Computer control of an HF chirp radar

THESIS

Submitted in fulfilment of the

requirements for the degree of

MASTER OF SCIENCE (ELECTRONICS)

of Rhodes University

by

DESMOND BRYAN GRIGGS

December 1990

(Revised March 1991)

Abstract

This thesis describes the interfacing of an IBM compatible microcomputer to a BR Communications chirp sounder. The need for this is twofold: Firstly for *control* of the sounder including automatic scheduling of operations, and secondly for *data capture*.

A signal processing card inside the computer performs a Fast Fourier Transform on the sampled data from two phase matched receivers. The transformed data is then transferred to the host computer for further processing, display and storage on hard disk or magnetic tape, all in *real time*.

Critical timing functions are provided by another card in the microcomputer, the timing controller. Built by the author, the design and operation of this sub-system is discussed in detail. Additional circuitry is required to perform antenna and filter switching, and a possible design thereof is also presented by the author.

The completed system, comprising the chirp sounder, the PC environment, and the signal switching circuitry, has a dual purpose. It can operate as either a *meteor radar*, using a fixed frequency (currently 27,99 MHz), or as an *advanced chirp ionosonde* allowing frequency sweeps from 1,6 to 30 MHz. In the latter case fixed frequency doppler soundings are also possible. Examples of data recorded in the various modes are given.

Declaration

I declare the contents of this thesis to be my own unaided work, with the exception of the following software routines:

TMS320 FFT – Delanco Spry

DSP16 assembler driver routines – Ariel

TMS320 ionosonde routines – BT Bonnevie

Pascal DCS and Tape driver – BT Bonnevie

In addition all the processed meteor data results presented here were produced by Prof G Poole.

This thesis is being submitted for the degree of Master of Science in Electronics at Rhodes University, Grahamstown. It has not been submitted before for any degree or examination at any other university.

DB Griggs

689G4433

On this the 31'st day of December 1990

Acknowledgements

Firstly thanks must go to my supervisor Dr Allon Poole for proposing and funding the project.

Secondly to my other supervisor Prof Graham Poole for the use of his equipment and for tolerating my interference with his meteor research programme.

To Mr Andries Delport and Mr Ronnie Seeber of Mikomtek, for the use of their sounder and signal processing card, and general interest expressed by them in the project.

To all the staff in the Physics department and Electronic Services at Rhodes, particularly the late Mr Barry Guthrie, for their advice and expertise.

Prof Pat Terry from the Computer Science department for his assistance during my software evaluation.

Mr Andy Soper for the loan of his PCMOS operating system for evaluation purposes.

Finally many thanks to Mr Bo Bonnevie for his much needed advice and encouragement.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

An ionosonde is a piece of equipment that is used to probe the upper atmosphere and is a type of variable frequency radar. Ionisation of the neutral gas in the ionosphere, which extends from approximately 60 km upwards, affords this region the characteristic property of refracting and reflecting radio waves propagating through it. Long distance shortwave radio communications is possible as a result of this, and the reliability of these communications was originally the main driving force behind investigation of the ionosphere.

Traditionally ionosondes measure only the *virtual height* of the reflecting layer at different frequencies, and from this electron density profiles can be calculated. Two broad types of ionosondes are in use. The first uses pulsed sounding techniques, and the second uses frequency modulated continuous wave (FMCW) or 'chirp' sounding. The ionosondes under consideration here fall into the second category [Poole AWV 1983, Evans GP 1984].

1.1 Ionospheric research

In the absence of an accurate model of the ionosphere, predictions of optimum frequencies for radio transmissions have been based on the large data-base collected mainly over the last 50 to 60 years. Modern investigation of the ionosphere requires parameters other than just the virtual height. Digital or advanced sounders have been developed over approximately the past 20 years (less for advanced chirpsounders). Parameters which can be measured with such a sounder include Doppler velocity of the reflecting layer, angle of arrival of the received signal, determination of polarisation mode, and vernier group height

from phase information. It also allows for quantitative evaluation of the amplitude, as well as, numerical rather than 'graphical' recording of the data. This requires a special type of segmented frequency sweep, as well as continual switching between pairs of spatially separated antennas. To achieve this, advanced sounders are under microprocessor control, and digital data are stored on magnetic storage media as opposed to the traditional recording of ionograms on photographic film [Evans GP 1984, Reinisch BW 1986]. The basic principles of chirpsounder operation are outlined in Appendix A, and an explanation is given of the terms *cell*, *sounding* and *ionogram*.

Of particular interest to the Hermann Ohlthaver Institute for Aeronomy is the ability to perform oblique digital sounding, using two chirpsounder systems some distance apart. This requires precise timing synchronization of the two systems.

Ionisation due to solar radiation and other sources may produce several distinct layers in the ionosphere which vary according to many factors such as time of day, season, solar cycle and unusual solar activity. The maximum plasma frequency of a given layer is called the layer's critical frequency and its maximum for the ionosphere is called the penetration frequency (see Figure [A-1]). Above this frequency vertically incident radio waves are not reflected, though they may be scattered [URSI 1978]. The penetration frequency for the region of the ionosphere directly above Grahamstown rarely exceeds 15 MHz.

1.2 Meteor Research

Ionisation of the ionosphere, though due mainly to incident solar energy, may also be caused by *meteoroids* as they 'burn up' on entry into the earth's atmosphere. The resulting phenomenon, which is marked by a visible streak of light and an ionisation trail along its path, is known as a *meteor* [McKinley DWR 1961].

Ionisation due to meteors is localised and extremely intense, though short lived, and can reflect frequencies well above the penetration frequency of the ionosphere. The chirpsounder at a suitably high fixed frequency (in this instance 27,99 MHz), provides an accurate and efficient facility for logging meteors which, for the most part, is independent of weather conditions and time of day.

Maximum heating of the meteoroid, and thus maximum ionisation, occurs at a height of approximately 100 to 110 km. See Appendix B for a background to the meteor pro-

gramme.

1.3 Project requirements

At the time this project was started the Hermann Ohlthaver Institute for Aeronomy had in its possession three chirp sounders, one with no digital capabilities, one advanced chirpsounder, and the newest of the three possessing potential digital capabilities but lacking external command and timing control. The latter sounder (a BR Communications 9034) was at the time being used solely for meteor research and it was decided to complete this system, to allow for both meteor and ionosonde operation, by linking it to an IBM AT compatible PC for control purposes. It is this project with which this thesis concerns itself.

The following initial requirements were of primary concern:

1. Upgrading the existing meteor data logging routine to allow for validation of echoes in 'real time' and compression of the stored data.

2. Connection of an IBM AT for system control allowing for specification of sweep structure, antenna selection and real time data processing, as well as automatic scheduling of operations to be run.

3. Development of the controller software and user interface in a suitable 'high level' language.

4. Design and construction of a timing controller, allowing for accurate timing adjustment, even during the execution of system operations (particularly important for oblique ionosonde operation as allowance has to be made for synchronization of two independent sounders).

5. The design and construction of antenna switching circuitry.

6. Design of compatible software for ultimate inclusion of the data capture system (DCS) developed by BT Bonnevie.

7. Inclusion of a device driver for a future film recording system (FRS).

8. Implementation of the above with the minimum of interference with the existing meteor and ionosonde programmes.

9. Finally, the overall system design was to take into account the existing hardware, structure and requirements of the older two ionosondes, so as to maintain compatibility between all three as far as possible. This would assist in the future rationalization of all three systems.

Chapter 2

Controller requirements

2.1 Existing equipment

At the time of commencement of this project the institute had in its possession three Barry Research / BR Communications 'chirp' ionosondes. The VIS-1, the eldest of the three, having been purchased around 1971.

2.1.1 VIS-1

The VIS-1 (Vertical Incidence Sounder) has over the past twenty years undergone minimal modification and consists of two 19″ racks containing the following units:

Logchirp Control (1015) – This performs the overall control functions. These include setting of the vertical and oblique sweep rates, sweep limits, sweep formats (log or linear) in vertical mode, frequency mark generation and sounding interval. Ionogram records may be initiated manually, automatically at 5 minute intervals, or remotely. All options are selectable via front panel controls.

Frequency synthesizer (5006) – To produce the vertical incidence mode linear and logarithmic sweep rates the Logchirp Control unit programs the synthesizer through 1/2 sec (25 kHz) segments of a 50 kHz/s sweep. This *basic rate* determines the relationship between the sounder height range (1000 km) and the sounder receiver bandwidth (500 Hz). See Appendix A for a detailed discussion of chirpsounder principles and operation. The two oblique incidence sweep rates, 50 kHz/s and 100 kHz/s, are continuous linear frequency ramps.

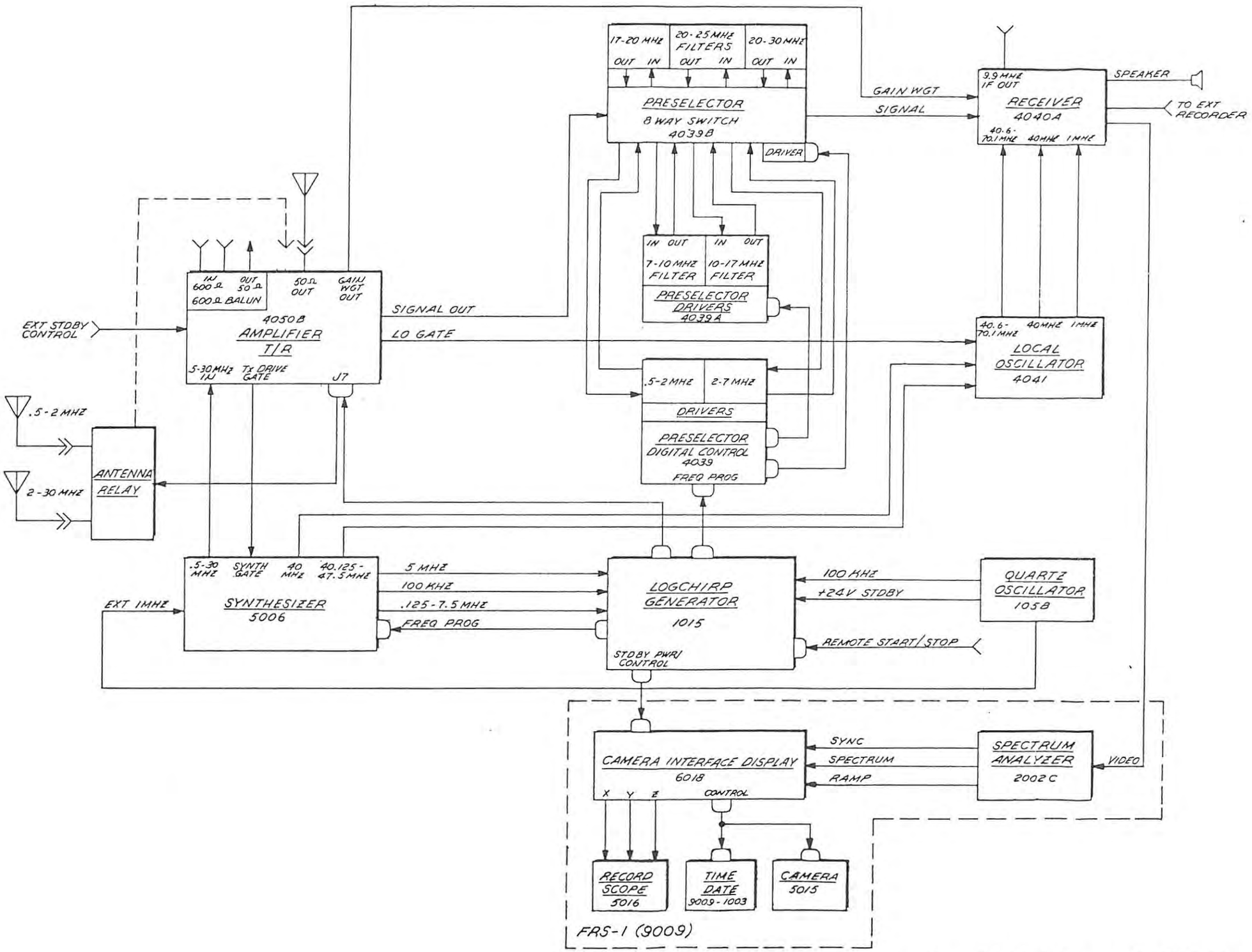


Figure 2-1: Block diagram of the VIS-1 chirp sounder [Barry Research 1972]

Amplifier-T/R (4050B) – Signals between 0.5 and 30 MHz generated by the synthesizer are amplified to approximately 8 W (peak) by the amplifier-T/R unit in vertical or oblique transmit mode. A T/R switch and its associated timing circuitry enables the use of a single antenna for both transmitting and receiving. The T/R rate used was a 'warbled' function which partially eliminated blind height ranges but produced considerable interference with other HF equipment. This has since been replaced by an M-sequence function.

Preselectors (4039) – The received RF signal originally passed through one of three preselectors, which consisted of digitally tuned filters. As these filters were switched by mechanical relays, they proved unreliable for the limited increase in performance and have since been removed from circuit.

Receiver (4040A) – At the input of the receiver a variable gain amplifier (under the control of the amplifier-T/R) is used to gain weight the received signals in vertical incidence mode. The receiver baseband output is in the frequency range 0–500 Hz and an audio monitor is also provided.

Local oscillator (4041) – Generates all the necessary mixer frequencies.

Spectrum analyser (2002) – Produces a frequency spectrum for film records.

Film recording system (FRS-1) – This includes the Camera interface (6018), monitor and record oscilloscopes, and camera, and produces a 35 mm film record of each ionogram. Record identification was originally achieved with a mechanical time/date unit but this has been replaced by an electronic, microprocessor controlled T/D unit.

Quartz oscillator (HP 105B) – A stable, temperature controlled, battery backed up, quartz oscillator provides the necessary 1 MHz and 100 kHz signals to the system. There is an additional unused 5 MHz output.

2.1.2 VOS-1

This sounder, being only a year younger than its predecessor, had the same basic units. The only marked difference between the two was the inclusion of circuitry in the camera interface to display the time and date on the two oscilloscope screens, thus doing away with the need for a separate time/date unit.

However, in the early 1980's the VOS-1 underwent extensive modifications, with the replacement of Logchirp Control by a microprocessor based controller. In addition the single receiver was replaced by two phase matched receivers, and an array of antenna switching circuitry was added. A RF power amplifier was also used with the system for the first time.

The result of these modifications was the ability to perform complex structured frequency sweeps, and the digital storage of data on magnetic tape. This was achieved through the use of a hardware FFT box [Fisher JS 1979] and a 6800 based data capture system (DCS) [Poole AWV 1983]. These latter two units have since been replaced and it suffices to say here that the principle of operation of the new DCS remains the same. The VOS-1 thus includes the following new units:

Vertichirp controller – This replaces Logchirp Control, but the two are interchangeble with only a few minor modifications. The Vertichirp controller is based on a South West Techniques Corporation 6809 microprocessor system. In addition to the processor board, two 32 k memory boards containing both EPROM and RAM, and a calculator board are used. Communication with the controller is via an RS-232 link from a dumb terminal. The majority of the software is written in the language Forth, and stored in EPROM.

All system timing circuitry and RAM are on battery backup. System functions (operations) such as *ionograms* and *timing* shifts can be defined completely from the dumb terminal, and scheduled to run at any time.

Phase matched receivers – A pair of phase matched receivers are required for phase comparisons which allows for the measurement of such parameters as angle of arrival. Remote control of the AGC is possible with the receiver gain being reported.

RF power amplifier (T500A) – Maximum of 70W in, into 50Ω. Nominal 10 dB gain.

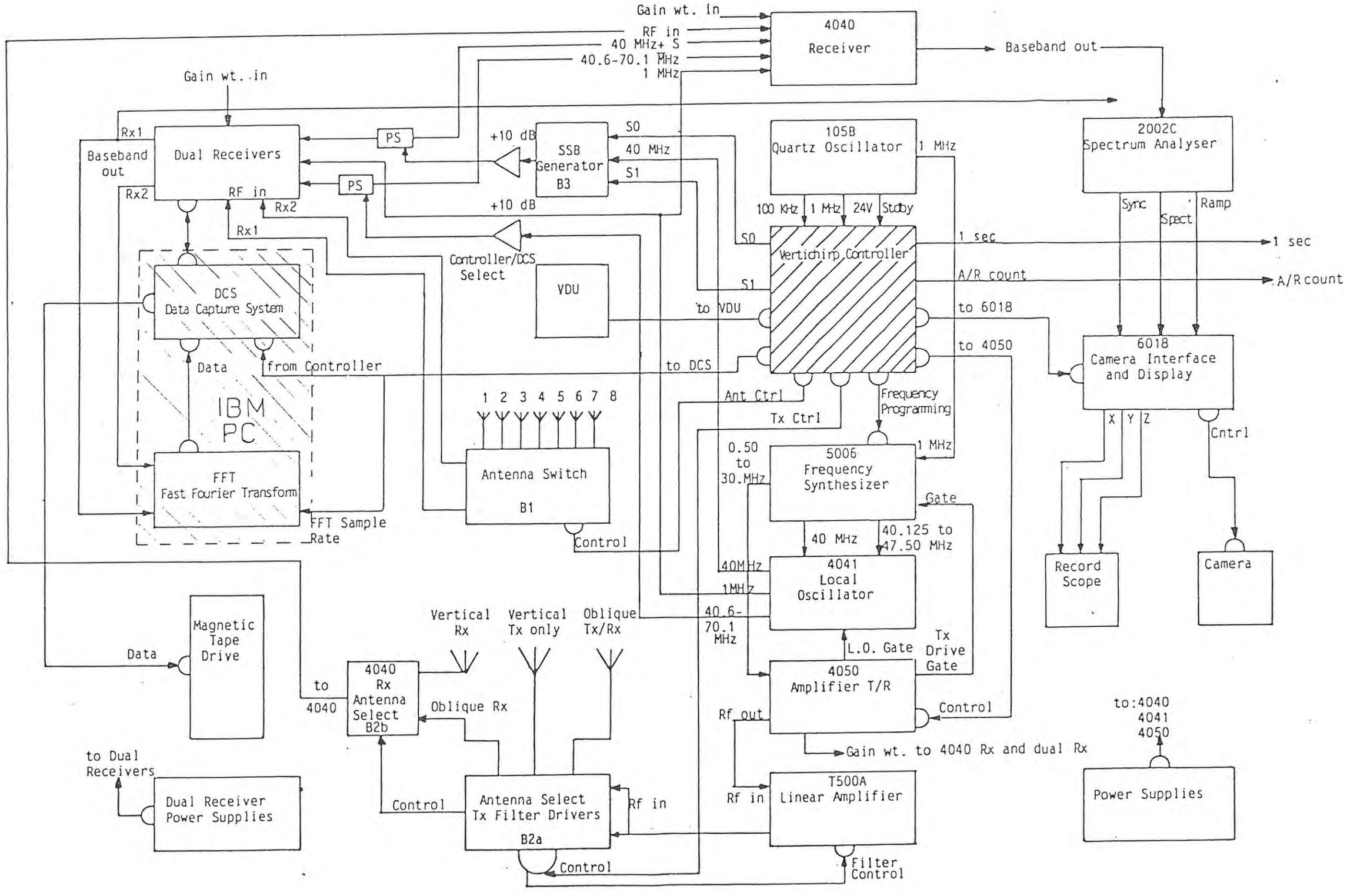


Figure 2-2: Block diagram of the VOS-1 chirp sounder system [Evans GP 1984]

Remote switching of filters is required as well as power blanking.

Antenna switching – The transmit RF power out can be switched to one of two antennas, generally either a vertical or oblique antenna, at the start of an ionogram. Mechanical relays are used and 100 W can be switched.

The two receivers can be switched to any pair of a possible eight receive antennas (four LF and four HF). All unused antenna inputs are grounded. Switching is again performed via mechanical relays, and while this has the advantage that the RF signal is not attenuated, the relays have a finite life expectancy, and a fault may not be readily identifiable. Remember that the receive antennas may be required to switch every cell.

Digital signal processor – A Delanco Spry, TMS32010 based signal processing card is used to perform the real time transformations from the time to the frequency domain, as well as computation of the power spectrum. The power spectrum is then indexed in descending order of magnitude to a preselectable number of points per cell (PPC), scaled to retain all significant bits, and the processed data transferred to the PC. The card has an onboard analog to digital converter (ADC) and digital to analog converter (DAC) which are not used.

IBM PC – An IBM PC compatible is used to perform all the data capture functions. It contains two 12 bit ADC cards (PC-26), each connected to a receiver output, and these cards are strobed by the external sample strobe. After every sample one of these cards interrupts the PC, which then services the two cards and writes the samples to an appropriate area in memory. At the end of a cell the processed data are retrieved from the signal processing card and the raw data down-loaded to the card. After further processing of the transformed data, it is plotted to a graphics screen and if required, written to magnetic tape. All necessary information as regards the ionogram, such as cell length, sample rate, etc, is obtained from the Vertichirp Controller via a link through a parallel port.

Magnetic tape drive – Data can be stored to magnetic tape. The present tape format used is 1/2 inch, 9 track, PE, 1600 bpi.

2.1.3 BR-9034

The 9034 was purchased in 1985 and consists of three units:

Transmit exciter (1070) – This emits a continuous wave signal at a fixed frequency or is swept in an upward linear ramp starting at any frequency in the range 1.6 to 30 MHz. Four basic rates are available; 25, 50, 100 and 200 kHz/s. The transmit frequency can be offset from the receive frequency by up to ±10 kHz in 1 Hz steps. The transmit exciter also contains an RF power detector to warn of RFPA failure.

Dual channel receiver (4070) – This contains two phase matched receivers, the receive frequency synthesizer, 5 MHz oscillator, and a microprocessor for control purposes. All system operating parameters such as tuned frequency, sweep rate, bandwidth, gain, etc. are digitally controlled and remotely programmed via an RS-232C serial data port. The internal 5 MHz signal may be replaced by an external source via rear panel BNC J22. Start and jump strobing is possible, via rear panel connectors J10 to J14.

Power amplifier (5018) – Nominal 100 W is fed back to the transmit exciter where it passes through a final T/R switch ('dark' switch) before being fed to the antenna. Input to the RFPA is 0 dBm and a 10 W output is available as well. All inputs and outputs are matched to 50Ω.

Control of the 9034 system was, at the start of this project, achieved through the use of a dumb terminal for programming and a push button switch to provide the start strobe. This is adequate for continuous, fixed frequency operation as required for meteor research, but operation as an ionosonde is not at all possible.

The T/R switching function was interchangeable between a 500 Hz square wave and an M-sequence, via the reversal of a jumper on the interface board in the dual channel receiver. For meteor research the 500 Hz square wave option is used, but ionosonde operation requires the M-sequence.

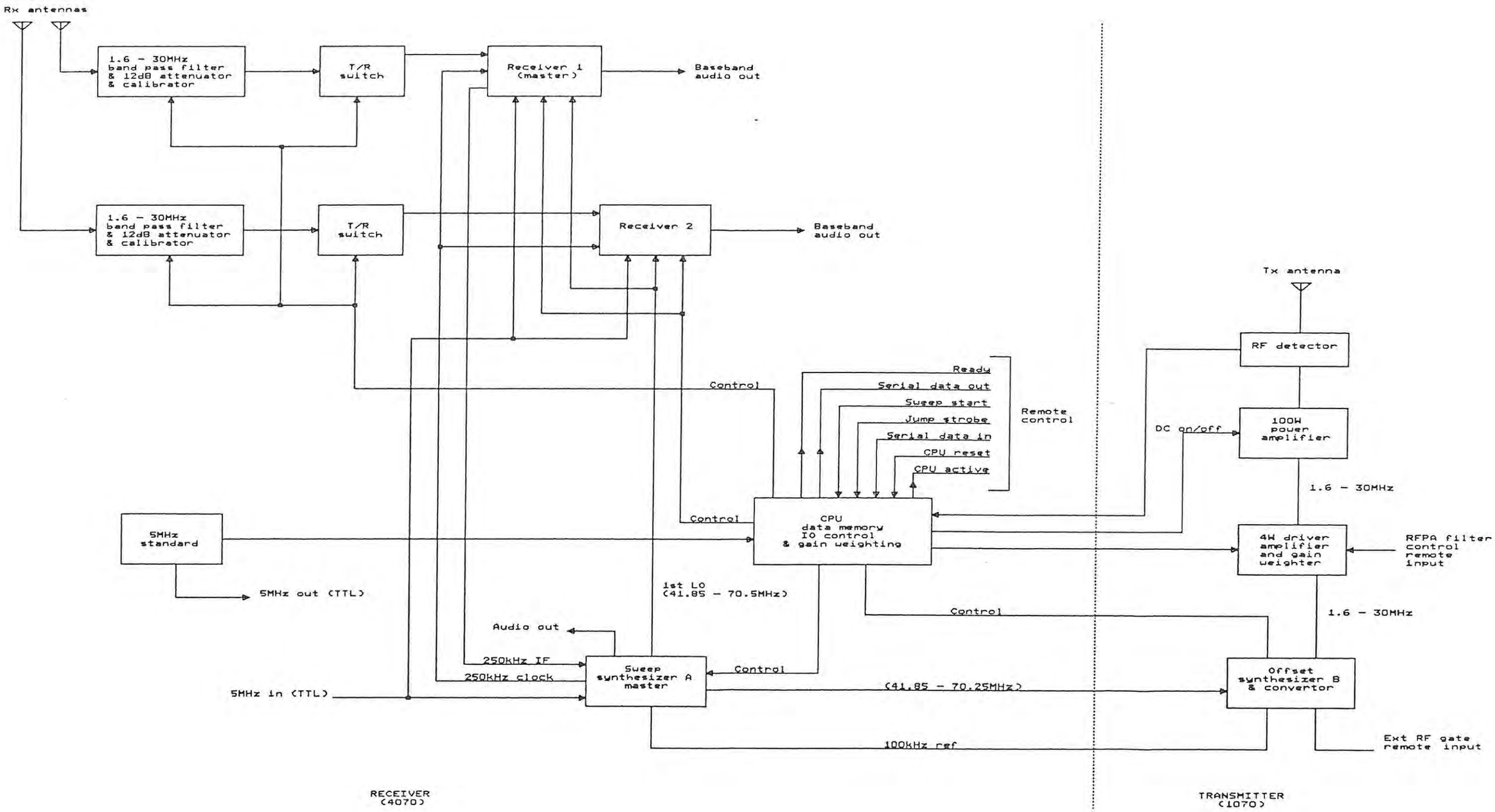


Figure 2-3: Block diagram of the BR-9034 chirp sounder [BR Communications 1984]

2.2 Upgrading the BR-9034

It was decided to use an IBM AT compatible to perform the necessary control functions, as well as simultaneously doubling as the data capture system. This imposed certain speed (time) requirements on the system which initially the author was unsure could be met.

An Ariel DSP-16 digital signal processing card is used to perform the 'front end' data capture. It contains two analog to digital converters and the sampled data is transformed to the frequency domain. After some additional processing the resultant data are uploaded to the host computer for display and storage.

The timing controller (TC) provides all timing and strobe inputs to the ionosonde. It was decided to develop this as a plug in card for the IBM PC[1] so that accurate timing control of the PC could be achieved as well. This also allows rapid data transfer to and from the card. The TC requires battery backup so as not to lose system timing during power failures. In addition the TC has to allow for system timing to be advanced and retarded by small amounts, even while the sounder is sweeping.

It was decided to use a Real Time Clock chip (RTC) to provide time and date information to the PC. The reason the PC's RTC was not used is simply the necessity for the clock to remain in phase with the system frequency standard. The use of a RTC chip substantially reduced the amount of circuitry that would have otherwise been required to implement this function. The only drawback is that the clock input frequencies to these devices are not readily obtainable from a 5 MHz signal.

One forced modification to the 9034 was that to enable remote selection of the T/R waveform (500 Hz square wave or M-sequence) required for operation as both an ionosonde and meteor radar. A small additional amount of circuitry was added to the interface card in the dual channel receiver, and a control line added to the rear panel connector J10 (see Appendix E).

An additional standard 19″ rack module, the Signal Switching unit (SS), is required and contains all the antenna, RF, baseband, and filter switching circuitry. The signal path followed varies depending on whether the sounder is operating as an ionosonde or

[1]It will in future be taken as understood that PC refers to an IBM AT compatible unless otherwise stated.

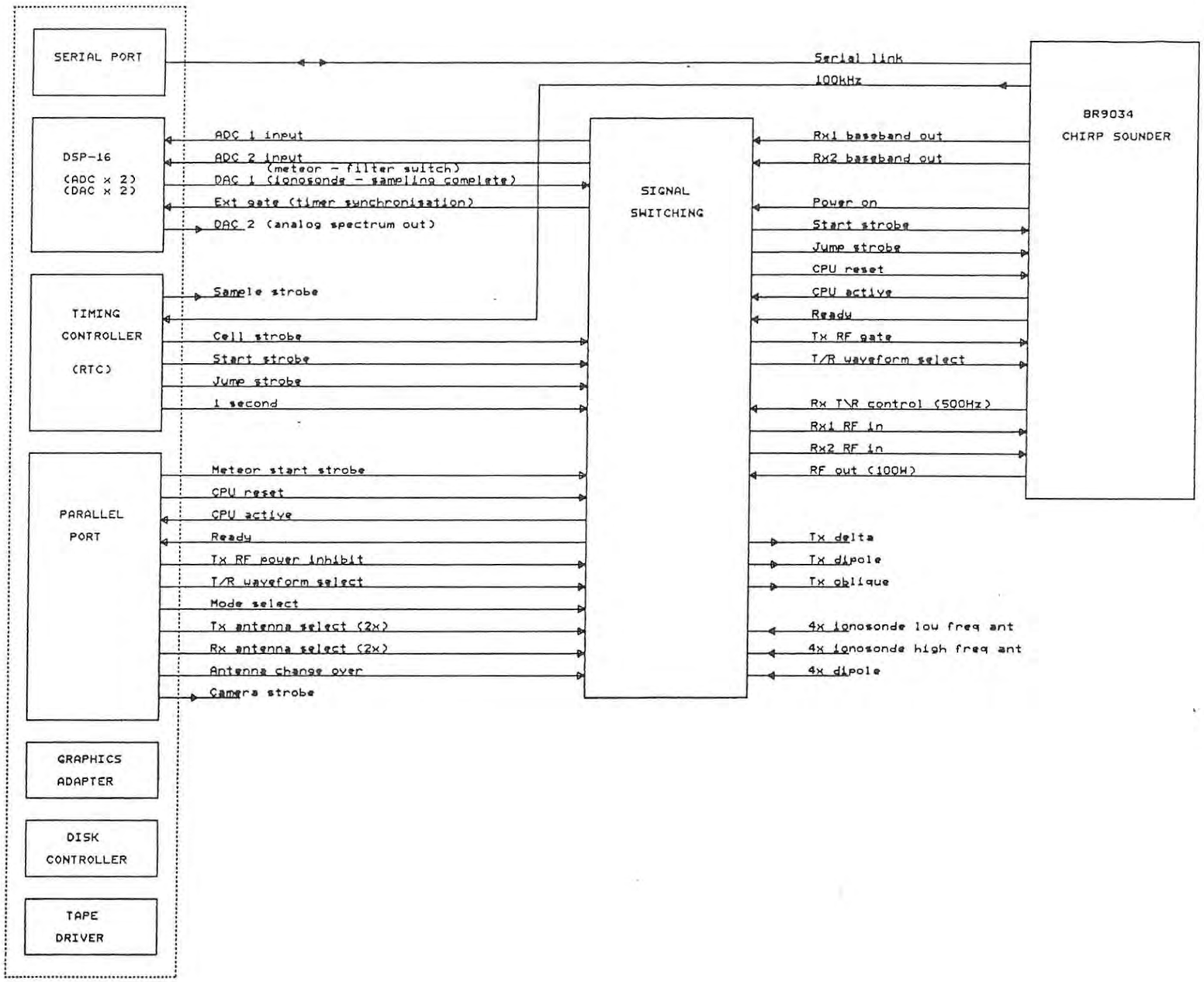


Figure 2-4: Block diagram of the upgraded BR-9034 system

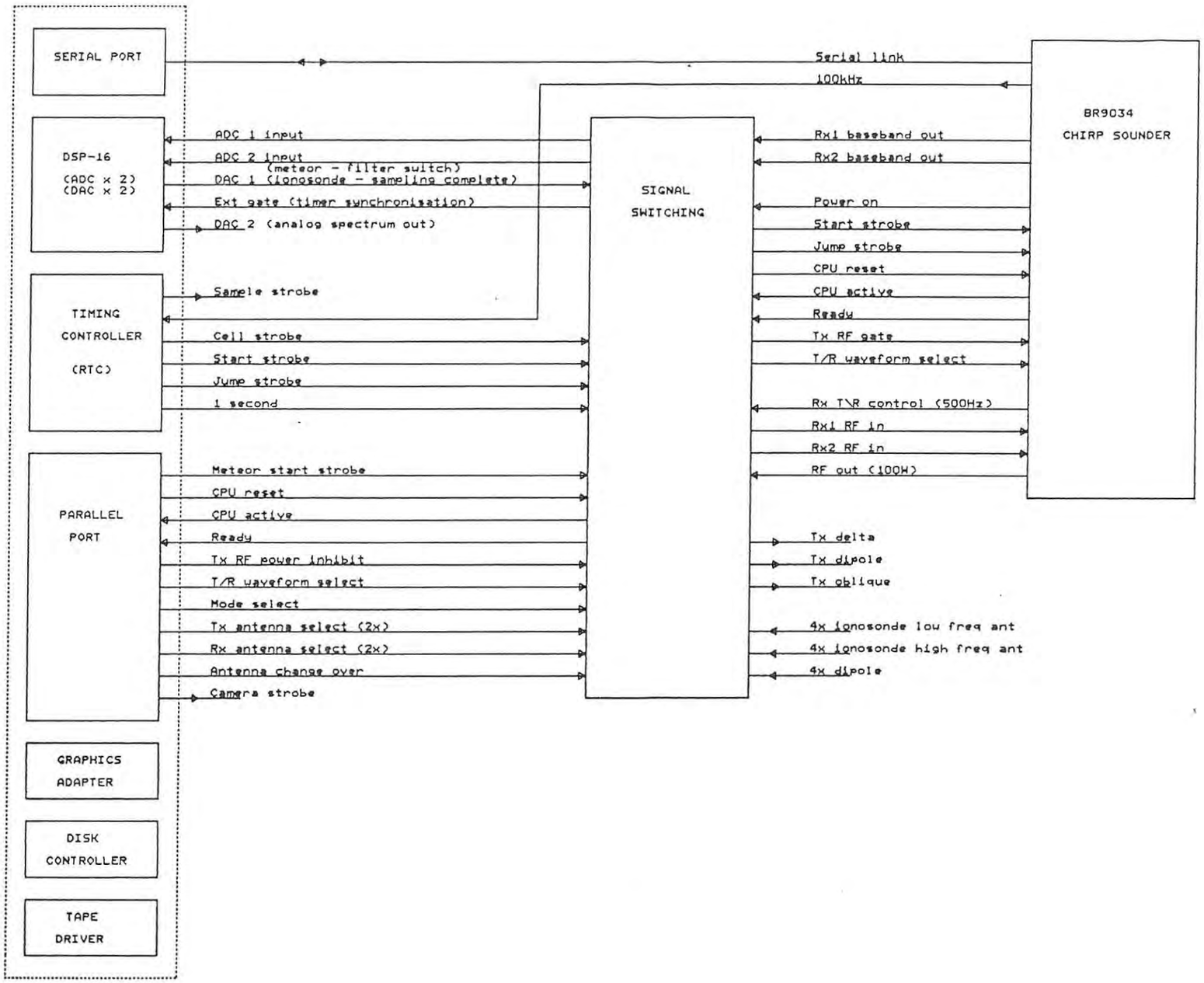


Figure 2-4: Block diagram of the upgraded BR-9034 system

a meteor radar. Control of the SS is largely via the parallel port on the PC. With the exception of the serial link (and internal frequency standard if used), all input and output signals associated with the 9034 connect to the outside world via the SS.

The SS also contains the circuitry necessary for synchronising the sample rate timer on the DSP16 at the start of each cell. The need for this will be described in detail in section 3.4.3.

2.3 Rationalisation of ionosondes

Figure [2-4] gives a block diagram of the overall 9034 system showing all control lines, signal lines, etc. The rationalisation programme is intended to introduce some compatibility between the three chirpsounder systems, both on a software and a hardware level. The outermost 'shell' of the system, the man - machine interface, will, within the bounds of any hardware constraints, remain the same on all three systems.

Figure [2-5] is a block diagram of the rationalised VOS-1/VIS-1 sounder systems. The Interface Adapter (IA) will replace the vertichirp controller in the present VOS-1 system. The IA will be substantially simpler in design than vertichirp control and Figure [2-6] shows the functional blocks that will be retained from this controller. A microprocessor is still required to interpret the serial string from the PC and program the frequency synthesizer. The IA will also have to retrieve the IF gain from the receivers and return this to the PC serially.

2.4 Man – machine interface

The architecture of the IBM PC is shown in Figure [2-7]. The BIOS[2] is stored in ROM on the motherboard. It gains immediate control of the system on power-up and is responsible for system checks, hardware identification and all low level hardware operations. If the application software makes numerous BIOS calls, then the BIOS can be loaded into 'shadow' RAM for increased speed.

DOS[3] is the operating system and affords control of the hardware at a higher level

[2]Basic Input Output System

[3]Disk Operating System

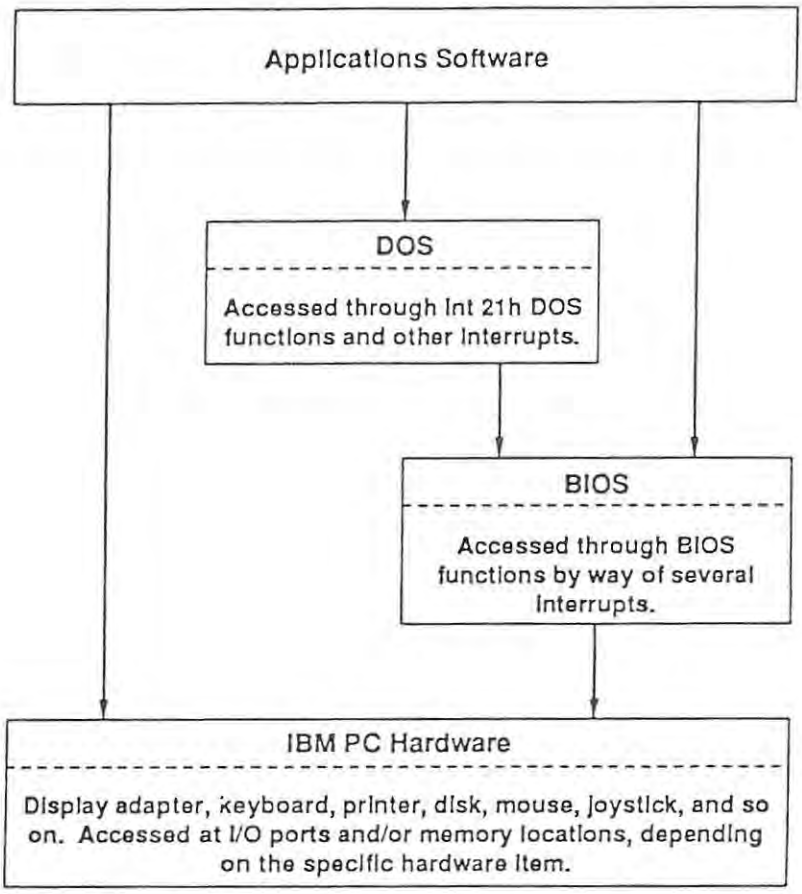


Figure 2-7: DOS and BIOS systems software as a control and interface layer [Borland (assembler) 1988]

than the BIOS. It is consequently easier to use and a single DOS function call may make use of a number of BIOS routines.

If all hardware operations are performed solely via the above two systems software layers, then the exact hardware configuration will be invisible to the applications software. However, if specific control of the hardware is required, or a non-standard device is being used, then these interface layers can be bypassed. DOS allows access to BIOS calls through software interrupt 21h. For additional information consult the DOS programmers guide [Microsoft 1984].

The applications layer itself may consist of a number of levels. We require that at the highest level, the user interface, the hardware configuration be invisible. Ideally this means that the user need not know what ionosonde, signal processing card, magnetic tape unit, etc. are configured in any particular system in order to operate it.

The software driver routines for the sounder itself and signal processing card will obviously vary for the three systems. However, if these routines can be placed in separate libraries then all that is required will be to recompile the main program with the appropriate libraries. This requires that the libraries have a common, predefined interface or declaration section. The 'high level' language chosen for software development must allow for this.

The user interface must be such that the program is easy to use. This requires a menu-driven approach with as much relevant information as possible being displayed on the screen. All user input must be validated before being accepted. Parameters associated with a particular mode of operation are to be stored on disk in a file of system operations. This will avoid problems during power failures. In addition system files could be transferred from one ionosonde system to the next, allowing for immediate duplication of system operation, scheduling, etc. These modes of operation must be rationalised, and more clearly defined than those used by the VOS-1 system.

Chapter 3

Description of additional hardware

3.1 Digital Signal Processing board (DSP-16)

The DSP-16 is a general purpose signal processing card (based upon the TMS32020 digital signal processor) which interfaces as a slave processor to an IBM PC. It forms a self-contained subsystem, with on- board RAM, two analog input and two analog output channels, and a programmable sample rate timer [Ariel 1987].

Communication is via four consecutive I/O ports in the host PC (switch selectable). The DSP-16 also interrupts the PC with the interrupt request line being jumper selectable. A 64 kbyte page in the PC's memory is required to accommodate the DSP-16's dual ported RAM (see Appendix I for available locations).

Control of the on-board sample rate timer is possible via an external gate, allowing hardware synchronisation of sampling between cells. A *cell* is the period of time over which the data for a single FFT was collected. In ionosonde mode this cell length corresponds to a linear chirp segment of the sounder sweep as described in Appendix A. A detailed overview of the DSP-16 is given in Appendices F & G.

3.2 Serial link

Serial communications between the PC and the BR-9034 is via an RS-232C link at 9600 baud. The transmission format is shown in Figure [3-1]. Parity is switch selectable (on the BR-9034 interface board) between none and space parity. It was left at space parity to allow for operation of the link by the existing dumb terminal instead of the PC, particularly while testing the system.

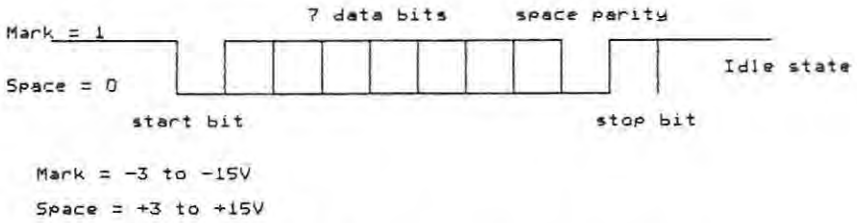


Figure 3-1: Serial transmission format

Serial port 1 on the PC is used (COM 1), and is configured during the controller software initialization routine. The PC is connected as Data Terminal Equipment (DTE) and the BR-9034 as Data Communications Equipment (DCE). Transmit, receive, ground and Data Set Ready (DSR) are the only lines used. DSR is held at +12 V when the BR-9034 is powered on and is used by the PC to check that the sounder is 'on-line'. If the link is disconnected then this check can be bypassed by a switch on the connector to the PC. This switch allows DSR to float to +12 V by disconnecting it from the 100Ω pull-down resistor. The connector wiring diagram is given in Appendix J. The serial card is configured to interrupt the PC whenever it receives a character or is ready to transmit the next character. [IBM 1983, BR Communications 1984, Nathanson P 1987]

3.3 Timing controller (TC)

A number of alternatives presented themselves for effecting advance/retard timing shifts. Firstly, the actual 5 MHz signal to the BR-9034 could be increased/decreased in frequency. This results in a smooth time adjustment even during sounder operation, as all the sounder frequencies are adjusted accordingly. With the VOS-1 a simple technique of pulse insertion and deletion is employed to produce a modified signal from the standard frequency of 100 kHz. The insertion of a pulse is in effect a doubling of the frequency over that

period. A similar technique could not be used with the 5 MHz signal however, as this is running close to the frequency limit of CMOS logic firstly, and secondly would almost certainly cause problems with the internal timing of the 9034, as it uses phase locked loops which have a limited input frequency range. Instead a small but continuous frequency adjustment would be necessary to effect these changes (4.5 – 5.5 MHz).

This was the approach first adopted by the author and was pursued to the stage of wire wrapping the proto-type. However, it can be seen (by the author now as well) that as all the receiver mixer frequencies are locked to the 5 MHz standard, a 10% change in the 5 MHz signal will produce a corresponding 10% change in the receiver tuned frequency (shown graphically in Figure [3-2]). This fundamental oversight proved to be a costly mistake time wise.

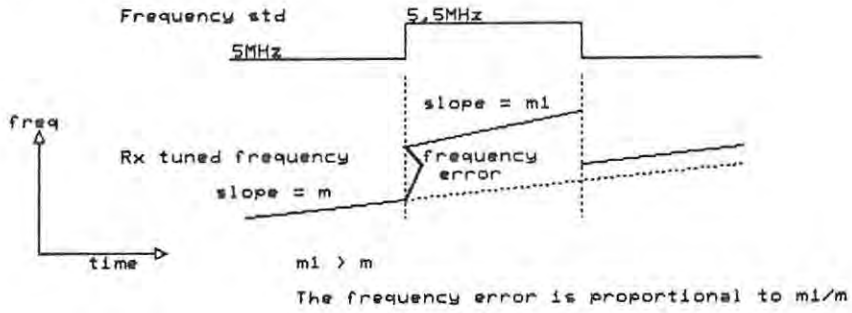


Figure 3-2: Graphical illustration of effecting timing shifts by adjusting the 5 MHz input

A modification on the above approach would be to effect a burst adjustment of the 5 MHz signal over say 10% of the cell duration. For the remaining 90% of the time the receiver would tune to the correct frequency. However, to produce suitably high shift rates would require adjustment of the input frequency by an order of magnitude. This is again unacceptable.

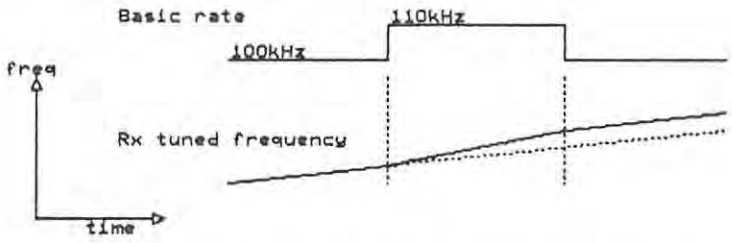


Figure 3-3: Timing shifts effected by adjusting the basic rate

An elegant solution would be to adjust the basic rate alone (see Figure [3-3]). This is

the technique currently used with the VOS-1, and if the timing controller could produce an adjusted 100 kHz signal compatibility with the VOS-1 system would be maintained, which complements the rationalisation programme. However, this signal is not readily accessible on the 9034 system and the potential for synchronisation problems if this signal was modified, were just too high to have pursued this approach even on an experimental basis. It would also have required numerous modifications to the 9034 in conflict with the authors philosophy of effecting the minimum number of changes.

From a software point of view the above approaches are relatively straight forward as no adjustment of programmed cell frequency jumps is required. In fact the timing shift is invisible to the software, apart from monitoring the size of the time shift.

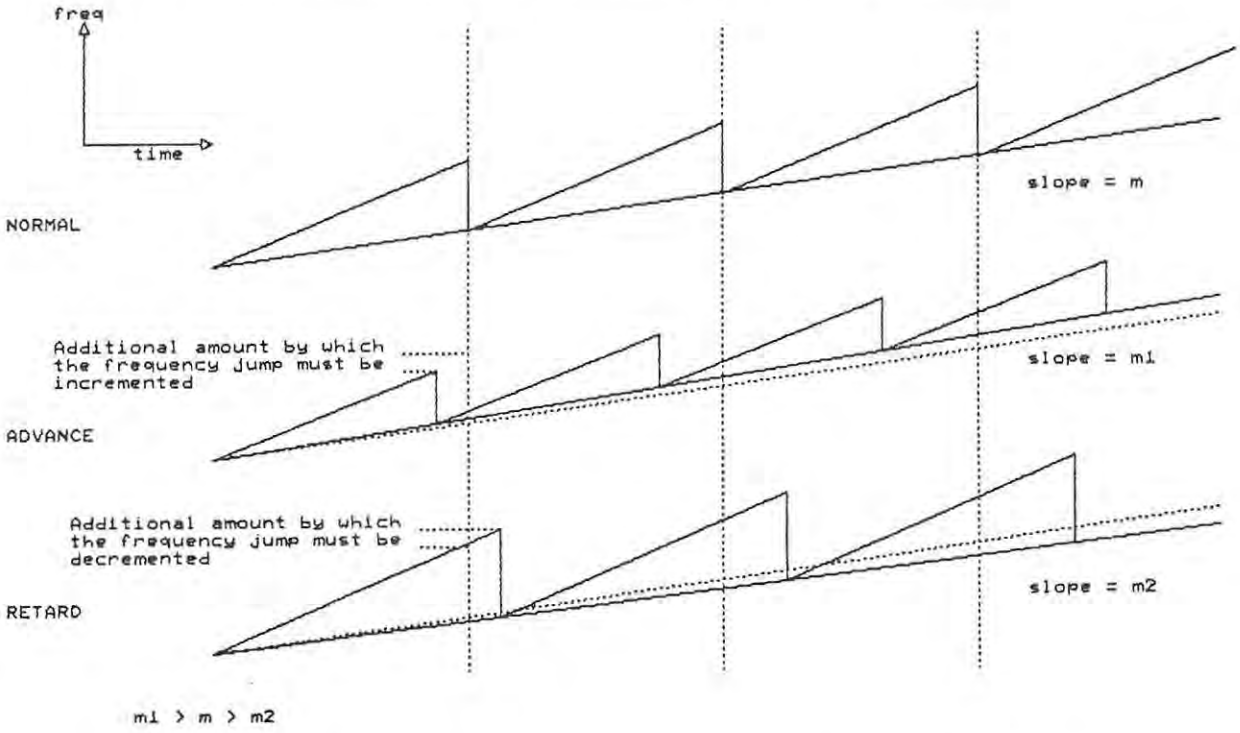


Figure 3-4: Implementation of timing shifts by adjusting the jump strobe

An entirely different approach would be not to modify any of the signals used internally by the 9034, which would avoid many of the timing problems. Cell start times are defined by either a cell start or jump strobe. Cell end times are not defined except by the reception of a jump strobe. Thus the timing could effectively be adjusted during operation by advancing or delaying these strobes.

This requires a software modification to the jump frequency programmed in the serial

string during timing shifts. Different jump offsets would be required for different timing advance/retard rates. *Adjustment to the system timing would only become apparent at the start of the next cell.* That is timing shifts would not appear smooth and continuous during sounder operation. Timing shifts could also only be implemented in fixed, relatively large steps. This method involves complex synchronisation of the software with the TC hardware.

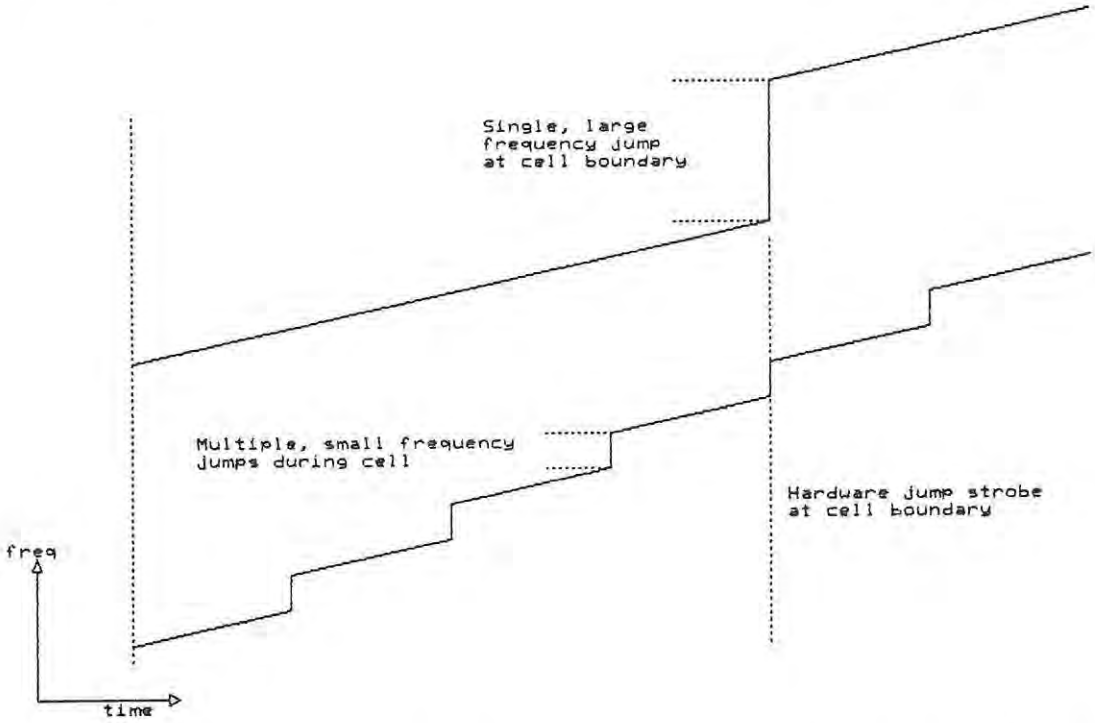


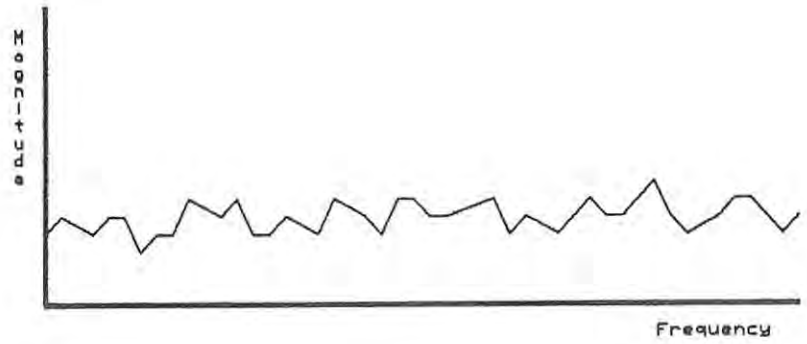
Figure 3-5: Graphical representation of software frequency jumps to effect 'smooth' timing shifts

All of the above approaches have the distinct advantage of providing a direct method of adjusting the Real Time Clock (RTC) by simply deriving its clock input from the adjusted output frequency. The RTC cannot be set with sufficient accuracy from the PC and changing its clock input frequency proved to be the only solution to adjusting the RTC by small amounts.

A disadvantage of the above methods is that they all effect the cell length. Adjusting the cell length would interfere with re-synchronisation of the external sample rate timer on the DSP16 (see section 3.4.3). This is not a problem if separate ADC cards are being used which require an external sample strobe, as this strobe could be suitably adjusted.

A final solution was to effect all timing shifts during sounder operation purely in software. In other words no adjustment is made of any of the input frequencies to the 9034, nor any of the strobe inputs. Consequently the cell length remains unchanged and all timing and synchronisation problems, both with the 9034 and sample rate timer on the DSP16, are avoided. Timing shifts are effected by adjusting the programmed jump frequency to the 9034. As with the previous method, timing shifts can only be performed in fixed jumps. The size of these jumps is however greatly variable. In addition it is possible to effect a number of small frequency increments at regular intervals during each cell, by using software jump 'strobes', instead of a single jump at the end of the cell (Figure [3-4]). This means that time shifts could be controlled more finely with prompt response to operator commands.

(a)



(b)

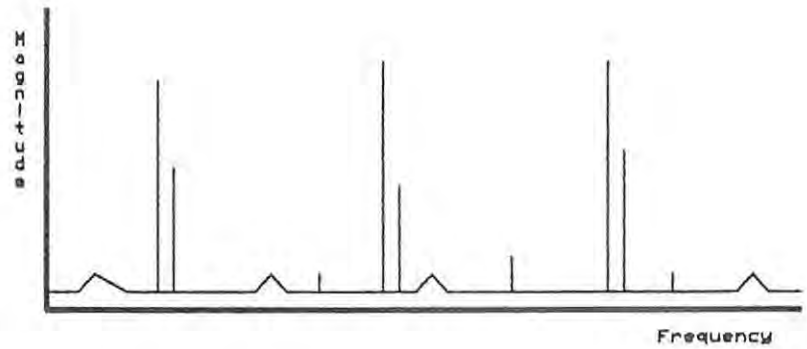


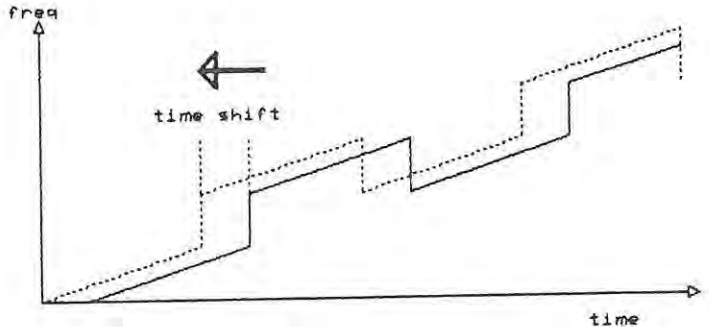
Figure 3-6: Comparison of frequency spectra for: (a) the hardware shift methods and (b) the software shift methods

As FFT's are performed on a cell by cell basis, a series of spikes will be observed as the receiver 'steps' through the signal being searched for. With the majority of the

other methods discussed no useful data could be expected from the FFT as the signal would sweep through the whole frequency band of the receiver. Figure [3-6] compares the expected FFT output for these cases.

This was the method finally implemented. It is completely effective for linear sweeps where the basic rate and overall rate are equal. A linear sweep where these two rates differ, or a structured cellular sweep where local frequency offsets are used within cells, will produce no results if the timing error is of the order of the cell length. The problem is illustrated in Figure [3-7]. Logarithmic sweeps could be used but would require involved calculations of frequency jumps taking into account the frequency offset. These limitations can for the most part be disregarded.

(a)



(b)

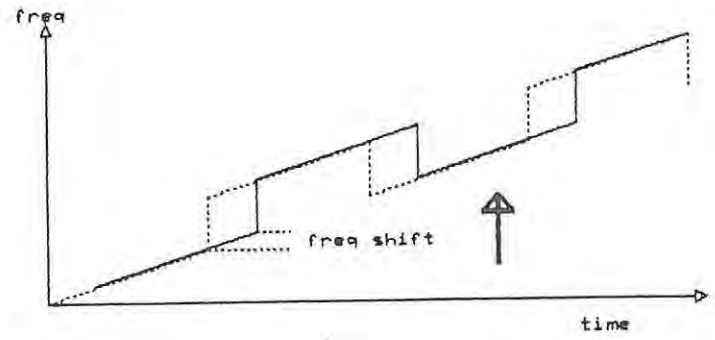


Figure 3-7: Signal synchronisation by (a) effecting a hardware timing shift and (b) synthesizing a timing shift by a frequency shift, illustrating the problems associated with a structured cellular sweep.

A block diagram of the timing controller is given in Figure [3-8]. From speed and power considerations the TC has been constructed largely from high speed CMOS devices.

Circuit descriptions are given in Appendix K and circuit diagrams in Appendix L.

Note: *The actual TC hardware and RTC are only adjusted at the end of a sweep.* Timing shifts of up to ± 1 second are possible in 0,1 ms steps. The rate of adjustment is such that to effect a hardware shift of 1 second in either direction takes 1 second to complete.

An onboard crystal oscillator is provided to maintain approximate system timing if the input signal from the frequency standard is lost.

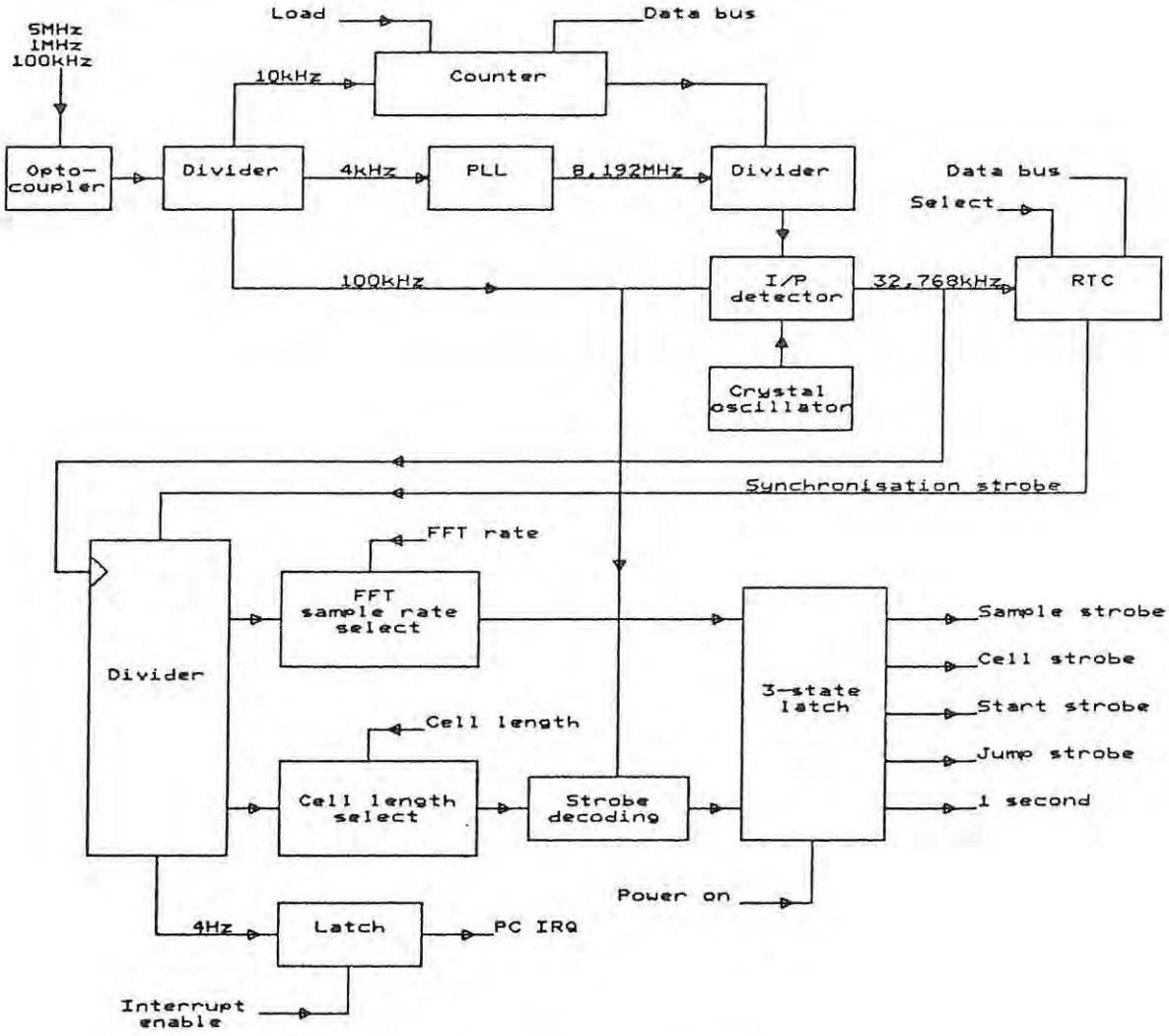


Figure 3-8: Timing controller block diagram

3.4 Signal Switching (SS)

3.4.1 Antenna switching

In ionosonde mode four spacially located antennas are needed for determination of angle of arrival, doppler velocity, etc (Appendix A). Antenna switching occurs between cells, ie at a maximum rate of 4 Hz. If mechanical relays were used to achieve this, then signal attenuation would be negligible but reliability questionable. Even if the programme schedule were only a simple synoptic consisting of a single 5 minute ionogram every hour, the majority of these devices could only guarantee 1 year of reliable operation. A malfunction might not be readily detectable by the operator either. So, some type of solid state switching device is required. The CMOS analog switches that will be discussed in section 3.4.2 would not be suitable as attenuation would be too high for such small signals. BR Communications use two baluns and some diodes to perform RF switching. Single package devices (PAS-3) are available in this configuration (Figure [3-9]), and are in fact the devices that are used by the meteor system as will be described shortly.

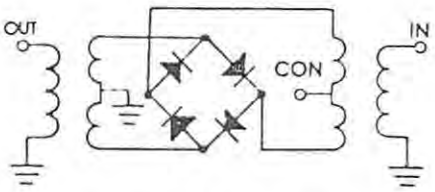


Figure 3-9: RF switch

For maximum flexibility any of the four ionosonde receive antennas should be able to connect to either or both the receivers. In addition it may or may not be desirable to ground unused antenna inputs as done presently in the VOS-1 system (see Figure [3-10]). In the authors opinion this is likely to degrade system performance, but this would have to be determined empirically.

An alternative design for antenna switching in ionosonde mode, suggested by the author and Mr BT Bonnevie, is shown in Figure [3-11]. This approach simplifies the required hardware substantially, while sacrificing minimal system flexibility. Requirements on antenna orientation are more rigid, and the exact configuration will have to be taken into consideration during post analysis of recorded data. Programming of multi-cell soundings will also be simplified, as antenna allocation will be independent of station location. It is

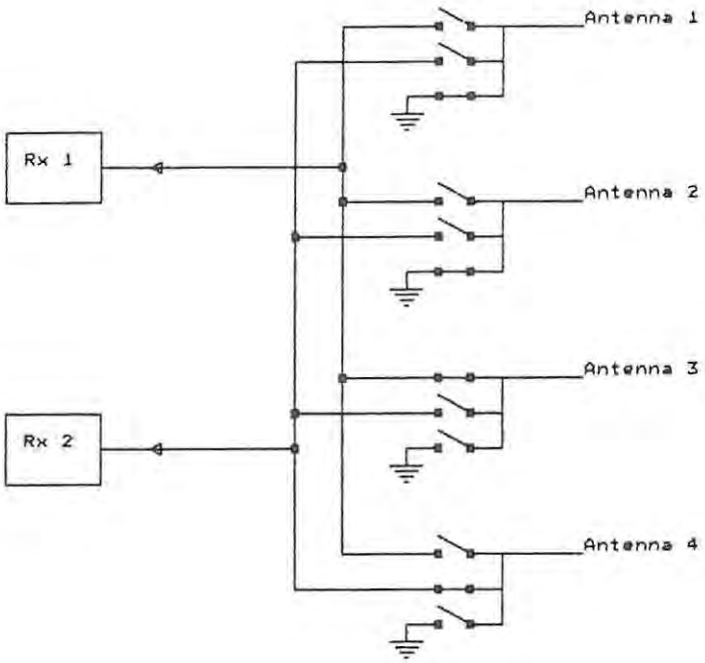


Figure 3-10: Antenna switching in the VOS-1 system

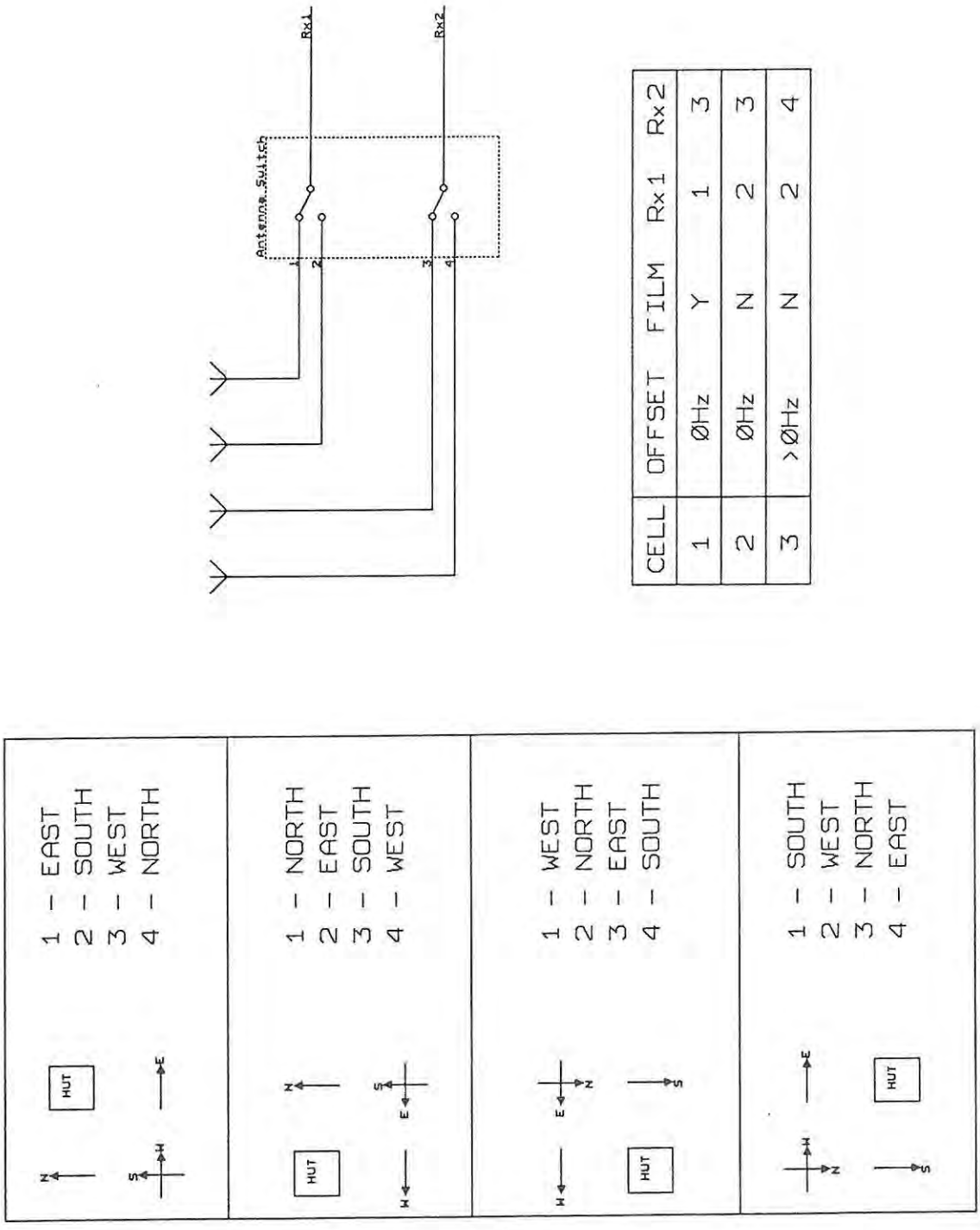
important that in the physical construction of the SS all RF signal paths be kept equal in length. Antenna selection is via the parallel port on the PC and some decoding logic in the SS itself.

In meteor mode pairs of receive antennas are switched to the receivers at 250 Hz (Appendix B). The circuitry to perform this was designed by Prof G Poole and uses four PAS-3 switches. The switching signal is provided by the 4070 dual channel receiver, *Rx T/R control out* (Appendix E).

Depending on whether the system is operating in ionosonde or meteor mode, one bank of antennas will be disabled. This control line comes directly from the parallel port.

The PAS-3 switches were very expensive but no suitable alternative could be found. The author has thus provided a rough circuit diagram (Appendix L) of the RF switching circuitry, but this will have to be constructed at some future date when the devices can be purchased.

The RF power out can be connected to one of three transmit antennas or a 50 Ω dummy load. Should the SS unit lose power or the antenna select input, the transmitter output will be switched into the dummy load.



CELL	OFFSET	FILM	Rx1	Rx2
1	ØHz	Y	1	3
2	ØHz	N	2	3
3	>ØHz	N	2	4

Figure 3-11: RF signal path in ionosonde mode, showing all possible antenna orientations, and a sample 3 cell sounding structure (Mr BT Bonnevie)

3.4.2 Base-band signal and filter switching

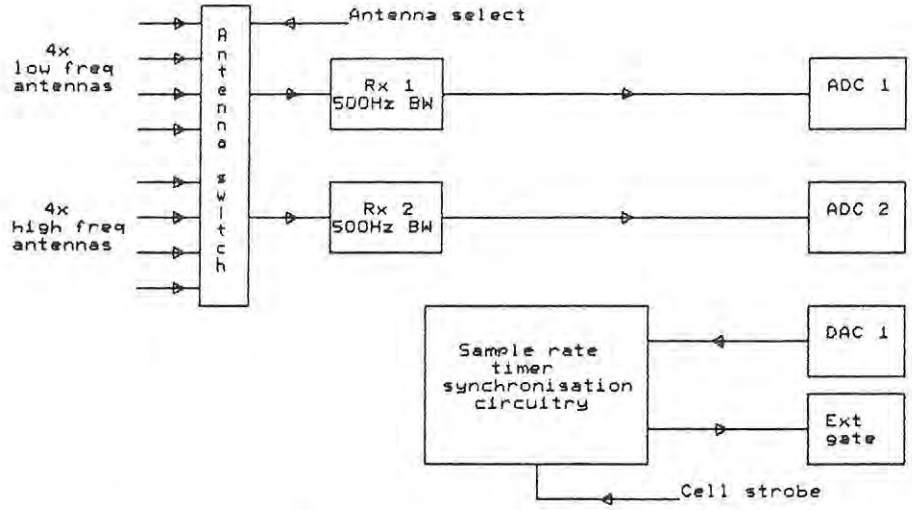


Figure 3-12: Effective signal path in ionosonde mode

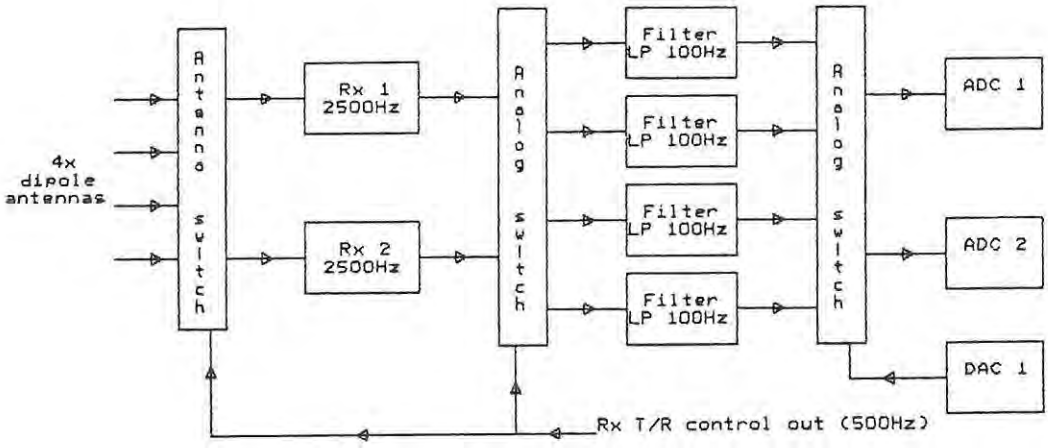


Figure 3-13: Effective signal path in meteor mode

In ionosonde mode, receiver outputs are sampled directly by the ADC's (Figure [3-12]). In meteor mode the receiver bandwidth is 2500 Hz and the receiver outputs pass through 100 Hz low pass filters before being sampled. An analog demultiplexer switches each receiver output between a pair of these filters. Switching is synchronous with the RF section, with allowance for transmission delay through the receivers (Appendix B). Pairs of filter outputs are then switched to the ADC's by an analog multiplexer as shown in Figure [3-13]. Control of this multiplexer is provided by the DAC 1 on the signal processing card (or the parallel port if separate ADC cards are being used). A timing

diagram of multiplexer operation is given in Figure [F-5]. Signal switching is done using CMOS analog switches (type 4053 & 4066). An additional switch is used to bypass these low pass filters in ionosonde mode.

3.4.3 Synchronisation of the external sample rate timer

Two approaches to sampling the receiver outputs exist. Either the ADC card(s) use an external sample strobe, provided by the timing controller, or alternatively the card may use an onboard sample rate timer, in which case synchronisation of the timer with the cell strobe is required.

The second alternative is applicable to the DSP-16 and synchronisation is performed at the start of every cell. This is achieved by using a slightly higher sample rate (1024,6 Hz) so that the last sample will be taken before the end of the cell. Immediately after this sample the *sampling complete* line (DAC 1) to the signal switching unit is raised hi. This positive edge clocks a flip flop which then disables the sample rate timer via an external gate on the DSP-16 card as shown in Figure [3-14]. This gate holds the timer reset until it is released by the next cell strobe from the TC. *A fundamental requirement of this approach is that the time interval between enabling the timer and first sample being taken, remain contstant.* A detailed timing diagram is given in Figure [F-4].

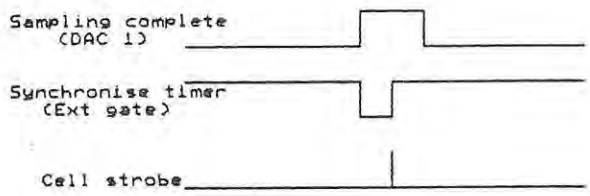


Figure 3-14: Synchronisation of the sample rate timer on the DSP-16

3.5 Film Recording System (FRS)

The VOS-1 system produces ionogram records by 'dragging' the film across a viewing slit on an oscilloscope screen. This is a cumbersome approach and is fraught with mechanical and electronic problems. The intention is to replace this with a system whereby the ionogram is first plotted on a graphics screen and then photographed using a camera

with a motor-drive. A single control line has been made available on the parallel port to provide the necessary strobe to the camera.

Ideally a second monitor is needed, that could be set up in a dark cabinet and used purely for this purpose. To produce ionograms of suitable quality a graphics screen (EGA[1] or VGA[2]) would be needed. This poses a problem however as every monitor requires a driver card in the PC and the memory location of the various cards has been predefined by IBM and is used by DOS and the ROM BIOS. It is theoretically possible to simultaneously operate a CGA[3] and monochrome monitor if both are running in text mode. However only one adapter can be installed if graphics mode is to be used due to over- lapping of memory addresses as a result of the additional memory requirements (see Figure [I-5] for CRT memory locations).

The final possibility is to use two similar monitors in parallel. This would require either installing two identical graphics adapter cards, or driving both monitors from one card. The first is not possible as the PC requires acknowledge signals from the adapter cards, and this would cause bus contention as the timing on the two cards could never be exactly the same. The second would almost certainly require additional buffering of one of the monitors.

The author has, where possible, allowed for the future inclusion of the FRS with the present software.

---

[1] Enhanced Graphics Adapter

[2] Video Graphics Array

[3] Colour Graphics Adapter

Chapter 4

Software

The PC performs two distinct functions. Firstly it has overall control of the complete sounder system, and is responsible for the programming of the ionosonde as well as automatic scheduling of operations. Secondly it is responsible for data capture, processing and data display, and finally storage to some mass storage medium. The software that performs the above two functions is contained in the file **CONTROL.EXE**.

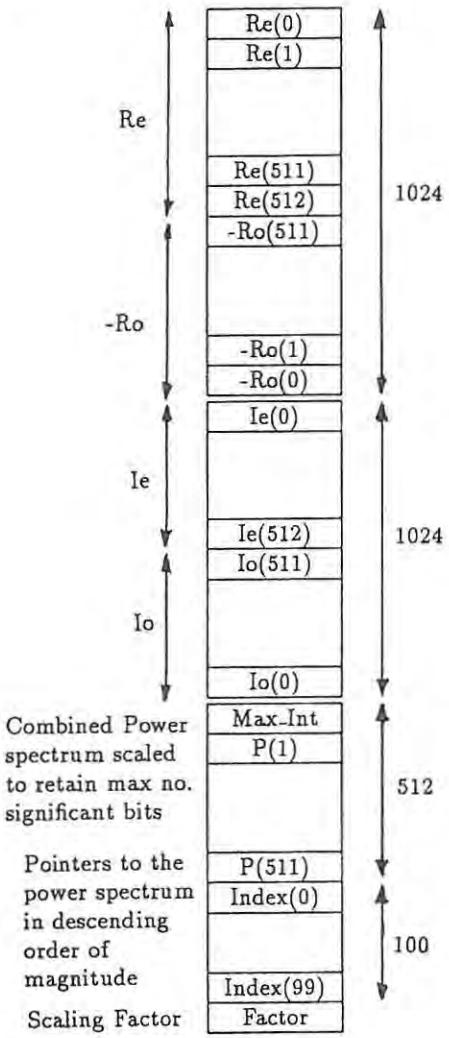
System timing is derived from the timing controller, a card which plugs into the PC and is programmable by the PC. The PC also plays host to the digital signal processing card (and analogue to digital converter cards) which performs initial data capture and processing.

4.1 Data capture and preliminary signal processing

This is performed using the Ariel Corporations, TMS32020 based, DSP-16. The two on-board 16 bit analog to digital converters (ADC) are used to sample the two input channels simultaneously. Hardware sample rates of approximately 1024 or 2048 Hz are used. Lower sample rates can be achieved by discarding samples in software. Samples from consecutive cells are stored in two separate buffers with one buffer being processed while the other is being filled (Figure [F-2]). The DSP-16 software is written such that it has two modes of operation, described as follows.

Data upload format

Ionosonde mode



Total: 2661 words

Meteor mode

Spectral maxima frequencies	$f_{max}1$
	$f_{max}2$
	$f_{max}3$
	$f_{max}4$
	R1 (at f_{ave})
	I1
	R2
	I2
	R3
	I3
	R4
	I4
	Freq spread 1
	Freq spread 2
	Freq spread 3
	Freq spread 4
	Sample max 1
	Sample max 2
	Sample max 3
	Sample max 4
	Noise Hi
	Noise Lo

Total: 22 words

Control data download format

Mode
No. samples / channel
Points / cell (PPC)
Echo criterion (Hz)
Spread threshold
State
Sample rate divider

Total: 7 words

Mode = 0 Meteor operation
Mode $\neq$ 0 Ionosonde mode
State = 0 Data output to PC (interrupt PC)
State > 0 Analogue spectrum only
Sample rate = Hardware rate / $2^{\text{Rate divider}}$

Figure 4-1: DSP-16 to PC data transfer format

4.1.1 Ionosonde mode

In ionosonde mode a 1024 point complex FFT simultaneously transforms data from both input channels into the frequency domain. If 1024 points per channel have not been sampled, as for example in the case of half and quarter second cells, the remainder of the buffer is filled with zeros. This has the effect of extending the period of time over which the signal is sampled in the time domain (see Appendix C). The FFT uses an in-place, scrambled input – natural output algorithm and was adapted from the one supplied by Delanco Spry with the model 10 DSP board [Langman D 1986].

The real and imaginary components, corresponding to channels A and B respectively, are separated using an unscrambling routine based on Equations [C-6] & [C-7]). A combined power spectrum of the two channels is then calculated, scaled to retain as many significant bits as possible, and indexed in descending order of magnitude to a preselectable number of points per cell (PPC). Finally all of the above information is transferred to the host PC, in the format shown in Figure [4-1], by consecutively writing each word to the host data port. The DSP-16 interrupts the PC when it is ready to upload data. The above routines were adapted from those written by Mr BT Bonnevie for the model 10. Compatibility of the two sets of code has been maintained as far as possible.

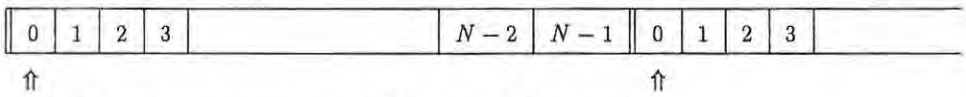


Figure 4-2: Graphical representation of data storage in ionosonde mode.

Re-synchronisation of the sample rate timer is required at the start of each cell in order to maintain phase relationships for cell comparisons. The external gate is used to disable the interrupt timer at the end of each cell until the next start cell strobe. This gate is disabled via one of two digital to analog converters which outputs a positive pulse after the required number of samples have been captured. The gate is reset by the next start cell strobe. The relevent timing diagram is shown in Figure [F-4]. [Ariel 1987]

The first valid sample is stored in position 1 of the respective channel buffer. The value stored in position 0 is conceptually the sample from time zero, and is set to zero in ionosonde mode before computing the FFT (see Figure [4-2]). This is unimportant as

there will be no valid data to sample at that time anyway if the ionosonde is 'chirping' (Figure [4-3]).

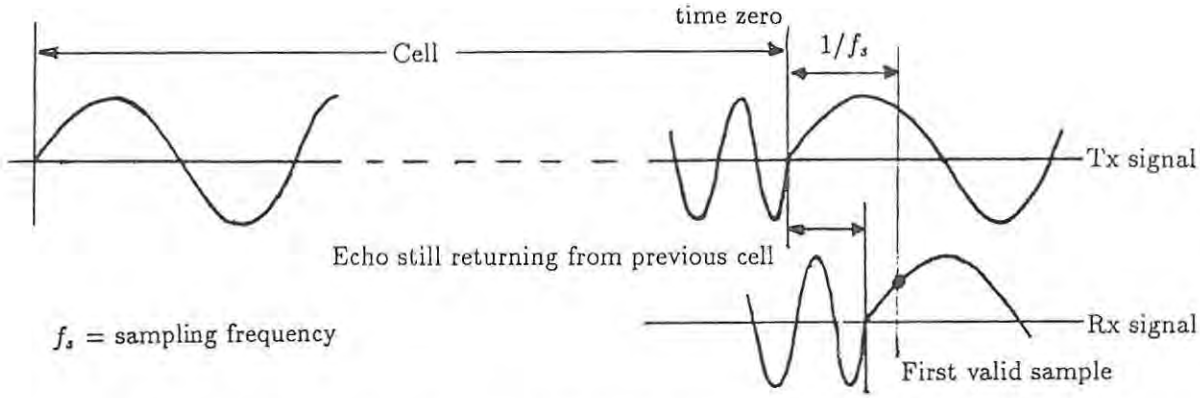


Figure 4-3: Transmit and received signals at cell boundaries in 'chirp' mode.

The error allowed in the sampling time of the two channels must not exceed 1% of the shortest period in the signal out of the receiver (500 Hz in ionosonde mode).

$$T_{min} = 1/f_{max} = 1/(500Hz) = 2\ ms$$

$$1\% \longrightarrow T_{error} = 20\ \mu s$$

The DSP-16 samples to within 1 ns, ie simultaneously for our application. Similarly the error allowed between appropriate samples in different cells must meet this criterion, ie. be less than 20 μs.

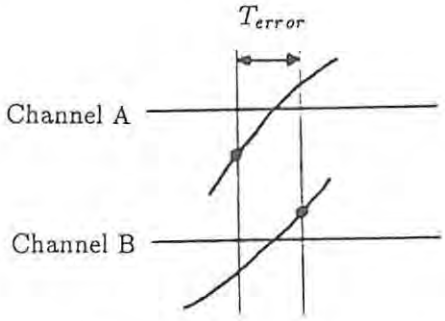
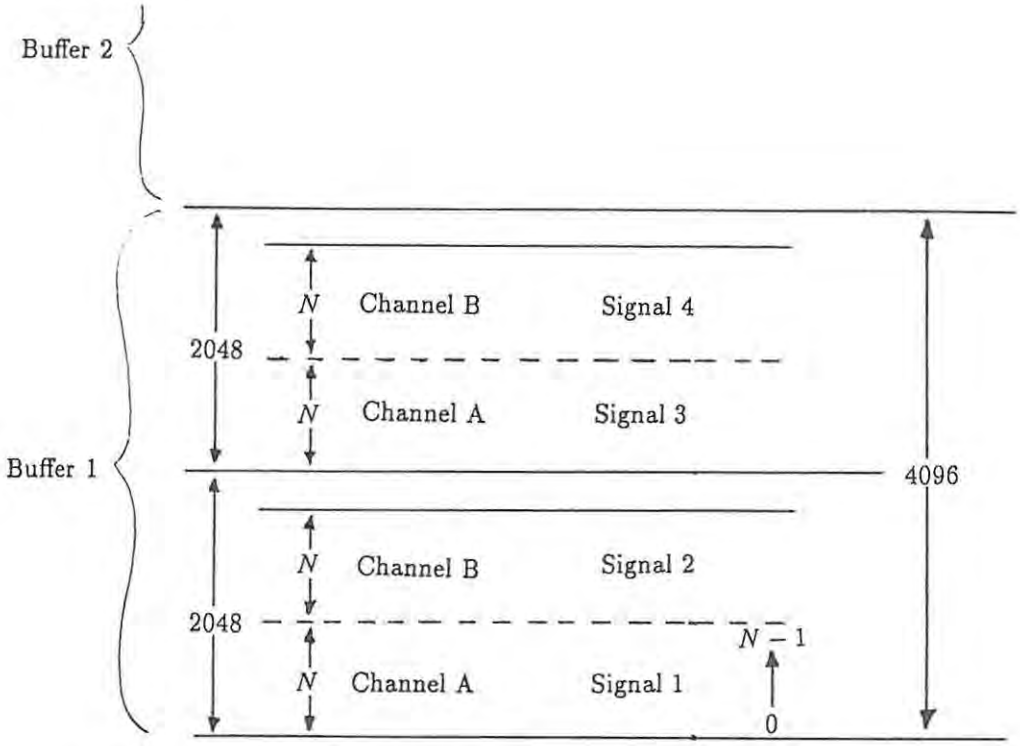


Figure 4-4: Sampling error

4.1.2 Meteor mode



N may have a maximum size of 1024

Figure 4-5: Storage of samples in meteor mode

For determination of angle of arrival four antennas are required. However only two phase matched receivers are available. With the receiver bandwidths set at 2500 Hz, the RF inputs and base-band outputs are 'synchronously' switched, with the outputs being band limited to 100 Hz by four lowpass filters (see Appendix B). In addition only two ADC's are available to sample these four channels. Thus an external analog multiplexer is used to switch the ADC's between pairs of outputs. This multiplexer is switched using one of two digital to analogue converters (DAC) as shown in Figure [F-5].

The ADC's sample at 1024 Hz but from alternate channels resulting in an effective sample rate of 512 Hz per channel. Using a 1024 point transform, points on the computed spectra will be separated by 0,5 Hz (512 values per spectrum covering the frequency range 0–255,5 Hz). Each of the two buffers used to store the samples is further subdivided as shown in Figure [4-5]. The number of points sampled per transform determines the length of each cell. These *time cells* are contiguous in meteor mode.

At the end of each cell, a power spectrum is obtained for each of the four effective channels, with the maximum of each and the frequency, f_{max}, at which it occurred being recorded. The four values of f_{max} are then compared and provided the difference between the lowest and highest frequency meet the *echo criterion*, the echo is assumed to be a valid meteor. The echo criterion is the maximum frequency difference allowed between spectral maxima, the width of which can be set from the host PC.

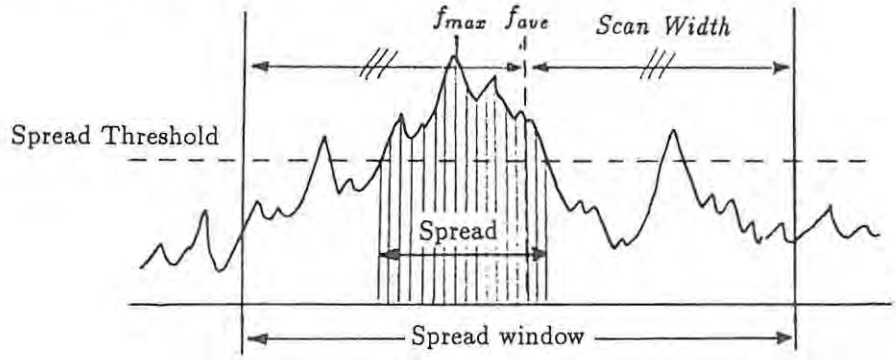


Figure 4-6: Determination of frequency spread in meteor mode

If the echo criterion is met, the real and imaginary components of the four spectra at an average[1] frequency, f_{ave}, are up-loaded to the PC. For each channel, f_{max}, together with a measure of the frequency spread of each of the transformed signals about f_{max}, and the maximum of each of the sampled signals in the time domain are also uploaded (Figure [4-1]).

The *frequency spread* is the range of frequencies about f_{max} for which all the computed spectral magnitudes exceed a certain threshold (the *spread threshold*, see Figure [4-6]). The number of points on the spectrum scanned either side of f_{ave} in order to find the spread, is determined by the value of *ScanWidth*, a TMS320 variable at present set equal to 20. Twice the product of ScanWidth and the sample separation gives the *Spread window.*

A measure of the average spectral noise is also up-loaded for display. This value is normalised to the value it would have if the cell length had been 1 second.[2] No adjustment is made of any other parameters to compensate for cell length.

Once echo data has been up-loaded to the PC it may first have to pass up to three

[1] f_{ave} is rounded to the nearest discrete value available.

[2] If the cell length is halved then the average noise spectrum amplitude is also halved.

software 'amplitude filters' before being accepted by the PC as a meteor echo. Each filter consists of a rectangular window function with the upper and lower frequency limits, as well as the amplitude threshold, selectable by the operator. The average amplitude of the four channels is used.

Note: These 'filters' are not filters in the normal sense and the relationship between their effective gain and the threshold is not simple, and depends on the size distribution of the meteor echoes.

4.1.3 General information

In both meteor and ionosonde modes an analog spectrum output is available from a second DAC which also provides a trigger pulse for an oscilloscope (Figure [F-6]).

Should the TMS320 run out of time to complete its instructions within a cell, it will interrupt the host PC and upload a single word so as to notify the host. Once the PC has acknowledged this interrupt, by reading the slave data port, the TMS320 will reset itself and continue execution.

Documented listings of all controller and TMS320 source code can be found in the separate text *Source code listings for Control of a BR-9034 chirpsounder, and Data Capture using a TMS320 signal processor.* Signal processing flow diagrams are given in Appendix H.

See Appendices F & G for detailed information on the DSP-16 and TMS32020 respectively. To assemble the source code run **ASM320 DSP16TMS**.

4.2 Operating system

One of the major problems experienced was not due to the choice of programming language but that of operating system. DOS, being non-reentrant, is not suited to real time or multitasking operations. All three languages use DOS function calls to perform I/O and thus great care has to be taken when requiring I/O in interrupt driven routines (ie If DOS is interrupted the interrupt service routine can not make a DOS function call). A number of compromises had to be made due to this limitation. The author did look at the possible use of alternative operating systems, in particular PCMOS, but these did not always offer true DOS compatibility and were costly.

[Microsoft 1984, Software Link 1987]

There are two possible solutions to the reentrancy problem of DOS. Firstly, the software can be written such that only the main process, or alternatively only an interrupt service routine, has access to DOS function calls. The second option is to check whether DOS is busy before making any calls to it and if so setting a flag and waiting until it has finished. This requires redirection of a number of interrupt vectors and the use of a regular interrupt, such as the timer (IRQ 0), to perform a 'DOS not busy' and 'flag set' check. The second solution is not desirable if an interrupt must be serviced immediately. It also reduces portability of the software. [Borland (Pascal) 1988, Waite Group, Ladd SR 1988, Prosise J 1987]

The first approach was adopted, with only the main program providing screen output. Disk access is allowed from within only one interrupt service routine, and from the main program only when this interrupt is gated off.

4.3 Choice of high level language

4.3.1 Language options

Three high level programming languages were considered for the development of the controller software and user interface. These were — *C*, *Pascal* and *Modula-2*. These languages all support 'high level' structured programs, modularity, a wide selection of data types, and have graphics capabilities.
The particular interpretations of these languages/compilers under consideration were:

- Turbo C version 2
- Turbo Pascal version 5
- TopSpeed Modula-2 version 1

All of these use a similar integrated development environment consisting of a more or less standard editor, fast (efficient code producing) compilers with immediate error location in the source code, smart linkers, and run time error location (and debugging facilities). These integrated environments greatly reduce software development time.

None of the above implementations are directly compatible with the respective ANSI standard language definitions. This reduces the portability of the source code to other

implementations on different machines or operating systems. However, this was not considered too important for two reasons. Firstly, if the code is easily readable hopefully any changes required to move to a more standard implementation could be made fairly readily. This would essentially consist of writing additional routines supplied as part of the language, or in associated libraries, with the above implementations. Secondly, as the controller hardware (ie IBM PC/AT) is unlikely to be replaced in the foreseeable future, if and when it is the software will probably require extensive revision anyway. (Additionally the standards are continually undergoing revision.)

4.3.2 Turbo C

C was developed in 1972 by Dennis Ritchie and has its roots in the language BCPL which in turn was a descendent of Algol (Figure [4-7]). The UNIX operating system is developed almost entirely from C.

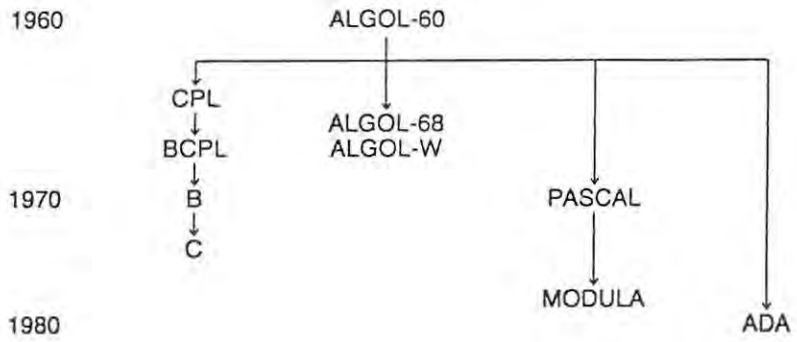


Figure 4-7: Schematic genealogy for members of the Algol family of programming languages [Brown DL 1985]

C is a subroutine orientated language, allowing separate compilation and linking of these routines. It does not allow nesting of subroutines (functions) within subroutines. It has a powerful set of control statements, similar to those of Pascal and Modula-2, however it does not support as wide an assortment of structured data types nor the ability to create new types. On the whole, it also lacks readability in comparison with the other two. "Even seasoned programmers may find themselves investing a significant amount of time trying to understand someone else's code ...or even code that they themselves have written in the past!" "For instance C permits (and even encourages) the construction of very

terse code." "Pascal programs are, on the whole, much more readable because they don't encourage this terse style" [Brown DL 1985]. This is not to say that readable code can not be written in C, it just requires greater effort on the part of the programmer.

"If a language tightly restricts the circumstances in which a location associated with a variable of one type may be interpreted as a different type, the language is said to be *strongly typed.* Pascal is strongly typed, C is not" [Brown DL 1985].

"What C may claim as a feature may easily be mistaken by a Pascal programmer as a bug." "Veteran C programmers respect and appreciate those special features that give this language its distinctive flavour and realize that they reflect the needs of the systems programmer." "C permits great programming flexibility and sacrifices (to some extent) program reliability; the C programmer has great control over the machine's operations. Pascal produces programs that are reliable but not as flexible" [Brown DL 1985]. "Turbo C and Turbo Pascal are moving towards the center of this C-Pascal language spectrum: Turbo C adds some structure to C, and Turbo Pascal adds some flexibility to Pascal" [Borland (C) 1988].

Turbo C allows linking to Turbo Pascal as well as assembler routines. This would be advantageous due to the amount of time already invested in developing the Data Capture System (DCS) in Turbo Pascal and the use of assembler routines for time critical operations. Turbo C has an advanced graphics library and routines to handle interrupts. (Although no mention is made of it in the manuals, the author suspects that certain sections of the language, such as I/O, may not be reentrant!)

Turbo C supports the draft proposed American National Standards Institute (ANSI) standard, fully supports the Kernighan & Ritchie definition (*The C Programming Language*), and includes certain optional extensions for mixed-language and mixed-model programming [Borland (C) 1988].

In addition, interface routines to the DSP-16 board already existed for Microsoft C and were almost directly compatible with Turbo C.

4.3.3 Turbo Pascal

Pascal is a general purpose, high level programming language developed by Niklaus Wirth in 1971, and was intended to aid the teaching of a systematic approach to computer programming.

Pascal is a strongly typed language, rich in data and control structures. Like Algol from which it descended it is a subroutine orientated language. Unlike C though, it allows nesting of a number of layers of subroutines (procedures and functions) within subroutines. In other words, better 'hiding' of data structures, resulting in simpler more reliable code.

Standard Pascal has a number of disadvantages. Firstly, "It promotes a monolithic overall program structure, not one in which re-usable code can be channeled into (or drawn out of) libraries" [Terry PD 1987]. Turbo Pascal overcomes this to some extent with the introduction of units, which are pre-compiled segments of code that can be used by other programs. Secondly, "The constructor mechanisms supported do not go quite far enough. In particular they are 'mid-level' and do not properly address the needs of programming closer to machine level than integer suggests" [Terry PD 1987]. Turbo Pascal includes *inline* declarations and directives, and linking to assembler routines to overcome this, as well as interrupt handling facilities.

Turbo Pascal includes the same advanced graphics routines as Turbo C. It does have the problem however, that the I/O and dynamic memory handling routines are non-reentrant. *Non-reentrant* means that the variables associated with a particular operation are stored in fixed memory locations, and not on the stack. Thus if an interrupt occurs during the execution of a non-reentrent operation, which calls this operation, all variable values will be corrupted after the return from interrupt [Borland (Pascal) 1988].

One of Pascal's main attractions though is its readability and the close resemblance of Pascal source code to English language if descriptive identifiers are used. Another important consideration was the amount of operational code already developed in Turbo Pascal for the DCS.

Although Turbo Pascal claims "ANSI standard compatibility" [Borland (Pascal) 1988], it also contains more adaptations and extensions to the standard than do either Turbo C or TopSpeed Modula-2. Thus it is probably the least portable of the three languages under consideration.

DSP-16 interface routines existed for Turbo Pascal version 3 and could be purchased for version 5, if they could not be adapted from the existing source code.

4.3.4 TopSpeed Modula-2

Modula-2 was also created by Niklaus Wirth who developed Pascal and it is a direct descendent of this language. "You can think of Modula-2 as a 'better', or more powerful, Pascal" [JPI 1988].

In Modula-2 the source code is broken up into modules which can be compiled separately and linked together at run time, as can be done with functions in C and units in Turbo Pascal. It is even more strongly typed than Pascal, but this can be more than a little irritating at times. For example it requires a type conversion in order to add an integer to a real. Modula-2 has built in support for multitasking, and the TopSpeed version includes an advanced time-sliced scheduler in its library, for easy implementation of concurrent processing. TopSpeed Modula-2 does not however have a library of graphics routines as advanced as those supplied with the Turbo packages [Terry PD 1989].

At present the *de facto* standard for Modula-2 is Niklaus Worth's *Programming in Modula-2.* TopSpeed's implementation is based on the description in this book. However it does contain a number of useful extensions not present in the language as defined by Wirth [JPI 1988].

Interface routines to the DSP-16 would have to be adapted from the given assembly routines as none existed for Modula-2.

4.3.5 Speed comparisons

Comparisons between the code generated by the above three compilers were made in three areas, namely:

- Graphics
- Numerical computations using integers
- Floating point computations

'Identical' programs were written in all three languages, and to the best of the author's knowledge all compiler directives that could affect the execution speed of the generated code (such as range and stack overflow checking) were turned off. Listings of the respective routines can be found in Appendix D.

For the sake of brevity only the language name shall be used and this will be understood to refer to the particular implementations under test.

High resolution graphics

Pascal and C use essentially the same graphics routines, thus a comparison was only made between Pascal and Modula-2. One of the most time consuming sections of the data capture system is the plotting of ionograms, Doppler velocity, etc. to the screen in real time. Pascal/C claim high speed graphics by bypassing the BIOS and writing directly to video memory [Borland (Pascal) 1988]. Modula-2 makes no such claim and essentially had no library to support high resolution graphics (at that stage). At the time of performing these tests 'in-house' routines were being used by the DCS for plotting ionograms. These routines also wrote directly to video memory and were faster than Pascal routines for plotting vertical and horizontal lines. Translating these routines to Modula-2, and modifying the Modula-2 Graph module to enable 'hi-res' graphics, produced similar time results as their Pascal counterparts. Therefore only a comparison of the times for the standard Pascal and Pascal 'in-house' routines is made here. From Table [4-1] it can be seen that the advantages of using 'in-house' routines over the standard routines is overshadowed by the choice of computer hardware.

Tests were performed on a Colour Graphics Adapter (CGA) card in 'hi-res' mode (640x200) and on a Video Graphics Array (VGA) card (640x480).

	PC (CGA)	AT (CGA)	AT (VGA) CGAhi	AT (VGA) VGAhi
Pascal	47	17	8	28
'in-house'	30	22	5	

Table 4-1: Time (in seconds) required to address an entire 'hi-res' graphics screen pixel by pixel.

Integer Computations

In addition to the standard 2-byte integer type, Pascal also supports a 4-byte longint type. However, Pascal is slower than either of the other two (Table [4-2]).
Note: A co-processor does not affect integer calculations.

	PC	AT
C	22	10
Pascal	45	17
Modula-2	23	10

Table 4-2: Comparison of times for integer computations (in seconds)

Floating point computations

		PC	AT	AT (80287)
C	- float (4-bytes)	45	19	2,8
	- double (8)		20	3,2
Pascal	- real (6)	20	10	4,1
	- extended (10)			3,3
Modula-2	- real (4)	48	22	3,0
	- longreal (8)	51	23	3,3

Table 4-3: Comparison of times for floating point computations (in seconds)

Pascal's floating point emulation routines were substantially faster than those supplied with C and Modula-2.

4.3.6 Language decision

In the time available, the author was unable to gain sufficient familiarity with all three languages to a depth that would allow an objective assessment of each. The final decision therefore, was based on a subjective overview of the reference manuals, the opinions of programmers experienced in one or more of the languages, and the author's prior programming experience. As all three of the languages appeared to fill our basic requirements the overriding criteria became readability and reliability of the program code.

The author choose to discard C as an option due to its lack of readability and absence of enforced 'good' programming practices in comparison with Pascal and Modula-2, and this appears to be the consensus amoungst a growing number of programmers; "My own feeling is that the current C craze may wane over the next couple of years, leaving the language back in the hands of the converted" [McIndoe T 1989].

Another important aspect was maintenance of the code. While both Pascal and C are well known and established languages, Modula-2 is the primary language of computer education at present (and for the forseeable future) at Rhodes University, our most likely source of future programmers. This however does not totally preclude Pascal as an option. "Modula-2 is very similar to Pascal, at least at a superficial level, and Pascal programmers can read Modula-2 code very easily." [Terry PD 1987]

In the author's opinion, Modula-2 seems to be a unification of the most desirable features of Pascal and C. With its stronger typing and increased readability, it allows the development of even more reliable code than Pascal, while at the same time allowing the flexibility of C by giving the programmer total control of the hardware at a very low level. With its inherent multitasking capabilities it is better suited to real time applications than either Pascal or C. However, TopSpeed Modula-2 is still in its early stages of development, with regular updates being brought out to correct bugs and affect minor changes to the language. In this regard Borland had more or less stablised their Turbo Pascal software package. Also, the TopSpeed integrated development environment does not provide the advanced debugging facilities of Turbo Pascal and is considerably slower in compiling and linking programs.

The final choice came down to Turbo Pascal for the following reasons:

1. Readability.
2. The extensive amount of software already written in it for the DCS.
3. The speed and reliability of the development system and of the compiled code.
4. The speed of its floating point emulation library as coprocessors are at the moment prohibitively expensive.
5. The extensive graphics library.
6. With the introduction of units, Turbo Pascal has moved towards the Modula-2 concept — "Two aspects of Turbo Pascal make this modular functionality work: (1) Its tremendous speed in compiling and linking and (2) its ability to manage several code files simultaneously, such as a program and several units" [Borland (Pascal) 1988].

7. Finally, Borland have hinted that Turbo Pascal may at some future date be ported to other machines and operating systems, with the introduction of compiler defined constants such as MSDOS and CPU86.

Having not totally discarded Modula-2 as an option, and accepting its advantages over Pascal paticularly as regards the format of the source code, all software was developed with Modula-2 labelling conventions. Not only does this make the Pascal code more readable, but will ease the conversion of the software to Modula-2 should this be desirable at some future date.

All the software was later upgraded to Turbo Pascal version 5.5.

4.4 User Interface

The user interface forms the highest level of abstraction of the system and at this level the exact hardware configuration is invisible to the user.

Once the system is operating (**CONTROL.EXE**) all user input is via the keyboard. The entire shell is menu driven with function keys to select options. A tree type hierarchy has been adopted. This is not too cumbersome as the user never transcends further than two levels. The menu structure is shown in Figure [4-8].

The monitor screen has been divided into a number of windows, though none of them are demarcated by a distinct boundary. The screen is used in graphics mode throughout as this allows ionograms to be plotted at the outer level of the system. Text mode would be suitable for most screen operations but swapping between text and graphics mode has some drawbacks. Most importantly it would render the plotting of ionograms to an off-screen page as a background operation impossible, should this be required at a future date.

The PC/sounder system can be configured into a number of modes of operation. User selectable parameters associated with each of these modes are stored in records, and these define what are called *system operations.*

All system operations can be created and edited from within the editor, as well as schedules of operations to be run, and are stored on disk in the system file. All operator input is validated before being accepted and error reports are given. The editor also allows operations to be duplicated or deleted. Every time an operation is used by another

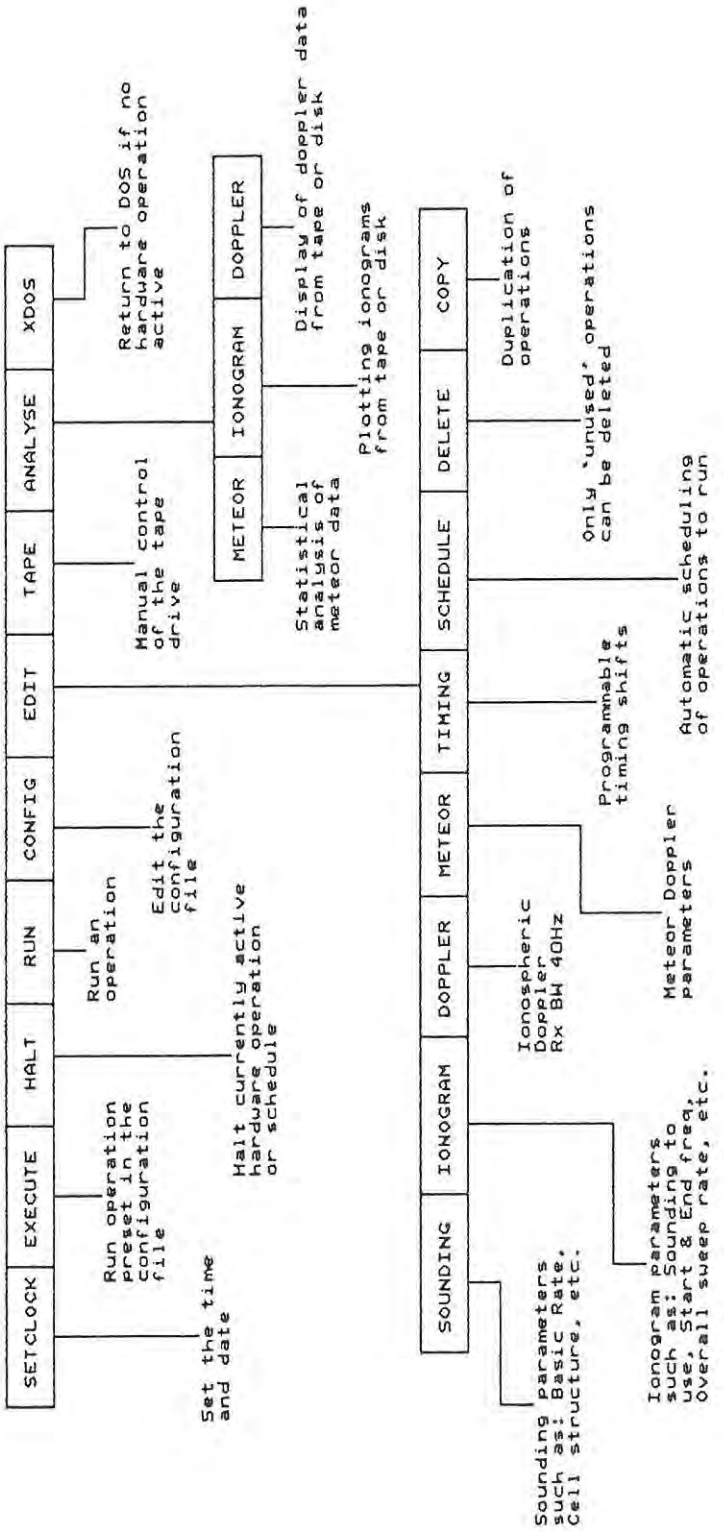


Figure 4-8: The hierarchial software menu structure

operation it is flagged and only operations not in use can be removed from the system file.

For an in-depth description of system operation, consult the *Operators Guide.*

4.4.1 Storage of system operations

There are six types of system operations and all information pertaining to an operation is stored in a system record with the following format:

Operation name : 8 characters maximum but commencing with an alpha character.

Type of operation : Sounding, Ionogram, Doppler, Meteor, Timing, Schedule.

User level : 0 if not used by any other system operations. Incremented by 1 every time the operation is used by another operation.

Operation details : An array of 21 records each containing 1 string and 4 integer values.

The first five operation types listed above are termed *hardware operations* in that they exercise some control over the system hardware. *Schedules* are purely software operations. System operations that are currently executing are termed *active* operations. Only one hardware operation can be active at any given time.
Note: *Soundings* are an exception to the general rules that apply to system operations in that they cannot be run directly. Soundings are merely building blocks of *Ionograms.*

Operations can be scheduled to execute on an hourly daily or weekly basis. The active schedule is scanned once a second while there is no hardware operation active. All hardware is initialised 1 second before the start of a hardware operation. The active schedule can also call another schedule which in turn becomes the active schedule. This operates on a co-process type principle in that control will not return to the first schedule unless the second schedule in turn calls the first.
System records are stored on disk in the system file (**CONTROL.SYS**) and loaded into memory when the control program executes. Changes to the system file are only updated on disk when no hardware operation is active, or when the program terminates. If the system file cannot be found on boot-up then the operator will be prompted to create one.

The first six system records in the system file are the standard operation specifications (Sstd, Istd, Dstd, Mstd, Tstd, Pstd). Whenever an edit is attempted on an operation

that does not exist it will automatically be created, provided there is space remaining in the system file, and will default to the appropriate standard operation parameters. These standard operations can be edited but not deleted.
Note: No two operations can be entered with the same name.

Another file read by the control program during initialisation is the configuration file (**CONTROL.CFG**) which stores basic system information such as station location, boot-up operation, etc. Again the operator is prompted to create this file if it cannot be located during initialisation.

4.4.2 Data storage to disk

Meteor data

Data are stored to the hard disk in a file with the filename:

[ppp][yy][ddd].MET

where [ppp] are the first three letters of the station name, [yy] the year and [ddd] the day number. The data associated with each echo, obtained as described in section 4.1.2, are stored in a *meteor record.* Each record consists of 38 bytes which contain the time, followed by data for each of the four effective channels, in the following format:

Time : Three word values (hour)(minute)(second) – 6 bytes.

Spectral maxima frequencies (f_{max}) : Four byte values (L1)(L2)(L3)(L4) – 4 bytes.

Real & imaginary components (f_{ave}) : Eight integers representing four complex numbers (R1)(I1)(R2)(I2)(R3)(I3)(R4)(I4) – 16 bytes.

Frequency spread : Four byte values (S1)(S2)(S3)(S4) – 4 bytes.

Sample maxima : Four words (M1)(M2)(M3)(M4) – 8 bytes.

The *time* recorded is the PC's time at the end of the echo time cell. *Spectral maxima frequencies* represent the points on the spectrum at which the maximum for each channel occurred (f_{max}), and are accurate to the nearest 0.5 Hz (ie. a value of 50 represents a frequency of 25 Hz). The *real & imaginary components* refer to the spectra of each of the four channels at the average frequency f_{ave}. *Frequency spread* is determined by the number of successive spectral points on either side of f_{max} for which the magnitude of the

spectrum is above the spread threshold (see Figure [4-6]). Time domain *sample maxima* are absolute values and lie in the range 0 to 32767.

On commencement of a meteor observation the current day's data file will be appended to, unless one does not exist in which case it will be created. The first record appended to a file, and the first record of a newly created file will be a *parameter record.* These parameters will apply to all meteor records stored thereafter until another parameter record is encountered or the end of the file is reached. Parameter records have the same format as meteor records but are identified by an hour value of 99. The manner in which parameters are stored in the record fields is as follows:

- Hour – 99
- Minute – century + year
- Second – day number
- R1 – echo criterion (Hz)
- R2 – spread threshold
- R3 – number of samples per FFT
- R4 – Rx attenuation (dB's)
- L4 – number of PC software filters
- L1 – filter 1 start frequency (Hz) - inclusive
  S1 – end frequency (Hz) - inclusive
  I1 – threshold
- L2 – filter 2 start frequency
  S2 – end frequency
  I2 – threshold
- L3, S3, I3 – filter 3
- S4 – digital filter reference number
- I4, M1, M2, M3, M4 – zero

Ionospheric data

Data are written to both disk and tape in the same format. Data are output as strings of 18 characters in length. To save space, most of the data are converted to hexidecimal and stored as ASCII strings. At the start of an ionogram a number of global parameters are immediately saved.

place ◇◇ century year ◇ day no ◇ hour : minute : second ◇

Rate ◇ = ◇ basic rate ◇ kHz/s ◇

Cells/Sounding: ◇ no cells per sounding

Cl ◇ = ◇ 1/32 ◇ x ◇ 2^ cell length ◇ s ◇

FFT ◇ Rate ◇ = ◇ 2^ sample rate ◇ Hz

Offset ◇ = ◇ Tx offset freq ◇ Hz ◇

Shift: ◇ time shift ◇ ms

Items underlined are data entries and ◇ represents the space character (ASCII 20h). Thereafter at the end of every sounding the following data are written to tape.

sounding start freq ◇ antennas cell no sounding no Rx attenuation

spectral freq amplitude A real B real A imag B imag

repeated to a possible maximum of PPC points per cell.

Data are stored to the hard disk in a file with the filename:

[pp][yy][hh][mm].[ddd]

where [pp] are the first two letters of the station name, [yy] the year, [hh] hour, [mm] minute, and [ddd] the day number.

4.5 Software description

All software development has been done using Borland's Turbo Pascal version 5.5. Due to the memory segmentation of Intel's 80x86 microprocessor architecture, the largest contiguous segment of code that can be handled is 64 kbytes. To overcome this limitation Borland has introduced units. An absolute limit on system memory of 640 kbytes exists, which is imposed by DOS. Turbo Pascal makes use of any expanded memory, into which it loads the file being edited. This releases an additional 64 kbytes of memory for executable

code while the compiler and integrated environment are memory resident. Editor files cannot exceed 64 kbytes in size.

4.5.1 Using units

Units are pre-compiled libraries that can be used by the main program. Units are linked at compile time, and any units that have been modified since the last compile will automatically be re-compiled before inclusion with the main program. A unit has two parts; an *interface*, and an *implementation* section. All types, variables, procedures, etc. declared in the interface section are available for use by the main program. Items declared in the implementation section are hidden, which provides a further level of abstraction to a Pascal program. Declarations in the interface section are compiled using the far call model (ie. segment:offset) and can be loaded into any available memory segment.

Units raise a number of interesting possibilities as far as the variety of hardware devices, such as signal processing cards and ionosondes, are concerned. It is possible to associate with each of these devices a unique unit specific to that device, and if care is taken to standardise the interface section (for example of the DSP units) then the exact card in use would be invisible to the control program. In addition any new DSP card purchased would simply have to have its driver software included in a unit which conforms to the standard, for inclusion in the system with no modifications to the main program. All the software has been developed with this in mind, and the author has attempted to remove all device specific functions from the main program. If this modular approach can be applied with sufficient rigour, then it should be possible to simply include the Data Capture System (written independently by Mr BT Bonnevie) as a separate unit(s). To this end the author has worked in conjunction with Mr Bonnevie, but at the time of writing this thesis, the DCS had not been included. An overview of the units that are in use follows.

Globals : This unit contains all the global constant, type, and and variable declarations that are common to more than one unit.

Util : All the general purpose routines such as writing integers and strings to the display, changing windows (viewports), etc.

Core : This contains the fundamental routines of the control program. It uses most of the remaining units in its interface section and is in turn used by most of these units in there implementation section. These circular unit references were required to overcome reentrancy problems associated with DOS (sections 4.2 & 4.5.3). The program sits in a loop testing status flags set in interrupt service routines while waiting for input from the keyboard.

Timer : Contains the interface software for the Timing Controller. If the TC is not present then it will use the PC's real time clock.

Dsp16Tp : Driver software for the DSP-16 as supplied by Ariel is included in this unit. The DSP-16 will automatically be initialised after this unit has loaded.

Br9034 : This unit configures the serial port to allow data to be received from the 9034 on an interrupt basis. It also sets a number of control lines in the parallel port.

Config : Firstly this contains a general setup routine which executes only after all the software has been loaded. It also allows editing of the configuration file.

Edit : Does not contain the line editor but uses it in the editing of all system operations. The updated system file is queued and written to disk by the Core unit.

Analyse : At the time of writing, this unit contained only routines to perform statistical analysis and graphical display of meteor data. Statistical results for each day are stored on disk with the file name: (ppp)(yy)(ddd).DAT

Program operation has been well documented in the source code which can be found in the separate text *Source code listings for Control of a BR-9034 chirpsounder, and Data Capture using a TMS320 signal processor*, Rhodes University, 1990.

4.5.2 System initialisation

When **CONTROL.EXE** is run the software automatically detects what hardware is present and initialises the system accordingly. The hardware checked is as follows:

- Timing Controller (TC) — If this card is present then it also warns of any possible timing errors that may have occurred due to a loss of input from the external

frequency (time) standard. If the Timing Controller is not found then the PC's real time clock will be used, and *Meteor* operations will be the only type of hardware operation that can be run.

- Signal processing card (DSP-16) — The TMS320 code (**DSP16TMS.HEX**) is down-loaded and the sample rate timers set as part of this cards initialisation routine. If it does not initialise properly then the sounder control software will still function but there will be no data capture system operation.

- Sounder (BR-9034) — The PC's serial port is configured for communication with the sounder and if this serial link is established the sounder is assumed to be on-line. If the sounder is off-line no commands will be sent to it but the data capture system will operate as normal (obviously no useful data can be recorded but this feature assists with system testing). While hardware operations are active status reports are expected from the sounder. If these are not received in the time allowed, then the sounder will be reset and in some instances re-started.

- Graphics card — At the time of writing this thesis, the software was compatible with only CGA (or EGA/VGA in CGA mode), but it would be preferable to detect which type of graphics system is present and configure screen output accordingly.

- Tape driver — At the time of writing, no software tape driver had been implemented but detection that the tape driver is on-line would be required.

After initialisation of the hardware the System and Configuration files are loaded from disk into memory. Any boot-up operations specified in the configuration file will immediately begin execution. During initialisation the system status and error conditions are reported. Fatal errors will prompt the operator for corrective action, or alternatively terminate the program.

4.5.3 Handling interrupts

In order to insure that time critical routines are executed immediately, this code has been associated with hardware interrupt lines. The control program intercepts three of the PC's interrupts for its own purposes (see Table [N-2]). Hardware interrupts are mapped to software addresses in the PC's interrupt vector table (Figure [I-3]), and interrupt

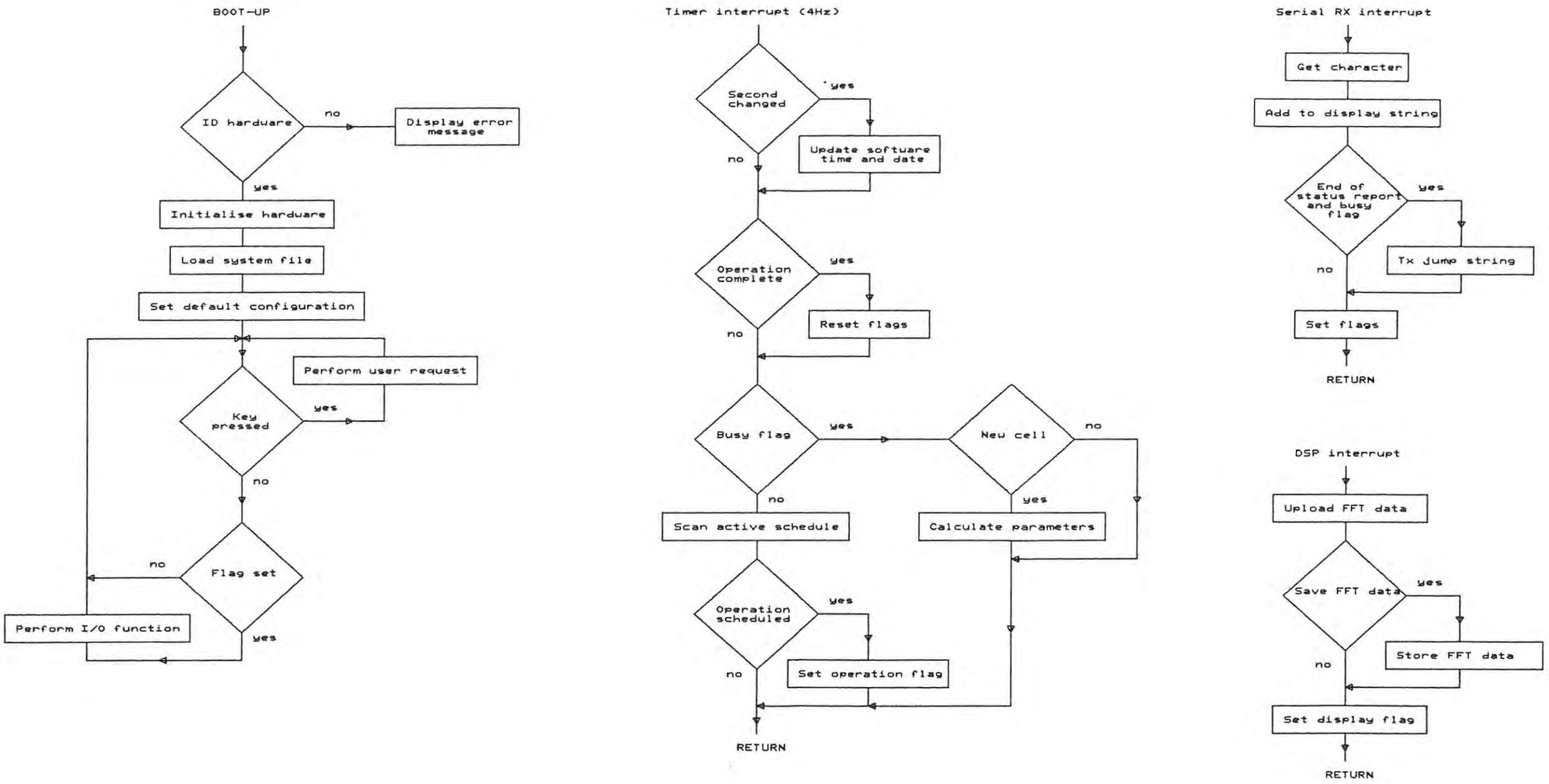


Figure 4-9: Simplified flow diagram of the controller software

requests (IRQ) are prioritised by the Interrupt Controller (PIC) chip. The AT has two interrupt controllers[3] which are prioritised as shown in Figure [I-1].

To avoid the reentrancy problems associated with DOS, no screen output is performed from within interrupt service routines. Instead a system of flags is employed to allow this output to be performed sequentially. The 'core' routine of the control program is a loop which continuously polls each of these flags in turn, and if any have changed status, performs the necessary screen updates.

This proved a satisfactory solution for screen output, as this is generally slow and not critical to system performance. In fact the screen has the lowest priority and is only updated when there is sufficient time to do so. This approach could not be used with disk I/O as this must be performed within interrupt routines, as well as being required by the main program. The solution to this was to queue disk operations required by the main program while the disk may be needed by an interrupt routine. Conversely, interrupts are disabled when the main program is using disk access. The DSP interrupt service routine is the only interrupt routine that has access to the disk.

A simplified flow diagram of the controller software is given in Figure [4-9].

4.5.4 Programing the Timing Controller

The Timing Controller is programmed via ports 300h though 31Fh on the PC. A detailed description of port allocation is given in Table [K-2]. Of particular importance are the control and status registers. Setting the RUN bit in the control register (Figure [4-10]) produces a start strobe on the next 1 second edge which sets the ACTIVE bit in the status register (Figure [4-11]). Thereafter a jump strobe will be produced at the start of every cell. Resetting the RUN bit will cause all strobes to halt at the end of the present cell. If IRQ ENABLE in the control register is not set then the 4 Hz interrupt generated by the TC will be inhibited. This bit is reset when the PC is reset. During initialisation, toggling the status of the ADV/RTD bit allows the PC to detect if the TC hardware is present.

The IRQ bit in the status register is checked to confirm that the interrupt was gen-

[3]The interrupt service routine is required to send an unconditional end-of-interrupt to the PIC, before this chip will accept further interrupts on the same IRQ line. If the interrupt was generated by IRQ 8 through IRQ 15 then an end of interrupt must be sent to both PIC's.

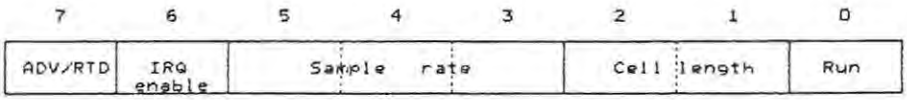


Figure 4-10: TC control register

erated by the TC and not some other device using the same IRQ line. FREQ FAIL is set if the input from the external frequency (time) standard has been lost at some stage. Reading the status register resets this bit. XTAL OSC is set whenever the TC is using its on-board oscillator in the absence of the frequency standard.

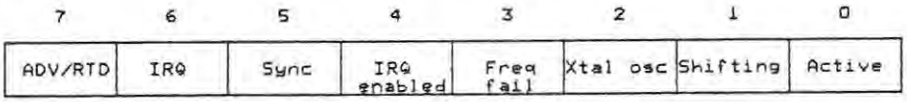


Figure 4-11: TC status register

Whenever the real time clock on the TC is set, the hardware 1 second signal must be synchronised with the RTC's internal 1 second. This is achieved through some hardware trickery by reading the RTC's interrupt status (port 310h). A detailed discussion on programming the Timing Controller is given in Appendix K.

4.5.5 Controlling the sounder

The BR-9034 sounder is programmed via a serial link to the PC. Only the format of commands used will be given here, although some additional commands and variations thereof exist. All command strings are terminated by a Carriage Return (CR).

MxRxBxFabcdefgCxGxxVabcdefAx : This is the start command string.

Mx - Tx, Rx, T/R & Tx+Rx mode

Rx - 25, 50, 100, 200 kHz/s basic rate

Bx - 500, 2500, 40 Hz Rx bandwidth

Fabcdefg - start frequency 1.60000 to 29.99999 MHz

Cx - 0.25, 0.5, 1, 2 second cell length

Gxx - Rx attenuation 0 to 114 dB's

Vabcdef - Tx offset frequency 0.000 to ±19.999 kHz

Ax - status reports on/off

JabcdefGxxD : Jump command string

Jabcdef – change frequency by 0.00 to ±999.99 kHz

Gxx – Rx attenuation

D – jump on hardware strobe (I jump on CR)

M0 : Terminate command

The formats of the status reports received from the 9034 are as follows:

XXXgYYZ :

XXX is the averaged IF-gain over the cell from Rx1

YY the receiver attenuation

Z this indicates an error condition

(P – RFPA failure, L – synthesizer out of lock)

E – command input error

T – sounder in standby mode

In the case of 0,25 second cells the status is returned 50 ms before the end of the cell. In the other three cases the status is returned 100 ms before the end. Initially the author tried transmitting jump commands a quarter of a second before the end of a cell (ie. immediately after a jump strobe in the case of 0,25 second cells). However the BR-9034 will not return a status report once it has received a jump command string. These commands are now transmitted immediately after receipt of the status report, as illustrated in Figure [4-12].

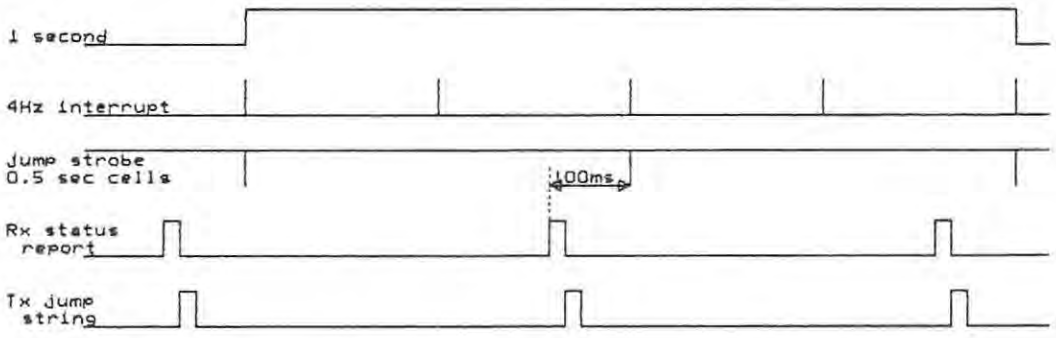


Figure 4-12: Timing diagram for transmission of the jump command string to the BR-9034

External strobing of the 9034 is required to perform structured frequency sweeps with suitable timing accuracy. Two strobes are required; a start strobe to begin sounder operation, and a jump strobe to effect step changes in frequency as well as receiver attenuation. Both these strobes are generated by the Timing Controller. [BR Communications 1984]

Chapter 5

System Testing

5.1 Phantom meteor echoes

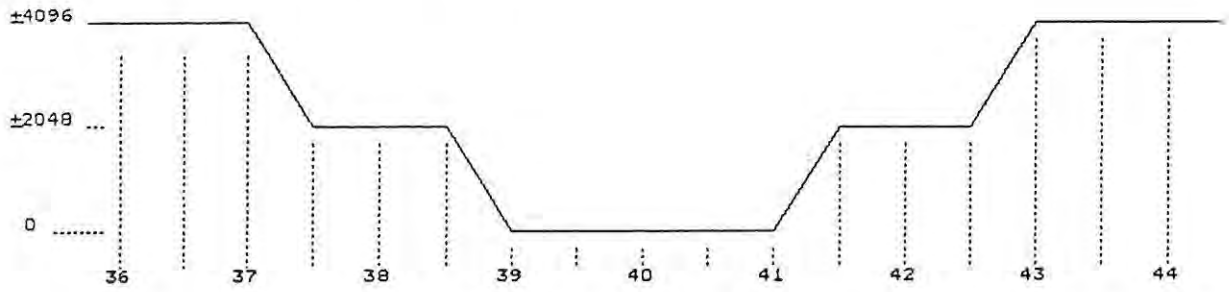


Figure 5-1: DSP-16 digital filter

Initial testing of the new meteor logging system showed clearly that some form of filtering was required around 40 Hz (transmitted frequency offset) to reduce the registration of echoes caused by 'breakthrough'.[1] It was decided to implement a digital band-stop filter in software on the digital signal processing card. This filter could initially only be adjusted by altering values in a look-up table in the TMS320 program itself and re-assembling this code. This approach proved too inflexible and was later modified to allow down-loading of new filter look-up tables from files on disk. These files require a .TBL extension and are now down-loaded to the DSP-16 immediately before the start of a meteor operation.

[1]This is a phenomenon that arises due to ionospheric conditions during peaks in the solar cycle. Historically the term arose as it was initially thought to be caused by transmitter power 'leaking' during the receive cycle.

All results given below were obtained using a filter of the shape shown in Figure [5-1].[2] The use of this filter allows for the detection of echoes with an amplitude less than that of any possible 'breakthrough', as processing of the data only occurs after filtering.

In addition the use of a look-up table allows for correction of the filter response of the external 100 Hz hardware low-pass filters, and this is the default look-up table. *It is the adjusted real and imaginary components that are up-loaded to the PC.*
Warning: Very little error checking is performed on these look-up tables before down-loading them to the signal processing card. The onus is on the user to insure that these files are correctly formatted. A reference number (0–255) is used to identify the filter in meteor parameter records. Thereafter 81 positive integer values are required. Multiplication of a point on the frequency spectrum by a weighting factor of 4096 (2^{12}) results in an effective gain of 1, which should not be exceeded. Comments in .TBL files must be preceded by a semicolon.

On occasions the new meteor logging system with its frequency discrimination criterion, logged continuously at some fixed frequency. Some of these phantom 'echoes' were traced to RF interference from the nearby ionosonde. The solution to this problem was to avoid using the VOS-1 ionosonde in fixed frequency (doppler) or stationary ionogram modes at those frequency bands which caused interference.

A periodic weak 'echo' at 17 Hz was traced to the VOS-1 data capture system PC. The offending frequency component was found to only be present when the DCS was waiting in record mode between sweeps. The exact cause was never traced, but was not of immediate importance as it fell outside the band of interest for meteor echoes (20–60 Hz) and was thus filtered out.

Interference from other sources has proved to be a real problem. One solution is to remove these frequencies with a suitable digital filter on the signal processing card. A secondary solution was to implement PC based software filters. Broad band filters are used to remove small amplitude echoes, and narrow band stop filters for specific spectral components.

[2]The shape of this filter is not ideal and was later found to have certain adverse effects due to the large step changes in amplitude.

5.2 Comparison of data from the old and new meteor logging systems

The new meteor logging system became operational at the end of November 1989 and ran continuously during December 89 and January 90 with a minimum of data loss. An echo criterion of 2 Hz was used throughout. Running in parallel was the old meteor logging system with the amplitude threshold set at 30 on a scale of 0 to 127 (see Appendix B).

All post processing of the data was performed by Prof G Poole on a VAX computer. Meteor files are transferred to the VAX as binary data and then decoded as Fortran records using routines written by Prof G Poole & Mr J Jonas.

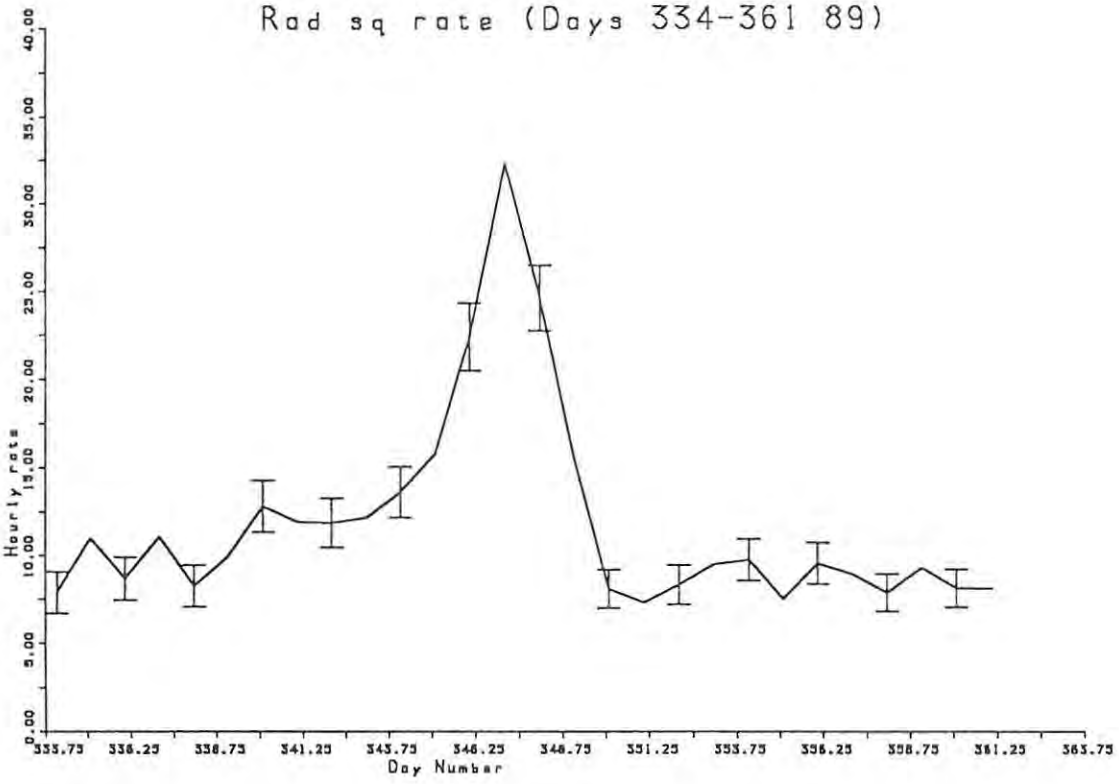
Of particular interest during December was the Geminid meteor shower[3] which peaked on 13 December (Day no 346). Shown in Figure [5-2] are the meteor rates for the old and new logging systems respectively, for a period of 28 days centered on the above date. As anticipated the rate of echoes is generally higher on the new logging system than on the old. The scales of both graphs plotted are the same, and it can be seen that on the 13th the rates were similar. The reason for this is not yet known but may be due to the increase in the ratio of large meteoroids to total number of meteoroids, near the center of the meteor stream. Also unexplained is the less gradual decrease in meteor rate shown by the new logging system on the 14th.

A contour map of radiant activity in the region of sky about the shower radiant is shown in Figure [5-3] for December 13th. As can be seen the shower radiants as given by the two logging systems agree closely (well within the approximate 2° uncertainty of the overall meteor radar system). In addition the general shape of the shower radiant as given by the two systems is similar. These shapes would be better defined had the shower rate been higher.

Another phenomenon of interest is the 2 day periodicity which has been very noticeable in upper atmosphere winds during January in previous years. However, neither the old nor new meteor logging systems showed a particularly large component at this frequency in 1990, although other expected periodicities (diurnal and semi-diurnal) were clearly present (Figures [5-4] & [5-5]).

[3]Meteor showers are generally named after the constellation in which the radiant is found. ('Radiant' defined in Appendix B.)

(a)



(b)

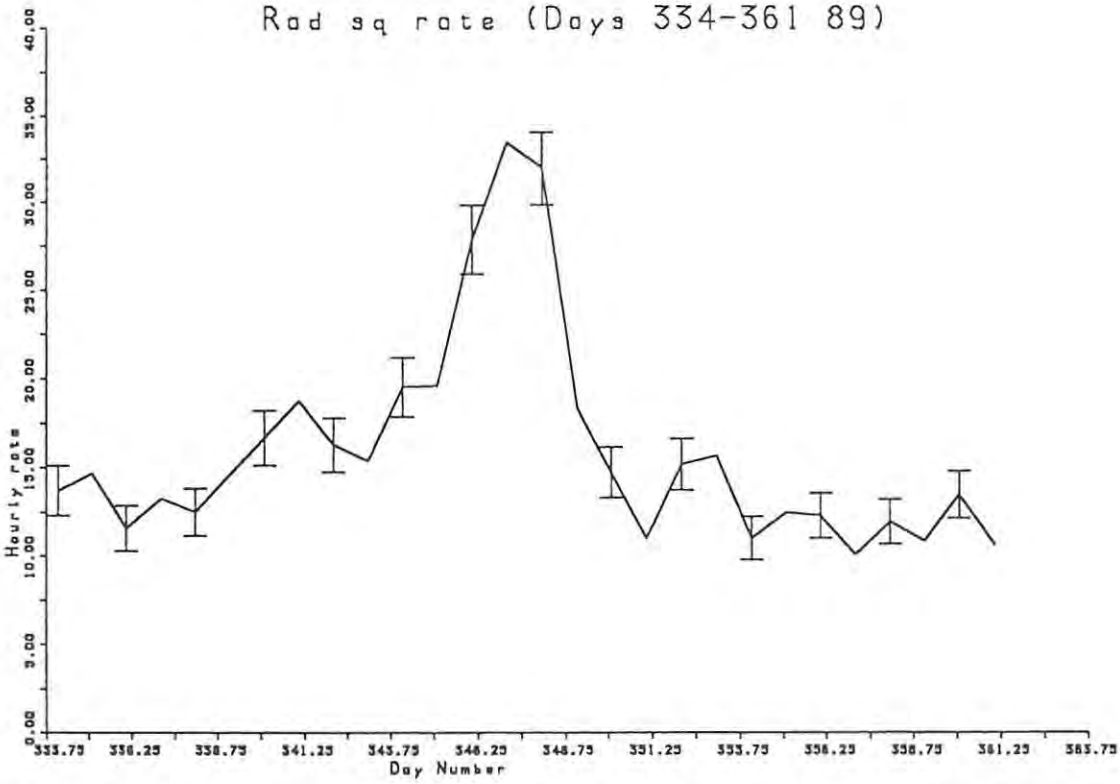
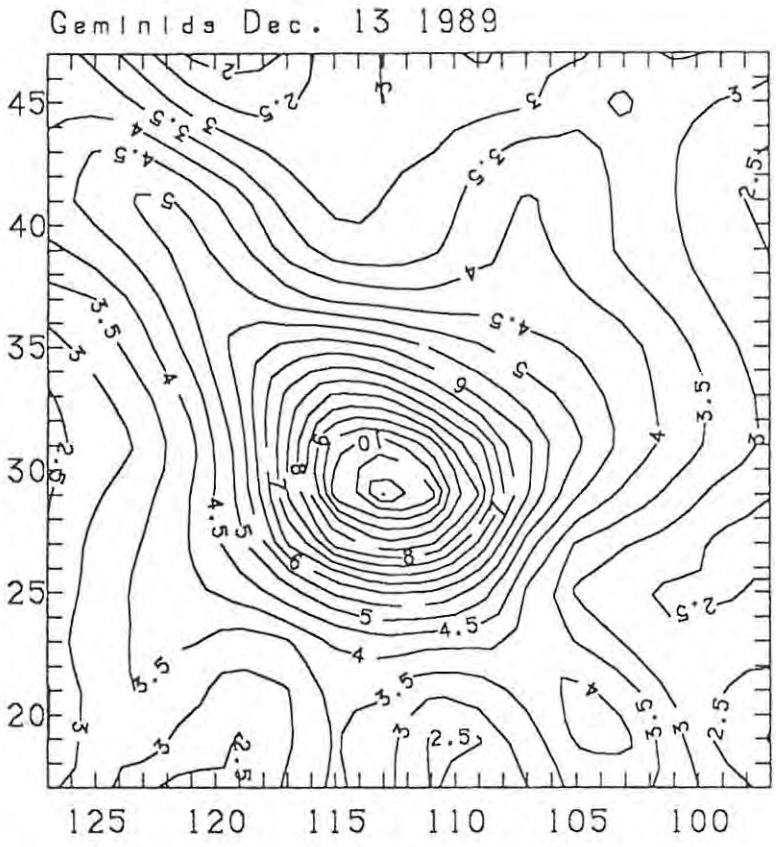


Figure 5-2: Meteor rates for the 10° by 10° region of sky about the Geminid shower radiant as recorded by (a) the old and (b) the new meteor logging systems (Prof G Poole)

(a)



(b)

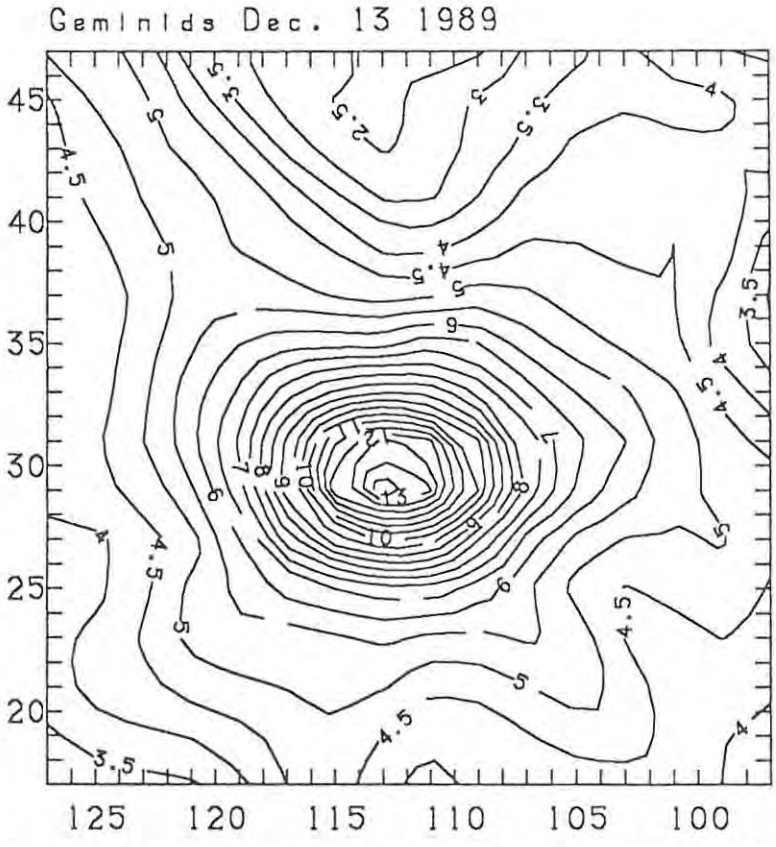


Figure 5-3: Contour map of the Geminid shower radiant as given by (a) the old and (b) the new meteor logging systems (Prof G Poole)

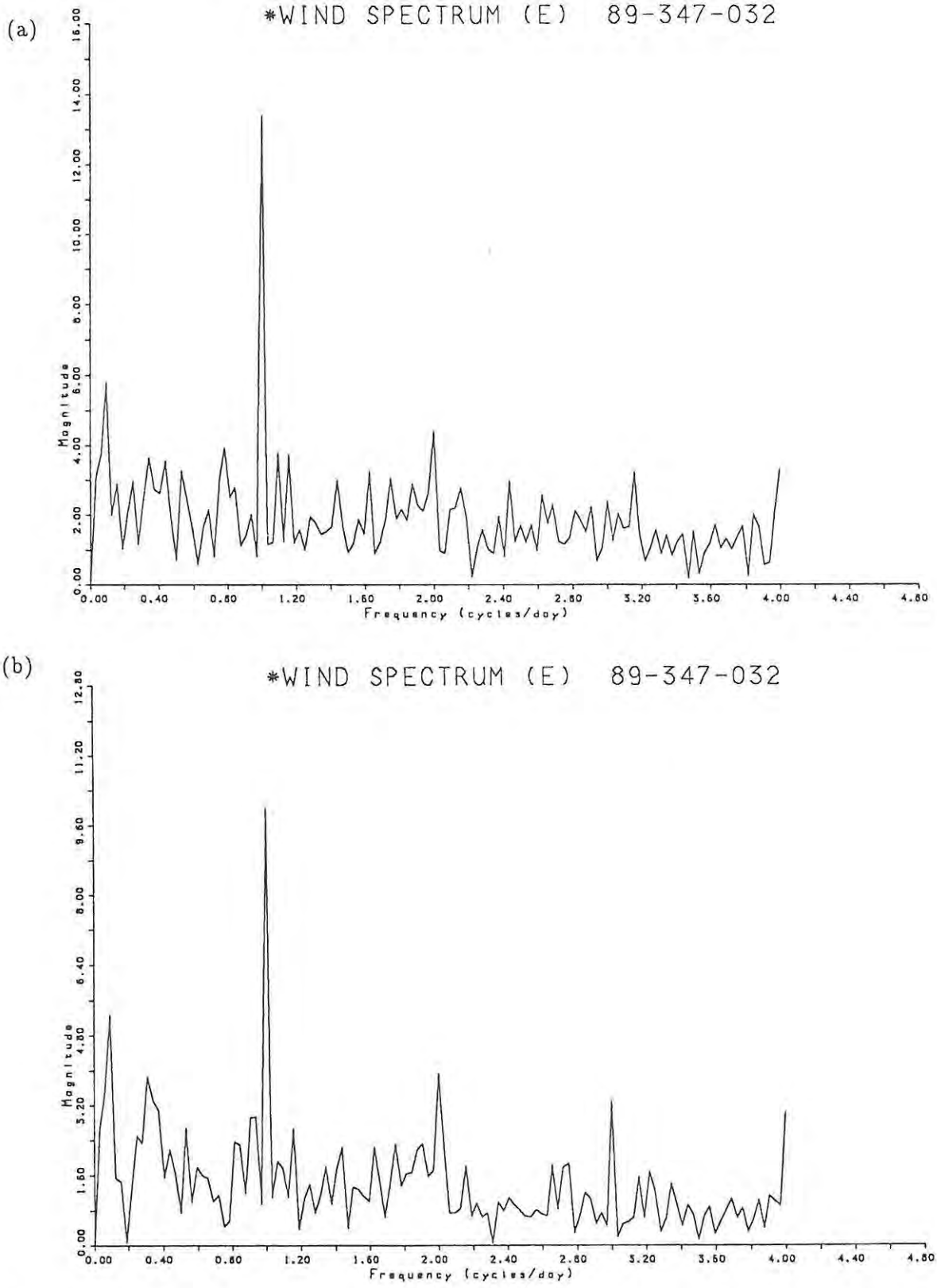
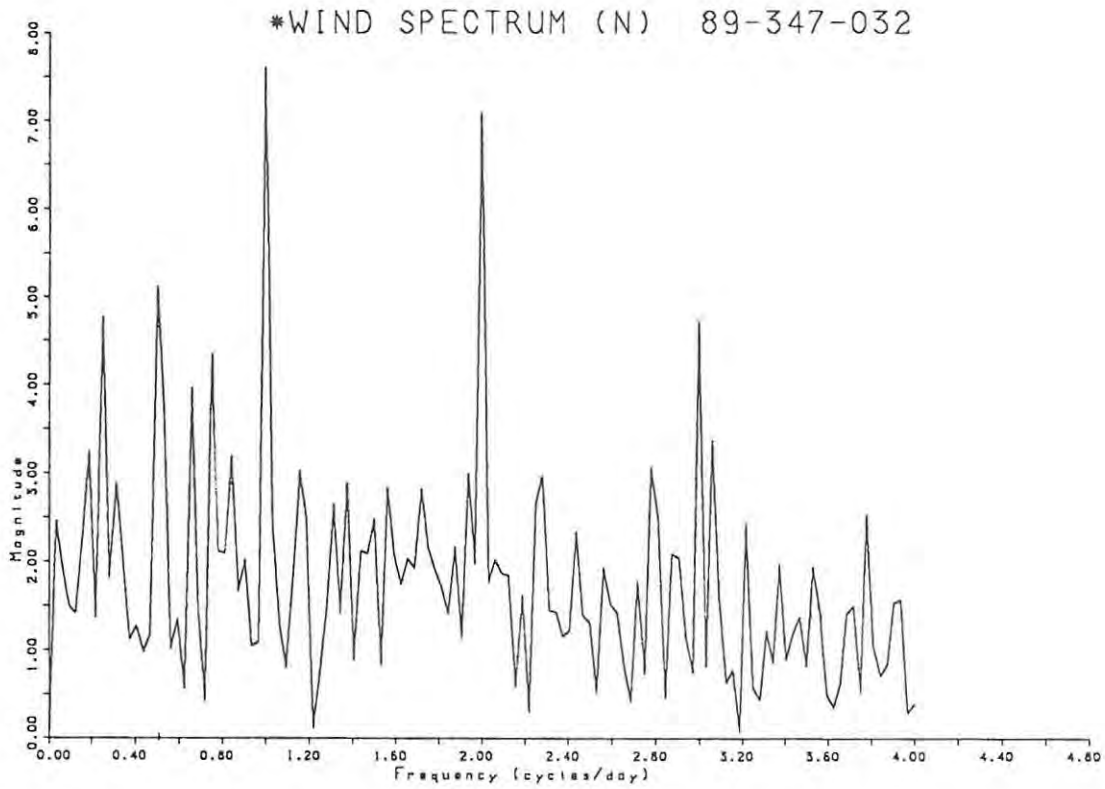


Figure 5-4: Comparison of upper atmosphere wind data (E - W) recorded by (a) the old and (b) the new meteor logging systems (Prof G Poole)

(a)



(b)

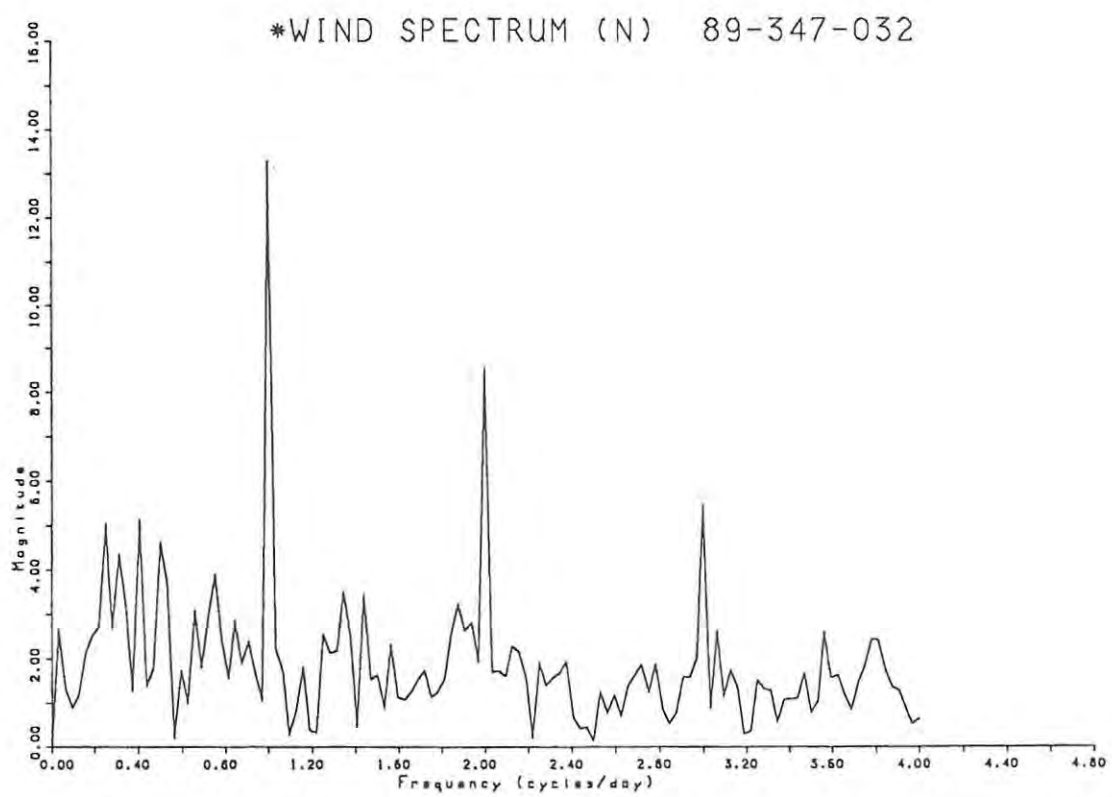
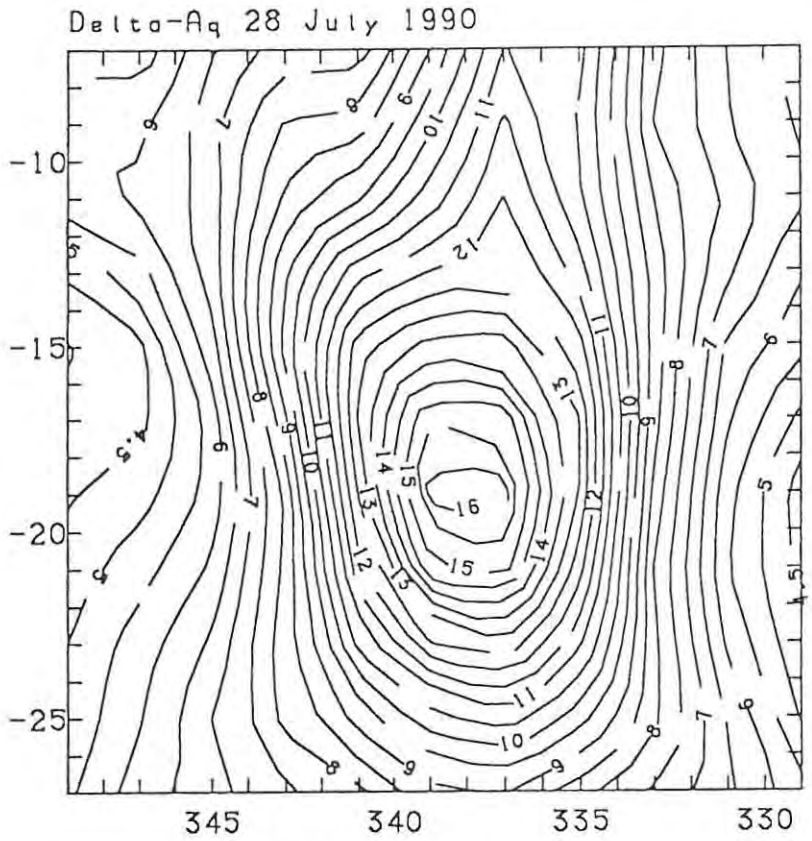


Figure 5-5: Comparison of upper atmosphere wind data (N - S) recorded by (a) the old and (b) the new meteor logging systems (Prof G Poole)

(a)



(b)

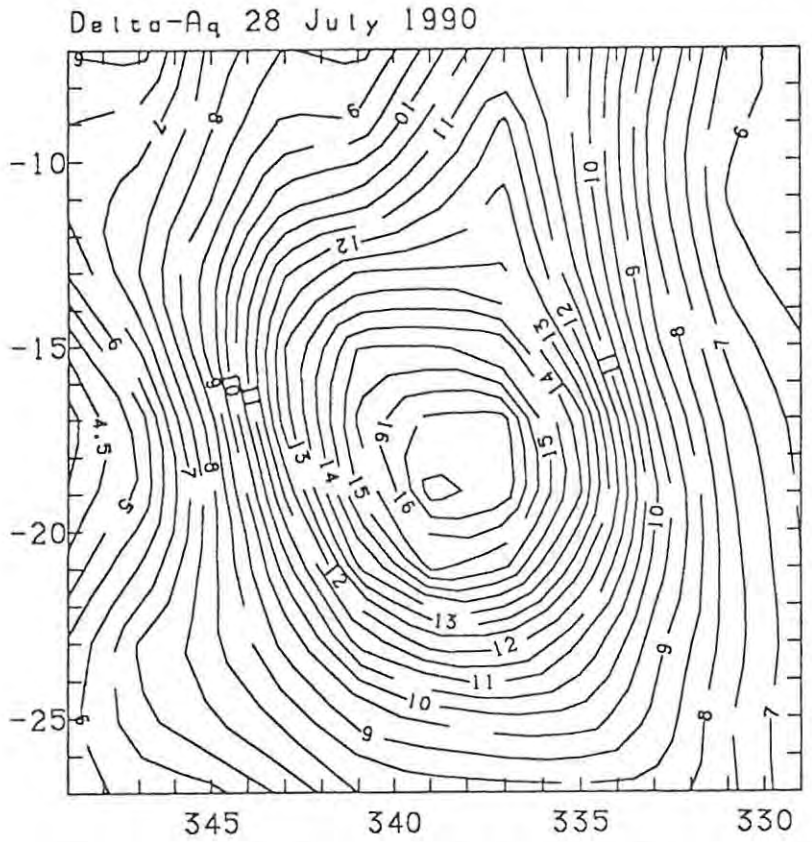


Figure 5-6: Contour map of the Delta-Aquarids shower radiant as given by (a) the old and (b) the new meteor logging systems (Prof G Poole)

Further comparisons between the two logging systems were made during the Eta-Aquarids in May 1990 and the Delta-Aquarids in July 1990. Radiant maps of the latter are given in Figure [5-6].

A weak day time shower, the Arietids, occurs in early June and was successfully logged by the new system. Figures [5-7] through [5-9] show the development and demise of this shower radiant over the period 3 to 17 June, peaking on the 9/10th.

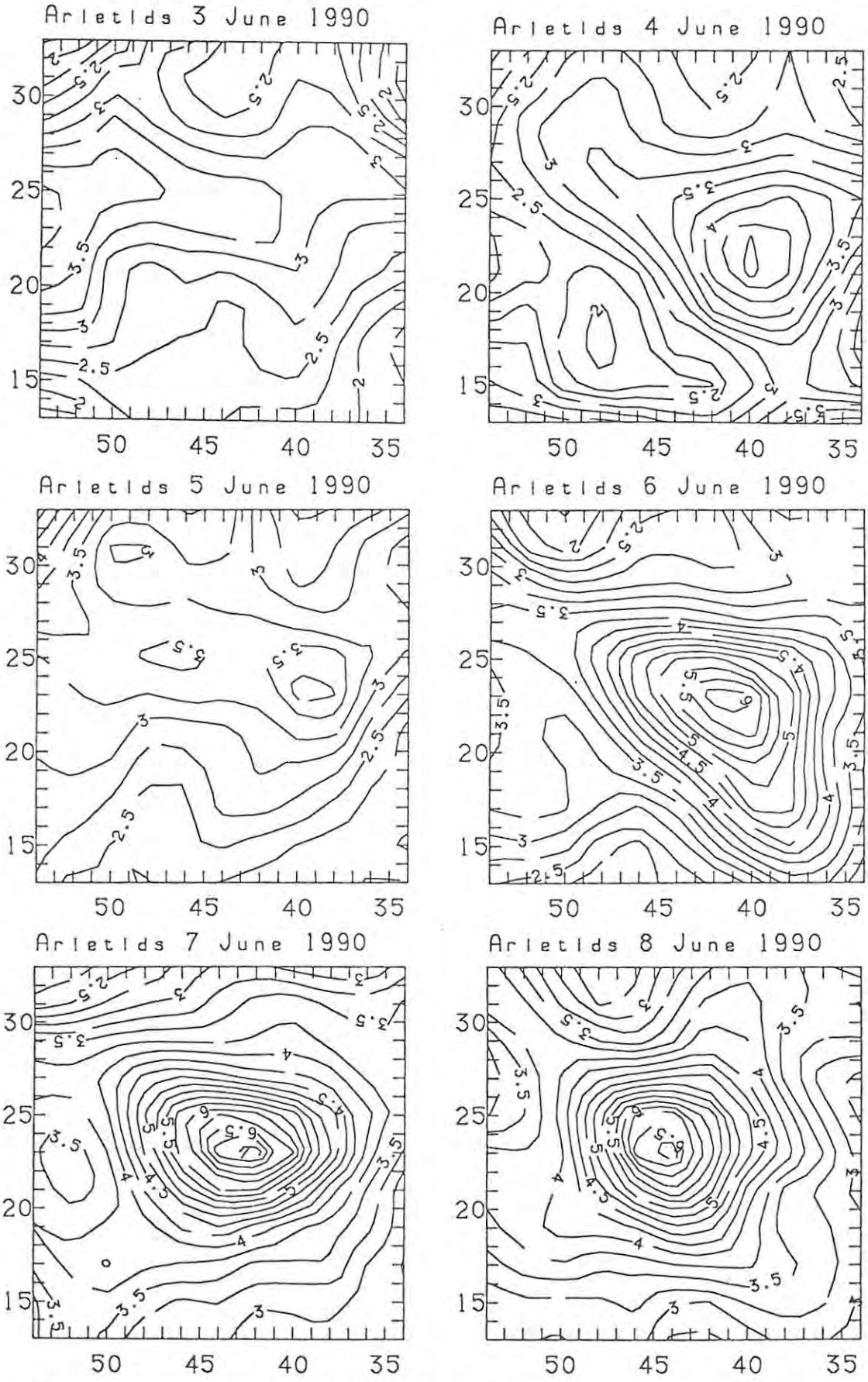


Figure 5-7: Development of the Arietid shower radiant - part 1 (Prof G Poole)

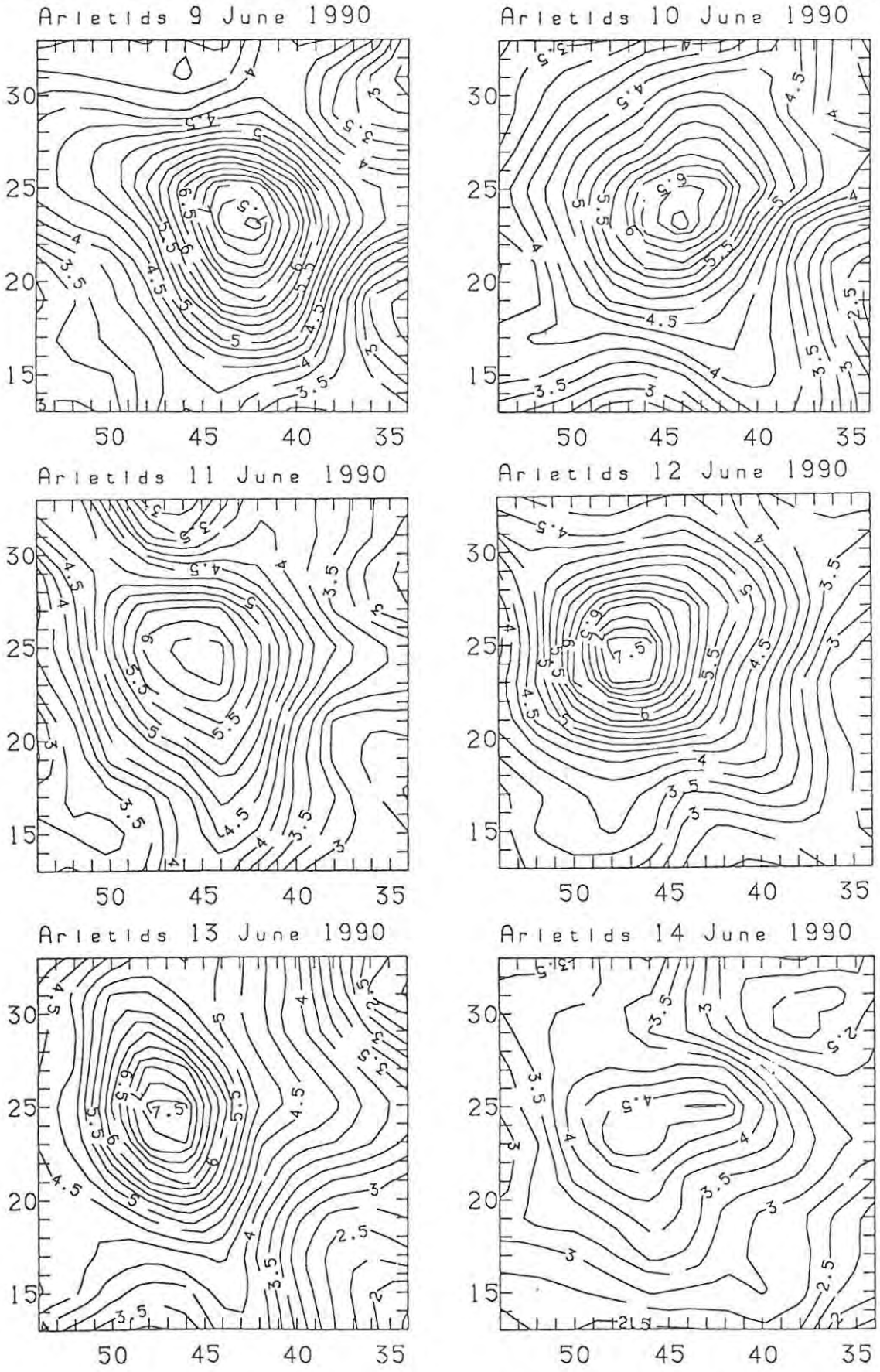


Figure 5-8: Development of the Arietid shower radiant - part 2 (Prof G Poole)

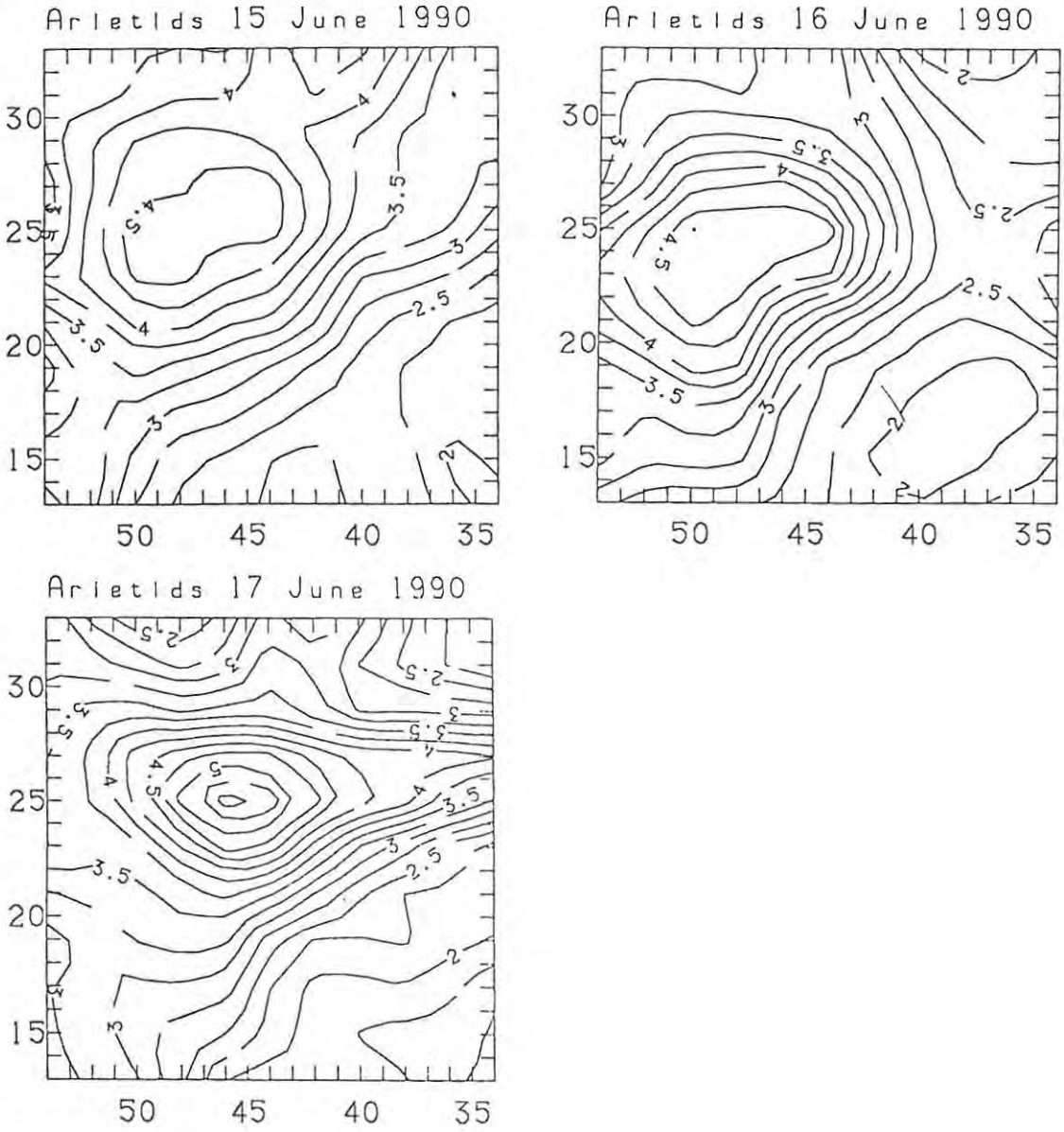


Figure 5-9: Development of the Arietid shower radiant - part 3 (Prof G Poole)

5.3 Performance of the BR-9034 as an ionosonde

At the time of writing this thesis the Signal Switching unit (SS) had not been constructed and without the necessary antenna switching circuitry multi-cell soundings are pointless. Thus all data has been recorded using single cell sounding structures and no phase calculations could be made. Conventional ionograms (group height versus frequency) displayed on the monitors of both systems, and recorded immediately after one another using the same antennas, compare favourably as shown in Figure [5-10]. These comparisons were made using a 1 second cell length, 50 kHz/s basic and overall rate, no window offset and a sample rate of 1024 Hz.

All options such as adjusting the basic rate, cell length, sample rate and window offset height have been tested. Figures [5-11] & [5-12]) illustrate the effect on vertical resolution of increasing the basic sweep rate in conjunction with a window offset.

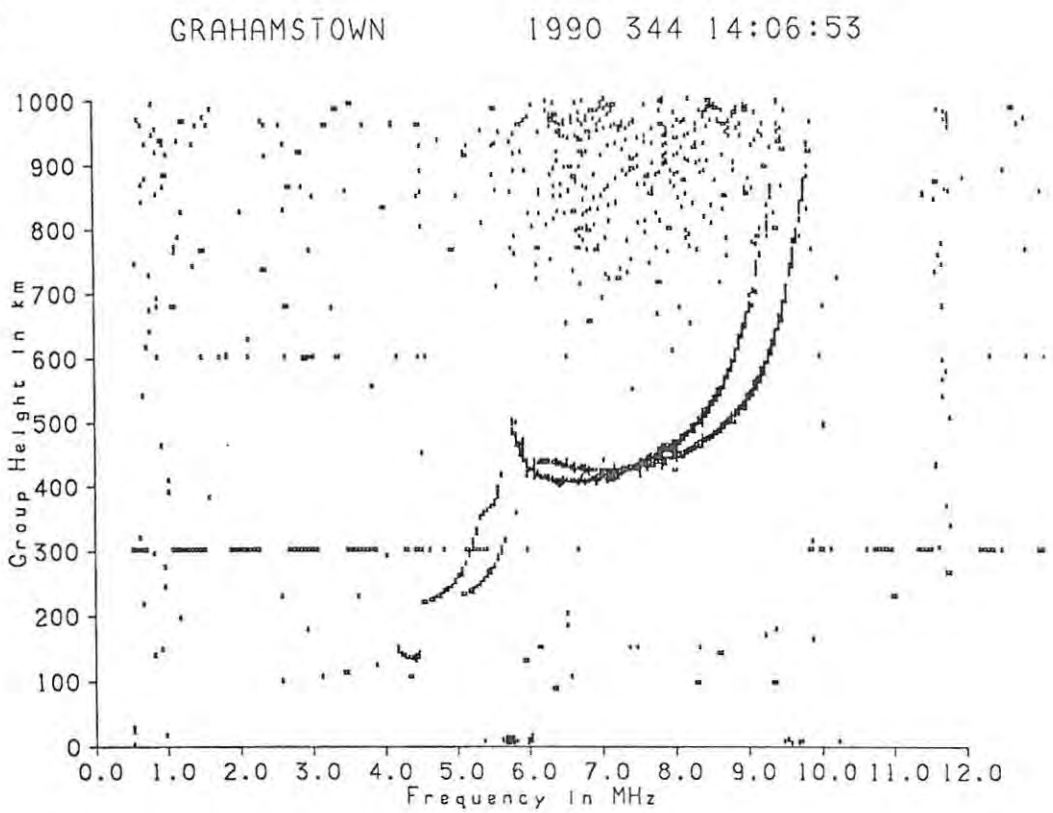
A variety of overall rates, both linear and logarithmic, are allowed, including zero overall rate for stationary ionograms. Figure [5-13] gives a comparison of two linear rates while Figure [5-14] compares two ionograms recorded using logarithmic rates. The number of points per cell to be recorded is selectable from 1 to 99, and a minimum relative amplitude threshold can also be set. Figure [5-15] demonstrates the usefulness of this criterion, particularly when the savings in storage space are considered.

Tests performed using the BR-9034 in Doppler mode showed promising results on the monitor but none of this data has been recorded. In Doppler mode the receiver bandwidth is set at 40 Hz and a sample rate of 128 Hz is used.

The Real Time Clock on the Timing Controller card (or PC if this card is not present) can be set from within the user interface, and small adjustments to the time can be effected using the TC. Effecting timing shifts during sounder operation by adjusting frequency jumps has not been tested.

A complex arrangement of schedules was successfully implemented on a trial basis. One second is required between operations to allow for configuration of the system hardware. Operations which have been specified with a time duration are shortened by a few seconds to accommodate this hardware setup time, but this is generally acceptable.

(a)



(b)

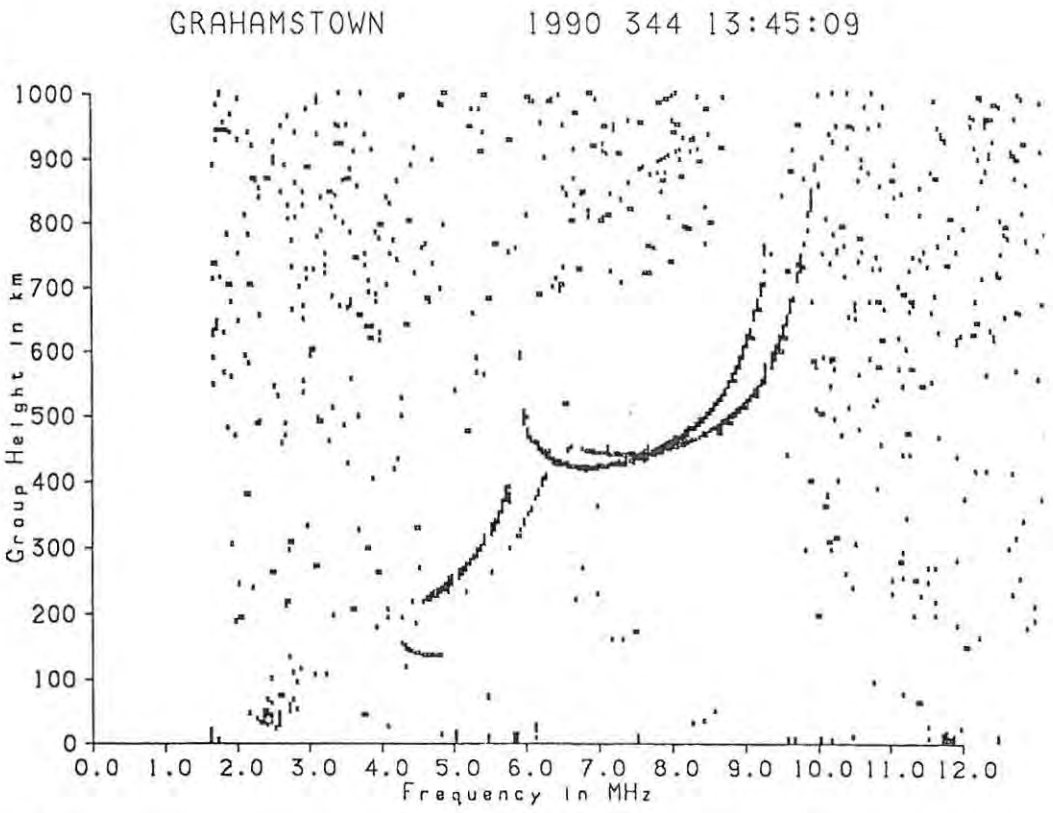
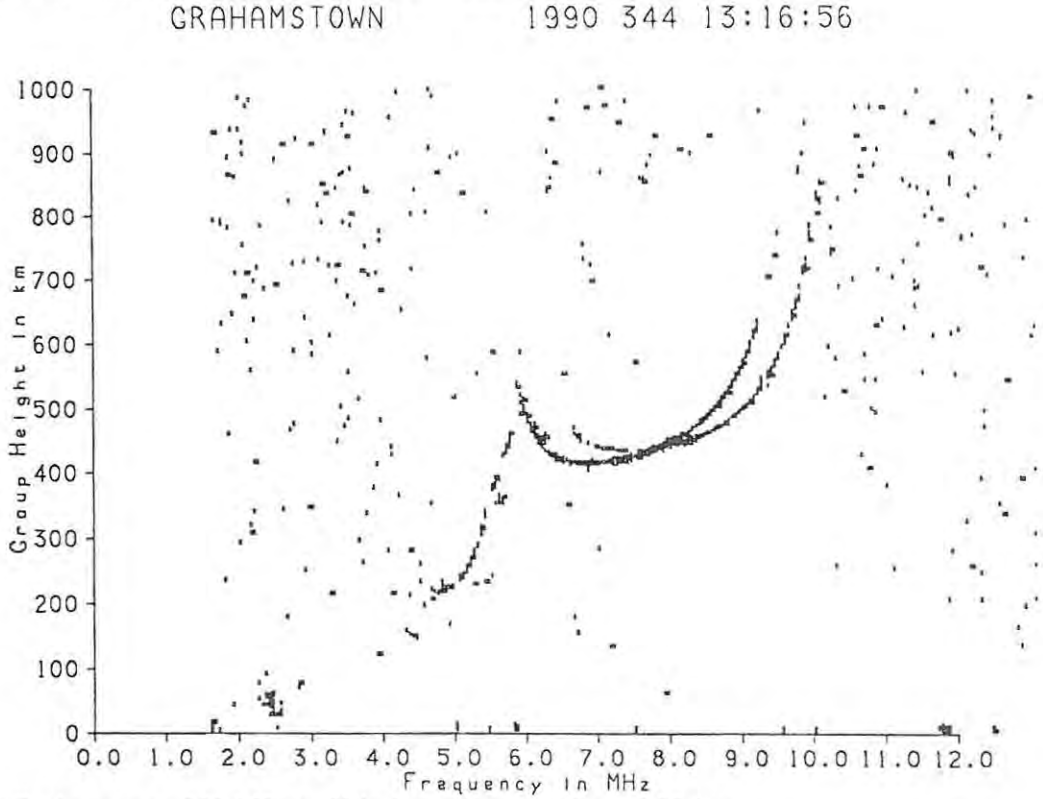


Figure 5-10: Comparison of ionograms recorded by (a) the VOS-1 and (b) the BR-9034 ionosondes (Dr A Poole)

(a) Basic rate = 50 kHz/s, Window offset height = 0 km



(b) Basic rate = 200 kHz/s, Window offset height = 200 km

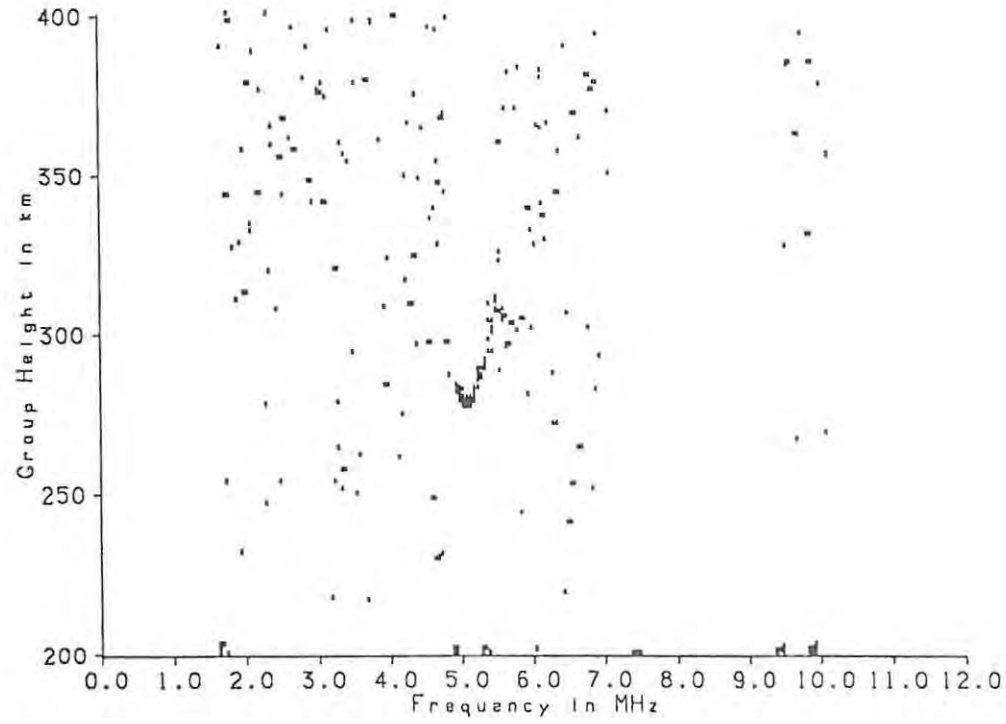
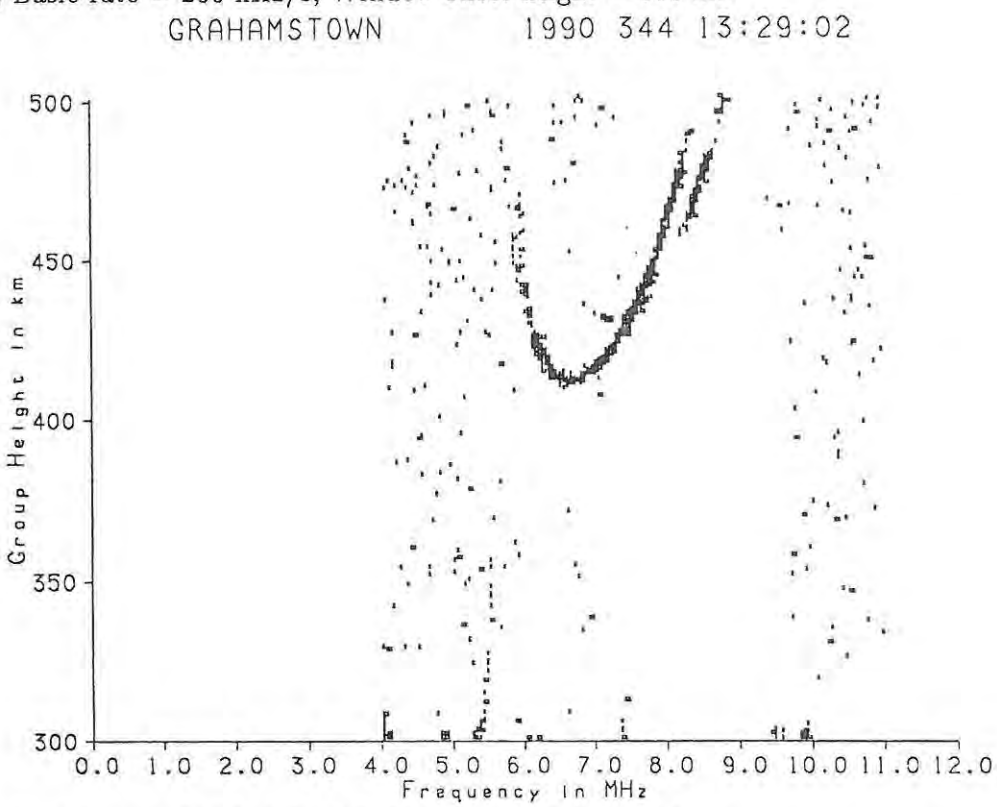


Figure 5-11: Improving vertical resolution by increasing the basic rate and using a window offset - part 1 (Dr A Poole)

(a) Basic rate = 200 kHz/s, Window offset height = 300 km



(b) Basic rate = 200 kHz/s, Window offset height = 400 km

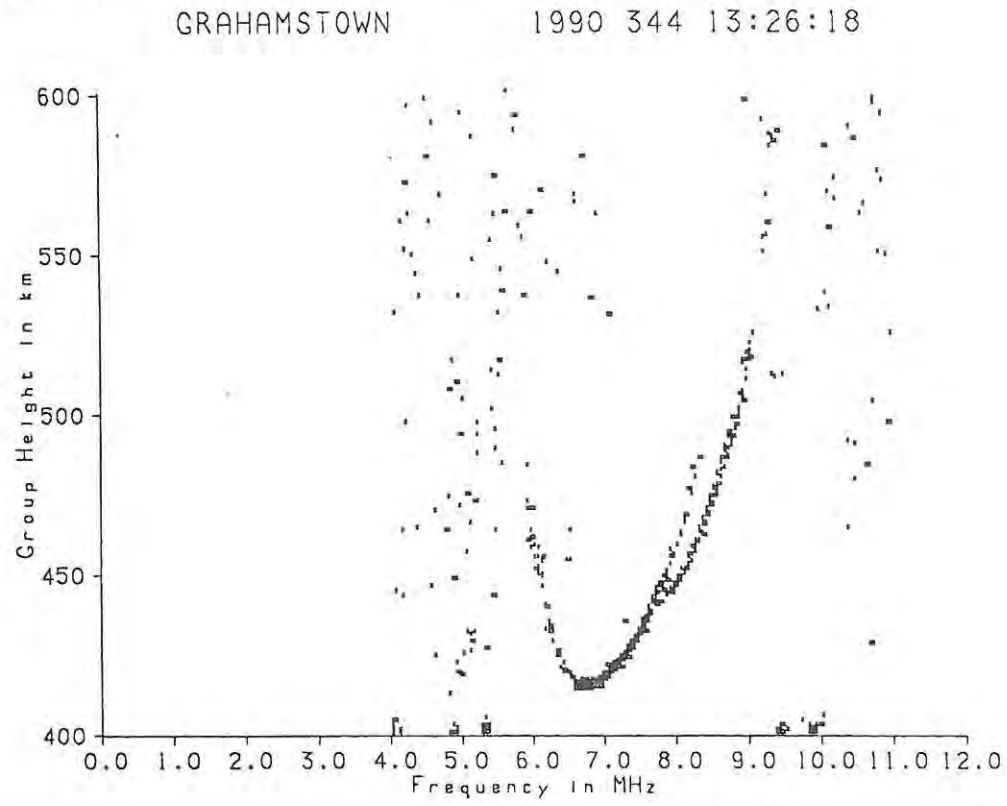
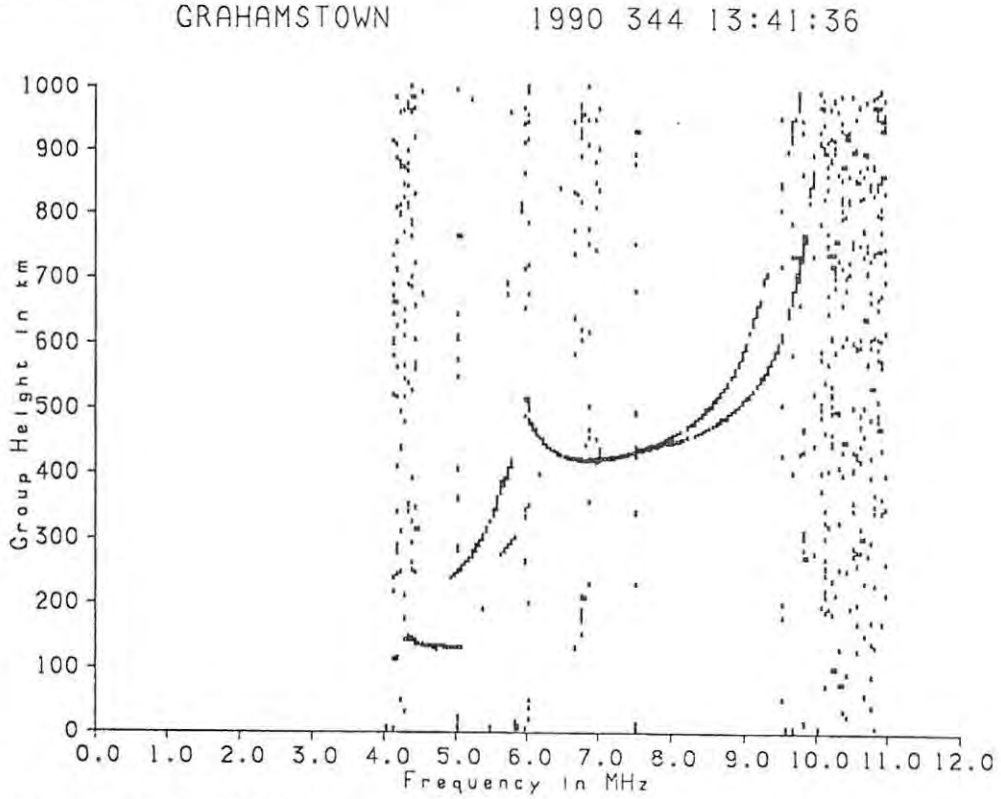


Figure 5-12: Improving vertical resolution by increasing the basic rate and using a window offset - part 2 (Dr A Poole)

(a) Overall rate = 50 kHz/s



(b) Overall rate = 25 kHz/s

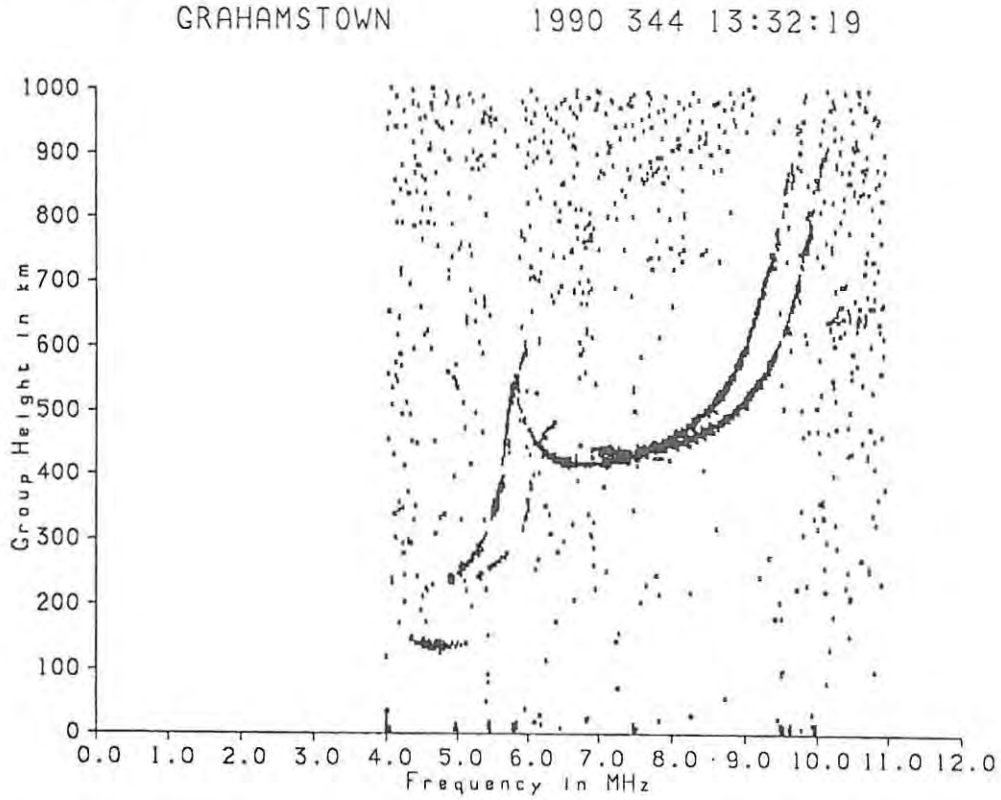
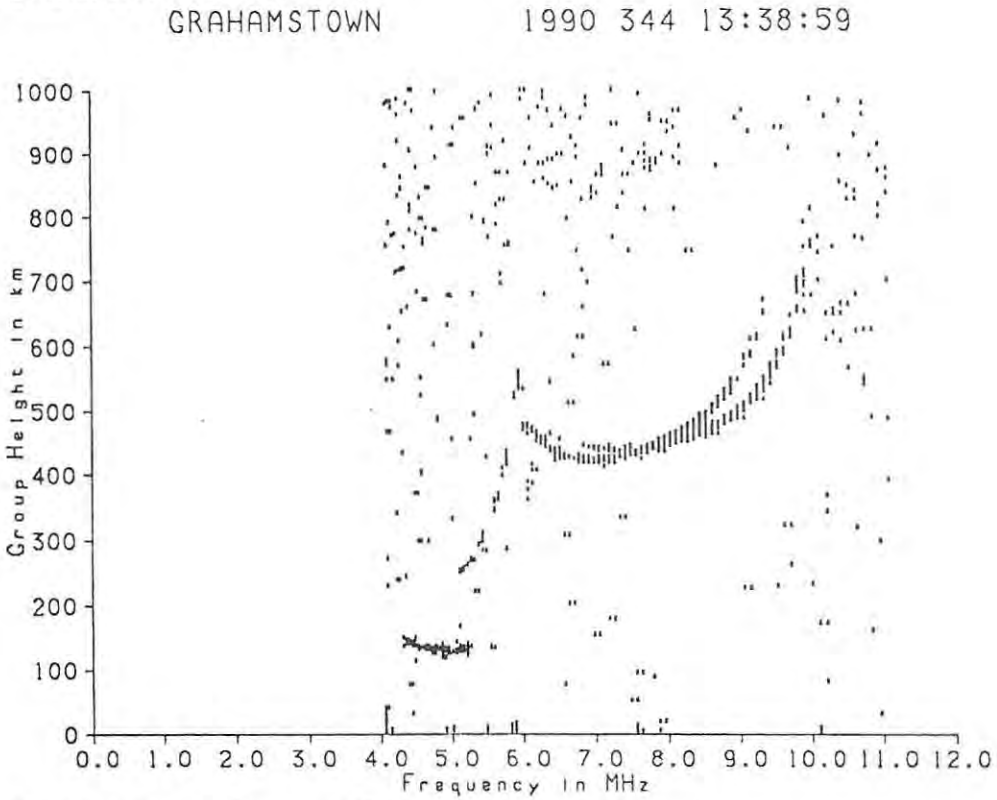


Figure 5-13: Comparison of ionograms recorded using different linear overall sweep rates (Dr A Poole)

(a) Overall rate = 0,01 octaves/s



(b) Overall rate = 0,02 octaves/s

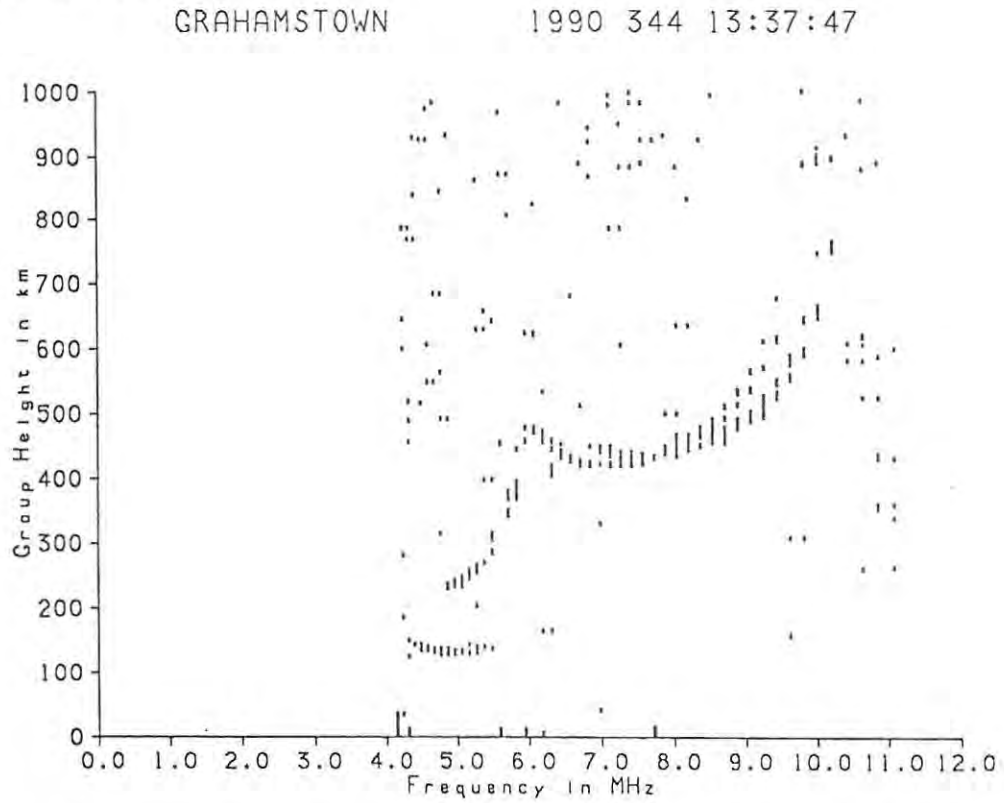
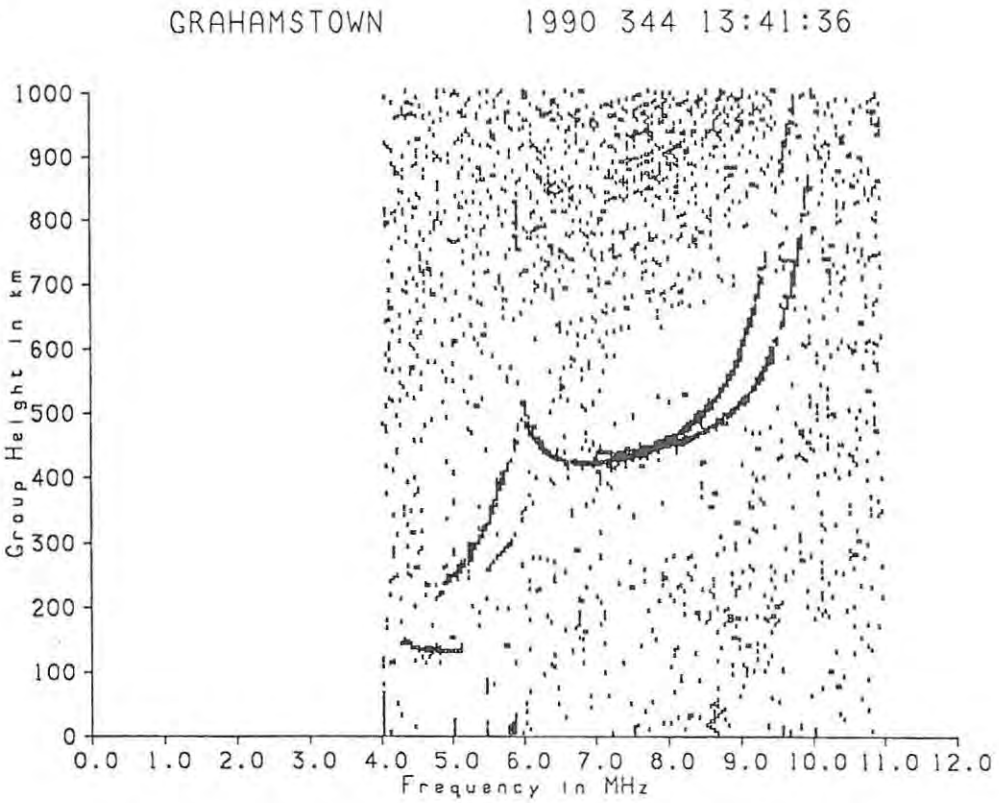


Figure 5-14: Comparison of ionograms recorded using different logarithmic overall sweep rates (Dr A Poole)

(a)



(b)

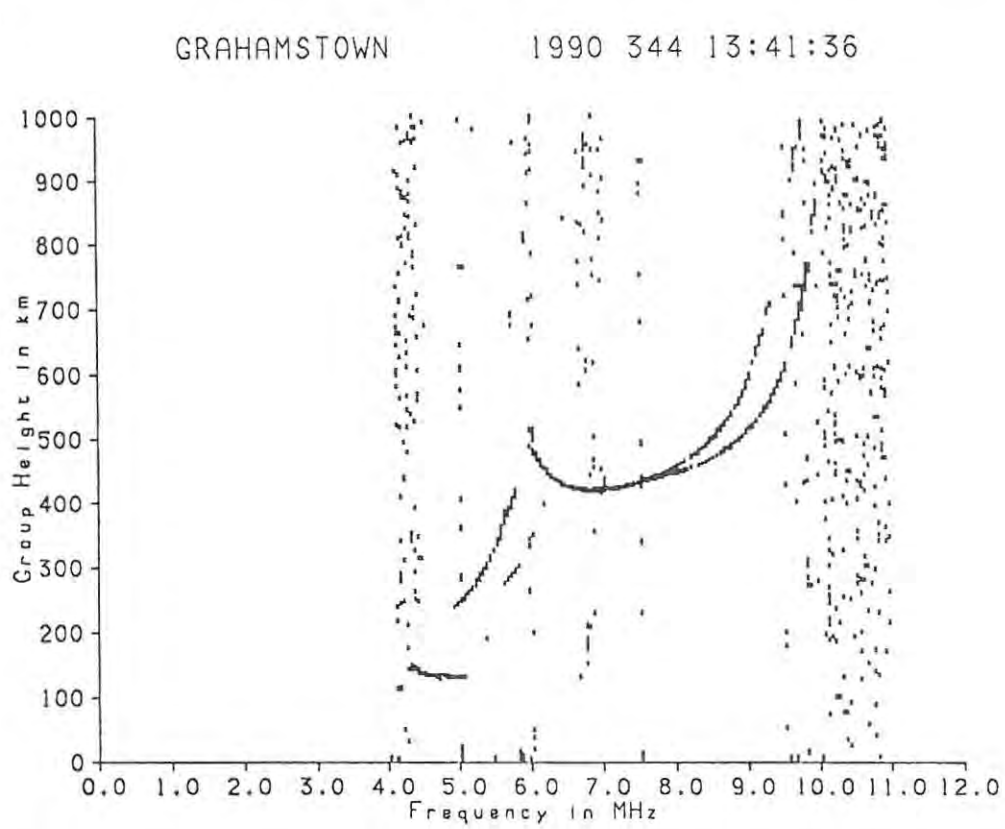


Figure 5-15: Results of recording an ionogram (a) without and (b) with an amplitude threshold criterion (Dr A Poole)

Chapter 6

Conclusion

In the author's design approach, the microcomputer is dedicated to the control of the BR-9034, as opposed to treating it as a peripheral device. This is necessary as the PC must integrate four separate functional blocks in order to operate the sounder and capture data from it. These are: the serial port, parallel port, signal processing card, and timing controller.

Real time operation necessitated the use of interrupts, with careful consideration having been given to the prioritization of interrupt requests. Limitations of the DOS operating system involving reentrancy, dictated the development of software void of screen output within interrupt routines. Instead, a system of software flags has been implemented. These flags are set during interrupts and continually polled by the 'core' routine of the program to sequentially display output.

6.1 Meeting the initial project specifications

Turbo Pascal proved an excellent choice of 'high level' programing language. From a language point of view this is due to the readability and reliability of Pascal. From a development point of view Borland's Turbo implementation proved a good choice because of its advanced integrated environment, and speed of compilation and linking.

Development of the initial hardware and software for meteor operation was completed in November 1989, and has been thoroughly tested while operating almost continuously since then. During this period the system has frequently been improved and numerous favourable comparisons of results with data from the old meteor logging system have been

made. The new logging system has a higher level of noise immunity than the old, and can record valid data even during periods of intense 'breakthrough'.

A simple menu driven user interface involving the function keys allows complete control of the sounder. System operations to effect structured frequency sweeps, as well as accurate timing adjustments, can be programmed via the line editor and saved on disk. All data entered is validated before being accepted. These operations can then be scheduled to execute at specific times providing a powerful yet flexible control system. Detailed descriptions of these functions can be found in the *Operators Guide.*

A plug-in card for the PC, the Timing Controller, provides all system timing requirements. A Real Time Clock chip is used for the clock/calendar functions and its input clock signal is derived from an external frequency standard via a phase-locked loop. This frequency standard is also used by the BR-9034 and thus phase coherency is maintained between the sounder and the Timing Controller. High speed CMOS technology has been used widely in the construction of the Timing Controller and most of the logic is provided with battery backup. Meteor operation is possible without this card, in which case the PC's Real Time Clock is used.

Diagrams for the antenna and signal switching circuitry have been provided, but the Signal Switching unit itself has not been constructed. Consequently no digital ionogram results have been recorded, although this potential capability exists. On the monitor screen, ionograms produced by the BR-9034 and the VOS-1 compare favourably.

Timing shifts during sounder operation are effected by programming frequency offsets. Timing shifts are only implemented in hardware after completion of the sounder sweep. No tests have been performed adjusting system timing during sounder operation.

6.2 Future development of the system

The first obvious improvement to the system would be the use of a VGA card for screen output. These cards can generally drive CGA monitors, and use could be made of the multiple graphics pages instead of continually re-drawing the screen. This would also enable plotting of ionograms to an off screen page as a background operation while, for instance, editing system operations. Alternatively, the higher resolution of a VGA monitor could be used for increased vertical resolution of ionograms.

Considering the astronomical cost of magnetic tape drives and the recent advances in optical drives, all data storage may in the future be on optical disks. High information densities can be achieved on these disks, the medium is non-volatile, and they could even replace film. This would alleviate both the need to process film, and to rewrite magnetic tapes, as well as providing rapid random access to any ionogram. Physical space required for data storage would no longer be a consideration, and scaling of ionograms could be done directly on the PC or even automatically.

Circuit boards are required for the Timing Controller and signal switching circuitry. In addition, much of the logic could be implemented on Field Programmable Gate Arrays (FPGA) which would reduce the number of discrete IC's. Some of the logic requires the high speed of HCMOS so care would have to be taken in implementing this step to avoid timing errors.

The chirp radar system discussed here has a dual research purpose. Primarily it is intended to probe the ionosphere using the refractive/reflective properties of this layer. Indirectly it provides insight into upper atmosphere winds from observation of meteor trails. The controller thus operates in both ionosonde and meteor mode on a time sharing basis. In ionosonde mode the duration of sounder operations is generally short but the start time is important, particularly for oblique operations. In meteor mode sounder operation is ideally continuous and it is only the approximate time of day that is important. Scheduling of operations has been orientated towards ionosonde operation and has some drawbacks for meteor operation. A compromise solution might be to schedule hardware operations with an exact start time, but Schedules with both a start and end time. In other words a Schedule will remain active as long as the time of day lies between its two limits. This means that when the PC boots-up it will automatically run the correct schedule for that time of day, something which is difficult to implement at present. For example, a daytime schedule and a nighttime schedule.

6.3 The ionosonde rationalisation programme

Standardization of the man - machine interface at a software level would render the exact hardware configuration invisible to the operator. At the hardware level the rationalisation programme would involve the development of a standard interface between the PC

and the chirp radar. It was decided to adopt the BR-9034 interface and to develop a microprocessor based Interface Adapter for the VOS-1 and VIS-1 sounder systems.

At present one possible solution would be a 8051/2 based micro-controller card with onboard EPROM, RAM and serial port, in addition to numerous input/output control lines. PC based development systems exist for these cards which would facilitate software development in a 'high level' language such as Pascal.

Very little additional hardware design would be required, as most of the functional blocks exist in the Vertichirp Controller. Additional features include the need to return the receiver IF amplitude and attenuation to the PC in the status report, as well as RF power amplifier failure detection. The VOS-1 and VIS-1 systems allow for advance/retard timing shifts during sounder operation by adjusting the basic rate. This feature would have to be forfeited, for the sake of consistency, in favour of the less elegant frequency adjustment method employed with the BR-9034 system. Some system flexibility allowed by the Vertichirp Controller would be lost, namely that of modifying the operating software while it is executing. This is a feature of the Forth programming language, and although conceptually very powerful, it can produce disasterous results if used by the inexperienced. Both the Interface Adapter hardware and software would be simpler than that of the Vertichip Controller, and the overall system would be significantly easier to use.

Chapter 6

Conclusion

In the author's design approach, the microcomputer is dedicated to the control of the BR-9034, as opposed to treating it as a peripheral device. This is necessary as the PC must integrate four separate functional blocks in order to operate the sounder and capture data from it. These are: the serial port, parallel port, signal processing card, and timing controller.

Real time operation necessitated the use of interrupts, with careful consideration having been given to the prioritization of interrupt requests. Limitations of the DOS operating system involving reentrancy, dictated the development of software void of screen output within interrupt routines. Instead, a system of software flags has been implemented. These flags are set during interrupts and continually polled by the 'core' routine of the program to sequentially display output.

6.1 Meeting the initial project specifications

To allow for immediate comparison with the requirements as laid down in Section 1.3 introduction, the results are listed in corresponding point form.

1. Development of the initial hardware and software for meteor operation was completed in November 1989, and has been thoroughly tested while operating almost continuously since then. During this period the system has frequently been improved and numerous favourable comparisons of results with data from the old meteor logging system have been made. The new logging system has a higher level of noise

immunity than the old, and can record valid data even during periods of intense 'breakthrough'.

2. A simple menu driven user interface involving the function keys allows complete control of the sounder. System operations to effect structured frequency sweeps, as well as accurate timing adjustments, can be programmed via the line editor and saved on disk. All data entered is validated before being accepted. These operations can then be scheduled to execute at specific times providing a powerful yet flexible control system. Detailed descriptions of these functions can be found in the *Operators Guide.*

3. Turbo Pascal proved an excellent choice of 'high level' programing language. From a language point of view this is due to the readability and reliability of Pascal. From a development point of view Borland's Turbo implementation proved a good choice because of its advanced integrated environment, and speed of compilation and linking.

4. A plug-in card for the PC, the Timing Controller, provides all system timing requirements. A Real Time Clock chip is used for the clock/calendar functions and its input clock signal is derived from an external frequency standard via a phase-locked loop. This frequency standard is also used by the BR-9034 and thus phase coherency is maintained between the sounder and the Timing Controller. High speed CMOS technology has been used widely in the construction of the Timing Controller and most of the logic is provided with battery backup. Meteor operation is possible without this card, in which case the PC's Real Time Clock is used. Timing shifts during sounder operation are effected by programming frequency offsets. Timing shifts are only implemented in hardware after completion of the sounder sweep. No tests have been performed adjusting system timing during sounder operation.

5. Diagrams for the antenna and signal switching circuitry have been provided, but the Signal Switching unit itself has not been constructed. Consequently no digital ionogram results have been recorded, although this potential capability exists. On the monitor screen, ionograms produced by the BR-9034 and the VOS-1 compare favourably.

6. An extensive amount of time was spent by the author and Mr B.T Bonnevie in streamlining both the controller software and data capture system for compatibility. At the time of writing however, the two had not been fully integrated, although the author has made use of many of the routines written by Mr Bonnevie.

7. No hardware has been designed for the film recording system, but allowance has been made to drive such a device from software.

8. Only limited tests were made for comparison between the VOS-1 and BR-9034 and as such there was no interference with the ionosonde programme. The old Apple microcomputer based meteor logging system was finally shut down in November 1990. Prior to this it had been running in parallel with the new system, and as a result very little data loss was incurred directly as a result of failure of the new logging system.

9. The ionosonde rationalisation programme is summarised in Section 6.3.

6.2 Future development of the system

The first obvious improvement to the system would be the use of a VGA card for screen output. These cards can generally drive CGA monitors, and use could be made of the multiple graphics pages instead of continually re-drawing the screen. This would also enable plotting of ionograms to an off screen page as a background operation while, for instance, editing system operations. Alternatively, the higher resolution of a VGA monitor could be used for increased vertical resolution of ionograms.

Considering the astronomical cost of magnetic tape drives and the recent advances in optical drives, all data storage may in the future be on optical disks. High information densities can be achieved on these disks, the medium is non-volatile, and they could even replace film. This would alleviate both the need to process film, and to rewrite magnetic tapes, as well as providing rapid random access to any ionogram. Physical space required for data storage would no longer be a consideration, and scaling of ionograms could be done directly on the PC or even automatically.

Circuit boards are required for the Timing Controller and signal switching circuitry. In addition, much of the logic could be implemented on Field Programmable Gate Arrays

(FPGA) which would reduce the number of discrete IC's. Some of the logic requires the high speed of HCMOS so care would have to be taken in implementing this step to avoid timing errors.

6.3 The ionosonde rationalisation programme

Standardization of the man - machine interface at a software level would render the exact hardware configuration invisible to the operator. At the hardware level the rationalisation programme would involve the development of a standard interface between the PC and the chirp radar. It was decided to adopt the BR-9034 interface and to develop a microprocessor based Interface Adapter for the VOS-1 and VIS-1 sounder systems.

At present one possible solution would be a 8051/2 based micro-controller card with onboard EPROM, RAM and serial port, in addition to numerous input/output control lines. PC based development systems exist for these cards which would facilitate software development in a 'high level' language such as Pascal.

Very little additional hardware design would be required, as most of the functional blocks exist in the Vertichirp Controller. Additional features include the need to return the receiver IF amplitude and attenuation to the PC in the status report, as well as RF power amplifier failure detection. The VOS-1 and VIS-1 systems allow for advance/retard timing shifts during sounder operation by adjusting the basic rate. This feature would have to be forfeited, for the sake of consistency, in favour of the less elegant frequency adjustment method employed with the BR-9034 system. Some system flexibility allowed by the Vertichirp Controller would be lost, namely that of modifying the operating software while it is executing. This is a feature of the Forth programming language, and although conceptually very powerful, it can produce disastrous results if used by the inexperienced. Both the Interface Adapter hardware and software would be simpler than that of the Vertichip Controller, and the overall system would be significantly easier to use.

References

- Ariel, 1987, Operating Manual for the DSP-16 Data Acquisition Processor, New Jersey USA
- Barry Research, 1972, Vertichirp Sounder VOS-1 Operating and Service Manuals, Palo Alto California
- Booth IS, 1988, *Meteor Radar Data Collection and Analysis System* (Honours project report), Rhodes University
- Borland, 1988, Turbo Assembler ver 1.0 User's & Reference Guide, California
- Borland, 1988, Turbo C ver 2.0 User's & Reference Guide, California
- Borland, 1988, Turbo Pascal ver 5.0 User's & Reference Guide, California
- BR Communications, 1984, Operating and Service Manual for Model 9034 Chirpsounder System, Sunnyvale California
- Brown DL, 1985, *From Pascal to C*, Wadsworth, Belmont California
- Burr-Brown, 1987, *Handbook of Personal Computer Instrumentation*, 2nd Edition, USA
- Evans GP, 1984, *Computer Control of a Barry Research Chirpsounder* (MSc thesis), Rhodes University
- Fisher JS, 1979, *Real Time Fast Fourier Transform Analyser* (MSc thesis), Rhodes University
- Griggs DB, 1986, *Data Communications Analyser* (design/project report), University of the Witwatersrand

- Hawkins GS, *The Physics and Astronomy of Meteors, Comets and Meteorites*, McGraw Hill
- Horowitz P & Hill W, 1984, *The Art of Electronics*, Cambridge University Press, New York
- IBM, 1983, PC XT/AT Technical Reference, USA
- Jenson & Partners International (JPI), 1988, TopSpeed Modula-2 User's Manual & Language Tutorial, USA
- Ladd SR, July 1988, *A Turbo TSR*, Byte Magazine, 301
- Lamport L, 1986, LaTeX: *A Document Preparation System*, Addison-Wesley, USA
- Langman D, 1986, Model 10 Data Acquisition and Signal Processing Board Documentation, Delanco Spry, Rochester New York
- Marsh MJC, 1986, *Phase Doppler Chirp Ionosonde*, CSIR/NITR, JHB
- McIndoe T, June 1989, *Software Productivity*, Electronics & Wireless World, 624
- McKinley DWR, 1961, *Meteor Science and Engineering*, 1961, McGraw Hill
- Microsoft, 1984, MS-DOS ver 3.1 Users Reference & Programmers Guide
- Motorola, 1979, European CMOS Selection Data Book
- Nathanson P & Grant R, 1987, Interfacing course notes, Rhodes University
- OrCAD, 1989, OrCAD/SDT III, Oregon
- Poole AWV, 1983, *Advanced Ionospheric ChirpSounding* (PhD thesis), Rhodes University
- Poole LMG, 1988, *Grahamstown All Sky Meteor Radar*, Journal of Atmospheric and Terrestrial Physics, **50**, 585
- Prosise J, April 1987, *Instant Access to Directories*, PC Magazine, 313
- Reinisch BW, August 1986, *The Digisonde 256 system and Ionospheric Research*, Ionosonde Network Advisory Group (INAG) bulletin no. 48, 13

- Rubenking NJ, November 1987, *Optimising Turbo Pascal: A primer*, PC Magazine, 331
- Software Link, 1987, PC-MOS/386 Modular Operating System release 2.1 Users Guide
- Stanley WD, 1982, *Digital Signal Processing*, 2nd Edition, Reston Publishing
- Terry PD, 1987, *Modula-2 for Pascal Programmers*, Rhodes University
- Terry PD, 1989, *MS-DOS and JPI Modula-2*, Rhodes University
- Texas Instruments (TI), 1986, TMS32020 Users Guide
- URSI, 1978, Handbook of Ionogram Interpretation & Reduction, World Data Centre A for Solar - Terrestrial Physics, 2nd Edition, USA
- Waite Group, *MS-DOS Papers*, Howard W Sams & Company Hayden Books, Indiana
- Wakai N, 1985, Ohyama H & Koizami T, Manual of Ionogram Scaling, Radio Research Laboratories – Ministry of Posts and Telecommunications, Japan
- Wood K, June 1989, *The PC Graphics Maze*, Electronics & Wireless World, 560

Glossary

ADC – Analog to Digital Converter

ANSI – American National Standards Institute

ASCII – American Standard Code for Information Interchange

BIOS – Basic Input Output Sysytem

This is the ROM based code responsible for the lowest level hardware operations on the IBM PC.

CGA – Colour Graphics Adapter

CMOS – Complementary Metal Oxide Semi-conductor

DAC – Digital to Analog Converter

DCE – Data Communications Equipment

DCS – Data Capture System

DMA – Direct Memory Access

DOS – Disk Operating System This is the name of a very popular operating system for the IBM PC.

DSP – Digital Signal Processing

This thesis involves itself with the DSP-16, a signal processing card manufactured by the Ariel Corporation.

DSR – Data Set Ready

A status line used by the RS-232 standard to indicate to the DTE that the DCE is 'on-line'.

DTE – Data Terminal Equipment

EGA – Enhanced Graphics Adapter

HCMOS – High speed CMOS

IA – Interface Adapter

This box contains a microprocessor used for basic control of the VIS-1 and VOS-1 sounder systems, and provides a standardised interface to the PC similar to that of the BR-9034.

I/O – Input/Output

IRQ – Interrupt Request

PC – Personal Computer

In this thesis PC is understood to imply an IBM compatible AT.

PIC – Priority Interrupt Controller

RF – Radio Frequency

RFPA – Radio Frequency Power Amplifier

SS – Signal Switching unit

This box contains all the RF, baseband, and filter switching hardware.

TC – Timing Controller

This is the hardware card responsible for providing all critical timing operations such as strobing the sounder.

VGA – Video Graphics Array / Versatile Graphics Adapter

Appendix A

Advanced chirpsounding basic principles

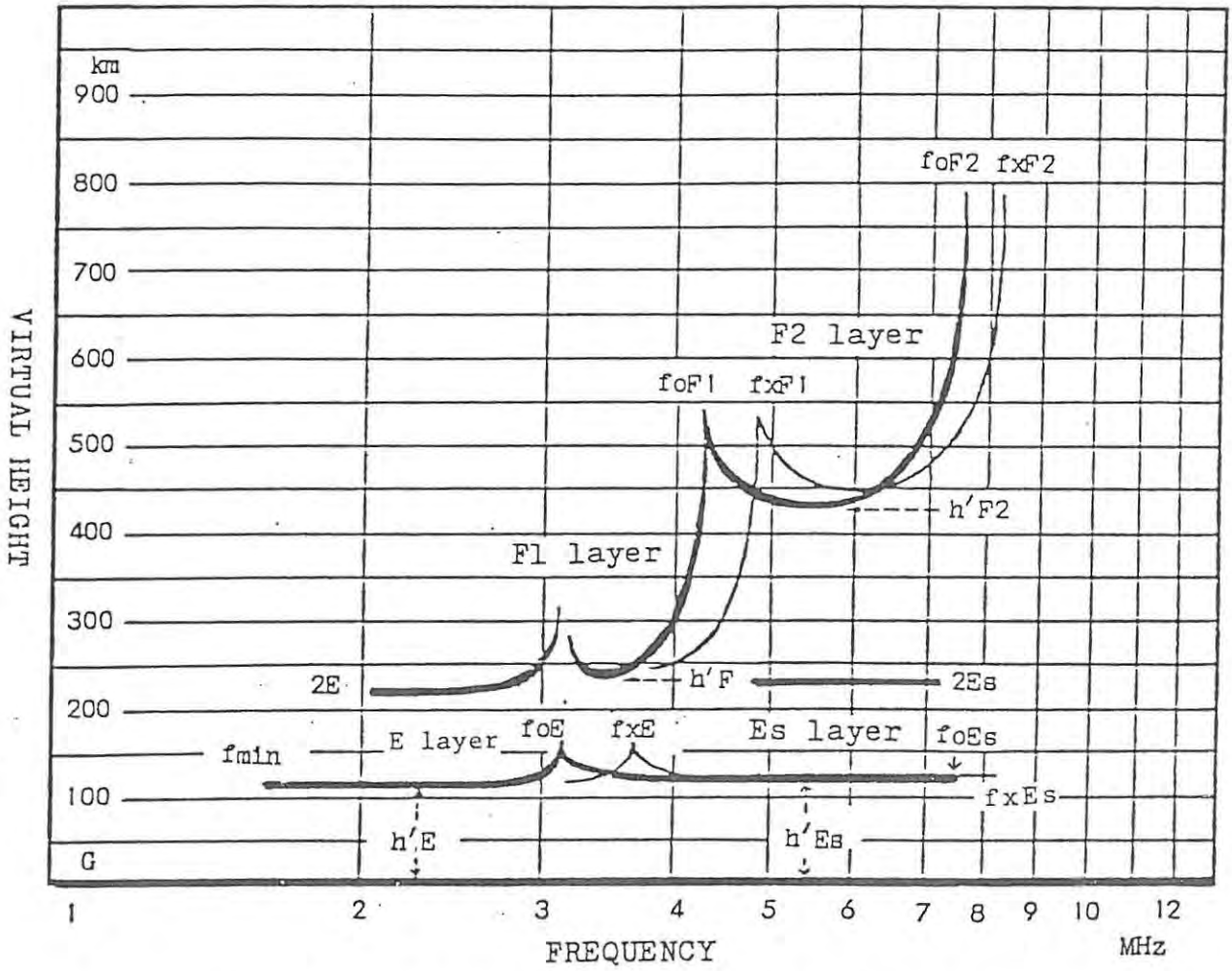


Figure A-1: Typical ionogram [Wakai N *et al* 1978]

Both pulse and chirp ionosondes measure the return delay of a transmitted signal. However the way this is done differs for the two techniques as shown in Figures [A-2] & [A-3]. As a result of the narrow bandwidth of the transmitted signal, chirp sounders can produce results equivalent to those of pulse sounders at a fraction of the peak transmitted power (factor of 10^3). The price paid for this is the duration of the required sweep (minutes as opposed to seconds).

In order to determine such information as angle of arrival, doppler velocity, and signal polarization, a segmented frequency structure is required. This *ionogram structure* will now be outlined in more detail [Evans GP 1984].

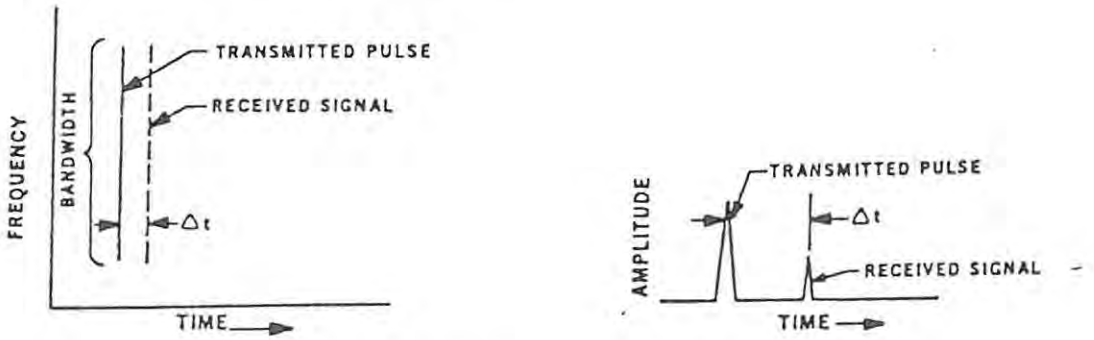


Figure A-2: Pulse ionosonde [Barry Research 1972]

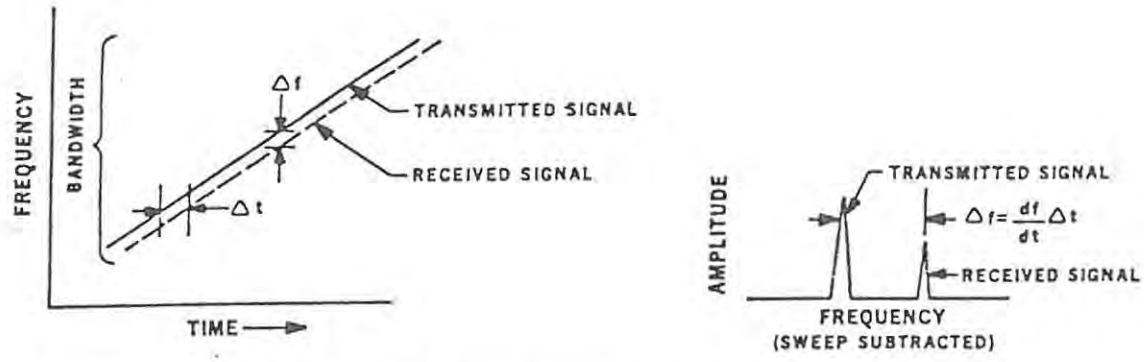


Figure A-3: Chirp Ionosonde [Barry Research 1972]

A.1 A cell

This is the fundamental building block of a digital ionogram. During a cell the transmitter frequency increases from the cell start frequency f_C, at some linear rate called the *basic rate* (K_B). The cell length (T_C), basic rate and receive antennas are variable. The cell start frequency may also be offset by an amount $\triangle f$ from the sounding start frequency.

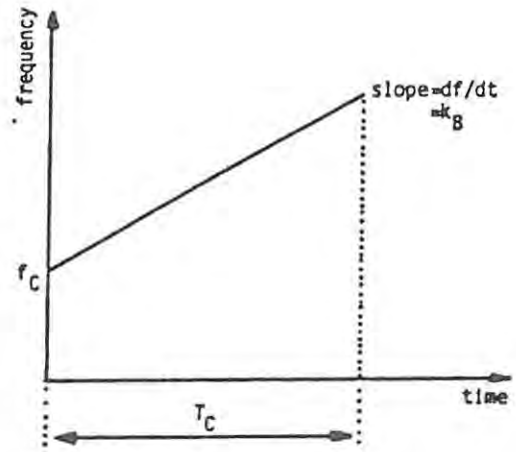
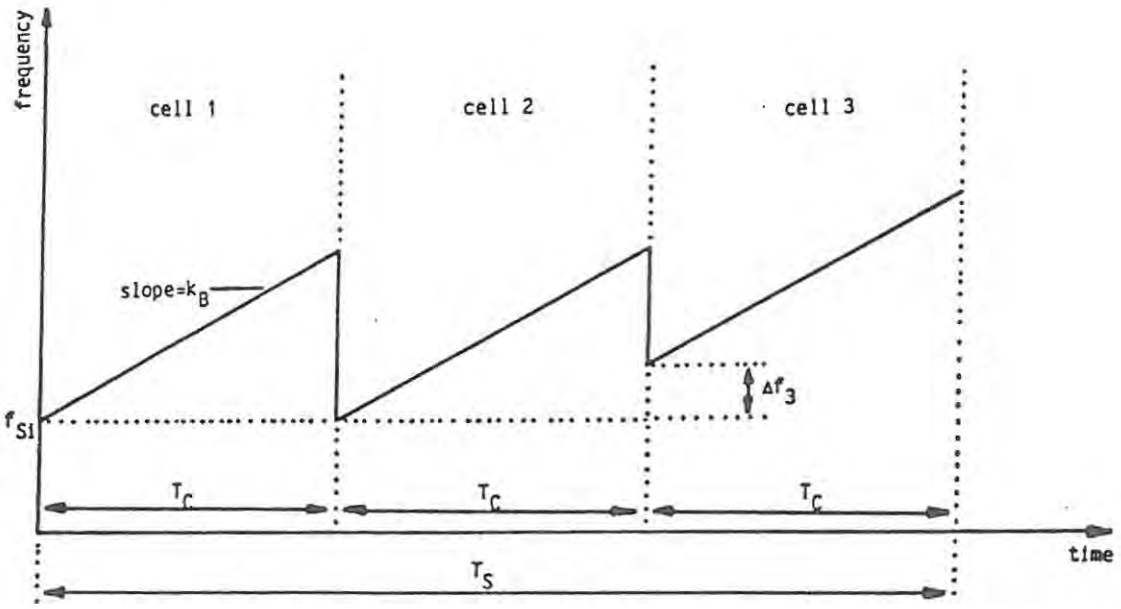


Figure A-4: A cell [Evans GP 1984]

A.2 A sounding

This consists of n adjacent cells which together allow for the evaluation of certain ionospheric parameters at a specific (characteristic) frequency. The maximum number of points per cell to be recorded for a sounding can be specified as well as the minimum amplitude threshold.



T_S = sounding period

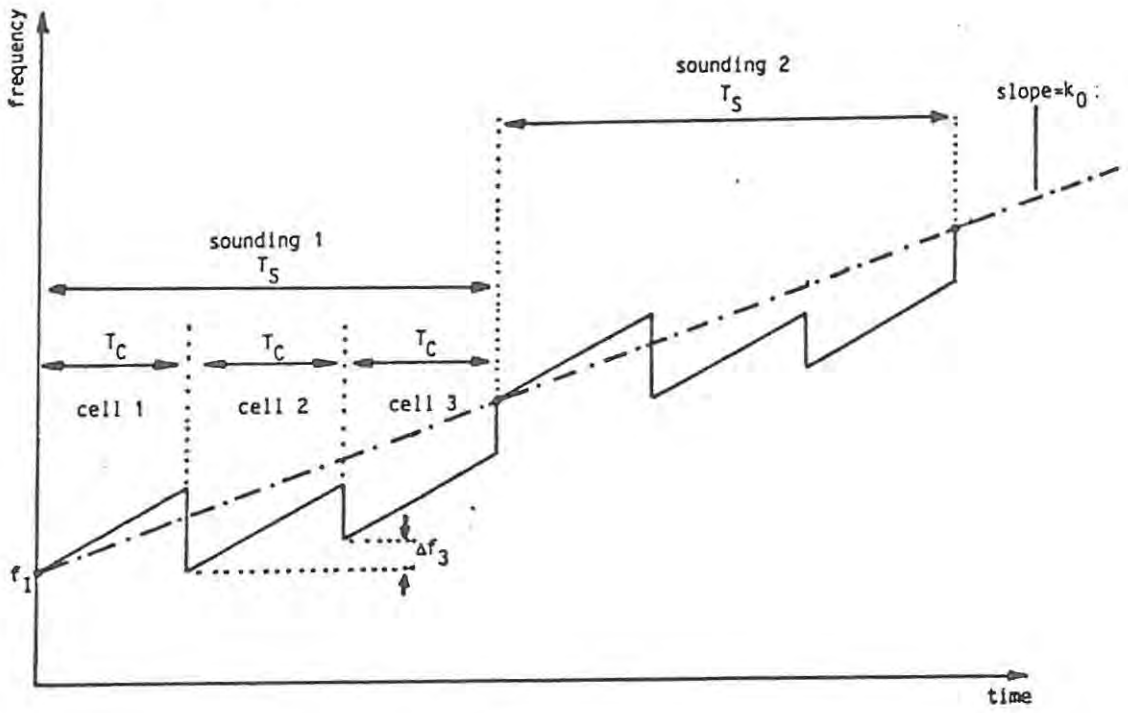
f_{Si} = sounding i start frequency

Δf_3 = cell 3 offset frequency

Figure A-5: A 3 cell sounding [Evans GP 1984]

A.3 An ionogram

An ionogram consists of a number of consecutive soundings whose characteristic (or sounding start) frequencies are related by some linear or logarithmic rate of change called the *overall rate* (k_0). The ionogram start and end frequencies may be defined, and output directed to magnetic tape, disk or film.



f_I = ionogram start frequency

k_0 = overall sweep rate, linear frequency scale

Figure A-6: Digital ionogram structure [Evans GP 1984]

Depending on the basic rate and number of cells per sounding, frequency jumps forward or backwards may be required at the start of a sounding in order to approximate the overall rate. Both may be required if the overall rate is logarithmic.

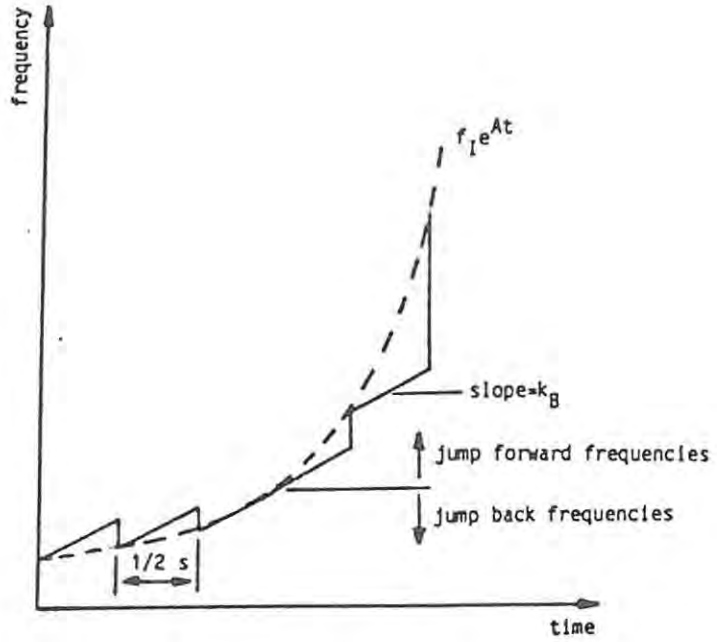


Figure A-7: Logarithmic sweep [Evans GP 1984]

A.4 A stationary ionogram

This is a normal ionogram structure but with an overall sweep rate of zero. That is the normal sounding structure is maintained but the sounding start frequency does not change. Instead of specifying an ionogram end frequency a time duration is given.

A.5 A doppler sounding

Here a single cell structure is used and both the basic and overall rates are set equal to zero. In other words the transmit frequency is fixed and is usually offset from the receiver tuned frequency. A doppler sounding (also called a stationary doppler) is distinctly different from a sounding in that it is not a building block of an ionogram.

A.6 Basic equations

The following is a list of all symbols used and their definition:

f_{Cij} start frequency of cell j in sounding i — Hz

T_C cell period (invariant for ionogram) — s

k_B basic sweep rate — Hz/s

n total number of cells per sounding

f_{Si} start frequency of sounding i — Hz

T_S sounding period (invariant for ionogram) — s

Δf_j cell j offset frequency — Hz

f_I ionogram start frequency — Hz

f_E ionogram end frequency — Hz

m total number of soundings per ionogram

T_I ionogram period or duration — s

k_0 overall sweep rate, linear frequency scale — Hz/s

A overall sweep rate exponential time constant, logarithmic frequency scale — 1/s

All derivations for the following formulas can be found in the MSc thesis by GP Evans, *Computer Control of a Barry Research Chirpsounder*, Rhodes University, 1984.

Sounding period:

$$T_S = nT_C \tag{A-1}$$

Start frequency of cell j in sounding i:

$$f_{Cij} = f_{Si} + \Delta f_j \tag{A-2}$$

Ionogram start frequency:

$$\begin{aligned} f_I &= f_{S1} \\ &= f_{C11} - \Delta f_1 \end{aligned} \tag{A-3}$$

Start frequency of sounding i:

$$f_{Si} = f_I + k_0 i T_S \quad \text{— linear} \tag{A-4}$$

$$f_{Si} = f_I e^{AiT_S} \quad \text{— logarithmic} \tag{A-5}$$

Total number of soundings in an ionogram:

$$m = \frac{(f_E - f_I)}{k_0 T_S} \quad \text{— linear} \tag{A-6}$$

$$m = \frac{\ln(f_E/f_I)}{AT_S} \quad \text{— logarithmic} \tag{A-7}$$

Ionogram duration:

$$T_I = mT_S \tag{A-8}$$

A.7 Windowing

The virtual height of the ionosphere is

$$h' = c(f_T - f_R)/2k_B$$

where $(f_T - f_R)_{max}$ is the effective receiver bandwidth (500 Hz) or 'window'.

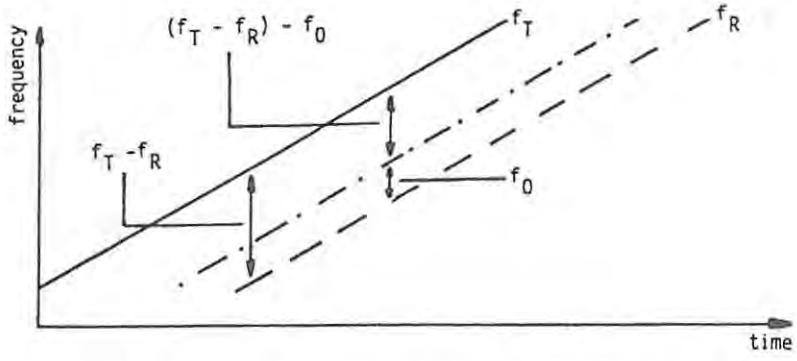


Figure A-8: Graphical representation of windowing. [Evans GP 1984]

The virtual height scale has minimum and maximum values given by:

$$h'_{min} = 0 \text{ km}$$

$$h'_{max} = c(f_T - f_R)_{max}/2k_B \text{ km}$$

h'_{max} is the *window height.*

This range could be adjusted to cover any window of 500 Hz by subtracting a fixed offset f_0 from the difference frequency as shown in Figure [A-8].

$$\begin{aligned} h' &= \frac{c[(f_T - f_R) - f_0]}{2k_B} \\ &= \frac{c(f_T - f_R)}{2k_B} - \frac{cf_0}{2k_B} \end{aligned} \tag{A-9}$$

$cf_0/2k_B$ is the *window offset* (Figure [A-9]).

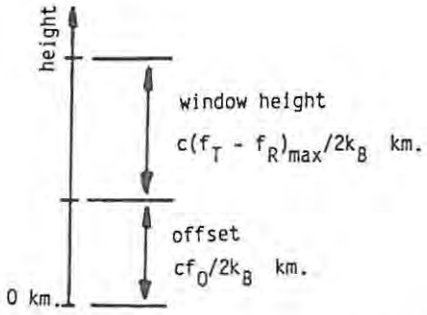


Figure A-9: Window offset and window height. [Evans GP 1984]

Appendix B

Background to the meteor programme

When a small object moving through interplanetary space encounters the earth's atmosphere it leaves in its wake a train of visible light and a trail of ionized gas. The object is known as a *meteoroid* and the phenomenon as a *meteor.* It is estimated that over the entire earth approximately a hundred million visible meteors burn themselves out daily. [McKinley DWR 1961] A meteor that survives entry and strikes the earth is termed a *meteorite.* Most visible radiation comes from the vicinity of the vapourising meteoroid known as the head.

Meteors can be classified into two main groups — sporadic and shower meteors. The *radiant* of a meteor is the point where the meteor path, extrapolated backwards, intersects the celestial sphere (Figure [B-1]). Shower meteors are characterised by the fact that the

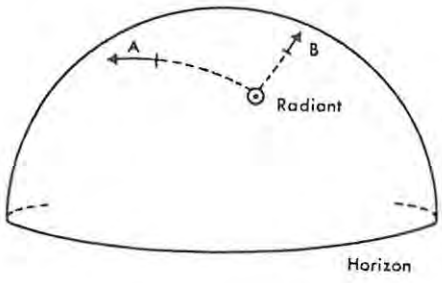


Figure B-1: The meteor path as projected on the celestial sphere by two spaced observers. [Hawkins GS]

meteoroids all enter the earth's atmosphere along parallel tracks and thus, when observed from the ground, all appear to have originated from a single point on the celestial sphere, known as the *shower radiant.*

The observation of meteor showers forms an important component of meteor astronomy. Showers occur when the earth's orbit traverses a stream of meteoroids, and many showers recur on an annual basis. Nearly all meteor streams have been associated with existing comets. It should be noted however, that the comet may or may not be the direct cause of the meteor stream and the association may be simply due to the fact that they both have a common origin.

B.1 Observation techniques

For many years visual observations, either by naked eye, telescope, or using cameras and long exposures, have formed the backbone of meteor observations. Limitations of such observations are that they are confined to clear night-time conditions, and that only comparatively bright meteors are detectable. These methods are prone to human error and accuracy is low given that the meteor is generally travelling quite rapidly.

The ionization trail can be detected using radio waves which are reflected from the region of enhanced ionization to give a result analogous to the detection of material objects using radar. With radar it is possible to record meteors not visible to the naked eye, and it can be used during daylight hours and under adverse weather conditions. The reflection point will lie on a line drawn from the radar station perpendicular to the meteor trail. There are two main radar techniques — the forward scatter (bistatic) type and the backscatter (monostatic) type. In the monostatic case, the transmitter and receiver are co-located while in the bistatic case they are spatially separated. [McKinley DWR 1961]

B.2 The Grahamstown meteor radar

The Grahamstown meteor radar [Poole LMG 1988] has been operational since April 1986 and is of the monostatic type. A continuous wave at a fixed frequency of 27,99 MHz is square wave modulated at 500 Hz. An array of four receive antennas and one transmit antenna is used to provide near all sky coverage. All antennas are half-wave dipoles,

mounted one third of a wave length above a conducting ground plane with their axes aligned North - South. This configuration allows for the measurement of not only the Doppler shift of any meteor echo but also angle of arrival. Two phase matched receivers with bandwidths of 2500 Hz are used and the transmitter has a negative offset of 40 Hz relative to the receiver 'tuned' frequency (the frequency which produces a DC baseband output). Mean transmitter power is approximately 30 W.

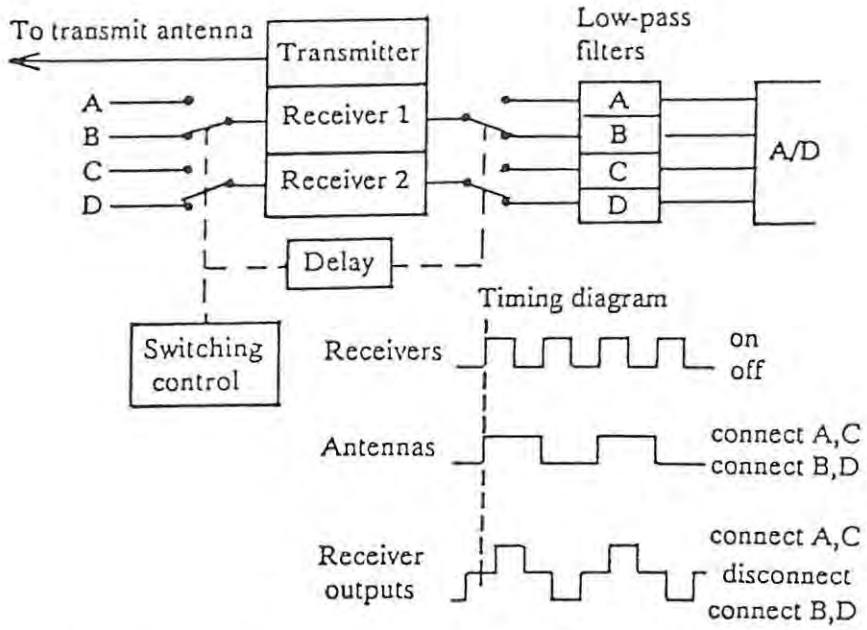


Figure B-2: Schematic of meteor operation antenna switching. [Poole LMG 1988]

The two receivers serve the four effective channels by switching the receiver inputs between pairs of antennas at 250 Hz while synchronously switching each output between lowpass filters, with cut off frequencies around 100 Hz, as shown in Figure [B-2]. Meteor echoes appear at the output as an audio tone generally doppler shifted about 40 Hz. Antennas are positioned as shown in Figure [B-3]. Comparison of the phase differences between antennas A, B and C provide a range of possible directions for the echo location and D is situated so as to optimally resolve the ambiguity.

The original data logging system used an Apple-type microcomputer with a multi-channel A/D converter board. The sample rate was 256 Hz and logging of data was triggered by any signal exceeding a certain amplitude threshold. Data was transferred to floppy disk once the computer's memory was full. [Poole LMG 1988]

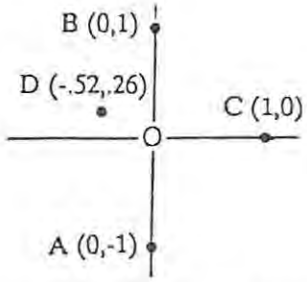


Figure B-3: Receive antenna array with co-ordinates given in wavelengths relative to the origin. [Poole LMG 1988]

B.3 Shortcomings of the old data logging system

1. No signal processing was performed by the Apple which resulted in the collection of a large amount of useless data not due to meteor echoes.

2. Raw data from the A/D was stored to floppy disk which requires a large amount of space.

3. In addition to the above the Apple disk drives are not capable of storing the same amount of data as the newer disk drives.

4. Items 1 to 3 above all resulted in frequent operator intervention being required to change disks.

5. Frequency resolution was poor due to the low sample rate

6. Data was only written to disk once the Apples memory was full, thus large amounts of data could be lost due to power dips.

7. Due to the co-location of an ionosonde with the meteor radar, and the use of an amplitude threshold criterion for data logging, which was suspended during ionosonde runs to avoid interference problems, a substantial loss of data was incurred. [Booth IS 1988]

Appendix C

Revision of the Fast Fourier Transform

The *Fast Fourier Transform* (FFT) is a high speed algorithm for computing the Fourier transform of a discrete time signal.

C.1 Discrete Fourier transform

A non-periodic, discrete time function will have as its transform a continuous, periodic frequency function (see Table [C-1]).

Time Function	Frequency Function
Non-periodic and Continuous	Non-periodic and Continuous
Periodic and continuous	Non-periodic and Discrete
Non-periodic and Discrete	Periodic and Continuous
Periodic and Discrete	Periodic and Discrete

Table C-1: Comparison of Fourier transform pairs [Stanley WD 1982]

If we assume that our sampled signal in the time domain is periodic then the result is a sampled frequency function. This is known as the *discrete Fourier transform* (DFT) and is the one used for digital computations.

If the time signal is denoted $x(n)$ with sample time T understood, and similarly with

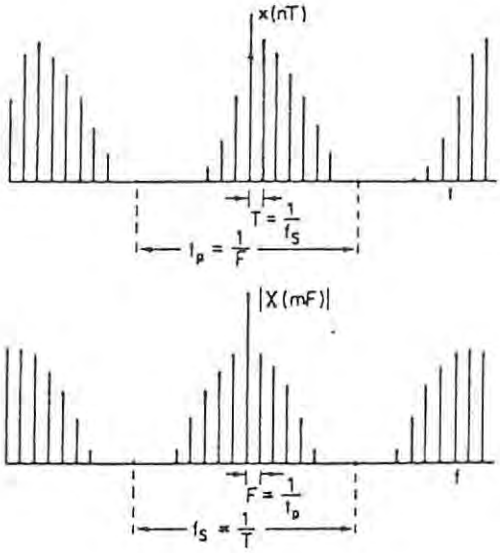


Figure C-1: A periodic, discrete time function and its Fourier transform which is a periodic, discrete frequency function [Stanley WD 1982]

$X(m)$ and the frequency increment F, for

$$0 \leq n \leq N-1$$

$$0 \leq m \leq N-1$$

and

$$FT = \frac{1}{N} \tag{C-1}$$

Then the DFT transform pair can be stated

$$X(m) = \sum_{n=0}^{N-1} x(n) W_N^{mn} \tag{C-2}$$

$$x(n) = \frac{1}{N} \sum_{m=0}^{N-1} X(m) W_N^{-mn} \tag{C-3}$$

In both the time and frequency domain, one period corresponds to N points $(0 \ldots N-1)$ with the point N corresponding to the beginning of a new period (Figure [C-2]). One important difference between the time and frequency functions though, is that the frequency function over half the period is related to the other half. Thus the maximum unambiguous frequency is $f_S/2$, which is the well known folding frequency (f_o).[1] Since

[1]This only applies if the time function is real

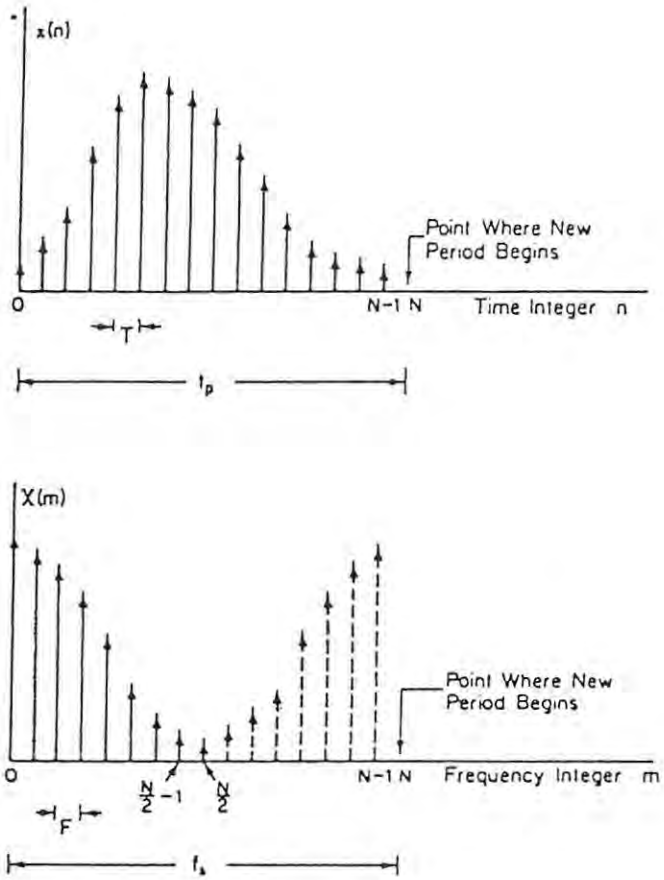


Figure C-2: Illustration of some of the properties of the DFT [Stanley WD 1982]

$m = \frac{N}{2} - 1$ is the maximum unambiguous frequency integer and $m = 0$ is the first, there are $N/2$ unique transform components when the time function has N points.

It can be shown that the behaviour of of either $x(n)$ or $X(m)$ over the range $N/2$ to N is identical to the behaviour from $-N/2$ to 0. Thus the range $N/2$ to N can be considered as a sort of 'negative' time or frequency range in interpreting various even or odd properties of the transform functions.

Definitions:

even function	$x(N-n) = x(n)$	$X(N-m) = X(m)$
odd function	$x(N-n) = -x(n)$	$X(N-m) = -X(m)$

We can write

$$X(m) = X_r(m) + jX_i(m) \tag{C-4}$$

where $X_r(m)$ is the real part of $X(m)$, and $X_i(m)$ the imaginary part. It can be shown that when $x(n)$ is real, $X_r(m)$ is an even function of m and $X_i(m)$ an odd function.

Similarly, when $x(n)$ is imaginary $X_r(m)$ is an odd function of m, and $X_i(m)$ an even function. Additional properties of the DFT are summarised in Table [C-2].

$x(n)$	$X(m)$
Real	Real part is Even Imaginary part is Odd
Imaginary	Real part is Odd Imaginary part is Even
Real and Even	Real and Even
Real and Odd	Imaginary and Odd
Imaginary and Even	Imaginary and Even
Imaginary and Odd	Real and Odd

Table C-2: Even and odd properties of the DFT [Stanley WD 1982]

Assume a complex time function

$$x(n) = r(n) + ji(n)$$

where $r(n)$ and $i(n)$ are both real sequences. Let $R(m)$ be the transform of $r(n)$ and $I(m)$ that of $i(n)$. It can be shown that

$$r(n) + ji(n) \rightleftharpoons R(m) + jI(m) \tag{C-5}$$

However $R(m)$ and $I(m)$ are both complex, therefore on the right hand side the overall real and imaginary parts are not directly identifiable. It can further be shown that

$$\begin{aligned} R(m) &= \frac{1}{2}[R_e + jI_o] \\ &= \frac{R(m) + R(N-m)}{2} + \frac{j[I(m) - I(N-m)]}{2} \end{aligned} \tag{C-6}$$

$$\begin{aligned} I(m) &= \frac{1}{2}[I_e - jR_o] \\ &= \frac{I(m) + I(N-m)}{2} + \frac{j[R(m) - R(N-m)]}{2} \end{aligned} \tag{C-7}$$

where R_e is the even function of the real input channel and R_o the odd function. Similarly for the imaginary channel. These results are important in that they allow two real signals $r(n)$ and $i(n)$ to be combined in a single complex function, transformed simultaneously,

$W_N^N = 1$	$W_N^{\frac{N}{2}} = -1$
$W_N^{\frac{N}{4}} = -j$	$W_N^{\frac{3N}{4}} = j$
$W_N^{KN} = 1$	$W_N^{KN+r} = W_N^r$
$W_{2N}^K = W_N^{\frac{K}{2}}$	

Table C-3: Properties of W_N [Stanley WD 1982]

and then separated into two distinct transforms $R(m)$ and $I(m)$. This implies that in our application the sampled data from both receivers can be transformed in one computational sweep. [Fisher JS 1979, Stanley WD 1982]

C.2 Fast Fourier transform

Certain special algorithms permit the implementation of the DFT with considerable savings in computational time (Table [C-4]). These class of algorithms are known as the *fast Fourier transform* (FFT). Note that the FFT is not a different transform from the DFT. If Equation [C-2] is expressed in matrix form then it can be written

$$\vec{X} = [W]\vec{x} \tag{C-8}$$

where $\vec{X}$ is the column vector defining the transform, $\vec{x}$ is the column vector defining the discrete time signal, and $[W]$ is an NxN matrix. To solve this equation requires approximately N^2 complex operations. [Stanley WD 1982]

The basic FFT algorithm was developed by JW Cooley and JW Tukey in 1965. Maximum efficiency of FFT algorithms is obtained by constraining the number of points in the time domain to be an integer power of two. [Fisher JS 1979]

Assume

$$N = 2^L$$

then $[W]$ can be factorised into L matrices

$$[W] = [W_1][W_2]\dots[W_L] \tag{C-9}$$

From the properties of W (summarised in Table [C-3]) it can be shown that each row of the individual matrices have only two non-zero terms. Substitution of Equation [C-9] into

Equation [C-8] yields

$$\vec{X}_s = [W_1][W_2]\dots[W_L]\vec{x} \qquad \text{(C-10)}$$

where the subscript on $\vec{X}_s$ denotes the fact that the components of this vector appear in a different order than assumed in the original definition of $\vec{X}$. To solve this equation requires approximately $N \log_2 N$ complex computations. [Stanley WD 1982]

Most FFT algorithms can be classified as either

in-place : Memory requirements for storage are kept at a minimum but either the input requires scrambling or the output unscrambling.

natural input-output : Input and output vectors are not scrambled but requires more memory.

The way in which an algorithm works is best illustrated by means of an example showing what happens on each iteration (butterfly). Such examples can be found in a number of texts and will not be repeated here.

The actual algorithm used was supplied by Delanco Spry [Langman D 1986] and is a 1024 point, in-place, scrambled input – natural output FFT.

N	N(DFT)	N(FFT)	N(FFT)/N(DFT)
16	256	64	0.250
32	1024	160	0.156
64	4096	384	0.0938
128	16,384	896	0.0547
256	65,536	2048	0.0312
512	262,144	4608	0.0176
1024	1,048,576	10,240	0.0098
2048	4,194,304	22,528	0.0054
4096	16,777,216	49,152	0.0029

N = Number of points in time sample.

$N(\text{DFT}) = N^2$ = Approximate number of complex arithmetic operations with DFT.

$N(\text{FFT}) = N \log_2 N$ = Approximate number of complex arithmetic operations with FFT.

Table C-4: Comparison of times for the DFT and FFT [Stanley WD 1982]

C.3 Practical implications of the FFT

From Equation [C-1] we have that

$$F = \frac{1}{NT}$$

where F is the frequency separation between two adjacent points on the spectrum. The frequency separation can be adjusted in a number of ways, one of which is to use a larger transform and 'zero fill' for the required number of additional samples. This has the effect of extending the length of the sampled signal in the time domain.

Note: No additional information about the signal is obtained, but merely less information is hidden due to the so called 'picket fence' effect. That is, more points on the spectrum are given and the likelihood of anomalies being hidden between points is reduced.

The smallest amount by which two similar signals must differ in frequency in order for them to still be resolved in the frequency domain is known as the *frequency resolution* (f_{res}). This is solely dependent on the duration of the sampled signal in the time domain and is given by:

$$f_{res} = \frac{1}{\text{duration sampled signal}} \tag{C-11}$$

Note: 'Zero filling' has no effect on the frequency resolution.

If the number of samples used in the transform is halved and the remainder zero filled, then the amplitude is halved. Remember also that in order to avoid aliasing the highest frequency contained in the sampled signal can only be half the sampling frequency. A summary of the various combinations of transform size and sampling frequency is given in Table [C-5].

Transform size N	Sample rate (Hz)	# samples taken	Relative amplitude	Separation of spectral points F (Hz)	Frequncy resolution (Hz)
256	256	256	1	1	1
	512	256	1	2	2
	1024	256	1	4	4
512	256	256	0.5	0.5	1
		512	1	0.5	0.5
512	512	256	0.5	1	2
		512	1	1	1
512	1024	256	0.5	2	4
		512	1	2	2
1024	256	256	0.25	0.25	1
		512	0.5	0.25	0.5
		1024	1	0.25	0.25
1024	512	256	0.25	0.5	2
		512	0.5	0.5	1
		1024	1	0.5	0.5
1024	1024	256	0.25	1	4
		512	0.5	1	2
		1024	1	1	1

Table C-5: Effects of FFT size and sampling frequency on frequency resolution and separation of points on the frequency spectrum

Appendix D

Pascal, C & Modula-2 test programs

D.1 Graphics routines

Pascal

```
PROGRAM TestPlot;

USES
  Graph;

VAR
  Pt_even: ARRAY[0..7999] OF BYTE ABSOLUTE $B800:$0000;
  Pt_odd:  ARRAY[0..7999] OF BYTE ABSOLUTE $BA00:$0000;
  x, y, GraphDriver, GraphMode: INTEGER;

PROCEDURE Plot_Pt (Xpos, Ypos: INTEGER);
(* Fast routine for plotting a white point on a CGA screen *)
VAR
  Byte_Pos: INTEGER;
  Mask: BYTE;

BEGIN
  IF (Xpos < 640) AND (Ypos < 200)
    THEN BEGIN
           Mask :=  1 Shl (7 - Xpos MOD 8);
           IF ODD(Ypos)
             THEN BEGIN
                    Byte_Pos := Xpos Shr 3 + (Ypos-1) * 40;
                    Pt_odd[Byte_Pos]  := Pt_odd[Byte_Pos] OR Mask;
                  END
             ELSE BEGIN
                    Byte_Pos := Xpos Shr 3 + Ypos * 40;
                    Pt_even[Byte_Pos] := Pt_even[Byte_Pos] OR Mask;
```

```
                END;
        END;
END;

BEGIN
  GraphDriver := cga;
  GraphMode := cgahi;
  InitGraph(GraphDriver,GraphMode,'\tp');
  FOR x := 1 TO 639 DO
    FOR y := 1 TO 199 DO Plot_Pt(x,y);          (* 'in-house routine *)
(*  FOR y := 1 TO 199 DO PutPixel(x,y,7);  *)  (* std routine *)
  CloseGraph;
  {Textmode;}
END. (* TestPlot *)
```

Modula-2

```
MODULE TESTPLOT;

FROM Graph IMPORT TextMode, GraphMode, Plot;

VAR
  PT_EVEN [0B000H:0]: ARRAY[0..7999] OF BYTE;
  PT_ODD [0BA00H:0]: ARRAY[0..7999] OF BYTE;
  X,Y: INTEGER;

PROCEDURE BitwiseOR(w1, w2: BYTE): BYTE;
BEGIN
  RETURN BYTE(BITSET(w1) + BITSET(w2))
END BitwiseOR;

PROCEDURE PLOT_PT(XPOS, YPOS: INTEGER);
(* Fast routine for plotting a white point on a CGA screen *)
VAR
  BYTE_POS: INTEGER;
  MASK: BYTE;
BEGIN
  IF (XPOS < 640) AND (YPOS < 200)
    THEN
    MASK :=  VAL(BYTE,1 << (7 - XPOS MOD 8));
    IF ODD(YPOS)
    THEN
      BYTE_POS := XPOS >> 3 + (YPOS-1) * 40;
      PT_ODD[BYTE_POS]  := BitwiseOR(PT_ODD[BYTE_POS],MASK)
    ELSE
      BYTE_POS := XPOS >> 3 + YPOS * 40;
      PT_EVEN[BYTE_POS] := BitwiseOR(PT_EVEN[BYTE_POS],MASK)
    END
```

```
  END
END PLOT_PT;

BEGIN
  GraphMode;
  FOR X := 1 TO 319 DO
    FOR Y := 1 TO 199 DO PLOT_PT(X,Y) END;     (* 'in-house' routine *)
(*  FOR Y := 1 TO 199 DO Plot(X,Y,7) END;  *)  (* modified std routine *)
  END;
  TextMode
END TESTPLOT.
```

D.2 Integer computations

Pascal

```
PROGRAM TestInt;

VAR
  a, b, c: INTEGER;

BEGIN
  FOR c := 1 TO 100 DO
    BEGIN
      b := 1;
      FOR a := 1 TO 15000
      DO BEGIN
           b := a - 2000 + b * 3;
         END;
    END;
  Writeln(b);
END.
```

Modula-2

```
MODULE TestInt;

FROM IO IMPORT WrInt;

VAR
  a, b, c: INTEGER;

BEGIN
  FOR c := 1 TO 100
    DO b := 1;
      FOR a := 1 TO 15000
        DO   b := a - 2000 + b * 3;
      END;
```

```
  END;
  WrInt(b,10);
END TestInt.
```

C

```
#include <stdio.h>

main()

{
  int  a, b, c;

  for (c = 1; c <= 100; c++)
    {
      b = 1;
      for (a = 1; a <= 15000; a++)
b = a - 2000 + b * 3;
    };
  printf("%d \n",b);
}
```

D.3 Floating point computations

Pascal

```
PROGRAM TestReal;

VAR
  a, b: INTEGER;
  x, y: REAL;

BEGIN
  x := 1;
  y := 1;
  b := 1;
  FOR a := 1 TO 15000
      DO BEGIN
           b := a - 2000 + b * 3;
           y := x * 50 - (y/10) + b;
         END;
  Writeln(b);
  Writeln(y:3);
END.
```

Modula-2

```
MODULE TestReal;

FROM IO IMPORT WrInt, WrReal;

VAR
  a, b: INTEGER;
  x, y: REAL;

BEGIN
  x := 1.0;
  y := 1.0;
  b := 1;
  FOR a := 1 TO 15000
      DO   b := a - 2000 + b * 3;
           y := x * 50.0 - (y/10.0) + VAL(REAL,b);
      END;
  WrInt(b,10);
  WrReal(y,3,10);
END TestReal.
```

C

```
#include <stdio.h>

main()

{
  int  a, b;
  float  x, y;

  x = 1;
  y = 1;
  b = 1;
  for (a = 1; a <= 15000; a++)
    {
      b = a - 2000 + b * 3;
      y = x * 50 - (y/10) + b;
    }
  printf("%d \n",b);
  printf("%e \n",y);
}
```

Appendix E

Modifications to the 9034 chirpsounder

E.1 Remote selection of the T/R waveform

This modification was made to allow for remote switching of the T/R waveform between the 500 Hz required in meteor radar mode, and M-sequence for ionosonde operation. The carrier in socket U54 on the interface board (4070-5541) was replaced by a quad, 2-input NAND gate (74LS00) and the following wiring alterations were made.

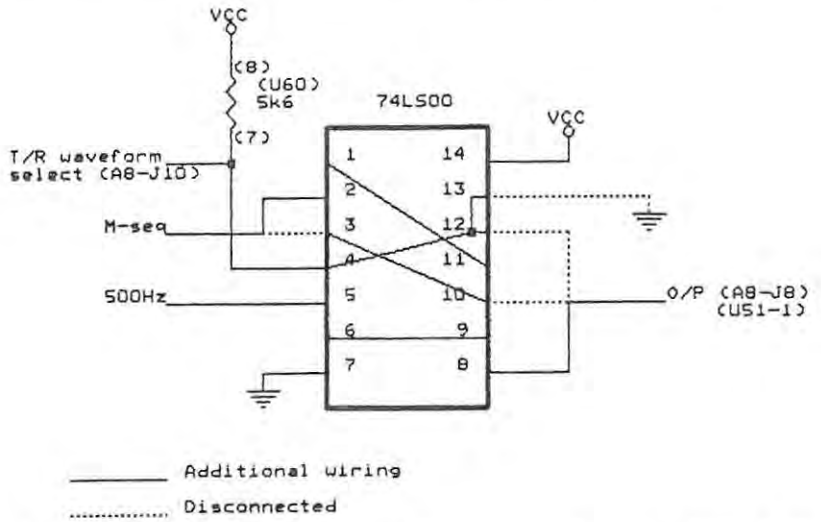


Figure E-1: Wiring modifications to enable remote selection of the T/R waveform

The *T/R waveform select* line has been connected to J10-3 on the rear panel. *Rx RF*

front end overload out has been disconnected. If T/R waveform select is held hi or left floating the output waveform will be a 500 Hz square wave. This output is available for use in meteor mode from a BNC (J24) that has been added to the rear panel, labeled *Rx T/R control out.* This output has been buffered and all modifications have been updated in the BR-9034 manuals.

E.2 Power on indicator

Rx calibrator on in has been disconnected from J10-4 and connected to a rear pannel switch on the 4070. J10-4 is now held at +5 V via a 100 Ω resistor connected to the supply in from the 1070. This *Power on* signal is used to switch AC power to the signal switching unit.

E.3 Gain weighting of external receivers

The *Rx gain weighting* signal has been made available on the back panel via a BNC (J25) to allow for the inclusion of additional receivers at a future date. This output is unbuffered.

Appendix F

DSP-16 Overview

F.1 Hardware description

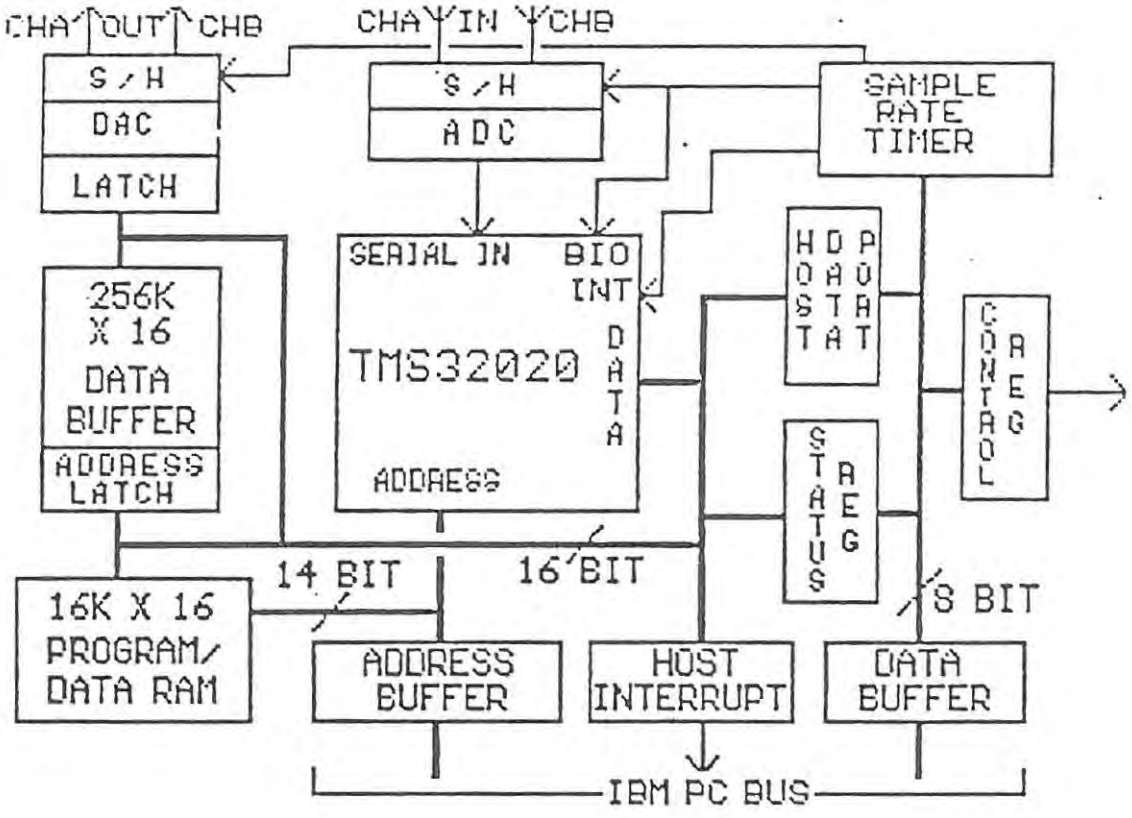


Figure F-1: Block diagram of the DSP-16 signal processing board. [Ariel 1987]

Ariel's DSP-16 is a signal processing 'plug in' card for an IBM PC/AT. It is based upon the Texas Instruments TMS32020 (TMS320C25) digital signal processor. It forms a self contained sub-system with 16k words of on-board RAM, 256k of DRAM (not used for this application), two 16 bit analog-to-digital converters, two 16 bit digital-to-analog converters, and a programmable sample rate timer.

Both analog input and output channels can operate at rates up to 50 kHz, with bandwidths of 20 kHz (-3 dB).

The DSP-16 communicates with the host via four consecutive I/O ports on the PC. DSP-16 RAM is mapped to a 64 kbyte page in the host computers memory space. This memory is only enabled during operation of the driver software and may be accessed by DMA. (Requires the TMS320 be placed on hold.) A more detailed discussion can be found in section F.6 entitled *Hardware configuration.*

In addition the DSP-16 can interrupt the host PC if enabled to do so by the driver software. In this application the DSP-16 interrupts the host PC whenever data is available for uploading. The interrupt request line is jumper selectable on the DSP-16 board should this be necessary for compatibility with existing hardware (section F.6.3). [Ariel 1987] The TMS320 software can be initialised from the PC into a state where only the analogue spectrum is output (ie. no data is transferred to the PC, implying no interrupts).

An *external gate* is available to disable the interrupt timer from hardware. Functionality of the external gate can be enabled or disabled by the PC in software. When the external gate is pulled to ground the interrupt timer is reset.

F.2 Memory map

Although the DSP-16 board only has 16k of memory, the TMS32020 is capable of addressing program and data memory of 64k each. The DSP-16's 16k consists of dual ported RAM. When the PC accesses this memory, it is mapped from the base address upwards as one contiguous block. The TMS32020, through the use of the Global Allocation Register (GREG), has the ability to map this 16k into various combinations of data and program memory (Actually local and global memory for use with multiple processors). The GREG is initialised to F8h which allocates 2k as data memory and 14k as program memory (see Table [G-1]). The program RAM is addressed by the TMS32020 from 0h to 37FFh. How-

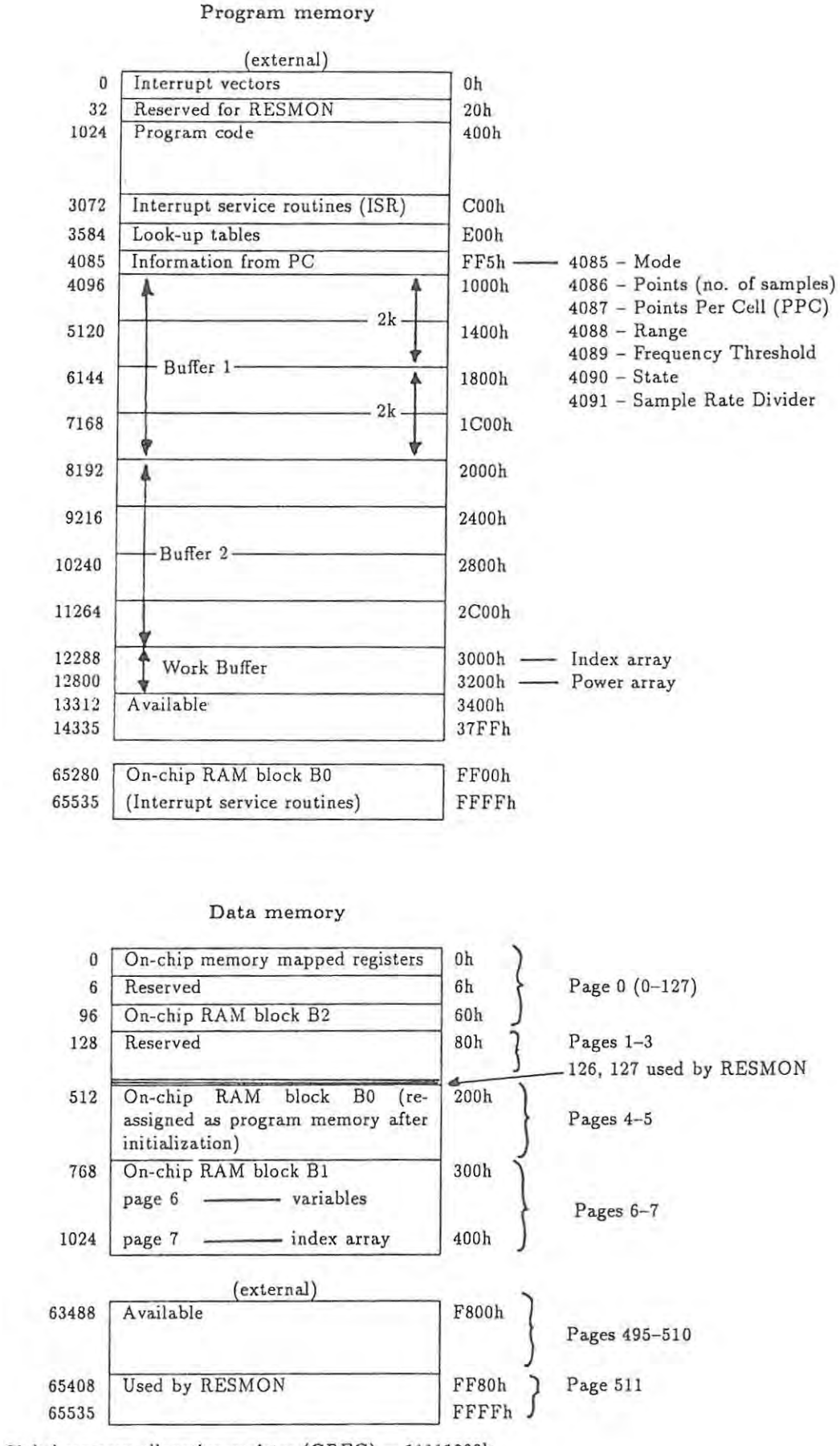


Figure F-2: Memory map of the DSP-16

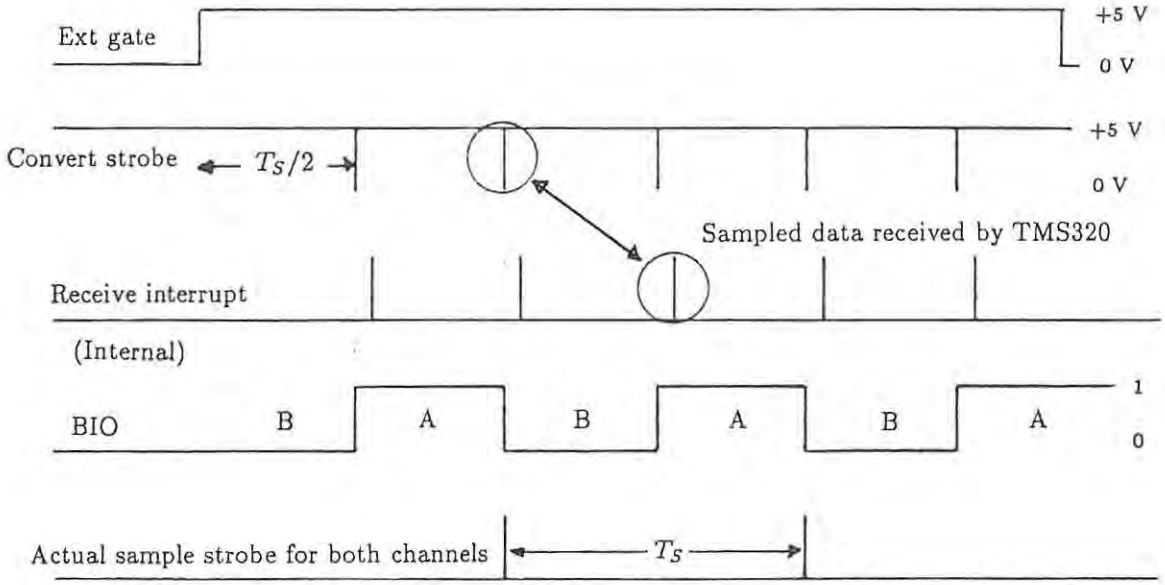
ever, through some hardware trickery, the external data RAM resides at address F800h to FFFFh (see Figure [F-2]). [Ariel 1987]
Note: From the PC's point of veiw this memory is still contiguous from the base address upwards!

The TMS32020 also has 544 words (16 bits) of 'on chip' memory located in three blocks (B0 = 256, B1 = 256, B2 = 32). These blocks are initialised as data RAM, however, block B0 can be relocated as program memory at addresses FF00h to FFFFh. Access to on chip RAM is twice as fast (minimum) [Ariel 1987] as external RAM, therefore it has been used for the storage of time critical code namely the interrupt service routines (ISR), as well as for the storage of all variables. On chip RAM can not be accessed directly by the PC, so the ISR's are transferred to memory block B0 during the initialisation routine of the TMS32020 assembler program. All branch instructions to this block have there destination addresses modified by the necessary offset from their assembled addresses. [TI 1986]

For compatibility with RESMON (the resident monitor used by Ariels debugger), block B2 has not been used and no code is located below 400h, apart from the modified interrupt vectors. In fact no external data memory has been used either, but RESMON requires the upper 128 words (FF80h to FFFFh) so if necessary more program memory can be allocated using GREG. If, however, the full 16k is allocated as program RAM you will forfeit the use of the powerful debugger.
Note: In order to run the TMS32020 program (**DSP16TMS.ASM**) with the debugger certain modifications to the code are required. These are documented in the source code.

The TMS320 data memory is paged with a maximum of 512 pages each containing 128 words. Page 0 represents the physical addresses 0h to 127h. The three basic addressing modes available on the TMS320 are, immediate, direct, and indirect (see Figure[G-4]). With direct addressing only the offset in the currently selected page (set by the Data Page pointer register) is used. Memory block B1 contains pages 6 and 7. Page 6 is used for storage of all data variables, while page 7 temporarily stores the index array used in sorting the power spectrum in ionosonde mode. [Ariel 1987, TI 1986]

F.3 Internal timing diagrams

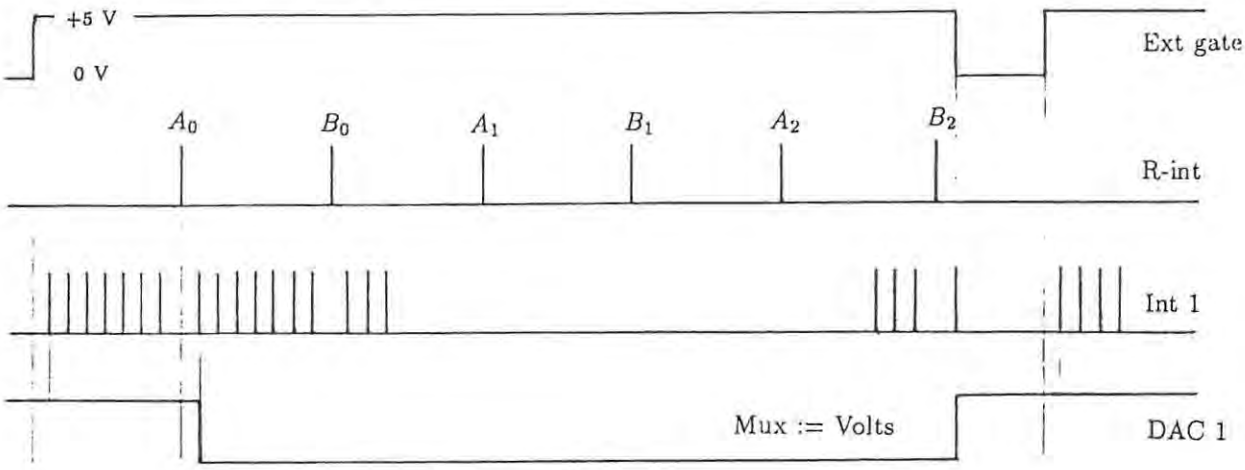


$T_S = 1/f_s$ (Sampling period)

Figure F-3: Derived timing diagram for DSP-16 interupts in conjunction with the external gate

A sample from channel A will result in an internal serial receive interrupt (R-int) in just over half a sample period later. A sample from channel B will cause an interrupt in just over one sample period later, as B is effectively sampled at the same time as A.[1] BIO is not reset or synchronised by the external gate, thus in the example in Figure [F-3], on the first convert strobe BIO will be set to 1 and the first interrupt will be received from channel A. This first value will be arbitrary as well as the second sample, received from channel B. The first actual sample strobe will occur exactly one sample period after the external gate goes hi. This is the intended state of operation in ionosonde mode. Had BIO initially been 1 then the first interrupt would come from channel B (sample would be arbitrary) and the first sample strobe would occur after half a sample period of the external gate going hi. In this case the data for the very first cell would be corrupt if used for cell comparisons, but timing would be correctly synchronised thereafter.

[1] This is achieved via the use of an additional sample and hold circuit on the channel B input line which is triggered by the convert strobe for channel A



$T_{Int\ 1} = T_{Int\ 2} \gg T_{R-int}$ Interrupt rates

Int 2 is 180° out of phase with Int 1

Figure F-4: Synchronisation of the sample rate timer in ionosonde mode via the external gate (simplified timing diagram)

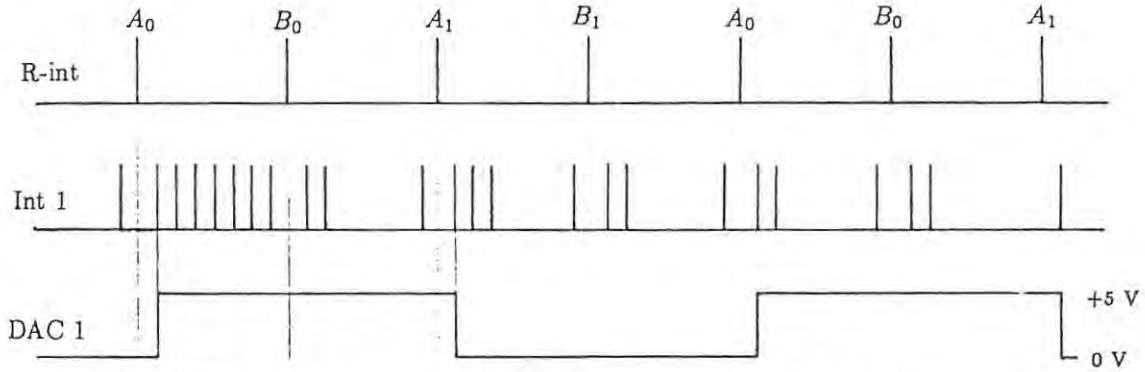


Figure F-5: Dual channel multiplexing in meteor mode (simplified timing diagram)

The ADC interrupt timer is set to twice the required sample rate as is the timer that generates interrupts 1 & 2. This second timer is set at a much higher frequency than the first. Interrupts 1 & 2 are 180° out of phase, with Int 1 being associated with DAC 1 and Int 2 with DAC 2. The lowest sample rate allowed by the hardware sample rate timer is 1 kHz. However, lower sample rates can be achieved by discarding samples in software, as is done in ionosonde mode. Hardware sample rates are programmed from the host PC by setting the rate dividers for the ADC and DAC timers.

$$\text{hardware sample rate} = \frac{2,5 * 10^6}{2 * \text{rate divider}} \text{Hz}$$

The hardware sample rate in meteor mode is 1024,6 Hz (divider = 1220). In ionosonde

mode two sample rates are used, the second being 2049,2 Hz (divider = 610).

DAC 1 is used in conjunction with the external gate to synchronise the interrupt timer in ionosonde mode (Figure [F-4]). In meteor mode it is used to switch the external analog multiplexer to enable sampling of four channels. Due to the delays associated with samples being received by the TMS320, the multiplexer is switched at what appears to be an illogical time from a software point of view. That is it is switched immediately after the sample from channel A is received, but analysis of the timing diagram in Figure [F-5] shows that this makes sense. The only time the DSP-16 operates in meteor mode is when running a *Meteor* system operation.

DAC 2 is used in both modes to provide an analog spectrum out. This is the combined and scaled power spectrum for channels A & B (either antennas 1 & 2 or 3 & 4 in meteor mode). The power spectrum array is simply scanned and the value at each position output to the DAC. Periodically a noisy signal drifts across the scope. This occurs whenever the power spectrum array is output to the DAC at exactly the same time the new power spectrum is being calculated. A synchronisation pulse is provided to trigger the scope as shown in Figure [F-6].

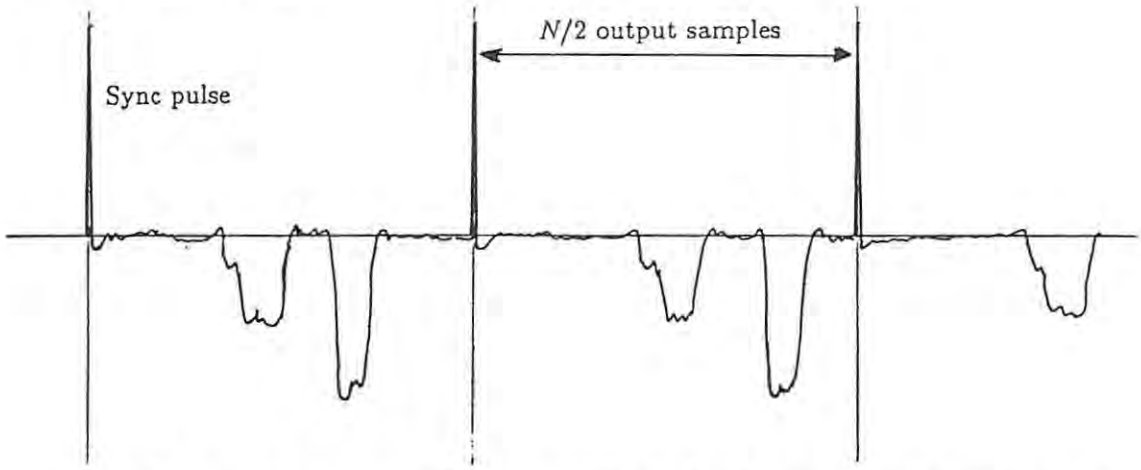


Figure F-6: Analog spectrum from the DSP-16's DAC 2 output to an oscilloscope

Note: Due to the length of the ADC interrupt service routine, interrupts 1 & 2 can be lost as interrupts are disabled within interrupt service routines. This does not affect overall system performance.

F.4 Analog connectors

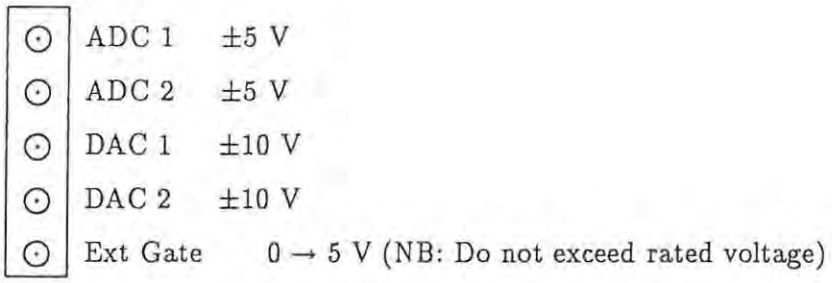


Figure F-7: Analog connectors on the backplane of the DSP-16 card [Ariel 1987]

The analog to digital converters have an adjustable range of between ±1 V to ±10 V peak. The external gate uses TTL logic levels and has no input protection. When the external gate is above 3,5 V or left floating, the interrupt timer is enabled (see Figure [F-8]).

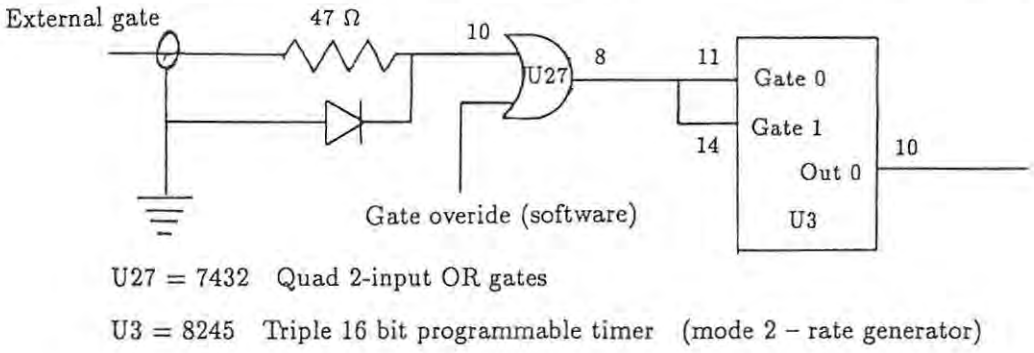


Figure F-8: DSP-16 external gate circuitry

F.5 Driver software

Originally control of the DSP-16 from the host PC was via an interrupt driven, memory resident assembly language routine supplied by Ariel. This routine was loaded from DOS on boot-up. A Turbo Pascal version 5 language interface to this memory resident routine was adapted from the version 3 interface (supplied by Ariel), by the author, basically to allow for version 5's stronger type checking. This approach has now been superseded as Turbo Pascal 5 allows for linking of assembler routines directly, with the use of the $L filename.obj compiler directive.

To produce the object file DSP30TP5.OBJ place the following assembler files in the

same directory:

DSP300.ASM

DSP301.ASM

DSP302.ASM

DSP303.ASM

DSP304.ASM

DSP30TP5.ASM

SEG.MAC

and then run: **TASM[2] /DUNIT DSP30TP5**

The driver functions are located in the unit DSP16TP.PAS, where they are declared as EXTERNAL. The compiled unit DSP16TP.TPU contains the assembler routine code, and after compilation the object file is no longer required.

The TMS320 hex file DSP16TMS.HEX is down-loaded to the DSP-16 card during initialisation of the system. The TMS320 source code can be found in the files:

DSP16TMS.ASM

DSP16TM1.ASM

To assemble the source code insure these files are both in the current directory and run:

ASM320 DSP16TMS

To run the debugger enter:

DSPBUG DSP16TMS

If a filename is not specified then the debugger will down-load the resident monitor (RESMON) to the DSP-16. Certain modifications are required to the DSP16TMS source code to allow it to run with the debugger, but these changes are documented in the code itself.

Files with the extension .TBL each contain a look-up table used to perform digital filtering in meteor mode. Such a file will be down-loaded to the DSP card on commencement of a meteor operation, if specified in the system operation. Alternatively the existing look-up table will be used. On boot-up, the default table contains correction factors for the external 100 Hz low pass filters only.

---

[2]Borlands Turbo Assembler must be used!

F.6 Hardware configuration

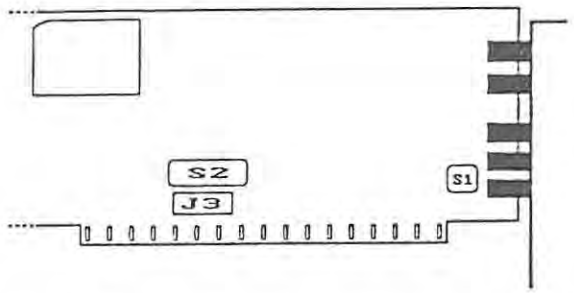


Figure F-9: DSP-16 switch locations [Ariel 1987]

F.6.1 I/O ports

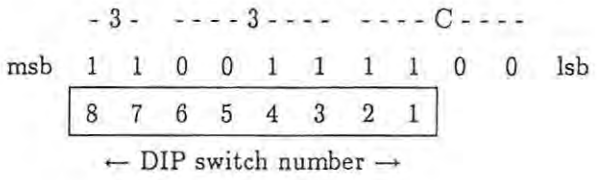


Figure F-10: DSP-16 I/O address switch with with default setting of 33Ch [Ariel 1987]

The DSP-16 can be relocated to any valid I/O address by changing the settings of the 8 position DIP switch S2. The DSP-16 uses 4 consecutive ports on the PC and at present spans addresses 33Ch–33Fh (828–831). If these settings are altered then the driver software needs to be informed of this fact. The DSP-16 I/O address may be represented as a 10 digit binary number. The correlation between the DIP switch setting and the bus address is shown in Figure [F-10] where 0=ON and 1=OFF. [Ariel 1987] Allocation of I/O ports on the IBM PC/AT is given in Figure [I-4].

F.6.2 Memory space

The DSP-16 occupies one 64 kbyte page of the PC's memory. This memory is only enabled when the assembly language drivers are actually accessing the memory. At any other time the memory will be invisible to all other programs. The 4 position DIP switch S1 is used to select one of sixteen possible pages as shown in Table [F-1]. [Ariel 1987] Assignment of memory on the IBM PC/AT is given in Figure [I-2]. Currently page D

(D0000h–DFFFFh) is used, but should this be changed then the driver software will need to be notified of the new memory location.

Memory page	Four position switch 1	2	3	4
0	ON	ON	ON	ON
1	ON	ON	ON	OFF
2	ON	ON	OFF	ON
3	ON	ON	OFF	OFF
4	ON	OFF	ON	ON
5	ON	OFF	ON	OFF
6	ON	OFF	OFF	ON
7	ON	OFF	OFF	OFF
8	OFF	ON	ON	ON
9	OFF	ON	ON	OFF
A	OFF	ON	OFF	ON
B	OFF	ON	OFF	OFF
C	OFF	OFF	ON	ON
D	OFF	OFF	ON	OFF
E	OFF	OFF	OFF	ON
F	OFF	OFF	OFF	OFF

Table F-1: DSP-16 memory address switch options [Ariel 1987]

F.6.3 Interrupts

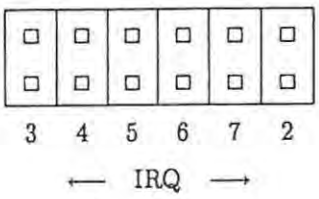


Figure F-11: DSP-16 interrupt jumper switch J3 [Ariel 1987]

The DSP-16 host interrupt may be selected by connecting a shorting plug vertically to two pins of the 12 pin header J3 at the bottom edge of the DSP-16 board as shown in

Figure [F-11]. In addition the driver software must explicitly enable interrupts, and an interrupt handler must be installed. At present IRQ 5 is used to notify the PC of data to be uploaded. Note that on the IBM AT IRQ 2 is mapped to IRQ 9 due to its expanded interrupt capability. IBM assignment of interrupts is given in Figure [I-1]. [Ariel 1987]

F.7 DSP-16 Plus

This is the upgraded version of the DSP-16 and provides greater noise immunity on the analog inputs. The original jack plugs on the analog signal lines have been upgraded to mini-XLR audio connectors on the later version.

From a software aspect, the two are compatible with one exception. A 'bug' on the original DSP-16 which resulted in the DAC outputs being inverted with respect to ground, has been corrected on the DSP-16 Plus. Thus the constant *Volts*, used for signal switching via DAC 1, has a negative value when used with the DSP-16 and a positive value when used with its successor.

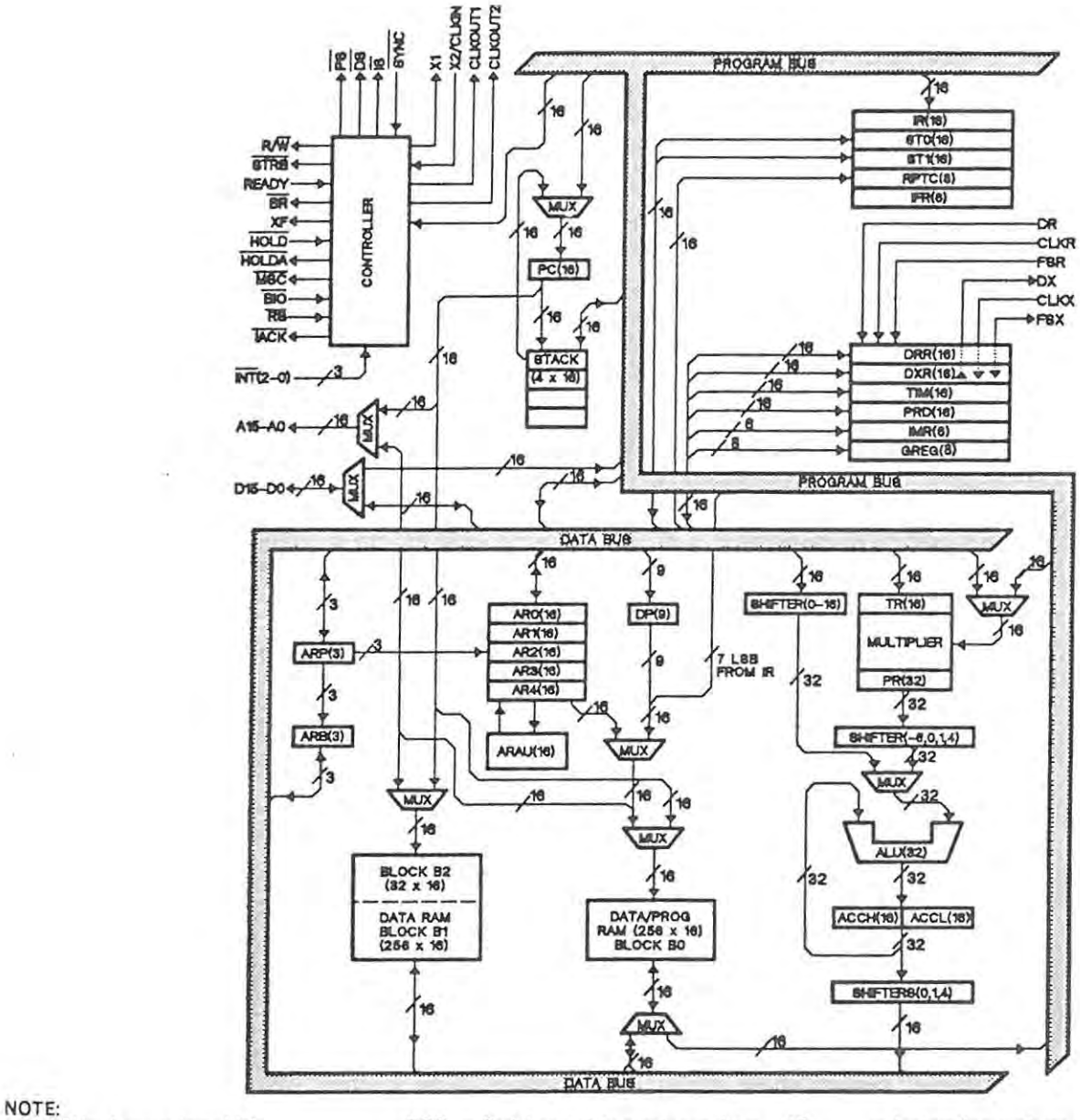
Appendix G

TMS32020 overview

G.1 Architecture

The TMS32020 is a 16 bit microprocessor designed specifically for digital signal processing applications. With a 200 *ns* instruction cycle time,[1] separate program and data busses, single cycle multiplication with a 32 bit accumulator, pipelining, and a high degree of parallelism affording it 'real time' capabilities. It features separate data and program address spaces of 64k words each and allows direct transfers between the two. In addition it has 544 words of 'on chip' data RAM, of which 256 words can be configured as program RAM (see Figure [F-2]). [TI 1986]

[1] The TMS320C25 which is an upgraded CMOS version has a 100 *ns* instuction cycle time and can be used as a plug-in replacement for the TMS32020



NOTE:

ACCH	= Accumulator high	DRR	= Serial port data receive register	IR	= Instruction register
ACCL	= Accumulator low	DXR	= Serial port data transmit register	PR	= Product register
ARAU	= Auxiliary register arithmetic unit	GREG	= Global memory allocation register	PRD	= Period register for timer
ARB	= Auxiliary register pointer buffer	IFR	= Interrupt flag register	TIM	= Timer
ARP	= Auxiliary register pointer	IMR	= Interrupt mask register	TR	= Temporary register
DP	= Data memory page pointer	RPTC	= Repeat instruction counter	ST0,ST1	= Status registers

Figure G-1: Block diagram of the TMS32020 Digital Signal Processor [TI 1986]

G.2 Registers, I/O ports and interrupts

GREG value			Data memory		Program memory	
Binary	Hex	Decimal	Range	# Words	Range	# Words
0000 0000	00	0		0	0-3FFF	16,384
1111 1111	FF	255	FF00-FFFF	256	0-3EFF	16,128
1111 1110	FE	254	FE00-FFFF	512	0-3DFF	15,872
1111 1100	FC	252	FC00-FFFF	1,024	0-3BFF	15,360
1111 1000	F8	248	F800-FFFF	2,048	0-37FF	14,336
1111 0000	F0	240	F000-FFFF	4,096	0-2FFF	12,288
1110 0000	E0	224	E000-FFFF	8,192	0-1FFF	8,192
1100 0000	C0	192	C000-FFFF	16,384		0

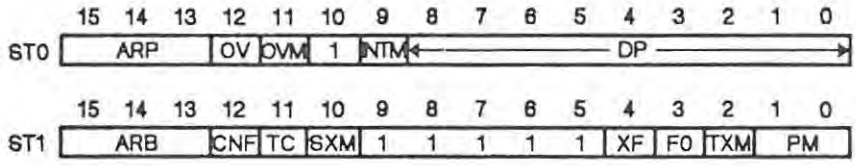
Table G-1: TMS320 global memory allocation register (GREG) as used by the DSP-16

The TMS32020 has an 'on chip' serial port which can operate bi-directionally. This port generates two interrupts (X-int and R-int) to trigger receive and transmit events. The DSP-16 uses this port to receive data from the two analogue to digital converters. This port is mapped to the base of the TMS320's data memory (Table [G-2]). The memory mapped registers may be accessed in the same manner as any other data memory location, with the exception that block moves are not allowed.

Register Name	Address location	Definition
DRR (15-0)	0	Serial port data receive register
DXR (15-0)	1	Serial port data transmit register
TIM (15-0)	2	Timer register
PRD (15-0)	3	Period register
IMR (5-0)	4	Interrupt mask register
GREG (7-0)	5	Global memory allocation register

Table G-2: TMS320 memory mapped registers [TI 1986]

The two status registers ST0 & ST1, can be written to data memory, allowing the processor status to be saved and restored during interrupts.



ARP	=	Auxiliary register pointer	OV	=	Overflow flag
OVM	=	Overflow mode	INTM	=	Interrupt mode
DP	=	Data memory page pointer	ARB	=	Auxiliary register pointer buffer
CNF	=	On-chip RAM configuration	TC	=	Test/control flag
SXM	=	Sign-extension mode	XF	=	XF pin status
FO	=	Format bit	TXM	=	Transmit mode bit
PM	=	Product shift mode			

Figure G-2: TMS320 status registers [TI 1986]

The DSP-16 uses the eight TMS320 I/O ports numbered 8 through 16. These ports are designated as shown in Table [G-3].

I/O Port	Description
8	Read/write BUFFER data
9	Write BUFFER address port
10	Read data from host I/O port
11	Enable/disable refresh mode
12	Host interrupt
13	Write data to DAC
14	Read host I/O port status
15	Write data to host I/O port

Table G-3: I/O ports of the TMS320 used by the DSP-16 [Ariel 1987]

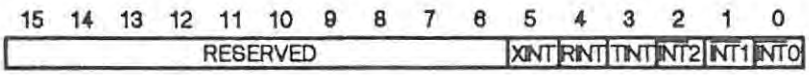


Figure G-3: TMS320 interrupt mask register [TI 1986]

Interrupt name	Memory Location	Priority	Function
$\overline{RS}$	0	1(highest)	External reset signal
$\overline{Int\ 0}$	2	2	External user interrupt #0
$\overline{Int\ 1}$	4	3	External user interrupt #1
$\overline{Int\ 2}$	6	4	External user interrupt #2
T-int	24	5	Internal timer interrupt
R-int	26	6	Serial port receive interrupt
X-int	28	7(lowest)	Serial port transmit interrupt
TRAP	30	N/A	TRAP instruction address

Table G-4: TMS320 Interrupts [TI 1986]

G.3 Addressing modes

There are three addressing modes offered by the TMS32020 (see Figure [G-4]). In the direct addressing mode, a 7 bit address is used to point to the memory location of the operand in the currently selected data page.[2] In indirect addressing mode the Auxiliary Register Pointer (ARP) points to the auxiliary register containing the 16 bit address of the required operand. Finally, the immediate operand addressing mode has the operand stored either in the instruction word, or in the word following the instruction opcode.

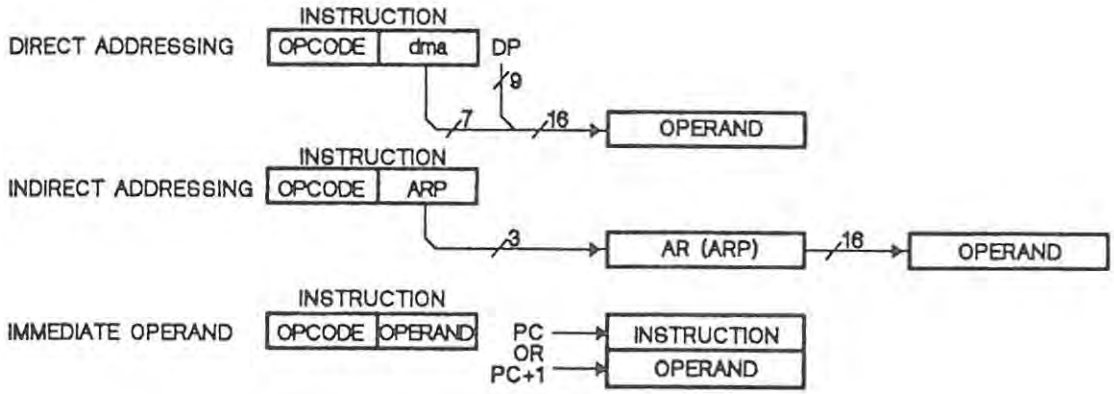


Figure G-4: TMS320 addressing modes [TI 1986]

[2]The 9 bit Data Page pointer (DP) is used to select 1 out of a possible 512 data pages of 128 words each (Figure [G-2])

G.4 Instruction set

Instruction Set Summary

Mnemonic	Description	# Words	Instruction Bit Code
			FEDCBA9876543210
ABS	Absolute value of accumulator	1	1100111000011011
ADD	Add to accumulator with shift	1	0000→S→I←D→
ADDH	Add to high accumulator	1	01001000I←D→
ADDS	Add to low accumulator with sign extension suppressed	1	01001001I←D→
ADDT	Add to accumulator with shift specified by T register	1	01001010I←D→
ADLK	Add to accumulator long immediate with shift	2	1101→S→0000001 0
AND	AND with accumulator	1	01001110I←D→
ANDK	AND immediate with accumulator with shift	2	1101→S→00000100
APAC	Add P register to accumulator	1	1100111000010101
B	Branch unconditionally	2	111111111←D→
BACC	Branch to address specified by accumulator	1	1100111000100101
BANZ	Branch on auxiliary register not zero	2	11111 0111←D→
BBNZ	Branch if TC bit ≠ 0	2	11111 0011←D→
BBZ	Branch if TC bit = 0	2	11111 0001←D→
BGEZ	Branch if accumulator ≥ 0	2	11111 01001←D→
BGZ	Branch if accumulator > 0	2	11111 00011←D→
BIOZ	Branch on I/O status = 0	2	11111 01010←D→
BIT	Test bit	1	1001←B→I←D→
BITT	Test bit specified by T register	1	01010111I←D→
BLEZ	Branch if accumulator ≤ 0	2	11111 00101←D→
BLKD	Block move from data memory to data memory	2	11111 1011←D→
BLKP	Block move from program memory to data memory	2	11111 1001←D→
BLZ	Branch if accumulator < 0	2	11111 00111←D→
BNV	Branch if no overflow	2	11111 01111←D→
BNZ	Branch if accumulator ≠ 0	2	11111 01011←D→
BV	Branch on overflow	2	11111 00001←D→
BZ	Branch if accumulator = 0	2	11111 01101←D→
CALA	Call subroutine indirect	1	1100111000100100
CALL	Call subroutine	2	11111 1101←D→
CMPL	Complement accumulator	1	1100111000100111
CMPR	Compare auxiliary register with auxiliary register AR0	1	11001110010100 CM
CNFD	Configure block as data memory	1	1100111000000100
CNFP	Configure block as program memory	1	1100111000000101
DINT	Disable interrupt	1	1100111000000001
DMOV	Data move in data memory	1	01010110I←D→
EINT	Enable interrupt	1	1100111000000000
FORT	Format serial port registers	1	1100111000000111FO
IDLE	Idle until interrupt	1	1100111000011111
IN	Input data from port	1	1000←PA←I←D→
LAC	Load accumulator with shift	1	0010→S→I←D→
LACK	Load accumulator immediate short	1	11001010←K→
LACT	Load accumulator with shift specified by T register	1	01000010I←D→
LALK	Load accumulator long immediate with shift	2	1101→S→00000001
LAR	Load auxiliary register	1	00110→R→I←D→
LARK	Load auxiliary register immediate short	1	11000→R←K→
LARP	Load auxiliary register pointer	1	0101010110001→R→
LDP	Load data memory page pointer	1	01010010I←D→
LDPK	Load data memory page pointer immediate	1	1100100←K→
LPH	Load high P register	1	01010011I←D→
LRLK	Load auxiliary register long immediate	2	11010→R→00000000
LST	Load status register ST0	1	01010000I←D→
LST1	Load status register ST1	1	01010001I←D→
LT	Load T register	1	00111100I←D→
LTA	Load T register and accumulate previous product	1	00111101I←D→
LTD	Load T register, accumulate previous product, and move data	1	00111111I←D→
LTP	Load T register and store P register in accumulator	1	00111110I←D→

Instruction Set Summary (concluded)

Mnemonic	Description	# Words	Instruction Bit Code
			FEDCBA9876543210
LTS	Load T register and subtract previous product	1	01011011I←D→
MAC	Multiply and accumulate	2	01011101I←D→
MACD	Multiply and accumulate with data move	2	01011100I←D→
MAR	Modify auxiliary register	1	01010101I←D→
MPY	Multiply (with T register, store product in P register)	1	00111000I←D→
MPYK	Multiply immediate	1	101←K→
NEG	Negate accumulator	1	1100111000100011
NOP	No operation	1	0101010100000000
NORM	Normalize contents of accumulator	1	1100111010100010
OR	OR with accumulator	1	01001101I←D→
ORK	OR immediate with accumulator with shift	2	1101→S→00000101
OUT	Output data to port	1	1110→PA→I←D→
PAC	Load accumulator with P register	1	1100111000010100
POP	Pop top of stack to low accumulator	1	1100111000011101
POPD	Pop top of stack to data memory	1	01111010I←D→
PSHD	Push data memory value onto stack	1	01010100I←D→
PUSH	Push low accumulator onto stack	1	1100111000011100
RET	Return from subroutine	1	1100111000100110
ROVM	Reset overflow mode	1	1100111000000010
RPT	Repeat instruction as specified by data memory value	1	01001011I←D→
RPTK	Repeat instruction as specified by immediate value	1	11001011←K→
RSXM	Reset sign-extension mode	1	1100111000000110
RTXM	Reset serial port transmit mode	1	1100111000100000
RXF	Reset external flag	1	1100111000001100
SACH	Store high accumulator with shift	1	01101→X→I←D→
SACL	Store low accumulator with shift	1	01100→X→I←D→
SAR	Store auxiliary register	1	01110→R→I←D→
SBLK	Subtract from accumulator long immediate with shift	2	1101→S→00000011
SFL	Shift accumulator left	1	1100111000011000
SFR	Shift accumulator right	1	1100111000011001
SOVM	Set overflow mode	1	1100111000000011
SPAC	Subtract P register from accumulator	1	1100111000010110
SPM	Set P register output shift mode	1	1100111000000 10 PM
SQRA	Square and accumulate	1	00111001I←D→
SQRS	Square and subtract previous product	1	01011010I←D→
SST	Store status register ST0	1	01111000I←D→
SST1	Store status register ST1	1	01111001I←D→
SSXM	Set sign-extension mode	1	1100111000000111
STXM	Set serial port transmit mode	1	1100111000100001
SUB	Subtract from accumulator with shift	1	0001→S→I←D→
SUBC	Conditional subtract	1	01000111I←D→
SUBH	Subtract from high accumulator	1	01000100I←D→
SUBS	Subtract from low accumulator with sign extension suppressed	1	01000101I←D→
SUBT	Subtract from accumulator with shift specified by T register	1	01000110I←D→
SXF	Set external flag	1	1100111000001101
TBLR	Table read	1	01011000I←D→
TBLW	Table write	1	01011001I←D→
TRAP	Software interrupt	1	1100111000011110
XOR	Exclusive-OR with accumulator	1	01001100I←D→
XORK	Exclusive-OR immediate with accumulator with shift	2	1101→S→00000110
ZAC	Zero accumulator	1	1100101000000000
ZALH	Zero low accumulator and load high accumulator	1	01000000I←D→
ZALS	Zero accumulator and load low accumulator with sign-extension suppressed	1	01000001I←D→

Figure G-5: Summary of the TMS32020 instruction set [TI 1986]

Appendix H

DSP-16 software flow diagrams

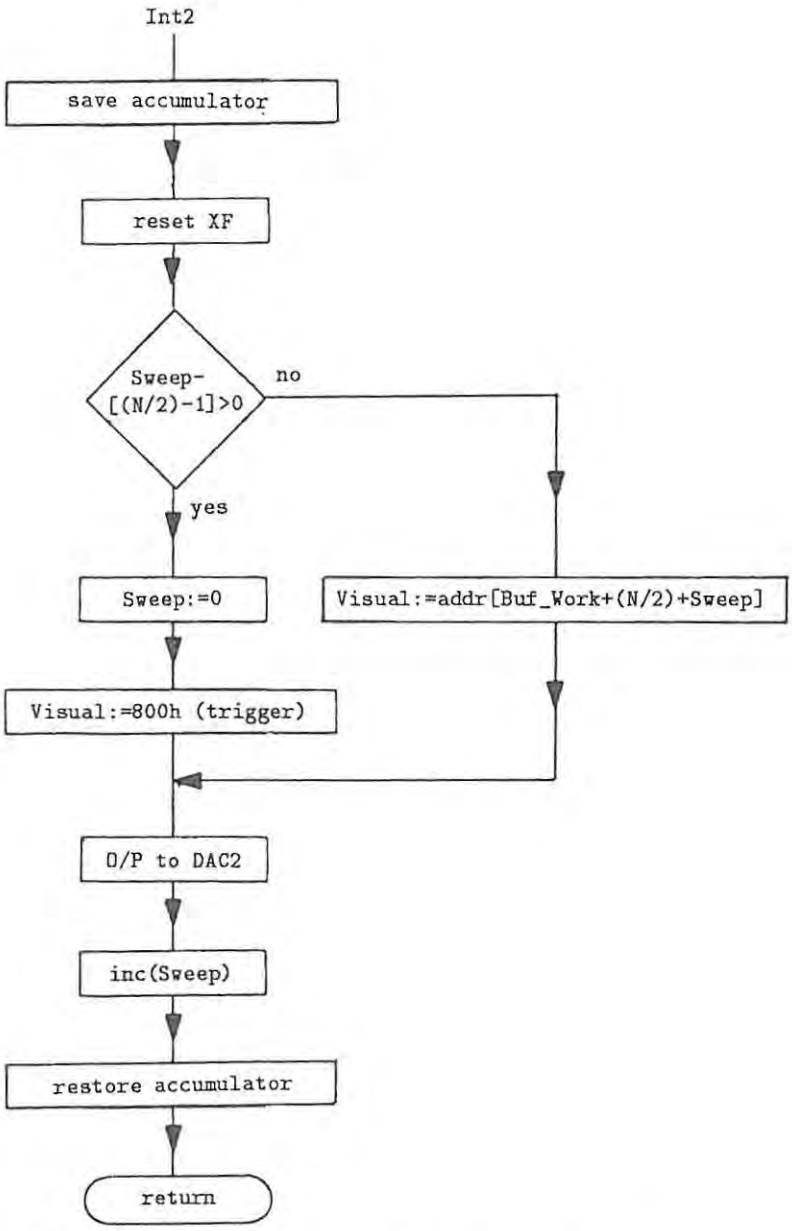


Figure H-1: DSP-16 interrupt 2 service routine

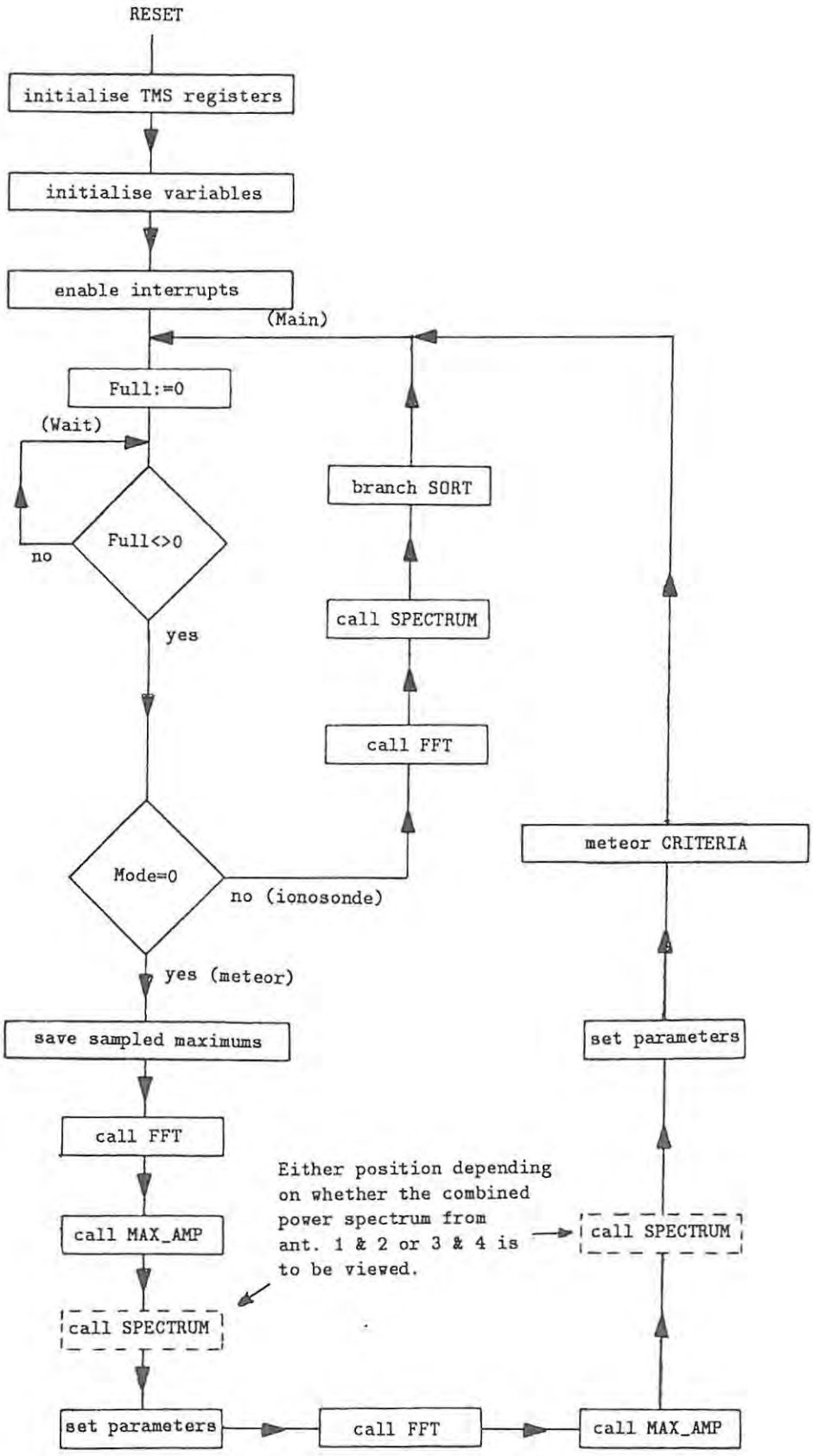


Figure H-2: DSP-16 program flow

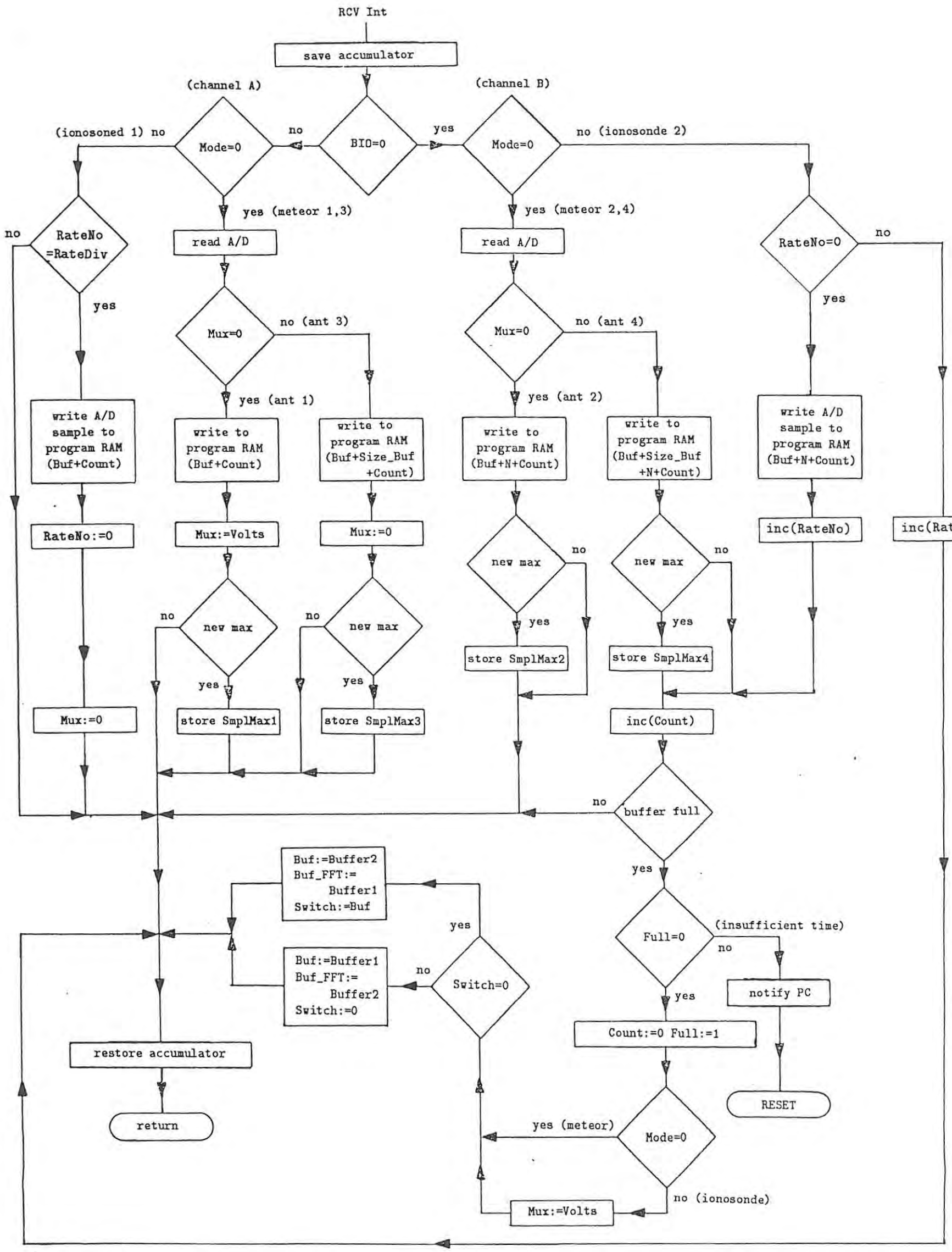


Figure H-3: DSP-16 ADC interrupt service routine

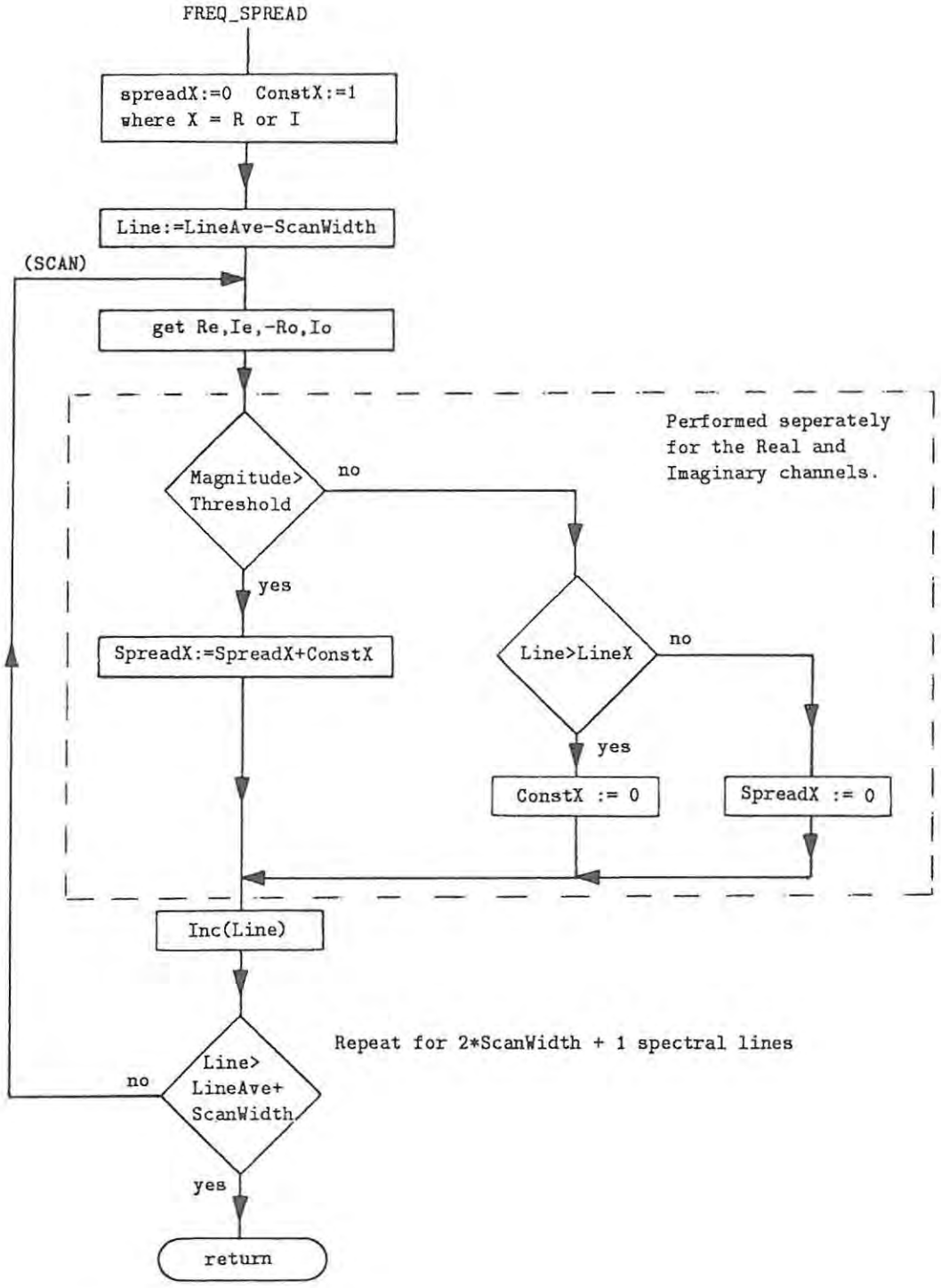


Figure H-4: Determination of frequency spread in meteor mode

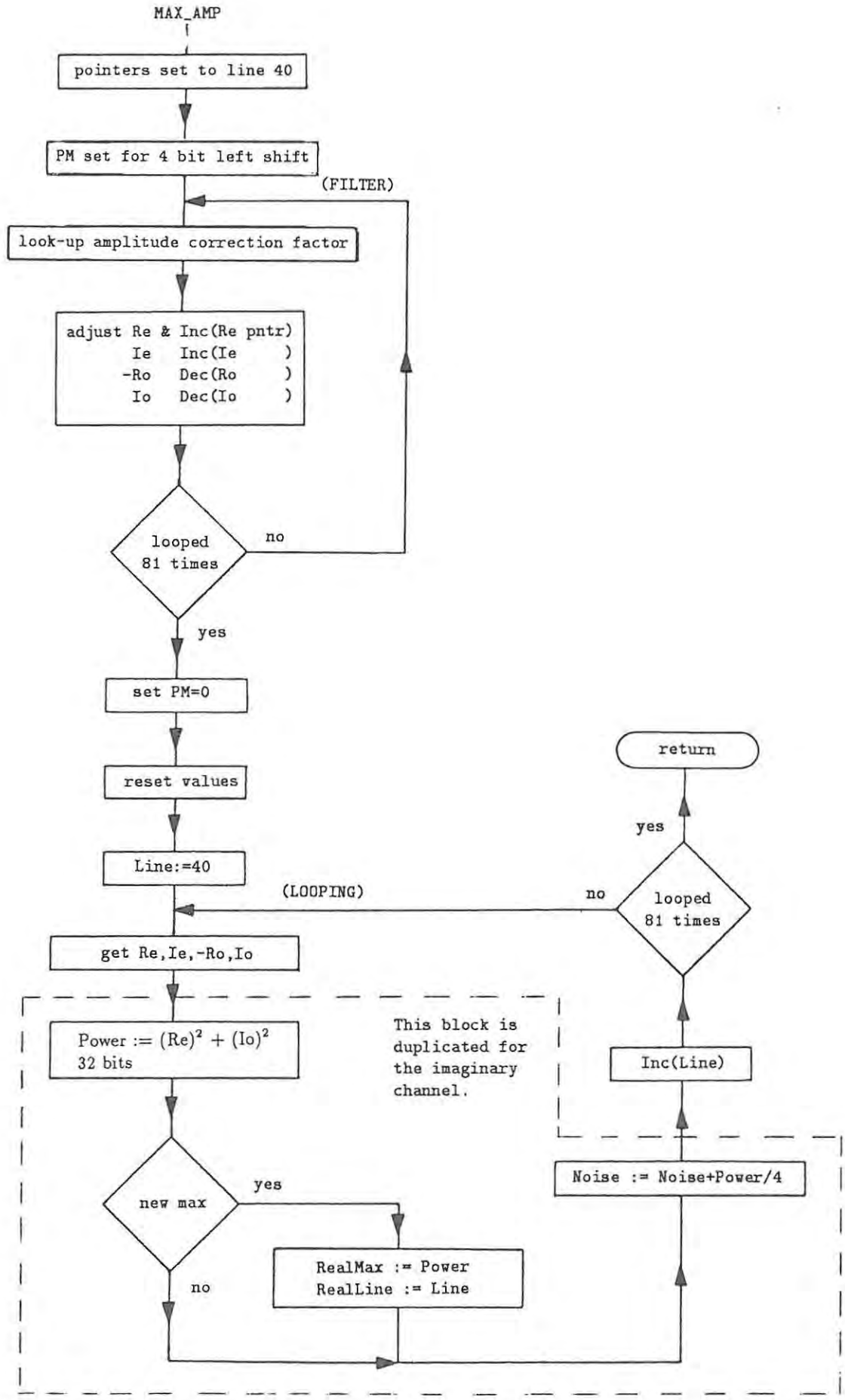


Figure H-5: Digital filter and determination of the maximum frequency component

Appendix I

PC memory map, I/O and interrupt vector table

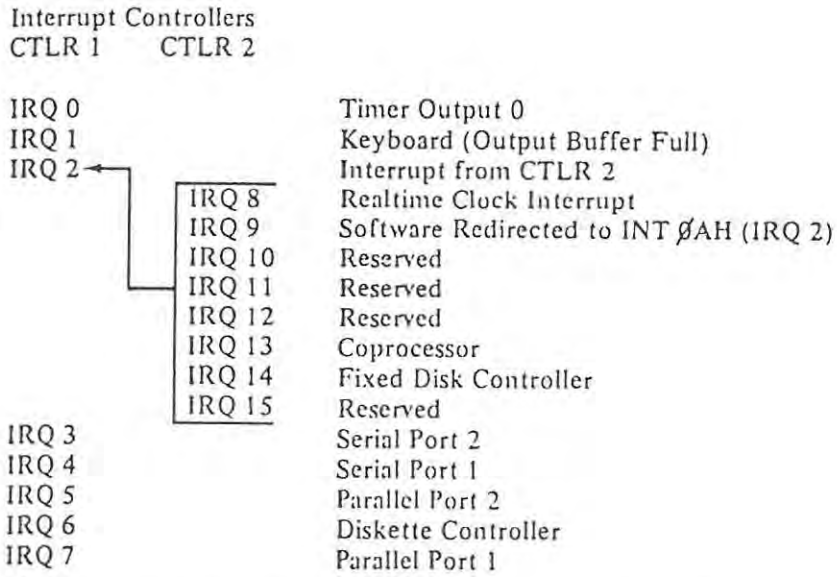


Figure I-1: IBM AT interrupt request lines [IBM 1983]

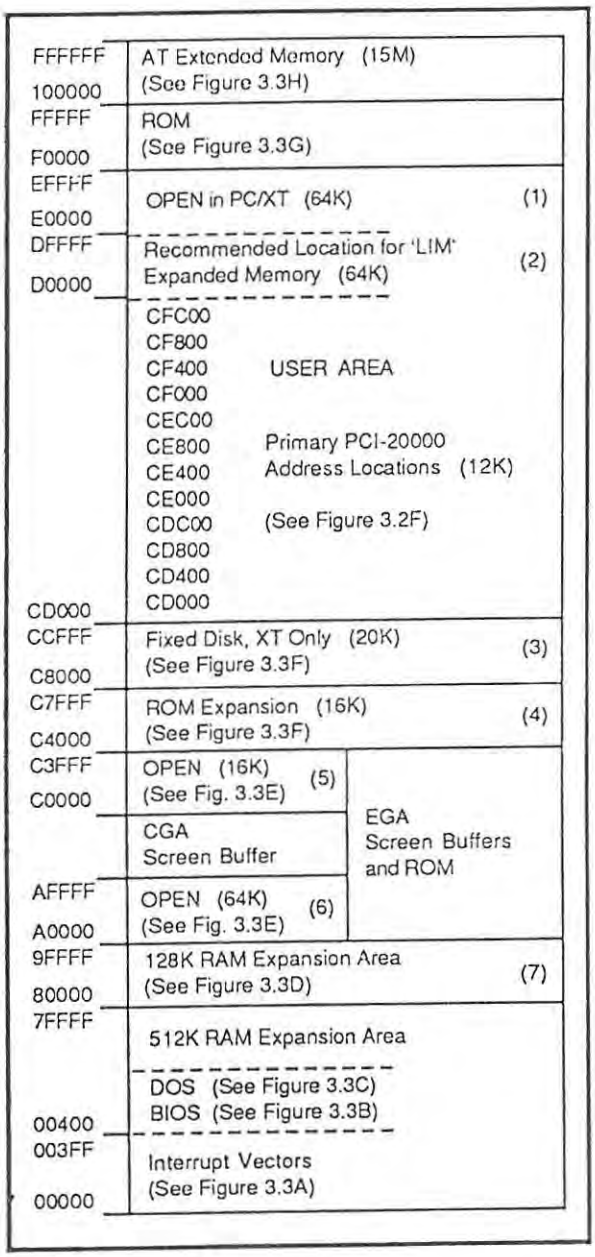


Figure I-2: Memory map for PC/XT/AT [Burr-Brown 1987]

```
00000 -00003 =Interrupt 0, divide-by-zero error.
00004 -00007 =Interrupt 1, single-step operation.
00008 -0000B=Interrupt 2, non-maskable interrupt.
0000C-0000F =Interrupt 3, break-point.
00010 -00013 =Interrupt 4, arithmetic overflow.
00014 -00017 =Interrupt 5, BIOS print-screen routine.
00018 -0001B=Interrupt 6, reserved.
0001C-0001F =Interrupt 7, reserved.
00020 -00023 =Interrupt 8, hardware timer 18.2/sec.
00024 -00027 =Interrupt 9, keyboard.
00028 -0002B=Interrupt A, reserved.
0002C-0002F =Interrupt B, communications.
00030 -00033 =Interrupt C, communications.
00034 -00037 =Interrupt D, alternate printer.
00038 -0003B=Interrupt E, floppy disk atten signal.
0003C-0003F =Interrupt F, printer control.
00040 -00043 =Interrupt 10, invokes BIOS video I/O service routine.
00044 -00047 =Interrupt 11, invokes BIOS equipment configuration
              check.
00048 -0004B=Interrupt 12, invokes BIOS memory-size check.
0004C-0004F =Interrupt 13, invokes BIOS disk I/O service routines.
00050 -00053 =Interrupt 14, invokes BIOS RS-232 I/O routines.
00054 -00057 =Interrupt 15, invokes BIOS cassette I/O, extended AT
              service routines.
00058 -0005B=Interrupt 16, invokes BIOS keyboard I/O routine.
0005C-0005F =Interrupt 17, invokes BIOS printer I/O.
00060 -00063 =Interrupt 18, ROM BASIC.
00064 -00067 =Interrupt 19, invokes BIOS boot-strap start-up routine.
00068 -0006B=Interrupt 1A, invokes BIOS time-of-day routines.
0006C-0006F =Interrupt 1B, BIOS ctrl-break control.
00070 -00073 =Interrupt 1C, gen at timer clock tick.
00074 -00077 =Interrupt 1D, video initialization control param pointer.
00078 -0007B=Interrupt 1E, disk parameter table pointer.
0007C-0007F =Interrupt 1F, graphics character table pointer.
00080 -00083 =Interrupt 20, invokes DOS program termination.
00084 -00087 =Interrupt 21, invokes all DOS function calls.
00088 -0008B=Interrupt 22, user-created, DOS-controlled interrupt
              routine invoked at program end.
0008C-0008F =Interrupt 23, user-created, DOS-controlled interrupt
              routine invoked on keyboard break.
00090 -00093 =Interrupt 24, user-created, DOS-controlled interrupt
              routine invoked at critical error.
00094 -00097 =Interrupt 25, invokes DOS absolute disk read service.
00098 -0009B=Interrupt 26, invokes DOS absolute disk write service.
0009C-0009F =Interrupt 27, ends program and keeps program in
              memory under DOS.
000A0-000FF=Interrupts 28 through 3F, reserved.
00100 -00103 =Interrupt 40, disk I/O (XT).
00104 -00107 =Interrupt 41, fixed disk parameters (XT).
00108 -00123 =Interrupts 42 through 48, reserved.
00124 -00127 =Interrupt 49, keyboard supplement translation table
              pointer.
00128 -0017F=Interrupts 49 through 5F, reserved.
00180 -0019F=Interrupts 60 through 67, user-defined interrupts.
  PCI-20046S can be programmed to use any one of the interrupts in
  the range of 60 thru 67. Interrupt 60 is used by ASYST, version 1.53.
  Interrupt 60 is used by ASYST, version 1.53.
  Interrupt 67 is used by the Expanded Memory Manager.
001A0-001FF=Interrupts 68 through 7F, not used.
00200 -00217 =Interrupts 80 through 85, reserved for BASIC.
00218 -003C3=Interrupts 86 through F0, BASIC interpreter.
003C4-003FF=Interrupts F1 through FF, not used.
```

Figure I-3: The interrupt vector table [Burr-Brown 1987]

```
000 -01F  = DMA controller (8237A-5).
020 -03F  = Interrupt controller (8259A).
040 -05F  = Timer (8254).
060 -06F  = Keyboard (8042).
070 -07F  = NMI mask register, real-time clock.
080 -09F  = DMA page register (74LS612).
0A0 -0BF  = Interrupt controller 2 (8259A).
0C0 -0DF  = DMA controller 2 (8237A).
0F0 -0FF  = Math coprocessor.
1F0 -1F8  = Fixed disk.
200 -207  = Joystick (game controller).
258 -25F  = Intel "Above Board".
278 -27F  = Parallel printer (secondary).
300 -31F  = Prototype card.
060 -36F  = Reserved.
378 -37F  = Parallel printer (primary).
080 -38F  = SDLC or bisynchronous communications (secondary).
3A0 -3AF  = Bisynchronous communications (primary).
3B0 -3BF  = Monochrome adapter/printer.
3C0 -3CF  = EGA, reserved.
3D0 -3DF  = Color/graphics adapter.
3F0 -3F7  = Diskette controller.
3F8 -3FF  = Serial port (primary).
```

Figure I-4: IBM AT I/O map [Burr-Brown 1987]

```
A0000 -AFFFF = Enhanced Graphics Adapter (EGA) screen buffers.*
B0000 -B7FFF = Monochrome adapter or EGA.
B0000 -B0FFF = Monochrome screen buffer.
B1000 -B7FFF = Reserved for screen buffers.
B8000 -BFFFF = Color/graphics adapter (CGA) or EGA.
B8000 -BBFFF = CGA buffer.
BC000 -BFFFF = CGA/EGA screen buffers.
C0000 -C3FFF =EGA BIOS.*
*Suggested memory location for installation of PCI-20000 products.
```

Figure I-5: CRT screen buffers [Burr-Brown 1987]

Appendix J

Phase-locked loops

J.1 Comparison of types

A phase-locked loop (PLL) consists of a phase detector, amplifier and voltage controlled oscillator (VCO) and represents a blend of analog and digital techniques (Figure [J-1]). PPL's are now available as single chip devices in a range of technologies, and this powerful building block has been used in our application to perform frequency multiplication by hooking a counter device (divider) between the VCO and phase detector.

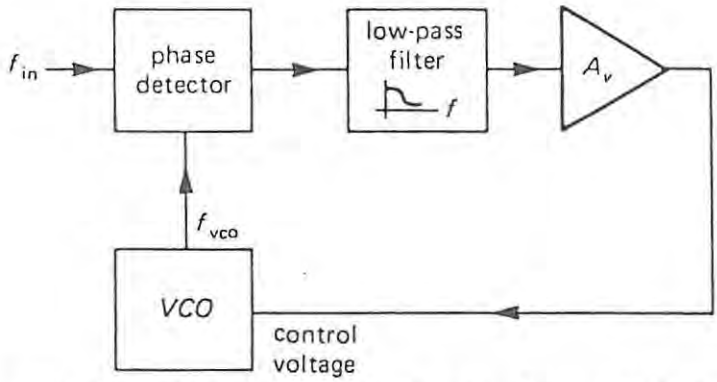


Figure J-1: Phase-locked loop block diagram [Horowitz & Hill 1984]

The phase detector compares two input frequencies generating an output that is a measure of there phase difference. If f_{IN} doesn't equal f_{VCO}, the phase error signal, after being filtered and amplified, causes the VCO frequency to deviate in the direction of f_{IN}. Under suitable conditions the VCO will quickly 'lock' to f_{IN} maintaining a fixed phase relationship with the input signal.

There are two basic types of phase detectors. Type-1 is designed to be driven by analog signals or digital square-wave signals, whereas the type-2 phase detector is driven by digital transitions (edges). Table [J-1] gives a comparison of the properties of the two types. For our application, where we have a relatively clean input signal, type-2 offered the best performance. A CMOS 74HC4046 device was used for its low power consumption and high speed.

	TYPE-1 exclusive-OR	TYPE-2 edge-triggered
Input duty cycle	50% optimum	irrelevant
Lock on harmonic?	yes	no
Rejection of noise	good	poor
Residual ripple at $2f_{IN}$	high	low
Lock range (L)	full VCO range	full VCO range
Capture range	fL ($f < 1$)	L
Output frequency when out of lock	f_{centre}	f_{min}

Table J-1: Comparison of the two types of phase detectors [Horowitz & Hill 1984]

PLL'S differ from most feedback devices in that the quantity measured to generate the error signal is not the same as the feedback quantity Here we measure phase, but adjust frequency which introduces a 90° phase lag in the loop (phase being the integral of the frequency). To compensate for this a second order loop is required which has additional low-pass filtering as shown in Figure [J-1]. This provides some 'flywheel' action, but reduces the capture range and increases the capture time. However, with type-2 phase detectors, a second order loop generates phase lock with zero phase difference between the reference and VCO.

J.2 Design study

Voltage controlled oscillator

$$f_{min} = \frac{1}{R_1(C_1 + 32\text{pF})}$$

Let $f_{min} = 7,5$ MHz and $C_1 = 10$ pF (empirical)
therefore $R_1 \approx 3k\Omega$ (2k2)

$$f_{max} = \frac{1}{R_2(C_1 + 32\text{pF})} + f_{min}$$

Let $f_{max} = 9$ MHz therefore $R_2 \approx 16k\Omega$ (15k)

These design equations differ from those given in the data books in that symbols R_1 and R_2 have been exchanged, as the author found this to give better results at the frequency of interest. Capacitor values were determined experimentally and all component values have been optimised for minimum jitter on the output signal. The 4046 has a chronic supply voltage dependency problem and the frequency range of operation has consequently been adjusted to allow for operation from 3,8 to 5,6 V. Values given in brackets represent the optimised values used.

Second order lowpass filter

$$R_4C_2 \approx \frac{6N}{f_{max}} - \frac{N}{2\pi\Delta f}$$

$N = 2048$ and let $C_2 = 100\mu$F
therefore $R_4 \approx 11k\Omega$ (1k)

$$(R_3 + 3k\Omega)C_2 \approx \frac{100N\Delta f}{(f_{max})^2} - R_4C_2$$

therefore $R_3 \approx 24k\Omega$ (120k)

Appendix K

Circuit descriptions

K.1 Timing Controller (TC)

The timing controller has been constructed using CMOS devices. This was necessitated by the need for battery backup. To meet the speed requirements high speed CMOS has been used, which is comparable in speed with LS TTL logic. A nominal battery voltage of 4.8 V is provided by four NiCad size AA cells.

A block diagram of the timing controller is given in Figure [K-1]. A 5 MHz, 1 MHz or 100 kHz signal is input from the frequency (time) standard via BNC conector J1. This input is isolated from the timing controller circuitry by the high speed opto-coupler U1. A minimum input signal of 2 Vrms is required at 8 mA.

The inverted input signal then passes to the clock frequency generator circuitry and the output 32,768 kHz from this is used to clock the real time clock (RTC) chip. In addition all the output strobes are derived from this signal by the strobe decode circuitry. Finally the various strobes and output signals pass through 3-state buffers U46 & U47 to connectors J3 and J2 respectively.

J2 is the sample strobe if required by external analog to digital converter cards. The output can be configured for operation on either the rising or falling edge of the strobe by shifting jumper JP3. J3 connects the timing controller to the signal switching unit and the BR-9034.

U46 and U47 disable all outputs during battery operation. The POWER OFF control line goes hi immediately the PC's +5 V rail drops below 4 V, via U31E and diodes D1–3.

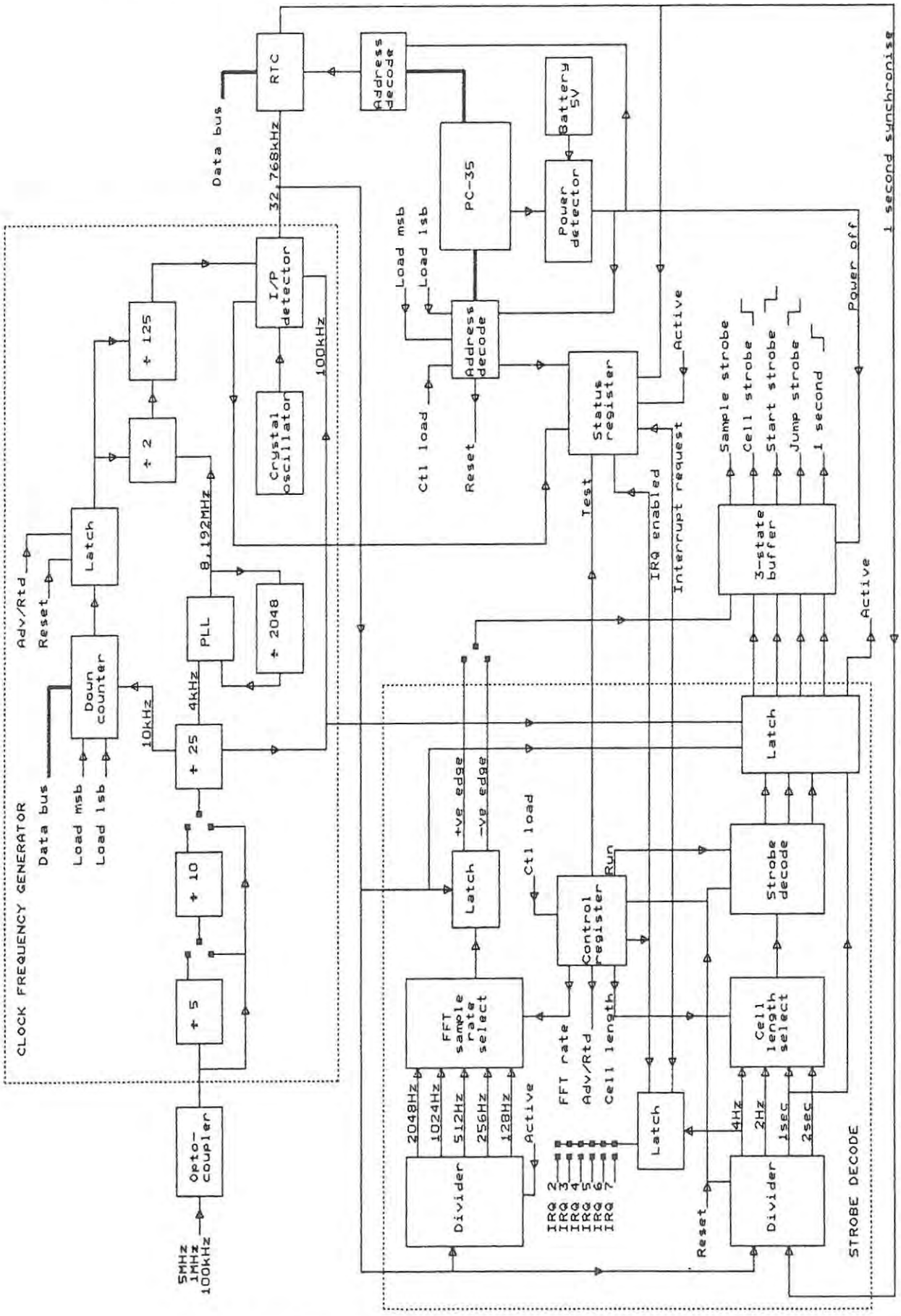


Figure K-1: Detailed block diagram of the timing controller

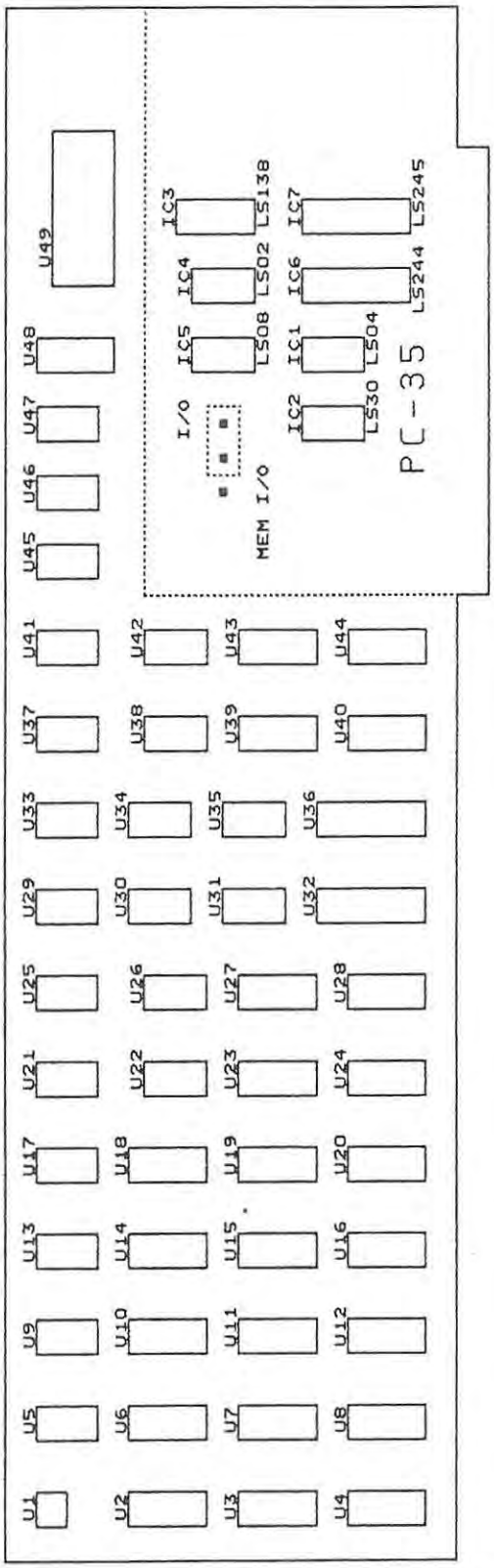


Figure K-2: Timing controller board layout

In addition all outputs are disabled when the signal from the external frequency standard is lost as the strobe decode circuitry is thus rendered non-functional. The POWER OFF line is used to inhibit all chip selects during power outings.

U49 is an ICM7170 and is the RTC chip. Pin 12 ($\overline{INTERRUPT}$) is used to synchronise the external circuitry and the real time clock. This is achieved by configuring the interrupt mask register on the ICM7170 to provide a one second interrupt. On interrupt this line goes lo and remains there until the interrupt status register is read. Reading the interrupt status register returns SYNC hi and holds U19 & U20 reset until the next one second interrupt, as shown in Figure [K-3]. Synchronisation is only necessary on power up and after setting the time on the RTC.

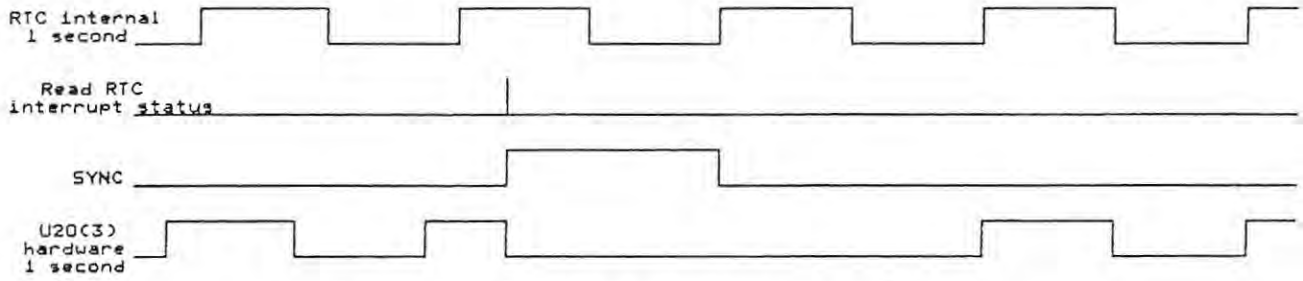


Figure K-3: Synchronisation of the hardware with the RTC

The strobe decode circuitry also provides a 4Hz output, used to interrupt the PC when the IRQ ENABLE bit in the control register (U32) is set (see Table [K-4]). This 1/4 second interrupt is used to drive the PC's software clock. The IRQ line remains hi until U33A is reset by reading the status register (U36), when it is pulled lo via resistor R9.

The status register (Table [K-3]) is not affected by reset and indicates the current board status. U33B is latched when the input signal from the frequency standard is lost, and remains latched until reset by reading the status register. The timing controller is reset by writing to I/O port 31Fh on the PC.

K.1.1 Clock frequency generator

Jumpers J1 & J2 allow for an input frequency of 5 MHz, 1 MHz or 100 kHz. This input is divided down to 4 kHz and fed to U7 which is configured as a type-2 phase locked loop (PLL, see Appendix J). The synthesised 8,192 MHz from the voltage controlled oscillator

(VCO) and 4,096 MHz from U10 then pass to the advance/retard circuitry.

When the SHIFTING control line is hi the output 32,768 kHZ from U14 is either doubled in frequency, ADV/-RTD line hi, or inhibited if ADV/-RTD is lo. Thus to shift system timing by one second in either direction, takes one second to complete.

Binary up/down counters U39, U40, U43 & U44 latch the shift required in steps of 100 μs. In other words, if these counters are loaded with the value 10, the resultant timing shift will be 1 ms. U44 & U43 are addressed at port 31Ch and contain the least significant byte (LSB). This byte must be loaded first as shifting begins immediately the MSB is loaded into counters U40 & U39.

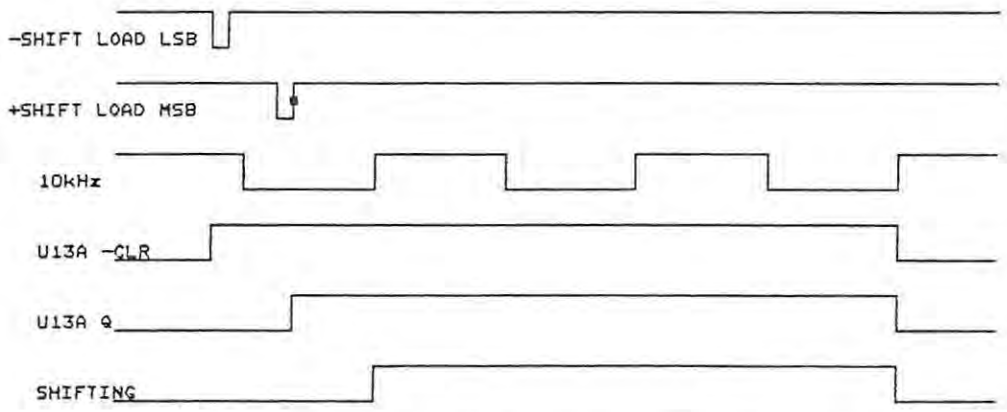


Figure K-4: Timing diagram for a 0.2 ms time shift

Once a shift greater than zero has been loaded, reset to flip flop U13A is released by OR-ing all the binary outputs of the counters together. This flip flop then changes state on the -SHIFT LOAD MSB line going hi. U13B is then enabled and sets SHIFTING hi on the rising edge of the next 10 kHz clock pulse. With SHIFTING hi the binary counters are enabled and will begin decrementing with the next 10 kHz clock pulse (see Figure [K-4]).

Note: If a hardware or software reset occurs while shifting, all flip flops will be reset. The output frequency will return to 32,768kHz and remaining programmed shift will be lost, unknown to the PC.

Decade counters U15 & U16,[1] and flip flop U21A form an input frequency detector circuit. If the signal from the frequency standard is lost, then the RTC will swap to the

[1]U15 & U16 are standard CMOS devices as their slower switching times provide a reset pulse long enough to be seen by flip flop U21A.

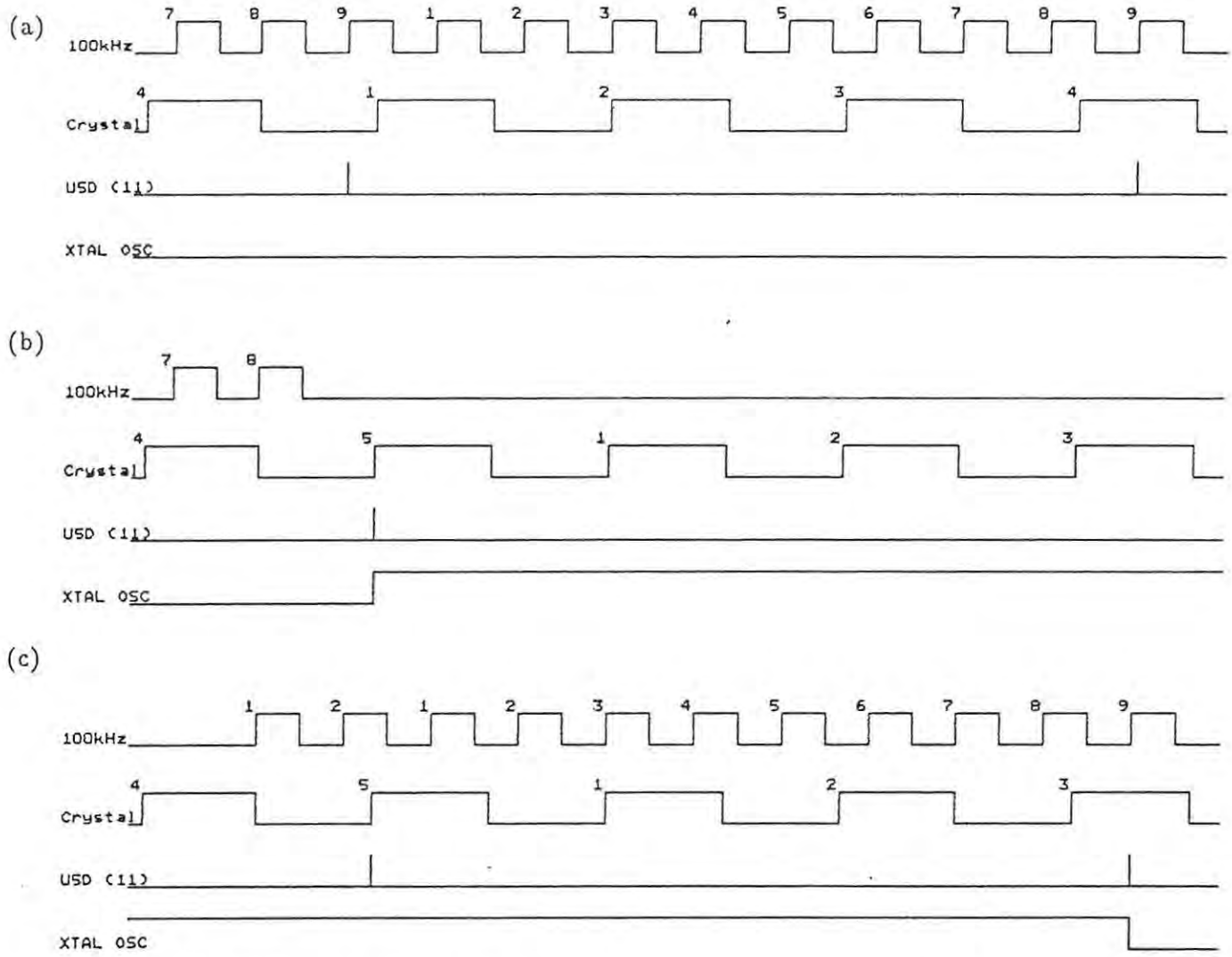


Figure K-5: Timing diagrams for the input frequency detector :- (a) normal operation, (b) frequency failure, (c) input re-established

onboard 32,768 kHz crystal oscillator with a dead time of less than 150 μs (see Figure [K-5] for timing diagrams). This sets the CRYSTAL OSCILLATOR line hi and disables all outputs to the signal switching unit. Should the input signal be re-established, then timing will revert back to the frequency standard, with a dead time of less than 100 μs and output strobes will be re-enabled. This is particularly useful for maintaining approximate timing position during temporary input frequency outings. The FREQ FAIL bit in the status register (U36) will remain set however until the status register is read.

Note: If a frequency failure occurs during a timing shift, then execution of the shift will halt until the input is re-established, after which it will continue unless the hardware is

reset in the meantime.

K.1.2 Strobe decode

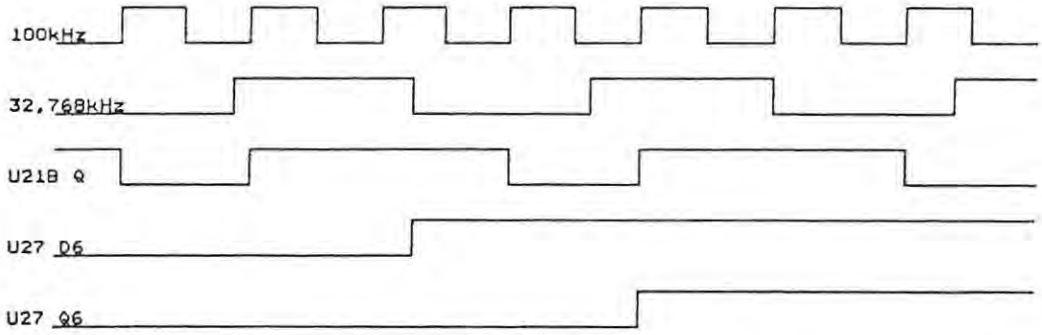


Figure K-6: Synchronisation of strobes with the frequency standard

The 32,768 kHz from the clock frequency generator circuitry is divided by binary dividers U19 and U20 to provide 4Hz, 2Hz and 1Hz signals. U19 & U20[2] are reset and inhibited while the SYNC line from the RTC (U49) is hi. This allows the 1Hz output to be synchronised with the second pulse of the RTC. The outputs of the 74HC4040 can not be used directly as strobes as they contain decoding 'glitches'. U19 and U20 thus change state on the falling edge of the 32,768 kHz clock, but these changes are only clocked through on the positive edge of the clock by flip flop U27.

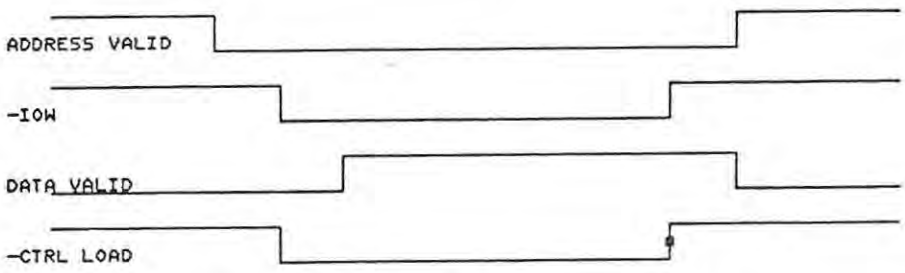


Figure K-7: Writing to the control register

In addition as the 32,768 kHz is derived from a phase locked loop it is subject to small variations in frequency. However the BR-9034 once locked to the external 5 MHz frequency standard has a ± 200 ns tolerance for variations in timing of the external start and jump strobes over a 10 μs interval. The positive edge of the 32,768 kHz clock is thus

[2]U19, U20, U52 & U54 are standard CMOS devices to reduce signal bounce.

synchronised, via flip flop U21, with the positive edge of the 100 kHz derived from the frequency standard (see Figure [K-6]).

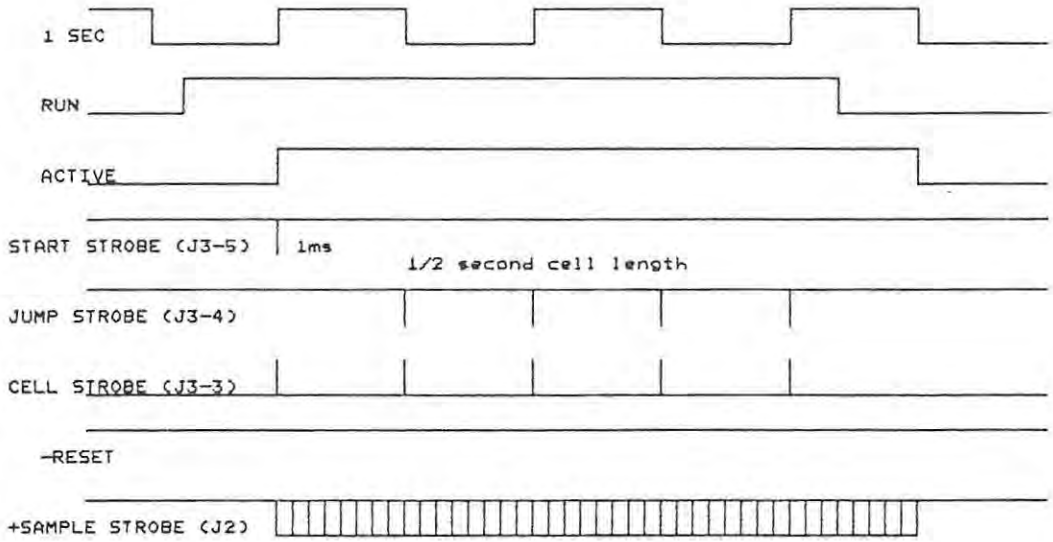


Figure K-8: Timing controller strobe decoding timing diagram

U32 is the control register. It is loaded when -CTRL LOAD line transits from a lo to a hi synchronously with the -IOW line going hi (Figure [K-7]). When bit 0 (RUN) in this register is set, then a start strobe is produced synchronously with the rising edge of the next one second pulse. Decade counter U24 in conjunction with flip flop U30B limit the width of the start, jump and cell strobes to 1 ms. U30A decodes the first cell strobe as the start strobe and there after as jump strobes for the duration of the sweep. The timing diagram is shown in Figure [K-8].

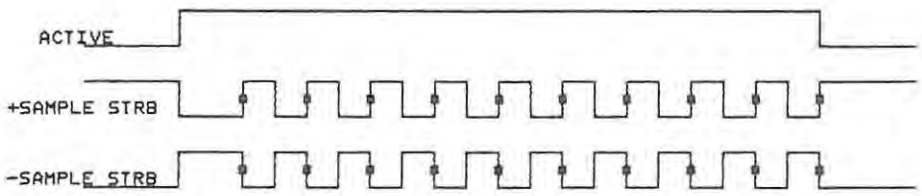


Figure K-9: Timing controller sample strobe timing diagram

The sample rate generator is provided for use with externally triggered analog to digital converter cards, allowing for triggering on either the rising or falling edge. The first sample is made exactly one sample period after the start of a cell. The sample rate generator is inhibited between sweeps by holding binary counter U54 reset (see Figure [K-9]).

Simultaneous with the start strobe the ACTIVE line goes hi and is held there by flip flop U41A until the end of the last cell unless -RESET goes lo. Operation is normally terminated by setting bit 0 (RUN) of the control register to zero. This allows sampling to continue till the end of the current cell. The control register is cleared on reset and sampling halts immediately (Figure [K-10]).

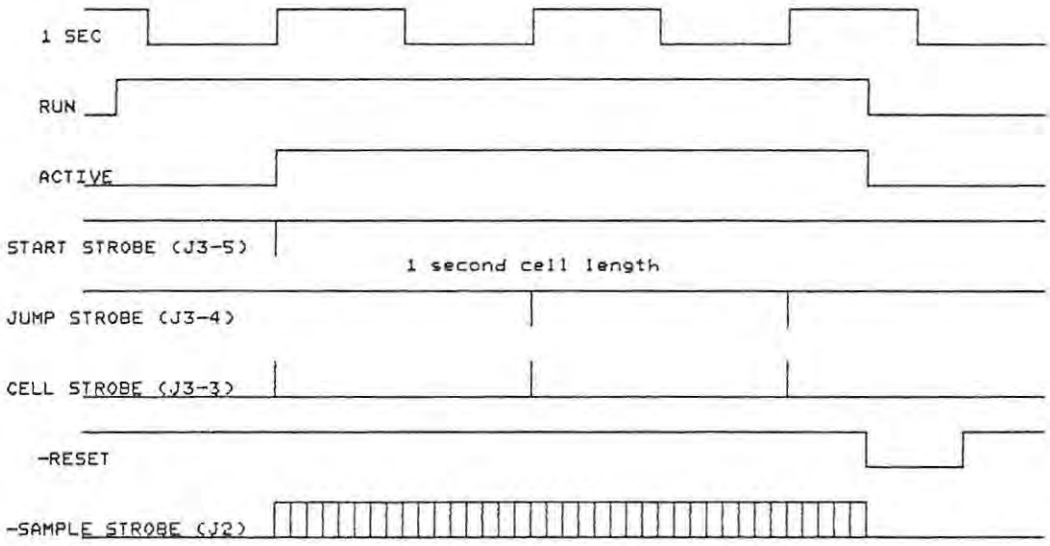


Figure K-10: Strobe decode timing diagram on reset

U52 is configured as a divide by two stage and allows for cell lengths of two seconds.

K.1.3 Connectors

J1 (miniature BNC) 5 MHz, 1 MHz, 100 kHz input from frequency standard. This input is optically isolated and requires a minimum input of 2 Vp-p at 8 mA.

J2 (miniature BNC) Sample strobe to external ADC cards. Either positive or negative edge strobing can be selected via a jumper.

J3 (9 pin D-type)

1 - GND

2 - 1 second; positive edge, 50% duty cycle square wave.

3 - cell strobe; positive edge, 1 ms duration.

4 - jump strobe; GND when active, 3-state, 1 ms.

5 - start strobe; GND when active, 3-state, 1 ms.

All outputs (J2 & J3) are 3-state during battery operation or loss of input from the frequency standard.

K.1.4 Power requirements

Supply voltage	4.0–5.5V
Battery current at 5V	33mA – Crystal Oscillator
	20mA – External frequency standard (100 kHz)
Battery charge current	Max 40 mA from 12 V rail

Table K-1: Timing controller power requirements

The prototype TC has been wire wrapped on a PC-35 prototyping card which comes complete with bus buffering and address decoding circuitry. The board layout is shown in Figure [K-2]. This TC is a simplified version and does not contain the sample strobe circuitry nor does it allow for cell lengths of 2 seconds. At present IRQ 3 is used for the quarter second interrupt.

K.1.5 Register addressing

300	1/100 seconds	R/W
301	hours	R/W
302	minutes	R/W
303	seconds	R/W
304	month	R/W
305	date	R/W
306	year	R/W
307	day-of-week	R/W
310	interrupt status	R
310	mask reg	W
311	command reg	W
31C	shift load lsb	W
31D	shift load msb	W
31E	control reg	W
31F	status reg	R
31F	Reset	W

Table K-2: TC port address allocation

7	Timing slip direction (also used to test for TC hardware)
6	Interrupt request pending
5	Synchronisation of the hardware with the RTC in progress
4	Interrupts enabled
3	Input frequency failure has occurred
2	Clock input from onboard oscillator
1	Time shift in progress
0	Hardware strobing active

Table K-3: Significance of TC status register bits

7	Timing slip direction	0 = retard
		1 = advance
6	Enable interrupt requests to the PC	
5,4,3	Sample rate	000 = 2048 Hz
		001 = 1024 Hz
		010 = 512 Hz
		011 = 256 Hz
		100 = 128 Hz
2,1	Cell length	00 = 0.25 sec
		01 = 0.5 sec
		10 = 1 sec
		11 = 2 sec
0	Start stobes on next 1 second edge (RUN)	

Table K-4: Programing the TC control register

K.2 Signal Switching unit (SS)

Figure [K-11] shows the RF and analog signal paths. Solid state devices[3] are used to perform the RF receive input signal switching. Mechanical relays are used to select the transmit antenna, as high power devices are required and the switching rate is low (once per ionogram). Figure [K-12] gives the antenna connections to each receiver.

The +5 V POWER ON line from the 4070 is used to power the Signal Switching unit on and off in conjunction with the BR-9034 on/off switch. A green LED on the front panel of the SS is used to indicate that all the required supply voltages are present.

[3] PAS-3 attenuator/switches

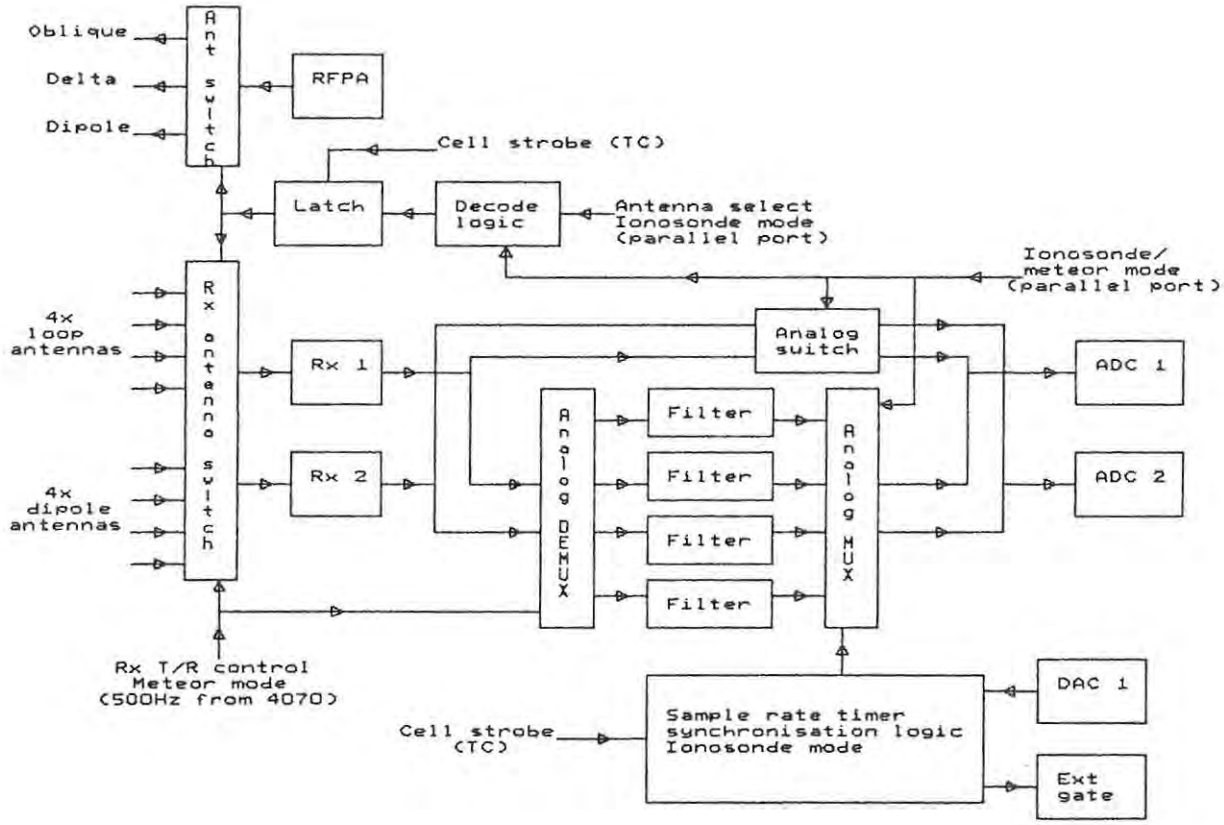


Figure K-11: Block diagram of the Signal Switching unit

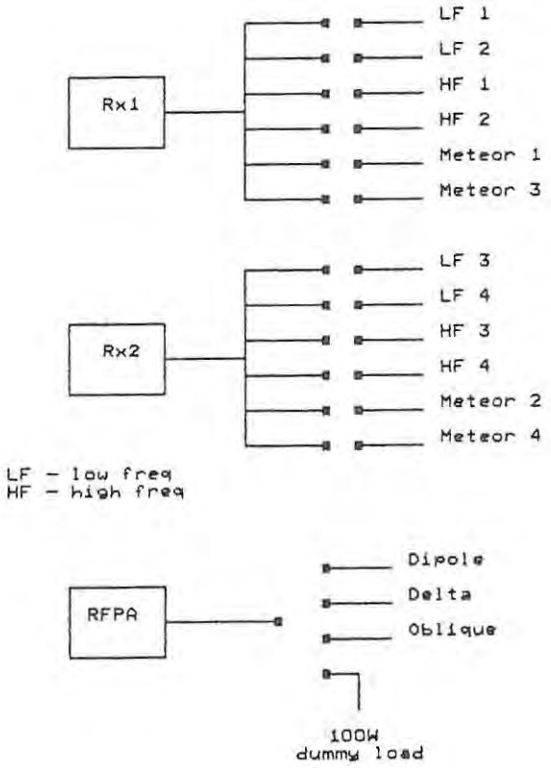


Figure K-12: RF signal path

Appendix L

Circuit diagrams

Clock Frequency Generator

STROBE DECODE

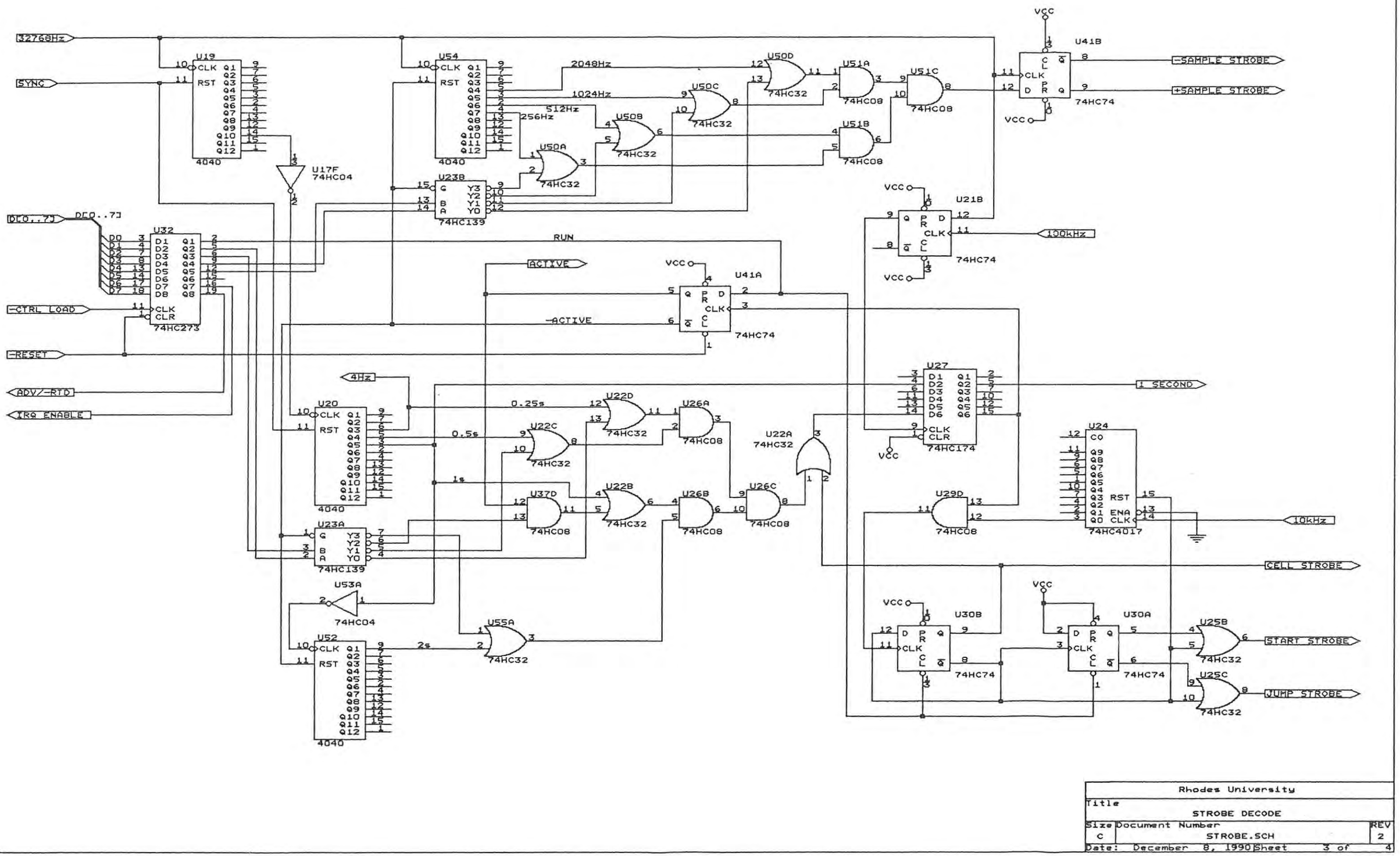
POWER OFF

PC-35

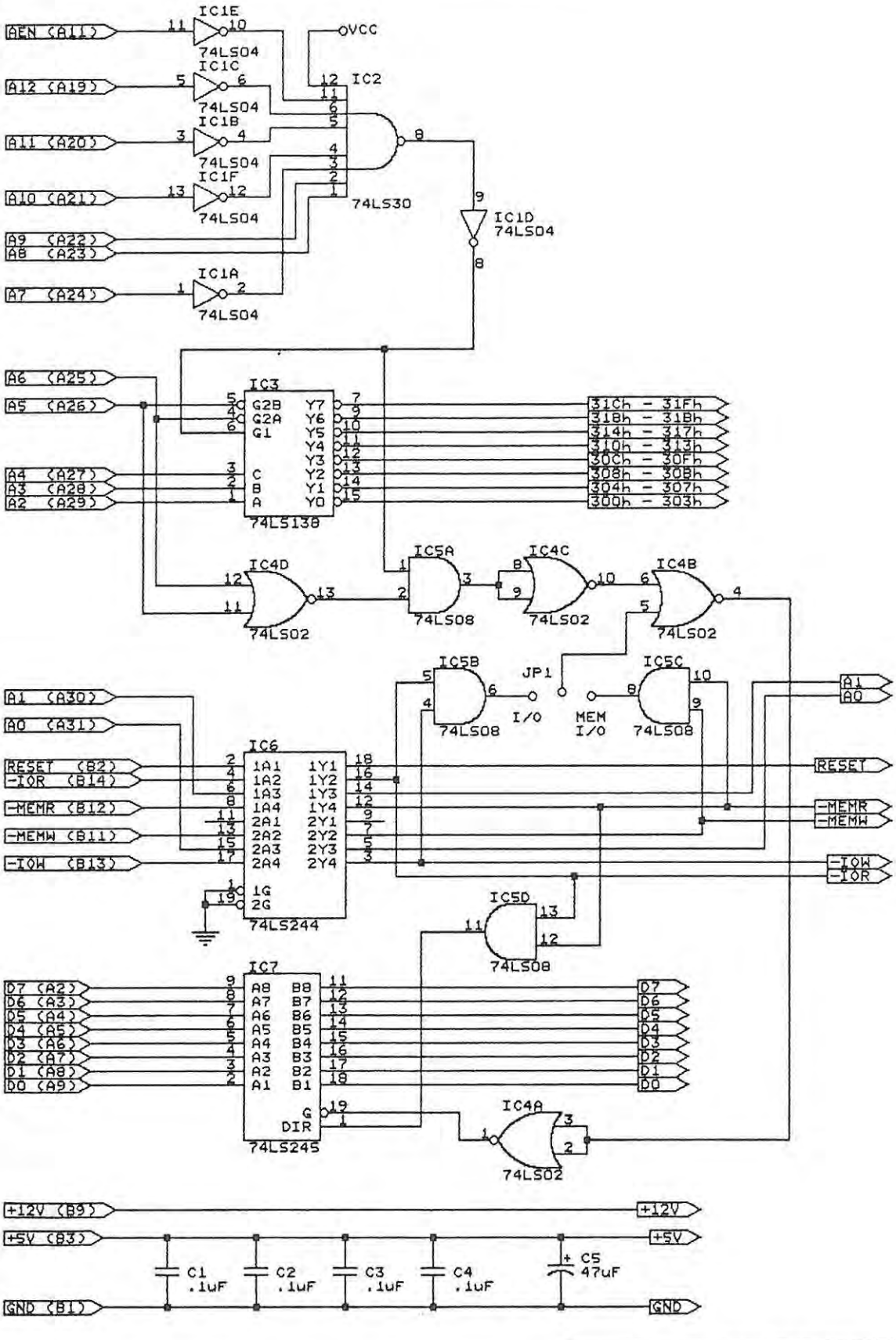
Rhodes University	
Title	TIMING CONTROLLER
Size: C	Document Number: TC.SCH
REV: 2	
Date: December 7, 1990	Sheet 1 of 4

5MHz, 1MHz, 100kHz
100kHz
10kHz
JP2
JP3
1MHz
100kHz
4kHz
8192kHz
U2
U3
U4
U5A
U8
U7
74HC4017
74HC32
74HC4046
C1 10pF
C2 100nF
R1 2k2
R2 15k
R3 120k
R4 1k
U6(2)
U6(4)
U6(8)
U6(9)
U6(10)
U6(12)
U6(13)
U6(15)
SHIFTING
ADV/-RTD
VCC
U13A
U13B
74HC74
U5C
U17C
74HC04
U9B
74HC08
U17A
U17B
U10
74HC4040
-RESET
U29A
DC0..7]
[DC0..7
U44
U42A
U34C
U25D
4075
74HC191
U34B
74HC32
U43
U42B
U42C
4075
U40
U38A
U38B
U39
U38C
-SHIFT LOAD MSB
-SHIFT LOAD LSB
U9D
U9C
U5B
4096kHz
U11
U12
U14
U17E
U18(2)
U18(15)
R5 10M
R6 100k
U18(3)
X1 32768Hz
C3 20pF
C4 20pF
U18(13)
U18(6)
U15
U16
4017
U5D
U21A
U9A
U25A
U26D
32768Hz
XTAL OSC
U17D

Rhodes University		
Title: CLOCK FREQUENCY GENERATOR		
Size: C	Document Number: CLK-FREQ.SCH	REV: 2
Date: December 8, 1990	Sheet 2 of 4	

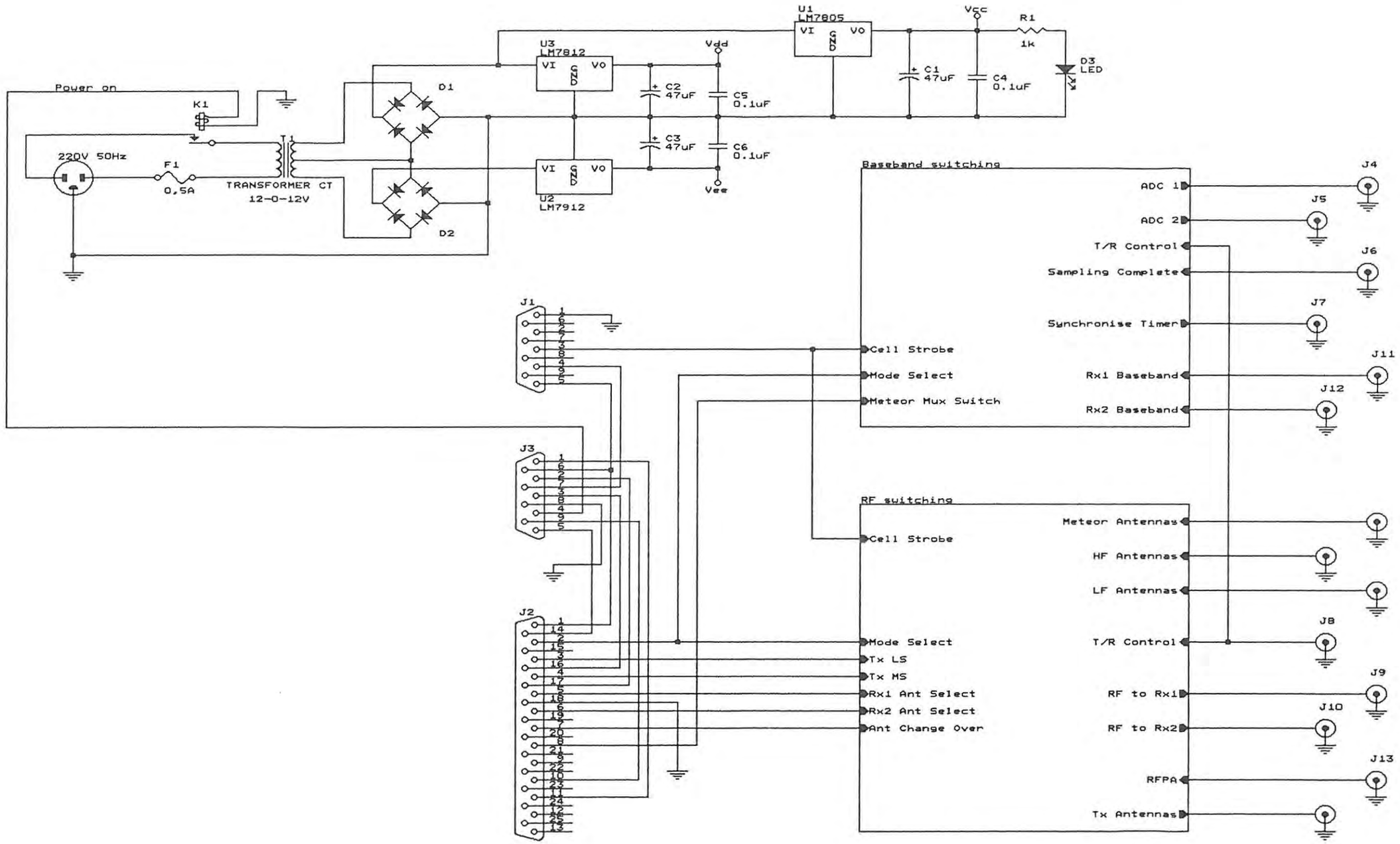


Rhodes University		
Title: STROBE DECODE		
Size: C	Document Number: STROBE.SCH	REV: 2
Date: December 8, 1990	Sheet: 3 of 4	

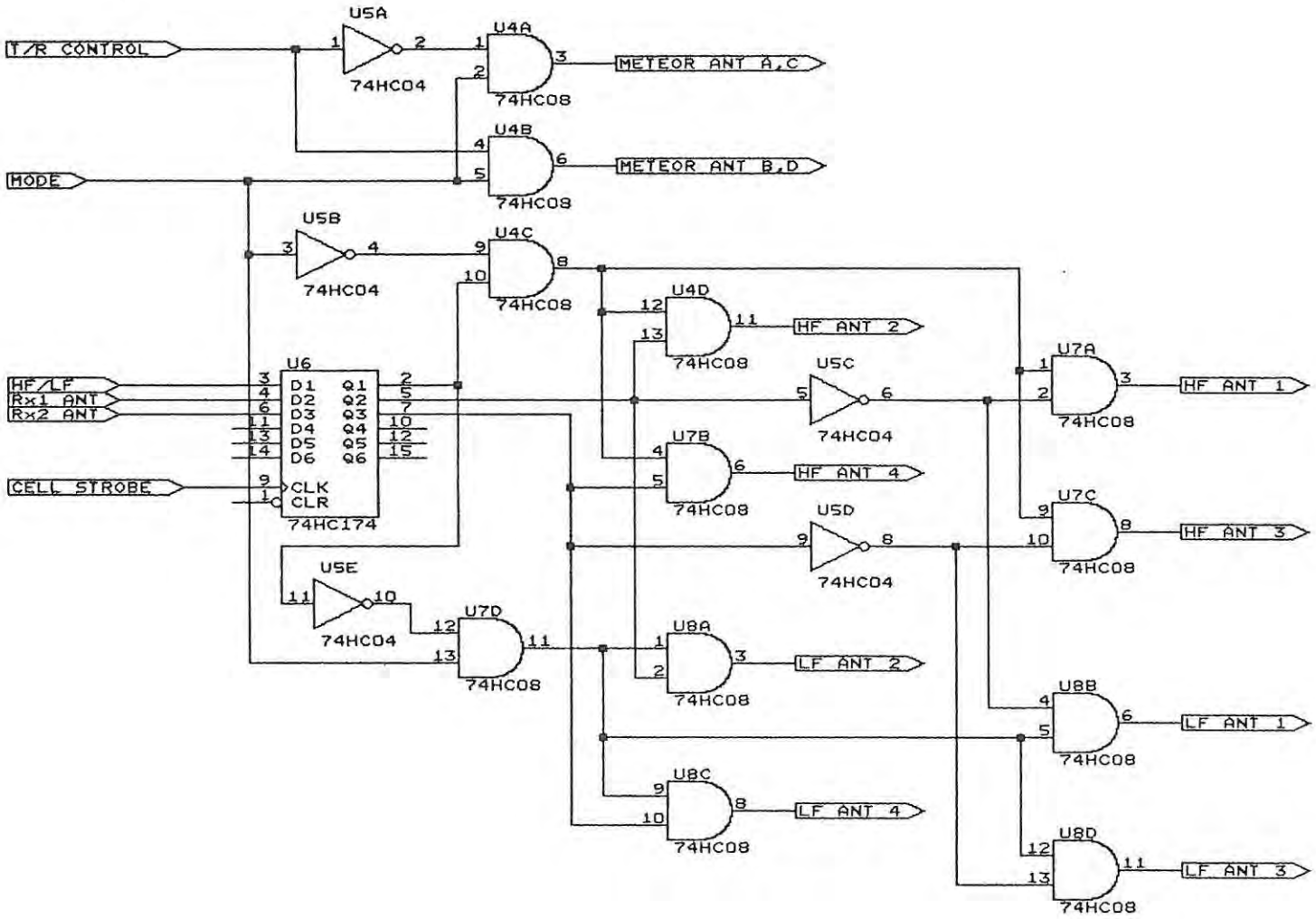


U3 : 8 = GND, 16 = Vcc
U6, U7 : 10 = GND, 20 = Vcc
U1, U2, U4, U5 : 7 = GND, 14 = Vcc

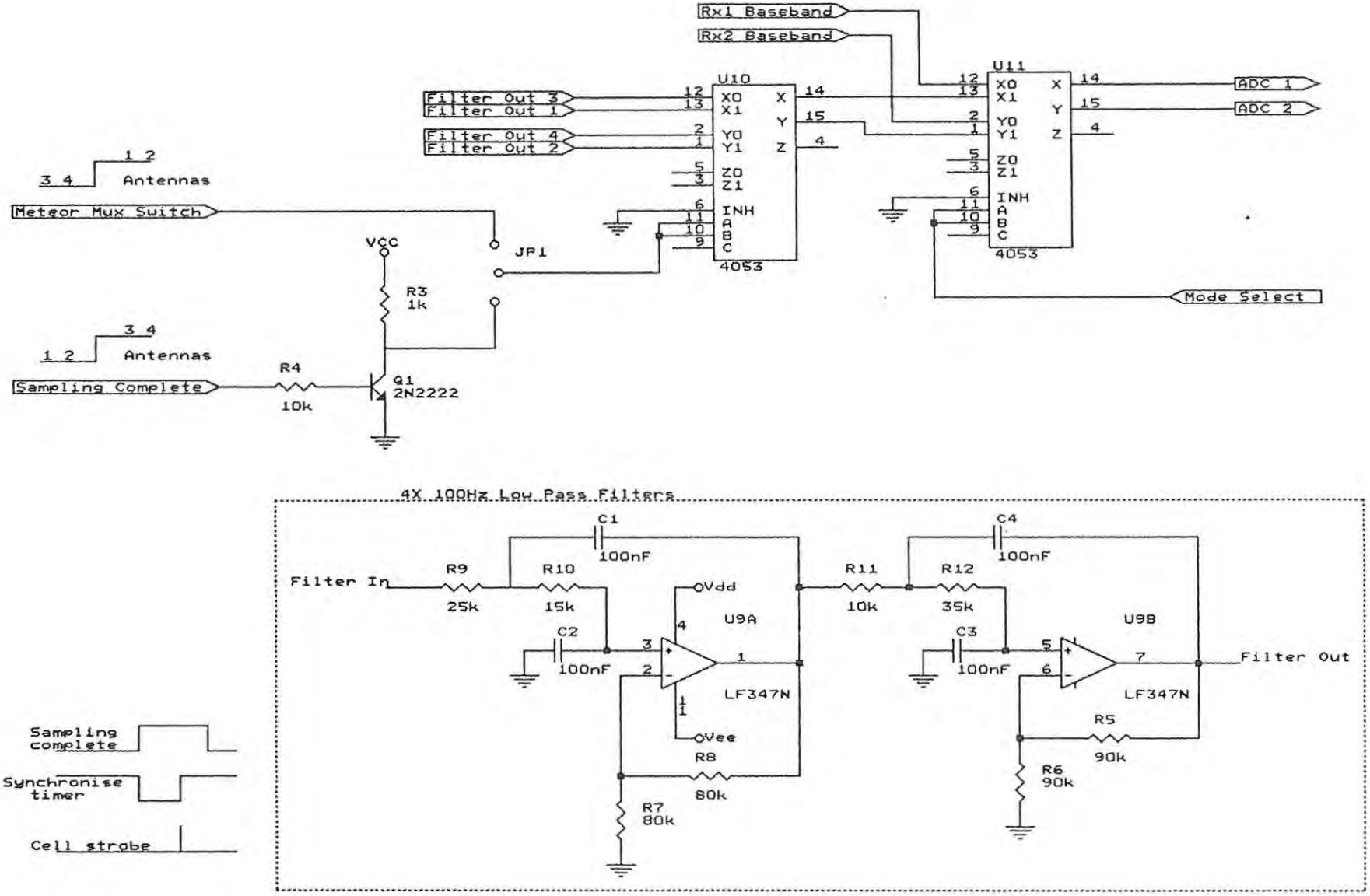
Eagle Electric		
Title: PC-35 Prototyping board for the IBM PC		
Size: E	Document Number: PC-35.SCH	REV:
Date: August 13, 1990	Sheet 4 of 4	



Rhodes University		
Title: SIGNAL SWITCHING		
Size: C	Document Number: SS.SCH	REV: 1
Date: December 14, 1990	Sheet 1 of 3	



Rhodes University		
CIRCUIT DIAGRAM INCOMPLETE		
Title: RF CONTROL LOGIC		
Size: A	Document Number: RF.SCH	REV: 1
Date: December 14, 1990	Sheet: 2 of 3	



4053 : 7 = Vee, 8 = Vss, 16 = Vdd

Vee = -12V, Vss = 0V, Vdd = 12V

Rhodes University		
CIRCUIT DIAGRAM INCOMPLETE		
Title: BASEBAND SIGNAL SWITCHING		
Size: A	Document Number: BASEBAND.SCH	REV: 1
Date: December 14, 1990	Sheet 3 of 3	

Appendix M

Parts list

Item	Quantity	Reference	Part
1	1	U1	HCPL2602
2	7	C6,C1,C2,C3,C4,C7,C8	.1uF
3	1	C5	10uF
4	1	U36	74HC373
5	6	U37,U9,U26,U29,U45,U51	74HC08
6	7	U35,U5,U22,U25,U34,U50, U55	74HC32
7	5	U33,U13,U21,U30,U41	74HC74
8	2	J1,J2	BNC
9	1	U49	ICM7170
10	3	U31,U17,U53	74HC04
11	2	U28,U23	74HC139
12	4	R7,R4,R8,R9	10k
13	1	J3	CONNECTOR DB9
14	2	U47,U46	74HC126
15	1	JP5	HEADER 7X2
16	4	JP4,JP1,JP2,JP3	JUMPER-2
17	3	D3,D1,D2	1N4148
18	1	Q1	2N2222
19	1	R10	680
20	1	BT	4.5V
21	1	D4	1N4007
22	1	U7	74HC4046
23	1	R2	2k7
24	1	R3	330k
25	1	R1	2k2
26	1	C1	100pF

Item	Quantity	Reference	Part
27	1	C2	10nF
28	8	U2,U3,U4,U8,U11,U12,U14, U24	74HC4017
29	5	U10,U19,U20,U52,U54	74HC4040
30	2	U16,U15	4017
31	4	U44,U39,U40,U43	74HC191
32	2	U42,U38	4075
33	1	R5	10M
34	1	R6	100k
35	1	X1	32768Hz
36	2	C3,C4	20pF
37	1	U32	74HC273
38	1	U27	74HC174
39	1	C5	47uF
40	1	IC1	74LS04
41	1	IC3	74LS138
42	1	IC2	74LS30
43	1	IC4	74LS02
44	1	IC5	74LS08
45	1	IC6	74LS244
46	1	IC7	74LS245

Item	Quantity	Reference	Part
1	1	U1	LM7805
2	1	F1	0,5A
3	1	U2	LM7912
4	1	U3	LM7812
5	12	J1,J2,J4,J5,J6,J7,J8,J9, J10,J11,J12,J13	BNC
6	1	J13	220V 50Hz
7	2	R1,R3	1k
8	3	C1,C2,C3	47uF
9	2	D2,D1	BRIDGE
10	3	C5,C4,C6	0.1uF
11	1	D3	Green
12	2	J3,J1	CONNECTOR DB-9
13	1	J2	CONNECTOR DB-25
14	1	T1	TRANSFORMER CT
15	1	K1	SOLIDSTATE RELAY
16	3	U4,U7,U8	74HC08
17	1	U5	74HC04
18	1	U6	74HC174
19	2	R4,R11	10k
20	1	Q1	2N2222
21	2	U10,U11	4053
22	1	U9	LF347N
23	4	C1,C2,C3,C4	100nF
24	2	R5,R6	90k
25	2	R7,R8	80k
26	1	R9	25k

Appendix N

System installation

N.1 PC addressing and interrupts

Device	Memory (hex)	Port addresses (hex)
DSP-16	D0000 – DFFFF	33C – 33F
TC		300 – 31F

Table N-1: Memory and port addresses required by system hardware

Interrupt source	IRQ
Timer (not disabled)	0
Keyboard	1
Real Time Clock	8
ADC card(s)	9
Timing Controller	3
Serial port (com1)	4
DSP card	5

Table N-2: Allocation of Interrupt Request (IRQ) lines

If the DSP card in question uses on-board ADC's then it will utilise IRQ 5, otherwise it will be replaced by IRQ 9 from the ADC card(s). If the Timing Controller is not located then IRQ 3 will be replaced by IRQ 8 from the PC's Real Time Clock and *Meteor* will be the only type of operation that can execute.

N.2 System interconnections

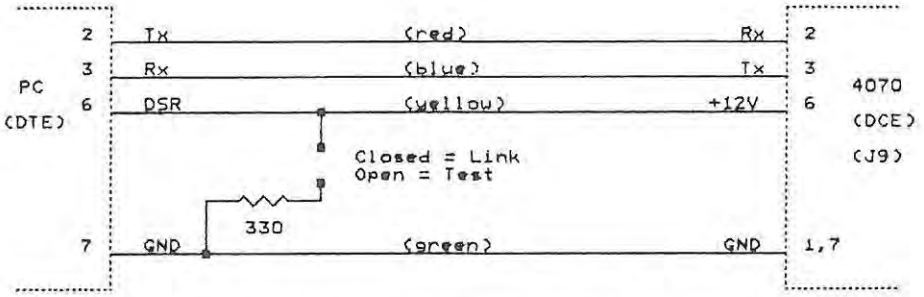


Figure N-1: Wiring the RS-232 link

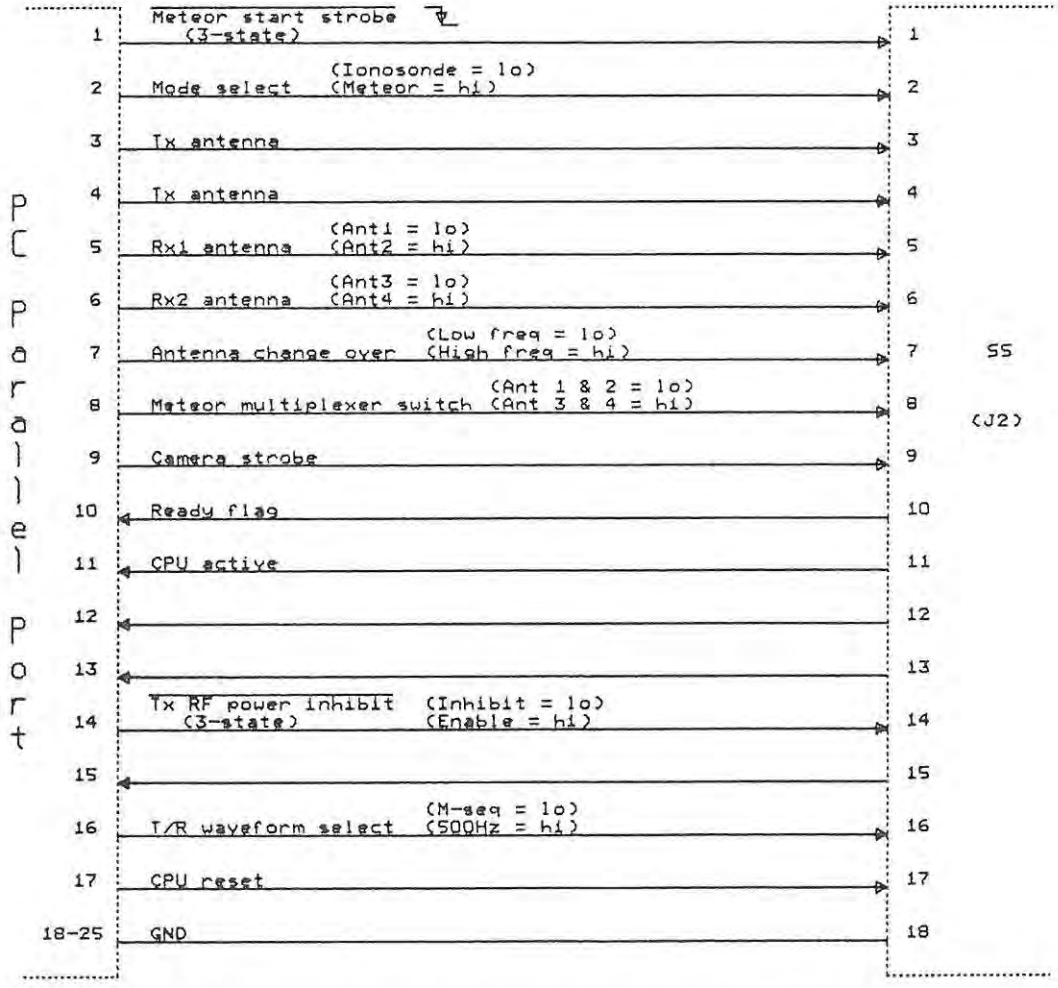


Figure N-2: Connecting the PC's parallel port to the Signal Switching unit

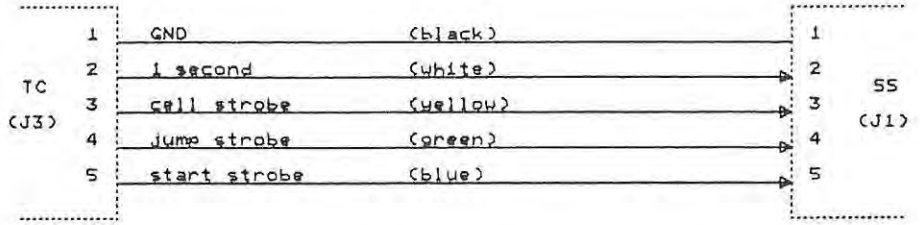


Figure N-3: Connecting the Timing Controller to the Signal Switching unit

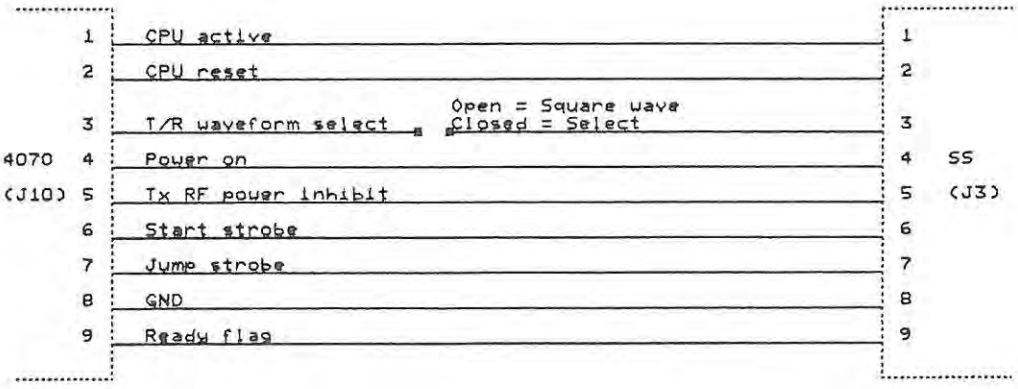


Figure N-4: Connecting the BR-9034 to the Signal Switching unit

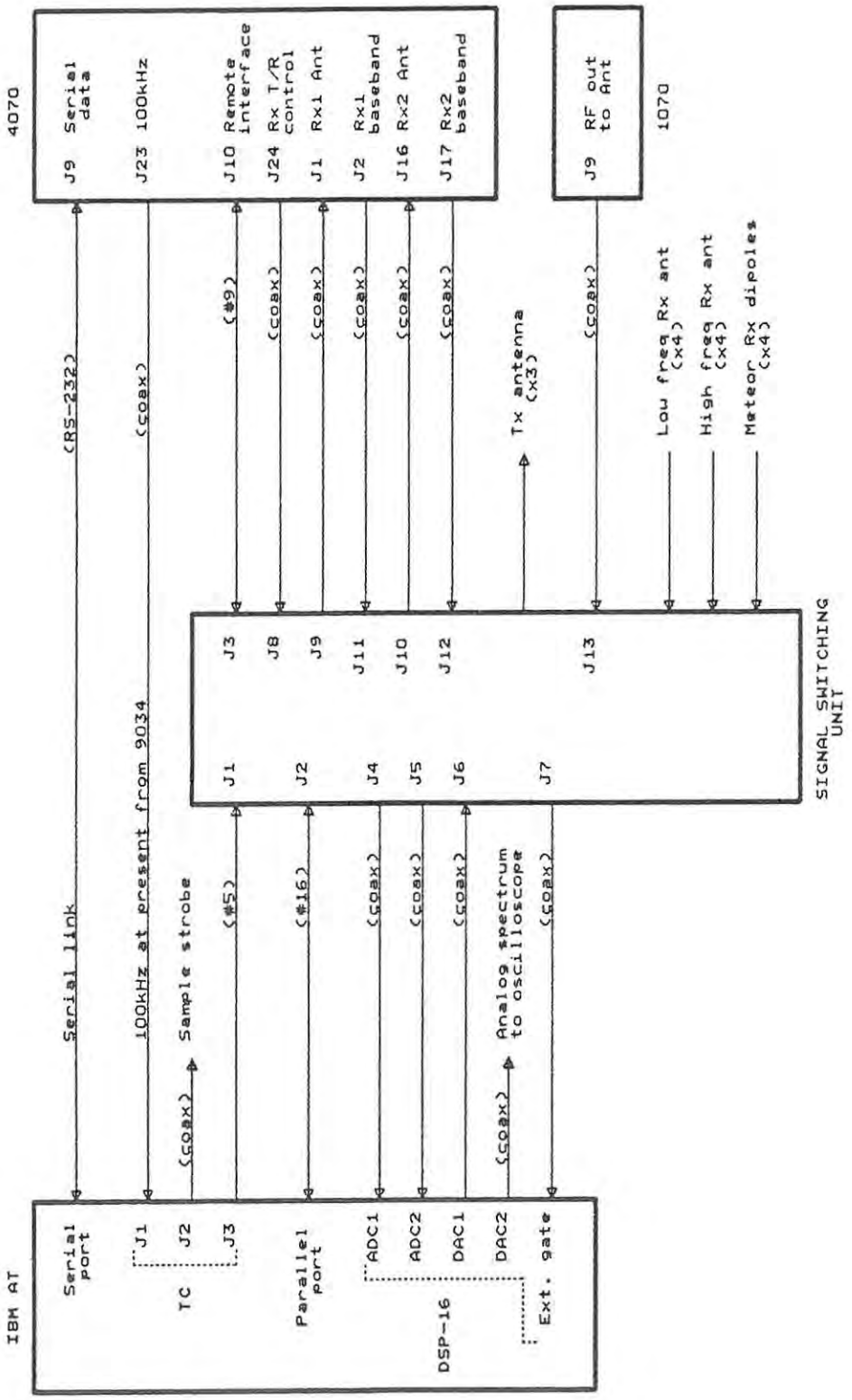


Figure N-5: System wiring diagram

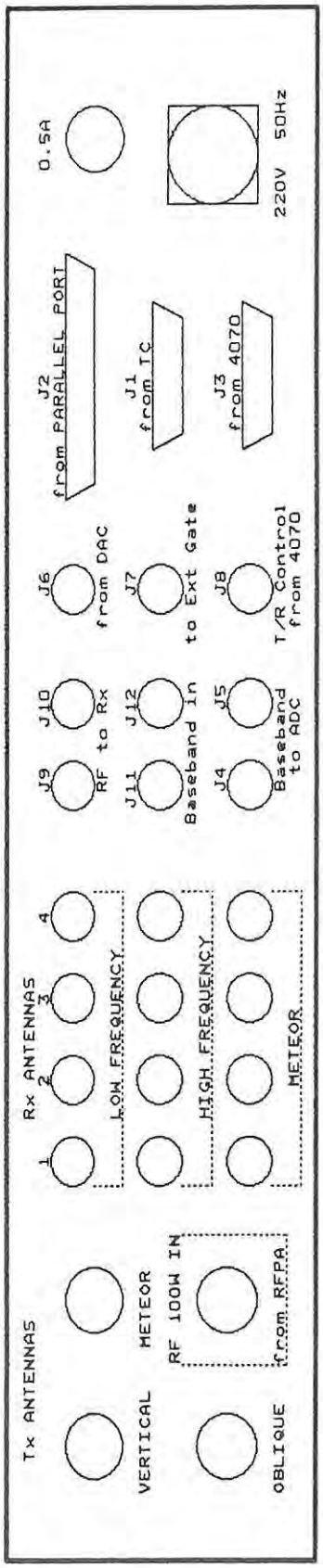


Figure N-6: The Signal Switching unit rear panel

Operators Guide

Contents

1 File management

1.1 Files used by the control program

The following files are required for operation of the system. They must all be located in the same directory, but not necessarily the root directory.

CONTROL.EXE is the executable version of the Turbo Pascal program which runs on the PC and controls all sounder and data capture operations.

CONTROL.CFG is read by the control program during initialisation and is used to set such system parameters as station location and boot operation. If the configuration file cannot be found the operator will be prompted to create one or alternatively abort execution.

CONTROL.SYS is also loaded into memory during initialisation of the control program and contains all the previously defined system operations. If the system file cannot be found, the operator will be given the option to create the default system file, or alternatively abort the program.

DSP16TMS.HEX is the TMS320 object code down-loaded to the DSP16 card on initialisation of the control program.

*.TBL files contain the look-up table for the DSP16's digital filter used in meteor mode. They contain 82 positive integer values with 4096 equating to a weighting factor of 1. Very little error checking is performed on these files and the onus is on the operator to ensure they are correctly formatted. The first entry in the file is the digital filter reference number (0–255) stored in the meteor parameter record. The second value corresponds to the 20 Hz weighting factor and the last to 80 Hz. Comments must be preceded by a semicolon.

CGA.BGI is the graphics driver for a CGA monitor.

1.2 Files created for data storage

(ppp)(yy)(ddd).MET is the format of the filename used to store the meteor data, where (ppp) represents the first three characters of the station name, (yy) the year, and (ddd) the day number. A new file is created everyday for meteor data.

(ppp)(yy)(ddd).DAT store the analysed meteor data for each day giving meteor rates, frequency histograms and the average power spectrum.

(pp)(yy)(hh)(mm).(ddd) is the filename format for ionograms, where (hh) represents hours and (mm) minutes

All of the above output files can be directed to separate sub-directories from within the configuration file.

2 Definition of system operations

The following types of system operations exist: *Sounding, Ionogram, Doppler, Meteor, Timing* and *Schedule.* Any operation created will be one of these types and will be specified by a unique operation name. A *Schedule* simply programs other operations to run at specific times (software operation) whereas all the other operations exercise some form of control over the system hardware configuration and/or timing, and are termed *hardware operations.*

Note: A *Sounding* is somewhat of an exception to the general rules that apply to system operations as will be explained in section 4.6.

Each operation is identified by a unique operation name of not more than 8 characters in length, but commencing with an alpha character. Changes to the system file will only be updated on hard-disk when a hardware operation is not executing.

3 Screen utilisation

The screen is divided into a number of windows although no window boundaries are visibly displayed. The general screen layout is shown in Figure 1.

3.1 Clock/calendar window

The top right hand corner shows the time and date. The display format is as follows:

DayOfWeek Century+Year DayNo Hour:Min:Sec

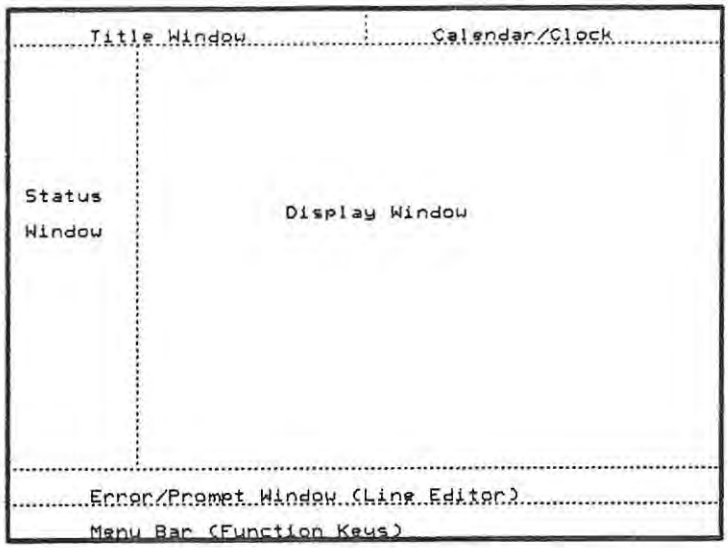


Figure 1: Screen window layout

3.2 Title window

The title window displays the station name at the outer level of the menu structure. As you descend this hierarchial structure it is replaced by the name and path of the current level.

level 0 : station name

level 1 : level 1 title

level 2 : level 1 title — level 2 title

3.3 Menu window

The bottom of the screen is used for the menu bar, which gives the current definition of valid function keys. The user interface is largely function key driven.

3.4 Error/prompt window

Immediately above the menu window is the error/prompt window. When an error message is displayed all function keys etc. are inactive until the message is cleared by pressing any key. Error messages will also time out after 10 seconds. The prompt window is the only window that can be physically edited, through the use of a simple line editor.

3.5 Display window

The display window, which is the default window, is used for the plotting of ionograms and displaying meteor echo data when at the outer level of the menu structure. At lower levels it is used for the display of such things as the configuration file and system operations. The display window is only cleared when necessary, so in many instances the previously viewed data will remain displayed even when you change levels, unless there is new information to display.

The following data pertaining to meteor echoes is displayed:

```
Time of last meteor echo

Frequency of maximum amplitudes for antennas
      A      B      C      D

Average amplitude                Average noise

Absolute sample maximums for antennas
      A      B      C      D



Time of last echo removed by the PC's software filter

Average frequency of maximum amplitudes

Average amplitude
```

For ionograms and doppler soundings, height is plotted on the vertical axis and time on the horizontal. Height markers are plotted at the start of a sweep. If the display is cleared during a run, by for instance entering the editor, then these will be lost as will the existing data plotted. Frequency or time marks may be plotted during a sweep. As time

advances, data is plotted from left to right. If the right hand edge of the display window is reached, then plotting will continue from the left hand edge and will overwrite existing data.

Note: Screen output has the lowest priority, and if there is insufficient time, screen output may be lost.

3.6 Status window

The status window is used to indicate the current status of the system and any fault conditions that are present or may have occured. The status screen layout is as follows:

```
Active schedule
Active hardware operation
DSP errors
TC errors
Sounder errors
Timing slip
Sounder frequency
Status report
```

Active schedule – This is the currently active schedule, which is scanned once a second when no hardware operation is active. The active schedule is scanned even while it is being edited, so operations may be entered immediately before they are scheduled to run. If the operator is busy with disk access when a hardware operation is scheduled to start running, then the operation will be cancelled to avoid any possible timing errors that may occur, but the speaker will beep to notify the operator of this.

Active hardware operation – Displays the name and type of hardware operation that is currently active.

The following error conditions are reported:

- DSP – Signal processing card not present or could not be initialised.
  Filter – The digital filter look-up table required in meteor mode could not be downloaded.
  DspError – An error occurred during communications with the Dsp card.

Format – There was a format error in the data transferred to the PC, possibly due to insufficient time, in which case the DSP card would reset itself.

- TC – Timing controller card not present. All operations requiring the TC will be inactive. This implies all hardware operations other than *Meteor*.
  Xtal – Input from the external frequency standard has been lost and the timing controller is using its on-board oscillator.
  FreqFail – Input from the frequency standard was lost at some stage, but has since been re-established.

- Sounder – Serial link not connected.
  RFPA – Transmitted power dropped below 15 W.
  Lock – Sounder frequency synthesizer broke lock.

Status report – The status report received from the sounder is displayed here. This report during operation looks as follows:

(XXX)g(YY)[Z]

XXX is the averaged IF amplitude of the receiver during the cell (0–255)

YY the receiver attenuation where 1=3 dB's, 2=6 dB's, etc. (0–38)

Z is not always present and represents one or more error conditions.

P – RF power out less than 15 W

L – Synthesizer out of lock

Other status reports are:

T – Sounder in standby mode

E – Input command error

4 Description of function keys

The user interface is menu driven and options are selectable via the use of function keys. The menu bar at the bottom of the screen gives the definition of each function key. This bar may change depending on the operators level within the menu hierarchy. Figure 2 gives an overview of this hierarchy with a brief description of the function of each key within the menu. A detailed description now follows.

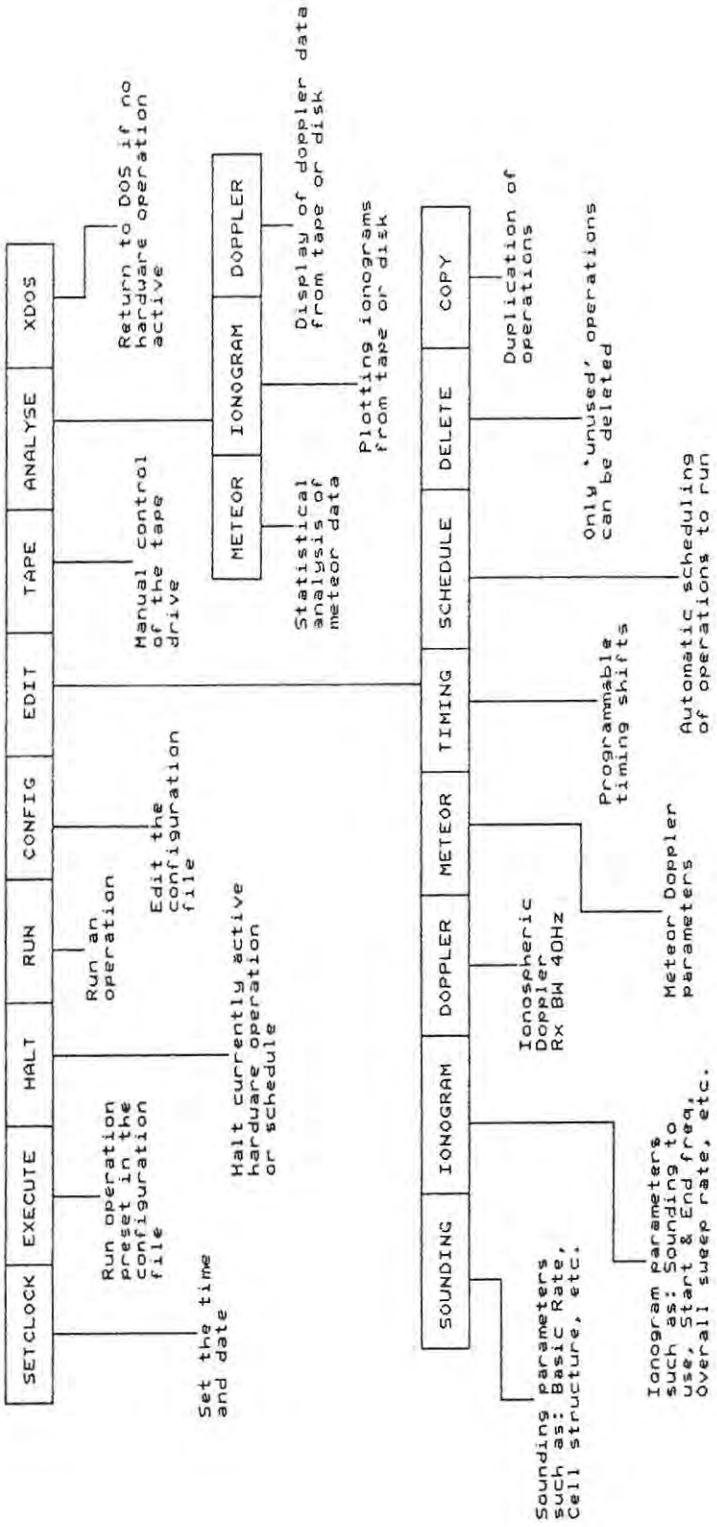


Figure 2: Software menu structure

4.1 SetClk — F1

Sets the real time clock on the timing controller card, or PC if this card is not present. The command format is as follows:

[WeekDay Year Month Day] [Hour] minute second

DayOfWeek is Mon, Tues .. Sun. Year can be either the century and year or just the year, and hour is in the 24 hour format. Parameters in square brackets are optional, but if one of these parameters is entered, then the remainder of the parameters in that set of parenthesis must also be given, and in the order specified.

4.2 Execute — F2

This function immediately executes the default operation specified in the configure environment (see section 4.5). If this is a hardware operation (any operation other than a Schedule) then this key is ineffectual while another hardware operation is active.

4.3 Halt — F3

Immediately halts the active hardware operation unless this is a *Timing* shift. Once a timing shift has been initiated it cannot be stopped due to a hardware constraint of the timing controller. If a hardware operation is not running then the currently active *Schedule* will be cancelled.

4.4 Run — F4

Prompts the operator for an operation to execute. If the entered operation is anything other than a *Schedule*, then it will be ignored if a hardware operation is active. A list of executable operations is displayed.

4.5 Configure — F5

In the configure environment, system parameters can be set which will be saved to disk in the CONTROL.CFG file.

Station Name : Consists of 3 to 20 characters with no spaces allowed between the first 3 characters as these are used for station identification in filenames.

EXECute Operation : The default operation to be run in conjunction with the Execute function key (F2).

Boot Operation : This operation will be executed immediately after the system boots up. The longest time for a fixed duration operation is 60 minutes. This is to limit the amount of data that can be lost after power failures, by having to wait a long time before the next scheduled operation.

Day Schedule, Night Schedule : If specified then the PC will run the appropriate schedule on boot-up.

Dawn, Dusk : Used to specify the start and end of day and night.

Meteor Directory, Statistical Directory, Ionogram Directory : Output data is directed to the appropriate sub-directory.

4.6 Editor — F6

The editor allows for the creation and deletion, as well as editing of all system operations. The menu bar is redefined (Figure 2) and the display screen lists all the currently defined system operations. Preceding every operation name is a number, and this value represents the operations *user-level.* It is a counter associated with each operation, initialised to zero, and incremented by one every time that operation is used by another operation. The need for this will be seen when the delete function key (F7) is described later.

Operations to be edited are selected with the aid of function keys and options are changed using the arrow keys. When input is required the operator will be prompted for this. Only the prompt window can physically be edited, and all input will be validated before updating the display window. In the line editor `Backspace` deletes the last character while `Delete` clears the input line. The `Esc` key always returns you to the next highest logical level in the command structure.

There are two 'global' editing features, delete (F7) and copy (F8), which operate on whole system operations while function keys F1 to F6 enable editing of the operations themselves. Once the type of system operation has been selected you will be prompted for the unique name of that particular operation. If this operation does not exist it will be created, provided there is sufficient space in the system file, with all parameters set

to the default values specified by the standard operations (Sstd, Istd, Dstd, Mstd, Tstd, Pstd). A description of each type of system operation is given below, followed by a list of its parameters.

Sounding — F1

A *Sounding* is the basic building block of an ionogram and as such is not a true system operation in that it can not be run by itself but only from within an *Ionogram*. A *Sounding* consists of a number of cells, and it is this ability to construct segmented sweep patterns that provides the chirp sounder with its advanced capabilities.

Basic Rate : The linear sweep rate used during each 'chirp' segment or cell. Selected from 25, 50, 100, 200 kHz/s.

Window Offset : The transmitter offset from the receiver tuned frequency, specified as a window offset height of between 0–500 km.

Rx Attenuation : Fixed attenuation programmable between 0 and 114 dB's in 3 dB steps, otherwise software AGC operational.

Cell Length : 0.25, 0.5, 1, 2 seconds.

Note: The maximum transform size is 1024 points. Therefore, if a cell length of 2 seconds is selected the highest sampling rate possible is 512 Hz. Inversely if the sampling rate is 2048 Hz the cell length cannot exceed 0.5 seconds.

Sample Rate : 256, 512, 1024, 2048 Hz. In ionosonde mode the bandwidth of the receivers is 500 Hz. This implies that sample rates below 1024 Hz require additional band limiting of the sampled signal to avoid aliasing problems.

Points / Cell : Number of points per cell on the power spectrum to be sorted in descending order of magnitude and plotted to the screen. Selectable between 1–99.

Threshold : Magnitude squared threshold below which points on the power spectrum are rejected, irrespective of the number of points per cell (0–99).

No. of Cells : Number of cells selectable between 1 and 10. Cells within a sounding will be executed in the order specified.

(Rx1 Antenna) (Rx2 Antenna) (Local Offset) : The local offset is the positive offset of the cell start frequency from the sounding start frequency (0 to 9.99 kHz in 10 Hz steps). Each receiver must be switched to one of the antennas within its antenna pair.

Ionogram — F2

This specifies the 'macro' type parameters of the sweep as well as re-direction of the output data for storage purposes.

Mode :

Tx – Transmit only.

Rx – Receive only.

T/R – Transmit-receive switching using an M-sequence.

Tx+Rx – Simultaneous transmit and receive.

Sounding : Name of the sounding to be used, from a list of available soundings.

Start Freq : Start frequency in kHz, in the range 1.6 to 30 MHz. If set greater than the end frequency, then the end frequency will be adjusted to equal the start frequency.

Data Storage :

None – Data only displayed on screen

Disk – Store data to hard disk

Tape – Data written to magnetic tape

Film Record : Generation of film record.

Ionogram Type :

Normal – Non-zero overall sweep rate.

Stationary – Frequency reset to start frequency at the beginning of every sounding.

Normal ionogram

End Freq : End frequency in kHz, in the range Start Freq to 30 MHz.

Overall Rate :

Linear : 25–200 kHz/s

Logarithmic : 10–30 mOct/s (milli-Octaves)

Stationary ionogram

Duration[1] : Length of run in minutes (1–60). When set to continuous the operation can only be halted by operator intervention.

Time Mark : When set will produce a time mark on the display (and on film) after time intervals of between 1–60 minutes.

Doppler — F3

A *Doppler* operation is a fixed frequency sounding where the transmitter frequency is generally offset from the receiver tuned frequency. Only the doppler shift of the signal, reflected from an ionospheric layer, is measured. The receiver bandwidth is set to 40 Hz and a sample rate of 256 Hz is used.

Mode :

Tx – Transmit only.

Rx – Receive only.

T/R – Transmit receive switching using an M-sequence.

Tx+Rx – Simultaneous transmit and receive.

Rx Frequency : Receiver tuned frequency, variable between 1.6 and 30 MHz in 1 kHz steps.

Tx Offset Freq : Transmitter offset frequency, variable between 0 and 40 Hz in 1 Hz steps (negative offset – LSB receivers)

Rx Attenuation : Fixed attenuation programmable between 0 and 114 dB's in 3 dB steps, otherwise software AGC operational.

[1] Operations specified with a time duration are shortened by a couple of seconds to allow for hardware setup time.

Duration : Length of run in minutes (1–60) less a couple of seconds to allow for hardware setup time. When in continuous mode the operation can only be terminated by the operator.

Time Mark : If set will produce a time mark on the display (and on film) after time intervals of between 1 and 60 minutes.

Cell Length :

1 second – 256 samples per channel

2 – 512

3 – 768

4 – 1024

Points / Cell : Number of points per cell on the power spectrum to be sorted in descending order of magnitude and plotted to the screen. Selectable between 1–99.

Threshold : Magnitude squared threshold below which points on the power spectrum are rejected, irrespective of the number of points per cell (0–99).

Data Storage :

None – Data only displayed on screen

Disk – Store data to hard disk

Tape – Data written to magnetic tape

Film Record : Generation of film record.

Rx Antennas (Rx1) (Rx2) : Select receive antennas.

Meteor — F4

In *Meteor* operation a fixed frequency, well above the critical frequency of the ionosphere, is used to detect the doppler shift from meteors. The transmitter frequency is offset from the receiver tuned frequency. There is no explicit cell structure associated with a meteor operation. Angle of arrival of the returned signal is also measured by using four antennas simultaneously. To reduce crosstalk introduced in the synthesis of the two additional channels, the receiver bandwidth is set to 2500 Hz and the receiver baseband output is

band-limited to 100 Hz. Each channel is sampled at 512 Hz but channel pairs are sampled 180° out of phase.

Mode :

Tx – Transmit only.

Rx – Receive only.

T/R – Transmit receive switching using a 500 Hz square wave.

Tx+Rx – Simultaneous transmit and receive.

Rx Frequency : Receiver tuned frequency, variable between 20 and 30 MHz in 1 kHz steps.

Tx Offset Freq : Transmitter offset frequency, variable between 20 and 60 Hz in 1 Hz steps (negative offset – LSB receivers)

Rx Attenuation : Fixed attenuation programmable between 0 and 114 dB's in 3 dB steps.

Duration : Length of run in minutes (1–60). When in continuous mode the operation can only be terminated by the operator.

Data to Disk : Storage of data on hard disk.

Echo Criterion : The maximum frequency difference allowed between spectral maxima for recognition as a valid meteor echo (1 to 9 Hz)

Sprd Threshold : The threshold used to determine the spread of the signal in the frequency domain (10–999, units undefined).

Cell Length :

.3 seconds – 153 samples per channel

.4 – 204

.5 – 256

.6 – 307

.7 – 358

.8 – 409

.9 – 460

1 - 512

(4 channels with a sample rate of 512 Hz/channel)

Filter File : The name of the file containing the digital filter look-up table. If this file cannot be found the existing look-up table will be used. The file name must start with an alpha character.

No. Filters : Software filtering of echo data is possible, using a simple rectangular window function and the averaged signal amplitude for the four channels. The number of filters is selectable between 0 and 3.

Filter (Start frequency) (End frequency) (Threshold) : Inclusive of start and end frequencies. Frequency range is 20 to 60 Hz. (Threshold units are undefined.)

Timing — F5

Programmable timing shifts can be made using this operation. The timing adjustment will only be made if the timing controller is being clocked by the external frequency standard and not the onboard oscillator.

Direction : Advance or retard system timing.

Shift : Shift system timing in the specified direction by an amount between 0.1 and 999.9 ms in 0.1 ms steps.

Schedule — F6

A *Schedule* allows for the programming of operations to run at specific times. These operations can be run on an hourly, daily or weekly basis. The active Schedule only executes hardware operations while in record mode (ie. whenever the operator is not accessing one of the mass data storage devices). The command format is as follows:

[Day] [Hour] Minute Second Operation

Day : Sun, Mon, Tues, Wed, Thurs, Fri, Sat. Specification of Day is optional.

These operations will have the highest priority and will only execute on a weekly basis.

Hour : 0 to 23. Specification of this parameter is optional.

These operations have the next highest priority and will execute on a daily basis.

Minute : 0 to 59.

If no Hour is set then these operations will execute on an hourly basis.

Second : 0 to 59.

Operation : Name of operation to be run from list of system operations. A *Sounding* cannot be entered as it is in effect only a building block for an *Ionogram.*

Schedules are sorted and displayed vertically, firstly in descending priority, and secondly in chronological order. When searching an active schedule for an operation to run, it is scanned from top to bottom until an appropriate operation is found. If an entry is made with the same time information as an existing entry then the new one will be inserted above the old, and will consequently be the one executed.

While a hardware operation is running, the active Schedule is not scanned as a new operation cannot commence until the current one has terminated. Note that a Schedule can call another Schedule, which allows for the development of large and complex programmes if required.

Delete — F7

Whenever an operation is used or called by another operation, the first operation has its *user-level* adjusted. A user-level greater than zero indicates that the operation is currently used bv at least one other operation. For instance every entry of an operation in a *Schedule* increments the operations user-level by one and similarly when a *Sounding* is used in an *Ionogram.* The user-level of each operation is displayed in the outer layer of the editor environment. The default operations cannot be deleted, nor any operations with a user-level greater than zero. An active schedule can also not be deleted.This function is only available from the outer layer of the editor and you will be prompted for the name of the operation to delete.

Note: There is no way to retrieve an operation that has been accidentally deleted!

Copy — F8

Provided there is sufficient space in the system file, any operation can be copied. The command format is as follows:

(Old operation name) (New operation name)

This function is only available from the outer layer of the editor.

4.7 Tape — F7

Allows operator control of the magnetic tape unit (not operational).

4.8 Analyse — F9

Post processing and analysis of data. At the time of writing only meteor data could be analysed. A frequency histogram, daily rate, hourly rate, and averaged power spectrum are calculated for a specified days data. The results are plotted on the monitor and written to a data file.

4.9 XDOS — F10

Terminates CONTROL.EXE and returns to DOS. Only functional if no hardware operation is active.