https://ntrs.nasa.gov/search.jsp?R=19720003677 2020-03-11T21:55:14+00:00Z

UNIVERSITY OF ILLINOIS
URBANA

AERONOMY REPORT NO. 42

PARTIAL-REFLECTION DATA COLLECTION AND PROCESSING USING A SMALL COMPUTER

by
M. H. Birley
C. F. Sechrist, Jr.

June 1, 1971

Supported by
National Aeronautics and Space Administration
Grant NGR-013
NGR 14-005-013

Aeronomy Laboratory
Department of Electrical Engineering
University of Illinois
Urbana, Illinois

UILU-ENG71-2501

AERONOMY REPORT

NO. 42

PARTIAL-REFLECTION DATA COLLECTION AND PROCESSING USING A SMALL COMPUTER

by

M. H. Birley
C. F. Sechrist, Jr.

June 1, 1971

Supported by
National Aeronautics and Space Administration
Grant ~~NGR-013-~~ NGR 14-005-013

Aeronomy Laboratory
Department of Electrical Engineering
University of Illinois
Urbana, Illinois

ABSTRACT

On-line data collection of the amplitudes of circularly polarized radio waves, partially reflected from the D region of the earth's ionosphere, has enabled the calculation of an electron-density profile in the height region 60-90 km. A PDP 15/30 digital computer with an analog to digital converter and magnetic tape as an intermediary storage device are used. The computer configuration, the software developed, and the preliminary results are described.

TABLE OF CONTENTS

LIST OF SYMBOLS

A = amplitude of reflection wave

c = velocity of light = 3×10^{8} ms^{-1}

e = electron charge = 1.6×10^{-19} C

h = height above Earth's surface, m

K = absorption coefficient

m = electron mass = 9.1×10^{-31} kg

N = electron density, m^{-3}

n = complex refractive index = $\mu - i\, ck/\omega$

o = ordinary mode of circular polarization

R = reflection coefficient

x = extraordinary mode of circular polarization

$\mathcal{C}(x) = \frac{1}{p!} \int_0^\infty \frac{\varepsilon^p}{\varepsilon^2+x^2} e^{-\varepsilon}\, d\varepsilon$

ω = angular frequency of the transmitted wave

ω_o^2 = plasma frequency = $Ne^2/m\varepsilon_o$

ω_L = longitudinal component of the angular gyrofrequency

ε_o = permittivity of free space = 8.85×10^{-12} Fm^{-1}

ν_m = mono-energetic collision frequency, s^{-1}

Δh = the height interval, m

μ = phase refractive index

LIST OF ABBREVIATIONS AND ACRONYMS

ADC	Analog to digital converter
ADC system	A real-time data acquisition system which inputs data directly into the digital computer via the ADC
ADC	The name of the PDP 15/20 1/0 Handler program, which supervises operation of the ADC
API	Automatic Priority Interrupt
AO75	Ordinary mode amplitude reflected from 75 km
AO80	Ordinary mode amplitude reflected from 80 km
AX75	Extraordinary mode amplitude reflected from 75 km
AX80	Extraordinary mode amplitude reflected from 80 km
B/F	Background/Foreground
BFKM+ADC	The name of the B/F system tape which incorporates the ADC
CPU	Central processing unit
.DAT	Device assignment table
DEC	Digital Equipment Corporation
HP	Hewlett Packard
I/O	Input/Output
IPU	Input output processing unit
PDP	Programmed Data Processor
PRADDL	Partial-reflection automated digital data logging
SOMED	Synoptic observation of mesospheric electron densities
SOMED system	(Reynolds or papertape system) A data acquisition system which utilized papertape for temporary storage of experimental data

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

The history of the partial-reflection experiment from its proposal by Gardner and Pawsey (1953), to its inception at the University of Illinois in 1968, is adequately described elsewhere (Pirnat, 1968; Reynolds, 1970). The same sources contain a description of the theory related to the experiment, the assumptions involved, the derivation of the Sen-Wyller (1960) quasi-longitudinal approximate equation, the reflection of radio waves from step discontinuities in the D region of the ionosphere, and the form of the reflected waves at the receiver output. They also describe the partial-reflection system in use at the University of Illinois, which is a hardware system.

The signal at the output of the partial-reflection receiver is a time-varying voltage with the form shown in Figure 5.1. A circularly-polarized wave is transmitted in the ordinary mode and the reflected signal is of interest for the following 800 microseconds; 33 milliseconds later the process is repeated for the extraordinary mode. Two methods of storing these data have been used to date; this report describes a third.

The first method is that used by Pirnat (1968). The short time interval between the ordinary and extraordinary wave transmission makes it possible to display the reflected signal from both simultaneously on an oscilloscope with a reasonably persistent phosphor. The extraordinary wave amplitude is inverted for clarity. The resulting A-scan can be photographed with a camera driven by a synch pulse. Data are collected and a photograph is taken once every two seconds. In order to be meaningful an average over several minutes must be used. It would be convenient to have amplitude data at 1.5 km intervals in the region 60-90 km. Therefore, 2700 data points must be read off the photographic record. The process of reading a data point is one of measuring the ordinate of the trace. Clearly, the problem of data reduction makes the method prohibitive.

The second method is that of Reynolds (1970). The receiver output is sampled three times for each mode. The sampling times correspond to waves reflected from the heights: 45, 75 and 80 km. The first sample constitutes a measure of the noise level during the interval. The second and third samples constitute the data. Hard-wired logic was used and two data samples, for each mode, were punched on paper tape. In this way an average electron density could be calculated at the mean height, 77.5 km, only. Two 30-minute runs were made each day, May 1970 through January 1971. The data were to be processed on the CDC G-20 facility in the Electrical Engineering Building on campus. Because the paper tape reader of the G-20 would not read the special paper tape used, the processing took place on the Digital Equipment Corporation PDP-15/30 computer in the Aeronomy Laboratory. Consequently, the paper tape system constitutes part of this report.

The third method of data collection utilized the PDP-15/30 computer, interfaced to the output of the receiver via an ADC (analog to digital converter). In this way, samples of the reflected wave could be collected directly at height intervals corresponding to 1.5 km, in the height range 60-90 km. The method of collection and analysis was completely automatic, allowing regular observation without human intervention. The results were filed on magnetic tape.

The addition of the University of Illinois' partial-reflection experiment to the network of stations studying the lower ionosphere using ground-based techniques (the SOMED network), as well as the rocket program directed toward the same height region, will allow the first long-term study, on a regular diurnal basis, of the dynamics and chemistry of the mesospheric region of the earth's atmosphere.

Described in this report is the software developed at the University of Illinois for the partial-reflection facility. Included is the special system software necessary to drive the peripheral devices, the data analysis for both the paper tape and the ADC systems and the method of file organization and retrieval.

2. THEORY OF THE PARTIAL-REFLECTION EXPERIMENT

The simple theory of the partial reflection, or differential absorption, experiment has been described by Pirnat (1968) and Reynolds (1970). Consider a radio wave polarized in the extraordinary mode and propagating in a magneto-ionic medium. Over a given distance this wave suffers an absorption which is greater than that which a wave of the ordinary mode of circular polarization would suffer. This difference in absorption between the extraordinary mode and the ordinary mode of circular polarization is termed differential absorption. The differential absorption is proportional to the electron density in the region through which the radio wave propagates, and it is this differential absorption which is indirectly measured by the partial-reflection experiment and used to deduce electron density. If waves, polarized in this fashion, are propagated vertically into the ionosphere, they will be partially reflected from any sharp discontinuity in the ionization (see Figure 2.1).

Assuming small, step discontinuities in a stratified, magnetoionic medium, the Fresnel reflection coefficient at such a boundary is given by

$$R \approx \frac{dn}{2n} \tag{2.1}$$

for each polarization (Budden, 1961). If ν_m is assumed continuous across the boundary (Belrose and Burke, 1964) then R, as given by the Sen-Wyller (1960) theory for quasi-longitudinal propagation, is proportional to the electron density N. As a result, the ratio of the reflection coefficients may be obtained as a function of ν_m and N.

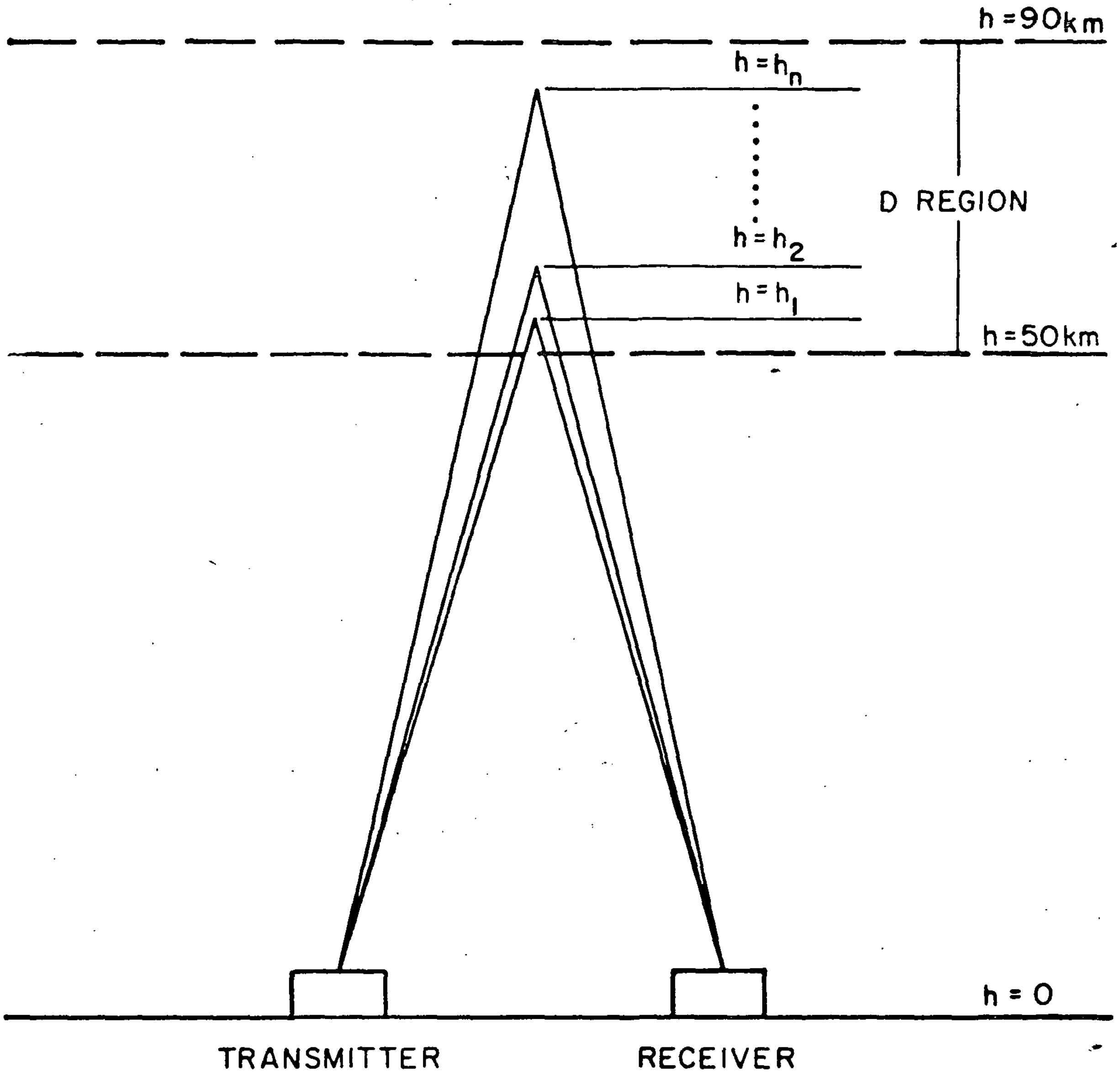

Figure 2.1 The partial-reflection model.

The amplitude of the reflected wave which suffers absorption is given by

$$A = R \exp(-2 \int_o^h K \, dh) \qquad (2.2)$$

(Belrose and Burke, 1964). Hence, the ratio of the ratios of Ax/Ao at the heights h_1 and h_2 is

$$\frac{(Ax/Ao)_2}{(Ax/Ao)_1} = \frac{(Rx/Ro)_2}{(Rx/Ro)_1} \exp \left[-2 \int_{h_1}^{h_2} (K_x/K_o) dh\right] \quad . \qquad (2.3)$$

Assuming constant differential absorption in the height range h_1-h_2, then

$$\int_{h_1}^{h_2} (K_x-K_o) \; dh = (K_x-K_o)\Delta h \quad .$$

Rearranging terms,

$$K_x-K_o = \frac{1}{2\Delta h} \ln\left\{\left(\frac{Ax/Ao}{Rx/Ro}\right)_{h_1} / \left(\frac{Ax/Ao}{Rx/Ro}\right)_{h_2}\right\} \quad . \qquad (2.4)$$

However, the absorption coefficient is a function of electron density and collision frequency alone, from which N may be obtained explicitly as

$$N = [(\frac{5\Delta he^2}{2cm\varepsilon_o \nu_m})\left\{\mathcal{C}_{5/2}[\frac{(\omega-\omega_L)}{\nu_m}]-\mathcal{C}_{5/2}[\frac{(\omega+\omega_L)}{\nu_m}]\right\}]^{-1} x[\ln\left\{\left(\frac{Ax/Ao}{Rx/Ro}\right)_{h_1} / \left(\frac{Ax/Ao}{Rx/Ro}\right)_{h_2}\right\}] \quad . \qquad (2.5)$$

To summarize, since $Rx/Ro = f_1(\omega,\nu_m)$, then

$$N = f\left\{\frac{Ax}{Ao}, \nu_m, \Delta h\right\} \quad . \qquad (2.6)$$

Given Equation (2.5), it is a simple matter to write a FORTRAN subroutine whose input data are the signal amplitude ratios and collision frequencies at two heights, along with the size of the height interval, and whose output is the electron density at the mean height. A function has been written for this calculation; it has been given the name ELDEN and a source listing may be found in the Appendix.

The Sen-Wyller (1960) quasi-longitudinal approximation, along with the assumption of a stratified ionosphere, represents only a first-order approximation to the radio wave propagation in the D region. The validity of the results may be expected to vary from height to height.

In order to calculate electron density by the method summarized above, a collision-frequency model must be assumed. The collision frequency is a function of height, since it is proportional to the number density of particles available for collisions. Many researchers (Pirnat, 1968; Reynolds, 1970; Mechtly and Smith, 1968) have utilized the fact that collision frequency varies directly proportionally with atmospheric pressure

$$\nu_M = kp \quad .$$

The constant of proportionality may be found by experiment. Although this itself is not a constant, a reasonable estimate has been made by Mechtly and Smith (1968) from rocket measurement results for a quiet sun. The atmospheric pressure varies with latitude, season and local meteorology. Using the mean atmospheric model from CIRA 1965, a first estimate of collision frequency may be made for the height interval 60-90 km. The seasonal variation of collision frequency is illustrated in Figure 2.2; these profiles are based on rocket measurements (Lodato and Mechtly, 1971).

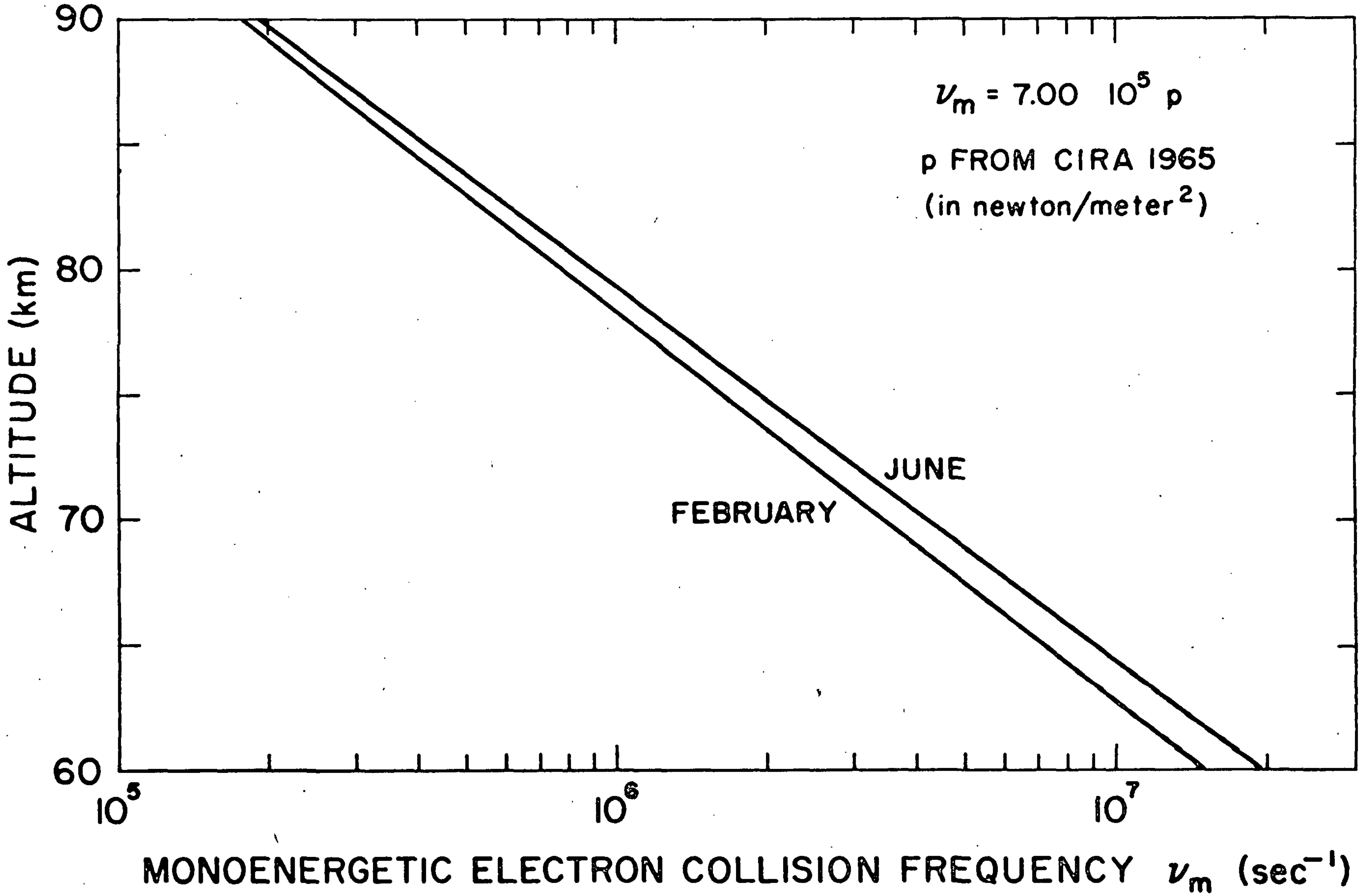

Figure 2.2 Seasonal variation in collision frequency.

Equation 2.6 indicates that the electron density at the mean of a height interval is a function of the amplitude ratio and the electron collision frequency at each end of the height interval $(Ax/Ao)_1$, $(Ax/Ao)_2$, ν_{m1}, ν_{m2}, respectively. Since the electron density is a function of four variables, it is difficult to characterize its sensitivity to any one. Such a characterization is a measure of the validity of the experimental results. It is noteworthy that since the original paper by Gardner and Pawsey (1953), few results of this nature have been published (see Thrane and Piggott, 1965; Belrose, 1969).

3. SYSTEM DESCRIPTION

The University of Illinois Aeronomy Laboratory's digital computer facility is composed of the following hardware and peripherals:

DEC PDP-15/30 central processing unit

16K 18-bit, nanosecond core memory

KSR-35 console teletype

KSR-33 console teletype

PC15 high speed paper tape reader and punch

KE15 extended arithmetic element

4 TU55 DEC tape transports

TC020 DEC tape control

KW15 real-time clock, line frequency

Special computer interface for: HP561A analog to digital converter

Figure 3.1 is a photograph of the system.

The system software includes a subset of FORTRAN IV developed for a 16K machine and incorporating all major features of the language. The machine language is MACRO, the main feature of which is a set of unified I/O statements controlling data handling. Selection of physical devices is determined at load time on the actual machine and not when a program is written.

The system is designated as a background/foreground system. The background/foreground monitor permits certain limited multiple use of the system, although an extensive amount of core is tied up in a supervisory function. A very small (1K) real-time data-logging program may be run in the foreground while the background is used for program editing and debugging. A simple change to a single user 16K system may be made when large programs are to be run.

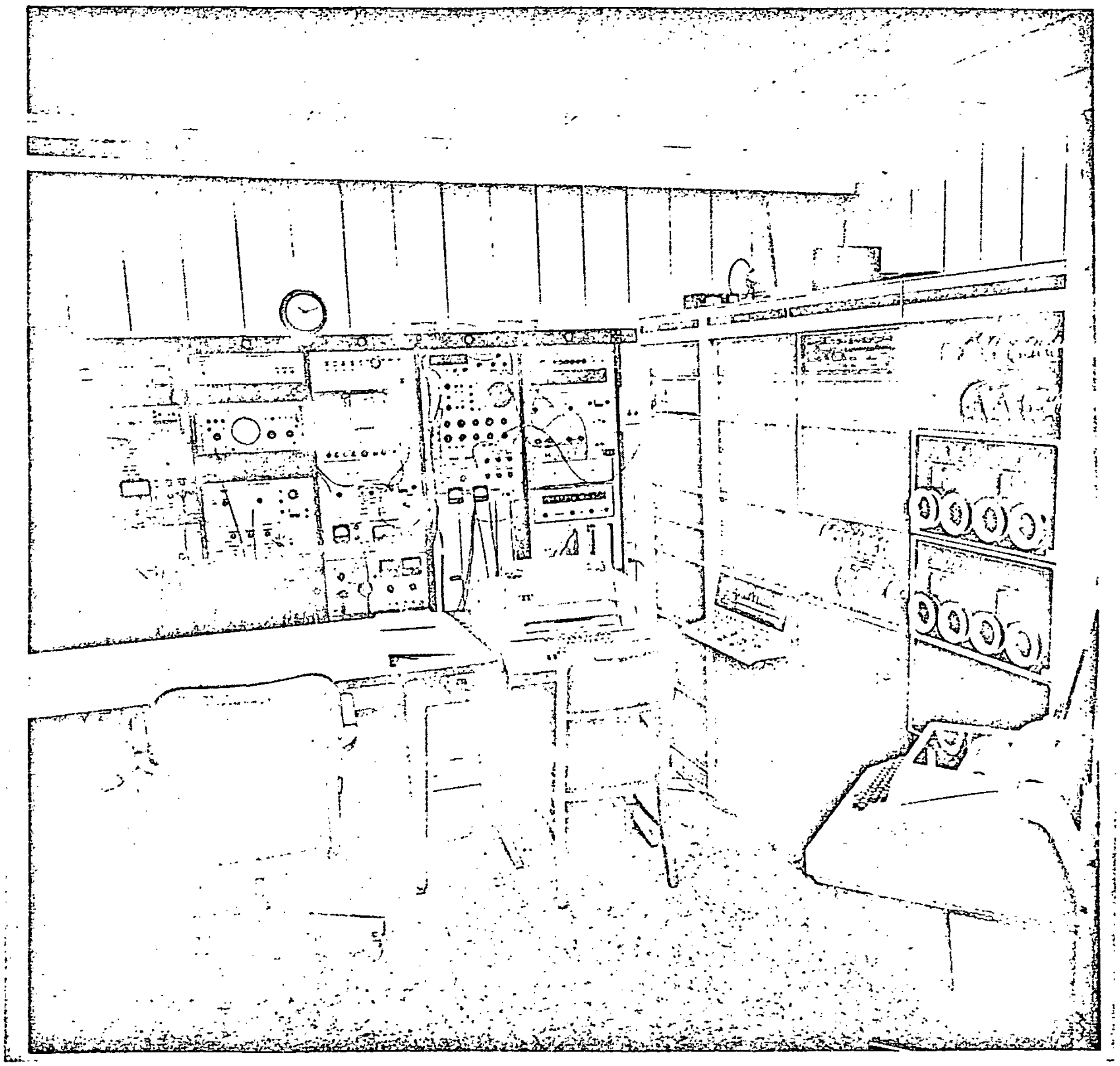

Figure 3.1 PDP 15/30 digital computer and partial reflection receiving system.

A detailed description of the computer may be found in the DEC Users Handbook Volume 1 (1970) and the Advanced Monitor Software System Programmers Manual (1970). This chapter is intended as a general description of those parts of the system related to data acquisition in a real-time environment using the analog to digital converter as well as a summary of the paper tape system which preceded it.

3.1 Timing Considerations

The sequence of events associated with the collection of one complete set of data is illustrated by Figure 3.2. A trigger cycle, with a periodicity of 2 seconds, initiates transmission of an ordinary mode radio pulse at time 0. The transmitted pulse has a width of 50 μsec and approximately 400 μsec after the peak of the pulse, signals reflected from the lower boundary of the D region are received. The form of the signal at the receiver output is illustrated by Figure 5.1. Since the timing requirements of the paper tape and the ADC system differ considerably they will be described separately. Figure 3.2 refers to the ADC system.

When the computer is used for the collection of partial-reflection data, a special word count register must be set, for each transmitter pulse, with the number of words to be converted and stored in the following 800 μsec. The actual time in which the conversions are made is determined by the encode signal illustrated in Figure 3.2. The experiment is designed to take 4 noise samples at a height corresponding to 37-41.5 km, at which no signal is expected to be present, and 21 data samples at heights corresponding from 60 to 90 km at 1.5 km intervals. Whenever the encode signal is present, as described below, the analog to digital converter will "free run" at a 100 KHz repetition rate. A converted sample will be present at the output buffer once every 10 μsec,

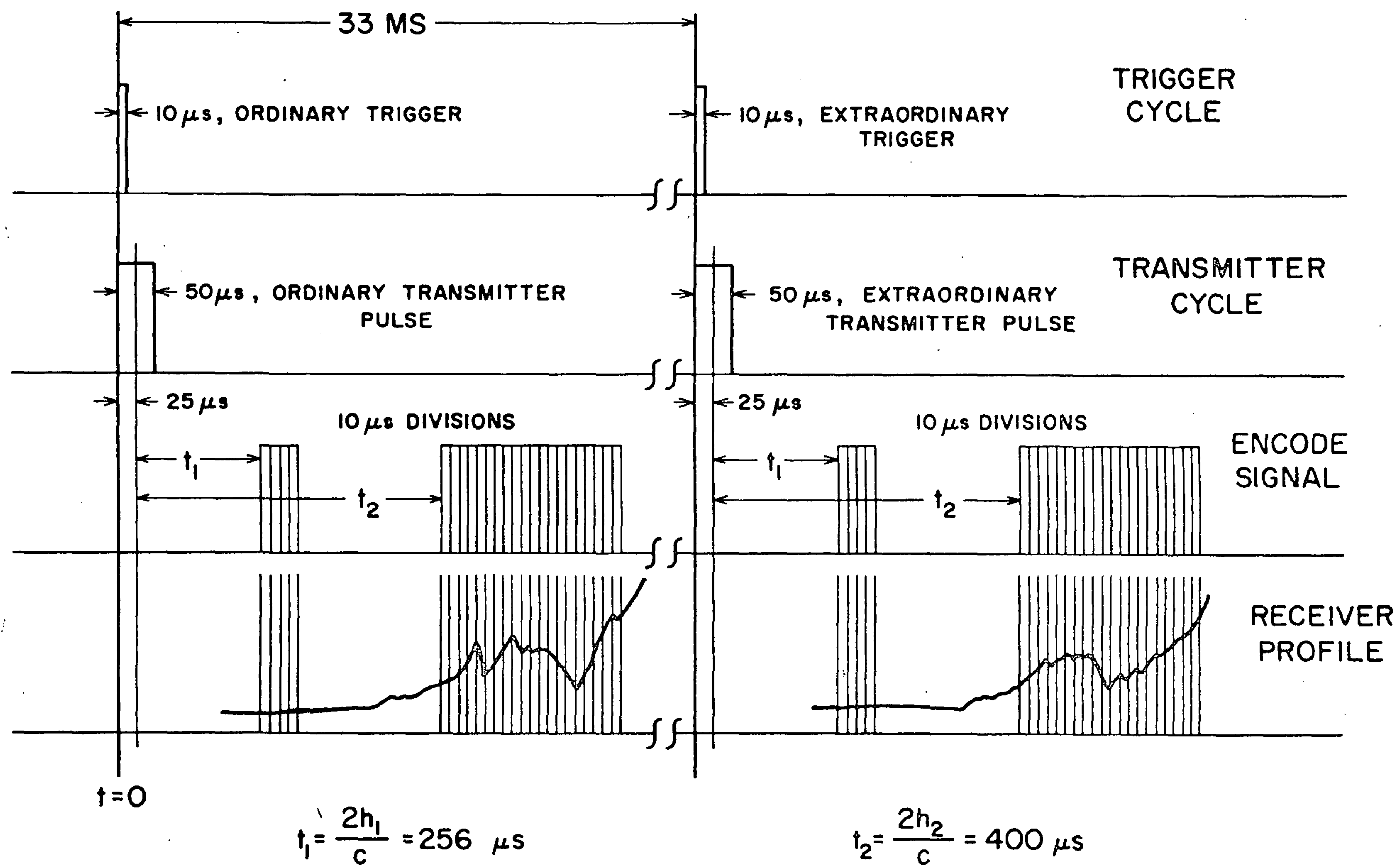

Figure 3.2 Timing diagram.

corresponding to a height interval of 1.5 km. The encode signal should be present for 40 + 5 μsec in order to obtain the 4 noise samples. 144 μsec later the encode signal should again be set for a minimum of 300 μsec; during the second interval 21 data samples are converted and transferred to the magnetic core memory of the computer. At this point the word count register indicates that transfer is complete and no further data can be stored in the core until it has been reset and the line buffer emptied.

The operation of the SOMED paper tape system (or Reynolds system, see below) is similar. However, the noise criteria are satisfied by hardware logic and only two samples of data are recorded for each transmitter pulse. The first corresponds to a height of 75 km and the second to 80 km; weak echoes are reflected from these heights and received on the ground 533 μsec after occurrence of the transmitter pulse. The encode command signal is not used nor present. This system is described in detail elsewhere.

3.2 The Reynolds System

The Reynolds (1970) system utilized a Vidar 1025019 Digital Data Acquisition System in conjunction with four sample and hold circuits and hard wired logic to record four data words every 2 seconds on oiled paper tape. An adjustable noise discriminator reproduced a previous data point if excessive noise was encountered. Sample times were controlled by an external timing device as described in the last section.

The four data points were recorded in the sequence AO75, AO80, AX75 and AX80. Each data point had a format of six paper tape frames as illustrated by Table 3.1, and coded as in Figure 3.3. Twenty-four frames were punched every 2 seconds. The first frame (the sign) and the last frame (the end of line) were redundant as data points and the four intermediate frames corresponded to decimal digits of a number with four significant figures and two decimal places.

TABLE 3.1 SOMED papertape format

A075	Sign First Frame Second Frame Third Frame Fourth Frame End-of-line
A080	Sign First Frame Second Frame Third Frame Fourth Frame End-of-line
AX75	Sign First Frame Second Frame Third Frame Fourth Frame End-of-line
AX80	Sign First Frame Second Frame Third Frame Fourth Frame End-of-line

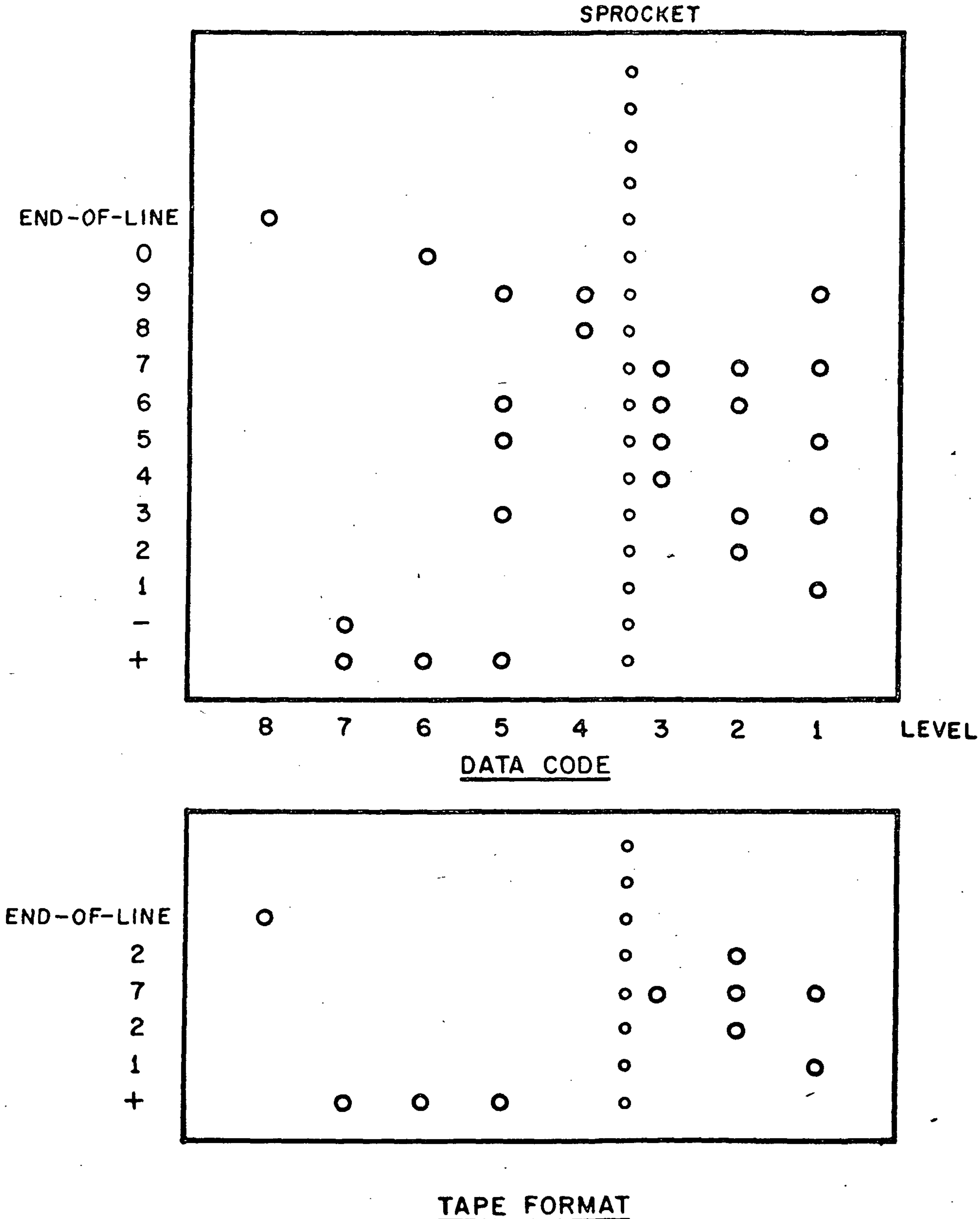

Figure 3.3 Data code and recording format.

Errors could occur during punching; typically, these included intermediate zeros or a missing end of line or sign character. The redundant data frames were utilized to keep track of the position of the tape; and the correct operation of the experiment depended on an accurate knowledge at all times of the quantity to which a particular data frame referred. Section 4.2.3 contains further details of the system.

The Reynolds system was used in the nine-month period prior to the installation of the PDP-15/30 digital computer and the paper tapes were processed by the program SOMED 1.

3.3 General Description of the Computer

3.3.1 .DAT slot assignment

Table 3.2 lists the .DAT slot assignment for the BFKM + ADC and the KM15 + ADC systems. The Device Assignment Table is a default assumption made by a particular operating system about the connection of peripherals to the central processing unit. A hardware analogy would be as follows:

Consider a set of connectors (A) attached to the input/output processing unit of the computer and a set of connectors (B) attached to peripheral devices so that each peripheral device can be connected to the IPU. A default assignment is provided which defines the normal connector combination, but if the user wishes, any pair of A connectors can be interchanged. The central processing unit sees no further than its B connectors; information is transmitted to these from the peripheral in a device independent format. Certain illegal connections are possible, such as writing to a papertape reader, and error routines signal these as they occur. The .DAT slot number refers to the IPU side of the connection and the device handler name to the peripheral connection.

TABLE 3.2 .DAT slot assignment for both the 15/20 and 15/30 incorporating the analog to digital converter

.DAT Number	BFKM + ADC		KM15 + ADC
	Background	Foreground	
-15	DTA1	None	DTA1
-14	DTA2	None	DTA2
-13	DTA2	None	DTA2
-12	TTA0	None	TTA0
-11	DTA2	None	DTA2
-10	PRA0	None	PRA0
-7	DTA0	DTA0	DTA0
-6	None	None	NON
-5	DTA2	DTA6	DTC2
-4	DTA2	DTA6	DTC2
-3	TTA0	TTA1	
-2	TTA0	TTA1	
-1	DTA0	DTA0	DTC0
1	DTA1	DTA7	DTA1
2	DTA2	DTA4	DTA4
3	DTA3	ADB	ADC
4	TTA0	TTA1	TTA0
5	PRA0	PRA0	PRA0
6	TTA0	TTA1	TTA0
7	PPA0	PPA0	PPA0
10	DTA0	DTA0	DTA0

The actual system must be described in terms of software. The .DAT slot number refers to a core location in the computer which contains a vector address. When a READ or WRITE is initiated by a program, a specific .DAT slot number is referenced. A supervisory routine, which is part of the system monitor, called the CAL Handler, recognizes the request and references the core location of the .DAT. The vector address points to the starting location of a device handler program which bears a name identical to those in the right hand columns of Table 3.2. These programs are each specifically tailored to the requirements of the devices which they serve. Chapter 4 contains a description of a device handler program called ADC, which was written for the HP 5610A analog to digital converter and assigned to .DAT slot 3 of the KM15 + ADC system tape. .DAT 3 of the BFKM + ADC system has been reserved for a B/F device handler ADB.

3.3.2 The function of the system monitor

A small section of the computer core can never be overlayed by a user program, during normal operation. This section contains a supervisory program (the resident monitor) which "listens" to the teletype. If the character ↑C is typed, at any time, the resident monitor will halt execution of every other program in core and call a larger supervisory program (the non-resident monitor) off the system tape currently associated with DEC tape unit 0 (see Figure 3.4).

The non-resident monitor can accept a large number of control commands from the teletype and can reply with a large repertoire of answers. In particular it will accept commands to re-allocate the .DAT slots described in the last section and call into core, from the system tape, a loader program which overlays it. The loader has been designed to locate any program specified by the user, which is currently on a storage device, to bring it into core, and to supervise the start of its execution.

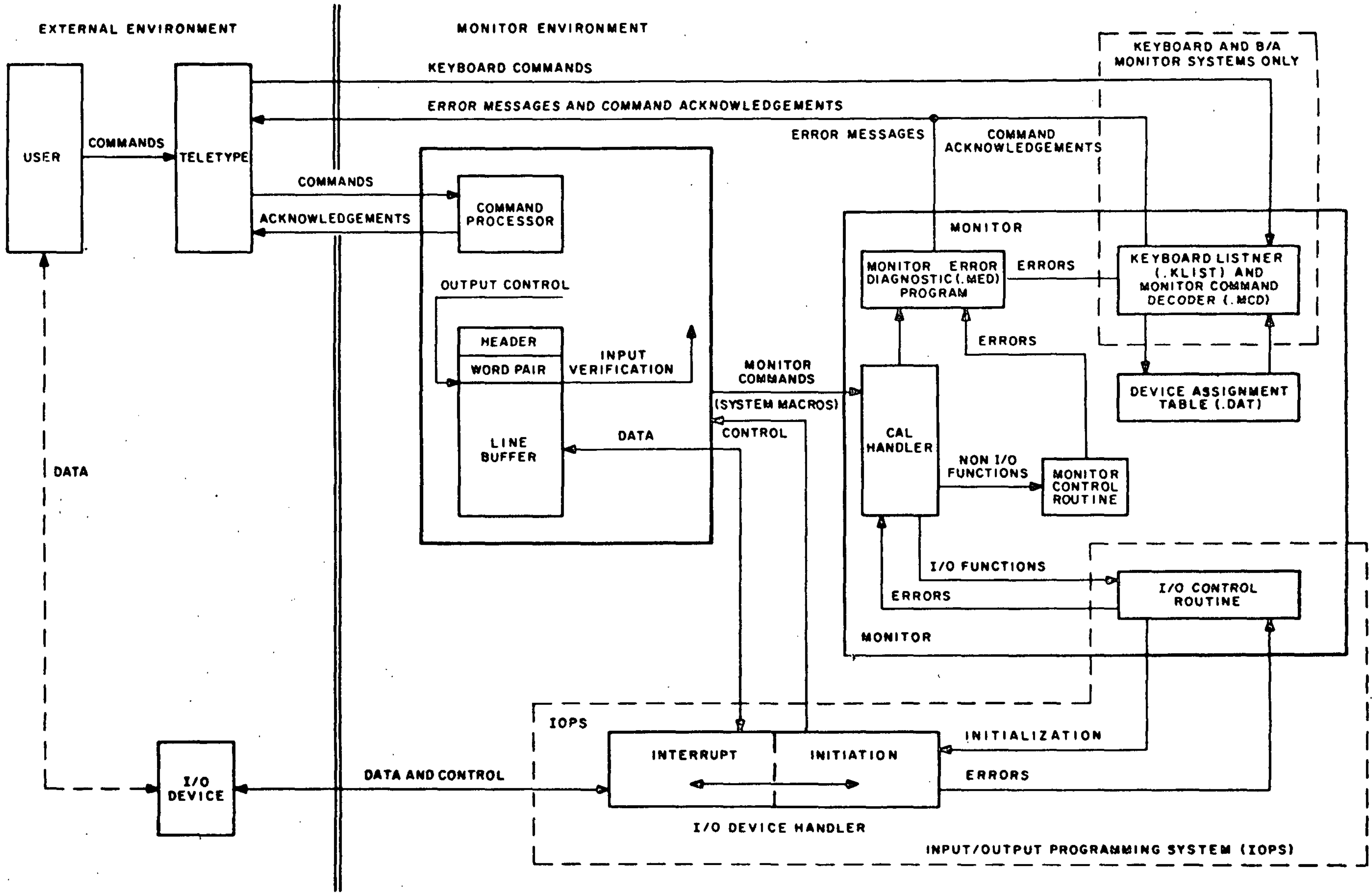

Figure 3.4 Command, control, and data flow in monitor environment (see DEC Advanced Monitor System Programmers Manual).

If the user program references a .DAT slot number, the loader will check which device handler program is currently associated with that slot and bring that program into core, also. The I/O device handler controls the operation of the peripheral device to which it is attached and passes all error conditions to the resident monitor which can write appropriate error messages on the teletype.

3.3.3 Multicycle databreak

Data transfer from the ADC to core and from core to DEC tape takes place under multicycle databreak control, the fastest I/O facility available in the current system configuration. The maximum data rate using this technique is 250 KHz. However, the actual data rate is controlled by the particular device using the facility.

When a peripheral is ready to send or receive one word of data, it raises a data channel request flag, to which the IPU responds with a data channel grant. The transfer then takes place during three special cycles which operate independently of the CPU and the program being executed. During the first cycle a word count register is incremented and tested. During the second cycle a current address register is incremented and the address to which it points is taken to be the address to or from which data are to be transferred. During the third cycle, the transfer of the data word is initiated and completed.

If the word count register is tested and found to contain a zero, a signal (Word Count Overflow) is sent to the device to terminate all future transfers. The resident monitor is also notified that this condition exists. In the case of the ADC, an automatic priority interrupt at level 0 is requested.

3.3.4 Automatic priority interrupt

Program execution can take place at 8 different levels of priority. The highest priority is called level 0 and the lowest, level 7. A higher level request will halt execution of a lower level program segment and start execution of another program segment. When the higher priority program is completed, control is returned to the point of interruption. In a real-time environment, where each peripheral operates at a different speed, a queue of API requests will develop. In this way the CPU can make use of time during which it is waiting for some action from a peripheral device.

In our system, the ADC has the highest priority - API 0 - and can interrupt any other device at any time that its word count register overflows.

When a device is granted an API break, it sends the address in core memory, with which it is associated, to the processor. This address is called the channel register and is located in absolute location 57_8 for the ADC. The channel register contains a pointer to the address of a special subroutine, part of the I/O device handler, associated with the device.

3.4 HP 5610A ADC Interface

Figure 3.5 illustrates the flow of command signals and data between the HP 5610A analog to digital converter, the interface and the input/output processor of the computer. This section is not intended as an exhaustive description, for which the reader is referred to the Computer Special Systems manual.

To effect a data transfer, the word count address (26_8) must first be loaded with the two's complement of the number of words to be transferred and the current address register (27_8) with the address (-1) in which the first word is to be stored. The interface is activated by issuing IOT WRITE (703724_8). When an external input control command is issued, conversion is initiated and a binary data word (9 bits + sign) is placed in the output register of the converter. The data ready flag is raised and causes the interface to issue ADC REQUEST. The processor sets the Data Channel Flag and the interface

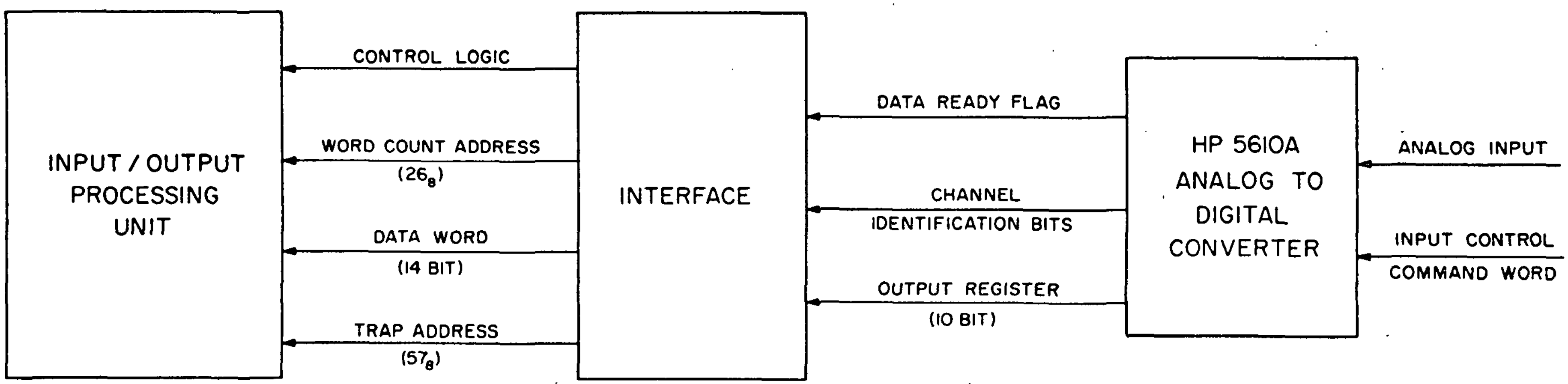

Figure 3.5 Signal flow in the HP 5610A *ADC interface.*

issues a DATA CHANNEL REQUEST. When the Data Channel Grant is received, the interface posts the word count address back to the computor and signals that a word is to be read into memory. The 18-bit data word which is read into the memory has the format illustrated by Figure 3.6. The ADC is designed to multiplex up to 8 separate analog input channels; in the present application, only channel 0 is available.

The process is repeated at 10 μsec intervals for as long as the external control command is set and until the word count register (26_8) overflows.

When the word count register overflows, I/O OFLO is set, which inhibits the operation of the 14-bit I/O Bus, and API 0 REQUEST is posted. The controller responds with API 0 GRANT and the trap address (57_8) is posted back to the computer.

API 0 is requested by either the word count overflow or a device timing error flag which is set if the condition DATA READY FLAG "OR" DCH FLAG "OR" DCH REQ "OR" DCH ENA is set.

In both cases, after processing the request, the IOT CLEAR DEVICE TIMING ERROR flag and the IOT CLEAR OVERFLOW flag must be issued.

3.4.1 External control command

Five input lines are provided for controlling the operation of the HP 5610A analog to digital converter. They are called respectively HP bit 6, HP bit 5, HP bit 4, encode and ESP. For the Uniplexer configuration which is currently provided, only two modes of operation are available:

1. Free run internal sequence:

Bit 6	Bit 5	Bit 4	Encode	ESP
1	1	1	0	0

0-7			0 or 4			0 or 1			0-7			0-7			0-7		
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Channel ID				Not Used				Converted analog data									

Figure 3.6 The 18-bit data word transmitted to memory from the ADC via the interface.

2. Internal Sequence:

Bit 6	Bit 5	Bit 4	Encode	ESP
1	0	1	1	0

For operation in both these modes bits 4 and 6 and the ESP Line do not change and in the free run internal sequence mode the encode signal is ignored. In the free run internal sequence mode the converter samples channel 0 every 10 μs. In the internal sequence mode the converter will sample channel 0 each time the encode switch is set.

In order to perform the operation described in this paper, the free run internal sequence mode was used. HP bit 5 was set to a 1 (positive logic) each time that a group of data samples were to be taken.

4. SOFTWARE

This chapter includes a description of all programs written for the partial-reflection facility at the University of Illinois. Since these programs are inoperable without an explanation of how to call them into core, the first part of the chapter details the file organization adopted at this time by the users of the machine.

In general, each format of a program is stored on a different magnetic tape spool. There are two tape spools assigned to the user programs associated with the partial-reflection experiment and two other magnetic tape spools which contain related system programs. The magnetic tape spools will in the future be referred to as DEC tapes.

The user programs are stored in two forms. The first is the source language, in which the programs were written. In the case of the programs described below the source language was either MACRO or FORTRAN IV. The source listings are all stored on the DEC tape labeled Standby Files. Additional program listings may be obtained from the standby files by use of either the editor or the peripheral interchange program. The machine code level programs are stored on the Partial-Reflection tape.

The Partial-Reflection tape contains two sections, one of which is for the mainstream program; the other section, known as the user library, contains all subroutines which the mainstream program may call. When a program is loaded into core from the partial-reflection tape, it is not necessary to specify any of the subroutines associated with the mainstream segment; the loader processes all subroutine calls that it finds on the mainstream segment and then loads, from the library, all files with names corresponding to those subroutine calls.

The library file may be thought of as a superfile whose components are themselves files and each of which contain one subroutine. Mainstream programs which were written in FORTRAN may call macro-level subroutines and vice versa. For a complete list of subroutines included in the partial-reflection library file see Table 4.1. For a list of the mainstream programs on the partial-reflection tape see Table 4.2.

User programs would be inoperable if it weren't for a set of system programs whose purpose is to manipulate the user programs and to supervise their operation. The editor program, which aids the programmer in writing his user programs, has already been mentioned above. It allows him to open a file on a DEC tape and to write a program into it from the teletype, while performing corrections to the program in a convenient manner. The peripheral interchange program (PIP) can be used to transfer files from one DEC tape to another or from one peripheral device to another. The UPDATE program is used to build library files. The LINKING LOADER loads user programs from the users DEC tape into core, and performs the linkages between mainstream programs and subroutines. Detailed descriptions of these systems programs may be found in the DEC PDP-15 Utility Programs manuals. There are also system programs whose purpose is to control the flow of data from peripheral devices into and out of core. These are generally called I/O Handlers and they are stored in the system library. The system programs described above comprise part of the system tape. Since two operating systems have been written for the analog to digital converter there are two system tapes.

The system tapes are identical in most respects. The first, for the 15/20, incorporates an I/O Handler for the analog to digital converter called ADC. The second, for the 15/30, incorporates an I/O Handler called ADB. Both of

TABLE 4.1 Library file listing for .LIBR5

PROGRAM NAME	PROGRAM SIZE
EMED2	252
AVERA2	103
DUMPR	562
DREAD	425
TTYPL	1222
EHIST	171
AVERAG	101
EMED	245
LREC	56
DECOD	263
ELDEN	1241
RDCHEK	1412
PTCHAR	231
SET	21

TABLE 4.2 Mainstream binary files on the partial-reflection tape

Directory Listing
636 Free Blks
15 User Files
10 System Blks

.LIBR	BIN	4	21
SOMEDA	XCT	37	1
SOMED1	BIN	2	13
SOMED2	BIN	1	6
DLOG2	BIN	5	2
WLOG1	BIN	3	3
WLOG2	BIN	36	2
DLOG3	BIN	56	3
DLOG4	BIN	22	3
NCALC	BIN	50	3
.LIBR5	BIN	112	43
PROC2	BIN	104	23
SOMEDA	XCU	17	61

these tapes differ from other system tapes available in that they make provision for an interrupt at API 0 from the analog to digital converter which is a non-standard peripheral device. Special programs are available for generating new system tapes. The system tapes also contain the nonresident portion of the monitor as described in Chapter 3.

4.1 Instructions for Loading User Programs

Several of the programs included in the partial-reflection program package were designed to be run in the foreground of the 15/30 system. In order to load and execute these programs, the following steps must be carried out.

1. Locate the system DEC tape labeled BFKM + ADC and place on DEC tape unit 0. Locate the partial-reflection tape and place on DEC tape unit 6. Make sure that both units are "write lock". Turn on both teletypes. Set the address switches on the console to 37646_8 and press stop, reset, start. DEC tape unit 0 should spin and a $ should be typed on teletype KSR 33; this is the foreground teletype.
2. If data are to be collected, place the data file DEC tape on DEC tape unit 4. Make sure that it is on "write enable".
3. At the foreground teletype, type the command MPOFF. This command enables the background to use input and output.
4. Type GLOAD after which DEC tape unit 0 should spin. When an angle bracket is echoed on the foreground teletype, type ← program name and follow it with the ALTMODE key.
5. DEC tape unit 6 should spin and the user program should be loaded; control should then be transferred to the background teletype which echoes $. The user program is now executing.

Many of the programs in the partial-reflection package are designed for operation in the 15/20 system. In which case they should be loaded as follows.

1. If the program references the analog to digital converter, the system tape KM15 + ADC must be used.
2. Follow standard operating procedures to load the 15/20 monitor.
3. Place the data file tape on DEC tape unit 4 and the partial-reflection tape on DEC tape unit 2.
4. Type GLOAD and when the teletype echoes an angle bracket, type ← program name and then press the ALTMODE key.
5. DEC tape units 2 and 0 will spin and the program should begin execution.

All programs which reference the data files but not the analog to digital converter may use any form of the 15/20 system tape. However, if the above system tape is not used the data tape must be assigned to .DAT 2. The partial-reflection tape should be placed on DEC tape unit 2. The programs can then be loaded as above.

Certain commonly used computer commands and replies will be encountered during execution of the programs in the partial-reflection package. Some of the most frequently encountered are listed below.

IOPS4 This reply is typed on the teletype when some device referenced by the program being executed is not ready. Check that all devices likely to be used are in the correct operating condition. When the error has been rectified, type ↑R and program loading or execution will continue. In the 15/30 system a device not ready message serves the same function.

↑P Certain of the user programs require this command in order to restart program execution from the beginning.

4.2 A Program for Processing Papertapes

4.2.1 SOMED 1

Program type - Mainstream

Source language - FORTRAN

Subroutine/function calls - ELDEN, RDCHEK, DECOD, AVERAG, EMED, PTCHAR, SET, LREC, EHIST

Program requirements - 15/20 monitor tape, KSR35, one DEC tape unit, papertape reader, Load using the linking loader

Listed in - Appendix

Summary

This program analyzes partial-reflection data from the heights of 75 and 80 km recorded on paper tape using the system described in Chapter 2, developed by D. A. Reynolds (1970). The program contains four COMMON, 900-word single dimension arrays. These are sufficient to store the ordinary and extraordinary mode receiver amplitudes for both heights for a 30-minute run. The raw data are recorded in non-standard code on paper tape and special subroutines were necessary to transfer it into core. Paper tape punch and reader errors necessitate a subroutine for checking the raw data. The output is both average and median values of the recorded amplitudes as well as amplitude ratios and electron density at three-minute intervals.

4.2.2 ELDEN

Program type - real function

Source language - FORTRAN

Called by - SOMED 1, PROC2

Size - 1241_8

Program requirements - the function is stored on the partial-reflection tape in library file called .LIBR5. It requires three input parameters. The first is a two element single dimensional array of the amplitude ratios and a three-element single-dimension array of the collision frequencies at the lower and upper bounds of the height interval. The third parameter is the size of the height interval in kilometers.

Listed in - Appendix

Call statement = ELDEN (AXBYAO, GNU, DELTAH)

Summary

The method used to derive electron density in the differential absorption experiment utilizes the ratio of extraordinary to ordinary wave amplitudes and collision frequency at both ends of the height interval, as described in Chapter 2. Although the function is complex it is possible to write Equation (2.6) where the variable names are listed below. The semiconductor integrals have been approximated by Burke and Hara (1963) and their formulas are listed in Table 4.3.

This form of the expression for obtaining electron density is ideal for programming as a FORTRAN function. The program assumes a radio frequency of 2.66 MHz and values of the constants given in the List of Symbols.

Symbol List

GNU = collision frequency = ν_m

O = $(\omega+\omega_L)/\nu_m$

X = $(\omega-\omega_L)/\nu_m$

CTO = $\mathcal{C}_{3/2}[(\omega+\omega_L)/\nu_m]$

CTX = $\mathcal{C}_{3/2}[(\omega-\omega_L)/\nu_m]$

CFO = $\mathcal{C}_{5/2}[(\omega+\omega_L)/\nu_m]$

TABLE 4.3 The approximate equations for the $\mathcal{F}$ integrals used in function ELDEN

$$\mathcal{F}_{3/2}(X) = (X^4+a_3X^3+a_2X^2+a_1X+a_o)/(X^6+b_5X^5+b_4X^4+b_3X^3+b_2X^2+b_1X+b_o)$$

where

$a_o = 2.3983474 \times 10^{-2}$

$a_1 = 1.1287513 \times 10$

$a_2 = 1.1394160 \times 10^2$

$a_3 = 2.4653115 \times 10$

$b_o = 1.8064128 \times 10^{-2}$

$b_1 = 9.3877372$

$b_2 = 1.4921254 \times 10^2$

$b_3 = 2.8958085 \times 10^2$

$b_4 = 1.2049512 \times 10^2$

$b_5 = 2.4656819 \times 10$

$$\mathcal{F}_{5/2}(X) = (X^3+a_2X^2+a_1X+a_o)/(X^5+b_4X^4+b_3X^3+b_2X^2+b_1X+b_o)$$

where

$a_o = 1.1630641$

$a_1 = 1.6901002 \times 10$

$a_2 = 6.6945939$

$b_o = 4.3605732$

$b_1 = 6.4093464 \times 10$

$b_2 = 6.3920505 \times 10$

$b_3 = 3.5355257 \times 10$

$b_4 = 6.6314497$

CFX = $\mathscr{F}_{5/2}[(\omega-\omega_L)/\nu_m]$

RX = R_x

RO = R_o

RXBYRO = R_x/R_o

AXBYAO = A_x/A_o

RATIO = $(A_x/A_o)/(R_x/R_o)$

DELTAH = Δh

AVGNU = ν_m at the center of $\Delta h = (\nu_m)_{\Delta h}$

FO = $(5e^2)/m\varepsilon_o)\,\mathscr{F}_{5/2}[(\omega+\omega_L)/(\nu_m)_{\Delta h}]/4c(\nu_m)_{\Delta h}$

FX = $5(e^2/m\varepsilon_o)\;\mathscr{F}_{5/2}[(\omega-\omega_L)/(\nu_m)_{\Delta h}]/4c(\nu_m)_{\Delta h}$

FD = FX - FO

ELDEN = Electron Density = $\dfrac{\ln\{[(A_x/A_o)/(R_x/R_o)]_{h_1}/[(A_x/A_o)/(R_x/R_o)]_{h_2}\}}{2\Delta h FD}$

4.2.3 RDCHEK

Program type - real subroutine

Source language - FORTRAN

Called by - SOMED 1

Size - 1412_8

Program requirements - teletype

Listed in - Appendix

Call statement - RDCHEK (INR,X,J,INWDCT,IALRED,IABORT)

Summary

The input array for this subroutine is raw data read from paper tape, with the format described in Chapter 3. Since the paper tape punch and the paper tape reader are prone to error, certain decisions must be made in deciphering the coded data. Redundant characters, which precede and succeed each data point, are used both to count the data points and to aid in their retrieval. Since four different quantities are measured in succession, it is particularly important that the exact position of the tape is known at all times. If the program is unable to decipher the data, it presents the problem on the teletype for the operator's intervention. The operator is asked to reconstruct the estimated number of data points which have been lost. The program may be restarted from the beginning by replying with a carriage return.

Symbol List

INR - one dimensional integer array of input data

X - single dimensional real output array of ordered raw data with all redundant characters removed

J - on output this contains the number of data points recognized by the subroutine

INWDCT - the number of elements in array INR

IALRED - assigned FORTRAN statement number to which control is passed if thirty blank frames of paper tape have been read

IABORT - assigned FORTRAN statement number to which control is passed if an irrecoverable error occurs

4.2.4 DECOD

Program type - real function

Source language - FORTRAN

Size - 263_8

Listed in - Appendix

Call statement - DECOD(X,I)

Summary

This function accepts an array X of raw data (as described in Chapter 3) minus redundant characters and decodes locations I + 1 through I + 4. From these it constructs one real number.

Symbol List

X - one dimensional real input array

I - integer specifying element of array X

4.2.5 AVERAG

Program type - real function

Source language - FORTRAN

Called by - SOMED 1

Size - 101_8

Listed in - Appendix

Call statement - AVERAG (A,NMIN,NMAX,SUM)

AVERA2 (A,LEL,NMIN,NMAX,SUM)

Summary

AVERAG averages a one-dimensional array A for terms NMAX through NMIN. On exit SUM is the sum of the elements averaged. AVERA2 performs the same function but A is a two-dimensional array and LEL specifies the column.

Symbol List

A - Real input array of data to be averaged (see above)

LEL - Integer specifying array column

NMIN - lower bound of array

NMAX - upper bound of array

SUM - sum of elements which were averaged

4.2.6 EMED

Program type - real function

Source language - FORTRAN

Called by - SOMED 1

Size - 245_8

Listed in - Appendix

Call statement - EMED(F,NMIN,LCOUNT,AVE)

EMED2(F,LEL,NMIN,LCOUNT,AVE)

Summary

The output of this program is the median of real positive array F.

Symbol List

F - one-dimensional real input array

NMIN - lower bound of array F

LCOUNT - number of elements of array F to be considered

AVE - any estimate of the median, must be a positive real number

4.2.7 PTCHAR

Program type - MACRO subroutine

Source language - MACRO

Called by - SET,LREC

Subroutine function calls - System macro calls, to paper tape reader handler

Size - 231_8

Program requirements - DAT 5 assigned to the paper tape handler

Listed in - Appendix

Call statement - JMS indirect

The program must be called twice. The first call is to PTON, which initializes the device. The main call is to PTCHAR, from which a frame of the paper tape which has been read is passed as an integer parameter.

Summary

Paper tape is read alternately to two 252_{10} ring buffers. The next sequential data frame is passed each time the subroutine is called. Updating of ring buffers is performed automatically.

Symbol List

POINT - contains address for the next sequential data frame to be passed to the main program

BUF1 - ring buffer

BUF2 - ring buffer

SEVN - flags end of ring buffer

4.2.8 SET

Program type - subroutine

Source language - MACRO

Called by - SOMED 1

Subroutine/function calls - PTCHAR,PTON

Size - 21_8

Listed in - Appendix

Call statement - SET(112,1)

Summary

SET starts the paper tape reader and locates the first occurrence of character 112. On exit the first ring buffer of PTCHAR is full and points to the first character following 112. 112_{10} is normally the first nonblank character on the paper tape.

4.2.9 LREC

Program type - subroutine

Source language - FORTRAN

Called by - SOMED 1

Subroutine/function calls - PTCHAR

Size - 56_8

Listed in - Appendix

Call statement - LREC(KK,LC,J,NMIN,NMAX)

Summary

LREC places a specified number of paper tape frames in an integer array. Transmission halts at character LC or after a given number of characters. J is returned as number of characters that were read into the integer array.

Symbol List

KK - integer input array

LC - integer numeral which causes paper tape reader to halt if encountered

J - output as number of data frames passed to array KK

NMIN - lower bound of array KK

NMAX - upper bound of array KK

4.2.10 EHIST

Program type - real subroutine

Source language - FORTRAN

Called by - SOMED 2

Size - 171_8

Listed in - Appendix

Call statement - EHIST(WO, WI,NINT,A,NU,FR)

Summary

This program calculates a frequency distribution from a set of real input data. It is used by later versions of the SOMED 1 program, in conjunction with a graph plotting routine, for analysis of partial-reflection amplitude data.

Symbol List

WO - lower bound of amplitude window

WI - upper bound of amplitude window

NINT - number of intervals within window

A - real input array

NU - size of array A

FR - real output array, size NINT, of the amplitude frequency distribution

4.3 I/O Handlers and Service Routines for the Analog to Digital Converter

Every peripheral attached to the computer has a program associated with it which performs input and/or output operations. This saves the programmer the task of writing separate I/O routines each time he wishes to transfer data between the CPU and a peripheral. For instance, a standard set of call statements are provided for communication between the two programs. The programmer can request a .READ at macro level. The device to be read is specified by its device number or .DAT slot number, in the parameter list of the .READ statement.

Table 3.3 is a listing of the .DAT slot default assignment. This can be changed by a typed command at any time. For instance, if a program asks for a .READ to unit -15, the computer will look up which device is attached to this unit. It is normally DTA2, DEC tape unit 2, but it could be changed by the operator to PRA0, the paper tape reader. The .DAT assignment table is the software equivalent of unplugging the DEC tape connection and plugging in the paper tape reader connection in its place.

The four character name which follows the .DAT slot number is the name of a program whose sole purpose is to deal with input and output to the device specified. This program is called an I/O handler.

An I/O service routine is identical to a handler, except that it is program dedicated and must be called by a standard subroutine call. Whereas I/O handler programs are incorporated in the system tape, as part of the system I/O library, I/O service routines are part of the user's program library. The I/O service routine described in this section is appended to the end of the programs from which it is called. The main advantage of an I/O service routine is that it requires less core. For this reason I/O service routines were developed to operate in the foreground of the 15/30 system thus freeing the maximum amount of core for use in the background.

The analog to digital converter utilizes the API 0 and the multicycle data break facility of the 15/30 computer, as described in Chapter 3. The trap address was hardware assigned to channel register 17 and the following IOTs were provided with the device.

703724_8 - initialize interface. It must be issued before data can be transferred from the converter to core.

703701_8 - skip on word count register overflow. The word count is stored in 26_8, and interrupts at API 0.

703704_8 - clear word count overflow.

703721_8 - skip on device timing error flag. This flag is set when a hardware fault develops.

703744_8 - clear the device timing error flag.

703702_8 - read the status of the device flags to the accumulator. If bit 15 is set, the overflow flag is set. If bit 16 is set, the device timing error flag is set.

Input to the analog to digital converter procedes as follows. The two's complement of the number of words to be sampled is stored in the word count register, location 26_8. At the same time the address to which the first word is to be transferred in core is stored in location 27_8, the current address register. The "Initialize Interface" IOT is requested and executed and program control is relinquished to either a wait loop or to the background. Each time that an external encode command is given to the analog to digital converter, the contents of the output buffer are transferred to the current address plus 1. This occurs without interruption of the program segment being executed. At the same time the word count register and the current address register are incremented by 1. Finally, the word count register reaches 0 and the word count overflow flag is set. API 0 is requested, the interface is effectively turned off, and if no other API 0 was being processed, control is transferred via JMS* ADINT, located in address 57_8, to the interrupt service routine.

The word count and current address registers, as well as the interface, must be reset before more data can be read from the converter.

If certain logic conditions exist in the interface the device timing error flag is set, indicating a hardware fault. An API 0 request is issued and control is transferred to the interrupt service routine, as before. Consequently the interrupt service routine must test for the source of the interrupt, by determining the status of both flags. If a device timing error has occurred, a message is typed at the control teletype and recovery procedures, unique to each handler, must be performed by the operator. Both interrupt flags are cleared before the final exit from the interrupt service routine.

In the following sections the I/O handler and the I/O service routine developed for the analog to digital converter are described in detail.

4.3.1 ADC.

Program type - I/O handler

Source language - MACRO

Called by - DLOG2

Subroutine/function call - IOPS4, IOPS6

Size - 135_8

Program requirements - analog to digital converter, control teletype, API 0, 15/20 monitor, external encode commands

Listed in - Appendix

Binary file stored in - .LIBR system library file on system tape KM15+ADC

Call statements - system MACROs

Summary

The form of I/O handler ADC. is consistent with the advanced software system manual. The CAL handler interprets all system macro calls made by the user program and passes the CAL address, in the accumulator, to the first word of the handler. The parameter list of the system macro contains an argument which permits dispatch to the correct portion of the handler. The following system macros are legal: .INIT, .READ, .WAIT, and .CLOSE.; macros other than these are either ignored or generate IOPS 6. When .INIT is called, the program segment ADINIT is executed. In this program segment a special monitor subroutine is called which sets up the address of the I/O interrupt handler in the channel register associated with the device. A standard buffer size is returned to the

MACRO parameter list and the handler busy flag is cleared. .READ must be issued every time that a set of samples is to be transferred to core. The action of this MACRO is to execute program segment ADREAD. The handler busy flag is tested and if it is already set the READ is ignored, otherwise the busy flag is set and the word count and current address registers are set up. Interface initialize is requested before the handler finally debreaks and returns control to the main program. All CAL handler requests are handled at API level 4.

The handler waits for an interrupt signifying that data have been transferred from the converter and that the word count register has overflowed. While waiting, the mainstream program can execute a loop or a .WAIT macro may be used to perform the same function. Finally the overflow flag is set and an interrupt at API level 0 is requested. Control is passed to the I/O interrupt service routine whose address, ADINT was set up in the channel register, or trap address, during the initialization routine. The program checks that the source of interrupt was the overflow flag, in which case the busy flag is cleared and a debreak and exit is performed to the interrupted address. If the device timing error flag is set, an irrecoverable IOPS 4 error occurs and a message is printed at the control teletype via the resident monitor IOPS routine. The .CLOSE macro serves merely to clear the handler busy flag.

4.3.2 Summary of 15/20, analog to digital converter handler calls

A. System

a. Advanced monitor software system

b. System library, LIBR

c. Multicycle data break

d. Word count in location 26_8, line buffer address in 27_8

e. API 0 and channel register 17_8

B. Functions

1. .INIT

   a. Return standard buffer size 30_{10}

   b. .SETUP API channel register 17_8

2. .DLETE ignore

   .RENAM ignore

   .FSTAT ignore

3. .SEEK ignore

4. .ENTER illegal function

5. .CLEAR illegal function

6. .CLOSE ignore

7. .MTAPE ignore

10. .READ

   a. set busy flag

   b. busy loop if busy flag already set

   c. set word count and buffer address

   d. issue IOT to initialize interface

   e. free run mode: 10 μsec wait, or wait until encode command, from time of execution of IOT

11. .WRITE illegal

12. .WAIT, .WAITR

   a. check I/O underway

   (1) busy: return to CAL address (.WAIT) or to address in CAL+2 (.WAITR)

   (2) nonbusy: return to CAL+2 or CAL+3 (.WAITR)

13. .TRAN illegal function

C. Legal data modes

1. Dump mode (no header word pair). Stores data in two's complement form, nine bits plus sign, right hand justified, one per word, in line buffer.

D. Program size - 164_8

E. Unrecoverable errors

1. Illegal function IOPS6

2. Device not ready IOPS4

F. Processing of interrupt

1. An interrupt at API 0 occurs if either the word count overflows or a device timing error occurs.

2. The busy flag is cleared.

3. If the overflow flag is set then it is cleared and the normal exit executed.

4. If the device timing error flag is set it is cleared before a call to IOPS 4, as an irrecoverable error.

4.3.3 <u>I/O service subroutine for 15/30 foreground</u>

Program type - dedicated subroutine

Source language - MACRO

Called by - DLOG4

Size - 50_8

Program requirements - 15/30 BFKM + ADC system tape, background or foreground, and an HP5610A analog to digital converter

Listed in - Appendix

Call statement - JMS ADINIT

Summary

The design philosophy behind the development of this I/O service routine, was for a program to input data from the analog to digital converter and output it to DEC tape while using the smallest amount of core possible. The program should be suitable for foreground use while leaving sufficient core for useful operation of the background in a 16K environment. The result was a dedicated handler, or I/O service routine for the analog to digital converter which used only 50_8 core locations, in contrast with the I/O handler ADB. which uses 316_8 core locations. Further savings of core were achieved by making the subroutine call as simple as possible and appending it to the main program, thus avoiding linkage addresses and complex parameter passing procedures. When the call statement listed above is issued in the main program, the program segment ADINIT of the I/O service routine calls the monitor .SETUP routine and establishes a JMS* ADINT in channel register 17, to the interrupt service routine. The word count register [26_8] and the current address register [(27_8)] are set up and the "Initialize Interface" IOT issued before return to the main program, which is then free to relinquish control to the background.

The interrupt service routine processes all interrupts as real-time requests, and issues a new "Interface Initialize" IOT, as follows.

When an interrupt, at API 0, is passed to the interrupt service routine the word count overflow flag and the device timing error flag are checked. In the case of a word count overflow control is passed to the real-time processor and the real-time subroutine is executed at API level 5. It is up to this subroutine to reset the word count and the buffer address. If the device timing error flag was set, control is passed via the real-time processor to a second real time subroutine which issues a device not ready message at the control teletype. In both cases the interrupt flag is cleared and control is returned to the interrupted address. A device timing error is irrecoverable. This subroutine may be better understood in conjunction with the main program DLOG4 described elsewhere.

4.4 Data Logging Programs

The programs described in this section require different I/O handlers or I/O service routines. Yet certain features are common to them both. The object is to input and store data from the analog to digital converter in a real-time environment. The form in which data are stored on DEC tape is similar in both cases and the same minimum time requirements apply.

Data are input from the analog to digital converter in 26_{10} word groups, timed externally through the analog to digital converter as described in Chapter 3. Data are output periodically to DEC tape in file oriented, DUMP mode. The program automatically assigns the file DATAFnDAT, where n = 1, 2, etc.; it cannot be over written without a positive command from the operator. If a file of a given file name is present, the operator may choose to increment n and thus create a new file.

The sampled data from the analog to digital converter are composed of 9 bits plus sign. On input the least significant bit is removed and the remaining 9 bits are packed into an 18-bit computer word. An identification sequence number is attached to every group of 26_{10} data words which are packed into 14, 18-bit, computer words. (See Table 4.4.) The 14 word groups, or sample sets, are stored sequentially in a 252_{10} ring buffer. One of the programs described in this section uses two ring buffers, which increases the throughput rate since data may be written into one while the other is being output to DEC tape, though it increases the core requirements of the program considerably.

TABLE 4.4 Format of packed data (one data set)

18 bit computer word		
1	IDENTIFICATION NUMBER	
2	Noise 1	Noise 2
3	Noise 3	Noise 4
4	Data 1	Data 2
5	Data 3	Data 4
6	Data 5	Data 6
7	Data 7	Data 8
8	Data 9	Data 10
9	Data 11	Data 12
9	Data 13	Data 14
11	Data 15	Data 16
12	Data 17	Data 18
13	Data 19	Data 20
14	Data 21	0

The main limitation to the throughput rate of the system is the speed at which a 252 word data buffer can be output to DEC tape. DEC tape is organized in 256_{10} word blocks. In file oriented mode every fifth block is written. Each call for a write to DEC tape causes one DEC tape block to be output. The programs described in this section align the DEC tape blocks by initially writing two dummy blocks on to tape. In this way a minimum time delay is incurred each time a block is written. The DEC tape spins after every 18 sets of data are input from the converter. At the standard data collection rate this is once every 18 seconds.

↑P will close the current file and restart the program whenever typed. When a file is closed, a coded number is added as the last data point and this signals the retrieval program (DUMPR) that an end of file has been reached.

Both the programs include the TIMER routine. With this facility the length of a data run may be automatically controlled by the computer. When bit 0 and 1 of the data register on the computer console are set, the remaining data switches are assumed to indicate the maximum number of data sets to be recorded. The switches must be set to a binary number larger than the nearest decimal integer multiple of 18 which is greater than the number of data sets required. The data switches will then be interrogated every time data are output to DEC tape until the required condition is meet. At this point the ↑P routine is automatically entered.

4.4.1 DLOG2

Program type - Mainstream

Source language - MACRO

Subroutine/function calls - system MACRO calls to the 15/20 ADC handler

Size - 1365_8

Program requirements - 15/20 KM15+ADC system tape, control teletype, DEC tape unit 4

Listed in - Appendix

Summary

This program operates in the 15/20 environment and utilizes the design consideration discussed in Section 4.4. Two ring buffers are used for input of data from the converter and the TIMER option is included. A thirty-minute data run may be obtained automatically by setting the data switches on the computer console to 603420_8.

The program uses the handler ADC which is assigned by default to .DAT 3. Output is to DEC tape on DEC tape unit 4 which is assigned, by default to .DAT 2.

4.4.2 DLOG4

Program type - Mainstream

Source language - MACRO

Subroutine/function calls - B/F I/O service routine

Size - 1234_8

Program requirements - BFKM + ADC system tape, control teletype, DEC tape unit 4, default assignments for foreground use

Listed in - Appendix

Summary

This program follows the design considerations discussed in Section 4.4 and uses one ring buffer and the I/O service routine to keep core requirements to a minimum. The TIMER option is present.

Before loading, the foreground commands MPOFF and SHARE should be given at the foreground teletype if the operator desires to use the background facility to the full. Data input from the converter to core is inhibited during data output from core to the data tape. This limitation is not relevant at the normal throughput rate, but as the throughput is increased a point will be reached where data are lost. When this program is executed there is sufficient core available to run the majority of the utility programs in the background.

4.5 General Data File Read Programs

4.5.1 WLOG1

Program type - Mainstream

Source language - FORTRAN

Subroutine/function calls - DREAD, DINIT, DUMPR

Program requirements - assign the data file DEC tape unit to .DAT slot 2

Listed in - Appendix

Summary

WLOG1 checks the contents of a user specified data file of the form generated by the programs described in Section 4.4 and prints at the teletype all the data points which correspond to a given partial-reflection height. Output is millivolts units.

4.5.2 WLOG2

Program type - Mainstream

Source language - FORTRAN

Subroutine/function calls - DINIT, DUMPR

Program requirements - assign the data file DEC tape to .DAT 2.

Listed in - Appendix

Summary

This program prints the contents of a user specified data file whose format conforms to that which is described in Section 4.4. The complete contents of the data file are output in decimal coded binary integers.

4.6 A Program to Calculate Electron Density Profiles

4.6.1 PROC2

Program type - Mainstream

Source language - FORTRAN

Subroutine/function calls - DREAD, DINIT, DUMPR, ELDEN, EMED2, AVERA2

Program requirements - assign data file to DAT slot 2

Summary

This program processes data stored on DEC tape to produce an electron-density profile in the height range 60-90 km at 1.5 km intervals. The output also provides extraordinary to ordinary amplitude ratios at each height processed averaged over 90 samples. Collision frequency profiles derived from rocket measurements are used in this program (Lodato and Mechtly, 1971). The user must supply the name of the data file to be processed, the heights for which output is required, a minimum acceptable signal-to-noise ratio and a maximum acceptable noise level.

The data are supplied in sets of 25 words, the first 4 words represent a noise measurement and they are averaged to produce a mean noise level for the time at which the data were collected. The remaining 21 words of data correspond to amplitudes reflected from the D region. Ordinary and extraordinary data sets are stored alternately on the DEC tape.

Results are printed to three significant figures, which represent the limit of sensitivity of the analog to digital converter.

4.6.2 DREAD

Program type - subroutine

Source language - FORTRAN

Called by - PROC2, WLOG1

Subroutine/function calls - DUMPR

Size - 425_8

Program requirements - See DUMPR

Listed in - Appendix 1

Call statement - DREAD(II,MIN,A,BMEAN,IERR, ID, KEOF)

Summary

This subroutine obtains three words of data from each data set stored on DEC tape in the manner described in Section 4.4. The first word of data is the sequence number associated with the data set, the second is the mean of the four noise samples associated with the same data set and the third is the data associated with a particular height at which data were sampled. In addition, a flag is raised if an end of file character is detected. A test is made for negative data, if a negative datum point is detected a message will be printed at the control teletype. Two conditions will give rise to a terminal error, in which case control will be passed to an assigned statement number. The most important condition is detection of a sequence number which is neither consecutive nor the end of file character. A terminal error will also occur if a datum point is requested which is greater than the maximum number of datum points recorded. Output is an array of datum points in millivolt units, each representing the same recording height and each obtained from consecutive data sets.

Symbol List

II - number of data sets to be read on input; on output the number of data sets read

MIN - the height code, 1-21, plus 5

A(II) - On output this contains the data in millivolt units referenced by MIN

DMEAN(II) - the mean noise level for a given data set

IERR - the assigned statement number for terminal errors

ID - on output this contains the sequence number of the last data set read

KEOF - end of file flag

4.6.3 DUMPR

Program type - subroutine

Source language - MACRO

Called by - DREAD, PROC2

Subroutine/function calls - system macro call to DEC tape handler

Size - 562_8

Program requirements - the data file DEC tape should be associated with DAT slot 2

The DEC tape handler DTD may be used

Listed in - Appendix

Call statement - DINIT

DUMPR(IDAT,NEGF)

Summary

The first call should be made from the main program to DINIT. This indicates to the linking loader that the subroutine should be loaded. The main call is to DUMPR and causes one 252_8 block to be read from DEC tape to a ring buffer. The subroutine unpacks data which were packed as described in Section 4.4, and outputs 26_{10} integer words into an array each time that it is called. The 26 words output have the following format:

Word 1 - ID number

Word 2 - 5 - noise samples

Word 6 - 26 - data

If a negative word is read, a flag is set.

Symbol List

IDAT(26) - the contents of one data set stored on DEC tape

NEGF - a flag which is set if a negative number is detected

4.7 NCALC

Program type - mainstream

Source language - FORTRAN

Subroutine/function calls - ELDEN

Program requirements - stored on the partial-reflection tape and interacts with the control teletype

Listed in - Appendix

Summary

This program was developed to investigate the behavior of the function ELDEN; it requests the following data at the teletype:

1. the height interval
2. the extraordinary to ordinary amplitude ratio at the lower bound of the height interval
3. the extraordinary to ordinary amplitude ratio at the upper bound of the height interval

4. the collision frequency at the lower bound

5. the collision frequency at the upper bound

The output is the calculated electron density.

The following rules should be borne in mind when using the program.

1. All input quantities are real numbers and a decimal point must be typed.

2. Each input quantity must be followed by a carriage return.

3. The height interval must be in kilometer units.

4. The collision frequency is assumed to be times 10^7.

5. The height interval will normally only be requested once, to return to that point in the program reply to all data requests with a carriage return.

5. RESULTS AND DISCUSSION

Figure 5.1 shows typical A-scan photographs of the ordinary and the extraordinary partial reflections. The horizontal axis represents time, or virtual height. The far left vertical graticule corresponds to the center of the transmitted pulse. The scale is 100 μsec per division, or 15 km per centimeter. The center vertical graticule corresponds to 75 km. The data shown were recorded at 1400 CST on April 28, 1970, by Reynolds (1970). Figure 5.2 shows electron-density profiles deduced from rocket data obtained at Wallops Island, Virginia (Mechtly and Smith, 1968); these are included for comparison purposes. The ultimate measure of the validity of the experimental results must be as a comparison with results obtained from a different method.

The Partial Reflection Automated Digital Data Logging system (PRADDL) commenced operation in early 1971. Figures 5.3 through 5.14 illustrate the results obtained on May 7, 1971 at solar zenith angles 60°, 47°, 36°, 23.5°, 34° and 50°. Data were collected for 25-minute periods and those data samples which did not meet the $S/N > 1.5$ criterion in the same two-second interval were rejected. The data for each height were then averaged over the 25-minute period. The averages of the 25-minute groups were used to calculate the amplitude ratio and the electron-density profiles. Table 5.1 presents the number of data samples rejected vs. altitude for each of the data collection periods mentioned above. Actually, 25-minute average values of A_x and A_o were separately determined at height intervals of 1.5 between 60 and 90 km; and Figures 5.3-5.8 are in reality height profiles of $\overline{A}_x/\overline{A}_o$ or $\Sigma A_x/\Sigma A_o$. The electron-density profiles shown in Figures 5.9-5.14 were derived from the corresponding $\overline{A}_x/\overline{A}_o$ profiles presented in Figures 5.3-5.8, assuming an electron collision frequency profile for equinox conditions (Lodato and Mechtly, 1971).

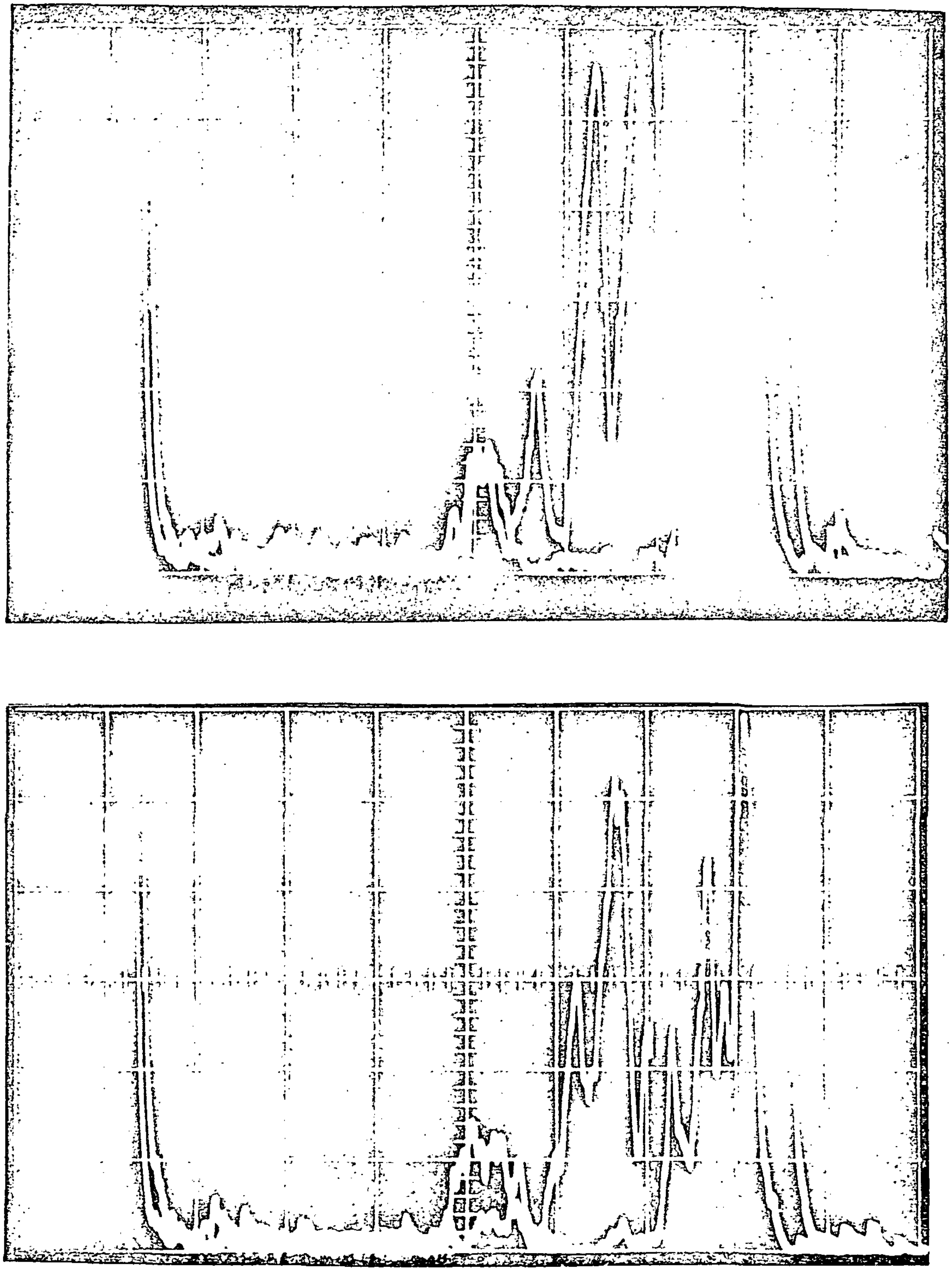

Figure 5.1 A-scan photographs of partial reflections.

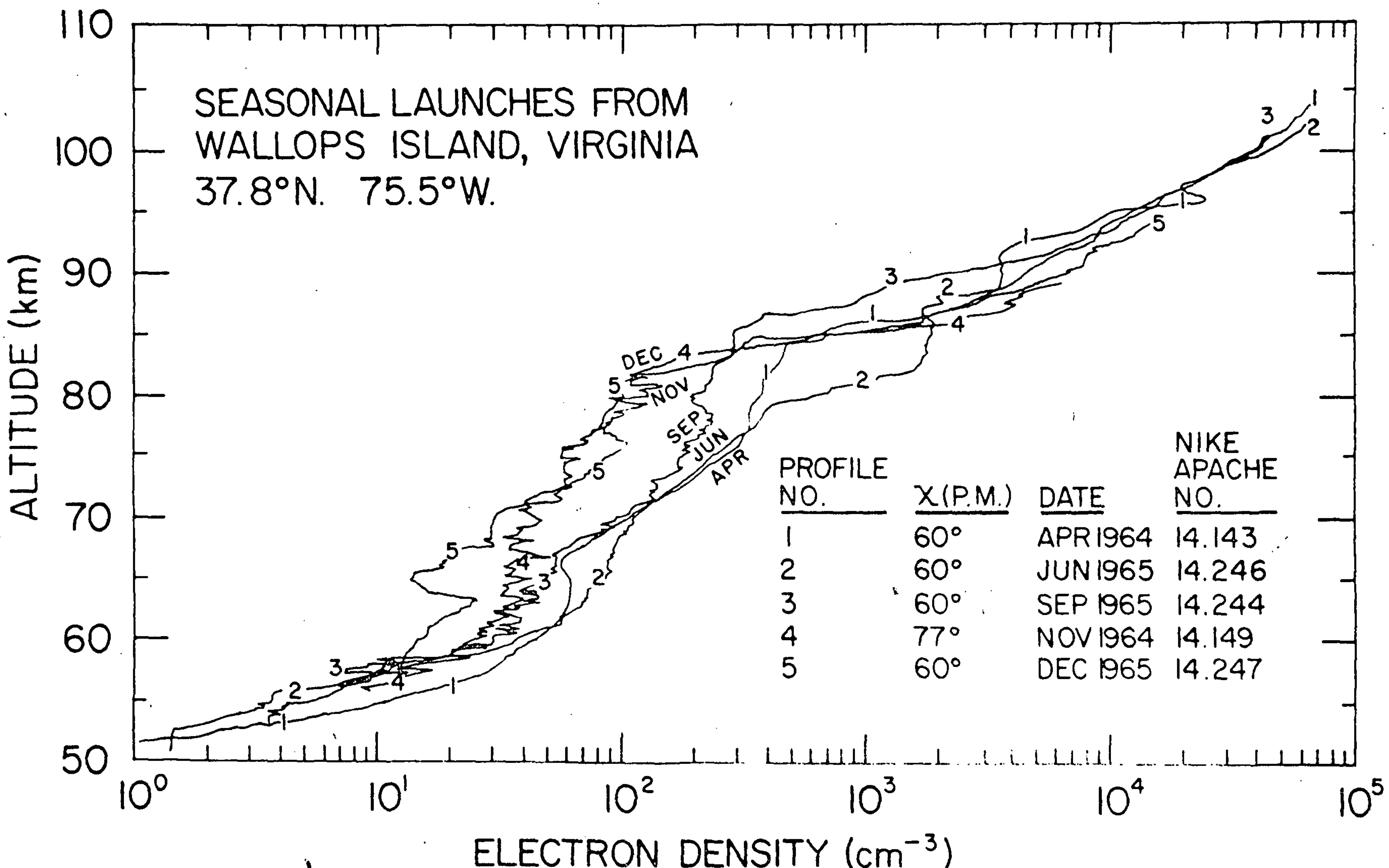

Figure 5.2 Electron-density profiles derived from rocket measurements.

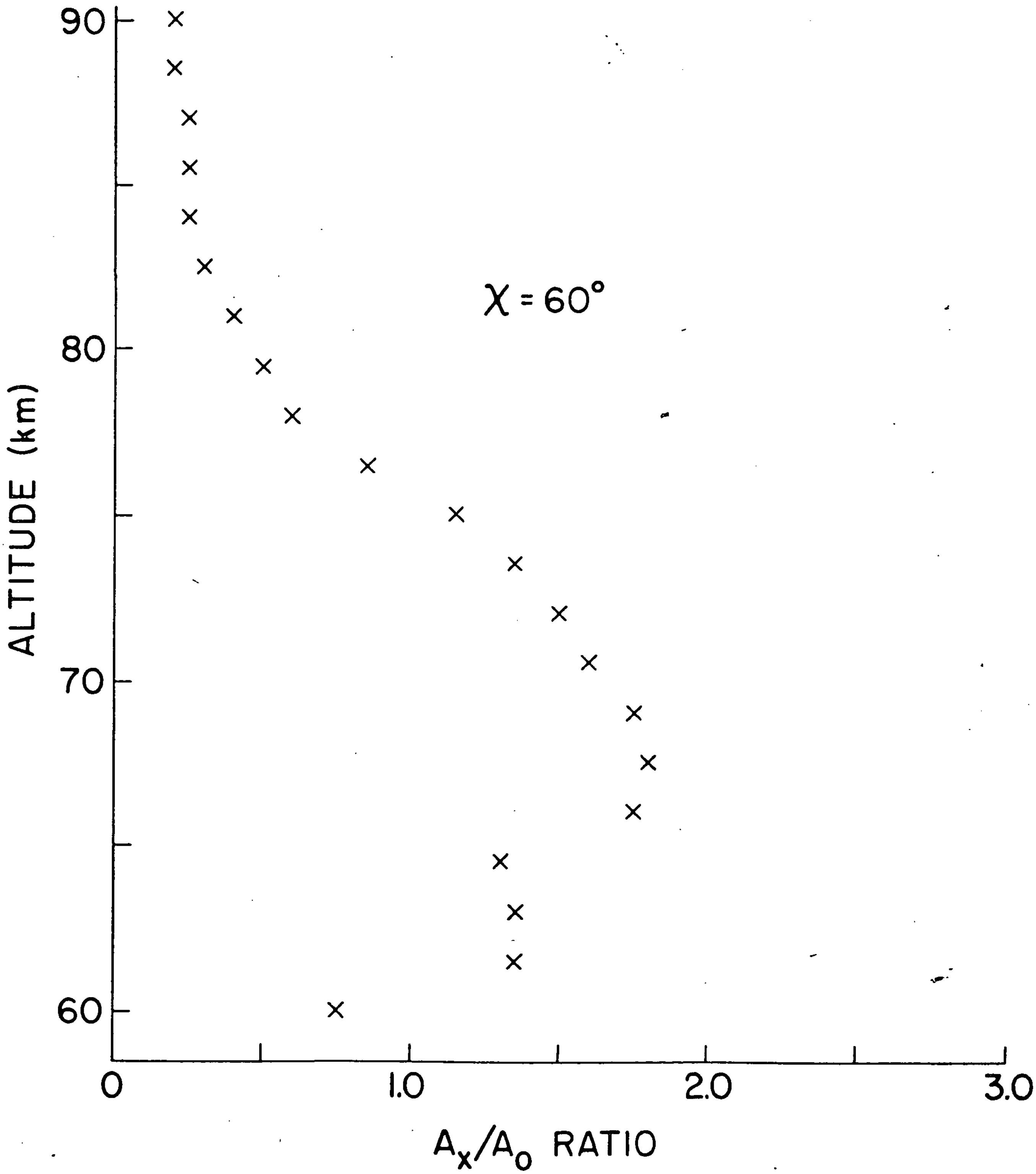

Figure 5.3 Amplitude ratio profile. Date: May 7, 1971, Time: 7:30-7:55 CST.

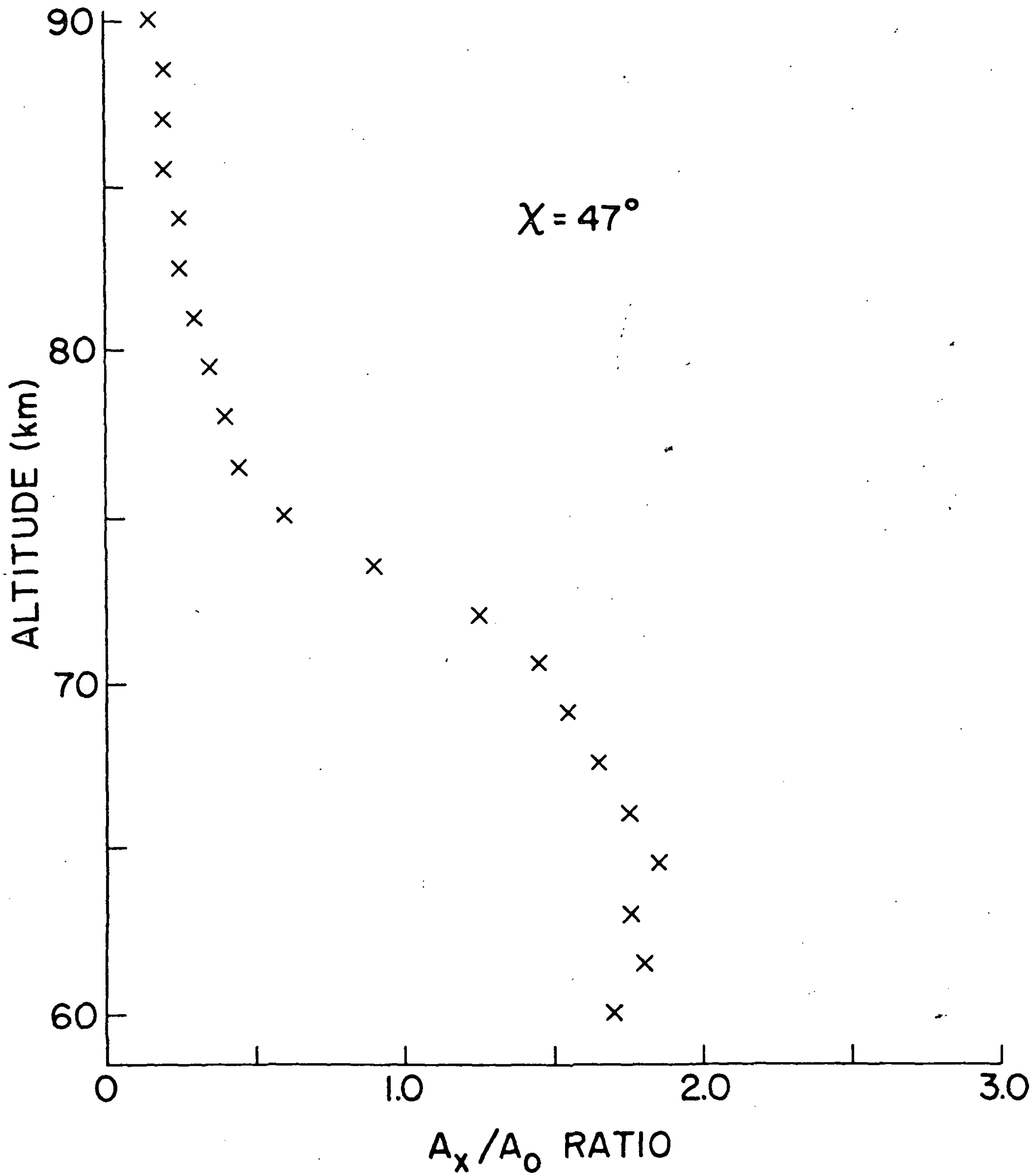

Figure 5.4 Amplitude ratio profile. Date: May 7, 1971, Time: 8:25-8:50 CST.

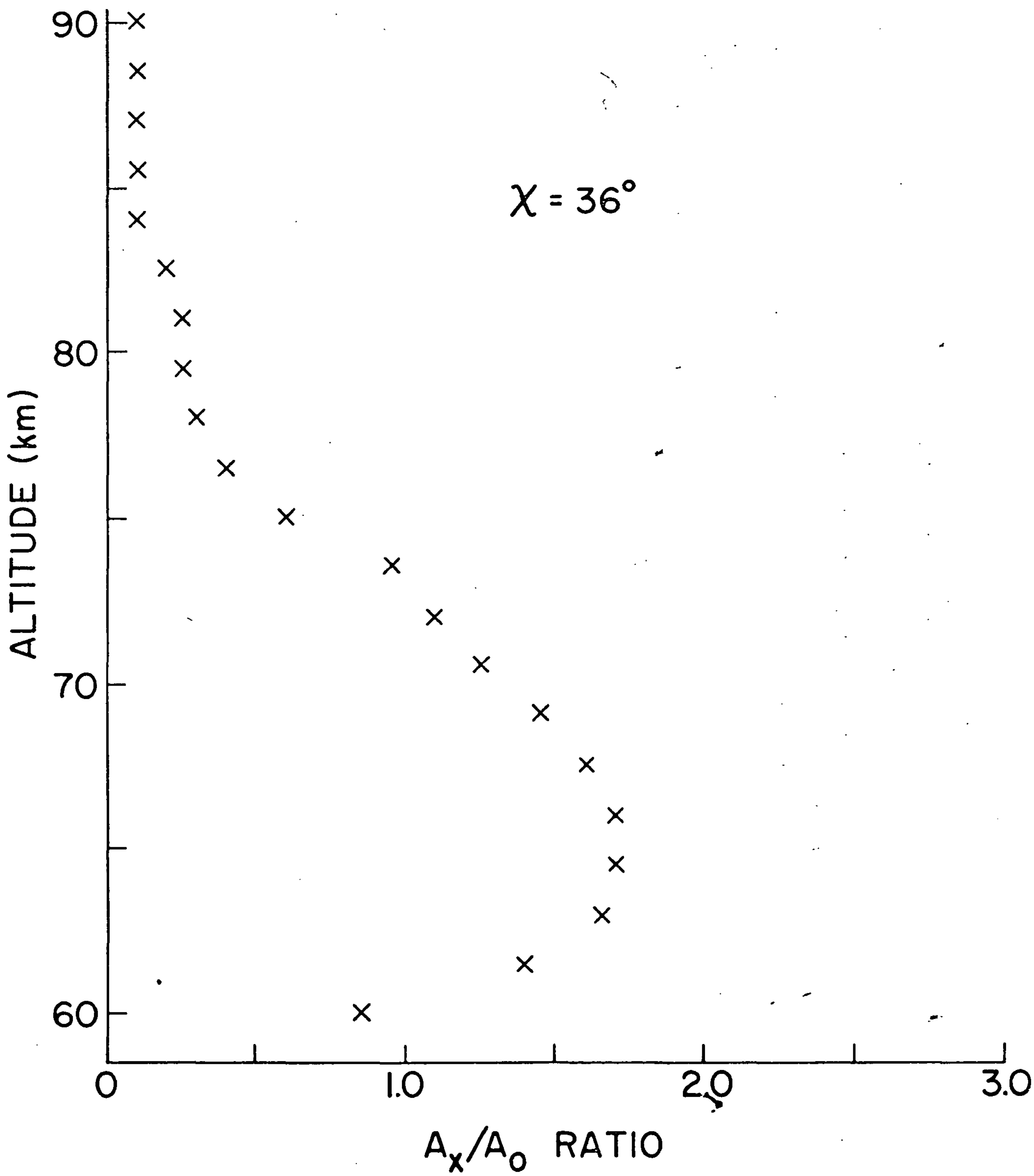

Figure 5.5 Amplitude ratio profile. Date: May 7, 1971, Time: 9:27-9:52 CST.

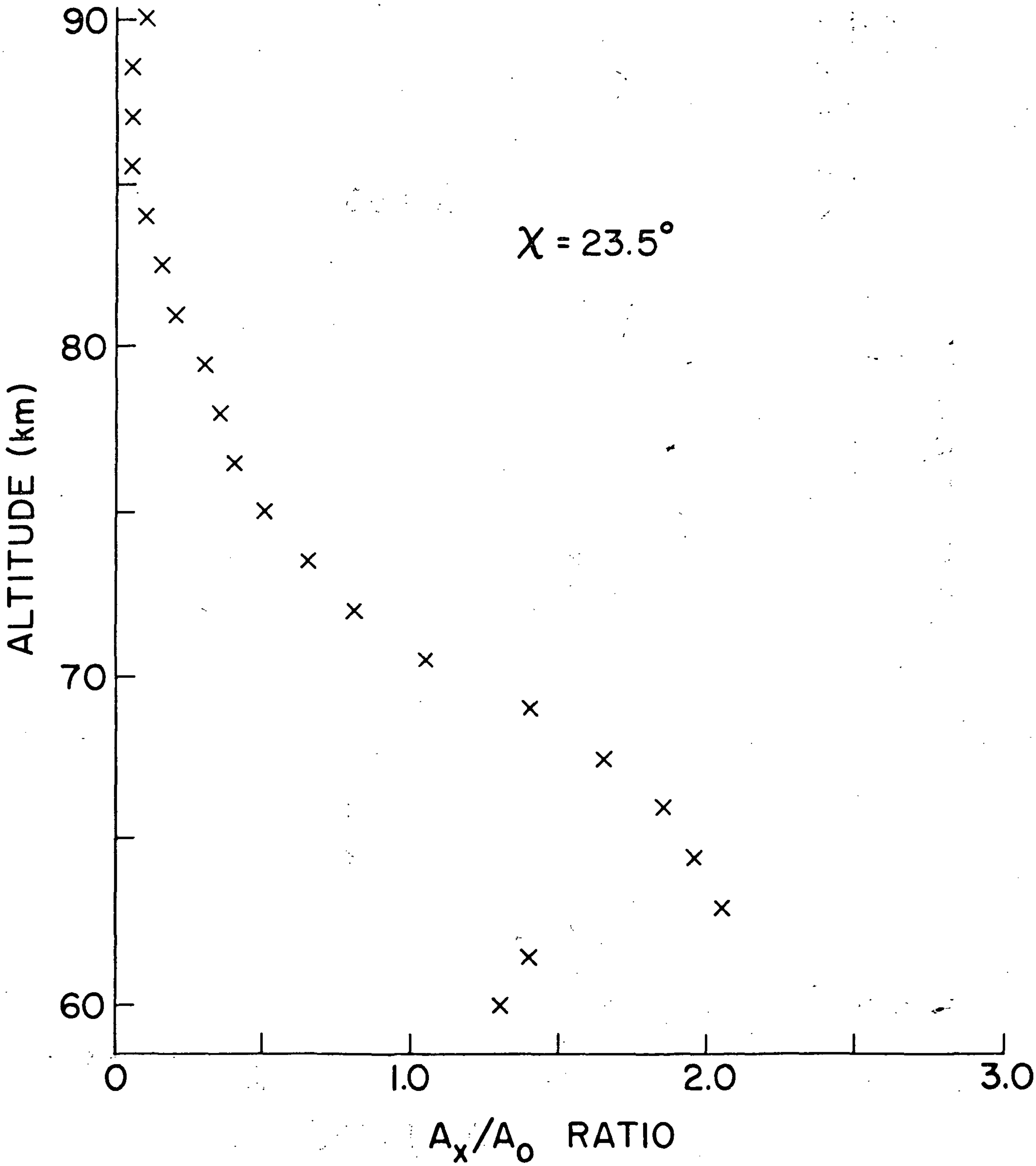

Figure 5.6 Amplitude ratio profile. Date: May 7, 1971, Time: 11:47-12:12 CST.

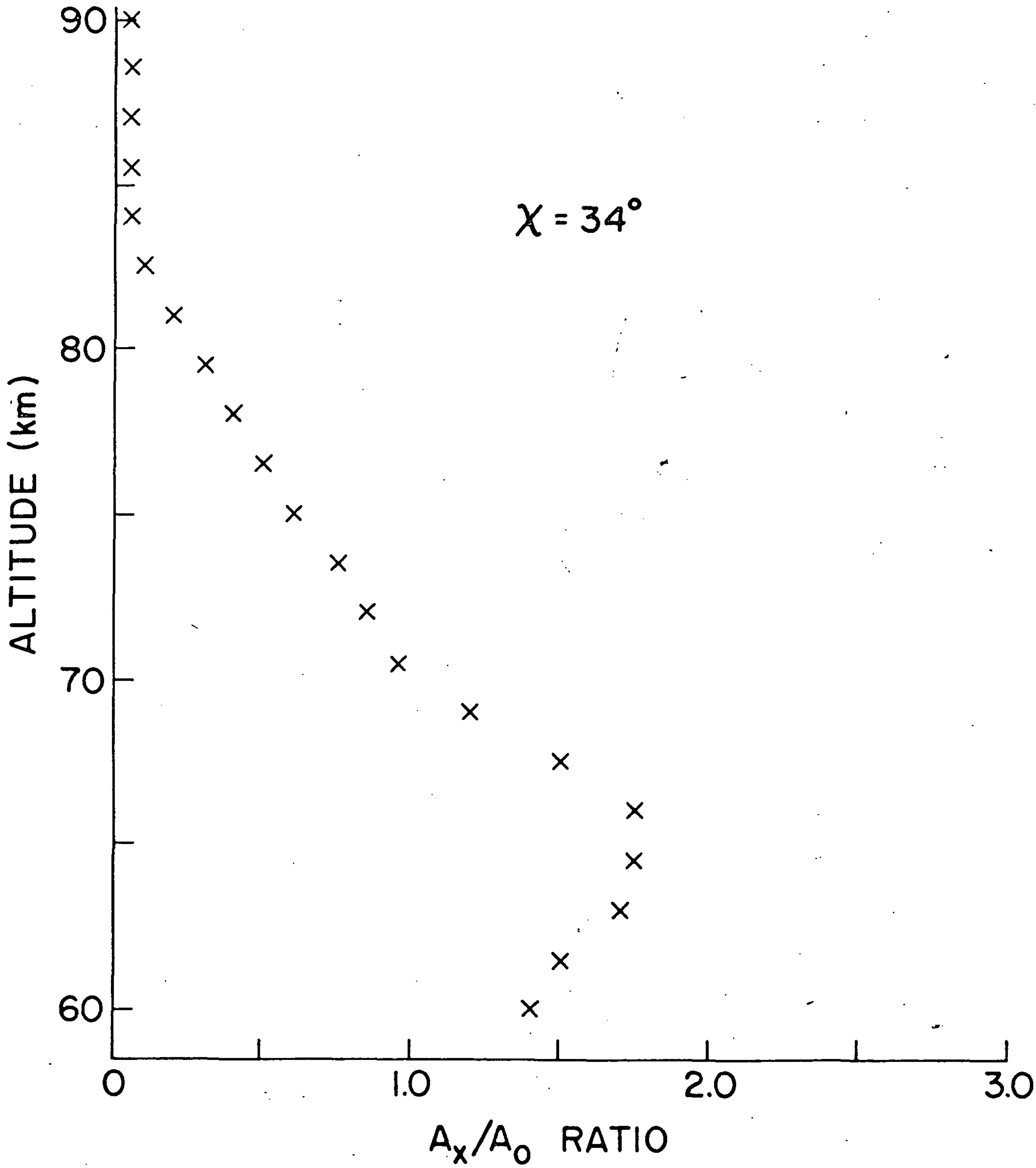

Figure 5.7 Amplitude ratio profile. Date: May 7, 1971, Time: 13:30-13:55 CST.

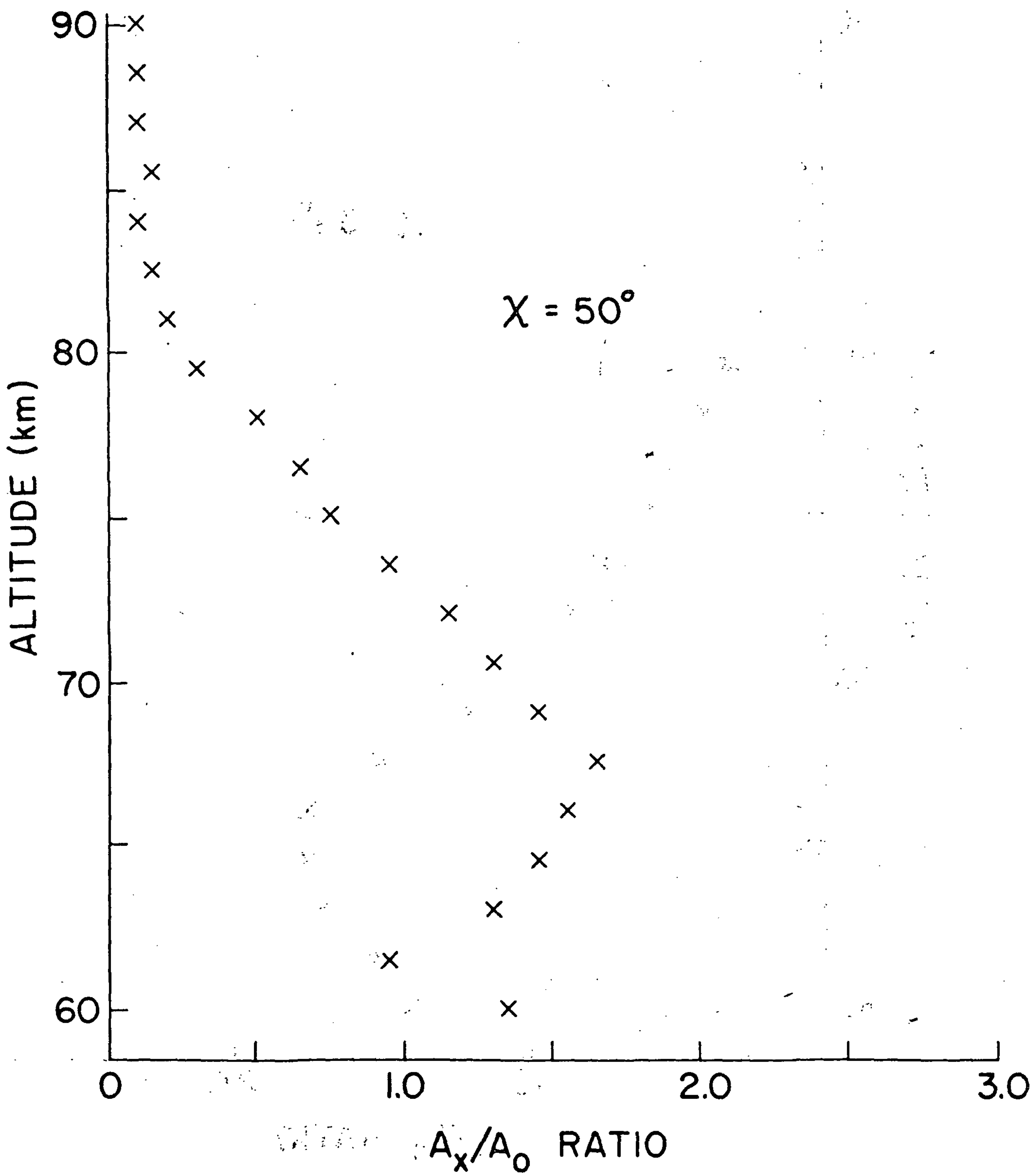

Figure 5.8 Amplitude ratio profile. Date: May 7, 1971, Time: 15:00-15:25 CST.

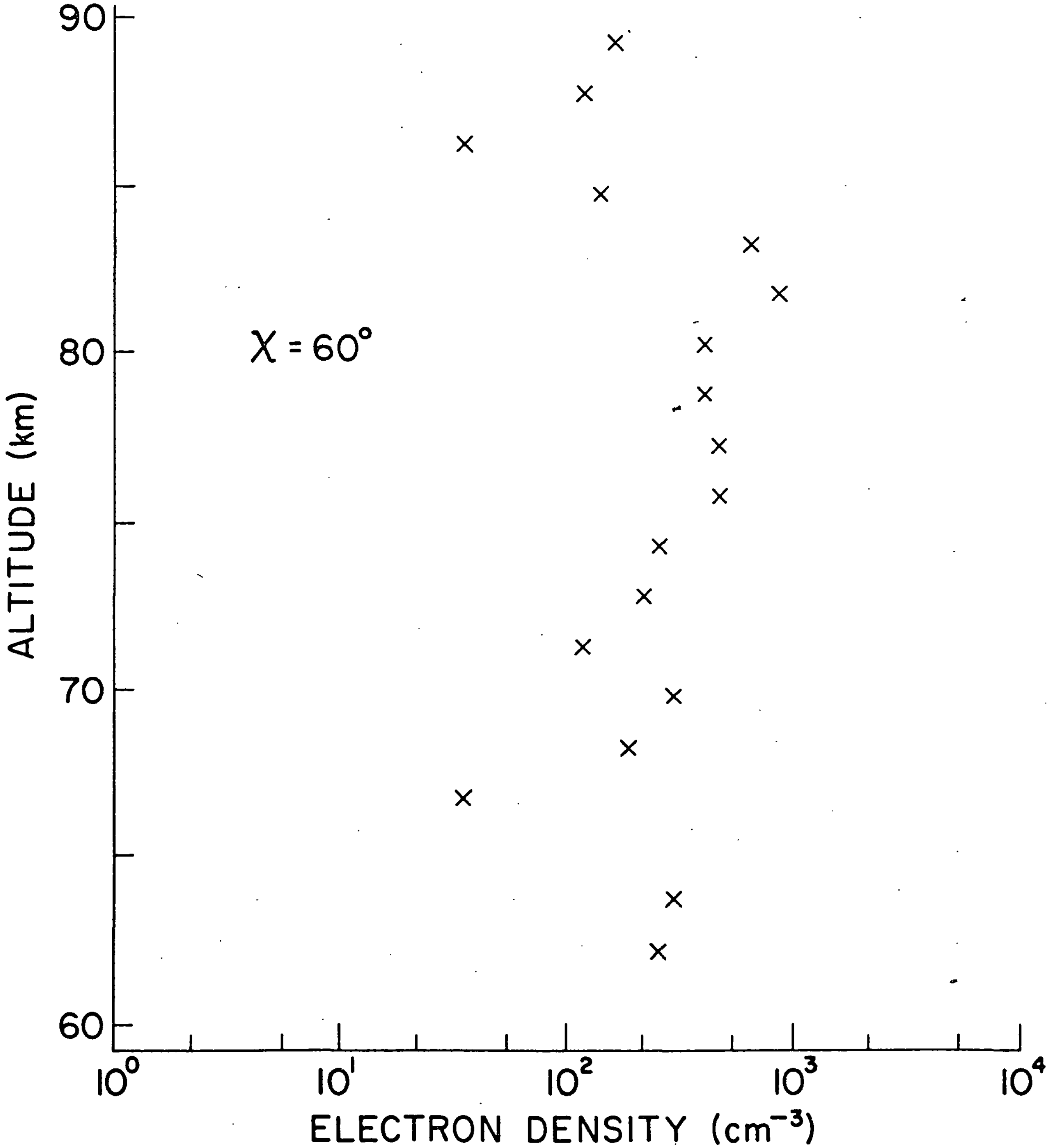

Figure 5.9 Electron-density profile. Date: May 7, 1971, Time: 7:30-7:55 CST.

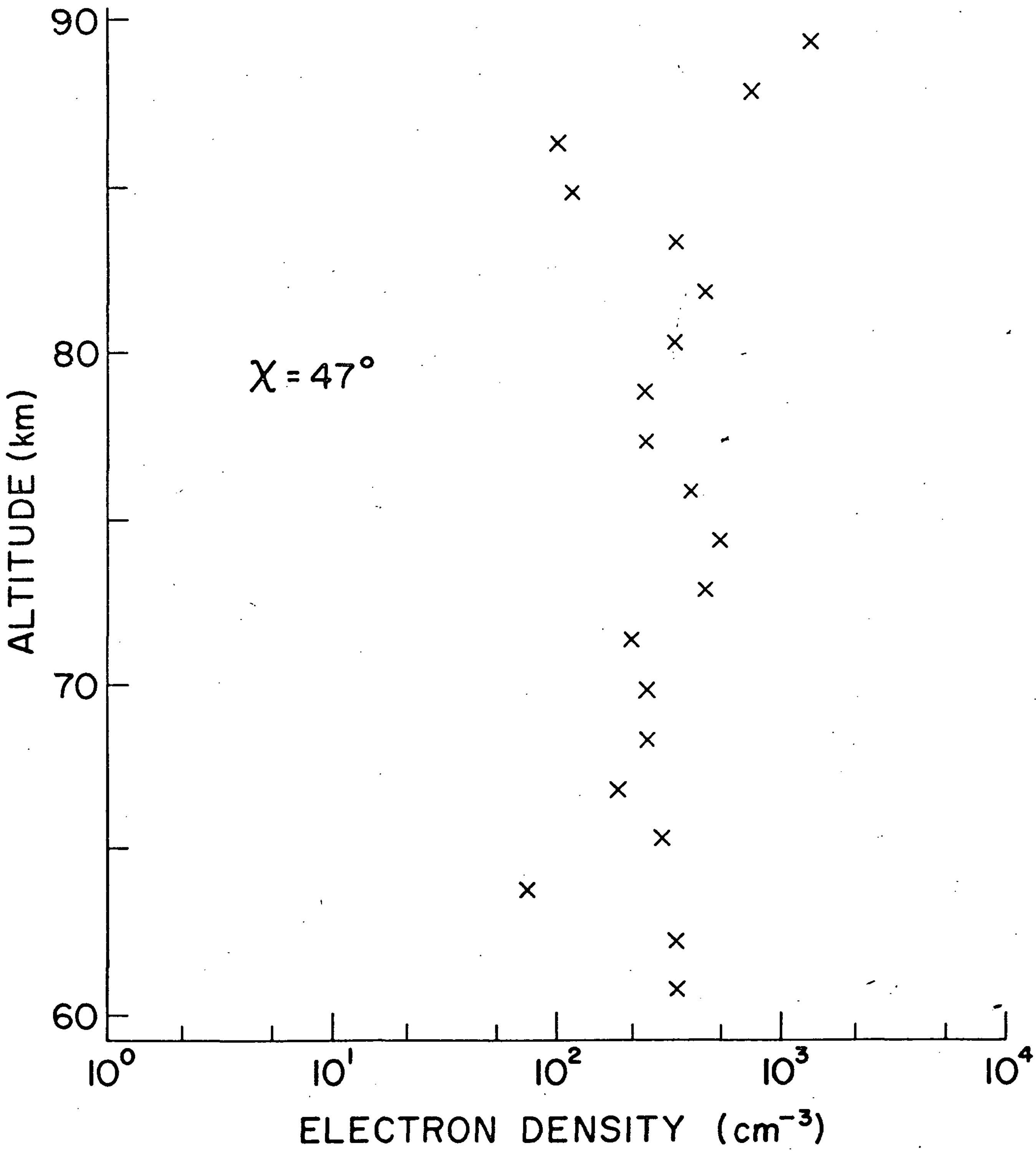

Figure 5.10 Electron-density profile. Date: May 7, 1971, Time: 8:25-8:50 CST.

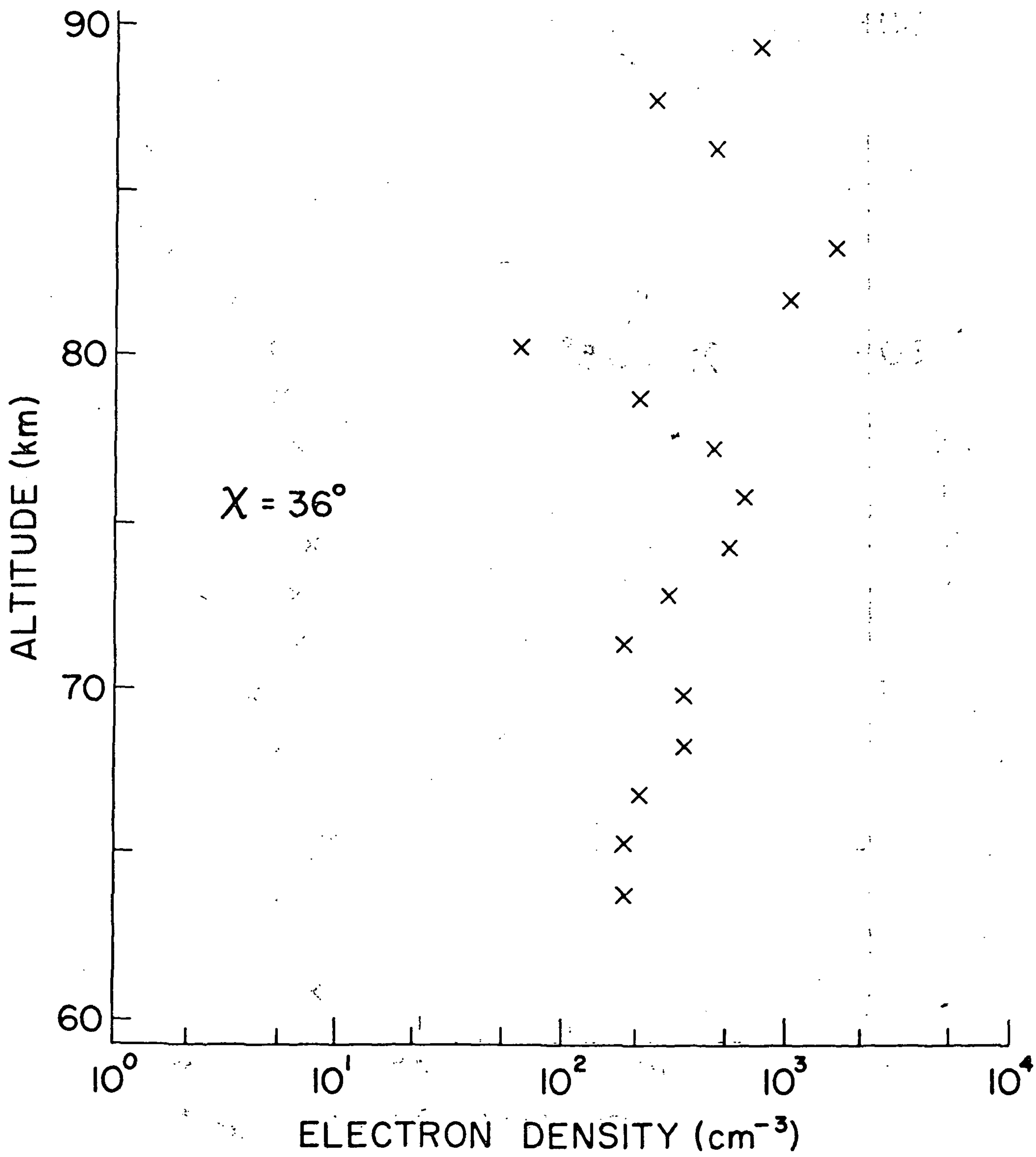

Figure 5.11 Electron-density profile. Date: May 7, 1971, Time: 9:27-9:52 CST.

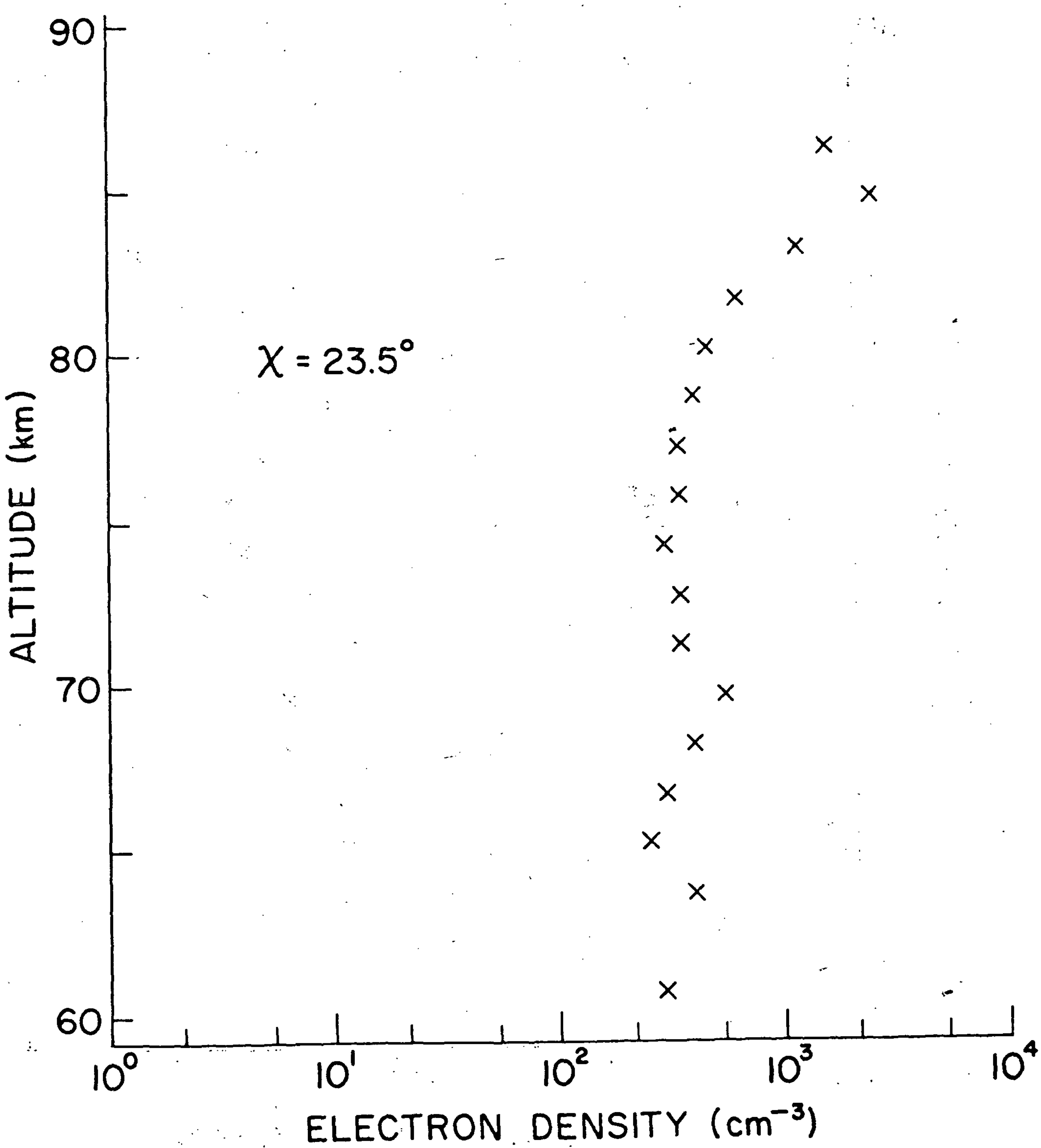

Figure 5.12 Electron-density profile. Date: May 7, 1971, Time: 11:47-12:12 CST.

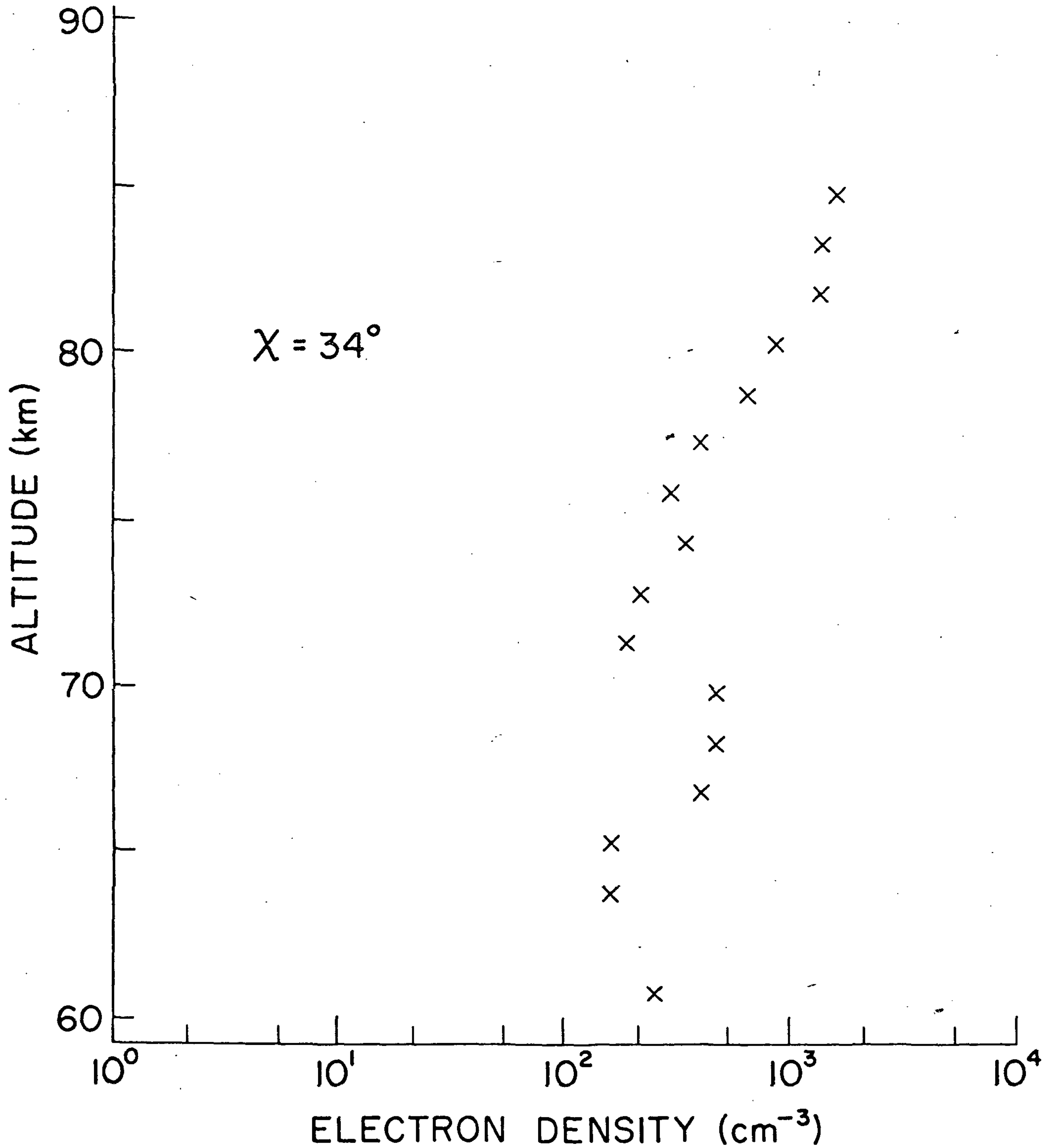

Figure 5.13 Electron-density profile. Date: May 7, 1971, Time: 13:30-13:55 CST.

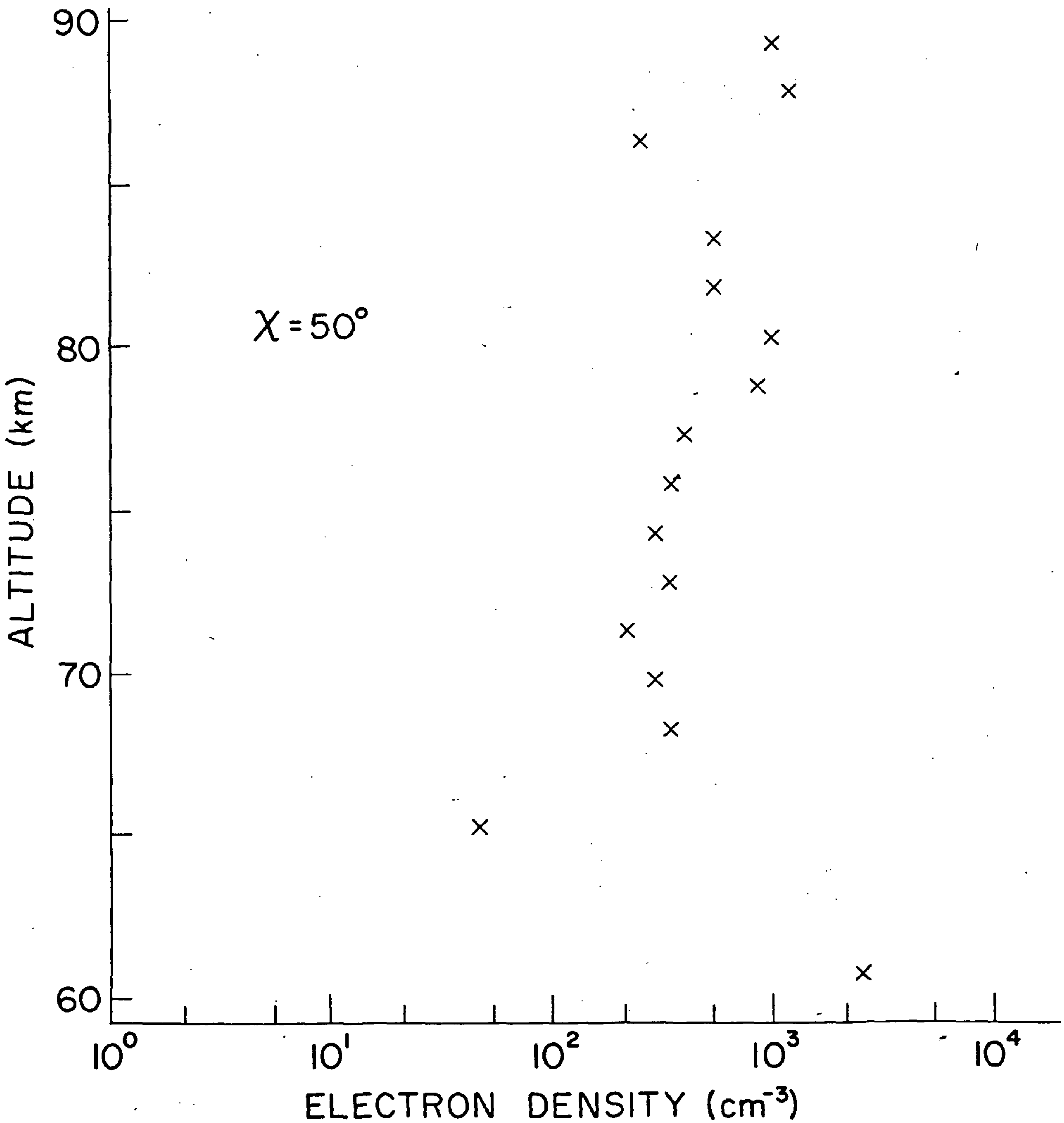

Figure 5.14 Electron-density profile. Date: May 7, 1971, Time: 15:00-15:25 CST.

TABLE 5.1 Number of data samples rejected vs. altitude for May 7, 1971. Times given are CST. Rejection signal-to-noise ratio was $S/N \leqq 1.5$. Total number of samples for each profile was 756.

Altitude (km)	0825-0850 $\chi = 47°$	0927-0952 $\chi = 36°$	1147-1212 $\chi = 23.5°$	1330-1355 $\chi = 34°$	1500-1525 $\chi = 50°$
60	736	736	754	700	734
61.5	711	641	748	545	692
63	580	468	607	414	534
64.5	406	364	458	323	414
66	274	289	365	225	331
67.5	181	247	304	140	257
69	148	205	266	79	188
70.5	116	192	252	69	150
72	109	199	228	67	132
73.5	109	190	249	63	119
75	100	216	291	70	109
76.5	148	283	339	94	114
78	186	321	437	132	115
79.5	196	346	579	196	153
81	234	388	668	325	203
82.5	335	485	706	453	264
84	418	545	724	565	289
85.5	445	606	730	662	320
87	440	620	728	662	348
88.5	415	627	730	642	413
90	430	684	715	601	512

It is evident that a small error in the $\overline{A}_x/\overline{A}_o$ ratio around the 60 or 85 km levels can cause larger errors in the computed electron density than equivalent errors around 75 km, as demonstrated by Belrose (1970). Because the method of determining the electron density is dependent on the difference of the logarithms at closely spaced intervals, and because the $\overline{A}_x/\overline{A}_o$ values have some degree of uncertainty, the resulting error could be considerable. The accuracy with which the $\overline{A}_x/\overline{A}_o$ profile can be measured thus determines the precision with which the electron density can be determined. For a given precision in the measurement of $\overline{A}_x/\overline{A}_o$, the accuracy with which the electron density can be determined depends strongly on height.

Figure 5.9 presents the electron-density profile derived from the partial-reflection experiment on May 7, 1971 (7:30-7:55 CST) for a solar zenith angle near 60°. Comparison of Figure 5.9 with the rocket electron-density profiles in Figure 5.2 reveals that reasonable agreement exists in the 67.5 to 82.5 km altitude range. The poor agreement below 67.5 km may be ascribed to the relatively large number of A_o and A_x data samples rejected below this height. For example, of the 756 data samples taken at 60, 61.5, 63, 64.5, 66 and 67.5 km, the number of samples rejected was 745, 745, 728, 670, 581 and 482, respectively. It is quite clear that below 67.5 km, the signal-to-noise ratio on this particular occasion was unsatisfactory; this is further evident in the erratic behavior of the $\overline{A}_x/\overline{A}_o$ profile below 67.5 km, as shown in Figure 5.3. Thus, below 67.5 km, small errors in the $\overline{A}_x/\overline{A}_o$ ratio cause larger errors in the computed electron-density profile.

The differential absorption is small below 70 km, under normal undisturbed conditions at middle latitudes, and the derived electron-density values are critically dependent on small changes in the slope of the $\overline{A}_x/\overline{A}_o$ profile. Apparently, this profile is not determined with sufficient accuracy to give reliable electron-density data at low D-region heights. However, electron densities are sometimes estimated assuming that the electron density at low heights is proportional to the ordinary-mode echo amplitude A_o (Belrose, 1970). The poor agreement between the rocket and partial-reflection electron densities above 80 km may be attributable to uncertainties in the $\bar{A}_x/\bar{A}_o$ ratio above that altitude. That is, above 80 km the $\bar{A}_x/\bar{A}_o$ ratio is difficult to measure accurately because it decreases rapidly to low values; and there is a minimum mean ratio that can be measured because the A_x amplitude must be above the noise level and the A_o amplitude must be less than its saturation level.

At times, there is a minimum in the electron-density profile in the height range 80-85 km. The exact details of the electron-density profile around 80-85 km depend critically on the accuracy of the $\bar{A}_x/\bar{A}_o$ profile there. Minima in the electron-density profiles sometimes appear near altitudes where the occurrence frequency of partial reflections is a maximum. This suggests the presence of steep gradients in electron density at discrete heights, which are very persistent. Furthermore, it is possible that the steep ledge of electron density, observed around 82 km in the rocket electron-density profiles, is responsible for the "valley" in the electron-density profile around 80 km, derived from partial-reflection data.

The A-scan recording technique represents the most basic method of providing partial-reflection amplitudes. For this reason it was used to determine the validity of the PRADDL system. A single photographic record of the ordinary and extraordinary amplitude profiles was compared with profiles obtained at the same instant of time from the PRADDL system. The results are summarized in Figure 5.15.

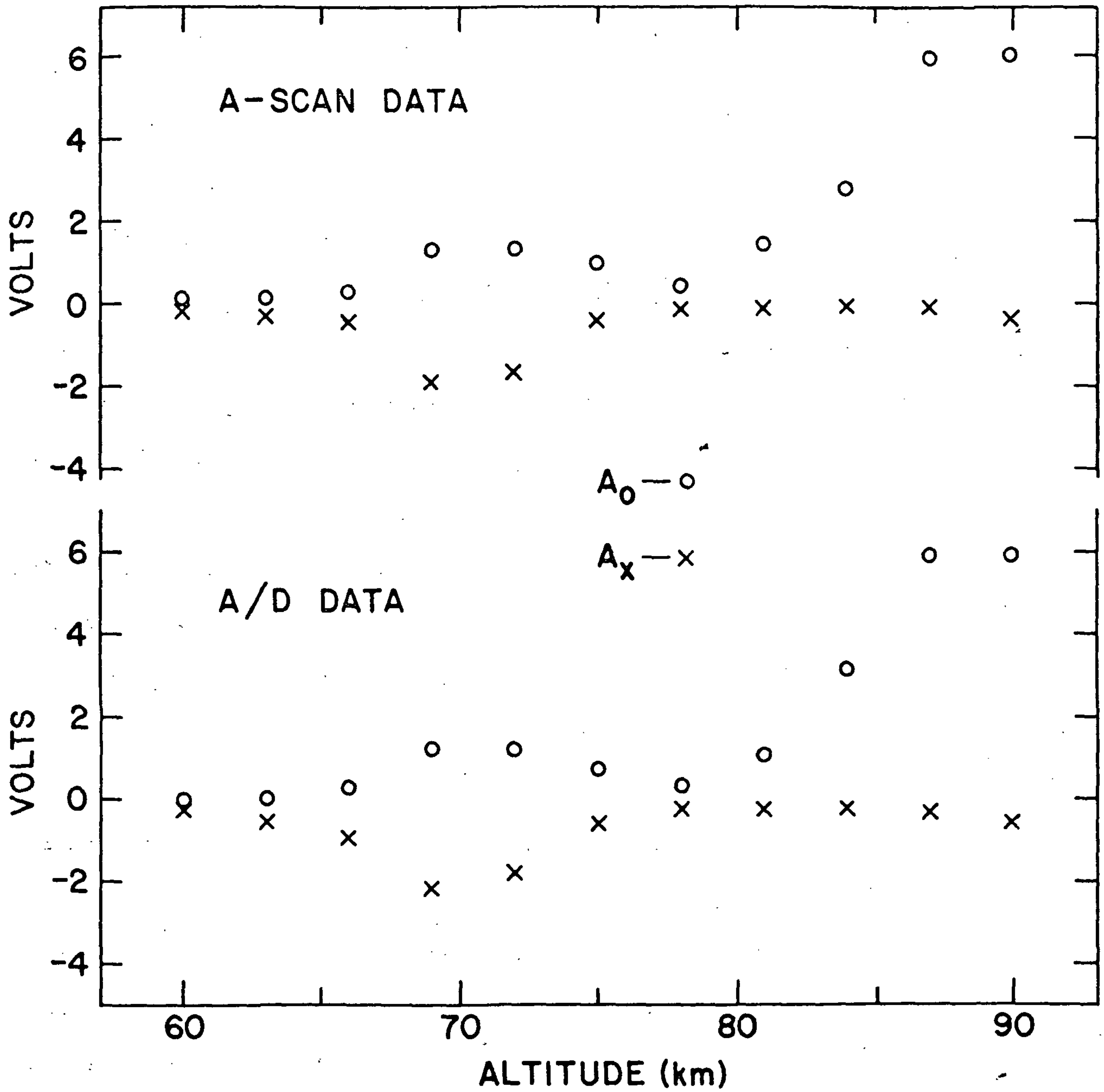

Figure 5.15 A comparison of the PRADDL system with the A-scan technique.

6. CONCLUSIONS AND RECOMMENDATIONS

The partial-reflection facility at the University of Illinois has been in existence since 1968, but the labor involved in reducing the experimental results has not made it possible to operate on a regular basis for most of that time. The incorporation of an automated data logging and data reduction system has proved to be a first step in solving this problem. At present, amplitude data are collected for 25-minute periods. An additional 15 minutes is required to calculate an electron-density profile.

The chief conclusions of this investigation may be summarized as follows:

(a) Below approximately 70 km, the electron columnar density is too small to permit the electron density profile to be derived accurately by differential absorption measurements, during undisturbed daytime conditions at middle latitudes;

(b) The lower altitude limit of the electron-density profiles (~70 km) is caused by the difficulty in measuring weak partial reflections with sufficient accuracy to obtain reliable $\overline{A}_x/\overline{A}_o$ and electron-density profiles;

(c) Under normal, undisturbed conditions in the D region above Urbana, it should be possible to determine accurate electron-density profiles between the altitudes of 68 and 80 km, approximately;

(d) The upper altitude limit of the electron-density profile (~80 km) is established by the decreasing accuracy of the $\overline{A}_x/\overline{A}_o$ values with increasing altitude;

(e) The percentage error in the computed electron-density profiles increases very rapidly at lower levels, because the D-region electron density decreases rapidly with decreasing altitude;

(f) It is essential to determine the slope of the $\overline{A}_x/\overline{A}_o$ profile much more accurately at low and at high D-region altitudes than at altitudes around 75 km, for the same sensitivity in measuring a given electron density;

(g) Preliminary study reveals that minima at certain heights in computed electron-density profiles tend to be associated with a high occurrence frequency of partial reflections near these heights. It is possible that the steep gradient observed in the rocket electron-density profiles near 82 km is responsible for the minimum observed at times in the partial-reflection electron-density profile.

Recommendations for future work include the following:

(a) Higher transmitter power would improve the signal-to-noise ratio for both ordinary- and extraordinary-mode partial reflections. This would reduce the number of rejected data samples below ∿70 km and thus improve the accuracy of the $\overline{A}_x/\overline{A}_o$ profile there;

(b) Another possible solution at the low D-region altitudes is to assume that the electron density below ∿70 km is approximately proportional to $\overline{A}_o$;

(c) Further study is needed of the relationship between electron density minima and a high occurrence frequency of partial reflections at certain altitudes;

(d) Implementation of a differential phase measuring system would permit the estimation of electron densities at the upper D-region altitudes where Faraday rotation of the partially-reflected waves occurs. The differential phase method suggested by Von Biel, et al. (1969) would provide an independent measure of electron density in the upper D region.

REFERENCES

Belrose, J. S. (1970), Radio wave probing of the ionosphere by the partial reflection of radio waves (from heights below 100 km), J. Atmosph. Terr. Phys. 32, 567-596.

Belrose, J. S. and M. J. Burke (1964), Study of the lower ionosphere using partial reflection. 1. Experimental technique and methods of analysis, J. Geophys. Res. 69, 2799-2818.

Budden, K. G. (1961), Radio Waves in the Ionosphere, University Press, Cambridge.

Burke, M. J. and E. H. Hara (1963), Tables of the semiconductor integrals $C_p(X)$ and their approximations for use with the generalized Appleton-Hartree magnetoionic formulas, Defence Research Telecommunications Establishment, Ottawa, Canada, Report 1113.

CIRA (1965), COSPAR International Reference Atmosphere, North-Holland Publishing Co., Amsterdam.

Digital Equipment Corporation Programming Manuals:

FOCAL Programming Manual	DEC-15-KJ2A-D
PDP-15 FORTRAN IV	DEC-15-KJ7B-D
MACRO-15 Assembler	DEC-15-AM2A-D
Advanced Monitor Software System for PDP-15/20/30/40	DEC-15-MR2A-D
PDP-9 Background/Foreground Monitor	DEC-9A-MR2B-D
PDP-15 Utility Programs	DEC-15-7W2A-D
PDP-15 Operator's Guide	DEC-15-H2CA-D
User Handbook Volume I, Processor	DEC-15-H2DA-D
User Handbook Volume II, Peripherals	DEC-15-H2DA-D
HP 5610A ADC Special Interface	CSS-MS-E-70-2

Gardner, F. F. and J. L. Pawsey (1953), Study of the ionospheric D-region using partial reflections, J. Atmosph. Terr. Phys. 3, 321-344.

Lodato, R. F. and E. A. Mechtly (1971), Comparisons of electron collision frequency in the lower ionosphere obtained from simultaneous rocket measurements of the absorption of radio waves and of 1216-Å radiation, presented at the 1971 Spring URSI Meeting, Washington, D. C.

Mechtly, E. A. and L. G. Smith (1968), Seasonal variation of the lower ionosphere at Wallops Island during IQSY, J. Atmosph. Terr. Phys. 30, 1551-1561.

Pirnat, C. and S. A. Bowhill (1968), Electron densities in the lower ionosphere deduced from partial reflection measurements, University of Illinois Aeronomy Report No. 29, University of Illinois, Urbana, Illinois.

Reynolds, D. A. and C. F. Sechrist, Jr. (1970), Measurement of average electron density between 75 and 80 kilometers, University of Illinois Aeronomy Report No. 36, University of Illinois, Urbana, Illinois.

Sen, H. K. and A. A. Wyller (1960), On the generalization of the Appleton-Hartree magnetoionic formulas, J. Geophys. Res. 65, 3931-3950.

Thrane, E. V. and W. R. Piggott (1965), The effect of irregularities in collision frequency on the amplitude of weak partial reflections, J. Atmosph. Terr. Phys. 28, 311-314.

Von Biel, H. A., W. A. Flood and H. G. Camnitz (1970), A differential-phase partial reflection technique for determining low altitude D-region ionization, J. Geophys. Res. 75, 4863-4870.

```
C#####################################################################
C       THIS FUNCTION ACCEPTS AN ARRAY AXBYAO:
C               AXBYAO(1) - ABSORPTION RATIO AT LOWER HEIGHT
C               AXBYAO(2) - ABSORPTION RATIO AT HIGHER HEIGHT
C       ALONG WITH COLLISION FREQUENCY AT THESE HEIGHTS, GNU(1),GNU(2).
C       AN AVERAGE ELECTRON DENSITY AT THE MEAN HEIGHT IS CALCULATED.
C               DELTAH    - HEIGHT INTERVAL
C       THE METHOD USED IS THE QUASI-LONGITUDINAL APPROXIMATION
C       OF SEN-WYLLER. THE IONOSPHERE IS ASSUMED TO BE SLABS OF
C       CONSTANT ELECTRON DENSITY.
C#####################################################################
        FUNCTION ELDEN(AXBYAO,GNU,DELTAH)
        DIMENSION AXBYAO(3),RXBYRO(3),RX(3),RO(3),GNU(3),RATIO(3)
C       APPROX INTEGRAL PARAMETERS
        A4=2.3983474E-2
        A3=1.1287513E+1
        A2=1.139416ØE+2
        A1=2.4653115E+1
        B6=1.8064128E-2
        B5=9.3877372
        B4=1.4921254E+2
        B3=2.8958085E+2
        B2=1.2Ø49512E+2
        B1=2.4656819E+1
        D3=1.163Ø641
        D2=1.69Ø1ØØ2E+1
        D1=6.6945939
        E5=4.36Ø5732
        E4=6.4Ø93464E+1
        E3=6.892Ø5Ø5E+1
        E2=3.5355257E+1
        E1=6.6314497
        AXBYAO(3)=Ø
C       GNU(3) IS MEAN COLLISION FREQUENCY AT THE INTERMEDIATE HEIGHT
C       CALCULATE C INTEGRALS AT BOTH HEIGHTS AND FOR AVERAGE GNU
        DUM=GNU(1)+GNU(2)
        GNU(3)=Ø.5*DUM
        DO 22 K=1,3
        O=(2.59614E+7)/GNU(K)
        X=7.3886E+6/GNU(K)
        CTN=O*(O*(O*(O+A1)+A2)+A3)+A4
        CTD=O*(O*(O*(O*(O*(O+B1)+B2)+B3)+B4)+B5)+B6
        CTO=CTN/CTD
        CTXN=X*(X*(X*(X+A1)+A2)+A3)+A4
        CTXD=X*(X*(X*(X*(X*(X+B1)+B2)+B3)+B4)+B5)+B6
        CTX=CTXN/CTXD
        CFO=(O*(O*(O+D1)+D2)+D3)/(O*(O*(O*(O*(O+E1)+E2)+E3)+E4)+E5)
        CFX=(X*(X*(X+D1)+D2)+D3)/(X*(X*(X*(X*(X+E1)+E2)+E3)+E4)+E5)
C       CALCULATE RATIOS
        RX(K)=SQRT((X*CTX)**2+(2.5*CFX)**2)
        RO(K)=SQRT((O*CTO)**2+(2.5*CFO)**2)
        RXBYRO(K)=RX(K)/RO(K)
        RATIO(K)=AXBYAO(K)/RXBYRO(K)
22      CONTINUE
C       CALCULATE FD FROM FINAL BALUES OF DO LOOP
        FO=(5.*3.1824E+3*CFO)/(4.*3.ØE+8*GNU(3))
        FX=(5.*3.1824E+3*CFX)/(4.*3.ØE+8*GNU(3))
        FD=FX-FO
C       ELECTRON DENSITY AT MEAN HEIGHT
        ELDEN=ALOG(RATIO(1)/RATIO(2))/(2.*DELTAH*FD)
        RETURN
        END
```

```
        .TITLE DUMPR
/ .READ, DUMP MODE FROM DECTAPE ON .DAT 2;
/ FILLS 252 DEC WORD BUFFER AND OUTPUTS 26
/ WORDS TO ARRAY IDAT EVERY TIME CALLED.
/ THESE ARE UNPACKED FROM 14 WORDS OF THE BUFFER.
/ IDAT: WORD 1          I.D. #
/       WORD2-5         NOISE SAMPLES
/       WORD6-26        DATA
/ NEGF:         SET IF A NEGATIVE NUMBER READ
        .GLOBL  DUMPR,.DA,DINIT
        .IODEV  2
DINIT   0
        .INIT   2,0,DINIT
        JMP*    DINIT
DUMPR   0
        JMS*    .DA     /PICKUP ADDR OF ADDR
        JMP     .+3     /OF ARRAY
A       0
FLAG    0       /SET ON NEG #
        LAC*    A
        DAC     A       /ADDR OF ARRAY
        LAC     (-15    /OCTAL #
        DAC     COUNT#
        LAC*    POINT
        SAD     SIX     /END OF BUFFER?
        SKP
        JMP     LBB
LBA     .READ   2,4,BUF1,252
        .WAIT   2
LCA     ISZ     SWITC   /READ TWO DUMMY BLOCKS
        JMP     LBA
        LAC     (-3
        DAC     SWITC
        LAC     (JMP LCB
        DAC     LCA
LCB     ISZ     POINT
        LAC*    POINT
        DAC     POINT   /POINT TO BUF1
LBB     LAC*    POINT
        SAD     SEVN    /END OF FILE ID?
        JMP     ENF
        LAC*    POINT
        DAC*    A
LOOP    ISZ     A
        ISZ     POINT
        JMS     UNPAC
        LAC     HI
        DAC*    A
        ISZ     COUNT
        SKP
        JMP     OUT
        ISZ     A
        LAC     LO
        DAC*    A
        JMP     LOOP
OUT     ISZ     POINT
        JMP*    DUMPR
SWITC   777775
        .DEC
BUF1    .BLOCK  252
SIX     66666
        .DSA    BUF1
SEVN    77777
        .OCT
POINT   .DSA    SIX
HI      0
LO      0
/-----------------------------------------
/ UNPACK TWO  DATAPOINTS FROM WORD STORED ON
/ DECTAPE
UNPAC   0
        LAC*    POINT
        AND     (400000 /SIGN OF HI?
        SAD     (400000
        JMP     HIMINUS
        DZM*    FLAG    /CLEAR FLAG
        LAC*    POINT
        AND     (777000 /HI IS POSITIVE
        .REPT   4
        CLL!RTR
        JMP     STOR1
HIMINUS LAC*    POINT   /TWOS COMP NOTATION
        DAC*    FLAG    /SET FLAG
        AND     (777000
```

```
        .REPT   4
        CLL!RTR
        TAD     (776000
STOR1   DAC     HI
        LAC*    POINT
        AND     (400
        SAD     (400    /SIGN OF LO?
        JMP     LOMINUS
        LAC*    POINT
        AND     (777    /MASK LO
        CLL!RAL
        JMP     STOR2
LOMINUS LAC*    POINT
        DAC*    FLAG    /SET FLAG
        AND     (777
        CLL!RAL
        TAD     (776000
STOR2   DAC     LO
        JMP*    UNPAC
/ END OF FILE ROUTINE
ENF     LAC     (SIX
        DAC     POINT   /ADDR OF SIX
        LAC     SEVN
        DAC*    A
        LAC     (ISZ SWITC
        DAC     LCA
        .CLOSE  2
        JMP*    DUMPR
        .END
```

```
C###############################################
C DREAD TRANSFERS DATA FROM DECTAPE TO CORE. THE DATA IS ASSUMED
C TO BE IN 26 WORD SETS. THE FIRST WORD OF EACH SET IS AN
C ID NUMBER.THESE ARE CONSECUTIVE ELSE A TERMINAL ERROR OCCURS.
C IF AN ID OF 77777 IS DETECTED, IT IS ASSUMED TO INDICATE
C AN END OF FILE AND II IS SET TO NEAREST LOWER EVEN INTEGER
C (O MODE SAMPLE SET), KEOF IS RAISED.
C SUBROUTINES CALLED: DUMPR
C II     # SETS TO BE READ
C MIN    HT OF INTEREST, HTCODE+5
C A(II)  OUTPUT, MILLIVOLT UNITS, AT THE HT REFERENCED BY MIN
C BMEAN(II)  MEAN NOISE LEVEL FOR GIVEN SET
C IERR   ASSIGNED STATEMENT # FOR TERMINAL ERRORS
C ID     SET SEQUENCE #
C KEOF   END OF FILE FLAG
C###############################################
        SUBROUTINE DREAD(II,MIN,A,BMEAN,IERR,ID,KEOF)
        DIMENSION A(1),BMEAN(1),IDAT(26)
        IF(MIN.GT.26) GO TO IERR
        KEOF=0
        ISWIT=0
        DO 200 ISET=1,II
        CALL DUMPR(IDAT,NEGF)
C CHECK ID CONSECUTIVE
        IF(ID-IDAT(1)+1) 10,15,10
C CHECK FOR EOF
10      IF(IDAT(1).NE.77777) GO TO IERR
C       E.O.F.
        II=(II/2)*2
        KEOF=1
        GO TO 210
15      ID=IDAT(1)
        IF (ISWIT) 40,20,40
20      IF (NEGF) 30,40,30
30      ISWIT=1
C NEGATIVE NUMBER HAS BEEN DETECTED
        WRITE (6,35)
35      FORMAT(1X38HAT LEAST ONE NEGATIVE VOLTAGE RECORDED)
40      A(ISET)=IDAT(MIN)*100/51
C GET MEAN NOISE
        SUM=0
        DO 130 JEL=2,5
        S=IDAT(JEL)
        SUM=S+SUM
130     CONTINUE
        BMEAN(ISET)=SUM*100./(51.*4.)
200     CONTINUE
210     CONTINUE
        RETURN
        END
```

```
C#########################################################A
C       FUNCTION DECOD ACCEPTS AN ARRAY X AND DECODES LOCATIONS
C       I+1 THRU I+4. IT THEN DECIMALISES AND SUMS TO PRODUCE
C       ONE REAL NUMBER.
C##########################################################
        FUNCTION DECOD(X,I)
        DIMENSION X(4),Y(4)
        MA=(I-1)*4
        DO 50 K=1,4
        ME=MA+K
        IF (X(ME)-32.) 20,10,20
10      Y(K)=0
        GO TO 50
20      IF(X(ME)-16.) 30,40,40
30      Y(K)=X(ME)
        GO TO 50
40      Y(K)=X(ME)-16.
50      CONTINUE
        DECOD=10.*Y(1)+Y(2)+0.1*Y(3)+0.01*Y(4)
        RETURN
        END

C##########################################################
C       RDCHEK - CHECKS ARRAY INR.
C               INWDCT  SIZE OF INR.
C               J       NUMBER OF FOUR DIGIT GROUPS.
C       LAST ELEMENT OF INR IS AN END OF LINE XTER (128).
C               IALRED  EXIT TO THIS ADDRESS WHEN END OF TAPE
C                       ENCOUNTERED   -   30 BLANK FRAMES.
C               IABORT  EXIT TO THIS ADDRESS IF TERMINAL PRINT ERROR
C       RDCHEK EXTRACTS FOUR FRAMES OF DATA FROM EACH GROUP OF SIX
C       FRAMES OF DATA WHOSE FIRST ELEMENT IS A SIGN AND WHOSE
C       LAST ELEMENT IS AN END OF LINE. IF EITHER IS MISSING
C       THE DATA CAN STILL BE EXTRACTED. INTERMEDIATE BLANKS ARE IGNORED
C               X       ON EXIT, THIS ARRAY CONTAINS THE FOUR WORD
C       GROUPS.
C##########################################################
        SUBROUTINE RDCHEK(INR,X,J,INWDCT,IALRED,IABORT)
        DIMENSION F(30),INR(1),X(1),TEMP(20)
C       INTEGER TO REAL CONVERSION.
        DO 5 I=1,INWDCT
        F(I)=INR(I)
5       CONTINUE
        J=0
        IF(INWDCT-30) 30,10,10
10      DO 20 I=1,30
C       CHECK FOR END OF TAPE
C       30 BLANKS
        IF (F(I)) 15,20,15
15      WRITE(6,16)
16      FORMAT(2X21H30 FRAMES, NOT BLANKS)
        GO TO 400
20      CONTINUE
        GO TO IALRED
C       EXIT FOR END OF TAPE CONDITION
30      IF (INWDCT-6) 340,35,340
35      DO 38 L=1,4
C       IF NO PUNCH ERRORS GET X AND EXIT
        IDUM=L+J*4
        X(IDUM)=F(L+1)
38      CONTINUE
        GO TO 180
C       LOCATE AND CORRECT ERROR.THERE ARE GT 6 FRAMES IN F.
C       LOCATE 4 DIGITS BETWEEN  ANY PAIR OF  SIGN-E.L,SIGN-SIGN,EL-EL.
340     WRITE (6,350) (F(K),K=1,INWDCT)
350     FORMAT(2X6F6.0)
40      L=1
        DO 110 K=1,INWDCT
        IF(F(K))60,110,60
60      IF(F(K)-128.) 70,120,70
C       IF FIFTH FRAME FROM AN EL OR SIGN IS A 32 THEN DISCARD
C       FOLLOWING GROUP
70      IF(F(K)-32.) 74,72,74
72      IF(L-5) 74,272,74
272     IFLAG1=1
273     WRITE (6,274)
274     FORMAT (1X18HSYSTEM RESET ERROR)
74      IF(F(K)-112.) 75,80,75
C       ASSUME NEG SIGN (64) IS PUNCH ERROR, TREAT AS POS SIGN.
75      IF (F(K)-64.) 90,80,90
80      IF(K-1) 140,110,140
C       THIS FRAME IS A DIGIT TO BE DECODED,STORE IT IN X.
90      IDUM=L+J*4
        X(IDUM)=F(K)
        L=L+1
```

```
      IF(L-6) 110,110,100
C     IGNORE EXTRA FRAMES IN THE MIDDLE
100   WRITE(6,105)
105   FORMAT(2X18HE.L. AND SIGN LOST)
      GO TO 400
110   CONTINUE
      WRITE(6,115)
115   FORMAT(2X24HNO E.L. IN INWDCT FRAMES)
      GO TO 400
120   IF(L-4) 130,380,300
380   IDUM=4*(J+1)
      WRITE (6,385) (X(K),K=1,IDUM)
385   FORMAT(2X4F6.0)
      GO TO 180
C     DATAPOINT LOST PRINT MESSAGE
130   WRITE(6,135)
135   FORMAT(2X27HE.L. BUT LESS THAN 4 DIGITS)
      GO TO 400
140   IF(L-4)145,150,150
145   WRITE(6,146)
146   FORMAT(2X20H SIGN IN WRONG PLACE)
C..................................................................
C     OPERATOR CORRECTS ABORT ERRORS.
C<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<>>>>>>>>>>
400   WRITE(6,410)
410   FORMAT(2X16H HOW MANY GROUPS)
      READ(4,420) J
420   FORMAT(I4)
      IF(J) 425,422,425
422   GO TO IABORT
425   IDUM=J*4
      READ(4,430) (X(I),I=1,IDUM)
430   FORMAT(F8.0)
      WRITE(6,435) (X(I),I=1,IDUM)
435   FORMAT(4F5.0)
      WRITE(6,440)
440   FORMAT(1X22HTYPE 0 IF O.K., ELSE 1)
      READ(4,420) IDUM
      IF(IDUM) 400,182,400
C<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<>>>>>>>>>>
C     PUSH UP LIST. REINITIALISATION.
150   IF(IFLAG2) 152,154,152
C     IF IFLAG2 WAS SET THEN WRITE THIS GROUP OVER LAST GROUP
C     BECAUSE LAST GROUP WAS ERRONIOUS DUE TO NON-PRINTING EL.
C     AS SPECIFIED BY FLAG1
152   IFLAG2=0
      GO TO 155
154   J=J+1
155   IDUM1=K+1
      DO 160 N=IDUM1,INWDCT
      IDUM3=N-K
      TEMP(IDUM3)=F(N)
160   CONTINUE
      IDUM2=INWDCT-K
      DO 170 N=1,IDUM2
      F(N)=TEMP(N)
170   CONTINUE
      INWDCT=IDUM2
      IF(IFLAG1) 171,174,171
171   IFLAG2=1
172   IFLAG1=0
174   GO TO 40
C,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
180   IF(IFLAG2) 182,181,182
181   J=J+1
182   IFLAG2=0
      IFLAG1=0
      RETURN
      END

        .TITLE PTCHAR
        .GLOBL PTCHAR,PTON,PTOFF,.DA
        .IODEV  5
PTON    0               /EP FOR INITIALIZATION
GO      .INIT   5,0,GO
        LAC     (SEVN
        DAC     POINT
        .READ   5,3,BUF2,52     /FILL BUFFER 2
        JMP*    PTON    /RETURN
PTOFF   0
        .CLOSE  5
        JMP*    PTOFF
PTCHAR  0               /TO GET A PAPER TAPE XTER
        JMS*    .DA     /PICK UP PARM ADDRESS
```

```
        JMP     .+2
K       .DSA    0
        LAC*    POINT
        SAD     SEVN
        JMP     LBA       /IF AC =777777
LBB     ISZ     POINT     /BUMP POINTER
        DAC*    K         /PUT VALUE IN PARM
        JMP*    PTCHAR    /RET
LBA     ISZ     POINT
        .WAIT   5         /IF NEW BUFFER NOT FILLED
        LAC*    POINT
        DAC     .+3
        .READ   5,3,0,52            /SET OLD BUFFER ADDR AND ISSUE .READ
        ISZ     POINT
        LAC*    POINT
        DAC     POINT     /SET POINTER TO NEW BUFFER
        LAC*    POINT
        JMP     LBB       /TO COMMON EXIT
        .OCT
BUF1    .BLOCK  64
SEVN    777777
        .DSA    BUF1
        .DSA    BUF2+2
BUF2    .BLOCK  64
        777777
        .DSA    BUF2
        .DSA    BUF1+2
POINT   .DSA    SEVN
        .END

        .TITLE  SET
        .GLOBL  SET,PTCHAR,PTON,.DA
/0000000000000000000000000000000000000000000000000000000000000000
/       SUBROUTINE SET - STARTS PAPERTAPE READ AND LOCATES FIRST OCCUR
/-ENCE OF CHARACTER 'NC'. 'ND' IS NOT USED. PAPERTAPE IS READ IN BLOCKS
/OF 52 FRAMES INTO ALTERNATE BUFFERS. ON EXIT NC IS PREVIOUS CHARACTER.
/00000000000000000000000000000000000000000000000000000000000000000
SET     0
        JMS*    .DA
        JMP     .+2+1
NC      .DSA    0         /XTER TO BE LOCATED ON PTAPE.
ND      .DSA    0         /NOT USED.
        JMS*    PTON
LOOP    JMS*    PTCHAR
        JMP     .+1+1
        .DSA    FRAM      /ADDRESS OF XTER READ FROM PTAPE.
        LAC     FRAM
        SAD*    NC
        JMP*    SET
        JMP     LOOP
FRAM    0
        .END

C###############################################################
C       SUBROUTINE LREC - PLACES XTERS FROM PTAPE READ BUFFER IN
C       ARRAY KK, ELEMENTS NMIN THRU NMAX. TRANSMISSION HALTS AT
C       XTER 'LC', OR AFTER (NMAX-NMIN) XTERS. J IS RETURNED
C       AS NUMBER OF XTERS READ.
C################################################################
        SUBROUTINE LREC (KK,LC,J,NMIN,NMAX)
        DIMENSION KK(32)
        DO 10 J= NMIN,NMAX
        CALL PTCHAR (KK(J))
        IF (KK(J)-LC) 10,20,10
10      CONTINUE
20      CONTINUE
        RETURN
        END

C#############################################################
C       ELECTRON DENSITY AT 77.5 KM IS CALCULATED FROM GROUND BASED
C       PARTIAL REFLECTION AMPLITUDE DATA RECORDED ON PTAPE.
C
C       MAINSTREAM VERSION 1. READS 1 TAPE, OUTPUTS RESULTS TO TTA.
C##############################################################
        DIMENSION IN(35),OUT(16),A075(900),AX75(900),A080(900)
        DIMENSION AX80(900),AVA075(10),AVA080(10),AVAX75(10)
        DIMENSION EMA075(10),EMA080(10),EMAX75(10),EMAX80(10)
        DIMENSION AVAX80(10), RA(3),RM(3),CFREK(3)
        COMMON /BLOCK1/A075/BLOCK2/A080/BLOCK3/AX75/BLOCK4/AX80
        EQUIVALENCE (A075(1),RA(1)),(A075(4),RM(1)),(A075(7),CFREK(1))
        ASSIGN 160 TO NOMOR
        ASSIGN 150 TO MESS
5       IRCOU=1
```

```
C       IRCOU=NUMBER OF RUNS TO BE PROCESSED
        DO 400 MRCOU=1,IRCOU
C"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
C       READS,CHECKS,DECODES AND CONTINUES WITH ABSOPTION ARRAYS,
C       EACH SIZE L. NFRAM IS NUMBER OF FRAMES READ DURING EACH
C       READ TO EOL
C"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
        PAUSE 1
        CALL SET(112,1)
        L=1
        NPTR=0
        IN(1)=112
        PAUSE 2
110     CALL LREC(IN,128,NFRAM,1,30)
        CALL RDCHEK (IN,OUT,J,NFRAM,NOMOR,MESS)
        DO 140 I=1,J
        NPTR=NPTR+1
        GO TO (131,132,133,134),NPTR
131     A075(L)=DECOD(OUT,I)
        GO TO 140
132     A080(L)=DECOD(OUT,I)
        GO TO 140
133     AX75(L)=DECOD(OUT,I)
        GO TO 140
134     AX80(L)=DECOD(OUT,I)
        IF(L-900) 135,138,138
138     NRUN=10
        GO TO 180
135     L=L+1
        NPTR=0
140     CONTINUE
        GO TO 110
C
150     WRITE(6,155)
155     FORMAT(1X,7HABORT 1)
        GO TO 500
C"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
C       WHEN ALL TAPE IS READ CONTINUE HERE. CHECK LAST FOUR XTER
C       GROUP READ PROPERLY
C"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
160     WRITE(6,161) L
161     FORMAT(1X4HL = ,I4)
        IF (NPTR) 165,170,165
165     L=L-1
170     CONTINUE
C       ABSORPTION ARRAYS NOW AVAILABLE
C...................................................................
C       PDATA    NUMBER OF TWO SECOND READINGS
C       AVAZZZ(I)       3 MINUTE ABSORPTION AVERAGES
C       EMAZZZ(I)       3 MINUTE ABSORPTION MEDIANS
C       ZVAVZZ(I)       OVERALL AVERAGE
C       ZMEDZZ          OVERALL MEDIANS
C       FOR ZZZ READ O OR X AND 75 OR 80
C       NRUN            NUMBER  COMPLETE 3 MINUTE INTERVALS
C...................................................................
        PDATA=L
        NRUN=INT(PDATA/90.)
        WRITE (6,175) NRUN
175     FORMAT (1X,7HNRUN = ,I4)
C       CALCULATE 3 MINUTE AVERAGES AND MEDIANS
180     TOTO7=0
        TOTO8=0
        TOTX7=0
        TOTX8=0
        DO 220 I=1,NRUN
        NMIN=90*(I-1)+1
        NMAX=90*I
        AVAO75(I)=AVERAG(AO75,NMIN,NMAX,SO)
        TOTO7=SO+TOTO7
        AVAO80(I)=AVERAG(AO80,NMIN,NMAX,SO)
        TOTO8=SO+TOTO8
        AVAX75(I)=AVERAG(AX75,NMIN,NMAX,SO)
        TOTX7=TOTX7+SO
        AVAX80(I)=AVERAG(AX80,NMIN,NMAX,SO)
        TOTX8=SO+TOTX8
        EMAO75(I)=EMED(AO75,NMIN,90,AVAO75(I))
        EMAO80(I)=EMED(AO80,NMIN,90,AVAO80(I))
        EMAX75(I)=EMED(AX75,NMIN,90,AVAX75(I))
        EMAX80(I)=EMED(AX80,NMIN,90,AVAX80(I))
220     CONTINUE
C       CALCULATE OVERALL AVERAGE AND MEDIAN
        RUN=NRUN
        DIV=RUN*90.
        IDIV=NRUN*90
```

```
      OVAV75=TOTO7/DIV
      OVAV80=TOTO8/DIV
      XVAV75=TOTX7/DIV
      XVAV80=TOTX8/DIV
      OMED75=EMED(AO75,1,IDIV,OVAV75)
      OMED80=EMED(AO80,1,IDIV,OVAV80)
      XMED75=EMED(AX75,1,IDIV,XVAV75)
      XMED80=EMED(AX80,1,IDIV,XVAV80)
C.....................................................................
C       CALCULATE ELECTRON DENSITIES AND ABSORPTION RATIOS AND
C       PRINT WITH HEADING.
C,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
      CFREK(1)=7.0E+5*2.67
      CFREK(2)=7.0E+5*1.13
      DEL=5.0E+3
      WRITE(6,310) NRCOU
310   FORMAT(10X,11HRUN NUMBER ,I3)
      WRITE(6,320)
320   FORMAT(1X,21HTHREE MINUTE AVERAGES,2X,20HTHREE MINUTE MEDIANS)
      WRITE(6,330)
330   FORMAT(1X,20HAX/AO AX/AO ELECTRON,2X,
     120HAX/AO AX/AO ELECTRON,4X19HAO75 AO80 AX75 AX80)
      WRITE(6,340)
340   FORMAT(1X,21H 75    80    DENSITY ,2X,20H 75    80    DENSITY)
      DO 350 I=1,NRUN
      RA(1)=AVAX75(I)/AVAO75(I)
      RA(2)=AVAX80(I)/AVAO80(I)
      RM(1)=EMAX75(I)/EMAO75(I)
      RM(2)=EMAX80(I)/EMAO80(I)
      ELAV=ELDEN(RA,CFREK,DEL)/(10.**9)
      ELME=ELDEN(RM,CFREK,DEL)/(10.**9)
      WRITE(6,345) RA(1),RA(2),ELAV,RM(1),RM(2),ELME,AVAO75(I),
     1AVAO80(I),AVAX75(I),AVAX80(I)
345   FORMAT(1XF5.3,1XF5.3,1XF7.4,4XF5.3,1XF5.3,1XF7.4,4X4(F5.3,1X))
350   CONTINUE
      MIN=NRUN*3
      WRITE (6,360) MIN
360   FORMAT(//5X18HOVERALL VALUES FOR,I4,11H MINUTE RUN/)
      RA(1)=XVAV75/OVAV75
      RA(2)=XVAV80/OVAV80
      RM(1)=XMED75/OMED75
      RM(2)=XMED80/OMED80
      ELAV=ELDEN(RA,CFREK,DEL)/(10.**9)
      ELME=ELDEN(RM,CFREK,DEL)/(10.**9)
      WRITE (6,345) RA(1),RA(2),ELAV,RM(1),RM(2),ELME,OVAV75,OVAV80,
     1XVAV75,XVAV80
C     END OF PRINTING
C
      GO TO 400
C
C     ON ABORT CONDITION END
C
500   CONTINUE
400   CONTINUE
      CALL PTOFF
      GO TO 5
      STOP 1
      END
C####################################################################
C       EMED CONSIDERS LCOUNT ELEMENTS OF ARRAY F STARTING
C       WITH ELEMENT NMIN. AVE IS ANY  GUESS AT THE MEDIAN.
C####################################################################
      FUNCTION EMED(F,NMIN,LCOUNT,AVE)
      DIMENSION F(1)
      SVAL=0
      HVAL=AVE
      EMED=AVE
      NMAX=NMIN+LCOUNT-1
5     ILCOU=0
      IHCOU=0
      DO 20 I=NMIN,NMAX
      IF (F(I)-EMED) 10,10,15
10    ILCOU=ILCOU+1
      GO TO 20
15    IHCOU=IHCOU+1
20    CONTINUE
      IF(ILCOU-IHCOU) 30,90,40
C     EMED IS TOO LOW
30    SVAL=EMED
      GO TO 50
C     EMED IS TOO HIGH
40    IF(ILCOU-IHCOU-1) 45,90,45
45    HVAL=EMED
```

```
      GO TO 50
C     FIND NEW EMED BETWEEN SVAL AND HVAL
50    DO 80 I=NMIN,NMAX
      IF(F(I)-SVAL) 80,80,60
60    IF (F(I)-HVAL) 70,80,80
70    EMED=F(I)
      GO TO 5
80    CONTINUE
C     IF NO EMED BETWEEN SVAL AND HVAL DO THE FOLLOWING
      EMED=HVAL
90    CONTINUE
      RETURN
      END
C########################################################
C WO    LOWER LIMIT OF MASK
C WI    UPPER LIMIT OF MASK
C NINT  # OF INTERVALS
C A     INPUT MATRIX
C NU    SIZE OF A
C FR    OUTPUT MATRIX - FREQUENCY OF OCCURENCE OF INTERVAL
C########################################################
      SUBROUTINE EHIST(WO,WI,NINT,A,NU,FR)
      DIMENSION A(1),FR(1)
      XN=NINT
      X= (WI-WO)/XN
      DO 10 I=1,NINT
      FR(I)=0.
10    CONTINUE
      DO 100 K=1,NU
      IL=(A(K)-WO)/X+1.
      IF(IL.LE.0.OR.IL.GT.NINT) GO TO 100
      FR(IL)=FR(IL)+1.0
100   CONTINUE
      RETURN
      END
```

```
        .TITLE ADC.
/EDITION 1. EXTERNAL INTERRUPT CONTROLLED READ.
/PDP-15/20 SYSTEM
/ANALOG TO DIGITAL CONVERTER (HP 5610A)
                ##################
ADWI=703724     /WRITE INITIALISE
ADSO=703701     /SKIP ON W.C. OFLO
ADST=703721     /SKIP ON DEVICE TIMING ERROR
ADCO=703704     /CLEAR OFLO FLAG
ADCT=703744     /CLEAR DEVICE T ERR
ADRS=703702     /READ STATUS TO AC
                /BIT 15 OFLO
                /BIT 16 T ERR
ADRA=703722     /READ AC. PROCESS CONTROLLED READ
/ API LEVEL 0. POSITIVE I/O BUS. W.C. AT 26.
/ BUFFER ADDRESS AT 27. TRAP ADDRESS AT 57.
/
        .GLOBL ADC.
.MED=3
ADC.    DAC     ADCALP  /SAVE CAL POINTER
        DAC     ADARGP  /AND ARGUMENT POINTER
        ISZ     ADARGP  /POINTS TO FUNCTION CODE
        LAC*    ADARGP  /GET CODE
        ISZ     ADARGP  /POINTS TO CAL+2
        TAD     (JMP DSPCH
        DAC     DSPCH   /MODIFIED JMP
DSPCH   XX
        JMP     ADINIT
        JMP     ADIGN
        JMP     ADIGN
        JMP     ADERR6
        JMP     ADERR6
        JMP     ADCLOS
        JMP     ADOK
        JMP     ADREAD
        JMP     ADERR6
        JMP     ADWAIT
        JMP     ADERR6
/
ADERR4  LAW     4
        JMP*    (.MED+1
/
ADERR6  LAW     6       /ERROR CODE 6
        JMP*    (.MED+1 /TO MONITOR
/
ADINIT  ISZ     ADARGP  /RETURN STANDARD BUFFER SIZE
        LAC     (36
        DAC*    ADARGP
        ISZ     ADARGP
SETUP   CAL     57      /SETUP API TRAP ADDR 57
ADBA    16              /TWO SKP IOTS
ADWC    ADSO
        ADINT
ADUND   CAL     57
ADSAC   16              /SAVE AC ON INT
ADCOUT  ADST            /PC,L,EX,MP
        ADINT
        LAC     (JMP    ADSTOP  /OVERWRITE SETUP
        DAC     SETUP
ADSTOP  DZM     ADUND   /NON-0 WHEN BUSY
ADWAIT  LAC     ADUND
        SNA
        JMP     ADOK
ADBUSY  DBR             / I/O UNDERWAY LOOP
        XCT     .+1
        XCT     .+1
        JMP*    ADCALP
ADREAD  LAC     ADUND   /CHECK I/O UNDERWAY
        SZA!CMA         /BUSY FLAG?
        JMP     ADBUSY
        DAC     ADUND   /NO - SET BUSY FLAG
        LAC*    ADARGP  /GET BUFFER ADDRESS
        TAD     (-1     /DATABREAK REQUIRES ADDR-1
        DAC*    (27     /NO HEADER WORD
        DAC     ADBA
        ISZ     ADARGP
        LAC*    ADARGP  /GET WORD COUNT
        DAC*    (26
        DAC     ADWC
        ISZ     ADARGP  /POINT TO EXIT
        ADWI            /WRITE INIALISE
/               ###############
/NORMAL RETURN FROM CAL
ADOK    DBR
```

```
        XCT     .+1
        XCT     .+1
        JMP*    ADARGP  /RETURN AFTER CAL
/               ################
/               INTERRRUPT HANDLER
/               ################
ADPIC   DAC     ADSAC   /SAVE AC
        LAC*    (0)     /SAVE PC,L,EX,MP
        DAC     ADCOUT
        LAC     ADION   /FORCE ION AT DISMISSAL
        JMP     ADSTON
ADINT   JMP     ADPIC   /PIC ENTRY
        DAC     ADSAC
        LAC     ADINT   /SAVE PC,L,EX,MP
        DAC     ADCOUT
        LAC     (JMP ADPIC)     /RESTORE PIC ENTRY
        DAC     ADINT
        IORS    /CHECK STATUS OF PIC
        SMA!CLA
        LAW     17740   /REBUILD FOR RESTORATION
ADSTON  TAD     ADION   /AT EXIT
        DAC     ADSWCH
/               #####################
/CHECKS SOURCE OF INTERRUPT FLAG
/               #####################
        ADRS            /BIT 15 OFLO
        AND     (4
        SAD     (4      /BIT 16 T ERR
        ADCO            /CLEAR OFLO FLAG
        ADRS
        AND     (2
        SAD     (2
        SKP
        JMP     ADDISM
/ TIMING ERROR FLAG IS SET.
/ DO MONITOR ERROR ROUTINE EXIT AFTER CLEARING FLAGS
        ADCT            /CLEAR T ERR FLAG
        DZM     ADUND   /CLEAR BUSY FLAG
        JMP     ADERR4
/               ###################
/INTERRUPT HANDLER DISMISSAL ROUTINE
/               ###################
ADDISM  DZM     ADUND   /CLEAR BUSY FLAG
        LAC     ADSAC   /RESTORE AC
ADSWCH  ION
        DBR
        XCT     .+1
        XCT     .+1
        JMP*    ADCOUT
/
ADCALP  0       /L,EM,MP,+CAL POINTER
ADION   ION
ADARGP  0       /CAL ARGUMENT POINTER
/               ###################
/               IGNORED FUNCTIONS
/
ADIGN   ISZ     ADARGP  /BYPASS FILE POINTER
        JMP     ADOK
/
/ CLOSE MACRO
ADCLOS  DZM     ADUND   /CLEAR BUSY FLAG
        JMP     ADOK
        .END

        .TITLE DLOG2
/               ########################
/READS ADC ON ENCODE COMMAND WITH THE SEQUENCE: 4 SAMPLES,
/21 SAMPLES, REPEAT. LOWEST ORDER DATA BIT IS ERASED. TWO
/DATA POINT PACKED PER WORD. OUTPUTS 252 WORD BLOCKS (18 SETS)
/TO DECTAPE ON .DAT 2.ALTERNATES STORAGE BUFFERS. FILENAME
/STARTS AS DATAFI DAT AND IS INCREMENTED IF ALREADY PRESENT AND
/USER REQUESTS. A SET HAS THE FORMAT: I.D., 4 NOISE FRAMES, 22
/DATA FRAMES, PACKED WITH TWO  FRAMES PER WORD.
/DATATAPE ON DT4 BY DEFAULT
/       INCLUDES TIMER OPTION
/               ###########################
        .IODEV 4,6,2,3
START   .INIT   4,0,RESTAR      /TT IN
        .INIT   6,1,RESTAR      /TT OUT
        .INIT   3,0,RESTAR      /INITIALISE ADC
        .INIT   2,1,RESTAR      /DT OUT
        JMS     INIT
RD      .READ   3,4,BUF,26      /26 SAMPLES READ ON INT
/ EXTERNAL ENCODE PULSE DEFINES TIME OF SAMPLES
```

```
        LAC     TRANF   /IS BUF1 FULL?
        SZA
        JMS     DTRANS  /THEN TRANSFER IT TO DTAPE
/RSUB ENTERED AT MAINSTREAM LEVEL
        .WAIT   3
        LAC     TRANF   /STILL SET?
        SZA
        HLT     /THEN READ BEFORE WROTE
        JMS     RSUB
        JMP     RD
BUF     .BLOCK  36
NAME    .SIXBT  "DATAF1DAT"     /NAME OF FILE TO STORE DATA
INIT    0
        LAC     (BUF1
        DAC     POINT
        DAC     TBUF
        DZM     IDCOU#
        DZM     TRANF#
/OPEN DECTAPE FILE TO RECEIVE DATA
UP      .FSTAT  2,NAME  /IS FILE PRESENT?
        SZA
        JMP     UPDATE  /YES
WRITE   .ENTER  2,NAME
        JMS     DTRANS  /WRITE DUMMY BLOCK
        JMS     DTRANS  /TWICE
        JMP*    INIT
/               ########################
/ENQUIRES WHETHER NEW FILE OR OVERWRITE OLD ONE.
/               ########################
UPDATE  .WRITE  6,2,MSG1,34     /TT OUT, ASCII
        .WAIT   6
        .WRITE  6,2,MSG2,34
        .WAIT   6
        .READ   4,2,COM,6       /READ RESPONSE
        .WAIT   4
        LAC     COM+2   /GET FIRST CHAR
        AND     (774000         /MASK SEVEN BITS
        SAD     (544000         /IS CHAR A Y?
        SKP
        JMP     WRITE
        ISZ     NAME+1  /YES, INCREMENT NAME
        JMP     UP
MSG1    MSG2-MSG1/2*1000
        0
        .ASCII  "FILE ALREADY "
        .ASCII  "PRESENT !!"<15>
MSG2    COM-MSG2/2*1000
        0
        .ASCII  "DO YOU WISH TO KEEP IT?"
        .ASCII  "(Y OR NO) AND CR."<15>
COM     .BLOCK  10              /REPLY GOES HERE
/               ########################
/ ENTERED AFTER ADC DATABREAK
/IS COMPLETED BY AN INTERRUPT. ISSUES SUBSEQUENT READ PARAMETERS
/AND STORES DATA INPUT.
/               ########################
RSUB    0
        DAC     SAVEAC#
NOISE   ISZ     IDCOU   /PUTS I.D. IN NEXT LOCATION
        LAC     IDCOU
        DAC*    POINT
        ISZ     POINT
DATA    LAC     (-15    /EVEN NUMBER OF DATA SAMPLES.LAST IS ARBIT.
        JMS     PAC
        LAC     SAVEAC
        JMP*    RSUB
/               ##########################
/PACK TWO DATA FRAMES INTO ONE WORD. REMOVE LOWEST ORDER BIT AND
/ROTATE. PLACES INPUT DATA IN BUF1 AND BUF2. WHEN ONE BUFFER IS FULL
/TRANSFER IT WHILE FILLING OTHER.
/               ##########################
PAC     0
        DAC     COUNTP# /STORE SAMPLE NUMBER
        LAC     (BUF-1
        DAC     BPOINT# /BUF POINTER
PAKING  ISZ     BPOINT  /DATA STARTS AT BUF
        LAC*    BPOINT  /GET INPUT DATA WORD
        AND     (1776   /LOOSE BIT 17
        .REPT   4       /ROTATE TO LHS
        CLL!RTL
        DAC*    POINT   /STORE
        ISZ     BPOINT
        LAC*    BPOINT  /GET NEXT INPUT DATA WORD
```

```
        AND     (1776
        CLL!RAR
        TAD*    POINT   /PACK IN PREVIOUS ONE
        DAC*    POINT   /STORE IN BUFI
        ISZ     POINT
        ISZ     COUNTP  /ALL DONE?
        JMP     PAKING  /NO
        LAC*    POINT   /IS CURRENT BUFFER FULL?
        SAD     SEVN
        SKP
        JMP*    PAC     /NO
        ISZ     POINT   /YES
        LAC*    POINT
        DAC     TBUF#   /PASS NAME OF FULL BUF TO DTRANS
        DAC     TRANF   /SET TRANSFER FLAG
        ISZ     POINT
        LAC*    POINT
/SET POINTER TO NEW BUFFER
        DAC     POINT
        JMP*    PAC
        .DEC
BUFI    .BLOCK  252
SEVN    77777
        .DSA    BUFI
        .DSA    BUF2
BUF2    .BLOCK  252
        77777
        .DSA    BUF2
        .DSA    BUFI
POINT   .DSA    SEVN
        .OCT
/               ################################
/ROUTINE TO TRANSFER FILLED DATA BUFFER TO DECTAPE
/               ################################
DTRANS  0
        LAC     TBUF
        DAC     .+3
        .WRITE  2,4,0,252
/DUMP BUFI TO OPENED DECTAPE FILE
        .WAIT   2
        DZM     TRANF
        JMS     TIMER
        JMP*    DTRANS
/       ####################################
/ OPTION INITIATED BY SETTING AC SWITCHES BITS 0,1.
/ FILE CLOSED IF REST OF SWITCHES IDENTICAL TO NUMBER OF
/ SECONDS OF RUN AT TIME OF DTRANS CALL.
/       ########################################
TIMER   0
        LAS
        AND     (600000
        SAD     (600000
        SKP
        JMP*    TIMER
        LAS
        AND     (177777
        CMA!CLL
        TAD     IDCOU
        CLL!SZL
        JMP     RESTAR
        JMP*    TIMER
/               ################################
/↑P CLOSE ROUTINE. TERMINATES RECORDING. AN INCOMPLETE BUFFER
/IS ERASED.
/               ################################
RESTAR  LAC     TRANF   /COMPLETE TRANSFER OF BLOCK IF IN PROGRESS
        SZA
        JMS     DTRANS
        .WRITE  2,4,SEVN,2      /WRITE EOF ID#
        .WAIT   2
        .CLOSE  4
        .CLOSE  3
        .CLOSE  2       /CLOSE OUTPUT FILE
        JMP     START
        .END    START

        .TITLE DLOG4
/       ########################
/       B/F     SERVICE ROUTINE METHOD
/       INCLUDES AC SWITCH TEST (TIMER)
/READS ADC ON ENCODE COMMAND WITH THE SEQUENCE: 4 SAMPLES,
/21 SAMPLES, REPEAT. LOWEST ORDER DATA BIT IS ERASED. TWO
/DATA POINT PACKED PER WORD. OUTPUTS 252 WORD BLOCKS (18 SETS)
/TO DECTAPE ON DAT 2. SINGLE BUFFER VERSION. FILENAME
```

```
/STARTS AS DATAFI DAT AND IS INCREMENTED IF ALREADY PRESENT AND
/USER REQUESTS. A SET HAS THE FORMAT: I.D., 4 NOISE FRAMES, 22
/DATA FRAMES, PACKED WITH TWO  FRAMES PER WORD.
/DATATAPE ON DT4 BY DEFAULT
/               ##############################
        .IODEV 4,6,2
START   .INIT   4,0,RESTAR      /TT IN
        .INIT   6,1,RESTAR      /TT OUT
        .INIT   2,1,RESTAR      /DT OUT
        JMS     INIT
        .IDLE
BUF     .BLOCK  200
NAME    .SIXBT  "DATAFIDAT"     /NAME OF FILE TO STORE DATA
INIT    0
        LAC     (BUF1
        DAC     POINT
        DZM     IDCOU#
        DZM     TRANF#
/OPEN DECTAPE FILE TO RECEIVE DATA
UP      .FSTAT  2,NAME  /IS FILE PRESENT?
        SZA
        JMP     UPDATE  /YES
WRITE   .ENTER  2,NAME
        JMS     DTRANS  /WRITE DUMMY BLOCK
        JMS     DTRANS  /TWICE
        JMS     ADINIT  /READY FOR ENCODE SIGNAL
        JMP*    INIT
/               ##########################
/ENQUIRES WHETHER NEW FILE OR OVERWRITE OLD ONE.
/               ##########################
UPDATE  .WRITE  6,2,MSG1,34     /TT OUT, ASCII
        .WAIT   6
        .WRITE  6,2,MSG2,34
        .WAIT   6
        .READ   4,2,COM,6       /READ RESPONSE
        .WAIT   4
        LAC     COM+2   /GET FIRST CHAR
        AND     (774000         /MASK SEVEN BITS
        SAD     (544000         /IS CHAR A Y?
        SKP
        JMP     WRITE
        ISZ     NAME+1  /YES, INCREMENT NAME
        JMP     UP
MSG1    MSG2-MSG1/2*1000
        0
        .ASCII  "FILE ALREADY "
        .ASCII  "PRESENT !!"<15>
MSG2    MSG3-MSG2/2*1000
        0
        .ASCII  "DO YOU WISH TO KEEP IT?"
        .ASCII  "(Y OR NO) AND CR."<15>
MSG3    COM-MSG3/2*1000
        0
        .ASCII  "DEVICE TIMING ERROR, "
        .ASCII  "CTL P TO RESTART"<15>
COM     .BLOCK  10              /REPLY GOES HERE
/               ##########################
/ ENTERED AFTER ADC DATABREAK
/IS COMPLETED BY AN INTERRUPT. ISSUES SUBSEQUENT READ PARAMETERS
/AND STORES DATA INPUT.
/               ##########################
RSUB    0
        DAC     SAVEAC
NOISE   ISZ     IDCOU   /PUTS I.D. IN NEXT LOCATION
        LAC     IDCOU
        DAC*    POINT
        ISZ     POINT
DATA    LAC     (-15    /EVEN NUMBER OF DATA SAMPLES.LAST IS ARBIT.
        JMS     PAC
        LAC     (-32
        DAC*    (26     /SET WORD COUNT FOR NOISE
        LAC     (BUF-1
        DAC*    (27
EX      LAC     TRANF   /IS BUF1 FULL?
        SZA
        JMS     DTRANS  / TRANSFER  TO DECTAPE
        LAC     SAVEAC
        .RLXIT  RSUB
/               ############################
/PACK TWO DATA FRAMES INTO ONE WORD. REMOVE LOWEST ORDER BIT AND
/ROTATE. PLACES INPUT DATA IN BUF1 AND BUF2. WHEN ONE BUFFER IS FULL
/TRANSFER IT WHILE FILLING OTHER.
/               ############################
PAC     0
```

```
        DAC     COUNTP# /STORE SAMPLE NUMBER
        LAC     (BUF-1
        DAC     BPOINT# /BUF POINTER
PAKING  ISZ     BPOINT  /DATA STARTS AT BUF
        LAC*    BPOINT  /GET INPUT DATA WORD
        AND     (1776   /LOOSE BIT 17
        .REPT   4       /ROTATE TO LHS
        CLL!RTL
        DAC*    POINT   /STORE
        ISZ     BPOINT
        LAC*    BPOINT  /GET NEXT INPUT DATA WORD
        AND     (1776
        CLL!RAR
        TAD*    POINT   /PACK IN PREVIOUS ONE
        DAC*    POINT   /STORE IN BUF1
        ISZ     POINT
        ISZ     COUNTP  /ALL DONE?
        JMP     PAKING  /NO
        LAC*    POINT   /IS CURRENT BUFFER FULL?
        SAD     SEVN
        SKP
        JMP*    PAC     /NO
        ISZ     POINT   /YES
        LAC*    POINT
        DAC     TRANF   /SET TRANSFER FLAG
        ISZ     POINT
        LAC*    POINT
/SET POINTER TO NEW BUFFER
        DAC     POINT
        JMP*    PAC
        .DEC
BUF1    .BLOCK  252
SEVN    77777
        .DSA    BUF1
        .DSA    BUF1
POINT   .DSA    SEVN
        .OCT
/               ##############################
/ROUTINE TO TRANSFER FILLED DATA BUFFER TO DECTAPE
/               ##############################
DTRANS  0
        .WRITE  2,4,BUF1,252
/DUMP BUF1 TO OPENED DECTAPE FILE
        .WAIT   2
        DZM     TRANF
        JMS     TIMER
        JMP*    DTRANS
/               ##############################
/ OPTION INITIATED BY SETTING AC SWITCHES BIT 0+1.
/ FILE CLOSED IF REST OF SWITCHES IDENTICAL TO NUMBER OF
/ SECONDS OF RUN AT TIME OF DTRANS CALL.
/               ##############################
TIMER   0
        LAS
        AND     (600000 /BITS 0 AND 1 SET?
        SAD     (600000
        SKP
        JMP*    TIMER
        LAS
        AND     (177777
        CMA!CLL         /COMPLIMENT AND T.ADD
        TAD     IDCOU   /SKIP IF IDCOU.GE.LAS
        CLL!SZL         /IE:SKIP IF L .NE.0
        JMP     RESTAR
        JMP*    TIMER
/               ##############################
/↑P CLOSE ROUTINE. TERMINATES RECORDING. AN INCOMPLETE BUFFER
/IS ERASED.
/               ##############################
RESTAR  LAC     (NOP
        DAC     ADIOT   /DISABLE INTERFACE
        LAC     TRANF
        SZA
        JMS     DTRANS
        .WRITE  2,4,SEVN,2      /WRITE EOF ID#
        .WAIT   2
        .CLOSE  4
        .CLOSE  2       /CLOSE OUTPUT FILE
        JMP     START
/################################################################
/               I/O SERVICE ROUTINE
/################################################################
ADSO=703701     /SKIP ON OVERFLOW FLAG
```

```
ADST=703721     /SKIP ON DEVICE TIMING ERROR
ADCO=703704     /CLEAR OVERFLOW FLAG
ADCT=703744     /CLEAR DEVICE  T'ERR FLAG
ADWI=703724     /WRITE INITIALISE
.SCOM=100
ADINIT  0
        LAC*    (.SCOM+55
.SETUP  DAC     .       /SETUP INTERRUPT TRAP ADDRESS
TMP1    LAC*    (.SCOM+51
REALTP  DAC     .       /REAL TIME PROCESSOR SUBR
AC0     LAC     (400010 /API 4
SAVEAC  ISA
        JMS*    .SETUP
        ADSO            /PARAMETERS PASSED TO .SETUP
        ADINT
        DBK             /FROM API4
        LAC     (JMP PASS       /OVERWRITE SETUP
        DAC     ADINIT+1        /ROUTINE
PASS    LAC     (-32    /SET WORD COUNT
        DAC*    (26
        LAC     (BUF-1  /AND BUFFER ADDR
        DAC*    (27     /FOR FIRST INTERRPT
        LAC     .+2
        DAC     ADIOT
        ADWI            /INIIALISE INTERFACE
        JMP*    ADINIT
/------------------------------------------------
/               INTERRUPT SERVICE ROUTINE
/------------------------------------------------
ADINT   0
        DAC     AC0     /INSERT DBA FOR V3A
        ADSO
        JMP     TERR
        ADCO
        LAC     (RSUB+500000    /API 5
        JMP     RUNIT
TERR    ADCT            /CLEAR T'ERR
        LAC     (RSUB2+600000   /API 6
RUNIT   JMS*    REALTP  /EXECUTE NAMED SUBROUTINES
                        /RSUB OR RSUB2
/EXIT SEQUENCE FOR INTERRUPT
        LAC     (404000 /REQUEST API 4
        ISA
        LAC     AC0
ADIOT   ADWI            /INITIALISE INTERFACE
        DBR
        JMP*    ADINT
/------------------------------------------------
/                       REALTIME SUBROUTINE TO DEAL WITH
/ DEVICE TIMING ERROR: TERMINAL
/------------------------------------------------
RSUB2   0
        DAC     SAVEAC
        .WRITE  6,2,MSG3,34
        .WAIT   6
        LAC     (NOP    /CANCEL INTERFACE INITIALISE
        DAC     ADIOT
        LAC     SAVEAC
        .RLXIT  RSUB2
        .END    START
```

```
C###########################################################
C PROC2 PROCESSES GROUND BASED PARTIAL REFLECTION DATA RECORDED
C ON MAGNETIC TAPE AND OUTPUTS ELECTRON DENSITY PROFILES AS
C 3 MINUTE AVERAGES AND MEDIANS, TO TELETYPE.
C ASSUMES SEN-WYLLER Q.L. APPROXIMATION AND EQUINOX COLLISION
C FREQUENCY PROFILE.
C HEIGHT RANGE: 60 -90 KM
C INTERVAL: 1.5 KM
C DATA ON .DAT SLOT 2
C###########################################################
      REAL MATA(2,2,24)
      LOGICAL IHTF,IPROF
      DIMENSION FNAM(2),ISIZ(2),BNOIS(22),ARRAY(22)
      DIMENSION DATA(920,2),P(21),RA(3),RM(3),CF(3)
      DIMENSION NOIS(2),GEN(30),LL(3)
      COMMON //DATA,MATA,ARRAY,BNOIS,P
      ASSIGN 220 TO MESS
C-----------------------------------------------------------
C EQUINOX COLLISION FREQUENCY PROFILE SET UP AS A DATA ARRAY
C FOR THE HEIGHTS 60-90 KM. IN 1.5 KM INCREMENTS
C
C-----------------------------------------------------------
      P(1)=265.
      P(2)=170.
      P(3)=135.
      P(4)=103.
      P(5)=79.5
      P(6)=68.
      P(7)=50.
      P(8)=34.5
      P(9)=31.
      P(10)=24.5
      P(11)=19.5
      P(12)=15.
      P(13)=11.8
      P(14)=8.7
      P(15)=6.75
      P(16)=5.15
      P(17)=3.95
      P(18)=3.1
      P(19)=2.35
      P(20)=1.75
      P(21)=1.35
      DO 1 I=1,21
      P(I)=P(I)*(10.**5)
1     CONTINUE
C-----------------------------------------------------------
C ASK FILE NAME AND PARAMETERS
C-----------------------------------------------------------
2     WRITE(6,5)
5     FORMAT(15H NAME DATAFILE )
      READ (4,10) FNAM
10    FORMAT(A5,A4)
      CALL DINIT
      CALL FSTAT(2,FNAM,LOG)
      IF (LOG.NE.0) GO TO 20
C FILE NOT FOUND
      WRITE(6,15) FNAM
15    FORMAT(6H FILE ,A5,A4,19H NOT FOUND ON DAT 2)
      GO TO 2
20    WRITE(6,21)
21    FORMAT(24H PROFILE ONLY ( T OR F ))
      READ (4,22) IPROF
22    FORMAT(1L1)
      IF(.NOT.IPROF) GOTO 23
      KHT=0
      GOTO 31
23    WRITE(6,25)
25    FORMAT(56H GIVE HT CODE: 0 FOR ALL 20, 1 FOR 60-61.5 INTERVAL
     1, ETC)
      READ(4,30) KHT
30    FORMAT(I2)
31    WRITE(6,32)
32    FORMAT(47H GIVE MINIMUM ACCEPTABLE SIGNAL TO NOISE
     1 RATIO.,/32H THEN MAX ACCEPTABLE NOISE LEVEL)
      READ(4,33) SNR,ANOIS
33    FORMAT(F8.3)
      WRITE(6,34) SNR,ANOIS
34    FORMAT(2XF8.3,2X,F8.3)
      IF(KHT) 40,35,40
35    MHT=20
      KHT=1
      LBOUND=5
      GO TO 50
```

```
40      MHT=KHT
        LBOUND=KHT+4
50      IHTF=.FALSE.
C===================================================
C END INITIALISATION, BEGIN OVERALL DO LOOP, REPEATED
C FOR EACH HEIGHT PROCESSED
C===================================================
        DO 400 JHT=KHT,MHT
        IF(IHTF) GO TO 54
52      MCO=1
        GO TO 60
54      MCO=2
60      DO 300 KCO=MCO,2
C"""""""""""""""""""""""""""""""""""""""""""""""""""
C OBTAIN ORDINARY AND XORDINARY DATA ARRAY FROM TAPE AT
C GIVEN HEIGHT, FOR EACH KCO ISSUED
C SNR   SIGNAL TO NOISE RATIO
C JHT   HEIGHT CODE OF REQUIRED HT INTERVAL
C ISIZ()        SIZE OF  DATA ARRAY
C LBOUND        UNPACKING CODE
C BNOIS()       MEAN NOISE DURING GIVEN TIME INTERVAL
C 900           ASSUMED  SIZE OF RUN
C NOIS()        # OF REJECTED DATA POINTS
C"""""""""""""""""""""""""""""""""""""""""""""""""""
90      LED=1
        ID=0
        NOIS(1)=0
        NOIS(2)=0
        LBOUND=LBOUND+1
95      CALL SEEK(2,FNAM)
        LEND=22
C LEND WILL BE CHANGED BY DREAD WHEN EOF ENCOUNTERED
100     CALL DREAD(LEND,LBOUND,ARRAY,BNOIS,MESS,ID,KEOF)
C"""""""""""""""""""""""""""""""""""""""""""""""""""
C       NOISE DISCRIMINATION AND O X SORTING
C 1. MEAN NOISE (BNOIS) SHOULD BE .LE. ANOIS
C 2. S/N RATIO SHOULD BE .GE. SNR
C ELSE INCREMENT NOISE COUNT (NOIS)
C"""""""""""""""""""""""""""""""""""""""""""""""""""
        DO 150 KEL=1,LEND,2
        RES=ARRAY(KEL)/BNOIS(KEL)
        IF(BNOIS(KEL).GT.ANOIS) GO TO 120
        IF(RES.GE.SNR) GO TO 130
120     NOIS(1)=NOIS(1)+1
        DATA(LED,1)=0
        GO TO 135
130     DATA(LED,1)=ARRAY(KEL)
C REPEAT FOR X
135     RES=ARRAY(KEL+1)/BNOIS(KEL+1)
        IF(BNOIS(KEL+1).GT.ANOIS) GO TO 140
        IF(RES.GE.SNR) GO TO 145
140     NOIS(2)=NOIS(2)+1
        DATA(LED,2)=0
        GO TO 148
145     DATA(LED,2)=ARRAY(KEL+1)
148     LED=LED+1
150     CONTINUE
C COMPLETES RECOVERY OF 11 DATAPOINTS EACH OF O AND X,GIVEN HT
        IF(LED-900) 160,160,180
160     IF(KEOF.EQ.0) GO TO 100
C END OF FILE ENCOUNTERED
        ISIZ(KCO)=LED-1
162     IF(IHTF.AND.IPROF) GOTO  215
        WRITE(6,165) ISIZ(KCO)
165     FORMAT(11H ARRAY SIZE ,I4)
        GO TO 200
180     ISIZ(KCO)=900
C DUMMY LOOP TO ENSURE FILE CLOSED PROPERLY
181     CALL DREAD(LEND,LBOUND,ARRAY,BNOIS,MESS,ID,KEOF)
        IF(KEOF) 162,181,162
C"""""""""""""""""""""""""""""""""""""""""""""""""""
C               ARRAYS NOW AVAILABLE
C DATA(ISIZ,1)  ORDINARY MODE AT GIVEN HT
C DATA(ISIZ,2)  XORDINARY MODE SAME HT
C ISIZ()        SIZE OF THESE ARRAYS
C"""""""""""""""""""""""""""""""""""""""""""""""""""
200     WRITE(6,210) NOIS(1), NOIS(2),SNR,ANOIS
210     FORMAT(6H NOISE ,2I6,/11H S/N RATIO ,F8.3,/13H
       1 NOISE LEVEL ,F8.3//)
215     CALL CLOSE(2)
        GO TO 250
220     WRITE(6,230) ID
230     FORMAT(17H SEQUENCE NUMBER ,I3, 6H ERROR)
```

```
      KEOF=0
      GO TO 2
C------------------------------------------
C FINDS THE AVERAGE AND MEDIAN OF ARRAY DATA IN 90 ELEMENT
C AND ALL ELEMENT GROUPS AND PLACES THE RESULTS IN ARRAY:
C MATA(1,1,1-22)          ORD,LOWER HT,
C MATA(1,2,1-22)          XORD,LOWER HT,
C MATA(2,1,1-22)          ORD,UPPER HT,
C MATA(2,2,1-22)          XORD,UPPER HT,
C WHERE THIRD DIMENSION ALTERNATES AVERAGE AND MEDIAN
C-----------------------------------------------
250   NRUN=ISIZ(KCO)/90
      MRUN=2*NRUN-1
      NRUN0=2*NRUN+1
      DO 280 LOD=1,2
      IHAL=0
      TOTO=0
      DO 270 I=1,MRUN,2
      IHAL=IHAL+1
      NMIN=90*(IHAL-1)+1
      NMAX=90*IHAL
      MATA(KCO,LOD,I)=AVERA2(DATA,LOD,NMIN,NMAX,SO)
      TOTO=SO+TOTO
      MATA(KCO,LOD,I+1)=EMED2(DATA,LOD,NMIN,90,MATA(KCO,LOD,I))
270   CONTINUE
C     90 ELEMENT GROUPS DONE
      R=NRUN*90
      MATA(KCO,LOD,NRUN0)=TOTO/R
      IDIV=NRUN*90
      MATA(KCO,LOD,NRUN0+1)=EMED2(DATA,LOD,1,IDIV,MATA(KCO,LOD,NRUN0))
C     OVERALL GROUP DONE
280   CONTINUE
300   CONTINUE
      IF(ISIZ(1)-ISIZ(2))305,310,305
305   WRITE(6,306)
306   FORMAT(11H SIZE ERROR)
      GO TO 2
C-----------------------------------------------------
C     ALL FOUR ARRAYS NOW AVAILABLE
C GET RATIOS AND ELECTRON DENSITIES FOR BOTH AVE AND MED
C-----------------------------------------------------
310   HT=JHT
      RHT=HT*1.5+58.5
      SHT=RHT+1.5
      CF(1)=P(JHT)
      CF(2)=P(JHT+1)
      IF(IHTF.AND.IPROF) GOTO 341
      IF(IPROF) GOTO 325
      WRITE(6,315)RHT,SHT
315   FORMAT(7X,16HHEIGHT INTERVAL ,F6.1,2XF6.1,3H KM)
      WRITE(6,316) (CF(I),I=1,2)
316   FORMAT(3X,19HCOLLISION FREQUENCY,E10.2,2X,E10.2)
      WRITE(6,320)
320   FORMAT(1X21HTHREE MINUTE AVERAGES,2X20HTHREE MINUTE MEDIANS
     1,4X22H THREE MINUTE AVERAGES)
325   WRITE(6,330)
330   FORMAT(1X21HAX/AO AX/AO ELECTRON,2X,
     120HAX/AO AX/AO ELECTRON,4X22HAOLO  AOHI  AXLO  AXHI)
      WRITE(6,340)
340   FORMAT(1X20H LO    HI    DENSITY,2X20H LO    HI    DENSITY)
C     MRUN SET FOR ODD NUMBER-LAST 3 MIN AV RECORDED
341   DO 350 I=1,MRUN,2
      RA(1)=MATA(1,2,I)/MATA(1,1,I)
      RA(2)=MATA(2,2,I)/MATA(2,1,I)
      RM(1)=MATA(1,2,I+1)/MATA(1,1,I+1)
      RM(2)=MATA(2,2,I+1)/MATA(2,1,I+1)
C     PROTECT AGAINST LOG 0
      IF(RA(1)*RA(2)*RM(1)*RM(2).GT.0.) GO TO 342
      ELAV=0.
      ELME=0.
      GO TO 343
342   ELAV=ELDEN(RA,CF,1.5E+3)/(10.**9)
      ELME=ELDEN(RM,CF,1.5E+3)/(10.**9)
343   IF(IPROF) GOTO 350
      WRITE(6,345) RA(1),RA(2),ELAV,RM(1),RM(2),ELME,
     1((MATA(L,J,I),L=1,2),J=1,2)
345   FORMAT(2(1XF5.2),1XF7.2,3X2(1XF5.2),1XF7.2,4X4(F5.1,1X))
350   CONTINUE
351   MINIT=NRUN*3
      IF(IPROF) GOTO 353
      WRITE(6,352) MINIT
352   FORMAT(/5X18HOVERALL VALUES FOR,I3,11H MINUTE RUN/)
353   RA(1)=MATA(1,2,NRUN0)/MATA(1,1,NRUN0)
      RA(2)=MATA(2,2,NRUN0)/MATA(2,1,NRUN0)
```

```
      RM(1)=MATA(1,2,NRUN0+1)/MATA(1,1,NRUN0+1)
      RM(2)=MATA(2,2,NRUN0+1)/MATA(2,1,NRUN0+1)
      IF(RA(1)*RA(2)*RM(1)*RM(2).GT.0.) GOTO 420
      ELAV=0.
      ELME=0.
      GOTO 354
420   ELAV=ELDEN(RA,CF,1.5E+3)/(10.**9)
      ELME=ELDEN(RM,CF,1.5E+3)/(10.**9)
354   WRITE(6,355) RA(1),RA(2),ELAV,RM(1),RM(2),ELME,
      1((MATA(L,J,NRUN0), L=1,2), J=1,2)
355   FORMAT(2(1XF5.2),1XF7.2,3X2(1XF5.2),1XF7.2,4X4(F5.1,1X))
      IF(IPROF) GOTO 358
      WRITE(6,357)
357   FORMAT(1H1)
358   MRUN=MRUN+3
C     MRUN=NRUN*2+2, LAST PAIR ARE OVERALL AV,OVERALL MED
      DO 360 I=1,MRUN
      MATA(1,1,I)=MATA(2,1,I)
      MATA(1,2,I)=MATA(2,2,I)
360   CONTINUE
      INTF=.TRUE.
400   CONTINUE
410   INTF=.FALSE.
      GO TO 2
      STOP 1
      END
```

```
C       WLOGI, WRITES DATA FROM DECTAPE TO TELETYPE
C ASKS ELEMENT (IHT) OF UNPACKED DATA ARRAY WHICH
C IS TO BE TYPED. GETS THIS ELEMENT FROM EACH DATA
C SET, 10 SETS AT A TIME. WRITES SET SEQ ID # TOO
C NOTE IHT IS HEIGHT CODE
        DIMENSION FNAM(2),DATA(10),BMEAN(10)
        ASSIGN 200 TO IERR
10      CALL DINIT
C       PERMITS LOADING OF DUMPR FROM .LIBR5
        WRITE(6,20)
20      FORMAT(15H WHICH DATAFILE)
        READ(4,30) FNAM
30      FORMAT(2A5)
        CALL FSTAT(2,FNAM,LOG)
        IF(LOG.NE.0) GO TO 40
        WRITE(6,35) FNAM
35      FORMAT(6H FILE ,2A5,19H NOT FOUND ON DAT 2)
        GO TO 10
40      CALL SEEK(2,FNAM)
        WRITE(6,50)
50      FORMAT(1H1,17H GIVE HEIGHT CODE)
        READ(4,60) IHT
60      FORMAT(I3)
        IHT=IHT+5
        ID=0
        DO 100 I=1,200
        NSET=10
        CALL DREAD(NSET,IHT,DATA,BMEAN,IERR,ID,KEOF)
        WRITE(6,70) ID,(DATA(J),J=1,NSET)
70      FORMAT(I5,10(1X,F5.1))
        IF(KEOF) 10,100,10
100     CONTINUE
200     WRITE(6,210)
210     FORMAT(12H DREAD ERROR)
        GO TO 10
        STOP 1
        END

C WLOG2 WRITES CONTENTS OF NAMED DATAFILE OFF OF DECTAPE
        DIMENSION IDATA(30),FNAM(2)
10      CALL DINIT
        WRITE(6,20)
20      FORMAT(15H WHICH DATAFILE)
        READ(4,30) FNAM
30      FORMAT(2A5)
        CALL FSTAT(2,FNAM,LOG)
        IF(LOG.NE.0) GO TO 40
        WRITE(6,35) FNAM
35      FORMAT(6H FILE ,2A5,19H NOT FOUND ON DAT 2)
        GO TO 10
40      CALL SEEK(2,FNAM)
50      DO 100 IO=1,200
        CALL DUMPR(IDATA,NEGF)
        WRITE(6,60) (IDATA(I),I=1,26)
60      FORMAT(1X,10I5)
        IF(IDATA(1).GE.77777) GO TO 10
100     CONTINUE
        GO TO 10
        STOP 1
        END
```

```
C NCALC CALCULATES ELECTRON DENSITIES FORM USER SUPPLIED
C DATA USING SUBROUTINE ELDEN
      DIMENSION AXBYAO(3),CF(3)
1     WRITE(6,2)
2     FORMAT(12H HT INTERVAL)
      READ(4,20) DEL
      DEL=DEL*10.**3
9     WRITE(6,10)
10    FORMAT(19H AX/AO  LO,HI, F6.3)
      READ(4,20)(AXBYAO(I),I=1,2)
20    FORMAT(F6.3)
      IF(AXBYAO(1)) 25,1,25
C ENQUIRE HT INTERVAL AGAIN IF NO RATIO TYPED
25    WRITE(6,30)
30    FORMAT(27H COLL FREQ  TIMES E+7  F6.3)
      READ(4,20)(CF(I),I=1,2)
      CF(1)=CF(1)*10.**7
      CF(2)=CF(2)*10.**7
      IF(AXBYAO(1)*AXBYAO(2)*CF(1)*CF(2).EQ.0.) GO TO 1
      EL=ELDEN(AXBYAO,CF,DEL)
      WRITE(6,50) (AXBYAO(I),I=1,2),(CF(I),I=1,2),DEL,EL
50    FORMAT(//1X2F8.3,2X2E10.2,2X,2E10.2//)
      GO TO 9
      STOP 1
      END
```