THE ESTABLISHMENT OF A LIDAR FACILITY AT RHODES UNIVERSITY

THESIS

Submitted in Fulfilment of the

Requirements for the Degree of

MASTER OF SCIENCE

of Rhodes University

by

RICHARD PETER JAMES SETON GRANT

January 1988

TABLE OF CONTENTS

CHAPTER 1

An Overview of LIDAR

CHAPTER 2

Choice of Transmitter and Receiver and Set-up Procedure

CHAPTER 3

Design, Construction and Testing of the Pulse Counting System

CHAPTER 4

Design and Testing of the Pulse Amplifier

CHAPTER 5

The Photomultiplier Tube Tester and Evaluator

CHAPTER 6

Performance of the DL 2100B Laser and Alignment with the Receiver

CHAPTER 7

Operation of the Rhodes LIDAR

CHAPTER 8

LIDAR Theory and Algorithms for the Analysis of LIDAR Data

CHAPTER 9

Results Obtained from the LIDAR Data

CHAPTER 10

Conclusion and Suggestions for Future Work

APPENDICES

APPENDIX 1. LIDAR Operating Skeleton Program

APPENDIX 2. Machine Code Routines called by GIANT.BAS

APPENDIX 3. PMT Non-linearity Compensation Calculations

APPENDIX 4. Temperature Calculations

APPENDIX 5. Scattering Ratio Calculations

ACKNOWLEDGEMENTS

I would like to thank the Foundation for Research and Development of the CSIR for financial support for this project through the South African National Committee for Weather, Climate and Atmospheric Research.

I also wish to thank my supervisor, Professor Gledhill, for his help in acquiring many of the components of the LIDAR, from the laser and computer to the searchlight mirror used in the receiver. I am grateful too for his help in finding a site for the LIDAR and for his open door approach during the course of the project.

I would like to thank Peter Michaelis of Associated Electronics for his help in dealing with Phase-R Corporation, the suppliers of the laser and for shopping around in Johannesburg on my behalf for small orders of components,

I would like to thank Terence Coupé and Joey Mackay of the Physics Department workshop for their help in building those pieces of equipment that required welding.

I wish to thank Jeff Lucas, Barry Guthrie, John McKinnell, Joy Smith and Del Gillam in the electronics section for their help and advice on the availability and suitability of components used to build the signal processing equipment. I am particularly grateful for the way Jeff Lucas made components available to me with the minimum of fuss when they were urgently needed.

My thanks too to Michael Russ for his help during the initial setting up of the equipment and for procuring a large number of papers on LIDAR through Inter-Library Loan.

Finally I would like to thank Professor Baart, Head of the Physics Department for his support and advice. I am particularly grateful for the way he tolerated the removal of various items of test equipment to dark and distant locations for indefinite periods, from time to time.

ABSTRACT

LIDAR is the optical equivalent of RADAR. A LIDAR facility has been established at Rhodes University using a flashlamp-pumped dye laser as the transmitter and a photomultiplier tube at the focus of a searchlight mirror as the receiver. The setting up of the receiver and transmitter as well as the design and construction of the photon counting electronics is described. The LIDAR has been used to measure aerosol scattering ratios and temperature profiles in the stratosphere and these results are presented with the algorithms and software used to reduce the data. Finally some recommendations are made for future work.

CHAPTER 1

An Overview of LIDAR

1.1 A Background to the Development of LIDAR

The term LIDAR is an acronym for light detection and ranging or light identification, detection and ranging. (Measures, 1984 p. 205). It has been coined to describe the optical equivalent of radar. At visible and near visible wavelengths various atmospheric consituents interact with radiation in several ways making LIDAR more than just a ranging device. It is a powerful method of remotely sensing the atmosphere.

Optical probing of the atmosphere had been done for some time before the advent of the laser. E.O. Hulburt (1937) was able to determine atmospheric densities up to 28 km using a searchlight beam and an offset receiver so the beams overlapped at a known altitude. Observations of the sky at twilight as the Earth's shadow sliced through the atmosphere demonstrated the existence of scattering by substances in a range from 15 km up to as high as 90 km. Investigations of the sodium layer at 90 km were performed by D.M. Hunten using the twilight scattering technique as early as 1954.

Early attempts to use a pulsed light source were made with pulsed discharge tubes by Friedland (1956) and just before the availability of the high powered laser an optical radar was used by Horman (1961) to measure range gated backscatter as related to visibility. The light source was a spark producing light pulses with a duration of 1 μs comparable to the period of many LIDAR pulses.

The development of Q-switching by McClung and Hellwarth (1962) made short pulses and the capacity for good range determinations available. (Measures 1984 p. 7). Since then the invention of many different types of laser and the development of sophisticated receiving equipment has led to a large body of literature on remote sensing of the atmosphere. The main areas of research and some key papers that began each branch are listed below.

1.2 Present Major Fields of LIDAR Research

1.2.1 Metals

The first measurements of sodium in the atmosphere by resonant scattering were performed by Bowman et al. (1969) using the tunability of a flashlamp pumped dye laser. Since then there have been determinations of the concentrations of lithium and its isotopic ratio, (Jegou et al. 1980), potassium, (Felix et al. 1973) calcium and Ca^{+} (Granier, 1985) using the same method. The technique has now been refined to the extent that daytime determinations of sodium are possible, keeping the background to acceptable limits with a receiver bandwidth of only 20 pm and a field of view of 3×10^{-4} steradians. (Granier et al. 1982).

1.2.2 Aerosols

The suggestion that there were scattering layers in the upper atmosphere with a possible meteoric origin had been made by G. Fiocco (1963) who claimed to detect scattering from as high as 140 km. However it is now suggested (Hall (1974) that this and other early attempts were influenced by such sources of interference as overloading of the photomultiplier tube from low altitude scattering and fluorescence in the rod of the ruby laser which was responsible for some of the delayed light. The first non-controversial detection of tropospheric and stratospheric aerosol layers was made by Fiocco et al. (1964) and since this time stratospheric aerosols and their connection with events such as volcanic eruptions releasing large amounts of sulphur dioxide and dust have come to be routinely monitored by lidar stations in several parts of the world.

1.2.3 Ozone

The technique of differential absorption and scattering in which two wavelengths are employed, one tuned to be absorbed by the species of interest and the other to the

shoulder of the absorption band to allow a determination of the amount of light absorbed was first suggested by Schotland (1964). In upper atmosphere research it has been used to determine ozone concentrations using UV laser light produced by frequency doubling in non-linear crystals (Megie, 1977). The acronym DIAL is now used to describe all DIfferential Absorption Lidar techniques.

1.2.4 Temperature

Remote temperature determinations have been performed using a variety of techniques. If it is assumed that in the clear air above about 10 km light is scattered principally by Rayleigh elastic scattering then the ideal gas laws and hydrostatics can be combined to yield temperature in terms of the density change (Sandford, 1967). Line broadening has also been used to determine temperature. (Fiocco et al., 1971).

1.2.5 Winds

If the atmosphere doing the scattering is in motion then the returned radiation is Doppler shifted. Various methods of detecting the small change in frequency have made possible remote wind measurements. Benedetti-Michelangeli (1972) used a scanning spherical Fabry-Perot interferometer. Eberhard (1980) used two laser beams with a frequency difference in the microwave region. (Dual-frequency Doppler-lidar or DFDL) The Doppler shift of the difference frequency yielded the air velocity but only over a very limited range.

It is possible to obtain even more information about conditions in the light path if the polarisation of the light is taken into account. McNeil and Carswell (1975) transmitted linearly polarised light and measured the polarisation of the returned signal. They found that the extent to which the returned light was depolarised depended on atmospheric conditions in a complex way. Some of the effects were associated with low lying layers of dust.

Although this discussion shows the extent and variation of lidar research it was envisaged that the lidar to be established at Rhodes University would concern itself initially with the detection of the intensity of returned light and the investigation of mesospheric and stratospheric temperature and aerosols.

There are now many lidar sites in various parts of the world. Most are in the northern hemisphere but there are sites in South America, Australia and Antarctica. As far as is known there is none in Africa south of the Sahara and so a site in South Africa able to monitor the middle atmosphere fills a large void in the global coverage. Rhodes University was chosen for the location of a lidar as it is the home of the Hermann Ohlthaver Institute for Aeronomy under the direction of Professor J. Gledhill. Research into the middle and upper atmosphere and ionosphere is undertaken here as well as at the South African base at SANAE in Antarctica.

1.3 An Outline of the Rest of this Thesis

In chapter 2 the reason for choosing the laser used in the Rhodes LIDAR is discussed as well as the design of the receiver. Chapters 3, 4 and 5 deal with the design, construction and testing of the pulse counting and photomultiplier tube testing electronics. In chapter 6 the performance of the laser is evaluated and the method used to align it with the receiver is described. Chapter 7 describes how the Rhodes LIDAR is operated. The processing of the data and results obtained are dealt with in chapters 8 and 9. Chapter 10 contains some suggestions for future work. Listings of software written for the project are given in the appendices.

CHAPTER 2

Take your hare when it is cased... Hannah Glasse, The Art of Cookery, 1747.

Choice of Transmitter and Receiver and Set-up Procedure

2.1 Requirements for a LIDAR Laser.

To be suitable for a LIDAR the laser must be capable of producing short pulses of high energy well collimated light. In order to investigate atmospheric constituents by resonant scattering it is also necessary that the laser be tunable over a wide range.

2.2 Reasons for Choosing the Phase-R DL 2100B Flashlamp Pumped Dye Laser.

The dye laser is able to fulfil all these requirements. The pulses are short, typically less than 1 μs, each pulse may contain several joules of energy, the light is fairly well collimated with a beam divergence typically less than 3 mrad in a flat-flat cavity and depending on the choice of dye, tunability across the entire visible spectrum is possible.

A dye laser is commonly pumped in one of the following ways

(i) with another laser, typically an argon-ion device.
(ii) with a co-axial flashlamp.

The first method was ruled out by the high cost of the pumping laser and the expected alignment difficulties in a comparatively hostile environment. Advantages of a flashlamp pumped system are low cost, tuning versatility, ease of alignment and simplicity. (Candela Co. brochure.)

With the type of laser selected it was necessary to choose a make and model. The lasers of Candela Corporation and Phase-R Corporation were evaluated. Those of the former, although of high quality, were ruled out by cost at a time when the exchange rate was less than $ 0,50 to

the rand. From stated specifications the Phase-R DL 2100B with high rate adaptor (HRA) was selected for the Rhodes LIDAR as its pulse characteristics were suitable and its cost was almost within budget restrictions. Components for grating-convex unstable (GCU) resonator or plano-convex unstable (PCU) resonator cavities are also available to further enhance performance. The specifications of the DL 2100B as given by Phase-R corporation in its brochure are shown in table 2.1. These specifications equalled or exceeded those of lasers used by other researchers in the field.

Rated output energy per pulse	3 J
Laser pulse rise time	0,1 µs
Pulse width FWHM	0,5 µs
HRA operation rate	0,5 Hz
Beam diameter (with HRA)	10 mm
Maximum stored energy	600 J
Divergence (flat-flat cavity)	< 3 mrad
(PCU/GCU)	< 0,5mrad
Stability (intensity variation)	1 - 10 %
(spectral variation)	< $\Delta\lambda/5$
Line width (flat-flat cavity)	4 nm and Eo J
(DP-60 prisms)	1 nm and 0,95 Eo J
(PCU/GCU)	0,02 nm and 0,7 Eo J
Cavity conisting of:-	
Narrow band unstable resonator (NBUR)	
#2 etalon and #3 etalon for future	
resonant scattering experiments	3,4 pm and 0,5 Eo J

Table 2.1 Pulse characteristics of the DL 2100B flashlamp pumped dye laser. (Phase-R brochure DL-2100/3100 High Energy Series)

The use of DP-60 prisms would produce a beam adequate for the investigation of atmospheric temperature and aerosol concentrations. With etalons the linewidth of 3,4 pm would enable investigation of the sodium layer. In practice certain difficulties were experienced with the achievement of the above specifications.

2.3 Procurement History

The laser was ordered in January 1985. It arrived in June but was found to have been damaged in transit. The laser was returned for repair and arrived back in September. First beam was obtained by October.

In addition to the laser and DP-60 prisms the following major items have been purchased.

(i)	2 prs protective goggles (low pass)	September 1985
(ii)	Model RJP-736 detector probe manufactured by Laser Precision Co. for monitoring beam energy.	December 1985
(iii)	Plano-convex unstable resonator (PCU)	December 1985
(iv)	#3 etalon for tuning to 589 nm	December 1985
(v)	Spare flashlamp	July 1986
(vi)	Dye filter and papers	July 1986

The laser is mounted on a copper cladded concrete slab for stability. The slab rests on two tyres on a strong wooden table in a room on the ground floor to decouple it from ground vibrations.

The light that issues from the laser is passed through an aperture in a window of the laser room and is then directed vertically upwards by a right angled prism mounted on the end of a rigid aluminium beam. The arrangement of the transmitter is shown in the two following figures.

The way in which the laser works and its actual performance are described in chapter 6.

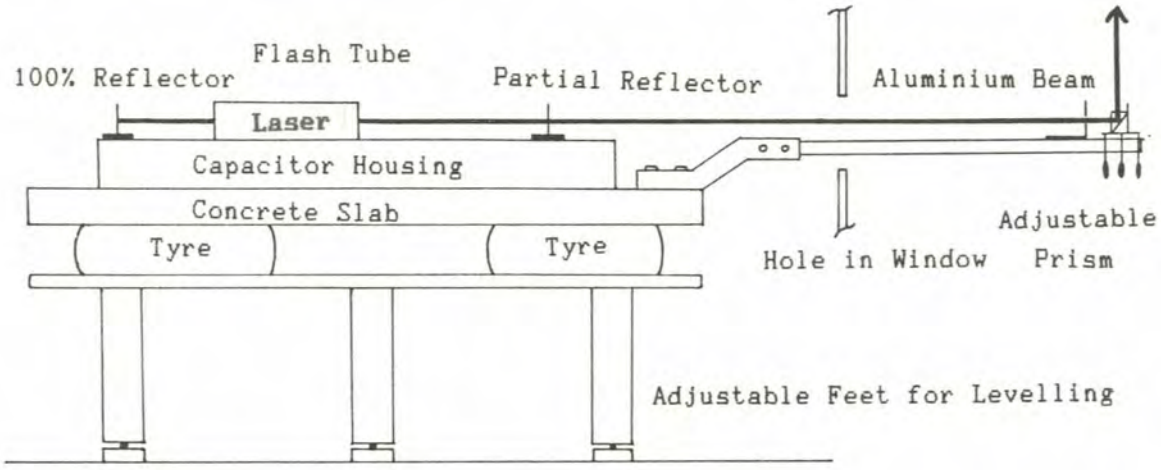


Figure 2.1 Schematic diagram of the laser transmitter

Figure 2.2 The Laser room with aluminium beam supporting the prism visible protruding from the window. The receiver cover can be seen in the background on the left.

2.4 Receiver Design

Early LIDARs suffered from the effects of saturation of the photomultiplier tube (PMT) by near field scattering. Two methods have been used to overcome this.

1. Shielding the PMT. Two techniques have been employed.

   (i) Use of a rotating shutter to eliminate fluorescence from the laser and cover the PMT during the arrival of near-field scattered light. The shutters are typically driven by a synchronous motor at from 12000 to 24000 rpm and have a radius of up to 10 cm (for example Kent et al, 1967 and S.K. Poultney, 1972 a).

   (ii) Switching of the voltages on the PMT. Charles S. Cook et al (1972) report an attempt to limit the saturation effects of the return from a smoke plume in a short-range LIDAR by applying a voltage pulse to the PMT focus electrode and some of the dynodes. This had no effect on afterpulsing and they concluded that afterpulsing originates between the photocathode and the first dynode and is due to photoelectron ionisation of the residual or surface adsorbed gas in this region. These ions then accelerate to the photocathode, their time of arrival being dictated by their mass, point of origin and the local electric field, where they produce the secondary electrons that comprise the afterpulsing. This source of afterpulsing is therefore not eliminated by removing the voltages applied to the tube for a few microseconds as the ions accelerate and produce afterpulses as soon as the voltages are re-applied.

2. Spatial separation between the transmitter and receiver.

   A separation of 10 m is stated to be sufficient to eliminate fluorescence effects and near-field scattering. (Measures, 1984 p 233 and p 264). A spatial separation of

50 cm has also been used for this purpose in an air pollution radar. (Hamilton, 1966). In early experiments with resonant scattering by sodium atoms, Bowman et al (1969) used a separation of 7 m to ensure no overlap of transmitter and receiver beams below 15 km.

The system most commonly used at present is the rotating shutter because it allows a monostatic coaxial arrangement in which the receiving mirror can also be used for transmitting the beam after it has passed through expanding optics. This permits an extremely low beam divergence in a diffraction limited system. Another advantage is ease of alignment and the fact that the region of illumination does not track across the field of view. A rotating shutter also has a high switching ratio. For the Rhodes LIDAR a rotating shutter is not available at present and so spatial separation was used.

2.5 The Rhodes LIDAR Receiver

To concentrate the returned light it was possible to obtain a 1,56 m diameter back silvered searchlight mirror manufactured by Bausch and Lomb Optical Co. of Rochester N.Y. It bears the inscription "F 641 mm to C.C surface" but the centre of the paraboloid is not marked. It has been stopped down with black paper to a diameter of 1,05 m to match the field of view of the collimating optics. The mirror is mounted in a steel frame and stands on a stable platform with its centre a distance of 39,8 m from the prism directing the laser beam vertically upwards. The light concentrated by the mirror passes through an adjustable iris, is collimated by a three-lens system with a full angle field of view of 79,2° and then passes through a 589 nm thin film filter manufactured by Oriel Corporation, with a bandwidth of 10 nm and 50% transmission, before falling on the photocathode of a RCA 9558B photomultiplier tube.

The separation between transmitter and receiver is considerably more than the 10 m mentioned earlier but even at this distance the PMT is driven close to saturation by light from lower levels when observations at the maximum range are being attempted.

The mirror and PMT housing are surrounded by a 2 m high tubular steel frame on which thick curtains of a dark material are hung to reduce light from nearby ground level sources such as windows and street lamps.

The pulses produced by the photomultiplier tube are amplified and counted by the electronics described in chapters 3 and 5.

Figure 2.5 is a photograph of the receiver.

Figure 2.5 The Rhodes LIDAR receiver.

2.6 Geometrical Analysis of the Rhodes Spatially Separated (Monostatic Biaxial) LIDAR

The iris at the primary focus of the searchlight mirror can be set to 5 different diameters which determine the divergence of the receiver beam. The full angle divergence varies with angle of incidence at the aperture with a maximum value, 2α, given sufficiently accurately by

$$2\alpha = \frac{a}{f}$$

where a is the aperture diameter and f is the focal length of the mirror in the same units, and a minimum value, 2β, of

$$2\beta = \frac{a.\cos \Gamma}{f}$$

where 2Γ is the full angle field of view of the aperture and collimating optics. This is shown in figure 2.6.

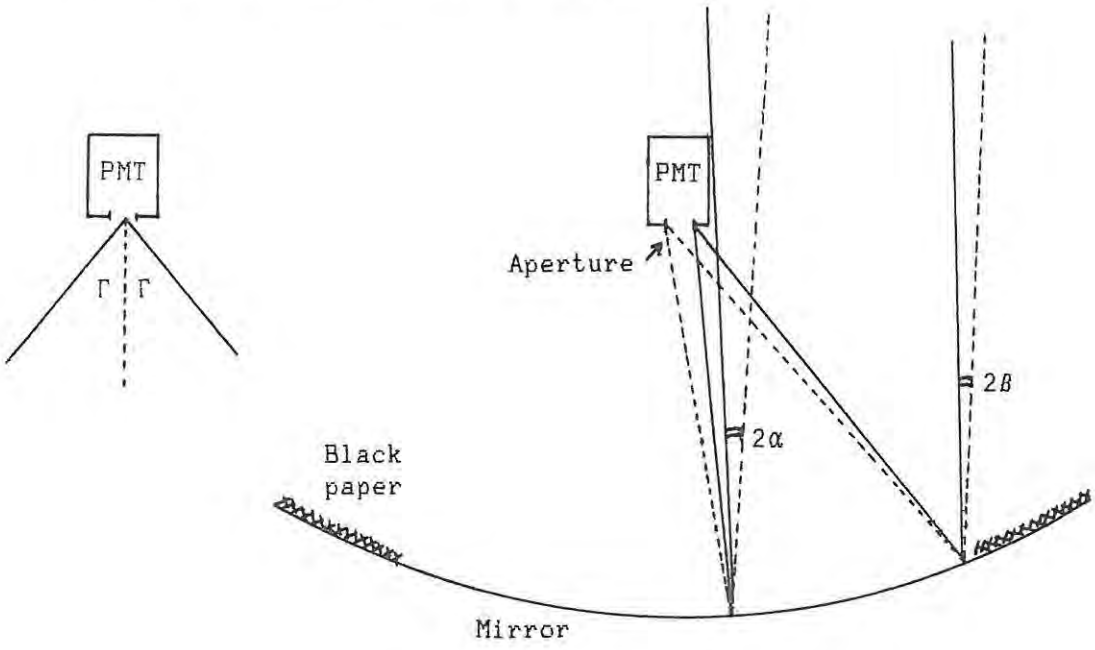


Figure 2.6 The Rhodes LIDAR receiver showing variation in beam divergence.

Requirements of the transmitter-receiver pair are

1. The laser beam must have a divergence less than that of the receiver beam.

2. The receiver beam should be as narrow as possible to limit sky background.

3. The laser beam should begin to enter the beam of the receiver at a distance far enough to ensure the PMT is not saturated by near field scatter.

4. Overlap of the beams should be completed within as short a distance as possible.

5. The laser beam should then remain within the beam of the receiver.

These conditions are fulfilled if the beam divergence requirements are met and the minimum divergence edge of the receiver beam furthest from the receiver is parallel to the edge of the laser beam closest to the receiver. This arrangement is shown in figure 2.7. Acute angles have been exaggerated for clarity. D is the horizontal separation of laser and mirror, ϕ is the half angle of the laser beam, h_1 is the height at which the laser beam begins to overlap the receiver beam, h_2 is the height at which entry is complete and h_3 is the height at which the laser beam is centered over the receiver.

With angles expressed in radians and using the small angle approximation where appropriate

$$h_1 = BR = \frac{D}{\alpha + \beta}$$

$$h_2 = AR = \frac{D}{2\beta - 2\phi}$$

$$\Delta h = h_2 - h_1$$

and

$$h_3 = XR = \frac{D}{\beta - \phi}$$

The divergence of the laser beam, measured as detailed in chapter 6, is 2 mrad. This fact, together with the above equations was used to prepare table 2.2

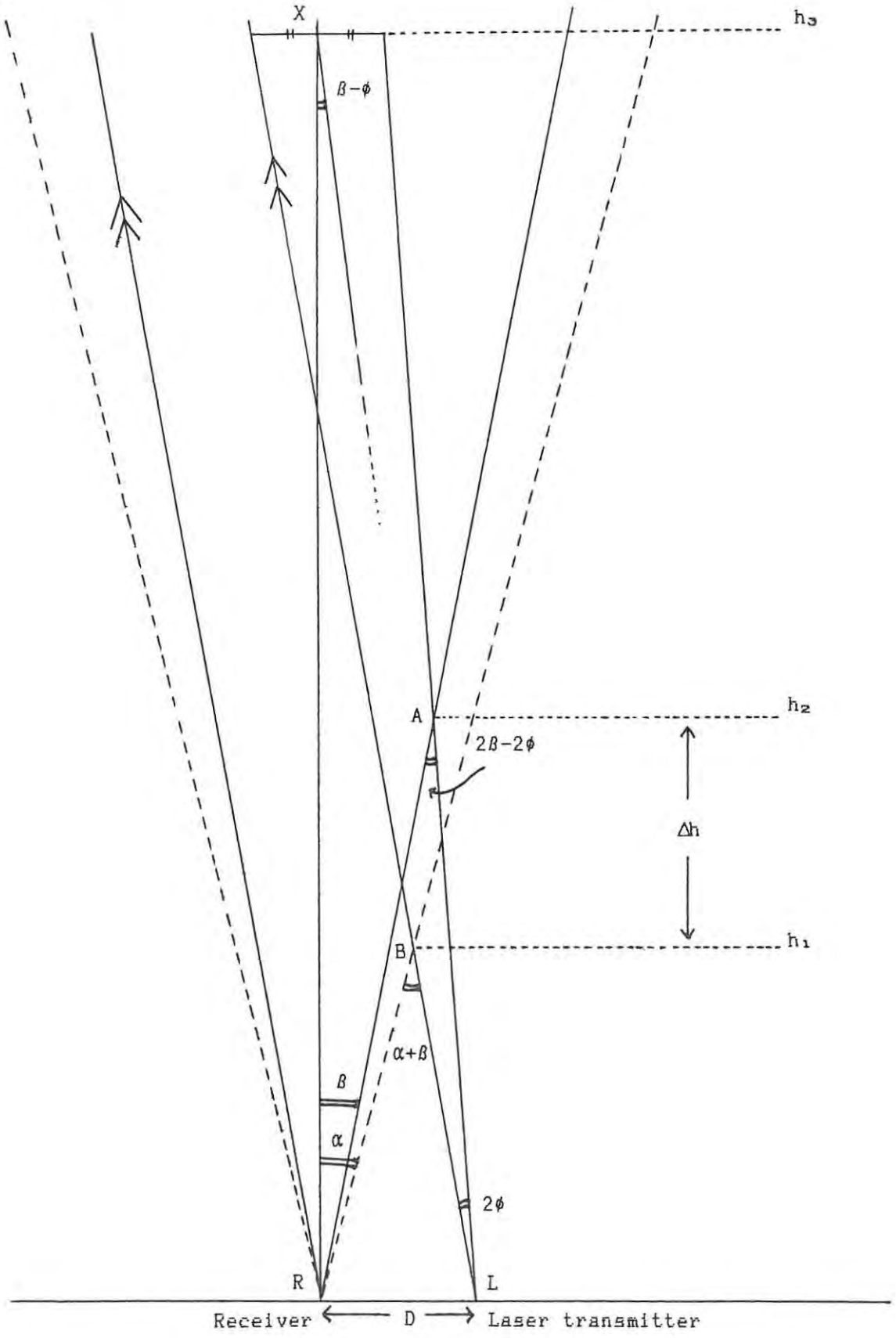


Figure 2.7 Overlap of the transmitter and receiver beams

Aperture [mm]	Minimum receiver divergence [mrad]	Maximum receiver divergence [mrad]	h_1 [km]	h_2 [km]	Δh [km]	h_3 [km]
2,50	2,95	3,82	11,8	42,1	30,3	84,2
4,00	4,71	6,12	7,4	14,7	7,3	29,3
5,50	6,50	8,41	5,3	8,9	3,5	17,8
7,50	8,8	11,5	3,9	5,8	1,9	11,6
10,50	12,4	16,1	2,8	3,8	1,0	7,7

Table 2.2 Receiver characteristics

2.7 Alignment of the Receiver.

The following procedure was used to focus the receiver on infinity and point it at the zenith.

1. The mirror was levelled with a spirit level placed across the rim succesively in two directions at right angles.

2. The bottom of the mirror was located by using a spirit level to find the point where the glass was horizontal

3. The iris was placed 641 mm vertically above the glass surface at the bottom of the dish using a plumb line passing through the iris.

4. The alignment was checked by sighting down a plumb line suspended at various points above the mirror. Since vertical rays must pass through the centre of the iris the reflection of the centre of the iris should appear plumb in line. It was found that the iris had to be raised about 3 mm to complete the alignment. This gives a value of 654 mm for the focal length of the mirror when the apparent thickness of the 15 mm glass in front of the aluminised surface is taken into account.

CHAPTER 3

If it works, leave it alone! (Personal motto.)

Design, Construction and Testing of the Pulse Counting System

3.1 Specifications for the Pulse Processing Electronics.

After the laser is fired, a pulse of light of duration about 0,4 μs, that is a beam of length about 120 m, is emitted and propagates vertically upwards. This light is scattered in all directions by air molecules and aerosols in its path. Some of the scattered light reaches the ground close to the transmitter and is collected by the receiver described in chapter 2 where it produces electrical pulses that can be amplified and counted. The time of arrival at the receiver, assuming only one scattering event per photon, is dependent on the height from which the light was scattered. It is necessary to measure this time as well as the number of photons returning per unit time interval in order to investigate the structure of the atmosphere at specific heights above the LIDAR.

Since the greatest height of interest is that of the sodium layer at about 90 km it was decided to make the vertical range of the LIDAR 100 km. This means collecting and counting photons for 667 μs after the laser is fired. The periods in which the counts are sampled after firing determine the vertical resolution of the system. It was decided to make this resolution as great as possible commensurate with the speed of readily available components and the fact that any vertical structure would be convolved with the vertical beam intensity. If the count is sampled every microsecond this gives a range bin of 150 m and it is this convenient sampling period that was decided upon.

A block diagram of the arrangement of the pulse processing electronics is shown in figure 3.1.

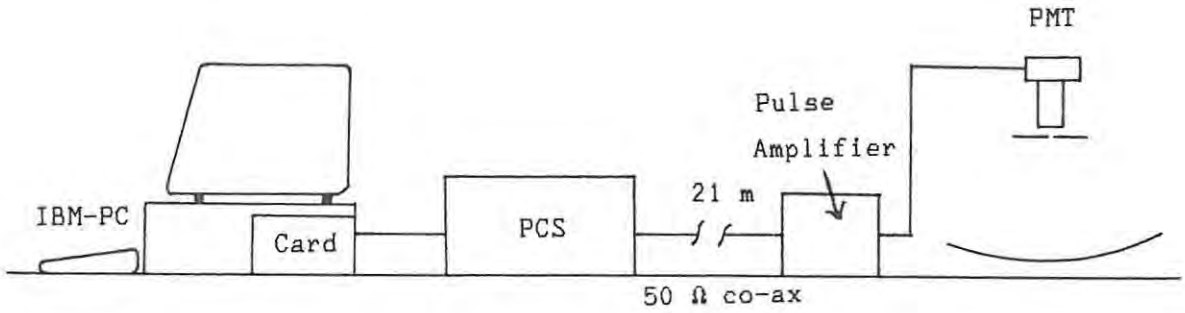


Figure 3.1 Arrangement of the receiver pulse processing electronics.

As the sampling and storing of counter values at a frequency of 1 MHz is beyond the scope of an IBM-PC a separate box of electronics had to be designed and built, the pulse counting system (PCS).

After the PCS had finished its work there would then be time before the next laser shot to transfer the counts to an IBM-PC and display an integrated atmospheric scattering profile on the screen.

3.2 The Pulse Counting System

A block diagram of the PCS designed to fulfil these requirements is shown in figure 3.2.
An overall description of operation of the PCS is as follows.

The PCS clock is derived from a 4 MHz crystal oscillator. This is immediately divided by 2 to yield a symmetrical 2 MHz waveform.

The two 8 bit counters, V and W, operate with their count, store and reset cycles 180° out of phase. While one counter counts for 1 μs the other stores its count and is reset. The counts are stored sequentially in a RAM, the addresses being generated by 10 bits of the 16 bit address counter. This counter counts down from 669 to zero. On reaching zero the clock signal is gated off terminating the process. A signal is then sent to the IBM-PC informing it that counting has ceased. The IBM-PC then winds the address counter back up from zero to 669 reading as it does so the counts stored in the RAM from addresses 0 to 667. The PCS is thus readied for the next shot and the counts in the IBM-PC are available for processing.

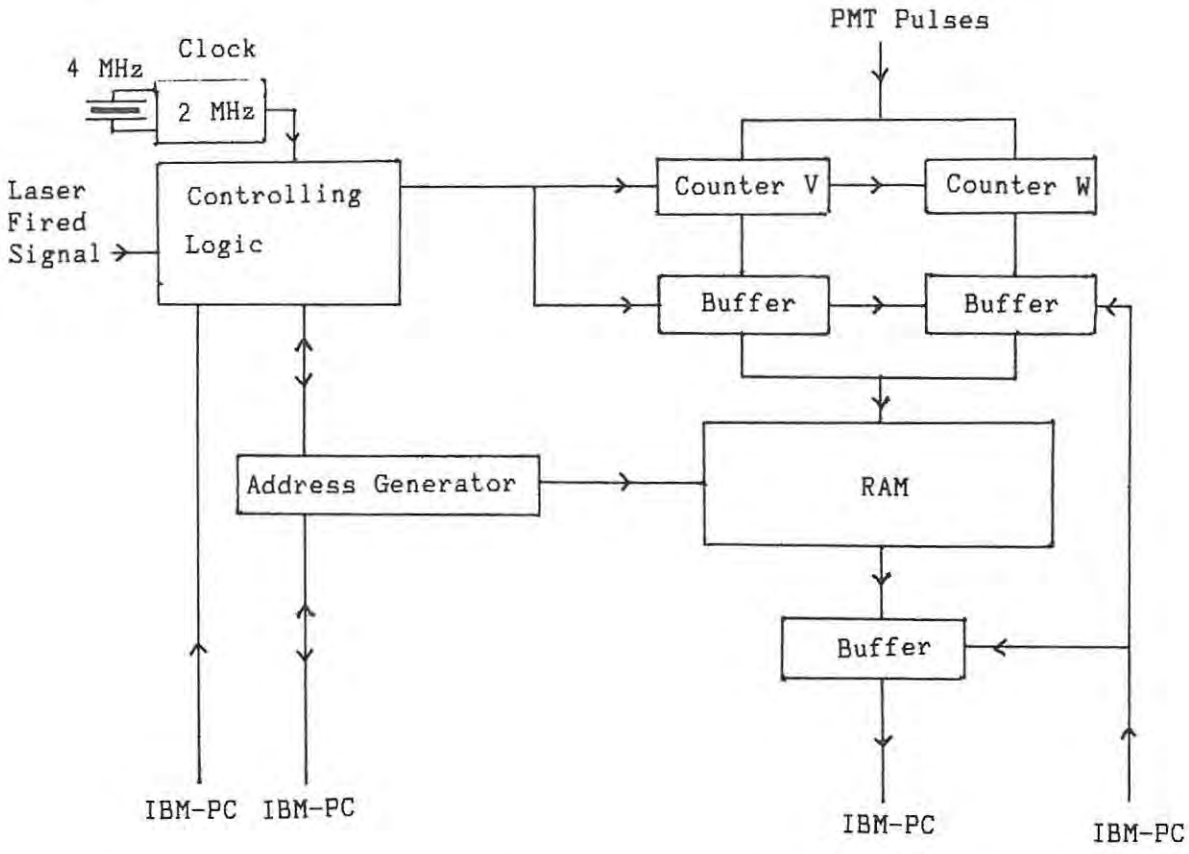


Figure 3.2 Block Diagram of the Pulse Counting System

3.3 IBM-PC Interface Card

The PCS requires 8 output control lines and 1 input line for its counters and buffers and to signal its status to the IBM-PC as well as 8 lines on which to transfer the stored counts. As no commercial parallel interface card was available at the time, one was designed and built. The Intel 8255 PIA chip was selected as suitable for implementing the interface. The addresses chosen for the card were hexadecimal 2BC to 2BF inclusive as these are not reserved in the IBM-PC I/O address space. A schematic diagram of the interface is shown in figure 3.3.

The circuit was built on a piece of "Veroboard" which was cut to fit one of the slots in the IBM PC. It was tested by hardwiring various digital signals to the port lines of the 8255 card configured as inputs and ensuring that they could be read correctly and also by checking that the correct digital values appeared when the lines were configured as outputs.

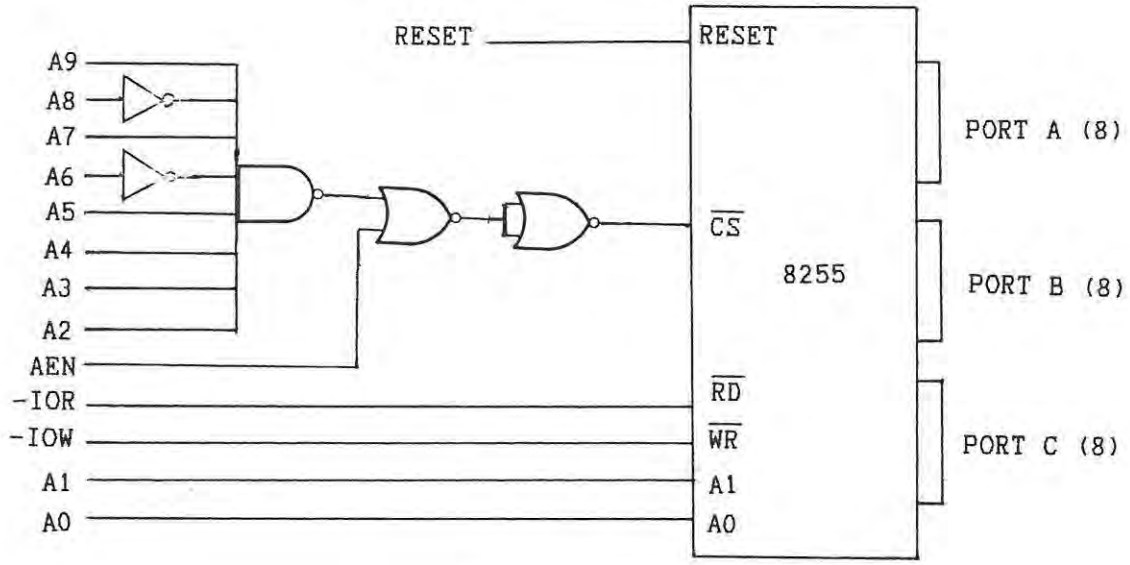


Figure 3.3 Schematic Diagram of the Parallel Interface Card.

3.4 Choice of Logic Family for the Pulse Counting System

Two overriding considerations in the design of the PCS were the requirements that each of the counters should count for exactly 1 μs and that changeover from one counter to the other should be as near instantaneous as possible. The logic selected was the Fairchild fast Schottky TTL family (FAST) manufactured by Motorola which has typical propagation delays of a few nanoseconds. The 74F109 D type flip-flops used to change from one set of counters to the other have propagation delays of 8 ns maximum and the outputs will change state in 2 ns. If path lengths are kept equal only state change times would affect the counting symmetry and as these were of the order of 0,2% of the total 1 μs counting interval they were felt to be insignificant. The 74F191 counters used for address generation and photon counting have a typical maximum count frequency of 110 MHz, well above the maximum pulse frequency of the photomultimpier tube to be used and fast enough to respond to pulses with a duration of as little as 9 ns.

The FAST logic family is as fast as Schottky high-speed TTL while drawing about one quarter of the power per gate.

3.5 Timing Considerations

The waveforms needed for the count, store and reset cycles of both counters and the enabling of the buffers through which the counts were transferred to the RAM are shown in figure 3.4. These waveforms were implemented within the control logic block.

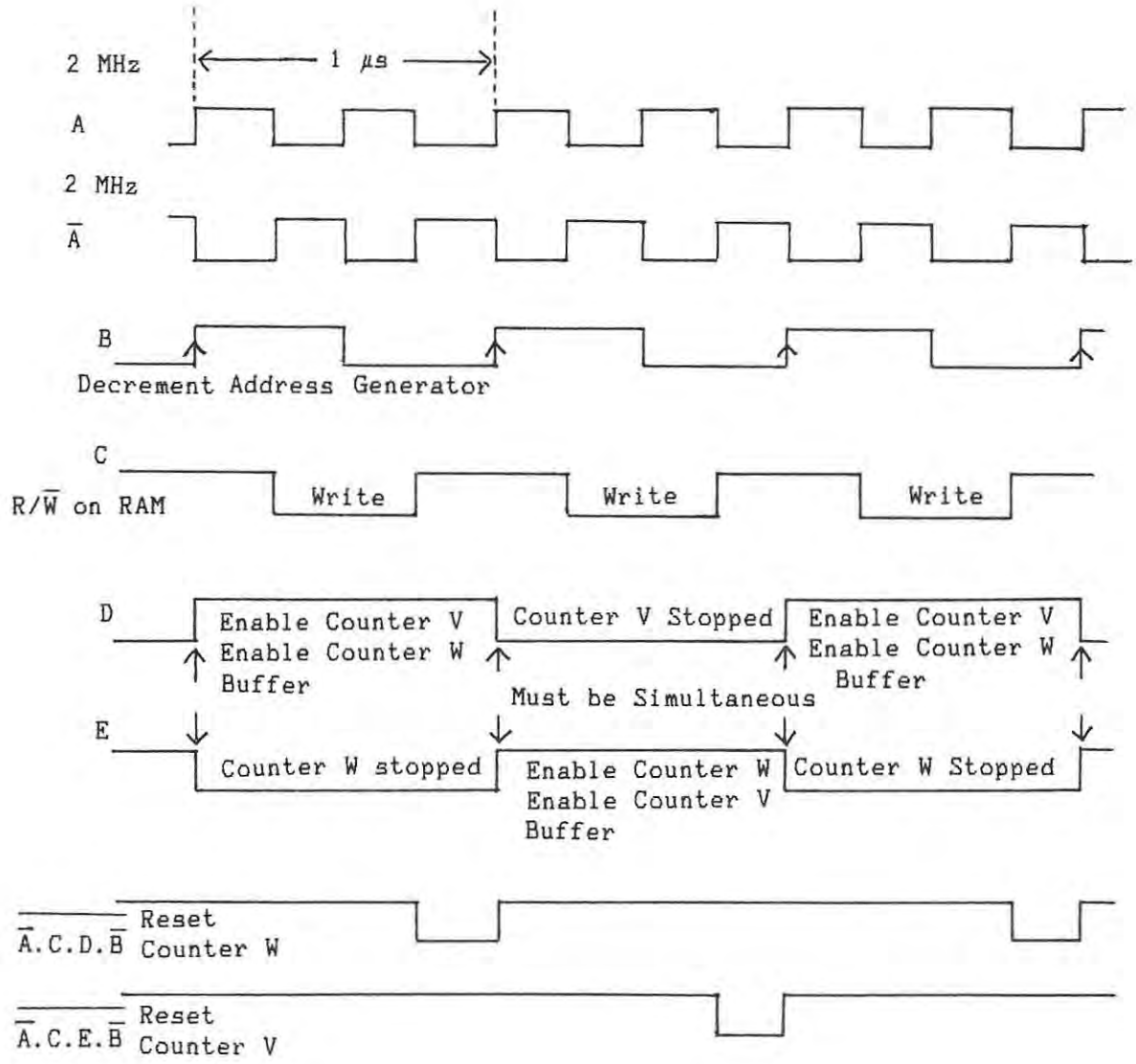


Figure 3.4 Waveforms Generated in the Logic Block of the Photon Counting System.

In order that the IBM-PC should be able to set up the initial condition of the control logic as well as be able to exercise it in a step by step fashion for testing purposes the following features were also implemented:

1. The flip-flops generating signals B and C could be set and cleared respectively by the computer to set the initial conditions of the logic block.

2. The 2 MHz clock was gated to the logic through a NAND gate controlled by a flip-flop. The flip-flop was set into one state by a signal from the computer or the firing of the laser and when the address counters reached zero it was flipped into its other state, gating off the 2 MHz clock and allowing the computer to determine that the pulse counting had stopped.

3. The direction of counting of the address generator counters as well as the source of their clock signal (PCS clock or computer) was placed under computer control.

4. The write-enable on the 6116 RAM was gated so that it could be disabled by the computer when the contents of the RAM were being read.

The functions of the connections between the PCS and the 3 8-bit ports of the 8255 chip on the interface card, PORT A, PORT B and PORT C are as set out in table 3.1.

A complete diagram of the PCS is shown in figure 3.5 which is on a fold-out sheet.

PORT A

LINE	INPUT/OUTPUT	FUNCTION
A0	input	
.	.	Lines for the data stored in the RAM
.	.	
A7	input	

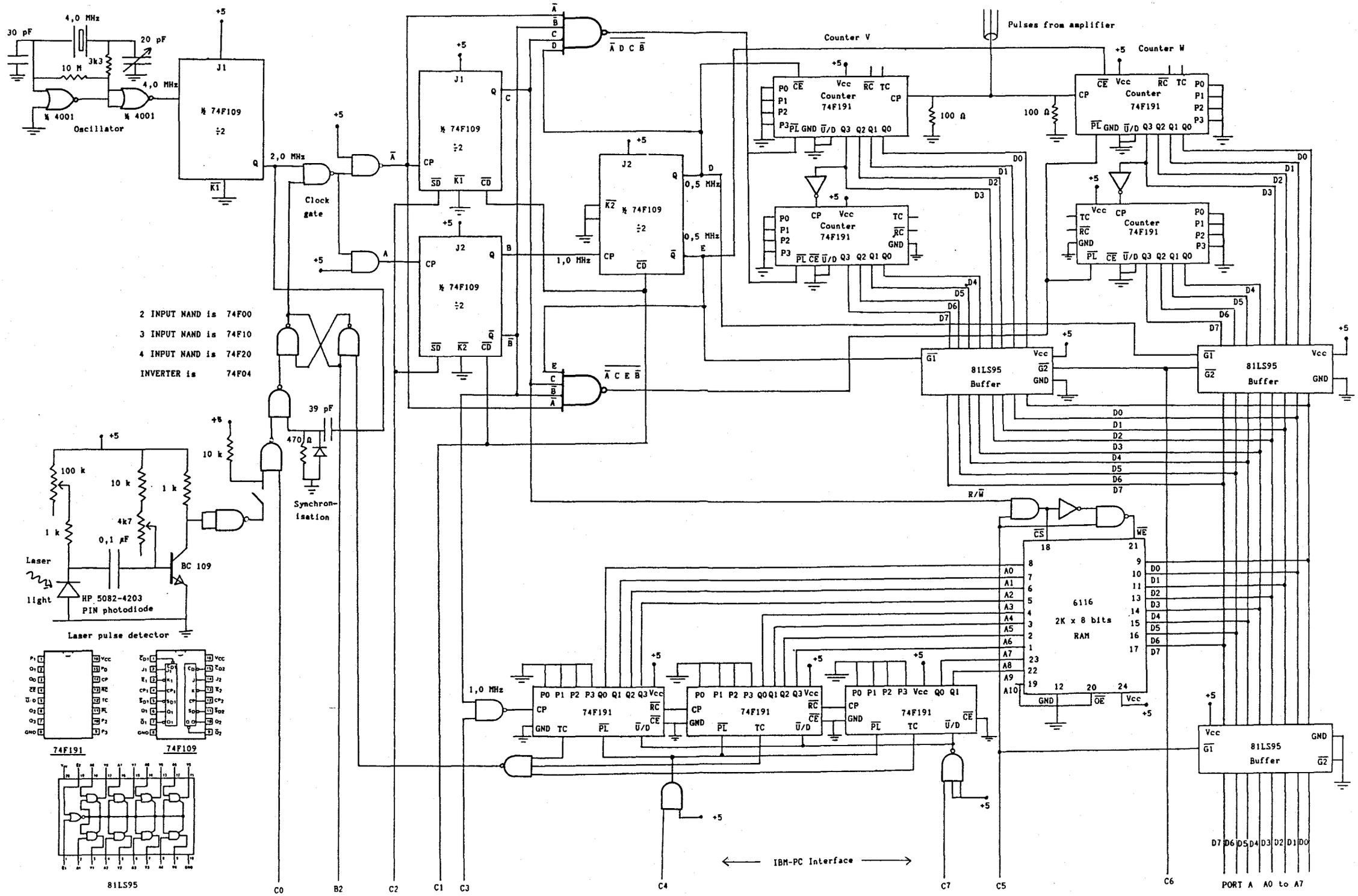
PORT B

LINE	INPUT/OUTPUT	FUNCTION
B2	input	HI when clock gated on, LO when clock gated off.

PORT C

LINE	INPUT/OUTPUT	FUNCTION
C0	output	Pulsed LO to gate clock on.
C1	output	Pulsed LO to set logic block signal B LO.
C2	output	Pulsed LO to set logic block signal C HI.
C3	output	Normally HI but with logic block signal $\overline{B}$ HI it can be used by the computer to generate a clock signal for the address counters when the contents of the RAM are being read back.
C4	output	Pulsed LO to parallel load the address counters with zero.
C5	output	LO when reading from the PCS RAM, otherwise HI.
C6	output	HI to disable both buffers between the photon counters and the PCS RAM, otherwise LO.
C7	output	LO for address counters counting up, HI for down.

Table 3.1 Connections between the PCS and the IBM-PC


4,0 MHz
30 pF
20 pF
10 M
3k3
4,0 MHz
¼ 4001
¼ 4001
Oscillator
+5
J1
½ 74F109
÷2
Q
K1
2,0 MHz
Clock gate
A
B
C
D
Ā D C B̄
J2
K2
CP
SD
CD
Q̄
0,5 MHz
1,0 MHz
E
Ā C E B̄
2 INPUT NAND is 74F00
3 INPUT NAND is 74F10
4 INPUT NAND is 74F20
INVERTER is 74F04
39 pF
470 Ω
10 k
Synchron-isation
100 k
10 k
1 k
4k7
1 k
0,1 µF
Laser light
BC 109
HP 5082-4203 PIN photodiode
Laser pulse detector
74F191
74F109
81LS95
Counter V
Counter W
Pulses from amplifier
100 Ω
Counter 74F191
P0 P1 P2 P3 CE RC TC CP PL GND U/D Q3 Q2 Q1 Q0
Vcc
81LS95 Buffer
G1
G2
GND
D0 D1 D2 D3 D4 D5 D6 D7
R/W
CS
WE
6116
2K x 8 bits RAM
A0 A1 A2 A3 A4 A5 A6 A7 A8 A9 A10
OE
IBM-PC Interface
C0 B2 C2 C1 C3 C4 C7 C5 C6
PORT A A0 to A7

3.6 Testing the Pulse Counting System

In order to accomodate the 667 data points graphically on the IBM-PC screen the computer was fitted with a Hercules graphics card giving a resolution of 720 x 350 pixels. The high level language capable of writing text and drawing lines on this screen available at the time was a version of BASIC, HBASIC. As the time-critical parts of the program were to be written in machine code it was decided to write a high level skeleton in HBASIC and call machine code routines from it as necessary. HBASIC supports the same easy-to-use commands as BASIC for the saving to disk of portions of memory as well as the loading of machine code files and data blocks to specific locations.

The first test program was written purely in assembler. It exercised the PCS cycle by cycle and wrote messages to the screen stating the stage reached.

The next stage was to inject pulses of a known frequency to the PCS and check that the correct number of pulses per microsecond were stored in the RAM and transferred to the IBM-PC. As it was intended to add the counts from successive firings of the laser and present the data graphically, a suite of assembler programs was written, callable from a simple HBASIC skeleton. They were loaded at 1 K boundaries for convenience. A memory map showing these programs and giving their function as well as showing the areas in memory where fresh and integrated data was stored is given in table 3.2.

The assembler listings of these programs as well as the HBASIC skeleton GIANT.BAS are given in the appendices.

3.7 Major Problems with the PCS

During testing it was found that the first two bytes of data corresponding to the lowest range bins contained random values. This was due to one of the counters counting dark current pulses continuously before the laser was fired. This count was latched after the first microsecond. As range bins below a few kilometers are not of interest this could be ignored and the data in these bins set to zero.

Absolute Address	Procedure Name	Function
1900:0000	BTSTGDL	Assembler program for testing the PCS system.
2000:0000	PROCA	Configure the PCS system so it is ready for triggering
2100:0000	PROCB	Read the PCS RAM and store in the PC. Wind the PCS counter from 0 to 669
2200:0000	PROCC	Wait for the laser to fire, ie wait for the laser to trigger the PCS countdown
2300:0000	PROCD	Use the computer to trigger the PCS and then wait for it to trigger itself off.
2400:0000	PROCE	Plot the stored points on the Hi-Res screen.
2500:0000	PROCF	Dump the Hi-Res screen to the FX-80 printer.
2600:0000		Storage area for accumulated counts.
2600:0800		Temporary storage of raw contents of the PCS RAM for inspection before addition to the accumulated data.

Table 3.2 **Location of Machine Code Routines and Data Storage.**

Before triggering the PCS the address counter is set to 669. Within 0,72 μs of triggering, the address counter decrements to 668 and one of the counters records the pulses for the subsequent microsecond. When this time is up the counter decrements again to 667 and the counts for the first microsecond are stored at this address in the PCS RAM. The counts for the second microsecond after firing are stored at address 666 and so on.

A more subtle problem to emerge was that when the pulse rate was 15 MHz each of the the RAM contents was found to be 31. The reason for this

was that the 8 bit counters consisting of two 4 bit stages were wired to ripple carry. The counters advance on the rising edge of the clock pulse. A ripple carry output is provided on the first stage which goes LO on a count of 15 and HI on the next count to bump up the next stage. The count is latched for subsequent storage by bringing the count enable HI. However this had the effect of forcing ripple carry HI. If the first stage contained 15 then after bringing count enable HI a total count of 31 would be stored. The problem was solved by using an inversion of bit 3 of the first stage to advance the second. Since the state of bit 3 is frozen on bringing count enable HI the correct count was then recorded.

Some graphs of count versus time are presented in figures 3.6 and 3.7. From these it can be seen that the range bins are symmetrical and that all are functioning correctly. Small, systematic irregularities are due to the pulse period not being an exact multiple of 1 μs.

3.8 Synchronisation of the PCS

The signal triggering the PCS to begin its countdown, whether emanating from the laser or the IBM-PC is asynchronous to the PCS clock. In order that the address counters decrement in an ordered and reproducible way it was necessary to synchronise the pulse triggering the flip-flop gating the 2 MHz signal to a rising edge of the 2 MHz clock. This was done by differentiating the 2 MHz clock and NANDing this signal with an inversion of the triggering signal from the LASER or the IBM-PC to produce a negative going pulse at the first rising edge of the 2 MHz clock after the asynchronous trigger input. The negative going pulse was used to gate the 2 MHz clock through to the rest of the logic in the PCS and start the countdown.

3.9 Accuracy of the PCS Timebase

It is necessary to know how accurately a particular address of the PCS RAM corresponds to a certain time after the PCS was triggered by the LASER or the IBM-PC as this determines the accuracy and precision of any distance determination.

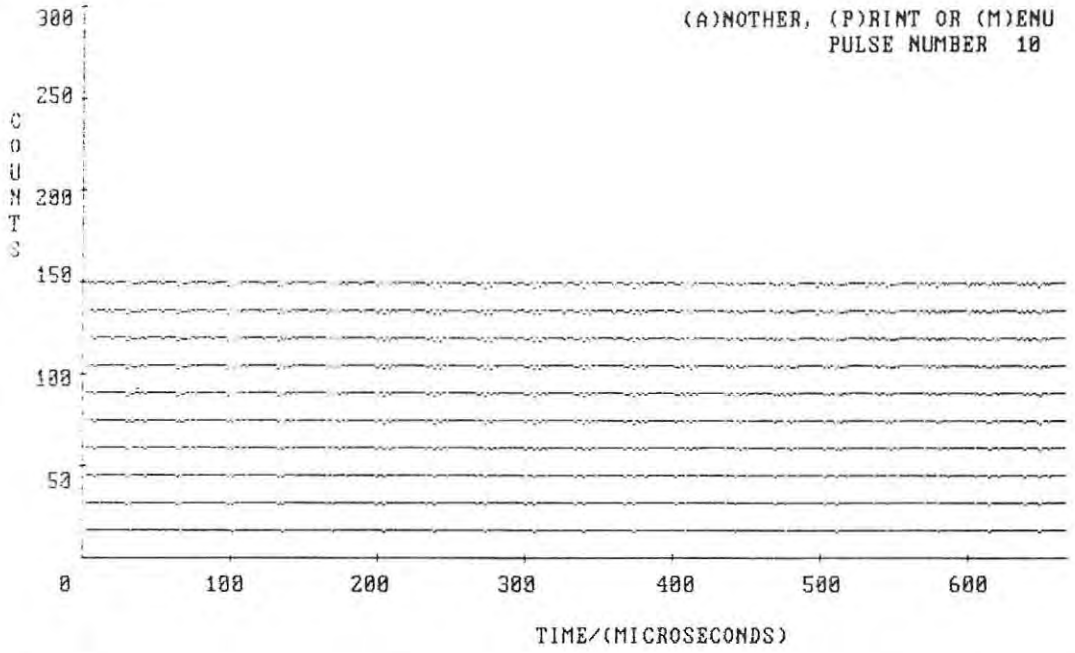


Figure 3.6 Accumulated data after 10 triggerings of the PCS while connected to a frequency source close to 15 MHz.

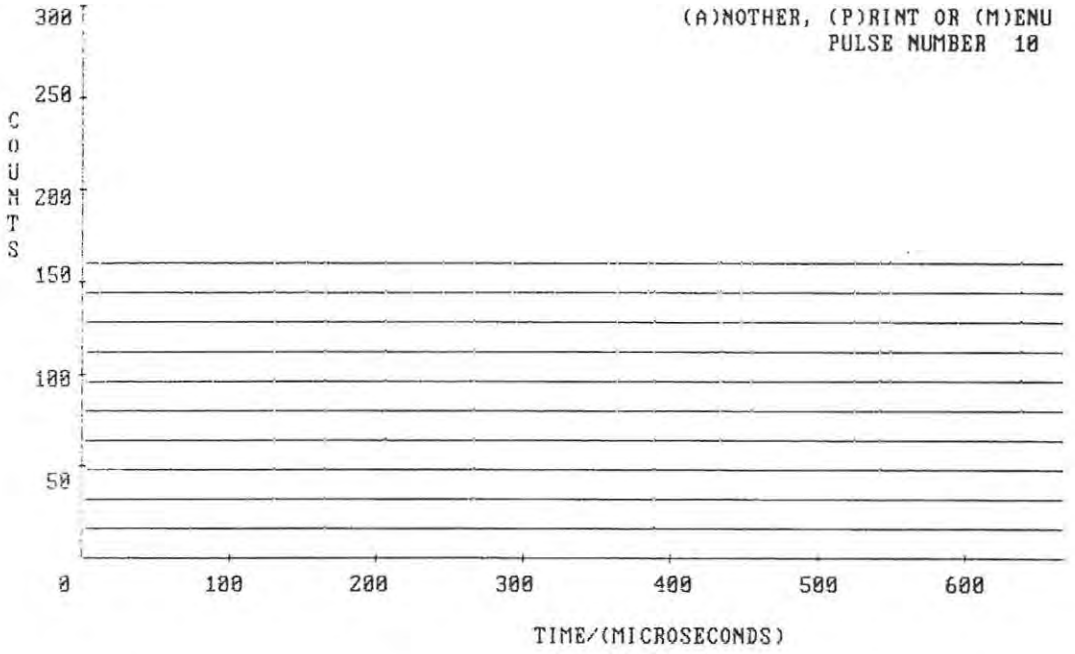


Figure 3.7 Accumulated data after 10 triggerings of the PCS while connected to a frequency source close to 16 MHz.

Consider the n^{th} PCS RAM location written to after the PCS is triggered. The uncertainty in the time interval spanned by this RAM location comes from two sources.

(i) Accumulated error due to the clock frequency of the PCS not being exactly 2,0 MHz.

(ii) An offset error due to the need for synchronisation of the triggering pulse with the PCS clock.

The crystal in the PCS was tweaked until the frequency of the PCS clock was 2,00000 MHz as measured with an HP 5314 A universal counter, hence the PCS clock is accurate to 1 part in 2×10^5 although this can be expected to drift by a few percent with time and change of temperature. After 667 μs the maximum accumulated error due to an inaccurate clock is still negligible compared to the error from (ii) above.

The PCS counter is set initially to 669. Inspection of the A0 address line with a Tektronix digital storage oscilloscope shows that the address decrements to 668 anywhere from 0,43 to 0,72 μs after the triggering signal. The first 1 μs interval then commences. The count from this interval is stored in the next interval when the address is 667 and so on. This is shown in the timing diagram in figure 3.8.

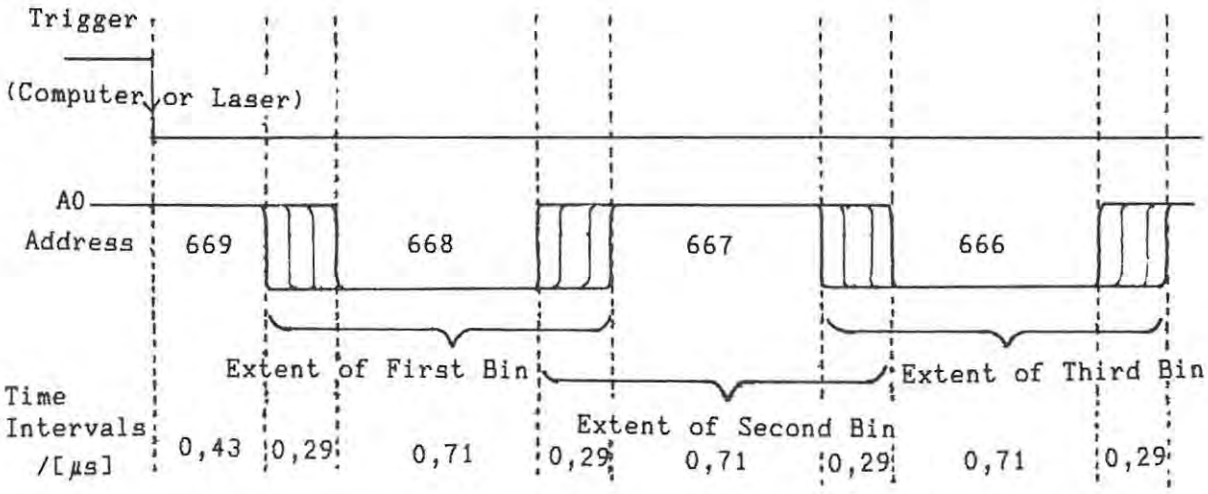


Figure 3.8 Timing diagram for address generation in the PCS.

It follows that the n^{th} bin encompasses counts produced a time t after triggering, where

$$t = (n-1) + 0,43 + 0,29 + \frac{0,71}{2} \pm \frac{(0,71 + 2 \times 0,29)}{2} \ \mu s$$

In terms of distance, taking the speed of light to be $3,00 \times 10^8$ m/s and remembering that the light must make a round trip, the n^{th} interval encompasses pulses produced by photons scattered from a distance d, where

$$d = \frac{3 \times 10^8}{2} \ . \ [(n - 1) + 1,08] \times 10^{-6} \pm 98 \ m$$

The photomultiplier tube tester which is the subject of chapter 6 is able to produce a light waveform with clearly defined edges. It has its own clock, the frequency of which is accurately known and so the time to various points in the waveform is also known. If the tester is triggered simultaneously with the PCS an inspection of the contents of the RAM at various addresses provides an independent method of confirming the accuracy of the PCS timebase. It was found that the edges of the light waveform occured where expected commensurate with propagation delays of the order of 1 μs in the analog electronics producing the waveform.

CHAPTER 4

Design and Testing of the Pulse Amplifier

4.1 Specifications for the Pulse Amplifier

The photomultiplier tube available for detecting the scattered light after collimation and passage through a thin film interference filter in the receiver was an EMI 9558B with a cathode sensitivity of 156 μA/L. It was first used at Rhodes in 1978 for airglow research.

This tube produces negative going pulses with an amplitude of about 50 mV into a 50 Ω load. The manufacturers stated "dead time" of 200 ns gives a nominal upper limit to the pulse frequency of 5 MHz. In order that these pulses be counted by the 74H191 counter chips in the PCS it is necessary that they be inverted and amplified to at least 2,8 V with a duration of at least 4,5 ns to equal or exceed the half-period of the maximum count frequency of 110 MHz after passing through some tens of metres of 50 Ω transmission line.

It is also necessary for the amplifier to include a discriminator which is able to ignore those pulses due to electrons released spontaneously within the dynode chain. These pulses are smaller than those due to photons which release electrons from the photocathode and cause multiplication throughout the chain.

A block diagram of the amplifier is shown in figure 4.1.

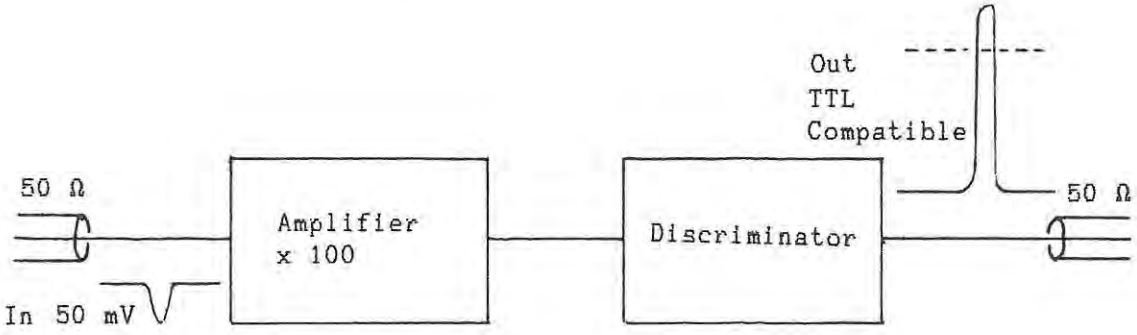


Figure 4.1 Block diagram of the pulse amplifier and discriminator

The circuit diagram is shown in figure 4.2.

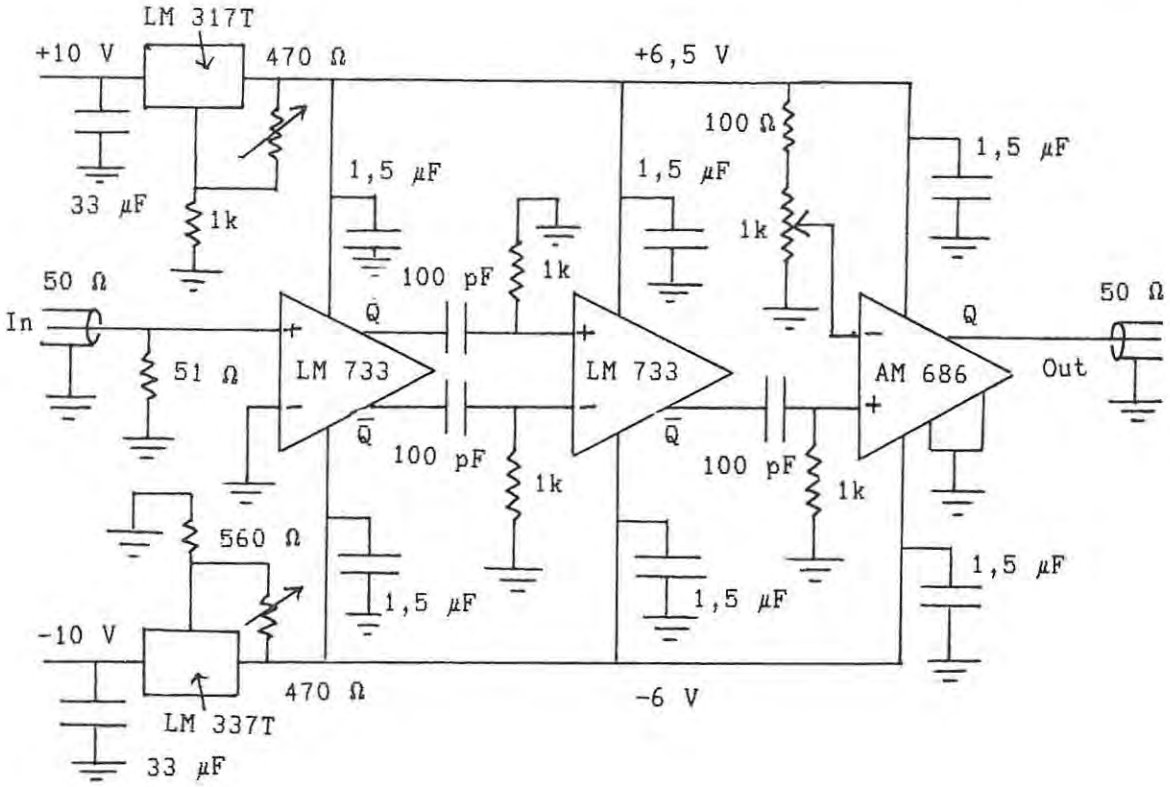


Figure 4.2 Circuit diagram of the pulse amplifier and discriminator

4.2 Description of Operation.

The LM 733 is a video amplifier with variable gain and bandwidth. When used with gain x10 it has a bandwidth of 120 MHz. Two LM 733 stages in series give a gain of 100 and a bandwidth of 77 MHz (Alley and Atwood p. 392). This is well in excess of the maximum pulse rate envisaged but the high frequency capability is necessary if the profile of pulses with a half power period of a few nanoseconds is to be maintained. The amplified pulses are then fed to the AM 686 comparator which, with the potential on its inverting input externally adjustable, functions as a discriminator. The AM 686 has a maximum propagation delay of 12 ns at a differential input voltage of 5 mV giving a maximum pulse frequency of over 40 MHz and with a rise time of a few nanoseconds pulse profile is again maintained. With components of these speeds available in an easy-to-use TTL compatible form it was not felt necessary to build the amplifier and discriminator in ECL.

The amplifier was built on double sided PC board with connections on the top side and with the the bottom side functioning as a ground plane. The board was fastened inside a thick aluminium box and situated about 30 cm from the photomultiplier tube in the receiver.

4.3 Testing the pulse amplifier

A source of fast pulses of variable frequency and amplitude was required in order to show that the pulse amplifier was working correctly but as a suitable one was not available, one was designed and built. It functions by using a ring oscillator to produce a fast square wave which is fed to both inputs of an exclusive-OR gate. One of the negative going edges is delayed by a variable amount so the gate produces positive going pulses with a frequency equal to that of the ring oscillator and of variable width. These pulses are inverted and used to drive a 1 kΩ load. Negative going pulses are then obtained by capacitatively tapping a point on the load resistor and referencing the pulses to ground.

The circuit diagram of the pulse generator is shown in figure 4.3.

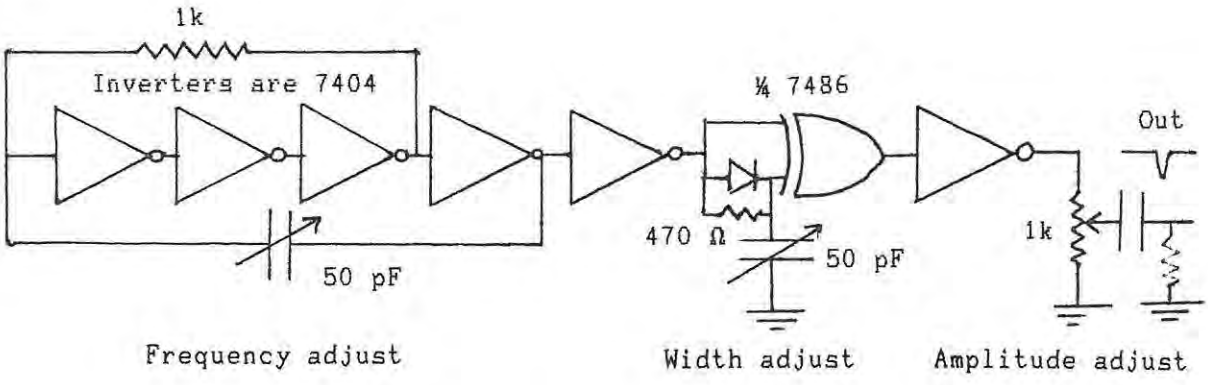


Figure 4.3 Circuit diagram of the pulse generator

4.4 Results of the Tests

The photograph in figure 4.4 shows in the top trace a stream of pulses from the pulse generator attenuated to a peak amplitude of 50 mV. These are fed to the pulse amplifier. The lower trace shows the output of the pulse amplifier fed into 50 Ω after passing down 22,4 m of co-axial cable. These pulses have an amplitude of 2,8 V. The timebase is set at 50 ns per division. There is considerable noise in the input pulse stream and the lower trace shows that this has been successfully discriminated against.

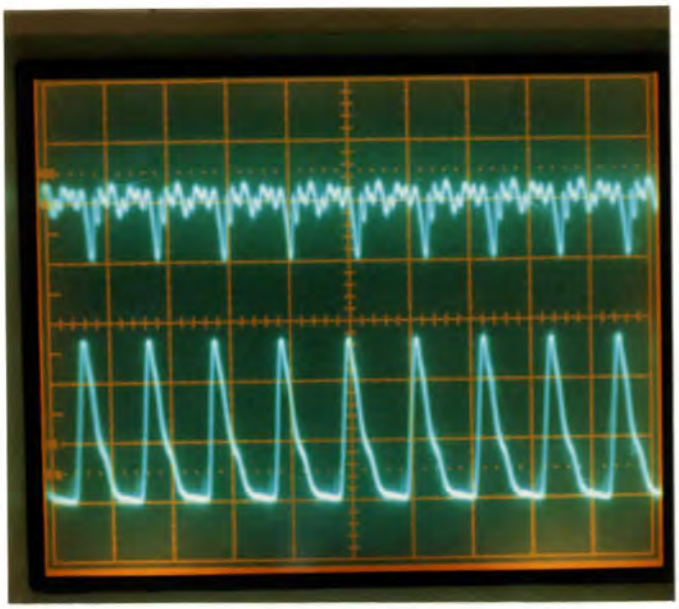


Figure 4.4 Top trace, pulses from the pulse generator, 50 mV/division
Bottom trace, amplified pulses, 1 V/division
Timebase 50 ns/division

The top trace in figure 4.5 shows pulses from the 9558B PMT. The trigger level of the oscilloscope is set at 50 mV. The lower trace shows the output of the amplifier into 50 Ω at the end of the 22,4 m co-axial cable. The amplitude of the amplified pulses is 3,6 V. The timebase is still 50 ns per division.

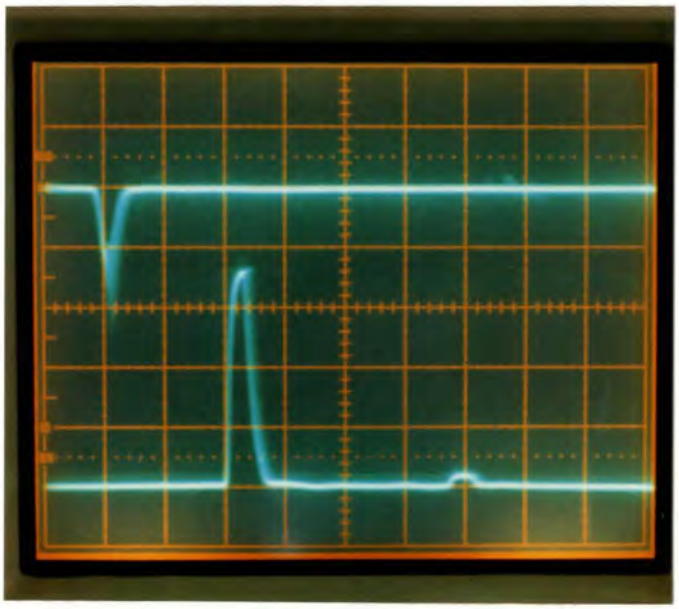


Figure 4.5 Top trace, pulses from the PMT, 50 mV/division
Bottom trace, Amplified pulses, 1 V/division
Timebase 50 ns/division

Figure 4.6 shows the effect of changing the timebase to 20 ns per division. The amplified pulses are consistently above a valid TTL HI for about 8 ns, long enough for the counters in the PCS to reliably register a count.

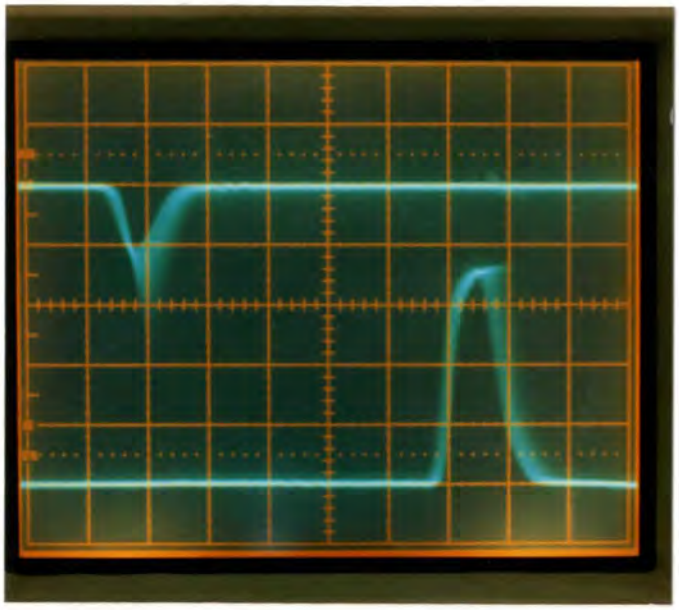


Figure 4.6 Top trace, pulses from the PMT, 50 mV/division
Bottom trace, amplified pulses, 1 V/division
Timebase 20 ns/division

From these results the pulse amplifier and discriminator was deemed to be adequate for pulses from the the 9558B PMT in the Rhodes LIDAR.

CHAPTER 5

Know thine enemy. (anon)

The Photomultiplier Tube Tester and Evaluator

5.1 Reasons for Building the Photomultiplier Tube Tester and Evaluator

The photomultiplier tube (PMT) is mounted in the receiver in order to count returning photons after the laser is fired. As each pulse is emitted from the laser the PMT may be stressed in the following ways.

1. Light scattered from the dense lower levels of the atmosphere may fall on the PMT cathode with an intensity far exceeding that of light from the much higher regions of interest, causing the tube to saturate for several microseconds and possibly affecting its performance for some time afterwards.

2. Light scattered from the beginning of the region of interest is often intense enough to cause pulse rates to exceed the maximum for which the tube is linear. If the response of the tube to these levels is known some compensation can be attempted.

In order to analyse the counts obtained from the PMT when the LIDAR is operating it is necessary to know the count rate up to which the tube is linear, the count rate at which the tube saturates and the effect of saturation on later performance as the tube comes out of saturation. It is also necessary to know how the pulse amplifier reacts to the stresses on the PMT.

5.2 Design of the Photomultiplier Tube Tester and Evaluator

It was decided that the PMT could be tested by exposing it repeatedly to an accurately reproducible light intensity waveform and integrating the pulse counts every microsecond so that a profile of its response could be built up. This is the function of the tester and evaluator. The waveform chosen was a linear ramp falling abruptly to zero,

followed by a rectangular pulse. A graph of the chosen waveform is shown in figure 5.1.

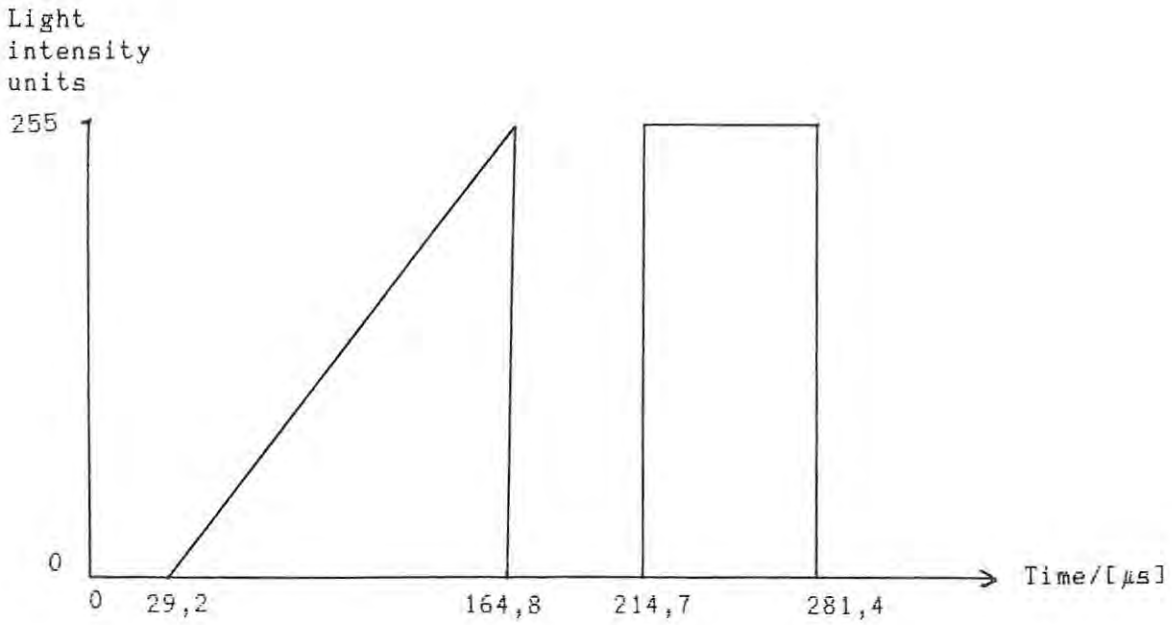


Figure 5.1 Light waveform produced by the PMT tester

The device chosen to produce the light was a yellow high intensity light emitting diode (LED) type HLMP 3858 manufactured by Hewlett-Packard, as this device has a light output directly proportional to current and, with a speed of response of 90 ns, reacts to changes in current in a time short compared to the sampling period of 1 μs. The centre wavelength of 585 nm is also close to that to which the laser is tuned when using rhodamine 6G dye. The average intensity of the light to which the PMT is exposed is adjusted by having a variable width slit in front of the LED to remove any non-linear effect in changing the brightness by varying the average current.

The waveform is produced by clocking data from an EPROM to a digital-to-analog converter (D/A) with a counter operating at a preset clock rate to generate addresses for the EPROM. The counters are triggered under computer control so that the waveform is synchronised with the pulse counting electronics and is repeatable.

The output of the PMT is amplified by the pulse amplifier discussed in chapter 4 and then sent to the photon counting system (PCS) discussed in chapter 3.

The software used is a slightly altered version of the LIDAR program GIANT.BAS which triggers the PCS and tester simultaneously and displays the integrated counts graphically with the horizontal axis calibrated in time rather than distance units.

The PMT being evaluated is placed in a darkened tube sufficiently far from the slit for the light from the LED to illuminate the photocathode uniformly. The area of the photocathode exposed is made the same as when the PMT is in the LIDAR receiver, that is a circle of diameter 23 mm equal to that of the narrow band filter. A diagram of the PMT receptacle is shown in figure 5.2.

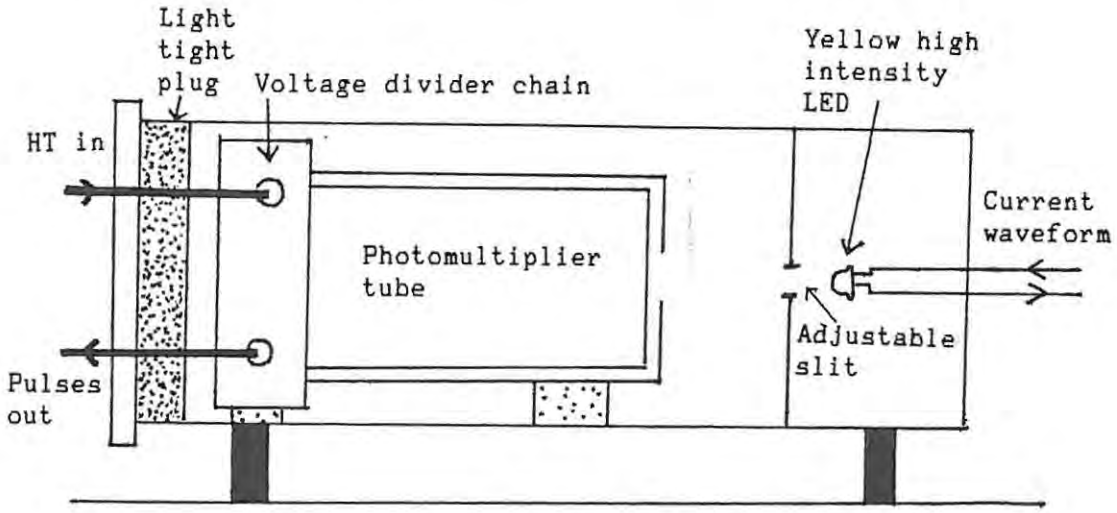


Figure 5.2 Enclosure for the PMT under test

A block diagram of the electronics used to produce the current waveform for the LED is shown in figure 5.3.

A circuit diagram of the PMT tester electronics is shown on the fold-out sheet, figure 5.4.

The clock frequency of the PMT tester is 1,881414 MHz as measured by an HP 5314 A universal counter. A frequency different to that of the PCS was chosen to avoid any beating between the two.

The contents of the EPROM used to generate the waveform are as shown in table 5.1.

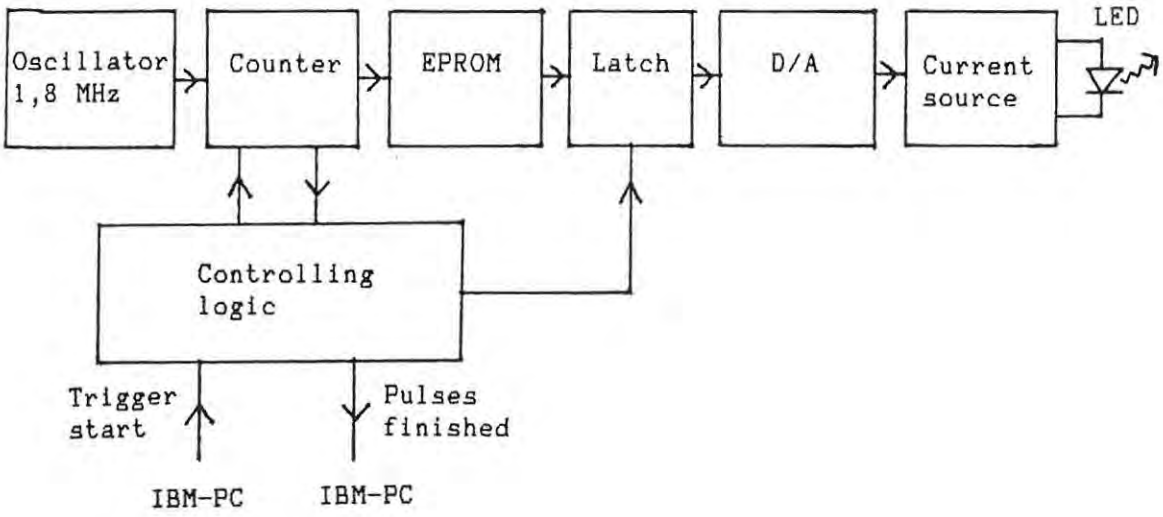


Figure 5.3 Block diagram of the electronics in the PMT tester

Address (decimal)	Contents (decimal)
0 to 108	0
109	1
110	1
111	2
112	2
113	3
.	.
.	.
615	254
616	254
617	255
618	255
619 to 806	0
807 to 1057	255
1058 to 2047	0

Table 5.1 Contents of the EPROM in the PMT Tester

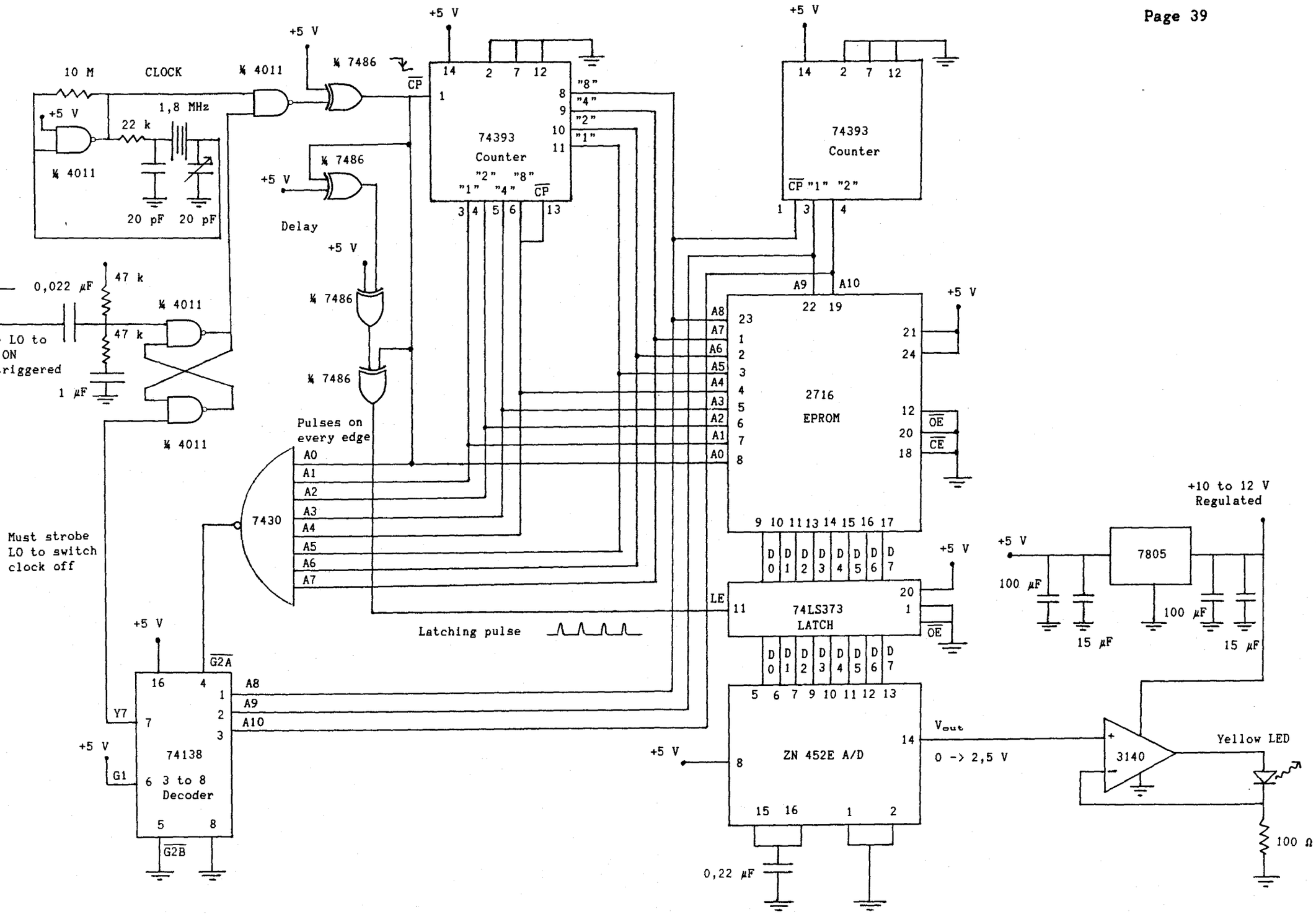


Figure 5.4 Circuit diagram of the PMT tester electronics.

It was necessary to choose a suitable cathode voltage at which to operate the PMT. Foord et al (1969) have tested several photomultiplier tubes and report that for the RCA 9558B tube, satisfactory results were achieved for photon counting work with an operating voltage of 1525 V and external amplification of 100. In the initial tests and later in the LIDAR receiver the PMT was operated with a cathode voltage of -1525 V as measured with a Beckman digital multimeter.

The first test performed on the 9558B PMT was a determination of the dark current pulse rate. The PCS was triggered 1000 times while the PMT was maintained in total darkness at room temperature. The result is shown in figure 5.5.

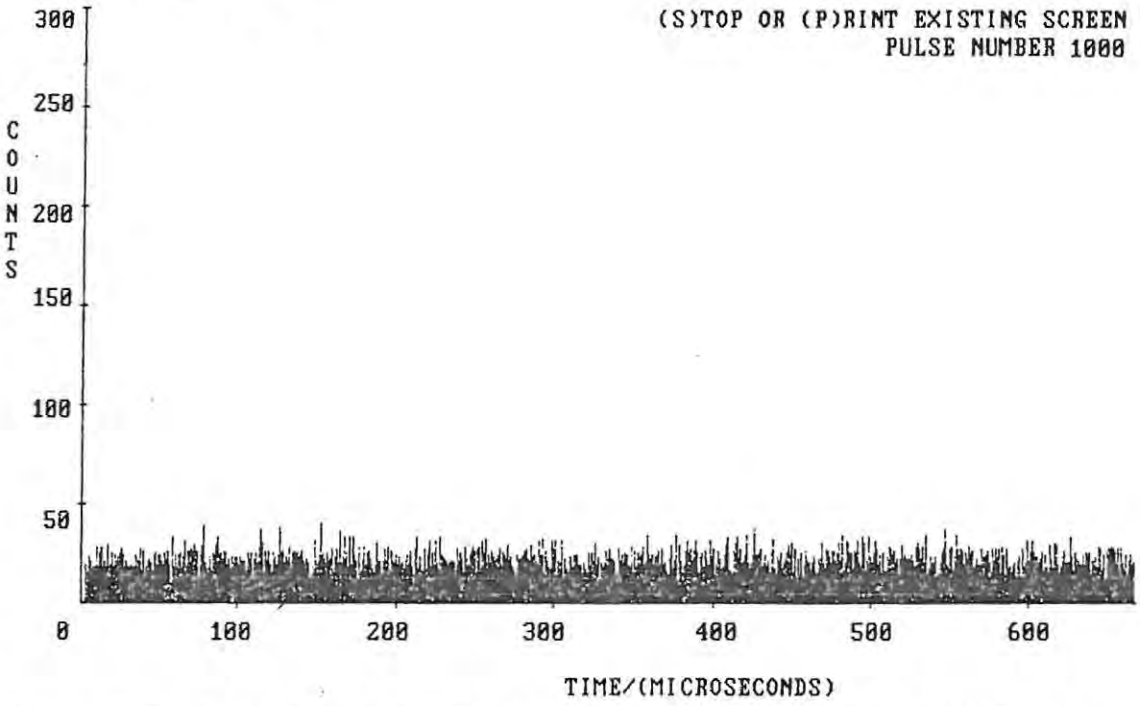


Figure 5.5 Accumulated dark current pulses after 1000 triggerings

A dump of the portion of the computer memory containing the accumulated counts showed the average of the counts in 50 of the bins to be 21,5 with a standard deviation of 4,3. This yields a value for the dark current count rate of the tube of 21,5 ± 4,3 kHz. This count rate is much less than the count rate obtained when the tube is in the receiver under a "dark" sky showing that there would be no point in going to the extra complication and expense of cooling the tube to reduce the dark current with the receiver as it is at present.

Figure 5.6 shows how the tube responds to low light levels.

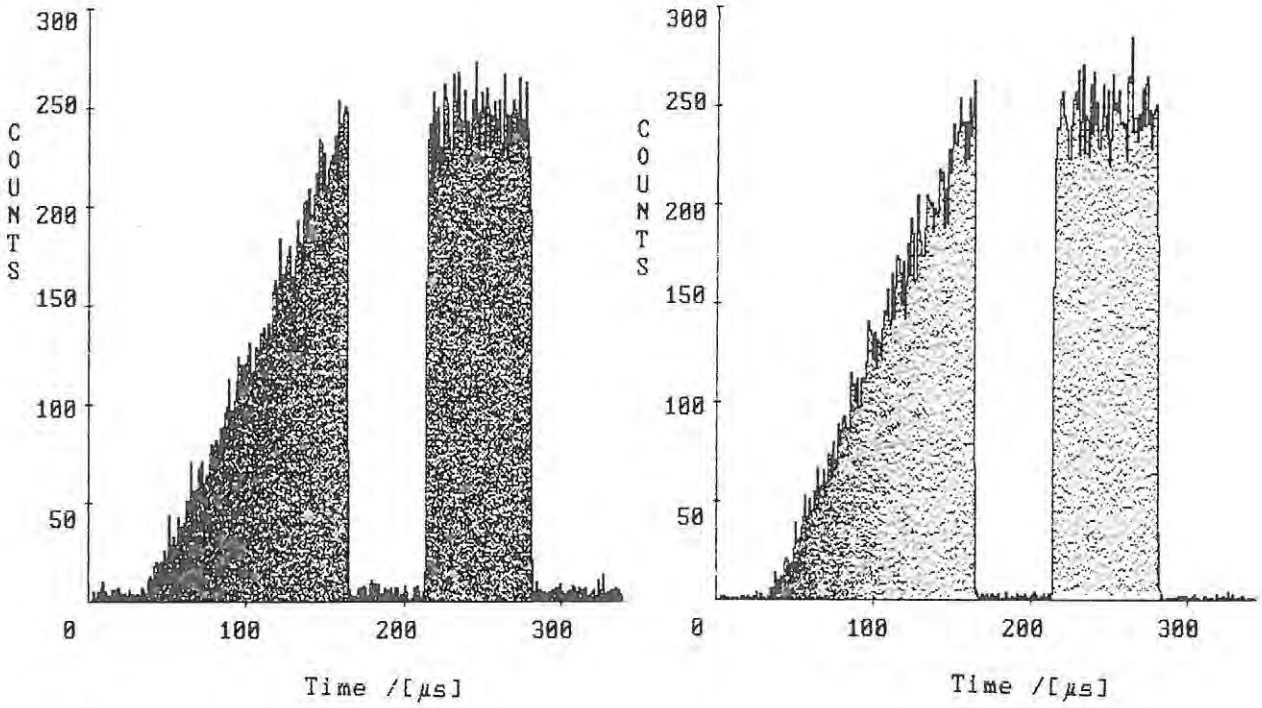


Figure 5.6 Tube response to low light levels.
Left graph, 200 pulses. Right graph, 40 pulses

Figure 5.6 can be interpreted as follows.

1. The linear current ramp supplied to the LED translates to an increase in the count rate which is linear within the bounds of the noise due to the statistical nature of the pulse generating process.

2. The count rate falls to the dark rate and rises again to its maximum value with no evidence of afterpulses.

3. From the right hand graph it is seen that the bins at the top of the triangular waveform have accumulated some 250 counts in 40 triggerings. This means that this particular tube and amplifier respond linearly for count rates up to at least 6,25 MHz.

Figure 5.7 shows the effect of allowing sufficient light onto the photocathode for some non-linearity in response to be evident. Even at these levels the count rate still increases with light intensity and the tube does not exhibit afterpulses. It is therefore still useful for measuring light intensity if the non-linearity can be compensated for.

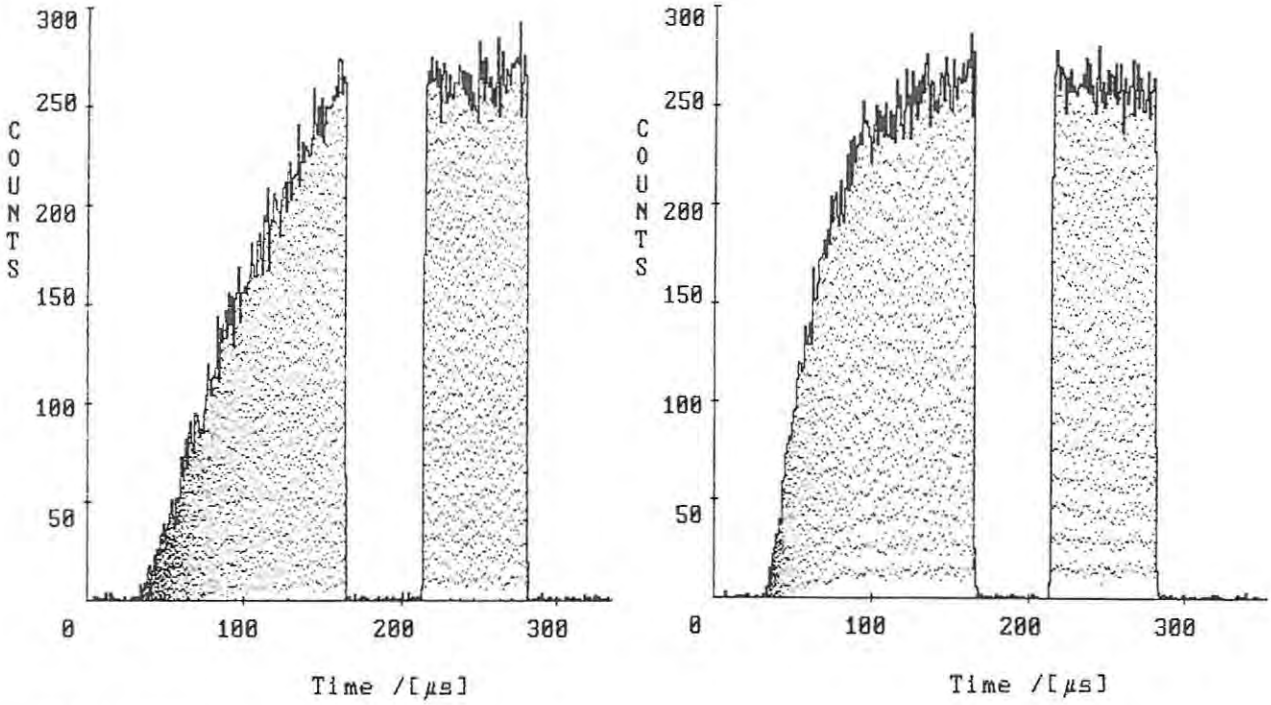


figure 5.7 Non-linearity in PMT response
Left graph, 27 pulses. Right graph, 18 pulses.

Figure 5.8 shows the effect of exposing the PMT to considerably more light. At this slit setting the tube is still generating pulses at about 14 MHz when exposed but when the light is switched off the tube generates afterpulses.

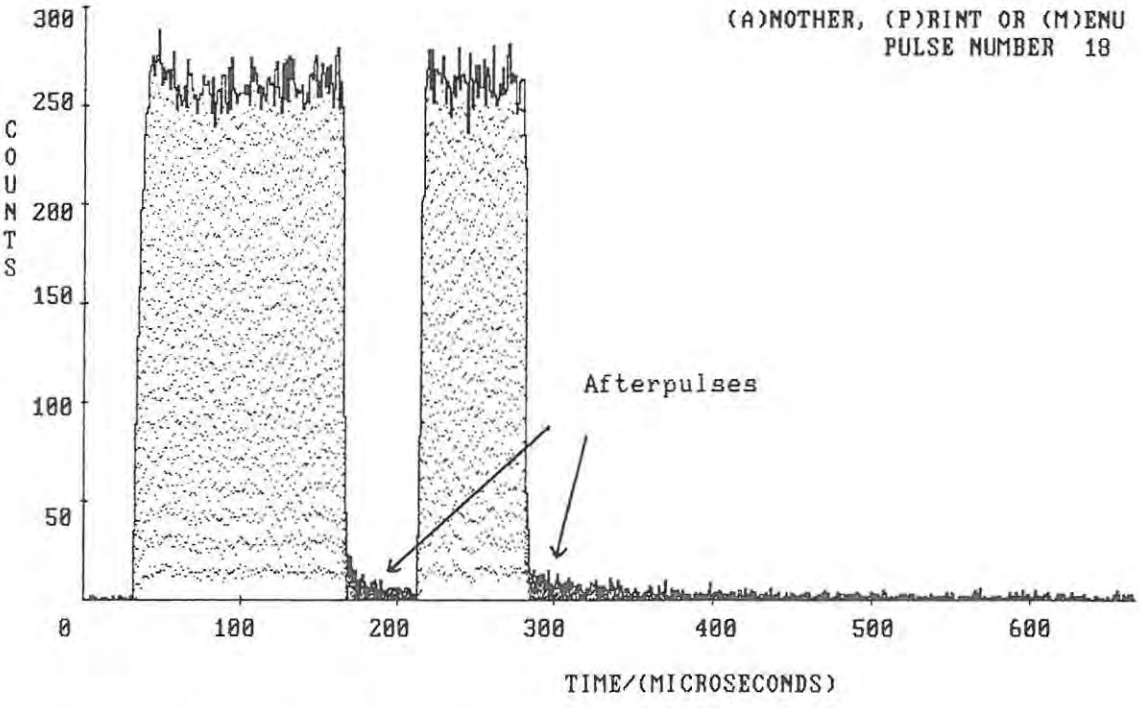


Figure 5.8 Afterpulses from the PMT. 18 light pulses.

If even more light is allowed through, it is possible for the pulse rate to be reduced to zero as shown in figure 5.9. When the light is extinguished however, considerable afterpulsing is evident for several hundreds of microseconds. This signal could be mistaken for photons returning from high in the atmosphere unless care is taken to prevent overexposure of the PMT when the laser is fired.

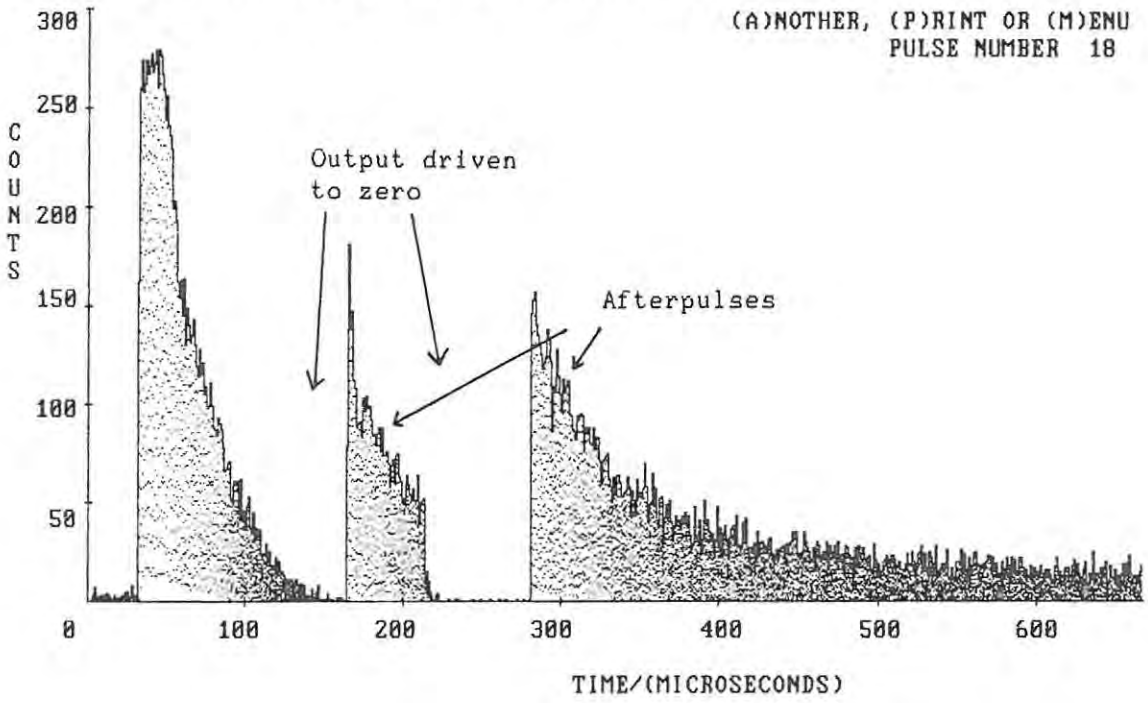


Figure 5.9 Afterpulses after severe overexposure of the PMT. 18 light pulses.

5.3 Effect of Changing the Cathode Voltage

Further tests showed that when the PMT is severely overdriven as in figure 5.9 there is no noticeable improvement in signal to afterpulsing ratio when the cathode voltage was reduced to as low as 1330 V. However when the tube is lightly overdriven and no afterpulses are being produced the response of the tube becomes linear although with considerably reduced sensitivity if the cathode voltage is reduced.

Figure 5.10 shows the effect of reducing the cathode voltage from 1530 V in stages to 1230 V while keeping the amount of light falling on the PMT constant.

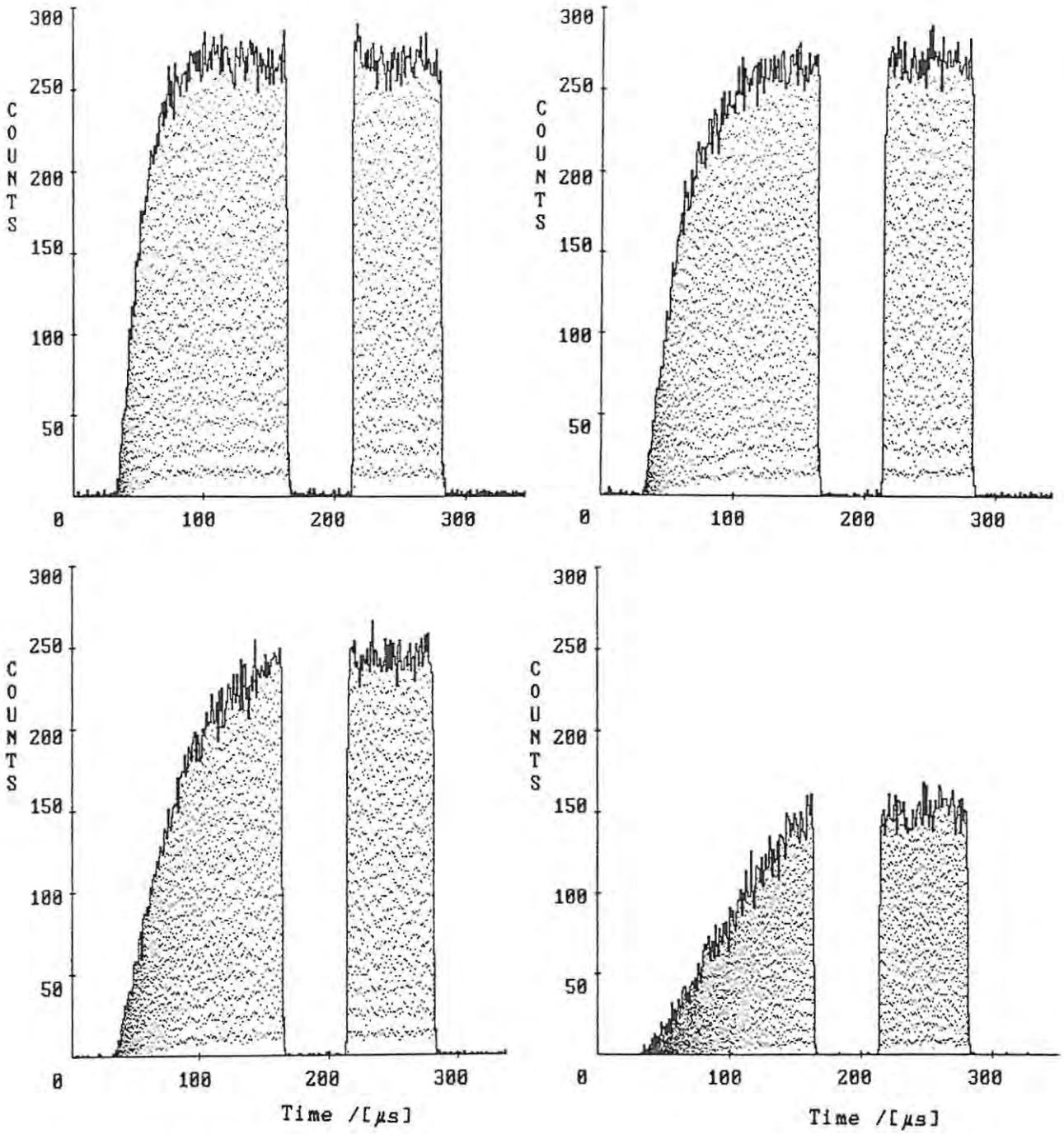


Figure 5.10 Effect of reducing the cathode voltage.
18 light pulses per graph.
Top left, 1530 V. Top right, 1430 V
Bottom left, 1330 V. Bottom right, 1230 V

If low light levels are to be examined, reducing the cathode voltage is not the solution to the saturation problem because of the accompanying reduced sensitivity. Foord et al (1969) showed that in the PMT-amplifier system the best division of gain is to be had when the tube is operated at as high a voltage as possible, provided cold field emission does not take place, and the amplifier gain can therefore be kept down.

5.4 Reasons for PMT Non-linearity and Saturation Effects

Foord et al (1969) and Poultney (1972 b) report that the upper frequency limit for a single PMT depends on the pulse shape, this being roughly gaussian due to different electron transit times between stages. At low light levels the response of a PMT is found to be linear but as the intensity increases the tube can produce more pulses than would be expected due to "pulse pile up" (Foord et al, 1969). There is an increased probability of the discriminator detecting a pulse if it sits on the trailing edge of a previous one. These pulse correlations can be minimised if a low impedance resistor is used in series with the anode and the recommended connection between anode and amplifier is a 50 Ω cable, properly terminated at both ends. As the light intensity is increased still further the PMT response levels off due to anticorrelations. Pulses fail to be detected because of a change in the voltage distribution across the stages of the resistor chain due to the high current in the tube (Foord et al, 1969). The gaussian pulse shape may also cause the discriminator to detect two pulses as one if they are sufficiently close together. An upper frequency limit is also fixed by the dead time of the amplifier and discriminator (Poultney, 1972 b).

The phenomenon of afterpulses, due to ionisation of adsorbed gas by photoelectrons from the photocathode, has been diccussed in chapter 2.

Foord et al (1969) examined a 9558B PMT and reported a quantum counting efficiency of 0,55 % and correlated pulses of less than 1 per 1000 when a dead time of 200 ns was allowed. This is equivalent to a count rate of 5 MHz and is in satisfactory agreement with the observation in the above tests that the response of the Rhodes RCA 9558B tube is linear up to a count rate of 6,25 MHz.

5.5 Compensation for Non-linearity in the Rhodes 9558B PMT

Knowing the tube is linear in its response to 6,25 MHz it was possible to generate a translation table yielding true count rates for intensities beyond the linear limit using a graph like that shown in figure 5.11. This table is used to extend the dynamic range of the tube when analysing counts obtained during the operation of the LIDAR. Actual count rates up to 14 MHz can be compensated for and translate

into true count rates up to 22 MHz.

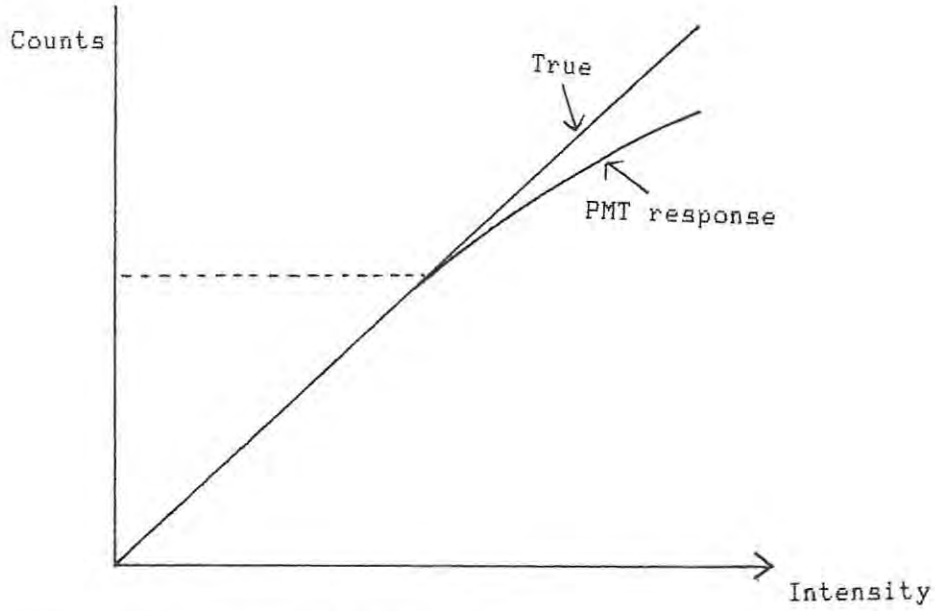


Figure 5.11 Extension of the PMT range.

The success of the translation process can be seen in figure 5.12. On the left is the original PMT response and on the right the compensated counts for two different average light intensities. If the actual PMT count rate is above 14 MHz the count rate is set to zero by the software.

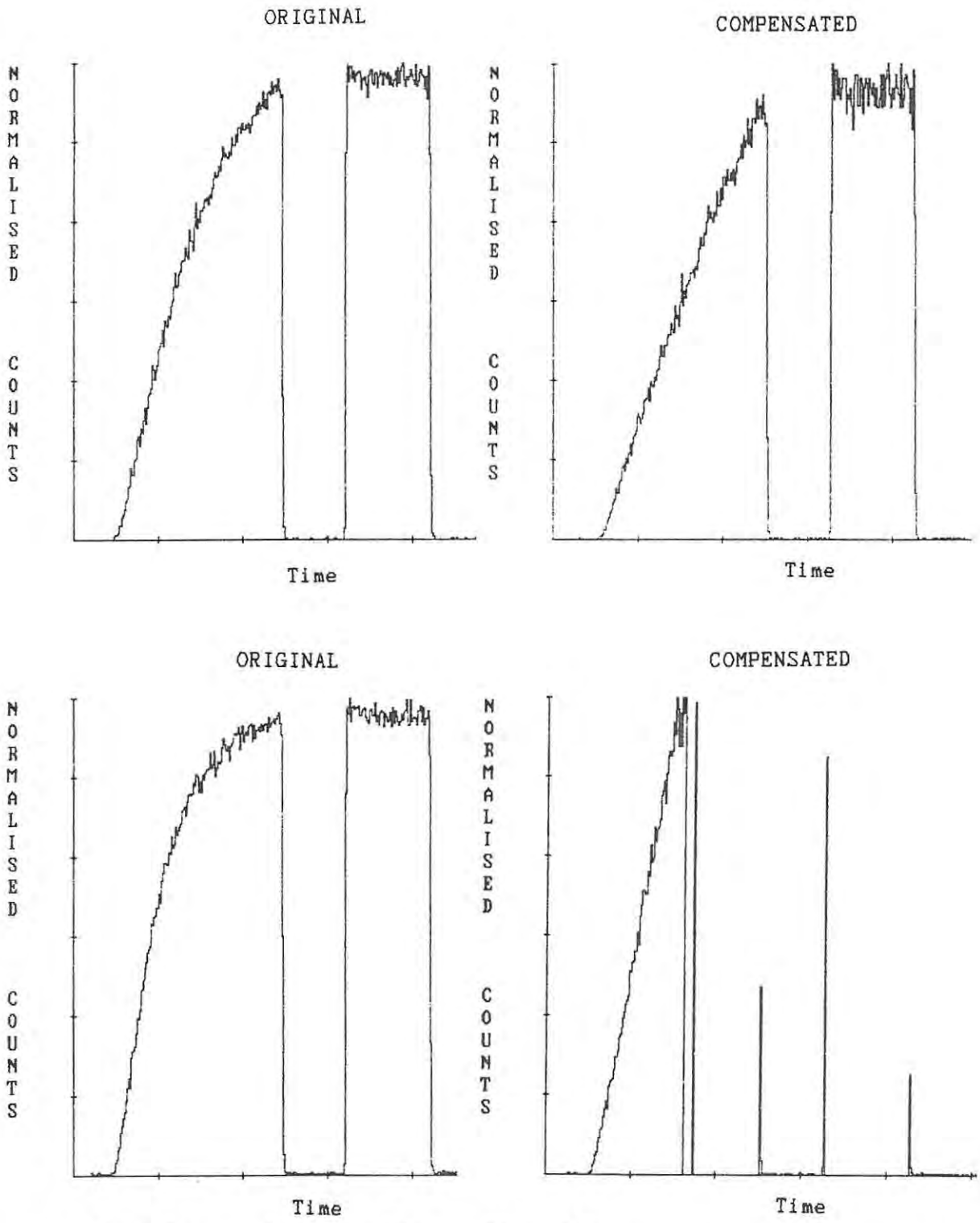


Figure 5.12 Compensation for PMT non-linearity.
Top case, slight non-linearity.
Bottom case, greater non-linearity.

Chapter 6

No matter how long you run the cold tap, you don't get hot water. (Confucius.)

Performance of the DL 2100B Laser and Alignment with the Receiver

6.1 Present Requirements of the Laser in the Rhodes LIDAR

At present the Rhodes LIDAR is being used to investigate atmospheric temperature and scattering ratio profiles and as a result the wavelength requirements for the light produced are fairly modest, being fixed by the centre wavelength and bandwidth of the narrow-band filter in the receiver. This filter has a centre wavelength of 589,0 nm and a bandwidth of 10,0 nm which means the laser has to be tuned to 589 nm but no special narrowing of the bandwidth or spectral stabilisation need be undertaken. The PMT counts are sampled every microsecond to give a range bin length of 150 m and if this resolution is to be useful the pulse emitted by the laser must be as short as possible and certainly less than 1 μs in duration. One of the methods used to reduce background counts is to make the receiver beam as narrow as possible and the laser beam has to be even narrower if the inverse square law is to apply in the data processing. Quoted beamwidth for the DL 2100B is 3 mrad and it is necessary that this be realised or bettered.

6.2 How Flashlamp Pumped Dye Lasers Work

A proposal for the use of organic materials as active substances in a laser cavity was published in 1961 (Brock et al) and later Stockmann, Mallory and Tittel (1964) suggested a laser based on singlet-state fluorescence of dye molecules. Stimulated emission from organic molecules was produced by Sorokin (1966) using a ruby laser as a pump and later by pumping with a fast flashlamp. (Sorokin and Lankard, 1967.)

One of the most attractive properties of the dye laser is tunability. In contrast to most other laser media the emission spectra of laser dyes are broad, permitting the lasing wavelength to be tuned to any chosen value within a reasonably broad range, typically from 20 to

50 nm. (2100B Operator's Manual, p.30.) Furthermore the number of fluorescent dyes is very large and compounds may be selected for emission in any given region of the optical spectrum. The dye laser is the first truly tunable laser which operates throughout the visible spectrum. (Snavely 1969.)

An energy level diagram typical of an organic dye molecule is shown in figure 6.1. The energy spacing between the rotational levels represented by the lighter lines is typically 140-170 cm^{-1}, small enough to provide a near continuum of states between the vibrational levels represented by the dark lines.

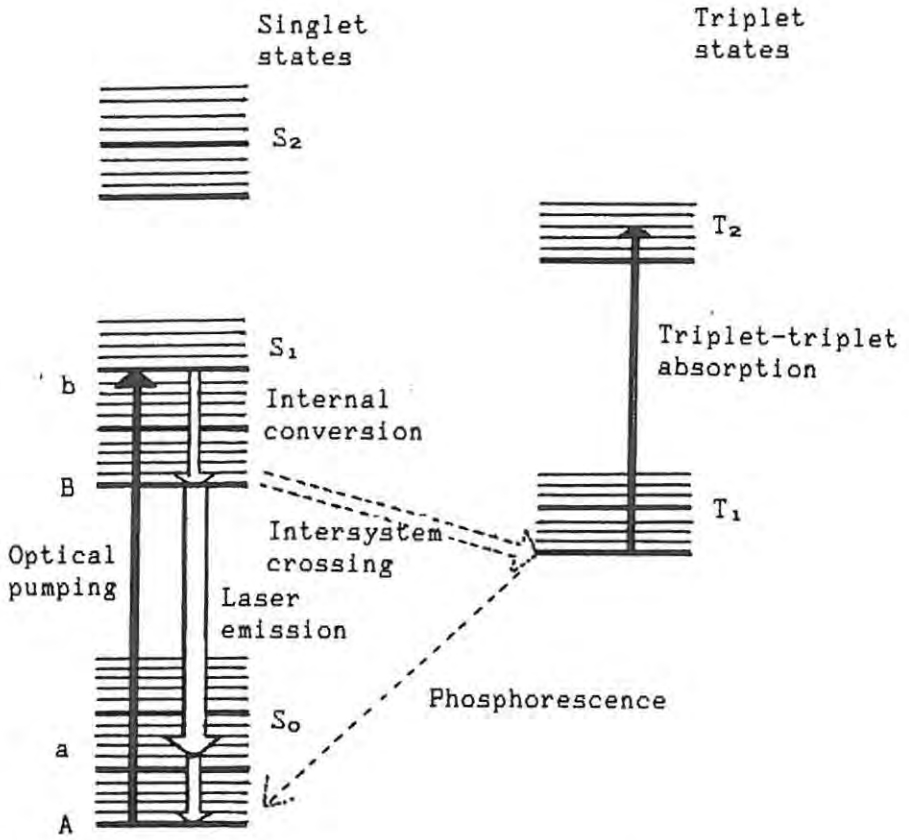


Figure 6.1 Energy level diagram for a typical dye molecule. Adapted from Measures (1984) p.192 and Snavely (1969).

Under the action of intense optical pumping from a suitable short wavelength source, electrons undergo the transition A -> b. They then decay nonradiatively to level B. If sufficient molecules have been excited for population inversion to have occured the fluorescence transition B -> a can be stimulated by photons of the corresponding wavelength already present and laser emission occurs. An important

factor which tends to limit the time for which a population inversion can be maintained is the relaxation by a non-radiative process to a triplet state. This process, known as inter-system crossing, effectively removes dye molecules from the lasing process as the triplet states are relatively long-lived, lasting for several microseconds and depending on the amount of dissolved oxygen in the solvent. The quenching due to triplet state formation is responsible for the short pulse duration. (Snavely 1969.)

If significant energy is to be extracted as a laser pulse then pumping must be done in a time short compared to that required for the triplet state to build up and so fast flashlamps are required with a risetime typically less than 100 ns. (Measures, 1984, p. 193.) The flashlamp and associated capacitors must therefore be designed to have a very low inductance. In order that thermal inhomogeneities do not degrade the cavity it is also necessary to circulate the dye through a heat exchanger. A schematic diagram of a flashlamp pumped dye laser is shown in figure 6.2. The Phase-R DL 2100B is a laser of this type.

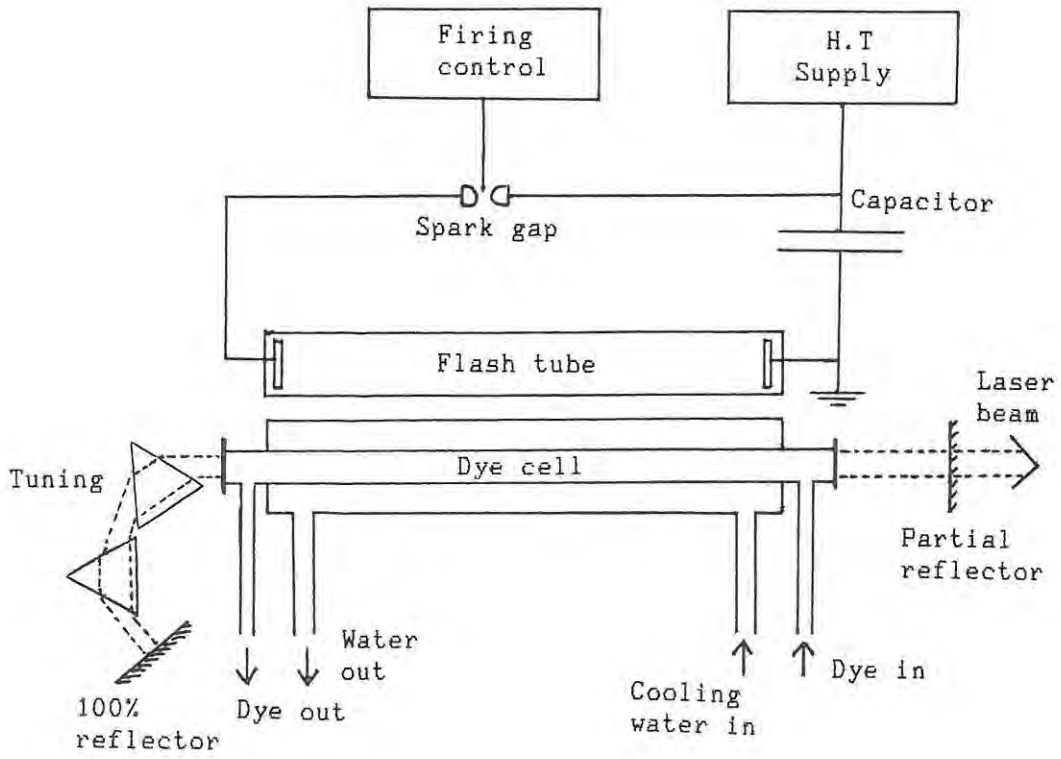


Figure 6.2 Schematic diagram of a flashlamp pumped dye laser.

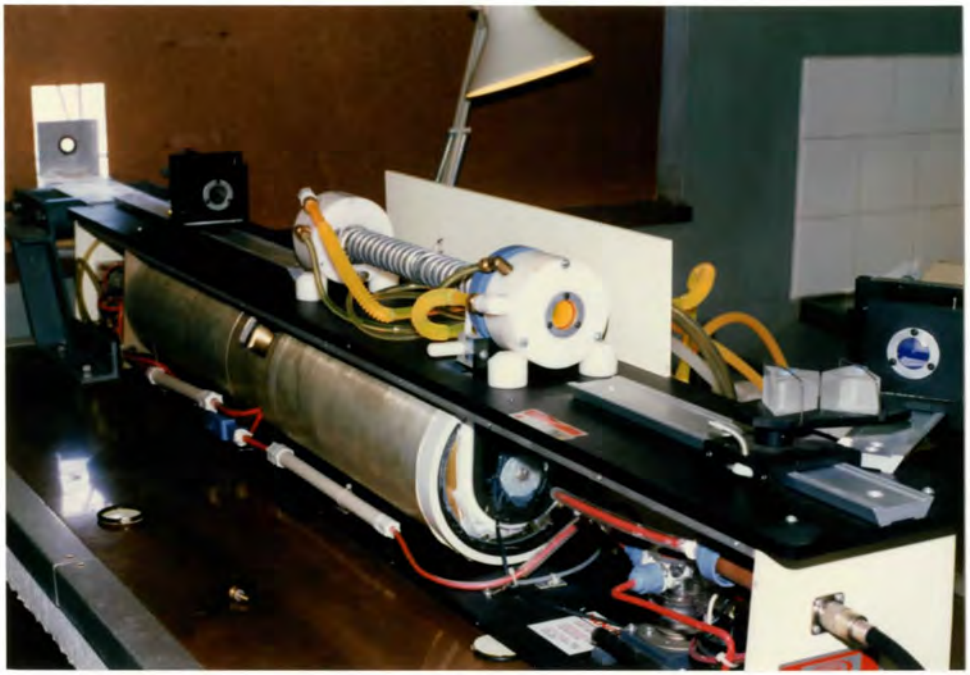


Figure 6.3 DL 2100B laser without the flashlamp and capacitor covers.

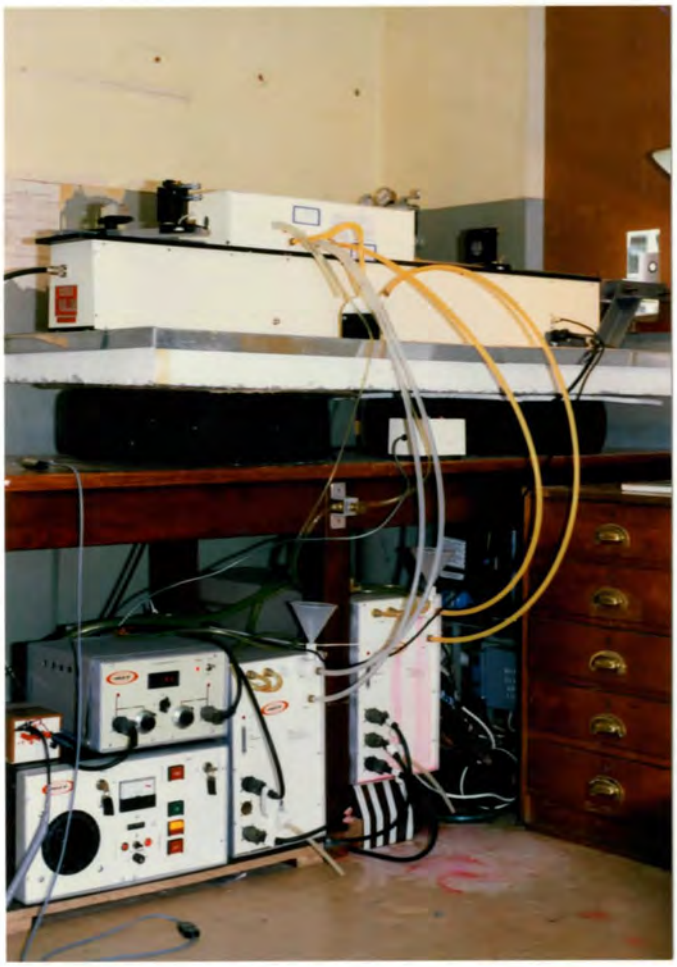


Figure 6.4 DL2100B laser with control console and solvent circulators

Figure 6.3 shows the laser with side panel and flash-lamp cover removed. The DP-60 prisms used for tuning and the 100% reflector are visible on the right. Figure 6.4 shows the laser on its table with the control console and solvent circulators underneath.

Tunability is achieved by making the cavity wavelength specific. This can be done by including one or more prisms as in the Rhodes DL 2100B or by use of a tilted diffraction grating. (For example Soffer and McFarland, 1967.) Tuning as well as narrowing of the spectral range of the pulse can be achieved by placing tilted etalons in the cavity. (For example Gibson, 1969.)

6.3 Adjustment and Performance of the Rhodes LIDAR Laser

6.3.1 Choice of wavelength and dye concentration

The choice of dye depends on the wavelength required. As monitoring of the mesospheric sodium layer is a future aim it was decided to operate the laser at 589,0 nm. The best dyes for this wavelength are the rhodamine family and rhodamine 6G (R6G) is being used at present. The concentration recommended by Phase-R Corporation is from 0,5 to 2,5 x 10^{-4} moles per litre and the dye solution that was found to work satisfactorily was 0,5 x 10^{-4} moles per litre of R6G in methanol. As recommended by Phase-R Corporation a glass fibre filter has been included between the dye circulator and dye cell to remove small bubbles and any debris from the dye.

6.3.2 Dye Lifetime

R6G is susceptible to degradation by the short-wavelength ($\lambda < 300$ nm) component of light from the xenon flashlamp. In the Rhodes LIDAR laser a high-rate adaptor has been installed which is a co-axial tube surrounding the quartz dye cell and carrying distilled water maintained at the same temperature as the dye. This permits operation up to 1 Hz. Calkins et al (1982) have found that the addition of 1,0 gram per litre of caffeine absorbs the damaging wavelengths and greatly prolongs dye lifetime. Accordingly caffeine at this concentration has been added to the cooling water circulating through the flashtube. The lifetime of a particular dye solution depends on the energy at which the flashtube

is fired but with caffeine in the water there was a noticeable improvement and the dye only requires changing every few thousand shots.

6.3.3 Alignment and tuning

If the laser is used in a flat/flat configuration with no tuning element in the cavity then alignment is fairly simple and is achieved by placing covers with central pin-holes over the mirrors and dye cell windows and passing the beam of a helium-neon (He-Ne) laser through the assembly. The mirrors are then aligned by making the reflected spots co-axial. When the prisms are included this method is inapplicable due to the difference in wavelength of the tuned cavity and the He-Ne laser and has to be done by starting with an aligned flat/flat configuration, mounting the prisms and then firing the laser repeatedly while adjusting only the 100% reflector until maximum output at the desired wavelength is obtained.

The maximum voltage recommended on the capacitors is 18 kV. When so charged the manufacturers state that the laser can produce several joules of energy per pulse. This results in an extreme danger to the eyes of people nearby, even from diffusely reflected light and so low-pass goggles with a 0,5% transmission at 589 nm are worn. This makes it impossible to examine the beam quality but it was found that when operated at just above the lasing threshold of 8,0 kV the beam could be looked at when shone onto a dark surface. This made alignment far easier. At low energies the beam has a doughnut shape since light from the flash-tube is absorbed before it reaches the centre of the dye cell. At operating voltages the dye concentration is correct if the beam produces an even burn on a suitable surface such as black polaroid film or a cheaper black hard-cover exercise book.

Once lasing has been achieved, tuning is accomplished by rotating the 100% reflector about a vertical axis. The change in wavelength is easily visible to the eye as the colour goes from a brittle yellow-green to an orange-red. Precise wavelength adjustment is accomplished by observing light reflected from one of the faces of the prisms through a spectrometer and diffraction grating previously adjusted so that the the cross-hairs lie on the 589,0 D line from a sodium lamp.

When laser light was viewed through the spectrometer several lines were sometimes seen centered on 589 nm but spanning up to 7 nm. These lines also varied from shot to shot with no adjustment having taken place. This effect is described by Sorokin (1969) and attributed to non-uniform heating of the dye causing random changes in its refractive index. It is also suggested that the laser jumps from one line to another rather than emitting all wavelengths simultaneously. (Furumoto and Ceccon, 1968.) As the light falls within the pass-band of the receiver filter this matter was not pursued further but if resonant scattering and absorption studies are to be made with the Rhodes LIDAR a great deal of attention will have to be given to beam quality and wavelength stability.

6.3.4 Beam divergence.

The manufacturer's specifications for the DL 2100B laser give the maximum full angle beam divergence as 3 mrad. The beam divergence was measured by tuning the laser to 589 nm, aligning the cavity by allowing the beam to fall on a dark surface 17 m away and maximising beam intensity and then measuring the beam diameter at 2 m intervals from the output reflector out to a distance of 16 m. The laser was charged to 12 kV but the beam could be examined by the naked eye while working in bright daylight to constrict the pupils. A graph of the results was plotted and a "worst case" line drawn. This yielded a beam divergence of 1,9 mrad. As beam divergence is dependent on such variables as alignment, dye concentration and age and as alignment tends to drift during operation a value of 2 mrad was taken for the beam divergence when calculating intersection heights with the receiver beam.

6.3.5 Pulse Energy.

The energy in each pulse of the laser beam was measured with an RjP-736 energy probe manufactured by Laser Precision Corp. This works by generating a current in a slice of ferroelectric material posessing a permanent electric polarization, the current being proportional to the time rate of change of the element temperature. (Operating instructions, p. 2) A circuit in the probe integrates the current and produces a voltage step proportional to the pulse energy.

In addition to the laser beam the laser produces considerable spontaneous fluorescence, which is not well collimated, from the dye and in order not to include this energy in the measurements, the probe was placed 10 m from the output reflector. The energy in 10 shots of the laser tuned to 589 nm at several capacitor voltages and with fresh R6G dye with a concentration of 0,5 x 10^{-4} moles per litre were averaged and are plotted on the graph in figure 6.5. At 17 kV the laser was producing 0,455 J per pulse.

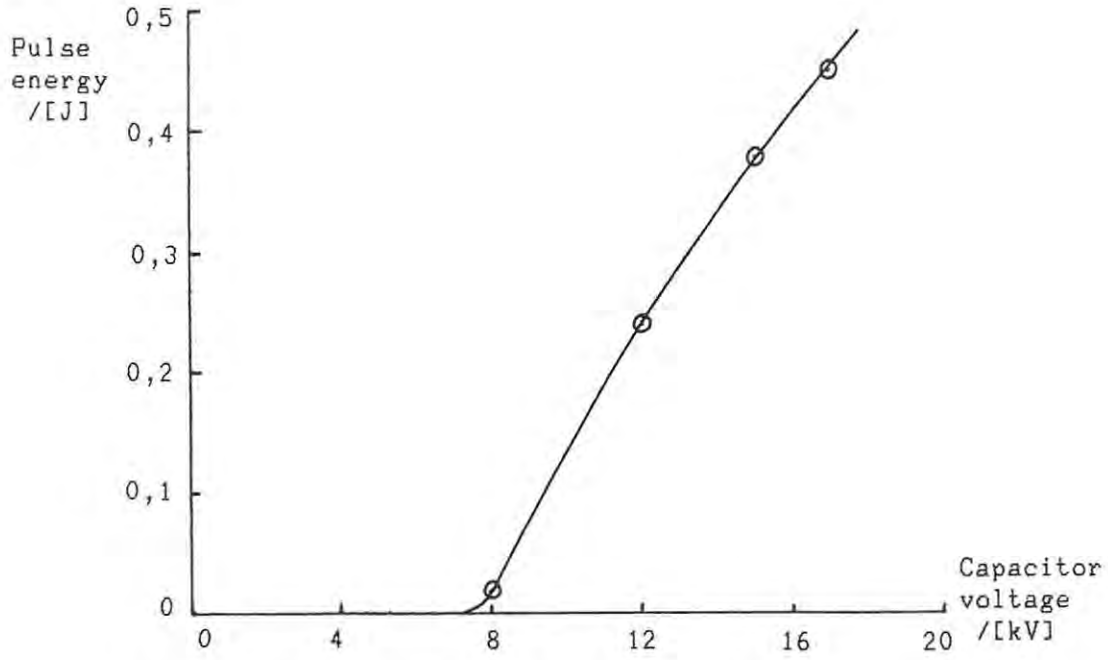


Figure 6.5 Pulse energy versus capacitor voltage for the DL 2100B.

Phase-R recommends not testing the quality of the beam with a burn closer than 4" from the output reflector. A burn close to the output reflector includes the fluorescence energy. This may explain why the measured energies fall rather short of the manufacturer's specifications. The energy of the beam is nevertheless ample for present LIDAR work and exceeds that used by many other LIDARs.

6.3.6 Pulse duration.

If the LIDAR is to be precise as a ranging device it is necessary that the pulse be as short as possible and certainly less than the sampling time of 1 μs. The duration of the pulses was measured by allowing the

attenuated beam to fall on an HP 5082-4203 PIN photodiode which with a rise-time of 1,5 ns is sufficiently fast to follow the pulse. The voltage across a 50 Ω resistor in series with the reverse biased diode was recorded with a digital storage oscilloscope with a bandwidth of 100 MHz. A photograph of a typical trace is shown in figure 6.6. The FWHM duration is 0,51 μs and the rise time close to 0,1 μs. This is close to the manufacturer's specifications of 0,4 μs and 0,1 μs respectively and shows the laser to have a pulse duration suitable for the Rhodes LIDAR.

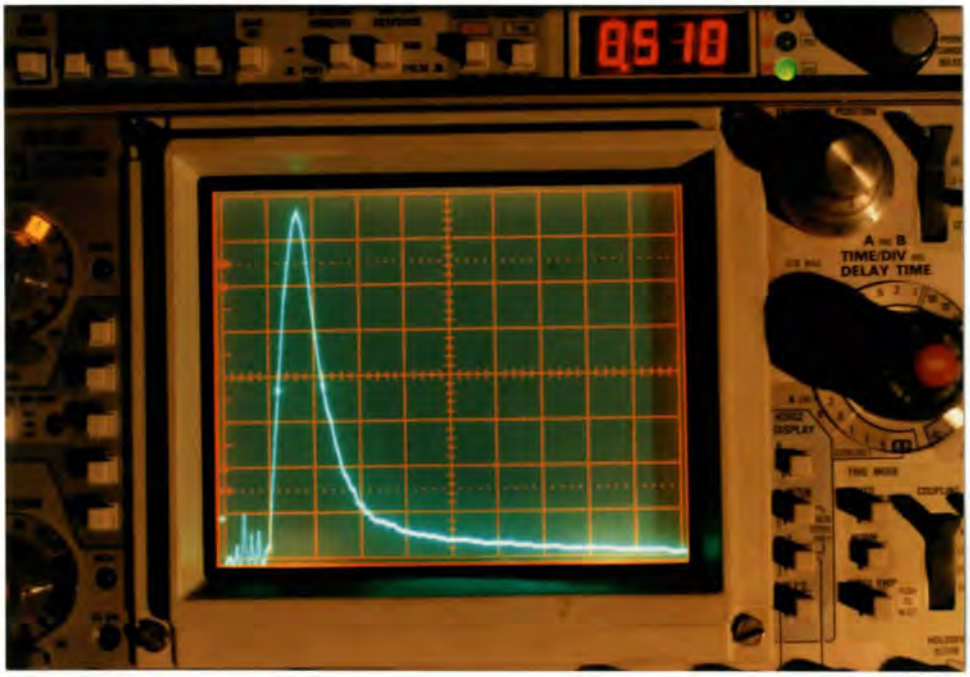


Figure 6.6 Pulse duration of the DL 2100B laser.

The performance of the DL 2100B matches or exceeds that of most lasers in other LIDARS in respect of beam energy and pulse duration but falls short in respect of collimation, bandwidth and spectral stability. These are facets that will require attention in the future.

6.4 Alignment of the Laser Transmitter with the Receiver

In order to make measurements of scattering by the atmosphere the laser has to be aligned with the receiver as detailed in chapter 2. When the laser is fired, some of the light falls on the PIN photodiode shown in figure 3.5 and produces a pulse which triggers the photon counting system (PCS). The return from the atmosphere is then presented

graphically on the IBM-PC screen. This allows adjustments to be made to the prism on the end of the aluminium beam protruding from the laser room in order to direct the laser light into the receiver beam.

Table 2.2 shows that the 5,5 mm diameter aperture in the receiver is the most suitable for observing the stratosphere where aerosols may be expected. This is selected, the laser fired repeatedly and the prism directing the beam upwards adjusted until three conditions are met simultaneously.

1. A return commences from near the 5 km level.

2. The maximum count rate of the PMT does not exceed 14 MHz indicating that the tube is not being driven far enough into saturation to produce afterpulses.

3. A figure of merit calculated by summing the counts in several range bins near the 20 km level is maximized.

This completes the alignment.

CHAPTER 7

Lets do it. (Gary Gilmore.)

Operation of the Rhodes LIDAR

7.1 Setting Up

The LIDAR is prepared for operation in the late afternoon when the sun is no longer high enough to be damagingly focussed by the receiver mirror and it can be seen that the sky will be clear and the likelihood of dew on the mirror is small, at least for the early part of the night. The power supplies for the photomultiplier tube (PMT) and pulse amplifier are set up close to the receiver and cables are run out. The curtains are hung on the frame surrounding the receiver.

A check is made that the receiver is pointing at the zenith and focussed on infinity as described in chapter 2 and a suitable aperture for the height to be observed is selected from table 2.2, generally that with the 5,5 mm diameter.

When it is dark the power supplies to the PMT and amplifier are switched on and the pulse shape and background count rate checked by examining the pulses with a fast oscilloscope and triggering the photon counting system repeatedly so that the counts per microsecond can be estimated. Generally the background is considered low enough for LIDAR observations to begin if it is less than 1 count every 2 μs.

The laser is switched on and the temperature in the solvent circulators is allowed to stabilize at 17°C. The integrity of the flashlamp is confirmed by checking the threshold at which it fires. It normally fires at 6 kV but this threshold will increase as the end of its lifetime nears or if one of the quartz-metal seals should develop a leak. Should the tube shatter the methanol solvent would probably catch fire and the dye circulator could then pump most of the 2,5 l in the reservoir onto the flames. For this reason a CO_2 fire extinguisher is kept handy and the laser also posesses a noise shutdown interlock which responds to any loud sound.

The laser is then fired at a dark board 17 m away and the alignment checked and the wavelength adjusted to 589 nm as described in chapter 6.3.3. Finally the right-angle prism is installed on the end of the aluminium beam protruding from the window of the laser room and the laser aligned with the receiver as described in chapter 6.4.

7.2 Gathering Data

Observations are normally carried out under clear sky conitions but on the occasions when thin high-altitude cirrus is present, running the LIDAR with the laser operating at low power produces returns from the clouds that show the instrument is working and provide a measure of its precision.

Every time the laser is fired the PCS is triggered simultaneously and within 0,3 s the data has been transferred to the IBM-PC and a scattering profile of the atmosphere displayed on the screen. The laser can operate at 1 Hz but is generally run at 0,5 to 0,3 Hz. Figure 7.1 shows the progressive dissipation of some thin high-altitude cirrus cloud over a period of about 30 minutes while the laser was being run with the capacitors charged to 9 kV.

The top left graph shows the accumulated counts from 19 pulses. The return from clear air starts at about 4 km where the laser and receiver beams begin to overlap, reaches a maximum at around 6 km and then decreases. In the absence of cloud the decrease would continue into the background but the cirrus produces a strong return starting at about 8 km. Some stratification in the cirrus is visible and the cloud top appears to be at 10 km. There is the suggestion of some attenuated return from the clear air above 10 km. Because the count rate from the cirrus is so high it is not possible to obtain quantitative information about the optical density of the clouds from these data.

In the middle right and bottom left graphs the band of cirrus produced a narrow spike that is only two range bins wide. This corresponds to cloud not more than 300 m thick. That the LIDAR is able to place counts of photons returning from such a narrow feature consistently into the same bins gives confidence in the precision of the instrument.

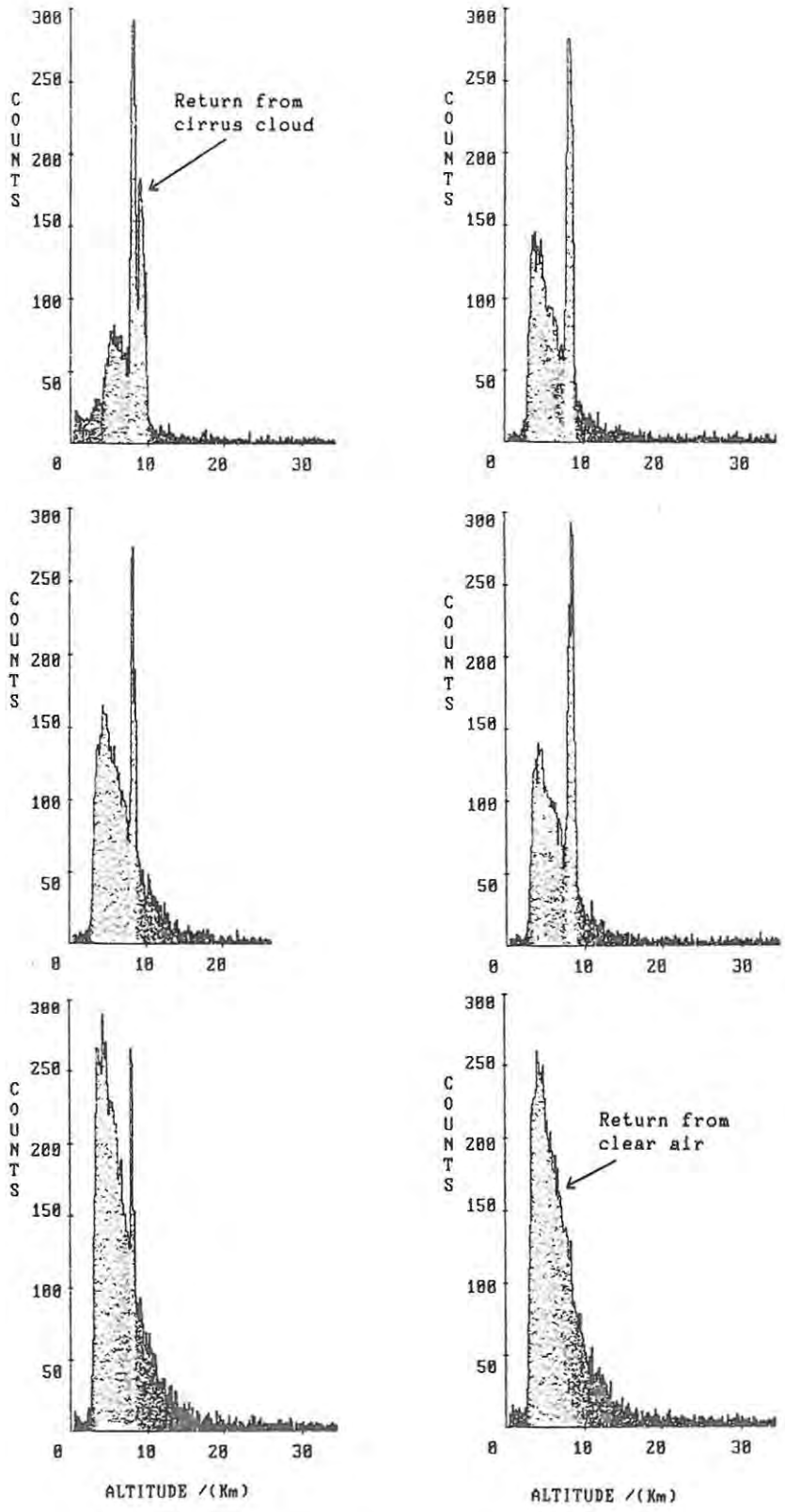


Figure 7.1 Progressive dispersion of high altitude cirrus cloud.

Top left	19 shots	Top right	29 shots
Middle left	26 shots	Middle right	22 shots
Bottom left	40 shots	Bottom right	40 shots

Eventually the cirrus dispersed as shown by the bottom right hand graph.

There appears to be some structure in the downward sloping portion of the last graph in figure 7.1. This is visible because the laser is operating at low power and could be due to dust or aerosols in the troposphere. However altitudes of less than 10 km are not investigated further in this thesis.

Once the sky has cleared the laser power can be increased and returns obtained from greater altitudes. Figure 7.2 shows how the computer screen appears after 11 shots into a clear sky. The returns from the first three shots can be seen as separate lines of dots in the lower portion of the graph. Figure 7.3 shows how the screen appears after 95 shots. On this occasion the laser was delivering about 360 mJ per pulse and the signal can be seen to merge with the noise between the ranges of 40 and 50 kilometers.

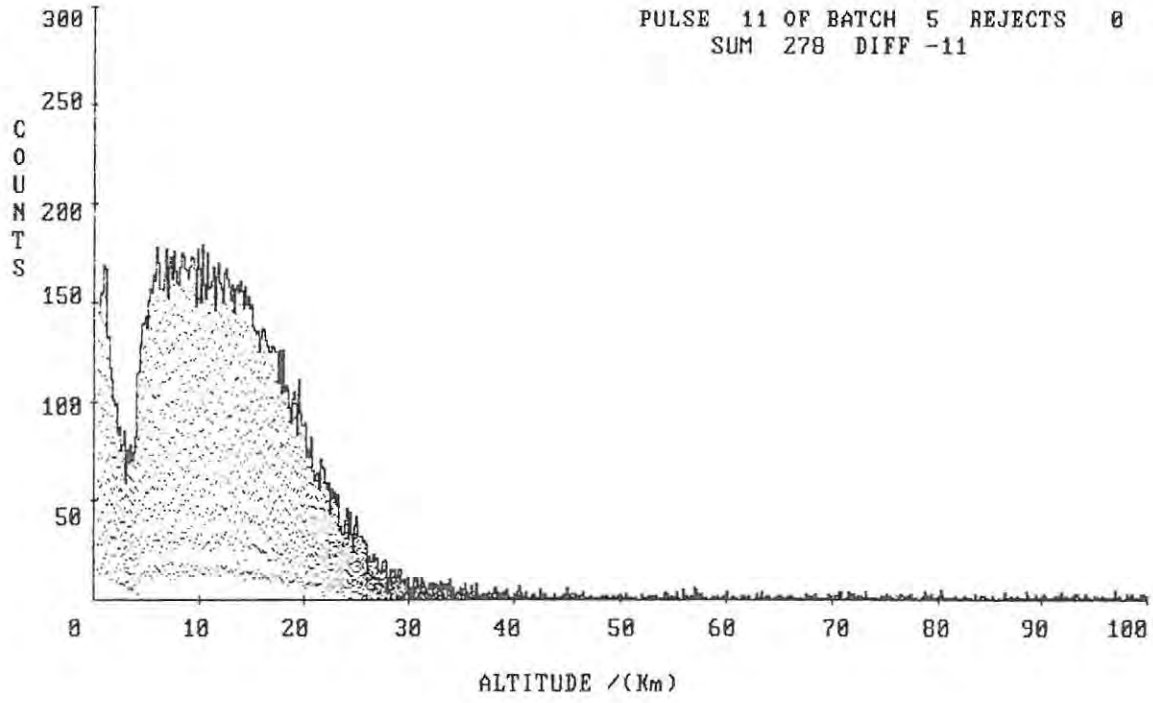


Figure 7.2 Appearance of the computer screen after 11 shots into a clear sky.

During the course of an observing session the laser is fired typically 100 times and the batch of integrated counts then saved to disk. Several sets, typically 10, of different batches of counts are then recorded while the laser power and optics remain unadjusted. Some 1000 shots therefore yield data which can then be analysed for temperature and scattering ratio information. If weather conditions remain favourable several sets of data can be gathered with the laser operating at different powers in order to yield information about different regions of the atmosphere.

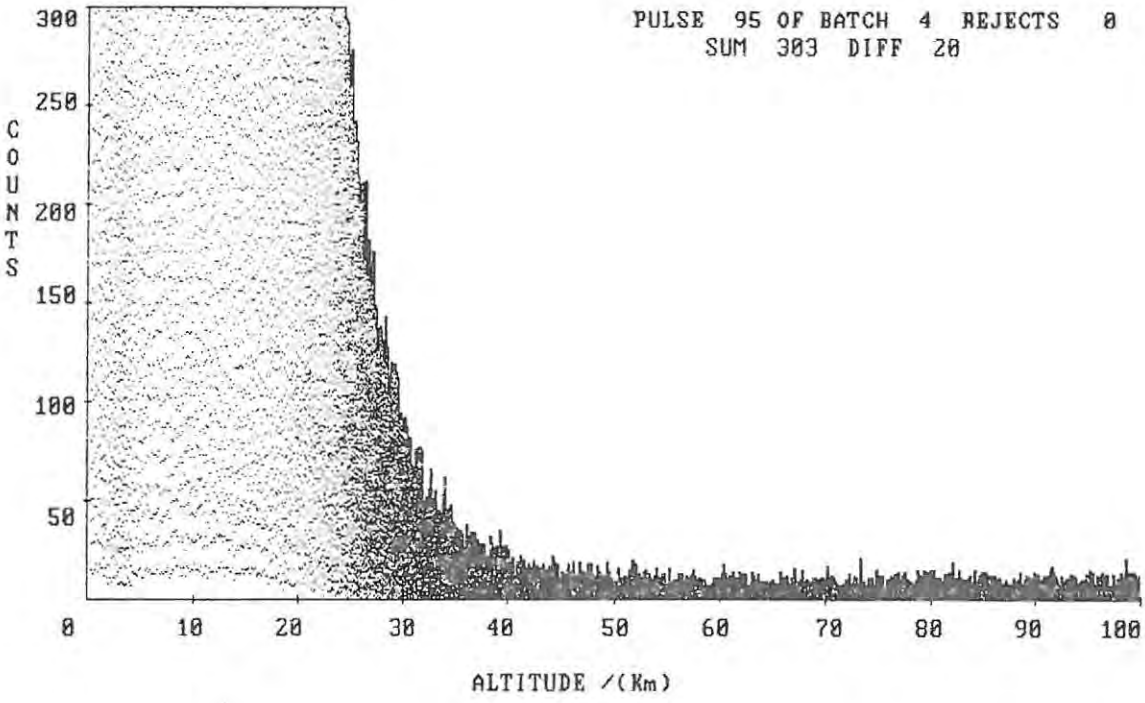


Figure 7.3 Appearance of the computer screen after 95 shots into a clear sky and shortly before saving the data to disk.

At the end of the observing period the equipment is shut down, the cover replaced over the receiver mirror and the laser log book updated. Early the next morning a telephone call is made to the Weather Bureau at the airport in Port Elizabeth to obtain temperature, pressure and altitude information from the balloon that was released at midnight.

The information provided extends from 10 km up to about 26 km where the balloon usually bursts.

The computer program which gathers the data permits the storing of the integrated counts on disk for future analysis. The algorithms used to analyse the data are described in chapter 8 and the results of this analysis in chapter 9.

7.3 Safety of Aircraft Overflying the LIDAR Site

Concern was expressed by the Commissioner for Civil Aviation that if an aeroplane was flying over the LIDAR site at the moment the laser was fired, the occupants could suffer serious eye damage. In order to persuade him that this was extremely unlikely, letters were written to Dr G. Megie in France, Dr B.R. Clemesha in Brazil and Professor L Thomas in Wales who head LIDAR research groups in areas where there is a great deal more air traffic. The replies included the information that in order to be blinded, one would have to be looking directly into the beam, which for a vertical LIDAR would mean the aeroplane would have to be flying upside-down. It had been calculated that for the Brazil LIDAR, situated some 2 km from an airfield, there was a chance of blinding an aviator once in a billion years if all the planes crossing the LIDAR site did so upside down! Nevertheless, at all these sites, a watch is kept for aircraft and firing is stopped if one should come close by. Permission was eventually received from the Commissioner for Civil Aviation to operate the LIDAR at Rhodes provided a similar watch is kept.

Chapter 8

LIDAR Theory and Algorithms for the Analysis of LIDAR Data

After the counts for each range bin have been integrated over several shots of the laser and stored on floppy disk they can be processed to yield profiles of atmospheric temperature and scattering ratio.

8.1 General LIDAR Theory for Non-resonant Atmospheric Scattering

When monochromatic light propagates through the atmosphere in the absence of particles large compared with its wavelength such as dust grains or water droplets, it is both Rayleigh and Mie scattered. The Rayleigh scattering is due to gas molecules and the Mie scattering to aerosols of particles with dimensions similar to the wavelength of the light. The transmitted and scattered light have the same wavelength.

For a pulsed LIDAR in which only scattered light reaches the receiver the general form of the lidar equation is (Measures 1984 p. 238)

$$\Delta P(\lambda,R) = \int J(\lambda,R,r)\ \Delta\lambda\ \Delta R\ p(\lambda,R,r)\ dA(R,r)$$

in which $\Delta P(\lambda,R)$ is the increment of signal power received by the detector in the wavelength interval $\lambda,\lambda+\Delta\lambda$.

$J(\lambda,R,r)$ represents the laser induced spectral radiance at wavelength λ, at position r in the target plane located at range R, per unit range interval;

$dA(R,r)$ represents the element of target area at position r and range R; and

$p(\lambda,R,r)$ represents the probability that radiation of wavelength λ emanating from position r at range R will strike the detector.

If the backscattered light is due only to Rayleigh and Mie scattering and if some method of counting the returning photons exists then this

equation can be written in the following form (Hauchecorne and Chanin, 1980)

$$N(z_i) = \frac{N_o A K R_q T^2(z_o,z_i)}{4\pi(z_i-z_o)^2} \cdot [n_r(z_i)\beta_r+n_m(z_i)\beta_m(z_i)]\Delta z \qquad [1]$$

where $N(z_i)$ is the number of detected photons from one laser pulse, from a layer of thickness Δz centered at a height z_i.

N_o is the number of photons emitted per laser pulse

A is the telescope area

K is the optical efficiency of the lidar system

R_q is the quantum efficiency of the photomultiplier tube.

$T(z_o,z_i)$ is the atmospheric transmission between the altitude of the lidar site and the height of the emitting layer.

$n_r(z_i)$ and $n_m(z_i)$ are the concentrations of air molecules and aerosols and;

β_r and $\beta_m(z_i)$ are the Rayleigh and Mie backscattering cross-sections

If the height range is such that the contribution of Mie scattering is negligible, then the returned light depends only on the density $\rho(z)$ and is given by the expression (Chanin and Hauchecorne, (1984) equation (2) corrected)

$$\rho(z) = \frac{C[S_L(z)-B(z)]}{T^2(z_o,z)}$$

where $S_L(z)$ is the signal coming from altitude z in a constant solid angle and corrected for any non-linearity in the photomultiplier tube if it is close to saturation.

$B(z)$ is the background signal due to dark current and sky background.

C is a normalisation constant that can be evaluated by fitting the data to a model eg CIRA 1972 or balloon data at a particular altitude.

The atmospheric transmission term between the LIDAR and the altitude z, $T^2(z_o,z_i)$, is complicated and difficult to evaluate. That part of the atmosphere below the lowest range bin of interest, the troposphere, is where most scattering takes place due to the low altitude aerosols and comparatively high gas density. The transmission term for this region is therefore constant provided the readings are taken over a reasonably short time and can be lumped with the constant C.

Within the region of interest, from 10 km through the stratospheric aerosols to about 25 km and then up through very clear air to the highest range bin measurable at around 40 km at present, two factors affect the estimation of atmospheric density from scattered light.

1. The energy of the beam at the top of the range is not as great as at the bottom due to attenuation by scattering of the beam within the range.

2. The strength of returns from the bottom of the range, detected at the ground, is greater than that of equal returns from the top since the light from the bottom has not had to propagate through the attenuating layers of the range itself.

In the evaluation of atmospheric density from the intensity of scattered light it is necessary to work from the top of the range, where it is assumed that the air is aerosol free and only Rayleigh scattering takes place, down to the bottom.

The author has estimated the effect of 1. by taking the energy of the beam in the i^{th} range bin being evaluated to be the energy of the beam in the range above $E_{(i+1)}$ plus that part of the beam scattered in that range, $e_{(i+1)}$. That is

$$E_i = E_{(i+1)} + e_{(i+1)}$$

In the $(i+1)^{th}$ bin the fraction of the beam that is scattered is equal to the fraction of the sky "covered" by air molecules in the bin. This is equal to the total scattering cross-section of the air molecules in a column of unit cross section and height 150 m, this being the height of a range bin and can be evaluated as follows.

The Rayleigh cross section of an air molecule for scattering in all directions is given by (Measures 1984 p 344)

$$\sigma^R(\lambda) = \frac{8\pi}{3}\sigma_\pi^R(\lambda)$$

where $\sigma_\pi^R(\lambda)$ is the Rayleigh backscatter cross-section given by

$$\sigma_\pi^R(\lambda) = 5{,}45\left[\frac{550}{\lambda(nm)}\right]^4 \times 10^{-31} \quad m^2.sr^{-1}$$

which for λ = 589 nm evaluates to $\sigma^R(589) = 3{,}47 \times 10^{-31}$ $m^2.sr^{-1}$

The total attenuation coefficient at this altitude is

$$\overline{\chi^R} = N^R\sigma^R(589) \quad m^{-1}$$

where N^R is the number density of air molecules in the range bin. Over a distance of 150 m the attenuation is given by $150.N^R.\sigma^R(589)$ which is the proportion of the incident beam attenuated, so that

$$\frac{e_{(i+1)}}{E_{(i+1)}} = 150.N^R.\sigma^R(589)$$

The energy of the beam at the top of the range in the n^{th} bin can be normalised to unity and the energy of the beam in successively lower levels calculated. It turns out that the correction is small but it is interesting to make it nevertheless.

The author has estimated the effect of 2. as follows. Consider two sources A and B equidistant in an attenuating medium and transmitting in unit time numbers of photons $N_S(A)$ and $N_S(B)$ in the direction of the receiver as shown in figure 8.1. Then

$$\frac{N_S(A)}{N_S(B)} = \frac{N_R(A)}{N_R(B)}$$

where $N_R(A)$ and $N_R(B)$ are the numbers of photons received from each source.

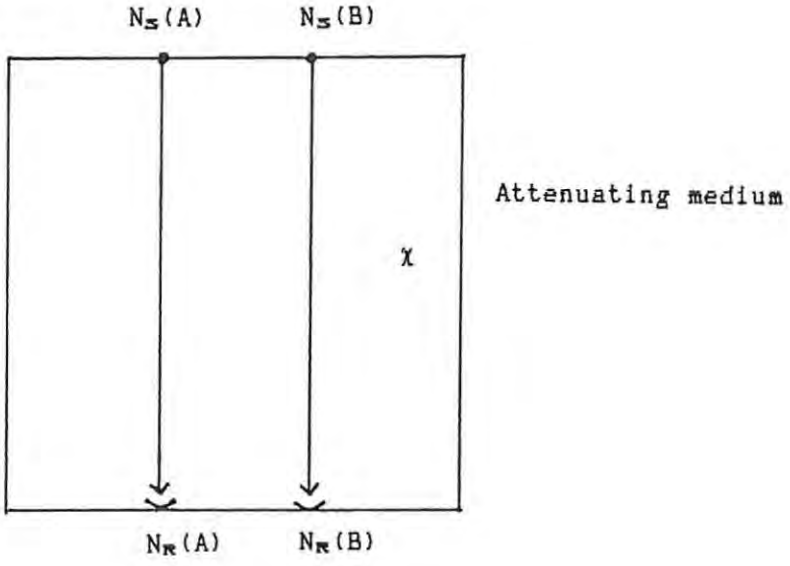


Figure 8.1 Sources and receivers in an attenuating medium.

Suppose the source $N_S(B)$ is now moved a distance Δz towards the receiver as shown in figure 8.2.

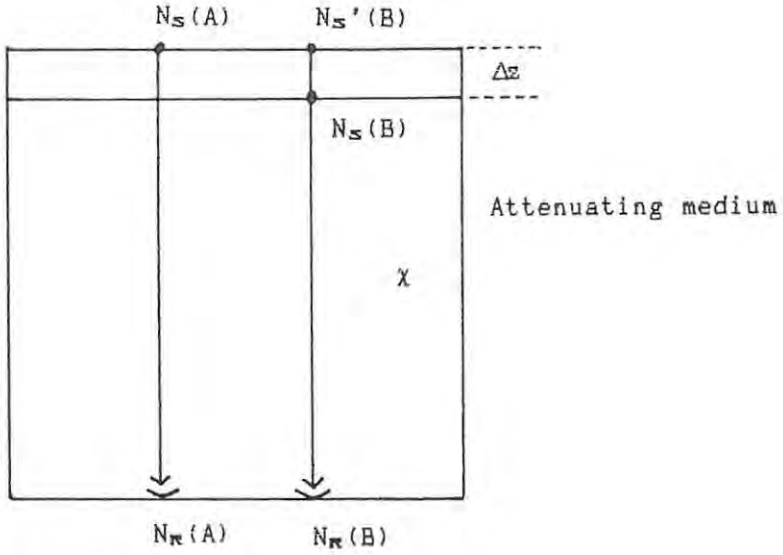


Figure 8.2 Sources and receivers with different separation in an attenuating medium

In order to be able to compare the strengths of the sources $N_S(A)$ and $N_S(B)$ knowing only $N_R(A)$ and $N_R(B)$ we reconstruct the source $N_S'(B)$ at the same distance from the receiver as $N_S(A)$ where

$$N_S(B) = N_S'(B) - \Delta N$$

ΔN is the number of photons absorbed in the distance Δz. From the definition of the extinction coefficient χ and taking the number of photons to be proportional to the light intensity, since the light is monochromatic

$$\Delta N = \chi N_S'(B) . \Delta z$$

$$\therefore \quad N_S(B) = N_S'(B) - \chi N_S'(B) \Delta z$$

$$\therefore \quad N_S'(B) = \frac{N_S(B)}{1-\chi\Delta z}$$

so that with $$\frac{N_R(B)}{N_R(A)} = \frac{N_S'(B)}{N_S(A)}$$

$$\frac{N_R(B)}{N_R(A)} = \frac{N_S(B)}{N_S(A)} \cdot \frac{1}{1-\chi\Delta z}$$

Inversion yields $$N_S(B) = \frac{N_R(B)}{N_R(A)} . N_S(A) . (1-\chi\Delta z)$$

Now if $N_S(B)$ results from scattering after illumination by a laser pulse it is proportional to the density of the layer in which the emitter is situated. Indeed if $N_S(A)$ is taken to be at the top of the range in bin n and $N_S(B)$ is taken to be in the next lowest bin, number (n-1) then.

$$\frac{N_S(B)}{N_S(A)} = \frac{\rho(n-1)}{\rho(n)} \quad \text{and letting} \quad \frac{N_R(A)}{N_R(B)} = \frac{N(n)}{N(n-1)}$$

So $$\frac{\rho(n-1)}{\rho(n)} = \frac{N_R(B)}{N_R(A)} (1-\chi_n\Delta z)$$

or $$\rho(n-1) = \rho(n).\frac{N(n-1)}{N(n)}. (1-\chi_n\Delta z)$$

similarly $$\rho(n-2) = \rho(n-1).\frac{N(n-2)}{N(n-1)}. (1-\chi_{n-1}\Delta z)$$

so $$\rho(n-2) = \rho(n).\frac{N(n-2)}{N(n)}. (1-\chi_n\Delta z).(1-\chi_{n-1}\Delta z)$$

Generally, for the j^{th} layer

$$\rho(j) = \rho(n).\frac{N(j)}{N(n)}.(1-\chi_n\Delta z).(1-\chi_{(n-1)}\Delta z)\ldots(1-\chi_{(j+1)}\Delta z)$$

Since $\chi\Delta z << 1$ this expression becomes

$$\rho(j) = \rho(n).\frac{N(j)}{N(n)}.(1-\sum_{i=j+1}^{n}\chi_i\Delta z). \qquad [2]$$

and we see that in an expression to calculate the density of the j^{th} layer based on the comparative counts from the j^{th} and the n^{th} layers, multiplication by the factor

$$(1-\sum_{i=j+1}^{n}\chi_i\Delta z)$$

will help correct for attenuation of the returning light in the range itself and in an expression to predict the number of photons from the j^{th} layer, N(j), the attenuation correcting term becomes, by a rearrangement of [2],

$$X(n-j) = \frac{1}{(1-\sum_{i=j+1}^{n}\chi_i\Delta z)}$$

Again the corrections turn out to be small.

Using the expressions developed above, the number of photons received by the detector due to both Rayleigh and Mie scattering in the j^{th} layer can now be written

$$N(j) = K.f[\rho(j),z(j),E(j),X(n-j)]$$

where $\rho(j)$ is the density of the jth layer

$z(j)$ is the distance to the jth layer

$E(j)$ is the energy of the beam at the j^{th} layer normalised to unity at the top of the range.

$X(n-j)$ allows for attenuation of the returning signal as it passes through the range

K is a constant which includes the area of the collector, the efficiency of the receiver and the transmittance between the bottom of the range and the ground.

If the laser beam is wholly within the receiver beam so that the amount of light gathered from an illuminated layer depends on the square of its distance from the receiver then the LIDAR equation in the form of equation [1] applies and can be re-written

$$N(j) = \frac{K.\rho(j).E(j).X(n-j)}{z^2(j)}$$

At the top of the range j = n and with the beam energy normalised to unity, E(n) = 1. The returning light has not benefitted from not having travelled through any of the range so X(n-j) = 1 /1

and

$$N(n) = \frac{K.\rho(n).1.1}{z^2(n).1}$$

$\rho(n)$ is known from tables of atmospheric density, eg CIRA 1972 or balloon measurements, so K can be evaluated.

Writing the equation for the number of photons received from the $(n-1)^{th}$ layer, N(n-1), and solving for $\rho(n-1)$ yields

$$\rho(n-1) = \frac{N(n-1).z^2(n-1).(1-\chi_n \Delta z)}{K(E_n+e_n)}$$

and generally for the density of the j^{th} layer

$$\rho(j) = \frac{N(j).z^2(j).(1-\sum_{i=j+1}^{n} \chi_i \Delta z)}{K.(E_{j+1}+e_{j+1})}$$

8.2 Determinations of Atmospheric Temperature Profiles

At a height above 30 to 35 km, or lower if conditions permit, the air is clear enough for the Mie scattering due to aerosols to be negligible and so for a lidar pulse the backscattered echo is proportional to atmospheric density.

If it is assumed that turbulence in the atmosphere causes a negligible departure from hydrostatic equilibrium and that it obeys the perfect gas law then a temperature profile can be deduced from a measure of the atmospheric density (Sandford 1967).

The following expressions for evaluating a temperature profile are taken from Chanin and Hauchecorne, (1984).

Using the perfect gas law the air pressure P(z), density $\rho(z)$ and temperature T(z) are related by

$$P(z) = \frac{R.\rho(z).T(z)}{M} \qquad \text{and} \qquad dP(z) = -\rho(z).g(z).d(z)$$

where M is the air mean molecular weight and R is the universal gas constant.

Combining these two equations leads to

$$\frac{dP(z)}{P(z)} = \frac{M.g(z).dz}{R.T(z)} = d(\mathrm{Log}P(z))$$

where Log refers to the natural logarithm.

Assuming the temperature and pressure to be constant across the i^{th} layer, the pressure at the top and bottom of the layer are related by

$$\frac{P(z_i-\Delta z/2)}{P(z_i+\Delta z/2)} = \exp \frac{M.g(z_i)}{R.T(z_i)} .\Delta z$$

and the temperature is expressed as

$$T(z_i) = \frac{M.g(z_i).\Delta z}{R.\text{Log } P.(z_i-\Delta z/2)/P(z_i+\Delta z/2)}$$

The pressure at the top of the highest layer measurable is fitted to a standard from an independent source. The top and bottom pressures of the i^{th} layer are then:

$$P(z_i+\Delta z/2) = \sum_{j=i+1}^{n} \rho(z_j).g(z_j).\Delta z + P_m(z_n + \Delta z/2)$$

$$P(z_i-\Delta z/2) = P(z_i+\Delta z/2) + \rho(z_i).g(z_i).\Delta z$$

Setting

$$X = \frac{\rho(z_i).g(z_i).\Delta z}{P(z_i+\Delta z/2)}$$

the temperature is then given by

$$T(z_i) = \frac{M.g(z_i).\Delta z}{R.\text{Log}(1+X)}$$

The flow diagram for a computer program to calculate the vertical temperature profile based on the equations from Chanin and Hauchecorne, (1980) given above is shown in figure 8.3.

8.3 Determination of Aerosol Scattering Ratio Profiles

The above measurements of temperature depend on the assumption that the returned light is due only to Rayleigh scattering by air molecules. It was first shown by Junge and Manson (1961) that between about 15 km and 25 km there are layers of aeroticulates which also scatter light, this being by the Mie scattering process. The first lidar experiments on this Junge layer were conducted by Fiocco and Grams (1964). If the Mie scattering contribution is neglected then the calculated atmospheric density is too big and the temperature too small.

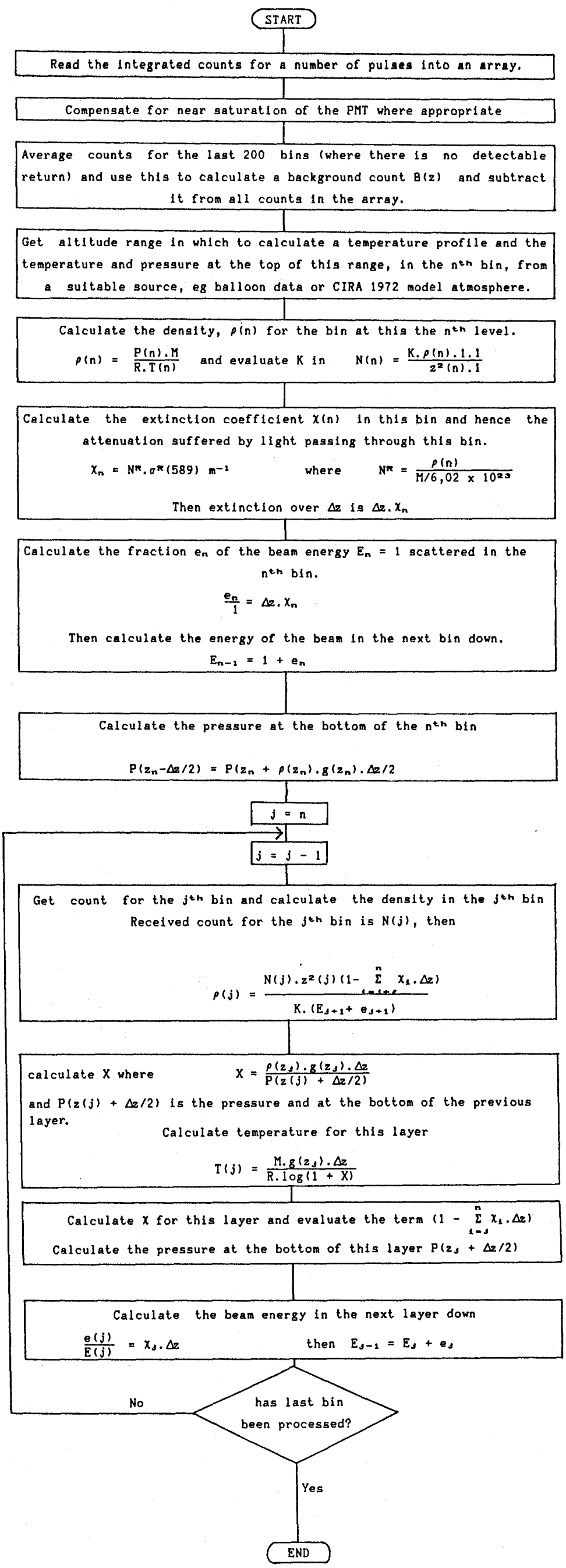


Figure 8.3 Flowchart for a computer program to calculate a temperature profile of the atmosphere. Symbols are as defined in the text.

If some alternative measurements of temperature and pressure are available for these regions, for example the CIRA 1972 model atmosphere for a fix at the upper level of 25 km or better still throughout the region from balloon-borne sensors, then molecular densities and the return from Rayleigh scattering can be computed. This allows the separation of Mie and Rayleigh scattering contributions. If further in-situ measurements of the number density of the aeroticulates are available or if measurements can be made at more than one wavelength, then their scattering cross sections can be obtained. (Northam et al. 1974 and Kerker 1976.)

If $\beta(R)$ is the atmospheric backscattering coefficient then

$$\beta(R) = \beta^{R}(R) + \beta^{M}(R)$$

and the scattering ratio is defined as (Measures, 1984 p. 344)

$$K_s(R) = \frac{\beta(R)}{\beta^{R}(R)}$$

where $\beta^{M}(R)$ and $\beta^{R}(R)$ represent the Mie and Rayleigh volume backscattering coefficients respectively at wavelength λ and range R.

If the return is due only to Rayleigh and Mie scattering, which is true if the wavelength of the laser is not resonant scattered by any of the species present, then (Measures 1984, p.357)

$$K_s(\lambda_L,R) \approx \frac{E(\lambda_L,R)}{E^{R}(\lambda_L,R)}$$

where $E^{R}(\lambda_L,R)$ is the lidar return in the absence of Mie scattering. In a system which counts returning photons the count is proportional to the total return $E(\lambda_L,R)$ and so the expression for the scattering ratio becomes

$$K_s(\lambda_L,R) \approx \frac{N(\lambda_L,R)}{N^{R}(\lambda_L,R)}$$

where $N^{R}(\lambda_L,R)$ can be calculated from independent measurements of atmospheric density. The calculation of density from temperature and

pressure measurements assumes that the contribution to the mass of the atmosphere by the aerosols is negligible.

In the stratosphere the laser beam and returning photons are attenuated by the aerosols as well as air molecules and the total extinction coefficient χ_e is given by (Measures, 1984 p. 344)

$$\overline{\chi}_e = \chi^R + \chi^M$$

The contribution from aerosols, χ^M, is difficult to estimate and will be neglected as the scattering ratio is found to be small at present (1987). χ^R alone will be used in the expressions to compensate for the attenuation of transmitted and returning light within the range developed in 8.1.

The flow diagram for a computer program to calculate the scattering ratio through the stratosphere based on the equations given above and those for the determination of temperature is given in figure 8.4.

8.4 Software Written to Process the Rhodes LIDAR Data

In order to process the counts obtained from the PMT after the laser is fired a number of programs were written for the IBM-PC. The function and operation of these is summarised below. Full listings are given in the appendices.

8.4.1 FIRST.BAS (Appendix 3.1)

This is the first of the data processing programs. It was written in HBASIC like the LIDAR operating program GIANT.BAS in order to make use of the same graphing routines. It loads a file of raw data and displays it graphically, scaled to fit the screen exactly. If there are no major imperfections due, for example, to mis-alignment of the apparatus, the program can compensate the data for near-saturation of the PMT as described in chapter 5.5 and graph it again. Finally the data can be saved to disk as a .BAS file ready for use by a program which will calculate temperature or scattering ratio profiles.

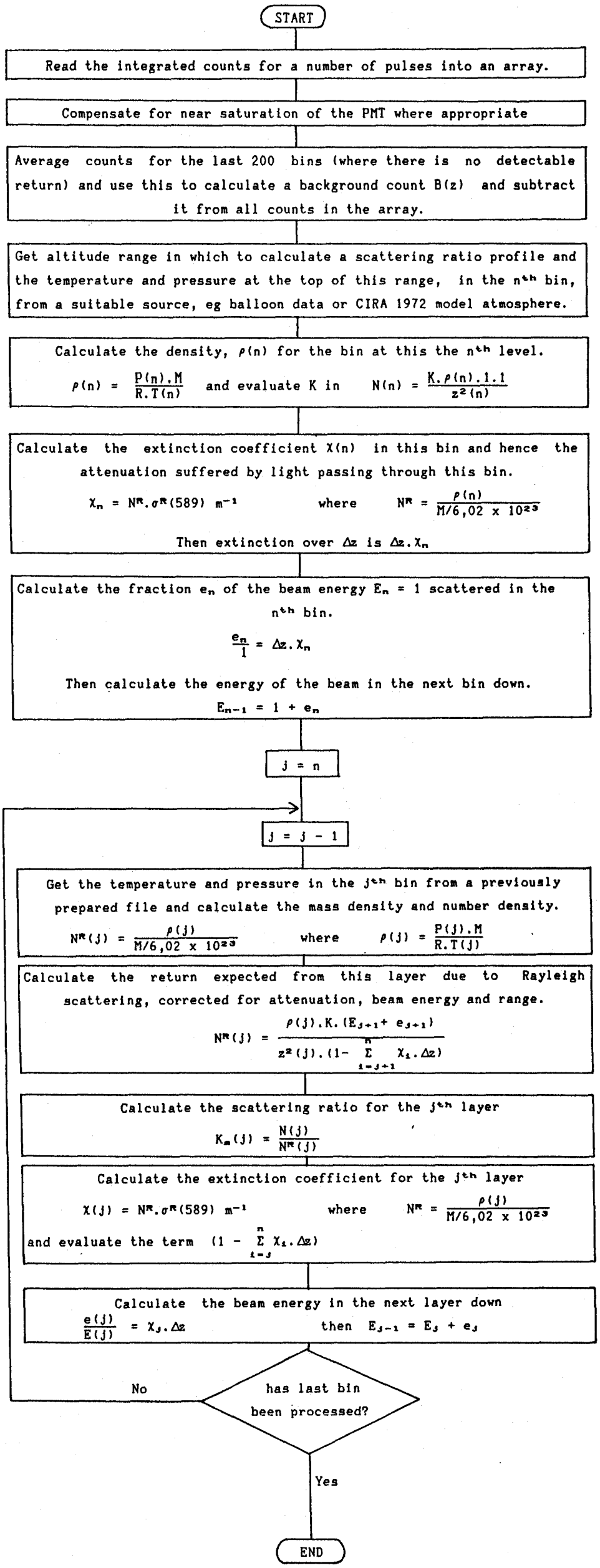


Figure 8.4 Flowchart for a computer program to calculate a scattering ratio profile of the atmosphere. Symbols are as defined in the text.

8.4.2 TEMPER.PAS (Appendix 4.1)

This program, written in Turbo Pascal, loads a file created by FIRST.BAS, calculates a background count by averaging the contents of 200 bins in which there is no detectable laser return and subtracts this from all the bins. The program then asks for the altitude range over which temperature is to be calculated as well as the temperature and pressure at the top of the range. Calculated altitudes and temperatures are then printed on the screen and there is a choice of whether to graph the results or save them to disk. The graph is drawn using routines from the Turbo Graphix Toolbox, a library of routines for Turbo Pascal by Borland. Once the graph is on the screen it can be printed on the FX-80 parallel printer using PROC F.ASM (appendix 2.6) or the results saved to disk as a file of pairs of reals containing altitude and temperature respectively. Prompts are issued throughout except after the graph is drawn when typing 'P' will print the graph, 'S' will save the results or 'X' will permit the processing of another file. 'Q' will quit the program and return to DOS.

8.4.3 SCATRAT.PAS (Appendix 5.1)

This Turbo Pascal program starts in the same way as TEMPER.PAS and subtracts a background count but then allows the creation of a file of temperatures and pressures at altitudes incrementing by 150 m drawn from balloon data and CIRA 1972 or the loading from disk of a previously created file. The program then asks for the range over which to calculate scattering ratio, does so and sends the results to the screen. One can then either repeat the calculations with a new starting count to ensure for example that the results are sensible and do not containing scattering ratios less than 1, graph the results or save them to disk. Again prompts are issued throughout except after the graph is drawn when typing 'P' will print the graph, 'S' will save the results or 'X' will permit the processing of another file. 'Q' will quit the program and return to DOS.

8.4.4 TEMPAVE.PAS (Appendix 4.2)

This program written in Turbo Pascal reads a number of files of altitude and temperature and stores the information in arrays which enable the mean temperature and its standard deviation to be calculated for each range bin. The files do not need to overlap exactly as the number of values of temperature available for each bin is also stored. After a number of files have been collated in this way the means can be graphed and error bars representing ± one standard deviation drawn at each point. The altitudes, means and standard deviations can also be saved to disk for archiving. Again prompts are issued throughout except after the graph is drawn when typing 'P' will print the graph, 'S' will save the results or 'X' will permit the processing of another file. 'Q' will quit the program and return to DOS.

8.4.5 SCATAVE.PAS (Appendix 5.2)

This Turbo Pascal program does the same for files of altitude and scattering ratio. as TEMPAVE.PAS does for altitude and temperature. Again prompts are issued throughout except after the graph is drawn when typing 'P' will print the graph, 'S' will save the results or 'X' will permit the processing of another file. 'Q' will quit the program and return to DOS.

CHAPTER 9

Results Obtained from the LIDAR Data

9.1 Compensation of the Raw Data for Non-linearity of the PMT

The first stage in the data reduction is to compensate the counts recorded per range bin for photomultiplier tube (PMT) non-linearity. The HBASIC program FIRST.BAS reads the file of counts per range bin from disk and draws a graph of counts per bin versus range on the screen, scaled to fill the screen exactly. If there are no obvious imperfections the program can then go on to calculate the average counts per shot and use the translation table described in chapter 5.5 to substitute, where appropriate, the count that would have been obtained if the response of the tube was linear up to a maximum of 22 counts per shot per bin. If the count would have exceeded this it is taken to be beyond redemption and set to zero. The data is also referenced to sea-level at this stage and since the LIDAR site in Grahamstown is at an altitude of 548 m the data is shifted 4 range bins up. The results are then graphed again and can be saved back to disk for future attention. The effect of this initial processing is demonstrated in figure 9.1 which shows raw and compensated data from 100 shots at an energy of 360 mJ per pulse. In the compensated data the return from 5 to 15 km was bright enough to produce a real count rate greater than 14 MHz and so the data has been set to zero.

9.2 Construction of a File of Altitude and Temperature

The temperature and altitude data obtained from the balloonsonde released by the Weather Bureau in Port Elizabeth during the night of the LIDAR observations is graphed. This usually goes as far as 26 km so to extend the range, the temperature up to an altitude of 40 km, which is the current upper limit of the LIDAR, is obtained from the 1972 COSPAR International Reference Atmosphere (CIRA 1972) for the corresponding month and latitude. A best line is drawn through the points and a file of temperature and altitude at 150 m intervals from 10 to 40 km is created on disk.

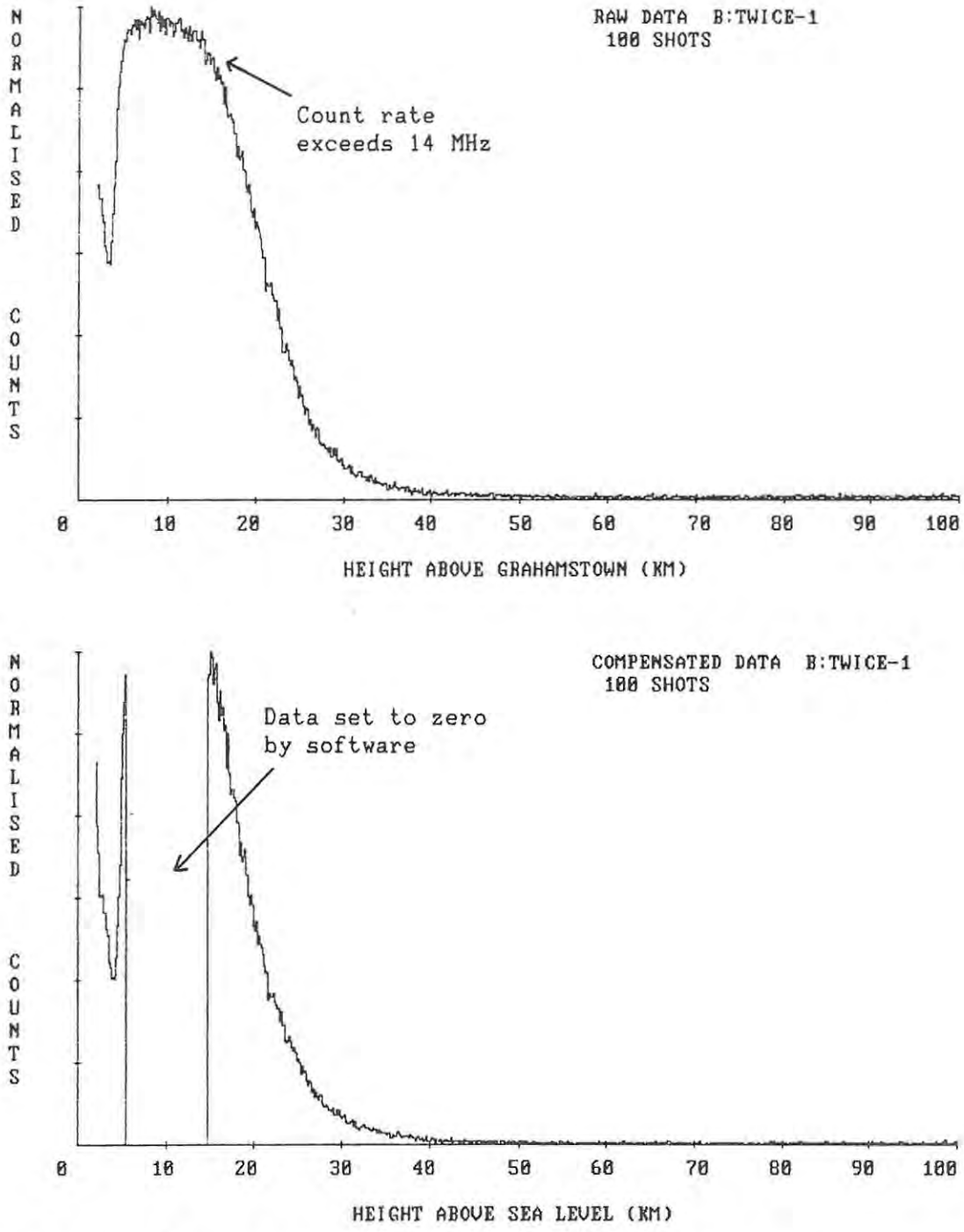


Figure 9.1 Top graph. Raw data from 100 shots at 360 mJ per pulse. Bottom graph. Same data compensated for PMT non-linearity.

9.3 Construction of a File of Altitude and Pressure

The altitude of the balloonsonde is not determined by an absolute method such as radar but from the pressure it experiences during its ascent. When it is released allowance is made for the surface pressure at the time and the Weather Bureau is confident the altitude given is correct to better than ± 10 m. An exponential regression calculation is

then applied to the Weather Bureau data from 10 to 25 km and to the CIRA 1972 data from 25 to 40 km. The coefficients so obtained predict the measured values of pressure to within 2% and generally better than 1%. The pressure at the altitudes of the range bins is then calculated using these coefficients rather than graphing the measured values, drawing a curve and reading off the pressure as is done for the temperature data.

The effectiveness of this method is shown in table 9.1.

Alt. — m	Meas. press. — mb	Source	Coefficients in Press. = A.exp(B.Alt.) A	B x 10^{-4}	Corr. coeff.	Calc. press. — mb	Diff. %
40000	3,08	CIRA	831,505	-1,40418	-0,99973	3,024	1,8
35000	5,99	CIRA	831,505	-1,40418	-0,99973	6,013	1,9
30000	12,1	CIRA	831,505	-1,40418	-0,99973	12,315	1,8
25000	25,3	CIRA	831,505	-1,40418	-0,99973	24,850	1,8
25000	25,3	CIRA	1386,651	-1,59881	-	25,47	0,7
23970	30,0	Balloon	1386,651	-1,59881	-0,99996	30,03	0,1
20746	50,0	Balloon	1386,651	-1,59881	-0,99996	50,29	0,58
18669	70,0	Balloon	1386,651	-1,59881	-0,99996	70,09	0,13
16479	100,0	Balloon	1386,651	-1,59881	-0,99996	99,48	0,52
13961	150,0	Balloon	1386,651	-1,59881	-0,99996	148,79	0,80
12158	200,0	Balloon	1386,651	-1,59881	-0,99996	198,50	0,75
10714	250,0	Balloon	1386,651	-1,59881	-0,99996	250,05	0,02
9497	300,0	Balloon	1386,651	-1,59881	-0,99996	303,76	1,25

Table 9.1 Comparison of Calculated and Measured Pressure and Altitude

9.4 Creation of a File of Altitude, Pressure and Temperature

The values of temperature and pressure at the altitudes of the range bins throughout the range over which LIDAR observations have been made are then saved to disk for future reference by the program which calculates scattering ratios.

9.5 Scattering Ratio Calculations

The Turbo-Pascal program SCATRAT is used to calculate scattering ratios. It first reads the BASIC file of compensated counts created by FIRST.BAS as described in 9.1 and strips off the header put there by BASIC. It then reads the file of temperature and pressure created as described in 9.2 and calculates scattering ratios using the algorithm shown in figure 8.4.

After the calculations have been performed the results are sent to the screen. The beginning and end of a typical calculation are shown in figure 9.2. The measured counts and calculated counts per range bin due to Rayleigh scattering track very closely across almost two orders of magnitude.

As shown in figure 8.2 the algorithm takes into account attenuation of the laser beam as well as the attenuation of returning photons within the range of observations. The energy of the beam at the bottom of the range compared to an energy of unity at the top as well as the extinction factor by which the predicted counts are respectively multiplied and divided are also shown. The maximum correction will occur at the bottom of the range and in the densest layers and in this case gives a correction of 3,8% or 75 counts in 1965 to that predicted from the lowest layer in the range. Elsewhere in the range and at greater altitudes it is proportionally less and can therefore be seen to provide a very small correction.

After the calculations of scattering ratio the results can be graphed and saved to disk ready for incorporation with other files created from different batches of shots. The graph corresponding to the results in figure 9.2 is shown in figure 9.3.

Bin	Height	Measured	Rayleigh	Scattering ratio
167	25061	26	30	0.86667
166	24911	40	31	1.29032
165	24761	40	32	1.25000
164	24611	26	34	0.76471
163	24461	40	35	1.14286
162	24311	48	36	1.33333
161	24161	33	38	0.86842
160	24011	33	39	0.84615
159	23861	33	40	0.82500
158	23711	40	42	0.95238
157	23561	40	44	0.90909
156	23411	40	46	0.86957
155	23261	55	47	1.17021
154	23111	48	49	0.97959
153	22961	55	51	1.07843
152	22811	55	53	1.03774
151	22661	55	55	1.00000
⋮	⋮	⋮	⋮	⋮
85	12761	892	871	1.02411
84	12611	914	910	1.00440
83	12461	955	955	1.00000
82	12311	986	998	0.98798
81	12161	1079	1049	1.02860
80	12011	1106	1097	1.00820
79	11861	1118	1142	0.97898
78	11711	1140	1195	0.95397
77	11561	1242	1252	0.99201
76	11411	1253	1311	0.95576
75	11261	1438	1367	1.05194
74	11111	1489	1433	1.03908
73	10961	1514	1496	1.01203
72	10811	1569	1563	1.00384
71	10661	1659	1633	1.01592
70	10511	1659	1715	0.96735
69	10361	1828	1794	1.01895
68	10211	1907	1877	1.01598
67	10061	1975	1965	1.00509

```
Bottom beam energy =   1.0188278432E+00
Extinction factor  =   9.8134478570E-01
(R)epeat with another starting count, (S)ave the results,
plot the (G)raph or (Q)uit ?
```

Figure 9.2 Scattering ratio calculations for data from 110 shots.

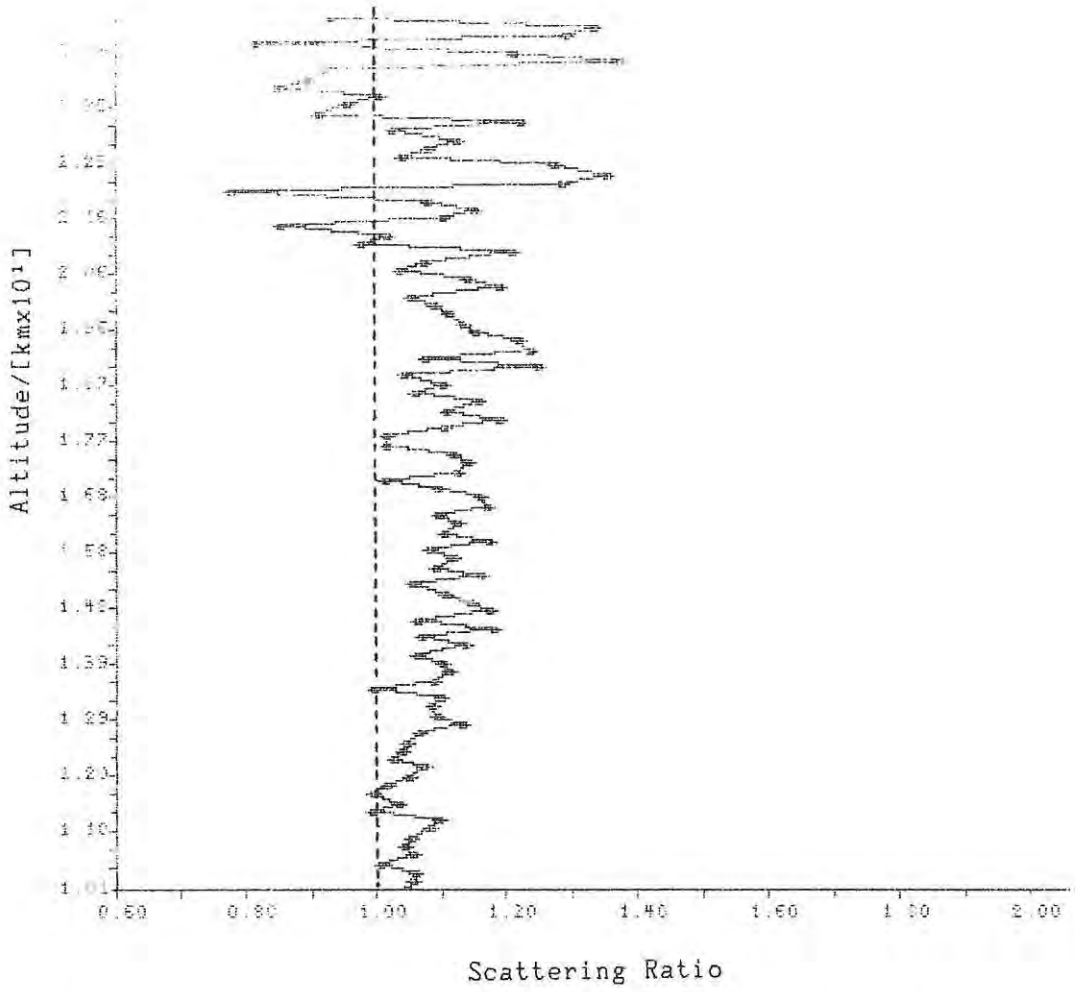


Figure 9.3 Scattering ratio vs. altitude using the results of figure 9.2. Data from 110 shots on 27 October 1987.

9.6 Averaging Several Files of Scattering Ratio

After typically 5 to 10 files of counts from typically 100 shots have been processed as described in 9.5, the results can be combined with the Pascal program SCATAVE. The values of scattering ratio for each bin are averaged and the standard deviation calculated. The average values can then be graphed and error bars of ± 1 standard deviation drawn for each point. The result of combining 10 files from data created on the night of 27 October 1987 is shown in figure 9.4.

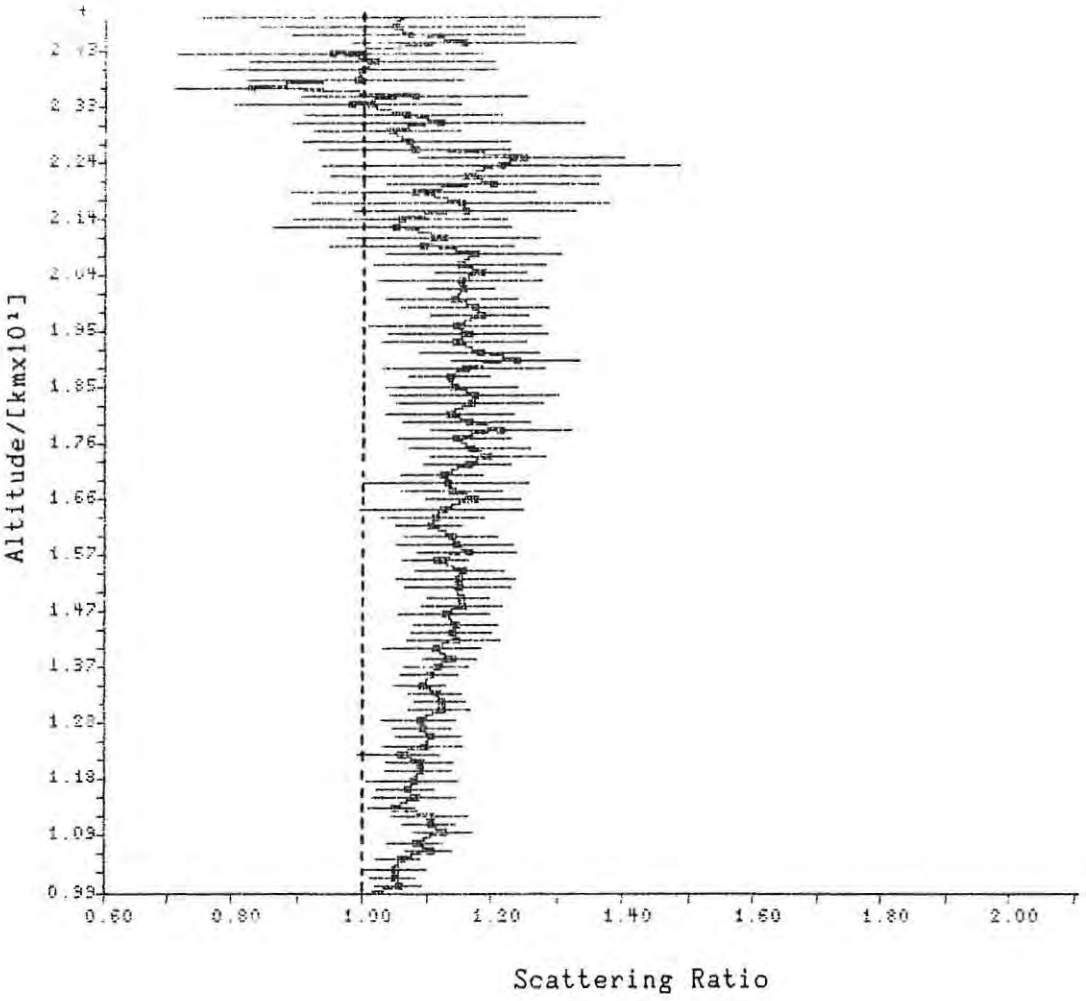


Figure 9.4 Scattering ratio vs. altitude. 10-25 km. 27 October 1987.
Combination of 10 files of data.
Error bars are ± 1 standard deviation.

The next clear night suitable for LIDAR observations was 20 November 1987. In the meantime the laser had been overhauled and beam quality improved. The result of combining 5 files of scattering ratio calculations, each from files of counts from 100 shots at 370 mJ per pulse is shown in figure 9.5.

Figure 9.5 turns out to have useful diagnostic properties because the kink at 25 km and fall-off in scattering ratio can be interpreted as follows. Below 25 km the laser beam was still entering the receiver beam and entry was only complete above this altitude where the scattering ratio remains close to unity. Below 25 km more scattered photons are expected than actually received and the scattering ratio evaluates to less than unity.

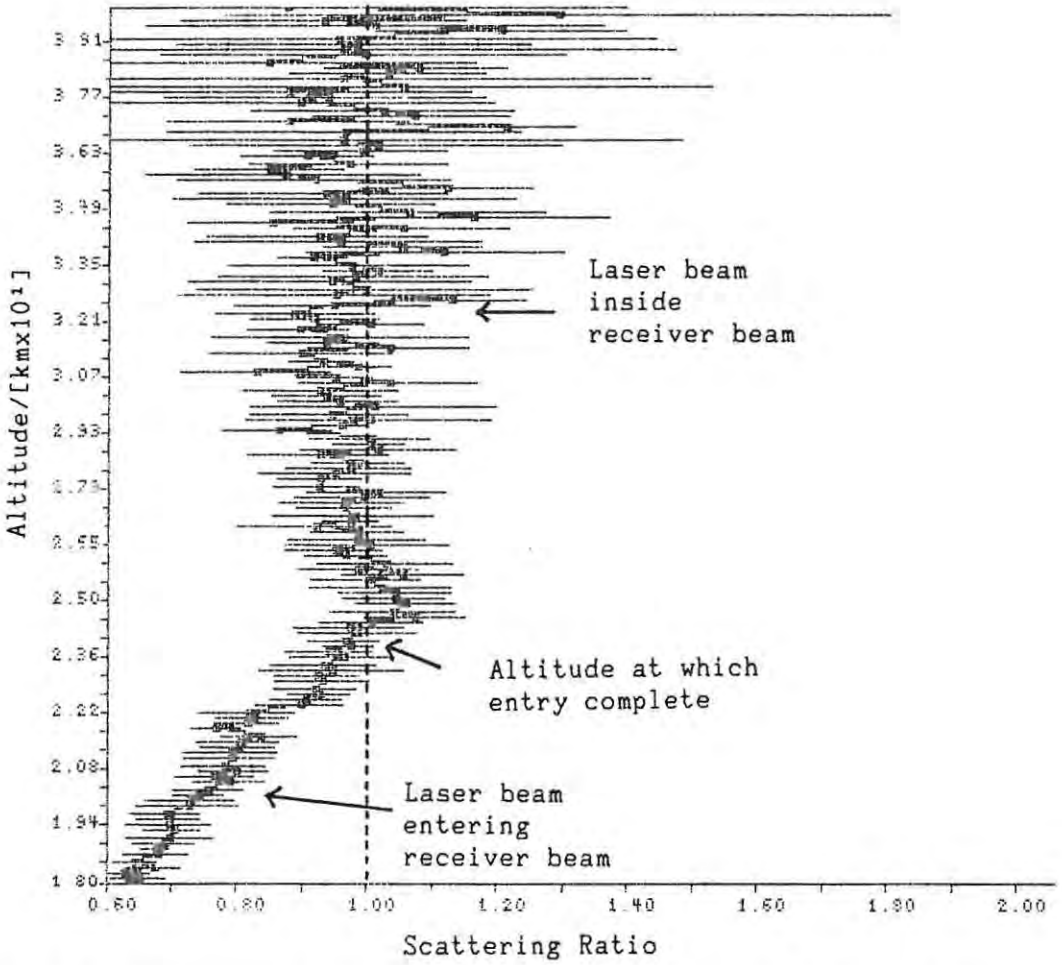


Figure 9.5 Scattering ratio vs. altitude. 18-40 km. 20 November 1987
Combination of 5 files of data.
Error bars are ± 1 standard deviation.
Evidence of transmitter-receiver mis-alignment.

On the same night, after realignment, a further 10 batches of shots at the slightly lower energy of 290 mJ per pulse produced results which when analysed for scattering ratio between 37 and 22 km gave the values shown in figure 9.6.

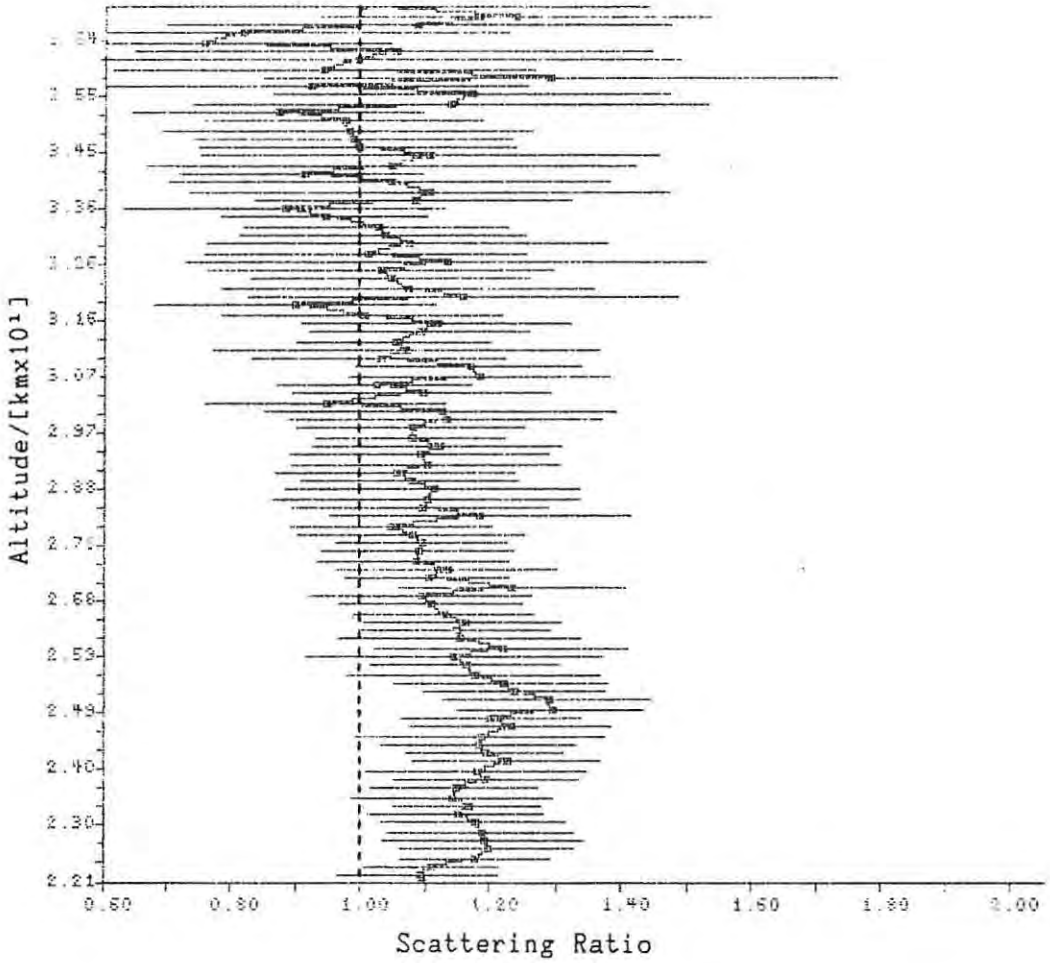


Figure 9.6 Scattering ratio vs. altitude. 22-37 km. 20 November 1987
Combination of 10 files of data.
Error bars are ± 1 standard deviation.

On the next clear night an attempt was made to investigate further the possible existence of an aerosol at an altitude of 25 km. However, as observations were beginning, the laser suffered a serious malfunction. Some plastic insulating material came between one of the spark gaps and was carbonised. This resulted in a dead short which caused the full output of the HT power supply to be applied continuously to a resistor through which the capacitors are normally charged. The resistor overheated, melting its plastic casing and burning off some of the oil in which it was immersed. The resulting cloud of acrid smoke prompted the use of the CO_2 fire extinguisher and the damage sustained prevented any further observations that night.

The resistor was repaired by replacing the melted plastic casing with garden hose filled with transformer oil and stoppered at the ends with silicone sealant. However the laser still did not work and it was found that the trigger transformer which initiates the spark gap discharge was open circuited. As no direct replacement was available a car ignition

coil was dismantled and the primary and secondary electrically separated as required by the driving circuit. Some capacitors were changed in the electronics in order to produce a satisfactory spark and after this repair the laser was brought back into operation. It was found that the triggering was now more reliable than it had been previously.

The next clear night was that of 17-18 December. Inspection of the raw data obtained showed a small but significant enhancement in the returned signal near 25 km while the laser was operating at both low and high power. Some of this data is shown in figure 9.7.

With the laser operating at the energy that produced the right-hand graph in figure 9.7, 3 sets of 5 batches of 100 shots were recorded. Scattering ratio profiles were computed for each batch and the 5 batches combined to yield graphs of scattering ratio between 20 km and 37 km with error bars representing ± 1 standard deviation on the value in each range bin for each batch. One of these profiles is shown in figure 9.8 and the layer near 25 km is now clearly visible.

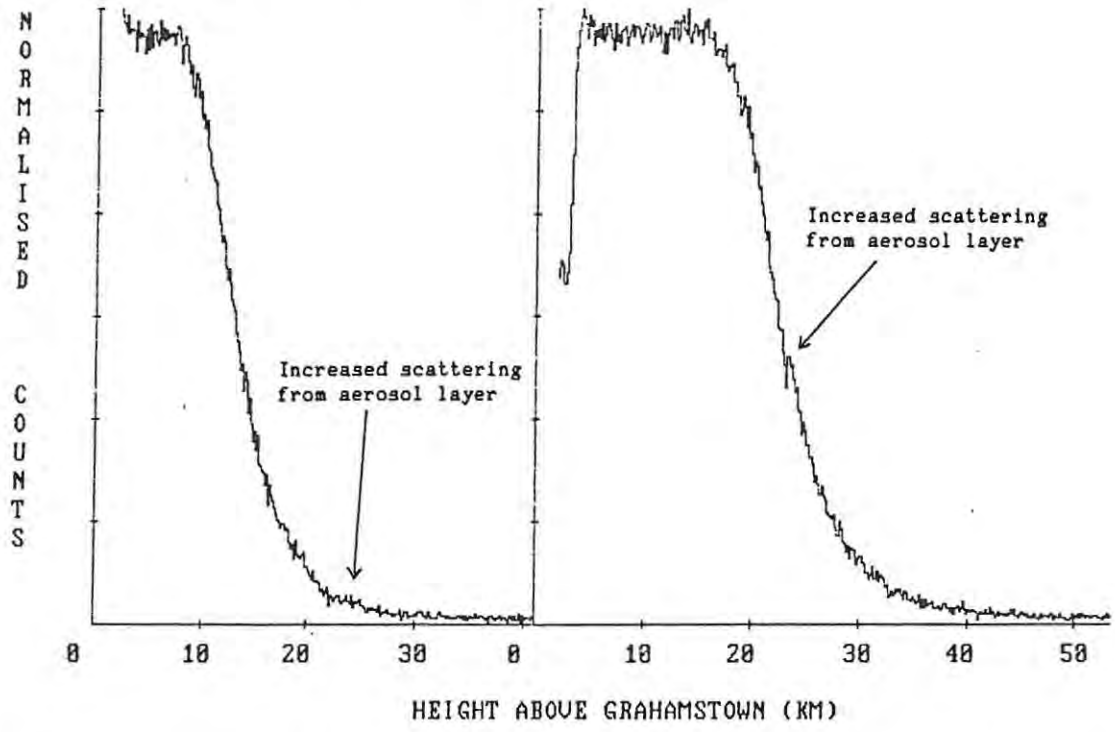


Figure 9.7 Raw data from 17-18 December showing enhanced scattering near an altitude of 25 km
Error bars are ± 1 standard deviation.

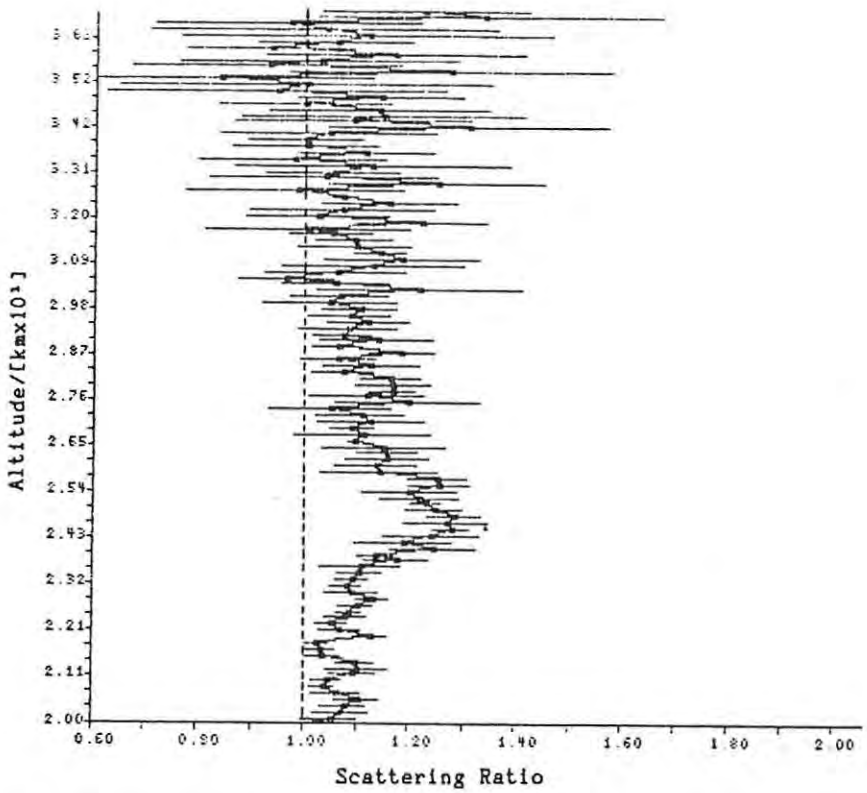


Figure 9.8 Scattering ratio vs. altitude. 20-37 km. 17-18 December showing layer near 25 km
Error bars are ± 1 standard deviation.

In order to examine the layer more closely the 3 sets were analysed for scattering ratio between 22 km and 27 km. The results are shown in figure 9.9, together with the time during which the observations were made.

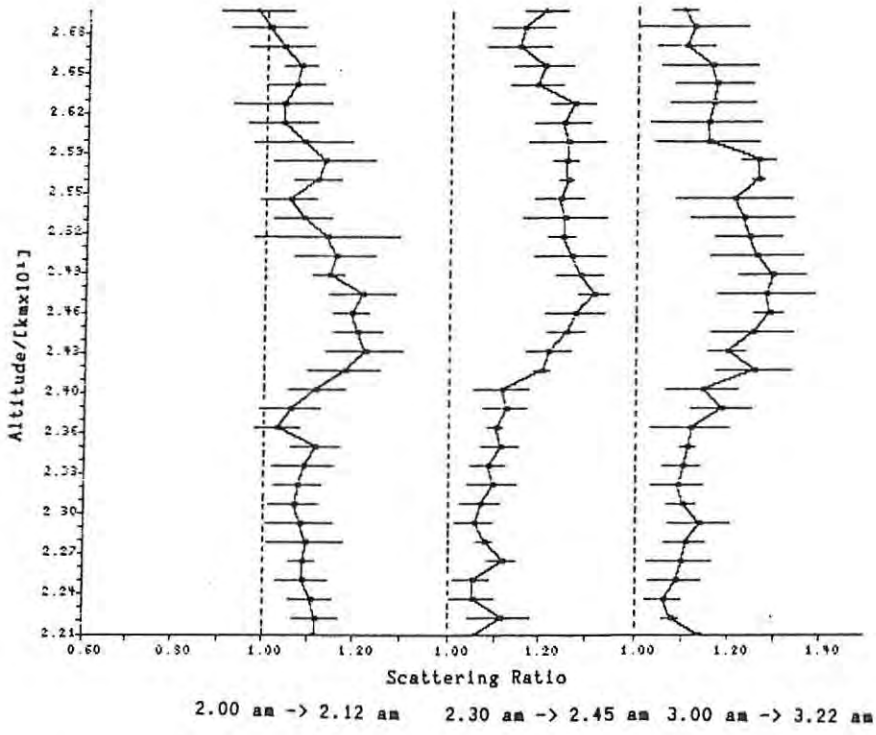


Figure 9.9 Scattering ratio between 22 km and 27 km
17-18 December 1987

9.7 Discussion of the Scattering Ratio Results

It has been shown (eg Hirono et al, 1984 and Shibata et al, 1984) that changes in the stratospheric scattering ratio follow large volcanic eruptions which inject quantities of gas and dust into the middle atmosphere. The aerosols so formed also display their presence by being responsible for spectacular sunsets for some time afterwards. (Meinel and Meinel, 1983, plates 6-4 to 6-6 etc.) The results of the El Chichón eruption on 4 April 1984 were visible in Grahamstown for at least a year but any twilight afterglow due to stratospheric aerosols has recently been noticeably absent. For this reason scattering ratios close to unity are to be expected from current LIDAR observations.

Scattering ratio measurements made by Shibata et al. (1984) at Kyushu University, Fukuoka, Japan are shown in figure 9.10. The enhancement due to the El Chichón eruption is clearly visible but the scattering ratio prior to the arrival of the cloud is very similar to the present results from the Rhodes LIDAR.

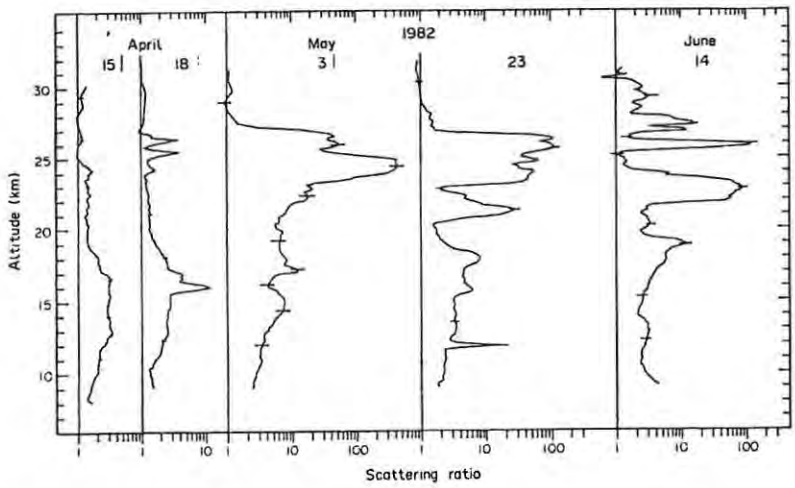


Figure 9.10 Scattering ratios following the El Chichón eruption. (Shibata et al. 1984)

There is a sulphur layer in the lower stratosphere even during volcanically quiet periods, composed mainly of droplets of an aqueous solution of sulphuric acid (Shibata et al. 1984). This layer may be responsible for the broad region of enhanced scattering ratio around 17 km shown in figure 9.4. The eruption of El Chichón caused a dramatic increase in scattering ratio near 25 km as shown in figure 9.10 and the feature seen at this altitude by the Rhodes LIDAR may be a remnant of this aerosol in the absence of any known contribution since that time.

The differences in the profile of this layer over the period of about 1 hour, shown in figure 9.9, may not be significant judged against the extent of the error bars but temporal changes in the layer would be a topic for further investigation.

9.8 Uncertainty of the Measured Scattering Ratio

The extent of the error bars in figures 9.4 and 9.6 indicates that the scattering ratio has been measured to within about 10% at the bottom of the range but is uncertain by up to 40% at the top. This uncertainty is greater than that introduced by such systematic sources of error as attenuation within the range (less than 4% maximum) and neglecting the variation in the acceleration due to gravity across the range (0,9% between 10 km and 40 km) or taking the speed of light to be 3,00 x 10^8 m/s. (about 0,2% depending on air density). The extent of the error bars is due to variations in the measured values of scattering ratio computed from several batches of counts, taken while neither the optics nor the voltage on the capacitors in the laser had been adjusted. They can therefore be attributed to statistical variations in the photon counting process. Foord et al. (1969) point out that it has been shown to great accuracy by E. Jakeman et al (1968) that the probability that n photoelectrons will be detected in a time T follows a Poisson distribution. The precision of the scattering ratio measurements could be increased if necessary by improving receiver efficiency to increase the number of counts recorded per bin or integrating over a greater number of shots with a consequent extension of observing time and reduction of laser flashlamp lifetime.

The accuracy of the measured scattering ratio over large height intervals is influenced by mis-alignment of the transmitter and receiver. The returned light from low altitudes is focussed by the mirror to a point outside the aperture which allows light onto the PMT. As the beams overlap, the focal point moves onto and then across the aperture. If the focal point was to skirt the edge of the aperture the signal could be mistaken for a broad band of enhanced scattering ratio. It is also assumed that the aperture is uniformly sensitive. If it is not, this could also apply a large scale bias to the data. The ability of the mirror to focus light to a region small compared with an aperture only a few millimeters in diameter is also assumed. There is also some out-of-focus light from the front surface of the back-silvered mirror arriving at the aperture. These problems can be

overcome by having the transmitter and receiver beams co-axial which requires a receiver of considerably greater sophistication than that currently in use.

Although the ability of the Rhodes LIDAR to see broad features is questionable and the broad region of slightly enhanced scattering ratio around 17 km shown in figure 9.4 may be produced in the receiver, it is difficult to see how narrow features could arise from imperfections in the equipment particularly when they appear at different laser energies and different alignments. The narrow band near 25 km shown in figure 9.8 is therefore almost certainly an atmospheric phenomenon.

Although present determinations of the scattering ratio by the Rhodes LIDAR are uncertain by up to 40% at the top of the range, this altitude is above the region where most middle atmosphere aerosols have been detected in the past and as previous volcanic eruptions have given rise to scattering ratios of several times unity and in some cases several orders of magnitude greater than unity as shown in figure 9.10, the Rhodes LIDAR is clearly capable of detecting and monitoring such enhancements when they next occur.

9.9 Temperature Calculations

If the scattering ratio is close to unity then the scattered light is proportional to atmospheric density and if hydrostatic equilibrium is assumed, the files of counts per range bin can be used to calculate atmospheric temperature as discussed in chapter 8.2 and using the algorithm shown in figure 8.3.

The same files of data used to create figure 9.4 were processed with the Pascal program TEMPER. This yielded profiles of temperature which were averaged using the program TEMPAVE and standard deviations calculated for the 10 values of temperature obtained for each range bin. These results were then plotted and appear in figure 9.11. As the temperature over the range is actually known from balloon data taken the same night it is also plotted.

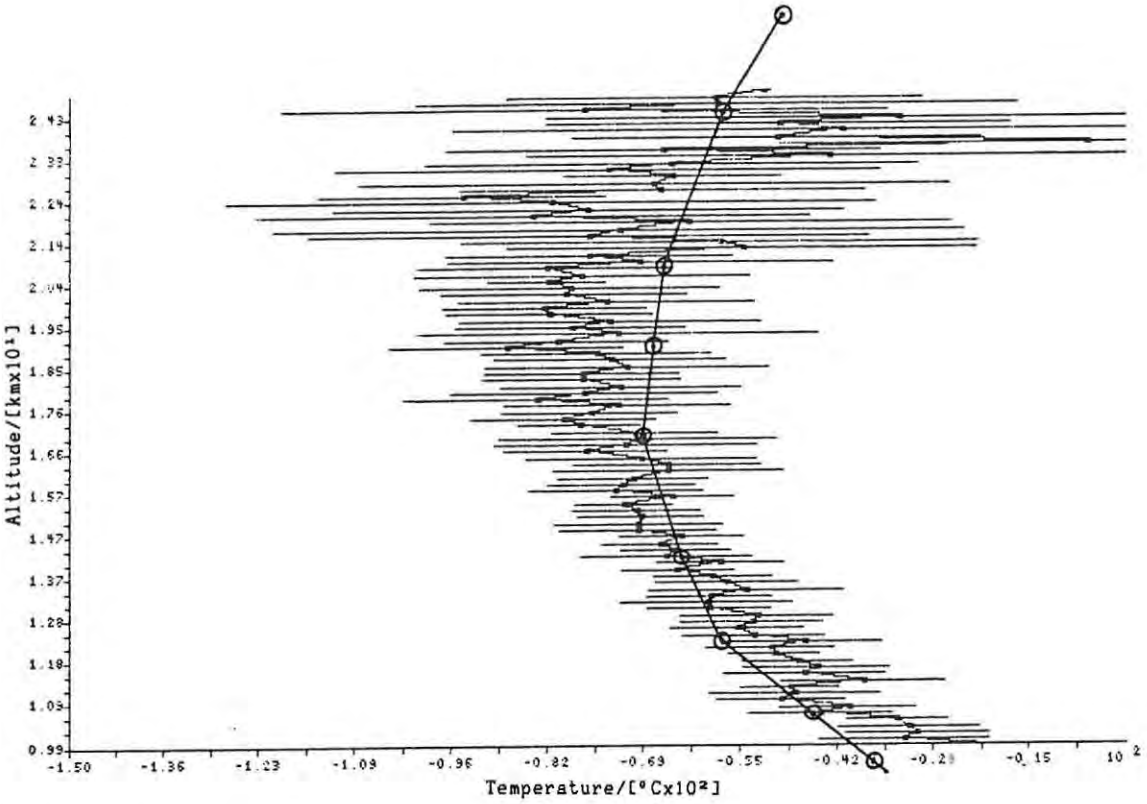


Figure 9.11 Temperature vs. altitude. 10-25 km. 27 October 1987
Combination of 10 files of data.
Error bars are ± 1 standard deviation.
——⊖—— Balloon data

The files used to create figure 9.6 were also processed for temperature information, the temperature at the top of the range being obtained from CIRA 1972. In this case each of the 10 files of temperature and altitude was smoothed with a 3-point running mean before being combined using the program TEMPAVE. The result is shown in figure 9.12. Again the CIRA 1972 and balloon temperatures are also plotted for comparison.

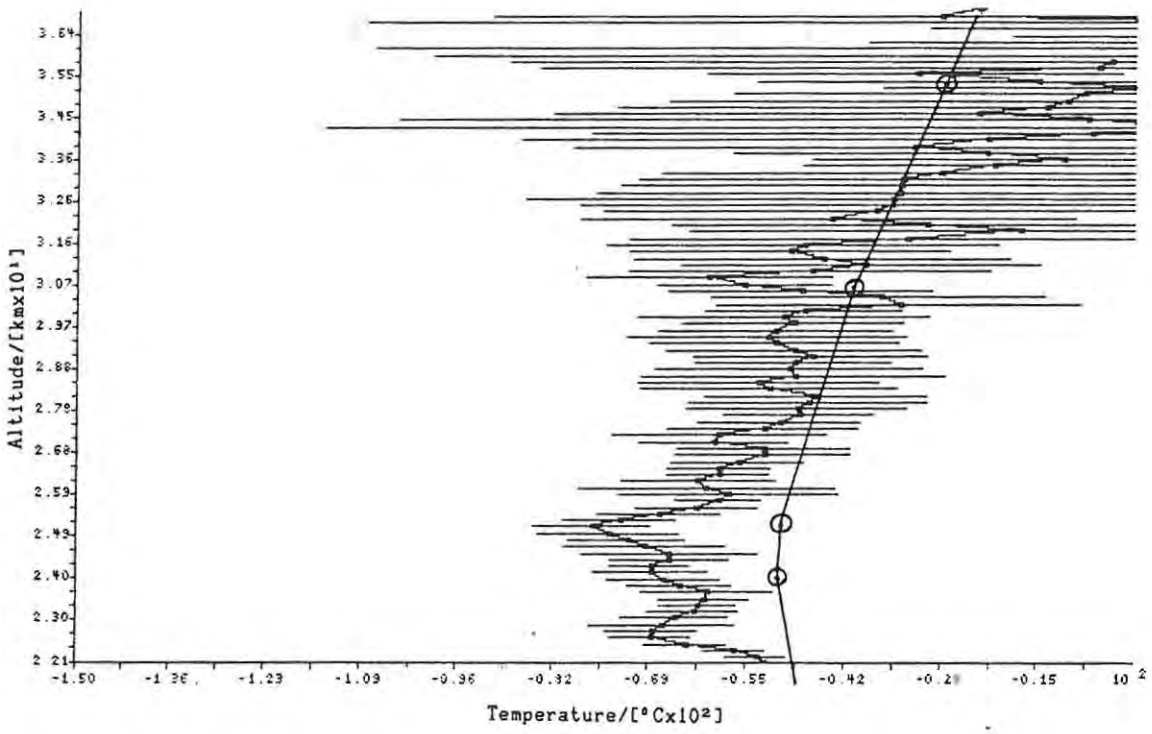


Figure 9.12 Temperature vs. altitude. 22-37 km. 20 November 1987
Combination of 10 files of data.
Error bars are ± 1 standard deviation.
———⊖——— Balloon and CIRA data.

9.10 Discussion of the Temperature Results

Figures 9.11 and 9.12 show that the Rhodes LIDAR is able to determine atmospheric temperature profiles up to an altitude of about 40 km at present. The uncertainties on the temperature at each point range from ± 12°C at 12 km in figure 9.11 to ± 100°C at 40 km near the top of figure 9.12. These large uncertainties can be attributed to the small size of the range bins in the Rhodes LIDAR compared to many other LIDARs which result in a comparatively small number of counts per bin and hence a large percentage variation between the counts in neighbouring bins and in the same bins in different batches of shots.

The symmetry of the error bars and the fact that the general trend of the graphs in figures 9.11 and 9.12 follows what the temperature is known to be suggests that greater precision could be obtained with an artificial increase in range bin size brought about by summing the counts in several adjacent bins. This would mean however that the vertical resolution of the system was reduced. This reduction would be acceptable if it is known that there are no small scale features to be

detected, unlike the case of the scattering ratio where thin layers of aerosols are of particular interest. Greater precision could also be obtained by increasing receiver efficiency and observing time and hence the number of counts recorded per bin.

LIDAR measurements of temperature are only valid if it is known that aerosols are absent. The presence of an aerosol layer causes an increase in the number of received photons and this is translated by the software to a layer of increased atmospheric density. To exist at the calculated pressure of the layer it has to be colder than the surroundings and hence the calculated temperature in an aerosol layer is too low if it is assumed aerosols are absent. Evidence of this phenomenon can be seen in figure 9.12 where a region of low temperature at 25 km coincides with a possible aerosol layer at this altitude shown in figure 9.6.

Because temperature calculations depend on data over broad ranges they are also particularly sensitive to tne performance of the receiver as discussed in 9.8.

Atmospheric temperatures have been measured by this method by Hauchecorne and Chanin (1980) at the Haute Provence Observatory in France. They used a LIDAR which was set up to measure mesospheric alkali metals and recorded backscattered light from 35 to 70 km. The height resolution of the LIDAR was 1,2 km, reduced to 4,8 km by a running average in order to decrease the standard deviation. After analysing the counts from 14600 laser pulses of 0,8 J at 670 nm they quote uncertainties of from 0,8°C at 35 km to 12°C at 66 km if any aerosol presence is neglected. The Rhodes LIDAR could probably repeat these observations if it was felt that the consequent wear and tear on the laser would be worthwhile.

When the data used to create figure 9.12 is smoothed with a 31 point running mean this produces a graph in which the curve passes through

the same points as would the line in a plot in which the range bins are 4.65 km long, comparable with those of Hauchecorne and Chanin. This plot is shown in figure 9.13. Further reduction in the extent of the error bars would be obtained by integrating data from more shots.

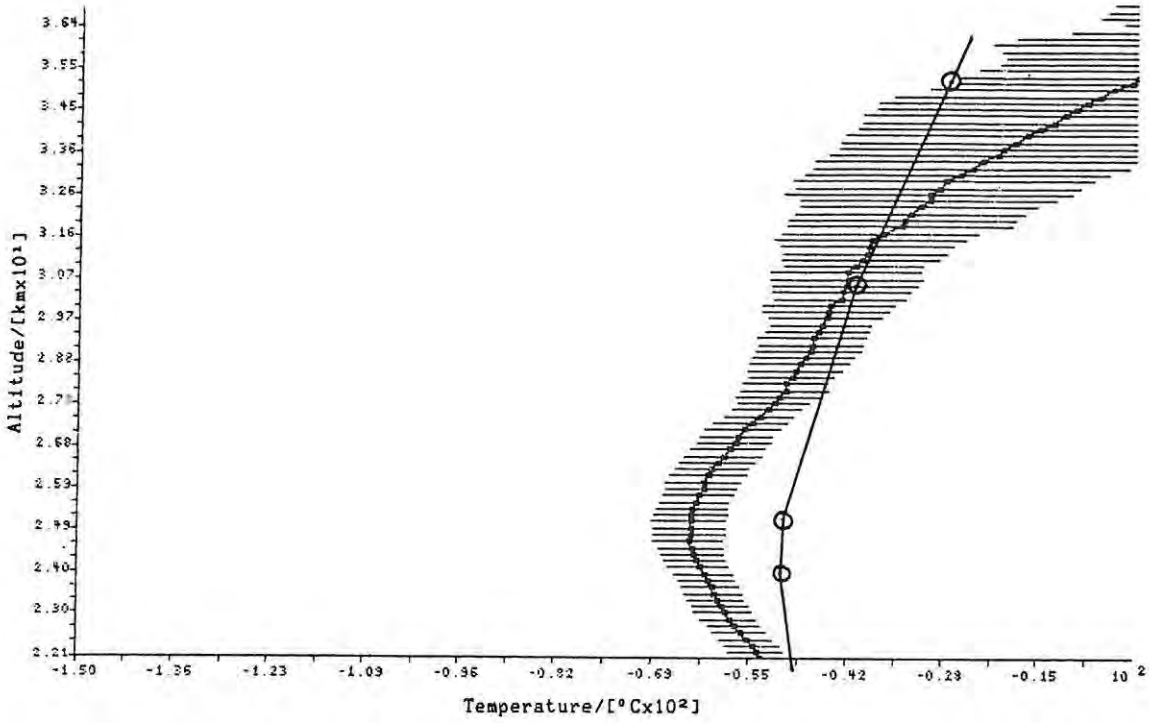


Figure 9.13 Data from figure 9.9 smoothed with a 31 point running mean. Error bars are ± 1 standard deviation.

——⊙—— Balloon and CIRA data.

CHAPTER 10

Conclusion and Suggestions for Future Work

10.1 Conclusion

A LIDAR using a tunable flashlamp pumped dye laser has been established at Rhodes University. The LIDAR is designed to investigate the middle atmosphere and as far as is known is the only one in Africa.

With present equipment the LIDAR is able to measure the atmospheric scattering ratio between 10 and 40 km. The LIDAR can also measure the temperature of the atmosphere within this range provided the scattering ratio is known to be close to unity.

10.2 Suggestions for further Work

The Rhodes LIDAR has been brought to the stage where it is a functional unit but certain aspects of it remain primitive in the extreme. It is now ready to be worked on further by a team of people who can bring their specialized knowledge to bear on various parts of it. There are aspects that could form projects of widely ranging complexity, from Honours level to PhD. The following are suggested as fields for development and improvement.

10.2.1 The Receiver

The receiver is separated from the transmitter to protect it from near-field scatter. Most LIDARs use a rotating shutter to protect the PMT. A rotating shutter would provide an interesting project for the Rhodes LIDAR and with one fitted the receiver and transmitter beams could be made co-axial. This would solve many of the alignment problems and the region of atmosphere illuminated by the laser pulse would also no longer track across the receiver's field of view.

The field of view of the collimating lens is 79,1 degrees and this does not make full use of the searchlight mirror surface. If a Cassegrain configuration was introduced the efficiency of the receiver would be significantly increased. The design and installation of a Cassegrain

reflector and a new set of collimating lenses would be another project.

The RCA 9558B is an old photomultiplier tube and greater count rates with a corresponding increase in dynamic range would be achieved with a state-of-the-art tube. Two have been purchased but they require faster pulse amplifiers possibly using GaAs technology. The design and building of a suitable amplifier would be another project.

10.2.2 The Laser

If the sodium layer between 90 and 100 km is to be studied then a method of tuning the laser to one of the D lines and narrowing the band width to a few picometers needs to be implemented. The laser also has to be stabilized at these settings for long periods in order to integrate enough pulses to build up a profile. Extra elements need to be obtained for the laser cavity and this project should only be tackled once the LIDAR has demonstrated its reliability and efficiency when observing features lower in the atmosphere.

The stratospheric ozone concentration is currently of worldwide concern. Ozone can be studied with the DIAL technique but this would require either another laser or a method of rapidly switching the wavelength of the existing one. Since ozone absorbs strongly in the ultra-violet, light at this wavelength can be produced by frequency doubling the visible output of a dye laser with a non-linear crystal. Again extra components will need to be obtained but ozone monitoring could form a future project.

10.2.3 Software and Computing Power

The software written so far is functional and accurate. However it could be tidied up and made more user-friendly. Several languages have been used, depending on which was most appropriate at the time. In particular the slow HBASIC routines would benefit from translation to Turbo Pascal now that a library of graphing routines is available for the Hercules card used for the display.

The computer presently being used is a first generation IBM-PC with 256 K of RAM. All the software and data are stored on floppy disk. The software has been kept in small pieces to minimise the impact of a disk crash. If a more up-to-date 8 MHz machine with a hard disk was obtained software development and data processing would be considerably speeded up. Figure 9.5 shows how receiver mis-alignment reveals itself even though it is not obvious in the raw data. At present it takes about 15 minutes to obtain a graph like figure 9.5 from a file of counts from a number of laser shots which considerably lengthens set-up time since the prism directing the laser beam then has to be adjusted and observations recommenced. This time would be very much shorter with all the necessary routines linked together on the hard disk of a faster machine.

10.2.4 A Home for the LIDAR System

The laser is currently housed in a building due to be taken over by the Ichthyology Department. This is its third home. Initially it was accomodated in two rooms on the third floor of the Geology Department. Leaning out of the window in order to adjust the prism directing the laser beam was not a popular task. Also the computer had to be placed close to the laser with the result that the electromagnetic pulse emitted by the laser was capable of corrupting the data and code in the machine, causing it to hang up. The next home was the lift room on the roof of the Zoology Department but the LIDAR had to move from there because the room contained machinery for operating lifts licenced to carry people. It will have to be moved from its present location at the end of December 1987 and at the moment no new site is available.

If a LIDAR capable of contributing meaningfully to worldwide remote monitoring of the atmosphere is to be established then a permanent air-conditioned and constant temperature site is required, preferably away from the effects of scattered city light.

The receiver stands in the open under a galvanized iron cover. Expensive components are subjected to considerable changes of temperature between daytime in the full sun and night under a clear sky. Also the receiver cannot be worked on during much of the day when

the sun is shining as removing the cover allows the mirror to focus sunlight sufficiently to ignite objects placed near the focus. Bending over the mirror in these circumstances is dangerous.

At present observations are not possible on a windy night because the curtains billow over the receiver and threaten to demolish the frame. If the receiver was housed in a brick building with a sliding roof these problems would be overcome and the receiver could be adjusted and aligned with ease.

Once a permanent working LIDAR is in operation, housed in a secure structure, consideration can be given to monitoring other atmospheric species such as sulphur dioxide and nitrogen oxides as well as the measurement of winds by detecting Doppler shifts of the scattered light.

Maintenance of a LIDAR at Rhodes University would provide a constant source of data for publication in local and international journals. When there is another large volcanic eruption which affects the stratosphere, contributions from Grahamstown will fill a void in global monitoring that used to extend from South America to Australia.

APPENDICES

APPENDIX 1. LIDAR Operating Skeleton Program

Appendix 1.1 GIANT.BAS

```
1 REM ************************************************************************
2 REM *            LIDAR OPERATING SKELETON PROGRAM                          *
3 REM *                        by    R. GRANT                                 *
4 REM ************************************************************************

10 SCREEN 2
15 CLS:KEY OFF:IP=0:PULSE=0:BATCH=1:DUMP=0:REJECTS=0
20 GOSUB 4000
25 LOCATE 2,15:PRINT "LIDAR CONTROLLER PROGRAM WRITTEN BY R.GRANT
   (1986)
26 LINE (14*9,2*14)-(64*9,2*14)
27 LOCATE 24,2:PRINT "LOADING MACHINE CODE ROUTINES.......";
31 GOSUB 6000:REM LOAD UP MACHINE CODE
32 LOCATE 24,38:PRINT "FINISHED";
35 LOCATE 5,2
40 A$=DATE$:PRINT "THE DATE IS ";A$;:INPUT " IS THIS CORRECT ? (Y/N)
   ",B$
45 IF B$="Y" THEN 100
50 LOCATE 6,2:INPUT "PLEASE TYPE THE CORRECT DATE AS mm-dd-yy ",D$
55 DATE$=D$
100 LOCATE 8,2
105 A$=TIME$:PRINT "THE TIME IS ";A$;:INPUT " IS THIS CORRECT ? (Y/N) "
    ,B$
110 IF B$ = "Y" THEN 130
120 LOCATE 10,2:PRINT "PLEASE TYPE THE CORRECT TIME AS hh:mm:ss AND
    PRESS RETURN WHEN IT IS CORRECT"
121 LOCATE 12,2:INPUT B$
125 TIME$=B$
129 CLS:GOSUB 4000
130 LOCATE 13,2:PRINT "DO YOU WISH TO"
135 LOCATE 15,18:PRINT "(1) RUN THE LIDAR SYSTEM"
140 LOCATE 17,18:INPUT "(2) TEST THE COUNTER HARDWARE ? ",B$
145 IF B$="1" THEN 1000
150 IF B$="2" THEN 200
151 GOTO 135
200 DEF SEG=&H1900
210 BLOAD "TESTMC",0
220 IP=0
230 CALL IP
240 GOTO 130
1000 CLS:GOSUB 4000
1005 PULSE=0:BATCH=1:REJECTS=0
1010 LOCATE 2,5:PRINT "DO YOU WISH TO :-"
1020 LOCATE 4,7:PRINT "(1)  RUN THE HARDWARE ONCE"
```

```
1030 LOCATE 6,7:PRINT "(2)  RUN THE HARDWARE CONTINUOUSLY"
1040 LOCATE 8,7:PRINT "(3)  RUN THE LIDAR SYSTEM. OR"
1050 LOCATE 10,7:PRINT "(4)  EXIT ?"
1060 LOCATE 12,5:INPUT "TYPE THE NUMBER OF YOUR CHOICE ",C$
1070 C=VAL(C$)
1080 ON C GOSUB 1200,1300,2000,1400
1090 GOTO 1000
1200 GOSUB 5000:REM DRAW AXES
1205 DEF SEG =&H2000:CALL IP:REM CONFIGURE PCS HARDWARE AND PREPARE
     FOR TRIG
1210 DEF SEG =&H2300:CALL IP:REM TRIGGER PCS ON, PROCD
1220 DEF SEG =&H2100:CALL IP:REM READ AND STORE RAM, PROCB
1230 DEF SEG =&H2400:CALL IP:REM PLOT THE POINTS, PROCE
1240 LOCATE 1,55:PRINT "(A)NOTHER OR (M)ENU      "
1241 Q$=INKEY$
1250 IF Q$="A" THEN 1210
1260 IF Q$="M" THEN 1000
1270 GOTO 1240
1300 GOSUB 5000
1310 LOCATE 1,47:PRINT "(S)TOP OR (P)RINT EXISTING SCREEN"
1315 DEF SEG =&H2000:CALL IP:REM CONFIGURE PCS HARDWARE AND PREPARE
     FOR TRIG
1320 DEF SEG =&H2300:CALL IP:REM TRIGGER PCS ON, PROCD
1330 DEF SEG =&H2100:CALL IP:REM READ AND STORE RAM, PROCB
1340 DEF SEG =&H2400:CALL IP:REM PLOT THE POINTS, PROCE
1350 Q$=INKEY$
1360 IF Q$="S" THEN 1380
1370 IF Q$="P" THEN 1390
1375 GOTO 1320
1380 Q$=INKEY$:IF Q$="" THEN 1380:REM PAUSE FOR ANOTHER KEY
1385 GOTO 1000
1390 DEF SEG=&H2500:CALL IP:REM DUMP SCREEN TO PRINTER
1395 GOTO 1000
1400 CLS:END:REM EXIT TO BASIC
2000 CLS
2010 GOSUB 4000
2020 LOCATE 2,5:INPUT "NUMBER OF PULSES TO BE INTEGRATED ";A$
2025 PULSETOT = VAL(A$)
2030 LOCATE 4,5:INPUT "SUBTRACT NOISE OR ALLOW COUNTS TO ACCUMULATE ?
     (S/A) ",B$
2040 IF B$<>"S" AND B$<>"A" THEN 2030
2050 LOCATE 6,5:INPUT "NUMBER OF DATA BATCHES TO BE TAKEN ";D$
2055 BATCHTOT = VAL(D$)
2060 LOCATE 8,5:INPUT "COMPUTER TRIGGER OR EXTERNAL TRIGGER OF THE
     LASER ? (C/E) ",C$
2063 IF C$<>"C" AND C$<>"E" THEN 2060
2065 IF C$="E" THEN 2080
2067 LOCATE 10,5:INPUT "TYPE THE DESIRED DELAY BETWEEN PULSES IN
     SECONDS ",DELAY
2068 DELAY = INT(340 * DELAY)
```

```
2070 IF C$<>"C" AND C$<>"E" THEN 2060
2080 LINE (9,15*14)-(74*9,15*14)
2090 LOCATE 15,2:INPUT "PLACE THE DISK FOR THE DATA IN DRIVE B THEN
     PRESS 'G' AND RETURN TO START ",E$
2100 IF E$<>"G" THEN 2090
2110 GOSUB 5000:REM DRAW AXES
2120 LOCATE 25,1:PRINT "(S)TOP";
2125 LOCATE 1,45:PRINT "PULSE 0   OF BATCH     REJECTS     ";
2126 LOCATE 1,64:PRINT BATCH:LOCATE 1,77:PRINT REJECTS
2130 DEF SEG =&H2000:CALL IP:REM PREPARE PCS HARDWARE
3000 IF C$="E" THEN 3050:REM WAIT FOR TRIGGERING OF PCS BY COMPUTER
3010 FOR I = 1 TO DELAY
3015 NEXT I:REM DELAY BETWEEN FIRINGS
3020 OUT 688,170:OUT 688,0:REM FIRE THE LASER
3050 DEF SEG =&H2200:CALL IP:REM WAIT FOR START OF PCS COUNTDOWN
3060 B2 = INP(701) AND 4
3070 IF B2 = 4 GOTO 3060:REM WAIT FOR B2 TO GO LO IE END OF PCS
     COUNTDOWN
3080 DEF SEG =&H2100:CALL IP:REM READ AND STORE PCS RAM, CHECK FOR
     REJECTS
3081 OLDMERIT = MERIT: MERIT = PEEK(&H5FFC)+ 256 * PEEK(&H5FFD):
     POKE &H5FFC,0:POKE &H5FFD,0
3082 LOCATE 2,50:PRINT "SUM ";MERIT;" DIFF ";MERIT-OLDMERIT;"    "
3083 IF PEEK(&H5FFF) = 0 THEN 3090:REM DATA OK, FLAG IS AT ABSOLUTE
     ADD 26FFF
3085 REJECTS=REJECTS+1
3087 LOCATE 1,77:PRINT REJECTS
3088 DEF SEG =&H2800:CALL IP:REM RECONFIGURE PCS HARDWARE
3089 GOTO 3150:REM DONT INC PULSE COUNT OR PLOT POINTS, JUST DO NEXT
     PULSE
3090 IF B$="A" THEN 3120:REM ACCUMULATE COUNTS
3100 DEF SEG =&H2300:CALL IP:REM TRIG PCS ON AND WAIT FOR END OF PCS
     COUNTDOWN
3110 DEF SEG =&H2700:CALL IP:REM READ AND SUBTRACT PCS RAM
3120 DEF SEG =&H2400:CALL IP:REM PLOT THE POINTS
3130 PULSE = PULSE + 1
3140 LOCATE 1,51:PRINT PULSE
3150 Q$=INKEY$
3160 IF Q$="S" THEN 3500:REM STOP AND GIVE OPTIONS
3170 IF PULSE < PULSETOT THEN 3000:REM DO THE NEXT PULSE
3180 GOSUB 7000
3190 BATCH = BATCH + 1
3200 LOCATE 1,64:PRINT BATCH
3210 IF BATCH <= BATCHTOT THEN 3450
3300 CLS:GOSUB 4000:T$ = TIME$
3400 LOCATE 10,5:PRINT BATCHTOT;" BATCHES OF ";PULSETOT;
     " PULSES COMPTETED";
3410 LOCATE 12,5:PRINT "AT ";T$;
3415 Q$ = INKEY$:IF Q$ = "" THEN 3415
3420 END
```

```
3450 CLS:PULSE=0:GOTO 2110
3500 LOCATE 25,1:PRINT "(C)ONTINUE  (M)ENU  (P)RINT SCREEN  OR
     (D)UMP TO DISK            ";
3510 Q$=INKEY$
3520 IF Q$ = "C" THEN 3570
3530 IF Q$ = "M" THEN 1000
3540 IF Q$ = "P" THEN GOTO 3600
3550 IF Q$ = "D" THEN GOTO 9000
3555 IF Q$ = "J" THEN GOSUB 10000 : REM JOIN DOTS
3556 IF Q$ = "R" THEN CLS:GOTO 2110: REM USEFUL FOR ALIGNING
3560 GOTO 3510
3570 GOSUB 8000
3575 DEF SEG =&H2800:CALL IP:REM RECONFIG PCS H/WARE
3580 GOTO 3170
3600 GOSUB 8000
3610 DEF SEG =&H2500:CALL IP:GOTO 3500:REM RESTORE OPTION LINE AND WAIT
4000 LINE (0,0)-(719,0):LINE -(719,347):LINE -(0,347):LINE -(0,0):
     RETURN
5000 CLS
5010 LINE (52,0)-(52,300)
5030 LINE -(719,300)
5035 FOR I= 0 TO 250 STEP 50:LINE (50,I)-(54,I):NEXT I
5040 PRINT"  300":PRINT:PRINT:PRINT"  250":PRINT"C":PRINT"O":PRINT"U"
     :PRINT"N 200"
5045 PRINT"T" :PRINT"S":PRINT  "  150":PRINT:PRINT:PRINT:PRINT
     "  100":PRINT:PRINT:PRINT:PRINT  "   50"
5050 A=119:B=66.7
5055 FOR I = 1 TO 10
5060 X = 52+ INT(I*B+.5)
5065 LINE (X,298)-(X,302)
5070 NEXT I
5075 PRINT:PRINT:PRINT:PRINT"    0       10     20      30     40
     50     60      70     80     90    100";
5078 LOCATE 25,1
5080   PRINT"                                          ALTITUDE /(Km)";
5200 RETURN
6000 REM DEF SEG =&H1900:BLOAD "BTSTGDL",0
6010 DEF SEG =&H2000:BLOAD "PROCA",0
6020 DEF SEG =&H2100:BLOAD "PROCB",0
6030 DEF SEG =&H2200:BLOAD "PROCC",0
6040 DEF SEG =&H2300:BLOAD "PROCD",0
6050 DEF SEG =&H2400:BLOAD "PROCE",0
6060 DEF SEG =&H2500:BLOAD "PROCF",0
6070 DEF SEG =&H2700:BLOAD "PROCG",0
6080 DEF SEG =&H2800:BLOAD "PROCH",0
6090 RETURN
7000 T$=STR$(-BATCH)
7005 N$="B:BATCH"+T$:REM THIS IS THE START OF THE DISK SAVE ROUTINE
7010 LOCATE 25,1:PRINT "SAVING BATCH ";BATCH;".......";R$;
7020 DEF SEG =&H2600:REM POINT CS AT STORAGE AREA
```

```
7025 POKE 2*667+1,INT(PULSE/256):POKE 2*667,PULSE-INT(PULSE/256)
7030 BSAVE N$,0,1402
7040 PRINT "DONE";
7050 RETURN
8000 LOCATE 25,1:R$=SPACE$(33):PRINT R$;"ALTITUDE /(Km)
     ";:RETURN
9000 DUMP=DUMP+1
9010 T$=STR$(-DUMP)
9020 U$=STR$(-PULSE)
9030 N$="B:DMP"+T$+U$
9040 LOCATE 25,1:PRINT "SAVING DUMP ";DUMP;".....     ";
9050 DEF SEG =&H2600:REM POINT AT STORAGE AREA
9055 POKE 2*667+1,INT(PULSE/256):POKE 2*667,PULSE-INT(PULSE/256)
9060 BSAVE N$,0,1402
9070 LOCATE 25,24:PRINT "DONE";
9080 GOTO 3500:REM RESTORE OPTIONS LINE AND WAIT
10000 DEF SEG =&H2600 :REM START OF DATA AREA
10010 X=51+4 :REM FIRST COLUMN ,MISSING FIRST 4 BINS WITH GARBAGE
10020 COUNT=256*PEEK(2*667+1-2*4)+PEEK(2*667-2*4):REM COUNT OF
      FIRST COLUMN
10030 YOLD=300-COUNT
10040 IF YOLD<0 THEN YOLD = 0:REM COUNTS > 300 ARE AT THE TOP OF THE
      SCREEN
10050 FOR I=2*667-2-2*4 TO 0 STEP -2
10060 COUNT=256*PEEK(I+1)+PEEK(I)
10070 Y=300-COUNT:IF Y < 0 THEN Y=0
10080 X=X+1:REM NEXT COLUMN
10090 LINE(X,YOLD)-(X,Y)
10100 YOLD=Y
10110 NEXT I
10120 RETURN
20000 STOP
```

APPENDIX 2. Machine Code Routines called by GIANT.BAS

Appendix 2.1

```
;*************************************************************************
;*                                                                       *
;*  THIS IS PROCA, A PROCEDURE TO CONFIGURE THE PCS FROM BASIC           *
;*                                                                       *
;*                              by                                       *
;*                                                                       *
;*                           R. GRANT                                    *
;*                                                                       *
;*************************************************************************

;EQUATES:-
CONTROL  EQU  02BFH
PORT_C   EQU  02BEH
PORT_B   EQU  02BDH
COUNT    =    667
STORAGE  =    06000H     ;STORAGE AREA FOR DATA FROM THE PCS RAM IS
                         ;1900:7000 OR 26000. THE CS REGISTER WILL BE
                         ;2000 FOR PROCA, AND 2000:6000 IS THE SAME
;
;ALL SEGMENT REGISTERS ARE EQUATED TO THE CS REGISTER WITH THE
;EXCEPTION OF THE SS AND SP REGISTERS WHICH ARE GIVEN THE VALUES
;SS=1800H, SP=0FFEH.
;
;DATA IS PLACED IN THE CODE SEGMENT AFTER THE CODE, HENCE ALL IS WELL
;AND THE CODE IS RELOCATABLE IF DS=CS.
;
CSEG SEGMENT PUBLIC 'CODE'
;
ASSUME CS:CSEG,DS:CSEG,ES:CSEG
;
SUBROUT PROC FAR     ;THE 'FAR' ENSURES THE RET AT THE END IS A FAR ONE
;
CONFIG: PUSH CS
        MOV AX,DS          ;HOLD VALUE OF DATA SEGMENT REGISTER ON ENTRY
        POP DS             ;INITIALISE DATA SEGMENT REGISTER
        MOV [OLD_DS],AX    ;SAVE VALUE OF DATA SEGMENT REGISTER ON ENTRY
        MOV [OLD_ES],ES    ;SAVE VALUE OF EXTRA SEGMENT REG. ON ENTRY
        MOV [OLD_SP],SP    ;SAVE VALUE OF STACK POINTER ON ENTRY
        MOV [OLD_SS],SS    ;SAVE VALUE OF STACK SEGMENT ON ENTRY
        MOV AX,DS          ;INITIALISE EXTRA SEGMENT REGISTER
        MOV ES,AX
        MOV AX,01800H      ;NEW VALUE FOR STACK SEGMENT
        MOV SS,AX
        MOV AX,0FFEH       ;NEW VALUE FOR STACK POINTER
        MOV SP,AX
;
```

```
;ANY EXTRA CODE GOES HERE
;
        MOV AL,10010010B
        MOV DX,CONTROL
        OUT DX,AL          ;A,B INPUT C OUTPUT MODE 0
        MOV DX,PORT_C
        MOV AL,11011111B
        OUT DX,AL
        MOV AL,11011110B   ;TRIGGER PCS ON
        OUT DX,AL
        MOV AL,11011111B
        OUT DX,AL
        MOV DX,PORT_B
SETTLE: IN AL,DX
        AND AL,00000100B   ;WAIT FOR PCS TO REACH ZERO AND GATE OFF
        JNZ SETTLE
        ;
        MOV AL,11001101B   ;C=0, R/W=R B=0 SO ADD COUNTER CAN COUNT
                           ;WE BAR HI
        MOV DX,PORT_C
        OUT DX,AL
        MOV AL,11011111B   ;PARALLEL LOAD ADD COUNTER WITH 0000
        OUT DX,AL          ;PULSE COUNTER BUFFERS TRI-STATE COUNT UP
;
        MOV CX,COUNT
WIND_UP: MOV AL,11010111B
        OUT DX,AL
        MOV AL,11011111B   ;GENERATE COUNTER ADDRESS
        OUT DX,AL
        LOOP WIND_UP
  ;
        CLD                ;COUNT UP WITH DI REGISTER
        MOV CX,COUNT       ;CLEAR DATA AREA SINCE DATA IS ADDED TO IT
        MOV DI,STORAGE
        MOV AX,0
        REP STOSW
  ;
        MOV AL,00111101B
        OUT DX,AL          ;C=B=0 R/W=W ENABLE PULSE COUNTER BUFFERS
        MOV AL,00111111B   ;COUNT DOWN
        OUT DX,AL
;
;
;NOW RESTORE THE SEGMENT REGISTERS
        MOV ES,OLD_ES
        MOV SS,OLD_SS
        MOV SP,OLD_SP
        MOV DS,OLD_DS      ;THIS ONE LAST IN ORDER TO BE ABLE TO GET
                           ;AT THE OTHERS
;
```

```
;ALL IS NOW RESTORED EXCEPT CS AND IP WHICH ARE RESTORED BY THE
;FAR RETURN
;
        RET                 ;FAR RETURN TO BASIC
;
SUBROUT ENDP
;
;PLACE ANY DATA HERE IN THE CODE SEGMENT. PLACING THE DATA AFTER THE
;CODE MEANS THE ENTRY POINT STAYS AT IP=0000
;
OLD_DS  DW  ?               ;THESE LOCATIONS ARE ESSENTIAL
OLD_ES  DW  ?               ;STORAGE FOR THE SEGMENT REGISTERS
OLD_SP  DW  ?               ;AND STACK POINTER
OLD_SS  DW  ?
;
CSEG    ENDS
END     CONFIG              ;SO THE LINKER GENERATES THE CORRECT ENTRY
                            ;ADDRESS
```

Appendix 2.2

```
;*************************************************************************
;*                                                                       *
;*THIS PROCEDURE, PROCB, READS THE PCS RAM AND STORES THE DATA IN THE    *
;*                        STORAGE AREA AT 26000.                         *
;*IT ALSO PREPARES A FIGURE OF MERIT BY ADDING THE COUNTS FROM 18 TO     *
;*      25 KM AND STORING THE SUM AT ABSOLUTE ADDRESS 26FFC              *
;*                                                                       *
;*                                by                                     *
;*                                                                       *
;*                              R. GRANT                                 *
;*                                                                       *
;*************************************************************************

;FOR THIS PROCB THE CS REG IS 2100 SO STORAGE IS 5000
;
;ALL SEGMENT REGISTERS ARE EQUATED TO THE CS REGISTER WITH THE
;EXCEPTION OF THE SS AND SP REGISTERS WHICH ARE GIVEN THE VALUES
;SS=1800H, SP=0FFEH.
;
;DATA IS PLACED IN THE CODE SEGMENT AFTER THE CODE, HENCE ALL IS WELL
;AND THE CODE IS RELOCATABLE IF DS=CS.
;
;EQUATES:-
;
PORT_A    EQU 02BCH
PORT_C    EQU 02BEH
COUNT     =   667
STORAGE   = 05000H        ;ACCUMULATED GOOD DATA
TEMPSTORE = 05800H        ;RAW DATA FROM PCS RAM
FLAG      = 05FFFH
TOPFIFTY  = 330
MOST      = 63
MERIT     = 05FFCH        ;OFFSET FOR FIG OF MERIT
STMERIT   = 120           ;uS AFTER FIRING TO START OF MERIT INTERVAL
ENDMERIT  = 167           ;TIME TO END OF INTERVAL
;
CSEG SEGMENT PUBLIC 'CODE'
;
ASSUME CS:CSEG,DS:CSEG,ES:CSEG
;
SUBROUT PROC FAR          ;THE 'FAR' ENSURES THE RET AT THE END
                          ;IS A FAR ONE
;
CONFIG: PUSH CS
        MOV AX,DS         ;HOLD VALUE OF DATA SEGMENT REGISTER ON ENTRY
        POP DS            ;INITIALISE DATA SEGMENT REGISTER
        MOV [OLD_DS],AX   ;SAVE VALUE OF DATA SEGMENT REGISTER ON ENTRY
```

```
        MOV [OLD_ES],ES    ;SAVE VALUE OF EXTRA SEG. REGISTER ON ENTRY
        MOV [OLD_SP],SP    ;SAVE VALUE OF STACK POINTER ON ENTRY
        MOV [OLD_SS],SS    ;SAVE VALUE OF STACK SEGMENT ON ENTRY
        MOV AX,DS          ;INITIALISE EXTRA SEGMENT REGISTER
        MOV ES,AX
        MOV AX,01800H      ;NEW VALUE FOR STACK SEGMENT
        MOV SS,AX
        MOV AX,0FFEH       ;NEW VALUE FOR STACK POINTER
        MOV SP,AX
;
;ANY EXTRA CODE GOES HERE
;
;
        MOV AL,11101101B   ;ENABLE BUFFER TO READ
        MOV DX,PORT_C      ;AND PARALLEL LOAD COUNTER WITH 0
        OUT DX,AL          ;DISABLE BOTH COUNTER BUFFERS
        MOV AL,11011111B   ;STATE OF C= R/W IS IRRELEVANT
        OUT DX,AL
  ;
        MOV CX,COUNT
        MOV BX,TEMPSTORE   ;POINT BX REG AT STORAGE FOR DATA
        MOV DX,PORT_A
READ:   IN AL,DX           ;GET BYTE FROM PCS RAM
        MOV [BX],AL        ;STORE RAW DATA
        INC BX             ;POINT BX AT NEXT LOCATION, BYTES REMEMBER
        MOV AL,11010111B
        MOV DX,PORT_C
        OUT DX,AL
        MOV AL,11011111B
        OUT DX,AL          ;INCREMENT ADDRESS, COUNT UP
        MOV DX,PORT_A
        LOOP READ
;
        MOV DX,PORT_C
        MOV AL,11010111B
        OUT DX,AL
        MOV AL,11011111B
        OUT DX,AL          ;BUMP COUNTER TO 668
        MOV AL,11010111B
        OUT DX,AL
        MOV AL,11011111B
        OUT DX,AL          ;BUMP COUNTER TO 669 SO DATA FROM FIRST RANGE
                           ;BIN 0-150M ENDS UP IN PCS RAM ADDRESS 667
;
;NOW CHECK THAT THE RAW DATA IS OK
;
CHECK:  MOV CX,TOPFIFTY    ;CHECK HIGHEST 330 COUNTS IE TOP 50 KM OF AIR
        MOV BX,TEMPSTORE+4 ;POINT BX AT START OF RAW DATA IGNORE FIRST
                           ;4 BYTES THE FIRST OF WHICH COLLECTS NOISE
        MOV AL,MOST        ;NO MORE THAN "MOST" COUNTS ALLOWED PER USEC
```

```
TEST:   CMP AL,[BX]
        JS GARBAGE          ;IF COUNTS MORE THAN MAX ALLOWED THEN QUIT
        INC BX              ;POINT AT NEXT LOCATION
        LOOP TEST
        XOR AH,AH
        MOV BX,FLAG
        MOV [BX],AH         ;DATA IS OK SO CLEAR FLAG
;
;NOW CALCULATE A FIGURE OF MERIT BY ADDING THE COUNTS FROM STMERIT TO
;ENDMERIT uS AFTER THE LASER FIRES
;
        MOV CX,ENDMERIT-STMERIT
        MOV SI,COUNT-STMERIT ;POINT SI REQUIRED NO. OF BYTES INTO
                             ;STORAGE
        ADD SI,TEMPSTORE     ;OFFSET SI INTO TEMPSTORE AREA
        XOR AX,AX
        MOV DI,MERIT         ;ES:DI NOW POINTS AT 26FFC ABSOLUTE
ADDUP:  MOV AL,[SI]          ;GET THE COUNTS AS BYTES
        ADD [DI],AX
        DEC SI               ;SI STARTS AT 547 AND MUST FINISH AT 500
        LOOP ADDUP
;
;
;NOW COMBINE THE RAW DATA WITH THE OLD
;
        MOV SI,TEMPSTORE
        MOV DI,STORAGE
        MOV CX,COUNT
COMBINE: MOV AL,[SI]        ;GET RAW DATA AS BYTES
        ADD [DI],AX         ;ADD TO OLD DATA AS WORDS
        INC SI              ;POINT AT NEXT RAW DATA BYTE
        INC DI
        INC DI              ;POINT AT NEXT GOOD DATA WORD IN STORAGE
        LOOP COMBINE
        JMP QUIT            ;FINISHED
;
;DEAL WITH GARBAGE
;
GARBAGE: MOV AL,0AAH
        MOV BX,FLAG
        MOV [BX],AL         ;DATA WAS GARBAGE SO SET FLAG TO TELL BASIC
;
;NOW RESTORE THE SEGMENT REGISTERS
QUIT:   MOV ES,OLD_ES
        MOV SS,OLD_SS
        MOV SP,OLD_SP
        MOV DS,OLD_DS       ;THIS ONE LAST IN ORDER TO BE ABLE TO GET
                            ;AT THE OTHERS
;
;ALL IS NOW RESTORED EXCEPT CS AND IP WHICH ARE RESTORED BY
```

```
;THE FAR RETURN
;
        RET                 ;FAR RETURN TO BASIC
;
SUBROUT ENDP
;
;PLACE ANY DATA HERE IN THE CODE SEGMENT. PLACING THE DATA AFTER THE
;CODE MEANS THE ENTRY POINT STAYS AT IP=0000
;
OLD_DS  DW  ?               ;THESE LOCATIONS ARE ESSENTIAL
OLD_ES  DW  ?               ;STORAGE FOR THE SEGMENT REGISTERS
OLD_SP  DW  ?               ;AND STACK POINTER
OLD_SS  DW  ?
;
CSEG    ENDS
END     CONFIG              ;SO THE LINKER GENERATES THE CORRECT
                            ;ENTRY ADDRESS
```

Appendix 2.3

```
;*************************************************************************
;*                                                                       *
;*      THIS IS PROC C. IT WAITS FOR THE LASER TO FIRE AND THEN          *
;*              WAITS FOR THE END OF THE PCS COUNTDOWN                   *
;*                                                                       *
;*                              by                                       *
;*                                                                       *
;*                           R. GRANT                                    *
;*                                                                       *
;*************************************************************************

;ALL SEGMENT REGISTERS ARE EQUATED TO THE CS REGISTER WITH THE
;EXCEPTION OF THE SS AND SP REGISTERS WHICH ARE GIVEN THE VALUES
;SS=1800H, SP=0FFEH.
;
;DATA IS PLACED IN THE CODE SEGMENT AFTER THE CODE, HENCE ALL IS WELL
;AND THE CODE IS RELOCATABLE IF DS=CS.
;
;EQUATES:-
;
PORT_B  EQU  02BDH
PORT_C  EQU  02BEH
;
CSEG SEGMENT PUBLIC 'CODE'
;
ASSUME CS:CSEG,DS:CSEG,ES:CSEG
;
SUBROUT PROC FAR          ;THE 'FAR' ENSURES THE RET AT THE END IS A
                          ;FAR ONE
;
CONFIG: PUSH CS
        MOV AX,DS         ;HOLD VALUE OF DATA SEGMENT REGISTER ON ENTRY
        POP DS            ;INITIALISE DATA SEGMENT REGISTER
        MOV [OLD_DS],AX   ;SAVE VALUE OF DATA SEGMENT REGISTER ON ENTRY
        MOV [OLD_ES],ES   ;SAVE VALUE OF EXTRA SEG. REGISTER ON ENTRY
        MOV [OLD_SP],SP   ;SAVE VALUE OF STACK POINTER ON ENTRY
        MOV [OLD_SS],SS   ;SAVE VALUE OF STACK SEGMENT ON ENTRY
        MOV AX,DS         ;INITIALISE EXTRA SEGMENT REGISTER
        MOV ES,AX
        MOV AX,01800H     ;NEW VALUE FOR STACK SEGMENT
        MOV SS,AX
        MOV AX,0FFEH      ;NEW VALUE FOR STACK POINTER
        MOV SP,AX
;
;ANY EXTRA CODE GOES HERE:-
;
        MOV DX,PORT_C
```

```
        MOV AL,00111111B  ;COUNTERS TO COUNT DOWN
        OUT DX,AL         ;WHEN LASER TRIGGERS PCS
        MOV DX,PORT_B
LOOP1:  IN AL,DX
        AND AL,00000100B  ;WAIT FOR B2 TO GO HI
        JZ LOOP1
;
;NOW RESTORE THE SEGMENT REGISTERS
        MOV ES,OLD_ES
        MOV SS,OLD_SS
        MOV SP,OLD_SP
        MOV DS,OLD_DS     ;THIS ONE LAST IN ORDER TO BE ABLE TO GET
                          ;AT THE OTHERS
;
;ALL IS NOW RESTORED EXCEPT CS AND IP WHICH ARE RESTORED BY THE
;FAR RETURN
;
        RET               ;FAR RETURN TO BASIC
;
SUBROUT ENDP
;
;PLACE ANY DATA HERE IN THE CODE SEGMENT. PLACING THE DATA AFTER THE
;CODE MEANS THE ENTRY POINT STAYS AT IP=0000
;
OLD_DS  DW  ?             ;THESE LOCATIONS ARE ESSENTIAL
OLD_ES  DW  ?             ;STORAGE FOR THE SEGMENT REGISTERS
OLD_SP  DW  ?             ;AND STACK POINTER
OLD_SS  DW  ?
;
CSEG    ENDS
END     CONFIG            ;SO THE LINKER GENERATES THE CORRECT
                          ;ENTRY ADDRESS
```

Appendix 2.4

```
;*************************************************************************
;*                                                                       *
;*         THIS IS PROC D. IT TRIGGERS THE PCS ON AND THEN               *
;*              WAITS FOR THE END OF THE PCS COUNTDOWN.                  *
;*                                                                       *
;*                                by                                     *
;*                                                                       *
;*                             R. GRANT                                  *
;*                                                                       *
;*************************************************************************

;ALL SEGMENT REGISTERS ARE EQUATED TO THE CS REGISTER WITH THE
;EXCEPTION OF THE SS AND SP REGISTERS WHICH ARE GIVEN THE VALUES
;SS=1800H, SP=0FFEH.
;
;DATA IS PLACED IN THE CODE SEGMENT AFTER THE CODE, HENCE ALL IS WELL
;AND THE CODE IS RELOCATABLE IF DS=CS.
;
;EQUATES:-
;
PORT_B  EQU 02BDH
PORT_C  EQU 02BEH
;
CSEG SEGMENT PUBLIC 'CODE'
;
ASSUME CS:CSEG,DS:CSEG,ES:CSEG
;
SUBROUT PROC FAR          ;THE 'FAR' ENSURES THE RET AT THE END IS A
                          ;FAR ONE
;
CONFIG: PUSH CS
        MOV AX,DS         ;HOLD VALUE OF DATA SEGMENT REGISTER ON ENTRY
        POP DS            ;INITIALISE DATA SEGMENT REGISTER
        MOV [OLD_DS],AX   ;SAVE VALUE OF DATA SEGMENT REGISTER ON ENTRY
        MOV [OLD_ES],ES   ;SAVE VALUE OF EXTRA SEG. REGISTER ON ENTRY
        MOV [OLD_SP],SP   ;SAVE VALUE OF STACK POINTER ON ENTRY
        MOV [OLD_SS],SS   ;SAVE VALUE OF STACK SEGMENT ON ENTRY
        MOV AX,DS         ;INITIALISE EXTRA SEGMENT REGISTER
        MOV ES,AX
        MOV AX,01800H     ;NEW VALUE FOR STACK SEGMENT
        MOV SS,AX
        MOV AX,0FFEH      ;NEW VALUE FOR STACK POINTER
        MOV SP,AX
;
;ANY EXTRA CODE GOES HERE:-
;
        MOV DX,PORT_C
```

```
        MOV AL,00111110B
        OUT DX,AL
        MOV AL,00111111B  ;GATE CLOCK ON
        OUT DX,AL
  ;
        MOV DX,PORT_B
POLL:   IN AL,DX
        AND AL,00000100B  ;WAIT FOR B2 TO GO LO IE FOR PCS TO FINISH
        JNZ POLL          ;ITS COUNTDOWN
  ;
;NOW RESTORE THE SEGMENT REGISTERS
        MOV ES,OLD_ES
        MOV SS,OLD_SS
        MOV SP,OLD_SP
        MOV DS,OLD_DS     ;THIS ONE LAST IN ORDER TO BE ABLE TO GET
                          ;AT THE OTHERS
;
;ALL IS NOW RESTORED EXCEPT CS AND IP WHICH ARE RESTORED BY THE
;FAR RETURN
;
        RET               ;FAR RETURN TO BASIC
;
SUBROUT ENDP
;
;PLACE ANY DATA HERE IN THE CODE SEGMENT. PLACING THE DATA AFTER THE
;CODE MEANS THE ENTRY POINT STAYS AT IP=0000
;
OLD_DS  DW  ?             ;THESE LOCATIONS ARE ESSENTIAL
OLD_ES  DW  ?             ;STORAGE FOR THE SEGMENT REGISTERS
OLD_SP  DW  ?             ;AND STACK POINTER
OLD_SS  DW  ?
;
CSEG    ENDS
END     CONFIG            ;SO THE LINKER GENERATES THE CORRECT ENTRY
                          ;ADDRESS
```

Appendix 2.5

```
;*************************************************************************
;*                                                                       *
;*THIS PROCEDURE, PROC E, PLOTS THE POINTS STORED IN STORAGE AT 26000H*
;*       IN THE COMPUTER RAM ON THE HI RES SCREEN AT B0000.              *
;*                                                                       *
;*                                by                                     *
;*                                                                       *
;*                             R. GRANT                                  *
;*                                                                       *
;*************************************************************************
;
;ALL SEG. REGISTERS ARE EQUATED TO THE CS REGISTER WITH THE EXCEPTION
;OF THE SS AND SP REGISTERS WHICH ARE GIVEN THE VALUES
;SS=1800H, SP=0FFEH.
;
;DATA IS PLACED IN THE CODE SEGMENT AFTER THE CODE, HENCE ALL IS WELL
;AND CODE IS RELOCATABLE IF DS=CS.
;
;EQUATES:-
;
STORAGE   = 2000H          ;CS WILL BE 2400H FOR THIS PROCEDURE,
                           ;ABSOLUTE ADDRESS OF STORAGE IS 26000H
COUNT     = 667
SCREEN  EQU 0B000H
;
CSEG SEGMENT PUBLIC 'CODE'
;
ASSUME CS:CSEG,DS:CSEG,ES:CSEG
;
SUBROUT PROC FAR           ;THE 'FAR' ENSURES THE RET AT THE END
                           ;IS A FAR ONE
;
CONFIG: PUSH CS
        MOV AX,DS          ;HOLD VALUE OF DATA SEGMENT REGISTER ON ENTRY
        POP DS             ;INITIALISE DATA SEGMENT REGISTER
        MOV [OLD_DS],AX    ;SAVE VALUE OF DATA SEGMENT REGISTER ON ENTRY
        MOV [OLD_ES],ES    ;SAVE VALUE OF EXTRA SEG. REGISTER ON ENTRY
        MOV [OLD_SP],SP    ;SAVE VALUE OF STACK POINTER ON ENTRY
        MOV [OLD_SS],SS    ;SAVE VALUE OF STACK SEGMENT ON ENTRY
        MOV AX,DS          ;INITIALISE EXTRA SEGMENT REGISTER
        MOV ES,AX
        MOV AX,01800H      ;NEW VALUE FOR STACK SEGMENT
        MOV SS,AX
        MOV AX,0FFEH       ;NEW VALUE FOR STACK POINTER
        MOV SP,AX
;
;ANY EXTRA CODE GOES HERE,
```

```
;
;THIS ROUTINE PLOTS THE POINTS READ FROM THE PCS RAM AND STORED IN
;"STORAGE" ON THE AXES DRAWN ON THE HI RES SCREEN BY HBASIC
;INITIAL X VALUE IS 52 ON THE X=0 MARK
;
       PUSH ES
       MOV AX,0B000H
       MOV ES,AX           ;POINT ES AT THE SCREEN
       MOV CX,COUNT        ;NUMBER OF POINTS TO BE PLOTTED
       SUB CX,3            ;DO THIS TO SKIP THE LOWEST ALTITUDE
                           ;COUNTS (GARBAGE)
       MOV SI,STORAGE      ;POINT SI AT START OF DATA
       ADD SI,COUNT        ;POINT SI AT LAST COUNT STORED IE FIRST
                           ;RESULT TAKEN
       ADD SI,COUNT        ;THEY ARE WORDS REMEMBER!
       SUB SI,8            ;POINT SI AT LAST LOCATION WRITTEN TO LESS
                           ;TWO
                           ;AND SKIP THREE LOWEST ALTITUDE COUNTS
       MOV AX,55           ;X CO-ORD OF FIRST RESULT TO BE PLOTTED
                           ;(X=52 IS X AXIS)
       MOV BX,300
       SUB BX,[SI]         ;GENERATE FIRST Y VALUE IN BX
                           ;(300-ACTUAL VALUE)
                           ;PLOT THE POINT IF [SI] LESS THAN 300
       JNS DRAW            ;OTHERWISE LET THE POINT BE AT THE TOP OF
                           ;THE SCREEN
       XOR BX,BX           ;SO SET BX = 0
  ;
DRAW:  CALL DOT_PLOT       ;PLOT THE POINT AX HAS X VALUE,
                           ;BX HAS Y VALUE
       DEC SI
       DEC SI              ;POINT SI AT NEXT COUNT
       MOV BX,300
       SUB BX,[SI]        ;GENERATE NEXT DATA
       JNS OKAY
       XOR BX,BX           ;AGAIN LET POINT BE AT THE TOP OF THE SCREEN
OKAY:  INC AX              ;GENERATE NEXT X VALUE
       LOOP DRAW
       ;
       POP ES
;
;NOW RESTORE THE SEGMENT REGISTERS
        MOV ES,OLD_ES
        MOV SS,OLD_SS
        MOV SP,OLD_SP
        MOV DS,OLD_DS      ;THIS ONE LAST IN ORDER TO BE ABLE TO GET
                           ;AT THE OTHERS
;
;ALL IS NOW RESTORED EXCEPT CS AND IP WHICH ARE RESTORED BY
;THE FAR RETURN
```

```
;
        RET                 ;FAR RETURN TO BASIC
;
SUBROUT ENDP
;
;PROCEDURE TO PLOT A DOT ON THE HIGH RES SCREEN
;ES MUST POINT AT THE SCREEN TO BE PLOTTED ON,
;IE ES MUST CONTAIN EITHER 0B000H OR 0B800H
;
; AX CONTAINS X VALUE BETWEEN 0 AND 719
; BX CONTAINS Y VALUE BETWEEN 0 AND 347
;
DOT_PLOT PROC NEAR
;
          PUSH DX
          PUSH CX
          PUSH BX
          PUSH AX
          PUSH AX           ;SAVE X VALUE ON THE STACK
          PUSH BX           ;SAVE Y VALUE ON THE STACK
          AND BX,0003       ;CALCULATE (Y MOD 4)
          MOV CX,BX         ;PLACE INTO CL WITH 00 IN CH
          INC CL            ;MAKE SURE CL > 0
          MOV BX,-2000H     ;IF (YMOD4)=0 THEN BX MUST BE 0
START1:   ADD BX,2000H
          LOOP START1       ;CALCULATE 2000H * (Y MOD 4) AND PLACE IN BX
          POP AX            ;PUT Y COUNT INTO AX
          SHR AX,1
          SHR AX,1          ;PUT INTEGER Y/4 INTO AX
          MUL NINETY        ;CALC 90*INTEGER(Y/4), RESULT IN AX
          MOV DI,AX         ;PLACE RESULT IN DI
          POP AX            ;PUT X VALUE IN AX
          MOV CL,AL         ;PLACE 8 LOWEST BITS IN CL
          AND CL,00000111B ;CALC XMOD8 IN CL
          SHR AX,1
          SHR AX,1          ;CALCULATE INTEGER X/8 IN AX
          SHR AX,1
          ADD BX,AX         ;ADD NEXT COMPONENT OF ADDRESS TO BX
          MOV AL,10000000B
          SHR AL,CL         ;PUT A 1 IN (7-XMOD8) BIT
          OR  ES:[BX+DI],AL ;TURN THE BIT ON
          POP AX
          POP BX            ;RESTORE CO-ORDS OF THE DOT PLOTTED
          POP CX
          POP DX            ;RESTORE INNOCENT REGISTERS
          RET
DOT_PLOT ENDP
;
;PLACE ANY DATA HERE IN THE CODE SEGMENT. PLACING THE DATA AFTER
;THE CODE ;MEANS THE ENTRY POINT STAYS AT IP=0000
```

```
;
OLD_DS  DW  ?           ;THESE LOCATIONS ARE ESSENTIAL
OLD_ES  DW  ?           ;STORAGE FOR THE SEGMENT REGISTERS
OLD_SP  DW  ?           ;AND STACK POINTER
OLD_SS  DW  ?
NINETY  DB  90
VALUE   DW  ?
;
CSEG    ENDS
END     CONFIG          ;SO THE LINKER GENERATES THE CORRECT
                        ;ENTRY ADDRESS
```

Appendix 2.6

```
;*************************************************************************
;*                                                                       *
;*THIS PROCEDURE, PROC F, DUMPS THE HI RES SCREEN ON THE IBM WITH THE    *
;*MONOCHROME GRAPHICS CARD TO THE FX-80 PRINTER VIA THE PARALLEL PORT    *
;*                        ON THE VIDEO CARD                              *
;*                                                                       *
;*                              by                                       *
;*                                                                       *
;*                           R. GRANT                                    *
;*                                                                       *
;*************************************************************************

;ALL SEG. REGISTERS ARE EQUATED TO THE CS REGISTER WITH THE EXCEPTION
;OF THE SS AND SP REGISTERS WHICH ARE GIVEN THE VALUES
;SS=1800H, SP=0FFEH.
;
;DATA IS PLACED IN THE CODE SEGMENT AFTER THE CODE, HENCE ALL IS WELL
;AND THE CODE IS RELOCATABLE IF DS=CS.
;
;EQUATES
;
SCREEN      EQU 0B000H
PRINSTATUS  EQU 03BDH
PRINDATA    EQU 03BCH
PRINCNTRL   EQU 03BEH
CR          EQU 0DH
LF          EQU 0AH
;
CSEG SEGMENT PUBLIC 'CODE'
;
ASSUME CS:CSEG,DS:CSEG,ES:CSEG
;
SUBROUT PROC FAR          ;THE 'FAR' ENSURES THE RET AT THE END IS A
FAR ONE
;
CONFIG: PUSH CS
        MOV AX,DS         ;HOLD VALUE OF DATA SEGMENT REGISTER ON ENTRY
        POP DS            ;INITIALISE DATA SEGMENT REGISTER
        MOV [OLD_DS],AX   ;SAVE VALUE OF DATA SEGMENT REGISTER ON ENTRY
        MOV [OLD_ES],ES   ;SAVE VALUE OF EXTRA SEG. REGISTER ON ENTRY
        MOV [OLD_SP],SP   ;SAVE VALUE OF STACK POINTER ON ENTRY
        MOV [OLD_SS],SS   ;SAVE VALUE OF STACK SEGMENT ON ENTRY
        MOV AX,DS         ;INITIALISE EXTRA SEGMENT REGISTER
        MOV ES,AX
        MOV AX,01800H     ;NEW VALUE FOR STACK SEGMENT
        MOV SS,AX
        MOV AX,0FFEH      ;NEW VALUE FOR STACK POINTER
```

```
        MOV SP,AX
;
;ANY EXTRA CODE GOES HERE,
;
;PROGRAM TO DUMP THE HI RES SCREEN FROM B0000 TO B7FFF
;TO THE FX-80 PRINTER ON THE PARALLEL PORT OF THE GRAPHICS CARD
;
            MOV DX,PRINCNTRL
            MOV AL,00000100B
            OUT DX,AL      ;INITIALISE HI
            MOV AL,00000000B
            OUT DX,AL      ;INITIALISE LO
            MOV CX,50
LOOP5:      LOOP LOOP5     ;STRETCH INIT LO GREATER THAN 50 US
            MOV AL,00000100B
            OUT DX,AL      ;INITIALISE HI AGAIN
            LEA BX,TABLE2
LOOP6:      MOV AH,[BX]
            CMP AH,04      ;IS IT EOT?
            JE LABEL1
            CALL SENDBYTE ;INIT PRINTER TO 8 DOTS PER LINE
            INC BX
            JMP LOOP6
LABEL1:     CALL PRTSCREEN    ;AND PRINT IT
;
;NOW RESTORE THE SEGMENT REGISTERS
        MOV ES,OLD_ES
        MOV SS,OLD_SS
        MOV SP,OLD_SP
        MOV DS,OLD_DS      ;THIS ONE LAST IN ORDER TO BE ABLE TO GET
                           ;AT THE OTHERS
;
;ALL IS NOW RESTORED EXCEPT CS AND IP WHICH ARE RESTORED BY THE
;FAR RETURN
;
        RET                ;FAR RETURN TO BASIC
;
SUBROUT ENDP
;
INIT_PTR PROC NEAR         ;PROCEDURE TO INITIALISE THE PRINTER TO 90
                           ;DOTS
            PUSH AX        ;PER INCH AND 720 COLUMNS OF DOTS
            PUSH BX
            LEA BX,TABLE1
NXTINIT:    MOV AH,[BX]    ;GET BYTE
            CMP AH,04      ;IS IT EOT ?
            JE OVER        ;STOP IF IT IS
            CALL SENDBYTE ;SEND TO PRINTER
            INC BX         ;POINT AT NEXT BYTE
            JMP NXTINIT
```

```
OVER:       POP BX
            POP AX
            RET
INIT_PTR    ENDP
;
;
SENDBYTE PROC NEAR          ;PROCEDURE TO SEND A BYTE TO THE PRINTER
            PUSH DX         ;BYTE MUST BE IN AH
LOOP1:      MOV DX,PRINSTATUS
            IN AL,DX        ;GET PRINTER STATUS
            AND AL,10000000B ;BIT 7 IS FX-80 BUSY SIGNAL
            JZ LOOP1        ;PRINTER STILL ACCEPTING DATA. BUSY SIGNAL
                            ;INVERTED
            MOV DX,PRINDATA
            MOV AL,AH       ;PUT DATA IN AL
            OUT DX,AL       ;SEND DATA
            MOV DX,PRINCNTRL
            MOV AL,00000101B
            OUT DX,AL       ;STROBE LO. (INTERFACE INVERTS BIT 0)
            MOV AL,00000100B
            OUT DX,AL       ;STROBE HI
            POP DX
            RET
SENDBYTE    ENDP
;
;
PRTSCREEN PROC NEAR         ;PROCEDURE TO PRINT THE SCREEN AS 44 ROWS
            PUSH AX
            PUSH CX
            PUSH ES
            MOV AX,SCREEN
            MOV ES,AX           ;POINT ES REGISTER AT SCREEN
            MOV AX,0
            MOV YCO_ORD,AX      ;ZERO THE Y CO_ORD
            MOV CX,44           ;DO 44 ROWS
LOOP4:      CALL PRTROW         ;PRINT A ROW
            MOV AX,YCO_ORD
            ADD AX,8
            MOV YCO_ORD,AX      ;UPDATE Y CO_ORD FOR NEXT ROW
            LOOP LOOP4
            POP ES
            POP CX
            POP AX
            RET
PRTSCREEN   ENDP
;
;
PRTROW PROC NEAR                ;PROCEDURE TO PRINT A ROW OF
                                ;720 DOTS 8 THICK
            PUSH CX             ;FROM XCO_ORD=0 TO XCO_ORD=719
```

```
            PUSH AX
            CALL INIT_PTR     ;INITIALISE FOR 720 DOTS OF GRAPHICS
            MOV AX,0
            MOV XCO_ORD,AX    ;SET X CO_ORD TO 0
            MOV CX,720        ;DO IT 720 TIMES
LOOP3:      CALL ASEMBYTE     ;ASSEMBLE BYTE INTO AH
            CALL SENDBYTE     ;AND SEND TO PRINTER
            INC XCO_ORD
            LOOP LOOP3
            CALL CRLF         ;PRINTER TO START OF NEXT ROW
            POP AX
            POP CX
            RET
PRTROW      ENDP
;
;
ASEMBYTE PROC NEAR          ;PROCEDURE TO ASSEMBLE A BYTE IN AH FROM THE
            PUSH DX         ;SCREEN POINTS XCO_ORD AND YCO_ORD TO
            PUSH CX         ;YCO_ORD + 7
            MOV AX,XCO_ORD
            MOV BX,YCO_ORD
            MOV CX,8        ;8 BITS IN THE BYTE
LOOP2:      CALL GET_DOT    ;GET STATUS OF DOT AT AX,BX INTO CARRY
            RCL DL,1        ;PLACE IN BIT 0 OF DL, PUSH OTHERS UP
            INC BX          ;NEXT BIT DOWN
            LOOP LOOP2
            MOV AH,DL       ;PUT ASSEMBLED BYTE IN AH
            POP CX
            POP DX
            RET
ASEMBYTE    ENDP
;
;
CRLF PROC NEAR              ;SENDS A CR AND LF TO THE PRINTER
            PUSH AX
            MOV AH,CR
            CALL SENDBYTE
            MOV AH,LF
            CALL SENDBYTE
            POP AX
            RET
CRLF        ENDP
;
;
GET_DOT PROC NEAR
;THIS PROCEDURE IS A SLIGHT VARIATION OF DOT_PLOT. SEE THAT FILE FOR
;MORE COMMENTS. THIS VERSION FETCHES THE STATE, 0 OR 1, OF THE DOT AT
;CO_ORDS X IN THE AX REGISTER AND Y IN THE BX REGISTER AND RETURNS
;WITH IT IN THE THE CARRY FLAG
;
```

```
;ES MUST POINT AT THE SCREEN
;
;AX MUST CONTAIN X VALUE BETWEEN 0 AND 719
;BX MUST CONTAIN Y VALUE BETWEEN 0 AND 347
;
             PUSH DX
             PUSH CX
             PUSH BX
             PUSH AX
             PUSH AX
             PUSH BX
             AND BX,3
             MOV CX,BX
             INC CL
             MOV BX,-2000H
START1:      ADD BX,2000H
             LOOP START1
             POP AX
             SHR AX,1
             SHR AX,1
             MUL NINETY
             MOV DI,AX
             POP AX
             MOV CL,AL
             AND CL,00000111B
             SHR AX,1
             SHR AX,1
             SHR AX,1
             ADD BX,AX
             MOV AL,10000000B
             SHR AL,CL              ;PUT A 1 IN (7-XMOD 8) BIT
             CLC                    ;CLEAR CARRY IN ANTICIPATION
             AND AL,ES:[BX+DI]      ;TEST THE BIT
             JZ COMPLETE
             STC                    ;SET THE CARRY, ITS 1
COMPLETE:    POP AX
             POP BX
             POP CX
             POP DX
             RET
GET_DOT      ENDP
;
;
;PLACE ANY DATA HERE IN THE CODE SEGMENT. PLACING THE DATA AFTER THE
;CODE MEANS THE ENTRY POINT STAYS AT IP=0000
;
OLD_DS  DW  ?               ;THESE LOCATIONS ARE ESSENTIAL
OLD_ES  DW  ?               ;STORAGE FOR THE SEGMENT REGISTERS
OLD_SP  DW  ?               ;AND STACK POINTER
OLD_SS  DW  ?
```

```
;
XCO_ORD  DW ?
YCO_ORD  DW ?
BYTE_TO_SEND DB ?
NINETY DB 90
;
TABLE1 DB 27,"*",6,208,2,4    ;90 DOTS PER INCH AND 720 DOTS ACROSS PAGE
TABLE2 DB 27,"A",8,4          ;8 DOTS VERTICALLY PER LINE AND EOT

CSEG    ENDS
END     CONFIG                ;SO THE LINKER GENERATES THE CORRECT
                              ;ENTRY ADDRESS
```

APPENDIX 3. PMT Non-linearity Compensation Calculations

Appendix 3.1 FIRST.BAS

```
1 REM ************************************************************************
2 REM *       CALCULATIONS OF PMT NON-LINEARITY COMPENSATION                  *
3 REM *                      by    R. GRANT                                   *
4 REM ************************************************************************

5 SCREEN 2:CLS:KEY OFF:IP=0
6 DEF SEG=&H3A00:BLOAD "PROCF",0:REM LOAD CODE FOR PRINTER DUMP
10 REM FIRST READ THE DATA FILE TO BE PROCESSED
20 INPUT "PLEASE TYPE THE NAME OF THE DATA FILE TO BE PROCESSED  ",N$
25 INPUT "PLEASE TYPE THE NUMBER OF SHOTS USED TO CREATE THIS DATA
   ",SHOTS
30 DEF SEG =&H3000
40 BLOAD N$,0
42 CODEOFFSET=&H0:GOSUB 4000:REM PLOT THE RAW DATA
44 LOCATE 25,1:PRINT"                                HEIGHT ABOVE
   GRAHAMSTOWN (KM)";
46 LOCATE 1,50:PRINT"RAW DATA  ";N$:LOCATE 2,50:PRINT SHOTS;"SHOTS"
   :BEEP
47 Q$=INKEY$
48 IF Q$="P" THEN DEF SEG=&H3A00:CALL IP:REM DUMP TO PRINTER
49 IF Q$ <> "C" THEN 47:REM CONTINUE ONLY WHEN C IS TYPED
50 REM****************************************
60 REM NOW STRAIGHTEN THE DATA
70 REM FIRST CREATE A LOOKUP TABLE OF STRAIGHTENED POINTS
75 DEF SEG =&H3000
80 J=0
90 FOR I = &H8000 TO &H8000+2*99 STEP 2
100 POKE I+1,INT(J/256):POKE I,J-256*INT(J/256)
110 J=J+10
120 NEXT
130 BLOAD "BASPTS",&H8000+200: REM THAT FILLS IN THE REST
140 REM NOW NORMALISE THE ACTUAL DATA TO 15 PULSES
145 CLS
150 PRINT"NORMALISING TO 15 PULSES AND LOOKING UP STRAIGHT VALUES"
160 DEF SEG =&H3000
170 FOR I = 2*667-4 TO 0 STEP -2
180 ACTUAL = 256*PEEK(I+1)+PEEK(I): REM GET REAL COUNT
190 B=INT(ACTUAL*15/SHOTS+.5)
200 IF B > 210 THEN B=0:REM CHECK FOR OUT OF RANGE SATURATION OF PMT
210 STR = 256*PEEK(2*B+&H8000+1)+PEEK(2*B+&H8000):REM LOOKS UP STRAIGHT
    VALUE
215 STR = INT(STR*SHOTS/150+.5):REM MAKE UP TO NUMBER OF SHOTS AGAIN
220 POKE I+1-8+&H2000,INT(STR/256):POKE I-8+&H2000,STR-256*INT(STR/256)
225 REM -8 IS TO COMPENSATE FOR GRAHAMSTOWN BEING 4 RANGE BINS ABOVE
    SEA LEVEL
230 NEXT
```

```
240 REM*****************************************
250 REM NOW PLOT THE NEW DATA
270 CODEOFFSET =&H2000:GOSUB 4000:REM PLOT THE POINTS
273 LOCATE 25,1:PRINT"                                HEIGHT ABOVE SEA
    LEVEL (KM)";
275 LOCATE 1,50:PRINT"COMPENSATED DATA  ";N$:LOCATE 2,50:PRINT SHOTS;
    "SHOTS"
276 BEEP
280 Q$=INKEY$
290 IF Q$="P" THEN DEF SEG=&H3A00:IP=0:CALL IP
300 IF Q$<>"C" THEN 280
310 REM NOW SAVE THE STRAIGHTENED DATA
320 CLS:INPUT"DO YOU WISH TO SAVE THE STRAIGHTENED DATA? (Y/N)",M$
330 IF M$ = "Y" THEN GOSUB 6000
999 STOP
4000 REM SUBROUTINE TO PLOT A SET OF DATA
4001 REM CODEOFFSET MUST BE OFFSET TO RAW DATA
4010 REM FIRST FIND LARGEST NUMBER IN DATA
4020 PRINT"FINDING LARGEST":LARGEST=0
4030 FOR I=2*667-2*14+CODEOFFSET TO CODEOFFSET STEP -2:REM OMIT FIRST
     2 KM
4040 COUNT=256*PEEK(I+1)+PEEK(I)
4050 IF LARGEST < COUNT THEN LARGEST = COUNT
4060 NEXT
4070 REM NOW NORMALISE DATA TO 300 MAX AND MOVE TO DEF SEG:$3700
4080 PRINT"NORMALISING AND MOVING DATA"
4090 FOR I=2*667-2*14+CODEOFFSET TO CODEOFFSET STEP -2
4100 COUNT = 256*PEEK(I+1)+PEEK(I)
4110 NORMCOUNT=INT(COUNT*300/LARGEST)
4120 POKE I+1+&H7000-CODEOFFSET,INT(NORMCOUNT/256)
     :POKE I+&H7000-CODEOFFSET,NORMCOUNT-256*INT(NORMCOUNT/256)
4130 NEXT
4140 DEF SEG=&H3700:REM POINT DEF SEG AT DATA TO BE PLOTTED
4150 GOSUB 5000:GOSUB 10000:REM PLOT POINTS
4160 RETURN
5000 CLS
5010 LINE (52,0)-(52,300)
5030 LINE -(719,300)
5035 FOR I= 0 TO 250 STEP 50:LINE (50,I)-(54,I):NEXT I
5040 PRINT"N":PRINT"O":PRINT"R":PRINT"M":PRINT"A":PRINT"L":PRINT"I"
     :PRINT"S":PRINT"E":PRINT "D":PRINT:PRINT:PRINT
5045 PRINT"C":PRINT"O":PRINT"U":PRINT"N":PRINT"T":PRINT"S"
5050 A=119:B=66.7
5055 FOR I = 1 TO 10
5060 X = 52+ INT(I*B+.5)
5065 LINE (X,298)-(X,302)
5070 NEXT I
5075 PRINT:PRINT:PRINT:PRINT"    0       10     20      30     40
     50     60      70     80     90    100";
5200 RETURN
```

```
6000 REM ROUTINE TO SAVE THE STRAIGHTENED DATA ******************
6010 PRINT"THE FILE OF ORIGINAL DATA WAS CALLED";N$
6020 INPUT"PLEASE TYPE A NAME FOR THE STRAIGHTENED DATA ",M$
6030 DEF SEG=&H3200:BSAVE M$,0,1402
6040 PRINT "DONE"
6050 RETURN
6060 REM ********************
10000 REM DOT JOINING ROUTINE
10005 REM DEF SEG MUST POINT AT THE START OF THE DATA AREA
10010 X=51+14 :REM FIRST COLUMN ,MISSING FIRST 14 BINS WITH GARBAGE
10020 COUNT=256*PEEK(2*667+1-2*14)+PEEK(2*667-2*14)
      :REM COUNT OF FIRST COLUMN
10030 YOLD=300-COUNT
10040 IF YOLD<0 THEN YOLD = 0:REM COUNTS > 300 ARE AT THE TOP OF THE
      SCREEN
10050 FOR I=2*667-2-2*14 TO 0 STEP -2
10060 COUNT=256*PEEK(I+1)+PEEK(I)
10070 Y=300-COUNT:IF Y < 0 THEN Y=0
10080 X=X+1:REM NEXT COLUMN
10090 LINE(X,YOLD)-(X,Y)
10100 YOLD=Y
10110 NEXT I
10120 RETURN
20000 STOP
```

APPENDIX 4. Temperature Calculations

Appendix 4.1 TEMPER.PAS

```
(*************************************************************************)
(*                                                                       *)
(*       TURBO PASCAL PROGRAM TO CALCULATE ATMOSPHERIC TEMPERATURE       *)
(*                                  by                                   *)
(*                               R. GRANT                                *)
(*                                                                       *)
(*************************************************************************)

PROGRAM Temper; (* Computes temperature profiles  from LIDAR data *)

{$I TYPEDEF.SYS}
{$I GRAPHIX.HGC}
{$I KERNEL.SYS}
{$I WINDOWS.SYS}
{$I FINDWRLD.HGH}
{$I AXIS.HGH}
{$I POLYGON.HGH}

CONST
  MAX       = 1420;   (* Number of bytes in BASIC'S data files *)
                      (* +  header *)
  MAXDATA   =  667;   (* Number of data points taken by PCS *)
  BinLength =  150;   (* Length of a range bin in meters *)
  g         = 9.81;   (* Acceleration due to gravity at the surface *)

VAR
  RawDataFileName        :  STRING[15];
  FinishedDataFileName   :  STRING[15];
  Dummy                  :  STRING[15];
  I                      :  INTEGER;
  TopBinCount            :  INTEGER;
  BeamEnergy             :  REAL;
  ExtCoeff               :  REAL;
  K                      :  REAL;
  X                      :  REAL;
  Extinction             :  REAL;      (* Kappa * delta Z *)
  SumOfExtinctions       :  REAL;      (* Sum of Kappa * delta Z *)
  TopTemperature         :  REAL;
  TopPressure            :  REAL;
  TopOfBinPressure       :  REAL;
  NumberDensity          :  INTEGER;
  StartingAlt            :  REAL;
  StoppingAlt            :  REAL;
  TopBin                 :  INTEGER;
  BottomBin              :  INTEGER;
  PCS                    :  INTEGER;
```

```
  Alt                    :  ARRAY[0..MAXDATA] of REAL;
  Actual                 :  ARRAY[0..MAXDATA] of integer
                            ABSOLUTE $3000:$0000;
  KelvinTemp             :  ARRAY[0..MAXDATA] of real;
  CelciusTemp            :  INTEGER;
  MassDensity            :  ARRAY[0..MAXDATA] of Real;


PROCEDURE ReadLidarData;

(* This procedure reads a basic file into memory at $3000:0000 *)

CONST
  MAX  =  1420;          (* Size of BASIC'S data files plus a few *)

VAR
  Filvar           :     File of byte;
  I                :     Integer;
  RawData          :     ARRAY[0..MAX] of byte ABSOLUTE $3000:$0000;
  RawDataFileName  :     STRING[15];

BEGIN
  ClrScr;
  WRITE('Please type the name of the file of raw data to be processed
        ');
  READLN(RawDataFileName);
  WRITE('Loading raw data.....');
  Assign(Filvar,RawDataFileName);
  Reset(Filvar);
  I := 0;

    WHILE NOT EOF (Filvar) DO
      BEGIN
        Read (Filvar,RawData[I]);
        I := I+1;
      END;

  CLOSE (Filvar);

  WRITELN('Stripping BASIC''''S Header');
    FOR I := 0 TO MAX DO
      BEGIN
        RawData[I] := RawData[I+7];
      END;

END;   (* ReadLidarData *)
```

```
PROCEDURE DrawTheGraph; (* Draws a graph of temperature vs altitude *)

VAR
  NumberOfPoints     :  INTEGER;
  a                  :  PlotArray;
  I                  :  INTEGER;
  Temp               :  REAL;

 BEGIN
   DefineWindow(1,0,0,XMaxGlb,YMaxGlb);

   SelectWindow(1);

  NumberOfPoints := TopBin - BottomBin;
  PCS := 668 - BottomBin;
   FOR I := 1 TO NumberOfPoints DO
     BEGIN
       a[I,1] := KelvinTemp[PCS]-273;  (* Convert to Celcius *)
       a[I,2] := Alt[PCS];
       PCS := PCS - 1;
     END;

(****  FindWorld(1,a,NumberOfPoints,1,1);   (***** Auto findworld ****)

 DefineWorld(1,-150.0,StartingAlt,-1.0,StoppingAlt);
   (*** remove for auto-world ***)

(*  WITH World[1] DO
(*    BEGIN
(*      Temp := Y1;           (* Flip the Y co-ordinates *)
(*      Y1 := Y2;             (* Only needed for FindWorld *)
(*      Y2 := Temp;
(*    END;    ****)

  SelectWorld(1);

  SelectWindow(1);

  SetLineStyle(0);

  DrawAxis(8,7,0,0,0,0,0,0,False);

  DrawPolygon(a,1,-NumberOfPoints,7,2,0);

 END;  (* DrawTheGraph *)
```

```
PROCEDURE PrntGrph; (* Dumps the HI RES Hercules screen to the FX-80 *)

(* This procedure reads a basic file into memory at $3000:0000 *)
(* and then runs it as code *)

CONST
  MAX  =  340;          (* Size of BASIC'S PROCF.BAS file plus a few *)

VAR
  Filvar           :     File of byte;
  I                :     Integer;
  PrtScrnCode      :     ARRAY[0..MAX] of byte ABSOLUTE $3000:$0000;

BEGIN
  Assign(Filvar,'PROCF.BAS');
  Reset(Filvar);
  I := 0;

    WHILE NOT EOF (Filvar) DO
      BEGIN
        Read (Filvar,PrtScrnCode[I]);
        I := I+1;
      END;

  CLOSE (Filvar);

    FOR I := 0 TO MAX DO
      BEGIN
        PrtScrnCode[I] := PrtScrnCode[I+7];
      END;

  (* Now run the code *)
  INLINE($9A/>$0000/>$3000);

END;   (* PrntGrph *)

PROCEDURE GetStartingInfo;

VAR
  BIN        : INTEGER;
  RealBIN    : REAL;
  RealTopBin : REAL;

BEGIN
  WRITELN('What is the highest altitude to start calculating
           the temperature?');
  WRITE('Type the height in meters   ');
  READLN(StartingAlt);
  TopBin := ROUND((StartingAlt-11) / 150);
```

```
  WRITELN('Down to what altitude should temperature be calculated?');
  WRITE('Again type height in meters  ');
  READLN(StoppingAlt);
  BottomBin := ROUND((StoppingAlt-11) / 150);

  RealTopBin := TopBin;
  WRITE('Please type the temperature at ',RealTopBin * 150 + 11,'
        meters in kelvin ');
  READLN(TopTemperature);
  WRITE('Please type the pressure at ',RealTopBin * 150 + 11,' meters
        in millibars ');
  READLN(TopPressure);

  FOR BIN := 1 TO MAXDATA DO
    BEGIN
      PCS := 668 - BIN;        (* This initialises the array Alt[PCS] *)
      RealBIN := BIN;
      Alt[PCS] := 150*RealBIN+11;
    END;


END;  (* Get starting info *)

PROCEDURE SubtractNoise;

(* This averages the last 200 bins where no return is detected and *)
(* subtracts this from the rest of the data (last 8 skipped)       *)

CONST
  BinsToAverage  = 200;

VAR
  I        :  INTEGER;
  Sum      :  INTEGER;
  Average  :  INTEGER;

BEGIN
  Sum     := 0;
  Average := 0;
  FOR I   := 8 TO BinsToAverage + 8 DO
    BEGIN
      Sum     := Sum + Actual[I];
    END;
  Average := ROUND(Sum DIV 200);  (* Calculate average *)
  FOR I := 0 TO MAXDATA DO
    BEGIN
      Actual[I] := Actual[I] - Average;
      IF Actual[I] < 0 THEN Actual[I] := 0; (* Subtract noise *)
    END;
END;  (* SubtractNoise *)
```

```
PROCEDURE FindTopBinCount;

(* This finds the count to use for that in the highest bin by *)
(* averaging the count in this bin with that of "spread" bins on *)
(* either side *)

CONST
  Spread = 5;

VAR
  I,N    :  INTEGER;
(* Gives the global variable PCS the index for the data in *)
(* "Actual[n]" *)

BEGIN
  N  := 0;

  FOR I := TopBin - Spread TO TopBin + Spread DO
    BEGIN
      PCS := 668 - I;    (* Array index of corresponding data *)
      N   := N + Actual[PCS];
    END;

  PCS := 668-TopBin; (*Init PCS to point to TpoBin data in the arrays*)
  TopBinCount := ROUND(N DIV (2 * Spread + 1)); (*Average the counts *)
WRITELN('Top bin count is  ',TopBinCount); (* FindTopBinCount *)

PROCEDURE CalcTopMassDensity (Press,Temp : REAL);

(* Calculates mass density at top of range in the REAL array *)
(* MassDensity *)

CONST
  MolMass = 0.02896; (* Mean molecular mass of air in kg per mol *)
  R       = 8.314;   (* Universal gas constant in joules per mol *)
                     (* per kelvin *)

VAR
  PressMKS    :  Real;

BEGIN
  PressMKS   :=  Press * 100;  (* Convert millibars to N/m2 *)
  MassDensity[PCS] := (PressMKS * MolMass )/(R * Temp );

END;  (* CalcMassDensity *)
```

```
PROCEDURE CalcK (Count,Height,MassDensity : REAL);

BEGIN
  K := Count * Height * Height / MassDensity ;
END;  (* CalcK *)

PROCEDURE CalcExtinction (MassDensity  :  REAL);

CONST
  MolMass = 0.02896;
  AvogadroNo = 6.02E+23;
  BinLength = 150;
  Sigma{589} = 3.4714E-31; (* Total scattering X section of an air   *)
                           (*  molecule in square meters at 589 nm   *)

VAR
  NumberDensity  :  REAL;
  Kappa          :  REAL;

BEGIN
  NumberDensity := MassDensity * AvogadroNo / MolMass;
  Kappa := NumberDensity * Sigma{589};
  Extinction := BinLength * Kappa;
  SumOfExtinctions := SumOfExtinctions + Extinction;

END; (* CalcExtinction *)

PROCEDURE CalcNewBeamEnergy (Extinction : REAL);

VAR
  ScatteredPortion   :  REAL;

BEGIN
  ScatteredPortion := Extinction * BeamEnergy;
  BeamEnergy := BeamEnergy + ScatteredPortion;
                                          (* Finds beam energy in  *)
                                          (* next bin down         *)
END;  (* CalcNewBeamEnergy *)

PROCEDURE CalcBottomOfTopBinPress (Pressure,Density : REAL);

CONST
  g         = 9.81;
  BinLength = 150;

VAR
  PressureMKS   : REAL;
```

```
BEGIN
  PressureMKS := Pressure * 100;
  TopOfBinPressure := PressureMKS + Density * g * BinLength / 2 ;
END; (* CalcBottomOfTopBinPressure *)

PROCEDURE CalcMassDensity (Count,Height  : REAL );

VAR
  Kappa      :  REAL;
  Atten      :  REAL;      (* 1 - SumOfExtinctions *)

BEGIN
  Atten  := 1 - SumOfExtinctions;
  MassDensity[PCS] :=Count * Height * Height * Atten /(K * BeamEnergy);
END; (* CalcMassDensity *)

PROCEDURE CalcX (MassDensity,TopOfBinPressure : REAL);

BEGIN
  X := MassDensity * g * BinLength / TopOfBinPressure;
END;  (* CalcX *)

PROCEDURE CalcTemp (X : REAL);

CONST
  MolMass   = 0.02896;
  g         = 9.81;
  R         = 8.314;
  BinLength = 150;

BEGIN
  KelvinTemp[PCS] := MolMass * g * BinLength / (R * LN (1 + X));
END;  (* CalcTemp *)

PROCEDURE CalcTopOfNextBinPressure (MassDensity : REAL);

CONST
  g         = 9.81;
  BinLength = 150;

BEGIN
  TopOfBinPressure := TopOfBinPressure + MassDensity * g * BinLength ;
END; (* CalcTopOfNextBinPressure *)
```

```
PROCEDURE SaveResults; (* Writes results to disk *)

VAR
  Filvar          :      File of real;
  DiskFileName    :      String[12];
  NumberOfReals   :      Integer;
  I               :      Integer;

BEGIN
  ClrScr;
  WRITELN('This procedure saves the calculated altitude and
           temperature');
  WRITELN('to disk as reals in the order altitude then temperature.');
  WRITELN('Please type a name for the file of data ');
  READLN(DiskFileName);
  Assign(Filvar,DiskFileName);
  WRITE('Saving  ',DiskFileName);
     Rewrite(Filvar); (* Open the file and delete any data *)

     PCS := 668-TopBin;
       WRITE(Filvar,Alt[PCS],TopTemperature);  (* Do initial values *)

        FOR I := TopBin-1 DOWNTO BottomBin DO
          BEGIN
            PCS := 668-I;
            WRITE(Filvar,Alt[PCS],KelvinTemp[PCS]);
          END;
  CLOSE(Filvar);

END; (* SaveResults *)


BEGIN (* Main Program *)
    ReadLidarData;
    SubtractNoise;

  REPEAT
      GetStartingInfo; (** Zero relevant variables **)
      FindTopBinCount;
      CalcTopMassDensity (TopPressure,TopTemperature);
      CalcK (TopBinCount,Alt[PCS],MassDensity[PCS]);
        SumOfExtinctions  := 0;
      CalcExtinction  (MassDensity[PCS]);
        BeamEnergy  := 1.0;
      CalcNewBeamEnergy (Extinction);
      CalcBottomOfTopBinPress (TopPressure,MassDensity[PCS]);


    WRITELN('Thinking.........');
```

```
    FOR I := TopBin - 1 DOWNTO BottomBin DO
      BEGIN
        PCS := 668-I;                (* Index into the arrays *)
        CalcMassDensity              (Actual[PCS],Alt[PCS]);
        CalcX                        (MassDensity[PCS],TopOfBinPressure);
        CalcTemp                     (X);
        CalcExtinction               (MassDensity[PCS]);
        CalcNewBeamEnergy            (Extinction);
        CalcTopOfNextBinPressure     (MassDensity[PCS]);
      END;

    WRITELN('Bin':3,'Height':18,'Measured':18,'Temperature (C)':18);
    WRITELN;
      PCS := 668-TopBin;
      WRITE(TopBin:3,Alt[PCS]:18:5,Actual[PCS]:18);
      WRITELN(TopTemperature-273:18:2); (* Does the starting *)
                                        (* values           *)
    FOR I := TopBin-1 DOWNTO BottomBin DO
      BEGIN
        PCS := 668-I;
        WRITE(I:3,Alt[PCS]:18:5,Actual[PCS]:18);
        WRITELN(KelvinTemp[PCS]-273:18:2);
      END;

    WRITELN('Final beam energy = ',BeamEnergy);
    WRITELN('Final extinction factor = ',1-SumOfExtinctions);
    WRITELN('(R)epeat with another starting temperature,
             (S)ave the results');
    WRITELN('draw the (G)raph or (Q)uit?');
      REPEAT
        READLN(Dummy);
      UNTIL (Dummy = 'R') OR (Dummy = 'Q') OR (Dummy = 'G')
            OR (Dummy = 'S');

    IF Dummy = 'G' THEN
      BEGIN
        InitGraphic;
        DrawTheGraph;
          REPEAT
            READLN(Dummy);
          UNTIL (Dummy = 'P') OR (Dummy = 'X') OR (Dummy = 'S');

          IF Dummy = 'P' THEN PrntGrph;
        LeaveGraphic;
    END;
     IF Dummy = 'S' THEN SaveResults;
  UNTIL Dummy = 'Q';

END.
```

Appendix 4.2 TEMPAVE.PAS

```
(*************************************************************************)
(*                                                                       *)
(*        TURBO PASCAL PROGRAM TO AVERAGE SEVERAL FILES OF               *)
(*                    ALTITUDE AND TEMPERATURE DATA                      *)
(*                                by                                     *)
(*                             R. GRANT                                  *)
(*                                                                       *)
(*************************************************************************)

PROGRAM Tempave;
{$I TYPEDEF.SYS}
{$I GRAPHIX.HGC}
{$I KERNEL.SYS}
{$I WINDOWS.SYS}
{$I FINDWRLD.HGH}
{$I AXIS.HGH}
{$I POLYGON.HGH}

(* This averages the values of temperature  contained in several  *)
(* files of altitude and temperature. It computes and draws error *)
(* bars of +/- one sample standard deviation on the graph         *)

CONST
  MAXDATA      = 667;     (* Largest number of data points possible *)

VAR
  Dummy                  :  STRING[12];
  CurrentFileName        :  STRING[16];
  FileNumber             :  INTEGER;
  BIN                    :  INTEGER;     (* Pointer into the array *)
  NewData                :  Array[1..MAXDATA] of REAL;
  Sum                    :  Array[1..MAXDATA] of REAL;
  SumOfSquares           :  Array[1..MAXDATA] of REAL;
  AvTemp                 :  Array[1..MAXDATA] of REAL;
  Altitude               :  Array[1..MAXDATA] of REAL;
  StdDeviation           :  Array[1..MAXDATA] of REAL;
  NoOfResults            :  Array[1..MAXDATA] of INTEGER;

PROCEDURE PrntScrn; (* Dumps the HI RES Hercules screen to the FX-80 *)

(* This procedure reads a basic file into memory at $3000:0000 *)
(* and then runs it as code *)

CONST
  MAX  =  340;          (* Size of BASIC'S PROCF.BAS file plus a few *)

VAR
  Filvar           :     File of byte;
```

```
  I                :     Integer;
  PrtScrnCode      :     ARRAY[0..MAX] of byte ABSOLUTE $3000:$0000;

BEGIN
  Assign(Filvar,'PROCF.BAS');
  Reset(Filvar);
  I := 0;

    WHILE NOT EOF (Filvar) DO
      BEGIN
        Read (Filvar,PrtScrnCode[I]);
        I := I+1;
      END;

  CLOSE (Filvar);

    FOR I := 0 TO MAX DO
      BEGIN
        PrtScrnCode[I] := PrtScrnCode[I+7];
      END;

  (* Now run the code *)
  INLINE($9A/>$0000/>$3000);

END;   (* PrntScrn *)

PROCEDURE DrawTheGraph; (* Draws a graph of temperature vs altitude  *)
                        (* and puts in error bars of +/- one std. dev*)

VAR
  NoOfPoints        :  INTEGER;
  a                 :  PlotArray;
  I                 :  INTEGER;
  Temp              :  REAL;
  StartingAlt       :  REAL;
  StoppingAlt       :  REAL;
  Left              :  REAL;
  Right             :  REAL;
  BottomBin         :  INTEGER;
  TopBin            :  INTEGER;

 BEGIN
   InitGraphic;
   DefineWindow(1,0,0,XMaxGlb,YMaxGlb);
   SelectWindow(1);

  NoOfPoints := 0;
  StartingAlt := 0;
  StoppingAlt := 0;
   FOR BIN := 1 TO MAXDATA DO
```

```
    BEGIN
      IF NoOfResults[BIN] >0 THEN
        BEGIN
          IF StartingAlt = 0 THEN StartingAlt := Altitude[BIN];
          NoOfPoints      := NoOfPoints + 1;
          a[NoOfPoints,1] := AvTemp[BIN]-273; (* Convert to celcius *)
          a[NoOfPoints,2] := Altitude[BIN];
          StoppingAlt := Altitude[BIN];
        END;
    END;

(******* FindWorld(1,a,NumberOfPoints,1,1);   ** Auto findworld **)
DefineWorld(1,-150.0,StartingAlt,-1.0,StoppingAlt);
(*** remove for auto-world ***)

    WITH World[1] DO
      BEGIN
        Temp := Y1;          (* Flip the Y co-ordinates *)
        Y1 := Y2;
        Y2 := Temp;
      END;

  SelectWorld(1);
  SelectWindow(1);
  SetLineStyle(0);

  DrawAxis(8,7,0,0,0,0,0,0,False);

  DrawPolygon(a,1,-NoOfPoints,7,1,0);

  (* Now do the error bars *)

  BottomBin := ROUND((StartingAlt-11)/150);
  TopBin    := ROUND((StoppingAlt-11)/150);
    FOR BIN := BottomBin TO TopBin DO
      BEGIN
        Left := AvTemp[BIN]-273-StdDeviation[BIN];
        Right := AvTemp[BIN]-273+StdDeviation[BIN];
        a[1,1] := Left;
        a[1,2] := Altitude[BIN];
        a[2,1] := AvTemp[BIN]-273;
        a[2,2] := Altitude[BIN];
        a[3,1] := Right;
        a[3,2] := Altitude[BIN];
        AxisGlb := TRUE; (* Keep same scaling as axes  12.27 am !!!! *)
        DrawPolygon(a,1,-3,0,1,0);
      END;

  REPEAT
    READLN(Dummy);
```

```
  UNTIL (Dummy = 'P') OR (Dummy = 'X');

  IF Dummy = 'P' THEN PrntScrn;  (* Send to the FX-80 printer *)

  LeaveGraphic;

 END;  (* DrawTheGraph *)


PROCEDURE ClearArrays; (*Initialises altitude array and clears others*)

VAR
  RealBIN : REAL;

BEGIN
  FOR BIN := 1 TO MAXDATA DO
    BEGIN
      RealBIN           := BIN;
      Altitude[BIN]     := 150*RealBIN+11;   (* Bin length 150m *)
      Sum[BIN]          := 0;
      SumOfSquares[BIN] := 0;
      NoOfResults[BIN]  := 0;
      StdDeviation[BIN] := 0;
    END;
  END; (* ClearArrays *)

PROCEDURE GetNameOfFile;

  BEGIN
    WRITELN('Please type the name of the file of altitude
             and temperature ');
    WRITE(' to be included ');
    READLN(CurrentFileName);
  END; (* GetNameOfFile *)

PROCEDURE GetData;

(* This procedure reads a file of altitude and temperature *)
(* and updates the relevant arrays *)

VAR
  Filvar          :     File of real;
  Altitude        :     REAL;
  Temperature     :     REAL;

BEGIN
  WRITE('    Loading raw data.....');
  Assign(Filvar,CurrentFileName);
  Reset(Filvar);
```

```
    WHILE NOT EOF (Filvar) DO
      BEGIN
        Read (Filvar,Altitude,Temperature);
        BIN               := ROUND((Altitude-11)/150);
                             (* Find the right bin *)
        NewData[BIN]      := Temperature;  (* and put the data in  *)
        Sum[BIN]          := Sum[BIN] + NewData[BIN];
        SumOfSquares[BIN] := SumOfSquares[BIN]+SQR(NewData[BIN]);
        NoOfResults[BIN]  := NoOfResults[BIN]+1;
        AvTemp[BIN]       := Sum[BIN]/NoOfResults[BIN];
      END;

  CLOSE (Filvar);

END;   (* GetData *)

PROCEDURE CalcStdDeviations;

  BEGIN
    WRITELN('Calculating standard deviations ');
      FOR BIN := 1 TO MAXDATA DO
        BEGIN
          IF NoOfResults[BIN] > 1 THEN
          StdDeviation[BIN] := SQRT((SumOfSquares[BIN]-SQR(Sum[BIN])/
                               NoOfResults[BIN])/(NoOfResults[BIN]-1));
        END;
  END;  (*CalcStdDeviations *)

PROCEDURE ShowArrays; (* Sends the contents of some arrays *)
                      (* to the screen *)

  BEGIN
    WRITE('Bin':3,'Altitude':18,'Temperature':18,'No of results':18);
    WRITELN('Standard Deviation':18);
      FOR BIN := 1 TO MAXDATA DO
        BEGIN
          IF NoOfResults[BIN] >0 THEN
            BEGIN
              WRITE(BIN:3,Altitude[BIN]:18:0,AvTemp[BIN]:18:6);
              WRITELN(NoOfResults[BIN]:18,StdDeviation[BIN]:18:6);
            END;
        END;
  END;  (* ShowArrays *)
```

```
PROCEDURE SaveArrays; (* Writes results to disk *)

VAR
  Filvar          :      File of real;
  DiskFileName    :      String[12];

BEGIN
  ClrScr;
  WRITELN('This procedure saves the calculated altitudes,
           temperature');
  WRITELN('and standard deviations to disk as reals in the order
           altitude');
  WRITELN('then temperature then standard deviation.');
  WRITELN('Please type a name for the file of data ');
  READLN(DiskFileName);
  Assign(Filvar,DiskFileName);
  WRITE('Saving  ',DiskFileName);
     Rewrite(Filvar); (* Open the file and delete any data *)

        FOR BIN := 1 to MAXDATA DO
          BEGIN
            IF NoOfResults[BIN] > 0 THEN
            WRITE(Filvar,Altitude[BIN],AvTemp[BIN],StdDeviation[BIN]);
          END;
  CLOSE(Filvar);

END; (* SaveArrays *)

PROCEDURE UpdateAndPrintArrays;

  BEGIN
    CalcStdDeviations;
    ShowArrays;
  END; (* UpdateAndPrintArrays *)

BEGIN (* Main program *)
  ClearArrays;
    REPEAT
      GetNameOfFile;
      GetData;
        REPEAT;
          WRITELN('(A)nother file, (V)iew arrays, (G)raph,
                   (S)ave results or (Q)uit');
            REPEAT
              READLN(Dummy);
            UNTIL (Dummy = 'A') OR (Dummy = 'V') OR (Dummy = 'G')
            OR (Dummy = 'S') OR (Dummy = 'Q');

          CASE Dummy OF
```

```
            'V'   :   UpdateAndPrintArrays;
            'G'   :   DrawTheGraph;
            'S'   :   SaveArrays;
          END; (* CASE *)
      UNTIL (Dummy ='A') OR (Dummy = 'Q');

  UNTIL Dummy = 'Q'

END. (* Main program *)
```

APPENDIX 5. Scattering Ratio Calculations

Appendix 5.1 SCATRAT.PAS

```
(*************************************************************************)
(*                                                                       *)
(*       TURBO PASCAL PROGRAM TO CALCULATE SCATTERING RATIOS             *)
(*                                  by                                   *)
(*                              R. GRANT                                 *)
(*                                                                       *)
(*************************************************************************)

PROGRAM ScatRat; (* Computes scattering ratios from LIDAR data *)

{$I TYPEDEF.SYS}
{$I GRAPHIX.HGC}
{$I KERNEL.SYS}
{$I WINDOWS.SYS}
{$I FINDWRLD.HGH}
{$I AXIS.HGH}
{$I POLYGON.HGH}

CONST
  MAX     =   1420;    (* Number of bytes in BASIC'S data files *)
                       (* +  header *)
  MAXDATA =   667;     (* Number of data points taken by PCS *)

VAR
  TempPressFileName      :  STRING[15];
  RawDataFileName        :  STRING[15];
  FinishedDataFileName   :  STRING[15];
  Dummy                  :  STRING[15];
  I                      :  INTEGER;
  TopBinCount            :  INTEGER;
  BeamEnergy             :  REAL;
  ExtCoeff               :  REAL;
  K                      :  REAL;
  Extinction             :  REAL;      (* Kappa * delta Z *)
  SumOfExtinctions       :  REAL;      (* Sum of Kappa * delta Z *)
  StartingScatRatio      :  REAL;      (* Assumed scattering ratio at *)
                                       (* the start *)
  NumberDensity          :  INTEGER;
  StartingAlt            :  REAL;
  StoppingAlt            :  REAL;
  TopBin                 :  INTEGER;
  BottomBin              :  INTEGER;
  PCS                    :  INTEGER;
  Alt                    :  ARRAY[0..MAXDATA] of real;
  Actual                 :  ARRAY[0..MAXDATA] of integer
                            ABSOLUTE $3000:$0000;
```

```
  Rayleigh                : ARRAY[0..MAXDATA] of integer;
  ScatRatio               : ARRAY[0..MAXDATA] of real;
  Temp                    : ARRAY[0..MAXDATA] of real;
  Press                   : ARRAY[0..MAXDATA] of real;
  MassDensity             : ARRAY[0..MAXDATA] of real;


PROCEDURE PrntScrn; (* Dumps the HI RES Hercules screen to the FX-80 *)

(* This procedure reads a basic file into memory at $3000:0000 *)
(* and then runs it as code *)

CONST
  MAX  =  340;          (* Size of BASIC'S PROCF.BAS file plus a few *)

VAR
  Filvar          :     File of byte;
  I               :     Integer;
  PrtScrnCode     :     ARRAY[0..MAX] of byte ABSOLUTE $3000:$0000;

BEGIN
  Assign(Filvar,'PROCF.BAS');
  Reset(Filvar);
  I := 0;

    WHILE NOT EOF (Filvar) DO
      BEGIN
        Read (Filvar,PrtScrnCode[I]);
        I := I+1;
      END;

  CLOSE (Filvar);

    FOR I := 0 TO MAX DO
      BEGIN
        PrtScrnCode[I] := PrtScrnCode[I+7];
      END;

  (* Now run the code *)
  INLINE($9A/>$0000/>$3000);

END;   (* PrntScrn *)

PROCEDURE DrawTheGraph; (* Draws a graph of scat. ratio vs altitude *)

VAR
  NumberOfPoints    :  INTEGER;
  a                 :  PlotArray;
  I                 :  INTEGER;
  Temp              :  REAL;
```

```
 BEGIN
   DefineWindow(1,0,0,XMaxGlb,YMaxGlb);

   SelectWindow(1);

  NumberOfPoints := TopBin - BottomBin;
  PCS := 668 - BottomBin;
   FOR I := 1 TO NumberOfPoints DO
     BEGIN
       a[I,1] := ScatRatio[PCS];
       a[I,2] := Alt[PCS];
       PCS := PCS - 1;
     END;

(******* FindWorld(1,a,NumberOfPoints,1,1);   *** Auto findworld **)
DefineWorld(1,0.40,StartingAlt,3.0,StoppingAlt);
(* remove for auto-world ***)

(** WITH World[1] DO
(*    BEGIN
(*      Temp := Y1;           (* Flip the Y co-ordinates *)
(*      Y1 := Y2;             (* Only needed for FindWorld *)
(*      Y2 := Temp;
(*    END;    **)

  SelectWorld(1);

  SelectWindow(1);

  SetLineStyle(0);

  DrawAxis(8,7,0,0,0,0,0,0,False);

  DrawPolygon(a,1,NumberOfPoints,7,2,0);

 END;  (* DrawTheGraph *)

PROCEDURE GetDiskAltTempPress;

VAR
 Filvar    :  File of REAL;
 FileName  :  STRING[12];
 Index     :  REAL;
 PCS       :  INTEGER;

BEGIN
 WRITELN('Please type the name of the disk file containing
          the altitude,');
 WRITELN('temperature and pressure data ');
```

```
 READLN(FileName);
 WRITE('Loading altitude, temperature and pressure from disk file '
       ,FileName);

 Assign(Filvar,FileName);
 Reset(Filvar);
 PCS := 0;

   WHILE NOT EOF (Filvar) DO
     BEGIN
       READ(Filvar,Index,Alt[PCS],Temp[PCS],Press[PCS]);
       IF PCS <> ROUND(Index) THEN
         BEGIN
           WRITELN('Data mismatch PCS = ',PCS);
           CLOSE(Filvar);
           EXIT;
         END;
       PCS := PCS + 1;
     END;

  CLOSE(Filvar);

END; (* GetDiskAltTempPress *)

PROCEDURE GetTempAlt;

CONST
  MAXDATA   = 667;
  BINHEIGHT = 150;

VAR
  PCS      :  INTEGER;
  BIN      :  INTEGER;
  Key      :  STRING[2];
  Altitude :  REAL;
  NUMBER   :  INTEGER;
  FileName :  STRING[12];
  Filvar   :  File of INTEGER;
  RealBIN  :  REAL;
  Temp     :  Array[0..MAXDATA] of INTEGER; (* Save as integers, *)
                                            (*process to reals with *)
                                            (*MAKETEMP.PAS *)
BEGIN
  ClrScr;
  FOR BIN := 1 TO MAXDATA DO
    BEGIN
      PCS := 668 - BIN;       (* This initialises the array Alt[PCS] *)
      RealBIN := BIN;         (* Likes it like this for some reason *)
      Alt[PCS] := 150*RealBIN+11;
    END;
```

```
  WRITELN('Altitude temperature and pressure input from (K)eyboard
        or (D)isk?');
    REPEAT
      READLN(Key);
    UNTIL (Key = 'K') OR (Key = 'D');

IF Key = 'D' THEN GetDiskAltTempPress

ELSE
  BEGIN
    (* First set arrays to zero *)
    FOR I := 0 to MAXDATA DO
      BEGIN
        Temp[I]  := 0;
        Press[I] := 0;
      END;

    (* Now input whatever data is available *)
    WRITE('Please type starting altitude in meters   ');
    READLN(Altitude);
    BIN := ROUND((Altitude-11)/150);
    PCS := 668-BIN; (* First range bin is in PCS 667 which counts *)
                    (* down *)
    WRITELN('Type temperature in kelvin, pressure in millibars,
             999 to exit');

  WHILE NUMBER <> 999 DO
    BEGIN
      Altitude := 150*BIN+11;
      WRITE('BIN  ',BIN,'  Altitude   ',Altitude,'  Temp   ');
      READLN(NUMBER);
      Temp[PCS] := NUMBER;
      BIN := BIN+1;
      PCS := 668-BIN;
    END;

    WRITELN('Please type a name for the disk file of this data');
    READLN(FileName);

    (* Now save the temperature  array to disk              *)
    (* so that the array index corresponds to that of the   *)
    (* counts in the LIDAR data file                        *)

    WRITELN('Saving ',FileName);

    Assign(Filvar,FileName);
    REWRITE(Filvar);  (* Open the file and delete any data *)
```

```
    FOR I := 0 TO MAXDATA DO
      BEGIN
        WRITE(Filvar,I,Temp[I]);
      END;

    CLOSE(Filvar);

    END;

END;  (* GetTempAlt *)

PROCEDURE ReadLidarData;

(* This procedure reads a basic file into memory at $3000:0000 *)

CONST
  MAX  =  1420;          (* Size of BASIC'S data files plus a few *)

VAR
  Filvar           :     File of byte;
  I                :     Integer;
  RawData          :     ARRAY[0..MAX] of byte ABSOLUTE $3000:$0000;
  RawDataFileName  :     STRING[15];

BEGIN
  WRITE('Please type the name of the file of raw data to be
        processed  ');
  READLN(RawDataFileName);
  WRITE('    Loading raw data.....');
  Assign(Filvar,RawDataFileName);
  Reset(Filvar);
  I := 0;

    WHILE NOT EOF (Filvar) DO
      BEGIN
        Read (Filvar,RawData[I]);
        I := I+1;
      END;

  CLOSE (Filvar);

  WRITELN('Stripping BASIC''''S Header');
    FOR I := 0 TO MAX DO
      BEGIN
        RawData[I] := RawData[I+7];
      END;

END;   (* ReadLidarData *)
```

```
PROCEDURE GetStartingInfo;

BEGIN
  ClrScr;
  WRITELN('What is the highest altitude to start calculating the
          scattering ratio?');
  WRITE('Type the height in meters   ');
  READLN(StartingAlt);
  TopBin := ROUND((StartingAlt-11) / 150);
  WRITELN('Down to what altitude should scattering ratios be
          calculated?');
  WRITE('Again type height in meters  ');
  READLN(StoppingAlt);
  BottomBin := ROUND((StoppingAlt-11) / 150);
END;

PROCEDURE SubtractNoise;

(* This averages the last 200 bins where no return is detected and *)
(* subtracts this from the rest of the data (last 8 skipped)       *)

CONST
  BinsToAverage  = 200;

VAR
  I        :  INTEGER;
  Sum      :  INTEGER;
  Average  :  INTEGER;

BEGIN
  Sum     := 0;
  Average := 0;
  FOR I   := 8 TO BinsToAverage + 8 DO
    BEGIN
      Sum     := Sum + Actual[I];
    END;

  Average := ROUND(Sum DIV 200);  (* Calculate average *)

  FOR I := 0 TO MAXDATA DO
    BEGIN
      Actual[I] := Actual[I] - Average;
      IF Actual[I] < 0 THEN Actual[I] := 0; (* Subtract noise *)
    END;

END;  (* SubtractNoise *)
```

```
PROCEDURE FindTopBinCount;

(* This finds the count to use for that in the highest bin by *)
(* averaging the count in this bin with that of "spread" bins on *)
(* either side      *)

CONST
  Spread = 5;

VAR
  I,N    :  INTEGER;
  Key    :  STRING[15];
(*Gives the global variable PCS the index for the data in "Actual[n]"*)

BEGIN
  N  := 0;

  FOR I := TopBin - Spread TO TopBin + Spread DO
    BEGIN
      PCS := 668 - I;    (* Array index of corresponding data *)
      N   := N + Actual[PCS];
    END;

  PCS := 668-TopBin; (*Init PCS to point to TpoBin data in the arrays*)
  TopBinCount := ROUND(N DIV (2 * Spread + 1));  (*Average the counts*)
  WRITELN('Top bin count is  ',TopBinCount);

END;  (* FindTopBinCount *)

PROCEDURE CalcMassDensity (Press,Temp : REAL);

(* Calculates mass density in the REAL array MassDensity *)

CONST
  MolMass = 0.02896; (* Mean molecular mass of air in kg per mol *)
  R       = 8.314;   (* Universal gas constant in joules per mol per *)
                     (* kelvin *)

VAR
  PressMKS    :  Real;

BEGIN
  PressMKS   :=  Press * 100;  (* Convert millibars to N/m2 *)
  MassDensity[PCS] := (PressMKS * MolMass )/(R * Temp );

END;  (* CalcMassDensity *)
```

```
PROCEDURE CalcK (Count,Height,MassDensity : REAL);

BEGIN
  K := Count * Height * Height / MassDensity ;
END;  (* CalcK *)

PROCEDURE CalcExtinction (MassDensity  :  REAL);

CONST
  MolMass = 0.02896;
  AvogadroNo = 6.02E+23;
  BinLength = 150;
  Sigma{589} = 3.4714E-31; (* Total scattering X section of an air   *)
                           (*  molecule in square meters at 589 nm   *)

VAR
  NumberDensity  :  REAL;
  Kappa          :  REAL;

BEGIN
  NumberDensity := MassDensity * AvogadroNo / MolMass;
  Kappa := NumberDensity * Sigma{589};
  Extinction := BinLength * Kappa;
  SumOfExtinctions := SumOfExtinctions + Extinction;

END; (* CalcExtinction *)

PROCEDURE CalcNewBeamEnergy (Extinction : REAL);

VAR
  ScatteredPortion   :  REAL;

BEGIN
  ScatteredPortion := Extinction * BeamEnergy;
  BeamEnergy := BeamEnergy + ScatteredPortion; (* Finds beam energy *)
                                               (* in next bin down  *)
END;  (* CalcNewBeamEnergy *)

PROCEDURE CalcExpectedReturn (MassDensity,Height : REAL);

VAR
  Atten    :  REAL;

BEGIN
  Atten := 1.0 - SumOfExtinctions;
  Rayleigh[PCS] := ROUND(MassDensity * K * BeamEnergy /(Height * Height
                         * Atten));

END; (* CalcExpectedReturn *)
```

```
PROCEDURE SaveResults; (* Writes results to disk *)

VAR
  Filvar          :      File of real;
  DiskFileName    :      String[12];
  NumberOfReals   :      Integer;
  I               :      Integer;

BEGIN
  ClrScr;
  WRITELN('This procedure saves the calculated altitude and scattering
           ratio.');
  WRITELN('to disk as reals in the order altitude then scattering
           ratio.');
  WRITELN('Please type a name for the file of data ');
  READLN(DiskFileName);
  Assign(Filvar,DiskFileName);
  WRITE('Saving  ',DiskFileName);
     Rewrite(Filvar); (* Open the file and delete any data *)

     PCS := 668-TopBin;
       WRITE(Filvar,Alt[PCS],StartingScatRatio); (*Do initial values*)

        FOR I := TopBin-1 DOWNTO BottomBin DO
          BEGIN
            PCS := 668-I;
            WRITE(Filvar,Alt[PCS],ScatRatio[PCS]);
          END;
  CLOSE(Filvar);

END; (* SaveReals *)

BEGIN (* Main Program *)
  GetTempAlt;
    REPEAT
        GetStartingInfo;  (* Zero relevant vars *****)
        ReadLidarData;
        SubtractNoise;
        FindTopBinCount;
        WRITELN('PCS = ',PCS);
        CalcMassDensity (Press[PCS],Temp[PCS]);
        CalcK (TopBinCount,Alt[PCS],MassDensity[PCS]);
        SumOfExtinctions  := 0;
        CalcExtinction  (MassDensity[PCS]);
        BeamEnergy  := 1.0;
        CalcNewBeamEnergy (Extinction);
```

```
        REPEAT
          WRITELN('Thinking........');

          FOR I := TopBin - 1 DOWNTO BottomBin DO
            BEGIN
              PCS := 668-I;                (* Index into the arrays *)
              CalcMassDensity              (Press[PCS],Temp[PCS]);
              CalcExpectedReturn           (MassDensity[PCS],Alt[PCS]);
              ScatRatio[PCS] := Actual[PCS]/Rayleigh[PCS];
              CalcExtinction               (MassDensity[PCS]);
              CalcNewBeamEnergy            (Extinction);

            END;

      WRITE('Bin':3,'Height':18,'Measured':18,'Rayleigh':18);
      WRITELN('Scattering ratio':18);
      WRITELN;
      PCS := 668-TopBin;
      StartingScatRatio := Actual[PCS] / TopBinCount ;
      WRITE(TopBin:3,Alt[PCS]:18:0,Actual[PCS]:18); (*This does the*)
      WRITELN(TopBinCount:18,StartingScatRatio:18:5);
                                                 (* Starting values *)

        FOR I := TopBin-1 DOWNTO BottomBin DO
          BEGIN
            PCS := 668-I;
            WRITE(I:3,Alt[PCS]:18:0,Actual[PCS]:18,Rayleigh[PCS]:18);
            WRITELN(ScatRatio[PCS]:18:5);
          END;

WRITELN;
WRITE('Bottom beam energy = ');
WRITELN(BeamEnergy);
WRITE('Extinction factor  = ');
WRITELN(1-SumOfExtinctions);
WRITELN('(R)epeat with another starting count,
         (S)ave the results,');
WRITELN('plot the (G)raph or (Q)uit ?');
 REPEAT
   READLN(Dummy);
 UNTIL (Dummy = 'R') OR (Dummy = 'Q') OR (Dummy = 'G')
       OR (Dummy = 'S');

    IF Dummy = 'R' THEN
      BEGIN
        WRITE('Old count was ',TopBinCount,'  Type new count  ');
        READLN(TopBinCount);
        PCS := 668- TopBin;
        CalcK(TopBinCount,Alt[PCS],MassDensity[PCS]);
```

```
            SumOfExtinctions  := 0;
           CalcExtinction  (MassDensity[PCS]);
            BeamEnergy  := 1.0;
            CalcNewBeamEnergy (Extinction);
         END;

       IF Dummy = 'G' THEN
         BEGIN
           InitGraphic;
           DrawTheGraph;
            REPEAT
              READLN(Dummy);
            UNTIL (Dummy = 'P') OR (Dummy = 'X') OR (Dummy = 'S');

              IF Dummy = 'P' THEN  PrntScrn;

           LeaveGraphic;
         END;    (* Dummy = 'G' *)

        IF Dummy = 'S' THEN  SaveResults;

          UNTIL Dummy <> 'R';

      UNTIL Dummy = 'Q';

END.
```

Appendix 5.2 SCATAVE.PAS

```
(*************************************************************************)
(*                                                                       *)
(*       TURBO PASCAL PROGRAM TO AVERAGE SEVERAL FILES OF                *)
(*                    ALTITUDE AND SCATTERING RATIO                      *)
(*                                by                                     *)
(*                             R. GRANT                                  *)
(*                                                                       *)
(*************************************************************************)

PROGRAM ScatAve;

{$I TYPEDEF.SYS}
{$I GRAPHIX.HGC}
{$I KERNEL.SYS}
{$I WINDOWS.SYS}
{$I FINDWRLD.HGH}
{$I AXIS.HGH}
{$I POLYGON.HGH}

(*This averages the values of scattering ratio contained in several  *)
(*files of altitude and scattering ratio. It computes and draws error*)
(* bars of +/- one sample standard deviation on the graph            *)

CONST
  MAXDATA      = 667;    (* Largest number of data points possible *)

VAR
  Dummy                 :  STRING[12];
  CurrentFileName       :  STRING[16];
  FileNumber            :  INTEGER;
  BIN                   :  INTEGER;    (* Pointer into the array *)
  NewData               :  Array[1..MAXDATA] of REAL;
  Sum                   :  Array[1..MAXDATA] of REAL;
  SumOfSquares          :  Array[1..MAXDATA] of REAL;
  AvScatRatio           :  Array[1..MAXDATA] of REAL;
  Altitude              :  Array[1..MAXDATA] of REAL;
  StdDeviation          :  Array[1..MAXDATA] of REAL;
  NoOfResults           :  Array[1..MAXDATA] of INTEGER;

PROCEDURE PrntScrn; (* Dumps the HI RES Hercules screen to the FX-80 *)

(* This procedure reads a basic file into memory at $3000:0000 *)
(* and then runs it as code *)

CONST
  MAX  =  340;          (* Size of BASIC'S PROCF.BAS file plus a few *)
```

```
VAR
  Filvar           :     File of byte;
  I                :     Integer;
  PrtScrnCode      :     ARRAY[0..MAX] of byte ABSOLUTE $3000:$0000;

BEGIN
  Assign(Filvar,'PROCF.BAS');
  Reset(Filvar);
  I := 0;

    WHILE NOT EOF (Filvar) DO
      BEGIN
        Read (Filvar,PrtScrnCode[I]);
        I := I+1;
      END;

  CLOSE (Filvar);

    FOR I := 0 TO MAX DO
      BEGIN
        PrtScrnCode[I] := PrtScrnCode[I+7];
      END;

  (* Now run the code *)
  INLINE($9A/>$0000/>$3000);

END;   (* PrntScrn *)

PROCEDURE DrawTheGraph; (* Draws a graph of scat. ratio vs altitude  *)
                        (* and puts in error bars of +/- one std. dev*)

VAR
  NoOfPoints       :  INTEGER;
  a                :  PlotArray;
  I                :  INTEGER;
  Temp             :  REAL;
  StartingAlt      :  REAL;
  StoppingAlt      :  REAL;
  Left             :  REAL;
  Right            :  REAL;
  BottomBin        :  INTEGER;
  TopBin           :  INTEGER;

 BEGIN
   InitGraphic;
   DefineWindow(1,0,0,XMaxGlb,YMaxGlb);
   SelectWindow(1);
```

```
  NoOfPoints := 0;
  StartingAlt := 0;
  StoppingAlt := 0;
   FOR BIN := 1 TO MAXDATA DO
     BEGIN
       IF NoOfResults[BIN] >0 THEN
         BEGIN
           IF StartingAlt = 0 THEN StartingAlt := Altitude[BIN];
           NoOfPoints      := NoOfPoints + 1;
           a[NoOfPoints,1] := AvScatRatio[BIN];
           a[NoOfPoints,2] := Altitude[BIN];
           StoppingAlt := Altitude[BIN];
         END;
     END;

(******* FindWorld(1,a,NumberOfPoints,1,1);   *** Auto findworld ****)
DefineWorld(1,0.60,StartingAlt,2.8,StoppingAlt);
(*** remove for auto-world ***)

    WITH World[1] DO
      BEGIN
        Temp := Y1;          (* Flip the Y co-ordinates *)
        Y1 := Y2;
        Y2 := Temp;
      END;

  SelectWorld(1);

  SetLineStyle(0);

  DrawAxis(8,7,0,0,0,0,0,0,False);

  DrawPolygon(a,1,-NoOfPoints,7,1,0);
  (* Now do the error bars *)
  BottomBin := ROUND((StartingAlt-11)/150);
  TopBin    := ROUND((StoppingAlt-11)/150);
    FOR BIN := BottomBin TO TopBin DO
      BEGIN
        Left := AvScatRatio[BIN]-StdDeviation[BIN];
        Right := AvScatRatio[BIN]+StdDeviation[BIN];
        a[1,1] := Left;
        a[1,2] := Altitude[BIN];
        a[2,1] := AvScatRatio[BIN];
        a[2,2] := Altitude[BIN];
        a[3,1] := Right;
        a[3,2] := Altitude[BIN];
        AxisGlb := TRUE; (* Keep same scaling as axes  12.27 am !!!! *)
        DrawPolygon(a,1,-3,0,1,0);
      END;
```

```
REPEAT
    READLN(Dummy);
  UNTIL (Dummy = 'P') OR (Dummy = 'X');

  IF Dummy = 'P' THEN PrntScrn;  (* Send to the FX-80 printer *)

  LeaveGraphic;

 END;  (* DrawTheGraph *)


PROCEDURE ClearArrays; (*Initialises altitude array and clears others*)

VAR
  RealBIN  :  REAL;

BEGIN
  FOR BIN := 1 TO MAXDATA DO
    BEGIN
      RealBIN           := BIN;
      Altitude[BIN]     := 150*RealBIN+11;   (* Bin length 150m *)
      Sum[BIN]          := 0;
      SumOfSquares[BIN] := 0;
      NoOfResults[BIN]  := 0;
    END;
  END; (* ClearArrays *)

PROCEDURE GetNameOfFile;

  BEGIN
    WRITELN('Please type the name of the file of altitude and
             scattering ');
    WRITE('ratio to be included ');
    READLN(CurrentFileName);
  END; (* GetNameOfFile *)

PROCEDURE GetData;

(* This procedure reads a file of altitude and scattering ratio *)
(* and updates the relevant arrays *)

VAR
  Filvar           :     File of real;
  Altitude         :     REAL;
  ScatteringRatio  :     REAL;

BEGIN
  WRITE('    Loading raw data.....');
  Assign(Filvar,CurrentFileName);
  Reset(Filvar);
```

```
    WHILE NOT EOF (Filvar) DO
      BEGIN
        Read (Filvar,Altitude,ScatteringRatio);
        BIN               := ROUND((Altitude-11)/150);
                               (* Find the right bin *)
        NewData[BIN]      := ScatteringRatio; (* and put the data in *)
        Sum[BIN]          := Sum[BIN] + NewData[BIN];
        SumOfSquares[BIN] := SumOfSquares[BIN]+SQR(NewData[BIN]);
        NoOfResults[BIN]  := NoOfResults[BIN]+1;
        AvScatRatio[BIN]  := Sum[BIN]/NoOfResults[BIN];
      END;

  CLOSE (Filvar);

END;   (* GetData *)

PROCEDURE CalcStdDeviations;

  BEGIN
    WRITELN('Calculating standard deviations ');
      FOR BIN := 1 TO MAXDATA DO
        BEGIN
          IF NoOfResults[BIN] > 1 THEN
          StdDeviation[BIN] := SQRT((SumOfSquares[BIN]-SQR(Sum[BIN])/
                               NoOfResults[BIN])/(NoOfResults[BIN]-1));
        END;
  END;  (*CalcStdDeviations *)

PROCEDURE ShowArrays; (* Sends some arrays to the screen *)

  BEGIN
    WRITE('Bin':3,'Altitude':18,'Scattering ratio':18,
          'No of results':18);
    WRITELN('Standard Deviation':18);
      FOR BIN := 1 TO MAXDATA DO
        BEGIN
          IF NoOfResults[BIN] >0 THEN
            BEGIN
              WRITE(BIN:3,Altitude[BIN]:18:0,AvScatRatio[BIN]:18:6);
              WRITELN(NoOfResults[BIN]:18,StdDeviation[BIN]:18:6);
            END;
        END;
  END;  (* ShowArrays *)
```

```
PROCEDURE SaveArrays; (* Writes results to disk *)

VAR
  Filvar          :      File of real;
  DiskFileName    :      String[12];

BEGIN
  ClrScr;
  WRITELN('This procedure saves the calculated altitudes, scat. rats');
  WRITELN('and std. devs. to disk as reals in the order altitude');
  WRITELN('then scattering ratio then standard deviation.');
  WRITELN('Please type a name for the file of data ');
  READLN(DiskFileName);
  Assign(Filvar,DiskFileName);
  WRITE('Saving  ',DiskFileName);
     Rewrite(Filvar); (* Open the file and delete any data *)

        FOR BIN := 1 to MAXDATA DO
          BEGIN
            IF NoOfResults[BIN] > 0 THEN
            WRITE(Filvar,Altitude[BIN],AvScatRatio[BIN],
                  StdDeviation[BIN]);
          END;
  CLOSE(Filvar);

END; (* SaveArrays *)

PROCEDURE UpdateAndPrintArrays;

  BEGIN
    CalcStdDeviations;
    ShowArrays;
  END; (* UpdateAndPrintArrays *)

BEGIN (* Main program *)
  ClearArrays;
    REPEAT
      GetNameOfFile;
      GetData;
        REPEAT;
          WRITELN('(A)nother file, (V)iew arrays, (G)raph,
                   (S)ave results or (Q)uit');
            REPEAT
              READLN(Dummy);
            UNTIL (Dummy = 'A') OR (Dummy = 'V') OR (Dummy = 'G')
            OR (Dummy = 'S') OR (Dummy = 'Q');
```

```
          CASE Dummy OF
            'V'   :   UpdateAndPrintArrays;
            'G'   :   DrawTheGraph;
            'S'   :   SaveArrays;
          END; (* CASE *)
        UNTIL (Dummy ='A') OR (Dummy = 'Q');

    UNTIL Dummy = 'Q'

END. (* Main program *)
```

LITERATURE CITED

Alley, A. and K.W. Atwood. Electronic Engineering (Third Edition). J. Wiley and Sons, N.Y. 1973.

Benedetti-Michelangeli, G., F. Congeduti and G. Fiocco (1972) Measurement of Aerosol Motion and Wind Velocity in the Lower Troposphere by Doppler Radar. J. Atmos. Sci. 29: 906-910.

Bowman, M.R., A.J. Gibson and C.W. Sandford (1969) Atmospheric Sodium Measured by a Tuned Laser Radar. Nature 221: 456-457.

Brock, E.G., P.Czavinsky, E.Hormats, H.C. Nedderman, D. Stirpe and F. Unterleitner (1961) Coherent Stimulated Emission from Molecular Crystals. J. Chem. Phys. 35: 759-760.

Calkins, J., E. Colley and J. Hazle (1982) The use of Caffeine as a Liquid Filter in Coaxial Flashlamp Pumped Dye Lasers. Opt. Comm. 42: 275-277.

Chanin, M. L. and A. Hauchecorne (1984) Lidar Studies of Temperature and Density using Rayleigh Scattering. MAP Handbook 13: 87-98.

Cook, C.S., G.W. Bethke and William D. Conner (1972) Remote Measurement of Smoke Plume Transmittance Using LIDAR, Applied Optics. 11: 1742-1748.

Eberhard, W.L. and R.M. Schotland (1980) Dual-frequency Doppler-lidar method of wind measurement. Applied Optics 19: 2967-2976.

Felix, F., W. Keenliside, G.S. Kent and M.C.W. Sandford (1973) Laser Radar Observations of Atmospheric Potassium. Nature. 346: 345.

Fiocco, G., G. Benedetti-Michelangeli, K. Maischberger and L. Madonna (1971) Measurement of Temperature and Aerosol to Molecule Ratio in the Troposphere by Optical Radar. Nature. 229: 78-79.

Fiocco, G. and G. Grams (1964) Observations of the Aerosol Layer at 20 km by Optical Radar. J. Atmos. Sci. 21: 323-324.

Fiocco, G. and L.D. Smullin (1963) Detection of Scattering Layers in the Upper Atmosphere (60-140 km) by Optical Radar. Nature. 199: 1275-1276.

Foord R., R. Jones, C.J. Oliver, and E.R. Pike (1969) The Use of Photomultiplier tubes for Photon Counting, Applied Optics 8: 1975-1989.

Friedland, S.S., J. Katzenstein and M.M. Zatzick (1956) Pulsed Searchlighting the Atmosphere, J. Geophys. Res. 61: 415.

Furumoto H. and H. Ceccon (1968) Time Dependent Spectroscopy of Flashlamp Pumped Dye Lasers. Appl. Phys. Lett. 13: 335-337.

Gibson A.J., (1969) A Flashlamp-pumped Dye Laser for Resonance Scattering Studies of the Upper Atmosphere. J. Sci. Instr. 2: 802-806.

Granier, C., J.P. Jegou and G. Megie (1985) Resonant Lidar Detection of Ca and Ca^+ in the Upper Atmosphere, Geophys. Res. Lett. 12: 655-658.

Granier, C. and G. Megie (1982) Daytime Lidar Measurements of the Mesospheric Sodium Layer. Planet. Space Sci. 30: 169-177.

Hall, F.F. Jr. Laser Applications. Academic Press. N.Y. 1974.

Hamilton, P.M. (1966) The Use of LIDAR in Air Pollution Studies. Air and Wat. Pollut. Int. J. 10: 427-434.

Hauchecorne, A. and M. Chanin (1980) Density and Temperature Profiles Obtained By Lidar Between 35 and 70 km. Geophys. Res. Lett. 7: 565-568.

Hirono, M., M. Fujiwara, T. Shibata and N. Kugumiya (1984) Lidar Observations of Atmospheric Aerosols Following the 1980 Eruption of Mt. St. Helens. Part 1. J. Atmos. Terr. Phys. 46: 1147-1157.

Horman, M.H. (1961) J. Opt. Soc. Amer. 51: 681-691.

Hulburt, E.O. (1937) Observations of a Searchlight Beam to an Altitude of 28 km. J. Opt. Soc. Amer. 27: 337-382.

Hunten,D.M. (1954) A Study of Sodium in Twilight. J. Atmos. Terr. Phys. 5: 44-56.

Jakeman, E., C.J. Oliver and E.R. Pike (1968) J. Phys. A. 2: 497.

Jegou, J., M. Chanin, G. Megie, J.E. Blamont (1980) Lidar Measurements of Atmospheric Lithium. Geophys. Res. Lett. 7: 995-998.

Junge, Chr., D. C. Chagnon and J. Manson, (1961) Stratospheric aerosols, J. Meteorol. 18: 81-107.

Kent, G.S., B.R. Clemesha and R.W. Wright (1967) High Altitude Atmospheric Scattering of Light from a Laser Beam. J. Atmos. Terr. Phys. 29: 169-181.

Kerker, M. and D.D. Cooke (1976) Remote Sensing of Particle Size and Refractive Index by varying the Wavelength. Applied Optics. 15: 2105-2111.

Measures, R.M. Laser Remote Sensing. John Wiley and Sons, N.Y. 1984.

Megie, G., J.Y. Allain, M.L. Chanin and J.E. Blamont (1977) Vertical Profile of Stratospheric Ozone by Lidar Sounding from the Ground. Nature. 270: 329.

Meinel, A. and M. Meinel. Sunsets, Twilights and Evening Skies. Cambridge University Press. Cambridge. 1983.

McClung, F.J. and R.W. Hellwarth (1962) Giant Optical Pulsations from Ruby, J. Appl. Phys. 33 828-829.

McNeil, W.R., and A.I. Carswell (1975) Lidar Polarisation Studies of the Troposphere. Applied Optics. 14: 2158-2168.

Northam, G.B., J.M. Rosen, S.H. Melfi, T.J Pepin, M.P. McCormick, D.J. Hofmann and W.H. Fuller Jr. (1974) Dustsonde and Lidar Measurements of Stratospheric Aerosols: a Comparison. Applied Optics. 13: 2416-2421.

Poultney, S.K. (1972 a) Laser Radar Studies of Upper Atmosphere Dust Layers and the Relation of Temporary Increases in Dust to Cometary Micrometeoroid Streams, Space Research XII, Akademie-Verlag. Berlin 403-421.

Poultney, S.K. (1972 b) Single Photon Detection and Timing: Experiments and Techniques. Advances in Electronics and Electron Physics. 31: 39-117.

Sandford, M.C.W. (1967) Laser Scatter Measurements in the Mesosphere and Above. J. Atmos. Terr. Phys. 29: 1657-1662.

Schotland, R.M. (1964) The Determination of the Vertical Profile of Atmospheric Gases by means of a Ground Based Optical Radar, in Proceedings of the Third Symposium on Remote Sensing of the Environment. University of Michigan.

Schuster, B.G. (1970) Detection of Tropospheric and Stratospheric Aerosol Layers by Optical Radar (Lidar) J. Geophys. Res. 75: 3123-3132.

Shibata, T., M. Fujiwara and M. Hirono (1984) The El Chichon Volcanic Cloud in the Stratosphere: Lidar Observation at Fukuoka and Numerical Simulation. J. Atmos. Terr. Phys. 46: 1121-1146.

Snavely B.B., (1969) Flashlamp-Excited Organic Dye Lasers. Proc. IEEE. 57: 1374-1390.

Soffer B.H. and B.B. McFarland (1967) Continuously Tunable Narrow-band Organic Dye Lasers. Appl. Phys. Lett. 10: 266-267.

Sorokin P.P. (1969) Organic Lasers. Sci. Am. 220: 30-40.

Sorokin P.P. and J.R. Lankard (1966) Stimulated Emission Observed from an Organic Dye, Chloroaluminium Phthalocyanine. IBM J. Res. Develop. 10: 162-163.

Sorokin P.P. and J.R. Lankard (1967) Flashlamp Excitation of Organic Dye Lasers: a Short Communication. IBM J. Res. Develop. 11: 148.

Stockman D.L., W.R. Mallory and K.F. Tittel (1964) Stimulated Emission in Aromatic Organic Compounds.Proc. IEEE (Correspondence) 52: 318-319.