ORNL/TM-11607

**OAK RIDGE
NATIONAL
LABORATORY**

MARTIN MARIETTA

**Air Pollution Effects Field
Research Facility: 3. UV-B
Exposure and Monitoring System**

J. A. McEvers
M. S. Hileman
N. T. Edwards

**MANAGED BY
MARTIN MARIETTA ENERGY SYSTEMS, INC.
FOR THE UNITED STATES
DEPARTMENT OF ENERGY**

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

This report has been reproduced directly from the best available copy.

Available to DOE and DOE contractors from the Office of Scientific and Technical Information, P.O. Box 62, Oak Ridge, TN 37831; prices available from (615) 576-8401, FTS 626-8401.

Available to the public from the National Technical Information Service, U.S. Department of Commerce, 5285 Port Royal Rd., Springfield, VA 22161.

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

Instrumentation and Controls Division

**AIR POLLUTION EFFECTS FIELD RESEARCH FACILITY:
3. UV-B EXPOSURE AND MONITORING SYSTEM**

J. A. McEvers
M. S. Hileman
N. T. Edwards*

*Environmental Sciences Division, Oak Ridge National Laboratory.

Date Published—March 1993

Prepared by the
OAK RIDGE NATIONAL LABORATORY
Oak Ridge, Tennessee 37831-6285
managed by
MARTIN MARIETTA ENERGY SYSTEMS, INC.
for the
U. S. DEPARTMENT OF ENERGY
under contract DE-AC05-84OR21400

MASTER

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

CONTENTS

LIST OF FIGURES	v
ACKNOWLEDGMENTS	vi
ABSTRACT	vii
1. SCOPE	1
2. SYSTEM OVERVIEW	2
3. HAZARDS AND PRECAUTIONS	5
4. DESIGN BASIS	6
5. UV-B GENERATION AND EXPOSURE SUBSYSTEM	7
5.1 ULTRAVIOLET-B GENERATION	7
5.1.1 Ultraviolet Lamps	7
5.1.2 Filters	10
5.2 EXPOSURE ASSEMBLY	10
6. POWER SUPPLY SUBSYSTEM	13
6.1 LAMP FILAMENT POWER SUPPLY/CONTROL	13
6.2 LAMP ARC POWER SUPPLY/CONTROL	15
7. SENSOR SUBSYSTEM	16
7.1 ULTRAVIOLET SENSORS	16
8. MONITORING AND CONTROL SUBSYSTEM	20
8.1 COMPUTER CONFIGURATION	20
8.2 SENSOR SIGNAL INPUT	20
8.3 CONTROL SIGNAL OUTPUT	20
8.4 DATA STORAGE	21
8.5 SYSTEM POWER CONDITIONING	21
8.6 SYSTEM SOFTWARE	21
8.6.1 Overall System Display	21
9. DATA ANALYSIS AND CONTROL EVALUATION	23
10. SUMMARY	27
11. CONCLUSIONS AND RECOMMENDATIONS	28
12. REFERENCES	29
13. BIBLIOGRAPHY	29

APPENDIX A: SYSTEM PROGRAM SOFTWARE LISTINGS	31
APPENDIX B: SYSTEM PROCESS DATA BASE FILE LISTING	109
APPENDIX C: SYSTEM DISPLAY SCREEN CONTRCL FILES	113

LIST OF FIGURES

Figure	Page
1. Outside portion of UV-B exposure facility	3
2. Block diagram of the UV-B exposure and monitoring system	4
3. Spectral distribution of UVB-313 lamps, FS-40 lamps, and sunlight	8
4. Ultraviolet lamp mounting detail	9
5. Spectral transmission of cellulose triacetate and mylar films	11
6. Ultraviolet lamp support rack detail	12
7. Power supply and monitoring/control subsystem cabinets	14
8. UVX meter absolute response	18
9. Output of UVB-313 lamp filtered with CA film	24
10. Measurement of UV-B under CA filter lamp bank and ambient UV-B	25
11. Percentage of the difference between curves in Fig. 10	26

ACKNOWLEDGMENTS

The authors acknowledge the assistance provided by G. E. Taylor of the Oak Ridge National Laboratory Environmental Sciences Division (currently with the Desert Research Institute, Reno, Nevada) for allowing us to participate in the proposal preparation for this work and provide the overall UV-B Exposure and Monitoring Facility.

ABSTRACT

The Oak Ridge National Laboratory Outdoor Ultraviolet-B (UV-B) Exposure and Monitoring Facility was developed in 1989 to provide well-controlled and -monitored exposure of specific terrestrial plant species to elevated levels of ultraviolet (UV) radiation. The introduction of various anthropogenic agents into the earth's stratosphere has resulted in a decrease in the volume of ozone (O_3) present there. The decrease in O_3 has resulted in an increase in the level of UV radiation reaching the earth's surface. Of particular interest is the level of UV-B, because it has the most detrimental effect on living tissue. A thorough understanding of the effects of elevated levels of UV-B on living tissue is critical to the formulation of economic policy regarding production of such agents and alternative strategies. The UV region of interest is referred to as UV-B and corresponds to radiation with a wavelength of 290 to 320 nm. Design, operation, and performance of the automated generation, exposure, and monitoring system are described. The system has proved to be reliable and easy to maintain and operate, and it provides significant flexibility in exposure programs. The system software is described, and detailed listings are provided. The ability to expose plants to controlled set point percentages of UV-B above the ambient level was developed.

1. SCOPE

The Oak Ridge National Laboratory (ORNL) Outdoor Ultraviolet-B (UV-B) Exposure and Monitoring Facility was developed to support research relating to the effects of projected increases in ambient UV-B radiation caused by stratospheric ozone (O_3) depletion. The introduction of anthropogenic agents such as chlorinated fluorocarbons (CFCs) into the earth's stratosphere has resulted in a decrease in the volume of O_3 present. This stratospheric layer of O_3 is responsible for the attenuation of ultraviolet (UV) radiation reaching the earth's surface from space. A decrease in the effective volume of the stratospheric O_3 layer results in an increase in the level of UV radiation reaching the earth's surface. This reciprocal effect is called the radiation amplification factor (RAF). The transport delay associated with the release of agents at the earth's surface and their ultimate arrival in the stratosphere, with the eventual detrimental effects, requires that a better understanding of the long-term effects of increased levels of surface UV radiation be obtained. The large volume of agents released in past years will not reach the stratosphere for several years and, once there, will remain there ~100 years. The ability to forecast the effects of the increased levels of UV-B and formulate appropriate actions was the motivation for the development of the ORNL Outdoor UV-B Exposure and Monitoring Facility. The facility allows the introduction of ambient UV-B radiation and adds radiation provided and controlled by the UV-B exposure and monitoring system. This system, the key component in the facility, is responsible for UV-B generation, monitoring, storage, and exposure condition display.

This report describes the design basis, overall design, operational methodology, and performance parameters for the system. It does not attempt to discuss the results of specific exposures; rather, these discussions are left to the particular reports produced from specific exposure programs.

2. SYSTEM OVERVIEW

The UV-B exposure and monitoring system consists of four major components: (1) UV-B generation, (2) UV-B irradiance exposure measurement, (3) ambient UV-B irradiance measurement, and (4) percentage above ambient UV-B irradiance control. Figure 1 is a photograph of the outside of the exposure facility.

Organization of the exposure and monitoring system is shown diagrammatically in Fig. 2. As this figure shows, each subsystem is closely linked to the preceding and succeeding subsystems. The specific subsystems are discussed in detail below.

Fig. 1. Outside portion of UV-B exposure facility.

Fig. 2. Block diagram of the UV-B exposure and monitoring system.

3. HAZARDS AND PRECAUTIONS

Ultraviolet radiation associated with short wavelengths is extremely dangerous. The UV-B generation equipment used in the UV-B exposure and monitoring system can produce levels of UV radiation that result in permanent damage to unprotected skin and eyes. This radiation is the same as that which produces sunburn following excessive exposure to unfiltered sunlight and which results in eye damage from viewing an unshielded electric welding arc.

Because the intensity of UV radiation is inversely proportional to the square of the distance from the radiant source, exposure at a safe distance is acceptable. Actual measurements were made to provide safe boundary limits around the exposure facility.

Individuals near UV-producing lamps should take appropriate safety precautions:

- Turn off all power when the system is not being used for such activities as measuring irradiance. When the system is on, the proper full-face shields provided at the facility should be worn. These special face shields are designed and approved for reducing UV radiation below 400 nm. Ordinary face shields used to protect the face from spilled chemicals or flying debris do not afford proper UV radiation protection.
- Cover any exposed skin with appropriate clothing (e.g., long-sleeved shirt, long pants, gloves).
- Limit UV radiation exposure to the absolute minimum.

In addition to the UV radiation hazard, this system uses voltage and current levels that can be lethal. Use common sense and do not work on equipment when power is on and excessive levels of moisture are present (e.g., fog, dew, rain).

4. DESIGN BASIS

Recent concerns relating to the depletion of the stratospheric O_3 layer from continued use of CFCs have led to attempts to determine or forecast the long-range effects of elevated levels of solar UV-B radiation. One of the most obvious methods for determining the effects of increased UV-B is to place plant specimens in an area exposed to enhanced UV-B radiation, such as in a chamber or appropriately designated area within the laboratory. Of course, it is also necessary to provide the specimens adequate levels of "normal" light (i.e., ambient sunlight). Plants do not respond solely to the UV-B spectrum (290-320 nm) but require other wavelengths present in the solar spectrum for metabolism and cell repair. Because it is extremely difficult to accurately simulate the solar spectrum in the laboratory environment, field studies are particularly important. To solve this problem, the ORNL UV-B exposure and monitoring system was designed to provide a field-deployable method of appropriately supplementing UV-B radiation and maximizing plant exposure to ambient sunlight.

Just as it is difficult to accurately simulate the solar spectrum within the laboratory, it is vital that the system enhance only the portion of the UV spectrum that is expected to increase as a result of stratospheric O_3 depletion to the limit considered detrimental to plant life. Extensive analysis of the transmission characteristics of the earth's atmosphere with regard to UV-B radiation has been performed by Green (1983) and others. Additional work by Zardecki and Gerstl (1982) in the determination of reasonably expected RAF values was instrumental to proper system design. RAF is the ratio of the expected increase in the intensity of UV-B radiation at the earth's surface relative to a decrease in stratospheric O_3 and is obviously important in determining the design of an exposure system. Currently accepted RAF values range between 1.7 and 2.0. That is, a 1% decrease in the thickness of the stratospheric O_3 layer is expected to result in a 1.7 to 2% change in the intensity of UV-B radiation at the earth's surface. RAF may be applied conservatively to more severe extremes of O_3 depletion, as long as it is understood how the factor is determined. Additional research used to support system design and development work is cited in the references. Obviously, this work provides the basis for determining the required dynamic range and functional characteristics for a supplemental UV-B source and exposure system.

5. UV-B GENERATION AND EXPOSURE SUBSYSTEM

This chapter describes key components, assembly, and geometry of the field exposure units. Key components of the UV-B generation system, other than power supply components (Chap. 6), are the UV lamps, filter materials, and associated support structures.

5.1 ULTRAVIOLET-B GENERATION

5.1.1 Ultraviolet Lamps

The Model UVB-313 lamps used to obtain the necessary UV spectrum were obtained from the Q-Panel Company in Cleveland, Ohio. The UVB-313 lamp is similar to the familiar FS-40 tanning lamp but has a much higher output. These lamps are nominal 40-W, 48-in.-long fluorescent lamps that contain phosphors selected specifically to enhance irradiance in the UV-B portion of the UV spectrum. The UVB-313 lamp was developed by Q-Panel for use in its Model QUV Accelerated Weathering Tester unit and availability is limited. To obtain lamps of this type, it was necessary to assure the vendor in writing that they would be used only in a closely monitored scientific environment and that personnel would be fully instructed in the hazards and precautions associated with the lamps. The spectral distribution of the lamps relative to sunlight is shown in Fig. 3. As this figure shows, the lamps can supply a significant level of UV radiation, and a considerable amount of this radiation lies outside (< 290 and > 320 nm) the spectrum of interest. Using the entire spectrum available from the lamps could produce distorted results, depending on plant sensitivity to the other wavelengths present. Therefore, filters are required to limit the spectral irradiance to the region of interest. Filtering implementation is detailed in Sect. 5.1.2.

The lamps are designed to operate at a nominal filament voltage of 3.2 VAC root mean square (rms) and a steady state arc voltage of ~ 107 VAC rms at 400 mA. As in all fluorescent lamps, efficacy depends on envelope temperature. Maximum efficacy occurs at a bulb wall temperature of 100°F , which corresponds to an indoor ambient temperature of 77°F . Lamps must be installed as complete sets, because the light output of the individual lamp is a function of its operating time. As lamps age, output decreases significantly, and the replacement of a single lamp in a bank of aged lamps will result in nonuniform light distribution. Although the output of each individual lamp in a bank can be "trimmed" by its series ballast rheostat, trimming may not be practical in some cases. For example, if the light output of a lamp has been decreased by increasing the value of the ballast rheostat, then when the output of the entire bank needs to be reduced, the trimmed lamp may extinguish while other lamps are still emitting, thus resulting in nonuniform light distribution.

Individual lamps are held in place and powered by conventional, commercially available lamp end connectors. A detailed view of a typical connector and its mounting is shown in Fig. 4. The lamps are shielded from above by a semicircular section of aluminum to protect the lamps from falling objects such as hail or projected objects such as those from mowing equipment.

Individual lamps fixtures are attached to a lamp rack (a nominal 4- by 4-ft framework).

Fig. 3. Spectral distribution of UVB-313 lamps, FS-40 lamps, and sunlight.

Fig. 4. Ultraviolet lamp mounting detail.

5.1.2 Filters

Lamps used as a source of UV radiation produce radiation not only in the UV-B spectrum (290 to 320 nm) but emit significant radiation from 260 to 375 nm. To limit the exposure spectrum to the area of interest, filter materials must be placed over the lamps to block unwanted wavelengths. While reports on similar work performed in Europe suggest that glass bandpass filters (e.g., Schott UG-11) are employed for this purpose, researchers in the United States have used plastic films. Work performed by Green (1966) shows that plastic films can serve as effective cut-on filters. The two films that have been used extensively in this type of research are mylar and cellulose triacetate. Plots of spectral response for these films are shown in Fig. 5. The plots show that cellulose triacetate can absorb almost all UV-C (<290 nm) radiation, thus allowing UV-B and UV-A through. Mylar film can absorb UV-B and UV-C (<320 nm) radiation, thus allowing only UV-A through. No effective mechanisms exist for eliminating only the UV-A (320 to 400 nm) portion of the lamp spectrum. Therefore, an accommodation must be made by arranging the exposure environment in two areas: one that exposes plants to the UV-A and UV-B portions of the spectrum and a second that exposes the plants only to UV-A radiation. In this manner, the adverse effects of exposure to UV-B can be determined by comparing the damage to UV-B/UV-A-exposed plants to the damage resulting from exposure to UV-A alone and, of course, to the damage from the ambient exposure control group.

Because plastic film filter materials are susceptible to damage and degradation with prolonged exposure to UV radiation, they must be replaced at regular intervals—typically, every 2 weeks. To facilitate replacement, films are held in place on lamp fixtures by magnetic strips. The magnetic strips are attached to the fixtures by epoxy resin, and a companion strip is held in place on top of the other strip by the magnetic attraction between the two strips. Plastic film is sandwiched between the strips. No tools are required for replacing the film. The magnetic strips attached to the lamp fixtures can be seen in Fig. 4.

5.2 EXPOSURE ASSEMBLY

The exposure assembly consists of the aluminum support structure used to elevate the lamp racks. A detailed photograph of the lamp support rack is shown in Fig. 6.

The lamp rack assemblies are suspended from the support structure by stainless steel cables threaded through pulleys. The cables are attached to a winch mechanism that allows adjustment of the distance between the lamp bank and the plants to accommodate the height of the plants and to vary the exposure intensity. Currently, the winch is manually operated, but it could easily be adapted to electrical operation.

Fig. 5. Spectral transmission of cellulose triacetate and mylar films.

Fig. 6. Ultraviolet lamp support rack detail.

6. POWER SUPPLY SUBSYSTEM

The power supply subsystem supplies the filament and arc power for the UV-B lamps in the exposure facility. The UV-B lamps are the same as conventional fluorescent lamps except that the phosphor is selected to enhance the UV-B portion of the total power radiated. The UV-B lamps use hot cathode filaments coated with an electron-emissive material at both ends of the glass envelope. These heated filaments decrease the potential required for arc initiation.

With the application of a sufficient potential difference between the electrodes, electrons are released from the electron-emissive material and form an electric discharge or arc through the mercury plasma originating at the negative electrode and terminating at the relatively positive electrode.

The arc voltage is alternating to provide alternating current (ac) and uniform irradiance and to minimize migration of the mercury to one end of the glass envelope.

The power supply subsystem supplies the arc voltage required to create the electric field inside the lamp and also generates the voltage to heat the filaments (filament voltage). It consists of the variable frequency power supply for the lamp arc power cabinet and the silicon-controlled rectifier (SCR) power controller to supply the filament transformer cabinet. The power supply subsystem cabinet is shown in Fig. 7 (right cabinet).

6.1 LAMP FILAMENT POWER SUPPLY/CONTROL

The lamp filament heating power is derived from the power supply subsystem through the SCR power controller. The SCR power controller operates on a single-phase, 110-VAC rms power source. It is a versatile power supply with plug-in firing control boards that are changed easily from one firing technique to another. The phase-angle firing technique is used for this application.

The SCR power controller maintains a maximum output voltage across the lamp filament when the system is started. To accomplish this, a 20-mA filament control signal is output from the computer control cabinet to the lamp filament power controller when the ambient UV-B reaches the specified start-up level. The SCR power controller provides ~ 120 -V rms to the primary side of a 6.3-V, nominal center-tapped (CT) filament transformer. The CT voltage from this transformer (~ 3.2 V) is then applied across each of the cathode filaments. A separate transformer is provided for each lamp on the isolated end. A common transformer supplies several filaments (lamps) in parallel on the nonisolated end of each lamp.

As mentioned, the SCR power controller uses phase-angle firing to control the voltage level necessary to heat the lamp filament. This voltage level is adjusted by the 4- to 20-mA output signal controlled by the UV-B software. The filament voltage is operated at maximum for cold weather starting and at the minimum allowable value at other times to maximize lamp life.

Fig. 7. Power supply and monitoring/control subsystem cabinets.

6.2 LAMP ARC POWER SUPPLY/CONTROL

One of the main purposes of the control system is to maintain the level of UV-B to which the plants are exposed at a specified percentage above the ambient UV-B level throughout daylight hours.

Direct control of the lamp arc power supply is used to maintain the UV-B level. As the ambient UV-B level increases, the lamp arc power supply increases the output voltage level, resulting in increased lamp current and thus increased lamp intensity. Similarly, when the ambient UV-B decreases, the lamp arc voltage decreases, thereby decreasing the UV-B intensity to which the plants are exposed.

The lamp arc power supply subsystem consists of the single- to three-phase converter and the variable frequency power supply. These components are housed in the power supply cabinet.

The variable frequency power supply provides ac power over the frequency range of 60 to 3000 Hz for output to the lamp arc power cabinet. The power supply output provides a maximum output voltage of 230 VAC rms, which is adjustable between 0 and 230 VAC rms from the 4- to 20-mA output signal controlled by the UV-B software. Total available power is 3000 VA at full-rated output voltage.

The lamp arc power supply is contained in a standard 19-in. rack-mount enclosure. The front panel control and indicators consist of a power-on indicator lamp, voltage amplitude control, and power circuit breaker that applies line power to the unit. Output frequency is determined by a plug-in oscillator. The output frequency range is from 45 to 5000 Hz. The dimensions of the variable frequency power supply are 17.5 × 19 × 23 in. The power supply weighs ~310 lb.

7. SENSOR SUBSYSTEM

The main purpose of the UV-B project is to monitor the intensity of the ambient UV-B and maintain the experimental plant species in an environment at a specified percentage of UV-B intensity above the ambient level. Therefore, a monitoring system is necessary to continuously report the ambient UV-B intensity and the UV-B intensity received by the experimental plant species. By using the feedback from the measured values of UV-B intensity, the UV-B computer program can accurately control the UV-B intensity received by the experimental plants. This monitoring system is realized by the UVX radiometer and sensor connected to the input/output (I/O) ports of the computer system. The UVX radiometer has a digital readout in radiometric units that range from 0 to 20 mW/cm², with precision traceable to the National Institute of Standards and Technology. The radiometer features three sensitivity ranges covering the span from 200 μ W/cm² to 20 mW/cm².

The UVX radiometer is powered by a single 9-V battery and is thus completely portable. The radiometer has low-battery indicators and overranging capability. It also has a sensor input receptacle to allow the sensor cable to input the measured UV-B intensity from the UV-B sensor, and it has a recorder output jack for transmitting the UV-B measurement to the computerized data acquisition system via a cable connected to the I/O modules on the computer.

7.1 ULTRAVIOLET SENSORS

The UVX sensor is designed for measuring radiant incidences from line-type and phosphor-coated mercury sources. To ensure proper operation of the sensor, several factors (including calibration, spectral response, and sensor field of view) must be clearly understood and practiced. These concepts are discussed as follows.

To ensure accurate radiometer readings, the meter should be zeroed before the sensor is placed in the UV-B environment. Zeroing is accomplished by covering the sensor so that no UV-B light strikes the sensor window (usually, turning the sensor upside down so the white circular window on the front faces an opaque surface is sufficient). Then, the range switch on the radiometer is set in the 200- μ W/cm² position, and the zero adjustment trimpot is adjusted for a display reading of 0.0 μ W/cm².

The UV-B spectrum includes those wavelengths of light between 290 and 320 nm. For accurate measurement of UV-B intensity, the UV sensors are calibrated to a monochromatic sources of UV-B radiation at 310 nm. The UV-B sensors are calibrated by adjusting their response to match that of a transfer standard detector when identically exposed to radiation of the appropriate calibration wavelength from the stable 310-nm monochromatic source. The absolute radiation level from this source is set by simultaneously matching it with that of the corresponding National Institute of Standards and Technology standard of spectral irradiance. Because only the sensor itself is adjusted during this calibration, it is interchangeable with any other UVX UV-B sensor or UVX radiometer electronics without affecting the overall accuracy of the monitoring system. The UV-B sensors should be recalibrated every 6 months to ensure continued accuracy.

The UVX sensor incorporates a unique diffuser at the incident surface which permits radiation detection to incidence angles of $\pm 81^\circ$ from normal. This allows the sensor to have

an almost ideal cosine response. The sensor field of view is also a major factor in determining the accurate UV-B intensity from the UV-B sensor reading. Because the power incident on the sensor is being measured, it is important that the sensor's field of view include the entire length of the UV-B lamps to ensure that the power from the excluded ends is sensed and included in the measurement. If the field of view does not include the entire lamp length, increasing the distance of the sensor from the lamp would be necessary. Otherwise, the UV-B intensity would be less and a different reading would be obtained.

Sensor spectral response is an important factor in obtaining accurate readings from the UV-B sensor. An ideal UV-B radiometer would respond exactly the same to all wavelengths of radiation in the region of interest (290 to 320 nm) and have zero response to all other wavelengths (ideal bandpass filter). Unfortunately, real radiometers have a response that peaks at some wavelength and diminishes toward zero from this peak, as is the case with the UVX sensor, whose absolute spectral response is shown in Fig. 8. However, by knowing the relative spectral distribution of the UV-B sources (UVB-313 lamps and sunlight) and the absolute spectral response of the radiometer, it is possible to calculate correction factors that will allow accurate, absolute UV-B irradiance measurements. The correction factor is the percentage of the radiometer's response that is caused by only those wavelengths of interest (290 to 320 nm), because the total output of the radiometer will contain contributions from electromagnetic radiation outside the UV-B region.

The output of the UVX radiometer is given by the following equation:

$$UVX = \int_{-\infty}^{+\infty} B(\lambda) A(\lambda) d\lambda = k \int_{-\infty}^{+\infty} B(\lambda) D(\lambda) d\lambda , \quad (1)$$

where

$B(\lambda)$ = absolute spectral response of UVX radiometer (unitless),

$D(\lambda)$ = relative spectral power distribution of light source to be measured ($W/m^2/nm$),

k = amplitude multiplier for $D(\lambda)$,

$A(\lambda) = k * D(\lambda)$, absolute spectral power distribution of light source to be measured ($W/m^2/nm$),

UVX = UVX radiometer output (W/m^2).

As shown in Fig. 8 and in the spectral power data on the UVB-313 lamp sources and sunlight (Fig. 3), the limits of integration in Eq. (1) can be limited to 270 to 400 nm because the product of $B(\lambda)*D(\lambda)$ approaches zero outside this range. Thus, the total output of the UVX radiometer is a function of the product of the UVX radiometer's absolute spectral response with the spectral power distribution of the light source to be measured.

Fig. 8. UVX meter absolute response.

A correction factor for the UVX radiometer can be derived which will give the portion of the radiometer response caused only by electromagnetic radiation in the UV-B region of interest. Solving for k in Eq. (1) gives the following equation:

$$k = \frac{UVX}{\int_{270 \text{ nm}}^{400 \text{ nm}} B(\lambda) D(\lambda) d\lambda} \quad (2)$$

The desired measured UV-B irradiance is given by Eq. (3) as follows:

$$\int_{290 \text{ nm}}^{320 \text{ nm}} A(\lambda) d\lambda = k \int_{290 \text{ nm}}^{320 \text{ nm}} D(\lambda) d\lambda \quad (3)$$

Substituting k from Eq. (2) into Eq. (3) gives the following:

$$\int_{290 \text{ nm}}^{320 \text{ nm}} A(\lambda) d\lambda = \left[\frac{UVX}{\int_{270 \text{ nm}}^{400 \text{ nm}} B(\lambda) D(\lambda) d\lambda} \right] \int_{290 \text{ nm}}^{320 \text{ nm}} D(\lambda) d\lambda \quad (4)$$

Thus, a correction factor can be defined as

$$\text{Correction Factor} = \frac{\int_{290 \text{ nm}}^{320 \text{ nm}} D(\lambda) d\lambda}{\int_{270 \text{ nm}}^{400 \text{ nm}} B(\lambda) D(\lambda) d\lambda} \quad (5)$$

Thus, the desired output is given by Eq. (6) as follows:

$$\int_{290 \text{ nm}}^{320 \text{ nm}} A(\lambda) d\lambda = (UVX) * (\text{Correction Factor}) \quad (6)$$

The relative spectral power distribution of the UVB-313 lamp and ambient sunlight was shown in Fig. 3. The data used to generate these figures were used for $D(\lambda)$ in Eq. (5) to obtain correction factors for the UVX radiometer when measuring ambient sunlight or the UVB-313 lamps. These correction factors were stored in the computer and used by the UV-B software to correct the UVX meter's output so it would only be proportional to the UV-B irradiance.

8. MONITORING AND CONTROL SUBSYSTEM

The monitoring and control subsystem is designed to automatically perform the data acquisition and data storage functions. Equally important purposes of the monitoring and control subsystem are the generation of control commands and output of signals to both the lamp arc power cabinet and SCR power controller. The left cabinet in Fig. 7 shows the computerized monitoring and control subsystem.

8.1 COMPUTER CONFIGURATION

The computer configuration for the UV-B monitoring and control subsystem is centered around an International Business Machines (IBM) Model 7234 industrial personal computer AT (PC AT) with rack-mount capability. This computer was chosen for its direct applicability to the UV-B project, low cost, and mountability in a standard 19-in. industrial rack cabinet.

The IBM Model 7534, industrial rack-mount, color graphics monitor is used to display the varying real-time status information in formatted screens as well as display data in graphic form.

8.2 SENSOR SIGNAL INPUT

The transfer of information from the lamp banks into the computer is accomplished with two MetraByte Model EXP-16, 16-channel multiplexers and a MetraByte Model DAS-8, medium-speed analog-to-digital converter (ADC). The multiplexers are mounted in a 19-in., rack-mount frame and contain screw terminals for terminating all analog input field wiring. The ADC plugs into the back of the IBM PC AT at one of the available backplane expansion slots.

Connections to the ADC from the EXP-16 are made through one of two standard 37-pin, D male connectors. Besides multiplexing the input channels, the EXP-16 provides signal amplification, filtering, and conditioning. The EXP-16 allows for user switch-selectable and programmable gains. The EXP-16 analog input module accepts a signal in the 0- to 5-V dc range. All 16 differential analog input channels are selected by a 4-bit TTL/CMOS-compatible address.

8.3 CONTROL SIGNAL OUTPUT

Control signal outputs are generated by a MetraByte Model DDA-06 analog output interface card. The DDA-06 plugs into the back of the IBM PC AT at one of the available backplane expansion slots. The 4- to 20-mA controller output signals are interfaced to field wiring through the MetraByte Model STA-U screw termination accessory board. Field wiring terminates directly onto screw terminals provided on the STA-U. Connections from the STA-U and the DDA-06 are made via a standard 37-pin ribbon cable.

The control signal output is generated by the control algorithm of the UV-B software and is based on proportional-integral control theory. This algorithm uses the difference between

the desired percentage UV-B above ambient and the actual percentage UV-B above ambient to determine the required adjustment in control signal output to achieve the desired result.

8.4 DATA STORAGE

Data storage for the system is provided by two removable Bernoulli disk units. This type of unit was selected for its large data storage capability, ruggedness, and low cost. The removable disks allow data to be accumulated in the field and taken to the laboratory or office for analysis.

8.5 SYSTEM POWER CONDITIONING

Power for the control and data acquisition computer system is provided by the power line conditioner. Inclusion of the conditioner was necessary because of the remoteness of the site and the possibility of power supply voltage surges.

8.6 SYSTEM SOFTWARE

System software is responsible for the data acquisition and system control functions. The software is based on the enhanced version of the BASIC programming language known as Microsoft's QUICK-BASIC and is organized in a modular form, with each module designed to perform a particular function.

The interaction between the system and the user is via a series of screens accessed through the command line options displayed at the bottom of each screen.

The most commonly used—and the one that appears when the UV-B program is initiated—is the main menu/overall system display screen.

8.6.1 Overall System Display

The overall system display contains data that are presented on the system color monitor and updated in real time. Components of the display are as follows:

- Top left: Current date
- Top right: Time of day
- Top center: UV-B RESEARCH FACILITY DATA ACQUISITION AND CONTROL
SYSTEM DATA BASE POINTS DISPLAY
- Column headings: NAME (name of point), TYPE (type of point: analog output or input),
VALUE (numeric value at that instant), ENGR. UNITS (proper
engineering units), MIN (minimum values for point), MAX (maximum
values for point)
- Lines 1–4: Information on the UV-B flux under cellulose acetate film for lamp banks
- Line 5: Ambient UV-B measured reaching ground
- Line 6: Air temperature (°F) in environment of experimental plant species
- Line 7: Rms value of arc voltage
- Line 8: Rms value of filament voltage output to filament transformer cabinet
- Line 9: Arc voltage as a percentage of power supply capacity
- Line 10: Filament voltage as a percentage of the SCR power controller

- Line 11: Raw UVX measurement of the ambient UV-B intensity before the correction factor is applied
- Lines 12-15: Raw UV-B intensity under CA before the correction factor is applied
- Line 16: Average UV-B intensity under cellulose acetate
- Line 17: Set point percentage above ambient at which UV-B intensity under cellulose acetate is set
- Line 18: Control percentage above ambient UV-B current achieved
- Line 19: Percentage shade to include in shade factor calculations under cellulose acetate lamp bank
- Lines 20-21: Readings for quantum sensors (mmol)
- Line 22: Filament voltage as a percentage of capacity output to lamp filament
- Line 23: Arc voltage as a percentage of capacity output to lamp bank

9. DATA ANALYSIS AND CONTROL EVALUATION

System performance and exposure intensities were evaluated from data logged automatically by the UV-B data acquisition and control system over a major part of the plant-growing season.

Retrieval and analysis of the data are focused here. Data analysis has two important functions: (1) correlating plant behavior and conditions with the increased intensities of UV-B and (2) evaluating the performance of the data acquisition and control system.

Examples of data collected from the system are shown in Figs. 9, 10, and 11. These figures illustrate the general performance of the power supply subsystem and the data acquisition and control subsystem. Figure 9 shows the spectral power distribution beneath the lamp banks of the exposure facility at various output power settings and emphasizes the system's ability to supplement ambient UV-B. Figure 10 shows the variation of UV-B intensity under a cellulose-acetate-filtered lampbank and the ambient UV-B for a sunny day with clouds directly overhead. Figure 11 shows the percentage difference between the curves in Fig. 10 and the percentage variation over a 1-d period.

By comparing the desired conditions with the actual data collected and graphically displayed, a reasonable analysis of control system performance can be made.

For an accurate system performance evaluation, the initial specifications and requirements of the UV-B data acquisition system can be used as the standard of reference.

The principal aim of the data acquisition and control system is to maintain the UV-B intensity level reaching the experimental plant species at the specified controller set point percentage above ambient UV-B intensity.

As shown in Fig. 10, the UV-B intensity received by the experimental plant species (from the control system) is always above ambient UV-B intensity for the typical day.

In Fig. 11, the percentage difference between the UV-B intensities graphed in Fig. 10 is shown. As can be deduced from this graph, the percentage difference fluctuates around 32%, which is the specified controller set point.

Fig. 9. Output of UVB-313 lamp filtered with CA film.

Fig. 10. Measurement of UVB under CA filter lamp bank and ambient UV-B.

Fig. 11. Percentage of the difference between curves in Fig. 10.

10. SUMMARY

An automated system for generating and controlling UV-B radiation and for collecting data about analyzing plant species has been designed and implemented.

The system has exposed plants at elevated UV-B intensities over a season of growth, and plans are being made for future seasons of exposure.

The system is dependable, interactive with operators, and expandable to allow for additional parameters and increased versatility.

11. CONCLUSIONS AND RECOMMENDATIONS

The UV-B exposure and monitoring system functions well and has demonstrated good performance over the course of the plant growing season. It has provided much consistent and correct data for analysis and excellent control of exposure intensities.

The software is user friendly and provides flexibility and interaction with the system.

The system user interface has proved easy to learn and use and provides information on all important parameters in the exposure environment. The data display screens provide data, status information, and menus for changing the method of control.

12. REFERENCES

Green, A. E. S., Ed. (1966). *The Middle Ultraviolet: Its Science and Technology*, John Wiley & Sons, Inc., New York, pp. 332-38.

Green, A. E. S. (1983). "The Penetration of Ultraviolet Radiation to the Ground," *Physiol. Plant* **58**, 351-59.

Zardecki, A., and Gerstl, S. A. W. (1982). *Calculations of Increased Solar UV Fluxes and DUV Doses Due to Stratospheric Ozone Depletions*, Los Alamos National Laboratory report LA-9233-MS.

13. BIBLIOGRAPHY

Caldwell, M. M., Gold, W. G., Harris, G., and Ashurst, C. W. (1983). "A Modulated Lamp System for Solar UV-B (280-320 nm) Supplementation Studies in the Field," *Photochem. Photobiol.* **37**, 479-85.

Dave, J. V., and Halpren, P. (1976). "Effect of Changes in Ozone Amount on the Ultraviolet Radiation Received at Sea Level of a Model Atmosphere," *Atmos. Environ.* **10**, 547-55.

Green, A. E. S., et al. (1974). "The Middle Ultraviolet Reaching the Ground," *Photochem. Photobiol.* **19**, 251-59.

National Academy of Sciences (1982). "Causes and Effects of Stratospheric Ozone Reduction: An Update," *National Academy Press*, Washington, D.C.

APPENDIX A

SYSTEM PROGRAM SOFTWARE LISTINGS

'THIS FILE CONTAINS VARIABLES COMMON TO DATAL&C AND PDBINPUT

COMMON SHARED

NUM.ANI%,NUM.ANO%,NUM.DGI%,NUM.DGO%,NUM.TXT%,NUM.ADC%,NUM.MIC%,NUM.RTD%,NUM.PNT%,PNT.TYPE\$(1),PNT.NAME\$(1),ANI.TBL%(1),ANI.EUNIT\$(1)

COMMON SHARED

ANI.DEV\$(1),ANI.UNIT%(1),ANI.CHNL%(1),ANI.CNVTYPE\$(1),ANI.COEF(1),ANI.COEFB(1),ANI.COEF(1),ANI.LOLIM!(1)

COMMON SHARED

ANI.HILIM!(1),ANI.SCAN%(1),ANI.PLINTRVL!(1),ANI.STATUS%(1),ANI.ANO\$,ANO.TBL%(1),ANO.STATUS%(1),ANO.CHNL%(1),ANO.SP(1),ANO.MAN.OUTPUT%(1),ANO.INI.OUTPUT(1),ANO.KP(1),ANO.KI(1),ANO.VALUE(1)

COMMON SHARED

DGI\$,DGI.TBL%(1),DGI.DEV\$(1),DGI.UNIT%(1),DGI.CHNL%(1),DGI.PLINTRVL!(1),DGI.STATUS%(1)

COMMON SHARED

DGO\$,DGO.TBL%(1),DGO.DEV\$(1),DGO.UNIT%(1),DGO.CHNL%(1),DGO.STATUS%(1),TXT\$,TXT.TBL%(1),TXT.TEXT\$(1),TXT.STATUS%(1)

COMMON SHARED PDB.LOADED%,COMMA\$,DBDRIVE\$,TYPE.PNTR%(1)

COMMON SHARED TOT.POINTS%,MAXANI%,MAXANO%,MAXDGI%,MAXDGO%,MAXTXT%

COMMON SHARED ADC.LOC.ANI%(1),MIC.LOC.ANI%(1),RTD.LOC.ANI%(1),ADC.CHNL.TBL%(1)

COMMON SHARED ANI.COMM1.DB%(1)

TOT.POINTS% = 400

MAXANI% = 364

MAXANO% = 6

MAXDGI% = 10

MAXDGO% = 10

MAXTXT% = 10

MAXADC% = 350 ' Guess for now !

MAXRTD% = 0

MAXMIC% = 45

IF PDB.LOADED% GOTO ENDINC1

REDIM PNT.NAME\$(TOT.POINTS%) 'POINT NAMES

REDIM PNT.TYPE\$(TOT.POINTS%) 'TYPE (ANI,ANO,DI,DO,TXT)

REDIM TYPE.PNTR%(TOT.POINTS%) 'POINTER TO ENTRY IN CORRES. TYPE TABLE.

REDIM ANI.EUNIT\$(MAXANI%) 'ENGR. UNITS (E.G., PPB,DEGC, ETC.)

REDIM ANI.DEV\$(MAXANI%) 'TYPE OF DEVICE ANALOG INPUT COMES FROM.

REDIM ANI.UNIT%(MAXANI%) 'UNIT# (COM-PORT#,ADC-CARD#,ETC.)

REDIM ANI.CHNL%(MAXANI%) 'CHANNEL # (COM-ELEMENT#,ADC-CHANNEL OR BIT#)

REDIM ANI.CNVTYPE\$(MAXANI%) 'CONVERSION TYPE (E.G.,SPC,T,LIN)

REDIM ANI.COEF(1)(MAXANI%) 'ENGR. UNITS CONVERSION COEFFICIENT "A"

REDIM ANI.COEFB(1)(MAXANI%) 'ENGR. UNITS CONVERSION COEFFICIENT "B"

REDIM ANI.COEF(1)(MAXANI%) 'ENGR. UNITS CONVERSION COEFFICIENT "C"

REDIM ANI.LOLIM!(MAXANI%) 'MINIMUM ALLOWABLE VALUE

REDIM ANI.HILIM!(MAXANI%) 'MAXIMUM ALLOWABLE VALUE

REDIM ANI.SCAN%(MAXANI%) 'SAMPLING RATE

'REDIM ANI.SAMTIM!(MAXANI%)

'REDIM PREV.SAM.TIME!(MAXANI%)

'REDIM PREV.ANI.SAMTIM!(MAXANI%)

REDIM ANI.STATUS%(MAXANI%) 'STATUS (0-DISABLED,1-ENABLED) -- \$ OR % ??

'REDIM ANI.VALUE!(MAXANI%) 'CURRENT MEASURED VALU

```

'REDIM ANI.NEW.VAL%(MAXANI%) 'STATUS OF ANI.VALUE! FOR DISK STORAGE
'REDIM ANI.PREVAL!(MAXANI%) 'PREVIOUSLY MEASURED VALUE
'REDIM ANI.MAXVAL!(MAXANI%) 'MAX MEASURED VALUE SINCE START OF EXPERIMENT
'REDIM ANI.MINVAL!(MAXANI%) 'MIN MEASURED VALUE SINCE START OF EXPERIMENT
'REDIM ANI.AVERAGE!(MAXANI%) 'AVERAGE MEAS. VALUE SINCE EXPERIMENT START
'REDIM ANI.VARIANCE!(MAXANI%) 'MEASUREMENT VARIANCE SINCE EXPERIMENT
START
'REDIM ANI.VALSUM!(MAXANI%) 'VALUE SUM
'REDIM ANI.VALSQSUM!(MAXANI%) 'VALUE SQUARED SUM
'REDIM ANI.MVCOUNT%(MAXANI%) 'SAMPLING COUNTER
'REDIM ANI.CUMHRAV!(MAXANI%,24)'CUMULATIVE HOURLY AVERAGE
'REDIM ANI.DLTIM!(MAXANI%)
'REDIM ANI.PLTIM!(MAXANI%)
,
'REDIM DAOOUT(NUM.CNTLS%)
,
'REDIM PDB.DATA$(80)      'TEMP ARRAY FOR READING POINT DATA RECORDS
,
'ANALOG OUTPUT POINTS
'REDIM ANO.VALUE!(MAXANO%)
'REDIM ANO.STATUS%(MAXANO%)
'REDIM ANO.CHNL%(MAXANO%)
'REDIM ANO.SP(MAXANO%)
'REDIM ANO.KP(MAXANO%)
'REDIM ANO.KI(MAXANO%)
'REDIM ANO.MAN.OUTPUT%(MAXANO%)
'REDIM ANO.INI.OUTPUT(MAXANO%)
,
'REDIM ANI.TBL%(MAXANI%) 'ANALOG INPUT LOC RELATIVE TO SYS POINT DB
'REDIM ANO.TBL%(MAXANO%)
'REDIM DGI.TBL%(MAXDGI%) 'DIGITAL INPUT LOC RELATIVE TO SYS POINT DB
'REDIM DGO.TBL%(MAXDGO%) 'DIGITAL OUTPUT LOC RELATIVE TO SYS POINT DB
'REDIM TXT.TBL%(MAXTXT%)
'REDIM TXT.TEXT$(MAXTXT%) 'SCREEN TEXT DATA TABLE
'REDIM TXT.STATUS%(MAXTXT%)
,
'REDIM ANI.PLINTRVL!(MAXANI%) 'ANALOG INPUT PRINTER LOGGING INTERVAL
'REDIM DGI.PLINTRVL!(MAXDGI%) 'DIGITAL INPUT PRINTER LOGGING INTERVAL
,
'REDIM DGI.VALUE%(MAXDGI%)
'REDIM DGI.DEV$(MAXDGI%)
'REDIM DGI.UNIT%(MAXDGI%)
'REDIM DGI.CHNL%(MAXDGI%)
'REDIM DGI.STATUS%(MAXDGI%)
,
'REDIM DGO.VALUE%(MAXDGO%)
'REDIM DGO.DEV$(MAXDGO%)
'REDIM DGO.UNIT%(MAXDGO%)
'REDIM DGO.CHNL%(MAXDGO%)
'REDIM DGO.STATUS%(MAXDGO%)
,
'REDIM ADC.LOC.ANI%(MAXADC%)

```


REDIM MIC.LOC.ANI%(MAXMIC%)
REDIM RTD.LOC.ANI%(MAXRTD%)
REDIM ADC.CHNL.TBL%(MAXADC%)
REDIM ANI.COMM1.DB%(MAXANI%)

ENDINC1:

□

***** POINT DATA BASE INPUT ROUTINE *****
' FOR ULTRAVIOLET-B RESEARCH FACILITY SOFTWARE

REM \$INCLUDE: 'COMVARI.BAS'

'lprint " IN PDBUVB.BAS "

41810 OPEN DBDRIVE\$ + "uvb.PDB" FOR INPUT AS #1
41850 NUM.ANI%=0
41851 NUM.ANO%=0
41890 NUM.DGI%=0
41930 NUM.DGO%=0
41970 NUM.TXT%=0
42010 NUMPNT%=0
42050 IF EOF(1) GOTO 44570
42090 NUMPNT%=NUMPNT%+1
42130 I%=NUMPNT%
42170 LINE INPUT #1,PDB.DATAS\$
42210 PNT.TYPES\$(I%)=MID\$(PDB.DATAS\$,1,3)
42250 COMMA.LOC%=4
42290 GOSUB 44690 'GET NEXT FIELD
42330 'LPRINT FSTRT%,FLENGTH%,I%,PDB.DATAS\$,MID\$(PDB.DATAS\$,FSTRT%,FLENGTH%)
42370 PNT.NAMES\$(I%) = MID\$(PDB.DATAS\$,FSTRT%,FLENGTH%)
42410 GOSUB 44690
42450 'EXTRACT THE FIELD INFORMATION BASED ON POINT TYPE AND CONSTRUCT
42490 'VARIOUS DATA TABLES AS DATA BASE IS READ
42530 '
42570 IF PNT.TYPES\$(I%) <> ANI\$ GOTO 43572
42610 NUM.ANI%=NUM.ANI%+1
42650 ANI.TBL%(NUM.ANI%)=I%
42690 TYPE.PNTR%(I%)=NUM.ANI% 'CHAIN BACK TO POINT NAME TABLE
42730 ANI.EUNIT\$(NUM.ANI%)=MID\$(PDB.DATAS\$,FSTRT%,FLENGTH%)
42770 GOSUB 44690
42810 ANI.DEV\$(NUM.ANI%)=MID\$(PDB.DATAS\$,FSTRT%,FLENGTH%)
42850 GOSUB 44690
42890 ANI.UNIT%(NUM.ANI%)=VAL(MID\$(PDB.DATAS\$,FSTRT%,FLENGTH%))
42930 GOSUB 44690
42970 ANI.CHNL%(NUM.ANI%)=VAL(MID\$(PDB.DATAS\$,FSTRT%,FLENGTH%))
43010 GOSUB 44690
43050 ANI.CNVTYP\$(NUM.ANI%)=MID\$(PDB.DATAS\$,FSTRT%,FLENGTH%)
43090 GOSUB 44690
43130 ANI.COEFA(NUM.ANI%)=VAL(MID\$(PDB.DATAS\$,FSTRT%,FLENGTH%))
43170 GOSUB 44690
43210 ANI.COEFB(NUM.ANI%)=VAL(MID\$(PDB.DATAS\$,FSTRT%,FLENGTH%))
43250 GOSUB 44690
43290 ANI.COEFC(NUM.ANI%)=VAL(MID\$(PDB.DATAS\$,FSTRT%,FLENGTH%))
43330 GOSUB 44690
43370 ANI.LOLIM!(NUM.ANI%)=VAL(MID\$(PDB.DATAS\$,FSTRT%,FLENGTH%))
43410 GOSUB 44690
43450 ANI.HILIM!(NUM.ANI%)=VAL(MID\$(PDB.DATAS\$,FSTRT%,FLENGTH%))
43490 GOSUB 44690
43530 ANI.SCAN%(NUM.ANI%)=(VAL(MID\$(PDB.DATAS\$,FSTRT%,FLENGTH%)))
43570 GOTO 44530

43571 '

43572 IF PNT.TYPE\$(I%) <> ANOS GOTO 43610

NUM.ANO%=NUM.ANO%+1

ANO.TBL%(NUM.ANO%)=I%

TYPE.PNTR%(I%)=NUM.ANO%

ANO.STATUS%(NUM.ANO%)=VAL(MID\$(PDB.DATAS,FSTRT%,FLENGTH%))

GOSUB 44690

ANO.CHNL%(NUM.ANO%)=VAL(MID\$(PDB.DATAS,FSTRT%,FLENGTH%))

GOSUB 44690

ANO.SP(NUM.ANO%)=VAL(MID\$(PDB.DATAS,FSTRT%,FLENGTH%))

GOSUB 44690

ANO.MAN.OUTPUT%(NUM.ANO%)=VAL(MID\$(PDB.DATAS,FSTRT%,FLENGTH%))

GOSUB 44690

ANO.INI.OUTPUT(NUM.ANO%)=VAL(MID\$(PDB.DATAS,FSTRT%,FLENGTH%))

GOSUB 44690

ANO.KP(NUM.ANO%)=VAL(MID\$(PDB.DATAS,FSTRT%,FLENGTH%))

GOSUB 44690

ANO.KI(NUM.ANO%)=VAL(MID\$(PDB.DATAS,FSTRT%,FLENGTH%))

GOTO 44530

43573 '

43610 IF PNT.TYPE\$(I%) <> DGI\$ GOTO 43930

43650 NUM.DGI%=NUM.DGI%+1

43690 DGI.TBL%(NUM.DGI%)=I%

43730 TYPE.PNTR%(I%) = NUM.DGI%

43731 DGI.DEV\$(NUM.DGI%)=MID\$(PDB.DATAS,FSTRT%,FLENGTH%)

43732 GOSUB 44690

43770 DGI.UNIT%(NUM.DGI%)=VAL(MID\$(PDB.DATAS,FSTRT%,FLENGTH%))

43810 GOSUB 44690

43850 DGI.CHNL%(NUM.DGI%)=VAL(MID\$(PDB.DATAS,FSTRT%,FLENGTH%))

43851 GOSUB 44690

43854 DGI.PLINTRVL!(NUM.DGI%)=VAL(MID\$(PDB.DATAS,FSTRT%,FLENGTH%))

43855 GOSUB 44690

43856 DGI.STATUS%(NUM.DGI%)=VAL(MID\$(PDB.DATAS,FSTRT%,FLENGTH%))

43890 GOTO 44530

43930 IF PNT.TYPE\$(I%) <> DGO\$ GOTO 44250

43970 NUM.DGO%=NUM.DGO%+1

44010 DGO.TBL%(NUM.DGO%)=I%

44050 TYPE.PNTR%(I%) = NUM.DGO%

44051 DGO.DEV\$(NUM.DGO%)=MID\$(PDB.DATAS,FSTRT%,FLENGTH%)

44052 GOSUB 44690

44090 DGO.UNIT%(NUM.DGO%)=VAL(MID\$(PDB.DATAS,FSTRT%,FLENGTH%))

44130 GOSUB 44690

44170 DGO.CHNL%(NUM.DGO%)=VAL(MID\$(PDB.DATAS,FSTRT%,FLENGTH%))

44210 GOSUB 44690

44211 DGO.STATUS%(NUM.DGO%)=VAL(MID\$(PDB.DATAS,FSTRT%,FLENGTH%))

44212 GOTO 44530

44250 IF PNT.TYPE\$(I%) <> TXT\$ GOTO 44530

44290 NUM.TXT%=NUM.TXT%+1

44330 TXT.TBL%(NUM.TXT%)=I%

44370 TYPE.PNTR%(I%)=NUM.TXT%

44410 TXT.TEXT\$(NUM.TXT%)=MID\$(PDB.DATAS,FSTRT%,FLENGTH%)

44411 GOSUB 44690

44412 TXT.STATUS%(NUM.TXT%)=VAL(MID\$(PDB.DATAS,FSTRT%,FLENGTH%))

44530 GOTO 42050

```

44570 CLOSE #1
' FOR PLOOP% = 1 TO NUMPNT%      'DEBUG
' LPRINT PNT.NAME$(PLOOP%)      'DEBUG
' NEXT PLOOP%                    'DEBUG
*****
' CREATING AIN SUBSETS OF ADC,MIC,& RTD POINTS FOR CONVERSIONS

      NUM.ADC%=0
      NUM.MIC%=0
      NUM.RTD%=0
      FOR ANI.LOOP%= 1 TO NUM.ANI%
ADCOK:   IF ANI.DEV$(ANI.LOOP%) <> "ADC" THEN GOTO MICOK
      NUM.ADC%=NUM.ADC%+1
      ADC.LOC.ANI%(NUM.ADC%)=ANI.LOOP%
'      ADC.CHNL.TBL%(NUM.ADC%)=ANI.CHNL%(ANI.LOOP%)
      GOTO ANDLP
MICOK:   IF ANI.DEV$(ANI.LOOP%) <> "MIC" THEN GOTO RTDOK
      NUM.MIC%=NUM.MIC%+1
      MIC.LOC.ANI%(NUM.MIC%)=ANI.LOOP%
      GOTO ANDLP
RTDOK:   IF ANI.DEV$(ANI.LOOP%) <> "RTD" THEN GOTO ANDLP
      NUM.RTD%=NUM.RTD%+1
      RTD.LOC.ANI%(NUM.RTD%)=ANI.LOOP%
      GOTO ANDLP
ANDLP:   NEXT ANI.LOOP%
'PRINT NUM.ADC%,NUM.MIC%,NUM.RTD%

*****

      PDB.LOADED%=-1
44610 CHAIN "UVB"
44650 '
44690 'ENTRY POINT FOR LOCATING FIELDS IN PDB STRING - PDB FIELD LENGTH IS
44730 'COMPUTED AS A FUNCTION OF RECORD COMMA LOCATION
44770 '
44810 FSTRT%=COMMA.LOC%+1
44811 LASTLOC%=LEN(PDB.DATA$)
44850 COMMA.LOC%=INSTR(FSTRT%,PDB.DATA$,COMMA$)
44890 IF COMMA.LOC% < 0 GOTO 45010
44930 IF COMMA.LOC%=0 THEN FLENGTH%=LASTLOC%
44931 COMMA.LOC%=LASTLOC% - FSTRT% + 1
44970 IF COMMA.LOC% > LASTLOC% THEN CLS:PRINT "DATA BASE LOAD ERROR, RECORD #:
",I%:STOP
44971 RETURN
45010 FLENGTH%=COMMA.LOC%-FSTRT%
45050 RETURN
45090 '
45130 '***** END OF POINT DATA BASE INPUT ROUTINE *****
□

```

' ULTRAVIOLET-B RESEARCH FACILITY SOFTWARE
' J. A. MCEVERS AND M. S. HILEMAN
,

'Modification History
,

'Modified: 06/20/90 By: Mike Hileman

'Modification:

'Changed the Controller Output section so that the Lamp Filament

'Controller would always output 100 Percent'

REM \$INCLUDE: 'COMVARI.BAS' 'BRING IN COMMON DATA BASE DEFINITION.
'DEFINE LOGICAL TRUE AND FALSE FOR ENTIRE PROGRAM.

'LPRINT " IN UVB.BAS "

FALSE%=0

TRUE%=-1

OFFICE% = TRUE%

OFFICE% = FALSE% 'TRUE=DOES NOT NEED FIELD I/O HDW.

KEY 19,CHR\$(&H00)+CHR\$(&H49) 'page up key

KEY 20,CHR\$(&H00)+CHR\$(&H51) 'page down key

KEY 21,CHR\$(&H40)+CHR\$(&H49) 'page up key with caps lock

KEY 22,CHR\$(&H40)+CHR\$(&H51) 'page down key with caps lock

KEY 15,CHR\$(&H80)+CHR\$(&H49) 'page up key extended keyboard

KEY 16,CHR\$(&H80)+CHR\$(&H51) 'page down key extended keyboard

KEY 17,CHR\$(&H80+&H40)+CHR\$(&H49) 'PAGE UP KEY WITH CAPS LOCK

KEY 18,CHR\$(&H80+&H40)+CHR\$(&H51) 'PAGE DOWN KEY WITH CAPS LOCK

KEY(15) ON

KEY(16) ON

KEY(17) ON

KEY(18) ON

KEY(19) ON

KEY(20) ON

KEY(21) ON

KEY(22) ON

ON KEY(11) GOSUB CNVUP 'CONVERT UP ARROW INTO WORD "UP"

ON KEY(15) GOSUB CNVUP

ON KEY(17) GOSUB CNVUP

ON KEY(19) GOSUB CNVUP

ON KEY(21) GOSUB CNVUP

ON KEY(14) GOSUB CNVDN 'CONVERT DOWN ARROW INTO WORD "DOWN".

ON KEY(16) GOSUB CNVDN

ON KEY(18) GOSUB CNVDN

ON KEY(20) GOSUB CNVDN

ON KEY(22) GOSUB CNVDN

KEY OFF

COLOR 15,1

CLS

,

,

'***** DATA DEFINITION *****

,

DIM STACK%(50)

'TABLES DEFINING SYSTEM POINTS

```

MAX.TRND.RNG%=24
NUM.HRS%=1
WARP=1          ' SETS TIME WARP FOR NORMAL FIELD OPERATION.
IF (OFFICE%) THEN WARP=1 'SPEEDS THINGS UP FOR LABORATORY TESTING.
DISPL.POINTS% = 55 'NUMBER OF POINTS ALLOWABLE IN SCRNX.XX.PNT FILES
TREND.POINTS%=55 'ALLOW TRENDING OF MAX OF 55 POINTS
TREND.5MINS%=288 '24 HR TRND CYCLE @5 MIN FIXED TRND INTERVAL / POINT
TRND.PNT.LOC%=0
DIM TRND.TIME!(TREND.5MINS%) 'ACCUM. TRND DATA FOR A MAX. OF 24 HOURS
DIM ANI.SAMTIM!(MAXANI%)
DIM PREV.SAM.TIME!(MAXANI%)
DIM PREV.ANI.SAMTIM!(MAXANI%)
DIM ANI.VALUE!(MAXANI%) 'CURRENT MEASURED VALU
DIM ANI.NEW.VAL%(MAXANI%) 'STATUS OF ANI.VALUE! FOR DISK STORAGE
DIM ANI.PREVAL!(MAXANI%) 'PREVIOUSLY MEASURED VALUE
DIM ANI.MAXVAL!(MAXANI%) 'MAX MEASURED VALUE SINCE START OF EXPERIMENT
DIM ANI.MINVAL!(MAXANI%) 'MIN MEASURED VALUE SINCE START OF EXPERIMENT
DIM ANI.AVERAGE!(MAXANI%) 'AVERAGE MEAS. VALUE SINCE EXPERIMENT START
DIM ANI.VARIANCE!(MAXANI%) 'MEASUREMENT VARIANCE SINCE EXPERIMENT START
DIM ANI.VALSUM!(MAXANI%) 'VALUE SUM
DIM ANI.VALQSUM!(MAXANI%) 'VALUE SQUARED SUM
DIM ANI.MVCOUNT%(MAXANI%) 'SAMPLING COUNTER
DIM ANI.CUMHRAV!(MAXANI%,24)'CUMULATIVE HOURLY AVERAGE
DIM ANI.DLTIM!(MAXANI%)
DIM ANI.PLTIM!(MAXANI%)

```

'SCREEN DISPLAY TABLE DEFINITIONS

```

DIM DISPL.DYE%(DISPL.POINTS%) 'COLOR FOR TEXT OR VALUE TO BE DISPLAYED
DIM DISPL.ROW%(DISPL.POINTS%) 'ROW COORD. FOR DISPLAYED POINT VALUES
DIM DISPL.COLM%(DISPL.POINTS%) 'COLUMN COORD. DISPLAYED POINT VALUE
DIM DISPL.NAME$(DISPL.POINTS%) 'PNT NAMES DISPLAYED ON CURRENT SCREEN
DIM DISPL.LOC%(DISPL.POINTS%) 'PNT VAL LOC FOR CURR SCR REL TO SYS DB
DIM UNITS.LEN%(DISPL.POINTS%) 'LNGTH OF UNITS FOR DISP. ON CURR. SCREEN
DIM TREND.NAME$(TREND.POINTS%)
DIM TREND%(TOT.POINTS%)
DIM TREND.LOC%(TREND.POINTS%)
DIM TREND.BUFFS!(TREND.5MINS%,TREND.POINTS%) 'ASSUME TREND DATA IS
                                                'CAPTURED EVERY MINUTE
                                                'IN A CIRCULAR BUFFER FOR A
                                                'MAX TIME SPAN OF 24 HOURS

DIM SCRTXT.LEN%(DISPL.POINTS%) 'SCREEN TEXT LNGTH (USED FOR DATA DISP.
                                'SCREENS THAT REQUIRE PARAMETER UNITS)

DIM EOPM%(12)
EOPM%(1)=0
EOPM%(2)=31
EOPM%(3)=59
EOPM%(4)=90

```

```

EOPM%(5)=120
EOPM%(6)=151      ' Used in Julian date section
EOPM%(7)=181
EOPM%(8)=212
EOPM%(9)=243
EOPM%(10)=273
EOPM%(11)=304
EOPM%(12)=334
'

CURR.SCREEN%=0
LINE.START%=7
PRINT.COUNT%=1
BACKSPACE%=8
OPTION.LINE%=22
STATUS.LINE%=23
QUERY.LINE%=24
INPUT.LINE%=25
NEXT.CHAR%=1
CARRIAGE.RETURN%=13
BLANK.LINE$=SPACE$(50)
SCR.LAST.LINE%= INPUT.LINE%
CENTER%=40
STACK.SIZE%=50
ON$="ON"
OFF$="OFF"
COMMA$=", "
'

SYSDRIVE$="C:" 'SYSTEM FILES
DATADRIVE$="D:" 'OUTPUT FILES
ALTDRIVE$="E:" 'ALTERNATE DATA DRIVE
EMR.DRIVE$ = "C:" 'EMERGENCY DATA DRIVE
PROGDRIVE$="C:" 'SOURCE FILE(S)
CDRV$="C:"
DDRV$="D:"
EDRV$="E:"
FDRV$="F:"
SCNDRIVE$="C:" 'SCREEN STATIC DATA FILES
DBDRIVE$="C:" 'POINT DATA BASE
SCN.FILE.EXT$=".SCN"
PNT.FILE.EXT$=".PNT"
'

EXIT$="EXIT"
PREV$="PREV"
QUIT$="QUIT"
ANI$="ANI"
TXT$="TXT"
DGI$="DGI"
DGO$="DGO"
ANO$="ANO"
CDEG$="DEGC"
UP$="UP"
DOWN$="DOWN"

```

```
NULL$=""
DATA.FILES = NULL$
```

```
'DEFINE SOME OUTPUT FORMATS:
F7P2$="####.##"
```

```
'DEFINE POSSIBLE INPUT OPTION STRINGS:
```

```
START$="START"
STOP$="STOP"
GRAPH$="GRAPH"
CONTROL$="CONTROL"
HELP$="HELP"
TREND$="TREND"
BAR$="BAR"
PRINT$="PRINT"
DISK$="DISK"
DATA$="DATA"
DIAG$="DIAG"
SELOPT$="SELECT OPTION FROM MENU"
ENTOPT$="ENTER OPTION"
```

```
'DEFINE STATUS LINE CONDITIONS:
```

```
PRINTER.TITLE$="SYSTEM DATA POINT REPORT"
PRINTER.ENABLE$="PRINTING ENABLED "
PRINTER.DISABLE$="PRINTING DISABLED"
DISK.ENABLE$="DISK STORE ENABLED "
DISK.DISABLE$="DISK STORE DISABLED"
PRINT.STATUS$=PRINTER.DISABLE$ 'INITIAL PRINTER STATUS LINE.
DISK.STATUS$=DISK.DISABLE$ 'INITIAL DISK STORAGE STATUS LINE.
SYS.STATUS$ = "SYSTEM STARTED" 'INITIAL SYSTEM STATUS LINE.
STATUS.CHANGE%=TRUE%
```

```
DATA.TITLE$=" UV-B RESEARCH FACILITY ANALOG DATA "
```

```
NUMHDRS%=12      '# of headers in data file
```

```
PDB.STRING$=""
```

```
VAL.STRING$=""
```

```
BLANK$="<BLANK>,"
```

```
EOH$="<EOH>"
```

```
DAYS=MID$(DATE$,4,2)
```

```
MONTHS=MID$(DATE$,1,2)
```

```
YR1$=MID$(DATE$,9,2)
```

```
YR2$=MID$(DATE$,10,1)
```

```
SYS.PDB$="1" 'POINT DATA BASE DESIGNATOR
```

```
DLINTRVL! = 180.0      'Disk data logging interval in seconds.
```

```
PLINTRVL! = 300.0
```

```
MINDLI! = 60.0          'Min. allowable disk logging interval in Sec.
```

```
MAXDLI! = 30000.0       'Min. allowable disk logging interval in Sec.
```

```
MINPLI! = 120.0         'Min. allowable printer logging interval in Sec
```

```
MAXPLI! = 30000.0       'Max. allowable printer logging interval in Sec
```

```
TRNDINTRVL!=60.0*5
```

```
TIME.COUNT%=0
```

```
NO.POINT.DEF%=TRUE%      'First time, display 1st Trend point in file.
```



```

TREND.START%=TRUE%
TREND.GRAPH%=0
TREND.GRPH.START%=0
TREND.FLAG%=0
ERR.PASS%=0
TRND.FLAG%=0
PASS%=0
ADC.LOOP% = 0
,
WRAP% = FALSE%      'FULL TREND BUFFER INDICATOR
AUTO.SCALE% = TRUE% 'SETS Y AXIS TREND SCALE TO (MAX-MIN)*1.2
,
*****
,
*****INITIALIZATION ROUTINE*****

FOR MLOOP%=0 TO MAXANI%
ANI.MAXVAL!(MLOOP%)=-9.E+9
ANI.MINVAL!(MLOOP%)= 9.E+9
NEXT MLOOP%
IF PDB.LOADED% THEN GOTO TRNDBUF

'OPEN & READ POINT DATA BASE CONTAINING POINT DATA
'INTO SYSTEM POINT TABLES
,
CHAIN "PDBUVB"

***** BEGIN CREATION OF TREND BUFFERS *****

TRNDBUF:
FILE.NAME$ ="SCRN008.PNT"
GOSUB LDDYNPNT
NTPNTS% = NDPNTS%
'REDIM TREND.BUFFS!(TREND.5MINS%,NTPNTS%)
FOR TLOOP% = 1 TO NTPNTS%
TREND.NAME$(TLOOP%) = DISPL.NAME$(TLOOP%)
TREND.LOC%(TLOOP%) = DISPL.LOC%(TLOOP%)
'LPRINT TREND.NAME$(TLOOP%);" ";TREND.LOC%(TLOOP%)
NEXT TLOOP%

***** END CREATION OF TREND BUFFERS

***** BEGIN METRABYTE HARDWARE DEFFINITION *****

DASH8BASE%=&H300      'SET BASE ADDRESS FOR ANALOG INPUT CARD.
DDAO6BASE%=&H350 'SET BASE ADDRESS FOR ANALOG OUTPUT CARD.
START12BIT%=0        'DEFINE COMMAND TO START A/D CONVERSION.

DIM GAIN.EXP!(2)
GAIN.EXP!(1)=1.0      'SET GAIN'S FOR ALL OF THE EXP-16 CARDS
GAIN.EXP!(2)=1.0      'USED IN THIS SYSTEM

```

```
NUM.SAMPS% = 5          'TAKE 5 READINGS FROM A/D (AVERAGE MIDDLE 3).
DIM CNTS%(NUM.SAMPS%)
```

```
'***** END METRABYTE HARDWARE DEFFINITION *****
'
```

```
'***** BEGIN UVB LAMP AND FILAMENT CONTROL DEFFINITION'S *****
'
```

```
DIM CNTLOUT(NUM.ANO%) ' DIMENSION CONTROLLER OUPUT ARRAY
```

```
KP.LAMP = 0.3          ' INITIAL PROPORTIONAL CONTROL PARAMETER
```

```
KI.LAMP = 0.05         ' INITIAL INTEGRAL CONTROL PARAMETER
```

```
WNDUPLMT= 25.0         ' INTEGRAL WINDUP LIMIT
```

```
' INITIAL UVB CONTROLLER SETPOINT PERCENTAGE
```

```
' ABOVE AMBIENT SOLAR UVB FLUX (DEFAULT)
```

```
' CONTAINED IN DATA BASE FILE (UVB.PDB)
```

```
FOR CTL.CNT% = 1 TO NUM.ANO%
```

```
  IF PNT.NAMES$(ANO.TBL%(CTL.CNT%)) = "LAMPFLUX" THEN
```

```
    FLUX.PCT.ABOVE.AMB = ANO.SP(CTL.CNT%)
```

```
  END IF
```

```
NEXT CTL.CNT%
```

```
CPPCT = 40.95          ' CONTROLLER OUTPUT A/D GAIN FACTOR
```

```
T = 0.5                ' SCAN RATE INTERVAL
```

```
FILAMENT.TURN.ON = .01
```

```
ARCPLASMA.TURN.ON = .20
```

```
Q.AMBIENT = 1.0        ' INITIALIZE 1ST TIME TO PREVENT DIVISION BY 0
```

```
'***** END UVB LAMP AND FILAMENT CONTROL DEFFINITION'S *****
'
```

```
'***** BEGIN COSINE CORRECTION TERMS FOR UVB POLYCHROMATIC SENSOR *****
'
```

```
  LAT.DEG = 36.01      ' OAK RIDGE LATITUDE
```

```
  LONG.DEG = 84.14     ' OAK RIDGE LONGITUDE
```

```
  H.MIN = 0.0          ' SOLAR TIME IN MINUTES
```

```
  DEC.ANG = 0.0        ' DECLINATION ANGLE OF SUN
```

```
  ZENITH.ANG = 0.0     ' SOLAR ZENITH ANGLE
```

```
  DLST = 1.0           ' DAY LIGHT SAVINGS TIME
```

```
  EQU.TIM = 0.0        ' EQUATION OF TIME (PERTUBATIONS IN
                        ' EARTH'S RATE OF ROTATION
```

```
  PI = 3.14159265      ' VALUE OF PI
```

```
  RAD.CNV = PI/180     ' PI RADIANS / 180 DEGREE'S
```

```
  DEG.FROM.NOON = 0.0  ' # OF DEGREES FROM NOON
```

```
'***** END COSINE CORRECTION TERMS FOR UVB POLYCHROMATIC SENSOR *****
'
```

```
'***** END OF DATA INITIALIZATION *****
'
```

```
'***** BEGIN "IDLE-LOOP" *****
```

```

ENTRY: CLS
      PROGRAM.END%=FALSE%
      SYS.RUN%=TRUE%
      FOR ANO.PTR% = 1 TO NUM.ANO%
        ANO.MAN.OUTPUT%(ANO.PTR%) = FALSE%
      NEXT ANO.PTR%
      DISK.LOG% = FALSE%
      '

IDLE:
      ON ERROR GOTO ERRHANDLER
      IF PROGRAM.END% THEN GOSUB 54151

IDLE1: IF SYS.RUN% THEN
      GOSUB METRA.SCAN 'SERVICE ANALOG INPUTS
      GOSUB TREND.DATA.CAPTURE
      GOSUB CONTROLLER.OUTPUT

      'GOSUB      'SERVICE DIGITAL INPUTS
      'GOSUB      'PERFORM DIGITAL OUTPUTS
      ELSE
      GOSUB 60040
      END IF

      GOSUB DATTIM 'TIME & DATE UPDATE
      GOSUB JULIE 'EXECUTE ROUTINE TO DETERMINE JULIAN DATE

      'POWER FAIL CHECK ??

      'INITIATE WATCHDOG TIMER IF DA & C HAS STARTED ??

      IF CURR.SCREEN% = 0 THEN GOSUB SCRN0   'Main Menu
      IF CURR.SCREEN% = 1 THEN GOSUB SCRN1   'GRAPHICS
      IF CURR.SCREEN% = 2 THEN GOSUB SCRN2   'SYSTEM DIAGRAM
      IF CURR.SCREEN% = 3 THEN GOSUB SCRN3   'Printer Logging
      IF CURR.SCREEN% = 5 THEN GOSUB SCRN5   'Disk Storage
      IF CURR.SCREEN% = 6 THEN GOSUB SCRN6   'Control
      IF CURR.SCREEN% = 8 THEN GOSUB SCRN8   'Graphics Trending
      IF CURR.SCREEN% = 13 THEN GOSUB SCRN13 'Bar Graph's
      IF CURR.SCREEN% = 15 THEN GOSUB SCRN15 'Load Micristarr profiles
      IF CURR.SCREEN% = 16 THEN GOSUB SCRN16 'Control Parameter Display
      IF CURR.SCREEN% = 17 THEN GOSUB SCRN17 'Change Cntr. SP Display
      IF CURR.SCREEN% = 18 THEN GOSUB SCRN18 'Change Cntr. GAIN Display
      IF CURR.SCREEN% = 19 THEN GOSUB SCRN19 'Change Cntr. RESET Display
      IF CURR.SCREEN% = 20 THEN GOSUB SCRN20 'Change Cntr. Mode Display
      IF CURR.SCREEN% = 21 THEN GOSUB SCRN21 'Change Cntr. Output %
      IF CURR.SCREEN% = 25 THEN GOSUB SCRN25 'All Points Display

      IF PRNT.LOG% THEN GOSUB LOGPRINT 'PRINT LOG CHECK
      IF DISK.LOG% THEN GOSUB LOGDISK 'DISK LOG CHECK
      INPUT.STRING$=""
      GOSUB KEYBD 'KEYBOARD INPUT HANDLER

      'UPDATE SYSTEM STATUS LINE

```

```

IF STATUS.CHANGE% OR SCREEN.CHANGE% THEN GOSUB STATLN

GOTO IDLE

'***** END OF "IDLE-LOOP" *****
,
'***** KEYBOARD INPUT HANDLER *****

KEYBD: 'ENTRY POINT
      IF INPUT.PROMPT% GOTO KBINS
      LOCATE INPUT.LINE%,1
      PRINT "INPUT: ";
      INPUT.PROMPT%=TRUE%
KBINS: KB$=INKEY$
      IF LEN(KB$) <> 0 GOTO CHKBS
      GOTO EXITKB
CHKBS: IF ASC(KB$) <> BACKSPACE% GOTO CHKCR
,
      'INCREMENT BACKSPACE COUNTER
      ,
      BACK.CNT%=BACK.CNT%+1
      BACK%=TRUE%
      IF NEXT.CHAR% = 1 GOTO EXITKB
      NEXT.CHAR%=NEXT.CHAR%-1
      LOCATE INPUT.LINE%,LINE.START%+NEXT.CHAR%
      PRINT " ";
      ,
      'DEFINE NEW INPUT BUFFER DISCARDING "BACKSPACED OVER" CHARACTERS
      'FROM PREVIOUS INPUT BUFFER
      ,
      INPUT.BUFFER$=LEFT$(OLD.BUFF$,BUFF.LEN%-BACK.CNT%)
      GOTO EXITKB
CHKCR: IF ASC(KB$) <> CARRIAGE.RETURN% GOTO ADDCHR
      BACK.CNT%=0 'RESET BACKSPACE COUNTER FOR NEXT INPUT STRING
      LINE.LENGTH%=LEN(INPUT.BUFFER$)
      INPUT.STRING$=INPUT.BUFFER$ 'CAPTURE PREVIOUSLY INPUT CHARACTERS
      ,
      'DETERMINATION OF INPUT.STRING$ WHEN INKEY$=<CR> (INPUT.BUFFER$="")
      ,
      IF INPUT.BUFFER$="" THEN INPUT.STRING$=KB$
      INPUT.BUFFER$="" 'CLEAR INPUT BUFFER FOR NEXT INPUT
      NEXT.CHAR%=1 'RESET CHARACTER COUNTER
      LOCATE INPUT.LINE%,LINE.START%+NEXT.CHAR%:PRINT BLANK.LINE$;
      GOTO EXITKB
ADDCHR: IF ASC(KB$) > 96 AND ASC(KB$) < 123 THEN KB$ = CHR$(ASC(KB$) - 32)
      INPUT.BUFFER$=INPUT.BUFFER$+KB$
      ,
      'KEEP TRACK OF INPUT BUFFER LENGTH
      ,
      BUFF.LEN%=LEN(INPUT.BUFFER$)
      ,
      'RESET BACKSPACE COUNTER TO ZERO BECAUSE OF CHANGE FROM BACKSPACING TO
      'CHARACTER INPUT FOR OPERATOR INPUT

```

```

IF BACK% THEN BACK.CNT%=0
BACK%=FALSE%
OLD.BUFF$=INPUT.BUFFER$ 'CAPTURE INPUT BUFFER
LOCATE INPUT.LINE%,LINE.START%+NEXT.CHAR%
PRINT KB$;
NEXT.CHAR%=NEXT.CHAR%+1
EXITKB: RETURN

' ***** END OF KEYBOARD INPUT HANDLER *****
'
'*****BEGIN DATE & TIME TASK *****

DATTIM:
LOCATE 1,73: PRINT TIME$;
LOCATE 1,1: PRINT DATE$;
RETURN

'***** END OF DATE & TIME TASK *****
'
'***** SCREEN 0 - MASTER *****

SCRN0:
IF SCREEN.STATIC% GOTO SCRNO0DYN
FILE.NAME$="SCRN000.SCN"   'NAME OF FILE HOLDING STATIC INFO.
SCREEN 0
COLOR 15,1
SCRN.COLOR% = 15
CLS
INPUT.PROMPT% = FALSE%
GOSUB SHWSTATIC           'GO TO ROUTINE TO PUT INFO ON SCREEN.
SCREEN.STATIC% = TRUE%
QUERY$=SELOPT$
GOSUB QUERYLN             'PUT UP THE QUERY LINE.
SCRNO0DYN:
'
'INTERPRET INPUT STRING
'
SCRNO0INP:
IF INPUT.STRING$ = NULL$ GOTO SCRNO0RTN
IF INPUT.STRING$ <> "START" GOTO CHKSTOP
SYS.RUN%=TRUE%
SYS.STATUS$="SYSTEM RUNNING"
STATUS.CHANGE%=TRUE%
INVALID.QUERY%=FALSE%
GOTO SCRNO0RTN
CHKSTOP:
IF INPUT.STRING$ <> "STOP" GOTO CHKGRAPH
'IF A SYSTEM IS STOPPED SYSTEM PDB TABLES MUST BE CLEARED
SYS.RUN%=FALSE%
SYS.STATUS$="SYSTEM STOPPED"
STATUS.CHANGE%=TRUE%
INVALID.QUERY%=FALSE%

```

```

      GOTO SCRNO0RTN
      'CHECK FOR GRAPHICS OPTION.
CHKGRAPH:
      IF INPUT.STRING$ <> GRAPH$ GOTO CHKCNTL
      NEW.SCREEN%=1
      GOSUB NEWSN
      GOTO SCRNO0RTN
CHKCNTL:
      IF INPUT.STRING$ <> CONTROL$ GOTO CHKPRINT
      NEW.SCREEN%=16
      GOSUB NEWSN
      GOTO SCRNO0RTN

CHKPRINT:
      IF INPUT.STRING$ <> PRINT$ GOTO CHKDISK
      NEW.SCREEN%=3
      GOSUB NEWSN
      GOTO SCRNO0RTN
CHKDISK:
      IF INPUT.STRING$ <> DISK$ GOTO CHKDATA
      NEW.SCREEN%=5
      GOSUB NEWSN
      GOTO SCRNO0RTN
CHKDATA:
      IF INPUT.STRING$ <> DATA$ GOTO SCRNO02
      NEW.SCREEN%=25
      GOSUB NEWSN
      GOTO SCRNO0RTN

SCRNO02:
      GOSUB INSTRINTP      'Input String Interpreter
      IF STRING.INTERP% THEN GOSUB 35170      'Input Error Handler
SCRNO0RTN:
      RETURN

***** END OF SCREEN 0 *****
,
***** SCREEN 1 - GRAPHICS *****

SCRN1:
      IF SCREEN.STATIC% GOTO SCRN1DYN
      FILE.NAME$="SCRNO01.SCN"
      SCREEN 0
      COLOR 15,1
      SCRNO.COLOR% = 15
      CLS
      INPUT.PROMPT% = FALSE%
      GOSUB SHWSTATIC      'Screen Static Info
      SCREEN.STATIC% = TRUE%
      QUERY$=SELOPT$
      GOSUB QUERYLN      'Query Line Generator
SCRN1DYN:
      ,

```

'INTERPRET INPUT STRING

SCRN1INP:

IF INPUT.STRING\$ <> TRENDS\$ GOTO CHKBAR1
NEW.SCREEN%=8
GOSUB NEWSN
GOTO SCRNI1RTN

CHKBAR1:

IF INPUT.STRING\$ <> BAR\$ GOTO CHKDIAG1
NEW.SCREEN%=13
GOSUB NEWSN
GOTO SCRNI1RTN

CHKDIAG1:

IF INPUT.STRING\$ <> DIAG\$ GOTO SCRNI1INP1
NEW.SCREEN%=2
GOSUB NEWSN
GOTO SCRNI1RTN

SCRNI1INP1:

GOSUB INSTRINTP 'Input String Interpreter
IF STRING.INTERP% THEN GOSUB 35170 'Input Error Handler

SCRNI1RTN:

RETURN

***** END OF SCREEN 1 *****

***** SCREEN 2 - UVB FACILITY OVERVIEW GRAPHICS *****

SCRN2:

IF SCREEN.STATIC% GOTO SCRNI2DYN
SCREEN 9
COLOR 15,1
SCRN.COLOR% = 15
CLS
INPUT.PROMPT% = FALSE%

'DRAW CONTROL CABINET

LINE (176,224) - (264,224),15 'CONTROL SYSTEM BOX
LINE (177,225) - (263,279),9,BF
LINE (176,224) - (176,280),15
LINE (176,280) - (264,280),15
LINE (264,280) - (264,224),15
LINE (176,224) - (186,214),15
LINE (264,224) - (274,214),15
LINE (186,214) - (274,214),15
LINE (274,214) - (274,270),15
LINE (264,280) - (274,270),15

'DRAW FILAMENT CABINET

'FILAMENT CABINET

LINE (336,206) - (424,206),15
LINE (337,207) - (423,244),9,BF
LINE (336,206) - (336,244),15
LINE (336,244) - (424,244),15

LINE (424,244) - (424,206),15
 LINE (336,206) - (346,200),15
 LINE (346,200) - (434,200),15
 LINE (434,200) - (424,206),15
 LINE (434,200) - (434,228),15
 LINE (434,228) - (424,244),15

'DRAW ARC POWER

LINE (336,260) - (336,296),15
 LINE (337,261) - (423,296),9,BF
 LINE (336,296) - (424,296),16
 LINE (336,260) - (424,260),15
 LINE (424,292) - (424,260),15
 LINE (336,260) - (346,250),15
 LINE (346,250) - (434,250),15
 LINE (434,250) - (424,260),15
 LINE (434,250) - (434,276),15
 LINE (434,276) - (424,296),15

'LINES CONNECTING CABINETS

LINE (428,220) - (480,220),3
 LINE (480,220) - (472,216),3
 LINE (480,220) - (472,224),3
 LINE (428,276) - (480,276),3
 LINE (480,276) - (472,272),3
 LINE (480,276) - (472,280),3

'FIL. POWER TO LAMPS

'ARC POWER TO LAMPS

LINE (269,220) - (336,220),3
 LINE (269,268) - (336,268),3
 LINE (176,266) - (60,266),3
 LINE (60,266) - (60,145),3
 CIRCLE (60,142),3,5
 CIRCLE (60,142),1,5

'CONTROL TO FIL. POWER

'CONTROL TO ARC POWER

'AIR TEMP.

' " "

' " "

' " "

' LINE (220,145) - (220,168),3
 ' LINE (220,168) - (88,168),3
 ' LINE (88,168) - (88,252),3
 ' LINE (88,252) - (176,252),3
 ' CIRCLE (220,142),3,5
 ' CIRCLE (220,142),1,5

'EXPOSURE FLUX

LINE (400,145) - (400,178),3
 LINE (400,178) - (120,178),3
 LINE (120,178) - (120,238),3
 LINE (120,238) - (176,238),3
 CIRCLE (400,142),3,5
 CIRCLE (400,142),1,5

'CONTROL FLUX

LINE (550,145) - (550,186),3
 LINE (550,186) - (152,186),3
 LINE (152,186) - (152,224),3
 LINE (152,224) - (176,224),3
 CIRCLE (550,142),3,5
 CIRCLE (550,142),1,5

'AMBIENT FLUX

'DRAW LAMP RACKS

LINE (176,98) - (248,56),15
 LINE (248,56) - (320,56),15
 LINE (176,98) - (256,98),15
 LINE (176,98) - (176,140),15,
 LINE (256,98) - (256,140),15,
 LINE (256,98) - (320,56),15,
 LINE (257,99) - (320,58),15
 LINE (256,99) - (320,57),15
 LINE (320,56) - (320,98),15

' LEFT LIGHT FRAME

.....

.....

LINE (360,98) - (432,56),15
 LINE (360,98) - (440,98),15
 LINE (441,99) - (504,58),15
 LINE (441,98) - (504,57),15
 LINE (432,56) - (504,56),15
 LINE (440,98) - (504,56),15
 LINE (360,98) - (360,140),15
 LINE (440,98) - (440,140),15
 LINE (504,56) - (504,98),15

'RIGHT LIGHT FRAME

.....

.....

'DRAW SUN

CIRCLE (50,50),20,14,CF
 CIRCLE (50,50),19,14
 CIRCLE (50,50),18,14
 CIRCLE (50,50),17,14
 CIRCLE (50,50),16,14
 CIRCLE (50,50),15,14
 CIRCLE (50,50),14,14
 CIRCLE (50,50),13,14
 CIRCLE (50,50),12,14
 CIRCLE (50,50),11,14
 CIRCLE (50,50),10,14
 CIRCLE (50,50),9,14
 CIRCLE (50,50),8,14
 CIRCLE (50,50),7,14
 CIRCLE (50,50),6,14
 CIRCLE (50,50),5,14
 CIRCLE (50,50),4,14
 CIRCLE (50,50),3,14
 CIRCLE (50,50),2,14
 CIRCLE (50,50),1,14
 FOR D=25 TO 360 STEP 25
 DRAW "TA="+VARPTR\$(D)+"NU35"
 NEXT D

'LAMPS

LINE (184,94) - (260,94),3
 LINE (182,95) - (259,95),3
 LINE (180,96) - (258,96),3
 LINE (275,84) - (202,84),3

'LAMPS IN LEFT RACK

LINE (273,85) - (200,85),3
 LINE (272,86) - (198,86),3
 LINE (226,69) - (299,69),3
 LINE (224,70) - (297,70),3
 LINE (223,71) - (295,71),3
 LINE (246,58) - (315,58),3
 LINE (244,59) - (314,59),3
 LINE (243,60) - (312,60),3
 LINE (364,96) - (440,96),3
 LINE (366,95) - (441,95),3
 LINE (368,94) - (443,94),3
 LINE (383,85) - (457,85),3
 LINE (384,84) - (458,84),3
 LINE (386,83) - (460,83),3
 LINE (407,71) - (479,71),3
 LINE (408,70) - (480,70),3
 LINE (410,69) - (482,69),3
 LINE (430,58) - (500,58),3
 LINE (428,59) - (498,59),3
 LINE (427,60) - (497,60),3

'LAMPS IN RIGHT RACK

'DRAW PLANTS

LINE (296,98) - (304,105),11,BF
 LINE (300,98) - (300,90),10
 LINE (294,90) - (306,90),10
 LINE (294,90) - (300,80),10
 LINE (306,90) - (300,80),10
 CIRCLE (300,87),3,10
 CIRCLE (300,87),1,10
 LINE (280,119) - (288,112),11,BF
 LINE (284,112) - (284,104),10
 LINE (278,104) - (290,104),10
 LINE (278,104) - (284,94),10
 LINE (290,104) - (284,94),10
 CIRCLE (284,101),3,10
 CIRCLE (284,101),1,10
 LINE (272,126) - (264,133),11,BF
 LINE (268,126) - (268,119),11
 LINE (262,119) - (274,119),10
 LINE (262,119) - (268,109),10
 LINE (274,119) - (268,109),10
 CIRCLE (268,116),3,10
 CIRCLE (268,116),1,10
 LINE (448,126) - (456,133),11,BF
 LINE (452,126) - (452,119),10
 LINE (446,119) - (458,119),10
 LINE (446,119) - (452,109),10
 LINE (458,119) - (452,109),10
 CIRCLE (452,116),3,10
 CIRCLE (452,116),1,10
 LINE (464,112) - (472,119),11,BF
 LINE (468,112) - (468,105),10
 LINE (462,105) - (474,105),10

```

LINE (462,105) - (468,95),10
LINE (474,105) - (468,95),10
CIRCLE (468,102),3,10
CIRCLE (468,102),1,10
LINE (480,98) - (488,105),11,BF
LINE (484,98) - (484,91),10
LINE (478,91) - (490,91),10
LINE (478,91) - (484,81),10
LINE (490,91) - (484,81),10
CIRCLE (484,88),3,10
CIRCLE (484,88),1,10
CIRCLE (484,86),1,10

```

```
COLOR SCR.N.COLOR%
```

```

FILE.NAME$="SCRN002.SCN"
GOSUB SHWSTATIC
FILE.NAME$ = "SCRN002.PNT"
GOSUB LDDYNPNT      'FIND POINT NAMES IN THE DATABASE.
SCREEN.STATIC%=TRUE%
QUERY$ = ENTOPT$
GOSUB QUERYLN
'UPDATE DYNAMIC DATA ON SCREEN.

```

```
SCRN2DYN:
```

```
    GOSUB DSPDYNPNT      'PUT POINT VALUES ON SCREEN.
```

```
SCRN2INP:
```

```
    GOSUB INSTRINTP      'Input String Interpreter
    IF STRING.INTERP% THEN GOSUB 35170      'Input Error Handler

```

```
SCRN2RTN:
```

```
    RETURN
```

```
*****END OF SCREEN 2 - CHAMBER GRAPHICS*****
```

```
,
```

```
***** SCREEN 3 - PRINTER LOGGING *****
```

```
SCRN3:
```

```

IF SCREEN.STATIC% GOTO SCR3DYN
FILE.NAME$ = "SCRN003.SCN"
SCREEN 0
COLOR 15,1
SCR.N.COLOR% = 15
CLS
INPUT.PROMPT% = FALSE%
GOSUB SHWSTATIC      'Screen Static Info
SCREEN.STATIC%=TRUE%
QUERY$=SELOPT$
GOSUB QUERYLN      'Query Line Generator

```

```
SCRN3DYN:
```

```

LOCATE 15,15
PRINT "PRINTER LOGGING INTERVAL = ";PLINTRVL/60.0;" MINUTES"

```

```

LOCATE 16,15
PRINT "(ALLOWABLE INTERVAL, MIN. = ";MINPLI//60.0;" ,MAX. = " ,_
    MAXPLI//60.0;" );
LOCATE 17,15
PRINT PRINT.STATUS$

IF INPUT.STRING$ = NULL$ THEN GOTO SCR3RTN

IF((CARRIAGE.RETURN%=ASC(INPUT.STRING$))AND(AWAIT.INT%=TRUE%)) THEN
    AWAIT.INT% = FALSE%
    SCREEN.STATIC% = FALSE%
    GOTO SCR3RTN
END IF

IF PRNT.LOG% THEN GOTO CHKNLOG

IF INPUT.STRING$ <> "LOG" GOTO CHKNLOG
PRNT.LOG%=TRUE%
PRINT.STATUS$ = PRINTER.ENABLE$
STATUS.CHANGE%=TRUE%
GOTO SCR3RTN
CHKNLOG:
IF INPUT.STRING$ <> "NLOG" GOTO SCR3INP1
PRINT.STATUS$ = PRINTER.DISABLE$
STATUS.CHANGE%=TRUE%
PRNT.LOG%=FALSE%
GOTO SCR3RTN

SCR3INP1:
IF AWAIT.INT% THEN GOTO CHKPINT
IF INPUT.STRING$ <> "LOGIN" GOTO SCR3INP2
QUERY$ = "ENTER PRINT LOGGING INTERVAL IN MINUTES"
GOSUB QUERYLN
AWAIT.INT% = TRUE%
GOTO SCR3RTN
CHKPINT:
PLINT! = VAL(INPUT.STRING$) * 60.0
IF PLINT! < MINPLI! OR PLINT! > MAXPLI! THEN GOTO SCR3INP2
PLINTRVL! = INT(PLINT!)
AWAIT.INT% = FALSE%
SCREEN.STATIC% = FALSE%
INPUT.STRING$ = NULL$
GOTO SCR3RTN

SCR3INP2:
GOSUB INSTRINTP      'Input String Interpreter
IF STRING.INTERP% THEN GOSUB 35170 'Input Error Handler
SCR3RTN:
RETURN

```

***** END OF SCREEN 3 *****

***** SCREEN 4 - PRESENTLY NOT USED *****

SCRN4:

RETURN

***** END OF SCREEN 4 *****

***** SCREEN 5 - DISK STORE *****

SCRN5:

```
IF SCREEN.STATIC% GOTO SCR5DYN
FILE.NAME$ = "SCRN005.SCN"
SCREEN 0
COLOR 15,1
SCRN.COLOR% = 15
CLS
INPUT.PROMPT% = FALSE%
OLD.DSTATUS$ = NULL$
GOSUB SHWSTATIC      'Screen Static Info
SCREEN.STATIC%=TRUE%
QUERY$=SELOPT$
GOSUB QUERYLN       'Query Line Generator
```

SCRN5DYN:

```
LOCATE 15,15
PRINT "ACTIVE DATA STORAGE DRIVE = ";DATADRVES$;
LOCATE 16,15
PRINT "ALTERNATE DATA STORAGE DRIVE = ";ALTDRVES$;
LOCATE 17,15
PRINT "EMERGENCY DATA STORAGE DRIVE = ";EMR.DRVES$;
LOCATE 18,15
PRINT "DISK STORAGE INTERVAL = ";DLINTRVL//60.0;" MINUTES"
LOCATE 19,15
PRINT "(ALLOWABLE INTERVAL, MIN. = ";MINDLI//60.0;" ,MAX. = ";_
      MAXDLI//60.0;" )";

IF OLD.DSTATUS$ = DISK.STATUS$ THEN GOTO SCR5INP3
LOCATE 20,01
PRINT SPACES$(79);
LOCATE 20,15
PRINT "PRESENT DISK LOGGING FILE = ";
IF DISK.STATUS$ = DISK.ENABLE$ THEN
  PRINT DATA.FILES$;
  OLD.DSTATUS$ = DISK.ENABLE$
ELSE
  PRINT DISK.DISABLE$
  OLD.DSTATUS$ = DISK.DISABLE$
END IF
```

SCRN5INP3:

```
IF INPUT.STRING$ = NULL$ THEN GOTO SCR5RTN
```

```
IF((CARRIAGE.RETURN%=ASC(INPUT.STRING$))AND(AWAIT.DRV%=TRUE%)) THEN
  AWAIT.DRV% = FALSE%
  SCREEN.STATIC% = FALSE%
  GOTO SCRN5RTN
END IF
```

```
IF DISK.LOG% THEN GOTO CHKNSTOR
```

```
IF INPUT.STRING$ <> "STOR" GOTO CHKNSTOR
DISK.LOG%=TRUE%
DISK.STATUS$ = DISK.ENABLE$
STATUS.CHANGE%=TRUE%
GOTO SCRN5RTN
```

```
CHKNSTOR:
```

```
IF INPUT.STRING$ <> "NSTOR" GOTO SCRN5INP
DISK.STATUS$ = DISK.DISABLE$
STATUS.CHANGE%=TRUE%
DISK.LOG%=FALSE%
CLOSE #3
GOTO SCRN5RTN
```

```
SCRN5INP:
```

```
IF AWAIT.DRV% GOTO CHKDASN
IF INPUT.STRING$ <> "DSKASS" GOTO SCRN5INP1
QUERY$ = "ENTER DISK DRIVE ASSIGNMENT -- VALID DRIVES ARE D OR E"
GOSUB QUERYLN
AWAIT.DRV% = TRUE%
GOTO SCRN5RTN
```

```
CHKDASN:
```

```
IF LEN(INPUT.STRING$) > 1 THEN GOTO SCRN5INP1
IF INPUT.STRING$ = "e" THEN INPUT.STRING$ = "E"
IF INPUT.STRING$ = "d" THEN INPUT.STRING$ = "D"
IF INPUT.STRING$ <> "E" AND INPUT.STRING$ <> "D" THEN GOTO SCRN5INP1
```

```
IF INPUT.STRING$ = "E" AND DATADRIVE$ = DDRV$ THEN
  DATADRIVE$ = EDRV$
  ALTDRIVE$ = DDRV$
  DISK.HDR% = FALSE%
  AWAIT.DRV% = FALSE%
  SCREEN.STATIC% = FALSE%
  INPUT.STRING$ = NULL$
  GOTO SCRN5RTN
END IF
```

```
IF INPUT.STRING$ = "D" AND DATADRIVE$ = EDRV$ THEN
  DATADRIVE$ = DDRV$
  ALTDRIVE$ = EDRV$
  DISK.HDR% = FALSE%
  AWAIT.DRV% = FALSE%
  SCREEN.STATIC% = FALSE%
  INPUT.STRING$ = NULL$
  GOTO SCRN5RTN
```

END IF

```
IF INPUT.STRING$ = "D" AND DATADRIVES$ = DDRV$ OR_
  INPUT.STRING$ = "E" AND DATADRIVES$ = EDRV$ THEN
  AWAIT.DRV% = FALSE%
  INPUT.STRING$ = NULL$
  SCREEN.STATIC% = FALSE%
  GOTO SCRNS5RTN
END IF
```

SCRNS5INP1:

```
IF AWAIT.INT% THEN GOTO CHKDINT
IF INPUT.STRING$ <> "DSKINT" GOTO SCRNS5INP2
QUERY$ = "ENTER DISK LOGGING INTERVAL IN MINUTES"
GOSUB QUERYLN
AWAIT.INT% = TRUE%
GOTO SCRNS5RTN
```

CHKDINT:

```
DLINT! = VAL(INPUT.STRING$) * 60.0
IF DLINT! < MINDLI! OR DLINT! > MAXDLI! THEN GOTO SCRNS5INP2
DLINTRVL! = INT(DLINT!)
AWAIT.INT% = FALSE%
SCREEN.STATIC% = FALSE%
GOTO SCRNS5RTN
```

SCRNS5INP2:

```
GOSUB INSTRINTP 'Input String Interpreter
IF STRING.INTERP% THEN GOSUB 35170 'Input Error Handler
```

SCRNS5RTN:

RETURN

***** END OF SCREEN 5 *****

,

***** SCREEN 6 - CONTROL *****

SCRN6:

RETURN

***** END OF SCREEN 6 *****

,

***** SCREEN 7 - PRESENTLY NOT USED *****

SCRN7:

RETURN

***** END OF SCREEN 7 *****

,

,

***** SCREEN 13 - BAR GRAPH DISPLAYS

,

```

SCRN13:
  ON ERROR GOTO ERRHANDLER
  IF SCREEN.STATIC% GOTO SCR13DYN

  '
  FILE.NAMES$="SCRN013.SCN"
  SCREEN 0
  CLS
  SCREEN 9
  COLOR 1,7
  SCR13.COLOR% = 1
  CLS
  INPUT.PROMPT% = FALSE%
  GOSUB SHWSTATIC
  COLOR 1,7
  QUERY$ = ENTOPT$
  GOSUB QUERYLN      'Query Line Generator
  '
  'OPEN GROUP NAMES FILE FOR SCREEN AND READ
  '
  IF GTME% = FALSE THEN
    GROUP$ = "GROUP.001"
    AUTO.BSCALE% = TRUE%
    GTME% = TRUE%
  END IF

  FILE.NAMES$ = GROUP$
  GOSUB LDDYNPNT      'LOAD THE DYNAMIC POINTS FOR THIS SCREEN.
  NGPNTS% = NDPNTS%

  ROW.OFFSET% = 42
  BAR.WIDTH% = 14
  PNT.BAR.SPAN% = 28
  COL.START% = 100
  COL.END% = 600
  NAM.ROW% = 4
  DIM SCALE.OLD%(9)

  FOR GLOOP% = 1 TO NGPNTS%
    LOCATE NAM.ROW%,1
    PRINT DISPL.NAMES$(GLOOP%)
    NAM.ROW% = NAM.ROW% + 2
    SCALE.OLD%(GLOOP%) = COL.START%
  NEXT GLOOP%

  ' Put up the scale for the bar graph

  IF AUTO.BSCALE% THEN

    BARSC.MIN! = 9E+9
    BARSC.MAX! = 9E-9
    FOR GLOOP% = 1 TO NGPNTS%

```



```

IF ANI.MINVAL!(TYPE.PNTR%(DISPL.LOC%(GLOOP%))) < BARSC.MIN! THEN
  BARSC.MIN! = ANI.MINVAL!(TYPE.PNTR%(DISPL.LOC%(GLOOP%)))
END IF
IF ANI.MAXVAL!(TYPE.PNTR%(DISPL.LOC%(GLOOP%))) > BARSC.MAX! THEN
  BARSC.MAX! = ANI.MAXVAL!(TYPE.PNTR%(DISPL.LOC%(GLOOP%)))
END IF
NEXT GLOOP%
GSPAN! = (BARSC.MAX! - BARSC.MIN!)*1.2
IF GSPAN! < 1.0 THEN GSPAN! = 1.0
BARSC.MIN! = BARSC.MAX! - GSPAN!
BARSC.MID! = BARSC.MIN! + GSPAN!/2.0

```

```
ELSE
```

```

  BARSC.MAX! = BARSC1.MAX!
  BARSC.MIN! = BARSC1.MIN!
  GSPAN! = BARSC.MAX! - BARSC.MIN!
  BARSC.MID! = BARSC.MIN! + GSPAN!/2.0
END IF

```

```

LOCATE 2,1
COLOR 12
PRINT "UNITS:";ANI.EUNIT$(TYPE.PNTR%(DISPL.LOC%(1)));
LOCATE 3,11
COLOR 8
PRINT USING "####.#";BARSC.MIN!;
LOCATE 3,41
PRINT USING "####.#";BARSC.MID!;
LOCATE 3,72
PRINT USING "####.#";BARSC.MAX!;
COLOR 1

```

```
'Draw the Black border
```

```

LINE (COL.START%-1,ROW.OFFSET%-1)-(COL.END%+1,ROW.OFFSET%-1),8
LINE (COL.END%+1,ROW.OFFSET%-1)-(COL.END%+1,ROW.OFFSET%_
  +PNT.BAR.SPAN%*8+BAR.WIDTH%+1),8
LINE (COL.END%+1,ROW.OFFSET%+PNT.BAR.SPAN%*8+BAR.WIDTH%+1)-_
  (COL.START%-1,ROW.OFFSET%+PNT.BAR.SPAN%*8+BAR.WIDTH%+1),8
LINE (COL.START%-1,ROW.OFFSET%+PNT.BAR.SPAN%*8+BAR.WIDTH%+1)-_
  (COL.START%-1,ROW.OFFSET%-1),8

```

```
'Draw tic marks on top of graph
```

```

TIC.BAR% = FIX((COL.END% - COL.START% + 2) / 4)
FOR T% = 0 TO 4
  COL.B% = TIC.BAR%*T% + 100
  LINE (COL.B%,ROW.OFFSET%-1)-(COL.B%,ROW.OFFSET%-4),8
NEXT T%

```

```
SCREEN.STATIC%=TRUE%
```

```
'SCREEN DYNAMIC OUTPUT
```

SCRN13DYN:

IF NOT (OK.TRND1% AND OK.TRND2%) THEN GOTO SCR13INP1

FOR GLOOP% = 1 TO NGPNTS%

SCALE! = (500.0*(ANI.VALUE(TYPE.PNTR%(DISPL.LOC%(GLOOP%)))-BARSC.MIN!)/
/GSPAN!)+COL.START%

'LPRINT DISPL.NAME\$(GLOOP%);" SCALE! = ";SCALE!

IF SCALE! >= COL.START% AND SCALE! <= COL.END% THEN SCALE% =
CINT(SCALE!)

IF SCALE! < COL.START% THEN SCALE% = COL.START%

IF SCALE! > COL.END% THEN SCALE% = COL.END%

IF SCALE.OLD%(GLOOP%) = SCALE% THEN GOTO NXTGLP

ROW.START% = ROW.OFFSET% + (PNT.BAR.SPAN% * (GLOOP% - 1))

'LPRINT "SCALE%=" ;SCALE%

8032 LINE (COL.START%,ROW.START%)-(COL.END%,ROW.START%+BAR.WIDTH%),7,BF

8033 LINE (COL.START%,ROW.START%)-(SCALE%,ROW.START%+BAR.WIDTH%),9,BF
SCALE.OLD%(GLOOP%) = SCALE%

NXTGLP:

NEXT GLOOP%

SCR13INP1:

IF INPUT.STRING\$ = NULL\$ GOTO SCR13RTN

IF (MAX.WAIT% OR MIN.WAIT% OR AWAIT.SCL% OR AWAIT.GRP%)
AND (CARRIAGE.RETURN% = ASC(INPUT.STRING\$)) THEN

MAX.WAIT% = FALSE%

MIN.WAIT% = FALSE%

AWAIT.SCL% = FALSE%

AWAIT.GRP% = FALSE%

SCREEN.STATIC% = FALSE%

GOTO SCR13RTN

END IF

IF AWAIT.GRP% THEN GOTO GETGRP

IF INPUT.STRING\$ <> "GROUP" GOTO GRPSPAN

QUERY\$ = "ENTER GROUP FILE# EXTENSION (EX. 001 FOR GROUP 1) OR <CR>"

GOSUB QUERYLN

AWAIT.GRP% = TRUE%

GOTO SCR13RTN

GETGRP:

GROUP.SAV\$ = GROUP\$

GROUP\$ = "GROUP." + INPUT.STRING\$

ON ERROR GOTO GETGRP.ERR

OPEN GROUP\$ FOR INPUT AS #13

CLOSE #13

AWAIT.GRP% = FALSE%

AUTO.BSCALE% = TRUE%

SCREEN.STATIC% = FALSE%

ON ERROR GOTO ERRHANDLER
GOTO SCRNI3RTN

GETGRP.ERR:

CLOSE #13
GROUP\$ = GROUP.SAV\$
RESUME GETGRP.ERR1

GETGRP.ERR1:

ON ERROR GOTO ERRHANDLER
GOTO SCRNI3INP2

GRPSPAN:

IF MAX.WAIT% THEN GOTO GAXIS.MAX
IF MIN.WAIT% THEN GOTO GAXIS.MIN
IF AWAIT.SCL% THEN GOTO GSCALE.TYPE
IF INPUT.STRING\$ <> "SCALE" THEN GOTO SCRNI3INP2

QUERY\$ = "ENTER AUTO OR MAN FOR BAR GRAPH SCALING OR <CR> FOR NO CHANGE"
GOSUB QUERYLN
AWAIT.SCL% = TRUE%
GOTO SCRNI3RTN

GSCALE.TYPE:

IF INPUT.STRING\$ = "AUTO" THEN
AUTO.BSCALE% = TRUE%
AWAIT.SCL% = FALSE%
SCREEN.STATIC% = FALSE%
GOTO SCRNI3RTN
END IF

IF INPUT.STRING\$ = "MAN" THEN
AUTO.BSCALE% = FALSE%
AWAIT.SCL% = FALSE%
MIN.WAIT% = TRUE%
QUERY\$ = "ENTER BAR GRAPH AXIS MIN. VALUE OR <CR> FOR NO CHANGE"
GOSUB QUERYLN
GOTO SCRNI3RTN
END IF

GOTO SCRNI3INP2

GAXIS.MIN:

BARSC1.MIN! = VAL (INPUT.STRING\$)
MIN.WAIT% = FALSE%
MAX.WAIT% = TRUE%
QUERY\$ = "ENTER BAR GRAPH AXIS MAX. VALUE OR <CR> FOR NO CHANGE"
GOSUB QUERYLN
GOTO SCRNI3RTN

GAXIS.MAX:

BARSC1.MAX! = VAL (INPUT.STRING\$)
GSPAN! = BARSC1.MAX! - BARSC1.MIN!

```

IF GSPAN! <= 0 THEN AUTO.BSCALE% = TRUE%
MAX.WAIT% = FALSE%
SCREEN.STATIC% = FALSE%
GOTO SCRN13RTN

```

SCRN13INP2:

```

GOSUB INSTRINTP 'Input String Interpreter
IF STRING.INTERP% THEN GOSUB 35170 'Input Error Handler

```

SCRN13RTN:

```

RETURN
***** END OF SCREEN 13 *****

```

```

***** SCREEN 15 - MICRISTARR PROFILE DISPLAY *****

```

SCRN15:

```

IF SCREEN.STATIC% GOTO SCRN15DYN
,
FILE.NAMES$="SCRN015.SCN"
SCREEN 0
COLOR 15,1
SCRN.COLOR% = 15
CLS
INPUT.PROMPT% = FALSE%
GOSUB SHWSTATIC
QUERY$ = ENTOPT$
GOSUB QUERYLN 'Query Line Generator
,

```

```

SCREEN.STATIC%=TRUE%

```

```

'SCREEN DYNAMIC OUTPUT
,

```

SCRN15DYN:

```

IF INPUT.STRING$ = NULL$ GOTO SCRN15RTN

```

SCRN13INP:

```

GOSUB INSTRINTP 'Input String Interpreter
IF STRING.INTERP% THEN GOSUB 35170 'Input Error Handler

```

SCRN15RTN:

```

RETURN

***** END SCREEN OF SCREEN 15 *****
,
***** SCREEN 16 - CONTROL PARAM DISPLAY *****

```

SCRN16:

```

IF SCREEN.STATIC% GOTO SCRN16DYN
,
FILE.NAME$="SCRN016.SCN"
SCREEN 0
COLOR 15,1
SCRN.COLOR% = 15
CLS
INPUT.PROMPT% = FALSE%
GOSUB SHWSTATIC

FOR ANO.PTR% = 1 TO NUM.ANO%
LOCATE 17+ANO.PTR%,63
IF ANO.MAN.OUTPUT%(ANO.PTR%) = TRUE% THEN
    COLOR 12
    PRINT "MANUAL  ";
ELSE
    COLOR 10
    PRINT "AUTOMATIC";
END IF
NEXT ANO.PTR%

COLOR SCRN.COLOR%

QUERY$ = ENTOPT$
GOSUB QUERYLN      'Query Line Generator
,
'OPEN POINT NAMES FILE FOR SCREEN AND READ
,
FILE.NAME$ = "SCRN016.PNT"
GOSUB LDDYNPNT      'LOAD THE DYNAMIC POINTS FOR THIS SCREEN.
,
SCREEN.STATIC%=TRUE%
,
'SCREEN DYNAMIC OUTPUT
,
SCRN15DYN:

GOSUB DSPDYNPNT      'PUT THE VALUES ON THE SCREEN.
,
LOCATE 05,40:PRINT USING F7P2$;FLUX.PCT.ABOVE.AMB; 'DISP.LAMPFLUX S.P.
LOCATE 06,30:PRINT USING F7P2$;FLUX.AVG; 'DISP. AVG. UV-B LAMP FLUX
LOCATE 09,28:PRINT USING F7P2$;AMBFLUX.AVG; 'DISP. AMB. SOLAR UV-B FLUX
LOCATE 11,40:PRINT USING F7P2$;ANO.SP(1); 'DISPLAY LAMPFIL SETPOINT

'INTERPRET INPUT STRING
IF INPUT.STRING$ <> "SETPOINT" GOTO SCRN16INP1
NEW.SCREEN% = 17
GOSUB NEWSN
GOTO SCRN16RTN
,
SCRN16INP1:
IF INPUT.STRING$ <> "GAIN" GOTO SCRN16INP2

```

```

NEW.SCREEN% = 18
GOSUB NEWSN
GOTO SCRN16RTN
,

SCRN16INP2:
IF INPUT.STRING$ <> "RESET" GOTO SCRN16INP3
NEW.SCREEN% = 19
GOSUB NEWSN
GOTO SCRN16RTN
,

SCRN16INP3:
IF INPUT.STRING$ <> "MANAUTO" GOTO SCRN16INP4
NEW.SCREEN% = 20
GOSUB NEWSN
GOTO SCRN16RTN
,

SCRN16INP4:
IF INPUT.STRING$ <> "OUTPUT" GOTO SCRN16INP
NEW.SCREEN% = 21
GOSUB NEWSN
GOTO SCRN16RTN
,

SCRN16INP:
GOSUB INSTRINTP 'Input String Interpreter
IF STRING.INTERP% THEN GOSUB 35170 'Input Error Handler
SCRN16RTN:
RETURN

***** END OF SCREEN 16 *****
,
***** SCREEN 17 - CHANGE CONTROL SP DISPLAY *****

SCRN17:
IF SCREEN.STATIC% GOTO SCRN17DYN
,
FILE.NAME$="SCRN017.SCN"
SCREEN 0
COLOR 15,1
SCRN.COLOR% = 15
CLS
INPUT.PROMPT% = FALSE%
GOSUB SHWSTATIC

IF AWAIT.CTL% THEN
QUERY$ = "ENTER SET POINT "+CHR$(34)+"VALUE"+CHR$(34)+" FOR " _
        +PNT.NAME$(CTL.PTR2%)+ " OR <CR> FOR NO CHANGE."
ELSE
LOCATE 22,01
PRINT "OPTIONS: ENTER CONTROLLER NAME TO CHANGE SETPOINT OR EXIT OR <CR>";
QUERY$ = ENTPT$
END IF
GOSUB QUERYLN 'Query Line Generator
,

```

```
'OPEN POINT NAMES FILE FOR SCREEN AND READ
'
'FILE.NAME$ = "SCRN017.PNT"
'GOSUB LDDYNPNT      'LOAD THE DYNAMIC POINTS FOR THIS SCREEN.
'
SCREEN.STATIC%=TRUE%
'
'SCREEN DYNAMIC OUTPUT
'
```

SCRN17DYN:

```
FOR CTL.PTR% = 1 TO NUM.ANO%
  LOCATE 8+2*(CTL.PTR% - 1),37:PRINT ANO.SP(CTL.PTR%);
NEXT CTL.PTR%

'GOSUB DSPDYNPNT      'PUT THE VALUES ON THE SCREEN.
'
'INTERPRET INPUT STRING

IF INPUT.STRING$ = NULL$ THEN GOTO SCR17RTN

IF((CARRIAGE.RETURN%=ASC(INPUT.STRING$))AND(AWAIT.CTL%=TRUE%)) THEN
  AWAIT.CTL% = FALSE%
  SCREEN.STATIC% = FALSE%
  GOTO SCR17RTN
END IF

IF ((INPUT.STRING$=PREV$) OR (CARRIAGE.RETURN=ASC(INPUT.STRING$))) THEN
  AWAIT.CTL% = FALSE%
  GOTO SCR17INP
END IF
IF AWAIT.CTL% THEN
  ANO.SP(CTL.CNT%) = VAL(INPUT.STRING$)
  IF PNT.NAMES$(ANO.TBL%(CTL.CNT%)) = "LAMPFLUX" THEN
    FLUX.PCT.ABOVE.AMB = ANO.SP(CTL.CNT%)
  END IF
  AWAIT.CTL% = FALSE%
  CTL.CHANGE% = TRUE%
  SCREEN.STATIC% = FALSE%
  GOTO SCR17RTN
END IF

FOR CTL.CNT% = 1 TO NUM.ANO%
  CTL.PTR1% = ANO.TBL%(CTL.CNT%)
  IF PNT.NAMES$(CTL.PTR1%) <> INPUT.STRING$ THEN GOTO NXT.CTL.SP
  CTL.PTR2% = CTL.PTR1%
  AWAIT.CTL% = TRUE%
  SCREEN.STATIC% = FALSE%
  GOTO SCR17RTN
```

```
NXT.CTL.SP:
  NEXT CTL.CNT%
```

```

SCRN17INP:
    GOSUB INSTRINTP      'Input String Interpreter
    IF STRING.INTERP% THEN GOSUB 35170      'Input Error Handler
SCRN17RTN:
    RETURN

'***** END OF SCREEN 17 *****
'***** SCREEN 18 - CHANGE CONTROL GAIN DISPLAY *****

SCRN18:
    IF SCREEN.STATIC% GOTO SCRN18DYN
    FILE.NAME$="SCRN018.SCN"
    SCREEN 0
    COLOR 15,1
    SCRN.COLOR% = 15
    CLS
    INPUT.PROMPT% = FALSE%
    GOSUB SHWSTATIC

    IF AWAIT.CTL% THEN
        QUERY$ = "ENTER NEW GAIN "+CHR$(34)+"VALUE"+CHR$(34)+" FOR " _
            +PNT.NAMES$(CTL.PTR2%)+ " OR <CR> FOR NO CHANGE."
    ELSE
        LOCATE 22,01
        PRINT "OPTIONS: ENTER CONTROLLER NAME TO CHANGE GAIN OR EXIT OR <CR>";

        QUERY$ = ENTOPT$
    END IF
    GOSUB QUERYLN      'Query Line Generator

    'OPEN POINT NAMES FILE FOR SCREEN AND READ
    FILE.NAME$ = "SCRN018.PNT"
    'GOSUB LDDYNPNT      'LOAD THE DYNAMIC POINTS FOR THIS SCREEN.

    SCREEN.STATIC%=TRUE%
    'SCREEN DYNAMIC OUTPUT

SCRN18DYN:

    FOR CTL.PTR% = 1 TO NUM.ANO%
        LOCATE 8+2*(CTL.PTR% - 1),43:PRINT ANO.KP(CTL.PTR%);
    NEXT CTL.PTR%

    'GOSUB DSPDYNPNT      'PUT THE VALUES ON THE SCREEN.

    'INTERPRET INPUT STRING

    IF INPUT.STRING$ = NULL$ THEN GOTO SCRN18RTN

```



```

IF((CARRIAGE.RETURN%=ASC(INPUT.STRING$))AND(AWAIT.CTL%=TRUE%)) THEN
  AWAIT.CTL% = FALSE%
  SCREEN.STATIC% = FALSE%
  GOTO SCRNI8RTN
END IF

IF ((INPUT.STRING$=PREV$) OR (CARRIAGE.RETURN=ASC(INPUT.STRING$))) THEN
  AWAIT.CTL% = FALSE%
  GOTO SCRNI8INP
END IF
IF AWAIT.CTL% THEN
  ANO.KP(CTL.CNT%) = VAL(INPUT.STRING$)
  AWAIT.CTL% = FALSE%
  CTL.CHANGE% = TRUE%
  SCREEN.STATIC% = FALSE%
  GOTO SCRNI8RTN
END IF

FOR CTL.CNT% = 1 TO NUM.ANO%
  CTL.PTR1% = ANO.TBL%(CTL.CNT%)
  IF PNT.NAMES$(CTL.PTR1%) <> INPUT.STRING$ THEN GOTO NXT.CTL.DG
  CTL.PTR2% = CTL.PTR1%
  AWAIT.CTL% = TRUE%
  SCREEN.STATIC% = FALSE%
  GOTO SCRNI8RTN

NXT.CTL.DG:
  NEXT CTL.CNT%

,

SCRNI8INP:
  GOSUB INSTRINTP      'Input String Interpreter
  IF STRING.INTERP% THEN GOSUB 35170      'Input Error Handler
SCRNI8RTN:
  RETURN

***** END OF SCREEN 18 *****
,
***** SCREEN 19 - CHANGE CONTROL RESET DISPLAY *****

SCRNI9:
  IF SCREEN.STATIC% GOTO SCRNI9DYN
  ,
  FILE.NAME$="SCRNI9.SCN"
  SCREEN 0
  COLOR 15,1
  SCRNI9.COLOR% = 15
  CLS
  INPUT.PROMPT% = FALSE%
  GOSUB SHWSTATIC

```

```

IF AWAIT.CTL% THEN
  QUERY$ = "ENTER NEW RESET "+CHR$(34)+"VALUE"+CHR$(34)+" FOR " _
    +PNT.NAME$(CTL.PTR2%)+ " OR <CR> FOR NO CHANGE."
ELSE
  LOCATE 22,01
  PRINT "OPTIONS: ENTER CONTROLLER NAME TO CHANGE RESET OR EXIT OR <CR>";

  QUERY$ = ENTOPT$
END IF
GOSUB QUERYLN      'Query Line Generator
,
'OPEN POINT NAMES FILE FOR SCREEN AND READ
,
'FILE.NAME$ = "SCRN019.PNT"
'GOSUB LDDYNPNT      'LOAD THE DYNAMIC POINTS FOR THIS SCREEN.
,
SCREEN.STATIC%=TRUE%
,
'SCREEN DYNAMIC OUTPUT
,
SCRN19DYN:

FOR CTL.PTR% = 1 TO NUM.ANO%
  LOCATE 8+2*(CTL.PTR% - 1),43:PRINT ANO.KI(CTL.PTR%);
NEXT CTL.PTR%

'GOSUB DSPDYNPNT      'PUT THE VALUES ON THE SCREEN.
,
'INTERPRET INPUT STRING

IF INPUT.STRING$ = NULL$ THEN GOTO SCR19RTN

IF((CARRIAGE.RETURN%=ASC(INPUT.STRING$))AND(AWAIT.CTL%=TRUE%)) THEN
  AWAIT.CTL% = FALSE%
  SCREEN.STATIC% = FALSE%
  GOTO SCR19RTN
END IF

IF ((INPUT.STRING$=PREV$) OR (CARRIAGE.RETURN=ASC(INPUT.STRING$))) THEN
  AWAIT.CTL% = FALSE%
  GOTO SCR19INP
END IF
IF AWAIT.CTL% THEN
  ANO.KI(CTL.CNT%) = VAL(INPUT.STRING$)
  AWAIT.CTL% = FALSE%
  CTL.CHANGE% = TRUE%
  SCREEN.STATIC% = FALSE%
  GOTO SCR19RTN
END IF

FOR CTL.CNT% = 1 TO NUM.ANO%
  CTL.PTR1% = ANO.TBL%(CTL.CNT%)
  IF PNT.NAME$(CTL.PTR1%) <> INPUT.STRING$ THEN GOTO NXT.CTL.DR

```

```

CTL.PTR2% = CTL.PTR1%
AWAIT.CTL% = TRUE%
SCREEN.STATIC% = FALSE%
GOTO SCRNI9RTN

```

```

NXT.CTL.DR:
    NEXT CTL.CNT%

```

```

SCRNI9INP:
    GOSUB INSTRINTP    'Input String Interpreter
    IF STRING.INTERP% THEN GOSUB 35170    'Input Error Handler
SCRNI9RTN:
    RETURN

```

```

***** END OF SCREEN 19 *****

```

```

***** SCREEN 20 - CHANGE CONTROLLER OUTPUT MODE *****

```

```

SCRN20:
    IF SCREEN.STATIC% GOTO SCRNI20DYN
    FILE.NAME$="SCRNI20.SCN"
    SCREEN 0
    COLOR 15,1
    SCRNI.COLOR% = 15
    CLS
    INPUT.PROMPT% = FALSE%
    GOSUB SHWSTATIC

    IF AWAIT.CTL% THEN
        QUERY$ = "ENTER "+CHR$(34)+"AUTO"+CHR$(34)+" OR "+CHR$(34)+"_
                  "MAN"+CHR$(34)+" TO CHANGE MODE FOR "
                  +PNT.NAME$(CTL.PTR2%)+ " OR <CR> FOR NO CHANGE."
    ELSE
        LOCATE 22,01
        PRINT "OPTIONS: ENTER CONTROLLER NAME TO CHANGE OUTPUT MODE OR"+_
              " EXIT OR <CR>";

        QUERY$ = ENTOPT$
    END IF
    GOSUB QUERYLN    'Query Line Generator

    FOR ANO.PTR% = 1 TO NUM.ANO%
        LOCATE 8+2*(ANO.PTR% - 1),43

        IF ANO.MAN.OUTPUT%(ANO.PTR%) = TRUE% THEN
            COLOR 12
            PRINT "MANUAL ";
        ELSE
            COLOR 10
            PRINT "AUTOMATIC";
        END IF
    
```

NEXT ANO.PTR%

COLOR SCR.N.COLOR%

'OPEN POINT NAMES FILE FOR SCREEN AND READ

'FILE.NAMES\$ = "SCRN020.PNT"

'GOSUB LDDYNPNT 'LOAD THE DYNAMIC POINTS FOR THIS SCREEN.

SCREEN.STATIC%=TRUE%

'SCREEN DYNAMIC OUTPUT

SCRN20DYN:

'GOSUB DSPDYNPNT 'PUT THE VALUES ON THE SCREEN.

'INTERPRET INPUT STRING

IF INPUT.STRING\$ = NULL\$ THEN GOTO SCR.N20RTN

IF ((CARRIAGE.RETURN%=ASC(INPUT.STRING\$)) AND (AWAIT.CTL%=TRUE%)) THEN

AWAIT.CTL% = FALSE%

SCREEN.STATIC% = FALSE%

GOTO SCR.N20RTN

END IF

IF ((INPUT.STRING\$=PREV\$) OR (CARRIAGE.RETURN=ASC(INPUT.STRING\$))) THEN

AWAIT.CTL% = FALSE%

GOTO SCR.N20INP

END IF

IF AWAIT.CTL% THEN

IF (INPUT.STRING\$) = "MAN" THEN

ANO.MAN.OUTPUT%(CTL.CNT%) = TRUE%

IF (INPUT.STRING\$) = "AUTO" THEN

ANO.MAN.OUTPUT%(CTL.CNT%) = FALSE%

IF ((INPUT.STRING\$ <> "MAN") AND

(INPUT.STRING\$ <> "AUTO")) THEN GOTO SCR.N20ST

AWAIT.CTL% = FALSE%

CTL.CHANGE% = TRUE%

SCR.N20ST: SCREEN.STATIC% = FALSE%

GOTO SCR.N20RTN

END IF

FOR CTL.CNT% = 1 TO NUM.ANO%

CTL.PTR1% = ANO.TBL%(CTL.CNT%)

IF PNT.NAMES\$(CTL.PTR1%) <> INPUT.STRING\$ THEN GOTO NXT.CTL

CTL.PTR2% = CTL.PTR1%

AWAIT.CTL% = TRUE%

SCREEN.STATIC% = FALSE%

```

GOTO SCRN20RTN

NXT.CTL:
NEXT CTL.CNT%

SCRN20INP:
GOSUB INSTRINTP      'Input String Interpreter
IF STRING.INTERP% THEN GOSUB 35170      'Input Error Handler
SCRN20RTN:
RETURN

***** END OF SCREEN 20 *****

***** SCREEN 21 - CHANGE CONTROLLER OUTPUT PERCENTAGE *****

SCRN21:
IF SCREEN.STATIC% GOTO SCRN21DYN
FILE.NAME$="SCRN021.SCN"
SCREEN 0
COLOR 15,1
SCRN.COLOR% = 15
CLS
INPUT.PROMPT% = FALSE%
GOSUB SHWSTATIC

IF AWAIT.CTL% THEN
QUERY$ = "ENTER NEW OUTPUT "+CHR$(34)+"VALUE"+CHR$(34)+" FOR "_
        +PNT.NAME$(CTL.PTR2%)+ " OR <CR> FOR NO CHANGE."
ELSE
LOCATE 22,01
PRINT "OPTIONS: ENTER CONTROLLER NAME TO CHANGE OUTPUT MODE OR"+_
        " EXIT OR <CR>";

QUERY$ = ENTOPT$
END IF
GOSUB QUERYLN      'Query Line Generator

FOR ANO.PTR% = 1 TO NUM.ANO%
LOCATE 8+2*(ANO.PTR%-1),64
IF ANO.MAN.OUTPUT%(ANO.PTR%) = TRUE% THEN
COLOR 12
PRINT "MANUAL ";
ELSE
COLOR 26
PRINT "AUTOMATIC";
END IF
NEXT ANO.PTR%

COLOR SCRN.COLOR%

```

```

'
'OPEN POINT NAMES FILE FOR SCREEN AND READ
'
FILE.NAME$ = "SCRN021.PNT"
GOSUB LDDYNPNT      'LOAD THE DYNAMIC POINTS FOR THIS SCREEN.
'
SCREEN.STATIC%=TRUE%
'
'SCREEN DYNAMIC OUTPUT
'
SCRN21DYN:

GOSUB DSPDYNPNT      'PUT THE VALUES ON THE SCREEN.
'
'INTERPRET INPUT STRING

IF INPUT.STRING$ = NULL$ THEN GOTO SCR21RTN

IF((CARRIAGE.RETURN%=ASC(INPUT.STRING$))AND(AWAIT.CTL%=TRUE%)) THEN
  AWAIT.CTL% = FALSE%
  SCREEN.STATIC% = FALSE%
  GOTO SCR21RTN
END IF

IF ((INPUT.STRING$=PREV$) OR (CARRIAGE.RETURN=ASC(INPUT.STRING$))) THEN
  AWAIT.CTL% = FALSE%
  GOTO SCR21INP
END IF
IF AWAIT.CTL% THEN
  CNTLOUT(CTL.CNT%) = VAL(INPUT.STRING$)
  IF CNTLOUT(CTL.CNT%) > 100.0 THEN CNTLOUT(CTL.CNT%) = 100.0
  IF CNTLOUT(CTL.CNT%) < 0.0 THEN CNTLOUT(CTL.CNT%) = 0.0
  AWAIT.CTL% = FALSE%
  CTL.CHANGE% = TRUE%
SCRN21ST:  SCREEN.STATIC% = FALSE%
          GOTO SCR21RTN
END IF

FOR CTL.CNT% = 1 TO NUM.ANO%
  CTL.PTR1% = ANO.TBL%(CTL.CNT%)
  IF PNT.NAMES$(CTL.PTR1%) <> INPUT.STRING$ THEN GOTO NXT.CTL.OUT
  CTL.PTR2% = CTL.PTR1%
  AWAIT.CTL% = TRUE%
  SCREEN.STATIC% = FALSE%
  GOTO SCR21RTN

NXT.CTL.OUT:
  NEXT CTL.CNT%

'
SCRN21INP:
GOSUB INSTRINTP      'Input String Interpreter

```

```

      IF STRING.INTERP% THEN GOSUB 35170      'Input Error Handler
SCRN21RTN:
      RETURN

      '***** END OF SCREEN 21 *****
      '***** SCREEN #25 - ALL POINTS DISPLAY *****

8050  'DEBUG #
SCRN25:
      IF SCREEN.STATIC% GOTO SCRN25DYN
      FILE.NAME$="SCRN025.SCN"
      SCREEN 0
      COLOR 15,1
      SCRN.COLOR% = 15
      CLS
      INPUT.PROMPT% = FALSE%
      GOSUB SHWSTATIC      'READ STATIC INFO & DISPLAY
      QUERY$ = ENTOPT$
      GOSUB QUERYLN      'Query Line Generator
      SCREEN.STATIC% = TRUE% 'STATIC INFO IS ON SCREEN.
      UPDATE% = FALSE%

      'SET POINTERS INTO DATABASE FOR INITIAL DISPLAY.
      DSTART% = 1
      DSTOP% = DSTART% + 12
      IF DSTOP% > NUMPNT% THEN DSTOP% = NUMPNT%
      FIRST.LINE% = 6

SCRN25DYN:  'DYNAMIC DATA DISPLAY

      IF UPDATE% GOTO NOUPD 'DO NOT MOVE FORWARD OR BACKWARD IN DB
      FOR DLOOP% = DSTART% TO DSTOP%
      LINE.NO% = FIRST.LINE% + DLOOP% - DSTART%
      LOCATE LINE.NO%,14
      PRINT USING "\ \";PNT.NAMES$(DLOOP%)
      NEXT DLOOP%
      UPDATE% = TRUE%      'INDICATE THAT NEW POINTS ARE ON SCREEN.

NOUPD:
      FOR DLOOP% = DSTART% TO DSTOP%
      LINE.NO% = FIRST.LINE% + DLOOP% - DSTART%
      IF PNT.TYPE$(DLOOP%) <> TXT$ GOTO CHKANIUPD
      LOCATE LINE.NO%,25
      PRINT USING "\ \";PNT.TYPE$(DLOOP%)
      LOCATE LINE.NO%,30
      PRINT USING "\ \";
      TXT.TEXT$(TYPE.PNTR%(DLOOP%))

CHKANIUPD:
      IF PNT.TYPE$(DLOOP%) <> ANIS$ GOTO CHKDGIUPD
      LOCATE LINE.NO%,25
      PRINT USING "\ \";PNT.TYPE$(DLOOP%)

```

```

LOCATE LINE.NO%,31
PRINT USING "####.##",ANI.VALUE!(TYPE.PNTR%(DLOOP%))
LOCATE LINE.NO%,42
PRINT USING "\    \",ANI.EUNIT$(TYPE.PNTR%(DLOOP%))
LOCATE LINE.NO%,53
PRINT USING "####.##",ANI.MINVAL!(TYPE.PNTR%(DLOOP%))
LOCATE LINE.NO%,64
PRINT USING "####.##",ANI.MAXVAL!(TYPE.PNTR%(DLOOP%))
GOTO NXTUPNT
'CHECK FOR DIGITAL INPUT POINT.
CHKDGIUPD:
IF PNT.TYPE$(DLOOP%) <> DGI$ GOTO CHKDGOUPD
LOCATE LINE.NO%,25
PRINT PNT.TYPE$(DLOOP%);
LOCATE LINE.NO%,30
IF (DGI.VALUE%(TYPE.PNTR%(DLOOP%))) THEN PRINT ON$; ELSE PRINT OFF$;
GOTO NXTUPNT
'CHECK FOR DIGITAL OUTPUT.
CHKDGOUPD:
IF PNT.TYPE$(DLOOP%) <> DGO$ GOTO CHKANOUPTD
LOCATE LINE.NO%,25
PRINT PNT.TYPE$(DLOOP%);
LOCATE LINE.NO%,30
IF (DGO.VALUE%(TYPE.PNTR%(DLOOP%))) THEN PRINT ON$;ELSE PRINT OFF$;
GOTO NXTUPNT
'CHECK FOR ANALOG OUTPUT.
CHKANOUPTD:
IF PNT.TYPE$(DLOOP%) <> ANO$ GOTO NXTUPNT
LOCATE LINE.NO%,25
PRINT PNT.TYPE$(DLOOP%);
LOCATE LINE.NO%,31
PRINT USING F7P2$;ANO.VALUE(TYPE.PNTR%(DLOOP%));
LOCATE LINE.NO%,42
PRINT USING "\    \", "PERCENT"
NXTUPNT:
NEXT DLOOP%
'
'INTERPRET INPUT
'
SCRN25INP:
IF INPUT.STRING$ = NULL$ THEN GOTO SCR25RTN

IF LRESET% AND CARRIAGE.RETURN% = ASC(INPUT.STRING$) THEN
  LRESET% = FALSE%
  SCREEN.STATIC% = FALSE%
  GOTO SCR25RTN
END IF
IF INPUT.STRING$ <> DOWN$ GOTO GO.UP
'
'MOVE DOWN INTO DATA BASE -
'
'FIRST ADJUST POINTERS
'
```



```

IF DSTOP% = NUMPNT% GOTO SCRN25RTN
DSTART% = DSTOP% + 1
DSTOP% = DSTART% + 12
IF DSTOP% > NUMPNT% THEN DSTOP% = NUMPNT%
UPDATE% = FALSE%
GOTO ERASELN

GO.UP:
IF INPUT.STRING$ <> UP$ GOTO LO.HI.RESET
'
'MOVE UPWARD IN DATA BASE -
'
'FIRST ADJUST POINTERS
'
DSTART% = DSTART% - 13
IF DSTART% < 1 THEN DSTART% = 1
DSTOP% = DSTART% + 12
IF DSTOP% > NUMPNT% THEN DSTOP% = NUMPNT%
UPDATE% = FALSE%
GOTO ERASELN

LO.HI.RESET:
IF LRESET% THEN GOTO CHKHILO
IF INPUT.STRING$ <> "RESET" GOTO SCRN25INP1
QUERY$ = "ENTER MIN OR MAX TO RESET MIN OR MAX ANI VALUES OR <CR> FOR NO
CHANGE"
GOSUB QUERYLN
LRESET% = TRUE%
GOTO SCRN25RTN

CHKHILO:
IF INPUT.STRING$ = "MAX" THEN
  FOR MLOOP%=0 TO MAXANI%
    ANI.MAXVAL!(MLOOP%) = -9.E+9
  NEXT MLOOP%
  LRESET% = FALSE%
  SCREEN.STATIC% = FALSE%
  GOTO SCRN25RTN
END IF

IF INPUT.STRING$ = "MIN" THEN
  FOR MLOOP%=0 TO MAXANI%
    ANI.MINVAL!(MLOOP%) = +9.E+9
  NEXT MLOOP%
  LRESET% = FALSE%
  SCREEN.STATIC% = FALSE%
  GOTO SCRN25RTN
END IF

GOTO SCRN25INP3

' Check to see if input string is a point ID

```

SCRN25INP1:

```

FOR DLOOP% = 1 TO NUMPNT%
IF INPUT.STRING$ <> PNT.NAMES$(DLOOP%) THEN GOTO SCRN25INP2
DSTART% = DLOOP%
DSTOP% = DSTART% + 12
IF DSTOP% > NUMPNT% THEN DSTOP% = NUMPNT%
UPDATE% = FALSE%
GOTO ERASELN

```

SCRN25INP2:

```

NEXT DLOOP%
' Input string did not contain a valid point ID
GOTO SCRN25INP3

```

```

'ERASE ONLY THE POINT INFO, LEAVE STATIC INFO ALONE.

```

ERASELN:

```

FOR DLOOP% = FIRST.LINE% TO SCR.LAST.LINE%-7
LOCATE DLOOP%,1
PRINT SPACES$(80);
NEXT DLOOP%
GOTO SCRN25RTN

```

SCRN25INP3:

```

GOSUB INSTRINTP 'Input String Interpreter
IF STRING.INTERP% THEN GOSUB 35170 'Input Error Handler

```

SCRN25RTN:

```

RETURN

```

```

***** END OF SCREEN # 25 *****

```

```

***** SCREEN 8 - GRAPHICS TREND *****

```

SCRN8:

```

IF SCREEN.STATIC% GOTO SCRN8DYN
SCREEN 0
CLS
SCREEN 9,1,0,0 'SET FOR HIGH RESOLUTION MODE.

```

```

HI.RES%=TRUE%

```

```

COLOR 1,15

```

```

SCRN.COLOR% = 1

```

```

CLS

```

```

INPUT.PROMPT% = FALSE%

```

```

FILE.NAMES$ = "SCRN008.SCN"

```

```

GOSUB SHWSTATIC

```

```

COLOR 1,15

```

```

REFRESH% = TRUE%

```

```

TREND.GRPH.START% = FALSE%

```

```

QUERY$ = ENTOPT$

```

```

GOSUB QUERYLN

```

```

SCREEN.STATIC%=TRUE%

```

```

IF NO.POINT.DEF% THEN
  TRND.POINT.NAME$ = TREND.NAME$(1)
  TRND.PNT.LOC% = TREND.LOC%(1)
  TREND.BUFF.LOC% = 1
  AUTO.SCALE% = TRUE%
  TREND.GRAPH% = TRUE%
  TREND.RANGE% = 3
  TREND.RANGE$ = "3"
  NO.POINT.DEF% = FALSE%
  REFRESH% = TRUE%
END IF

```

'1st Time thru display the
'1st trend point listed in
'SCRN008.PNT with auto scale
'and with 8 hours for X-axis.

```

LOCATE 21,1
PRINT "POINT ID: ";
COLOR 4
PRINT TRND.POINT.NAME$;
COLOR 1
PRINT "  UNITS: ";
COLOR 4
PRINT ANI.EUNIT$(TYPE.PNTR%(TRND.PNT.LOC%));
COLOR 1

```

SCRN8DYN:

```

IF REFRESH% THEN GOTO RDWGRD
GOSUB PLTPNT

```

SCRN8INP:

```

IF INPUT.STRING$ = NULL$ THEN GOTO SCR8RTN

```

```

IF (MAX.WAIT% OR MIN.WAIT% OR AWAIT.SCL% OR AWAIT.PNT% OR AWAIT.TIM%)_
AND (CARRIAGE.RETURN% = ASC(INPUT.STRING$)) THEN
  MAX.WAIT% = FALSE%
  MIN.WAIT% = FALSE%
  AWAIT.SCL% = FALSE%
  AWAIT.PNT% = FALSE%
  AWAIT.TIM% = FALSE%
  SCREEN.STATIC% = FALSE%
  GOTO SCR8RTN
END IF

```

```

IF AWAIT.PNT% THEN GOTO GETPNT
IF INPUT.STRING$ <> "POINTID" THEN GOTO CHKTIM
QUERY$ = "PLEASE ENTER THE POINT ID "
GOSUB QUERYLN
AWAIT.PNT% = TRUE%
GOTO SCR8RTN

```

'Get the Point name to be trended.

GETPNT:

```

      FOR TLOOP% = 1 TO NTPNTS%
8004  'DEBUG--LINE #
      IF INPUT.STRING$ = TREND.NAMES$(TLOOP%) GOTO SAVLOC
      NEXT TLOOP%

```

GOTO SCR8INP2 'Didn't find the point, notify and try again.

SAVLOC:

```

      TRND.POINT.NAMES$ = NULL$
      TRND.POINT.NAMES$ = TREND.NAMES$(TLOOP%)
      TRND.PNT.LOC% = TREND.LOC%(TLOOP%)
      TREND.BUFF.LOC% = TLOOP%
      AWAIT.PNT% = FALSE%
      TREND.GRAPH% = TRUE%
      AUTO.SCALE% = TRUE%           ' When changing a point goto auto scale
      SCREEN.STATIC% = FALSE%
      GOTO SCR8RTN

```

CHKTIM:

```

      IF AWAIT.TIM% THEN GOTO GETSPN
      IF INPUT.STRING$ <> "SPAN" THEN GOTO GETSCL
      QUERY$ = "ENTER TIME IN HOURS FOR X-AXIS (MAX: 22 MIN: 1) OR <CR> FOR NO

```

CHANGE"

```

      GOSUB QUERYLN
      AWAIT.TIM% = TRUE%
      GOTO SCR8RTN

```

GETSPN:

```

      TREND.RANGE% = VAL(INPUT.STRING$)
      IF TREND.RANGE% < 1 OR TREND.RANGE% > (MAX.TRND.RNG% - 2) GOTO SCR8INP2
      TREND.RANGE$ = INPUT.STRING$
      AWAIT.TIM% = FALSE%
      SCREEN.STATIC% = FALSE%
      GOTO SCR8RTN

```

GETSCL:

```

      IF MAX.WAIT% THEN GOTO YAXIS.MAX
      IF MIN.WAIT% THEN GOTO YAXIS.MIN
      IF AWAIT.SCL% THEN GOTO SCALE.TYPE
      IF INPUT.STRING$ <> "SCALE" THEN GOTO SCR8INP3
      QUERY$ = "ENTER AUTO OR MAN FOR Y-AXIS SCALING OR <CR> FOR NO CHANGE"
      GOSUB QUERYLN
      AWAIT.SCL% = TRUE%
      GOTO SCR8RTN

```

SCALE.TYPE:

```

      IF INPUT.STRING$ = "AUTO" THEN
      AUTO.SCALE% = TRUE%
      AWAIT.SCL% = FALSE%
      SCREEN.STATIC% = FALSE%
      GOTO SCR8RTN

```

END IF

```
IF INPUT.STRING$ = "MAN" THEN
  AUTO.SCALE% = FALSE%
  AWAIT.SCL% = FALSE%
  MIN.WAIT% = TRUE%
  QUERY$ = "ENTER Y-AXIS MIN. VALUE OR <CR> FOR NO CHANGE"
  GOSUB QUERYLN
  GOTO SCR8RTN
END IF
```

GOTO SCR8INP2

YAXIS.MIN:

```
Y.MIN! = VAL (INPUT.STRING$)
MIN.WAIT% = FALSE%
MAX.WAIT% = TRUE%
QUERY$ = "ENTER Y-AXIS MAX. VALUE OR <CR> FOR NO CHANGE"
GOSUB QUERYLN
GOTO SCR8RTN
```

YAXIS.MAX:

```
Y.MAX! = VAL (INPUT.STRING$)
Y.20PC! = Y.MAX! - Y.MIN!
IF Y.20PC! <= 0 THEN AUTO.SCALE% = TRUE%
MAX.WAIT% = FALSE%
SCREEN.STATIC% = FALSE%
GOTO SCR8RTN
```

SCR8INP3:

```
IF INPUT.STRING$ <> "CLEAR" THEN GOTO SCR8INP2
SCREEN.STATIC% = FALSE%
GOTO SCR8RTN
```

SCR8INP2:

```
GOSUB INSTRINTP      'Input String Interpreter
IF STRING.INTERP% THEN GOSUB 35170      'Input Error Handler
```

SCR8RTN:

```
8005  'DEBUG--LINE #
      RETURN
```

20450 'SCREEN #8 DYNAMIC OUTPUT

20530 '

'GET & PLOT DATA

20540 '***** TREND PLOT GENERATOR *****

PLTPNT:
20546 'ENTRY POINT

```
IF (TRND.HR! > HR.MAX! AND TREND.GRAPH% AND TREND.GRPH.START%) THEN
  IF (HR.MIN%+2) >= 24 THEN TRND.LATCH%=FALSE%
  SCREEN.STATIC% = FALSE%
  RETURN
END IF
```

```
IF TREND.GRPH.START% = FALSE% THEN
  TREND.GRPH.START% = TRUE%
  NUM.TIME.CNTS% = TREND.RANGE%*12
  STRT.TIME.CNT% = TIME.COUNT% - NUM.TIME.CNTS%
  HR.MIN!=HR.MIN%
  HR.MAX!=HR.MAX%
  CALL SCREEN.DEFINE
  CALL
```

```
DETERMINE.TRNDBUFF.INDICES(TIME.COUNT%,NUM.TIME.CNTS%,STRT.TIME.CNT%,START.INDE
X%,END.INDEX%,WRAP%,ALPHA%,OMEGA%)
```

```
IF START.INDEX% <= 0 OR END.INDEX% <= 0 THEN 'LINE STATEMENT REQUIRES
  TREND.GRPH.START%=FALSE% 'AT LEAST 2 POINTS TO PLOT SINGLE LINE SEGMENT.
  GOTO TRND.PLOT.RTN 'NOTE: PSET STATEMENT REQUIRES ONLY 1 POINT.
```

```
END IF
```

```
20549 IF WRAP% THEN CALL AM.PM.PLOT (ALPHA%,OMEGA%,4)
```

```
IF WRAP% THEN CALL WRAP.PLOT (4)
```

```
FOR J%=START.INDEX% TO END.INDEX%
```

```
20550 LINE (TRND.TIME!(J%),TREND.BUFFS!(J%,TREND.BUFF.LOC%))-
```

```
(TRND.TIME!(J%+1),TREND.BUFFS!(J%+1,TREND.BUFF.LOC%)),4
```

```
IF ((TRND.TIME!(J%)-TRND.TIME(J%+1))>0 OR (TRND.TIME!(J%+1)-TRND.TIME!(J%))>0.25)
```

```
THEN
```

```
LPRINT TRND.TIME!(J%),TRND.TIME!(J%+1),START.INDEX%,END.INDEX%,J%,"20550"
```

```
END IF
```

```
20551 NEXT J%
```

```
IF TREND.FLAG% THEN
```

```
TREND.FLAG%=FALSE%
```

```
GOTO TRND.PLOT.RTN
```

```
END IF
```

```
IF WRAP% AND STRT.TIME.CNT%<1 THEN
```

```
TREND.FLAG%=TRUE%
```

```
START.INDEX%=1
```

```
END.INDEX%=TIME.COUNT% - 1
```

```
IF END.INDEX% <= 0 THEN GOTO TRND.PLOT.RTN 'SKIP PLOT - NOT ENOUGH POINTS
```

```
GOTO 20549
```

```
END IF
```

```
ELSE
```

```
IF TIME.COUNT% >= 1 THEN
```

```
'CONTINUE TREND GRAPH, ADDING A POINT TO THE PLOT EVERY TREND INTERVAL TIME
```

```
UNIT
```

```

TC%=TIME.COUNT%
IF TREND.CON% GOTO 20552
TREND.CON%=TRUE%
20552 IF TIME.COUNT%=1 AND WRAP% THEN
    CALL WRAP.PLOT (4)
    GOTO TRND.PLOT.RTN
END IF
IF TC% > 288 GOTO TRND.PLOT.RTN
20553 CALL AM.PM.PLOT (TC%-1,TC%-1,4)
END IF
END IF

TRND.PLOT.RTN:
    RETURN    '???'

'***** END OF TREND PLOT GENERATOR *****'
,
'***** TREND PLOT X-Y AXES GENERATOR *****'
,
,
'***** 'DRAW X-Y AXES & LABEL *****'

RDWGRD:
45250 '
45290 'ENTRY POINT
    ON ERROR GOTO ERRHANDLER
    X.ORIGIN%=70
    Y.ORIGIN%=259
    Y.TOP%=35
    T.WARP#=TIMER*WARP
    IF T.WARP# > 86400# THEN
        T.WARP#=(T.WARP# / 86400# - FIX(T.WARP# / 86400#))*86400#
    END IF
    TRND.DAY# = T.WARP#/86400#
    HR.FRAC! = TRND.DAY#*24
    IF CLEAR.POINT%=FALSE% THEN
        HR.START% = CINT(HR.FRAC! + 0.5)
        HR.MAX% = HR.START% + 2
    END IF
    TREND.RANGE%=VAL(TREND.RANGE$)
    IF TREND.RANGE% > (MAX.TRND.RNG% -2) THEN TREND.RANGE%=MAX.TRND.RNG% - 2
    IF HR.START% - TREND.RANGE% > 0 THEN
        HR.MIN% = HR.START% - TREND.RANGE%
        NUM.HRS% = HR.MAX% - HR.MIN%
        IF NUM.HRS%=0 THEN NUM.HRS%=1
    ELSE
        IF HR.START% - TREND.RANGE% < 0 THEN
            HR.MIN% = 24 + (HR.START% - TREND.RANGE%)
            NUM.HRS% = HR.MAX% + (24 - HR.MIN%)
            IF NUM.HRS%=0 THEN NUM.HRS%=1
        ELSE
            IF HR.START% - TREND.RANGE% = 0 THEN
                HR.MIN% = 0

```

```

        NUM.HRS% = HR.MAX%
        IF NUM.HRS%=0 THEN NUM.HRS%=1
    END IF
END IF
END IF
IF NUM.HRS% > MAX.TRND.RNG% THEN
    NUM.HRS%=MAX.TRND.RNG%
    HR.MIN%=HR.MAX% - NUM.HRS%
    IF HR.MIN% < 0 THEN HR.MIN%=HR.MIN% + 24
END IF
NUM.MINS% = NUM.HRS% * 60
NUM.5MINS% = NUM.MINS%/5
X.SCALE! = (635-X.ORIGIN%)/(NUM.5MINS%)
,
8029
8028  LINE (X.ORIGIN%,Y.ORIGIN%)-(635,20),1,B
,
'GENERATE AXES TIC MARKS & LABEL
45530 '
,      X -AXIS
,
X.TIC!=0.0
X.TIC.TIC!=0.0
TIC.DIF!=(NUM.5MINS%-0)/NUM.HRS%
TIC.TIC.DIF!=(60-0)/6
HRMIN%=HR.MIN%
FOR X.TICLOOP%=1 TO NUM.HRS%+1
PNT.X.OLD%=PNT.X%
PNT.X%=X.ORIGIN%+INT(X.TIC!*X.SCALE! + 0.5)
PNT.X!=CSNG(PNT.X%)
IF X.TICLOOP% < 2 GOTO 45532
X.X.SCALE!=(PNT.X%-PNT.X.OLD%)/(60-0)
FOR X.X.TICLOOP%=1 TO 7
IF (NUM.HRS% > 16) GOTO 45531
PNT.X.X%=PNT.X.OLD%+INT(X.TIC.TIC!*X.X.SCALE!+0.5)
LINE (PNT.X.X%,Y.ORIGIN%)-(PNT.X.X%,Y.ORIGIN%+2),9 '10 MINUTE LIGHT BLUE TIC MARKS
IF (X.X.TICLOOP% < 3 AND X.X.TICLOOP% < 5) THEN GOTO 45531
LINE (PNT.X.X%,Y.ORIGIN%)-(PNT.X.X%,20),9,,&H5555 '20 MINUTE (LIGHT BLUE) SHORT
DASHES
45531 X.TIC.TIC!=X.TIC.TIC!+TIC.TIC.DIF!
NEXT X.X.TICLOOP%
X.TIC.TIC!=0.0
45532 LINE (PNT.X%,Y.ORIGIN%)-(PNT.X%,Y.ORIGIN%+4),1 'HOUR (DARK BLUE) TIC MARKS
LINE (PNT.X%,Y.ORIGIN%)-(PNT.X%,20),1,,&HF0F0 'HOUR (DARK BLUE) LONG DASHES
IF X.TICLOOP%=NUM.HRS%+1 THEN
    LOCATE 20,79
ELSE
    IF X.TICLOOP%=NUM.HRS% THEN
        COL.FACTOR!=(PNT.X!-0)*80
        LOCATE 20,CINT(COL.FACTOR!/640)
    ELSE
        LOCATE 20,CINT((PNT.X!*80)/640)
    END IF

```



```

END IF
IF HRMIN% > 23 THEN HRMIN%=HRMIN%-24
IF (NUM.HRS% > 16 AND HRMIN% MOD 2 = 1) GOTO ADD1HR
PRINT USING "##" ; HRMIN%
  IF HRMIN% >= 23 THEN
    HRMIN% = 0
    GOTO 45533
  END IF
ADD1HR: HRMIN% = HRMIN% + 1
45533 X.TIC! = X.TIC! + TIC.DIF!
  NEXT X.TICLOOP%
  LOCATE 21,44
  PRINT "HOUR"
  ,
  ,          Y-AXIS
  ,
  Y.TIC!=0.00
  IF AUTO.SCALE% THEN
    Y.MIN!=ANI.MINVAL!(TYPE.PNTR%(TRND.PNT.LOC%))
    Y.MAX!=ANI.MAXVAL!(TYPE.PNTR%(TRND.PNT.LOC%))
    Y.20PC!=(Y.MAX!-Y.MIN!)*1.2
  END IF
  IF Y.20PC! < .50 THEN Y.20PC!=1.0
  Y.SCALE!=(Y.ORIGIN%-Y.TOP%)/Y.20PC!
  TREND.YLIMIT!=Y.MAX!+(Y.20PC!-(Y.MAX!-Y.MIN!))/2
  Y.MIN!=Y.MIN!-(Y.20PC!-(Y.MAX!-Y.MIN!))/2
  TIC.DIF!=Y.20PC!/16
  FOR Y.TICLOOP%=1 TO 17
    PNT.Y%=Y.ORIGIN%-INT(Y.TIC!*Y.SCALE!*1.00 + 0.5)
    PNT.Y!=CSNG(PNT.Y%)
    IF Y.TICLOOP% MOD 2 = 1 THEN
      LINE (X.ORIGIN%,PNT.Y!)-(X.ORIGIN%-9,PNT.Y!),1
    ELSE
      LINE (X.ORIGIN%,PNT.Y!)-(X.ORIGIN%-5,PNT.Y!),1
    END IF
    LINE (X.ORIGIN%,PNT.Y!)-(635,PNT.Y!),9,,&HC0C0
    IF Y.TICLOOP% MOD 2 = 1 THEN
      IF Y.TICLOOP%=1 THEN
        LOCATE INT(20-Y.TICLOOP%),INT(((X.ORIGIN%-48)*80)/640)
      ELSE
        LOCATE INT(20-Y.TICLOOP%),INT(((X.ORIGIN%-51)*80)/640)
      END IF
      PRINT USING "####.##";(TIC.DIF!*(Y.TICLOOP%-1)+Y.MIN!)
    END IF
  END IF
45534 Y.TIC!=Y.TIC!+TIC.DIF!
  NEXT Y.TICLOOP%

XYGRDRTN:
  REFRESH% = FALSE%
45560 RETURN ' To line 40130 , End of Trending Static Info
45565 '***** END OF X-Y AXES GENERATOR *****

```

TREND.DATA.CAPTURE: '*****'

```
8505  'DEBUG--LINE #
      OK.TRND2% = TRUE%
      IF OK.TRND1% AND OK.TRND2% THEN          'See if we have read all of
        GOTO TRTIME                          'the Channels before
      ELSE                                      'we store any trend data.
        GOTO TRND.DATA.RTN
      END IF
```

TRTIME:

```
  PREV.TRND.TIME! = TRNDTIME!
  TRNDTIME# = TIMER*WARP
  WARP.DATE = FIX(TRNDTIME#/86400#)
  TIME.CK! = TRNDTIME! * 24
  TRNDTIME! = TRNDTIME#/86400# + JULIAN.DATE%
  IF TRNDTIME! - PREV.TRND.TIME! < 0 THEN TRNDTIME! = TRNDTIME! + 1
```

```
  '
  '
  IF TRNDTIME! < TLTIME! THEN GOTO TRND.DATA.RTN
  LAST.TRND.HR! = TRND.HR!
  TRND.HR! = TRNDTIME!
  IF ((TRND.HR! - LAST.TRND.HR!) > 1) AND (LAST.TRND.HR! > 0) THEN
    TRND.HR! = TRND.HR! - 1
    TRNDTIME! = TRNDTIME! - 1
  END IF
```

```
48375 TIME.COUNT% = TIME.COUNT% + 1
  IF TIME.COUNT% = TREND.5MINS% THEN WRAP% = FALSE%
  IF TIME.COUNT% > TREND.5MINS% THEN ' I.E. 24 HOURS HAVE ELAPSED
    WRAP% = TRUE%
    TIME.COUNT% = 1
  END IF
  '48376 TREND%(TRND.PNT.LOC%) = TRUE%
  '
  'CAPTURE HOUR & MEASURED VALUES IN TREND POINT BUFFERS
  '
  TRND.TIME!(TIME.COUNT%) = TRND.HR!
  IF TREND.GRAPH.START% = FALSE% THEN
    TRND.TIME.STRT! = TRND.TIME!(TIME.COUNT%)
  END IF
  FOR TRLOOP% = 1 TO NTPNTS%
```

```
TREND.BUFFS!(TIME.COUNT%,TRLOOP%) = ANI.VALUE!(TYPE.PNTR%(TREND.LOC%(TRLOOP%)))
  NEXT TRLOOP%
  'LPRINT TREND.BUFFS!(TIME.COUNT%,1)
  PREV.TLTIME! = TLTIME!
  TLTIME! = TRNDTIME! + TRNDINTRVL!/86400#
```

TRND.DATA.RTN:
RETURN

```

***** BEGIN SUBROUTINES FOR TRENDING *****
-----

SUB DETERMINE.TRNDBUFF.INDICES
(TICNT%,NTCNT%,STCNT%,S.INDEX%,E.INDEX%,WRAP%,A.INDEX%,O.INDEX%) STATIC
  SHARED TREND.CON%,TRUE%,FALSE%,TRND.TIME!(),HR.MIN!,TREND.5MINS%
    S.INDEX% = 1
    E.INDEX% = TICNT% - 1
    IF WRAP% = TRUE% THEN
      A.INDEX% = TICNT% + 1
      O.INDEX% = TREND.5MINS% - 1
    END IF
  END SUB
-----

SUB SCREEN.DEFINE STATIC ***** SUBROUTINE *****
  SHARED
    HR.MIN!,HR.MAX!,TREND.YLIMIT!,Y.MIN!,X.ORIGIN%,Y.ORIGIN%,Y.TOP%,TREND.GRPH.START%,T
    RUE%,JULIAN.DATE%,WARP.DATE,TRND.TIME!(),TIME.COUNT%
    HR.MIN!=HR.MIN!/24 + JULIAN.DATE% + WARP.DATE + NEW.MIN.DAY
    NEW.MIN.DAY=0
    HR.MAX!=HR.MAX!/24 + JULIAN.DATE% + WARP.DATE + NEW.MAX.DAY
    NEW.MAX.DAY=0
    IF HR.MIN! > TRND.TIME!(TIME.COUNT%) THEN HR.MIN! = HR.MIN! -1
    IF HR.MAX! < TRND.TIME!(TIME.COUNT%) THEN HR.MAX! = HR.MAX! +1

    WINDOW (HR.MIN!,Y.MIN!)-(HR.MAX!,TREND.YLIMIT!)
    VIEW (X.ORIGIN%,Y.TOP%)-(635,Y.ORIGIN%)
  END SUB
-----

SUB AM.PM.PLOT (INDX1%,INDX2%,COL%) STATIC ***** SUBROUTINE *****
  SHARED TRND.TIME!(),TREND.BUFFS!(),TREND.BUFF.LOC%,HR.MIN!
    FOR J%=INDX1% TO INDX2%
20558   LINE (TRND.TIME!(J%),TREND.BUFFS!(J%,TREND.BUFF.LOC%))-
      (TRND.TIME!(J%+1),TREND.BUFFS!(J%+1,TREND.BUFF.LOC%)),COL%
      IF ((TRND.TIME!(J%)-TRND.TIME(J%+1))>0 OR (TRND.TIME!(J%+1)-TRND.TIME!(J%))>0.25)
THEN
        LPRINT TRND.TIME!(J%),TRND.TIME!(J%+1),INDX1%,INDX2%,J%,COL%,"20558"
      END IF
20559   NEXT J%
  END SUB
-----

SUB WRAP.PLOT (COL%) STATIC ***** SUBROUTINE *****
  SHARED TRND.TIME!(),TREND.BUFFS!(),TREND.BUFF.LOC%,HR.MIN!,TREND.5MINS%
    LINE (TRND.TIME!(TREND.5MINS%),TREND.BUFFS!(TREND.5MINS%,TREND.BUFF.LOC%))-
      (TRND.TIME!(1),TREND.BUFFS!(1,TREND.BUFF.LOC%)),COL%
    IF ((TRND.TIME!(TREND.5MINS%)-TRND.TIME(1))>0 OR (TRND.TIME!(1)-
TRND.TIME!(TREND.5MINS%))>0.25) THEN
      LPRINT TRND.TIME!(TREND.5MINS%),TRND.TIME!(1),COL%,"WRAP"
    END IF

```

END SUB

***** END SUBROUTINES FOR TRENDING *****

***** END OF SCREEN 8 *****

***** SCREEN EXIT HANDLER *****

SCNEXT:

CLS

STACK.POINTER%=1 'SET POINTER BACK TO MASTER SCREEN

SCREEN.STATIC%=FALSE%

CURR.SCREEN%=0

INPUT.PROMPT%=FALSE%

QUERY.PROMPT%=FALSE%

STATUS.PROMPT%=FALSE%

KEY(11) OFF

KFY(14) OFF

RETURN

***** END OF SCREEN EXIT HANDLER *****

***** PREV OR <CR> HANDLER *****

PREVCR:

CLS

SCREEN.STATIC%=FALSE%

CURR.SCREEN%=STACK%(STACK.POINTER%)

IF STACK.POINTER%=0 GOTO NOPOP

STACK.POINTER%=STACK.POINTER%-1 'DECREMENT POINTER (POP THE STACK)

NOPOP:

INPUT.PROMPT%=FALSE%

QUERY.PROMPT%=FALSE%

STATUS.PROMPT%=FALSE%

INPUT.STRING\$=""

KEY(11) OFF

KEY(14) OFF

RETURN

***** END OF PREV OR <CR> HANDLER *****

***** QUERY LINE GENERATOR *****

QUERYLN:

LOCATE QUERY.LINE%,1

PRINT SPACES\$(70);

LOCATE QUERY.LINE%,1

PRINT "QUERY: ";QUERY\$;

RETURN

***** END OF QUERY LINE GENERATOR *****

***** KEYBOARD INPUT ERROR HANDLER *****

```

35170 ENTRY POINT
      INPUT.ERROR%=FALSE%
      INVALID.QUERY%=TRUE%
      'RESPONSE TO INPUT ERR
      '
      LOCATE INPUT.LINE%-1,8
      PRINT SPACE$(70);
      LOCATE INPUT.LINE%-1,8
      PRINT "NOT A VALID SCREEN OPTION";
      'WAIT FEW SECONDS
      FOR K%=1 TO 32000:NEXT K%
      LOCATE INPUT.LINE%-1,8
      PRINT SPACE$(70);
      LOCATE INPUT.LINE%-1,8
      GOSUB QUERYLN
      STRING.INTERP%=FALSE%
      RETURN

      ***** END OF KEYBOARD INPUT ERROR HANDLER *****
      '
      ***** PRINTER LOGGER *****

LOGPRINT:
      IF FACILITY.ON% = FALSE% THEN GOTO PRNLGRTN
      PPLTIME#=TIMER*WARP
      IF PPLTIME# > 86400# THEN
        PPLTIME!=(PPLTIME# / 86400# - FIX(PPLTIME# / 86400#))*86400#
      ELSE PPLTIME!=PPLTIME#
      END IF
      PPLTIME! = PPLTIME! / 86400 + JULIAN.DATE%
      ON ERROR GOTO PRNTERR

      IF PPLTIME! < PLTIME! GOTO PRNLGRTN
NEXT.PRN:
      PRNT.LOOP% = PRNT.LOOP% + 1
      IF PRNT.LOOP% > NUMPNT% THEN
        PRNT.LOOP% = 0
        PRINT.DONE% = TRUE%
        PRINT.COUNT% = 1
        JULOUT% = FALSE%
        PLTIME! = PPLTIME! + PLINTRVL! / 86400
        RETURN
      END IF

      IF PNT.TYPE$(PRNT.LOOP%) <> ANI$ GOTO NEXT.PRN
      IF JULOUT% <> FALSE% GOTO PRN.DATA
      IF LINE.COUNT% >= 44 THEN
        TEST.CNT=LINE.COUNT%
        LINE.COUNT%=0
        LPRINT CHR$(12) 'FORMFEED
      END IF

```

```

PPL.DAY!=PPLTIME!/86400!
JUL.PPL!=JULIAN.DATE%+PPL.DAY!
LPRINT CHR$(10):LPRINT CHR$(10) 'SKIP 2 LINES
LPRINT TAB(11)
LPRINT USING "###.###";JUL.PPL!,
LPRINT SPACES(5);DATE$,TIME$
LPRINT CHR$(10) 'SKIP 1 LINE
JULOUT%=TRUE%

```

```

'OUTPUT COLUMNAR HEADERS

```

```

LPRINT TAB(3);"POINT";TAB(17);"POINT"
LPRINT TAB(4);"NAME";TAB(17);"VALUE"
LPRINT TAB(3);"-----";TAB(17);"-----"
LPRINT CHR$(10) 'SKIP 1 LINE
LINE.COUNT% = LINE.COUNT% + 12

```

```

PRN.DATA:

```

```

LPRINT USING "\_ _ _ _ _###.###";PNT.NAMES(PRNT.LOOP%)_
;ANI.VALUE!(TYPE.PNTR%(PRNT.LOOP%))

```

```

'UPDATE PRINTER LOG TIME FOR POINT

```

```

LINE.COUNT% = LINE.COUNT% + 1
IF LINE.COUNT% >= 53 THEN
  TEST.CNT=LINE.COUNT%
  LINE.COUNT%=0
  LPRINT CHR$(12) 'FORMFEED
  JULOUT% = FALSE%
END IF

```

```

PRINT.COUNT% = PRINT.COUNT% + 2
GOTO PRNLGRTN

```

```

PRNTERR:

```

```

IF (ERR=24) THEN PRINTER.STATUS$ = "PRINTER TIMED-OUT"
IF (ERR=25) THEN PRINTER.STATUS$ = "PRINTER FAULT"
IF (ERR=27) THEN PRINTER.STATUS$ = "PRINTER OUT OF PAPER"
PRNT.LOG% = FALSE%
STATUS.CHANGE% = TRUE%
GOTO IDLE

```

```

PRNLGRTN: RETURN

```

```

***** END OF PRINTER LOGGER *****
,
***** DISK LOGGER *****

```

```

LOGDISK:

```

```

ON ERROR GOTO DISK.ERR.HANDLER

```

```

IF FACILITY.ON% = FALSE% THEN GOTO DSKLGRTN

```

```

IF DISK.HDR% THEN GOTO DISKDATA
' File has not been opened, open file and write header.

SCREEN.STATIC% = FALSE% ' In case we are on Disk Menu, Update the
                          ' Data File Name for the Menu.

GOSUB JULIE
HR$=MID$(TIME$,1,2)
MIN$=MID$(TIME$,4,2)
NOW.DATE$=""
NOW.DATE$=DATE$
NOW.TIME$=""
NOW.TIME$=TIME$
NUMHDR$=STR$(NUMHDR%)
JULIAN.DATE$=STR$(JULIAN.DATE%)
JULEN%=LEN(JULIAN.DATE$)
STST$=RIGHT$(JULIAN.DATE$,JULEN%-1) 'GET RID OF THE LEADING BLANK
IF JULEN%-1=3 THEN STST$=STST$      'NO NEED FOR LEADING ZERO
IF JULEN%-1=2 THEN STST$="0"+STST$
IF JULEN%-1=1 THEN STST$="00"+STST$ 'PUT ON SOME LEADING ZEROES
PART2$=HR$+MIN$
DATEXT$=".DAT"
FIL.NAM$=STST$+PART2$
DATA.FILE$=DATA.DRIVE$+FIL.NAM$+DATEXT$
FIRSTRC$=NOW.TIME$+COMMA$+DATA.TITLE$+COMMA$+NUMHDR$_
+COMMA$+STR$(NUMPNT%)+COMMA$+STR$(NUM.ANI%)
OPEN DATA.FILE$ FOR APPEND AS #3
PRINT #3," "+NOW.DATE$
PRINT #3,FIRSTRC$
'PRINT #3," SHADING FACTOR = "+STR$(SHADE)
FOR NN%=1 TO NUMPNT%
IF PNT.TYPE$(NN%) <> "ANI" THEN GOTO P1
IF TYPE.PNTR%(NN%) <> NUM.ANI% THEN
PRINT #3,PNT.NAMES$(NN%)+COMMA$;
ELSE
PRINT #3,PNT.NAMES$(NN%)
END IF
P1: NEXT NN%

FOR DTYPE% = 1 TO 8
FOR NN% = 1 TO NUMPNT%
IF PNT.TYPE$(NN%) <> "ANI" THEN GOTO P2
NDX%=TYPE.PNTR%(NN%)

IF TYPE.PNTR%(NN%) <> NUM.ANI% THEN
IF DTYPE% = 1 THEN PRINT #3,ANI.EUNIT$(NDX%)+COMMA$;
IF DTYPE% = 2 THEN PRINT #3,ANI.DEV$(NDX%)+COMMA$;
IF DTYPE% = 3 THEN PRINT #3,STR$(ANI.UNIT$(NDX%))+COMMA$;
IF DTYPE% = 4 THEN PRINT #3,STR$(ANI.CHNL$(NDX%))+COMMA$;
IF DTYPE% = 5 THEN PRINT #3,ANI.CNVTP$(NDX%)+COMMA$;
IF DTYPE% = 6 THEN PRINT #3,STR$(ANI.COEFA(NDX%))+COMMA$;
IF DTYPE% = 7 THEN PRINT #3,STR$(ANI.COEFB(NDX%))+COMMA$;

```

```

IF DTYPE% = 8 THEN PRINT #3,STR$(ANI.COEFC(NDX%))+COMMA$;
ELSE
IF DTYPE% = 1 THEN PRINT #3,ANI.EUNITS$(NDX%)
IF DTYPE% = 2 THEN PRINT #3,ANI.DEV$(NDX%)
IF DTYPE% = 3 THEN PRINT #3,STR$(ANI.UNIT$(NDX%))
IF DTYPE% = 4 THEN PRINT #3,STR$(ANI.CHNL$(NDX%))
IF DTYPE% = 5 THEN PRINT #3,ANI.CNVTP$(NDX%)
IF DTYPE% = 6 THEN PRINT #3,STR$(ANI.COEFA(NDX%))
IF DTYPE% = 7 THEN PRINT #3,STR$(ANI.COEFB(NDX%))
IF DTYPE% = 8 THEN PRINT #3,STR$(ANI.COEFC(NDX%))
END IF

P2:  NEXT NN%
      NEXT DTYPE%

PRINT #3,"<EOH>"
CLOSE #3
DISK.HDR%=TRUE%

DISKDATA:
PDLTIME#=TIMER*WARP
IF PDLTIME# > 86400# THEN
PDLTIME!=(PDLTIME# / 86400# - FIX(PDLTIME# / 86400#))*86400#
ELSE PDLTIME!=PDLTIME#
END IF
PDLTIME! = PDLTIME! / 86400 + JULIAN.DATE%

IF PDLTIME! < DLTIME! GOTO DSKLGRTN

OPEN DATA.FILE$ FOR APPEND AS #3

'Get the Julian date with 5 decimal places.
JTIM#=TIMER/86400#
GOSUB JULIE
JULIAN.DATE$ = STR$(JULIAN.DATE%)
JULEN%=LEN(JULIAN.DATE$) 'GET LENGTH OF STRING
JDAY#=JULIAN.DATE%
DATTIM#=JDAY#+JTIM#           'PUT DATE & TIME TOGETHER
DATTIM$=STR$(DATTIM#)
LENDATTIM%=LEN(DATTIM$)
DAT.TIMES$=RIGHT$(DATTIM$,LENDATTIM%-1) 'DROP THE LEADING BLANK
ICUT%=INSTR(DAT.TIMES$,".") 'FIND THE DECIMAL POINT
IF ICUT%=0 THEN DAT.TIMES$=DAT.TIMES$+".0000000"
IF JULEN%-1=3 THEN DAT.TIMES$=DAT.TIMES$
IF JULEN%-1=2 THEN DAT.TIMES$="0"+DAT.TIMES$
IF JULEN%-1=1 THEN DAT.TIMES$="00"+DAT.TIMES$
DAT.TIM$=LEFT$(DAT.TIMES$,9)      'NEED 5 PLACES AFTER DECIMAL POINT
'End Get the Julian date

PRINT #3,DAT.TIM$+COMMA$;

FOR NN%=1 TO NUMPNT%
IF PNT.TYPES$(NN%) <> "ANI" THEN GOTO NOTAI

```



```

NDX%=-1*TYPE.PNTR%(NN%)
IF TYPE.PNTR%(NN%) <> NUM.ANI% THEN
PRINT #3,STR$(ANI.VALUE(NDX%))+COMMAS$;
ELSE
PRINT #3,STR$(ANI.VALUE(NDX%))
'print #3,STR$(COS.Z)
END IF
NOTAI: NEXT NN%

CLOSE #3

DLTIME!=PDLTIME! + DLINTRVL! / 86400

DSKLGRTN: RETURN

'+++++

'***** END OF DISK LOGGER *****
'
'***** INPUT STRING INTERPRETER *****

INSTRINTP:
IF INPUT.STRING$ = NULL$ THEN RETURN
INVALID.QUERY%=FALSE%
IF INPUT.STRING$ <> QUIT$ GOTO EXITCHK
PROGRAM.END%=TRUE%
RETURN
EXITCHK:
IF INPUT.STRING$ <> EXIT$ GOTO PREVCHK
GOSUB SCNEXT 'EXIT HANDLER
RETURN
PREVCHK:
IF INPUT.STRING$ <> PREV$ AND ASC(INPUT.STRING$)<>CARRIAGE.RETURN%_
GOTO NOTLEGAL
GOSUB PREVCR 'PREV OR <CR> HANDLER
RETURN
NOTLEGAL:
STRING.INTERP%=TRUE% 'FLAG SET - KEYBOARD INPUT IS NONE OF THE ABOVE
RETURN

'***** END OF INPUT STRING INTERPRETER *****
'
'***** SYSTEM STATUS LINE OUTPUT *****

STATLN: 'ENTRY POINT
LOCATE STATUS.LINE%,1
PRINT "STATUS: ";
STATUS.LINE$=""
IF SYS.STATUS$ <> "" THEN STATUS.LINE$=STATUS.LINE$+SYS.STATUS$
IF SYS.STATUS$ <> "" THEN STATUS.LINE$=STATUS.LINE$+","
'STATUS.LINE$=STATUS.LINE$+PRINT.STATUS$+","
'STATUS.LINE$=STATUS.LINE$+DISK.STATUS$
LOCATE STATUS.LINE%,9

```

```

IF SYS.STATUS$ = "SYSTEM STOPPED" THEN
  COLOR 12
ELSE
  COLOR 10
END IF

```

```

PRINT STATUS.LINE$;
COLOR SCR.N.COLOR%

```

```

IF PRINT.STATUS$ = PRINTER.DISABLE$ THEN
  COLOR 12
ELSE
  COLOR 10
END IF

```

```

PRINT PRINT.STATUS$+", ";
COLOR SCR.N.COLOR%

```

```

IF DISK.STATUS$ = DISK.DISABLE$ THEN
  COLOR 12
ELSE
  COLOR 10
END IF

```

```

PRINT DISK.STATUS$
COLOR SCR.N.COLOR%

```

```

IF (INVALID.QUERY%=FALSE% AND CURR.SCREEN%=0) THEN
  END IF
  STATUS.CHANGE%=FALSE%
  SCREEN.CHANGE%=FALSE%

```

```

39090 RETURN

```

```

***** END OF SYSTEM STATUS LINE OUTPUT *****

```

```

,

```

```

***** SCREEN STATIC INFO RETRIEVE & DISPLAY ROUTINE *****

```

```

SHWSTATIC:

```

```

  SCREEN.CHANGE%=TRUE%
  SCR.N.STATIC.FILE$=SCNDRIVE$+FILE.NAMES$
  ,

```

```

  OPEN SCR.N.STATIC.FILE$ FOR INPUT AS #15

```

```

NXTSTLN:

```

```

  IF EOF(15) GOTO CLSSTFL
  INPUT #15,RLOC%,CLOC%,DYE%,SCREEN.TEXT$
  LOCATE RLOC%,CLOC%
  COLOR DYE%
  PRINT SCREEN.TEXT$;
  COLOR 15
  GOTO NXTSTLN

```

```

CLSSTFL:

```

```

  CLOSE #15          'CLOSE THE STATIC DATA FILE.

```

STATRTN:

RETURN

'***** END OF STATIC INFO RETRIEVE & DISPLAY ROUTINE *****'

'***** ANALOG OUTPUT SECTION *****'

' See Micristarr scan section

'***** END OF ANALOG OUTPUT SECTION *****'

'***** ANALOG CONTROLLER SECTION *****'

CONTROLLER.OUTPUT:

IF CONTROL.FLAG% = FALSE% THEN RETURN
CONTROL.FLAG% = FALSE%

FOR CNTL%=1 TO NUM.ANO%

IF ANO.MAN.OUTPUT%(CNTL%) = TRUE% THEN GOTO C.OUT

IF PNT.NAMES\$(ANO.TBL%(CNTL%)) = "LAMPFLUX" THEN

'PI Algorithm for Lamp Flux

LAMP.SETPNT = AMBFLUX.AVG*((FLUX.PCT.ABOVE.AMB/100.0) + 1.0)

KP = ANO.KP(CNTL%)

R = ANO.KI(CNTL%)

A = R*T*T/240.0

B = R*T*T/120.0

ER.2 = ER.1

ER.1 = ER

SP.PV.DIFF = LAMP.SETPNT - FLUX.AVG

ER = SP.PV.DIFF

ERRSUM = ERRSUM + SP.PV.DIFF

IF ABS(ERRSUM) > WNDUPLMT THEN ERRSUM = WNDUPLMT*SGN(ERRSUM)

IF AMBFLUX.AVG < ARCPLASMA.TURN.ON THEN

CNTLOUT(CNTL%) = 0.0

FACILITY.ON% = FALSE% 'Disable Disk Storage

ELSE

 'CNTLOUT(CNTL%) = CNTLOUT(CNTL%) + ANO.KP(CNTL%)*_
 (SP.PV.DIFF+(ANO.KI(CNTL%)*ERRSUM)) CNTLOUT(CNTL%) = CNTLOUT(CNTL%) +_
 KP*((A+T/2)*ER+ B*ER.1+(A+T/2)*ER.2)

FACILITY.ON% = TRUE% 'Enable Disk Storage

END IF

ELSE

```

IF PNT.NAMES$(ANO.TBL%(CNTL%)) = "LAMPFIL" THEN

    FOR TEMP.PTR% = 1 TO NUM.ANI%
        IF PNT.NAMES$(ANI.TBL%(TEMP.PTR%)) = "AIRTEMP" THEN
            A.TEMP = ANI.VALUE(TEMP PTR%)
            GOTO TEMP.ALG
        END IF
    NEXT TEMP.PTR%

TEMP.ALG:      'Temperature Algorithm for Lamp Filament
                'CNTLOUT(CNTL%) = (-1.25*A.TEMP) + 186.25
                ' IF AMBFLUX.AVG < FILAMENT.TURN.ON THEN
                '   CNTLOUT(CNTL%) = 0.0
                ' ELSE
                '   CNTLOUT(CNTL%) = 100.0
                ' END IF

                CNTLOUT(CNTL%) = 100.0

            END IF

        END IF

        IF CNTLOUT(CNTL%) > 100.0 THEN CNTLOUT(CNTL%) = 100.0
        IF PNT.NAMES$(ANO.TBL%(CNTL%)) = "LAMPFIL" THEN
            IF CNTLOUT(CNTL%) > 90.0 THEN CNTLOUT(CNTL%) = 90.
        END IF
        IF CNTLOUT(CNTL%) < 0.0 THEN CNTLOUT(CNTL%) = 0.0

C.OUT: DAOUT% = CNTLOUT(CNTL%) * CPPCT

        DAH% = INT(DAOUT% / 256)
        DAL% = DAOUT% - DAH% * 256

        OUT DDAO6BASE% + 2 * (ANO.CHNL%(CNTL%)), DAL%
        OUT DDAO6BASE% + 1 + 2 * (ANO.CHNL%(CNTL%)), DAH%

        ANO.VALUE(CNTL%) = CNTLOUT(CNTL%)

    NEXT CNTL%

    RETURN

    ***** END OF ANALOG CONTROLLER SECTION *****
    ,
    ,
    ***** BEGIN METRABYTE ADC CONVERSION *****
    ,

METRA.SCAN:

```

```
8501  'DEBUG--LINE #
      ADC.LOOP% = ADC.LOOP% + 1
```

```
ADC.ANI.LOOP:
```

```
IF ADC.LOOP% > NUM.ANI% THEN
  ADC.LOOP% = 0
  OK.TRND1% = TRUE% 'Let's trend know data collected 1st time.
  RETURN
END IF
```

```
ANI.PTR% = ANI.TBL%(ADC.LOOP%)
```

```
PRESENT.TIME1# = TIMER * WARP 'GET CURRENT TIME
PRESENT.TIME1! = PRESENT.TIME1# / 86400# + JULIAN.DATE%
```

```
IF PRESENT.TIME1! < ANI.SAMTIM!(ANI.PTR%) THEN GOTO METRA.SCAN
```

```
IF ANI.DEV$(ADC.LOOP%) = "CAL" THEN GOTO CALCULATE.ANI
```

```
8503  'DEBUG--LINE #
```

```
IF OFFICE% THEN
  NEWDAT% = INT (409.6 * RND)
  GOTO ADC.EGU
END IF
```

```
'Start Analog to Digital Converter Routine
FOR READING = 1 TO NUM.SAMPS%
  OUT DASH8BASE%+2, ANI.CHNL%(ANI.PTR%)*16+ANI.UNIT%(ANI.PTR%)
  OUT DASH8BASE%+1, START12BIT%
```

```
ADCRDY:   EOC% = INP(DASH8BASE%+2) AND &H80
          IF EOC% GOTO ADCRDY
```

```
'End Analog to Digital Converter Routine
LO% = INP(DASH8BASE%)
HI% = INP(DASH8BASE%+1)
COUNTS% = HI% * 16 + LO% / 16
'LPRINT PNT.NAMES$(ANI.PTR%), " = "; COUNTS%
```

```
CNTS%(READING) = COUNTS%
NEXT READING
```

```
GOSUB SORT.SAMP
```

```
NEWDAT.1:
  NEWDAT% = COUNTS%
```

```

LOCATE 20,40
'PRINT NEWDAT%;
'PRINT PNT.NAME$(ANI.PTR%)," = ",NEWDAT%
669
L = LEN (PNT.NAME$(ANI.PTR%))
IF LEFT$(PNT.NAME$(ANI.PTR%),L-2) = "AMBFLUX" THEN
  CONTROL.FLAG% = TRUE%
END IF
670

GOTO ADC.EGU

CALCULATE.ANI:

IF OK.TRND1% = FALSE% THEN GOTO SKPIT 'Has data been collected once?

'IF PNT.NAME$(ANI.PTR%) = "RAWUVXAMB" THEN ANI.VALUE(ANI.PTR%) = UVXA
'IF PNT.NAME$(ANI.PTR%) = "RAWUVXCA" THEN ANI.VALUE(ANI.PTR%) = UVXC
'IF PNT.NAME$(ANI.PTR%) = "MYAVG" THEN ANI.VALUE(ANI.PTR%) = MYFLUX.AVG
'IF PNT.NAME$(ANI.PTR%) = "AMBAVG" THEN ANI.VALUE(ANI.PTR%) = AMBFLUX.AVG
'IF PNT.NAME$(ANI.PTR%) = "CAMYAVG" THEN ANI.VALUE(ANI.PTR%) = FLUX.AVG

IF PNT.NAME$(ANI.PTR%) = "CAAVG" THEN ANI.VALUE(ANI.PTR%) = CAFLUX.AVG

'Begin calculating Percent Shading from the Quantum Sensors

IF PNT.NAME$(ANI.PTR%) = "%SHADE" THEN

FOR Q.CNT% = 1 TO NUM.ANI%
  IF PNT.NAME$(ANI.TBL%(Q.CNT%)) = "QUANTUM01" THEN
    Q.SHADE = ANI.VALUE(ANI.TBL%(Q.CNT%))
  END IF
  IF PNT.NAME$(ANI.TBL%(Q.CNT%)) = "QUANTUM02" THEN
    Q.AMBIENT = ANI.VALUE(ANI.TBL%(Q.CNT%))
  END IF
NEXT Q.CNT%

SHADE = 1 - (Q.SHADE/Q.AMBIENT)
IF SHADE < 0.0 THEN SHADE = 0.0
ANI.VALUE(ANI.PTR%) = SHADE * 100.0

END IF

'End calculating Percent Shading from the Quantum Sensors

IF PNT.NAME$(ANI.PTR%) = "CNTRLPCT" THEN
  IF AMBFLUX.AVG < ARCPLASMA.TURN.ON THEN
    ANI.VALUE(ANI.PTR%) = 0.0
    GOTO AMB.CTL.ZERO
  END IF
  ANI.VALUE(ANI.PTR%) = ((FLUX.AVG - AMBFLUX.AVG) / AMBFLUX.AVG) * 100

```

AMB.CTL.ZERO:

END IF

IF PNT.NAME\$(ANI.PTR%) = "SETPOINT" THEN
ANI.VALUE(ANI.PTR%) = FLUX.PCT.ABOVE.AMB
END IF

GOSUB ANI.MAX.MIN

GOTO SKPIT

***** SORT ROUTINE USED TO AVERAGE METRABYTE READINGS

SORT.SAMP:

FOR SORT= 1 TO NUM.SAMPS%

FOR ORDER=1 TO NUM.SAMPS%

IF CNTS%(ORDER) < CNTS%(SORT) GOTO 47260

TEMP=CNTS%(SORT) : CNTS%(SORT)=CNTS%(ORDER) : CNTS%(ORDER)=TEMP

47260 NEXT ORDER

NEXT SORT

TALLEY% = 0

COUNTS% = 0

FOR SUMLP% = 2 TO (NUM.SAMPS% - 1)

COUNTS% = COUNTS% + CNTS%(SUMLP%)

TALLEY% = TALLEY% + 1

NEXT SUMLP%

COUNTS%=COUNTS%/TALLEY%

RETURN

***** END SORT ROUTINE

RETURN

***** END METRABYTE ADC CONVERSION *****

' BEGIN METRABYTE ADC EGU CONVERSION ROUTINES

ADC.EGU:

AUVTYPE:

IF ANI.CNVTYPE\$(ANI.PTR%) <> "AUV" THEN GOTO CUVTYPE
ICNTS1%=NEWDAT%

```

ANICA = 0.0
ANICB = 10.0 'For Raw Value Full Scale of UVX Meter (20 watt/sq.meter
ANICC = 0.0
ADC% = ANI.UNIT%(ANI.PTR%)
CALL CNVRTLIN(ICNTS1%,ANICA,ANICB,ANICC,ANIVAL,ADC%) 'CONVERT THE DATA
UVXA = ANIVAL
'LOCATE 20,40
'PRINT "UVXA = ";UVXA;
UVXA.CORRECTION.FACTOR = ANI.COEFB(ANI.PTR%)
ANI.VALUE(ANI.PTR%)= UVXA*UVXA.CORRECTION.FACTOR
GOSUB ANI.MAX.MIN
GOTO SKPIT

```

CUVTYPE:

```

IF ANI.CNVTP$(ANI.PTR%) <> "CUV" THEN GOTO TNRTYPE
ICNTS1%=NEW DAT%
ANICA = 0.0
ANICB = 10.0 'For Raw Value Full Scale of UVX Meter (20 watt/sq.meter
ANICC = 0.0
ADC% = ANI.UNIT%(ANI.PTR%)
CALL CNVRTLIN(ICNTS1%,ANICA,ANICB,ANICC,ANIVAL,ADC%) 'CONVERT THE DATA
UVXC = ANIVAL
'LOCATE 21,40
'PRINT "UVXC = ";UVXC;
UVXC.CORRECTION.FACTOR = ANI.COEFB(ANI.PTR%)
ANI.VALUE(ANI.PTR%)= UVXA.CORRECTION.FACTOR*UVXA*(1-SHADE) +
UVXC.CORRECTION.FACTOR*(UVXC-UVXA*(1-SHADE))
GOSUB ANI.MAX.MIN
GOTO SKPIT

```

TNRTYPE:

```

IF ANI.CNVTP$(ANI.PTR%) <> "TNR" THEN GOTO LTYPE
ICNTS1%=NEW DAT%
EU$=ANI.EUNIT$(TYPE.PNTR%(ANI.PTR%))
CALL CNVRTTTC(ICNTS1%,EU$,ANIVAL) 'CONVERT THE DATA
ANI.VALUE(ANI.PTR%)=ANIVAL 'STORE THE DATA
GOSUB ANI.MAX.MIN
GOTO SKPIT

```

LTYPE: IF ((ANI.CNVTP\$(ANI.PTR%) <> "LIN") AND _
(ANI.CNVTP\$(ANI.PTR%) <> "COS")) THEN GOTO SKPIT 'LINEAR POINT

```

LTYPE1: ICNTS1%=NEW DAT%
EU$=ANI.EUNIT$(ANI.PTR%)
ANICA=ANI.COEFA(ANI.PTR%);ANICB=ANI.COEFB(ANI.PTR%)
ANICC=ANI.COEF(ANI.PTR%)
ADC% = ANI.UNIT%(ANI.PTR%)
CALL CNVRTLIN(ICNTS1%,ANICA,ANICB,ANICC,ANIVAL,ADC%) 'CONVERT THE DATA
ANI.VALUE(ANI.PTR%)=ANIVAL 'STORE THE DATA

```

```

47261 IF ANI.CNVTP$(ANI.PTR%) = "COS" THEN
      GOSUB COSCORR.SUB
      'LOCATE 18,30:PRINT "COS.Z = ";COS.Z"
47262 ANI.VALUE(ANI.PTR%) = ANIVAL / COS.Z

```



```

ELSE
    ANI.VALUE(ANI.PTR%) = ANIVAL
END IF

GOSUB ANI.MAX.MIN
'LPRINT "NAME= ";PNT.NAMES$(ANI.PTR%); " COUNTS = ";ICNTS1%;_
"ANIVAL = ";ANIVAL;" CHAN=";ICHAN2%
GOTO SKPIT

```

SKPIT:

' Increment new sample update time

ANI.SAMTIM!(ANI.PTR%)=PRESENT.TIME1! + ANI.SCAN%(ANI.PTR%)/86400

RETURN 'Go back to Idle loop, then return to ADC section.
 'We are doing this for now to keep Data Acquisition from
 'choking the Screen Display and Key board Inputs.

```

'-----
'
'      BEGIN METRABYTE ADC EGU CONVERSION SUBROUTINES
'
'      *****

```

```

SUB CNVRTTTC(COUNTS%,ENGUS$,DEG) STATIC 'CONVERSION FOR "T" TC'S
'LOCATE 20,40 : PRINT COUNTS%
A0#=0.100860910
A1#=25727.94369
A2#=-767345.8295
A3#=78025595.81
A4#=-9247486589
A5#=6.97688D11
A6#=-2.66192D13
A7#=3.94078D14

```

```

MAXCOUNTS = 2047.5 ' Max counts per each 5 volt span
MAXVLTS=5.0
MINVLTS=-5.0

```

VOLTS#=(COUNTS%-MAXCOUNTS)*MAXVLTS/MAXCOUNTS

' Type T transmitter outputs 4-20 ma for -100 to 300 degrees C.

```

MVIN#=(4.5595*VOLTS# - 7.9375)/1000.0
X=MVIN#

```

```

TMPTTC=A1#+X*(A2#+X*(A3#+X*(A4#+X*(A5#+X*(A6#+A7#*X))))
TMPTTC=A0#+X*TMPTTC
DEG.C=TMPTTC
DEG.F=(1.8*TMPTTC)+32.0

```

IF ENGUS\$=CDEG\$ THEN

```

      DEG=DEG.C
ELSE
      DEG = DEG.F
END IF

```

```

EXIT SUB
END SUB

```

```

*****

```

```

SUB CNVRTLIN(CNTS%,COFA,COFB,COFC,VALU,ADC%) STATIC      'LINEAR CNVRT.

```

```

      IF (ADC% = 1) THEN AGIN = 1.0
      IF (ADC% = 2) THEN AGIN = 100.0
      AGIN = 1.0
      MAXVLTS=5.0
      MINVLTS=-5.0
      CAMACCNT! = 2047.5

      VOLTS#=(CNTS%-CAMACCNT!)*MAXVLTS/CAMACCNT!
      VINN=VOLTS#*1.0/AGIN          '5.0 V = FULL SCALE
      XIN=VINN
      VALU=(XIN*XIN*COFA)+(XIN*COFB)+(COFC)
      EXIT SUB
      END SUB

```

```

*****

```

```

COSCORR.SUB:

```

```

      DEC.ANG = 23.45*SIN((RAD.CNV*360/365)*(284+JULIAN.DATE%))

```

```

      B.1 = (360/364)*(JULIAN.DATE%-81)

```

```

      EQU.TIM = 9.87*SIN(2*B.1*RAD.CNV) - 7.53*COS(B.1*RAD.CNV)_
               - 1.5*SIN(B.1*RAD.CNV)

```

```

      H.MIN = (TIMER / 60) - DLST*60 + 4*(75.0-LONG.DEG) + EQU.TIM

```

```

' TAKE'S 4 MINUTES FOR THE SUN TO TRANSVERSE 1 DEGREE OF LONGITUDE

```

```

      MIN.FROM.NOON = (H.MIN-12*60)

```

```

      DEG.FROM.NOON = MIN.FROM.NOON/4

```

```

      COS.Z = SIN(LAT.DEG*RAD.CNV)*SIN(DEC.ANG*RAD.CNV)+_
              COS(LAT.DEG*RAD.CNV)*COS(DEC.ANG*RAD.CNV)*_
              COS(DEG.FROM.NOON*RAD.CNV)

```

```

      'SIN.Z = SQR(1 - COS.Z^2)

```

```

      'TAN.Z = SIN.Z/COS.Z

```

```

      'Z.ANG = ATN(TAN.Z)          'ZENITH ANGLE IN RADIANS

```

```

      'Z.ANG = Z.ANG/RAD.CNV      'ZENITH ANGLE IN DEGREES

```

```

      'LOCATE 17,25:PRINT "COS.Z = ",COS.Z

```

```
IF ABS(COS.Z) < .01 THEN COS.Z = .01 ' AVOID DIVIDING BY ZERO
IF COS.Z < 0 THEN COS.Z = .01      ' NIGHT TIME IS HERE!
```

```
COS.Z = 1      ' ONLY NOW FOR SYSTEM CHECKOUT !
```

```
RETURN
```

```
*****
```

```
END CAMAC ADC EGU CONVERSION SUBROUTINES
```

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
```

```
***** MEASURED DATA MAX,MIN,AVERAGE, & VARIANCE COMPUTATION *****
```

```
ANI.MAX.MIN:
```

```
IF ANI.VALUE!(ANI.PTR%) > ANI.MAXVAL!(ANI.PTR%) THEN_
  ANI.MAXVAL!(ANI.PTR%)=ANI.VALUE!(ANI.PTR%) 'MAXIMUM
IF ANI.VALUE!(ANI.PTR%) < ANI.MINVAL!(ANI.PTR%) THEN_
  ANI.MINVAL!(ANI.PTR%)=ANI.VALUE!(ANI.PTR%) 'MINIMUM
```

```
'RETURN      'FOR NOW WHILE I CALIBRATE UV-B SENSOR
```

```
' Average the ambient UV-B Flux sensors, and Average the UV-B Flux
' sensors under the lamps.
```

```
MY.PTR% = 0
CA.PTR% = 0
AMB.PTR% = 0
```

```
MYFLUX.TOT = 0
CAFLUX.TOT = 0
AMBFLUX.TOT = 0
```

```
FOR AVG.PTR% = 1 TO NUM.ANI%
```

```
  L = LEN (PNT.NAMES$(ANI.TBL%(AVG.PTR%)))
```

```
  IF LEFT$(PNT.NAMES$(ANI.TBL%(AVG.PTR%)),L-2) = "MYFLUX" THEN
    MY.PTR% = MY.PTR% + 1
    MYFLUX.TOT = MYFLUX.TOT + ANI.VALUE(AVG.PTR%)
  END IF
```

```
  IF LEFT$(PNT.NAMES$(ANI.TBL%(AVG.PTR%)),L-2) = "CAFLUX" THEN
    CA.PTR% = CA.PTR% + 1
    CAFLUX.TOT = CAFLUX.TOT + ANI.VALUE(AVG.PTR%)
  END IF
```

```

IF LEFT$(PNT.NAMES$(ANI.TBL%(AVG.PTR%)),L-2) = "AMBFLUX" THEN
    AMB.PTR% = AMB.PTR% + 1
    AMBFLUX.TOT = AMBFLUX.TOT + ANI.VALUE(AVG.PTR%)
END IF

```

```

NEXT AVG.PTR%

```

```

' Digital Filter to Smooth UVB Readings and Help Control

```

```

CAFLUX.5 = CAFLUX.4
CAFLUX.4 = CAFLUX.3
CAFLUX.3 = CAFLUX.2
CAFLUX.2 = CAFLUX.AVG
CAFLUX.OLD = (CAFLUX.2+CAFLUX.3+CAFLUX.4+CAFLUX.5)/4
CAFLUX.AVG = CAFLUX.OLD*(4/5)+(CAFLUX.TOT/CA.PTR%)*(1/5)

AMBFLUX.5 = AMBFLUX.4
AMBFLUX.4 = AMBFLUX.3
AMBFLUX.3 = AMBFLUX.2
AMBFLUX.2 = AMBFLUX.AVG
AMBFLUX.OLD = (AMBFLUX.2+AMBFLUX.3+AMBFLUX.4+AMBFLUX.5)/4
AMBFLUX.AVG = AMBFLUX.OLD*(4/5)+(AMBFLUX.TOT/AMB.PTR%)*(1/5)

```

```

'MYFLUX.AVG = MYFLUX.TOT / MY.PTR%
'CAFLUX.AVG = CAFLUX.TOT / CA.PTR%
'AMBFLUX.AVG = AMBFLUX.TOT / AMB.PTR%

```

```

FLUX.AVG = CAFLUX.AVG

```

```

RETURN

```

```

***** END OF MAX,MIN,AVERAGE & VARIANCE COMP ROUTINE *****

```

```

***** DIGITAL INPUT SECTION *****

```

```

' None used presntly

```

```

***** END OF DIGITAL INPUT SECTION *****

```

```

***** DIGITAL OUTPUT SECTION *****

```

```

' None used presently

```

```

***** END OF DIGITAL OUTPUT SECTION *****

```

```

***** JULIAN DATE ROUTINE *****

```

```

JULIE:

```

```

IF LY1ST% THEN GOTO JULIE1

```

```

LY1ST% = TRUE%
LYF=VAL(MID$(DATE$,9,2)) MOD 4
IF LYF > 0 GOTO JULIE1
FOR MLOOP%=3 TO 12
EOPM%(MLOOP%)=EOPM%(MLOOP%)+1
NEXT MLOOP%
JULIE1: DATECK#=TIMER*WARP
IF DATECK# >86400# THEN
DATECK#=(DATECK# / 86400# - FIX(DATECK# / 86400#))*86400#
END IF
MTH%=VAL(MID$(DATE$,1,2))
DAY%=VAL(MID$(DATE$,4,2))
NOW#=TIMER*WARP
IF DATECK# > ( NOW# / 86400# - FIX(NOW# / 86400#))*86400# GOTO JULIE1
JULIAN.DATE%=EOPM%(MTH%) + DAY%
RETURN

***** END OF JULIAN DATE ROUTINE *****
,
***** SYSTEM EXIT ROUTINE *****

54151 'ENTRY POINT
'CLEAR (NECESSARY ??)
54152 CLS
,
'ANY OTHER WORK TO DO ??
,

54157 END 'NOTE: THE END STATEMENT CLOSSES ALL FILES

***** END OF SYSTEM EXIT ROUTINE *****
,
***** BEGIN SYSTEM ERROR HANDLER *****

ERRHANDLER: 'ERROR HANDLER FOR DIRECTORY CREATION & FILE OPEN
LPRINT DATE$," ";TIME$," ERROR = ";ERR;" IN LAST LINE # ";ERL
IF ERR = 75 THEN RESUME NEXT 'DIRECTORY ALREADY EXISTS
IF ERR = 53 THEN GOTO IDLE 'FILE ALREADY EXISTS

IF ERR = 57 THEN      'DEVICE I/O ERROR -- COMM ERROR ???
NAKFLG%=FALSE%
MIC.GD.VAL%=FALSE%
ERROUT.FLAG%=FALSE%
TIMEOUT.FLAG%=FALSE%
NAKOUT.FLAG%=FALSE%
AWAIT.REPLY%=FALSE%
RCVD2$ = ""
BRCV2$ = ""
RESUME IDLE
END IF

IF ERR = 6 THEN      'OVER FLOW ERROR
'LPRINT "OVERFLOW---MSG2$= ";MSG2$
RESUME IDLE

```

END IF

LPRINT DATE\$," ";TIME\$_
; " ERROR IN ERRHANDLER BUT ERROR NOT ANTICIPATED...CONTINUING
RESUME IDLE

***** END SYSTEM ERROR HANDLER *****

***** BEGIN DISK ERROR HANDLER *****

DISK.ERR.HANDLER:

LPRINT DATE\$," ";TIME\$," [DISK.ERR.HANDLER] ";DATADRIVE\$," DISK ERROR = ";ERR

IF ERR = 61 THEN LPRINT DATE\$;TIME\$," [DISK.ERR.HANDLER] ";DATADRIVE\$," DISK
FULL"

IF ERR = 71 THEN LPRINT DATE\$;TIME\$," [DISK.ERR.HANDLER] ";DATADRIVE\$," DISK NOT
READY"

IF ERR = 72 THEN LPRINT DATE\$;TIME\$," [DISK.ERR.HANDLER] ";DATADRIVE\$," DISK
FLAW DETECTED"

IF DATADRIVE\$ = EMR.DRIVE\$ THEN PROGRAM.END%=TRUE%

DATADRIVE\$ = ALTDRIVE\$

ALTDRIVE\$ = EMR.DRIVE\$

DISK.HDR% = FALSE%

RESUME IDLE

***** END DISK ERROR HANDLER *****

54280 '

55000 'ENTRY POINT FOR PROCESSING PDB TABLES & CREATION OF DISK FILE HEADER

55010 '

55020 FOR K%=1 TO NUMPNT%

IF PNT.TYPE\$(K%) <> "ANI" GOTO 55030

IF TYPE.PNTR%(K%) <> NUM.ANI% THEN

PDB.STRING\$=PDB.STRING\$+PNT.NAMES\$(K%)+COMMA\$

ELSE

PDB.STRING\$=PDB.STRING\$+PNT.NAMES\$(K%)

END IF

GOTO 55040

55030 IF PNT.TYPE\$(K%) <> "TXT" GOTO 55040

OLDPDB.STRING\$=PDB.STRING\$

PDB.STRING\$=TXT.TEXT\$(TYPE.PNTR%(K%))

GOSUB HEADER:

PDB.STRING\$=OLDPDB.STRING\$

55040 NEXT K%

GOSUB HEADER: 'OUTPUT PDB.STRING\$ (RECORD #1 OF HEADER)

PDB.STRING\$=""

FOR K%=1 TO NUMPNT%

IF PNT.TYPE\$(K%) <> "ANI" GOTO 55050

IF TYPE.PNTR%(K%) <> NUM.ANI% THEN

```

    PDB.STRING$=PDB.STRING$+ANI.EUNIT$(TYPE.PNTR%(K%))+COMMA$
ELSE
    PDB.STRING$=PDB.STRING$+ANI.EUNIT$(TYPE.PNTR%(K%))
END IF
55050 NEXT K%
    GOSUB HEADER: 'OUTPUT PDB.STRING$ (RECORD #2 OF HEADER)
    PDB.STRING$=""
    FOR K%=1 TO NUMPNT%
    IF PNT.TYPE$(K%) <> "ANI" GOTO 55060
    IF TYPE.PNTR%(K%) <> NUM.ANI% THEN
        PDB.STRING$=PDB.STRING$+ANI.DEV$(TYPE.PNTR%(K%))+COMMA$
    ELSE
        PDB.STRING$=PDB.STRING$+ANI.DEV$(TYPE.PNTR%(K%))
    END IF
55060 NEXT K%
    GOSUB HEADER: 'OUTPUT PDB.STRING$ (RECORD #3 OF HEADER)
    PDB.STRING$=""
    FOR K%=1 TO NUMPNT%
    IF PNT.TYPE$(K%) <> "ANI" GOTO 55070
    IF TYPE.PNTR%(K%) <> NUM.ANI% THEN
        PDB.STRING$=PDB.STRING$+STR$(ANI.UNIT%(TYPE.PNTR%(K%)))+COMMA$
    ELSE
        PDB.STRING$=PDB.STRING$+STR$(ANI.UNIT%(TYPE.PNTR%(K%)))
    END IF
55070 NEXT K%
    GOSUB HEADER: 'OUTPUT PDB.STRING$ (RECORD #4 OF HEADER)
    PDB.STRING$=""
    FOR K%=1 TO NUMPNT%
    IF PNT.TYPE$(K%) <> "ANI" GOTO 55080
    IF TYPE.PNTR%(K%) <> NUM.ANI% THEN
        PDB.STRING$=PDB.STRING$+STR$(ANI.CHNL%(TYPE.PNTR%(K%)))+COMMA$
    ELSE
        PDB.STRING$=PDB.STRING$+STR$(ANI.CHNL%(TYPE.PNTR%(K%)))
    END IF
55080 NEXT K%
    GOSUB HEADER: 'OUTPUT PDB.STRING$ (RECORD #5 OF HEADER)
    PDB.STRING$=""
    FOR K%=1 TO NUMPNT%
    IF PNT.TYPE$(K%) <> "ANI" GOTO 55090
    IF TYPE.PNTR%(K%) <> NUM.ANI% THEN
        PDB.STRING$=PDB.STRING$+ANI.CNVTYP$(TYPE.PNTR%(K%))+COMMA$
    ELSE
        PDB.STRING$=PDB.STRING$+ANI.CNVTYP$(TYPE.PNTR%(K%))
    END IF
55090 NEXT K%
    GOSUB HEADER: 'OUTPUT PDB.STRING$ (RECORD #6 OF HEADER)
    PDB.STRING$=""
    FOR K%=1 TO NUMPNT%
    IF PNT.TYPE$(K%) <> "ANI" GOTO 55100
    IF TYPE.PNTR%(K%) <> NUM.ANI% THEN
        PDB.STRING$=PDB.STRING$+STR$(ANI.COEFA!(TYPE.PNTR%(K%)))+COMMA$
    ELSE
        PDB.STRING$=PDB.STRING$+STR$(ANI.COEFA!(TYPE.PNTR%(K%)))

```

```
END IF
55100 NEXT K%
GOSUB HEADER: 'OUTPUT PDB.STRING$ (RECORD #7 OF HEADER)
PDB.STRING$=""
FOR K%=1 TO NUMPNT%
IF PNT.TYPE$(K%) <> "ANI" GOTO 55110
IF TYPE.PNTR%(K%) <> NUM.ANI% THEN
PDB.STRING$=PDB.STRING$+STR$(ANI.COEFB!(TYPE.PNTR%(K%)))+COMMA$
ELSE
PDB.STRING$=PDB.STRING$+STR$(ANI.COEFB!(TYPE.PNTR%(K%)))
END IF
55110 NEXT K%
GOSUB HEADER: 'OUTPUT PDB.STRING$ (RECORD #8 OF HEADER)
PDB.STRING$=""
FOR K%=1 TO NUMPNT%
IF PNT.TYPE$(K%) <> "ANI" GOTO 55120
IF TYPE.PNTR%(K%) <> NUM.ANI% THEN
PDB.STRING$=PDB.STRING$+STR$(ANI.COEFB!(TYPE.PNTR%(K%)))+COMMA$
ELSE
PDB.STRING$=PDB.STRING$+STR$(ANI.COEFB!(TYPE.PNTR%(K%)))
END IF
55120 NEXT K%
GOSUB HEADER: 'OUTPUT PDB.STRING$ (RECORD #9 OF HEADER)
PDB.STRING$=""
FOR K%=1 TO NUMPNT%
IF PNT.TYPE$(K%) <> "ANI" GOTO 55130
IF TYPE.PNTR%(K%) <> NUM.ANI% THEN
PDB.STRING$=PDB.STRING$+STR$(ANI.LOLIM!(TYPE.PNTR%(K%)))+COMMA$
ELSE
PDB.STRING$=PDB.STRING$+STR$(ANI.LOLIM!(TYPE.PNTR%(K%)))
END IF
55130 NEXT K%
GOSUB HEADER: 'OUTPUT PDB.STRING$ (RECORD #10 OF HEADER)
PDB.STRING$=""
FOR K%=1 TO NUMPNT%
IF PNT.TYPE$(K%) <> "ANI" GOTO 55140
IF TYPE.PNTR%(K%) <> NUM.ANI% THEN
PDB.STRING$=PDB.STRING$+STR$(ANI.HILIM!(TYPE.PNTR%(K%)))+COMMA$
ELSE
PDB.STRING$=PDB.STRING$+STR$(ANI.HILIM!(TYPE.PNTR%(K%)))
END IF
55140 NEXT K%
GOSUB HEADER: 'OUTPUT PDB.STRING$ (RECORD #11 OF HEADER)
PDB.STRING$=""
FOR K%=1 TO NUMPNT%
IF PNT.TYPE$(K%) <> "ANI" GOTO 55150
IF TYPE.PNTR%(K%) <> NUM.ANI% THEN
PDB.STRING$=PDB.STRING$+STR$(ANI.SCAN%(TYPE.PNTR%(K%)))+COMMA$
ELSE
PDB.STRING$=PDB.STRING$+STR$(ANI.SCAN%(TYPE.PNTR%(K%)))
END IF
55150 NEXT K%
GOSUB HEADER: 'OUTPUT PDB.STRING$ (RECORD #12 OF HEADER)
```



```

RETURN
HEADER:
'OUTPUT HEADER RECORD TO FILE
PRINT #1,BLANK$;PDB.STRING$
RETURN

```

```

'***** END OF OUTPUT FILE SET-UP *****

```

```

'***** STOP DATA ACQUISITION & CONTROL *****

```

```

60040 'ENTRY POINT
60050 '
60060 'CLOSE ALL RELEVANT FILES
60070 'ANYTHING ELSE TO BE DONE ??
60080 '
60090 RETURN

```

```

60100 '***** END OF STOP DATA ACQUISITION & CONTROL ROUTINE *****

```

```

CNVUP:
    INPUT.STRING$=UP$
    RETURN

```

```

CNVDN:
    INPUT.STRING$=DOWN$
    RETURN

```

```

'*****
'SUBROUTINE TO LOAD DYNAMIC DATA POINT NAMES AND SCREEN LOCATIONS.

```

```

LDDYNPNT:
    TEMP.FILES = SCRNDRIVES$+FILE.NAMES$
    OPEN TEMP.FILES FOR INPUT AS #15
    NDPNTS%=0
NXTDYNPNT:
    IF EOF(15) GOTO LDDYNPNT0
    NDPNTS%=NDPNTS%+1
    INPUT #15,DISPL.ROW%(NDPNTS%),DISPL.COLM%(NDPNTS%),DISPL.DYE%(NDPNTS%)_
        ,DISPL.NAME$(NDPNTS%)
    GOTO NXTDYNPNT
'
'LOCATE SCREEN POINT NAMES RELATIVE TO SYSTEM POINT NAME ARRAY

```

```

LDDYNPNT0:
    CLOSE #15
    FOR DLOOP%=1 TO NDPNTS%
    FOR DLOOP1%=1 TO NUNPNT%
    IF DISPL.NAME$(DLOOP%)<>PNT.NAME$(DLOOP1%)GOTO DYNPNT1
    DISPL.LOC%(DLOOP%)=DLOOP1% 'SAVE LOCATION OF POINT.
    GOTO DYNPNT2
DYNPNT1:
    NEXT DLOOP1%
DYNPNT2:
    NEXT DLOOP%

```

RETURN

*****END OF DYNAMIC POINT LOADING SUBROUTINE*****

*****SUBROUTINE TO PUT DYNAMIC POINT VALUES ON THE SCREEN*****

DSPDYNPNT:

```
FOR DLOOP%=1 TO NDPNTS%
IF DISPL.LOC%(DLOOP%)=0 GOTO NXTDSPPNT
IF PNT.TYPE$(DISPL.LOC%(DLOOP%)) <> TXT$ GOTO ANIDSP
LOCATE DISPL.ROW%(DLOOP%),DISPL.COLM%(DLOOP%)
COLOR DISPL.DYE%(DLOOP%)
PRINT TXT.TEXT$(TYPE.PNTR%(DISPL.LOC%(DLOOP%)))
COLOR 15
GOTO NXTDSPPNT
```

ANIDSP:

```
IF PNT.TYPE$(DISPL.LOC%(DLOOP%)) <> ANI$ GOTO NXTDSPPNT
LOCATE DISPL.ROW%(DLOOP%),DISPL.COLM%(DLOOP%)
COLOR DISPL.DYE%(DLOOP%)
PRINT USING F7P2$;ANI.VALUE!(TYPE.PNTR%(DISPL.LOC%(DLOOP%)))
COLOR 15
```

NXTDSPPNT:

```
NEXT DLOOP%
RETURN
```

*****END OF DYNAMIC POINT SUBROUTINE*****

*****SUBROUTINE TO MAKE NEW SCREEN ACTIVE*****

NEWSCN:

```
IF STACK.POINTER% >= STACK.SIZE% THEN GOTO STKOVF
STACK.POINTER%=STACK.POINTER%+1 'PUSH CURRENT SCREEN ON STACK
STACK%(STACK.POINTER%)=CURR.SCREEN%
SCREEN.STATIC%=FALSE%
INPUT.PROMPT%=FALSE%
QUERY.PROMPT%=FALSE%
STATUS.PROMPT%=FALSE%
'SET NEW TO NEW SCREEN.
CURR.SCREEN%=NEW.SCREEN%
INPUT.STRING$=NULL$
CLS
RETURN
'STACK OVER FLOW, CANNOT GO DEEPER IN DISPLAY LEVELS.
```

STKOVF:

```
RETURN
```

*****END OF ROUTINE TO MAKE NEW SCREEN ACTIVE*****

□

APPENDIX B
SYSTEM PROCESS DATA BASE FILE LISTING

Ultraviolet-B Research Facility Software - Process Data Base
Oak Ridge National Laboratory
Oak Ridge, TN

ANI,CAFLUX01,WATT/M2,ADC,1,8,CUV,0.0,.45238,0.0,0.0,0.0,.5
ANI,CAFLUX02,WATT/M2,ADC,1,9,CUV,0.0,.45238,0.0,0.0,0.0,.5
ANI,AMBFLUX01,WATT/M2,ADC,1,13,AUV,0.0,.19683,0.0,0.0,0.0,.5
ANI,AIRTEMP,DEGF,ADC,1,10,TNR,0.0,0.0,0.0,0.0,0.0,60
ANI,ARCVOLT,VAC-RMS,ADC,1,11,LIN,0.0,49.2886,0.0,0.0,0.0,1
ANI,FILVOLT,VAC-RMS,ADC,1,12,LIN,0.0,29.6818,0.0,0.0,0.0,1
ANI,ARCOUPTPUT,PERCENT,ADC,1,14,LIN,0.0,20.0,0.0,0.0,0.0,1
ANI,FILOUTPUT,PERCENT,ADC,1,15,LIN,0.0,20.0,0.0,0.0,0.0,1
ANI,RAWUVXAMB,WATT/M2,ADC,1,13,LIN,0.0,10.0,0.0,0.0,0.0,1
ANI,RAWUVXCA1,WATT/M2,ADC,1,8,LIN,0.0,10.0,0.0,0.0,0.0,1
ANI,RAWUVXCA2,WATT/M2,ADC,1,9,LIN,0.0,10.0,0.0,0.0,0.0,1
ANI,CAAVG,WATT/M2,CAL,0,0,CAL,0.0,0.0,0.0,0.0,0.0,.5
ANI,SETPOINT,PCT,CAL,0,0,CAL,0.0,0.0,0.0,0.0,0.0,60
ANI,CNTRLPCT,PCT,CAL,0,0,CAL,0.0,0.0,0.0,0.0,0.0,.5
ANI,%SHADE,PCT,CAL,0,0,CAL,0.0,0.0,0.0,0.0,0.0,.5
ANI,QUANTUM01,MMOL/SM2,ADC,2,0,LIN,0.0,-2028.95,0.0,0.0,0.0,10
ANI,QUANTUM02,MMOL/SM2,ADC,2,1,LIN,0.0,-2358.45,0.0,0.0,0.0,10
ANO,LAMPFIL,1,1,100.0,-1.0,0.0,0.3,0.05
ANO,LAMPFLUX,1,0,32.0,-1.0,0.0,10.0,1.0□

APPENDIX C

SYSTEM DISPLAY SCREEN CONTROL FILES

SCRN000.SCN

01,13,14,UV-B RESEARCH FACILITY DATA ACQUISITION & CONTROL SYSTEM

03,34,14,MASTER MENU

05,01,15,START	- START DATA ACQUISITION AND CONTROL
06,01,15,STOP	- STOP DATA ACQUISITION AND CONTROL
07,01,15,GRAPH	- GENERATE GRAPHICS OF SYSTEM PARAMETERS
08,01,15,CONTROL	- DISPLAY PV AND CHANGE CONTROLLER SETPOINTS
09,01,15,DATA	- DISPLAY & SCROLL THRU POINT DATA BASE
10,01,15,DISK	- CONTROL DISK LOGGING/DRIVE ASSIGNMENT
11,01,15,PRINT	- CONTROL PRINTER LOGGING
12,01,15,QUIT	- EXIT SYSTEM SOFTWARE

SCRN001.SCN

01,13,14,UV-B RESEARCH FACILITY DATA ACQUISITION & CONTROL SYSTEM

03,36,14,GRAPHICS

05,01,15,TREND	- PLOT POINT TREND DATA
06,01,15,BAR	- GENERATE BAR CHART OF POINTS
07,01,15,DIAG	- ULTRAVIOLET-B EXPOSURE SYSTEM OVERVIEW
08,01,15,PREV OR <CR>	- RETURN TO PREVIOUS SCREEN
09,01,15,EXIT	- RETURN TO MASTER SCREEN

SCRN002.SCN

01,13,14,UV-B RESEARCH FACILITY DATA ACQUISITION & CONTROL SYSTEM

02,32,14,SYSTEM OVERVIEW

18,25,15,CONTROL

19,25,15,CABINET

16,45,15,FILAMENT

17,45,15,CONTROL

20,45,15,ARC

21,45,15,CONTROL

22,01,15,OPTIONS: PREV <CR>

09,05,15,AIR TEMP.

12,17,15,% CONTROL ABOVE AMBIENT

08,67,15,AMBIENT

09,68,15,FLUX

18,59,15,% OUTPUT TO LAMPS

09,25,15,MYLAR

09,48,15,CA FLUX

SCRN003.SCN

01,13,14,UV-B RESEARCH FACILITY DATA ACQUISITION & CONTROL SYSTEM

03,32,14,PRINTER CONTROL
 05,01,15,LOG - ENABLE PRINTER OUTPUT
 06,01,15,NLOG - DISABLE PRINTER OUTPUT
 07,01,15,LOGINT - CHANGE PRINTER LOGGING INTERVAL
 08,01,15,PREV OR <CR> - RETURN TO PREVIOUS SCREEN
 09,01,15,EXIT - RETURN TO MASTER SCREEN

SCRN005.SCN

01,13,14,UV-B RESEARCH FACILITY DATA ACQUISITION & CONTROL SYSTEM

03,32,14,DISK CONTROL MENU

05,01,15,STOR - ENABLE DISK STORAGE
 06,01,15,NSTOR - DISABLE DISK STORAGE
 07,01,15,DSKASS - ALTER DISK STORAGE DRIVE ASSIGNMENTS
 08,01,15,DSKINT - ALTER DISK STORAGE INTERVAL
 09,01,15,PREV OR <CR> - RETURN TO PREVIOUS SCREEN
 10,01,15,EXIT - RETURN TO MASTER SCREEN
 13,29,14,DISK CONTROL PARAMETERS

SCRN008.SCN

01,13,14,UV-B RESEARCH FACILITY DATA ACQUISITION & CONTROL SYSTEM

22,01,01,OPTIONS: POINTID SPAN SCALE CLEAR PREV OR <CR> QUIT EXIT

SCRN012.SCN

01,36,15,
 03,01,15,
 03,48,15,
 04,01,15,
 04,45,15,
 05,45,15,
 06,01,15,
 07,01,15,
 08,01,15,
 10,01,15,
 11,01,15,
 12,01,15,
 14,01,15,
 16,01,15,
 06,43,15,
 07,43,15,
 08,43,15,
 09,43,15,
 10,43,15,
 12,43,15,
 14,43,15,
 16,43,15,
 19,01,15,OPTIONS: PREV OR <CR> EXIT

SCRN013.SCN

01,13,14,UV-B RESEARCH FACILITY DATA ACQUISITION & CONTROL SYSTEM

02,34,08,BAR DISPLAY

22,01,01,OPTIONS: GROUP SCALE PREV OR <CR> QUIT EXIT

SCRN020.SCN

01,13,14,UV-B RESEARCH FACILITY DATA ACQUISITION & CONTROL SYSTEM

02,16,14,CHANGE UV-B EXPOSURE SYSTEM CONTROLLER OUTPUT MODE

05,15,10,CONTROLLER NAME CONTROLLER OUTPUT MODE

06,15,10,----- -----

08,18,07,LAMPFIL

10,18,07,LAMPFLUX

SCRN021.SCN

01,13,14,UV-B RESEARCH FACILITY DATA ACQUISITION & CONTROL SYSTEM

02,13,14,CHANGE UV-B EXPOSURE SYSTEM CONTROLLER OUTPUT PERCENTAGE

05,08,10,CONTROLLER NAME CONTROLLER OUTPUT VALUE CONTROLLER MODE

06,08,10,----- ----- -----

08,11,07,LAMPFIL

10,11,07,LAMPFLUX

16,11,11,CONTROLLER MODE MUST BE IN "MANUAL" BEFORE CHANGING OUTPUT

INTERNAL DISTRIBUTION

1. H. R. Brashear	19. S. H. Stow
2. J. B. Cannon	20. T. J. Tschaplinski
3. N. T. Edwards	21. G. A. Tuskan
4. M. P. Farrell	22. R. E. Uhrig
5. D. N. Fry	23. R. I. Van Hook
6. C. W. Gehrs	24. S. D. Wulschleger
7. S. G. Hildebrand	25. B. Chexal, Advisor
8. M. S. Hileman	26. V. Radeka, Advisor
9. D. W. McDonald	27. R. M. Taylor, Advisor
10. J. A. McEvers	28. D. F. Craig, Advisor
11. S. B. McLaughlin	29. M. Sevik, Advisor
12. D. R. Miller	30-31. Central Research Library
13. G. N. Miller	32. Y-12 Technical Reference Section
14. M. L. Simpson	33-34. Laboratory Records
15. F. E. Sharples	35. Laboratory Records—Record Copy
16. D. S. Shriner	36. ORNL Patent Section
17. L. R. Shugart	37. I&C Division Publications Office
18. J. O. Stiegler	

EXTERNAL DISTRIBUTION

- 38. Dr. Robert N. Farvolden, Professor, Department of Earth Sciences, University of Waterloo, Waterloo, Ontario N2L 3G1, Canada
- 39. Dr. Jerry F. Franklin, Bloedel Professor of Ecosystem Analysis, College of Forest Resources, University of Washington, Anderson Hall AR-10, Seattle, WA 98195
- 40. Dr. Robert C. Harriss, Institute for the Study of Earth, Oceans, and Space, Science & Engineering Research Building, University of New Hampshire, Durham, NH 03824
- 41. Dr. Ronald H. Olsen, University of Michigan, Med Science II #5606, 1301 East Catherine Street, Ann Arbor, MI 48109-0620
- 42. Assistant Manager for Energy Research and Development, DOE-OR, P.O. Box 2001, Oak Ridge, TN 37831-8600
- 43-44. Office of Scientific and Technical Information, U.S. Department of Energy, P.O. Box 62, Oak Ridge, TN 37831

END

DATE
FILMED
9 / 15 / 93