Cameron

5105-155
Solar Thermal Power Systems Project
Parabolic Dish Systems Development

DOE/JPL-1060-90
Distribution Category UC-62

Overview of Software Development at the Parabolic Dish Test Site

C.K. Miyazono

July 15, 1985

Prepared for
U.S. Department of Energy
Through an Agreement with
National Aeronautics and Space Administration
by
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

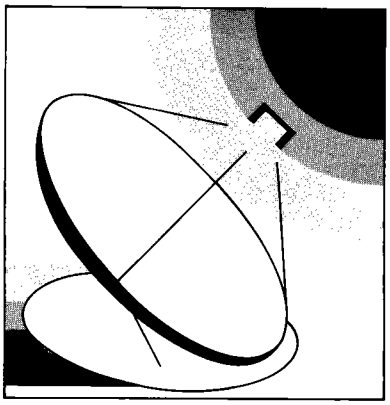
JPL Publication 85-56

5105-155
Solar Thermal Power Systems Project
Parabolic Dish Systems Development

DOE/JPL-1060-90
Distribution Category UC-62

Overview of Software Development at the Parabolic Dish Test Site

C.K. Miyazono



July 15, 1985

Prepared for
U.S. Department of Energy
Through an Agreement with
National Aeronautics and Space Administration
by
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

JPL Publication 85-56

Prepared by the Jet Propulsion Laboratory, California Institute of Technology, for the U.S. Department of Energy through an agreement with the National Aeronautics and Space Administration.

The JPL Solar Thermal Power Systems Project is sponsored by the U.S. Department of Energy and is part of the Solar Thermal Program to develop low-cost solar thermal and electric power plants.

ABSTRACT

The development history of the data acquisition and data analysis software is discussed in this report. The software development occurred between 1978 and 1984 in support of solar energy module testing at the Jet Propulsion Laboratory's Parabolic Dish Test Site, located within Edwards Test Station. The development went through incremental stages, starting with a simple single-user BASIC set of programs, and progressing to the relative complex multi-user FORTRAN system that was used until the termination of the project. Additional software in support of testing is discussed including software in support of the meteorological subsystem and the Test Bed Concentrator Control Console interface. Conclusions and recommendations for further development are discussed.

ACKNOWLEDGMENT

The work described herein was conducted by the Jet Propulsion Laboratory, California Institute of Technology, for the U.S. Department of Energy through an agreement with the National Aeronautics and Space Administration (NASA Task RE-152, Amendment 327; DOE/ALO/NASA Interagency Agreement No. DE-AM04-80AL13137).

CONTENTS

SECTION I

INTRODUCTION

This report covers the software development task that took place in support of the U.S. Department of Energy's (DOE's) Thermal Power Systems (TPS) task at the Jet Propulsion Laboratory (JPL) between 1978 and 1984.

The software development task was undertaken to gather data from solar tests in a timely manner and provide this information to cognizant engineers in a useful form. The actual testing of the solar equipment took place at JPL's Edwards Test Station (ETS) located at Edwards Air Force Base, Lancaster, California, approximately 150 km northeast of Pasadena. The software was developed at JPL's Foothill Facility in East Pasadena.

A. OVERALL DESIGN PHILOSOPHY

The overall design philosophy was the accurate and timely gathering of data from solar tests at ETS. This governed all decisions regarding software, hardware, and programming commitments. Software was also designed to provide the resulting information in a form that was easily understandable, both during the test and after the completion of the test.

Additional design requirements included ease of operation by the testing staff at ETS, and the eventual need for a multi-testing capability.

The ease of operation requirement developed as a consequence of the location of the test site. The testing staff at ETS did not have programming expertise and, therefore, could not handle complex programming tasks. Nor was there a sufficient workforce to dedicate an individual to operating the data acquisition system. This task had to be shared by all the testing staff. This meant that the testing procedure could not be too complicated nor time consuming. Changes in test parameters at the last moment were not uncommon and the software had to reflect the ability to change with the testing hardware on very short notice. These factors contributed to the need for simple and fast software to handle the data acquisition.

The testing at ETS was initially confined to one module at a time. However, hardware plans were made to have several test bed concentrators (TBCs) available for use simultaneously. This required that data acquisition equipment sufficient to handle all the tests be in place, and that software to support all these needs be available when the time came. This design requirement was to have a major impact on the selection of an operating system in the future.

All of these factors contributed to the overall design philosophy and the eventual software that was provided for test support at ETS.

B. DATA LOGGING HARDWARE

The data logger selected for the conversion of data from sensors to a digital form was the Acurex Autodata-Nine. This data logger has a modular design with input cards that permit the scanning of 10 sensors or channels of data. This modularity extended to remote scanner boxes that could be placed in a remote location, and be activated and scanned by a data logger on command. The remote scanners could also operate from a considerably remote location. The cabling required to connect the remote scanner to the data logger was far less than the cabling required to connect all the sensors to the data logger located hundreds of meters away.

The data loggers had a scan rate of 24 channels per second in standard resolution mode, and 10 channels per second in high resolution mode. The data loggers used a sample-and-hold type of scan, rather than an average for some period of time. These data loggers, first purchased in 1978, were most noted for the low signal-to-noise ratio. They were able to scan 1000 channels with the aid of the remote scanners. The scan rate, channels scanned, and sensor type were selectable from the front panel. A backup battery provided power for the clock and calendar so that the device always had the correct time and day. An impact paper tape printer was available, but was not used during testing due to the slowness of the printing, which adversely affected scanning during testing.

The selection of the data logger was based primarily on criteria determined by the test engineers within the group. The low signal-to-noise ratio was the most persuasive argument for their use. The ability of the data loggers to handle a wide range of sensors was also important in their selection. These data loggers could handle voltages up to 10 V with the appropriate input card. The thermocouple input cards included electronic temperature compensation, eliminating the need for a standard temperature lead. All the common forms for thermocouple leads were supported by the data logger. The necessary digital interface to the minicomputer was of secondary importance. This was a problem during later multi-testing implementation.

The data loggers became the standard logging device at the test site. Eventually, close to half a dozen were purchased and used for testing and data acquisition at ETS by the time that testing ended in 1984.

C. PRACTICAL RESTRICTIONS

Some practical restrictions had to be taken into account in developing the software. The most obvious were the restrictions placed on the software by the available hardware. These hardware restrictions were divided into two categories: the data loggers and the minicomputer used for data gathering.

As mentioned previously, the data loggers were selected for their low signal-to-noise ratio rather than their serial input/output (I/O) capabilities. The serial I/O was an added feature that was restricted to a relatively low speed. The I/O was also sensitive to noise and required exact control sequences to operate. Support for starting and stopping the data flow once

the scan command was sent was not provided by the data logger. This lack of XON/XOFF support restricted the minicomputer's availability to perform other tasks. The minicomputer's ability to accept the data when a scan was started was an essential criterion.

The minicomputer used was originally made available to the Test and Evaluation (T&E) Group from a previous NASA project. This was the start of the use of Digital Equipment Corporation (DEC) PDP-11 series minicomputers for the data gathering task. The expertise of the members of the programming staff was in the use of DEC equipment and software. This would restrict the T&E Group to DEC equipment and monetary considerations would restrict the type of DEC equipment available.

Another practical restriction was the location of the test site. The programmers developing the software were located 150 km from the actual test site. During periods of software installation or software problems at ETS, a programmer had to be available on site. This took up valuable development time by a programmer. Every attempt had to be made to keep the software as "friendly" as possible for the staff at ETS, thereby eliminating the need for a permanent programmer on site.

These were some of the general practical restrictions that had to be included in the software development. Additional restrictions due to new hardware and software additions are discussed in the appropriate sections.

SECTION II

RT-11 BASIC

A. DESIGN PHILOSOPHY

The overall design philosophy for this first stage of development was to provide testing capability as soon as possible and support the testing effort at ETS with the existing equipment. A second, but equally important, consideration was the need to keep the software as easy to use as possible for the testing staff at ETS.

The first requirement was that the testing capability be available at the first possible opportunity. However, the system was initially quite cumbersome and provided a great deal of input on user interfacing for future software efforts.

B. HARDWARE

The initial hardware available for the beginning of the project included equipment that was transferred from a previous project. This was the DEC PDP-11/10 minicomputer with two removable 2.5 MByte disk drives, and a Versatec printer/plotter. This equipment was originally purchased with NASA funds for a different project. The Solar Thermal T&E Group also purchased new equipment to supplement the transferred equipment. This included a nine-track magnetic tape drive and controller, and an Acurex Autodata-Nine data logger.

C. SOFTWARE

The operating system used was the standard single-user software supplied by DEC called RT-11. This operating software was designed for a single user to interface with the equipment. As such, all the system resources were available to the one user.

The initial data acquisition software was written in BASIC. This version was not very sophisticated and was more of a temporary measure until a FORTRAN version of the software was available. The BASIC program required that the program be started from the console terminal, and that the connecting cable from the terminal to the minicomputer be disconnected from the terminal and re-routed to the data logger. At that point, the data logger was set to the scanning mode and data acquisition began.

The data from the data logger consisted of American Standard Committee for the Interchange of Information (ASCII) characters, each character equalling 7 bits. The data were set out in a scan, with all data channels scanned in sequence preceded by a time mark. A selectable end-of-scan character was sent at the end. This character was an "!", ASCII character 41 octal. The structure of the characters within a data scan was fixed by the data logger. This served as a check of the integrity of the transmission of the data from the data logger to the minicomputer. A typical scan is shown in

Table 2-1. The presence of the colons separating the minutes, hours, and seconds were checked to perform a rudimentary check of the integrity of the data. The example shown in Table 2-1 is for the Meteorological Subsystem. The output for test data includes a three-digit channel identifier rather than the one-digit identifier (shown before the + or - sign) and includes six characters for units which have been deleted from this example. For more details, see the Acurex Auto-Data Nine Reference Manual.

The data logger was manually set to scan through all of its programmed channels once per interval. The interval usually used was 30 s or 1 min. The data logger's slow scan speed and the minicomputer's method of data acquisition prevented shorter intervals.

The data logger converted several types of analog data into useful digital data. Thermocouple voltages were directly converted to temperatures by the data logger. The data loggers were equipped with temperature compensating circuitry to perform the conversion. All other measurements had to be converted to voltages for measurement. These were digitized in sequence during a scan. These instantaneous values were then transmitted to the serial port of the data logger. No averaging of values during the interval was possible with the data logger.

The data from each scan was loaded into a buffer within the operating system software and was then transferred to the magnetic tape. All data in this version was sent directly to mag tape. No data was stored on any other medium. The format of the data on the magnetic tape was in the standard form for BASIC files. The scan information was transferred to magnetic tape in standard 512 byte blocks. Since the scans from the data logger may not be exactly 512 characters per scan, each block of data may include less than one scan of data. The magnetic tape drive used the standard IBM nine-track format for writing.

At the conclusion of the test, the cable from the minicomputer to the data logger was removed and re-attached to the terminal. From the terminal, a command to terminate the acquisition program was entered. At this point, an end-of-tape mark was placed onto the magnetic tape. The software would then perform a data dump of all data recorded on magnetic tape, either in a formatted output with all measurements converted to appropriate engineering units, or as a direct data dump in actual voltages as measured from the data logger.

This software was only an initial attempt at data acquisition for a limited number of data channels and was used only for the first few tests. During this period, development of FORTRAN data acquisition and printout software was underway. This BASIC software provided information that was helpful in generating a FORTRAN software package that was better suited to the needs of the test facility.

D. PROBLEMS/RECOMMENDATIONS

The major problem with this system had to do with the user interface. The constant moving of cable connectors and shuffling back and forth between the data logger and the terminal were quite complicated and required timing

Table 2-1. Example of Output from a Data Logger

193 12:48:48	, 0+000.00,	1+099.98,	2+04.445,	3+04.430,	4+000.47,	5+092.24,	6+093.21,	7+090.76,	8+090.23,	9+091.67,
193 12:48:58	, 0-000.00,	1+099.98,	2+04.466,	3+04.451,	4+000.48,	5+092.68,	6+093.65,	7+091.20,	8+090.67,	9+092.11,
193 12:49:08	, 0-000.00,	1+099.98,	2+04.480,	3+04.465,	4+000.48,	5+092.98,	6+093.96,	7+091.49,	8+090.96,	9+092.41,
193 12:49:18	, 0-000.00,	1+099.98,	2+04.500,	3+04.485,	4+000.48,	5+093.38,	6+094.36,	7+091.89,	8+091.35,	9+092.81,
193 12:49:28	, 0+000.00,	1+099.98,	2+04.500,	3+04.485,	4+000.48,	5+093.38,	6+094.37,	7+091.89,	8+091.35,	9+092.81,
193 12:49:38	, 0+000.00,	1+099.98,	2+04.050,	3+04.036,	4+000.43,	5+084.05,	6+084.93,	7+082.71,	8+082.23,	9+083.54,
193 12:49:48	, 0+000.00,	1+099.98,	2+04.080,	3+04.067,	4+000.43,	5+084.68,	6+085.56,	7+083.31,	8+082.83,	9+084.16,
193 12:49:58	, 0-000.00,	1+099.98,	2+04.080,	3+04.066,	4+000.43,	5+084.67,	6+085.56,	7+083.31,	8+082.83,	9+084.17,
193 12:50:08	, 0-000.00,	1+099.98,	2+04.097,	3+04.083,	4+000.44,	5+085.03,	6+085.93,	7+083.67,	8+083.19,	9+084.51,
193 12:50:18	, 0-000.00,	1+099.98,	2+04.097,	3+04.083,	4+000.44,	5+085.03,	6+085.92,	7+083.67,	8+083.19,	9+084.51,
193 12:50:28	, 0+000.00,	1+099.98,	2+04.120,	3+04.106,	4+000.44,	5+085.50,	6+086.40,	7+084.13,	8+083.64,	9+084.97,
193 12:50:38	, 0-000.00,	1+099.98,	2+04.132,	3+04.118,	4+000.44,	5+085.75,	6+086.65,	7+084.38,	8+083.89,	9+085.23,
193 12:50:43	, 0-000.00,	1+099.98,	2+04.132,	3+04.118,	4+000.44,	5+085.75,	6+086.65,	7+084.38,	8+083.89,	9+085.23,
193 12:50:58	, 0+000.00,	1+099.98,	2+04.159,	3+04.145,	4+000.44,	5+086.31,	6+087.22,	7+084.93,	8+084.44,	9+085.78,
193 12:51:08	, 0-000.00,	1+099.98,	2+04.162,	3+04.147,	4+000.44,	5+086.36,	6+087.32,	7+085.03,	8+084.54,	9+085.89,
193 12:51:18	, 0+000.00,	1+099.98,	2+04.176,	3+04.162,	4+000.44,	5+086.67,	6+087.58,	7+085.28,	8+084.79,	9+086.14,
193 12:51:28	, 0-000.00,	1+099.98,	2+04.191,	3+04.177,	4+000.45,	5+086.98,	6+087.89,	7+085.59,	8+085.09,	9+086.45,
193 12:51:38	, 0+000.00,	1+099.98,	2+04.208,	3+04.194,	4+000.45,	5+087.34,	6+088.26,	7+085.94,	8+085.44,	9+086.81,
193 12:51:48	, 0-000.00,	1+099.98,	2+04.223,	3+04.209,	4+000.45,	5+087.65,	6+088.57,	7+086.25,	8+085.75,	9+087.11,
193 12:51:58	, 0+000.00,	1+099.98,	2+04.239,	3+04.229,	4+000.45,	5+088.11,	6+089.04,	7+086.70,	8+086.19,	9+087.57,
193 12:52:08	, 0-000.00,	1+099.98,	2+04.253,	3+04.239,	4+000.45,	5+088.26,	6+089.19,	7+086.85,	8+086.35,	9+087.73,

from the testing staff. As the testing staff was not accustomed to using computerized data acquisition equipment, this only complicated matters. A major recommendation from this software was to simplify the user interface.

The data loggers can scan up to 1000 channels of information per scan. The data logger has to be set up to identify the channel number, the type of measurement to expect, the level of resolution to measure, as well as standard information such as scan frequency. This initial version of the software did not permit the automatic setting of the data logger. This required that the data logger be set by hand prior to the test. This can be a tedious task that could be automated through the data logger's serial I/O port.

The conversion values for each channel of data were hard-coded into the data processing software. To affect a change, the program had to be edited. This required that a programmer be available to make even the simplest changes of conversion constants. A better alternative would be to support the conversion constants in a table of values that could easily be edited.

These major problems were examined and corresponding solutions were incorporated into the next version of the software.

SECTION III

RT-11 FORTRAN

A. DESIGN PHILOSOPHY

This was the next increment of development of the software for the testing at the Parabolic Dish Test Site (PDTS). The overall design philosophy remained the same, namely, the accurate and timely gathering of data from tests.

Based on the experience from the BASIC programming, additional interfacing for ease of operator use was planned for the system. This experience not only provided information about the human-data-acquisition equipment interface, but also about the data logger-data-acquisition equipment interface.

B. HARDWARE AVAILABLE

The hardware was upgraded at this point. Rather than a DEC PDP-11/10 minicomputer, a larger, faster minicomputer (PDP-1134A) was substituted. This minicomputer provided more flexibility in accessories than could be included into the system.

The same peripherals were available for this new minicomputer. One major factor in the decision to use the PDP-1134A was the compatibility of peripherals. The two removable 2.5 MByte disk drives, the nine track magnetic tape drive, and the printer/plotter were all transferred to the new system. Additional peripherals that were new to this system included additional serial I/O ports for connection of the data loggers.

C. SOFTWARE

The software developed tried to take advantage of the BASIC programming experience. The inconvenience of manually changing the serial I/O port connector from the data logger to the console terminal was replaced by a much smoother system. As much automatic logging of information as possible was included.

The software consisted of six separate programs, each designed to be used in sequence. The first of the sequence was the program to set the parameters called SETPAR. This program was used to alter and/or update a table containing information about the channels to be scanned in the day's test. The information included the type of transducer that would be used, the conversion coefficient, if any, the resolution that the data acquisition software should expect, and a suitable title and appropriate units for each measurement. This software prepared the table that would be used in printing out the final results. It did not set up the data logger.

The next program assigned the test number and prepared the magnetic tape by placing a header file containing information about the test, such as test

number and date of test. The testing staff was encouraged to manually check the data logger to ensure that the time was synchronized with that in the data acquisition minicomputer, that the channels of interest would be scanned, that all alarm levels were properly set, that the scan rate was correct, and that the data logger was set to send data to its serial port.

The actual logging was performed by a program called LOG. This program acted acted as a buffer, collecting data from the data logger input port, performing some minor string checking, and outputting the information to the magnetic tape drive. The software also permitted the suspension and resumption of logging within a test and the entry of a message onto the tape. The checking of the string included the checking for colons in the time string and, in later versions, the checking of the data stream for warning messages. These warning messages were displayed on the screen of the console terminal for the testing staff to interpret and act upon.

At the conclusion of the data gathering, a program to place the parameter table onto the magnetic tape was run. This was the POST program. This provided an archive copy of the channels measured and some minor information about each channel. Later examination of the data showed that more information about the testing would have helped.

Two versions were available for printing out the data. The quickest was a program called QUICKY. This program started at the beginning of the data run on the magnetic tape and printed out each scan, line by line, without conversion of data or any additional information other than the time, channel number, and the raw data value. This was the printout most often requested at the test site. The other version provided a cleaner format with conversion to the appropriate engineering units, but could only display 10 channels of data at a time. Simple calculations were possible for this print program called PRINT. The simple calculations were included in the source code and had to be incorporated by a programmer. This precluded a member of the testing staff from altering or adding calculations during the testing. This further required the cognizant engineer to preplan his calculation needs. As a typical test had over 70 channels of data, the cognizant engineer would request the quick, raw printout and a formatted printout of only 10 of the most significant channels.

Note that the data logger was handled as a separate device. The software was set to permit rapid change of the channels to be scanned, the conversion factors to use, and the other information included in the output; but the testing staff had to set up the data logger manually, initiate testing manually, and terminate testing manually. In later versions of the software, this was replaced with automatic setup and use of the data logger by the minicomputer. Also note that graphs of data were not available at this time. The software supported several I/O channels so that the cabling from the console terminal did not have to be changed to a data logger during testing, as was the case with the BASIC software.

The files on magnetic tape consisted of three files. The first file contained starting information about the test such as test number and date. This file had the filename xxxxxx.PRE where the xxxxxx was the six-letter code for the test. This file was in a DEC RT-11 FORTRAN formatted sequential file. The main data file on the magnetic tape was in the form of a DEC RT-11

FORTRAN unformatted sequential file. Data from the log program was written onto the magnetic tape in this manner. The last file in the series was the file containing the parameters used to prepare the formatted output. It included all the information about the channel numbers used, the units, the titles, the resolution, and the type of sensor. This file was in the form of a DEC RT-11 FORTRAN unformatted sequential file. This information was important in reconstructing the test information at some future time. All three files were stored on magnetic tape at the test site.

Each tape could contain more than one test. However, one test could not span more than one tape. Running out of tape during a test would have resulted in a fatal error in the FORTRAN programming. Therefore, the testing staff was instructed to use a new tape whenever a long test was expected.

D. PROBLEMS/RECOMMENDATIONS

This version of the software package was a major improvement over the BASIC version. The amount of test equipment manipulation during testing was decreased, although not totally eliminated. The fact that the testing staff still had to set the data loggers by hand was a problem. The availability of several I/O ports accounted for the moderate amount of improvement.

This version also permitted easier changing of the testing parameter information, right up to the moment of the test. This ensured that the data was timely and correct. However, calculations still had to be prepared days in advance and last-minute changes could not be incorporated.

The data in tabular form, even with the appropriate units and formatting, still represented a staggering amount of data. Typical runs of several hours with scans of 75 channels every minute could produce a quarter of a million characters in the printout. Graphic representation of all these data would have assisted the cognizant engineer in the interpretation of the data. This major modification was added to the next software package.

Information about the test other than the raw data was limited. The parameters table was included; but this was minimal information at best. The software did support comments; but during the testing, time to include comments was usually not available. Additional information had to be stored with the test data for archival purposes.

Neither the operating system nor the application software permitted the conducting of two tests simultaneously. This was a major drawback. During the period of this testing, the two TBCs were being readied for simultaneous use. Software was developed to handle this situation.

SECTION IV

RSX-11M FORTRAN

A. DESIGN PHILOSOPHY

The basic design philosophy was simply an extension of the previous systems. The primary design goal was to ensure the accurate and timely gathering of data. Now, however, the requirements of testing dictated that the software be able to record data from two separate tests simultaneously.

The "user friendliness" of the software still needed to be improved. The design philosophy of providing the testing staff with software that would handle the majority of the data acquisition task had not yet materialized. This situation was extended to the cognizant engineer who had to interpret the data gathered.

B. HARDWARE AVAILABLE

The hardware available at this point changed significantly. To support the greater amount of data expected and the anticipated speed that would be required to handle this additional data, a large disk system was added to the minicomputer. This was a Control Data Corporation 7766 hard disk drive and Systems Industries disk controller. This disk had a capacity of 300 MByte of data. All software, with the exception of the operating system, was placed onto this disk. Additional serial I/O ports and video terminals were also added to the system in support of the expected multiple test situation.

The rest of the hardware remained the same. The minicomputer at the center of the data acquisition system was the DEC PDP-1134A with two 2.5 MByte removable hard disk drives, a nine track magnetic tape drive, an electrostatic printer/plotter, and assorted accessory equipment.

Additional data loggers were purchased from Acurex. In fact, the last data loggers received were from the final manufacturing run by Acurex. In support of the testing at a site far from the data acquisition equipment, the RS-232 signal would not ordinarily reach that far without the possibility of interference. To ensure that the data were received without problems, a duplex fiber optic cable was installed from the test site to the data acquisition site. This fiber optic link proved to be trouble free during the entire testing period.

C. SOFTWARE

1. Initial Attempt

The support of multiple tests simultaneously presented many problems. The first was the operating system that would support multiple users conducting multiple tests. Because of the hardware restraints, the multi-user real-time operating system available from DEC (called RSX-11M) was used. Along with this operating system, the applications software was written

in FORTRAN and assembly language. The version of RSX-11M that was used was version 3.1. Since the start of the development, RSX-11M had progressed to version 4.0. The last version represented an improvement in user interfacing, but at the price of speed and compactness. Also, each new version brought on new changes in the printer/plotter software and in the hard disk handler. These represented additional charges for software that were not expected. Also, a special driver was written to support the serial I/O in version 3.1, and it was not clear that it would function properly in later versions. For this reason, once the acquisition software was written in version 3.1, no further updating to support later versions was made.

The software effort at this point was quite large. Two programmers were assigned the task of data acquisition and recording, using a common memory approach. In this method, the data as received from the various data loggers was placed in a common area of memory that could be accessed by printout and storage programs running in turn. The common area of memory held the most recent scan of data; and it required coordination to receive, print out, and store the data in proper sequence in the time allotted. This task proved to be quite complicated.

Two events changed this approach. First, the individual in charge of software development, as well as the individual most intimately involved in the common memory approach, both left the project. The second event was financial. A budgetary problem resulted in releasing the two programmers working on the common memory approach and the abandonment of this approach. In its place, the need to get a multiple test system in operation quickly resulted in a much more conservative system. This system was based on hard disk file access for the data.

This hard disk file access was made possible because of the availability of the 300 MByte hard disk system. The data acquisition software stored the data in its raw form on a disk prior to any manipulations. This ensured that the data received was correct and that, at some future time, if a change in calibration were found, or some constant altered, the raw data received from the data logger would be available. The acquisition software was also designed to display warning messages on the console terminal should a raw data value exceed a predetermined limit. The software was designed to display on the console terminal the values obtained for several channels immediately after the scan to provide the cognizant engineer with a real-time estimate of the test situation.

This part of the software package was performed during the winter of 1981. The software needed to conduct the entire test consisted of several programs, each used in succession. This was necessary due to the limitations of the DEC FORTRAN software. Programs larger than 32 kilowords, where each word consists of two 8-bit bytes, could not be executed. This resulted in the segmentation of the overall program into subsets which performed specific tasks. Together, the software provided all the data acquisition and data retrieval necessary.

2. Operating System

At this point, a few words about the constraints placed upon the data acquisition software by the operating system is included. Under the original plan, several separate programs would run simultaneously within the operating system environment. One major problem that was encountered early, while working with the RSX-11M operating system, was the round-robin polling of tasks. Under RSX-11M, the operating system assigns a priority number to each task ranging from 1 to 250. Tasks with a high priority receive the resources of the minicomputer. Tasks with equal priority each get a turn at using the resources of the minicomputer for a set period of time. After the task's allocated period of time, the operating system goes to the next task with the same priority and gives the system resources to it. In this way, each task gets its turn using the minicomputer, and tasks that were entered later into the system get an opportunity to complete on an equal footing with tasks entered earlier into the system. However, this round-robin scheduling of resource allocation did not work well with the data loggers. The data loggers, as mentioned earlier, did not support XON-XOFF protocol and, therefore, could not be stopped once the scanning was begun. The problem developed that, during the scanning, the round-robin scheduler stopped the acquisition task and went on to another task, such as the data storage task or the printout task. This meant that the data for the latter portion of the task was lost. The only solution available to us at the time was to ensure that the minicomputer and its resources were always in the data acquisition task during the test procedure. This was done by setting the data acquisition task at a priority of 249 at startup. This effectively made the multi-user system a single user system.

One possible solution to this problem might have been the use of a "keyboard buffer" to store the information from the data logger in a serial input buffer until the operating system returned to the acquisition task after the round-robin attention to the other tasks. The RSX-11M operating system did not support an input buffer. The keyboard was active only when the operating system was polling that task. This meant that the common practice of "typing ahead" was not allowed. Any input was lost. This proved to be a major problem that was encountered using RSX-11M.

Another problem encountered with the operating system early in development was the difficulty with shuffling tasks within memory. The operating system monitored the free memory available for new tasks. As each user requested a task, each was loaded into memory as low as possible. As the tasks were completed, each task was removed from memory. A problem can occur if a small task is loaded low in memory with other larger tasks installed above in higher memory. When the small task is completed, it is removed from memory. If another task wants to enter into the system, it must be equal to or be smaller than the just-removed task. The operating system shuffler will consolidate any existing space in memory into one large contiguous region of memory and make it available for use to a new task. This did not always occur when tasks of different priorities were involved. Tasks would not be shuffled to make room and other requested tasks would be queued. This made it imperative that care be taken when running tasks of different priorities.

3. Disk File Access

The data acquisition system that was eventually used for the bulk of the testing at ETS was based on the RSX-11M operating system and the disk file access approach. The task was broken into many small tasks that were set to "daisy chain" together with each completing task calling the next task. This daisy chaining was accomplished through the use of indirect command files as well as the chaining of tasks. The tasks are listed in Table 4-1. Note that these are the major tasks. Each task shown has several subroutines that are not listed.

The daisy chaining of the tasks was accomplished by either calls from within an indirect command file or through the use of the RECEIV/SEND directives. In most cases, both methods were used to initiate the appropriate tasks. A sample of an indirect command file is included in Appendix F.

Table 4-1. Major Tasks for Data Analysis

Software Task	Description
DBGEN	Database generator
DECODE	Decodes the calculation subroutine and creates array for calculations
NWDBGN	Further database generation processor
EXECUT	Executive program from which the operator starts and stops logging
LOGGER	Main task that performs the actual logging
CNVRT	Compacts the data from one character per word to one character per byte
NEWVRT	Converts data from raw form to engineering units
NWPRNT	Prints out engineering unit data in tabular form
PLTSET	Extracts data from engineering unit file for plotting
PLOT	Generates plotting instructions for plotter using data extracted above

4. Testing Sequence

The sequence of testing began with the testing staff at ETS checking the parameters file. The parameters file contained the list of channels that were to be scanned including information concerning the type of sensor used, the resolution expected, the name and units to associate with the channel, limits (if any) to monitor for the various channels, the name and channel assignment of calculations to be included, the list of channels to display on the terminal in near real time, and a list of channels to plot at the conclusion of the test. The checking was done using the system editor. The test number was sequenced manually, and a check of time interval to be used was also made.

A parameters file existed for each module tested at ETS. As sensors were added or deleted, this file was modified. These parameters were used by subsequent programs to determine such functions as the channels to scan, the channels to display, and the channels to plot. To place this file into a readable form, the tasks DBGEN and NWDBGN were used to create and/or update the machine-readable file, SETPAR.00x, where x was either 1, 2, or 3. This file was the one used by all subsequent programs.

To include calculations in the data printout portion of the analysis sequence, the subroutine containing calculations had to be correlated with the channels used in the parameters file. To accomplish this, the program DECODE was used to correlate this information. This information was used to provide near real time displays of calculated values.

The testing staff interfaced with the data acquisition program EXECUT to start or stop a test. EXECUT asked a series of questions about the test to be performed to ensure that information important to the test would be archived with the data. The information included the test identification number, the date, the test bed being used, the mirror configuration (if the mirrors had been washed today), the cloud cover, and any notes of interest. This information was printed out at the end of the test. This information subroutine was added later and, therefore, was not available for all tests.

The EXECUT program performs a call to the operating system that initiates logging by calling the program LOGGER. This program sends a series of characters to the data logger serial port. The data logger interprets these characters and starts a scan. The program then waits for the characters to return from the data logger. As each character returns, the character is placed into a buffer maintained by the program. The characters are checked for warning and, if one is found, is displayed on the predetermined terminal. At the conclusion of the scan, the data in the buffer is transferred to a file on a disk called LOGx.ACQ, with the x either 1, 2, or 3. The file is opened, the information is transferred, and then the file is closed. This is to ensure that if there is a system crash, the data will not be lost. Under ordinary circumstances, an open file will be lost should a system crash occur.

After the data is safely on a disk, the logging task displays selected channels of converted data on a preassigned terminal. The selection of channels to display is set in the parameters table and the assignment of terminals is done at startup time by assignment of logical units. This is a feature of the RSX-11M operating system for installed tasks. The logging task

then checks its timer to see when the next scan is to be executed. In this way, the logging task determines when the logging is to be done, starts the scan, and processes the data as they are returned by the data logger. The logging task will skip the display of data should its timer indicate that the next scan is to start. In this way, the display of data is of secondary importance. The collection of data is paramount.

The EXECUT program can instruct the logging task to collect data from one, two, or all three available ports. The timing of the scans can be a problem, however. If each data logger scans approximately 75 channels of data, over 3 s are required simply to transfer the data from the data logger to the minicomputer. This does not include time to store, process, and display the data. The time given to perform the additional functions is approximately 15 s. The best scan rates that were used at ETS were 20-s scans for one data logger, and usually 30 s for two data loggers, and 1 min between scans for three data loggers. The logging task and the data loggers simply were not fast enough to handle data at higher rates.

At the conclusion of the test, an indirect command file (called POST) was executed that would handle the termination of the test and the post processing of the data. Whenever possible, indirect command files were used to ease the operator input at the test site. The indirect command files permitted a predetermined set of operating system-level commands to be entered. Each was then executed in turn. Some simple decision making logic was available in the indirect command processor and this was used to handle some individual differences between tests.

The indirect command file, POST, started by running the executive program EXECUT. From this task, the logging could be halted. At this point, additional comments could be entered for inclusion with the other data onto magnetic tape. The operator was asked to include the number of the logger used and the test ID run number. The test ID run numbers for the RSX-11M collected data was based upon a two-character module identification. The next four digits represented test numbers, usually starting with test 0100. The file extensions (the three characters after the decimal point) were used to indicate different types of files. Table 4-2 gives an example of the type of two-character module identifications that were used. A listing of all the tests performed is given in a JPL report.[1]

At this point, the processing of the data was started. The raw data was converted from one character per word to one character per byte, and stored in a file with the six-character file name followed by the three-character extension; .ACQ, by the task CNVRT. The .ACQ file was unique to this particular data run and contained all the raw data from the data logger. Upon completion, the CNVRT task called the next task, NEWVRT. This task took the raw data from the .ACQ file and, using the information from the parameters file, converted all the information to engineering units. This information was stored in a file given the name: filename.EUS. Upon completion of this task, the print formatting task was called by the NEWVRT

---

[1]Selcuk, M. K., Parabolic Dish Test Site: History and Operating Experience, JPL Publication 85-18, February 15, 1985.

Table 4-2. Module Identifications

Module ID	Module Tested
OG	Omnium-G Tests
BR	Brayton Air Receiver
CR	Carter Steam Engine
FM	Flux Mapper
CW	Cold-Water Cavity Calorimeter
ST	Stirling Cycle Engine
PY	Pyrheliometer Standards Comparison
MA	Materials Test
VA	Vanguard Tests

task. The print formatting task, called NWPRNT, formatted the output into seven columns of data in addition to the time stamp for each channel. This printout file was stored as TRANSx.LST where the x indicated the logger used.

The NWPRNT task is called the plot formatting task, PLTSET. This task extracted the necessary data from the .EUS file to create the graphs that were requested in the parameters file. The output from this task was stored in a temporary file denoted by the extension, .PLT. The formatting of the data into a plot format for use by the printer/plotter was performed by a task called PLOT. This task took the data from the PLT file and, using the proprietary software from the plotter vendor, performed the actual calls to the plotter subroutines. The outputs from this task were two binary files called VECTR1.BIN and PARM.BIN. At this point, many files were renamed to match the test identification filename to keep them unique to the just-completed test. A list of extensions used in the data analysis is given in Table 4-3.

All the files (with the exception of the .LST, .PLT, and .NOT files) were stored onto magnetic tape for archival storage. The files were also kept on the 300 MByte disk drive for backup. This twin backup system was used to ensure that the data would be available should one backup source be damaged. The transfer of data to the magnetic tape was done using the standard DEC peripheral interchange program (PIP), rather than the backup and recovery (BRU) utility. The BRU utility was not available when the RSX-11M data acquisition program was begun at ETS; and the initial versions proved to have a system bug. The PIP commands were well known to the testing staff and easy to use.

The storage of the data on the 300 MByte data disks required that the disks be changed periodically. The filled disks were stored at the Foothill Facility and a new disk was installed to store data. The data acquisition software was transferred from magnetic tape and occupied approximately 1 MByte of disk space.

Table 4-3. Filename Extensions Used for Data Analysis

Filename Extension	Contents
ACQ	Raw data from data logger
EUS	Data converted to engineering unit
LST	Engineering unit data formatted for printout
PLT	File containing data to plot
NOT	Note file containing notes from testing
CLC	Copy of calculations subroutine
SET	Copy of parameters table for this test
CMT	Comments file included on magnetic tape

The printing of the data and graphs was performed by the initiation of another indirect command file called PRINTOUT. This indirect command file could perform the printing and plotting of data for four different tests, each with different requirements for number of copies of data and number of sets of plots. This routine was normally initiated at the end of the day with a time delay and allowed to run overnight. The printout routine would begin the printout sometime overnight; and the completed printouts and plots would be available the next morning. Occasionally, problems with the printer such as a paper jam overnight would hinder the task. The biggest problem that developed was the printing out of large test runs overnight. The printer/plotter used an electrostatic process to write and, therefore, required specially processed paper. This paper was supplied in fanfold sizes of only 1000 sheets. Some of the later tests required from 300 to 400 sheets to print out. It was normal to require three sets of printouts and, therefore, only two sets could be printed out from one box of paper. This was especially annoying because printing was performed overnight and no one was available to add paper to the printer.

Once a week the testing staff cleaned up the disk by eliminating unnecessary files. Another indirect command file, CLEANUP, was used to perform this task. All .LST, .PLT, and .NOT files were eliminated. Executable tasks were purged so that only the last two versions were retained on the disk. This usually saved several hundred blocks of space on the disk.

5. File Structure

The structure of the data files stored on the disk was based upon the total number of channels that could possibly be recorded during a test. Initial versions used a maximum of 100 channels, but later this was increased

to 150 channels. All storage of data was by the sequence in which it was recorded, not by channel numbers. Because of this, information stored in the parameters file was very important in the correlating of stored data to the actual channel numbers used on the data logger.

The parameters file was an ASCII file that could be edited using the standard DEC editor. The structure of the file is shown in Table 4-4.

The first line is simply used to assist the testing staff to find the correct column for data entry. The second line contains information about the test to be performed. The first two characters represent the module being tested, the next four represent the test number, and the next three digits represent the total number of minutes during the test. This number is initially zero and, during the shutdown of the testing procedure, the correct number of minutes is placed into this table. The two digits starting at column 16 represent the time between scans in seconds. The last two sets of numbers were provided, but never used. The remaining lines, up to the words PRINTOUT ORDER, contained the actual information about each channel in the data logger. The first three-digit number was the channel number, as set in the data logger. The six-character code name starting at column 7 was a code input name for the calculation routines. This was normally set to any six-letter code word, such as HEADER, for channels not used in calculations. The information in columns 16 to 21 contained the conversion factor that the raw data from that channel was to be multiplied by in order to convert to engineering units. The eight-character engineering units were listed in columns 25 to 32. Columns 37 to 52 listed the 16-character name of the channel being recorded. This contained information about the test channel such as "INPUT STEAM PRES" or "THERM.ENRGY OUT."

Column 54 contained a single number that was to have been used to indicate the number of significant figures allowed in the printout. However, this was never implemented. The information in columns 56 and 57 were for use with the data logger. The logger can be set to measure thermocouples or voltages in the 100-millivolt range, 1-volt range, or 10-volt range with the simple selection of a switch. The first number corresponded to the selection switches on the data logger. The second character indicated the type of resolution that the data logger was to supply. The "H" indicated high resolution mode, while the "S" indicated standard resolution mode. The high resolution mode provided one additional decimal place of accuracy, but at the cost of speed. The scanning of a high-resolution channel was approximately 2.4 times slower than a standard resolution.

The last two sets of numbers, starting at columns 59 up to 63 for the lower and columns 65 to 70 for the higher, represented the lower and upper limits to be used by the data logger for tripping of alarms. If the two values were identically set to 9999., then no alarms were set. These last three sets of numbers were used to set up the data loggers.

All channels that would be scanned by the data logger were listed in order of appearance. The data loggers start from the low channels and work upward. At the end of the parameters list of channels, several channels are

Table 4-4. Parameter Table

```
TBC1 STIRLING ENGINE TEST, MOUNTED
12345678901234567890123456789012345678901234567890123456789012345678901234567890
ST 01 11  000  20  0  002
190   TTTTTT   1.0000   DEG. C      ENG.OIL.RES.TEMP 4 6S 9999. 9999.   45.67
191   TTTTTT   1.0000   DEG. C      INR MNFOLD QAD 6 4 6S 9999. 9999.   45.67
192   TTTTTT   1.0000   DEG. C      INR MNFOLD QAD 7 4 6S 9999. 9999.   45.67
193   TTTTTT   1.0000   DEG. C      BACK SURF.QUAD 1 4 6S 9999. 9999.   45.67
194   TTTTTT   1.0000   DEG. C      BACK SURF.QUAD 8 4 6S 9999. 9999.   45.67
197   TTTTTT   1.0000   DEG. C      ORIFICE LINE TMP 4 6S 9999. 9999.   45.67
198   TTTTTT   1.0000   DEG. C      PREHEATD AIR PRB 4 6S 9999. 9999.   45.67
199   TTTTTT   1.0000   DEG. C      PREHEATED AIR    4 6S 9999. 9999.   45.67
200   TTTTTT   10.000   GPM         ENG COOL FLOW    4 3S 2.000 9999.   45.67
201   TTTTTT   500.00   RPM         ENGINE SPEED     4 3S 9999. 4.000   45.67
202   TTTTTT   290.00   PSIG        HE PRESS SELECT  4 3S 9999. 8.000   45.67
203   TTTTTT   10.000   DEG.C       ENG  OIL TEMP    4 3S 9999. 7.500   45.67
204   VAL013   290.00   PSIG        HE PRESS MAX     4 3S 9999. 10.00   45.67
205   TTTTTT   290.00   PSIG        HE PRESS  -TANK  4 3S 9999. 10.00   45.67
206   VAL001   1.0000   VOLTS       ENG COOLANT TMP  4 3S 9999. 3.400   45.67
207   VAL002   1.0000   VOLTS       HIGH TUBE TEMP   4 3S 9999. 4.600   45.67
208   VAL003   1.0000   VOLTS       TUBE 1 QUAD1     4 3S 9999. 4.600   45.67
209   VAL004   1.0000   VOLTS       TUBE 2 QUAD6     4 3S 9999. 4.600   45.67
210   VAL005   1.0000   VOLTS       TUBE 3 QUAD7     4 3S 9999. 4.600   45.67
211   VAL006   1.0000   VOLTS       TUBE 4 QUAD8     4 3S 9999. 4.600   45.67
212   VAL007   1.0000   VOLTS       FLUE (CONE) TEMP 4 3S 9999. 5.000   45.67
213   VAL008   1.0000   VOLTS       PREHEAT EXHAUST  4 3S 9999. 5.000   45.67
214   VAL009   1.0000   VOLTS       ALT WINDING TEMP 4 3S 9999. 3.400   45.67
215   TTTTTT   50.000   VOLTS       VOLT PHASE A     4 3S 9999. 9999.   45.67
216   TTTTTT   50.000   VOLTS       VOLT PHASE B     4 3S 9999. 9999.   45.67
217   TTTTTT   50.000   VOLTS       VOLT PHASE C     4 3S 9999. 9999.   45.67
218   TTTTTT   5.0000   AMPS        CURRENT PHASE A  4 3S 9999. 9999.   45.67
219   TTTTTT   5.0000   AMPS        CURRENT PHASE B  4 3S 9999. 9999.   45.67
220   TTTTTT   5.0000   AMPS        CURRENT PHASE C  4 3S 9999. 9999.   45.67
221   VAL010   1.0000   VOLTS       ALT FREQUENCY    4 3S 9999. 9999.   45.67
222   VAL011   1.0000   VOLTS       POWER FACTOR     4 3S 9999. 9999.   45.67
223   TTTTTT   4.0000   KWATTS      ALTERN POWER     4 3S 9999. 9999.   45.67
224   VAL012   1.0000   VOLTS       OXYGEN ANALYZER  4 3S 9999. 9999.   45.67
225   TTTTTT   2.0000   GPM         FLT.PLT.COOL.FLO 4 3S 9999. 9999.   45.67
226   TTTTTT   1.6667   PSIG        ORIFCE LINE PRES 4 1S 9999. 9999.   45.67
227   TTTTTT   0.3333   PSID        ORIFCE DELTA PRS 4 1S 9999. 9999.   45.67
228   KENDAL   10.000   WATT/SQ.M   KENDALL PYRHELIO 4 1S 9999. 9999.   45.67
229   EPPLEY   117.10   WATT/SQ.M   EPPLEY PYRHILIO. 4 1S 9999. 9999.   45.67
230   TTTTTT   1.0000   (STATUS)    SLIDE PLATE STAT 4 2S 9999. 9999.   45.67
231   TTTTTT   0.0100   PSID        MANIFLD AIR PRES 4 1S 9999. 9999.   45.67
501                      DEG.C       ENG COOLANT TEMP
502                      DEG.C       HIGH TUBE TEMP
503                      DEG.C       TUBE 1 QUAD1
504                      DEG.C       TUBE 2 QUAD6
505                      DEG.C       TUBE 3 QUAD7
506                      DEG.C       TUBE 4 QUAD8
507                      DEG.C       FLUE(CONE)TEMP
508                      DEG.C       PREHEAT OUT TEMP
509                      DEG.C       ALT WINDING TEMP
510                      HERTZ       ALT FREQUENCY
511                        PF        POWER FACTOR
512                      PERCENT     OXYGEN ANALYZER
513                      MPA         SCALED CHAN 204
PRINTOUT ORDER
200,201,202,203,204,205,215,216,217,218,219,220,222,223,228,229,230,501,502,507,
510,511,
REAL TIME PLOTS
504            0.000    0000.00
END
```

listed with a number greater than 500. These were used to indicate calculations. These corresponded to the calculations that were included in the .CLC routines. Since the printout required titles, this was the place where that information was given. Note that information about conversion factors or data logger setups is not presented here. If there were no calculations, then the last scanning channel must be set to 599. This was used by the program to indicate the end of the data channel portion of the table.

The next section of the setup table is the printout order list. This list represented the channels to be displayed on the terminal during near real time display of data. The data displayed included units and the channel number. Conversion to engineering units was completed before displaying the information on the screen. The screen was set to display 23 channels of data. It was here that the testing staff could set which channels were to be displayed. The display was formatted with four rows each of six channels with the exception of the last, which only displayed five channels. This section had to be set prior to the start of the test and could not be changed during the test. Therefore, once set, the display was fixed until the next text. If the data logger fell behind in completing scans, such as when several data loggers were collecting data and there were many alarms that must be processed, then the display portion of the code was bypassed. The collection and storage of data and the display of warnings were considered more important than the display of near real time values.

The final section of the parameters table was the plotting of data marked "REAL TIME PLOTS." This section listed the channels to be plotted by the PLTSET and PLOT tasks in the data processing sequence. The first 16 characters of a single line indicated the channels to be plotted on one graph. As shown in the example, calculations were combined with data channels in any order. The next eight columns represented the starting value of the y-axis of the graph, and the last eight columns represented the ending value of the y-axis. If the values were both zero, then the task was requested to determine the minimum and maximum values. The plotting routines used the minimum listed and used either the higher limit or the next closest increment in plotting. A maximum of 20 different graphs was possible. There was no restriction on the use of channels, so that a channel could appear in all 20 graphs if desired.

The end of the parameters file was indicated by the word "END." This, then, is a summary of the parameters file. The file appears with each test on magnetic tape or on the 300 MByte data disks as xxxxxx.SET where xxxxxx indicates the test name.

The testing staff was instructed to verify this parameters file each day prior to testing. If a channel was added or deleted, the testing staff was instructed to make the appropriate changes. The most common change after the addition or deletion of a channel was the changing of a conversion factor. This information was critical for the proper analysis of the data at some later time. In many cases, this file was the only listing indicating the channels that were recorded during a test.

To create a machine-readable form of this parameters file, the tasks DBGEN and NWDBGN were used. The resulting file was called SETPAR.00x where x was the data logger number. This file contained information in a binary form. The first record of the file consisted of 20 words and contained the following information:

Word 1	Number of channels to acquire
Word 2 and 3	Test identification number
Word 4	Number of scans planned for test
Word 5	Time between scans
Word 6	Two-character test module identifier
Word 7	Number of channels for screen display
Word 8	Number of plots
Word 12	Number of calculations
Word 20	Actual number of scans taken during the test
All other words	Left blank for future use

The second record contained the channel numbers to scan and consisted of 150 words. The third record was 150 real variables that contained the conversion factors to translate the raw data to engineering units. The fourth record consisted of double-real variables containing the ASCII units description. This record was 150 double-real variables long. The fifth record was 300 double-real variables long and contained the 16-character channel descriptor. The sixth record of 150 words contained the significant digits information.

The seventh record, consisting of 150 double-real variables, contained the code input name used for the calculation routines. The eighth record, consisting of 150 words, contained the sequence numbers for the screen output for near real time displays. The ninth record, consisting of 150 integers, was followed by 150 pairs of real numbers. The integers contained the sequence number of channels to be plotted and the pair of real numbers were the lower and upper limits of the channel number. The tenth record was 100 words indicating the numbering sequence of the calculations. The eleventh record was 100 double-real variables containing the units of the calculations. The next record of 200 double-real variables contained the 16-character identifiers for the calculations. The thirteenth record of 200 double-real variables contained output information used for calculations. The fourteenth record of 150 words contained the input selection of each channel for the data logger. The fifteenth record contained the input sequence numbers for data needed for the calculations. The last record consisted of 150 pairs of real variables. These contained the lower and upper alarm limits for each logger channel. All of this information in the SETPAR file was accessible from each

of the subsequent tasks. This ensured that the setup information used by each task was identical. It also offered a relatively easy method of transmitting this information from task to task and still was easily accessible to the testing staff.

The procedure for data acquisition by the LOGGER program placed the data received from the data logger directly into a disk file called LOGx.ACQ, where the x indicated the logger used. This file consisted of records, each approximately 2700 words in length. This corresponded to 150 channels of data, each channel occupying 18 words of data space. The data from the data logger was stored in the record as it was received. The first 18 characters, which were the day and time, were stored in the first 18 words of data space. The next 18 characters, representing the first channel scanned, were stored in the next 18 words of data space, regardless of the channel number. In other words, the first channel collected, whether it was channel 001 or channel 367, was placed in the second 18 words of data. The next channel scanned was placed in the next 18 words of data space, and so on. As can be seen, only when 150 channels were scanned was it possible to fill the record. Under normal use, the records were not filled entirely. The reason for this type of structure was the type of access that was used in the logging programs. The direct access file structure was used to allow rapid opening, writing, and closing of the file. This was important to preserve data in case of a system crash. To use the direct access file structure, a fixed record length was required. Also, a future addition that was never implemented would have required that the record length be predetermined to some maximum value.

The LOGx.ACQ file held the data inefficiently. The DEC minicomputer memory is based on words, each of which consists of two bytes. Each byte can hold one character. This file held the character only in the lower byte of the words. This was done for speed considerations. However, for storage of data, this is inefficient. The first task called in the data analysis portion converted this information from word format to byte format. The structure of the resulting output was different. This output data file was given the name xxxxxx.ACQ where xxxxxx was the file name.

The day and time were stored in the 18 bytes of the ACQ file. The next 120 bytes were set aside for the inclusion of weather station data. This function was never implemented. More about the weather station is provided in a later section. Following the weather data section, the 150 channels of information were stored, each channel occupying 18 bytes. The same storage arrangement as for the LOGx.ACQ file was used for the storage of the channel data.

The processing of the raw data to engineering unit data generated another data file containing the engineering unit data and the results of the calculations. This file was denoted xxxxxx.EUS, where xxxxxx was the file name. This file consisted of two types of records. The first record contained a list of channels that were converted in this file. The second and subsequent records consisted of 260 real variable elements. The first two real elements were broken down into four words consisting of the time. The first word contained the day number of the test, measured from the start of the year as day number one. The next three words contained the time of the

scan in hours, minutes, and seconds. The next 10 real variables contained the converted weather data. The next 150 variables contained the converted channel data. The final 98 variables contained the calculations in sequential order. It should be noted that, under normal circumstances, not all the variables in the record were used. Typical data runs consisted of 100 data channels and 20 calculations.

For the plotting of the data, information had to be extracted from the EUS file. This information was included in a .PLT file. This file consisted of two records. The first record contained information about the plots. The number of channels to be plotted was placed in the first word. The second word of the record contained the time between scans. The next 640 words contained the 16-character channel identification information. The following 320 words contained the units to be matched with the channels. The next two sets of 80 words contained the sequence number of the channels to be plotted first, then all the limits of the channels as determined either from the parameters table or from a scan of the data. Word numbers 1123 to 1142 contained the number of graphs to be included in each plot, and the final nine words contained the beginning and ending times of the plots. The second and subsequent records contained the values in chronological order for each channel in ascending order. It should be noted that if more than 480 data points were taken for any given channel, the plotting routine took every other data point for plotting. This plotting information was then used to produce the actual plots which were generated by proprietary software from Versatec. The file structure of these .BIN files can be determined from Versatec.

All other files are sequential ASCII files that are direct representations of the data as viewed on the terminal screen. The data for these files are not altered in any way for storage onto a disk.

In summary, the file structures for the data files used in the RSX-11M FORTRAN version of the data acquisition system are based entirely on the order of acquisition, and are not dependent upon the channel number directly. All data files were set to the maximum allowable size prior to testing, even though the testing may not have required all the data record space to be used. The data files tended to be direct-access type of files that permitted quick access to a specific record, but required the predetermination of record size.

D. PROBLEMS/RECOMMENDATIONS

The RSX-11M FORTRAN data acquisition system was not without its problems. This system, in its end configuration, was quite different from the initial concept. This change was mainly due to changes in the testing requirements as well as to changes in our understanding of the RSX-11M operating system and its FORTRAN language.

The most important problem with the RSX-11M FORTRAN data acquisition software was the lack of a hardware interrupt routine that would permit the recording of the data from the data loggers on an as-needed basis, rather than the waiting-for-data approach that was used. The group lacked an expert in

RSX-11M systems applications and, consequently, was unable to implement a hardware interrupt routine. This would have alleviated the requirement for the minicomputer to be waiting for the data to arrive at the serial input port from the data logger. Rather, the minicomputer could have been processing the data, displaying the data in a real time display, or generating real time graphs, and returning to collect data on demand from the input port. This type of hardware interrupt would also have alleviated the input buffer problem if it had been implemented in all input ports such as terminal inputs. This would also have solved the round-robin scheduling problem that occurred. The hardware interrupt routine would have had the highest priority and, therefore, would always have been serviced first. Other tasks, not now time dependent, could have completed their functions unencumbered by timing restraints. Implementation of this type of routine is highly recommended if this system is to be used.

The problem with the shuffling of tasks within memory was a systems level problem that has since been corrected by later versions of the RSX-11M operating system. The reason that a newer version was not used at the test site was that several vendor-supplied software packages were written for the version of RSX-11M in use, and upgrades to newer versions were not purchased. Also, several assembly language device drivers had been written specifically for this version of the operating system and it was unclear how they would interface with the newer version of the operating system. Another problem with working with an older operating system is the lack of software support from DEC or from any other software vendors.

In summary, the inclusion of a hardware interrupt service routine would have simplified development of the data acquisition system. Time constraints as well as the lack of a systems applications background severely limited the type of data acquisition system that could be developed. Any new application of this software should include modifications to include a hardware interrupt service routine and a recompilation and relinking with the newer operating systems.

SECTION V

HARDWARE INTERFACING

A great deal of data acquisition and data analysis hardware was purchased during the course of the project. Much of this hardware included software from the vendor or software modifications to the data acquisition software that had to be taken into account during software development.

The most obvious area of interfacing was with the newly acquired DEC equipment. As the system grew, the operating system was changed as well as the programming language that was used. During the entire period of development, a software update agreement was in force with DEC. This provided the software development group with new updates of the operating system and the programming language as they became available. This agreement also provided software support by telephone. However, as the operating systems evolved to newer versions, this telephone support was no longer available.

The DEC hardware was normally installed by the software development group members. When a move of the entire system took place, such as the transfer of equipment from the Foothill Facility to ETS, DEC field service was contracted to provide support for the move. A service maintenance agreement was in force during the entire period of testing. This proved invaluable in repairing the system at ETS whenever there was a problem with the system. This also provided routine maintenance on a regular basis. This was most important at ETS due to the environmental conditions there.

Much of the peripherals were from outside third-party vendors. The hardware was checked as much as possible for compatibility with both the operating system software and the existing DEC hardware. In all cases, installation was purchased with the hardware. This was very cost effective in that any problems at installation time were quickly handled. Most of the peripherals did not require software changes nor did they come with software patches to be added to the system. The two major exceptions were the Control Data Corporation CDC-9766 disk drive and the Versatec Model 1100 printer/plotter. The disk drive required additional driver software that was provided by DEC and the third-party vendor. The printer/plotter came with a set of FORTRAN callable subroutines to be used to plot graphs and a printer driver to install within the operating system. Both the disk drive software and the printer/plotter software worked on a specific operating system, and worked only on one version. At operating system upgrade time, a new version of the peripheral software was needed. This required additional funds and the need to perhaps modify the existing data acquisition/analysis software to handle the new peripheral software. Partially for this reason, the operating system was never upgraded to the new system.

One critical system that had to be interfaced to the minicomputer was the E-Systems Control Console for the operation of the TBCs. The TBCs provided a single serial I/O port for eventual control of the TBC function from the minicomputer. However, for the initial (and only) use at ETS, it was desired that the minicomputer be able to load in ephemeris data for each day's testing. The software for this was written under the name CCUCON, and was set

to provide all the functions necessary for control by the minicomputer. The only function that was completely tested was the loading of the ephemeris data. The algorithm used to calculate the sun's position was based on an article by Robert Walraven.[2] This algorithm provided a generally accurate method of finding the sun. However, since the memory-based tracking system was used only to get the parabolic disk in the general vicinity of the sun, it was considered accurate enough initially. This was later updated to an algorithm based upon the work of John Stallkamp, of JPL, working with the Low Cost Concentrator. This proved to be much more accurate and required updating at the start of each calendar year with information from the Astronomical Almanac. This program loaded the TBC memory with the solar positions and was used each morning prior to the beginning of testing. It usually required 10 min to complete. This delay was due to the slowness of the TBC control to register and respond to the input from the minicomputer. After the loading, the testing staff ensured that the load was good by examining several coordinates by hand and comparing this with a printout from the minicomputer. In general, there were no problems.

The other major pieces of hardware that had to be interfaced to the minicomputer were the data loggers. A detailed description of the data logger is given above. The interfacing of the data loggers to the minicomputer was by an RS-232 serial I/O line. Setting up the data loggers was initially accomplished by hand setting toggle switches on the front. This was acceptable during the early stages of development, but later, when testing sequences reached 100 channels, this became tedious. To assist the testing staff, a program called LOGSET was written that scanned the SETPAR file for a particular logger and, taking that information, returned control sequence characters to set up the data logger as indicated by that file. The information in the parameters file that indicated the type of transducer, limits, channels to scan, resolution, and scanning frequency was loaded into the data logger. This procedure was quite lengthy, requiring 10 to 20 min, depending on the complexity of the channels being set. A listing of the control sequences for the data logger is included in Appendix C. This routine had to be used whenever the parameters file had been altered and whenever there was a power outage on the data loggers. This program relieved the testing staff from setting the data loggers and permitted them to perform other tasks.

Data acquisition from a different source, the JPL-developed flux mapper controller, is covered in a separate report.[3] The software used for this task was not used at ETS; but, rather, the data was gathered at ETS on digital cassette tapes and transported to the Foothill Facility for analysis. This analysis included the plotting of three-dimensional graphs depicting the flux at a location at or near the focal plane, printouts of the flux ratios, and contour plots showing lines of equal flux. Examples of these are given in the separate report.

---

[2]Walraven, R., "Calculating the Position of the Sun," Solar Energy, Vol. 20, pp. 393-397, 1978.

[3]Miyazano, C., Software Used with the Flux Mapper at the Solar Parabolic Dish Test Site, JPL Publication 84-76, September 15, 1984.

During all of the testing period, a separate meteorological subsystem was established and maintained at ETS near the test site. The instruments included in this meteorological subsystem were as follows:

Instrument	Manufacturer
Ambient Temperature	Meteorological Research
Wind Speed	Meteorological Research
Wind Direction	Meteorological Research
Dew Point	Meteorological Research
Pyrheliometer	JPL/Kendall
Pyranometer	JPL/Kendall
Barometric Pressure	Meteorological Research
Pyrheliometer	Eppley

This subsystem was operated 24 hours a day, 7 days a week, from late 1978 to mid-1984. Data was recorded at a frequency of one scan per minute. The system was active during the entire period with the exception of power failure periods and measuring device failures. This data was recorded using an Acurex Autodata-Nine data logger attached to a separate data buffer and formatting unit and a magnetic tape drive. The magnetic tape was changed monthly and all the weather information was provided on two charts each month to all cognizant engineers. One of these charts included graphs showing the pyrheliometer values for each day as a function of time. The other chart provided the minimum and maximum values for ambient temperature, wind speed, and barometric pressure for each day, along with an integrated total energy received per unit area measure for each day. A sample of this is provided in Appendix D.

The analysis of the data tape was performed at the Foothill Facility using a sequence of programs designed to store the data into FORTRAN-accessible files on the disk, search the data for the minima and maxima values, and generate the plots. The program, MS6DAY, performed the transfer from magnetic tape onto the disk. The data on magnetic tape was stored in 512 byte blocks of data as ASCII characters. Each scan was not an exact multiple of 512 bytes so that a single block on tape contained more than one scan. The program had to scan the data and determine the start of each new scan from the time mark, and store the data on the disk as a new scan. Another problem with this tape system was the lack of an end-of-tape mark on some of the tapes. This was due to such problems as operator error, power outages at the meteorological subsystem, and running out of tape. For these cases, the file on the disk would not close properly since the program did not terminate properly. This presented a problem that was solved only by monitoring the progress and instructing the software to terminate the scanning of the tape at the last known correct scan.

The magnetic tape drive handler was an assembly language routine that was developed by the software development staff to read one block of data from a tape and transfer the data to memory. This was one of the drivers which prevented the upgrading to updated versions of the operating system. It was not clear that this driver would function under the new environment without problems.

The program, MSS, took the disk data and created the monthly pyrheliometer charts. These were based on the values of the first pyrheliometer listed. If, from this chart, an anomaly appeared, then the second pyrheliometer value was used. The program, SEARCH3, searched the data files for the minima and maxima for each day as well as integrated the total energy received by the first pyrheliometer. This information was then used by the program, PLOT3, to produce the minima/maxima charts.

On several occasions, special plots were requested by various engineers for a variety of analyses. The most common request was wind speed data for several days. This was used for convective heat loss studies. Another common request was ambient temperature data for several days. This was used in conjunction with the Low Cost Concentrator expansion/contraction problem.

The weather station instrumentation was calibrated approximately once every 6 months. The pyrheliometers were calibrated by comparison of a laboratory standard with the instruments. All other calibrations were performed as stated in the manufacturers' manuals.

The weather station was dismantled to provide instrumentation for support of the parabolic disk at Rancho Mirage. The weather data tapes remain archived at JPL.

The environment at ETS presented major problems for the data acquisition equipment. The extremes of heat and cold were the most obvious problems. The minicomputer and peripheral equipment required constant air conditioning during the summer, and heating during the winter. Another problem not nearly as obvious was that of dust. The wind blew dust and sand into the minicomputer room, which were quite detrimental to the hard disk systems in the unit. In fact, one severe head crash of the 300-MByte disk drive was caused by the dust conditions in the minicomputer room. This dust problem demanded that both DEC and the third-party vendor for the 300-MByte disk drive make routine maintenance calls to clean and replace air filters. The power for the minicomputer room proved to be a problem as well. Additional power for just the minicomputer room had to be provided. Power in the trailer housing the minicomputer room was used by the adjacent building to power control panels for the TBCs as well as power data loggers and other analysis equipment.

The distance between the minicomputer room and the test control room where the terminals and data loggers were located was near the maximum of the RS-232 serial lines. Lower baud rates were used to ensure that data was not lost in transit. For the most remote test site, a 20mA current loop line was used for the terminal I/O, and a fiber optics cable was used for the data logger link.

SECTION VI

TRAINING

The training of the testing staff at ETS to use the data acquisition system was difficult at best. The testing staff was constantly short-handed and did not have time to go through a great deal of formal training. Most of the training that did occur was hands-on and, in short, one- to two-day sessions. Much of the initial use of the data acquisition software was done by an individual from the software development group that was at ETS for the day. He would conduct the data acquisition portion of the test and, simultaneously, show the testing staff the procedure for testing.

Also, several different versions of a User's Guide were produced. One for the RSX-11M FORTRAN is included in Appendix E of this report.

If a problem occurred during testing, someone from the software development group was available by telephone to assist. Occasionally, modifications were made by using a phone modem hookup between ETS and the Foothill Facility, although this was not done very often due to the noise that was present on the telephone lines.

For this type of development system, it is recommended that the testing staff either have one individual whose primary function is to operate the data acquisition system, or provide much more training for the testing staff on the use of the data acquisition software and hardware.

SECTION VII

RECOMMENDATIONS AND CONCLUSIONS

A. SOFTWARE

The software written was adequate for most of the testing at ETS. When several tests were conducted and shorter intervals between scans were desired, it became obvious that a hardware interrupt service routine was necessary. This is probably the one major software recommendation that can be made after 6 years of testing. The progression from the RT-11 BASIC to RT-11 FORTRAN to RSX-11M FORTRAN was inevitable. The flexibility provided by either newer versions of the RSX-11M operating system or the upgraded RSX-11M+ operating system may solve many of the difficulties associated with the operating systems. This will have to be re-evaluated. As for using this data acquisition software on another operating system, there are too many hardware and operating system-dependent functions. Transition to another operating system would be very difficult, despite the fact that the programming language is FORTRAN.

The disk access system of storage of data performed well; however, it used large amounts of disk space. Another system of storage may conserve disk space better and should be considered in any modification to the system.

B. HARDWARE

By starting with a DEC PDP-11/10 minicomputer, hardware alterations were impossible without a major investment in new hardware. However, to achieve higher scan rates and a larger number of channels, some method of direct memory access (DMA) for the input devices should be considered. The use of external data loggers would not be possible in this case. Several manufacturers provide DMA devices that will allow the input data to be scanned much more quickly and transferred to memory directly rather than through a serial port as used in this system. This would greatly increase the speed of all phases of data acquisition.

The selection of the data loggers should include a closer look at the interface between the data logger and the minicomputer. The data loggers used at ETS did not provide adequate interfacing for all of the features that would be needed for true compatibility between the data logger and the minicomputer.

C. CONCLUSIONS

The data acquisition system used at the PDTS was developmental, just as the modules tested there. The data acquisition software that was developed in support of testing at ETS was adequate for the task. Many problems were encountered in the development and use of the software, but data collection in support of testing was always handled in a timely manner. Supplemental software for the analysis of meteorological data, flux mapper data, and control of the TBC control console were developed on schedule. Any future use of this software should consider the recommendations concerning the use of hardware interrupt service routine in the software and the use of direct memory access data acquisition hardware.

APPENDIX A

SAMPLE GRAPHICAL OUTPUT FROM THE PLOT ROUTINE FOR TEST DATA TAKEN AT ETS

PDC1: CWCC 100% MIRR.

TEST RUN: CW0178 22-FEB-83

TIME INTERVAL BETWEEN SCANS: 30 SEC.

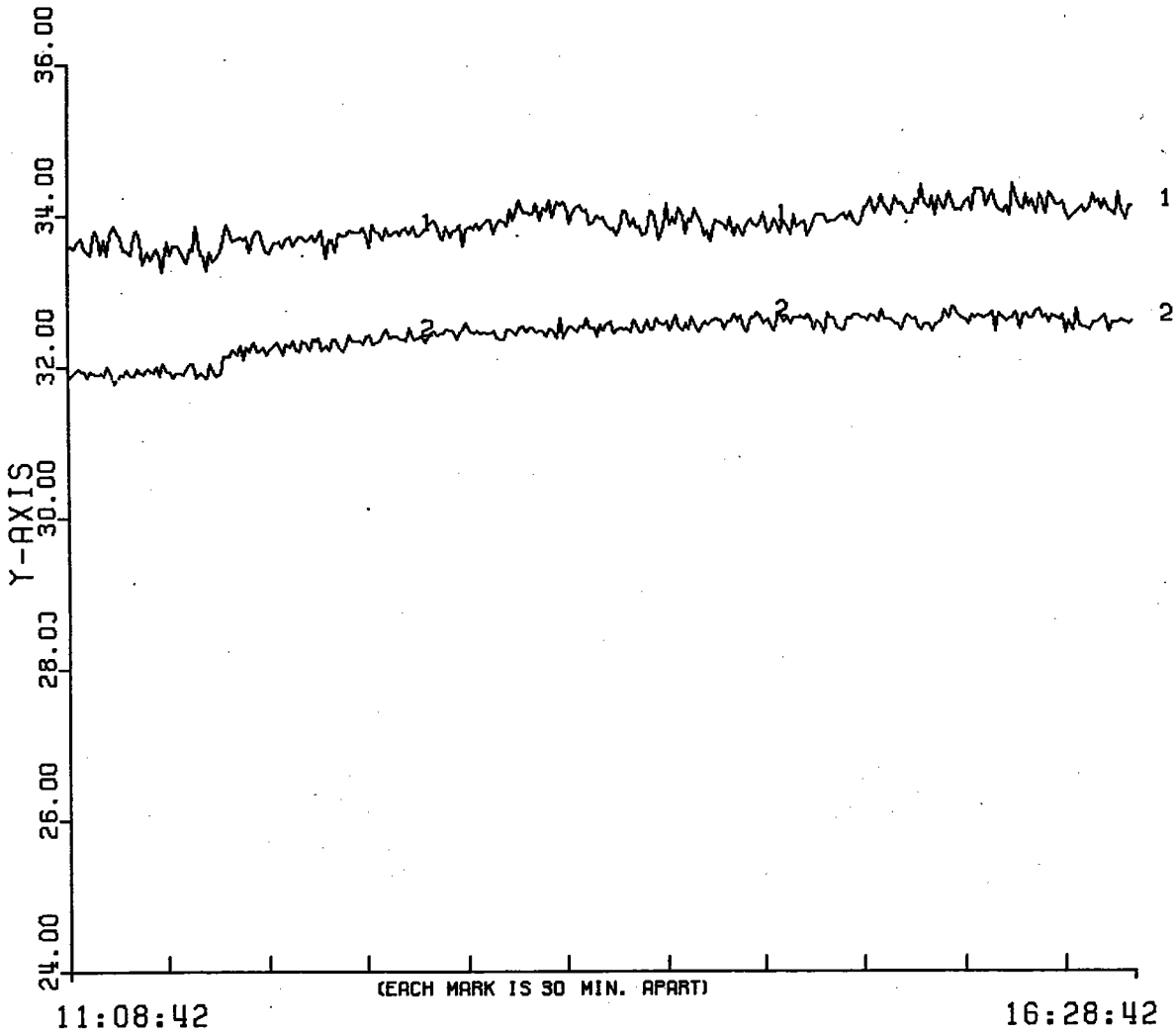
NUMBER OF SCANS: 642

1. FM CAL IN

GPM CHNL NO. 200

2. EX. APERTURE FLW

GPM CHNL NO. 213



PDC1: CWCC 100% MIRR.

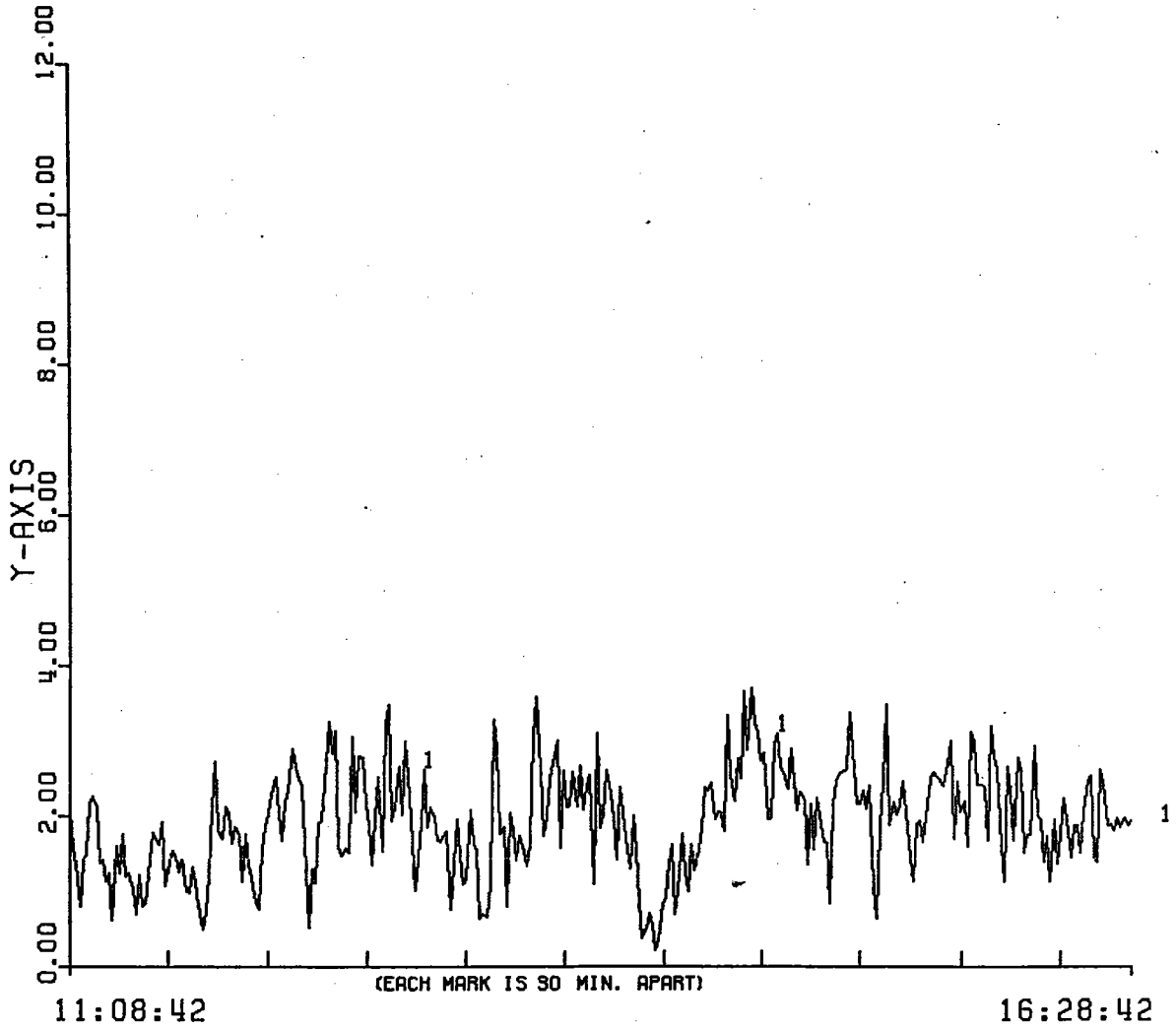
TEST RUN: CW0178 22-FEB-83

1. WIND SPEED

M/SEC CHNL NO. 211

TIME INTERVAL BETWEEN SCANS: 30 SEC.

NUMBER OF SCANS: 642



APPENDIX B

SAMPLE OUTPUT FROM THE COMMENTS (.CMT) FILE AND FROM THE NOTES (.NOT) FILE FOR A TEST

```
NOTE -- TEST NO.  ST0173
NOTE -- DATE:  21-JUN-82
NOTE -- TEST BED ID: TBC-2
NOTE -- MIRROR CONFIG: 100%
NOTE -- MIRRORS WASHED TODAY?  N
NOTE -- CLOUD COVER:  CLEAR
NOTE -- ENGINE/RECEIVER TESTED: ESOR II B HYDROGEN
NOTE -- ADD. MIRRORS COVERED:N
NOTE -- 457 MM Z AXIS POSITION
```

TEST RUN: STO173

DATE: 21-JUN-82 TEST MODULE: TBC-2

ENGINE/RECEIVER TESTED:
ESOR II B HYDROGEN

MIRROR CONFIGURATION: 100%

ADD. MIRRORS COVERED: NO

CLOUD COVER: CLEAR
MIRRORS WASHED TODAY: NO

ADDITIONAL NOTES:

457 MM Z AXIS POSITION
POST: THIS IS THE "MORNING" DATA FROM STO173
POST: ANOTHER RUN STO173 HAS THE "AFTERNOON" DATA FROM THIS TEST
POST: SEE OTHER STO173 RUN ON ARC TAPE 36 FOR DATA

APPENDIX C

ACUREX AUTODATA-NINE DATA LOGGER CONTROL SEQUENCES

Table A - Set Remote Commands

Command	Action	Command	Action
*	unlock remote command input	*MC.	mag tape on
*LO.	lock out remote command input	*MO.	mag tape off
*AA.	select "all data" record mode	*NC.	punch on
*AB.	select "all alarms" record mode	*NO.	punch off
*AF.	select "alarms once" record mode		
*AC.	alarms on	*PC.	printer on
*AO.	alarms off	*PO.	printer off
*AZ.	clear all alarms		
		*RC.	serial output on
*CC.	time, header and predata/digital inputs normal	*RO.	serial output off - (disables data and summary table outputs; does not affect echo)
*CD.	delete time, header and predata/digital inputs from serial output		
		*S1.	report summary Table 1
*DC.	predata/digital inputs on		
*DO.	predata/digital inputs off	*SC.	select continuous scan mode
		*SI.	select interval scan mode
*EC.	echo on	*S8.	select single scan mode
*EO.	echo off		
		*WC.	wait I/O on
*GO.	start scan	*WO.	wait I/O off
*HA.	halt scan and stop any operation in process		

Table B - Define Remote Commands

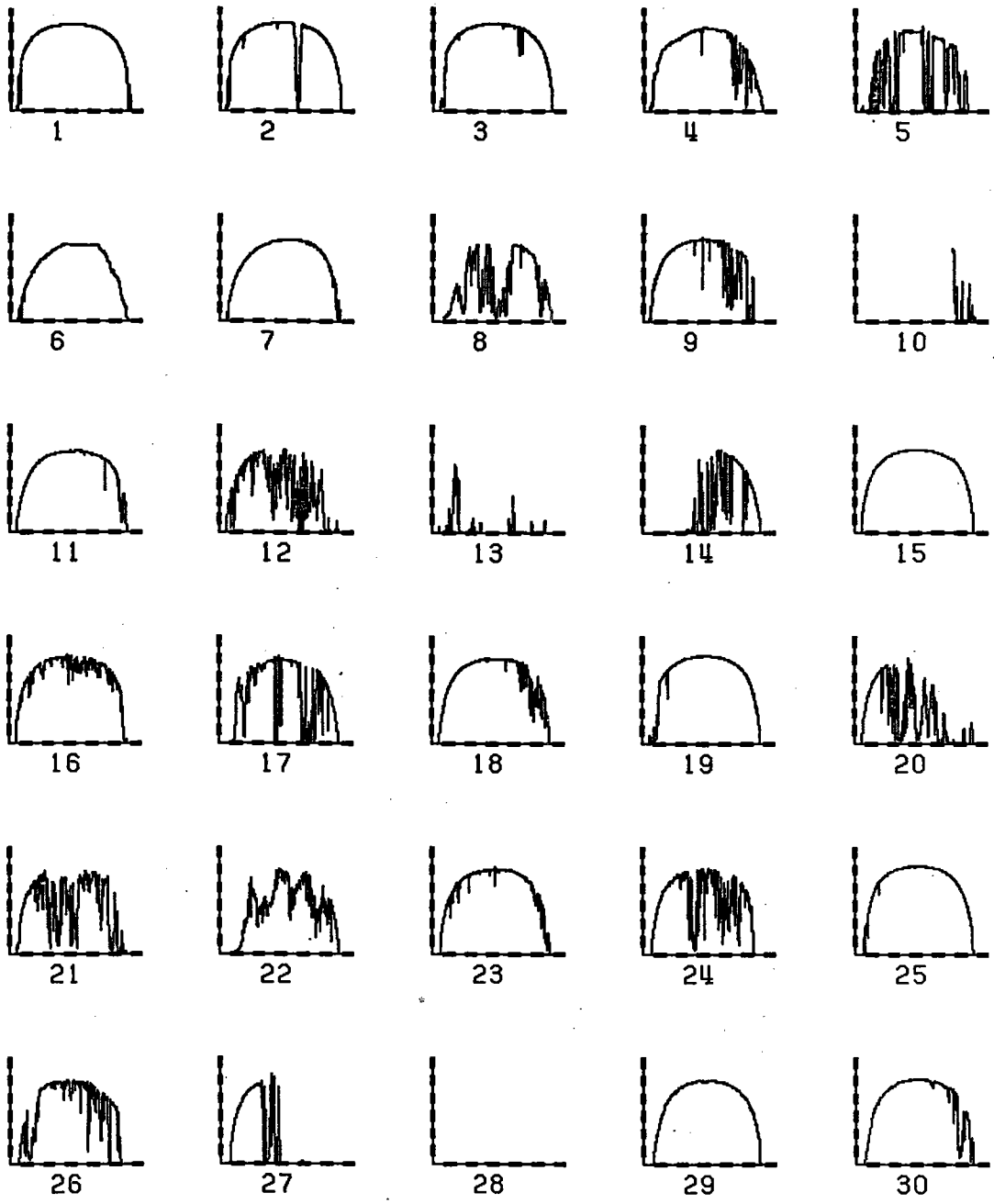
Command(a)*	Action	Example	
*AD(6).	define alarm beadband value	*AD 00060.	define beadband equal to 000060 units
*A1(9).	define alarm periodic log interval	*A1 000 00 15 30.	define log interval equal to 000 days, 00 hours, 15 minutes and 30 seconds
*AS(1).	define alarm state: D = clear 1 = +hi 2 = -hi 3 = +10 4 = -10	*A83.	define alarm state (+10)
*AV(9).	define limit number and value	*AV 001 100000.	define limit 001-equal to 100000 units
*AG(9).	assign limit to channels	*AG 001 012 015.	assign limit 001 to channels 012 through 015
*FC(3).	select first channel of scan	*FC 005.	select first channel 005
*LC(3).	select last channel of scan	*LC 099.	select last channel 099
*H1(6).	define header 1	*H1 123456.	define header 1 equal to 123456
*H2(6).	define header 2	*H2 987654.	define header 2 equal to 987654
*RA(3).	select one channel (random access) and inmate scan according to selected scan mode	*RA 017.	scan channel 017
*RT(9).	set real time	*RT 115 14.45.30.	set real time to day 115, 2:45:30 p.m.
*S2(6).	report summary Table 2 (I/T and resolution assignments) of selected channels	*S2 000 049.	report I/T and resolution assigned to channels 000 through 049 (see Figure 3)
*S3(6).	report summary Table 3 (limit assignments) of selected channels	*S3 014 019.	report limits assigned to channels 014 through 019 (see Figure 4)
*T1(9).	define scan interval	*T1 000 00 00 90.	define scan interval equal to 000 days, 00 hours, 00 minutes, 90 seconds
*TY(8).	assign resolution and I/T function to channels a. Resolution is "H" for hi, and "S" for standard. b. I/T function is "0" through "7" corresponding to pushbutton assignment on Autodata Nine front panel. 0 1 2 3 4 5 6 7	*TY 007 022 1H.	assign I/T function 1 and hi resolution to channels 007 through 022

*Digits in parentheses define the number of characters required in the argument. Spaces may be inserted anywhere in the remote command message for formatting convenience without without affecting command validity.

APPENDIX D

SAMPLE MONTHLY WEATHER SUMMARY PLOTS

The Weather Summary Plots show daily pyrheliometer readings and monthly minima/maxima for ambient temperature, wind speed, barometric pressure, relative humidity, and total solar energy integrated over the day.

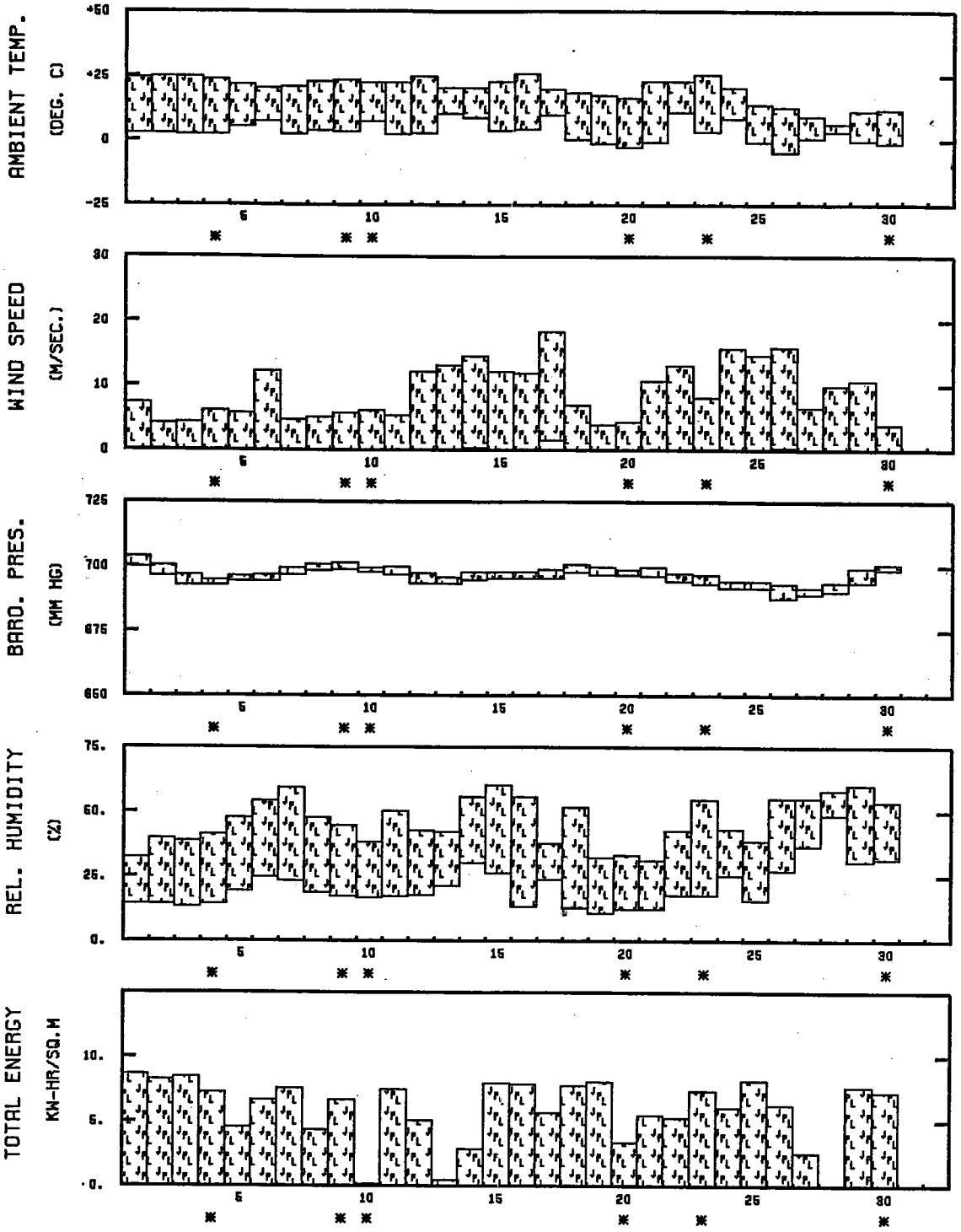


INSOLATION FOR NOV., 1981

JPL/PDTS

NORMAL INCIDENCE PYRHELIOMETER #1

MINIMUM/MAXIMUM VALUES FOR NOV., 1981



*INDICATES SOME DATA MISSING FOR THAT DAY. RESULTS MAY BE IN ERROR

APPENDIX E

RSX-11M FORTRAN VERSION "USER'S GUIDE," DATED MARCH 18, 1983

USER'S GUIDE

ETS COMPUTER SYSTEM
MARCH 18, 1983

CONTENTS

CONTENTS (Continued)

CHAPTER 1

INTRODUCTION

The document contains information on the use of the Digital Equipment Corporation PDP-11/34A computer located at Edwards Test Station.

In this document and in the computer file, the following convention is used to describe what is to be typed in by the operator. All messages to be typed in are in quotes (" ") except the carriage return which is displayed as '(CR)'. All your typing must be capitalized. In the text of this document, lower case words or letters means that you are to input the appropriate number, letter or word to complete the command.

If you need any help or have any questions, call 85-177-9131.

Thanks.

CHAPTER 2

STARTUP

This file assumes you wish to start up the computer.

2.1 COMPUTER

Startup procedure is as follows:

1. Turn on terminal TTO:
2. Power up mass storage unit (1st button on left).
3. Remove cover from computer.
4. Turn on computer (knob on right turn to DC ON).
5. Put RK05 disk marked 'System' into DKO: and move rocker switch from 'LOAD' to 'RUN'.
6. On the computer control panel, hit 'CNTRL' and 'BOOT' keys simultaneously.
7. On TTO: type

   "DK" (CR)

8. Follow the commands on TTO: (be sure to enter correct time)
9. Log in on TTO:

   "HEL DB1/DB1" (CR)

   (This is now a priviledged terminal).

10. Type in:

    "SET /UIC=[310,15]" (CR)

11. You have now completed the computer startup sequence.

2.2 TERMINAL

To log on to a terminal, do the following:

1. Turn on the terminal (toggle switch on back left side of terminal as viewed from front, toggle up is on). The terminal should reply with a beep to indicate that it has reset itself.

2. Log in on a terminal by typing

   "HEL DB1/DB1" (CR)

3. Type in:

   "SET /UIC=[310,15]" (CR)

4. Terminal is now ready for use.

5. If the terminal is for data display only, set the terminal to a slave condition by typing

   "SET /SLAVE=TTx:" (CR)

   Where x is the terminal number. To return the terminal to normal operation, type in, at another terminal, the following

   "SET /NOSLAVE=TTx: (CR)

   Where x is the terminal number.

2.3 CCU

Startup of CCU must be cleared with a Test Chief. The Test Chief or someone appointed by him will start up the system for you.

To update the ephemeris data in the CCU, do the following:

1. At the terminal, type in

   "RUN CCUCON" (CR)

2. Put the CCU in supervisory mode by pushing the following buttons in the control panel of the CCU in use, 'MAN', then 'ENTER', then 'SUPV', then 'ENTER'. The display should return with

   '[MPOS]'

3. Verify that the line printer is on in the trailer and that the three-way switch (next to terminal 5) is switched to the TBC you wish to load.

4. Follow the directions in the program.

5. Upon completion of the update, the full menu will appear again, at this point, verify a few points in the CCU memory by manually comparing the printout of ephemeral data with the contents of the CCU memory.

6. If the memory has loaded correctly, terminate the CCUCON program by typing in

"S" (CR)

7. Update the clock and the date on the CCU control panel.

8. You have now completed the loading of the memtrack memory in the CCU.

9. If the terminal keyboard is unresponsive, switch the three way switch to TT5 and hit the return key once. This should clear the computer line.

CHAPTER 3

PREPARATION

This file assumes you wish to prepare for a test run.

For software preparation, look at the SOFTWARE section.
For logger preparation, look at the LOGGER section.
For mag tape preparation, look at the MAG TAPE section.

3.1 SOFTWARE

To prepare the software for a test run, do the following:

(1) Using the DEC editor (EDT), make the necessary changes to the parameter table, such as STIRL.SET. This will include updating the following:

   (a) Title

   (b) Test number

   (c) Run number

   (d) Scan frequency

   (e) Scan interval between real-time printouts

   (f) Plots

   (g) Alarm changes

   (h) Deletion of channels

   (i) Addition of channels

   (j) Conversion factor changes

   Upon completion of any necessary changes, exit the editor by typing

   "EX (CR)"

(2) Use PIP to transfer this parameter table to the working table for the test run, by typing

"PIP TBLx.SET=STIRL.SET" (CR)

where the x is the concentrator number.

(3) To prepare the computer-understandable run table, type

"RUN DBGEN" (CR)

and answer with the correct table number.

(4) If calculations are to be made, type

"RUN DECODE" (CR)

and follow the directions.

(5) If a real-time display is desired, type

"RUN NWDBGN" (CR)

and follow the directions.

(6) At this point, a usable SETPAR table is ready.

3.2 LOGGER

To set-up the logger for a test run, do the following:

(1) Be sure the red lockout switch inside the logger front door is in the up position.

(2) Be sure the RS-232-C cable is connected to the port in the back of the logger.

(3) Check the logger front panel. A listing of the computer port number will be taped there.

(4) With a usable SETPAR table available, type

"LOG" (CR)

and answer the questions. The logging sequence takes approximately 15 minutes.

(5) When the LOG program terminates, correct the logger time with the program TRY (but not from terminal TTO:). Do this by typing

"TRY" (CR)

Enter the logger port number (listed on logger), enter the terminal port number (listed on the terminal -- 0, 1, 5, 6, or 7), and when the command is requested, type in

"*RTdddhhmmss," (not shown on screen)

where ddd is the day number, hh is the hour, mm is the minute, and ss is the second.

Enter the comma when the exact time is at hand [do not enter a (CR)].

(6) The logger is now ready for use.

3.3 MAG TAPE

To set-up the mag tape unit for a test run, do the following:

(1) Mount a magnetic tape on the tape drive. Be sure that the "WRITE-ENABLED" ring is in place.

(2) Make sure adequate tape is available (1200' needed for a 5 hour test).

(3) If this is a new tape, then you must initialize the tape first. This is done by typing

"INI MT:ARCxx" (CR)

where xx is the number of the ARC tape. This number can be obtained from the log book.

(4) Type in

"MOU MT:/OVR" (CR)

The tape should move forward 5 cm and rewind. This indicates the tape is now mounted and the system acknowledges this.

CHAPTER 4

ACQUISITION

This file assumes you wish to begin the acquisition of data. For inclusion of a comment, look at the COMMENT section below.

To being, do the following:

(1) Begin by verifying that the SETPAR table has been updated, the logger has been programmed, and the time is correct in both the computer and the logger.

(2) To begin the acquisition of data, type

"RUN EXECUT" (CR)

and follow the directions.

(3) Acquired raw data is written into a file,

LOG1.ACQ	For Table 1 Data
LOG2.ACQ	For Table 2 Data

and so forth.

(4) Messages of scan number should appear on the terminal from which you started the task.

(5) Alarms and screen display (if any) will appear on TT5: for Table 1 data, TT6: for Table 2 data, and TT7: for Table 3 data.

4.1 COMMENT

To enter a comment into the data record during a data run, do the following:

(1) Use a terminal not being used for screen display of data.

(2) Enter the comment by first typing

"RUN EXECUT" (CR)

and follow the directions listed. Be sure to wait until the terminal directs you to enter the comment.

(3) Try not to tie up the terminal too long. The longer you take to enter the comment, the less time the computer has to acquire data.

CHAPTER 5

TERMINATE

For standard post processing procedures, the following instructions should be followed. These instructions assume that there are multiple tests for the day and that some of these have been processed during the day.

5.1 TEST COMPLETED EARLY IN DAY

A test has been completed early in the day and there is sufficient time to process the data completely before terminating the active test. If this is the case, then the procedure to follow is:

(1) Use "@POST" (CR) to begin post processing. You will need to have the test ID number and the number of the table used in order to answer all the questions.

(2) The POST routine will process the data and store it on a disk using the test ID number as the file name. When data reduction is complete, a message will appear on the terminal in which the @POST command was given, indicating that the processing is complete.

(3) Now, all the files for this test have been processed. You can now proceed to the section dealing with the printing of the data.

5.2 TEST COMPLETED LATE IN DAY

A test has been completed late in the day and there is insufficient time to process the data completely before terminating another active test. If this is the case, then the procedure is:

(1) At the keyboard, type in

"RUN EXECUT" (CR)

and state that you do NOT wish to process immediately.

(2) When the remaining active test is terminated, you may begin analysis of this test overnight using the command

"@DELAY1" (CR)

Answer the questions that follow. When the task asks if you wish to process overnight, type in

"Y" (CR)

(3) The post processing will begin in 4 hours. However, it will not be printed out.

5.3 LAST TEST OF THE DAY

This is the last test of the day and all other tests have either completed analysis or have been halted using the "RUN EXECUT" command. If this is the case, then to begin immediate analysis of the data,

(1) Type in

"@POST" (CR)

and answer that you wish to begin analysis of the data immediately. Identify the logger used and the Test ID run number to the program. The pogram will now analyze the data to completion. However, it will not be printed out.

5.4 FAST RESTART

This file assumes that you have just completed a test run (or the system has had a problem and abnormally terminated the program) and you wish to restart immediately.

(1) Be sure that the system has not rebooted automatically. (This can be done by checking to see if the cursor returns when the "RETURN" key is pressed.) If necessary, reboot, otherwise go on to step 2.

(2) If everything appears to be alright, then copy the scan number from the control terminal. This will be necessary to reduce this first file later.

(3) Once you have this number (or the time of the last scan), type

"@FASTx" (CR)

Where x is the number of the logger or table being used. This will pip over the LOGx.ACQ file to a holding file named "HOLDx.RAW" and the NOTE.oox to a holding file named "HOLDx.NOT".

(4) The computer will automatically place you into EDT. You will have to change the number of the test run to the next higher number before you can begin.

(5) Once you have exited EDT, the program will automatically run DBGEN, DECODE, and NWDBGN (you will have to enter the table number).

(6) Once you have completed these, you are now ready to begin taking data once again. Follow the instructions listed in the "ACQUISITION" section of this report.

Warning

This can only be used once. Once it is used, the post processing must be done to both before being used again. Failure to do so will cause erasure of data.

5.5 LATER

This file assumes that you wish to terminate a test run and reduce the data at some convenient time later.

If you wish to start analyzing data from a previously completed run using this procedure, look at the section titled "REDUCTION LATER".

(1) To terminate the currently executing logging task and start the analysis later, type

"RUN EXECUT" (CR)

When the instructions appear, type in the appropriate letter to terminate the logger task in progress.

(2) A question will appear on the terminal asking if you wish to begin reduction of data, type

"N" (CR)

(3) You will see two stop messages on the terminal, one for the task EXECUT and one for the task LOGGER. After these two have appeared, the system has now completed logging and is free to be used for some other application.

(4) BEFORE DOING ANYTHING ELSE, use PIP to change the name of the raw data file. If you do not, you could write over the just-gathered data and eliminate it!! Type

"PIP yyyyyy.RAW=LOGx.ACQ/RE" (CR)

where yyyyyy is the test ID name and number (e.g. ST0111) and x is the number of the logger used (same as the TBLx.SET number).

WARNING

Failure to perform step 4 during this type of operation might result in the loss of all accumulated data from this test run.

CHAPTER 6

REDUCTION

This file assumes you wish to begin the reduction of data and have used the "@POST" (CR) command to terminate the test.

For fast restart recovery, look at the REDUCTION RESTART section. For reduction of data from a LATER termination, look at the REDUCTION LATER section.

(1) If, upon completion of the test run, "@POST" (CR) was typed into the command terminal, and a "Y" (CR) was typed when asked if the system should begin reduction immediately, then the system will automatically reduce the data and store the results on the large disk. However, it will not printout the results nor place them on mag tape.

(2) To transfer the data to mag tape and print out the results, see the section on printing the data.

(3) Be sure the line printer is 'ON' and that there is sufficient paper. Advance enough paper such that the fan-fold process can begin in the catch basket.

(4) Be sure that an archive mag tape is on the drive and that the tape has been mounted with the commands

"MOU MT:/OVR" (CR)

If this is a new tape, then see "HELP PREP MAG" for appropriate commands.

(5) The system can now be left to complete this task.

(6) Upon completion of this task, update the master file with the correct ARC tape number by using

"EDT MASTER.FIL" (CR)

6.1 DELAY

This file assumes you wish to begin the reduction of data later in the day or overnight and have used the "RUN EXECUT" (CR) command to terminate the test.

(1) Be sure that the terminal that you are using was not used to start any other command files (files that start with the '@' symbol).

Warning

> If there are several tests to reduce later, then be sure that the tests are staggered overnight. If two tests try to access the same program, the second test will abort abnormally and analysis will not occur. To prevent this, see the last instruction in this section.

(2) To begin data analysis, type in

"@DELAY1" (CR)

at a terminal not currently being used.

(3) Answer the questions that will be asked such as the identification number of the run and the TBLx.SET being used for the run.

(4) The computer will now ask if you wish to process the data overnight. If you answer "N" (CR), then the computer will wait until the command

"RES xxxxxx" (CR)

is typed in, where xxxxxx is the name of the command file. This name is listed on the command terminal at the time of the pause initiation.

(5) The computer will now ask if you wish to process immediately. If the answer is "Y" (CR), then proceed to the PRINTOUT OF DATA section in TERMINATE chapter. If the answer is "N" (CR), go to the next instruction.

(6) The computer will now ask if you wish to process overnight. If the answer is "Y" (CR), then the computer will begin analysis in 4 hours. If the answer is "N" (CR), then the computer will place you back into pause and you will have to go up two instruction steps.

(7) If there is already a test being reduced using the command file "DELAY1", then repeat the above instructions using the command file "DELAY". The only difference will be that the time delay for overnight processing will be 8 hours instead of 4.

6.2 RESTART RECOVERY

This file assumes that you had to use the FAST RESTART sequence to terminate a test run in an abnormal fashion, such as the system rebooting itself.

Warning

> All other logger data must be processed prior to the use of this sequence. Failure to do so may result in the loss of data.

(1) Be sure that all other logger data has been completely reduced.

(2) If RUN EXECUT was used, go to the next step. If RUN EXECUT was not used,

(a) Check to see that a comment file exists for the run by typing,

"PIP yyyyyy.CMT" (CR)

where yyyyyy is the test run number (e.g. ST0100). If it exists, go on to the next instruction. If not, then type in the following to create one.

"PIP yyyyyy.CMT/NV=NO.CMT" (CR)

(b) Take the scan number that you copied from the control terminal screen during the fast restart, divide that number by the number of scans per minute that were being taken during the test and round this result to the nearest whole number. This is the number of minutes of testing.

(c) Type in

"EDT TBLx.SET" (CR)

and edit the run number back to the original number and edit the number of minutes to the one calculated in the instructions above. (Be sure there are three digits in the number). Exit from edit.

(d) Type in

"PIP yyyyyy.SET/NV=TBLx.SET (CR)

where yyyyyy is the test run name (e.g. ST0100) and x is the number of the logger used (same as the TBLx.SET number).

(e) To add the NOTE file to the CMT file, type in

"RUN EXECUT" (CR)

and type in the appropriate letter to terminate the test. When asked, do you wish to process immediately, answer "N".

(3) Type in

"@PROCESS" (CR)

Answer the questions. This will now complete the analysis of the data. Upon completion, you can use '@PRINTOUT' to printout the results.

6.3 LATER

This file assumes that you terminated a test using the procedure outlined in TERMINATE LATER and are now ready to process the data, or that you have

followed the directions outlined in REDUCTION RESTART RECOVERY and are now ready to proceed with the data reduction.

(1) Be sure that the TBLx.SET table is correct with the proper test run number and the proper number of minutes of testing. The number of minutes can either be obtained from the yyyyyy.SET table by using the command

"PIP TI:=yyyyyy.SET"(CR)

and getting the number of minutes from the third line, or can be calculated as done in the REDUCTION RESTART RECOVERY section.

(2) Type in the following commands and enter the appropriate table numbers when asked:

"RUN DBGEN" (CR)
"RUN DECODE" (CR)
"RUN NWDBGN" (CR)

(3) If you terminated the test using the procedure outlined in TERMINATE LATER, then you must execute one additional command. Type in

"PIP LOGx.ACQ/NV=yyyyyy.RAW/RE" (CR)

where yyyyyy is the test ID name and number and x is the number of the table use (See instruction 4 in TERMINATE LATER section).

(4) Now type

"@DELAY1" (CR)
or "@DELAY" (CR)

and answer the questions.

(5) You will be asked if you wish to process overnight. If you wish to start processing overnight, type in "Y" (CR), and go to the PRINTOUT OF DATA section in the TERMINATE chapter. If you wish to process immediately, type in "N" (CR), and go to the next step.

(6) If you answered "N" (CR), then the computer will wait until the command

"RES xxxxxx" (CR)

is typed in, where xxxxxx is the name of the command file. This name is listed on the command terminal at the time of the pause initiation.

(7) The computer will now ask if you wish to process immediately. Type in the answer "Y" (CR), and proceed to the PRINTOUT OF DATA section in TERMINATE chapter.

CHAPTER 7

PRINTOUT OF DATA

Now that the analysis of data has either been completed or has yet to begin, you may now start the printout command file. Please note that this command file should be used only AFTER the last test of the day has been completed.

(1) At a terminal that is not currently processing a command file (any terminal that does not have an @xxxx file working) type in the command,

"@PRINTOUT" (CR)

(2) A series of questions about today's tests will appear on the screen. You will need to type in the test run ID number, the number of printouts and the number of sets of plots desired.

(3) If a set of test data is still being processed, then you will have to answer "N" when asked if the data is ready to print out. If this is the case, then a later question will ask you when the test should be ready. Type in the number of hours based on the terminating command file used. For example,

if DELAY1 is used, type in "6" hours,
if DELAY is used, type in "10" hours,
if POST is used, type in "2" hours.

If two of the three above command files are being used, then type in the greater of the two numbers.

(4) Be sure that the line printer is on and has sufficient paper. Be sure that the mag tape unit is on line and mounted.

(5) The printing should be completed before the next morning.

7.1 ADDITIONAL COPIES

(1) For more copies of the last set of plots, type

"RUN RASM" (CR)

(2) For more copies of the data, type

"PIP xxxxxx.LST/SP:y" (CR)

where xxxxxx is the file name of the data to print out, and y is the number of copies desired.
If only one copy is desired, the "/SP:y" can be left off.

CHAPTER 8

SHUTDOWN

This file assumes that you wish to shutdown the computer system in the trailer, T-1600.

To shutdown a terminal, look at the SHUTDOWN TERMINAL section.

(1) Log off on all terminals except TT0: in the trailer.

(2) From TT0:, type

"RUN SHUTUP" (CR)

(3) The computer will ask for the number of minutes before shutdown. Type in

"0" (CR)

(4) The computer will now execute the shutdown procedure.

(5) When the RUN light on the front panel of the computer goes out, then power down the RK05 disk drives by pushing the 'RUN-LOAD' rocker switch to the 'LOAD' position. When the 'LOAD' light is on, remove the RK05 disks and store in the appropriate cabinet.

(6) Turn off the mass storage unit by pressing the 'START' button (on the far left side). The red lights above the row of buttons should start to flash, indicating that the disk is spinning down.

(7) Turn the rotary switch on the right side of the computer front panel to the 'DC OFF' position. The 'DC ON' light just above that should go out.

(8) Go to the rear of the computer and turn off the power to the transformer. A switch on the left side of the box on the transformer should be in the 'DOWN' position. The red indicator light should be out.

(9) Cover the computer.

8.1 TERMINAL

To shutdown a terminal, do the following:

(1) At the keyboard, type

"BYE" (CR)

(2) A log off message will appear on the screen. When this message has completed, turn off the terminal. (Toggle switch on back left side of terminal as viewed from front, toggle down is off.)

CHAPTER 9

PROBLEMS

This file assumes that you are having a problem with one portion of the data acquisition system. For more detailed help, look at the appropriate section for the following devices:

COMPUTER
LOGGER
MAG TAPE UNIT
PRINTER
TERMINAL

Be sure to include a notation in the Computer Log book for all computer related problems.

If none of these fix the description, then try calling 85-177-9131.

9.1 COMPUTER

(1) If the system is crashing on a sporatic basis, the best advice is to call DEC service and request help.

(2) If the computer will not start up in the morning, and the vent fans on the top of the cabinets are working, check the circuit breakers in the computer cabinets.

(3) If during bootup, the run light does not come on, call DEC field service and request help.

(4) If, during routine running (not data logging, not loading the CCU and not setting up the logger), the computer becomes very sluggish and does not respond quickly, check to see if someone has turned one of the data loggers to manual control.

(5) For all other problems, the best advice is to call DEC service.

9.2 LOGGER

(1) If the logger is on, but you can not use the LOG or TRY commands, check to see if the RS-232-C connector is still connected, check to see if the "LOCK-OUT" switch is in the up position, and check to see if the "SERIAL OUT" toggle is in the up position.

(2) If the logger communicates with the computer but will not set alarms properly, first verify that the correct table is being used to set the alarms, then have the alarms card exchanged with another unit.

(3) If the logger shows "Stuck in Loop" messages during the LOG routine, then abort the LOG routine, check the connections on the logger, run WAK on the logger port, and try again.

(4) If the logger was being used in the LOG routine and the log routine aborted abnormally, run WAK to clear the logger port and try again.

9.3 MAG TAPE UNIT

(1) If a mag tape is on the drive and the computer refused to execute a "MOU" command, check to see if the magtape is new and has not been initialized yet. If so, initialize mag tape.

(2) If an old mag tape is on the drive and the computer refuses to initialize, demount the mag tape (Caution -- "DMO" command can crash system).

(3) If a new tape is on the drive and the system has been up all night and the computer refuses to initialize or mount the tape, then the system still assumes that a tape is mounted. Use the "DMO" command to demount the mag tape (Caution -- "DMO" command can crash system).

(4) If during an attempted transfer of data using PIP, the computer refuses to follow the command, check to see if the "WRITE-ENABLE" ring is installed.

9.4 PRINTER

(1) If print is light, check toner fluid level. Sometimes, shaking the tone bottle to mix the toner will darken print.

(2) If print is erratic, try moving the cable around the Versatec cable is very fragile and connections that are broken may come into contact again.

9.5 TERMINALS

(1) If a terminal is on but does not communicate with the computer, check the RS-232-C cable both at the computer end and the terminal end.

(2) If a terminal is on and has been working but does not permit you to type in, push "SET-UP" key and then the "RESET" key.

(3) If a terminal is on, has been working, and the "ON LINE" light is not on, check the keyboard cable. Be sure its plugged in.

(4) If a terminal is on and random characters appear on the screen, baud and/or parity is incorrect. Try pushing "SET-UP" key and then the "RESET" key.

(5) If a terminal was being used for weather data and is to be changed back to a computer terminal, be sure that the rotory switch on the box to the right of the terminal is set back to the designation, "TT5".

(6) If a terminal is being used for SCREEN-ALARM output and the output has stopped appearing, a key on the keyboard may have been pushed by accident. Hit the "RETURN" key a few times to clear this.

(7) If a terminal will show the characters you type on the screen, but the computer refuses to return a prompt or to execute your commands, there is a logger that is not suppose to be on, putting data into a computer line. The computer interprets this as a terminal trying to log on, so it spends time trying to match log on messages. Turn off the logger that should not be on.

(8) If a terminal was being used for a TRY or LOG command and the program was aborted abnormally, the terminal may not work. Try WAK from another terminal to clear it.

(9) If a command file will not work on a terminal (any file that is entered by typing "@xxxxxx"), then the terminal is being used by a command file already. Each terminal can only handle one command file at any one time. Try going to another terminal and entering the command.

CHAPTER 10

COMMANDS

10.1 COMMANDS

The following are system level commands that might come in handy.

(1)	To stop the printer when The "SP:y" option is used	"ABO PRT..." (CR)
(2)	To see what tasks are executing at your terminals	"ACT" (CR)
(3)	To see what task are executing at all terminals	"ACT /ALL" (CR)
(4)	To get out of pip	"(CTRL C) ABO PIP" (CR)
(5)	To get a listing of files on the storage (DBO) disk	"PIP DBO:/LI" (CR)
(6)	To get a listing of files on magnetic tape	"PIP MT:/LI" (CR)
(7)	To find out the amount of free space that remains on disk	"PIP /FR" (CR)
(8)	To see what devices are loaded and/or mounted on the computer	"DEV" (CR)
(9)	To see the time	"TIM" (CR)

CHAPTER 11

TAPE BACKUP

11.1 TAPE BACKUP

This section assumes that you wish to backup the DB1: disk onto tape for archival storage or as a precaution against a disk crash. This is to be done when the system is NOT performing any other tasks.

(1) If an arc tape is installed and mounted on the tape drive, remove it with the following command:

"DMO MOT:ARCxx" (CR)

where xx is the arc tape number. Note that this procedure may cause the system to crash.

(2) Install a new reel of magnetic tape onto the tape drive. It should be a 2400 foot (largest) reel. Be sure there is a write-enable ring installed on the reel. Place the tape unit 'ON LINE'.

(3) At the console terminal, type the following commands:

"DMO DB1:" (CR)
"RUN DB0:[100,4]BRU" (CR)

(4) You will now get a different cursor that identifies the program you are running as BRU. Type in the following commands at the terminal:

BRU)/VERIFY/BACKUP_SET:ddmmmyy/REVISED:AFTER:xx-yyy-zz
FROM: DB1:
TO: MT0:

where ddmmmyy represent today's data (e.g. 01JAN83) and xx-yyy-zz represent the day before the last backup was executed (e.g. 20-DEC-82). Note that the underlined works in the above command are typed by the computer.

(5) The tape will make two passes--the first to write all of the new and revised files onto tape and the second to compare the tape files with the disk file. If there are a large number of files to back up, then two reels of tape may be necessary. The terminal will inform you if another reel is necessary.

(6) Upon completion, remove the tape and label it as a BRU backup of ETS DB1: giving all the information you listed on the line marked BRU) above.

(7) To exit from BRU, type in:

(CTRL Z)

(8) Remount the DB1: disk by typing:

"MOU DB1:/OVR" (CR)

(9) You have now completed the backup of the disk.

CHAPTER 12

KNOWN BUGS AND PROBLEMS

12.1 KNOWN BUGS AND PROBLEMS

This section is a summary of the known problems and bugs with specific applications programs. A work-around, if it exists, is included to bypass the problem. Any new or different problems should be reported as soon as possible to the Foothill personnel. As problems and bugs are fixed, they will be deleted from this section.

(1) LOG problems.

(a) Problem - Does not set clock time properly.

(b) Cause - Slow sending of the time characters results in a lag between inquiry of time and final sending of time to data logger.

(c) Work-Around - Use TRY to update the time after LOG is completed, as indicated in the last instruction of section 3.2.

(2) LOGGER Problems.

(a) Problem - Screen display can only display 23 channels maximum.

(b) Cause - Unknown at this time.

(c) Work-Around - Limit display to 23 channels.

(a) Problem - screen display gives erroneous data if calculations are mixed in with the data channels.

(b) Cause - Unknown at this time.

(c) Work-Around - Place all the calculations that are to be displayed on the screen at the end of the display listing.

(3) PLOT problems.

(a) Problem - Plots without data are generated if only calculations are plotted.

(b) Cause - Unknown at this time.

(c) Work-Around - At least one data channel must be plotted at the beginning of the plot series, then all plots are correct.

(4) CCUCON Problems.

(a) Problem - Divide-by-zero errors appear on screen during CCU loading during certain times of the year.

(b) Cause - When the sun's azimuth angle approaches 90 degrees, a division is performed using the sin or cos of the angle, causing the error to appear.

(c) Work-Around - Ignore the error; it does not affect the results.

```
;TITLE POST.CMD
RUN EXECUT
.WAIT EXECUT
.5:
.IFNACT CNVRT  .DELAY 20.S
.IFNACT CNVRT  .GOTO 5
.ENABLE SUBSTITUTION
.ASKN [1:3] TBL ENTER LOGGER NUMBER
.ASKS [6:6] FILE ENTER TEST ID RUN NUMBER (6 CHRS)
.WAIT CNVRT
.WAIT NEWVRT
.WAIT NWPRNT
.WAIT PLTSET
.WAIT PLOT
PIP 'FILE'.BAN/NV=VECTR1.BIN/RE
PIP 'FILE'.BBN/NV=PARM.BIN/RE
PIP 'FILE'.LST/NV=TRANS'TBL'.LST/RE
PIP 'FILE'.NOT/NV=NOTE.00'TBL'/RE
;
;       >>>> END OF POST PROCESSING FOR TBL 'TBL'
;
.END:
```