipe depths. A series of tests will have to be conducted for pipes buried in soils exposed to various surface moisture conditions. Comparison of the pipe heat losses measured by the thermal probe method with those determined by other techniques such as the calorimetric method [2] on a section of buried pipes is needed for improved accuracy of this method. It is recommended to apply the thermal probe technique for measuring the heat loss through piping system anchors and supports on the site, which appears to be possible.

9. ACKNOWLEDGMENT

This work was sponsored by the Building Services Division, U.S. Department of Energy through an interagency agreement with the National Institute of Standards and Technology. The authors wish to express their appreciation to Mr. Prakash B. Kunjeer of the U.S. Department of Energy for guidance and assistance throughout the project. The authors would like to thank Douglas Pruitt of NIST for his assistance in constructing the automated instrumentation system, and Sandra Foltz of NIST for her help in developing the computer programs used in this study.

10. REFERENCES

1. Pan Am World Services, Inc., "Heat Distribution Systems Life-Cycle Cost Analysis - Comparison Between Direct Buried and Shallow Trench Systems," Report 130319, July 1985.
2. Kusuda, T., Fang, J. B. and Ellis, W. M., "A Method for Measuring Heat Loss from Underground Heat Distribution Systems," Thermal Insulation: Materials and Systems, ASTM STP 922, F. J. Powell and S. L. Matthews, Eds., American Society for Testing and Materials, Philadelphia, PA, 1987, pp. 52-68.
3. Beck, J. V. and Karnitz, M. A., "Parameter Estimation Study of Heat Loss from Underground Steam Pipelines," ORNL/TM-9928, Oak Ridge National Laboratory, Oak Ridge, TN, June 1986.

4. Eckert, E. R. G. and Drake, R. M., 'Analysis of Heat and Mass Transfer,' McGraw-Hill Book Co., New York, 1972.

5. Carslaw, H. S. and Jaeger, J. C., "Conduction of Heat in Solids," Oxford University Press, London, 2nd Edition, 1959.

6. Boggs, S. A. and Radhakrishna, H. S., "Soil Thermal Resistivity and Thermal Stability Measuring Instrument - Volume 4: Abridged Operating Instructions for the Thermal Property Analyzer," EL-2128, Electric Power Research Institute, Palo Alto, CA, November 1981.

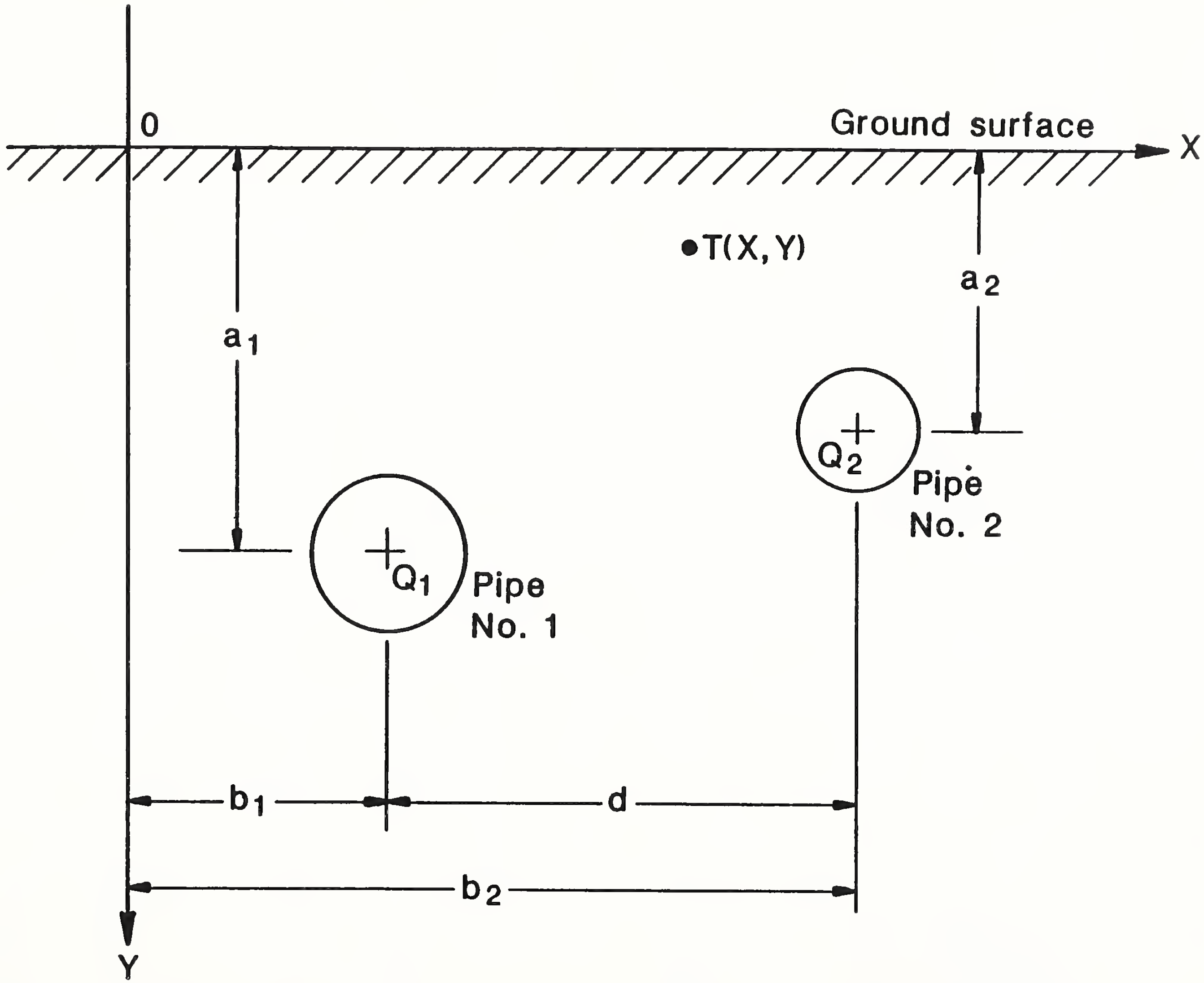

Figure 1. Schematic of a Two-Pipe Underground Heat Distribution System

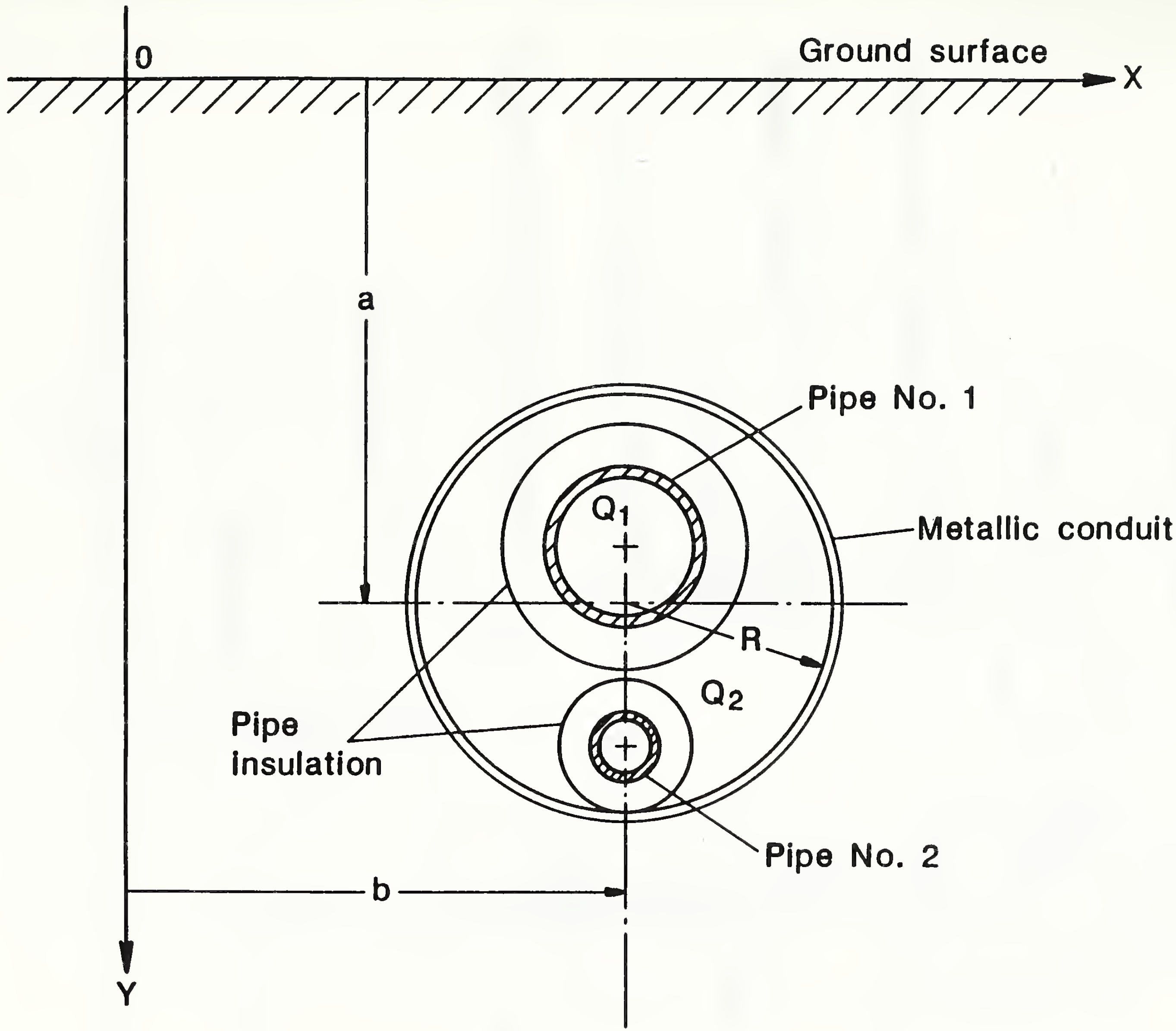

Figure 2. Two Underground Pipes Encased in a Metallic Conduit

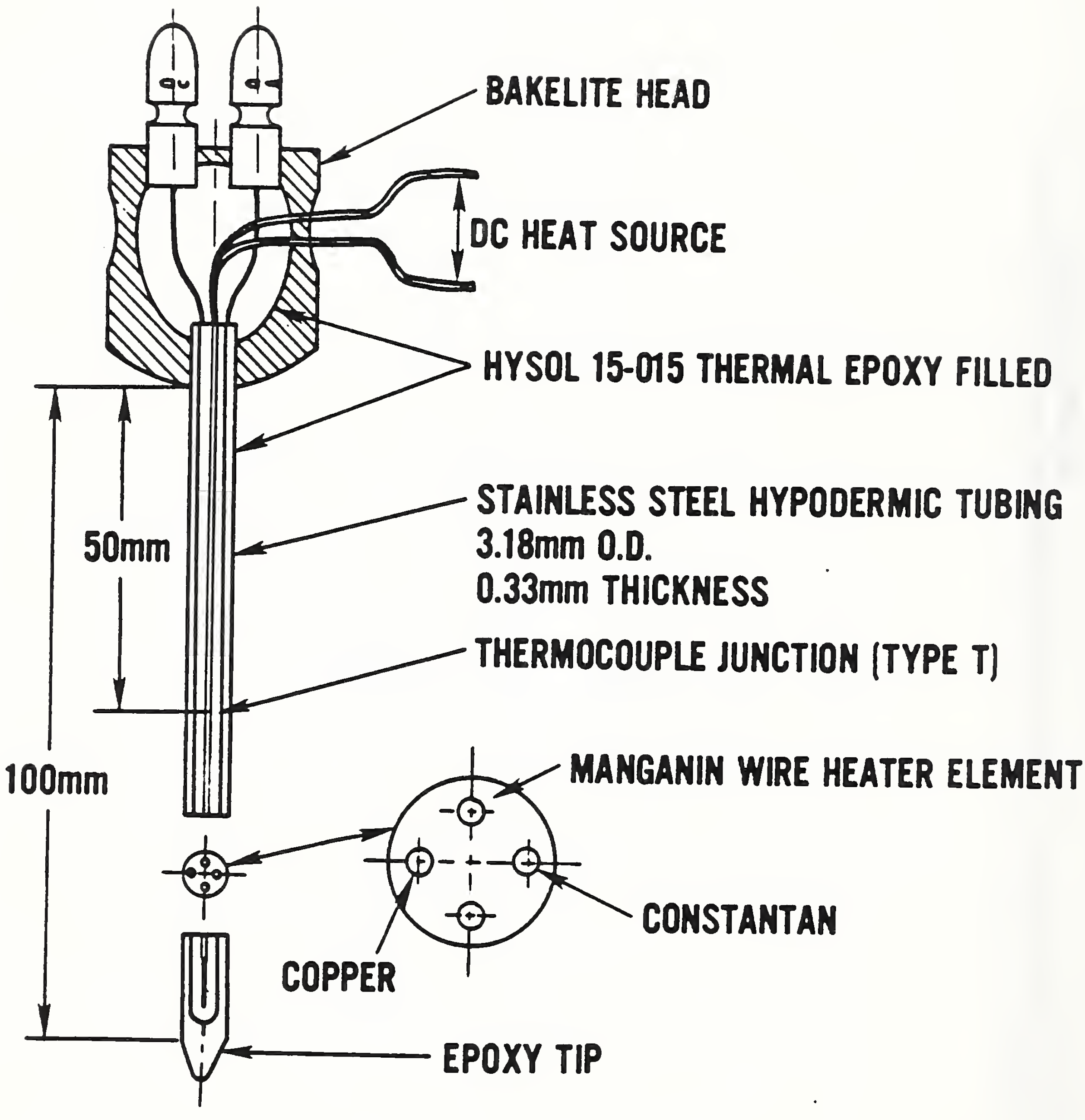

Figure 3. Sectional View of A Laboratory Thermal Conductivity Probe

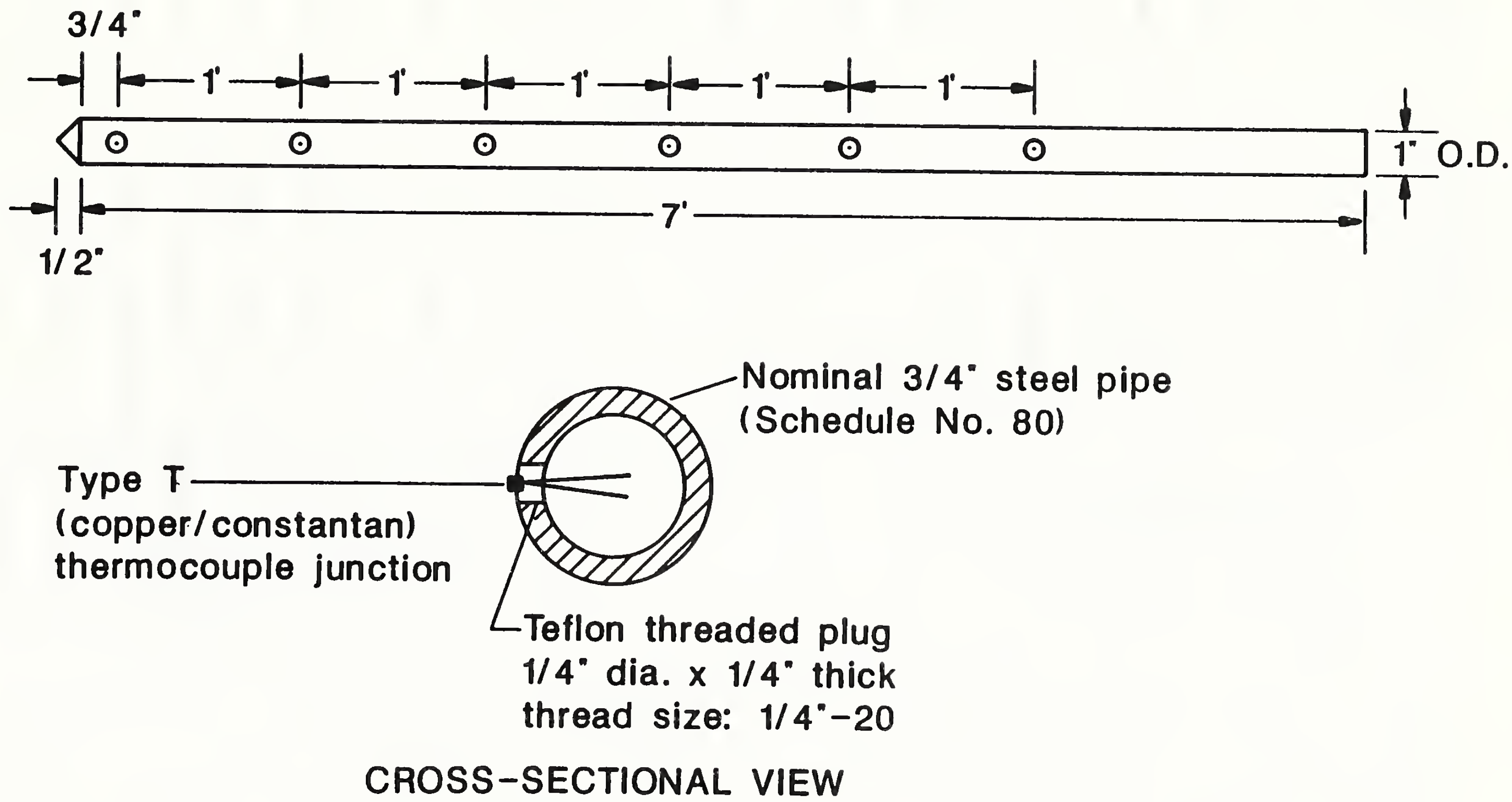

Figure 4. Construction Details of A Temperature Probe

Table 1. Pin Connections Between Connector J2 of the DC Power Supply and Connector J1 of the Data Acquisition System Board

J2 POWER SUPPLY		J1 DT2801	
Pin No.	Description	Signal Name	Pin No.
1	Outboard Sense	DAC 0 GND	23
24	CC Volt Prog	DAC 0 Out	22
22	- Sense	DAC 1 GND	25
25	CV Volt Prog	DAC 1 Out	24
3	Current Monitor	Channel 0	1
5	Voltage Monitor	Channel 1	3
6	Power-on Preset	Amp Low	18
20	-15V Reg.	AGND	17
4	+15V Reg.	AGND	21
23	+5V Reg.		
7	Power Supply Common	DGND	26,27,32,37,42,47,48
10	Control Isolator Bias		
29	$\overline{\text{Remote Reset}}$	DIO Port 0, Bit 0	28
30	$\overline{\text{Remote Trip}}$	DIO Port 0, Bit 1	29
31	$\overline{\text{Remote Inhibit}}$	DIO Port 0, Bit 2	30
16	$\overline{\text{Over temperature}}$ 4.7K	DIO Port 1. Bit 0	38
17	$\overline{\text{Over voltage}}$ 4.7K	DIO Port 1, Bit 1	39
18	$\overline{\text{Output Unregulated}}$ 4.7K	DIO Port 1, Bit 2	40
19	Low Bias 4.7K	DIO Port 1, Bit 5	44
35	$\overline{\text{CC Mode}}$ 4.7K	DIO Port 1, Bit 3	41
36	$\overline{\text{CV Mode}}$ 4.7K	DIO Port 1, Bit 4	43
37	Status Isolator Bias		
34	Status Isolator Common		

Appendix A. Descriptions of Major Subroutine Subprograms

Subroutine THERMA is used to read up to four thermocouples installed in a thermal conductivity probe, to calculate the thermal conductivity and diffusivity of soil surrounding each thermocouple location, and to store the test data in a file named by the user. Subroutine TEMPER reads earth temperatures from up to sixteen thermocouples positioned at different locations and various depths, and records both detailed and briefly summarized results of temperature measurements on two data files. Subprogram HLCALC performs calculations of the heat loss from underground pipes based on least squares fitting of the earth temperature data to a theoretically derived non-linear equation describing the underground temperature field. Subroutine LMMNL determines the parameters in the non-linear function based on the Levenberg, Marquardt and Morrison algorithm modified for one or more independent variables, and FUNVAL evaluates the function and its partial derivatives with respect to the parameters. Subroutine INITAL is used to initialize the analog interface card and DEGREE reads temperatures from thermocouple outputs. Subprogram POWERON resets and performs operations of the programmable DC power supply for the probe heater, and POWON calculates the electrical current of the desired power level and turns on the power to the thermal conductivity probe. Subroutine RDPOW reads the electrical current and voltage levels from the programmable DC power supply, and CONSTPOW regulates the electric power applied to the probe to be within $\pm$ 0.08 W of the desired value. Subroutine TEMPDT calculates the time to start and to terminate power to the probe heater, and reads the probe surface temperatures from up to four thermocouple input channels. Subprogram CALC computes soil thermal conductivity and diffusivity and coefficient of correlation and finds maximum and minimum values for graph plotting. Subroutines SETTEM, GTEMP, PLTTEM and PLTCHR are used for plotting temperature versus logarithm of time to the screen. Subroutine HILITE is used to highlight a cell by setting its attributes to reverse video while returning the previously highlighted cell attributes to normal. Subroutines CLOCK reads the system clock and PRTCLK writes time and date to the screen.

Appendix B. Operation Procedure of the Instrumentation System

The system connections required include the main power cords to the computer and the programmable DC power supply, a probe power plug and thermocouple inputs, analog card inputs, and the programmable DC power supply remote control. After the instrumentation system is setup and the DOS (Diskette Operating System) disk is inserted into drive A, both the computer and the programmable DC power supply are turned on to load the computer operating system. In response to the prompts from the computer, the user enters the date and time, and then inserts a formatted new blank disk into drive B. Replacing the DOS disk with the program disk, the user types 'HEAT' and presses the ENTER key to start the program. Pressing the ENTER key always terminates the input line and using the CTRL-C key terminates program execution of 'HEAT'and returns to the operating system.

In a few seconds, the screen will show the title of the software package, 'Underground Piping System Heat Loss Diagnostics. The main menu is displayed after depressing of the ENTER key. The main menu lists all of the options available for the user. The options are as follows: 1. Determine the soil thermal conductivity, 2. Measure the ground temperatures, 3. Calculate heat losses from buried pipes, 4. Exit. Using the arrow keys on the numeric key pad to move the cursor up and down, the user makes a selection of the desired option, which is displayed and highlighted on the screen, and presses the ENTER key. Option 4 in the main menu is used to exit from the program to the operating system.

B.1 Thermal Conductivity Measurement

The user selects option 1 for measuring the thermal conductivity of soil surrounding an underground heat distribution system. A test setup file must exist before a thermal conductivity measurement can be performed. By answering questions and typing data on the keyboard, the user either creates a new setup file or uses an existing one created and stored previously on the disk. The following is an example of a setup file created interactively in which typical input data are enclosed in parentheses.

FILE NAME to be 1 to 10 alphanumeric characters long (B:SETUP 01)

PROBE SERIAL NUMBER to be any 2 digit number (01)

RESISTANCE/UNIT LENGTH of the probe heater in milli-ohms/cm (60)

EFFECTIVE LENGTH of the probe heater in cm (102)

EFFECTIVE RADIUS of the probe heater in cm (0.546)

NUMBER OF THERMOCOUPLES in the probe to be 2 digit number (02)

START TIME is the time necessary to pass the startup transient of the probe and begin measuring thermal conductivity in seconds (1000)

FINISH TIME at which the power to the probe is to be turned off in seconds (1900)

TIME INCREMENT to be the scanned time in seconds (20)

POWER LEVEL of the probe heater in Watts/cm (0.19)

This sample file sets a test using a 1 in. (25 mm) diameter by 4.3 ft (1.3 m) long field probe containing two thermocouples and an electric heater having an electrical resistance of 60 mohm/cm, an effective length of 102 cm, and an effective radius of 0.546 cm. The thermal conductivity measurement will start at 1000 seconds after the power is supplied to the probe heater, and finish at 1900 seconds. The probe heater will be operated at a power level of 0.19 W/cm. Temperatures will be scanned, displayed on the screen and recorded on an output file every 20 seconds. The numerical values of probe parameters such as the resistance per unit length and the effective radius and length of the probe heater, can be found from the technical data relating to probe specifications provided by the manufacturer.

The power level to be applied to the probe heater for a given test is determined on the basis of the probe electrical resistance/unit length, which is dependent upon the thermal conductivity of soil at the test site. Boggs and Radhaknishna [6] developed the thermal property analyzer for measuring soil thermal resistivity and selected its power level based on the anticipated soil thermal conductivity to use one of three probe powers/unit length, which is suitable for soils of high, medium, or low thermal conductivity. The guidelines for selecting the power level are as follows:

Expected Thermal Conductivity			Suggested Power Level Per Unit Length
	Btu/h.ft.F	W/m.C	W/cm
High = >	0.96	1.7	0.36

Medium =	0.48 - 0.96	0.83 - 1.7	0.19
Low = <	0.48	0.83	0.10

Each thermal conductivity probe has a characteristic time constant which is the time required to pass the startup transient. The start time for acquiring the taest data should be equal to this time constant, and the finish time for data acquisition should be limited to within three times of this time constant.

An existing file can also be chosen as the setup file, and its contents are then read by the computer and displayed on the screen. The thermal conductivity test can be run automatically after the user interacts with the system to either select the setup file or input the probe parameters, and to name an output file for summarizing the measured results. The instrumentation system will display the start and finish times of the power supply to the probe, write the current time to the screen, read, display and record the time and surface temperatures of the probe every scan. The electric power is applied to the probe heater after a period of 200 seconds elapses for all thermocouples to attain an equilibrium state. The thermal conductivity measuring system regulates and measures the electrical current and voltage from the programmable DC power supply, and displays the desired and the actual power levels for the probe heater. For each time increment, both the time and each thermocouple temperature are continuously displayed on the screen and recorded on the output file.

When the test is finished, the best fitted values of thermal conductivity and diffusivity of soil based on equation 4 at each thermocouple location are printed on both the screen and the output file, along with the coefficient of correlation. A plot of temperature versus log of time during the test for all thermocouples attached on the probe surface can be seen on the screen by pressing the ENTER key. After this plotting is completed, the program stops for the user to view the graph. The user can choose to replot the temperature data over a different time span between the new start and finish times. A negative response by typing a "N" or "n" to the recalculation of thermal conductivity and diffusivity question returns the system to the main menu after pressing the ENTER key.

B.2 Earth Temperature Measurement

In the main menu displayed on the computer screen, option 2 is selected for measuring the earth temperatures at different depths using the arrow and ENTER keys. By responding to the program questions, the user either creates a new index file or uses an existing one stored on the disk. The details of a thermocouple index file to be created prior to executing the temperature data acquisition program, or a temperature measurement are given below. Typical user inputs are enclosed in parentheses.

FILE NAME to be 1 to 12 characters long (B:INDEX1)

NUMBER OF THERMOCOUPLES to be a number between 1 and 12 (02)

PROBE NUMBER to be any 2-digit number (01)

THERMOCOUPLE NUMBER* to be any number between 1 and 12 (01)

HORIZONTAL DISTANCE* of thermocouple from a reference pint in inches (12.0)

VERTICAL DEPTH* of thermocouple from the groun surface in inches (12.0)

THERMOCOUPLE NUMBER to be any 2-digit number (03)

HORIZONTAL DISTANCE in inches (12.0)

VERTICAL DEPTH in inches (36.0)

Note: Record numbers 4 to 6 are repeated for each additional thermocouple. This input data file states that the temperature measurement will be made using a temperature probe having two thermocouples positioned at a horizontal distance of 1 ft (0.30 m) from a reference point on the plane normal to buried pipes, and at depths of 1 and 3 ft (0.30 and 0.91m) from the ground surface, respectively.

A list of thermocouple arrangements will be displayed on the computer screen as soon as an index file is created from the user inputs or existent from the previous inputs. Soil temperature measurements are then performed automatically following a new output file named by the user. Upon updating the time and date to the screen, the system reads thermocouple outputs, and writes the thermocouple number, its temperature reading and location to both the screen and the output file. The measured values of earth temperatures can be updated or omittted by entering either "Y" or "N" from the key board when determining if another scan is needed.

To obtain more data on temperature distribution around the buried pipes, additional temperature probes are connected and carefully inserted into holes, and the preceding procedure of acquiring temperature data will be repeated. Depth temperature measurements need to be conducted on the thermally undisturbed soil situated far from the pipes. After acquiring sufficient data on the earth temperatures above and on both sides of the buried pipes, the user initiates the system to select the temperatures obtained from a probe located the farthest distance from the pipes as the undisturbed earth temperatures at various depths. Summary results of temperature measurements are automatically recorded on another new output file named by the user, and the system then returns to the main menu. This output file to be used for pipe heat loss calculations contains a tabulation of the earth temperature, the horizontal distance and depth of the measuring point, and the undisturbed earth temperature at the measurement depth, for each measuring point.

B.3 Heat Loss Calculations

Estimates of the heat loss from a two-pipe system, and of the horizontal locations and depths of the underground pipes are carried out by selecting option 3 in the main menu. The program heading, 'Buried Pipes Heat Loss Calculation Program' will appear on the screen. The user is then prompted by the system to determine if an existing data file stored in the disk is to be used or if a new data file should be created. The following information should be contained in an input data file in which the values in the parentheses are typical user inputs.

INPUT DATA FILE NAME = a file name of 1 to 12 alphanumeric characters (B:DATAFL1)

DISTANCE BETWEEN CENTERS OF PIPES (inch) = a 2 to 9-digit number with a decimal point (13.00)

NUMBER OF MEASURING LOCATION (xxx): to be any 3-digit number (002)

SOIL THERMAL CONDUCTIVITY (Btu/h·ft·F) = a 2 to 9-digit number with a decimal point (0.5240)

PROVIDE THE MODE OF INPUT OF TEST RESULTS:

1 = DATA OBTAINED DIRECTLY FROM OTHER SUBPROGRAMS AND FILES

2 = DATA INPUT THROUGH AN INTERACTIVE MANNER

MODE OF DATA INPUT (1 or 2) = a 3-digit number (002)

MEASURING LOCATION NUMBER* (xxx): any 3-digit number (001)

THE EARTH TEMPERATURE* (DEG F) = a 2 to 8-digit number with a decimal point (94.700)

HORIZONTAL DISTANCE* (inch) = a 2 to 8-digit number with a decimal point (72.000)

VERTICAL DISTANCE* (inch) = a 2 to 8-digit number with a decimal point (12.000)

UNDISTURBED EARTH TEMPERATURE* (DEG F) = a 2 to 8-digit number with decimal point (76.500)

MEASURING LOCATION NUMBER (xxx): (002)

THE EARTH TEMPERATURE (DEG F) = (135.300)

HORIZONTAL DISTANCE (inch) = (72.000)

VERTICAL DEPTH (inch) = (36.000)

UNDISTURBED EARTH TEMPERATURE (DEG F) = (70.000)

Note: The last 5 records in the input data file are repeated for each additional measuring location.

The values in the parentheses shown in this input data file are typical user inputs. If an existing file, which was created for summarizing the results of earth temperature measurements at the end of main menu option 2, is employed as the input data file, the user will be prompted to enter the file name. Its contents are then accessed and copied by the computer to a new file named by the user. The system prompts the user to supply the following information:

PROVIDE THE TYPE OF PIPE CONFIGURATION:

1 = TWO PIPES LOCATED INSIDE A SINGLE METALLIC CONDUIT

2 = TWO PIPES INSTALLED IN SEPARATE CONDUIT

TYPE OF PIPE CONFIGURATION (1 or 2) = a 3-digit number (002)

INPUT THE INITIAL PARAMETER ESTIMATES:

HEAT LOSS FROM PIPE NO. 1 (Btu/h·ft) = a 2 to 10-digit number with a decimal point (200.0)

HORIZONTAL DISTANCE OF PIPE NO. 1 (inch) = a 2 to 10-digit number with a decimal point (76.0)

VERTICAL DEPTH OF PIPE NO. 1 (inch) = a 2 to 10-digit number with a decimal point (48.0)

The numerical values in the parentheses are user inputs utilized to serve as an example. A data file can be established as the input file prior to execution of this heat loss calculation program. This existing file should contain data to specify the distance between the centers of the pipes, the number of data points, the soil thermal conductivity, the earth temperatures and their measuring locations, undisturbed earth temperatures, the type of pipe configuration, the initial estimates of the heat loss and the location of pipe No. 1, and the heat loss and vertical depth of pipe No.2. The contents of this file are read by the computer and displayed on the screen. The user names an output file to be created, and gives a diagnostics file a name to obtain detailed results of calculations or simply presses the RETURN key if a diagnostic file is not needed. The computer system performs calculations to determine statistically the values of the parameters in a theoretical expression (equation 2 or 3) for the temperature field around the underground system, and to estimate the heat losses and the locations of two insulated pipes based on the non-linear least squares method. A system warning will appear on the screen if the number of iterations exceeds the maximum allowable number of 50. When the calculations are finished, the pipe heat loss rates are printed on the screen along with the horizontal distance and vertical depth of the underground pipes. The final display is also written to the output file. The program then returns to the main menu awaiting further instructions.

```
      PROGRAM HEAT
C
C HEAT IS THE MAIN PROGRAM FOR THE UNDERGROUND PIPING SYSTEM HEAT
C LOSS DIAGONOSTICS SOFTWARE PACKAGE.   THIS PROGRAM CONTROLS ALL
C OPERATIONS OF A MICROCOMPUTER CONTROLLED INSTRUMENTATION AND DATA
C ACQUISITION SYSTEM IN A TOTALLY INTERACTIVE MANNER.   A MENU IS
C PROVIDED TO THE USER FOR SELECTION OF VARIOUS OPERATIONS INCLUDING
C MEASUREMENTS OF SOIL THERMAL CONDUCTIVITY AND GROUND TEMPERATURES
C AT DIFFERENT DEPTHS, AND CALCULATIONS OF THE HEAT LOSSES AND
C LOCATIONS OF THE DIRECT BURIED PIPES.
C THE SUBROUTINES CALLED FROM THIS PROGRAM ARE LOGO, MNMENU, THERMA,
C TEMPER, HLCALC, CURSOR AND PRT.
C
      INTEGER*2 ROW, COL, CONT
      INTEGER CHOICE
      CHARACTER*80 ITITL
      CALL LOGO
C
C CLEAR THE SCREEN AND SELECT VARIOUS OPERATIONS FROM A MAIN MENU
C
   10 ROW=0
      COL=0
      CALL CURSOR(COL,ROW)
      CALL MNMENU(CHOICE)
      IF (CHOICE .EQ. 1) THEN
         CALL THERMA
      ELSE
         IF (CHOICE .EQ. 2) THEN
            CALL TEMPER
         ELSE
            IF (CHOICE .EQ. 3) THEN
               CALL HLCALC
            ELSE
               STOP
            END IF
         END IF
      END IF
      GOTO 10
      END

      SUBROUTINE MENU(STRING,NLINES,NCHAR,MOTOP,NTITL,OPTION)
C
C SUBROUTINE MENU PROVIDES THE USER TO HILIGHT HIS OR HER CHOICE
C AMONG THE TYPE OF THERMAL MEASUREMENTS AND CALCULATIONS.  MENU
C CALLS THE ROUTINES FROM FORGRPHX.
C THE VARIABLES, NLINES, NCHAR, AND MOTOP MUST BE DECLARED AS
C INTEGER*2.
C   VARIABLES :
C      STRING - AN ARRAY OF 20 ELEMENT STRINGS. THE STRINGS MAY BE
C               UP TO 60 CHARACTERS IN LENGTH.
C      NLINES - THE NUMBER OF MENU OPTIONS
C      NCHAR  - THE NUMBER OF CHARACTERS IN THE LONGEST LINE
C      MOTOP  - THE ROW NUMBER AT WHICH THE MENU OPTION STARTS
C      NTITL  - THE NUMBER OF CHARACTERS IN THE TITLE
C      OPTION - THE OPTION SELECTED BY THE USER
C
      INTEGER*2    ROW , COL , NCHAR , NLINES , SLINES
      INTEGER*2    CONT , MOTAB ,MOTOP , SCTOP , NTITL
      INTEGER      OPTION
      CHARACTER*80 IDATA
      CHARACTER*60 STRING(20)
C
C SET SCREEN OUTLINE
C
C INITIALIZATIONS
C
C STAB - THE COLUMN NUMBER AT THE LEFT MARGIN OF THE MENU SCREEN
C MOTAB - THE COLUMN NUMBER AT THE LEFT MARGIN OF THE MENU OPTIONS
C SCTOP -THE ROW NUMBER AT WHICH THE MENU STARTS. NOTICE THAT THE
C        SUBSCRIPT OF STRING EQUALS THE ACTUAL ROW ONLY IF SCTOP
C        EQUALS ONE.
C SLINES THE NUMBER OF LINES OF THE MENU SCREEN.
C
```

```
      STAB   = 36 - (NTITL / 2)
      MOTAB    = 36 - (NCHAR / 2)
      SCTOP  =  5
      SLINES = 20
      NCHAR  = NCHAR + 1
C
C CLEAR SCREEN AND SET CURSOR POSITION
C
      ROW = 0
      COL = 0
      CALL CURSOR( COL , ROW )
      ROW = SCTOP
C
C PAINT SCREEN
C
      DO 275 I=1 , SLINES
        ITEMPO = I + SCTOP - 1
        IF (ITEMPO .GE. MOTOP) COL = MOTAB
        IF (ITEMPO .LT. MOTOP) COL = STAB
        CALL CURSOR( COL , ROW )
        WRITE(IDATA,300) STRING(I)
 300    FORMAT( A60, '$' )
        CALL PRT( IDATA )
        ROW = SCTOP + I
 275  CONTINUE
      COL = MOTAB
      OPTION = 0
      CALL RDMENU(COL, MOTOP, NLINES, NCHAR, OPTION)
      OPTION = OPTION + 1
C
C CLEAR AND NORMALIZE SCREEN
C
      DO 500 I = 1, 80
 500    CALL NORVID(I, ROW )
C
      ROW = 0
      COL = 0
      CALL CURSOR(COL, ROW )
      CALL TMODE
      RETURN
      END

      SUBROUTINE RDMENU(COL, MOTOP, NLINES, NCHAR, OFFSET)
C
C THIS SUBROUTINE RETURNS A CODE CORRESPONDING TO AN OPTION
C SELECTED FROM A MENU DISPLAYED ON SCREEN. THIS PROCEDURE
C ALLOWS VERTICAL DISPLACEMENT OF THE CURSOR IN ORDER TO
C HIGHLIGHT THE DESIRED OPTION.
C
C SUBROUTINES CALLED BY RDMENU ARE HILITE, KEYBD, CRT,AND REVVID
C
C    VARIABLES :
C       COL - THE COLUMN WHERE THE CURSOR IS LOCATED
C       MOTOP - THE ROW WHERE THE TOP OF THE OPTIONS IS LOCATED
C       NLINES - THE NUMBER OF OPTIONS AVAILABLE
C       NCHAR  - THE LENGTH OF THE LONGEST OPTION
C       OFFSET - THE LOCATION OF THE CURSOR UPON TERMINATION OF
C                THIS ROUTINE
C
      INTEGER*2 INCHAR, COL, ROW, ICOL, IROW, OLDCOL, OLDROW
      INTEGER*2 NLINES, NCHAR, OFFSET, MOTOP, MOTAB
C
C INITIALIZATIONS
C
      OLDCOL = COL
      ROW = MOTOP
C
C SET-A-CELL
C
      ICOL = COL - 1
      IROW = MOTOP
      DO 100 I = 1 , NCHAR
        JCOL = ICOL + I
```

```
  100   CALL REVVID( JCOL , IROW )
C
C WAIT FOR KEYBOARD INPUT
C
  200 CALL KEYBD( INCHAR )
      IF( INCHAR .EQ. 0 ) THEN
        CALL KEYBD( INCHAR )
        IF( INCHAR .EQ. 72 ) THEN
C
C CURSOR UP
C
          OLDROW = ROW
          IF( OFFSET .EQ. 0 ) THEN
            OFFSET = NLINES - 1
           ELSE
            OFFSET = OFFSET - 1
          END IF
          ROW = MOTOP + OFFSET
          CALL HILITE( OLDCOL , OLDROW , COL , ROW , NCHAR )
        ELSE
         IF( INCHAR .EQ. 80 ) THEN
C
C CURSOR DOWN
C
           OLDROW = ROW
           OFFSET = MOD( OFFSET+1 , NLINES )
           ROW = MOTOP + OFFSET
           CALL HILITE( OLDCOL , OLDROW , COL , ROW , NCHAR )
         ELSE
          CALL CRT( 7 )
         ENDIF
        ENDIF
      ELSE
       IF ( INCHAR .EQ. 13 ) THEN
         GO TO 400
        ELSE
         CALL CRT( 7 )
       END IF
      END IF
      GO TO 200
  400 RETURN
      END

      SUBROUTINE HILITE( OLDCOL , OLDROW , COL , ROW , NCHAR )
C
C THIS SUBROUTINE HILIGHTS A CELL BY SETTING ITS ATTRIBUTES TO REVERSE
C VIDEO WHILE RETURNING THE PREVIOUSLY HILIGHTED CELL ATTRIBUTES TO NORMAL.
C
C SUBROUTINES CALLED BY HILITE ARE NORVID AND REVVID
C
      INTEGER*2 NCHAR , COL , ROW , ICOL , IROW , OLDCOL , OLDROW
      INTEGER*2 JCOL , JROW
C
C CHANGE OLD CELL ATTRIBUTES TO NORMAL
C
      IROW  = OLDROW
      ICOL    = OLDCOL - 1
      DO 500 I = 1 , NCHAR
        JCOL = ICOL + I
      SUBROUTINE LOGO
C
C THIS SUBROUTINE PRINTS THE SOFTWARE PACKAGE TITLE, UNDERGROUND
C PIPING SYSTEM HEAT LOSS DIAGNOSTICS TO THE SCREEN.
C WARNING : LOGO MUST BE USED ONLY WITH A 'HERCULIES' BOARD .
C THE SUBROUTINES CALLED ARE GMODE, GPAGE, CLRSCR, DISP, PUTPT,
C DLINE, FILL, PRTCHAR, AND TMODE FROM FORGRPHX.ASM.
C
      INTEGER*2 X,Y,ROW,COL,NX,NY,N,WIDTH,LENGTH
      INTEGER*2 I,J
      CHARACTER*54 IDATA
      CALL GMODE
      CALL GPAGE(1)
      CALL CLRSCR
```

```
      CALL DISP(1)
C DIAMOND
      COL = 400
      ROW = 90
      CALL PUTPT(COL,ROW)
      I = 550
      J = 15
      CALL DLINE(I,J)
      I = 700
      J = 90
      CALL DLINE(I,J)
      I = 550
      J = 165
      CALL DLINE(I,J)
      CALL DLINE(COL,ROW)
C FAR LEFT DIAMOND
      I = 470
      J = 55
      CALL PUTPT(I,J)
      J = 125
      CALL DLINE(I,J)
   90 FORMAT(A1)
C     TOP BARRIER
      I = 480
      J = 50
      CALL PUTPT(I,J)
      I = I + 4
      CALL DLINE(I,J)
      J = J + 12
      CALL DLINE(I,J)
      I = 506
      J = 92
      CALL DLINE(I,J)
      J = J - 42
      CALL DLINE(I,J)
      I = 570
      CALL DLINE(I,J)
      J = J + 10
      I = I + 10
      CALL DLINE(I,J)
      J = J + 20
      CALL DLINE(I,J)
      I = I - 10
      J = J + 10
      CALL DLINE(I,J)
      I = I + 10
      J = J + 10
      CALL DLINE(I,J)
      J = J + 20
      CALL DLINE(I,J)
      I = I - 10
      J = J + 10
      CALL DLINE(I,J)
C AROUND LEFT SIDE OF B, DO THE RIGHT SIDE OF S (BOTTOM UP)
      I = I + 30
      CALL DLINE(I,J)
      I = I - 10
      J = J - 10
      CALL DLINE(I,J)
      J = J - 10
      CALL DLINE(I,J)
      I = I + 14
      CALL DLINE(I,J)
      J = J + 6
      CALL DLINE(I,J)
      I = I + 22
      CALL DLINE(I,J)
      J = J - 10
      CALL DLINE(I,J)
      J = J - 8
      I = I - 12
      CALL DLINE(I,J)
      I = I - 12
```

```
      J = J - 8
      CALL DLINE(I,J)
      I = I - 10
      J = J - 10
      CALL DLINE(I,J)
      J = J - 20
      CALL DLINE(I,J)
      J = J - 10
      I = I + 10
      CALL DLINE(I,J)
      I = I + 20
      CALL DLINE(I,J)
C BOTTOM BARRIER
      I = 480
      J = 130
      CALL PUTPT(I,J)
      I = I + 4
      CALL DLINE(I,J)
      J = J - 42
      CALL DLINE(I,J)
      I = 506
      J = 118
      CALL DLINE(I,J)
      J = J + 12
      CALL DLINE(I,J)
      I = 520
      CALL DLINE(I,J)
      J = J - 80
      CALL DLINE(I,J)
      I = I + 10
      CALL DLINE(I,J)
      J = J + 80
      CALL DLINE(I,J)
      I = I + 90
      CALL DLINE(I,J)
C RECTANGLES IN THE MIDDLE OF THE B
      N = 0
      J = 64
   10 N = N + 1
      I = 544
      CALL PUTPT(I,J)
      I = I + 22
      CALL DLINE(I,J)
      J = J + 16
      CALL DLINE(I,J)
      I = I - 22
      CALL DLINE(I,J)
      J = J - 16
      CALL DLINE(I,J)
C SECOND RECTANGLE
      J = J + 36
      CALL PUTPT(I,J)
      IF (N .EQ. 1) GO TO 10
C OUTSIDE CURVE OF THE S
      I = 640
      J = 60
      CALL PUTPT(I,J)
      J = J + 10
      CALL DLINE(I,J)
      I = I - 14
      CALL DLINE(I,J)
      J = J - 6
      CALL DLINE(I,J)
      I = I - 22
      CALL DLINE(I,J)
      J = J + 10
      CALL DLINE(I,J)
      J = J + 8
      I = I + 8
      CALL DLINE(I,J)
      J = J + 8
      I = I + 8
      CALL DLINE(I,J)
```

```
      J = J + 2
      I = I + 10
      CALL DLINE(I,J)
      I = I + 10
      J = J + 10
      CALL DLINE(I,J)
      J = J + 20
      CALL DLINE(I,J)
C FILL IN THE BACKGROUND
      COL = 410
      ROW = 90
      CALL FILL(COL,ROW)
      COL = 550
      ROW = 140
      CALL FILL(COL,ROW)
      COL = 650
      ROW = 90
      CALL FILL(COL,ROW)
      COL = 485
      ROW = 110
      CALL FILL(COL,ROW)
      COL = 500
      ROW = 70
      CALL FILL(COL,ROW)
      COL = 550
      ROW = 70
      CALL FILL(COL,ROW)
      COL = 550
      ROW = 110
      CALL FILL(COL,ROW)
      COL = 585
      ROW = 100
      CALL FILL(COL,ROW)
      COL = 610
      ROW = 65
      CALL FILL(COL,ROW)
      COL = 610
      ROW = 115
      CALL FILL(COL,ROW)
C
C UNDERGROUND
C
      CALL PRTCHAR(25,50,85,2,3)
      CALL PRTCHAR(50,50,110,2,3)
      CALL PRTCHAR(75,50,100,2,3)
      CALL PRTCHAR(100,50,101,2,3)
      CALL PRTCHAR(125,50,114,2,3)
      CALL PRTCHAR(150,50,103,2,3)
      CALL PRTCHAR(175,50,114,2,3)
      CALL PRTCHAR(200,50,111,2,3)
      CALL PRTCHAR(225,50,117,2,3)
      CALL PRTCHAR(250,50,110,2,3)
      CALL PRTCHAR(275,50,100,2,3)
C PIPING SYSTEM
      CALL PRTCHAR(150,140,80,2,3)
      CALL PRTCHAR(175,140,105,2,3)
      CALL PRTCHAR(200,140,112,2,3)
      CALL PRTCHAR(225,140,105,2,3)
      CALL PRTCHAR(250,140,110,2,3)
      CALL PRTCHAR(275,140,103,2,3)
C SPACE
      CALL PRTCHAR(325,140,83,2,3)
      CALL PRTCHAR(350,140,121,2,3)
      CALL PRTCHAR(375,140,115,2,3)
      CALL PRTCHAR(400,140,116,2,3)
      CALL PRTCHAR(425,140,101,2,3)
      CALL PRTCHAR(450,140,109,2,3)
C HEAT LOSS
      CALL PRTCHAR(275,230,72,2,3)
      CALL PRTCHAR(300,230,101,2,3)
      CALL PRTCHAR(325,230,97,2,3)
      CALL PRTCHAR(350,230,116,2,3)
C SPACE
```

```
      CALL PRTCHAR(400,230,76,2,3)
      CALL PRTCHAR(425,230,111,2,3)
      CALL PRTCHAR(450,230,115,2,3)
      CALL PRTCHAR(475,230,115,2,3)
C DIAGNOSTICS
      CALL PRTCHAR(425,320,68,2,3)
      CALL PRTCHAR(450,320,105,2,3)
      CALL PRTCHAR(475,320,97,2,3)
      CALL PRTCHAR(500,320,103,2,3)
      CALL PRTCHAR(525,320,110,2,3)
      CALL PRTCHAR(550,320,115,2,3)
      CALL PRTCHAR(575,320,116,2,3)
      CALL PRTCHAR(600,320,105,2,3)
      CALL PRTCHAR(625,320,99,2,3)
      CALL PRTCHAR(650,320,115,2,3)
      READ(*,90)
      CALL TMODE
      RETURN
      END


  500   CALL NORVID( JCOL , IROW )
C
C CHANGE NEW CELL ATTRIBUTES TO REVERSE VIDEO
C
      IROW = ROW
      ICOL = COL - 1
      DO 600 I = 1 , NCHAR
        JCOL = ICOL + I
  600   CALL REVVID( JCOL , IROW )
      RETURN
      SUBROUTINE MNMENU(OPTION)
C THIS SUBROUTINE PROVIDES A MAIN MENU LISTING ALL OF THE OPERATIONS
C TO BE CHOSEN BY THE USER TO CARRY OUT SOIL THERMAL CONDUCTIVITY AND
C TEMPERATURE MEASUREMENTS AND HEAT LOSS CALCULATIONS.
C THE SUBROUTINE CALLED IS MENU.
C  VARIABLES :
C    STRING - AN ARRAY OF 20 ELEMENT STRINGS WITH A LENGTH OF UP TO
C             60 CHARACTERS.
C    NLINES - TOTAL NUMBER OF THE MENU OPTIONS.
C    NCHAR - THE NUMBER OF CHARACTERS IN THE LONGEST LINE.
C    MOTOP - THE ROW NUMBER AT WHICH THE MENU OPTIONS START
C    NTITL - TOTAL NUMBER OF CHARACTERS IN THE TITLE
C
      INTEGER*2 NLINES,NCHAR,MOTOP,NTITL
      INTEGER OPTION
      CHARACTER*60 STRING(20)
      DATA (STRING(I),I=1,20) / 20*' '/
      DATA STRING(1) /' MAIN MENU '/
      DATA STRING(2) /' ---------- '/
      DATA STRING(4) /' 1 : DETERMINE THE SOIL THERMAL CONDUCTIVITY '/
      DATA STRING(5) /' 2 : MEASURE THE GROUND TEMPERATURES '/
      DATA STRING(6) /' 3 : CALCULATE HEAT LOSSES FROM BURIED PIPES '/
      DATA STRING(7) /' 4 : EXIT '/
      DATA STRING(10) /' PLEASE RESPOND BY HIGHLIGHTING YOUR CHOICE '/
      NLINES = 4
      NCHAR = 54
      MOTOP = 8
      NTITL = 11
      CALL MENU(STRING, NLINES, NCHAR, MOTOP, NTITL, OPTION)
      RETURN
      END


      SUBROUTINE THERMA
C
C THERMA READS THERMOCOUPLES OF THE PROBE AND CALCULATES THE SOIL
C THERMAL CONDUCTIVITY AND THERMAL DIFFUSIVITY AT EACH THERMOCOUPLE
C LOCATION. UP TO FOUR THERMOCOUPLE INPUT CHANNELS CAN BE USED IN
C THIS PROGRAM. THE OUTPUT DATA IS STORED IN A FILE NAMED BY THE
C USER.  THE SUBROUTINES CALLED FROM THIS PROGRAM ARE INITAL, DEGREE,
C GETFL, MAKEFL, TEMPDT, CALC, CLOCK, PRTCLK, DATAFL AND ROUTINES
C FROM THE FILES 'KEY.ASM' AND 'FORGRPHX.ASM'.
```

```
C
C VARIABLES
C     DATA - DATA HOLDS THE TEMPERATURE READINGS
C     ROW  - SPECIFIES THE ROW ON THE SCREEN WHERE INFORMATION IS TO
C            BE WRITTEN
C     COL  - SPECIFIES THE COLUMN ON THE SCREEN WHERE INFORMATION IS
C             TO BE WRITTEN
C
      CHARACTER*43 IDATA
      CHARACTER*1 ANSW,ANSWR
      INTEGER PROBE,DIST,LENGTH,START,FINISH,INC,TC,TIME,RESLEN
      INTEGER*2 ROW,COL,INCHAR
      REAL RADIUS,POWER,DATA(4,1000),PIE,POWR
      COMMON /VARS/ TIME,POWER,RADIUS,PIE,GAMMA,INC,TC,START,FINISH
      COMMON /ATURE/ DATA
      COMMON /HEATR/ POWR,LENGTH
      PIE = 3.141593
      GAMMA = 0.5772
C
C CLEAR SCREEN
C
      ROW = 0
      COL = 0
      CALL CURSOR(COL,ROW)
      ROW=1
      COL=1
      CALL CURSOR(COL,ROW)
C
C DETERMINE IF AN OLD TEST SETUP FILE IS TO BE USED OR A NEW SETUP
C FILE SHOULD BE CREATED.
C
      ROW = 5
      COL =1
      CALL CURSOR(COL,ROW)
      WRITE(*,500)
  500 FORMAT(20X,'UNDERGROUND DIRECT BURIED PIPE ANALYSIS PROGRAM',///)
   10 WRITE(*,100)
      READ(*,90) ANSW
      IF ((ANSW .EQ. 'Y') .OR. (ANSW .EQ. 'y')) THEN
         CALL GETFL(ANSWR)
      ELSE
         IF ((ANSW .EQ. 'N') .OR. (ANSW .EQ. 'n')) THEN
            CALL MAKEFL(ANSWR)
         ELSE
           WRITE(*,80)
   80      FORMAT(' Please try again (answer either Y or N).')
           GO TO 10
         END IF
      END IF
      IF ((ANSWR .EQ. 'N') .OR. (ANSWR .EQ. 'n')) GO TO 999
      CALL DATAFL(10,ANSWR)
      IF ((ANSWR .EQ. 'N') .OR. (ANSWR .EQ. 'n')) GO TO 999
C
C CALCULATE THE REQUESTED POWER FROM THE PROBE POWER LEVEL
C
      POWER = POWER * LENGTH
      ROW = 22
      COL = 1
      CALL CURSOR(COL,ROW)
      WRITE(*,5)
    5 FORMAT(' NOTE : IF THE TERMINAL BEEPS PLEASE REBOOT TO',
     *' REINITIALIZE')
      WRITE(*,90)
      CALL INITAL
C
C
  501 FORMAT(20X,'DATA ACQUISITION BOARDS INITIALIZED',/)
      CALL TEMPDT
      CALL CALC
      ROW = 24
      COL = 1
      CALL CURSOR(COL,ROW)
      WRITE(*,85)
```

```
      READ(*,90) ANSW
   85 FORMAT(' Please press RETURN to return to the menu. ')
   90 FORMAT(A1)
   91 FORMAT(I2)
  100 FORMAT(' Would you like to use an existing setup  file (Y/N) ? ')
  999 RETURN
      END

      SUBROUTINE TEMPDT
C
C SUBROUTINE TEMPDT GATHERS THE TEMPERATURE DATA DURING THE SPECIFIED
C TIME PERIOD. THIS SUBPROGRAM CAN HANDLE UP TO 4 INPUT CHANNELS. THE
C POWER TO THE PROBE IS TURNED ON AND THEN OFF (ACCORDING TO DELAY AND
C FINISH TIME) DURING THE DATA AQUISITION. THE SUBROUTINES CALLED BY
C THIS ROUTINE ARE CLOCK, DEGREE, POWERON, PRTCLK, DIGANA, CURSOR AND
C PRT.
C
C VARIABLES
C    BEGIN - BEGIN HOLDS THE TIME TO START THE POWER TO THE PROBE
C            BEGIN(2) = SEC, BEGIN(1) = MIN, BEGIN(0) = HOURS.
C    STOP  - STOP HOLDS THE TIME TO STOP THE POWER TO THE PROBE
C            STOP(2) = SEC, STOP(1) = MIN, STOP(0) = HOURS.
C    GET   - GET INDICATES THE SECOND ON WHICH THE NEXT SET OF DATA
C            SHOULD BE OBTAINED.
C    TIME  - TIME HOLDS THE NUMBER OF TIMES THE THERMOCOUPLES HAVE
C            BEEN READ (THE NUMBER OF ITEMS IN THE DATA ARRAY)
C    DATA  - DATA IS AN ARRAY OF ALL THE DATA READ
C    ROW   - INDICATES THE ROW ON THE SCREEN INFORMATION IS TO BE WRITTEN TO
C    COL   - INDICATES THE COLUMN ON THE SCREEN THE INFORMATION IS TO
C            BE WRITTEN TO
C
      INTEGER*2 JD(7),ROW,COL,RRWIN,CRWIN,RLWIN,CLWIN
      INTEGER PROBE,DIST,LENGTH,START,FINISH,INC,TC,TIME,RESLEN,
     *GET,STOP(0:2),BEGIN(0:2),TIM,DELAY
      REAL RADIUS,POWER,DATA(4,1000),TEMP(16),PIE,POWR
      CHARACTER IDATA*15, IDTAB*46
      COMMON /VARS/ TIME,POWER,RADIUS,PIE,GAMMA,INC,TC,START,FINISH
      COMMON /ATURE/ DATA
      COMMON /HEATR/ POWR,LENGTH
      DATA RRWIN/25/,CRWIN/80/,RLWIN/9/,CLWIN/1/
      TIME = 1
      TIM = 0
      WRITE(10,10) (I,I=1,TC)
   10 FORMAT(2X,'TIME',4X,4(2X,'TC#',I1,3X))
      CALL CLOCK(JD)
      CALL PRTCLK(JD)
C
C CALCULATE THE TIME TO START THE POWER TO THE PROBE
C
      DELAY = 200
      IJK = JD(6) + DELAY
      BEGIN(2) = MOD(IJK,60)
      BEGIN(1) = IJK / 60 + JD(5)
      IJK = BEGIN(1)
      KIJK = IJK / 60
      BEGIN(0) = JD(4) + KIJK
      BEGIN(1) = MOD(IJK,60)
      BEGIN(0) = MOD(BEGIN(0),24)
C
C CALCULATE THE TIME TO STOP THE POWER TO THE PROBE
C
      IJK = JD(6) + FINISH
      STOP(2) = MOD(IJK,60)
      STOP(1) = IJK / 60 + JD(5)
      IJK = STOP(1)
      KIJK = IJK / 60
      STOP(0) = JD(4) + KIJK
      STOP(1) = MOD(IJK,60)
      STOP(0) = MOD(STOP(0),24)
      GET = JD(6)
C
C CLEAR SCREEN
C
```

```
C CLEAR SCREEN
      ROW = 0
      COL = 0
      CALL CURSOR(COL,ROW)
      ROW=2
      COL=1
      CALL CURSOR(COL,ROW)
      WRITE(* ,500) BEGIN,STOP
  500 FORMAT(1H+,5X,'START TIME : ',I2,':',I2.2,':',I2.2,5X,
     *'STOP TIME : ',I2,':',I2.2,':',I2.2)
      ROW=3
      COL= 10
      CALL CURSOR(COL,ROW)
      WRITE(IDATA,15)
   15 FORMAT('GATHERING DATA$')
      CALL PRT(IDATA)
      ROW = 7
      COL = 1
      CALL CURSOR(COL,ROW)
      GO TO (416,417,418,419),TC
  416 WRITE(IDTAB,201)
      GOTO 420
  417 WRITE(IDTAB,202)
      GOTO 420
  418 WRITE(IDTAB,203)
      GOTO 420
  419 WRITE(IDTAB,204)
  420 CALL PRT(IDTAB)
      ROW = ROW + 1
      CALL CURSOR(COL,ROW)
      WRITE(IDTAB,421)
      CALL PRT(IDTAB)
C
C READ THE SURFACE TEMPERATURES OF THE PROBE
C
   20 CALL DEGREE(TEMP)
      DO 25 J = 1,TC
         DATA(J,TIME) = TEMP(J)
   25 CONTINUE
      TIM = TIM + INC
      WRITE(10,100) TIM,(DATA(J,TIME),J=1,TC)
  100 FORMAT(1X,I5,3X,4(F8.1,1X))
      ROW = ROW + 1
      IF(ROW.GT.25) THEN
      ROW = 25
      CALL SCRLUP(RLWIN,CLWIN,RRWIN,CRWIN)
      ENDIF
      COL=1
      WRITE(IDTAB,320) TIM
  320 FORMAT(1X,I5,'$')
      CALL CURSOR(COL,ROW)
      CALL PRT(IDTAB)
      DO 50 L=1,TC
      WRITE(IDTAB,220) DATA(L,TIME)
      COL=9+(TC-1)*9
      CALL CURSOR(COL,ROW)
      CALL PRT(IDTAB)
  220 FORMAT(F8.1,1X,'$')
   50 CONTINUE
      COL=COL+9
      CALL CURSOR(COL,ROW)
      IDTAB=' F$'
      IDTAB(1:1)=CHAR(248)
      CALL PRT(IDTAB)
C
C CALCULATE THE NEXT TIME TO READ THE TEMPERATURES
C
      GET = GET + INC
      GET = MOD(GET,60)
      TIME = TIME + 1
   30 CALL CLOCK(JD)
C
C WRITE THE CURRENT TIME TO THE SCREEN
```

```
C
      CALL PRTCLK(JD)
C
C CHECK TO SEE IF IT IS TIME TO TERMINATE THE PROGRAM, READ A TEMPERATURE
C OR WAIT FOR TIME TO ADVANCE.
C
      IF ((JD(4).EQ.BEGIN(0)).AND.(JD(5) .EQ. BEGIN(1)) .AND.
     * (JD(6) .EQ. BEGIN(2))) CALL POWERON(POWER,POWR)
      IF (JD(4).GT.STOP(0)) GO TO 40
      IF(JD(4).EQ.STOP(0).AND.JD(5).EQ.STOP(1).AND.JD(6).GT.STOP(2))
     *GO TO 40
      IF(JD(4).EQ.STOP(0).AND.JD(5).GT.STOP(1)) GO TO 40
      IF (JD(6) .EQ. GET) GO TO 20
      GO TO 30
C  TURN OFF THE POWER TO THE HEATER
   40 JDATA = 0
      ICHAN = 0
      CALL DIGANA(JDATA,ICHAN,IGAIN,IERROR)
      ICHAN = 1
      CALL DIGANA(JDATA,ICHAN,IGAIN,IERROR)
      TIME = TIME - 1
  201 FORMAT('  TIME      TC#1                                 $')
  202 FORMAT('  TIME      TC#1      TC#2                       $')
  203 FORMAT('  TIME      TC#1      TC#2      TC#3             $')
  204 FORMAT('  TIME      TC#1      TC#2      TC#3      TC#4   $')
  421 FORMAT('-----------------------------------------------$')
      RETURN
      END

      SUBROUTINE CALC
C
C THIS SUBROUTINE CALCULATES THE THERMAL CONDUCTIVITY AND THERMAL
C DIFFUSIVITY OF SOIL AT EACH THERMOCOUPLE LOCATION. THE SUBROUTINES
C CALLED ARE  CURSOR AND PRT IN THE FILE 'FORGRPHX.ASM'.
C
      REAL PIE,GAMMA,RADIUS,POWER,SLOPE(4),INTER(4),DATA(4,1000),
     *TICK,KS(4),ALPHA(4),POWR
      CHARACTER*80 IDATA
      CHARACTER*1 ANS
      INTEGER*2 ROW,COL,X,Y
      INTEGER LENGTH,START,FINISH,INC,TC,TIME,TIMHTR,DELAY,NSYMB(4)
      COMMON /VARS/ TIME,POWER,RADIUS,PIE,GAMMA,INC,TC,START,FINISH
      COMMON /ATURE/ DATA
      COMMON /HEATR/ POWR,LENGTH
      COMMON /PLTDAT/ YMAX,YMIN,XMAX,XMIN
      DATA NSYMB/4,254,88,43/
C
C CLEAR SCREEN
C
       DELAY=200
    2  ROW = 0
       COL = 0
       CALL CURSOR(COL,ROW)
       WRITE(IDATA,570) START
  570 FORMAT(10X,'Start = ',I4,' Seconds $')
       ROW =3
       COL = 1
       CALL PRT(IDATA)
       WRITE(10,580) START
  580 FORMAT(10X,'Start = ',I4,' Seconds')
       ROW = 5
       COL = 5
       CALL CURSOR(COL,ROW)
       WRITE(IDATA,100)
       CALL PRT(IDATA)
       ROW = ROW + 1
       WRITE(IDATA,150)
       CALL CURSOR(COL,ROW)
       CALL PRT(IDATA)
       ROW = ROW + 1
       WRITE(IDATA,160)
       CALL CURSOR(COL,ROW)
       CALL PRT(IDATA)
```

```
      ROW = ROW + 1
      WRITE(IDATA,175)
      CALL CURSOR(COL,ROW)
      CALL PRT(IDATA)
      WRITE(10,510)
      WRITE(10,520)
      IDELAY=(START+DELAY)/INC
      TIMHTR=DELAY/INC
      DO 1 J=1,TC
        NDATA=0
        SXY = 0
        SX = 0
        SY = 0
        S2X = 0
        S2Y = 0
        DO 10 I=IDELAY,TIME
          II=I-TIMHTR
          TICK = ALOG(REAL(II * INC))
          SXY = SXY + (DATA(J,I) * TICK)
          SY = SY + DATA(J,I)
          SX = SX + TICK
          S2Y = S2Y + (DATA(J,I) * DATA(J,I))
          S2X = S2X + (TICK * TICK)
          NDATA=NDATA+1
 10     CONTINUE
C
C CALCULATE THE COEFFICIENT OF CORRELATION, R SQUARED
C
        VART = NDATA * S2X - SX * SX
        VARV = NDATA * S2Y - SY * SY
        SLOPE(J) = (NDATA * SXY - (SX * SY)) / VART
        INTER(J) = (SY - SLOPE(J) * SX) / NDATA
        R2 = SLOPE(J) * SLOPE(J) * VART / VARV
C
C CONVERSION OF SI UNITS TO ENGINEERING UNITS
C
        RADFT = RADIUS / 30.48
        POWRR = 3.4144 * POWR
C
C CALCULATE SOIL THERMAL CONDUCTIVITY
C
        KS(J) = (POWRR * 30.48) / (4 * PIE * SLOPE(J) * LENGTH)
C
C CALCULATE THERMAL DIFFUSIVITY
C
        ALPHA(J) = RADFT * RADFT / 4 * EXP(INTER(J)/SLOPE(J) + GAMMA)
C
C WRITE THE THERMAL PROPERTIES OF SOIL
C
        ROW = ROW + 2
        CALL CURSOR(COL,ROW)
        WRITE(IDATA,200) J,KS(J),ALPHA(J),R2
        CALL PRT(IDATA)
        WRITE(10,550) J,KS(J),ALPHA(J),R2
  1 CONTINUE
C
C PAUSE TO VIEW DATA
C
     ROW =ROW +3
     CALL CURSOR(COL,ROW)
     WRITE(IDATA,650)
     CALL PRT(IDATA)
     READ(*,560) ANS
C
C FIND MAX & MIN VALUES FOR GRAPH
C
     YMAX=DATA(1,1)
     YMIN=DATA(1,1)
     XMAX=10.
     XMIN=10.
     DO 60 K=1,TIME
     DO 60 J=1,TC
       IF(DATA(J,K).LT.YMIN) YMIN=DATA(J,K)
```

```
      IF(DATA(J,K).GT.YMAX) YMAX=DATA(J,K)
 60 CONTINUE
    YTEMP=-50.
  3 IF(YTEMP.GT.YMIN) GO TO 4
    YTEMP=YTEMP+50.
    GO TO 3
  4 YMIN=YTEMP-50
  6 IF(YTEMP.GT.YMAX) GO TO 7
    YTEMP=YTEMP+50.
    GO TO 6
  7 YMAX=YTEMP
    XTIME=INC*TIME
  8 IF(XMAX.GT.XTIME) GO TO 9
    XMAX=XMAX*10
    GO TO 8
  9 CONTINUE
    CALL TMODE
    CALL GMODE
    CALL GPAGE(1)
    CALL LEVEL(1)
    CALL CLRSCR
    CALL GTEMP
    WRITE(IDATA,585) START
585 FORMAT('Start = ',I4,' Sec$')
    X=90
    Y=25
    NX=1
    NY=1
    NV=0
    CALL PRTTXT(X,Y,IDATA,NX,NY,NV)
    DO 420 J=1,TC
    WRITE(IDATA,590) J
590 FORMAT('TC # ',I1,'    $')
    Y=Y+12
    CALL PRTTXT(X,Y,IDATA,NX,NY,NV)
    X=90+64
    Y=Y+6
    CALL TEXT(X,Y,NSYMB(J))
    Y=Y-6
    X=90
420 CONTINUE
    Y=Y+20
    X=90+12*8
    CALL TEXT(X,Y,75)
    X=90+22*8
    CALL TEXT(X,Y,224)
    Y=Y-6
    DO 425 J=1,TC
    WRITE(IDATA,596) J,KS(J),ALPHA(J)
596 FORMAT('TC # ',I1,2X,F6.3,2X,1PE10.3,'$')
    Y=Y+12
    X=90
    CALL PRTTXT(X,Y,IDATA,NX,NY,NV)
425 CONTINUE
    DO 400 J=1,TC
       ISEC=INC
       DO 410 L=TIMHTR+1,TIME
         ISEC=INC*(L-TIMHTR)
         CALL PLTCHR(DATA(J,L),ISEC,NSYMB(J))
410    CONTINUE
       XX=INC
       XX=XX
       TEMP1=SLOPE(J)*ALOG(XX)+INTER(J)
       ISEC=INC
       IF(TEMP1.LT.YMIN) THEN
         XX=(YMIN-INTER(J))/SLOPE(J)
         XX=EXP(XX)
         ISEC=XX
         TEMP1=YMIN
       ENDIF
       CALL SETTEM(TEMP1,ISEC)
       XX=(TIME-TIMHTR)*INC
       XX=XX
```

```
         TEMP2=SLOPE(J)*ALOG(XX)+INTER(J)
         ISEC=XX
         CALL PLTTEM(TEMP2,ISEC)
  400 CONTINUE
      READ(*,560) ANS
      CALL TMODE
      ROW = 0
      COL = 0
      CALL CURSOR(COL,ROW)
      ROW = 5
      COL = 10
      CALL CURSOR(COL,ROW)
      IDATA='Recalculate Thermal Conductiviy & Diffusivity (Y/N) ? $'
      CALL PRT(IDATA)
      READ(*,560) ANS
      IF(ANS.EQ.'Y'.OR.ANS.EQ.'y') THEN
        ROW= 7
        COL = 10
        CALL CURSOR(COL,ROW)
        IDATA= 'Start (Seconds) = $'
        CALL PRT(IDATA)
        READ(*,*) START
        GO TO 2
      ENDIF
  560 FORMAT(A1)
  650 FORMAT(5X,'Type Return to view graph $')
  100 FORMAT('  THERMO-   SOIL THERMAL      THERMAL      C.C.$')
  150 FORMAT('  COUPLE    CONDUCTIVITY    DIFFUSIVITY     R2 $')
  160 FORMAT('    NO.     (BTU/H-FT-F)     (FT**2/H)         $')
  175 FORMAT('  ---------------------------------------------$')
  200 FORMAT(5X,I1,7X,F9.4,5X,F10.6,4X,F5.4,'$')
  300 FORMAT(A10)
  510 FORMAT('  THERMO-   SOIL THERMAL      THERMAL    C.C. ',
     *   /'  COUPLE    CONDUCTIVITY    DIFFUSIVITY    R2  ')
  520 FORMAT('    NO.     (BTU/H-FT-F)     (FT**2/H)',
     *   /2X,'--------------------------------------------')
  550 FORMAT(5X,I1,7X,F9.4,5X,F10.6,4X,F5.4)
      RETURN
      END
      SUBROUTINE MAKEFL(ANSWR)
C
C SUBROUTINE MAKEFL HELPS THE USER TO CREATE A PARAMETER FILE FOR
C EXECUTING THE THERMA PROGRAM. MAKEFL CALLS THE SUBROUTINE CURSOR.
C
C VARIABLES
C   NAME   - THE NAME OF THE TEST SETUP FILE TO BE CREATED
C   PROBE  - THE PROBE SERIAL NUMBER
C   RESLEN - THE RESISTANCE PER UNIT LENGTH OF THE HEATING ELEMENT
C            (mohm/cm)
C   LENGTH - THE EFFECTIVE LENGTH OF THE HEATING ELEMENT (cm)
C   RADIUS - THE EFFECTIVE RADIUS OF THE HEATING ELEMENT (cm)
C   TC     - THE NUMBER OF THERMOCOUPLES ON THE SURFACE OF THE PROBE
C   START  - THE START TIME IS THE TIME NECESSARY TO PASS THE STARTUP
C            TRANSIENT AND BEGIN MEASURING THERMAL CONDUCTIVITY (sec)
C   FINISH - THE TIME AT WHICH POWER TO THE PROBE IS TO BE TURNED
C            OFF (sec)
C   POWER  - THE POWER LEVEL OF THE PROBE HEATER (W/cm)
C   ROW    - THE ROW NUMBER WHERE INFORMATION IS TO BE WRITTEN
C   COL    - THE COLUMN NUMBER WHERE INFORMATION IS TO BE WRITTEN
C
      CHARACTER*1 ANSWR
      CHARACTER*10 NAME
      INTEGER*2 ROW,COL
      INTEGER PROBE,DIST,LENGTH,START,FINISH,INC,TC,TIME,RESLEN
      REAL RADIUS,POWER
      COMMON /VARS/ TIME,POWER,RADIUS,PIE,GAMMA,INC,TC,START,FINISH
      COMMON /HEATR/ POWR,LENGTH
      I = 0
C
C GET THE INFORMATION NEEDED FROM THE USER
C
   10 WRITE(*,100)
      READ(*,90) NAME
```

```
C
C CLEAR SCREEN
C
      ROW = 0
      COL = 0
      CALL CURSOR(COL,ROW)
      OPEN(11,FILE=NAME,STATUS='NEW',ERR=99)
      WRITE(11,100) NAME
      WRITE(*,110)
      READ(*,91) PROBE
      WRITE(11,110) PROBE
      WRITE(*,120)
      READ(*,93) RESLEN
      WRITE(11,120) RESLEN
      WRITE(*,130)
      READ(*,93) LENGTH
      WRITE(11,130) LENGTH
      WRITE(*,140)
      READ(*,94) RADIUS
      WRITE(11,140) RADIUS
    6 WRITE(*,150)
      READ(*,91) TC
      IF ((TC .LT. 1) .OR. (TC .GT. 4)) GO TO 6
      WRITE(11,150) TC
      WRITE(*,160)
      READ(*,95) START
      WRITE(11,160) START
      WRITE(*,170)
      READ(*,95) FINISH
      WRITE(11,170) FINISH
    9 WRITE(*,180)
      READ(*,92) INC
      IF ((INC .LT. 10) .OR. (INC .GT. 90)) GO TO 9
      WRITE(11,180) INC
      WRITE(*,190)
      READ(*,94) POWER
      WRITE(11,190) POWER
      WRITE(*,91)
      TIME = FINISH - START
   89 FORMAT(A1)
   90 FORMAT(A10)
   91 FORMAT(I2)
   92 FORMAT(I3)
   93 FORMAT(I4)
   94 FORMAT(F10.6)
   95 FORMAT(I5)
  100 FORMAT(' FILE NAME :',23X,A10)
  110 FORMAT(' PROBE SERIAL NUMBER  (XX):',8X,I2)
  120 FORMAT(' RESISTANCE/UNIT LENGTH (mohm/cm): ',I4)
  130 FORMAT(' EFFECTIVE LENGTH (cm) :',11X,I4)
  140 FORMAT(' EFFECTIVE RADIUS (cm) :',11X,F10.5)
  150 FORMAT(' NUMBER OF THERMOCOUPLES (XX):',4X,I2)
  160 FORMAT(' START TIME (sec) :',16X,I5)
  170 FORMAT(' FINISH TIME (sec) :',15X,I5)
  180 FORMAT(' TIME INCREMENT (sec) :',12X,I3)
  190 FORMAT(' POWER LEVEL (W/cm) :',14X,F10.5)
      RETURN
   99 WRITE(*,200)
      WRITE(*,90)
      I = I + 1
      IF (I .GE. 4) THEN
         WRITE(*,220)
         WRITE(*,90)
         READ(*,89) ANSWR
      END IF
      IF ((ANSWR .EQ. 'N') .OR. (ANSWR .EQ. 'n')) GO TO 98
      GO TO 10
  200 FORMAT(' The file you wish to create cannot be created by DOS.')
  210 FORMAT(' Please try again.')
  220 FORMAT(' Press Y  to continue trying to get a valid name, or press
     * N to exit (Y/N).')
   98 RETURN
      END
```

```
      SUBROUTINE DATAFL(NUNIT,ANSWR)
C
C DATAFL OBTAINS THE NAME OF THE DATAFL FROM THE USER.
C
      CHARACTER*1 ANSWR
      CHARACTER*10 NAME1
      DATA NAME1 /' '/
   50 WRITE(*,90)
      WRITE(*,60)
      WRITE(*,90)
   60 FORMAT(' Please enter the name of the OUTPUT file. ')
      READ(*,90) NAME1
      OPEN(NUNIT,FILE=NAME1,STATUS='NEW',ERR=66)
      RETURN
C
C ALLOW THE USER TO EXIT IF HE OR SHE CANNOT CREATE A FILE
C
   66 WRITE(*,200)
      WRITE(*,90)
      J = J + 1
      IF (J .GE. 4) THEN
         WRITE(*,220)
         WRITE(*,90)
         READ(*,89) ANSWR
      END IF
      IF ((ANSWR .EQ. 'N') .OR. (ANSWR .EQ. 'n')) GO TO 98
      GO TO 50
   89 FORMAT(A1)
   90 FORMAT(A10)
  100 FORMAT(' OUTPUT FILE NAME :',5X,A10)
  200 FORMAT(' The file you wish to create cannot be created by DOS.')
  210 FORMAT(' Please try again.')
  220 FORMAT(' Press Y  to continue trying to get a valid name, or press
     * N to exit. (Y/N)?')
   98 RETURN
      END

      SUBROUTINE INFILE(ANSWR)
C
C SUBROUTINE INFILE PROMPTS THE USER FOR THE NAME OF THE INPUT FILE.
C
      CHARACTER*1 ANSWR
      CHARACTER*10 NAME1
      DATA NAME1 /' '/
   50 WRITE(*,90)
      WRITE(*,60)
      WRITE(*,90)
   60 FORMAT(' Please enter the name of the INPUT file. ')
      READ(*,90) NAME1
      OPEN(8,FILE=NAME1,STATUS='OLD',ERR=66)
      RETURN
C
C ALLOW THE USER TO EXIT IF HE OR SHE CANNOT CREATE A FILE
C
   66 WRITE(*,200)
      WRITE(*,90)
      J = J + 1
      IF (J .GE. 4) THEN
         WRITE(*,220)
         WRITE(*,90)
         READ(*,89) ANSWR
      END IF
      IF ((ANSWR .EQ. 'N') .OR. (ANSWR .EQ. 'n')) GO TO 98
      GO TO 50
   89 FORMAT(A1)
   90 FORMAT(A10)
  200 FORMAT(' The file you wish to create cannot be created by DOS.')
  210 FORMAT(' Please try again.')
  220 FORMAT(' Press Y  to continue trying to get a valid name, or press
     * N to exit. (Y/N)?')
   98 RETURN
      END
```

```
      SUBROUTINE GETFL(ANSWR)
C
C SUBROUTINE GETFL ACCESS A PARAMETER FILE SPECIFIED BY THE USER.
C GETFL READ INFORMATION NEEDED IN THE PROGRAM. GETFL ECHO PRINTS
C THE INFORMATION IT READS TO THE SCREEN. THE SUBROUTINE CURSOR IS
C USED IN THIS ROUTINE.
C
C VARIABLES
C   NAME   - THE NAME OF THE DATA FILE ON THE SYSTEM
C   PROBE  - THE PROBE SERIAL NUMBER
C   RESLEN - THE RESISTANCE PER UNIT LENGTH OF THE HEATING ELEMENT
C            (mohm/cm)
C   LENGTH - THE EFFECTIVE LENGTH OF THE HEATING ELEMENT (cm)
C   RADIUS - THE EFFECTIVE RADIUS OF THE HEATING ELEMENT (cm)
C   TC     - THE NUMBER OF THERMOCOUPLES ON THE SURFACE OF THE PROBE
C   START  - THE STARTING TIME OF THERMAL CONDUCTIVITY MEASUREMENT
C            (sec)
C   FINISH - THE TIME AT WHICH POWER TO THE PROBE IS TO BE TURNED
C            OFF (sec)
C   POWER  - THE POWER LEVEL OF THE PROBE (W/cm)
C   ROW    - INDICATES THE ROW WHERE INFORMATION IS TO BE WRITTEN
C   COL    - INDICATES THE COLUMN WHERE INFORMATION IS TO BE WRITTEN
C
      CHARACTER*1 ANSWR
      CHARACTER*10 NAME
      CHARACTER*35 LABEL
      INTEGER*2 ROW,COL
      INTEGER PROBE,DIST,LENGTH,START,FINISH,INC,TC,TIME,RESLEN
      REAL RADIUS,POWER
      COMMON /VARS/ TIME,POWER,RADIUS,PIE,GAMMA,INC,TC,START,FINISH
      COMMON /HEATR/ POWR,LENGTH
      I = 0
   10 WRITE(*,100)
      WRITE(*,89)
      READ(*,88) NAME
C
C CLEAR SCREEN
C
      ROW = 0
      COL = 0
      CALL CURSOR(COL,ROW)
      OPEN(11,FILE=NAME,STATUS='OLD',ERR=99)
C
C READ THE PARAMETER FILE
C
      READ(11,90) LABEL,NAME
      WRITE(*,90) LABEL,NAME
      READ(11,91) LABEL,PROBE
      WRITE(*,91) LABEL,PROBE
      READ(11,93) LABEL,RESLEN
      WRITE(*,93) LABEL,RESLEN
      READ(11,93) LABEL,LENGTH
      WRITE(*,93) LABEL,LENGTH
      READ(11,94) LABEL,RADIUS
      WRITE(*,94) LABEL,RADIUS
      READ(11,91) LABEL,TC
      WRITE(*,91) LABEL,TC
      READ(11,95) LABEL,START
      WRITE(*,95) LABEL,START
      READ(11,95) LABEL,FINISH
      WRITE(*,95) LABEL,FINISH
      READ(11,92) LABEL,INC
      WRITE(*,92) LABEL,INC
      READ(11,94) LABEL,POWER
      WRITE(*,94) LABEL,POWER
      WRITE(*,89)
      TIME = FINISH - START
   88 FORMAT(A10)
   89 FORMAT(A1)
   90 FORMAT(A35,A10)
   91 FORMAT(A35,I2)
   92 FORMAT(A35,I3)
```

```
   93 FORMAT(A35,I4)
   94 FORMAT(A35,F10.5)
   95 FORMAT(A35,I5)
  100 FORMAT(' Please enter the name for the parameter file. ')
      RETURN
   99 WRITE(*,200)
      WRITE(*,90)
      I = I + 1
      IF (I .GE. 4) THEN
         WRITE(*,220)
         WRITE(*,90)
         READ(*,89) ANSWR
      END IF
      IF ((ANSWR .EQ. 'N') .OR. (ANSWR .EQ. 'n')) GO TO 98
  200 FORMAT(' The parameter file you specified cannot be opened in DOS
     *.')
  210 FORMAT(' Please try again.')
  220 FORMAT(' Press Y  to continue trying to get a valid name, or press
     * N to exit (Y/N).')
      GO TO 10
   98 RETURN
      END
      SUBROUTINE INITAL
C
C  INITAL IS SUBROUTINE USED TO INITIALIZE THE SYSTEM (THE OMEGA
C  BOARD) FOR READING TEMPERATURES FROM THERMOCOUPLE OUTPUTS.
C  INITAL USES SUBROUTINES BRDARD, GETBRD, RESET, LOCATE, INIT, AND
C  SETRNG, ALL OF THESE SUBROUTINES CAN BE FOUND IN THE FILE
C  'KEY.ASM'.
C
      INTEGER*2 IRANG,NCHAN,IDATA,IBRDS
      INTEGER*2 JDATA(1512)
      DIMENSION VOLT(16),CJTEMP(2),CCAL(2)
      DIMENSION A(0:7)
      DIMENSION B(0:6)
      COMMON /ANALDATA/ JDATA
      DATA ACAL/17415./
      DATA BCAL/23509./
      DATA DCAL/21933./
      DATA CCAL/14870.,14924./
      DATA A/0.100860910,25.72794369,-.7673458295,7.802559581E-2,
     1-9.247486589E-3,6.97688E-4,-2.66192E-5,3.94078E-7/
      DATA B/0.000579,0.039593,0.000017,-2.833469E-6,
     16.668596E-8,1.32534E-9,-2.98963E-11/
      NCHAN=1
      CALL BRDADR(IDATA,NCHAN)
      CALL GETBRD(IBRDS)
      CALL RESET
      NCHAN=1
      CALL BRDADR(IDATA,NCHAN)
      CALL GETBRD(IBRDS)
      CALL LOCATE
      NCHAN=1
      CALL BRDADR(IDATA,NCHAN)
      CALL GETBRD(IBRDS)
      CALL INIT
C
C SET RANGES TO 3
C
      DO 10 K=1,16
      IRANG=3
      NCHAN=K
      CALL SETRNG(IRANG,NCHAN)
   10 CONTINUE
      CALL INIT
      KR=0
      DO 40 K=1,8
      JJDATA=JDATA(K)
      IF(JDATA(K).LT.0) JJDATA=JJDATA+2**16
      JRANGE=JJDATA-256*(JJDATA/256)
      KR=KR+1
      JRANGE=JJDATA/256
      KR=KR+1
```

```
   40 CONTINUE
      RETURN
      END
      SUBROUTINE PRTCLK(JD)
C
C SUBROUTINE PRTCLK PRINTS THE DATE AND TIME.
C
      INTEGER*2 JD(7),ROW,COL,ROWP,COLP
      CHARACTER*22 IDATA
      CALL RDCUR(COLP,ROWP)
      ROW=1
      COL=59
      WRITE(IDATA,100) JD
  100 FORMAT(I2,'/',I2.2,'/',I2.2,1X,I2,':',I2.2,':',I2.2,'.',I2.2,'$')
      CALL CURSOR(COL,ROW)
      CALL PRT(IDATA)
      CALL CURSOR(COLP,ROWP)
      RETURN
      END
      SUBROUTINE DEGREE(TEMP)
C
C  SUBROUTINE DEGREE IS USED TO READ TEMEPRATURES FROM OMEGA BOARD.
C  DEGREE CALLS SUBROUTINES ANALOG, MEASURE, AND RESET.
C
C VARIABLES
C    TEMP    - TEMP IS THE ARRAY OF TEMPERATURES FORMED AS A
C              RESULT OF THIS ROUTINE.
C    CJTEMP  - CJTEMP HOLDS THE COLD JUNCTION TEMPERATURES FOR EACH
C              OF THE 16 AVAILABLE LINES.
C
      INTEGER*2 IRANG,NCHAN,IDATA,IBRDS
      INTEGER*2 JDATA(1512)
      DIMENSION VOLT(16),CJTEMP(2),CCAL(2)
      DIMENSION A(0:7)
      DIMENSION B(0:6),TEMP(16)
      COMMON /ANALDATA/ JDATA
      CHARACTER*1 NCONT
      DATA ACAL/17415./
      DATA BCAL/23509./
      DATA DCAL/21933./
      DATA CCAL/14870.,14924./
      DATA A/0.100860910,25.72794369,-.7673458295,7.802559581E-2,
     1-9.247486589E-3,6.97688E-4,-2.66192E-5,3.94078E-7/
      DATA B/0.000579,0.039593,0.000017,-2.833469E-6,
     16.668596E-8,1.32534E-9,-2.98963E-11/
   20 CALL MEASURE
C
C READ EACH OF THE 16 CHANNELS
C
      DO 30 K=1,16
      NCHAN=K
      CALL ANALOG(IDATA,NCHAN)
      IDATA=JDATA(9+K)
      VOLT(K)=IDATA
      VOLT(K)=(VOLT(K)/ACAL)*25.0
   30 CONTINUE
      NCHAN=17
      CALL ANALOG(IDATA,NCHAN)
      IDATA=JDATA(26)
C
C CONVERT FROM COLD JUNCTION TEMPERATURE TO MV.
C
      CJTEMP(1)=IDATA
      CJTEMP(1)=(CJTEMP(1)/BCAL)*(29816000./CCAL(1))/2.-273.16
      EMF1=B(0)
      DO 45 K=1,6
      EMF1=EMF1+B(K)*CJTEMP(1)**K
   45 CONTINUE
      NCHAN=18
      CALL ANALOG(IDATA,NCHAN)
      IDATA=JDATA(27)
C
C CONVERT FROM COLD JUNCTION TEMPERATURE TO MV.
```

```
C
      CJTEMP(2)=IDATA
      CJTEMP(2)=(CJTEMP(2)/BCAL)*(29816000./CCAL(2))/2.-273.16
      EMF2=B(0)
      DO 46 K=1,6
      EMF2=EMF2+B(K)*CJTEMP(2)**K
   46 CONTINUE
      DO 47 K=1,16
      VR=EMF1
      IF(K.GE.8) VR=EMF2
      VOLT(K)=VOLT(K)+VR
      TEMP(K)=A(0)
C
C CREATE AN ARRAY OF TEMPERATURES
C
      DO 48 L=1,7
      TEMP(K)=TEMP(K)+A(L)*VOLT(K)**L
   48 CONTINUE
C
C CONVERT SI UNIT INTO ENGINEERING UNIT
C
      TEMP(K)=1.8*TEMP(K)+32.
   47 CONTINUE
      CALL RESET
      RETURN
      END
      SUBROUTINE POWERON(POWER,POWR)
C THIS SUBROUTINE RESETS AND PERFORMS THE OPERATIONS OF THE
C PROGRAMMABLE DC POWER SUPPLY FOR THE PROBE HEATER.
C POWER DENOTES THE REQUESTED POWER. POWR DENOTES THE DELIVERED POWER.
C VOLTS IS MAXIMUM VOLTS REQUESTED.  AMPS IS THE MAXIMUM AMPS
C VOLTO IS THE VOLTS OUT, OR SCALED VOLTS, AND AMPO IS AMPS OUT, OR
C SCALED AMPS.  VOLTI DENOTES VOLTS IN, OR SCALED VOLTS, AMPI IS AMPS
C IN, OR SCALED AMPS.  VOLTB DENOTES DELIVERED VOLTS, AND AMPB IS THE
C DELIVERED AMPS.  OHM DENOTES THE RESISTANCE OF THE LOAD.
C THE SUBROUTINES CALLED ARE RESETP, PORTOT,DIGWR AND POWON.
      REAL POWER, VOLT,POWR
      INTEGER*2 IPORT,JDATA(8),IERROR,ROW,COL
      CHARACTER*1 ANSW
      DATA JDATA / 8*0 /
      CALL RESETP
   10 IPORT = 0
      CALL PORTOT(IPORT,IERROR)
C   JDATA(1) IS REMOTE RESET
      JDATA(1) = 1
C   JDATA(2) IS REMOTE TRIP
      JDATA(2) = 0
C   JDATA(3) IS REMOTE INHIBIT
      JDATA(3) = 1
      CALL DIGWR(IPORT,JDATA,IERROR)
      IF (IERROR .EQ. 1) WRITE (*,170)
      JDATA(1) = 0
      JDATA(2) = 1
      CALL DIGWR(IPORT,JDATA,IERROR)
      IF (IERROR .EQ. 1) THEN
      ROW = 3
      COL = 1
      CALL CURSOR(ROW,COL)
      WRITE(*,170)
  170 FORMAT(' ------ERROR IN I/O PORT WRITE ROUTINE-----  ')
      ENDIF
      JDATA(1) = 1
      COL=1
      ROW=3
      CALL CURSOR(COL,ROW)
      WRITE(*,180)
  180 FORMAT(' STARTING THE HEATING OF THE PROBE ')
      CALL POWON(POWER,POWR)
      ROW=4
      CALL CURSOR(COL,ROW)
      WRITE(*,190) POWER,POWR
  190 FORMAT(' POWER REQUESTED IS ',F10.5,'W',5X,' POWER DELIVERED IS
     * ', F10.5, 'W')
```

```
      WRITE(10, 190) POWER,POWR
      RETURN
      END

      SUBROUTINE POWON(POWER,POWR)
C
C SUBROUTINE POWON TURNS ON THE POWER TO THE PROBE. THEN POWON CHECKS
C TO SEE IF THE POWER LEVEL IS WITHIN + OR -0.08 WATT OF THE
C REQUESTED LEVEL.   THIS ROUTINE CALLS CONSTPOW AND DIGANA.
C
      INTEGER*2 IDATA,IGAIN,ICHAN,IERROR, ID(2),KD(2)
      REAL POWER,POWR,VOLTS,AMPS,VOLTO,AMPO,VOLTI,AMPI
      CHARACTER*1 ANSW
C
C SET THE STARTING VOLTAGE EQUAL TO 1.5 VOLTS, AND THEN CALCULATE
C THE AMPS NEEDED TO GET A STARTING POWER.
C
      ICHAN = 1
      IGAIN=0
      VOLTS = 1.5
      AMPS = POWER /VOLTS
      DATA = VOLTS * 4096 / 60
      IDATA = DATA
      IF (IDATA .GT. 4095) IDATA = 4095
      CALL DIGANA(IDATA,ICHAN,IGAIN,IERROR)
      IF (IERROR .EQ. 1) WRITE(*,100)
  100 FORMAT(' ----- ERROR IN DIGITAL TO ANALOG CONVERSION -----')
      DATA = AMPS * 4096 / 10
      IDATA = DATA
      IF (IDATA .GT. 4095) IDATA = 4095
      ICHAN = 0
      CALL DIGANA(IDATA,ICHAN,IGAIN,IERROR)
      IF (IERROR .EQ. 1) WRITE(*,100)
      CALL SEC(ID)
      CALL SEC(KD)
      KD(1) = KD(1) + 1
      IF (KD(1) .GE. 60) KD(1) = KD(1) - 60
  110 CALL SEC(ID)
      IF (KD(1) .NE. ID(1)) GOTO 110
      IF (KD(2) .LT. ID(2)) GOTO 110
      CALL CONSTPOW(POWER,POWR,VOLTS,AMPS)
      RETURN
      END

      SUBROUTINE RDPOW(AMPB,VOLTB)
C
C SUBROUTINE RDPOW READS THE LEVEL OF THE AMPS AND THE VOLTS. RDPOW
C CALLS ANADIG.
C
      INTEGER ISUM
      INTEGER*2 IDATA,IGAIN,ICHAN,IERROR,K,ID(2),KD(2)
      REAL VOLTB,AMPB,GAIN(4),VOLTI,AMPI
      DATA GAIN /1.0,2.0,4.0,8.0/
C
C READ CURRENT FROM THE POWER SUPPLY
C
      CALL SEC(ID)
      CALL SEC(KD)
      KD(1) = KD(1) + 1
      IF (KD(1) .GE. 60) KD(1) = KD(1) - 60
  100 CALL SEC(ID)
      IF (KD(1) .NE. ID(1)) GOTO 100
      IF (KD(2) .LT. ID(2)) GOTO 100
      IGAIN = 0
      ICHAN = 0
      ISUM = 0
      DO 110 II=1,10
         CALL ANADIG(IDATA,ICHAN,IGAIN,IERROR)
         ISUM = ISUM + IDATA
  110 CONTINUE
  120 IF (IERROR .EQ. 1) THEN
         WRITE(*,130)
  130    FORMAT(' ----- ERROR IN ANALOG TO DIGITAL CONVERSION -----')
```

```
         GO TO 160
      END IF
      IDATA = ISUM/10
      K = IGAIN + 1
      AMPI = (IDATA - 2048) * 10/(2048 * GAIN(K))
      IF (AMPI .LT. 0) AMPI = 0
      AMPB = AMPI * 2
C
C READ VOLTAGE FROM THE POWER SUPPLY
C
      IGAIN = 0
      ICHAN = 1
      ISUM = 0
      DO 140 II=1,10
         CALL ANADIG(IDATA,ICHAN,IGAIN,IERROR)
         ISUM = ISUM + IDATA
  140 CONTINUE
  150 IDATA = ISUM/10
      IF (IERROR .EQ. 1) THEN
         WRITE(*,130)
         GO TO 160
      END IF
      K = IGAIN + 1
      VOLTI = (IDATA - 2048) * 10 / (2048 * GAIN(K))
      IF (VOLTI .LT. 0) VOLTI = 0
      VOLTB = VOLTI * 12
  160 RETURN
      END
      SUBROUTINE CONSTPOW(POWER,POWR,VOLTS,AMPS)
C
C SUBROUTINE CONSTPOW MAINTAINS THE POWER LEVEL TO WITHIN + OR - 0.08
C WATT OF THE DESIRED POWER. CONSTPOW CALLS PORTIN, RDPOW, DIGANA,
C AND DIGRD.
C
      INTEGER*2 IDATA1,IDATA(8),IGAIN,ICHAN,IERROR,IPORT
      REAL POWER,POWR,VOLTS,AMPS,VOLTI,AMPI,VOLTO,AMPO,AMPB,VOLTB
C
      IPORT = 1
      CALL PORTIN(IPORT,IERROR)
      IF (IERROR .EQ. 1) WRITE(*,100)
  100   FORMAT(' ----- ERROR IN SETTING A PORT -----')
C
C FIND THE DIFFERENCE BETWEEN THE DESIRED POWER LEVEL AND THE ACTUAL
C POWER LEVEL.
C
  110 CALL RDPOW(AMPB,VOLTB)
      POWR = AMPB * VOLTB
      IF (AMPB .EQ. 0) THEN
         OHM = 1
      ELSE
         OHM = VOLTB/AMPB
      END IF
      DIFF = POWER - POWR
C
C READ THE CC (CONSTANT VOLTAGE) BIT
C
      IPORT = 1
      CALL DIGRD(IPORT,IDATA,IERROR)
      ICHAN = IDATA(5)
      CV = ((OHM*POWER)**.5)-((OHM*POWR)**.5)
      CI = (POWER/OHM)**.5-(POWR/OHM)**.5
      IF (ABS(DIFF) .GT. 0.08) THEN
        IF (ICHAN .EQ. 0) THEN
           VOLTS = VOLTS + CV
           DATA = VOLTS/60*4096
        ELSE
           AMPS = AMPS + CI
           DATA = AMPS/10*4096
        END IF
C
C CHECK FOR ANY ERROR FLAGS
C
        IF (IDATA(1).EQ. 0) WRITE(*,120)
```

```
120    FORMAT(' WARNING : over temperature ')
       IF (IDATA(2) .EQ. 0) WRITE(*,130)
130    FORMAT(' WARNING : over voltage ')
       IF (IDATA(3) .EQ. 0) WRITE(*,140)
140    FORMAT(' WARNING : output unregulated ')
       IF (IDATA(10) .EQ. 1) WRITE(*,150)
150    FORMAT(' WARNING : low bias or AC drop out ')
C
C
       IF (ICHAN .EQ. 0) THEN
          ICHAN = 1
       ELSE
          ICHAN = 0
       END IF
       IF (DATA .LT. 0) DATA = 0
       IF (DATA .GT. 4095) DATA = 4095
       IDATA1 = DATA
       CALL DIGANA(IDATA1,ICHAN,IGAIN,IERROR)
       IF (IERROR .EQ. 1) WRITE(*,160)
160    FORMAT(' ----- ERROR IN DIGITAL TO ANALOG CONVERSION -----')
       GO TO 110
      END IF
      RETURN
      END
      SUBROUTINE GTEMP
C
C ESTABLIZES AXES FOR PLOT OF TEMP VERSUS LOG OF TIME
C
C
      INTEGER*2 X,Y,LFULL
      CHARACTER*40 IDATA
C
      COMMON /PLTDAT/ YMAX,YMIN,XMAX,XMIN
      DATA LFULL/600/
C
      XSCALE=ALOG10(XMAX/XMIN)
      YSCALE=(YMAX-YMIN)
      CALL DISP(1)
      X=60
      Y=300
      CALL PUTPT(X,Y)
      X=60+LFULL
      CALL DLINE(X,Y)
      X=60
      Y=301
      CALL PUTPT(X,Y)
      X=60+LFULL
      CALL DLINE(X,Y)
      X=60
      Y=300
      CALL PUTPT(X,Y)
      Y=0
      CALL DLINE(X,Y)
      DO 30 K=1,2
      X=60+K
      Y=300
      CALL PUTPT(X,Y)
      Y=0
      CALL DLINE(X,Y)
   30 CONTINUE
      FN=ALOG10(XMAX/XMIN)+0.01
      MLAST=FN
      DO 10 K=1,MLAST
      DO 10 L=1,9
      XSEC=L*10**K
      XX=60+600*ALOG10(XSEC/XMIN)/XSCALE
      X=XX
      Y=300
      CALL PUTPT(X,Y)
      Y=295
      IF(L.EQ.1.OR.L.EQ.5) Y=290
      CALL DLINE(X,Y)
      IF(L.EQ.1.OR.L.EQ.5) THEN
```

```
      JSEC=XSEC
      X=X-8*(4-K)
      NX=1
      NY=2
      Y=306
      NV=0
      WRITE(IDATA,106) JSEC
  106 FORMAT(I4,'$')
      CALL PRTTXT(X,Y,IDATA,NX,NY,NV)
      ENDIF
   10 CONTINUE
      NSCALE=(YMAX-YMIN)/10
      NSCALE=NSCALE-1
      DO 20 K=0,NSCALE
      TEMP=YMIN+K*10
      YY=300-300*(TEMP-YMIN)/YSCALE
      Y=YY
      X=60
      CALL PUTPT(X,Y)
      X=78
      CALL DLINE(X,Y)
      KLABEL=TEMP
      WRITE(IDATA,102) KLABEL
  102 FORMAT(I4,'$')
      NV=0
      NX=1
      NY=1
      X=26
      Y=Y-4
      CALL PRTTXT(X,Y,IDATA,NX,NY,NV)
   20 CONTINUE
      WRITE(IDATA,100)
  100 FORMAT('TIME (seconds) $')
      X=210
      Y=328
      NX=2
      NY=2
      NV=0
      CALL PRTTXT(X,Y,IDATA,NX,NY,NV)
      NV=1
      WRITE(IDATA,101)
  101 FORMAT('  Temperature  $')
      X=0
      Y=20
      CALL PRTTXT(X,Y,IDATA,NX,NY,NV)
      WRITE(IDATA,105)
  105 FORMAT('Frequency$')
      NV=0
      X=200
      Y=0
      WRITE(IDATA,107)
  107 FORMAT('SOIL CONDUCTIVITY TEST $')
      CALL PRTTXT(X,Y,IDATA,NX,NY,NV)
      RETURN
      END
      SUBROUTINE PLTCHR(TEM,ISEC,NCHAR)
C
C SUBROUTINE FOR  PLOTTING A SYMBOL OF TEMPERATURE VERSUS LOG OF TIME
C IN SECONDS
      INTEGER*2 X,Y
C
C
      COMMON /PLTDAT/ YMAX,YMIN,XMAX,XMIN
      CALL LEVEL(1)
      Y=300
      IF(TEM.GT.YMIN) YY=300-((TEM-YMIN)/(YMAX-YMIN))*300
      Y=YY+4
      IF(Y.LT.0) Y=0
      XSEC=ISEC
      XX=60+600*(ALOG10(XSEC/XMIN)/ALOG10(XMAX/XMIN))
      X=XX-4
      IF(X.LT.60) X=60
      IF(X.GT.660) X=660
```

```
      CALL TEXT(X,Y,NCHAR)
      RETURN
      END
      SUBROUTINE SETTEM(TEM,ISEC)
C
C SUBROUTINE FOR PLOTTING TEMPERATURE VERSUS LOG OF TIME
C IN SECONDS
C
C
      INTEGER*2 X,Y
C
      COMMON /PLTDAT/ YMAX,YMIN,XMAX,XMIN
      YSCALE=YMAX-YMIN
      XSCALE=ALOG10(XMAX/XMIN)
      CALL LEVEL(1)
      YY=300
      IF(TEM.GT.YMIN) YY=300-300*(TEM-YMIN)/YSCALE
      IF(YY.LT.0.0) YY=0.
      IF(YY.GT.300.) YY=300.
      Y=YY
      XSEC=ISEC
      XX=60.+600.*(ALOG10(XSEC/XMIN)/XSCALE)
      X=XX
      IF(X.GT.660) X=660
      IF(X.LT.60) X=60
      CALL PUTPT(X,Y)
      RETURN
      END
      SUBROUTINE PLTTEM(TEM,ISEC)
C
C SUBROUTINE FOR PLOTTING TEMPERATURE VERSUS LOG OF TIME
C IN SECONDS
C
C
      INTEGER*2 X,Y
C
      COMMON /PLTDAT/ YMAX,YMIN,XMAX,XMIN
      YSCALE=YMAX-YMIN
      XSCALE=ALOG10(XMAX/XMIN)
      CALL LEVEL(1)
      YY=300
      IF(TEM.GT.YMIN) YY=300-300*(TEM-YMIN)/YSCALE
      IF(YY.LT.0.0) YY=0.
      IF(YY.GT.300.) YY=300.
      Y=YY
      XSEC=ISEC
      XX=60.+600.*(ALOG10(XSEC/XMIN)/XSCALE)
      X=XX
      IF(X.GT.660) X=660
      IF(X.LT.60) X=60
      CALL DLINE(X,Y)
      RETURN
      END
      SUBROUTINE PRTTXT(X,Y,IDATA,NX,NY,NV)
C
C
      INTEGER*2 X,Y,NCHAR,NX,NY,XX,YY
      CHARACTER*(*) IDATA
      CHARACTER*1 JCHAR
      NLEN=LEN(IDATA)
      JSTRG=INDEX(IDATA,'$')-1
      IF(JSTRG.LE.0) JSTRG=NLEN
      DO 10 K=1,JSTRG
      JCHAR=IDATA(K:K)
      NCHAR=ICHAR(JCHAR)
      IF(NV.EQ.0) THEN
      YY=Y
      XX=X+(K-1)*NX*8
      ELSE
      XX=X
      YY=Y+(K-1)*NY*8
      ENDIF
      CALL PRTCHAR(XX,YY,NCHAR,NX,NY)
```

```
      CALL LEVEL(1)
   10 CONTINUE
      RETURN
      END
      SUBROUTINE TEMPER
C THIS SUBPROGRAM READS THE TEMPERATURES OF SOIL AT DIFFERENT
C LOCATIONS AND VARIOUS DEPTHS. UP TO SIXTEEN TEMPERATURE INPUT
C CHANNELS CAN BE HANDLED BY THIS SUBPROGRAM  THE OUTPUT DATA AND
C SUMMARY RESULTS OF TEMPERATURE MEASUREMENTS ARE STORED IN TWO FILES
C NAMED BY THE USER.  THE SUBROUTINES CALLED BY THIS SUBPROGRAM ARE
C MAKEINX, GETINX, DATAFL, RESET, INITAL, DEGREE, SCRLUP, KEYBD,
C CLOCK AND PRTCLK.
C
C VARIABLES :
C  TEMP - CONTAINS DATA OF THE MEASURED TEMPERATURES.
C  NTC - TOTAL NUMBER OF THERMOCOUPLES USED.
C  ILABEL - THE IDENTIFICATION TITLE OF THE MEASURING LOCATION.
C  XH(I,J) - THE HORIZONTAL DISTANCE MEASURED FROM A REFERENCE POINT
C           TO THE I-TH THERMOCOUPLE OF THE J-TH PROBE, (INCH).
C  YV(I,J) - THE VERTICAL DEPTH FROM THE GROUND SURFACE FOR THE I-TH
C           THERMOCOUPLE OF THE J-TH PROBE, (INCH).
C  COL - THE COLUMN NUMBER WHERE INFORMATION IS TO BE WRITTEN TO THE
C        SCREEN.
C  ROW - THE ROW NUMBER WHERE INFORMATION IS TO BE WRITTEN  TO THE
C        SCREEN.
C  NPROB - THE THERMOCOUPLE PROBE NUMBER.
C  TDATA - AN ARRAY OF THE TEMPERATURE DATA.
C  TUERTH - AN ARRAY OF THE UNDISTURBED EARTH TEMPERATURE.
      CHARACTER*40 ILABEL
      CHARACTER*80 IDATA
      CHARACTER*1 ANSW,ANSWR
      CHARACTER*12 DTAFL
      REAL KSAV
      INTEGER*2 ROW,COL,JD(7),RULWI,CULWI,RLRWI,CLRWI
      DIMENSION XH(16,15),YV(16,15),TDATA(16,15),TEMP(16),LDTA(16,15),
     *TUERTH(16,15)
      COMMON / TCLOC/ NTC,NPROB,XH,YV
      COMMON / NDKS/ NDPT,KSAV
      COMMON / LOGA/ LDTA
      LOGICAL LDTA
      DATA RULWI/7/,CULWI/1/,RLRWI/23/,CLRWI/80/
      NPMAX=1
      XHMAX=1.0
      DO 8 I=1,16
        DO 5 J=1,15
          TDATA(I,J)=0.
          XH(I,J)=0.
          YV(I,J)=0.
          TUERTH(I,J)=0.
          LDTA(I,J)=.FALSE.
    5   CONTINUE
    8 CONTINUE
C
C CLEAR SCREEN
C
      COL=0
      ROW=0
      CALL CURSOR(COL,ROW)
      ROW=5
      COL=1
      CALL CURSOR(COL,ROW)
      WRITE(*,10)
   10 FORMAT(15X,'SOIL TEMPERATURE DATA ACQUISITION PROGRAM')
C
C DETERMINE IF AN EXISTING INDEX FILE IS TO BE USED OR A NEW INDEX
C FILE SHOULD BE CREATED
C
   20 ROW=8
      COL=1
      CALL CURSOR(COL,ROW)
      WRITE(*,100)
      READ(*,110) ANSW
      IF ((ANSW .EQ. 'Y') .OR. (ANSW .EQ. 'y')) THEN
```

```
         CALL GETINX(ANSWR)
      ELSE
        IF ((ANSW .EQ  'N') .OR. (ANSW  EQ. 'n')) THEN
           CALL MAKEINX(ANSWR)
        ELSE
           WRITE(*,120)
           GO TO 20
        END IF
      END IF
      IF ((ANSWR .EQ. 'N') .OR. (ANSWR .EQ. 'n')) GO TO 999
      COL=1
      ROW=24
      CALL CURSOR(COL,ROW)
      WRITE(*,88)
   88 FORMAT(20x,' Please press RETURN to continue.')
      READ(*,210)
      CALL NAME(600,'NEW',14,ANSWR,DTAFL)
      IF ((ANSWR .EQ. 'N') .OR. (ANSWR .EQ. 'n')) GO TO 999
C
C INITIALIZE THE SYSTEM FOR READING TEMPERATURES FROM THERMOCOUPLE
C OUTPUTS
C
      CALL INITAL
C
C INPUT THE IDENTIFICATION TITLE OF THE MEASURING LOCATION AND WRITE
C THE DATA TO THE SCREEN
C
      COL=0
      ROW=0
      CALL CURSOR(COL,ROW)
      COL=1
      ROW=2
      CALL CURSOR(COL,ROW)
      WRITE(*,225)
      READ(*,210) ILABEL
      WRITE(14,225) ILABEL
      COL=5
      ROW=ROW+1
      CALL CURSOR(COL,ROW)
      WRITE(IDATA,300)
      CALL PRT(IDATA)
      ROW=ROW+1
      WRITE(IDATA,310)
      CALL CURSOR(COL,ROW)
      CALL PRT(IDATA)
      ROW=ROW+1
      WRITE(IDATA,320)
      CALL CURSOR(COL,ROW)
      CALL PRT(IDATA)
      ROW=ROW+1
      WRITE(IDATA,330)
      CALL CURSOR(COL,ROW)
      CALL PRT(IDATA)
      WRITE(14,340)
  340 FORMAT(/,'  THERMO-   TEMPER-    HORIZONTAL    VERTICAL ',
     * '       DATE      TIME ')
      WRITE(14,341)
  341 FORMAT('  COUPLE     ATURE     DISTANCE      DEPTH ')
      WRITE(14,342)
  342 FORMAT('    NO.     (DEG F)     (INCH)       (INCH) ',
     *'                 HR:MIN ')
C
C READ CLOCK AND WRITE TIME AND DATE TO SCREEN
C
  350 CALL CLOCK(JD)
      CALL PRTCLK(JD)
C
C READ THE GROUND TEMPERATURES OF VARIOUS DEPTHS AND WRITE THE DATA
C TO THE SCREEN AND THE OUTPUT FILE
C
      CALL DEGREE(TEMP)
      DO 50 I=1,16
         TDATA(I,NPROB)=TEMP(I)
```

```
   50 CONTINUE
      DO 550 I=1,16
      IF (LDTA(I,NPROB)) THEN
        ROW=ROW+1
        IF (ROW .GT. 23) THEN
          ROW=23
          CALL SCRLUP(RULWI,CULWI,RLRWI,CLRWI)
        END IF
        COL=1
        CALL CURSOR(COL,ROW)
        WRITE(IDATA,500) I-4,TDATA(I,NPROB),XH(I,NPROB),YV(I,NPROB)
        CALL PRT(IDATA)
      END IF
  500 FORMAT(7X,I2,3(5X,F8.3),'$')
  550 CONTINUE
      RTEMPO=ROW
       DO 580 I=1,16
       IF (LDTA(I,NPROB)) WRITE(14,570) I-4,TDATA(I,NPROB),XH(I,NPROB),
     * YV(I,NPROB),(JD(K),K=1,5)
  580    CONTINUE
C
C DETERMINE IF THE TEMPERATURE DATA ARE TO BE UPDATED
C
      NCHAR=0
  585 ROW=23
      COL=1
      CALL CURSOR(COL,ROW)
      WRITE(*,600)
      ROW=24
      COL=56
      CALL CURSOR(COL,ROW)
      READ(*,110) ANSW
      IF ((ANSW .EQ. 'Y') .OR. (ANSW .EQ. 'y')) THEN
         ROW=RTEMPO
         GO TO 350
      ELSE
         IF((ANSW .EQ. 'N') .OR. (ANSW .EQ. 'n')) THEN
            CONTINUE
         ELSE
            GOTO 585
         END IF
      END IF
C
C FIND THE MAXIMUM VALUE FOR PROBE NUMBER
C
      IF (NPROB .GT. NPMAX) NPMAX=NPROB
C
C DETERMINE IF MORE TEMPERATURE DATA WITH ANOTHER PROBE ARE NEEDED
C
 586  ROW=23
      COL=1
      CALL CURSOR(COL,ROW)
      WRITE(*,650)
      ROW=24
      COL=62
      CALL CURSOR(COL,ROW)
      READ(*,110) ANSW
      IF ((ANSW .EQ. 'Y') .OR. (ANSW .EQ. 'y')) THEN
         CLOSE(12,STATUS='KEEP')
         COL=0
         ROW=0
         CALL CURSOR(COL,ROW)
         GO TO 20
      ELSE
         IF ((ANSW .EQ. 'N') .OR. (ANSW .EQ. 'n')) THEN
            CONTINUE
         ELSE
            GOTO 586
         END IF
      END IF
C
C OBTAIN THE UNDISTURBED EARTH TEMPERATURES AT VARIOUS DEPTHS. THE
C PROBE WITH THE FURTHEST HORIZONTAL DISTANCE IS THE PROBE USED FOR
```

```
C THE UNDISTURBED EARTH TEMPERATURES.
C
      DO 610 J=1,NPMAX
      DO 610 I=1,16
        IF(XH(I,J) .GT. XHMAX) XHMAX=XH(I,J)
  610 CONTINUE
      DO 622 J=1,NPMAX
        DO 620 I=1,16
          IF(XH(I,J) .EQ. XHMAX) THEN
             LDTA(I,J)=.FALSE.
             JSTAR=J
             TDATA(I,JSTAR)=TDATA(I,J)
             YV(I,JSTAR)=YV(I,J)
          END IF
  620   CONTINUE
  622 CONTINUE
      DO 640 J=1,NPMAX
       DO 630 I=1,16
        IF(LDTA(I,J)) THEN
          DO 625 K=1,16
            IF(YV(I,J) .EQ. YV(K,JSTAR)) TUERTH(I,J)=TDATA(K,JSTAR)
  625     CONTINUE
        END IF
  630  CONTINUE
  640 CONTINUE
C
C CREATE ANOTHER OUTPUT FILE AND SUMMARIZE THE TEMPERATURE DATA
C
      CALL NAME(170,'NEW',15,ANSWR,DTAFL)
      WRITE(15,700)
      WRITE(15,710)
      NDPT=0
      DO 900 J=1,NPMAX
       DO 800 I=1,16
         IF(LDTA(I,J)) THEN
           WRITE(15,750) TDATA(I,J),XH(I,J),YV(I,J),TUERTH(I,J)
           NDPT=NDPT+1
         END IF
  800  CONTINUE
  900 CONTINUE
  100 FORMAT(' Would you like to use an existing index file ? (Y/N)',
     *' : ')
  110 FORMAT(A1)
  120 FORMAT('Please try again .')
  210 FORMAT(A40)
  225 FORMAT(1H+,'  MEASURING LOCATION : ',A40)
  300 FORMAT('  THERMO-    TEMPER-    HORIZONTAL    VERTICAL $')
  310 FORMAT('  COUPLE     ATURE       DISTANCE      DEPTH $')
  320 FORMAT('    NO.      (DEG F)      (INCH)        (INCH) $')
  330 FORMAT(' ________________________________________________ $')
  570 FORMAT(4X,I2,3(5X,F8.3),3X,I2,2('/',I2),2X,I2,':',I2)
  600 FORMAT(' Please enter Y (or N) to have (or skip) another ',
     * 'scan :    ')
  650 FORMAT(' Would you like to get more data with another probe ?',
     *' (Y/N) :    ')
  700 FORMAT('  SUMMARY RESULTS OF TEMPERATURE MEASUREMENTS ',//,
     *'    TEMPER-    HORIZONTAL    VERTICAL     UNDISTURBED ')
  710 FORMAT('     ATURE       DISTANCE       DEPTH        TEMPERATURE',/,
     *'    (DEG F)      (INCH)         (INCH)        (DEG F) ',/)
  750 FORMAT(2X,F8.3,3(5X,F8.3))
      CALL RESET
      REWIND 15
      RETURN
  999 WRITE(*,1000)
 1000 FORMAT(' SOME ERRORS OCCUR IN DATA INPUT. ')
      CALL RESET
      RETURN
      END
      SUBROUTINE MAKEINX(ANSWR)
C  THIS SUBROUTINE HELPS THE USER TO CREATE AN INDEX FILE FOR
C  EXECUTING TEMPERATURE DATA ACQUISITION PROGRAM.  THE SUBROUTINES
C  CALLED ARE CURSOR AND PRT.
C
```

```
C   VARIABLES
C    FLNAME - THE NAME OF THE THERMOCOUPLE INDEX FILE TO BE CREATED.
C    NTC - THE TOTAL NUMBER OF THERMOCOUPLES USED.
C    NOTC - THE THERMOCOUPLE NUMBER.
C    XH(I,J) - THE HORIZONTAL DISTANCE OF THE I-TH THERMOCOUPLE OF THE
C              J-TH PROBE FROM A REFERENCE POINT, (INCH).
C    YV(I,J) - THE VERTICAL DEPTH OF THE I-TH THERMOCOUPLE OF THE J-TH
C              PROBE FROM THE GROUND SURFACE, (INCH).
C    COL - THE COLUMN NUMBER WHERE INFORMATION IS TO BE WRITTEN TO
C          SCREEN.
C    ROW - THE ROW NUMBER WHERE INFORMATION IS TO BE WRITTEN TO
C          SCREEN.
C    NPROB - THE THERMOCOUPLE PROBE NUMBER.
C
      CHARACTER*12  FLNAME
      CHARACTER*1  ANSWR
      CHARACTER*80  IDATA
      INTEGER*2  ROW,COL
      DIMENSION XH(16,15),YV(16,15),LDTA(16,15)
      COMMON / TCLOC/ NTC,NPROB,XH,YV
      COMMON / LOGA/ LDTA
      LOGICAL LDTA
C
C   PROVIDE THE INFORMATION FOR CREATING AN INDEX FILE
C
      CALL NAME(70,'NEW',12,ANSWR,FLNAME)
  100 FORMAT(' FILE NAME : ', 23X, A12)
      IF ((ANSWR .EQ. 'N') .OR. (ANSWR .EQ. 'n')) GO TO 1000
      WRITE(12, 100) FLNAME
  105 WRITE(*,110)
      READ(*,112) NTC
      IF ((NTC .LT. 1) .OR. (NTC .GT. 16)) GO TO 105
      WRITE(12,110) NTC
      WRITE(*,115)
      READ(*,112) NPROB
      WRITE(12,115) NPROB
      DO 150  J=1,NTC
        WRITE(*,120)
        READ(*,112) NOTC
        NOTC=NOTC+4
        WRITE(*,130)
        READ(*,113) XH(NOTC,NPROB)
        WRITE(*,140)
        READ(*,113) YV(NOTC,NPROB)
        LDTA(NOTC,NPROB)=.TRUE.
  150 CONTINUE
C
C   CLEAR SCREEN AND ARRANGE INFORMATION INTO A TABLE FORM
C
      COL=0
      ROW=0
      CALL CURSOR(COL,ROW)
      COL=1
      ROW=3
      CALL CURSOR(COL,ROW)
      WRITE(IDATA,160)
      CALL PRT(IDATA)
      WRITE(12,161)
      ROW=ROW+2
      CALL CURSOR(COL,ROW)
      WRITE(IDATA,170)
      CALL PRT(IDATA)
      ROW=ROW+1
      WRITE(IDATA,180)
      CALL CURSOR(COL,ROW)
      CALL PRT(IDATA)
      ROW=ROW+1
      WRITE(IDATA,190)
      CALL CURSOR(COL,ROW)
      CALL PRT(IDATA)
      ROW=ROW+1
      WRITE(IDATA,200)
      CALL CURSOR(COL,ROW)
```

```
      CALL PRT(IDATA)
      WRITE(12,205)
      WRITE(12,210)
      DO 350 I=1,16
         IF (LDTA(I,NPROB)) THEN
         ROW=ROW+1
         COL=1
         CALL CURSOR(COL,ROW)
         WRITE(IDATA,300) I-4,XH(I,NPROB),YV(I,NPROB)
         CALL PRT(IDATA)
         WRITE(12,310) I-4,XH(I,NPROB),YV(I,NPROB)
         END IF
  350 CONTINUE
  110 FORMAT(' NUMBER OF THERMOCOUPLES (XX) : ',4X,I2)
  112 FORMAT(I2)
  113 FORMAT(F8.3)
  115 FORMAT(' PROBE NUMBER (XX) : ',15X,I2)
  120 FORMAT(' THERMOCOUPLE NUMBER (XX) : ',8X,I2)
  130 FORMAT(' HORIZONTAL DISTANCE (INCH) :',1X,F8.3)
  140 FORMAT(' VERTICAL DEPTH (INCH) : ',5X,F8.3)
  160 FORMAT(' A LIST OF THERMOCOUPLE ARRANGEMENTS  $ ')
  161 FORMAT('  LOCATIONS OF THERMOCOUPLES IN THE GROUND ')
  170 FORMAT('  THERMO-   HORIZONTAL    VERTICAL  $ ')
  180 FORMAT('  COUPLE     DISTANCE      DEPTH  $ ')
  190 FORMAT('    NO.        (INCH)       (INCH)  $ ')
  200 FORMAT(' ________________________________________  $ ')
  205 FORMAT('  THERMO-   HORIZONTAL    VERTICAL  ',
     * / '  COUPLE     DISTANCE      DEPTH ')
  210 FORMAT('    NO.        (INCH)       (INCH)  ',
     * / ' ________________________________________  ')
  300 FORMAT(4X,I2,7X,F8.3,6X,F8.3,'$')
  310 FORMAT(4X,I2,7X,F8.3,6X,F8.3)
 1000 RETURN
      END

      SUBROUTINE GETINX(ANSWR)
C
C  SUBROUTINE GETINX ACCESS A THERMOCOUPLE INDEX FILE SPECIFIED BY
C  THE USER.  THIS SUBROUTINE READS THE FILE AND PRINTS THE
C  INFORMATION TO THE SCREEN.   SUBROUTINE CURSOR IS CALLED BY THIS
C  SUBPROGRAM.
C
C  VARIABLES :
C   FLNAME - THE NAME OF THE THERMOCOUPLE INDEX FILE.
C   NTC - TOTAL NUMBER OF THERMOCOUPLES USED.
C   XH(I,J) - THE HORIZONTAL DISTANCE MEASURED FROM A REFERENCE POINT
C             TO THE I-TH THERMOCOUPLE OF THE J-TH PROBE, (INCH).
C   YV(I,J) - THE VERTICAL DEPTH FROM THE GROUND SURFACE TO THE I-TH
C             THERMOCOUPLE OF THE J-TH PROBE, (INCH).
C   COL - THE COLUMN NUMBER WHERE INFORMATION IS TO BE WRITTEN TO THE
C         SCREEN.
C   ROW - THE ROW NUMBER WHERE INFORMATION IS TO BE WRITTEN TO THE
C         SCREEN.
C   NPROB - THE THERMOCOUPLE PROBE NUMBER.
C
      CHARACTER*1 ANSWR
      CHARACTER*12 FLNAME
      CHARACTER LABEL*36,LABELS(1:5)*43
      DIMENSION XH(16,15),YV(16,15),LDTA(16,15)
      COMMON / TCLOC/ NTC,NPROB,XH,YV
      COMMON / LOGA/ LDTA
      LOGICAL LDTA
      CALL NAME(900,'OLD',12,ANSWR,FLNAME)
C
C  READ THE THERMOCOUPLE INDEX FILE
C
      READ(12,155) LABEL,FLNAME
  155 FORMAT(A36,A10)
      WRITE(*,155) LABEL,FLNAME
      READ(12,160) LABEL,NTC
      WRITE(*,160) LABEL,NTC
  160 FORMAT(A36,I2)
      READ(12,200) LABEL,NPROB
```

```
      WRITE(*,200) LABEL,NPROB
  200 FORMAT(A36,I2)
      READ(12,220) (LABELS(J),J=1,5)
      WRITE(*,220) (LABELS(J),J=1,5)
  220 FORMAT(4(A43,/),A43)
      DO 300 K=1,NTC
         READ(12,250) I,XH(I,NPROB),YV(I,NPROB)
         WRITE(*,250) I,XH(I,NPROB),YV(I,NPROB)
         I=I+4
         LDTA(I,NPROB)=.TRUE.
  250 FORMAT(4X,I2,7X,F8.3,6X,F8.3)
  300 CONTINUE
 1000 RETURN
      END
       SUBROUTINE HLCALC
C THIS SUBROUTINE CALCULATES THE HEAT LOSS FROM DIRECTLY BURIED PIPES
C BASED ON UNCONSTRAINED, UNWEIGHTED NONLINEAR LEAST SQUARES FITTING
C OF THE EARTH TEMPERATURE DATA TO THE THEORETICAL EQUATIONS USING THE
C LEVENBERG/MARQUARDT/MORRISON ALGORITHM WITH ANALYTICAL DERIVATIVES.
C THESE DIMENSIONS ALLOW UP TO 100 OBSERVED VALUES, 5 INDEPENDENT
C VARIABLES, AND 10 PARAMETERS TO BE DETERMINED.
C SUBROUTINES CALLED INCLUDE CURSOR, LMMNL AND FUNVAL FOR TWO PIPES
C IN SEPARATE CONDUITS, OR FNVAL1 FOR TWO PIPES INSTALLED IN A SINGLE
C CONDUIT HAVING A CONSTANT TEMPERATURE.
C THE INPUT AND OUTPUT DATA ARE STORED IN FILES NAMED BY THE USER.
C
      IMPLICIT REAL*8 (A-F,S-Y)
      CHARACTER*1 ANSW,ANSWR
      CHARACTER*12 DTAFL
      INTEGER NCOND
      INTEGER*2 ROW,COL
      DIMENSION X(10),YY(100),XX(100,5),F(100),A(100,10)
      COMMON /CALHL/ YY,XX
      COMMON /UHIN/ ND,AK,DS,NCOND
C**
      IER=2
    . ITS=50
      TOL=1.D-6
      EPS=1.D-8
      EXPEND=1.5
      DECR=0.5
C
C  CLEAR SCREEN
C
      ROW=0
      COL=0
      CALL CURSOR(COL,ROW)
      ROW=5
      COL=1
      CALL CURSOR(COL,ROW)
      WRITE(*,19)
   19 FORMAT(18X,' BURIED PIPES HEATLOSS CALCULATION PROGRAM')
C
C  DETERMINE IF AN EXISTING DATA FILE IS TO BE USED OR A NEW DATA
C  FILE SHOULD BE CREATED
C
   20 ROW=8
      COL=1
      CALL CURSOR(COL,ROW)
      WRITE(*,30)
   30 FORMAT(' Would you like to use an existing data file ? (Y/N)',
     &' :        ')
      COL=56
      ROW=9
      CALL CURSOR(COL,ROW)
      READ(*,40) ANSW
   40 FORMAT(A1)
      IF((ANSW .EQ. 'Y') .OR. (ANSW .EQ. 'y')) THEN
         CALL GETDTA(X,ANSWR)
      ELSE
         IF((ANSW .EQ. 'N') .OR. (ANSW .EQ. 'n')) THEN
            CALL MAKEDTA(X,ANSWR)
         ELSE
```

```
              GO TO 20
           END IF
        END IF
        IF((ANSWR .EQ. 'N') .OR. (ANSWR .EQ. 'n')) GO TO 999
        CALL NAME(600,'NEW',16,ANSWR,DTAFL)
        IF((ANSWR .EQ. 'N') .OR. (ANSWR .EQ. 'n')) GO TO 999
C       COMPUTE RHO=1/(4*PAI*K),AND THE DISTANCE BETWEEN THE PIPE
C       CENTERS
C
        DO 60 I=1,ND
        XX(I,4)=1./(4.*3.14159*AK)
        XX(I,5)=DS
60      CONTINUE
        IF (NCOND .EQ. 1) THEN
         NIV=4
         NP=3
        ELSE
         NIV=5
         NP=5
        END IF
C SHOULD A DIAGNOSITC FILE BE CREATED?
        CALL NAME(100,'NEW',3,ANSW,DTAFL)
        IF ((ANSW .EQ. 'Y') .OR. (ANSW .EQ. 'y')) THEN
           IER=1
        ELSE
           IF ((ANSW .EQ. 'N') .OR. (ANSW .EQ. 'n')) IER=0
        END IF
        ROW=10
        COL=1
        CALL CURSOR(COL,ROW)
        WRITE(*,77)
   77   FORMAT(20X,' *** CALCULATING *** ')
        CALL LMMNL(X,F,A,SUMSQ,ND,NP,TOL,EXPEND,DECR,ITS,IER,NCOND)
        IF (IER .EQ. 2) WRITE(16,80)
80      FORMAT (1X,' MAXIMUM NUMBERS OF ITERATIONS EXCEEDED ')
C       PRINT THE HEAT LOSS RATES FROM THE UNDERGROUND PIPES AND THEIR
C       LOCATIONS
        COL=0
        ROW=0
        CALL CURSOR(COL,ROW)
        IF(NCOND .EQ. 2) THEN
           WRITE(16,90)
           DHL2=X(2)+DS
           WRITE(16,95) X(1),X(4),X(2),DHL2,X(3),X(5)
           WRITE(*,90)
           WRITE(*,95) X(1),X(4),X(2),DHL2,X(3),X(5)
        ELSE
           WRITE(16,91)
           WRITE(16,96) X(1),X(2),X(3)
           WRITE(*,91)
           WRITE(*,96) X(1),X(2),X(3)
        END IF
90      FORMAT(//36X,' PIPE NO. 1 ',6X,' PIPE NO. 2'/)
91      FORMAT(//36X,' PIPES 1 & 2'/)
95      FORMAT(2X,'HEAT LOSS RATE(Q),BTU/H-FT',2(8X,F10.4)/2X,'HORIZONTAL
     &DISTANCE(L),INCH',7X,F10.4,8X,F10.4/2X,'VERTICAL DEPTH(D),INCH',
     &12X,F10.4,8X,F10.4)
96      FORMAT(2X,'HEAT LOSS RATE(Q),BTU/H-FT',8X,F10.4/2X,'HORIZONTAL '
     &,'DISTANCE(L),INCH',7X,F10.4/2X,'VERTICAL DEPTH(D),INCH',
     &12X,F10.4)
        GO TO 101
  999 WRITE(*,1000)
 1000 FORMAT(' SOME ERRORS OCCUR IN DATA INPUT. ')
  101 COL=1
      ROW=23
      CALL CURSOR(COL,ROW)
      WRITE(*,97)
   97 FORMAT(' Press RETURN to get back to the main menu. ')
      READ(*,98)
   98 FORMAT(A1)
      RETURN
      END
      SUBROUTINE MAKEDTA(X,ANSWR)
```

```
C THIS SUBROUTINE ASSISTS THE USER TO CREATE AN INPUT FILE FOR
C CALCULATING THE PIPE HEAT LOSSES AND LOCATIONS FOR DIRECT BURIED
C CONDUIT DISTRIBUTION SYSTEMS.   THE SUBROUTINES CALLED ARE CURSOR
C AND PRT.
C VARIABLES :
C  DTAFL - THE NAME OF THE DATA FILE TO BE CREATED.
C  ND - THE NUMBER OF MEASURING LOCATIONS.
C  AK - THE AVERAGE VALUE OF SOIL THERMAL CONDUCTIVITY, (BTU/H-FT-F).
C  DS - SEPARATION DISTANCE BETWEEN THE CENTERS OF THE PIPES, (INCH).
C  X(I) - THE INITIAL ESTIMATE OF THE I-TH PARAMETER, WHICH INCLUDES :
C             I = 1  HEAT LOSS FROM PIPE NO. 1
C               = 2  HORIZONTAL DISTANCE OF PIPE NO. 1
C               = 3  VERTICAL DEPTH OF PIPE NO. 1
C               = 4  HEAT LOSS FROM PIPE NO. 2
C               = 5  VERTICAL DEPTH OF PIPE NO. 2
C  XX(I,J) - THE INDEPENDENT VARIABLES OF THE I-TH MEASURING LOCATION,
C             J = 1  HORIZONTAL DISTANCE, (INCH).
C               = 2  VERTICAL DEPTH, (INCH).
C               = 3  UNDISTURBED EARTH TEMPERATURE, (DEG F).
C  YY(I) - THE EARTH TEMPERATURE OF THE I-TH MEASURING LOCATION.
C  ROW - THE ROW NUMBER WHERE INFORMATION IS TO BE WRITTEN TO SCREEN.
C  COL - THE COLUMN NUMBER WHERE INFORMATION IS TO BE WRITTEN TO THE
C        SCREEN.
C
      IMPLICIT REAL*8 (A-G,R-Y)
      CHARACTER*12 DTAFL,OUTFL
      CHARACTER*1 ANSWR
      INTEGER*2 COL,ROW
      INTEGER PROMPT,UNITNUM
      CHARACTER*3 STAT
      REAL KSAV
      DIMENSION X(10),YY(100),XX(100,5)
      COMMON /CALHL/ YY,XX
      COMMON /UHIN/ ND,AK,DS,NCOND
      COMMON /NDKS/ NDPT,KSAV
C
C  PROVIDE THE INFORMATION FOR CREATING AN INPUT DATA FILE
C
      CALL NAME(70,'NEW',8,ANSWR,DTAFL)
      IF ((ANSWR .EQ. 'N') .OR. (ANSWR .EQ. 'n')) GO TO 1000
C
C  CLEAR SCREEN
C
      COL=0
      ROW=0
      CALL CURSOR(COL,ROW)
      WRITE(8,50) DTAFL
      WRITE(*,120)
      READ(*,*) DS
      WRITE(8,120) DS
      WRITE(*,70)
      READ(*,30) ND
      WRITE(8,70) ND
      WRITE(*,100)
      READ(*,*) AK
      WRITE(8,100) AK
      WRITE(*,122)
      WRITE(*,125)
      READ(*,30) NMODE
      IF(NMODE .EQ. 2) THEN
C
C USE AN INTERACTIVE MODE FOR DATA INPUT
C
        DO 200 J=1,ND
          WRITE(*,140)
          READ(*,*) NOLN
          WRITE(*,150)
          READ(*,*) YY(NOLN)
          WRITE(*,160)
          READ(*,*) XX(NOLN,1)
          WRITE(*,170)
          READ(*,*) XX(NOLN,2)
          WRITE(*,180)
```

```
          READ(*,*) XX(NOLN,3)
  200   CONTINUE
      ELSE
C
C OBTAIN THE DATA DIRECTLY FROM OTHER SUBPROGRAMS
C
      CALL NAME(200,'OLD',15,ANSWR,DTAFL)
      IF ((ANSWR .EQ. 'N') .OR. (ANSWR .EQ. 'n')) GO TO 1000
  212 READ(15,202)
  202 FORMAT(5(/))
      DO 205 J=1,ND
         READ(15,203) YY(J),(XX(J,K),K=1,3)
  203 FORMAT(2X,F8.3,3(5X,F8.3))
  205 CONTINUE
      END IF
      DO 220 J=1,ND
        WRITE(8,210) YY(J),(XX(J,K),K=1,3)
  220 CONTINUE
      WRITE(*,250)
      WRITE(*,260)
      READ(*,30) NCOND
      WRITE(8,260) NCOND
      WRITE(*,280)
      WRITE(*,300)
      READ(*,*) X(1)
      WRITE(8,300) X(1)
      WRITE(*,320)
      READ(*,*) X(2)
      WRITE(8,320) X(2)
      WRITE(*,340)
      READ(*,*) X(3)
      WRITE(8,340) X(3)
      IF(NCOND .EQ. 2) THEN
        WRITE(*,350)
        READ(*,*) X(4)
        WRITE(8,350) X(4)
        WRITE(*,360)
        READ(*,*) X(5)
        WRITE(8,360) X(5)
      END IF
      RETURN
   30 FORMAT(I3)
   40 FORMAT(A12)
   50 FORMAT('  INPUT DATA FILE NAME = ',21X,A12)
   70 FORMAT('  NUMBER OF MEASURING LOCATIONS (XXX): ',7X,I3)
  100 FORMAT('  SOIL THERMAL CONDUCTIVITY (Btu/h-ft-F) = ',3X,F9.4)
  120 FORMAT('  DISTANCE BETWEEN CENTERS OF PIPES (inch) = ',1X,F9.4)
  122 FORMAT('  PROVIDE THE MODE OF INPUT OF TEST RESULTS : ',/,
     &' 1 = DATA OBTAINED DIRECTLY FROM OTHER SUBPROGRAMS AND FILES ',
     &/,' 2 = DATA INPUT THROUGH AN INTERACTIVE MANNER ')
  125 FORMAT('  MODE OF DATA INPUT ( 1 OR 2 ) = ',12X,I3)
  140 FORMAT('  MEASURING LOCATION NUMBER (XXX): ',11X,I3)
  150 FORMAT('  THE EARTH TEMPERATURE (DEG F) = ',12X,F8.3)
  160 FORMAT('  HORIZONTAL DISTANCE (inch) = ',15X,F8.3)
  170 FORMAT('  VERTICAL DEPTH (inch) = ',20X,F8.3)
  180 FORMAT('  UNDISTURBED EARTH TEMPERATURE (DEG F) = ',4X,F8.3)
  210 FORMAT(1X,F8.3,2(2X,F8.3),2X,F8.3)
  250 FORMAT(' PROVIDE THE TYPE OF PIPE CONFIGURATION : ',/,
     &' 1 = TWO PIPES LOCATED INSIDE A SINGLE METALLIC CONDUIT ',/,
     &' 2 = TWO PIPES INSTALLED IN SEPARATE CONDUIT ')
  260 FORMAT('  TYPE OF PIPE CONFIGURATION (1 OR 2) = ',6X,I3)
  280 FORMAT('  INPUT THE INITIAL PARAMETER ESTIMATES : ')
  300 FORMAT('  HEAT LOSS FROM PIPE NO. 1 (Btu/h-ft) = ',5X,F10.4)
  320 FORMAT('  HORIZONTAL DISTANCE OF PIPE NO. 1 (inch) = ',1X,F10.4)
  340 FORMAT('  VERTICAL DEPTH OF PIPE NO. 1 (inch) = ',6X,F10.4)
  350 FORMAT('  HEAT LOSS FROM PIPE NO. 2 (Btu/h-ft) = ',5X,F10.4)
  360 FORMAT('  VERTICAL DEPTH OF PIPE NO. 2 (inch) = ',6X,F10.4)
 1000 RETURN
      END

      SUBROUTINE GETDTA(X,ANSWR)
C THIS SUBROUTINE READS THE DATA FILE REQUIRED AS THE INPUT FOR
C CALCULATING THE HEAT LOSS FROM THE UNDERGROUND PIPES.   GETDTA
```

```
C ECHOES THE INFORMATION IT READS TO THE SCREEN.   THE SUBROUTINE
C CURSOR IS CALLED IN THIS ROUTINE.
C VARIABLES :
C  DTAFL - THE NAME OF THE INPUT DATA FILE.
C  ND - TOTAL NUMBER OF MEASURING LOCATIONS.
C  AK - THE AVERAGE SOIL THERMAL CONDUCTIVITY, (Btu/h-ft-deg F).
C  DS - SEPARATION DISTANCE BETWEEN THE CENTERS OF THE PIPES, (inch).
C  X(I) - THE INITIAL ESTIMATE OF THE I-TH PARAMETER, WHICH INCLUDES :
C            I = 1   HEAT LOSS FROM PIPE NO. 1.
C              = 2   HORIZONTAL DISTANCE OF PIPE NO. 1.
C              = 3   VERTICAL DEPTH OF PIPE NO. 1.
C              = 4   HEAT LOSS FROM PIPE NO. 2.
C              = 5   VERTICAL DEPTH OF PIPE NO. 2.
C  XX(I,J) - THE INDEPENDENT VARIABLES OF THE I-TH MEASURING LOCATION,
C            J = 1   HORIZONTAL DISTANCE, (inch).
C              = 2   VERTICAL DEPTH, (inch).
C              = 3   UNDISTURBED EARTH TEMPERATURE, (deg F).
C  YY(I) - THE EARTH TEMPERATURE OF THE I-TH MEASURING LOCATION.
C  COL - THE COLUMN NUMBER AT WHICH INFORMATION IS TO BE WRITTEN TO
C        SCREEN.
C  ROW - THE ROW NUMBER AT WHICH INFORMATION IS TO BE WRITTEN TO THE
C        SCREEN.
C
      IMPLICIT REAL*8 (A-G,R-Y)
      CHARACTER*1 ANSWR
      CHARACTER  DTAFL*12, KLABEL*45
      INTEGER*2 ROW,COL
      DIMENSION X(10),YY(100),XX(100,5)
      COMMON /CALHL/ YY,XX
      COMMON /UHIN/ ND,AK,DS,NCOND
      CALL NAME(211,'OLD',8,EXIT,DTAFL)
      IF ((ANSWR .EQ.'N') .OR. (ANSWR .EQ. 'n')) GO TO 1000
C
C  READ AND ECHO THE EXISTING DATA FILE
C
      READ(8,60) KLABEL,DTAFL
      WRITE(*,60) KLABEL,DTAFL
      READ(8,80) KLABEL,DS
      WRITE(*,80) KLABEL,DS
      READ(8,70) KLABEL,ND
      WRITE(*,70) KLABEL,ND
      READ(8,80) KLABEL,AK
      WRITE(*,80) KLABEL,AK
      DO 120 J=1,ND
        READ(8,100) YY(J),(XX(J,K),K=1,3)
        WRITE(*,100) YY(J),(XX(J,K),K=1,3)
  120 CONTINUE
      READ(8,70) KLABEL,NCOND
      WRITE(*,70) KLABEL,NCOND
      READ(8,150) KLABEL,X(1)
      WRITE(*,150) KLABEL,X(1)
      READ(8,150) KLABEL,X(2)
      WRITE(*,150) KLABEL,X(2)
      READ(8,150) KLABEL,X(3)
      WRITE(*,150) KLABEL,X(3)
      IF(NCOND .EQ. 2) THEN
         READ(8,150) KLABEL,X(4)
         WRITE(*,150) KLABEL,X(4)
         READ(8,150) KLABEL,X(5)
         WRITE(*,150) KLABEL,X(5)
      END IF
C
C  PAUSE TO LET THE USER VIEW THE DATA
C
      COL=1
      ROW=24
      CALL CURSOR(COL,ROW)
      WRITE(*,98)
   98 FORMAT(24x,' Please press RETURN to continue.')
      READ(*,50)
      RETURN
   50 FORMAT(A12)
   60 FORMAT(A46,A12)
```

```
   70 FORMAT(A46,I3)
   80 FORMAT(A46,F9.4)
  100 FORMAT(1X,F8.3,2(2X,F8.3),2X,F8.3)
  150 FORMAT(A46,F10.4)
 1000 RETURN
      END

      SUBROUTINE NAME(PROMPT,STAT,UNITNUM,ANSWR,FILEN)
C SUBROUTINE NAME IS A TEMPLATE FOR GETTING THE NAME OF AN INPUT OR
C AN OUTPUT FILE AND OPENING THAT FILE.
C    PROMPT  - THE MESSAGE TO PROMPT  THE USER
C    STAT    - THE STATUS OF THE FILE TO BE OPENED
C    UNITNUM - THE UNIT NUMBER TO BE ASSOCIATED WITH FILE
C    ANSWR   - HAS THE FILE BEEN OPENED SUCCESSFULLY (Y/N)
C    FILEN   - THE NAME OF THE FILE OPENED
      INTEGER PROMPT,UNITNUM
      CHARACTER*3 STAT
      CHARACTER*1 ANSWR
      CHARACTER*12 FILEN
      INTEGER*2 COL,ROW
      I=1
C
C CLEAR SCREEN
C
      COL=0
      ROW=0
      CALL CURSOR(COL,ROW)
      COL=1
      ROW=5
      CALL CURSOR(COL,ROW)
      IF (PROMPT .EQ. 70) THEN
         WRITE(*,70)
      ELSE
          IF (PROMPT .EQ. 200) THEN
             WRITE(*,200)
          ELSE
             IF (PROMPT .EQ. 211) THEN
                WRITE(*,211)
             ELSE
                IF (PROMPT .EQ. 100) THEN
                   WRITE(*,100)
                   I=4
                ELSE
                   IF (PROMPT .EQ. 600) THEN
                      WRITE(*,600)
                   ELSE
                      IF (PROMPT .EQ. 900) THEN
                         WRITE(*,900)
                      ELSE
                         IF (PROMPT .EQ. 170) THEN
                            WRITE(*,170)
                         ELSE
                            GO TO 20
                         END IF
                      END IF
                   END IF
                END IF
             END IF
          END IF
      END IF
   10 COL=5
      ROW=7
      CALL CURSOR(COL,ROW)
      WRITE(*,40)
      COL=8
      ROW=8
      CALL CURSOR(COL,ROW)
      READ(*,50) FILEN
      OPEN(UNITNUM,FILE=FILEN,STATUS=STAT,ERR=99)
      GOTO 20
   99 COL=1
      ROW=10
      CALL CURSOR(COL,ROW)
```

```
      WRITE(*,999)
      I=I+1
      CALL CURSOR(COL,ROW)
   30 IF (I .GT. 4) THEN
         COL=1
         ROW=12
         CALL CURSOR(COL,ROW)
         WRITE(*,420)
         COL=59
         ROW=13
         CALL CURSOR(COL,ROW)
         READ(*,60) ANSWR
         IF ((ANSWR .EQ. 'Y') .OR. (ANSWR .EQ. 'y')) THEN
            GOTO 10
         ELSE
            IF ((ANSWR .EQ. 'N') .OR. (ANSWR .EQ. 'n')) THEN
               GO TO 20
            ELSE
               GOTO 30
            END IF
         END IF
      ELSE
         GO TO 10
      END IF
C
C CLEAR  SCREEN
C
   20 COL=0
      ROW=0
      CALL CURSOR(COL,ROW)
      RETURN
   40 FORMAT(' NAME :                          ')
   50 FORMAT(A12)
   60 FORMAT(A1)
   70 FORMAT(' Please enter a name for the INPUT file being',
     * ' created. ')
  100 FORMAT(' Please enter diagnostic file name or return',
     *' if one is not wanted.')
  170 FORMAT(' The summary being created can be used as an input  ',
     *'file in menu choice three. ')
  200 FORMAT(' Please enter input data file name (Menu choice #2',
     *' last file name entered). ')
  211 FORMAT(' Please enter the INPUT file name.')
  420 FORMAT(' Would you like to keep trying to get a valid name ',
     *'(Y/N)?      ')
  600 FORMAT(' Please enter the OUTPUT file name.')
  900 FORMAT(' Please enter the name of the thermocouple index file.',
     */)
  999 FORMAT(' The file you wish to use cannot be opened by DOS. ')
      END
      SUBROUTINE LMMNL(X,F,A,SUMSQ,ND,NP,TOL,EXPND,DECR,ITS,IER,NCOND)
      IMPLICIT REAL*8 (A-H,O-Z)
      INTEGER NCOND
      CHARACTER*1 ANSW
      REAL*8 B(10,10),DA(10),DU(10),D(10),C(10),DX(10),Y(10)
      DIMENSION X(10),YY(100),XX(100,5),F(100),A(100,10)
      COMMON /CALHL/ YY,XX
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C     THIS SUBROUTINE IS BASED ON LEVENBURG, MARQUARDT, MORRISON
C     ALGORITHM (SEE OSBORNE 'NONLINEAR LEAST SQUARES - THE LEVENBERG
C     ALGORITHM REVISITED', J. AUSTRAL. MATH. SOC. 19 (SERIES B) (1976),
C     PP. 343-357) AND IS MODIFIED FOR ONE OR MORE INDEPENDENT VARIABLES
C     IN THE NONLINEAR FUNCTION.
C
C  VARIABLES:
C     X(1)       Vector of parameters less than or equal to 10
C                Input   : Contains estimate of solution
C                Output  : Contains solution vector
C     A(N,NP)    Matrix containg the first partial derivatives of the function
C                with respect to each of the parameters.
C                Output  : Contains Upper Triangular Factor in orthogonal
C                factorization of GRAD F
C     F(1)       Storage for F vector of terms in sum of squares
```

```
C       SUMSQ       Output   : Contains final residual sum of squares
C       ND          Input    : Dimension of F
C       NP          Input    : Dimension of X
C       TOL         Input    : Tolerence on Calculation
C       EXPND       Input    : Factor by which EPS increased if test on sum of
C                   squares fails
C       DECR        Input    : Factor by which EPS decreased if test on sum of
C                   squares succeeds on first attempt
C       NCOND       Input    :  =1 Two pipes encosed in a single conduit
C                               =2 Two pipes installed in separate conduit
C       ITS         Input    : Max number of iterations
C                   Output   : Actual number of iterations
C       IER         Input    : =0 No Printing
C                              =1 Print Diagnostic Information
C                   Output   : =1 Successful Termination
C                              =2 Max ITS Exceeded
C                              =3 EPS exceeds 1.D6
C                              =4 Attainable Accuracy Reached Tol too small
C                   If IER =2,3 or 4 there may be errors in gradient
C                   calculation
C                          =500+I I'th column of A has a scale which is
C                          small compared to Euclidean norm of A by a
C                          Factor less than 1.D6
C       User supplied subroutine FUNVAL required to set values of SUMSQ,
C       F, A.  Declaration must be
C                           SUBROUTINE FUNVAL (A,F,X,SUMSQ,IFL,N)
C                           If IFL=1  sets all values
C                           If IFL=2  sets SUMSQ only; must not alter A or F
C       Diagonostic information contains in an output file: DIAGON.DTA
C
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
        NRDF=ND-NP
        IPRINT=IER
        IF (IPRINT.EQ.0) GO TO 41
        WRITE(3,102)
41      MAXITS=ITS
        WRITE(16,200)
        ITS0=0
C       CALL SUBROUTINE FOR CALCULATING PARTIAL DERIVATIVES ACCORDING TO
C       A SINGLE OR SEPARATE CONDUIT
        IF(NCOND .EQ. 1) CALL FNVAL1(A,F,X,SSF,1,ND)
        IF(NCOND .EQ. 2) CALL FUNVAL(A,F,X,SSF,1,ND)
        SDRES=DSQRT(SSF/NRDF)
        WRITE(16,201)ITS0,SDRES,X(1)
        DO 210 I=2,NP
        WRITE(16,202) X(I)
210     CONTINUE
        ITS=0
40      ITS=ITS+1
        NITS=0
C       CALL FUNCTION SUBROUTINE ACCORDING TO A SINGLE OR SEPARATE CONDUIT
        IF(NCOND .EQ. 1) CALL FNVAL1(A,F,X,SSF,1,ND)
        IF(NCOND .EQ. 2) CALL FUNVAL(A,F,X,SSF,1,ND)
C       COMPUTE ESTIMATE OF RESIDUAL STANDARD DEVIATION
CCCCCCCCCCCCCCCCCCCCCCCC
C       SCALE GRAD F     C
CCCCCCCCCCCCCCCCCCCCCCCC
        W=0.D0
        DO 1 I=1,NP
        S=0.D0
        DO 2 J=1,ND
2       S=S+A(J,I)**2
        W=W+S
        D(I)=DSQRT(S)
        W=DSQRT(W)
        DO 46 I=1,NP
        IF (D(I)/W.LT.1.D-6) GO TO 47
        S=1.0/D(I)
        DO 3 J=1,ND
3       A(J,I)=A(J,I)*S
46      CONTINUE
        GO TO 48
```

```
47    IER=500+I
      IF (IPRINT.EQ.0) GO TO 49
      WRITE(3,104) I
      WRITE(3,105) (D(I),I=1,NP)
49    GO TO 45
48    IF (ITS.EQ.1) EPS=1.0
      IF (IPRINT.EQ.0) GO TO 42
      WRITE(3,100) ITS,EPS,SSF
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C      HOUSEHOLDER TRANSFORMATION OF GRAD F,F     C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C     VECTOR DA CONTAINS DIAGONAL ELEMENTS OF UPPER
C     TRIANGULAR MATRIX A.
42    DO 4 I=1,NP
      S=0.D0
      DO 5 J=I,ND
5     S=S+A(J,I)**2
      S=DSQRT(S)
      IF (A(I,I).GT.0.0) S=-S
      DA(I)=S
      A(I,I)=A(I,I)-S
      IF (I.EQ.NP) GO TO 6
      IP1=I+1
      DO 7 K=IP1,NP
      S=0.D0
      DO 8 J=I,ND
8     S=S+A(J,I)*A(J,K)
      S=-S/(DA(I)*A(I,I))
      DO 9 J=I,ND
9     A(J,K)=A(J,K)-S*A(J,I)
7     CONTINUE
6     S=0.D0
      DO 20 J=I,ND
20    S=S+A(J,I)*F(J)
      S=-S/(DA(I)*A(I,I))
      DO 21 J=I,ND
21    F(J)=F(J)-S*A(J,I)
4     CONTINUE
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C     COMPUTE SUM OF SQUARES OF RESIDUALS     C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      NP1=NP+1
      SSR=0.D0
      DO 22 I=NP1,ND
22    SSR=SSR+F(I)**2
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C      FACTOR EPS APENDAGE,TRANSFORM RHS UPPER TRIANGLE OF       C
C      TRANSFORMED MATRIX STORED IN UPPER TRIANGLE OF B.         C
C     FILL IN B STORED COLUMNWISE IN ROWS IN LOWER TRIANGLE OF B.C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
19    DO 30 I=1,NP
      DO 31 J=1,NP
31    B(I,J)=0.D0
      C(I)=0.D0
30    B(I,I)=EPS
      DO 10 I=1,NP
      S=DA(I)**2
      IP1=I+1
      IL1=I-1
      DO 12 J=1,I
12    S=S+B(I,J)**2
      S=DSQRT(S)
      IF (DA(I).GT.0.D0) S=-S
      DU(I)=S
      W=DA(I)-S
      IF (I.EQ.NP) GO TO 18
      DO 13 K=IP1,NP
      S=A(I,K)*W
      IF (I.EQ.1) GO TO 11
      DO 14 J=1,IL1
14    S=S+B(I,J)*B(K,J)
11    S=-S/(DU(I)*W)
      B(I,K)=A(I,K)-S*W
```

```
      DO 15 J=1,I
15    B(K,J)=B(K,J)-S*B(I,J)
13    CONTINUE
18    S=F(I)*W
      DO 16 J=1,I
16    S=S+B(I,J)*C(J)
      S=-S/(DU(I)*W)
      DX(I)=F(I)-S*W
      DO 17 J=1,I
17    C(J)=C(J)-S*B(I,J)
10    CONTINUE
CCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C     BACK SUBSTITUTION     C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      DX(NP)=DX(NP)/DU(NP)
      DO 25 I=2,NP
      K=NP-I+1
      KP1=K+1
      S=0.D0
      DO 26 J=KP1,NP
26    S=S+B(K,J)*DX(J)
25    DX(K)=(DX(K)-S)/DU(K)
      SSS=SSR
      DO 32 I=1,NP
      SSS=SSS+C(I)**2
      DX(I)=DX(I)/D(I)
32    Y(I)=X(I)-DX(I)
      NITS=NITS+1
CCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C     CHECK CONVERGENCE     C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      IER=4
      IF (SSS.GE.SSF) GO TO 45
      IER=1
C     CALL THE FUNCTION SUBROUTINE ACCORDING TO A SINGLE OR SEPARATE CONDUIT
      IF(NCOND .EQ. 1) CALL FNVAL1(A,F,Y,SSN,2,ND)
      IF(NCOND .EQ. 2) CALL FUNVAL(A,F,Y,SSN,2,ND)
      S=.5D0*(SSF-SSN)/(SSF-SSS)
      IF (IPRINT.EQ.0) GO TO 43
43    IF (S.GE.1.D-4) GO TO 28
      EPS=EXPND*EPS
      IER=3
      IF (EPS.GT.1.D6) GO TO 45
      GO TO 19
28    SDRES=DSQRT(SSN/NRDF)
      DO 29 I=1,NP
29    X(I)=Y(I)
      IF (IPRINT.EQ.0) GO TO 44
      WRITE(16,203) ITS,EPS,S,SDRES,X(1)
      DO 211 I=2,NP
      WRITE(16,202) X(I)
211   CONTINUE
C     CHECK FOR CONVERGENCE OF SUM OF SQUARES OF RESIDUALS.
44    IF ((DSQRT(SSF)-DSQRT(SSS))/(1.D0 + DSQRT(SSF)).GE.TOL) GO TO 35
45    SUMSQ=SSN
      DO 33 I=1,NP
      A(I,I)=DA(I)
      S=D(I)
      DO 34 J=1,I
34    A(J,I)=A(J,I)*S
33    CONTINUE
C     PRINT ESTIMATES OF PARAMETERS AND THEIR STANDARD DEVIATIONS
      WRITE(16,204)
      DO 270 K=1,NP
      S1=0.D0
      D(K)=1/A(K,K)
      S1=S1+D(K)**2
      KP1=K+1
      DO 260 I=KP1,NP
      S2=0.D0
      IL1=I-1
      DO 250 J=K,IL1
250   S2=S2+A(J,I)*D(J)
```

```
      D(I)=-S2/A(I,I)
      S1=S1+D(I)**2
260   CONTINUE
      S1=SDRES*DSQRT(S1)
      WRITE(16,265) K,X(K),S1
270   CONTINUE
C     PRINT RESIDUAL STANDARD DEVIATION AND DEGREES OF FREEDOM
      WRITE(16,266) SDRES,NRDF
C     PRINT OVSERVATIONS,PREDICTED VALUES AND RESIDUALS
      WRITE(16,275)
C     CALL THE FUNCTION SUBROUTINE ACCORDING TO A SINGLE OR SEPARATE CONDUIT
      IF(NCOND .EQ. 1) CALL FNVAL1(A,F,X,SUMSQ,1,ND)
      IF(NCOND .EQ. 2) CALL FUNVAL(A,F,X,SUMSQ,1,ND)
      DO 280 I=1,ND
      PRED=YY(I)-F(I)
      WRITE(16,276) I,XX(I,1),XX(I,2),YY(I),PRED,F(I)
280   CONTINUE
      RETURN
35    IER=2
      IF (ITS.GE.MAXITS) GO TO 45
      IF (NITS.EQ.1) EPS=EPS*DECR
      GO TO 40
100   FORMAT ('    ITS=',I3,'  EPS=',F14.6,'   SUMSQ=',F14.6)
102   FORMAT ('1   NONLINEAR LEAST SQUARES FIT BY LEVENBERG ALGORITHM')
104   FORMAT ('SCALING ERROR NO. OF COLUMN =',I3)
105   FORMAT (4(' D(',I2,')=',F14.6))
190   FORMAT(1X,'DATA SET NO.',I3/)
200   FORMAT (2X,'ITERATION',27X,'RESIDUAL',5X,'PARAMETER ESTIMATES'/3X,
     &'NUMBER',7X,'EPS',9X,'PSI',7X,'STD DEV',9X,'X(1) TO X(5)'/)
201   FORMAT(4X,I3,29X,F10.5,4X,F15.6)
202   FORMAT(50X,F15.6)
203   FORMAT(/4X,I3,5X,F10.5,2X,F10.5,2X,F10.5,4X,F15.6)
204   FORMAT(//57X,'STANDARD'/10X,'PARAMETERS',13X,'ESTIMATE',16X,
     &'DEVIATION'/)
265   FORMAT(12X,'X(',I2,')',12X,F14.8,12X,F12.8/)
266   FORMAT(10X,'RESIDUAL STANDARD DEVIATION = ',F14.8//10X,'NUMBER OF
     &RESIDUAL DEGREES OF FREEDOM = ',I8//)
275   FORMAT(10X,'HORIZONTAL',3X,'VERTICAL',5X,'OBSERVED',5X,'PREDICTED
     &'/11X,'DISTANCE',5X,'DEPTH',9X,'TEMP',9X,'TEMP',8X,'RESIDUAL'/2X,
     & 'NUMBER',4X,'(IN.)',7X,'(IN.)',7X,'(DEG F)',6X,'(DEG F)',7X,
     &'(DEG F)'//)
276   FORMAT(3X,I3,2(4X,F8.3),3(4X,F10.5))
      RETURN
      END
      SUBROUTINE FUNVAL (A,F,X,SUMSQ,IFL,ND)
C     THIS SUBROUTINE IS USED WITH SUBROUTINE LMMNLF TO EVALUATE THE
C     FUNCTION G AND ITS DERIVATIVES.
      IMPLICIT REAL*8 (A-G,R-Y)
      DIMENSION X(10),YY(100),XX(100,5),F(100),A(100,10)
      REAL*8 NUM1,NUM2,NUM3,NUM4,NUM5,NUM6
      COMMON /CALHL/ YY,XX
      SUMSQ=0.D0
      DO 10 I=1,ND
      NUM1=(XX(I,1)-X(2))**2+(XX(I,2)+X(3))**2
      DEN1=(XX(I,1)-X(2))**2+(XX(I,2)-X(3))**2
      NUM2=(XX(I,1)-X(2)-XX(I,5))**2+(XX(I,2)+X(5))**2
      DEN2=(XX(I,1)-X(2)-XX(I,5))**2+(XX(I,2)-X(5))**2
      NUM3=XX(I,2)*(XX(I,1)-X(2))
      DEN3=NUM1*DEN1
      NUM4=XX(I,2)*(XX(I,1)-X(2)-XX(I,5))
      DEN4=NUM2*DEN2
      NUM5=XX(I,2)*((XX(I,1)-X(2))**2+(XX(I,2)**2-X(3)**2))
      NUM6=XX(I,2)*((XX(I,1)-X(2)-XX(I,5))**2+(XX(I,2)**2-X(5)**2))
C     CALCULATE THE VALUE OF FUNCTION G
      G=XX(I,4)*(X(1)*DLOG(NUM1/DEN1)+X(4)*DLOG(NUM2/DEN2))+XX(I,3)
      RESID=YY(I)-G
      SUMSQ=SUMSQ+RESID*RESID
      IF (IFL .EQ. 2) GOTO 10
C     SET VALUES FOR I-TH ROW OF GRADIENT G
      A(I,1)=-XX(I,4)*DLOG(NUM1/DEN1)
      A(I,2)=-8.*XX(I,4)*(X(1)*X(3)*NUM3/DEN3+X(4)*X(5)*NUM4/DEN4)
      A(I,3)=-4.*XX(I,4)*X(1)*NUM5/DEN3
      A(I,4)=-XX(I,4)*DLOG(NUM2/DEN2)
```

```
      A(I,5)=-4.*XX(I,4)*X(4)*NUM6/DEN4
      F(I)=RESID
10    CONTINUE
      RETURN
      END
      SUBROUTINE FNVAL1(A,F,X,SUMSQ,IFL,ND)
C     THIS SUBROUTINE IS USED WITH SUBROUTINE LMMNLF TO EVALUATE THE
C     FUNCTION G AND ITS PARTIAL DERIVATIVES WITH RESPECT TO THE
C     PARAMETERS TO BE DETERMINED. THE TOTAL HEAT LOSS FROM TWO PIPES
C     INSTALLED IN A METALLIC CONDUIT IS DETERMINED USING THIS
C     SUBROUTINE
      IMPLICIT REAL*8 (A-G,R-Y)
      DIMENSION X(10),YY(100),XX(100,5),F(100),A(100,10)
      REAL*8 NUM1,NUM3,NUM5,NUM6
      COMMON /CALHL/ YY,XX
      SUMSQ=0.D0
      DO 10 I=1,ND
      NUM1=(XX(I,1)-X(2))**2+(XX(I,2)+X(3))**2
      DEN1=(XX(I,1)-X(2))**2+(XX(I,2)-X(3))**2
      NUM3=XX(I,2)*(XX(I,1)-X(2))
      DEN3=NUM1*DEN1
      NUM5=XX(I,2)*((XX(I,1)-X(2))**2+(XX(I,2)**2-X(3)**2))
      NUM6=XX(I,2)*((XX(I,1)-X(2)-XX(I,5))**2+(XX(I,2)**2-X(5)**2))
C     CALCULATE THE VALUE OF FUNCTION G
      G=XX(I,4)*X(1)*DLOG(NUM1/DEN1)+XX(I,3)
      RESID=YY(I)-G
      SUMSQ=SUMSQ+RESID*RESID
      IF (IFL .EQ. 2) GO TO 10
C     SET VALUES FOR I-TH ROW OF GRADIENT G
      A(I,1)=-XX(I,4)*DLOG(NUM1/DEN1)
      A(I,2)=-8.*XX(I,4)*X(1)*X(3)*NUM3/DEN3
      A(I,3)=-4.*XX(I,4)*X(1)*NUM5/DEN3
      F(I)=RESID
10    CONTINUE
      RETURN
      END
;
; Subroutine CLOCK
;
; FOR MULTI I/O PLUS CARD
;
;       Use as Fortran callable subroutine
;
;                       CALL CLOCK(JD)
;
;                               where JD is declared as INTEGER*2 JD(7)
;
;
PARMBLK         STRUC
PARM1           DD      ?
PARMBLK         ENDS
P@CLOCK         segment Para    'Code'
;
;
CLOCK   Proc    Far

PORT_CLK        EQU     340H

Public  CLOCK
        Assume  Cs:P@CLOCK
        Lds     Si,Es:PARM1[Bx]
MONTH:  MOV     Dx,PORT_CLK+7
        IN      Al,Dx
        CALL    BCDBIN
        MOV     Ah,00H
        MOV     [Si],Ax
        Inc     Si
        Inc     Si
DAY:    MOV     Dx,PORT_CLK+6
        IN      Al,Dx
        CALL    BCDBIN
        MOV     Ah,00H
        MOV     [Si],Ax
```

```
        Inc     Si
        Inc     Si
YEAR:   MOV     Dx,PORT_CLK+9
        IN      Al,Dx
        CALL    BCDBIN
        MOV     Ah,00H
        MOV     [Si],Ax
        Inc     Si
        Inc     Si
HOUR:   MOV     Dx,PORT_CLK+4
        IN      Al,Dx
        CALL    BCDBIN
        MOV     Ah,00H
        MOV     [Si],Ax
        Inc     Si
        Inc     Si
MIN:    MOV     Dx,PORT_CLK+3
        IN      Al,Dx
        CALL    BCDBIN
        MOV     Ah,00H
        MOV     [Si],Ax
        Inc     Si
        Inc     Si
SECOND: MOV     Dx,PORT_CLK+2
        IN      Al,Dx
        CALL    BCDBIN
        MOV     Ah,00H
        MOV     [Si],Ax
        Inc     Si
        Inc     Si
HUN:    MOV     Dx,PORT_CLK+1
        IN      Al,Dx
        CALL    BCDBIN
        MOV     Ah,00H
        MOV     [Si],Ax
        Ret
CLOCK   Endp
BCDBIN  Proc    Near
        MOV     Ah,Al
        And     Al,0FH
        And     Ah,0F0H
        Shr     Ah,1
        Add     Al,Ah
        Shr     Ah,1
        Shr     Ah,1
        Add     Al,Ah
        Ret
BCDBIN  Endp
;
; SUBROUTINE SEC
;
;               Fortran callable subroutine
;
;               Use as CALL SEC(ID)
;
;                       With argument declared as INTEGER*2 ID(2)
;
SEC     Proc    Far

Public  SEC

        Assume  Cs:P●CLOCK
        Lds     Si,Es:PARM1[BX]
        MOV     Dx,PORT_CLK+2
        IN      Al,Dx
        CALL    BCDBIN
        MOV     Ah,00H
        MOV     [Si],Ax
        Inc     Si
        Inc     Si
        MOV     Dx,PORT_CLK+1
        IN      Al,Dx
        CALL    BCDBIN
```

```
        MOV     Ah,00H
        MOV     [Si],Ax
        Ret
SEC     Endp
P0CLOCK Ends
        End


PARMBLK                 STRUC
PARM1           DD       ?
PARM2           DD       ?
PARM3           DD       ?
PARM4           DD       ?
PARMBLK                 ENDS
P0POWER                 SEGMENT PARA    'CODE'
                        ASSUME CS:P0POWER


;**********************************************

SUBTTL SUBROUTINE RESETP

RESETP PROC FAR

PUBLIC RESETP

;
;THE RESET WILL TAKE PLACE REGARDLESS OF WHAT OPERATING SEQUENCE THE DT2801
;SERIES BOARD MAY BE EXECUTING PRIOR TO RUNNING THIS PROGRAM.
;
;
;  TO CALL FROM FORTRAN USE :  CALL RESETP
;
BASE_ADDRESS     EQU     2ECH
COMMAND_REGISTER EQU     BASE_ADDRESS + 1
STATUS_REGISTER EQU      BASE_ADDRESS + 1
DATA_REGISTER    EQU     BASE_ADDRESS
COMMAND_WAIT     EQU     4H
WRITE_WAIT       EQU     2H
READ_WAIT        EQU     5H
;
CRESET           EQU     0H
CSTOP            EQU     0FH
CCLEAR           EQU     1H
CADIN            EQU     0CH
CDAOUT           EQU     8H
CSIN             EQU     4H
CSOUT            EQU     5H
CDIOIN           EQU     6H
CDIOOUT          EQU     7H
;                 .
;
ERR              EQU     1H                 ; 1 = CODE FOR ERROR
NOERR            EQU     0H                 ; 0 = CODE FOR NO ERROR
ERRCK            EQU     80H                ; 80H = ERROR CHECK PATTERN

;
;STOP THE DT2801 SERIES BOARD AND EMPTY THE DATA OUT REGISTER.
;
        MOV     DX,COMMAND_REGISTER
        MOV     AL,CSTOP
        OUT     DX,AL
        MOV     DX,DATA_REGISTER
        IN      AL,DX
;
;WAIT UNTIL THE DT2801 SERIES BOARD DATA IN FLAG IS CLEAR AND READY FLAG
;IS SET, THEN WRITE THE RESET COMMAND BYTE TO THE COMMAND REGISTER.
;
        MOV     DX,STATUS_REGISTER
WRWAIT: IN      AL,DX
        AND     AL,WRITE_WAIT
        JNZ     WRWAIT
OKAY:   IN      AL,DX
```

```
        AND     AL,COMMAND_WAIT
        JZ      OKAY
        MOV     DX,COMMAND_REGISTER
        MOV     AL,CRESET
        OUT     DX,AL
;
;WAIT UNTIL THE DT2801 SERIES BOARD DATA OUT READY OR (READY) FLAG IS SET,
;THEN READ THE DATA OUT REGISTER TO EMPTY IT.
;
        MOV     DX,STATUS_REGISTER
PUT:    IN      AL,DX
        AND     AL,READ_WAIT
        JZ      PUT
        MOV     DX,DATA_REGISTER
        IN      AL,DX
        RET
RESETP  ENDP

;***********************************************

SUBTTL ANADIG

ANADIG PROC FAR

PUBLIC ANADIG

;
;ANADIG REQUESTS AN A/D INPUT GAIN CODE VALUE, AND A LEAGAL CHANNEL NUMBER.
;THE GAIN CODE MUST BE 0,1,2 OR 3.
;
; TO CALL FROM FORTRAN USE :  CALL ANADIG(IDATA,NCHAN,IGAIN,IERROR)
;    INTEGER*2 IDATA,NCHAN,IGAIN,IERROR
;


;
;STOP AND CLEAR THE DT2801 SERIES BOARD   .
;
        MOV     DX,COMMAND_REGISTER
        MOV     AL,CSTOP
        OUT     DX,AL
        MOV     DX,DATA_REGISTER
        IN      AL,DX

        MOV     DX,STATUS_REGISTER
WAIT3:  IN      AL,DX
        AND     AL,WRITE_WAIT
        JNZ     WAIT3
OKAY3:  IN      AL,DX
        AND     AL,COMMAND_WAIT
        JZ      OKAY3
        MOV     DX,COMMAND_REGISTER
        MOV     AL,CCLEAR
        OUT     DX,AL
;
;WAIT UNTIL DATA IN FLAG IS CLEAR AND READY FLAG IS SET, THEN WRITE THE
;READ A/D IMMEDIATE COMMAND BYTE TO THE DATA IN REGISTER.
;
        MOV     DX,STATUS_REGISTER
WAIT4:  IN      AL,DX
        AND     AL,WRITE_WAIT
        JNZ     WAIT4
OKAY4:  IN      AL,DX
        AND     AL,COMMAND_WAIT
        JZ      OKAY4
        MOV     DX,COMMAND_REGISTER
        MOV     AL,CADIN
        OUT     DX,AL
;
;WAIT UNTIL THE DT2801 SERIES BOARD DATA IN FULL FLAG IS CLEAR, THEN WRITE
;THE A/D GAIN BYTE TO THE DATA IN REGISTER.
;
        MOV     DX,STATUS_REGISTER
WAIT5:  IN      AL,DX
```

```
        AND     AL,WRITE_WAIT
        JNZ     WAIT5
        MOV     DX,DATA_REGISTER
        LDS     SI,ES:PARM3[BX]
        MOV     AL,[SI]
        OUT     DX,AL
;
;WAIT UNTIL DATA IN FLAG IS CLEAR, THEN WRITE THE A/D CHANNEL BYTE TO THE
;DATA IN REGISTER.
;
        MOV     DX,STATUS_REGISTER
WAIT6:  IN      AL,DX
        AND     AL,WRITE_WAIT
        JNZ     WAIT6
        MOV     DX,DATA_REGISTER
        LDS     SI,ES:PARM2[BX]
        MOV     AL,[SI]
        OUT     DX,AL
;
;READ TWO BYTES OF A/D DATA FROM DATA OUT REGISTER, WAITING FOR A SET DATA
;OUT READY (OR READY) FLAG BEFORE EACH READ, AND COMBINE THE TWO BYTES
;INTO ONE WORD.
;
        MOV     DX,STATUS_REGISTER
WAIT7:  IN      AL,DX
        AND     AL,READ_WAIT
        JZ      WAIT7
        MOV     DX,DATA_REGISTER
        IN      AL,DX
        MOV     CL,AL
        MOV     DX,STATUS_REGISTER
WAIT8:  IN      AL,DX
        AND     AL,READ_WAIT
        JZ      WAIT8
        MOV     DX,DATA_REGISTER
        IN      AL,DX
        MOV     AH,AL
        MOV     AL,CL
        LDS     SI,ES:PARM1[BX]
        MOV     [SI],AX
;
;WAIT UNTIL THE DT2801 SERIES BOARD DATA IN FULL FLAG IS CLEAR AND READY
;FLAG IS SET, INDICATING COMMAND COMPLETION, THEN CHECK THE STATUS REGISTER
;ERROR FLAG.
;
        MOV     DX,STATUS_REGISTER
WAIT9:  IN      AL,DX
        AND     AL,WRITE_WAIT
        JNZ     WAIT9
OKAY9:  IN      AL,DX
        AND     AL,COMMAND_WAIT
        JZ      OKAY9
        LDS     SI,ES:PARM4[BX]
        MOV     CL,NOERR
        MOV     [SI],CL
        IN      AL,DX
        AND     AL,ERRCK
        JZ      EXIT
;
;       ERROR HANDELING ROUTINE
;
        MOV     CL,ERR
        MOV     [SI],CL
EXIT:   RET
ANADIG  ENDP

;*****************************************************

SUBTTL DIGANA

DIGANA PROC FAR

PUBLIC DIGANA
```

```
;
;DIGANA CAUSES THE OUTPUT OF THE DATA PASSED TO IT ON A SPECIFIED CHANNEL.
;
; TO CALL FROM FORTRAN USE :    CALL DIGANA(IDATA,ICHAN,IGAIN,IERROR)
;       INTEGER*2 IDATA,ICHAN,IGAIN,IERROR
;
;


;
;STOP AND CLEAR THE DT2801 SREIES BOARD.
;
        MOV     DX,COMMAND_REGISTER
        MOV     AL,CSTOP
        OUT     DX,AL
        MOV     DX,DATA_REGISTER
        IN      AL,DX

        MOV     DX,STATUS_REGISTER
WAIT10: IN      AL,DX
        AND     AL,WRITE_WAIT
        JNZ     WAIT10
OKAY10: IN      AL,DX
        AND     AL,COMMAND_WAIT
        JZ      OKAY10
        MOV     DX,COMMAND_REGISTER
        MOV     AL,CCLEAR
        OUT     DX,AL
;
;WAIT UNTIL THE DT2801 SERIES BOARD DATA IN FULL FLAG IS CLEAR AND READY
;FLAG IS SET, THEN WRITE THE WRITE DAC IMMEDIATE COMMAND BYTE
;TO COMMAND REGISTER.
;
        MOV     DX,STATUS_REGISTER
WAIT11: IN      AL,DX
        AND     AL,WRITE_WAIT
        JNZ     WAIT11
OKAY11: IN      AL,DX
        AND     AL,COMMAND_WAIT
        JZ      OKAY11
        MOV     DX,COMMAND_REGISTER
        MOV     AL,CDAOUT
        OUT     DX,AL
;
;WAIT UNTIL THE DT2801 SERIES BOARD DATA IN FULL FLAG IS CLEAR, THEN WRITE
;THE DAC SELECT BYTE TO THE DATA IN REGISTER.
;
        MOV     DX,STATUS_REGISTER
WAIT12: IN      AL,DX
        AND     AL,WRITE_WAIT
        JNZ     WAIT12
        MOV     DX,DATA_REGISTER
        LDS     SI,ES:PARM2[BX]
        MOV     AL,[SI]
        OUT     DX,AL
;
;DIVIDE THE DATA INTO HIGH AND LOW BYTES AND WRITE BOTH BYTES TO THE DATA
;IN REGISTER, WAITING FOR A CLEAR DATA IN FULL FLAG BEFORE EACH WRITE.
;
        LDS     SI,ES:PARM1[BX]
        MOV     DX,STATUS_REGISTER
WAIT13: IN      AL,DX
        AND     AL,WRITE_WAIT
        JNZ     WAIT13
        MOV     DX,DATA_REGISTER
        MOV     AX,[SI]
        OUT     DX,AL
        MOV     DX,STATUS_REGISTER
WAIT14: IN      AL,DX
        AND     AL,WRITE_WAIT
        JNZ     WAIT14
        MOV     DX,DATA_REGISTER
        MOV     AX,[SI]
        MOV     AL,AH
```

```
        OUT     DX,AL
;
;WAIT UNTIL THE DT2801 SERIES BOARD DATA IN FULL FLAG IS CLEAR AND READY
;FLAG IS SET, INDICATING COMMAND COMPLETION, THEN CHECK THE STATUS REGISTER
;ERROR FLAG.
;
        MOV     DX,STATUS_REGISTER
WAIT15: IN      AL,DX
        AND     AL,WRITE_WAIT
        JNZ     WAIT15
OKAY15: IN      AL,DX
        AND     AL,COMMAND_WAIT
        JZ      OKAY15
        LDS     SI,ES:PARM4[BX]
        MOV     CL,NOERR
        MOV     [SI],CL
        IN      AL,DX
        AND     AL,ERRCK
        JZ      BYE
;
;    ERROR HANDELING ROUTINE
;
        MOV     CL,ERR
        MOV     [SI],CL
BYE:    RET
DIGANA  ENDP

;*****************************************************************

SUBTTL PORTIN

PORTIN PROC FAR

PUBLIC PORTIN

;
;PORTIN SETS THE DIGITAL PORT FOR INPUT. DIGRD SHOULD BE USED TO READ THE
;VALUE OF THE PORT.
;
;TO CALL FROM FORTRAN USE :  CALL PORTIN(IPORT,IERROR)
;      INTEGER*2 IPORT,IERROR
;


;
;STOP AND CLEAR THE DT2801 SERIES BOARD
;
        MOV     DX,COMMAND_REGISTER
        MOV     AL,CSTOP
        OUT     DX,AL
        MOV     DX,DATA_REGISTER
        IN      AL,DX

        MOV     DX,STATUS_REGISTER
WAIT1A: IN      AL,DX
        AND     AL,WRITE_WAIT
        JNZ     WAIT1A
OKAY1A: IN      AL,DX
        AND     AL,COMMAND_WAIT
        JZ      OKAY1A
        MOV     DX,COMMAND_REGISTER
        MOV     AL,CCLEAR
        OUT     DX,AL
;
;WAIT UNTIL DATA IN FLAG IS CLEAR AND READY FLAG IS SET, THEN WRITE THE
;SET DIGITAL PORT FOR INPUT COMMAND BYTE TO THE DATA IN REGISTER.
;
        MOV     DX,STATUS_REGISTER
WAIT16: IN      AL,DX
        AND     AL,WRITE_WAIT
        JNZ     WAIT16
OKAY16: IN      AL,DX
        AND     AL,COMMAND_WAIT
        JZ      OKAY16
```

```
        MOV     DX,COMMAND_REGISTER
        MOV     AL,CSIN
        OUT     DX,AL
;
;WAIT UNITL THE DATA IN FULL FLAG IS CLEAR, THE WRITE THE DIGITAL PORT
;SELECT BYTE TO THE DATA IN REGISTER.
;
        MOV     DX,STATUS_REGISTER
WAIT17: IN      AL,DX
        AND     AL,WRITE_WAIT
        JNZ     WAIT17
        MOV     DX,DATA_REGISTER
        LDS     SI,ES:PARM1[BX]
        MOV     AL,[SI]
        OUT     DX,AL
;
;WAIT UNTIL THE DATA IN FLAG IS CLEAR AND READY FLAG IS SET,
;INDICATING COMMAND COMPLETION, THEN CHECK THE STATUS REGISTER ERROR FLAG.
;
        MOV     DX,STATUS_REGISTER
WAIT18: IN      AL,DX
        AND     AL,WRITE_WAIT
        JNZ     WAIT18
OKAY18: IN      AL,DX
        AND     AL,COMMAND_WAIT
        JZ      OKAY18
        LDS     SI,ES:PARM2[BX]
        MOV     CL,NOERR
        MOV     [SI],CL
        IN      AL,DX
        AND     AL,ERRCK
        JZ      OVER
;
;   ERROR HANDELING ROUTINE
;
        MOV     CL,ERR
        MOV     [SI],CL
OVER:   RET
PORTIN  ENDP

;***********************************************************

SUBTTL PORTOT

PORTOT PROC FAR

PUBLIC PORTOT

;
;PORTOT SETS A PORT FOR OUTPUT. DIGWR SHOULD BE USED TO WRITE THE THE PORT
;SET UP FOR OUTPUT.
;
; TO CALL FROM FORTRAN USE :  CALL PORTOT(IPORT,IERROR)
;     INTEGER*2 IPORT,IERROR
;

;
;STOP AND CLEAR THE DT2801 SERIES BOARD
;
        MOV     DX,COMMAND_REGISTER
        MOV     AL,CSTOP
        OUT     DX,AL
        MOV     DX,DATA_REGISTER
        IN      AL,DX

        MOV     DX,STATUS_REGISTER
WAIT19: IN      AL,DX
        AND     AL,WRITE_WAIT
        JNZ     WAIT19
OKAY19: IN      AL,DX
        AND     AL,COMMAND_WAIT
        JZ      OKAY19
        MOV     DX,COMMAND_REGISTER
```

```
        MOV     AL,CCLEAR
        OUT     DX,AL
;
;WAIT UNTIL DATA IN FLAG IS CLEAR AND READY FLAG IS SET, THEN WRITE THE
;SET DIGITAL PORT FOR OUTPUT COMMAND BYTE TO THE DATA IN REGISTER.
;
        MOV     DX,STATUS_REGISTER
WAIT20: IN      AL,DX
        AND     AL,WRITE_WAIT
        JNZ     WAIT20
OKAY20: IN      AL,DX
        AND     AL,COMMAND_WAIT
        JZ      OKAY20
        MOV     DX,COMMAND_REGISTER
        MOV     AL,CSOUT
        OUT     DX,AL
;
;WAIT UNITL THE DATA IN FULL FLAG IS CLEAR, THE WRITE THE DIGITAL PORT
;SELECT BYTE TO THE DATA IN REGISTER.
;
        MOV     DX,STATUS_REGISTER
WAIT21: IN      AL,DX
        AND     AL,WRITE_WAIT
        JNZ     WAIT21
        MOV     DX,DATA_REGISTER
        LDS     SI,ES:PARM1[BX]
        MOV     AL,[SI]
        OUT     DX,AL
;
;WAIT UNTIL THE DATA IN FULL FLAG IS CLEAR AND READY FLAG IS SET,
;INDICATING COMMAND COMPLETION, THEN CHECK THE STATUS REGISTER ERROR FLAG.
;
        MOV     DX,STATUS_REGISTER
WAIT22: IN      AL,DX
        AND     AL,READ_WAIT
        JZ      WAIT22
OKAY22: IN      AL,DX
        AND     AL,COMMAND_WAIT
        JZ      OKAY22
        LDS     SI,ES:PARM2[BX]
        MOV     CL,NOERR
        MOV     [SI],CL
        IN      AL,DX
        AND     AL,ERRCK
        JZ      HOP
;
;    ERROR HANDELING ROUTINE
;
        MOV     CL,ERR
        MOV     [SI],CL
HOP:    RET
PORTOT  ENDP

;***********************************************************

SUBTTL DIGRD

DIGRD  PROC FAR

PUBLIC DIGRD

;
;DIGRD READS A DIGITAL INPUT BYTE FROM THE PORT SPECIFIED. PORTIN MUST BE
;USED ONCE BEFORE DIGRD TO INITIALIZE THE PORT FOR INPUT.
;
; TO CALL FROM FORTRAN USE :    CALL DIGRD(IPORT,JDATA,IERROR)
;       INTEGER*2 IPORT,JDATA(8),IERROR
;           JDATA : 1 = LOW BIT ........... 8 = HIGH BIT
;
;

;
;STOP AND CLEAR THE DT2801 SERIES BOARD
;
```

```
        MOV     DX,COMMAND_REGISTER
        MOV     AL,CSTOP
        OUT     DX,AL
        MOV     DX,DATA_REGISTER
        IN      AL,DX

        MOV     DX,STATUS_REGISTER
WAIT23: IN      AL,DX
        AND     AL,WRITE_WAIT
        JNZ     WAIT23
OKAY23: IN      AL,DX
        AND     AL,COMMAND_WAIT
        JZ      OKAY23
        MOV     DX,COMMAND_REGISTER
        MOV     AL,CCLEAR
        OUT     DX,AL
;
;WAIT UNTIL DATA IN FLAG IS CLEAR AND READY FLAG IS SET, THEN WRITE THE
;READ DIO IMMEDIATE COMMAND BYTE TO THE DATA IN REGISTER.
;
        MOV     DX,STATUS_REGISTER
WAIT24: IN      AL,DX
        AND     AL,WRITE_WAIT
        JNZ     WAIT24
OKAY24: IN      AL,DX
        AND     AL,COMMAND_WAIT
        JZ      OKAY24
        MOV     DX,COMMAND_REGISTER
        MOV     AL,CDIOIN
        OUT     DX,AL
;
;WAIT UNITL THE DATA IN FULL FLAG IS CLEAR, THE WRITE THE DIO PORT
;SELECT BYTE TO THE DATA IN REGISTER.
;
        MOV     DX,STATUS_REGISTER
WAIT25: IN      AL,DX
        AND     AL,WRITE_WAIT
        JNZ     WAIT25
        MOV     DX,DATA_REGISTER
        LDS     SI,ES:PARM1[BX]
        MOV     AL,[SI]
        OUT     DX,AL
;
;WAIT UNTIL THE DATA OUT READY FLAG IS SET, THEN READ THE DIO DATA BYTE
;FROM THE DATA OUT REGISTER.
;
        MOV     DX,STATUS_REGISTER
WAIT26: IN      AL,DX
        AND     AL,READ_WAIT
        JZ      WAIT26
        MOV     DX,DATA_REGISTER
        IN      AL,DX
        LDS     SI,ES:PARM2[BX]
        MOV     CX,8
SHR1:   MOV     DX,00H
        SHR     AL,1
        JNC     SHR2
        MOV     DX,01H
SHR2:   MOV     [SI],DX
        INC     SI
        INC     SI
        LOOP    SHR1
;
;WAIT UNTIL THE DATA IN FLAG IS CLEAR AND READY FLAG IS SET,
;INDICATING COMMAND COMPLETION, THEN CHECK THE STATUS REGISTER ERROR FLAG.
;
        MOV     DX,STATUS_REGISTER
WAIT27: IN      AL,DX
        AND     AL,WRITE_WAIT
        JNZ     WAIT27
OKAY27: IN      AL,DX
        AND     AL,COMMAND_WAIT
        JZ      OKAY27
```

```
        LDS     SI,ES:PARM3[BX]
        MOV     CL,NOERR
        MOV     [SI],CL
        IN      AL,DX
        AND     AL,ERRCK
        JZ      SKIP
;
;   ERROR HANDELING ROUTINE
;
        MOV     CL,ERR
        MOV     [SI],CL
SKIP:   RET
DIGRD   ENDP

;*******************************************************

SUBTTL DIGWR

DIGWR  PROC FAR

PUBLIC DIGWR

;
;DIGWR WRITES A DIGITAL OUTPUT BYTE TO THE PORT SPECIFIED BY IPORT. PORTOT
;MUST BE USED ONCE PRIOR TO DIGRD TO INITIALIZE THE PORT FOR OUTPUT.
;
; TO CALL FROM FORTRAN USE :  CALL DIGWR(IPORT,JDATA,IERROR)
;       INTEGER*2 IPORT,JDATA(8),IERROR
;            JDATA : 1 = LOW BIT ......... 8 = HIGH BIT
;


;
;STOP AND CLEAR THE DT2801 SERIES BOARD
;
        MOV     DX,COMMAND_REGISTER
        MOV     AL,CSTOP
        OUT     DX,AL
        MOV     DX,DATA_REGISTER
        IN      AL,DX

        MOV     DX,STATUS_REGISTER
WAIT28: IN      AL,DX
        AND     AL,WRITE_WAIT
        JNZ     WAIT28
OKAY28: IN      AL,DX
        AND     AL,COMMAND_WAIT
        JZ      OKAY28
        MOV     DX,COMMAND_REGISTER
        MOV     AL,CCLEAR
        OUT     DX,AL
;
;WAIT UNTIL DATA IN FLAG IS CLEAR AND READY FLAG IS SET, THEN WRITE THE
;READ DIO IMMEDIATE COMMAND BYTE TO THE DATA IN REGISTER.
;
        MOV     DX,STATUS_REGISTER
WAIT29: IN      AL,DX
        AND     AL,WRITE_WAIT
        JNZ     WAIT29
OKAY29: IN      AL,DX
        AND     AL,COMMAND_WAIT
        JZ      OKAY29
        MOV     DX,COMMAND_REGISTER
        MOV     AL,CDIOOUT
        OUT     DX,AL
;
;WAIT UNITL THE DATA IN FULL FLAG IS CLEAR, THE WRITE THE DIO PORT
;SELECT BYTE TO THE DATA IN REGISTER.
;
        MOV     DX,STATUS_REGISTER
WAIT30: IN      AL,DX
        AND     AL,WRITE_WAIT
        JNZ     WAIT30
        MOV     DX,DATA_REGISTER
```

```
        LDS     SI,ES:PARM1[BX]
        MOV     AL,[SI]
        OUT     DX,AL
;
;WAIT UNTIL THE DATA IN FLAG IS CLEAR ANF THE READY FLAG IS CLEAR, INDICATING
;COMMAND COMPLETION, CHECK THE STATUS REGISTER ERROR FLAG.
;
        MOV     DX,STATUS_REGISTER
WAIT31: IN      AL,DX
        AND     AL,WRITE_WAIT
        JNZ     WAIT31
        LDS     SI,ES:PARM2[BX]
        MOV     CX,8
SHR4:   MOV     AH,[SI]
        AND     AH,1H
        INC     SI
        INC     SI
        SHR     AX,1
        LOOP    SHR4
        MOV     DX,DATA_REGISTER
        OUT     DX,AL
;
;WAIT UNTIL THE DATA IN FLAG IS CLEAR AND READY FLAG IS SET,
;INDICATING COMMAND COMPLETION, THEN CHECK THE STATUS REGISTER ERROR FLAG.
;
        MOV     DX,STATUS_REGISTER
WAIT32: IN      AL,DX
        AND     AL,WRITE_WAIT
        JNZ     WAIT32
OKAY32: IN      AL,DX
        AND     AL,COMMAND_WAIT
        JZ      OKAY32
        LDS     SI,ES:PARM3[BX]
        MOV     CL,NOERR
        MOV     [SI],CL
        IN      AL,DX
        AND     AL,ERRCK
        JZ      JUMP
;
;    ERROR HANDELING ROUTINE
;
        MOV     CL,ERR
        MOV     [SI],CL
JUMP:   RET
DIGWR   ENDP
P0POWER ENDS
        END
TITLE  - Routine to operate the analog card
;
;
PARMBLK         STRUC
PARM1           DD      ?
PARM2           DD      ?
PARM3           DD      ?
PARMBLK         ENDS
;
;

SUBTTL  DATA AREA (part of code segment)
BRDDATA SEGMENT PARA COMMON 'DATA'
          ;PARAMETER LIST:
          ;These entered by the user or default used by program.

BRD1     DW 0         ;1st brd address
BRD2     DW 0            ;2nd brd
BRD3     DW 0            ;3rd brd
BRD4     DW 0            ;4th brd
BRD5     DW 0            ;5th brd
BRD6     DW 0            ;6th brd
BRD7     DW 0            ;7th brd
BRD8     DW 0            ;8th brd
BRD9     DW 0            ;9th brd
BRD10    DW 0            ;10th brd
```

```
BRD11     DW 0              ;11th brd
BRD12     DW 0              ;12th brd
RESOL     DB 128            ;fs count = RESOL*128
          DB 0
FILDEL    DB 1              ;Filter dyl = FILDEL/60 (Sec)
          DB 0
CHANS     DB 16             ;No. chans in use per brd
          DB 0
SSTEP     DB 0              ;1=read 1chan per call
          DB 0
SKEY      DB 1              ;1 = RET if key pressed
                            ;0 = ignore keyboard
          DB 0
          SUBTTL DATA AREA
NOZREF    DB 0              ;0 = normal
                            ;1 = no auto zero & scale
          DB 0
CADDR     DW 0FFFFH         ;Addr of cal nos. Used if <> 0FFFFH
                            ;Segment = RADDR

;PARAMETERS CALC BY PROGRAM

RADDR     DW 0FFFFH         ;Range list addr. Offset = 0
                            ;Segment addr of range and data table.
DADDR     DW 0FFFFH         ;Data list addr
SADDR     DW 0FFFFH         ;Scale list addr
OADDR     DW 0FFFFH         ;Offset list addr

BRDS      DB 1              ;No. of brds in use
          DB 0
BRDCNT    DW 1              ;Brd no. being read
CHANCNT   DW 1              ;Next chan # to read
RANGCNT   DW 0              ;Next range no. (data)
ZREFCNT   DB 7              ;Next 0 & ref range
          DB 0
CHSELFL   DB 0              ;1= new chan selected
          DB 0
TEMPCHAN  DW 0              ;CHANCNT in INIT
TEMPRANG  DW 0              ;RANGCNT in INIT
TEMPC     DW 0              ;Temporary registers
TEMPD     DW 0
TEMPF     DB 0
          DB 0
TIMR_CNT  DW 0              ;Last reading of timer
CLK_TICK  DB 0              ;Missed clk ticks * 2
          DB 0
         ;Locate brd by searching these addresses:
LOC_TABLE DB    2H,3H,0BH,0CH,12H,13H,1BH,1CH
REV        DB 0           ;PROGRAM REVISION
           DB    57H,4DH,61H,63H,6CH,61H,79H
           ;DATA

BRDDATA    ENDS

SUBTTL
ANALDATA SEGMENT PARA COMMON 'DATA'
           DW     1512     DUP(?)
ANALDATA   ENDS
POKEY SEGMENT PARA 'CODE'
;*********************************************
LOCATE   PROC      FAR
;Locate brds
;must pwr on 1st. or clear
;brd data to 0 (use RESET)
ASSUME   CS:POKEY,DS:BRDDATA,ES:NOTHING
PUBLIC   LOCATE
           PUSH   DS
           PUSH   ES
           PUSH   DI
           PUSH   SI
           MOV    AX,BRDDATA
           MOV    DS,AX
;Search I/O addr 100H to 0FFE0H
```

```
           MOV    BRDS,0             ;BRDS=0
           MOV    BP,1FE0H           ;1st. address to check
LOC_NEXT:  MOV    SI,0
SEARCH:    MOV    DX,BP
           ADD    DL,LOC_TABLE[SI]   ;Search address from table
           ADC    DH,0
           IN     AL,DX
           CMP    AL,0
           JNE    NEXT1              ;No brd here.
           INC    SI
           CMP    SI,8
           JNE    SEARCH             ;Keep looking, this may be a brd.
           ;Good so far. Now check addr 9. See if it decrements.
           MOV    DX,BP
           ADD    DX,9
           MOV    CX,10              ;Decrement 10 times
           MOV    SI,14              ;In 14 readings or less
           PUSHF
           CLI                       ;Disable interrupts
           CALL   TIMR_READ
           IN     AL,DX
           MOV    BL,AL              ;1st. reading
DECR:      SUB    BL,1
DECR1:     MOV    AX,68H             ;Wait 200 uS.
           CALL   WAIT
           DEC    SI
           JZ     NEXT               ;No brd here
           IN     AL,DX
           CMP    BL,AL              ;Data decr?
           JNE    DECR1              ;No
           LOOP   DECR               ;Yes
           ;A brd found
           CALL   ROLL_OVER          ;Update CLK_TICK
           POPF                      ;Enable interrupts
           SUB    BH,BH
           MOV    BL,BRDS
           SHL    BX,1               ;BX * 2
           MOV    BRD1[BX],BP        ;Save base address of brd
           INC    BRDS               ;No. of brds found
           CMP    BRDS,12
           JE     DONE
           JNE    NEXT1
NEXT:      CALL   ROLL_OVER          ;Update CLK_TICK
           POPF                      ;Enable interrupts
NEXT1:     SUB    BP,20H             ;Next base address
           CMP    BP,0100H           ;Last address checked?
           JAE    LOC_NEXT           ;No
           ;All addresses checked
DONE:      SUB    BH,BH
           MOV    BL,BRDS
           SHL    BX,1               ;BX * 2
DONE1:     MOV    BRD1[BX],0         ;If no brd, addr. = 0
           ADD    BX,2
           CMP    BX,24              ;12 words
           JNE    DONE1              ;Not done
           POP    SI
           POP    DI
           POP    ES
           POP    DS
           RET
           LOCATE ENDP
;*********************************************
;INITIALIZE OFFSET AND SCALE OF EACH CHANNEL.
           ;(Auto zero and scale)
;First call 'LOCATE'.

INIT       PROC   FAR
PUBLIC     INIT
           PUSH DS
           PUSH ES
           PUSH DI
           PUSH SI
           MOV    AX,BRDDATA
```

```
        MOV     DS,AX
        ;Setup parameters
        MOV     CHSELFL,0
        MOV     RANGCNT,0
        MOV     ZREFCNT,7
        MOV     CHANCNT,1
        MOV     BRDCNT,1
        ;Any brds located?
        CMP     BRDS,0
        JNE     CALC1                   ;Yes
        CALL    BEEP                    ;Error
        POP   SI
        POP   DI
        POP   ES
        POP   DS
        RET
        ;Calculate RADDR
CALC1:  MOV     AX,RADDR
        MOV     ES,AX                   ;Segment of range and data table
        CMP     AX,0FFFFH               ;Is it (RADDR) default?
        JNE     CALC2                   ;No, use it.
        MOV     AX,ANALDATA             ;Yes, it will follow THE_KEY
        MOV     RADDR,AX
        POP   SI
        POP   DI
        POP   ES
        POP   DS
        RET                             ;Enter ranges and call this subroutine
                                        ;again.
        ;Calc DADDR
CALC2:  MOV     AL,CHANS
        ADD     AL,2                    ;CHANS+2
        MUL     BRDS                    ;AX=BRDS*(CHANS+2)
        MOV     DADDR,AX                ;Size of range data
        ;Are ranges valid? (only checks 1st brd)
CALC3:  MOV     BL,CHANS
        SUB     BH,BH
        MOV     SI,BX
CALC4:  DEC     SI
        JS      CALC6                   ;All ranges valid
        CMP     BYTE PTR ES:[SI],7
        JA      CALC5
        CMP     BYTE PTR ES:[SI],4
        JE      CALC5
        CMP     BYTE PTR ES:[SI],1
        JAE     CALC4
CALC5:  CALL    BEEP
        MOV     AX,0FFFFH
        CALL    WAIT
        CALL    BEEP
        POP   SI
        POP   DI
        POP   ES
        POP   DS
        RET
        ;Calc SADDR,OADDR
CALC6:  MOV     AX,DADDR
        SHL     AX,1
        MOV     BX,AX
        ADD     AX,DADDR                ;AX=DADDR*3
        MOV     SADDR,AX
        ADD     AX,BX                   ;AX=DADDR*5
        MOV     OADDR,AX
        CALL    AENTR
        POP     SI
        POP     DI
        POP     ES
        POP     DS
        RET
        INIT    ENDP
;**************************************************
AENTR   PROC    NEAR                    ;****CHECK THIS, SHOULD IT BE "FAR"?
        MOV     AL,CHANS
```

```
          ADD     AL,2
          MOV     EX,BRDCNT
          SUB     BL,1
          MUL     BL
          MOV     TEMPRANG,AX          ;TEMPRANG=(CHANS+2)*(BRDCNT-1)=initial range
          MOV     TEMPCHAN,1           ;TEMPCHAN=initial chanel count
AENTR3:   MOV     TEMPF,0
          MOV     SI,TEMPRANG
          MOV     AL,ZREFCNT
          CMP     AL,ES:[SI]           ;Range=ZREFCNT?
          JNE     AEN                  ;No
          JMP     AENTR4               ;Yes, match
AEN:      INC     TEMPRANG
          INC     TEMPCHAN
          MOV     AL,CHANS
          INC     AL
          CMP     BYTE PTR TEMPCHAN,AL   ;TEMPCHAN=CHANS+1?
          JNE     SRCH1                ;No
          MOV     TEMPF,5              ;Range
          CMP     ZREFCNT,5            ;10V range?
          JE      AENTR4               ;Yes CJ sensor
          INC     TEMPCHAN             ;TEMPCHAN = CHANS+2
          INC     TEMPRANG
          MOV     TEMPF,7              ;Range
          CMP     ZREFCNT,7            ;TEMPCHAN=CHANS+2 & ZERFCNT=7?
RJMP:     JE      AENTR4               ;Yes, 50mV offset
SRCH1:    MOV     AX,TEMPCHAN
          CMP     AL,CHANS             ;Search done?
          JBE     AENTR3               ;No
SRCHCONT: CMP     CHANCNT,1            ;CHANCNT=1?
          JE      SRC                  ;Yes
          RET                          ;No, return to MEASURE
SRC:      MOV     TEMPCHAN,1
          INC     TEMPRANG
          INC     BRDCNT
          MOV     AX,BRDCNT
          CMP     AL,BRDS              ;Last board?
          JBE     AENTR3               ;No
          MOV     BRDCNT,1
          MOV     TEMPRANG,0
          SUB     ZREFCNT,1
          CMP     ZREFCNT,0            ;Last range?
          JE      SRC1                 ;Yes
          JMP     AENTR3               ;No
SRC1:     MOV     ZREFCNT,7
          MOV     BRDCNT,1
          MOV     CHANCNT,1
          MOV     RANGCNT,0
          RET                          ;Exit INIT
 AENTR4:  CALL    READZR
          CMP     CHSELFL,0
          JE      SRCHCONT             ;Continue
          RET                          ;Setup, don't read
          AENTR ENDP
;**************************************************
READZR    PROC    NEAR
;Read & save offset & scale
          CALL    PIA_SETUP
          MOV     AL,CHANS
          ADD     AL,4
          CMP     BYTE PTR CHANCNT,AL     ;CHANCNT=CHANS+4
          JE      RDSCALE              ;Yes, don't read offset
;READ OFFSET AND STORE IT
          MOV     DX,BP                ;I/O address of 1st 6522
          SUB     AL,3
          CMP     BYTE PTR TEMPCHAN,AL    ;TEMPCHAN=CHANS+1
          JB      RDOFF1               ;Yes
          MOV     AL,TEMPF             ;No range=CJ or 50mV
          JMP     RDOFF2
RDOFF1:   MOV     SI,TEMPRANG
          MOV     AL,ES:[SI]
RDOFF2:   OR      AL,10H
          OUT     DX,AL                ;Range
```

```
         ADD    DX,10H               ;I/O address of 2nd 6522
         MOV    AL,83H
         OUT    DX,AL                ;Channel = 0 reference
         CMP    CHSELFL,1            ;Read data or just setup?
         JNE    RDOFF25              ;Read data
         RET                         ;Setup and return to MEASURE
RDOFF25: CALL   READ_SETUP
         CALL   READ
         ;Calculate index for offset storage
RDOFF3:  MOV    SI,TEMPRANG
         SHL    SI,1
         ADD    SI,OADDR             ;SI=TEMPRANG*2+OADDR
         ;Save count at offset
RDOFF4:  MOV    ES:[SI],BX           ;Saved
;READ AND SAVE SCALE
         ;Skip reading of scale?
RDSCALE: MOV    AL,CHANS
         ADD    AL,3
         CMP    BYTE PTR CHANCNT,AL     ;TEMPCHAN=CHANS+2?
         JNE    RDSCL                ;No
         JMP    SRCH2                ;Yes, don't read scale
         ;First find scale index
RDSCL:   MOV    SI,TEMPRANG
         SHL    SI,1                 ;SI=TEMPRANG*2
         ADD    SI,SADDR             ;SI=scale index
         ;Find channel addr. & put in TEMPC
         MOV    TEMPC,82H            ;50mV ref
         MOV    BL,ZREFCNT
         AND    BL,02H               ;Bit 1=1?
         JNE    RDSCL0               ;Yes, use 50mV ref
         MOV    TEMPC,81H            ;6.9V ref
         ; Now finf.d range and put in AL
RDSCL0:  MOV    BL,CHANS
         INC    BL                   ;BL=CHANS+1
         CMP    BYTE PTR TEMPCHAN,BL    ;CJ sensor?
         JB     RDSCL1               ;No
         MOV    TEMPC,81H            ;6.9V ref channel
         MOV    AL,5                 ;CJ sensor range (10V)
         JMP    RDSCL2
RDSCL1:  MOV    DI,TEMPRANG          ;DI=range index
         MOV    AL,BYTE PTR ES:[DI]     ;Range
         CMP    AL,3                 ;25mV range?
         JNE    RDSCL15              ;No
         MOV    AL,7                 ;Yes, use 50mV range
RDSCL15: CMP    AL,1                 ;5V range?
         JNE    RDSCL2               ;No
         ;5V range: use 10V scale from CJ
         ;So calc CJ scale index =
         ;      (TEMPRANG-TEMPCHANG+1+CHANS)*2+SADDR
         INC    DI                   ;DI=TEMPRANG+1
         SUB    DI,TEMPCHAN
         SUB    BX,BX
         MOV    BL,CHANS
         ADD    DI,BX
         SHL    DI,1
         ADD    DI,SADDR             ;DI=CJ scale index
         ;Move CJ scale to 5V scale
         MOV    AX,ES:[DI]
         MOV    ES:[SI],AX
         JMP    RDDIV
         ;Range & channel found, now put into PIA
         ;Select range
RDSCL2:  MOV    DX,BP                ;1st 6522 addr
         OR     AL,10H
         OUT    DX,AL
         ;Select channel
         ADD    DX,10H               ;2nd 6522 addr
         MOV    AX,TEMPC
         OUT    DX,AL
         CMP    CHSELFL,1            ;Read data just set up?
         JNE    RDSCL25
         RET                         ;Set up & return to MEASURE
RDSCL25: CALL    READ_SETUP
```

```
        CALL    READ
        ;Save count at scale
        ;Calc scale index=TEMPRANG*2+SADDR
RDSCL3: MOV     SI,TEMPRANG
        SHL     SI,1
        ADD     SI,SADDR            ;SI = scale index
        MOV     ES:[SI],BX          ;Saved
        ;Calc offset index=TEMPRANG*2+OADDR
        MOV     DI,TEMPRANG
        SHL     DI,1
        ADD     DI,OADDR            ;DI = offset index
        MOV     BX,TEMPRANG
        CMP     BYTE PTR ES:[BX],3      ;25mV range?
        JNE     SUBOFF2             ;No
        ;Calc 50mV index =
        ;          (TEMPRANG-TEMPCHAN+2+CHANS)*2+OADDR
        MOV     DI,TEMPRANG
        ADD     DI,2
        SUB     DI,TEMPCHAN
        SUB     BX,BX
        MOV     BL,CHANS
        ADD     DI,BX
        SHL     DI,1
        ADD     DI,OADDR            ;DI=50mV offset index
        ;Subrtact offset
SUBOFF2: MOV    AX,ES:[DI]
        SUB     ES:[SI],AX
        MOV     BX,1
        ADD     BL,CHANS            ;BX=CHANS+1
        CMP     TEMPCHAN,BX
        JAE     SRCH2               ;CJ sensor or 50mV offset
        ;Adjust scale for 25mV range
ADJ:    MOV     BX,1
        CMP     ZREFCNT,3           ;25mV range?
        JNE     RDSCL4              ;No
        ;Divide scale by 2 if 25mV range
RDDIV:  SHR WORD PTR ES:[SI],1
        JMP     SRCH2
RDSCL4: CMP     ZREFCNT,2           ;250mV range?
        JNE     RDSCL5              ;No
        MOV     BX,3
RDSCL5: CMP     ZREFCNT,6           ;500mV range?
        JNE     RDSCL6              ;No
        MOV     BX,4
        ;Divide offset by 2↑(BX-1)
RDSCL6: MOV     CX,BX
        DEC     CX                  ;CL=BX-1
        JE      SRCH2               ;CL=0
        SHR     WORD PTR ES:[DI],CL
        ;Search for another range=ZREFCNT
SRCH2:  MOV     AX,TEMPCHAN
        MOV     TEMPC,AX            ;Save chan count
        MOV     AX,TEMPRANG
        MOV     TEMPD,AX            ;Save current range count
        JMP     SRCH4
SRCH3:  MOV     TEMPF,0
        MOV     SI,TEMPRANG
        MOV     AL,ES:[SI]
        CMP     AL,ZREFCNT          ;Range=ZREFCNT?
        JE      MOVOFF              ;Yes, match
SRCH4:  INC     TEMPRANG
        INC     TEMPCHAN
        MOV     TEMPF,5
        MOV     AL,CHANS
        INC     AL                  ;AL=CHANS+1
        CMP     BYTE PTR TEMPCHAN,AL    ;TEMPCHAN: CHANS+1
        JNE     SRCH5               ;No
        CMP     ZREFCNT,5           ;Yes, 10V range?
        JE      MOVOFF              ;Yes, match for CJ sensor
        INC     TEMPRANG
        INC     TEMPCHAN            ;TEMPCHAN=CHANS+2
        MOV     TEMPF,7
        CMP     ZREFCNT,7           ;50mV range?
```

```
         JE     MOVOFF              ;Yes, match for 50mV special
         ;adjust TEMPCHAN & TEMPRANG to <= CHANS+1
SRCH5:   MOV    AL,CHANS
         ADD    AL,2
         CMP    BYTE PTR TEMPCHAN,AL    ;TEMPCHAN<=CHANS+1?
         JBE    SRCH6               ;Yes
         DEC    TEMPCHAN
         DEC    TEMPRANG
         JMP    SRCH5
SRCH6:   MOV    AX,TEMPCHAN
         CMP    AL,CHANS            ;Search done?
         JBE    SRCH3               ;No
         RET                        ;Yes, try next board
         ;Calc new offset index
MOVOFF:  MOV    SI,TEMPRANG
         SHL    SI,1
         ADD    SI,OADDR
         ;SI=new offset index=TEMPRANG*2+OADDR
         ;Calc old offset index
MOVOFF1: MOV    DI,TEMPD
         SHL    DI,1
         ADD    DI,OADDR
         ;DI=old offset index=(old TEMPRANG)*2+OADDR
         ;Now move offset
         MOV    AX,ES:[DI]
         MOV    ES:[SI],AX
         CMP    TEMPF,7             ;50mV special?
         JE     MOVOFF2             ;Yes, don't move scale
         ;Calc new scale index
         MOV    SI,TEMPRANG
         SHL    SI,1
         ADD    SI,SADDR
         ;SI=new scale index=TEMPRANG*2+SADDR
         ;Calc old scale index
         MOV    DI,TEMPD
         SHL    DI,1
         ADD    DI,SADDR
         ;DI=old scale index=(old TEMPRANG)*2+SADDR
         ;Now move scale
         MOV    AX,ES:[DI]
         MOV    ES:[SI],AX
MOVOFF2: JMP    SRCH4
         READZR ENDP
;**************************************************************
MEASURE PROC FAR
PUBLIC MEASURE
         PUSH   BP
         PUSH   DS
         PUSH   ES
         PUSH   DI
         PUSH   SI
         MOV    AX,BRDDATA
         MOV    DS,AX
         ;Measure data
         ;First run LOCATE and INIT
M1:      MOV    ES,RADDR
         MOV    CHSELFL,0
M2:      MOV    AL,CHANS
         ADD    AL,3
         CMP    BYTE PTR CHANCNT,AL     ;CHANCNT<CHANS+3?
         JB     READDAT             ;Yes, read data
         JMP    AUTO                ;Do auto zero & scale
         ;Read the data
READDAT: CALL PIA_SETUP
         ;Find chan addr.
         MOV    AX,CHANCNT
         ADD    AX,3                ;Adjust
         CMP    AX,0CH              ;Chan addr < 0CH?
         JB     RDDAT1              ;Yes
         ADD    AX,4                ;No, adjust
         CMP    AX,14H              ;Chan addr < 14H?
         JB     RDDAT1              ;Yes
         ADD    AX,0CH              ;No, adjust
```

```
RDDAT1:   MOV   TEMPC,AX
          ;Find range
          MOV   AL,CHANS
          INC   AL
          CMP   BYTE PTR CHANCNT,AL     ;1st CJ sensor?
          JNE   RDD                 ;No
          MOV   TEMPC,80H           ;CJ Sensor chan
          MOV   AL,15H              ;CJ Sensor range
          JMP   RDDAT3
RDD:      INC   AL
          CMP   BYTE PTR CHANCNT,AL     ;2nd CJ sensor?
          JNE   RDDAT2              ;No
          MOV   TEMPC,0             ;2nd CJ chan
          MOV   AL,1DH              ;2nd CJ range
          JMP   RDDAT3
RDDAT2:   MOV   DI,RANGCNT
          MOV   AL,ES:[DI]
          OR    AL,10H              ;Range
RDDAT3:   MOV   DX,BP
          OUT   DX,AL               ;Select range
          MOV   AX,TEMPC
          ADD   DX,10H              ;2nd 6522 addr
          OUT   DX,AL               ;Select channel
          CMP   CHSELFL,1           ;Read data or just set up?
          JNE   RDDAT4              ;Read data
          JMP   CENTR               ;Set up & return
RDDAT4:   MOV   BX,2
          CALL READ
          ;Scale index=RANGCNT*2+SADDR
          MOV   SI,RANGCNT
          SHL   SI,1
          ADD   SI,SADDR
          ;Offset index= RANGCNT*2+OADDR
          MOV   DI,RANGCNT
          SHL   DI,1
          ADD   DI,OADDR
          ;Adj indices if 2nd CJ sensor
          MOV   AL,CHANS
          ADD   AL,2
          CMP   BYTE PTR CHANCNT,AL     ;2nd CJ sensor?
          JNE   RDDAT5              ;No
          SUB   SI,2                ;Yes,adjust
          SUB   DI,2
          ;Subtract offset
RDDAT5:   SUB   BX,ES:[DI]
          ;Divide by scale
DIV:      MOV   DX,BX
          ;Convert if negative
          MOV   BX,0                ;Sign flag=pos
          MOV   AX,8000H
          AND   AX,DX
          JZ    DIV0
          SUB   AX,AX
          SUB   AX,DX
          MOV   DX,AX
          MOV   BX,1                ;Sign flag=neg
DIV0:     SUB   AX ,AX              ;Data in DX:AX
          MOV   CX,2
DIV1:     SHR   DX,1                ;Div by 4
          RCR   AX,1
          LOOP  DIV1
          DIV   WORD PTR ES:[SI]
          ;Divide by calibration #
DIV2:     MOV   CX,0FFFFH
          CMP   CX,CADDR            ;Div by CAL?
          JE    CONV                ;No
          MOV   DX,AX               ;Data
          ;Calc CAL index
          MOV   AX,BRDCNT
          DEC   AX
          MOV   CL,6
          MUL   CL                  ;AX=(BRDCNT-1)*6
          MOV   SI,CADDR
```

```
         ADD    SI,AX                  ;SI=addr of ACAL
         MOV    AL,CHANS
         CMP    BYTE PTR CHANCNT,AL     ;CJ sensor?
         JA     DIV3                   ;Yes, use ACAL
         MOV    DI,RANGCNT
         MOV    AL,ES:[DI]             ;Get range
         CMP    AL,7
         JE     DIV3                   ;Range 7, use ACAL
         CMP    AL,3
         JE     DIV3                   ;Range 3, use ACAL
         ADD    SI,2                   ;SI=addr of BCAL
         CMP    AL,5
         JE     DIV3                   ;Range 5, use BCAL
         CMP    AL,1
         JE     DIV3                   ;Range 1, use BCAL
         ADD    SI,2                   ;Range 6 or 2, use DCAL
DIV3:    SUB    AX,AX
         MOV    CX,1
DIV4:    SHR    DX,1                   ;Divide by 2
         RCR    AX,1
         LOOP   DIV4
         DIV    WORD PTR ES:[SI]
         ;Convert back to negative
CONV:    CMP    BX,0
         JE     DSTORE                 ;It's positive
         SUB    BX,BX
         SUB    BX,AX
         MOV    AX,BX
        ;Calc data index=RANGCNT*2+DADDR & save data
DSTORE:  MOV    SI,RANGCNT
         SHL    SI,1
         ADD    SI,DADDR
         MOV    ES:[SI],AX             ;Save data
         JMP    BENTR
         ;Do auto zero and scale
AUTO:    CMP    NOZREF,1               ;Do auto zero & scale?
         JE     BENTR                  ;No
         CALL   AENTR                  ;Yes
         CMP    CHSELFL,1
         JE     CENTR
         ;Incr CHANCNT, RANGCNT, BRDCNT, ZREFCNT
         ;Set up chan & range & go to CENTR
         ;to continue 'MEASURE' or exit.
BENTR:   INC    CHANCNT
         INC    RANGCNT
         MOV    AL,CHANS
         ADD    AL,3
         CMP    AL,BYTE PTR CHANCNT
         JAE    B1                     ;CHANS+3>=CHANCNT
         ;RANGCNT dosen't incr when CHANCNT>=CHANS+3
         DEC    RANGCNT
         ADD    AL,2
         CMP    AL,BYTE PTR CHANCNT
         JNE    B1                     ;CHANS+5<>CHANCNT
         MOV    CHANCNT,1
         INC    BRDCNT
         MOV    AL,BRDS
         CMP    AL,BYTE PTR BRDCNT
         JAE    B1                     ;BRDS >= BRDCNT
         MOV    BRDCNT,1
         MOV    RANGCNT,0
         SUB    ZREFCNT,1
         CMP    ZREFCNT,-1             ;ZREFCNT = -1?
         JNE    B1                     ;No
         MOV    ZREFCNT,7              ;Yes, last range done
B1:      MOV     CHSELFL,1
         ;Set up chan & range & go to CENTR
         JMP    M2
         ;Exit from 'MEASURE' if
         ;all chans & all brds done,
         ;or key pressed & SKEY=1,
         ;or SSTEP=1
CENTR:   CMP    BRDCNT,1
```

```
        JNE   C1                  ;BRDCNT <> 1
        CMP   CHANCNT,1
        JNE   C1                  ;CHANCNT <> 1
        POP   SI
        POP   DI
        POP   ES
        POP   DS
        POP   BP
        RET                       ;Exit
C1:     CMP   SSTEP,1             ;SSTEP=1?
        JNE   C2                  ;No
        POP   SI
        POP   DI
        POP   ES
        POP   DS
        POP   BP
        RET                       ;Yes, exit
C2:     CMP   SKEY,1              ;SKEY=1?
        JNE   C3                  ;No, continue
        ;Check for keypress
        MOV   AH,0BH
        INT   21H                 ;Chk std input status
        CMP   AL,0FFH             ;Key pressed?
        JNE   C3                  ;No
        POP   SI
        POP   DI
        POP   ES
        POP   DS
        POP   BP
        RET                       ;Yes, exit
C3:     JMP   M1
        MEASURE ENDP
;*************************************************************
INCLOCK PROC   FAR
;System time set to clock time
PUBLIC INCLOCK
        MOV   AX,BRDDATA
        MOV   DX,AX
;First run 'LOCATE'
;AX & DX for input/output
;AX, CX, DX, for DOS time & date write
;SI for building CX, DI for building DX
;BP is clock address.
;Only BX, SP, and segment registers not changed
        ;Any brds located?
        CMP   BRDS,0
        JNE   READ1               ;Yes
        CALL  BEEP                ;No, error
        RET
        ;Setup DDRB's & 'HLD' & 'RD'
READ1:  MOV   DX,BRD1
        ADD   DX,2
        MOV   AL,0B0H
        OUT   DX,AL
        ADD   DX,10H
        MOV   AL,0BFH
        OUT   DX,AL
        PUSHF
        CLI
        CALL  HLD_HI              ;'HLD'=hi
        MOV   AL,20H
        MOV   DX,BRD1
        OUT   DX,AL               ;'HLD' & 'RD' = hi
        ;Read time
        SUB   BP,BP
        MOV   CL,0FH
        CALL  RBYTE               ;Read seconds
        MOV   AH,AL
        SUB   AL,AL
        MOV   DI,AX
        MOV   CL,0FH
        CALL  RBYTE               ;Read minutes
        MOV   SI,AX
```

```
        MOV     CL,3H               ;Bytes 2 & 3 lo for 10's
        CALL    RBYTE               ;Read hours
        MOV     CX,SI
        MOV     CH,AL
        CALL    HLD_LO              ;'HLD' & 'RD' = lo
        POPF
        MOV     DX,DI
        MOV     AH,2DH
        INT     21H                 ;Set system time
        CMP     AL,0FFH             ;Error setting time?
        JNE     READ2               ;No
        CALL    BEEP                ;Yes
        JMP     READ4
READ2:  PUSHF
        CLI
        CALL    HLD_HI              ;'HLD' = hi
        MOV     AL,20H
        MOV     BX,BRD1
        OUT     DX,AL               ;'HLD' & 'RD' = hi
        ;Read date
        INC     BP                  ;Skip week, not used
        MOV     CL,03H              ;Bytes 2 & 3 lo for 10's
        CALL    RBYTE               ;Read day
        MOV     DI,AX
        MOV     CL,0FH
        CALL    RBYTE               ;Read month
        MOV     DX,DI
        MOV     DH,AL
        MOV     DI,DX
        MOV     CL,0FH
        CALL    RBYTE               ;Read year
        SUB     CH,CH
        MOV     CL,AL
        ADD     CX,1900             ;Add century
        CMP     CX,1980             ;<1980
        JAE     READ3               ;No
        ADD     CX,100              ;Yes, next century
READ3:  CALL    HLD_LO              ;'HLD' & 'RD' = lo
        POPF
        MOV     DX,DI
        MOV     AX,2BH
        INT     21H                 ;Set system date
        CMP     AL,0FFH             ;Error setting date?
        JNE     READ4               ;No
        CALL    BEEP                ;Yes
        ;Set DDRB for input on HLD line
READ4:  MOV     DX,BRD1
        ADD     DX,2
        MOV     AL,0CFH
        OUT     DX,AL
        RET
        INCLOCK ENDP
;*********************************************************
RBYTE   PROC    NEAR
        CALL    RNIB                ;Low nibble
        MOV     AH,AL
        CALL    RNIB                ;High nibble
        ;Bytes 2 & 3 low for 10's place, hr or day
        AND     AL,CL
        ;Convert BCD to binary
        MOV     CX,AX
        MOV     AH,10
        MUL     AH
        ADD     AL,CH
        RET
        RBYTE ENDP
RNIB    PROC    NEAR
        AND     AL,0FH              ;'RD' = lo 'HLD' = hi
        MOV     DX,BRD1
        XCHG    AX,BP
        ADD     DX,10H
        OUT     DX,AL               ;Addr to clock
        INC     AX
```

```
            XCHG    AX,BP
            ;Wait >6 us
            PUSH    AX
            MOV     AX,1
            CALL    WAIT
            POP     AX
            ;Read data
            SUB     DX,10H
            IN      AL,DX                  ;Data from clock
            AND     AL,0FH                 ;Top 4 bits 0
            RET
            RNIB    ENDP
;*****************************************************
OUTCLOCK    PROC    FAR
PUBLIC  OUTCLOCK
            MOV     AX,BRDDATA
            MOV     DX,AX
;      CLOCK SET
;Clock is set to system time
;first run 'LOCATE'
;AX & DX for input/output
;AX, CX, DX for DOS time & date read
;CX saved in SI, DX saved in DI
;BP is clock address.
;Only BX, SP, and segment registers not changed
            ;Any brds located?
            CMP     BRDS,0
            JNE     SET1                   ;Yes
            CALL    BEEP                   ;No, error
            RET
            ;Set DDRB's for outputs
SET1:       MOV     DX,BRD1                ;Brd bsar addr
            ADD     DX,2
            MOV     AL,0BFH
            OUT     DX,AL
            ADD     DX,10H
            OUT     DX,AL
            ;Read system time
            MOV     AH,2CH
            INT     21H                    ;Read time
            MOV     SI,CX                  ;Save it
            ;Set clock (sec, min, hrs)
            PUSHF
            CLI
            SUB     BP,BP                  ;clk addr for seconds
            CALL    HLD_HI                 ;'HLD' = high
            SUB     AX,AX                  ;Sec = 0
            CALL    WBYTE                  ;Set seconds
            MOV     AX,SI
            SUB     AH,AH
            CALL    WBYTE                  ;Set minutes
            MOV     AX,SI
            MOV     AL,AH
            MOV     AH,8H                  ;Bit 3 = 1 for 24 hr format
            CALL    WBYTE                  ;Set hour
            CALL    HLD_LO                 ;'HLD' & 'WR' = low
            POPF
            ;Read system date
            MOV     AH,2AH
            INT     21H                    ;Read date
            MOV     SI,CX                  ;Save it
            MOV     DI,CX
            ;Set clock (day, mo, year)
            PUSHF
            CLI
            INC     BP                     ;Skip week, not used
            CALL    HLD_HI                 ;'HLD' = hi
            ;Set leap year bit if <1 yr before Feb 29
            MOV     CX,SI
            MOV     DX,DI
            MOV     AX,DI
            SUB     AH,AH
            AND     CL,03H                 ;Bits 0 & 1
```

```
        JNZ     LPYR1           ;Not 0, not leap yr
        CMP     DH,2            ;Before Feb 29?
        JA      LPYR1           ;No
        MOV     AH,4H           ;Yes, set leap yr bit
LPYR1:  MOV     CX,SI
        INC     CX
        AND     CL,03H          ;Bits 0 & 1
        JNZ     LPYR2           ;Not 0, not yr prec. leap
        CMP     DH,3            ;Before March 1?
        JB      LPYR2           ;No
        MOV     AH,4H           ;Yes, set leap yr bit
        ;Now really set the clock
LPYR2:  CALL    WBYTE           ;Set Day
        MOV     AX,DI
        MOV     AL,AH
        SUB     AH,AH
        CALL    WBYTE           ;Set month
        MOV     AX,SI
        SUB     AX,1900         ;Remove century from yr
        CMP     AX,100          ;>2000?
        JB      YR
        SUB     AX,100          ;Yes, remove another centry
YR:     CALL    WBYTE           ;Set year
        CALL    HLD_LO          ;'HLD' & 'WR' = lo
        POPF
        ;Set DDRB for input on HLD line
        MOV     DX,BRD1
        ADD     DX,2
        MOV     AL,0CFH
        OUT     DX,AL
        RET
        OUTCLOCK ENDP
;*******************************************************
WBYTE   PROC    NEAR
;Write a byte to the clock (2 nibbles)
        ;1st convert binary to BCD
        MOV     CL,AH
        SUB     AH,AH
BIN_BCD: SUB    AL,10
        INC     AH
        CMP     AL,0
        JGE     BIN_BCD
        ADD     AL,10
        SUB     AH,1
        ;Now write the byte
        CALL    WNIB            ;Low nibble
        MOV     AL,AH
        OR      AL,CL           ;Set bits in hr or day
        CALL    WNIB            ;High nibble
        RET
        WBYTE ENDP
WNIB    PROC    NEAR
        AND     AL,0FH          ;'WR'=lo 'HLD'=hi
        MOV     DX,BRD1
        OUT     DX,AL           ;Data to clock
        XCHG    AX,BP
        ADD     DX,10H
        OUT     DX,AL           ;Addr to clock
        INC     AX              ;Increment address
        XCHG    AX,BP
        OR      AL,80H          ;Bit 7 hi
        SUB     DX,10H
        OUT     DX,AL           ;'WR'=hi
        AND     AL,7FH          ;Bit 7 lo
        OUT     DX,AL           ;'WR' = lo
        RET
        WNIB    ENDP
HLD_HI  PROC    NEAR
        MOV     DX,BRD1         ;Base addr
        MOV     AL,0            ;'HLD' high
        OUT     DX,AL
        ;Wait >150 us
        MOV     AX,0C0H
```

```
          CALL  WAIT
          RET
          HLD_HI ENDP
HLD_LO    PROC  NEAR
          MOV   DX,BRD1             ;Base addr
          MOV   AL,10H              ;'HLD' low
          OUT   DX,AL
          RET
          HLD_LO ENDP
;**************************************************
BEEP      PROC  NEAR                ;Beep the speaker
          ;Only AX and flags are altered.
          ;Only CS segment register used.
          PUSH  CX
          MOV   AL,10110110B        ;Timer 2, mode 3
          OUT   43H,AL
          MOV   AX,840              ;1K Hz tone
          OUT   42H,AL              ;to timer 2
          MOV   AL,AH
          OUT   42H,AL
          IN    AL,61H              ;Port 61 data
          MOV   AH,AL               ;Save it
          OR    AL,03H
          OUT   61H,AL              ;Turn on speaker
          MOV   CX,08FFFH           ;Loop count
TONE:     LOOP  TONE
          MOV   AL,AH               ;Recover port 61 data
          AND   AL,0FDH             ;Bit 1 low
          OUT   61H,AL              ;Turn off speaker
          POP   CX
          RET
          BEEP  ENDP
;************************************************************
WAIT      PROC  NEAR
          ;Wait time = AX * 840nS + approx 100 uS
          ;if interrupts are disabled.
          ;      AX = 32500 maximum.
          ;         = 1 minimum.
          ;Missed clock ticks counted in CLK_TICK
          ;(Clock ticks = CLK_TICK / 2).
          ;All registers preserved except flags.
          PUSH  AX
          IN    AL,61H              ;Port 61 data
          OR    AL,01H              ;Bit 0 hi
          AND   AL,0FDH             ;Bit 1 low
          OUT   61H,AL              ;Timer 2, gate on
          MOV   AL,10110000B        ;Timer 2, mode 0
          OUT   43H,AL
          POP   AX
          PUSH  AX                  ;Delay count
          OUT   42H,AL              ;LSB to timer 2
          MOV   AL,AH
          OUT   42H,AL              ;MSB to timer 2
W1:       IN    AL,62H
          TEST  AL,20H              ;Timer 2 terminal cnt high?
          JE    W1                  ;No, loop until high
          CALL  ROLL_OVER           ;Incr clock tick
          MOV   AL,10110110B        ;Timer 2, mode 3
          OUT   43H,AL
          POP   AX
          RET
          WAIT  ENDP
ROLL_OVER PROC  NEAR
          ;Increment CLK_TICK if timer 0 rolled over
          PUSH  BX
          MOV   BX,TIMR_CNT
          CALL  TIMR_READ           ;Read timer 0
          CMP   BX,TIMR_CNT         ;Roll_over?
          JB    RO1                 ;Yes
          NOP                       ;No
          NOP                       ;Same time
          NOP
          NOP
```

```
         NOP
         NOP
         JMP    RO2
RO1:     INC    CLK_TICK
RO2:     POP    BX
         RET
         ROLL_OVER ENDP
TIMR_READ PROC  NEAR
         ;Read timer 0 count & save in TIMR_CNT
         SUB    AL,AL
         OUT    43H,AL            ;Freeze timer 0
         NOP
         NOP
         IN     AL,40H            ;Low byte, timer 0
         MOV    AH,AL
         NOP
         IN     AL,40H            ;High byte, timer 0
         XCHG   AL,AH
         MOV    TIMR_CNT,AX       ;Save it
         RET
         TIMR_READ ENDP
;*************************************************************
PIA_SETUP PROC  NEAR
         ;Setup both 6522 DDRB for data direction
         ;1st 6522 ACR:
         ;T1 counts down with pulses on PB6
         MOV    SI,BRDCNT
         DEC    SI
         SHL    SI,1
         MOV    BP,BRD1[SI]       ;Base I/O address
         ;Setup ACR
         MOV    DX,BP
         ADD    DX,0BH            ;Offset 1st ACR=0BH
         IN     AL,DX
         OR     AL,20H
         AND    AL,0E1H
         OUT    DX,AL             ;1st ACR data
         ;Setup DDRB
         SUB    DX,09H            ;Offset 1st DDRB=2
         MOV    AL,0BFH
         OUT    DX,AL             ;1st DDRB data
         ADD    DX,10H            ;Offset to 2nd DDRB=12H
         OUT    DX,AL             ;2nd DDRB data
         RET
         PIA_SETUP ENDP
;**********************************************************************
READ     PROC   NEAR
         ;CALL with no. of cycles in BX.
         ;This adjusts for less than full scale reference.
         ;Normally BX=2. In INIT BX=8  for
         ;250mV range ans BX=16 for 500mV range.
         ;RET with count in BX
         SUB    CH,CH
         MOV    CL,FILDEL         ;Filter delay
         MOV    CLK_TICK,0
         PUSHF
         CLI                      ;Disable interrupts
         CALL   TIMR_READ
DLY0:    MOV    AX,15870          ;16.7mS/CALL
         CALL   WAIT              ;Filter settling
         LOOP   DLY0
         MOV    DX,BP             ;I/O base address
         ADD    DX,8              ;Address of T2
DLY5:    MOV    AX,0
         MOV    CX,128
         ;Set counter T2 to 0 & start counting
         OUT    DX,AX
         ;Wait
DLY6:    IN     AX,DX
         CMP    AX,0FFFFH         ;Look for 1st transition
         JE     DLY7              ;Found it
         LOOP   DLY6
         JMP    DLY9              ;Transition not found
```

```
DLY7:     SUB   CH,CH
          MOV   CL,RESOL
DLY8:     MOV   AX,1885            ;1.65mS/cycle
          CALL  WAIT
          LOOP  DLY8
          DEC   BX
          JNE   DLY7
DLY9:     IN    AX,DX              ;Read counter
          SUB   BX,BX
          SUB   BX,AX              ;Count in BX
          POPF
DLY10:    CMP   CLK_TICK,1         ;>1/2 clock tick missed?
          JBE   DLY11              ;No
          INT   8H                 ;Time_of_day interrupt to
          SUB   CLK_TICK,2         ;Catch up one clock tick.
          JMP   DLY10
DLY11:    RET
          READ  ENDP
;***************************************************************
READ_SETUP PROC NEAR
          ;Setup BX for READ cycles if in INIT
          ;BX=2 normally, BX=8 if 250mV range.
          ;BX=16 if 500mV range.
          MOV   BX,2
          SUB   AH,AH
          MOV   AL,CHANS
          INC   AL
          CMP   AX,TEMPCHAN        ;CJ sensor?
          JBE   RS2                ;Yes
          CMP   ZREFCNT,2          ;250mV range?
          JNE   RS1                ;No
          MOV   BX,8
RS1:      CMP   ZREFCNT,6          ;500mV range?
          JNE   RS2                ;No
          MOV   BX,16
RS2:      RET
          READ_SETUP ENDP
;*****************************************************************
RESET     PROC  FAR
PUBLIC    RESET
          PUSH   DS
          PUSH   SI
          PUSH   BP
          MOV   AX,BRDDATA
          MOV   DS,AX
          ;Reset analog cards so they can be found by 'LOCATE'.
          MOV   BX,-2
RESET1:   INC   BX
          INC   BX
          MOV   BP,BRD1[BX]        ;Brd address
          CMP   BP,0               ;Past last brd?
          JE    RESDONE            ;Yes, done
          CMP   BX,24              ;Past 12th brd?
          JA    RESDONE            ;Yes, done
          SUB   SI,SI
          SUB   AL,AL
RESET2:   MOV   DX,BP
          ADD   DL,LOC_TABLE[SI]   ;I/O address
          OUT   DX,AL              ;Reset it
          INC   SI
          CMP   SI,8
          JNE   RESET2
          JMP   RESET1
RESDONE:
          POP   BP
          POP   SI
          POP   DS
          RET
          RESET ENDP
;
;
SETRES          PROC FAR
        PUBLIC  SETRES
```

```
;
; Subroutine for setting the resolution of the A/D conversion
;
;        use as CALL SETRES(IRES)
;            where IRES is an INTEGER*2 variable
;
          LDS    SI,Es:PARM1[BX]
          MOV    AX,[SI]
          MOV    BX,BRDDATA
          MOV    Ds,BX
          MOV    RESOL,AL
          RET
SETRES           ENDP
;
SETFIL           PROC    FAR
        PUBLIC   SETFIL
;
; SUBROUTINE FOR SETTING FILTER DELAY
;
;        use as CALL SETFIL(IFIL)
;                where IFIL is an INTEGER*2 variable
;
          LDS    SI,Es:PARM1[BX]
          MOV    AX,[SI]
SETFIL10: MOV    BX,BRDDATA
          MOV    Ds,BX
          MOV    FILDEL,AL
          RET
SETFIL           ENDP
SETCHN           PROC    FAR
        PUBLIC   SETCHN
;
; SUBROUTINE FOR SETTING CHANNELS PER BOARD
;
;        use as CALL SETCHN(NCHAN)
;                where NCHAN is an INTEGER*2 variable
;
          LDS    SI,Es:PARM1[BX]
          MOV    AX,[SI]
SETCHN10: MOV    BX,BRDDATA
          MOV    Ds,BX
          MOV    CHANS,AL
          RET
SETCHN           ENDP
SETSTP           PROC    FAR
        PUBLIC   SETSTP
;
; SUBROUTINE FOR SETTING CHANNELS PER BOARD
;
;        use as CALL SETSTP(ISTEP)
;                where ISTEP is an INTEGER*2 variable
;
          LDS    SI,Es:PARM1[BX]
          MOV    AX,[SI]
          CMP    AL,00H
          JE     SETSTP10
          MOV    AL,01H
SETSTP10: MOV    BX,BRDDATA
          MOV    Ds,BX
          MOV    SSTEP,AL
          RET
SETSTP           ENDP
SETKEY           PROC    FAR
        PUBLIC   SETKEY
;
; SUBROUTINE FOR SETTING KEY VARIABLE
;
;        use as CALL SETKEY(IKEY)
;                where IKEY is an INTEGER*2 variable
;
          LDS    SI,Es:PARM1[BX]
          MOV    AX,[SI]
          CMP    AL,00H
```

```
           JE      SETKEY10
           MOV     AL,01H
SETKEY10:  MOV     BX,BRDDATA
           MOV     Ds,BX
           MOV     SKEY,AL
           RET
SETKEY             ENDP
SETNOZ             PROC     FAR
        PUBLIC   SETNOZ
;
; SUBROUTINE FOR SETTING NOZREF
;
;       use as CALL SETNOZ(INOZ)
;              where INOZ is an INTEGER*2 variable
;
           LDS     SI,Es:PARM1[BX]
           MOV     AX,[SI]
           CMP     AL,00H
           JE      SETNOZ10
           MOV     AL,01H
SETNOZ10:  MOV     BX,BRDDATA
           MOV     Ds,BX
           MOV     NOZREF,AL
           RET
SETNOZ             ENDP
SETBRD             PROC     FAR
        PUBLIC   SETBRD
;
; SUBROUTINE FOR SETTING NUMBER OF BOARDS
;
;       use as CALL SETBRD(NBRDS)
;              where NBRDS is an INTEGER*2 variable
;
           LDS     SI,Es:PARM1[BX]
           MOV     AX,[SI]
SETBRD10:  MOV     BX,BRDDATA
           MOV     Ds,BX
           MOV     BRDS,AL
           RET
SETBRD             ENDP
GETBRD             PROC     FAR
        PUBLIC   GETBRD
;
; SUBROUTINE FOR SETTING NUMBER OF BOARDS
;
;       use as CALL GETBRD(NBRDS)
;              where NBRDS is an INTEGER*2 variable
;
           MOV     AX,BRDDATA
           MOV     Ds,AX
           MOV     AH,00H
           MOV     AL,BRDS
           LDS     SI,Es:PARM1[BX]
           MOV     [SI],AX
           RET
GETBRD             ENDP
;
;
BRDADR             PROC     FAR
        PUBLIC   BRDADR
;
;
; Routine for returning board address
;
;    use as CALL BRDADR(IWORD,N)
;
;     where IWORD is the board address (INTEGER*2 variable)
;           N   is the board number (1 to 12) (INTEGER*2 variable)
;
           MOV     AX,BRDDATA
           LDS     SI,Es:PARM2[BX]
           PUSH    BX
           MOV     BX,[SI]
```

```
          MOV    Ds,AX
          DEC    BX
          SHL    BX,1
          MOV    AX,BRD1[BX]
          POP    BX
          LDS    SI,Es:PARM1[BX]
          MOV    [SI],AX
          RET
BRDADR           ENDP
SETADR           PROC     FAR
        PUBLIC   SETADR
;
;
; Routine for returning board address
;
;    use as CALL SETADR(IWORD,N)
;
;     where IWORD is the board address (INTEGER*2 variable)
;           N   is the board number (1 to 12) (INTEGER*2 variable)
;
          LDS    SI,Es:PARM1[BX]
          MOV    CX,[SI]
          LDS    SI,Es:PARM2[BX]
          MOV    BX,[SI]
          DEC    BX
          SHL    BX,1
          MOV    AX,BRDDATA
          MOV    DS,AX
          MOV    BRD1[BX],CX
          RET
SETADR           ENDP
;
;
SETRNG           PROC     FAR
        PUBLIC   SETRNG
;
; Routine for setting the range of the NCHAN th channel
;
;       use as CALL SETRNG(IRANGE,NCHAN)
;
;          where IRANGE is the range parameter (1 to 7)
;              declared as an INTEGER*2 variable
;                NCHAN is the channel number (INTEGER*2 variable)
;
          LDS    SI,Es:PARM1[BX]          ;Get Range #
          MOV    AX,[SI]
          PUSH   AX                       ;Save on Stack
          LDS    SI,Es:PARM2[BX]          ;Get Channel #
          MOV    CX,[SI]                  ;Channel #
          DEC    CX                       ;Channel # minus 1
          MOV    AX,BRDDATA
          MOV    Ds,AX
          MOV    AX,CX
          DIV    CHANS            ;AL = BOARD #, AH = CHANNEL # (minus 1)
          MOV    CX,AX            ;CL = BOARD #, CH = CHANNEL # (minus 1)
          MOV    AL,CHANS
          ADD    AL,2
          MUL    CL               ;AX = (CHANS+2)*BOARD #
          MOV    CL,CH
          MOV    CH,00H
          ADD    AX,CX
          MOV    SI,AX            ;Range value are stored starting at offset
                                  ;0 of Segment ANALDATA
          POP    AX               ;Retrieve range setting
          MOV    BX,ANALDATA
          MOV    Ds,BX
          MOV    [SI],AX
          RET
SETRNG           ENDP
;
ANALOG           PROC     FAR
;
;
```

```
;  Routine for retrieving analog data
;
;       use as CALL ANALOG(IDATA,NCHAN)
;               whwre IDATA is the data (INTEGER*2)
;                     NCHAN is the position of the data (INTEGER*2)
        PUBLIC  ANALOG
          LDS   SI,Es:PARM2[BX]
          MOV   AX,[SI]
          MOV   CX,BRDDATA
          MOV   Ds,CX
          MOV   SI,DADDR
          MOV   CX,ANALDATA
          MOV   DS,CX
          DEC   AX
          SHL   AX,1
          ADD   SI,AX
          MOV   AX,[SI]
          LDS   SI,Es:PARM1[BX]
          MOV   [SI],AX
          RET
ANALOG          ENDP
;
P@KEY       ENDS
          END
                        TITLE   FORGRPHX
;
;  IBM PROFESSIONAL FORTRAN CALLABLE SUBROUTINE FOR THE HERCULES GRAPHICS
;  CARD - GRAPHX SUBROUTINES
;
PARMBLK         STRUC
PARM1           DD      ?
PARM2           DD      ?
PARM3           DD      ?
PARM4           DD      ?
PARM5           DD      ?
PARM6           DD      ?
PARMBLK         ENDS
P@GRAPHX        SEGMENT PARA 'CODE'
                ASSUME CS:P@GRAPHX
CIRC    PROC    FAR
;
;  SUBROUTINE FOR DRAWING A CIRCLE OF RADIUS R AT LOCATION X,Y
;
;       USE AS CALL CIRC(X,Y,R)
;
;               WHERE X,Y,R ARE INTEGER*2 VARIABLES
;
PUBLIC  CIRC
                LDS     SI,ES:PARM1[BX]
                MOV     AX,[SI]
                MOV     DI,AX
                LDS     SI,ES:PARM2[BX]
                MOV     AX,[SI]
                MOV     BP,AX
                LDS     SI,ES:PARM3[BX]
                MOV     AX,[SI]
                MOV     BX,AX
                MOV     AH,4DH
                INT     10H
                RET
CIRC            ENDP
CLRSCR          PROC    FAR
;
;  SUBROUTINE FOR CLEARING THE SCREEN
;
;       USE AS CALL CLRSCR
;
;               NO ARGUMENTS
;
PUBLIC          CLRSCR
                MOV     AH,42H
                INT     10H
                RET
```

```
CLRSCR          ENDP
DISP            PROC    FAR
;
;       SUBROUTINE FOR SETTING DISPLAY PAGE
;
;       USE AS  CALL DISP(IPAGE)
;
;               WHERE IPAGE IS INTEGER*2 VARIABLE = 0 OR 1
;
PUBLIC          DISP
                LDS     SI,ES:PARM1[BX]
                MOV     AX,[SI]
                MOV     AH,45H
                INT     10H
                RET
DISP            ENDP
GMODE           PROC    FAR
;
;       SUBROUTINE FOR SETTING GRAPHICS MODE
;
;               USE AS  CALL GMODE
;
;                       NO ARGUMENTS
;
PUBLIC          GMODE
                MOV     AH,40H
                INT     10H
                RET
GMODE           ENDP
GPAGE           PROC    FAR
;
;       SUBROUTINE FOR SETTING GRAPHICS PAGE
;
;               USE AS  CALL GPAGE(IPAGE)
;
;                       WHERE IPAGE IS INTEGER*2 VARIABLE = 0 OR 1
;
PUBLIC          GPAGE
                LDS     SI,ES:PARM1[BX]
                MOV     AX,[SI]
                MOV     AH,43H
                INT     10H
                RET
GPAGE           ENDP
TMODE           PROC    FAR
;
;       SUBROUTINE FOR SETTING TEXT MODE
;
;               USE AS  CALL TMODE
;
;               NO ARGUMENTS
;
PUBLIC          TMODE
                MOV     AH,41H
                INT     10H
                RET
TMODE           ENDP
ARC             PROC    FAR
;
;       SUBROUTINE FOR DRAWING QUARTER ARC OF RADIUS R AT LOCATION X,Y
;
;               USE AS  CALL ARC(X,Y,R,QUAD)
;
;               WHERE X,Y,R,QUAD ARE INTEGER*2 VARIABLES
;
;               AND QUAD = 1,2,3 OR 4
;
PUBLIC          ARC
                LDS     SI,ES:PARM1[BX]
                MOV     AX,[SI]
                MOV     DI,AX
                LDS     SI,ES:PARM2[BX]
                MOV     AX,[SI]
```

```
                MOV     BP,AX
                LDS     SI,ES:PARM3[BX]
                MOV     AX,[SI]
                PUSH    AX
                LDS     SI,ES:PARM4[BX]
                MOV     AX,[SI]
                MOV     AH,4CH
                POP     BX
                INT     10H
                RET
ARC             ENDP
BLKFIL          PROC    FAR
;
;       SUBROUTINE FOR FILLING RECTANGULAR REGION WHOSE LEFT
;       CORNER IS LOCATED AT X,Y AND WITH A WIDTH & LENGTH GIVEN
;
;               USE AS CALL BLKFIL(X,Y,WIDTH,LENGTH)
;
;               WHERE X,Y,WIDTH & LENGTH ARE INTEGER*2 VARIABLES
;
PUBLIC          BLKFIL
                LDS     SI,ES:PARM1[BX]
                MOV     AX,[SI]
                MOV     DI,AX
                LDS     SI,ES:PARM2[BX]
                MOV     AX,[SI]
                MOV     BP,AX
                LDS     SI,ES:PARM3[BX]
                MOV     AX,[SI]
                MOV     CX,AX
                LDS     SI,ES:PARM4[BX]
                MOV     AX,[SI]
                MOV     BX,AX
                MOV     AH,4AH
                INT     10H
                RET
BLKFIL          ENDP
DLINE           PROC    FAR
;
;       SUBROUTINE FOR DRAWING A LINE FROM THE CURRENT POSITION TO THE
;       POSITION GIVEN BY X,Y
;
;               USE AS CALL DLINE(X,Y)
;
;               WHERE X,Y ARE INTERGER*2 VARIABLES
;
PUBLIC          DLINE
                LDS     SI,ES:PARM1[BX]
                MOV     AX,[SI]
                MOV     DI,AX
                LDS     SI,ES:PARM2[BX]
                MOV     AX,[SI]
                MOV     BP,AX
                MOV     AH,49H
                INT     10H
                RET
DLINE           ENDP
FILL            PROC    FAR
;
;       SUBROUTINE FOR FILLING  A CONVEX POLYGON WITH A SEED X,Y
;
;               USE AS  CALL FILL(X,Y)
;
;               WHERE X,Y ARE INTEGER*2 VARIABLES
;
PUBLIC          FILL
                LDS     SI,ES:PARM1[BX]
                MOV     AX,[SI]
                MOV     DI,AX
                LDS     SI,ES:PARM2[BX]
                MOV     AX,[SI]
                MOV     BP,AX
                MOV     AH,4EH
```

```
                INT     10H
                RET
FILL            ENDP
GETPT           PROC    FAR
;
;       SUBROUTINE FOR GETTING THE INTENSITY AT THE POINT X,Y
;
;               USE AS  CALL GETPT(X,Y,INTEN)
;
;               WHERE X,Y,INTEN ARE INTEGER*2 VARIABLES
;
PUBLIC          GETPT
                LDS     SI,ES:PARM1[BX]
                MOV     AX,[SI]
                MOV     DI,AX
                LDS     SI,ES:PARM2[BX]
                MOV     AX,[SI]
                MOV     BP,AX
                MOV     AH,47H
                INT     10H
                XOR     AH,AH
                LDS     SI,ES:PARM3[BX]
                MOV     [SI],AX
                RET
GETPT           ENDP
LEVEL           PROC    FAR
;
;       SUBROUTINE FOR SETTING INTENSITY LEVEL
;
;               USE AS CALL LEVEL(INTEN)
;
;               WHERE INTEN IS INTEGER*2 VARIABLE = 0,1,2
;
;               LEVEL = 0 CAUSES BLACK POINT
;               LEVEL = 1 CAUSES WHITE POINT
;               LEVEL = 2 XORes THE SCREEN
;
PUBLIC          LEVEL
                LDS     SI,ES:PARM1[BX]
                MOV     AX,[SI]
                MOV     AH,44H
                INT     10H
                RET
LEVEL           ENDP
PUTPT           PROC    FAR
;
;       SUBROUTINE FOR MOVING THE IMAGINARY CURSOR TO LOCATION X,Y
;
;               USE AS CALL PUTPT(X,Y)
;
;               WHERE X,Y ARE INTEGER*2 VARIABLES
;
PUBLIC          PUTPT
                LDS     SI,ES:PARM1[BX]
                MOV     AX,[SI]
                MOV     DI,AX
                LDS     SI,ES:PARM2[BX]
                MOV     AX,[SI]
                MOV     BP,AX
                MOV     AH,48H
                INT     10H
                RET
PUTPT           ENDP
PLOT            PROC    FAR
;
;       SUBROUTINE SETTING, CLEARING OR XORing PIXEL AT LOCATION X,Y
;
;       USE AS CALL PLOT(X,Y)
;
;       WHERE X,Y ARE INTEGER*2 VARIABLES
;
PUBLIC          PLOT
                LDS     SI,ES:PARM1[BX]
```

```
                MOV     AX,[SI]
                MOV     DI,AX
                LDS     SI,ES:PARM2[BX]
                MOV     AX,[SI]
                MOV     BP,AX
                MOV     AH,46H
                INT     10H
                RET
PLOT            ENDP
TEXT    PROC    FAR
;
;       SUBROUTINE FOR WRITING CHARACTER AT LOCATION X,Y
;
;       USE AS  CALL TEXT(X,Y,CHAR)
;
;       WHERE X,Y, CHAR ARE INTEGER*2 VARIABLES
;
;
PUBLIC  TEXT
                LDS     SI,ES:PARM1[BX]
                MOV     AX,[SI]
                MOV     DI,AX
                LDS     SI,ES:PARM2[BX]
                MOV     AX,[SI]
                MOV     BP,AX
                LDS     SI,ES:PARM3[BX]
                MOV     AX,[SI]
                MOV     AH,4BH
                INT     10H
                RET
TEXT            ENDP
PRTCHAR         PROC    FAR
;
; FORTRAN CALLABLE SUBROUTINE FOR DRAWING A CHARACTER
;
; USE AS CALL PRTCHAR(X,Y,N,NX,NY)
;
;       WHERE
;               X,Y IS THE LOCATION OF THE CHARACTER
;               N   IS THE CHARACTER NUMBER
;               NX,NY IS THE MAGNIFICANTION IN THE X & Y DIRECTION
;
PUBLIC  PRTCHAR
                LDS     SI,ES:PARM1[BX]
                MOV     AX,[SI]
                MOV     DI,AX
                LDS     SI,ES:PARM2[BX]
                MOV     AX,[SI]
                MOV     BP,AX
                LDS     SI,ES:PARM3[BX]
                MOV     AX,[SI]
                MOV     CL,3
                SAL     AX,CL
                PUSH    AX
                MOV     AX,0F000H
                MOV     DS,AX
                POP     AX
                ADD     AX,0FA6EH
                MOV     SI,AX
                MOV     Cx,8
PRTCHAR_10:     PUSH    Cx
                PUSH    SI
                PUSH    DS
                PUSH    DI
                PUSH    DX
                MOV     AL,[SI]         ;GET CHARACTER
                PUSH    AX
                LDS     SI,ES:PARM5[BX]
                MOV     AX,[SI]
                MOV     CX,AX
                POP     AX
PRTCHAR_15:     PUSH    CX
                MOV     CX,8
```

```
                PUSH    AX
                PUSH    DI
PRTCHAR_20:     PUSH    CX
                ROL     Al,1
                PUSH    AX
                MOV     AL,1            ;SET INTENSITY, WHITE DOT
                JC      PRTCHAR_30
                MOV     AL,0            ;SET INTENSITY, BLACK DOT
PRTCHAR_30:     MOV     AH,44H          ;SET INTENSITY
                PUSH    ES
                PUSH    BX
                INT     10H
                POP     BX
                POP     ES
                PUSH    AX
                LDS     SI,ES:PARM4[BX]
                MOV     AX,[SI]
                MOV     CX,AX
                POP     AX
PRTCHAR_35:     PUSH    CX
                MOV     AH,46H          ;PLOT DOT
                PUSH    ES
                PUSH    BX
                INT     10H
                POP     BX
                POP     ES
                INC     DI              ;INCREASE X CO-ORDINATE
                POP     CX
                LOOP    PRTCHAR_35      ;LOOP X EXPAND
                POP     AX
                POP     CX
                LOOP    PRTCHAR_20      ;LOOP ON WIDTH OF CHARACTER
                POP     DI
                INC     BP              ;INCREASE Y CO-ORDINATE
                POP     AX
                POP     CX
                LOOP    PRTCHAR_15
                POP     DX
                POP     DI
                POP     DS
                POP     SI
                INC     SI
                POP     CX
                LOOP    PRTCHAR_10
                AND     AX,AX
                Ret
PRTCHAR         ENDP
Subttl  Subroutine CURSOR

CURSOR          Proc Far
Public  CURSOR

;
;       Fortran callable subroutine for positioning cursor
;       and clearing screen
;
;       Use as CALL CURSOR(COL,ROW)
;
;               where
;                       COL is the column number (1 to 80)
;                       ROW is the row number (1 to 25)
;
;               Declare COL and ROW as INTEGER*2 variables
;
;               If ROW and COL are both 0, the screen is erased the
;               the cursor returned to the home position.
;
                Lds     Si,Es:PARM1[Bx]
                Mov     Ax,[Si]
                Mov     Dl,Al           ;Column setup for INT 10H
                Lds     Si,Es:PARM2[Bx]
                Mov     Ax,[Si]
                Mov     Dh,Al           ;Row setup for INT 10H
```

```
                Mov     Ah,15             ;Commond number for retrieve state
                Int     10H
                Cmp     Dh,00H
                Jz      CURSOR_FCT        ;If ROW is zero, its a function
                Dec     Dh
                Dec     Dl
                Mov     Ah,2              ;Position cursor call
                Int     10H
                Jmp     CURSOR_RET
CURSOR_FCT:     Cmp     Dl,00H
                Mov     Ch,0              ;Use Scroll Active Page Up Routine
                Mov     Cl,0              ;to clear screen
                Mov     Dh,24
                Mov     Dl,79
                Mov     Bh,07H
                Mov     Al,0
                Mov     Ah,6
                Int     10H
                Mov     Ah,15
                Int     10H
                Mov     Dh,00h
                Mov     Dl,00h
                Mov     Ah,2
                Int     10H
CURSOR_RET:     Ret
CURSOR          Endp

Subttl Subroutine RDCUR

; Call as RDCUR(COL,ROW)
;
RDCUR   Proc Far
Public RDCUR
                Mov     Ah,15             ;Get Current Video State
                Int     10H
                Mov     Ah,3
                Int     10H
                Inc     Dl
                Inc     Dh
                Lds     Si,Es:PARM1[Bx]
                Mov     Al,Dl
                Mov     Ah,00H
                Mov     [Si],Ax
                Lds     Si,Es:PARM2[Bx]
                Mov     Al,Dh
                Mov     [Si],Ax
                Ret
RDCUR           Endp

RCHAR   Proc    Far
Public  RCHAR
;
; Fortran Callable Subroutine for reading the character at cuurent
; cursor position
;
; Use as CALL RCHAR(NCHAR)
;               where NCHAR IS AN INTEGER
;
                Mov     Ah,15
                Int     10H
                Mov     Ah,8
                Int     10H
                Mov     Ah,00H
                Lds     Si,Es:PARM1[Bx]
                Mov     [Si],Ax
                MOV     Al,00H
                Inc     Si
                Inc     Si
                Mov     [Si],Ax
                Ret
RCHAR           Endp

SCRLUP  Proc    Far
```

```
Public  SCRLUP
;
; Fortran Callable Subroutine for scrolling up window
;
; Use as          CALL SCRLUP(IWRLF,IWCLF,IWRRT,IWCRT)
;
;                 where  (IWRLF,IWCLF) ARE ROW & COL OF UPPER LEFT CORNER
;                        (IWRRT,IWCRT) ARE ROW & COL OF LOWER RIGHT CORNER
;
;
                  Lds     Si,ES:PARM1[Bx]
                  Mov     Ch,[Si]
                  Lds     Si,Es:PARM2[Bx]
                  Mov     Cl,[Si]
                  Lds     Si,Es:PARM3[Bx]
                  Mov     Dh,[Si]
                  Lds     Si,Es:PARM4[Bx]
                  Mov     Dl,[Si]
                  Dec     Ch
                  Dec     Cl
                  Dec     Dh
                  Dec     Dl
                  Mov     Al,1
                  Mov     Ah,6
                  Mov     Bh,07h
                  Int     10h
                  Ret
SCRLUP            Endp

SCRLDN  Proc      Far
Public  SCRLDN
;
; Fortran Callable Subroutine for scrolling down window
;
; Use as          CALL SCRLDN(IWRLF,IWCLF,IWRRT,IWCRT)
;
;                 where  (IWRLF,IWCLF) ARE ROW & COL OF UPPER LEFT CORNER
;                        (IWRRT,IWCRT) ARE ROW & COL OF LOWER RIGHT CORNER
;
;
                  Lds     Si,ES:PARM1[Bx]
                  Mov     Ch,[Si]
                  Lds     Si,Es:PARM2[Bx]
                  Mov     Cl,[Si]
                  Lds     Si,Es:PARM3[Bx]
                  Mov     Dh,[Si]
                  Lds     Si,Es:PARM4[Bx]
                  Mov     Dl,[Si]
                  Dec     Ch
                  Dec     Cl
                  Dec     Dh
                  Dec     Dl
                  Mov     Al,1
                  Mov     Ah,7
                  Mov     Bh,07h
                  Int     10h
                  Ret
SCRLDN            Endp


CONSOL  Proc      Far
Public  CONSOL
;
; FORTRAN CALLABLE SUBROUTINE TO TEST FOR TYPED CHARACTER
; FROM KEYBOARD
;
;                 CALL CONSOL(IBYTE)
;                         WHERE IBYTE IS INTEGER*2
;                         IBYTE= 0 MEANS NO CHARACTER TYPED
;                         IBYTE= -1 MEANS TYPED CHARACTER
;
                  Lds     SI,Es:[Bx]
                  MOV     AH,0BH
```

```
                INT     21H
                MOV     AH,AL
                MOV     [SI],AX
                Ret
CONSOL          Endp

Subttl Subroutine KEYBD(ICHAR)
;
; Fortran Callable Subroutine for Reading Key board
;
KEYBD Proc FAR
Public  KEYBD
                Lds     Si,Es:PARM1[Bx]
                Mov     Ah,7
                Int     21H
                Mov     Ah,00H
                Mov     [SI],Ax
                Ret
KEYBD           Endp

Subttl Subroutine CRT(ICHAR)

;
; Fortran Callable Subroutine for Outputing Character to Screen
;
CRT     Proc    Far
Public  CRT
                Lds     Si,Es:PARM1[Bx]
                Mov     Dl,[Si]
                Mov     Ah,02H
                Int     21H
                Ret
CRT             Endp


Subttl  SETUNL(COL,ROW)
;
; Subroutine for setting underline attribute at COL,ROW
;
SETUNL  Proc    Far
Public  SETUNL
                Lds     Si,Es:PARM1[Bx]
                Mov     Dl,[Si]         ;Column
                Dec     Dl
                Lds     Si,Es:PARM2[Bx]
                Mov     Dh,[Si]         ;Row
                Dec     Dh
                Mov     Ah,15           ;Get Current Video Page
                Int     10H
                Mov     Ah,2            ;Set Cursor position
                Int     10H
                Mov     Ah,8            ;Read Current Character
                Int     10H
                Mov     Ah,9            ;Set atribute
                Mov     Cx,1
                Mov     Bl,01H          ;Underline, no blink
                Int     10H
                Ret
SETUNL          Endp

Subttl  NORVID(COL,ROW)
;
; Subroutine for setting normal video attribute at COL,ROW
;
NORVID  Proc    Far
Public  NORVID
                Lds     Si,Es:PARM1[Bx]
                Mov     Dl,[Si]         ;Column
                Dec     Dl
                Lds     Si,Es:Parm2[Bx]
                Mov     Dh,[Si]         ;Row
                Dec     Dh
                Mov     Ah,15           ;Get Current Video Page
```

```
                Int     10H
                Mov     Ah,2            ;Set Cursor position
                Int     10H
                Mov     Ah,8            ;Read Current Character
                Int     10H
                Mov     Ah,9            ;Set atribute
                Mov     Cx,1
                Mov     Bl,07H          ;Normal Video, no blink
                Int     10H
                Ret
NORVID          Endp

Subttl  NORBLK(COL,ROW)
;
; Subroutine for setting normal blinking attribute at COL,ROW
;
NORBLK  Proc    Far
Public  NORBLK
                Lds     Si,Es:PARM1[Bx]
                Mov     Dl,[Si]         ;Column
                Dec     Dl
                Lds     Si,Es:PARM2[Bx]
                Mov     Dh,[Si]         ;Row
                Dec     Dh
                Mov     Ah,15           ;Get Current Video Page
                Int     10H
                Mov     Ah,2            ;Set Cursor position
                Int     10H
                Mov     Ah,8            ;Read Current Character
                Int     10H
                Mov     Ah,9            ;Set atribute
                Mov     Cx,1
                Mov     Bl,87H          ;Normal Video, blink
                Int     10H
                Ret
NORBLK          Endp
Subttl  REVVID(COL,ROW)
;
; Subroutine for setting reverse video no blinking attribute at COL,ROW
;
REVVID  Proc    Far
Public  REVVID
                Lds     Si,Es:PARM1[Bx]
                Mov     Dl,[Si]         ;Column
                Dec     Dl
                Lds     Si,Es:PARM2[Bx]
                Mov     Dh,[Si]         ;Row
                Dec     Dh
                Mov     Ah,15           ;Get Current Video Page
                Int     10H
                Mov     Ah,2            ;Set Cursor position
                Int     10H
                Mov     Ah,8            ;Read Current Character
                Int     10H
                Mov     Ah,9            ;Set atribute
                Mov     Cx,1
                Mov     Bl,70H          ;Reverse video, no blink
                Int     10H
                Ret
REVVID          Endp
Subttl  REVBLK(COL,ROW)
;
; Subroutine for setting reverse video, blinking attribute at COL,ROW
;
REVBLK  Proc    Far
Public  REVBLK
                Lds     Si,Es:PARM1[Bx]
                Mov     Dl,[Si]         ;Column
                Dec     Dl
                Lds     Si,Es:PARM2[Bx]
                Mov     Dh,[Si]         ;Row
                Dec     Dh
                Mov     Ah,15           ;Get Current Video Page
```

```
                Int     10H
                Mov     Ah,2            ;Set Cursor position
                Int     10H
                Mov     Ah,8            ;Read Current Character
                Int     10H
                Mov     Ah,9            ;Set atribute
                Mov     Cx,1
                Mov     Bl,0F0H         ;Reverse video,  blink
                Int     10H
                Ret
REVBLK          Endp

Subttl  BLKUNL(COL,ROW)
;
; Subroutine for setting underline, blinking attribute at COL,ROW
;
BLKUNL  Proc    Far
Public  BLKUNL
                Lds     Si,Es:PARM1[Bx]
                Mov     Dl,[Si]         ;Column
                Dec     Dl
                Lds     Si,Es:PARM2[Bx]
                Mov     Dh,[Si]         ;Row
                Dec     Dh
                Mov     Ah,15           ;Get Current Video Page
                Int     10H
                Mov     Ah,2            ;Set Cursor position
                Int     10H
                Mov     Ah,8            ;Read Current Character
                Int     10H
                Mov     Ah,9            ;Set atribute
                Mov     Cx,1
                Mov     Bl,81H          ;Underline, Blink
                Int     10H
                Ret
BLKUNL          Endp
Subttl  Subroutine PRT

PRT     Proc    Far

Public  PRT

;
;       Fortran callabel subroutine for printing a string
;
;
;       use as CALL PRT(STRING)
;       where STRING must end in '$'
;
                Lds     Si,Es:PARM1[Bx]
                Inc     Si
                Inc     Si
                Mov     Ax,[Si]
                Mov     Dx,Ax
                Inc     Si
                Inc     Si
                Mov     Ax,[Si]
                MoV     Ds,Ax
                Mov     Ah,09h          ;Print string function call
                Int     21H
                Ret
PRT             Endp

P0GRAPHX        ENDS
                END
```

NBS-114A (REV. 2-8C)

U.S. DEPT. OF COMM. BIBLIOGRAPHIC DATA SHEET (See instructions)	1. PUBLICATION OR REPORT NO. NISTIR 88-4009	2. Performing Organ. Report No.	3. Publication Date DECEMBER 1988

4. TITLE AND SUBTITLE

The Evaluation of Thermal Probe Method for Estimating the Heat Loss from Underground Heat Distribution Systems

5. AUTHOR(S)

Jin B. Fang and Richard A. Grot

6. PERFORMING ORGANIZATION (If joint or other than NBS, see instructions)

NATIONAL BUREAU OF STANDARDS
DEPARTMENT OF COMMERCE
WASHINGTON, D.C. 20234

7. Contract/Grant No.

8. Type of Report & Period Covered

9. SPONSORING ORGANIZATION NAME AND COMPLETE ADDRESS (Street, City, State, ZIP)

U.S. Department of Energy
Building Services Division
Washington, DC 20585

10. SUPPLEMENTARY NOTES

☐ Document describes a computer program; SF-185, FIPS Software Summary, is attached.

11. ABSTRACT (A 200-word or less factual summary of most significant information. If document includes a significant bibliography or literature survey, mention it here)

An automated, microcomputer controlled instrumentation system developed for in-situ measurements of the earth temperatures and soil thermal conductivities at different depths and for calculating the heat losses from the underground district heating pipes is described. Step-by-step use and operation procedures of the developed heat loss measuring system and computer software package are presented. The heat loss rates and locations of underground pipes are calculated from the measured values of soil thermal conductivity and the earth temperatures around the pipes using the non-linear least squares method. The thermal probe technique was used to estimate the heat loss rates and the depths of buried steam supply and condensate return pipes installed at the James Madison University, Harrisonburg, Virginia.

12. KEY WORDS (Six to twelve entries; alphabetical order; capitalize only proper names; and separate key words by semicolons)

computer software; district heating and cooling; earth temperature; heat loss; instrumentation system; nonlinear least squares fitting; soil; steel pipe; thermal conductivity; underground heat distribution system.

13. AVAILABILITY

[X] Unlimited

☐ For Official Distribution. Do Not Release to NTIS

☐ Order From Superintendent of Documents, U.S. Government Printing Office, Washington, D.C. 20402.

[X] Order From National Technical Information Service (NTIS), Springfield, VA. 22161

14. NO. OF PRINTED PAGES

125

15. Price

$18.95