AUSTRALIAN NATIONAL ANTARCTIC RESEARCH EXPEDITIONS

ANARE RESEARCH NOTES
60

An ionospheric Doppler and propagation delay monitor as operated at Macquarie Island 1986-87

Sjoerd Jongens, E.A. Essex and G.B. Burns

ANTARCTIC DIVISION
DEPARTMENT OF THE ARTS, SPORT,
THE ENVIRONMENT, TOURISM AND TERRITORIES

ANARE RESEARCH NOTES (ISSN 0729-6533)

This series allows rapid publication in a wide range of disciplines. Copies of this and other *ANARE Research Notes* are available from the Antarctic Division. Any person who has participated in Australian National Antarctic Research Expeditions is invited to publish through this series. Before submitting manuscripts authors should obtain a style guide from:

The Publications Office
Antarctic Division
Channel Highway
Kingston
Tasmania 7050
Australia

Published August 1988
ISBN: 0 642 13260 7

CONTENTS

APPENDIXES

FIGURES

AN IONOSPHERIC DOPPLER AND PROPAGATION DELAY MONITOR AS OPERATED AT MACQUARIE ISLAND 1986-87

by

Sjoerd Jongens(1), E.A. Essex(2) and G.B. Burns(1)

(1)Antarctic Division
Department of the Arts, Sport, the Environment, Tourism and Territories
Kingston, Tasmania, Australia

(2)Division of Theoretical and Space Physics
Latrobe University
Bundoora, Victoria, Australia

ABSTRACT

A monitoring system was developed to study the Doppler frequency shifts and time-of-arrival variations of VNG's carrier frequencies, and the propagation delay of their modulation. This report describes the equipment development, how the data were collected and presents some of the data in a more accessible form.

The 'Standard Frequency and Time Signal Service' VNG is located at Lyndhurst, Victoria, Australia (38°3.3' S, 145°15.7' E). This study is a continuation of similar monitoring of VNG by Cornelius and Essex (1978) on a much shorter path within Victoria, Australia. The present report describes the equipment used in creating a monitoring station for Macquarie Island at 54°30'S, 158°57'E. The monitoring station was constructed with minimal cost and development time. The system developed used slightly modified commercial equipment, and is easy to duplicate. Dedicated minicomputers were employed to automate the parameter recording and to control the HF receivers.

The VNG timing signals are transmitted on 4.5 MHz (0945-2130 UT), 7.5 MHz (2245-2230 UT) and 12.0 MHz (2145-0930 UT) each day. Only one frequency was monitored at any time.

This experiment was operated and developed between 19 August 1986 and 26 January 1987. The data collected include the Doppler frequency shift and HF signal strength recorded on chart, printed and digitally recorded sixty sample averages and standard deviation values, and some frequency modulated audio tapes. A few weeks of 'time-of-arrival' data were also recorded on chart.

The digitally recorded sixty sample averages of the Doppler frequency shift values are plotted in stacked diurnal form. Frequency spectra of the audio tape recorded Doppler frequency values are also presented in sonagram form.

It is demonstrated that group delay can be more accurately determined by software signal analysis, rather than by the use of a hardware level-discriminator.

[illegible]PHERIC DOPPLER AND PROPAGATION DELAY [illegible]
AS OPERATED AT MACQUARIE ISLAND 1985

by

Sjoerd Jongsma[1], [illegible] and G. B. Burns[2]

[1]Antarctic Division
[illegible]ment of the Arts, Sport, the Environment, Tourism and Ter[illegible]
Kingston, Tasmania, Australia

[2]Division of Theoretical and Space Physics
La Trobe University
Bundoora, Victoria, Australia

ABSTRACT

[illegible] system was developed to study the Doppler frequency shifts and [illegible] (VNG) carrier frequencies, and the propagation delay of their [illegible] the equipment development [illegible] the data were collected [illegible] more accessible form.

The [illegible] Frequency and Time Signal Service, VNG, is located at Lyndhurst [illegible] (38° [illegible] S, 145° [illegible] E). This study is a continuation of similar work [illegible] by [illegible] and Essex (1978) on a multi-station project in Victoria, Australia [illegible] the equipment used in creating a monitoring station for Macquarie [illegible] (1985). The monitoring station was constructed with [illegible]. The system developed [illegible] modified commercial [illegible]. Dedicated minicomputers were employed to automate [illegible] control the HF receivers.

The [illegible] signals are transmitted [illegible] (0945-2130 UT), 7.5 MHz [illegible] MHz (2245-0930 UT) [illegible] one frequency was [illegible]

The system was operated and developed between [illegible] August 1986 and [illegible] to provide the Doppler frequency shift and HF signal strength [illegible] digitally recorded [illegible] and standard [illegible] modulated audio tapes [illegible] of time-of-arrival [illegible]

[illegible] solid state sample averager [illegible] Doppler frequency shift [illegible] Frequency spectra [illegible] audio tape recorded [illegible] in conjugate [illegible]

[illegible] that group delay can [illegible] determined [illegible] analysis, [illegible] by the use of a [illegible]

1. INTRODUCTION

Doppler shifts cause signal degradation on high frequency (HF) radio circuits propagated via the ionosphere. This is particularly so on the more critical modulation modes such as data circuits or medium speed telegraph circuits. As the HF signal is refracted by the ionosphere, a Doppler shift may be introduced by the time-varying changes in the ionosphere, such as refractive index changes along the ray path or the physical motion of the ionosphere. These effects are more prevalent and more intense in the higher latitudes where the ionosphere shows rapid variations under the influence of auroral and magnetic activities and the presence of the ionospheric trough. Very little research into the ionosphere's behaviour has been done in the southern high latitude region, particularly in the Australian sector of the Antarctic and Southern Ocean region. As a contribution to the research of the near earth plasma and its effect on HF propagation, the Doppler monitoring and time-of-arrival HF receiver described here was developed. It can measure the received carrier frequency of any HF transmitter very accurately (limited mainly by the accuracy of the reference oscillator available on-site), and it also provides a 'clean' and useful detection of any time-signal modulation. This second facility enables the study of the variation in the propagation group delay, or the dispersion of the HF signal.

Macquarie Island is a good location for this project, as routine measurements of various parameters related to the ionospheric properties are made along the same path. Information on the path under study is given in Figure 1. Although a comparison with other data sets is not presented in this report, both at Beveridge in Victoria and at Macquarie Island geomagnetic pulsations are measured, and at Beveridge, Hobart and on Macquarie Island ionosondes operate routinely. A Faraday rotation experiment located on Macquarie Island (Lambert et al. 1986) measures the total electron content (TEC). This is maximally influenced at the sub-ionospheric point (SIP) of the ionosphere around 153°E, 48°S (approximately halfway between Macquarie Island and Hobart). If an appropriate frequency is selected for single hop propagation, the VNG monitoring equipment will indicate the effect of the ionospheric properties just north of this SIP, adding to the geographical chain of monitoring points.

Macquarie Island is conveniently close to VNG, while still enabling the monitoring of the effects of auroral, magnetic and ionospheric events, including the winter mid-latitude trough, by this method. Other Australian National Antarctic Research Expeditions (ANARE) stations will experience a much more complex signal over at least a two-hop path, with more auroral absorption and hence much lower signal-to-noise ratios.

This report deals with the hardware modifications required to commercially available equipment, the development of the software for the data collection, the presentation of some initial data reductions and some thoughts on possible future developments. It is written as a guide to those who may wish to use this equipment, develop similar equipment or proceed further with the analysis of the data collected.

A block diagram of the most developed experimental set up is shown in Figure 2.

2. DOPPLER FREQUENCY MEASUREMENTS

2.1 INTRODUCTION

The 'Standard Frequency and Time Signal Service', VNG Lyndhurst, Victoria transmissions on 4.5, 7.5 and 12 MHz are accurate to within 1 part in 10^{11} (expressed as a 24 hour average) of Telecom Australia's standard of frequency. The Telecom frequency standard is maintained to within a few parts in 10^{12} of the international definition of a time interval.

In order to obtain meaningful Doppler measurements, of the order of a few Hz for a 4.5 to 12 MHz carrier, the accuracy of the receiving equipment should be at least 1 part in 108. Previous studies (Cornelius 1976) have developed purpose built receivers to achieve this accuracy. This report indicates a method of modifying high-standard consumer-type receivers to give the high accuracy frequency measuring ability required.

This is achieved by phase-locking the receiver's local oscillators to the frequency standard of the laboratory clock used at the Macquarie Island station. The laboratory clock, a Systron-Donner model 8120 with a thermostatically controlled crystal oscillator (option B11), is accurate to 1 part in 109. A fine control adjustment of the crystal oscillator is used to set the long term accuracy compared with the VNG time signals to within 2 ms per week.

2.2 THE RECEIVERS

The first requirement of a Doppler monitor is a suitable short-wave receiver. A general purpose HF receiver keeps the equipment flexible.

As the Doppler shifts are small compared with the transmission frequencies, direct measurements would be hard to accomplish. The heterodyne principle overcomes this problem. The received frequency is mixed with a local reference, enhancing the resolution (Figure 3).

In our application, the first heterodyne in the system is designed into the HF receiver. After signal amplification the resulting frequency is then mixed again with the beat frequency oscillator (BFO), producing the Doppler frequency added to a 10 Hz offset (Figure 4).

Most modern receivers use the heterodyne principle and are designed with a phase locked loop (PLL) local oscillator, which synthesises the required receive frequency (Figure 5).

Heterodyne receivers are based on the principle that it is a lot easier to produce a variable local oscillator with a fixed frequency bandpass filtered amplifier, than a variable frequency bandpass filtered amplifier in a direct receiver. A direct receiver would need to maintain, for instance, a 3 kHz audio bandwidth anywhere within the 2 to 30 MHz range. A bandpass amplifier of that calibre is hard to produce. Introducing the heterodyne principle, where the received signal is mixed with a variable local oscillator. The intermediate frequency (IF) amplifier needed to bring the signal to a detectable level only operates around a fixed frequency of nominally 455 kHz.

In the early days, local oscillators were designed with an L-C type oscillator, where either the capacitor (C) or the coil (L) was made variable. Modern designs with higher accuracy use a crystal reference oscillator (Xtal), the frequency of which is nominally divided down to 10 kHz (refer Figure 5). The actual local oscillator is a voltage controlled oscillator (VCO) from which the frequency is also divided down to 10 kHz with a variable divider. That divider is fairly simple to build from digital ICs, and is often controlled by digital (push-button) circuitry. The two 10 kHz signals are then fed into a phase comparator (Φ in Figure

5), the output of which is low-pass filtered and is used to control the VCO. This comprises the PLL frequency synthesised local oscillator.

The most suitable and economical receiver found on the market at the end of 1985 was the ICOM model IC-R71A with the high stability crystal option CR-64 and the computer interface connector IC-EX309. The high stability crystal was advertised as being accurate to 1 part in 108, but no indication was given over what period it maintained this accuracy. As mentioned previously, in situ the reference oscillator was locked to the ANARE laboratory clock to ensure a sufficient margin between the available and required accuracy. The CR64 crystal oscillator is selected to produce a frequency of sufficient stability on its own account. This is necessary as the PLL circuitry is otherwise prone to introduce a certain amount of fast jitter, even when locked to the external reference.

A block diagram of R71A with the modifications is given in Figure 6. The IC-R71As are designed in a slightly more complex manner than the norm. First of all, there are four IF frequencies. This is done to improve the image rejection and the IF bandwidth control. At first glance, it would seem that four different local oscillators (LO) are used. The apparent need to lock all four LOs to the external reference could appear difficult. However, the first and second LOs are derived from the same reference oscillator (only LO1 is variable), and the third and fourth local oscillators have the same frequency. The IF3 of 455 kHz is produced to allow the IF bandwidth to be varied by front-panel control through a standard 455 kHz ceramic filter. The same frequency (LO3=LO4) is used to first mix down, then back up to the original IF2 frequency. Thus, any error introduced in the mixing with LO3 is eliminated when mixed back with LO4.

A noteworthy design feature of the PLL dividing circuit is that it offers full micro processor control. This means that it is possible to select the frequency to be monitored via the controlling external computer. Preparation for this was initiated but not completed in the available time.

Two receivers were in fact purchased. As will be described in detail later, one receiver was tuned to the main carrier of the transmitter station under study, and the second was used to synthesise a stable BFO, hence with this dual receiver system only one HF frequency can be monitored at any one time. The second receiver was also given the local oscillator frequency of the first, thus effectively tuning it to the same station. The second receiver was then operated in the AM-mode, detecting the one-second time marks, enabling time-of-arrival variations as well as Doppler frequency shifts to be monitored.

2.3 AERIALS

With both receivers operating in the CW mode, a comparison was made between two different receiving aerials. The aerial used under most conditions was a large VEE configuration with sides each about 150 m in length. It dramatically outperformed the alternative shorter, so called 'long wire' aerial of about 50 m on the two lower VNG frequencies of 4.5 and 7.5 MHz. On the 12 MHz frequency the signal strengths were very similar, averaged over time, although not simultaneously.

2.4 HARDWARE MODIFICATIONS

There are two main modifications required to make the ICOM IC-R71A receivers suitable for the measurements of the Doppler shifts. Firstly, the reference oscillators in both receivers must be locked to the laboratory clock. Secondly, in the second receiver LO1 must be mixed with LO2 and divided by 2, to obtain an accurate BFO to beat with the received carrier of VNG (Appendix VI). The block diagram of the Doppler shift monitoring interfacing is shown in Figure 7a.

2.4.1 Reference oscillator

The first stage of locking the reference oscillators is to divide a conveniently available output frequency down to a more suitable frequency for use by the phase comparator. The ICOM's reference frequency is available as 10.24 MHz after division by 3 from IC5 on the PLL UNIT board. The reference of 1 MHz from the Systron-Donner clock is first band-pass filtered through a tuned balun (eliminating ground loop inductions) into a CMOS Schmitt-trigger.

Both are then divided down by mostly CMOS dividers (to improve rejection of HF interference from nearby ANARE transmitters) to a frequency of 20 kHz. It was purposely chosen to differ from the ICOM's internal PLL frequency (10 kHz) to reduce the chance of 'frequency beating'. After active low-pass filtering, the output of the phase-comparator is then presented to the ICOM's 'calibrator' potentiometer circuit. The potentiometer pull-up resistor R1 of 1 kΩ needs to be disconnected, as it loads the active low pass filter too much. The 3dB cut-off frequency was chosen to be 160 Hz. This is a compromise between the expected short term error of the Xtal of 0.20 Hz (1 part in 10^5 of 20 kHz) and the actual 20 kHz output frequency of the phase-comparator. The phase-jitter of the resulting reference frequency was negligible as measured on an ordinary CRO (i.e. less than about 2° jitter).

2.4.2 Beat frequency oscillator

The second receiver's (ICOM#2) reference oscillator was also locked in the above manner. Then, to produce a frequency in the 9 MHz range to suit IF4, a mixing of LO1 and LO2 was required. The range of LO1, even when mixed with LO2, does not reach down to 9.0106 MHz. Hence, a synthesised frequency of 79 461.22 kHz is chosen for LO1. The ICOM#2 in the CW mode must actually be tuned (by the main dial) to 9.010.62, but the '2' (for 20 Hz) is not displayed on the front panel. Dialling the last 20 Hz can be verified either with an ordinary frequency counter on the output of the digital Δf interface to LSI#1, or by observing the chart recorder channel with the analogue Δf level (channel 2, Figure 2). After mixing with LO2 of 61.44 MHz we get 18 021.22 kHz. This is then divided down by 2 to arrive at 9010.61 kHz, which is 10 Hz above the heterodyned carrier of VNG. By comparison, the ICOM's standard BFO runs at 9.0098 MHz in the CW mode, producing an audio beat of 800 Hz. Figure 8 shows the bandwidths and centre frequencies of the IF stages in the IC-R71A.

In the mixing process, the amplifier stages IF1, IF2 and IF4 have inverted the spectrum of the received frequency. In other words, through these IF stages the original upper side-band (USB) is actually lower in frequency than the heterodyned carrier. For a positive Doppler shift to appear as an increase in '10 Hz+Δf' frequency, we would have to choose the BFO to be 10 Hz above rather than below the carrier. This was in fact not done during the recordings of 1986-87, so all uncorrected Δf data show an inversed polarity. The data shown in this report have been corrected.

In ICOM#2, the LO1 and LO2 are no longer connected to mixer 1 and mixer 2, but are rewired to the additional BFO mixer. LO1 and LO2 from ICOM#1 are then also fed into ICOM#2, effectively tuning it to the same HF signal. However, ICOM#1 is operated in the CW mode, and therefore tuned 900 Hz below what the AM mode requires. ICOM#2 is therefore actually mis-tuned by 900 Hz in the AM mode. As the bandwidth in the normal filter selection covers the 1 kHz audio plus 900 Hz sufficiently, this mis-tune has negligible effect.

The 18 MHz output of the additional BFO double-balanced mixer is fed into an active HF divider. This is a video opAmp which is used as an under-critical oscillator of which the output circuit is tuned to about 9 MHz. This divider is followed by a single stage amplifier, boosting the signal to more than what is required by the single side-band (SSB) product detectors. This BFO mixing circuit is fitted in a vacant location in ICOM#2. The output is wired to both receivers, making inter-comparison possible.

2.5 THE FREQUENCY METER

To measure the '10 Hz+Δf' signal and provide an analogue value of the Doppler shift (Δf) for use on the chart-recorder, a PLL frequency meter was built. The first stage is an active low-pass filter at 500 Hz. This removes the IF component in the signal from the SSB detector, but retains the fast response and boosts the signal to 5 V peak to peak (clipped). The output is also fed separately into an audio tape-recorder frequency modulator with a centre frequency of 3.375 kHz.

Initially an attempt was made to employ a frequency-to-voltage IC (LM331). This was rejected however, as the output is either too ripply or the response too slow, depending on the value of the output buffer capacitor chosen. This is due to the very low frequencies being measured (around 10 Hz).

Using the VCO control voltage of the PLL/VCO CMOS IC MC4046B, a compromise between the response time and low ripple could be much more easily found. The low-pass filter between the phase-comparator output and the VCO input is determined by data-sheet values for 10 Hz with a lock range of about 20 Hz. The buffered source follower (SF) output of the PLL IC is then fed into another active low-pass filter, designed to operate in the linear phase mode at 0.5 Hz. Its output carries the 10 Hz offset voltage, which can be eliminated by using a potentiometer between +5 V and ground (GND), set to the same voltage, and connected to the chart-recorder 'negative' input.

2.6 DIGITAL FREQUENCY RECORDINGS

The '10 Hz+Δf' was measured digitally by a standard DEC LSI-11 microcomputer as shown in the block diagram in Figure 2. This was achieved using the REQ A input of a DRVIIC parallel interface card.

3. TIME-OF-ARRIVAL MEASUREMENTS

3.1 INTRODUCTION

As has been noted earlier, a second receiver was required to synthesise a stable BFO for the Doppler frequency shift measurements. By feeding this second receiver the local oscillator frequency of the first, it could be tuned to the same station as the first receiver. The second receiver was operated in AM-mode detecting the one-second time marks.

A mini computer was used to sample the one-second time marks via a fast analog to digital interface with direct memory access. The unit chosen for this was a data translation model DT2758. A FORTRAN program was written to deduce the average arrival time of the first few zero-crossings of the 1000 Hz time marks. However, this time value includes a demodulation phase error due to frequency dependent refraction (i.e. dispersion) in the ionosphere. By software, an attempt was made to retrace the demodulation to the inferred start of the time mark.

Time measurement in this manner is considered more appropriate than the more common amplitude-envelope detectors. The latter are more adversely affected by signal fadings and rise-time distortion induced by narrow-band audio filters and rectifiers. Measuring the demodulation through a standard audio receiver circuit, and determining the start of the time mark retrospectively by software allows for considerably greater accuracy. Figure 9 illustrates how frequency selective fading introduces the above-mentioned time mark detection errors.

In order to maintain time accuracy, the triggering of the A-to-D converter was provided by a trigger burst synchronised to the external laboratory clock (see block diagram in Figure 7b). The length of the trigger burst is determined by the software, enabling the amount of collected data to fit the time needed for the required processing in the remainder of the one second period. To prevent the software time of day clock drifting with the unsynchronised mains power supply at the ANARE station, a substitute 50 Hz signal is also derived from the laboratory (Systron-Donner) clock to control the minicomputer's line-time-clock (LTC) (Figure 7c).

The A-to-D interface used can sample four channels simultaneously, enabling comparisons between propagations over more than one HF frequency used by the same station, or comparisons between both side-band demodulations. For this purpose, two small PLL exhalted carrier selective sideband (ECSS) short-wave receivers were obtained. However, time restrictions during the study period prevented these comparisons.

3.2 THE AUTOMATIC GAIN CONTROL

In order to make sensible time-of-arrival measurements further modifications are required to the receivers.

The IF automatic gain control (AGC) circuitry in the ICOM receivers not only reacts to the carrier, but also to the sidebands, to cater for the SSB modes. This is a disadvantage when monitoring the AM signal, during the Δt (time-of-arrival or TOA) measurements. Every second, when a time mark arrives, the AGC will adjust to the increased signal. However, due to the attack-time transient of the AGC, the detected audio signal envelope will also show a disturbing transient. This is of particular importance when tracing zero-crossings and monitoring the signal-to-noise ratios.

As ICOM#1 is operated in the CW mode with the narrow filter activated, both 1 kHz modulations in the sidebands are adequately supressed. Its AGC will then not be affected by the time-marks. Therefore, to remove the transients found in ICOM#2, its AGC is turned off, and the AGC of ICOM#1 is wired through and used instead. This is acceptable, since both receivers are tuned to the same HF frequency, and are sufficiently compatible in performance. Due to selective fading, the time mark amplitudes are still not perfectly levelled, but the transients are reduced to the remaining effect of the filtering in the audio amplifiers.

3.3 A-TO-D TRIGGERING

The trigger pulse train required for sampling the Δt time mark signal is limited to a rate of 25 kHz. That appears to be the limit of the LSI-11/02 using the provided software library package (DTLIB) in the fast-sweep mode. It suspends all other CPU operation, including the line-time-clock, and concentrates on data collection. Even the optional direct memory access (DMA) mode did not seem to be faster when sampling a single channel. The A-to-D card (data translation DT2758) can sample four channels simultaneously. Only when sampling more than one channel is the DMA mode faster than the fast-sweep mode. This is because the subsequent channels are sampled, converted and DMA stored by on-board control rather than by software or external triggering.

To produce a real-time accurate 25 kHz sampling train, a 400 kpps signal is derived from the divider chain which takes the 1 MHz reference from the Systron-Donner, and divides it down to 20 kHz. It is then gated with the '1-second' output of the Systron-Donner, producing a 25 kHz pulse train of 0.5 second duration. This is the default duration, allowing the LSI to synchronise to the trigger sequence. Once synchronised, a LSI output signal (via a DRV11C parallel board) indicating 'sampling done', controls a subsequent gate which limits the duration of the pulse train even further.

This 'sampling done' signal is generated by software after the required number of time mark cycles have been sampled, and resets the set/reset flip-flop driving this gate (Figure 7b).

4. SOFTWARE DEVELOPED

4.1 DOPPLER MEASUREMENTS

Various versions of FORTRAN programs were developed to measure the variations in the carrier phase delay thus yielding the Doppler shifts (Δf). The last version is listed in Appendix IV.

The version named VNGFSD counts the programmable LSI clock (MDB KW11P) running at 10 kHz through an overflowing buffer ('repeater cycle'). Whenever a positive-going transition of the Δf signal is sensed on the parallel interface (DRV11C) REQ A status line, the clock buffer is sampled. By subtracting two consecutive values from the clock buffer, the previous '10 Hz+Δf' period is determined. However, the clockbuffer is interpreted by the FORTRAN program as an INTEGER value with a sign bit, so only fifteen bits can be used for counting the 10 kHz clock. The longest period it can measure is 2^{15}/10 kHz=3.3 sec, or a Doppler shift of -9.7 Hz. This is more than sufficient for our use, but the resolution of 0.01 Hz (having nominally one thousand counts in each period of the 10 Hz signal) introduces a considerable error in the Δf standard deviation values, σ_f. Bringing the clock frequency up to 100 kHz does increase the resolution sufficiently, but limits the Δf shift range to -7 Hz. This could be overcome by starting the counter at the lowest integer value (-32768) rather than zero, thus doubling the count range (i.e. using all sixteen bits). However, the computing time required for testing and correcting for overflows often exceeds the one second available.

A version named TIMFSD (Appendix IV) was used to overcome computing time restrictions while using the clock running at 100 kHz. In this version, the KW11P clock (in the single cycle mode) was started at one positive going transition of the 10 Hz+Δf signal, and stopped at the next. The sixteen bit unsigned value was then converted to a floating-point value, allowing higher resolution. The unfortunate, but undramatic side effect is the halving of the number of samples taken, skipping each alternate 10 Hz+Δf period. The range is now also increased to 2^{16}/100 kHz=0.66 sec, corresponding to a Δf of -8.5 Hz. The initial software limit is set to + and -8 Hz, but later software revisions limits all Δf shifts to five times the previous minute's standard deviation σ_f (i.e. if five times σ_f at one stage is 3 Hz, and the sample taken is 5 Hz, the value recorded is 3 Hz). This variable limit method is adopted to prevent rare extreme excursions affecting the average value disproportionately. The number of occurrences of these clipped samples are given in the hourly summaries. Typically, over the average of eighteen thousand samples per hour, an average of one hundred are clipped, and one hundred are too long (i.e. less than 1.5 Hz), probably due to missed zero crossings. As no raw data were recorded in digital form, no study has been made of the effect on the averages when using this 'clipping' method.

Other versions of the programs selected the various output media, such as printer, chart recorder and finally the hard disk. This is indicated in the program names: F=Δf, S=signal-strength, P=print, C=chart, D=disk and CAL= a calibration version which displays averages over sixty samples on the VDU. In particular the printer version was written for the foreground-background operating system RT11-FB, allowing the output to be spooled and printout to be time-shared with data collection.

At one stage during the development, the software-derived Δf value was recorded on the chart, allowing comparison with the output from the PLL frequency meter. After introducing software smoothing approximating the PLL low-pass filtering effect, the results were comfortingly similar.

The average Δf with σ_f, and the average signal strength (S) with σ_s over six hundred periods for VNGFSD or about three hundred periods for TIMFSD are presented in numerical value and as a print/plot for easy assessment of the effects of periods of geomagnetic activity. Similarly, a summary and average is given each hour. In VNGFSD no system clock was used and six hundred periods of '10 Hz+Δf' were counted as one minute of real time in the computer printout. To overcome this inaccuracy, TIMFSD uses the system line-time-clock (synchronised to 50 Hz derived from the Systron-Donner clock) to interrupt sampling at the correct intervals.

The version using the disk rather than the printer has an extra subroutine producing an unique filename every hour, or whenever the program is started. The filename is derived from the system date and hour, at the time of opening the file (e.g. 23JAN7.H12). The filename extension is modified every time the program attempts to create a file more than once during the same hour. It is recommended for future use, to reverse the filename format (i.e. 7JAN23.H12), thus yielding sequential disk directory listings.

4.2 TIME OF ARRIVAL MEASUREMENTS

Various versions of FORTRAN programs were developed to determine the variation of the group path delay and the modulation delay. They are all named BLEEP with a version number added (Appendix V).

The first part of the BLEEP programs verifies whether the A-to-D sampling trigger arrives every second for a maximum duration of 500 ms, and has a rate of 25 kHz. It then synchronises the program so, that every new cycle of audio sampling commences during the second half of the 1-second cycle (without trigger pulses), and idles until the exact start of the next 1-second mark of the Systron-Donner clock. This idling period can be adjusted by software parameters, and verified on a CRO which monitors a spare D-to-A channel which changes level during the periods of idling.

Due to ionospheric dispersion and other distortion and fading of the time mark modulation, simple demodulation and envelope detection will not necessarily give an accurate value of the group path delay (often called the time-of-arrival or TOA). To prevent noise falsely triggering an envelope detector, a level discriminator (Schmitt-trigger) with a fixed trigger level above the expected noise is normally used to detect the start of the time mark reception. Often, the audio detection is followed by a narrow band 1 Hz filter, to further reduce the effect of noise. Due to the rise-time restrictions of such a filter and the effects of unpredictable signal fading, the resulting TOA value using this method may be biased (i.e. too late) by an erratic number of 1 ms cycles (Figure 8). As we are looking at variations of much less than 1 ms, these errors cannot be removed by statistics.

The BLEEP programs attempt to reduce this error by measuring (every second) the actual noise level during a 3 ms slot commencing 4 ms before the expected TOA. The trigger level is then set to twice the peak noise level, ensuring a 6dB S/N ratio minimum. The BLEEP1 version then searches for a certain number of zero crossings following the trigger. The number collected is limited only by the time required to process the data before the next time mark arrives. It calculates the average location of the first zero crossing, by superimposing the subsequent crossings brought forward by the appropriate number of half cycles. The actual start of the time mark is here assumed to have been a half cycle before that point (after the trigger), but drowned in the noise. This is illustrated by the dashed line in Figure 8. This we will call the modulation delay, to distinguish it from the total group path delay. The BLEEP1 program only collects a modulation delay estimation. A running average of the last sixty determinations of the modulation delay is output to the chart by this program.

The zero crossing timing calculations are limited in accuracy by the resolution available when sampling the audio at the rate of 25 kHz (twenty-five samples per 1 ms period). In BLEEP2 an array is generated with the number of occurrences of zero crossings in each of the twenty-five sampling slots. The peak slot is then regarded as the mean arrival time of the zero-crossings. Due to the transient zero-shifting of the signal in the band-passed audio section of the ICOM receiver, an erratic discrepancy of some significance is evident here. The calculated data were only shown on the VDU, and are not available in digital form, except for a period of about 10 minutes (printout). Over a typical example of sixty samples, the average Δt was 4.77 ms, with a $\sigma=4.3$, but about fifteen samples obviously skipped one (half-) cycle. This process was therefore rejected.

In BLEEP3 a different approach was tried, by summing the measured voltages over blocks of twenty-five samples. The level of each half-cycle peak is tested, to ensure it is higher than the noise threshold, in case we have a short time mark, or one of low amplitude. Note that except during BCD and quarter hour sequences, VNG transmits fifty-four normal marks of 50 ms, four short marks of 5 ms, and one skipped mark followed by the long 500 ms full mark. The resulting waveform represents a one-cycle average over the 25-30 ms sampling window, and from this the averaged modulation delay can be determined. The latter method was found to be consistent with observation of the time mark signal displayed on the CRO.

The modulation delay as determined from the zero-crossings of the demodulated time mark, is different from the group delay by a certain phase error Φ (Figure 10). The 1 kHz demodulated signal drifts within the envelope of the actual HF group signal. Hence, the phase error causes the 1 kHz signal to rise out of step with the commencement of the envelope. Ideally, the demodulated signal would step directly to the momentary value of that phase. The bandwidth limitations of both the transmitter and the receiver reduce that step with an inverse exponential factor. The resulting rise time will then reduce the peak amplitude of the first part cycle to such an extent, that it might drop below the noise discrimination level (solid line in 'Rx' of Figure 8). The resulting error still remains in the BLEEP programs, but the actual timing error as a result of missed peaks appeared to be only minor under typical S/N ratio conditions found at Macquarie Island (less than about ten degrees phase error under average conditions).

Determining the modulation delay using the zero-crossings is fairly easy, but determining the exact start of the group envelope is complicated by the noise preceeding the measurement triggering point. There may be other factors, but it appears that the phase error is mainly due to the difference between the delay the modulation encounters and that encountered by the carrier, f_o, resulting from the frequency dispersion in the ionosphere (Toman 1967, Cornelius 1979). The modulation consists of the upper side band (USB), f_o+1 kHz, and the lower side band (LSB), f_o-1 kHz (Figure 9). Even though the difference in frequency is relatively small, the ionospheric dispersion still causes them to follow different paths. Figure 10 illustrates the various terms using phasor diagrams. The phase difference resulting from the dispersion might be minimal in terms of time (δ), but is significant in terms of the phase relation ($\alpha\delta$) between the HF carrier and the side bands. This phase difference is carried through the AM demodulator to become a phase error (Φ) in the 1 kHz signal with respect to the HF envelope.

While program BLEEP1 only determines the modulation delay, programs BLEEP2 and BLEEP3 were written in an attempt to determine the group delay and the phase difference Φ. An estimation of the group delay is made by tracking back through the digitally sampled values of the audio signal strength to the first zero crossing prior to the triggering point (i.e. the point exceeding 6 dB S/N). This result may be corrupted by noise, but considering the time available to run the program between the 1 second marks, no higher level of sophistication was attempted. The individual determinations were always averaged over sixty consecutive successful samples before being output to the chart as a deflection on the modulation delay trace (Figure 11). The average over sixty determinations reduces the effect of noise corruptions on the individual determinations of the group delay.

In BLEEP2 and BLEEP3, after taking the first sixty time marks, the averaged group delay value is used to limit the range of an acceptable group delay determination to within +/-1 ms of this value. Any time marks starting before or after 1 ms from the averaged group delay value are rejected as being due to noise. After a sixty sample averaged group delay has been determined, the measurement of a peak noise value is extended from the initial 3 ms of the initialised sampling window, to from the beginning of the initialised sampling window, to within 1 ms of the averaged group delay value.

In the case of sixty consecutive rejects, the acceptance window is widened to the original settings. In the case of BLEEP1 and BLEEP2, the window is selectable via the keyboard to between 10 and 150 ms, but, at least initially, the first 3 ms of this period is used to measure the peak noise level. In practise windows longer than 25-30 ms require more computing time than is available between the one second time pips for analysis. Longer windows could be used by skipping alternate time marks, thus allowing a longer time for analysis. Averaging too many zero-crossings to determine the modulation delay may however introduce an error due to 'fluttery' ionospheric conditions. Hence BLEEP3 commences with a 25 ms window which alters only during the lock to the smoothed group delay.

5. DATA COLLECTED AND PRELIMINARY ANALYSIS

The VNG timing signals are transmitted on 4.5 MHz (0945-2130 UT), 7.5 MHz (2245-2230 UT) and 12.0 MHz (2145-0930 UT) each day. Only one frequency was monitored at any one time. The experiment was operated intermittently between 19 August 1986 and 26 January 1987. Appendix I indicates which frequencies were monitored at what times during this period. During the data recording period the experiment was being upgraded. At various stages data were recorded on chart, printed, stored on disk and collected on audio tape. Figure 2 shows, in block diagram form, the most developed stage the experiment reached.

Initially Doppler frequency shift and HF signal strength data only were monitored on chart. Some 1135 hours of data spread over 84 days were collected in this manner. An LSI 11 computer was incorporated into the system on 8 October and the average and standard deviation of sixty samples of the Doppler frequency shift and HF signal strength were calculated and printed from this time. One thousand one hundred and forty-four hours of data in 67 days were collected in this form. On 26 October the H component of magnetic pulsations was added to the chart records as a general indicator of magnetic activity. This was discontinued on 10 January 1987. A hard disk storage system was incorporated with the computing system on 23 December 1986, and from this time the Doppler frequency shift and HF signal strength averages and standard deviation values were also stored in digital form. Digital collection of these data amounted to 698 hours over 35 days.

The Doppler shift and HF signal strength data were also recorded, in frequency modulated form, on audio tape along with the raw VNG time mark audio and a locally generated time code. Seventy hours of data over 15 days from 29 December were collected in this manner.

From 11 January a second computing system was used to estimate the time-of-arrival of the time pip. Between 11 and 17 January the modulation delay was recorded on chart for 96 hours. From 18 January to 26 January a sixty sample averaged group delay value was superimposed as a deflection on the modulation delay trace (Figure 11). One hundred and ninety-seven hours of data in this form were collected.

The digitally recorded 1 minute averages of the Doppler frequency shift are presented in the form of stacked diurnal plots in Appendix II. The regularity of the diurnal variation of the ionosphere's electron density and effective height is apparent in these plots. In the afternoons and evenings the dominant Doppler shift is negative, and after sunrise it is generally positive for quite some hours. Nighttime variations are generally negligible, but even the slightest disturbance during minor magnetic storms has a considerable effect on the Doppler shifts. An active geomagnetic pulsation period accompanied by a green aurora at Macquarie Island, shows up clearly on the Δf graph on the night of 2 January 1987.

Results of modulation and group delays are only available on charts. Numerical values were displayed on the VDU screen during the software development, showing the high reliability of the procedure used to obtain the group path delay and the accompanying modulation phase shift measurement. Not many time marks were missed or rejected, as is visible on the chart traces. Whenever a time mark is missed, the smoothed delay values are 'held'. The envelope timing (group delay) is shown every sixty time marks, equivalent to 61 seconds during good reception, as the fifty-ninth time mark every minute is not transmitted by VNG. If any time marks are missed or rejected, the group delay marks are further apart. As the chart shows (as in the sample of Figure 11), that did not happen very often. It clearly shows the success of this software approach.

6. FUTURE STUDIES

In the future, the first step would be to develop the software further, to store the Dt parameters on disk. Furthermore, there are two PLL-ECSS receivers (ESKA RX12PL) available, allowing comparisons between Δt of two different HF frequencies of VNG, and/or the USB versus LSB delays. Ultimately, a third ICOM receiver could be incorporated for inter-comparison of Δf for two HF carrier frequencies.

Using a later (and faster) model LSI, the two Δf and Δt measurements could be combined into one computer system. Software could run in a time-sharing mode with the Δt measurements interrupting the Δf tracing every second for a short duration. The receiver has interfacing for a digital remote control, enabling automatic HF radio frequency changes at the appropriate times. With the improvement of the software, and the elimination of the need for a chart recorder, this would allow long periods of unattended operation.

The studies could be extended to other ANARE stations, comparing the ionosphere's behaviour over different paths. The monitoring is of course not limited to VNG, and other stable transmitters could be considered. Ideally, a dedicated ANARE transmitter could be used, located in Hobart for instance, which is modulated with two (or more) different frequencies simultaneously. This would allow the group delay to be determined more accurately, comparing the modulation delays of the audio frequencies. In this way, the ambiguity (when the phase-shift exceeds 2π) of the results could be eliminated. Also, it would allow closer study of non-linear frequency/delay behaviour close to the maximum usable frequency (MUF). A similar study on a shorter ionospheric hop between two points on Macquarie Island commenced during the 1987 winter season, comparable to previous studies in Victoria (Baulch 1984).

The VNG frequency and time signal service utilised in this research was closed down on 1 October 1987.

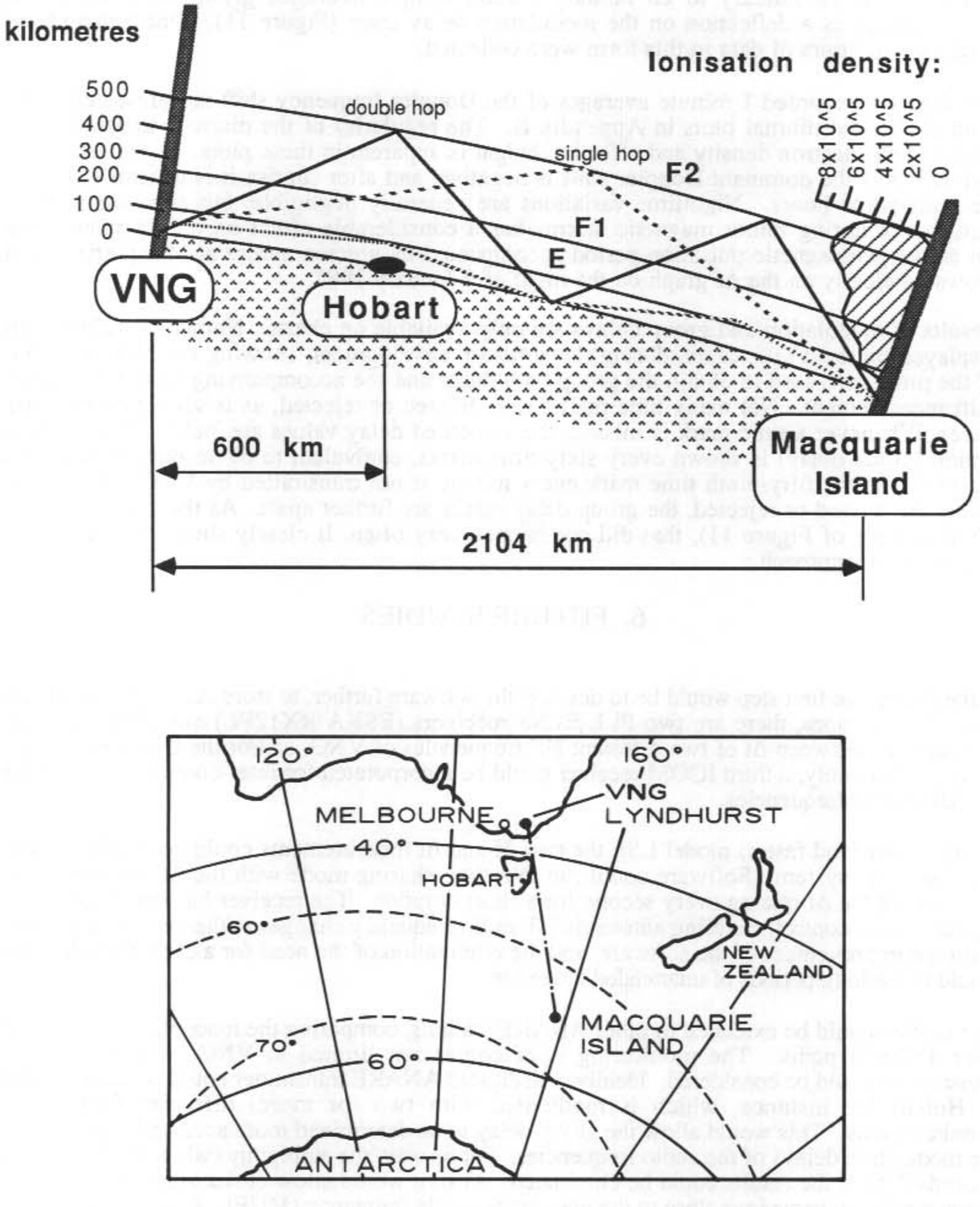


Figure 1. The path under study.

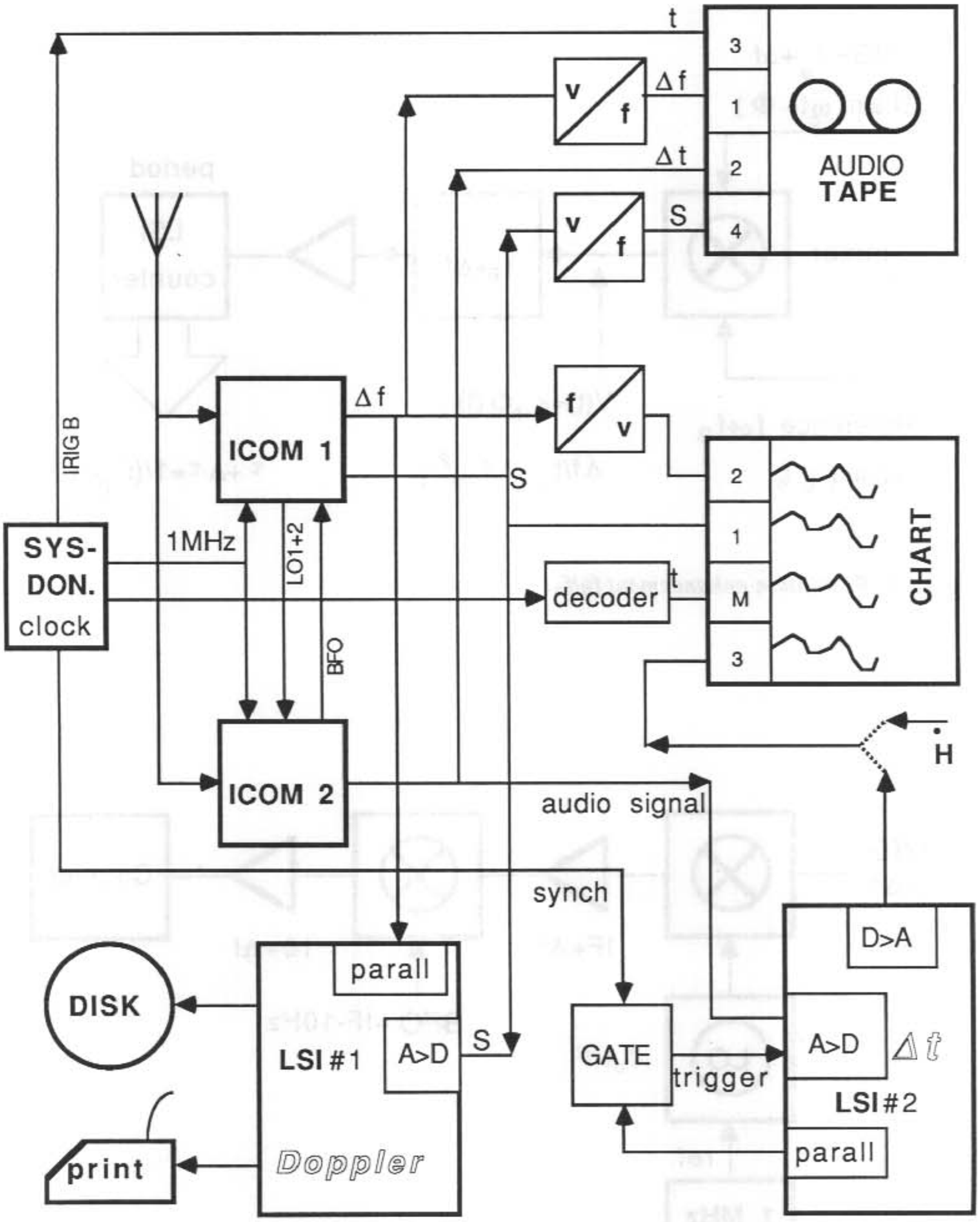


Figure 2. Main block diagram of the experiment at its most developed stage.

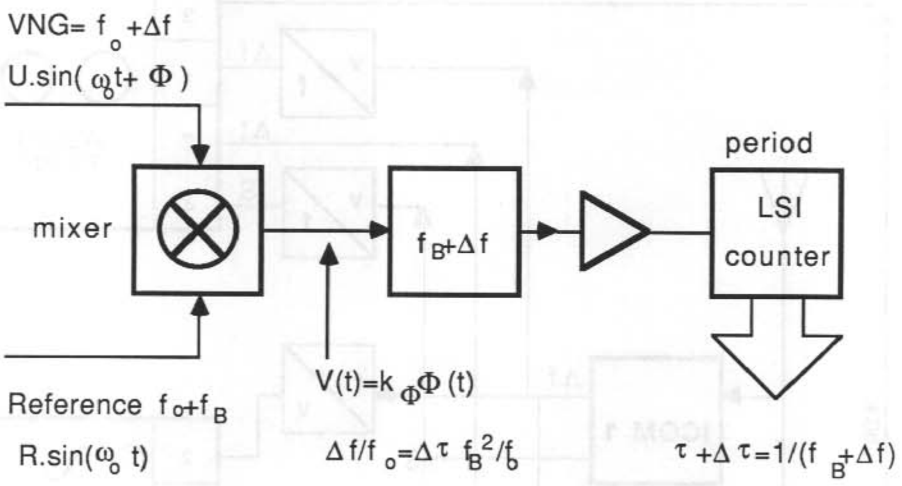


Figure 3. Resolution enhancement f_B/f_o.

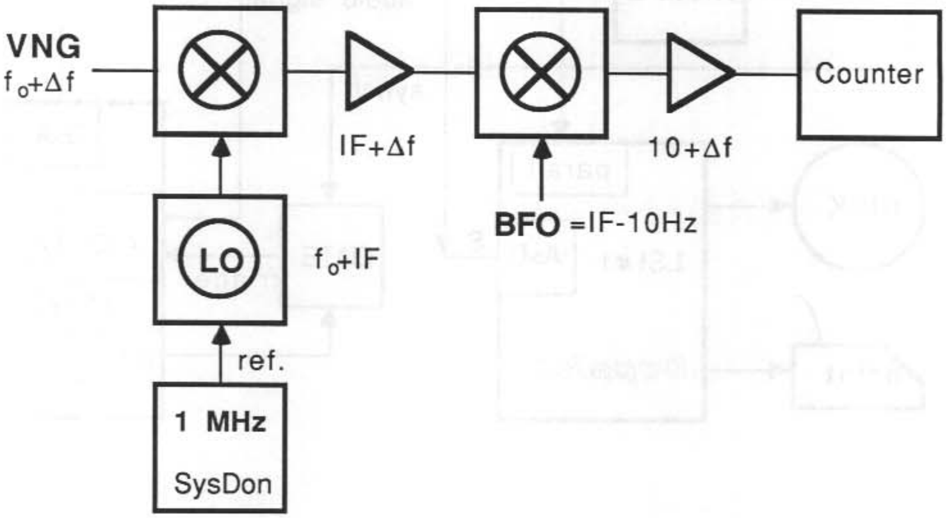


Figure 4. The Doppler monitor principle.

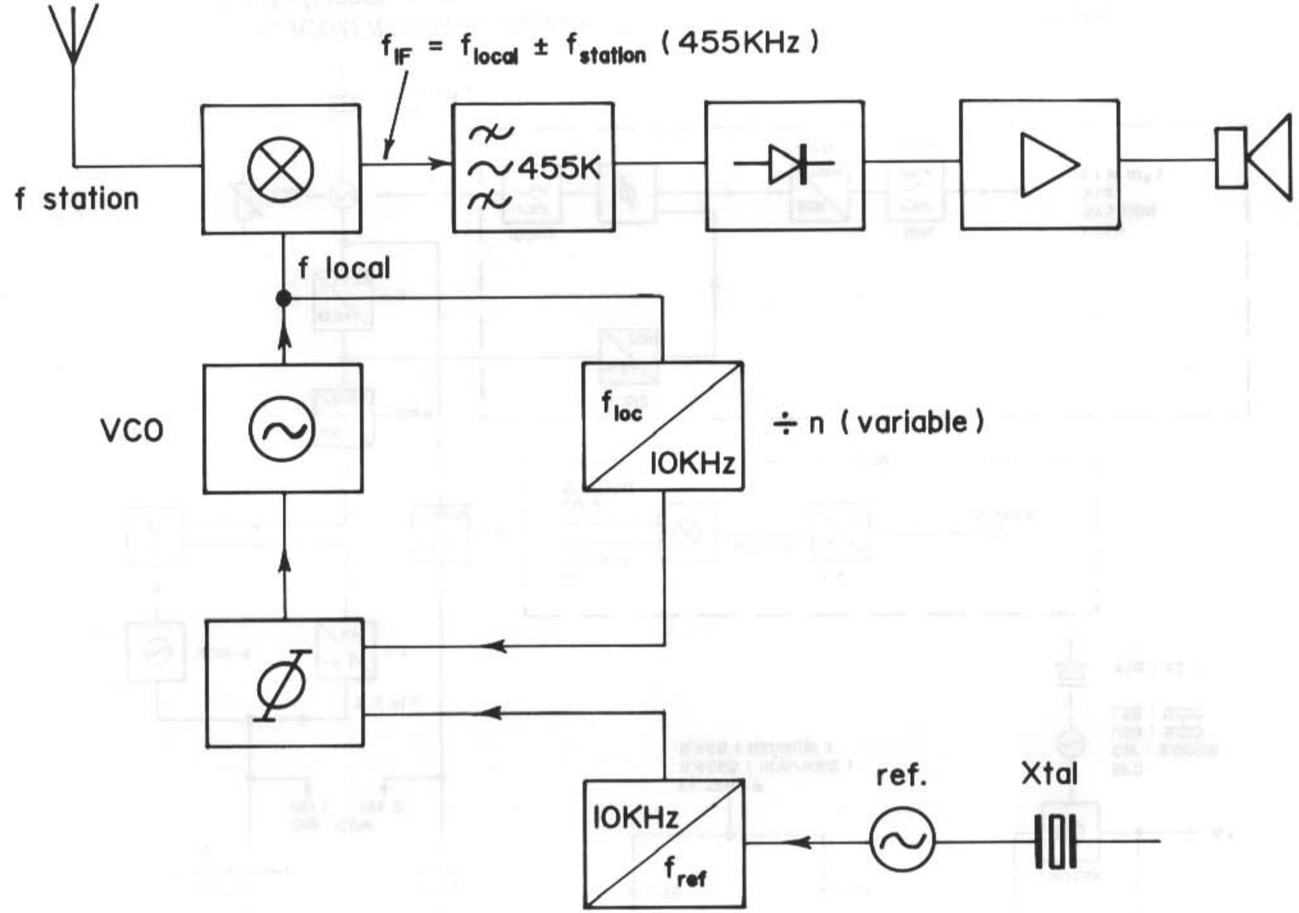


Figure 5. A normal heterodyne receiver with a phase locked loop local oscillator.

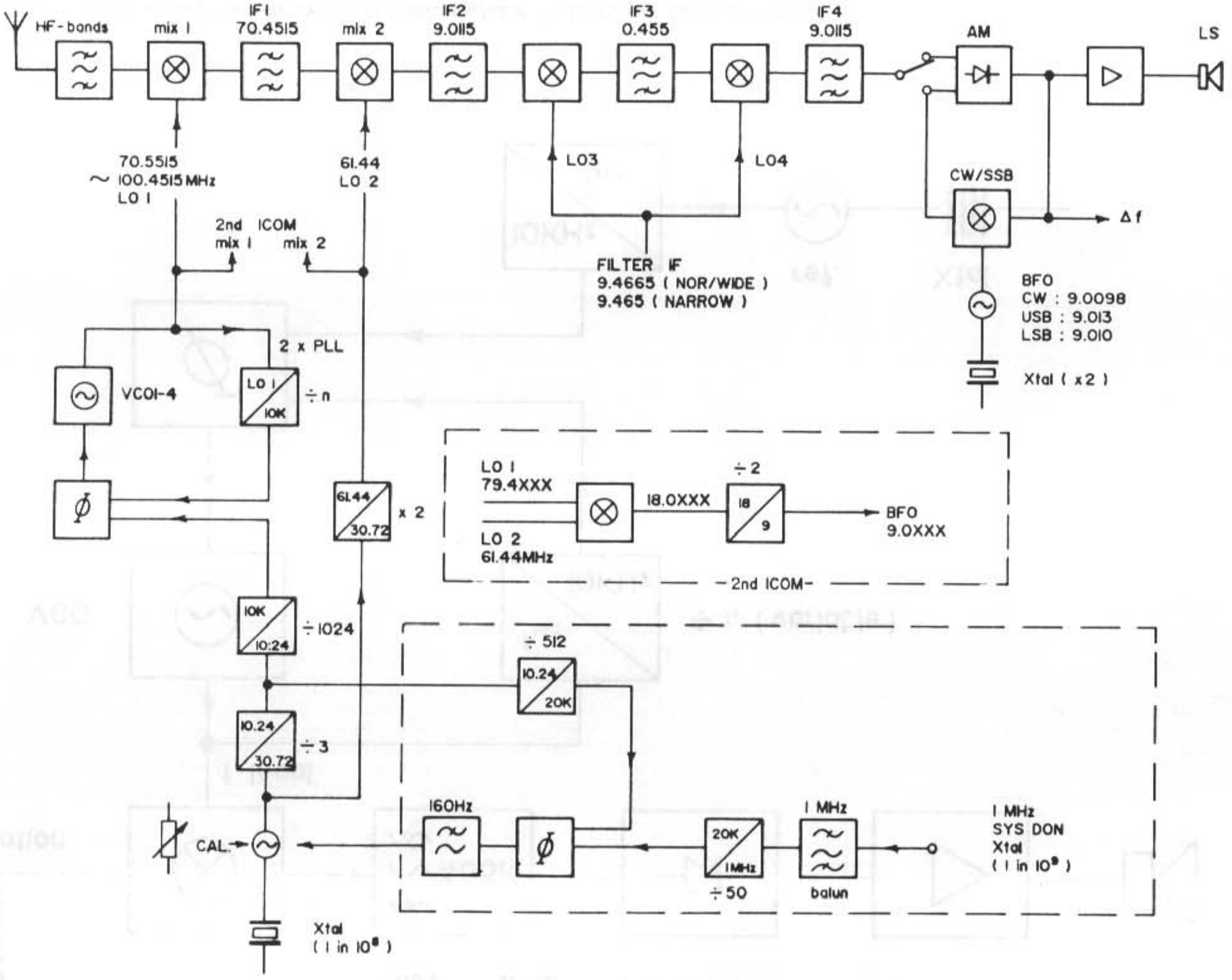
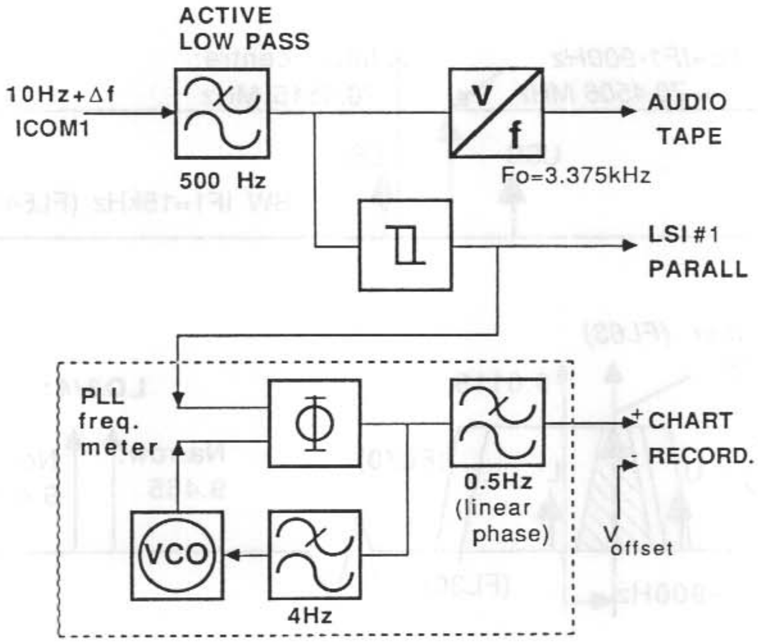
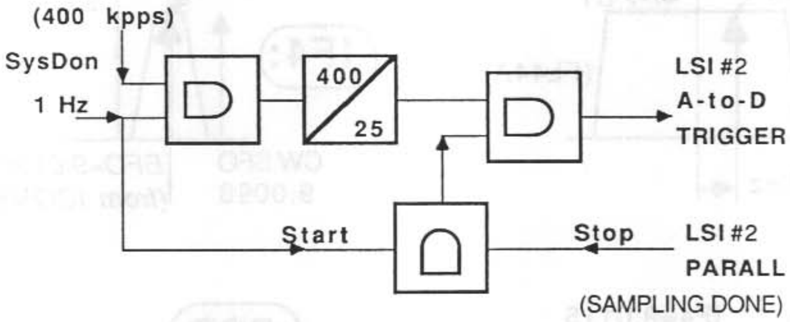


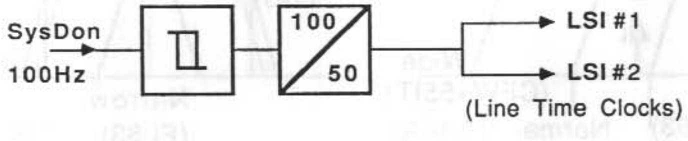
Figure 6. The ICOM R71A block diagram. The boxed sections are the additional circuits required for the Dopper monitor.



A: *Δf signal interfacing*



B: *Δt sampling trigger*



C: *LSI clock signals*

Figure 7. Interfacing block diagrams.

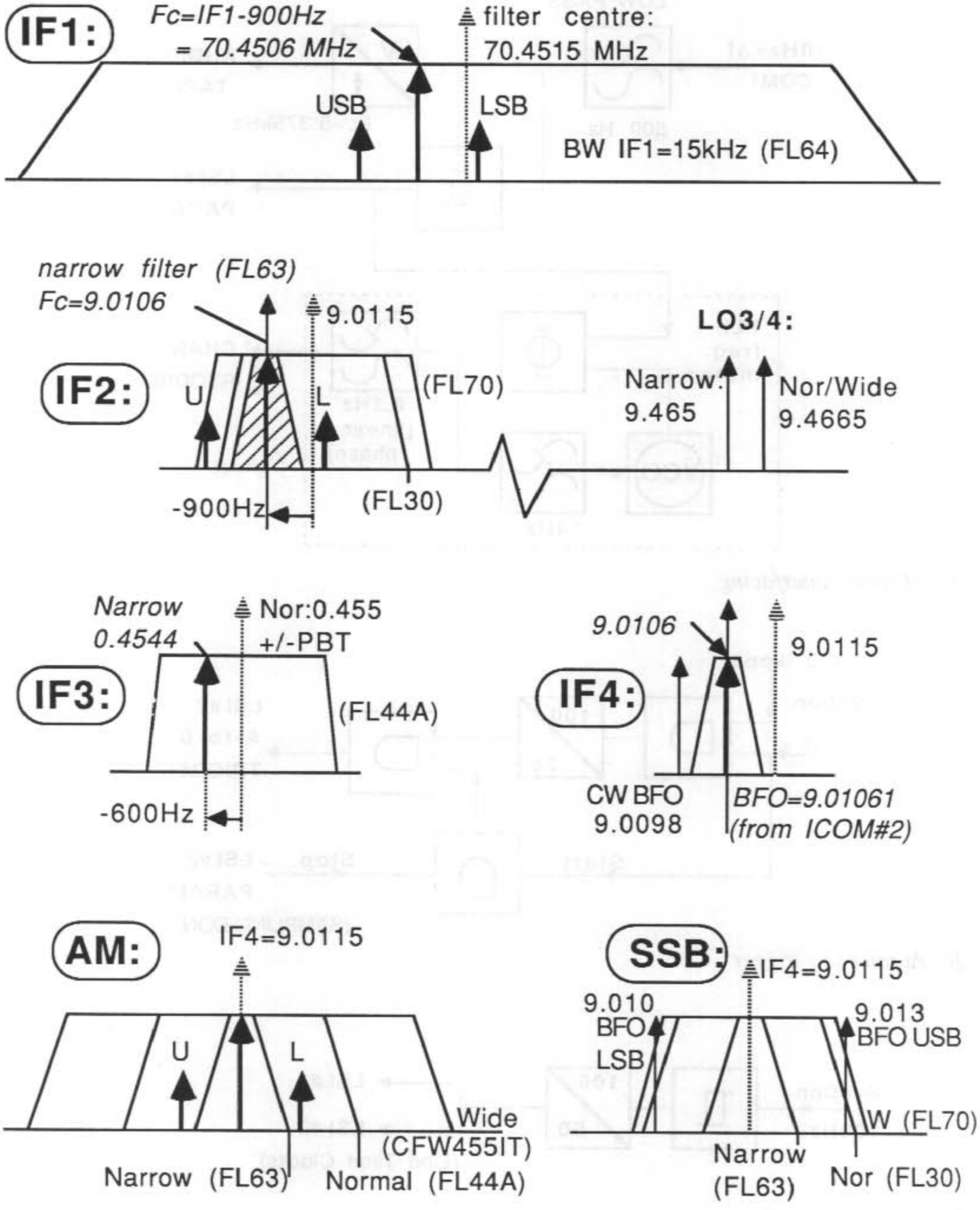


Figure 8. The IF band filters (frequencies in MHz).

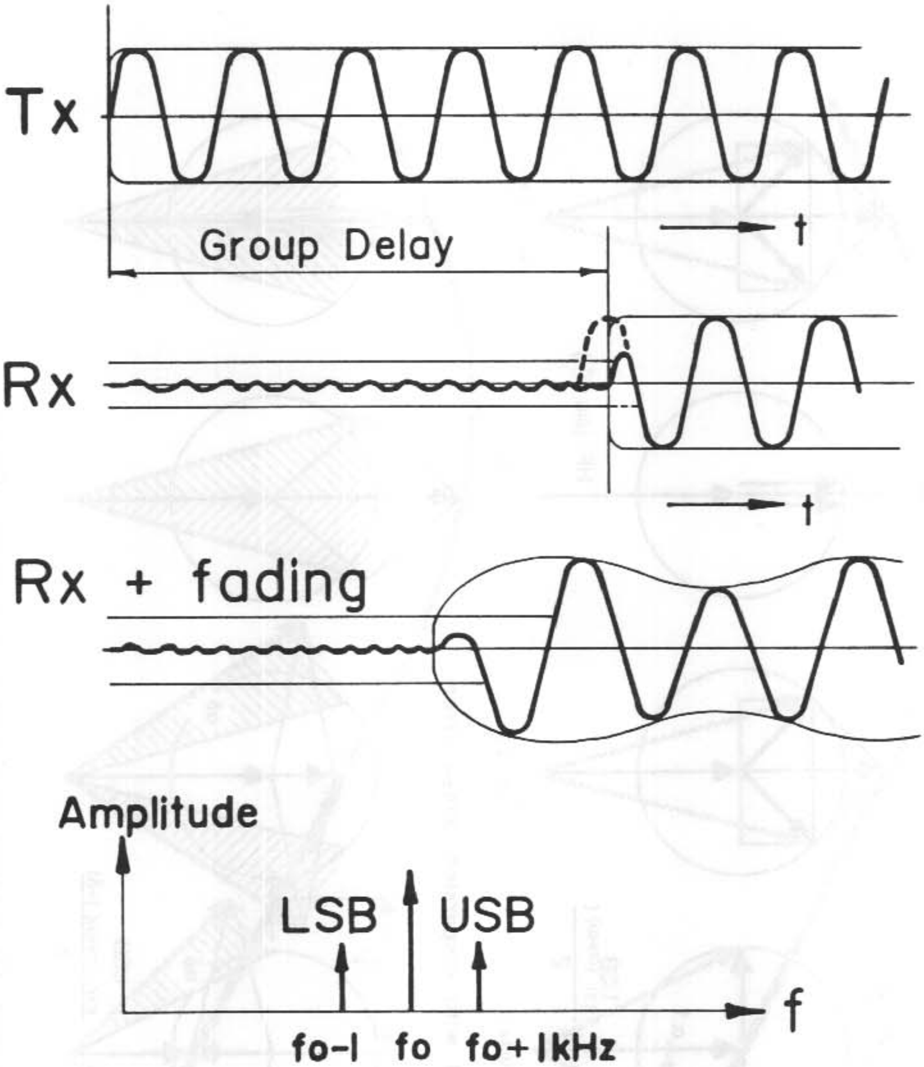


Figure 9. Errors in determining the group delay by demodulating the time marks. The first cycle shows the phase error due to frequency dispersion of the VNG signal with sideband components. Fading aggravates the error in time detection, when using fixed threshold detectors.

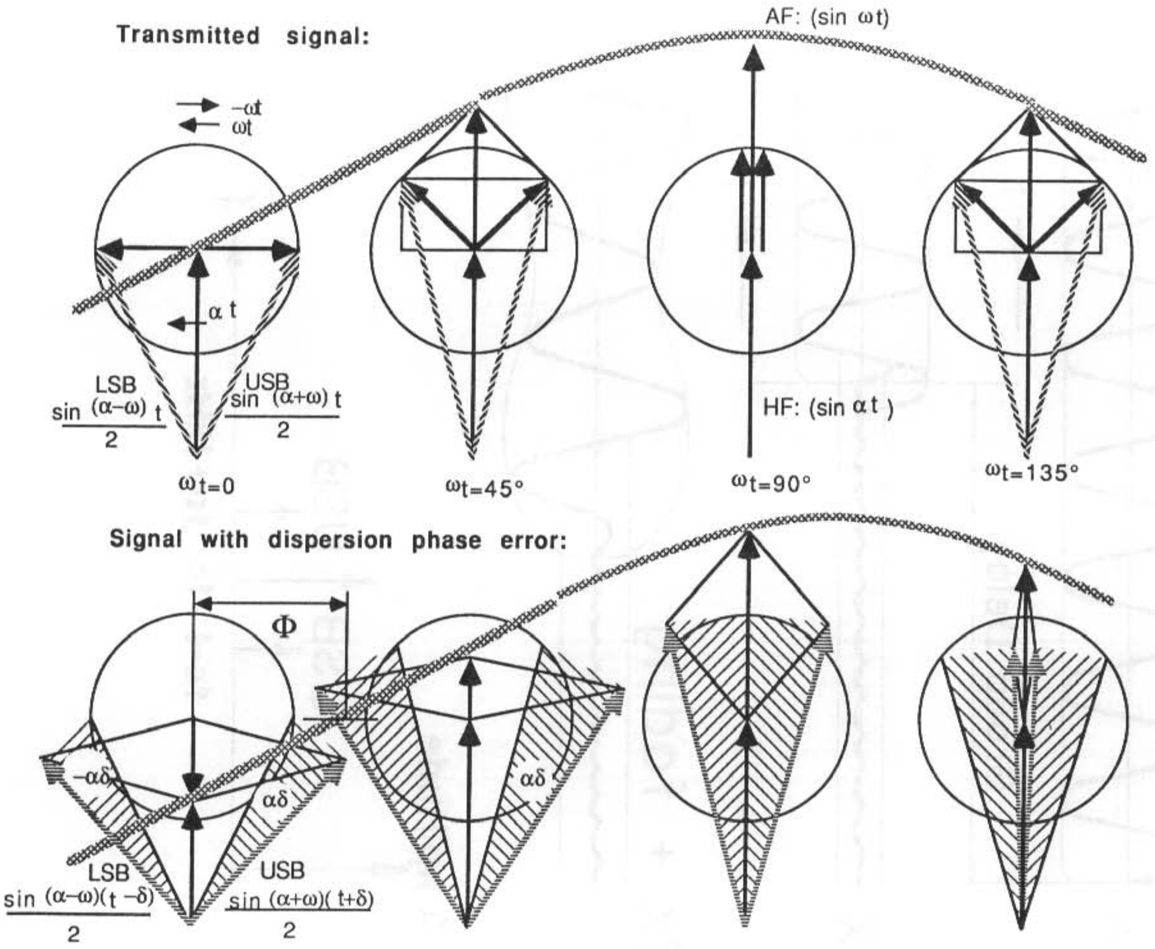


Figure 10. The effect of frequency dispersion on the audio signal. The LSB travels faster and the USB travels slower than the carrier (by time δ). The phasor diagrams show an integral number of rotations of αt.

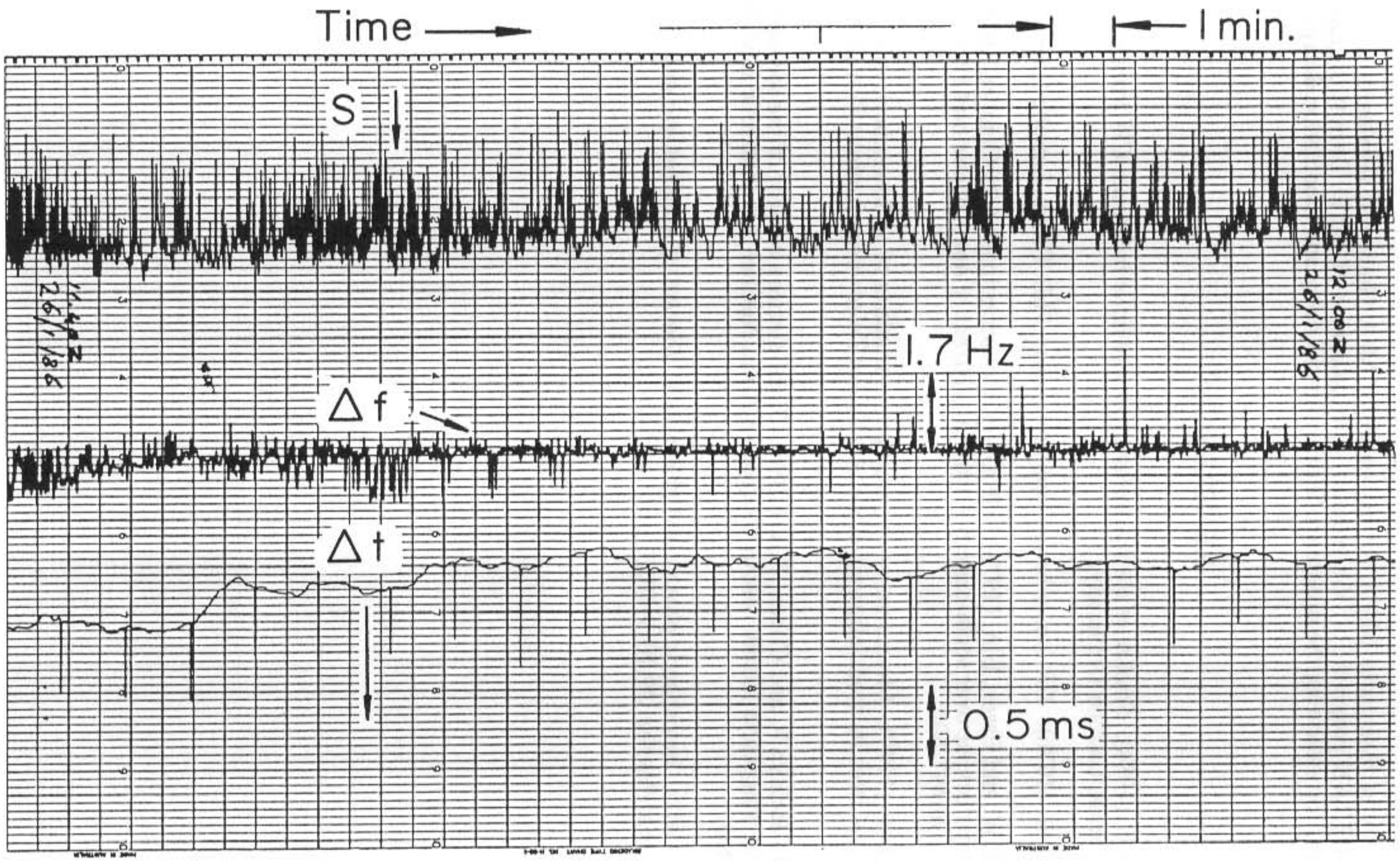


Figure 11. *A chart record showing signal strength S, Doppler shift Δf and the modulation delay Δt, with the group delay indicated once every 60 seconds.*

ACKNOWLEDGMENTS

During the stay at Macquarie Island, the assistance of Mr I. Grant was invaluable in getting the first FORTRAN programs off the ground. The assistance of Mr D. Barrett is gratefully acknowledged for indicating the difference between group delay and modulation delay.

REFERENCES

Baulch, R.N.E. (1984). *Studies of the ionosphere using oblique incidence CW transmission.* PhD Thesis, Latrobe University.

Cornelius, D.W. (1976). *HF ionospheric Doppler studies.* PhD Thesis, Latrobe University.

Cornelius, D.W. and Essex, E.A. (1978). HF Doppler observations of 23 October, 1976 total solar eclipse over south-eastern Australia. *Journal of Atmospheric and Terrestrial Physics 40*:497-502

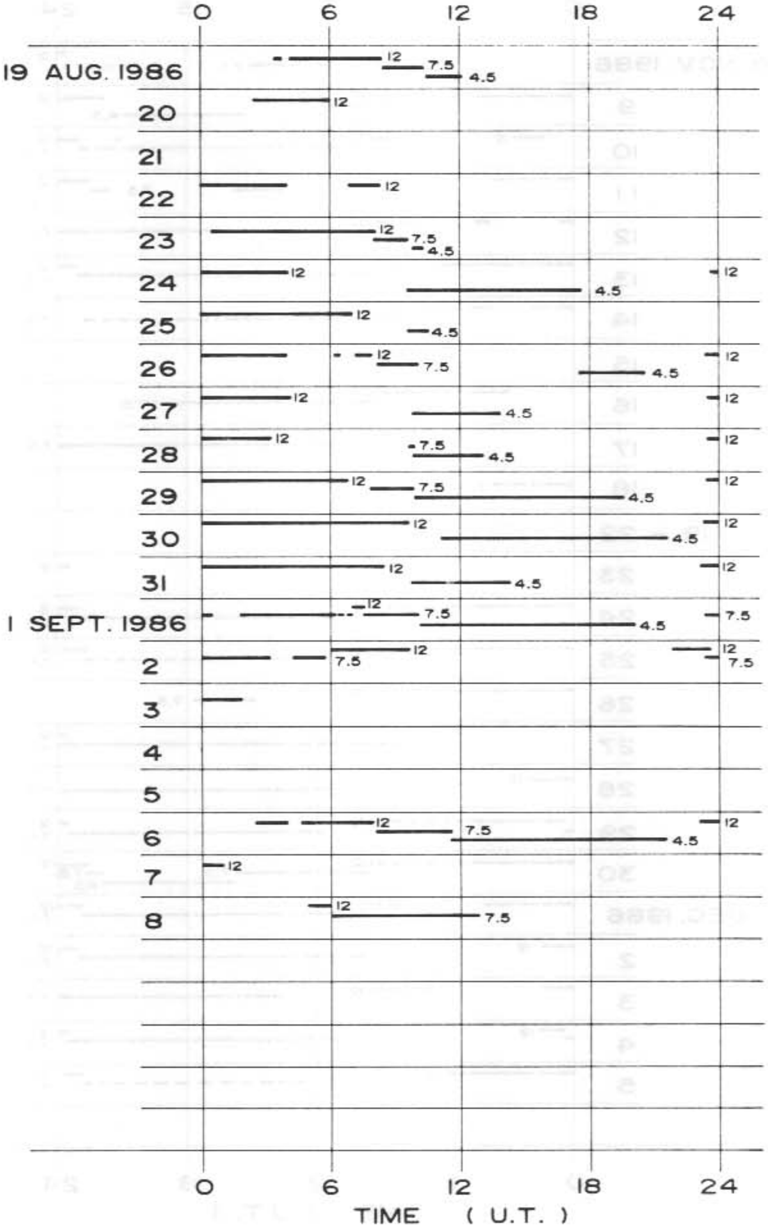
Cornelius, D.W. (1979). Measurement of group path variations of an ionospheric propagated HF signal from its modulation envelope. *Australian Telecommunications Research 13*(1):

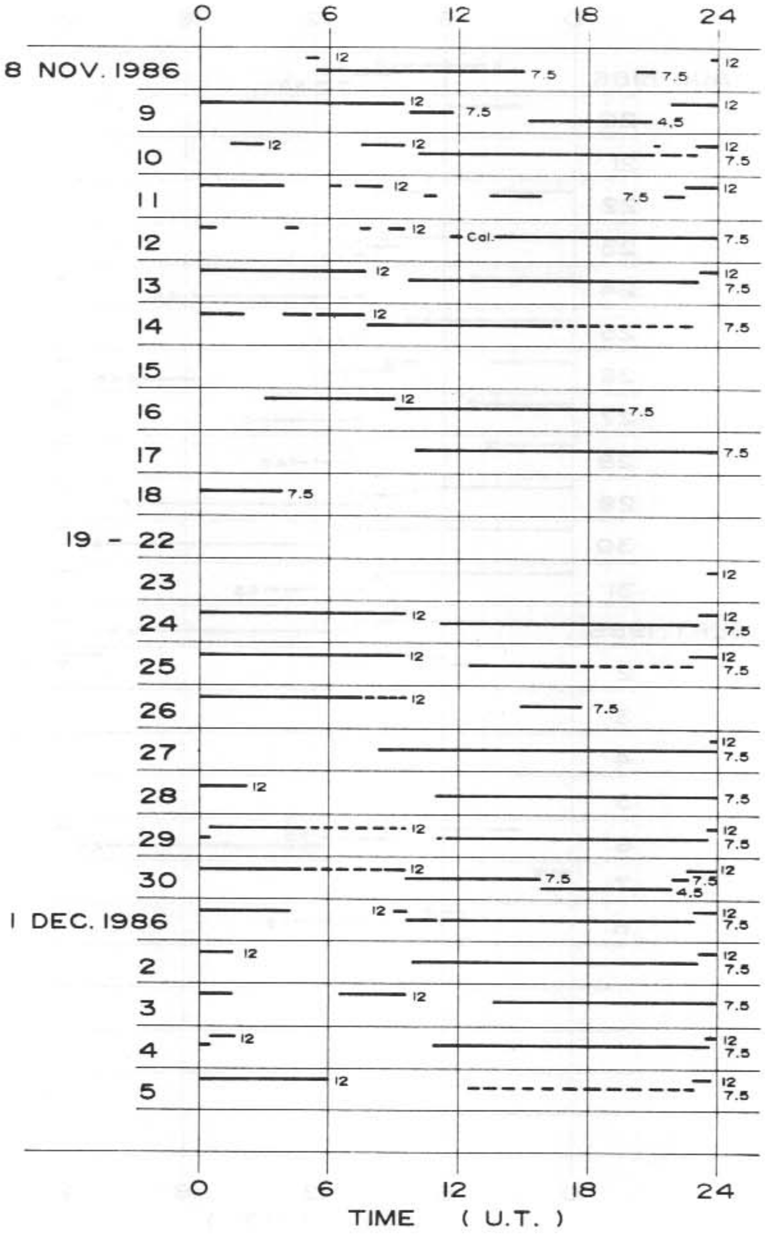
Lambert, M., Jacklyn P. and Cohen, E.A. (1986). Total electron content of the ionosphere north of Macquarie Island. *ANARE Research Notes Number 34*. Antarctic Division, Kingston.

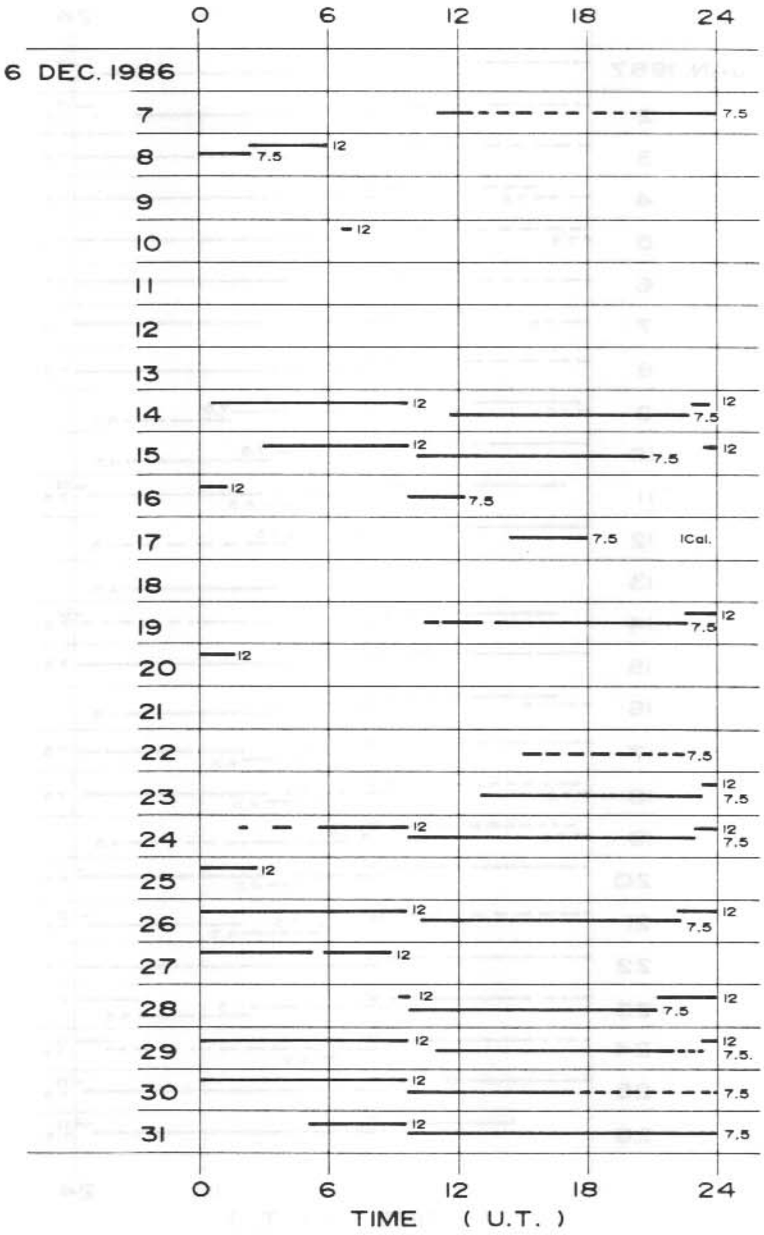
Toman, K. (1967). Ionospheric phase - and group path. *Journal of Atmospheric and Terrestrial Physics 29*:1019-1023.

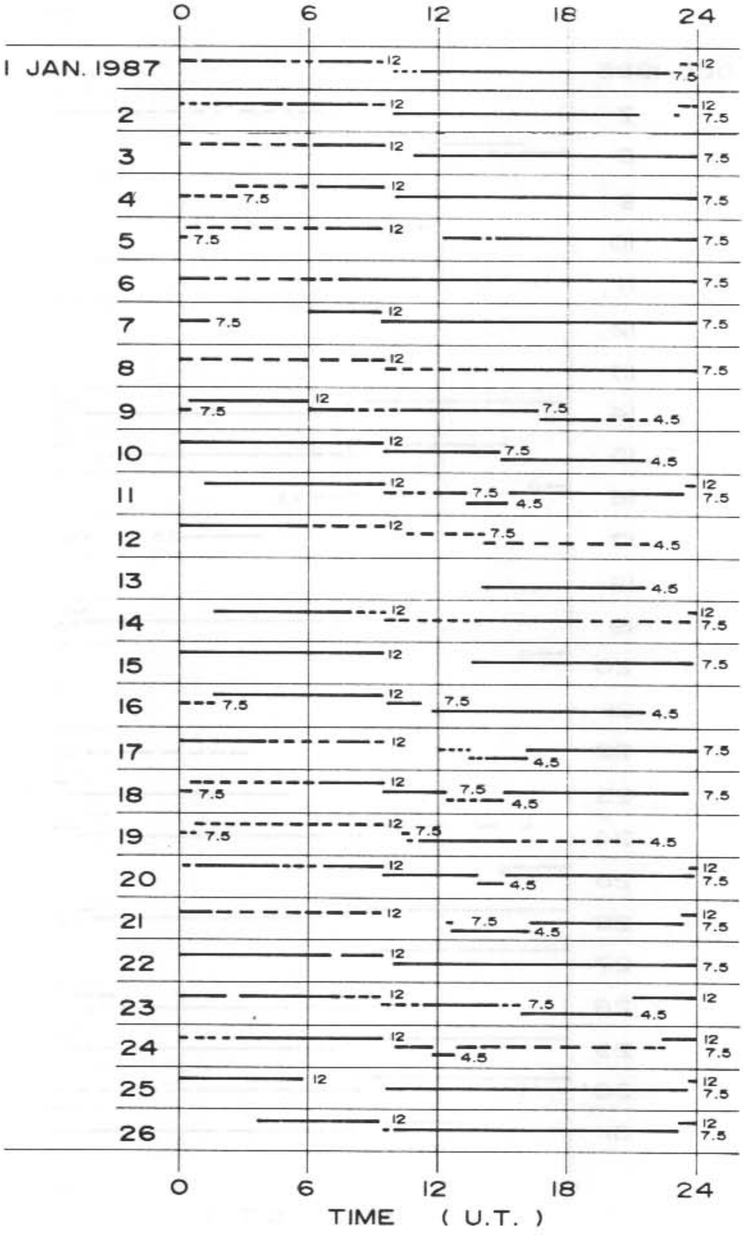
Appendix I. The VNG frequencies monitored

The frequency monitored is given in MHz by the values indicated at the end of each line indicating a recording session. From 19 August to 8 September only chart recordings were made and from 8 October digital recordings were also collected. Periods when digital records only (i.e. no charts) were obtained are indicated by the dashed lines.

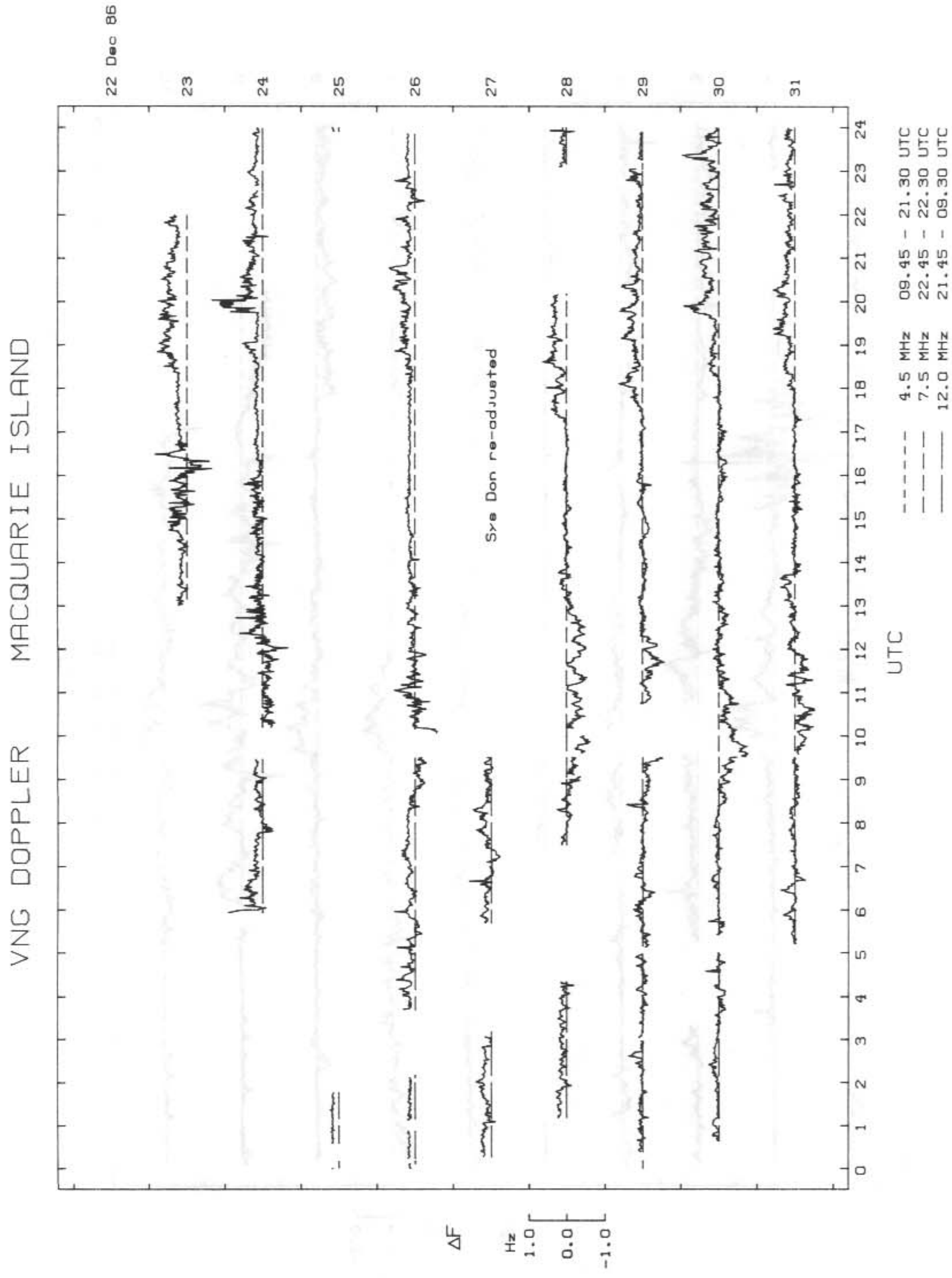


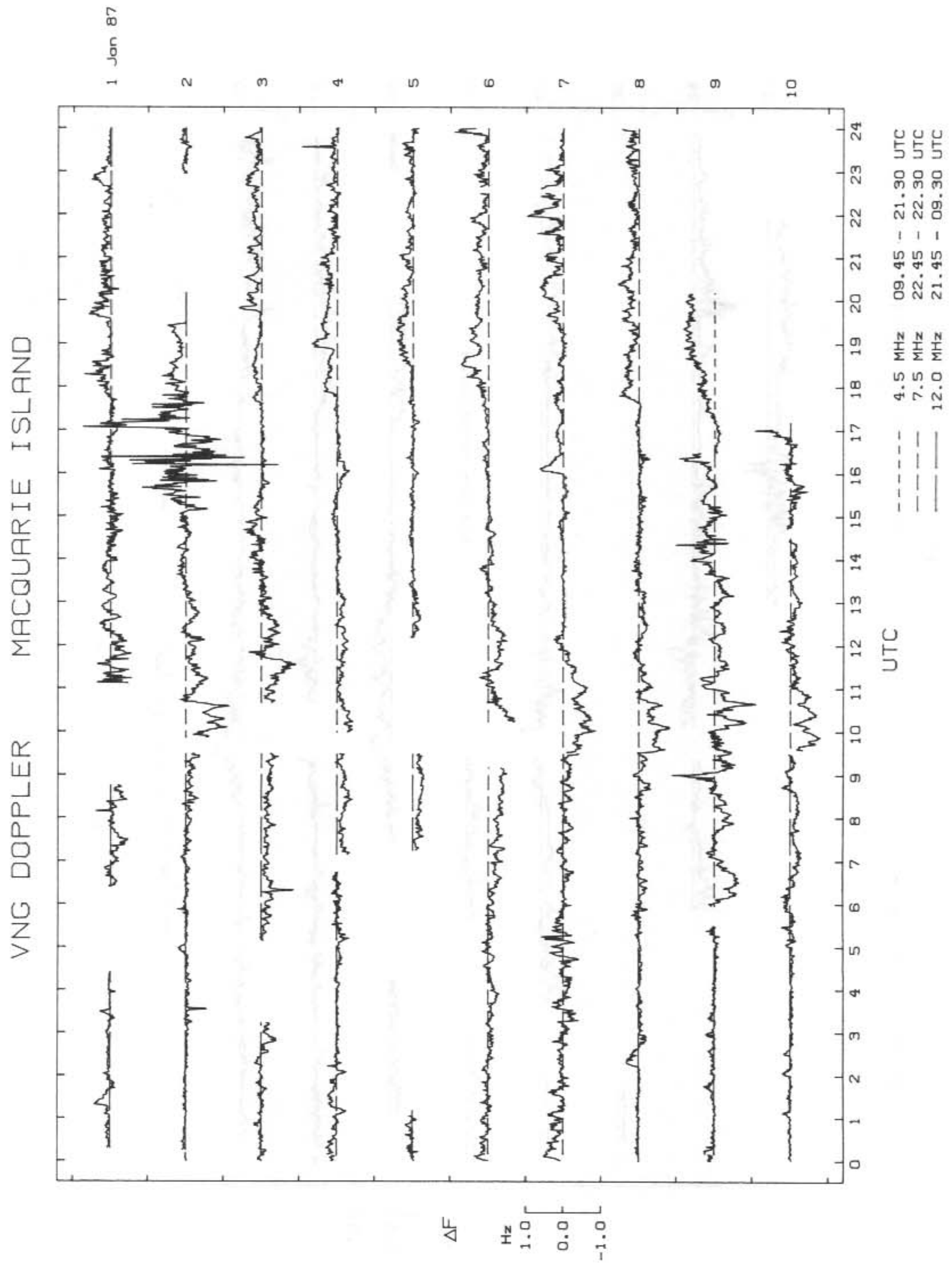

0
6
12
18
24
8 NOV. 1986
9
10
11
12
13
14
15
16
17
18
19 - 22
23
24
25
26
27
28
29
30
1 DEC. 1986
2
3
4
5
Cal.
12
7.5
4.5
TIME (U.T.)

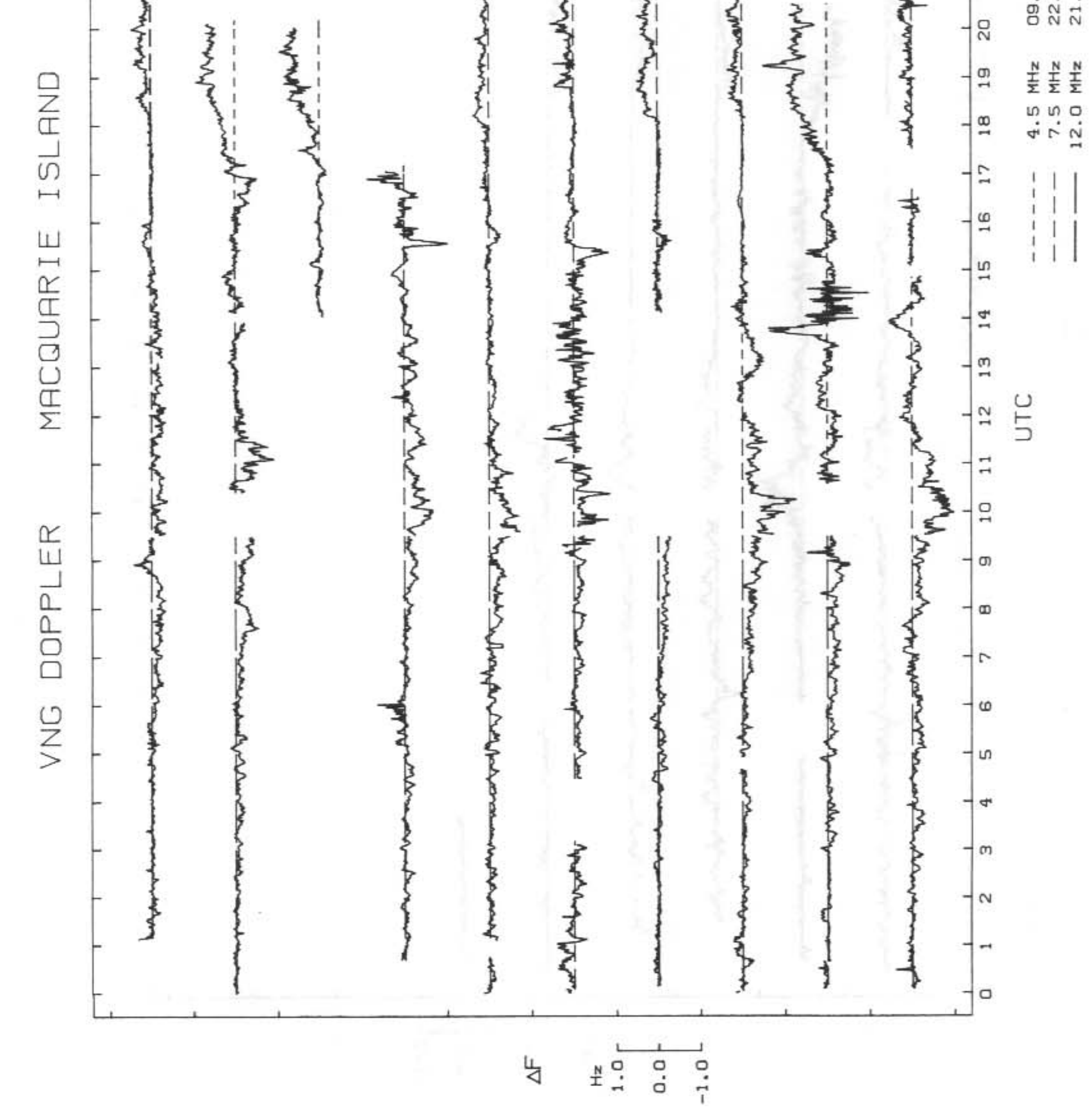

0
6
12
18
24
6 DEC. 1986
7
7.5
8
12
7.5
9
10
12
11
12
13
14
12
12
7.5
15
12
7.5
12
16
12
7.5
17
7.5
ICal.
18
19
12
7.5
20
12
21
22
7.5
23
12
7.5
24
12
12
7.5
25
12
26
12
12
7.5
27
12
28
12
12
7.5
29
12
12
7.5.
30
12
7.5
31
12
7.5
0
6
12
18
24
TIME (U.T.)

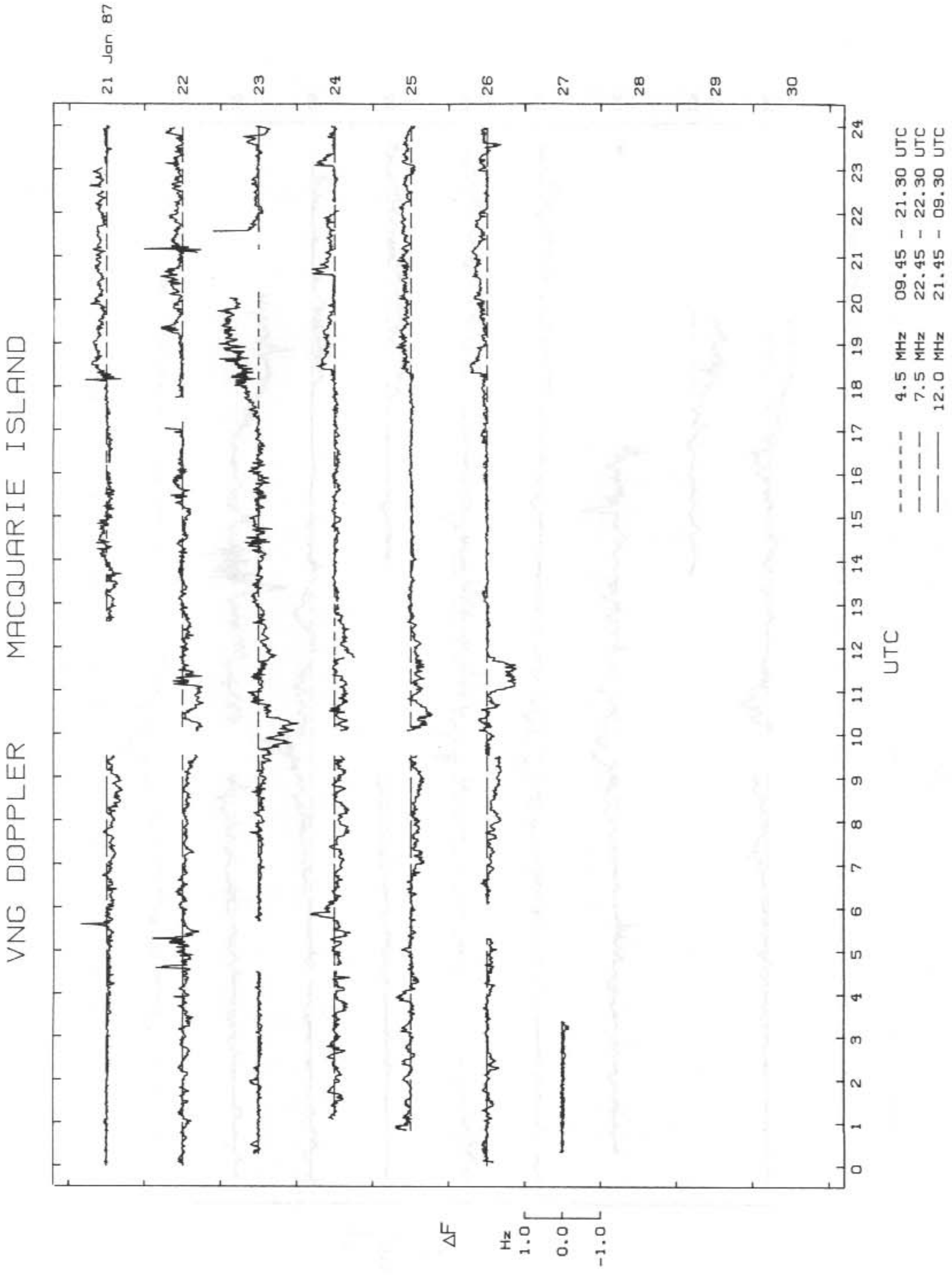
0
6
12
18
24
1 JAN. 1987
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
12
7.5
4.5
TIME (U.T.)

Appendix II. Stacked diurnal plots of the Doppler frequency shift values



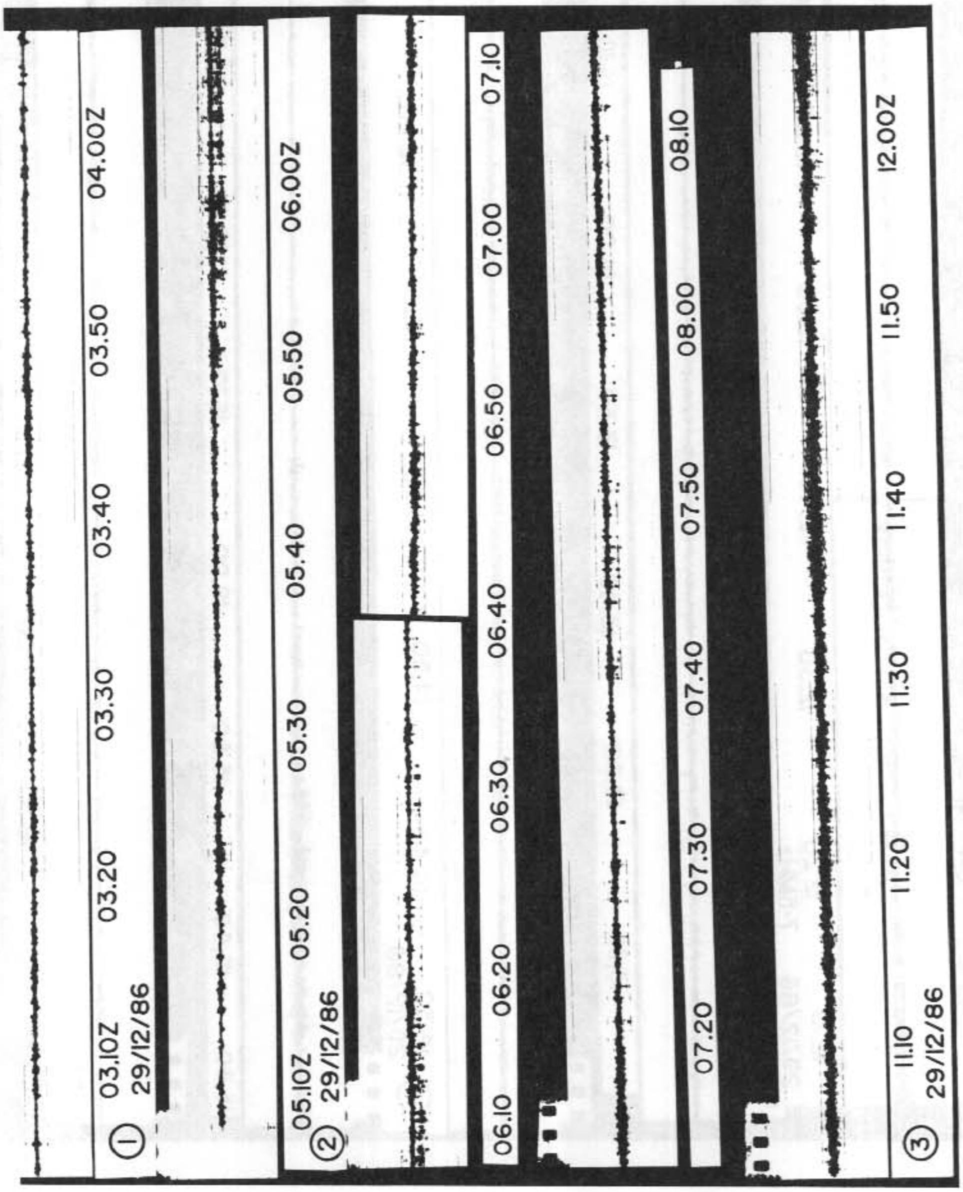

VNG DOPPLER MACQUARIE ISLAND
1 Jan 87
2
3
4
5
6
7
8
9
10
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
UTC
4.5 MHz 09.45 - 21.30 UTC
7.5 MHz 22.45 - 22.30 UTC
12.0 MHz 21.45 - 09.30 UTC
ΔF
Hz
1.0
0.0
-1.0

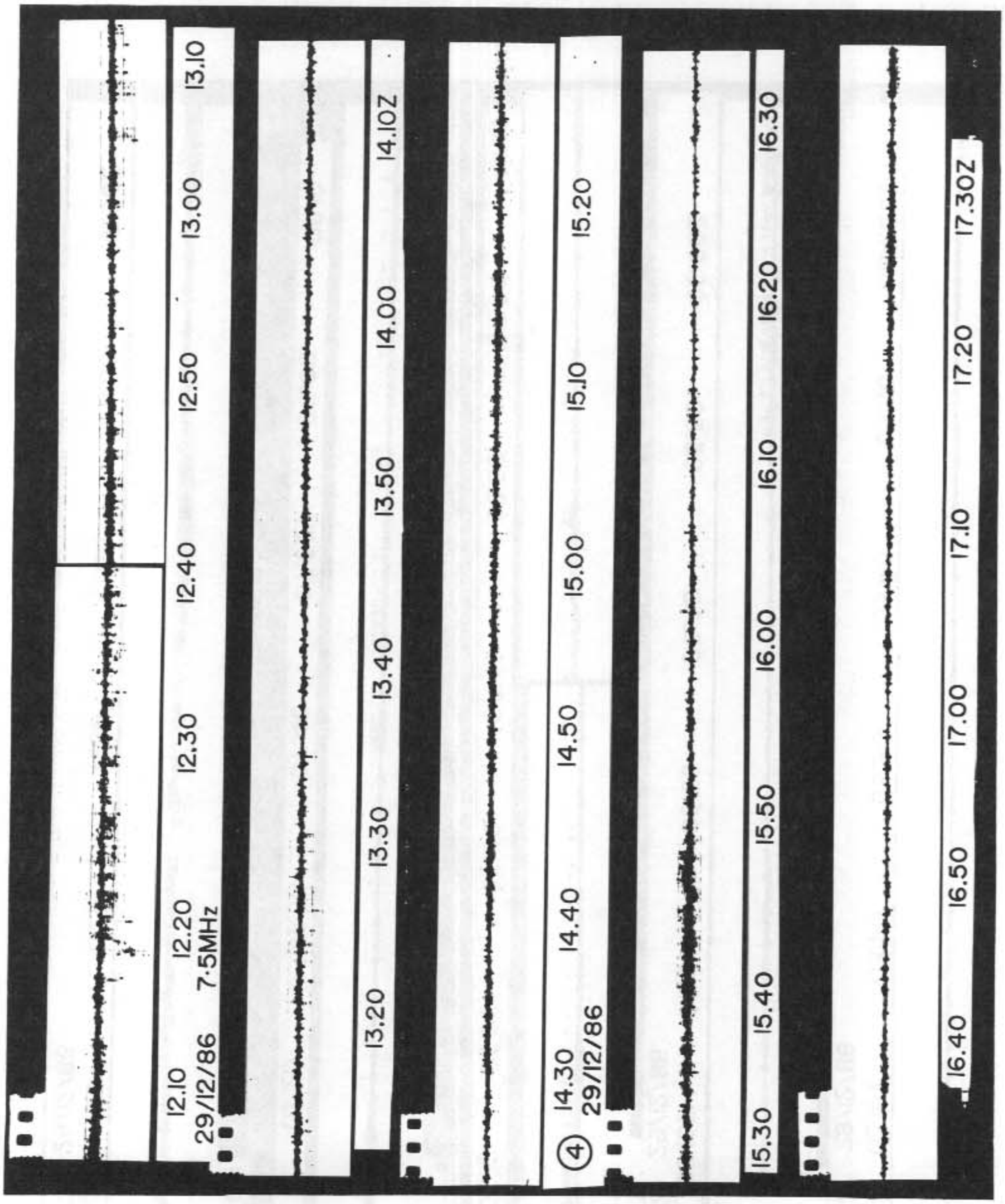

VNG DOPPLER MACQUARIE ISLAND
11 Jan 87
12
13
14
15
16
17
18
19
20
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
UTC
4.5 MHz 09.45 - 21.30 UTC
7.5 MHz 22.45 - 22.30 UTC
12.0 MHz 21.45 - 09.30 UTC
ΔF
Hz
1.0
0.0
-1.0

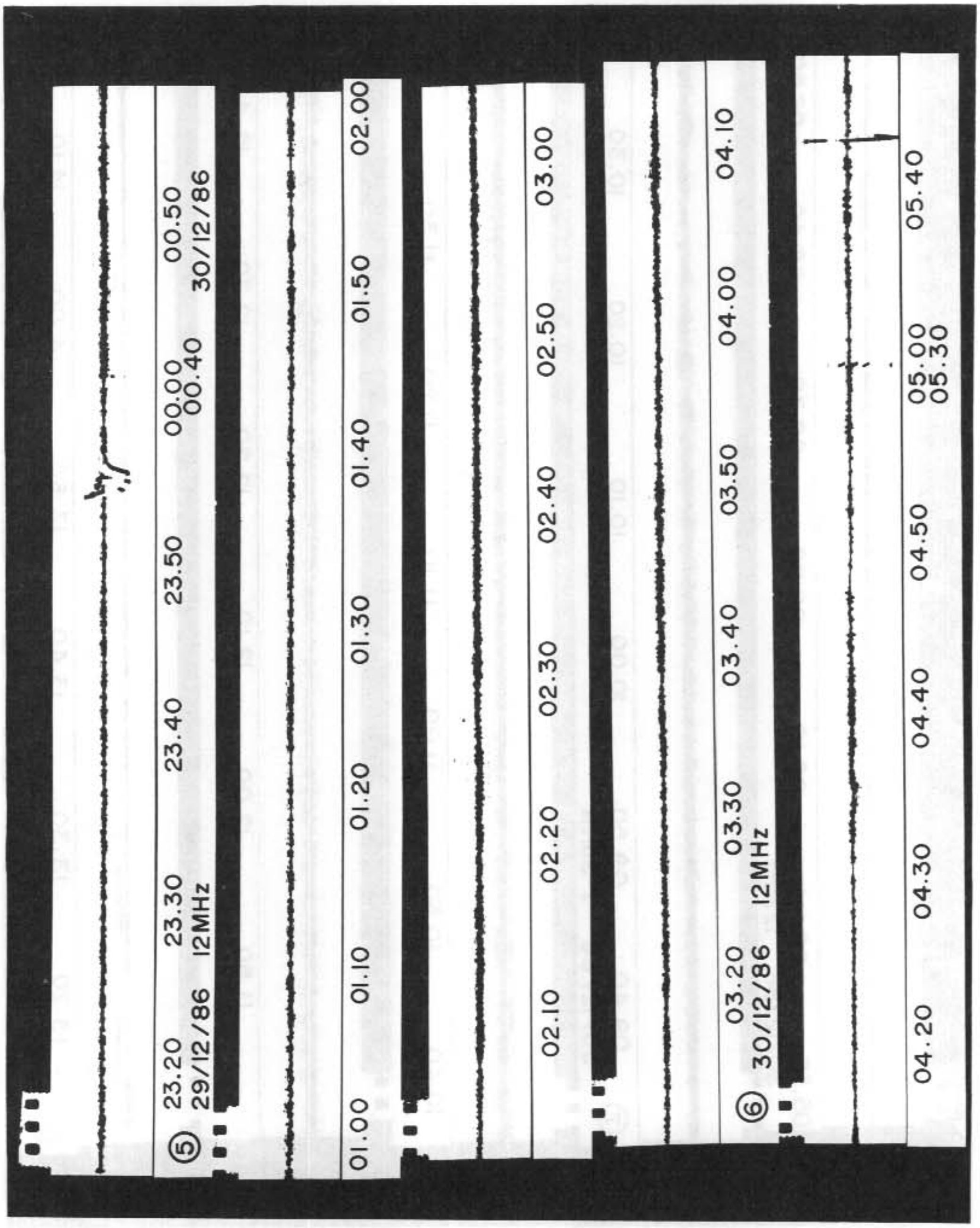

VNG DOPPLER MACQUARIE ISLAND
21 Jan 87
22
23
24
25
26
27
28
29
30
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
UTC
4.5 MHz 09.45 - 21.30 UTC
7.5 MHz 22.45 - 22.30 UTC
12.0 MHz 21.45 - 09.30 UTC
ΔF
Hz
1.0
0.0
-1.0

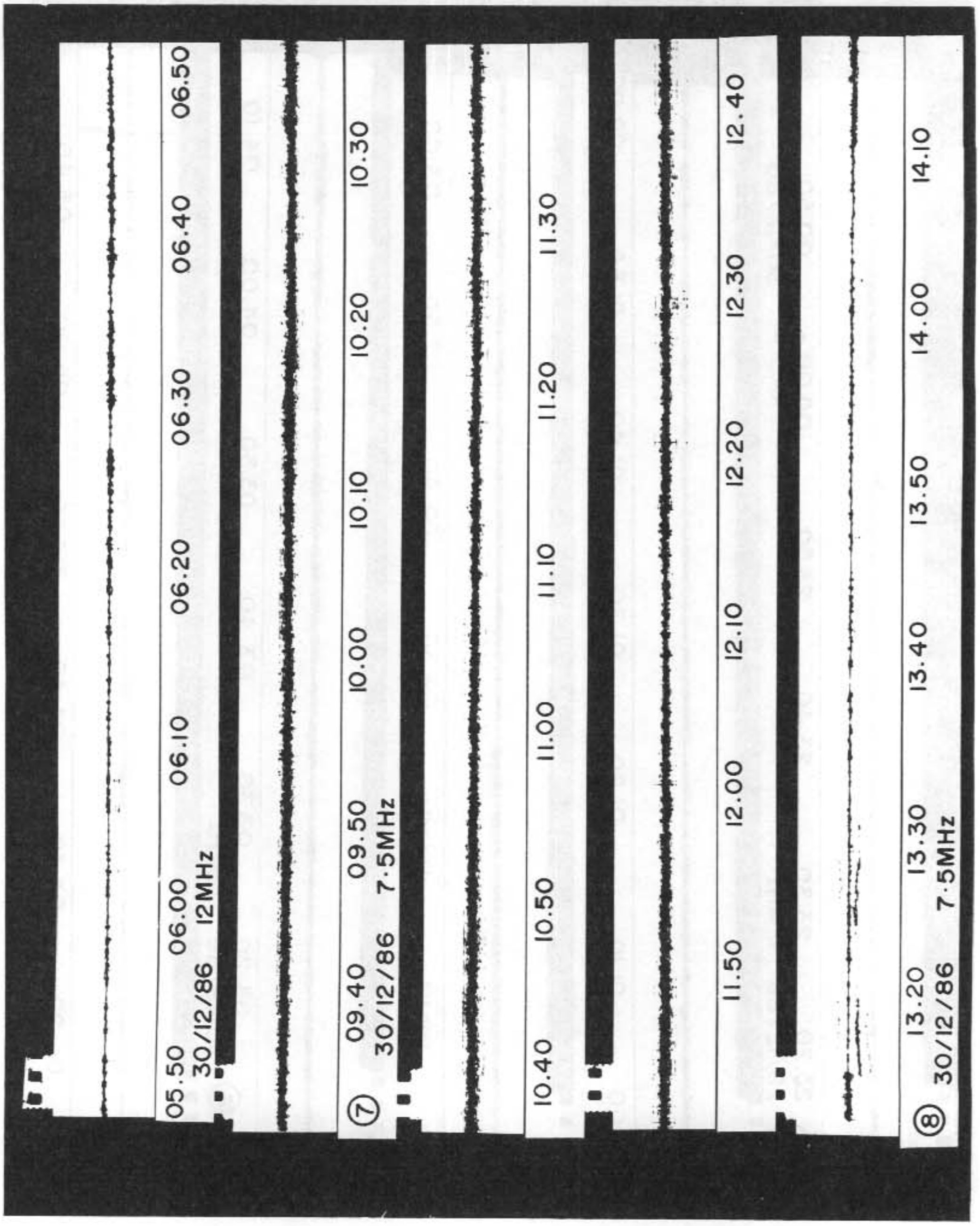
Appendix III. Frequency spectrum analysed Doppler shifts

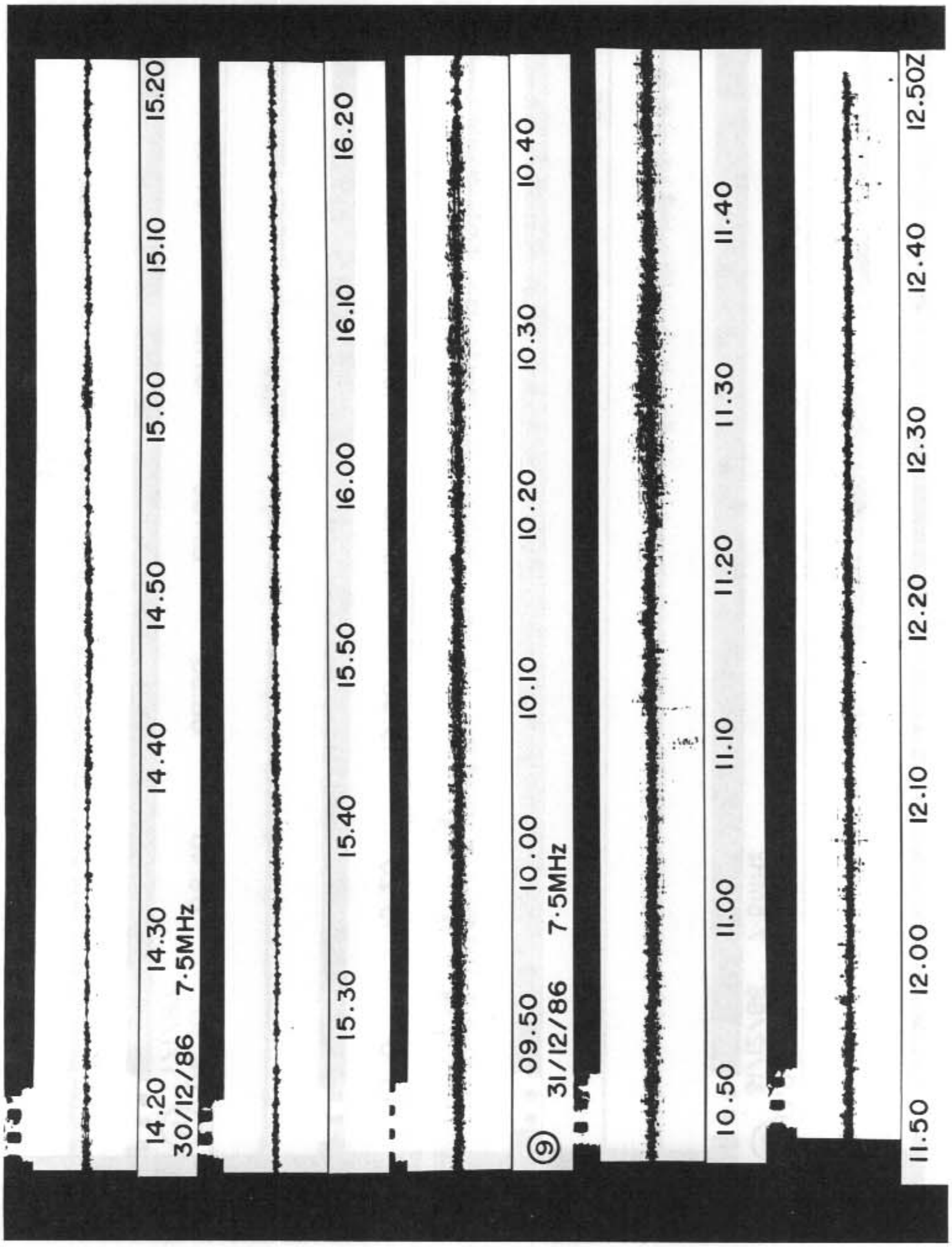
2Hz scale marks are shown every alternate 10 minutes.

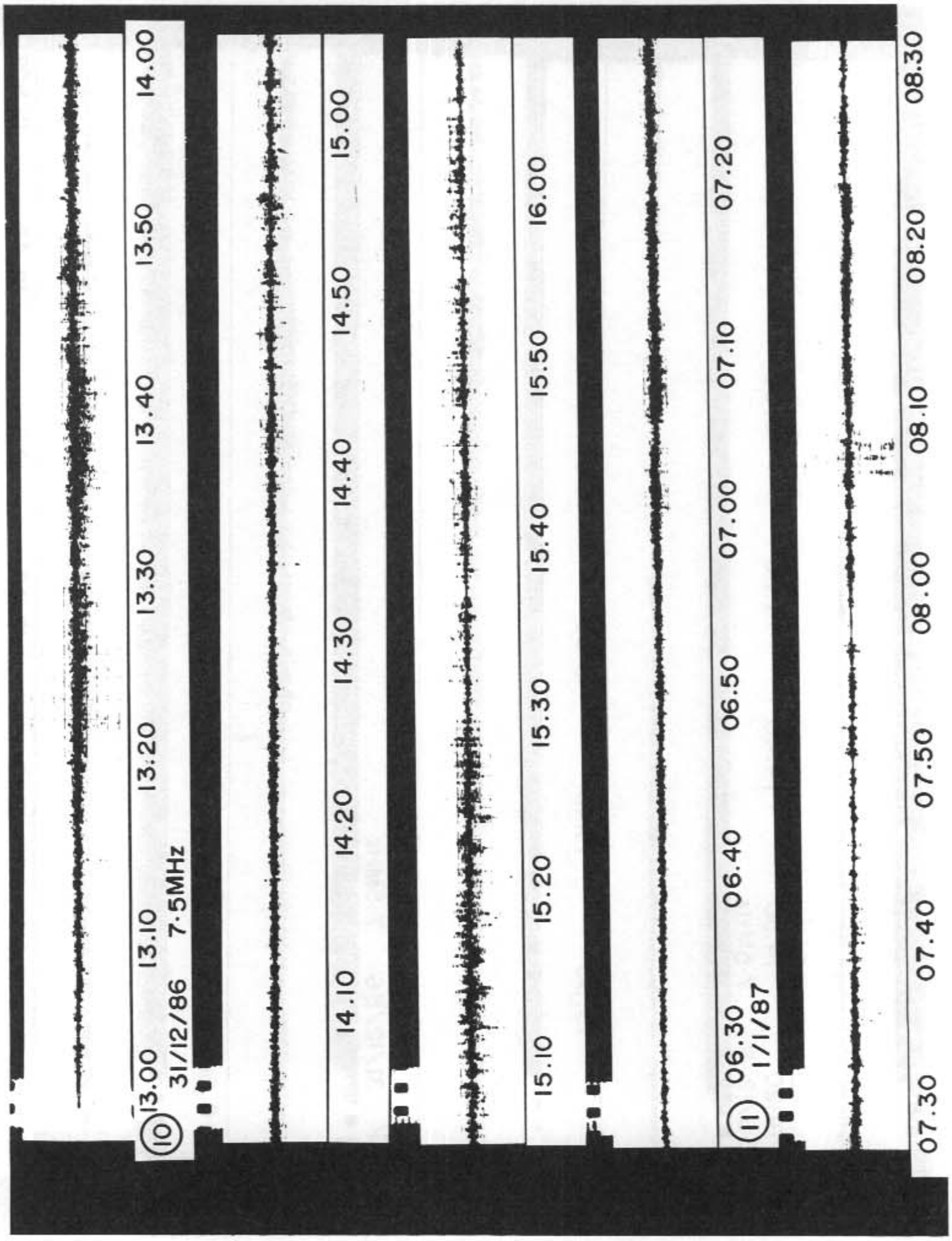



12.10
29/12/86
12.20
7·5MHz
12.30
12.40
12.50
13.00
13.10
13.20
13.30
13.40
13.50
14.00
14.10Z
④
14.30
29/12/86
14.40
14.50
15.00
15.10
15.20
15.30
15.40
15.50
16.00
16.10
16.20
16.30
16.40
16.50
17.00
17.10
17.20
17.30Z

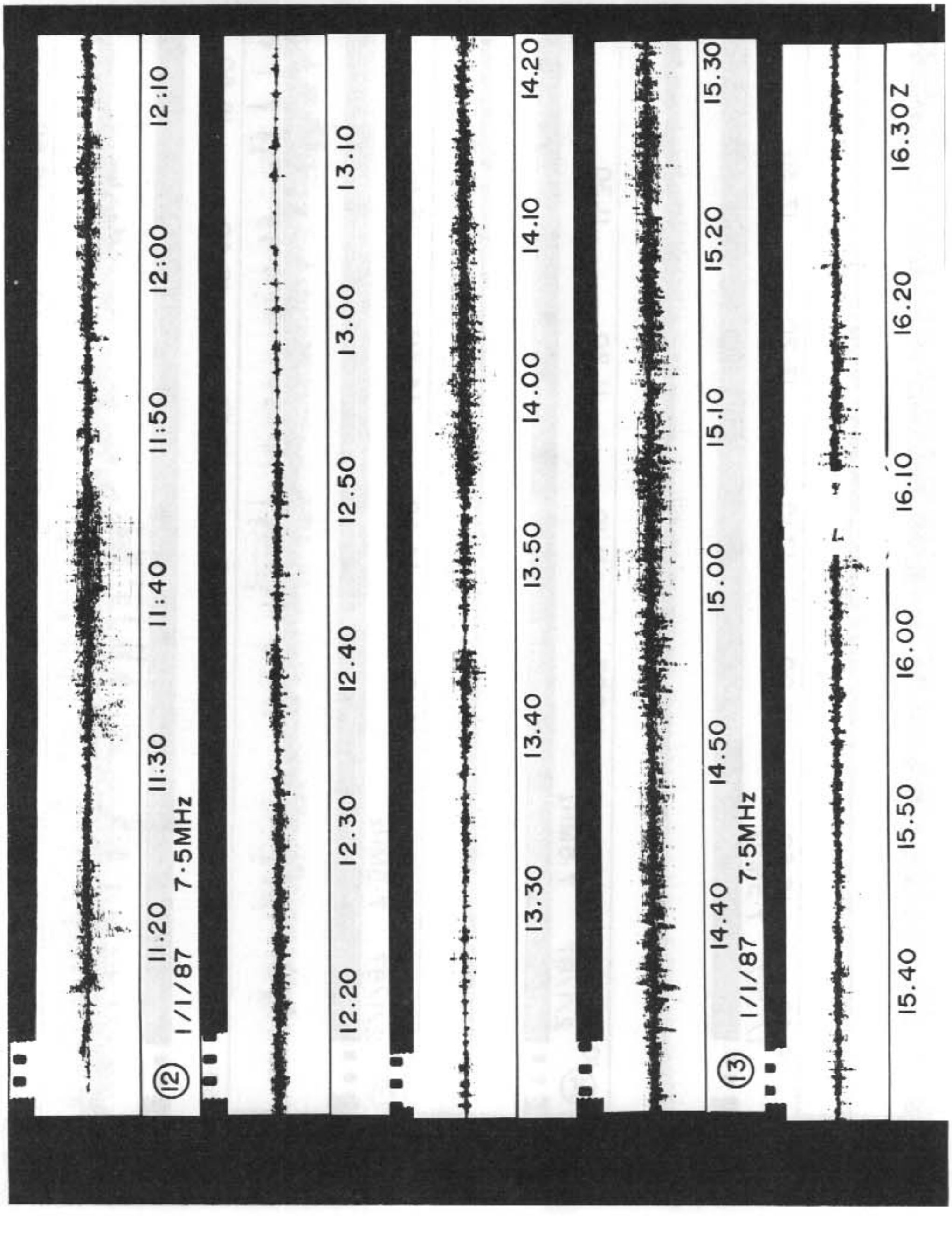

⑤
23.20
29/12/86
23.30
12MHz
23.40
23.50
00.00
00.40
00.50
30/12/86
01.00
01.10
01.20
01.30
01.40
01.50
02.00
02.10
02.20
02.30
02.40
02.50
03.00
⑥
03.20
30/12/86
03.30
12MHz
03.40
03.50
04.00
04.10
04.20
04.30
04.40
04.50
05.00
05.30
05.40

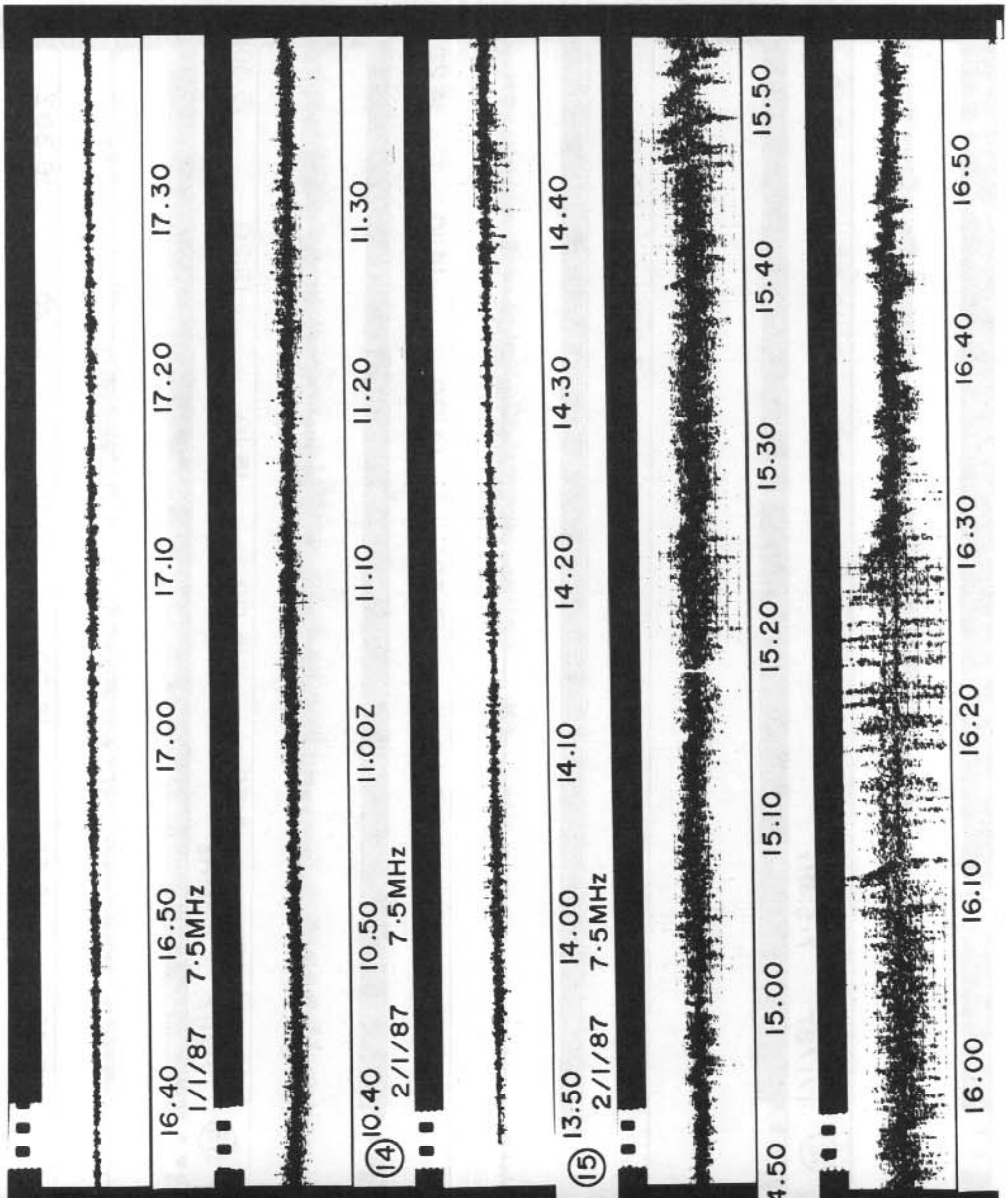

05.50
30/12/86
06.00
12MHz
06.10
06.20
06.30
06.40
06.50
⑦
09.40
30/12/86
09.50
7·5MHz
10.00
10.10
10.20
10.30
10.40
10.50
11.00
11.10
11.20
11.30
11.50
12.00
12.10
12.20
12.30
12.40
⑧
13.20
30/12/86
13.30
7·5MHz
13.40
13.50
14.00
14.10

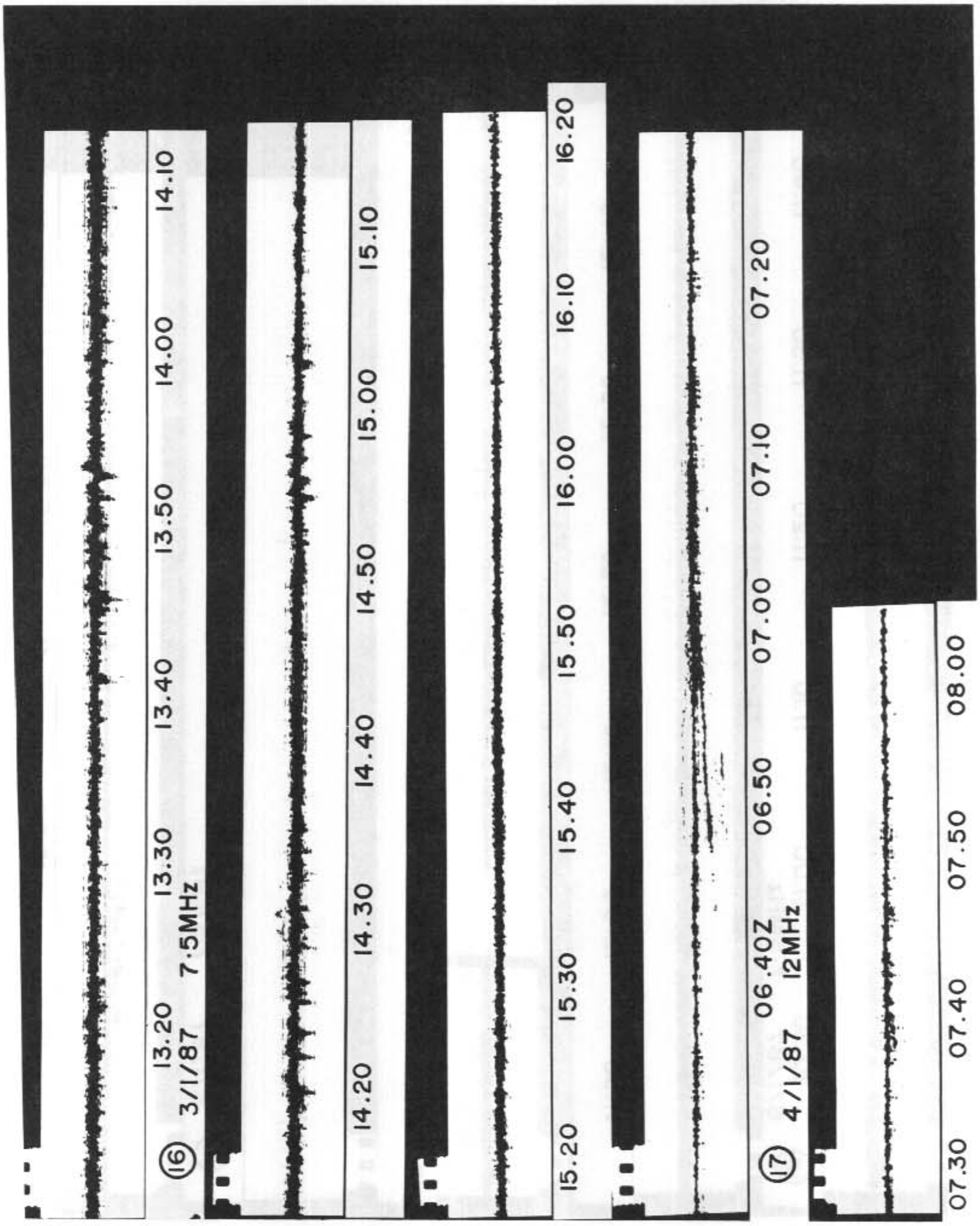

14.20
30/12/86
14.30
7·5MHz
14.40
14.50
15.00
15.10
15.20
15.30
15.40
15.50
16.00
16.10
16.20
⑨
09.50
31/12/86
10.00
7·5MHz
10.10
10.20
10.30
10.40
10.50
11.00
11.10
11.20
11.30
11.40
11.50
12.00
12.10
12.20
12.30
12.40
12.50Z

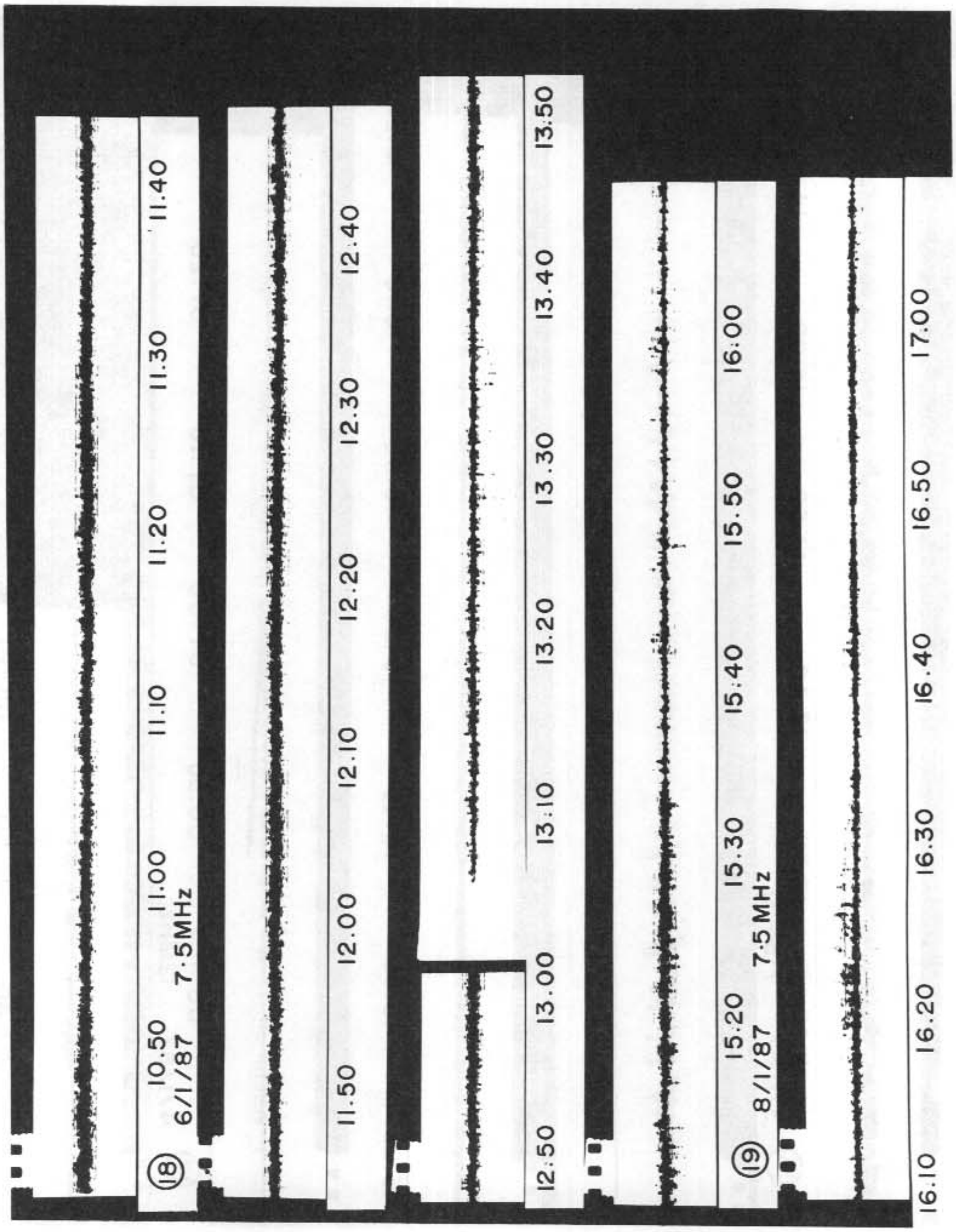


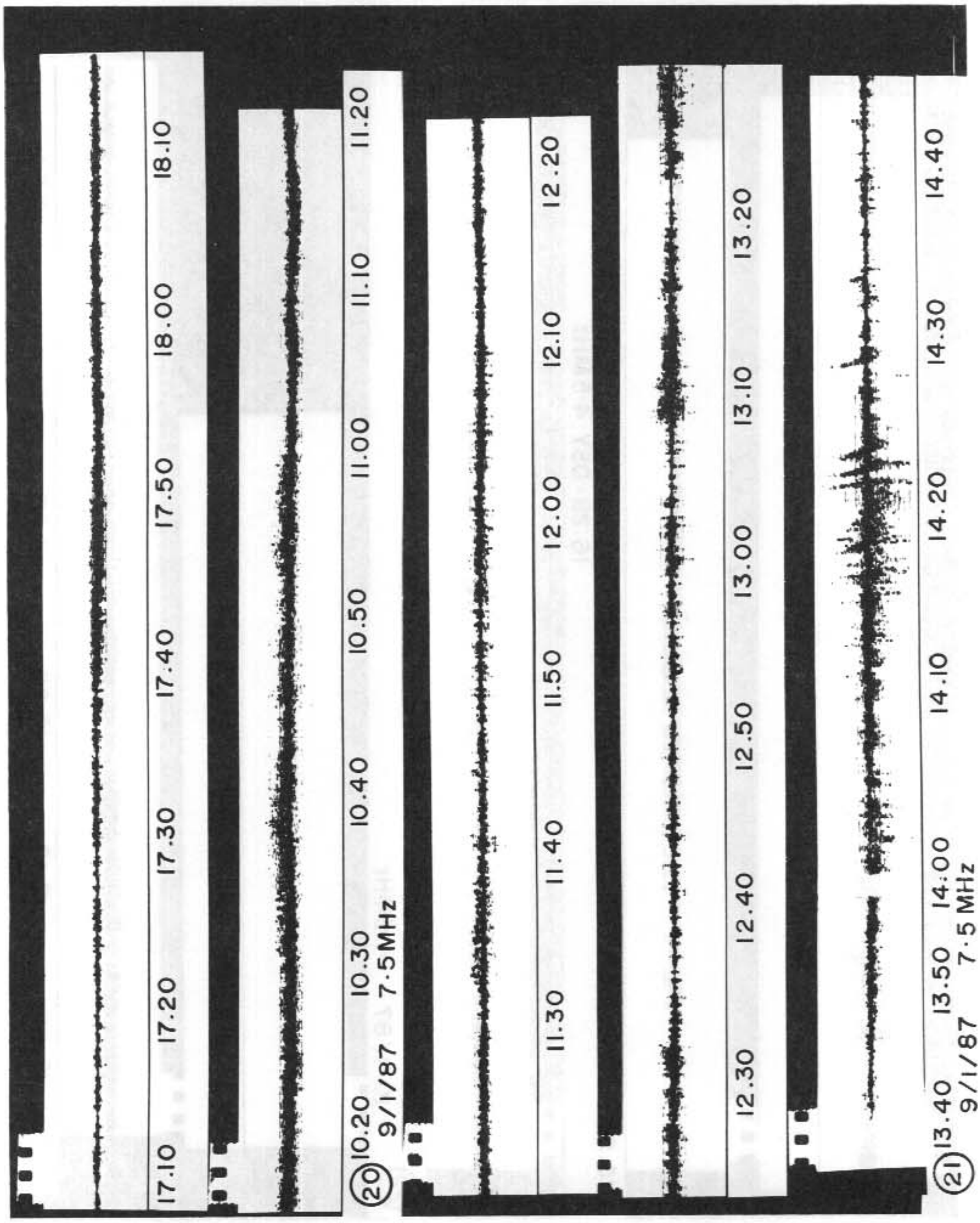
⑩ 13.00 31/12/86
13.10 7·5MHz
13.20
13.30
13.40
13.50
14.00
14.10
14.20
14.30
14.40
14.50
15.00
15.10
15.20
15.30
15.40
15.50
16.00
⑪ 06.30 1/1/87
06.40
06.50
07.00
07.10
07.20
07.30
07.40
07.50
08.00
08.10
08.20
08.30


⑫ 1/1/87 7·5MHz
11.20
11:30
11:40
11:50
12:00
12:10
12.20
12.30
12.40
12.50
13.00
13.10
13.30
13.40
13.50
14.00
14.10
14.20
⑬ 1/1/87 7·5MHz
14.40
14.50
15.00
15.10
15.20
15.30
15.40
15.50
16.00
16.10
16.20
16.30Z

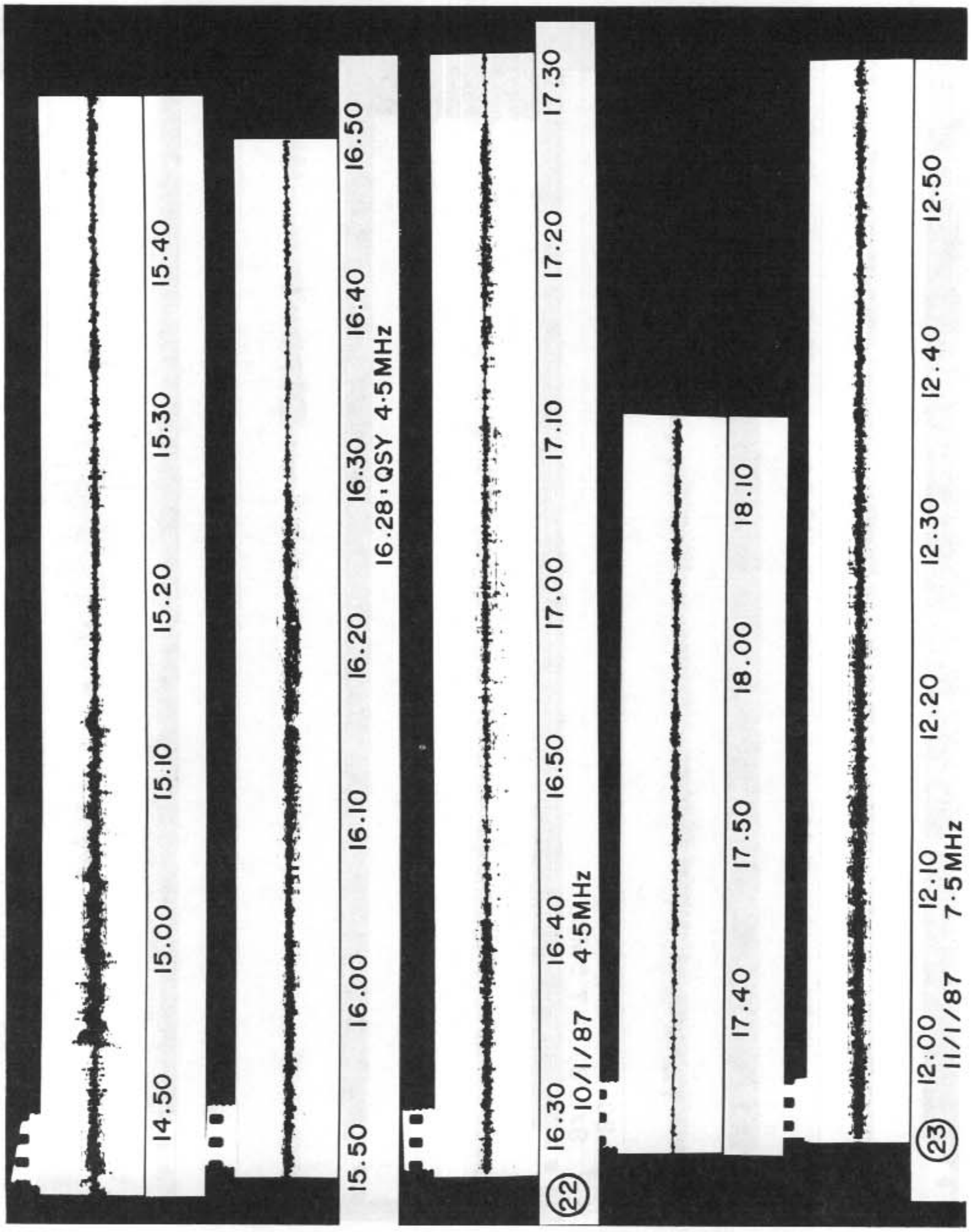

16.40
1/1/87
16.50
7·5MHz
17.00
17.10
17.20
17.30
⑭ 10.40
2/1/87
10.50
7·5MHz
11.00Z
11.10
11.20
11.30
⑮ 13.50
2/1/87
14.00
7·5MHz
14.10
14.20
14.30
14.40
14.50
15.00
15.10
15.20
15.30
15.40
15.50
16.00
16.10
16.20
16.30
16.40
16.50

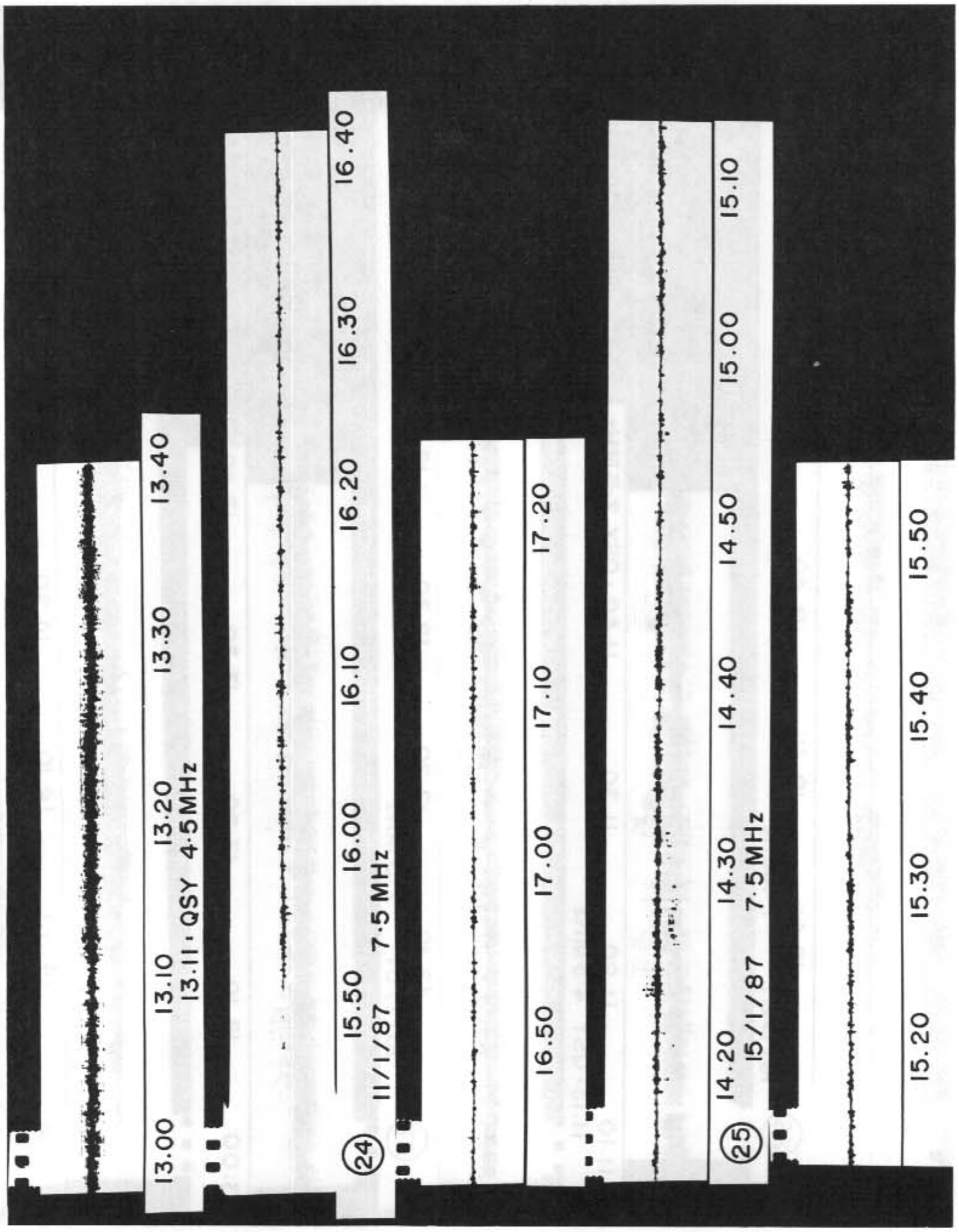

⑯
13.20
3/1/87
13.30
7·5MHz
13.40
13.50
14.00
14.10
14.20
14.30
14.40
14.50
15.00
15.10
15.20
15.30
15.40
15.50
16.00
16.10
16.20
⑰
4/1/87
06.40Z
12MHz
06.50
07.00
07.10
07.20
07.30
07.40
07.50
08.00

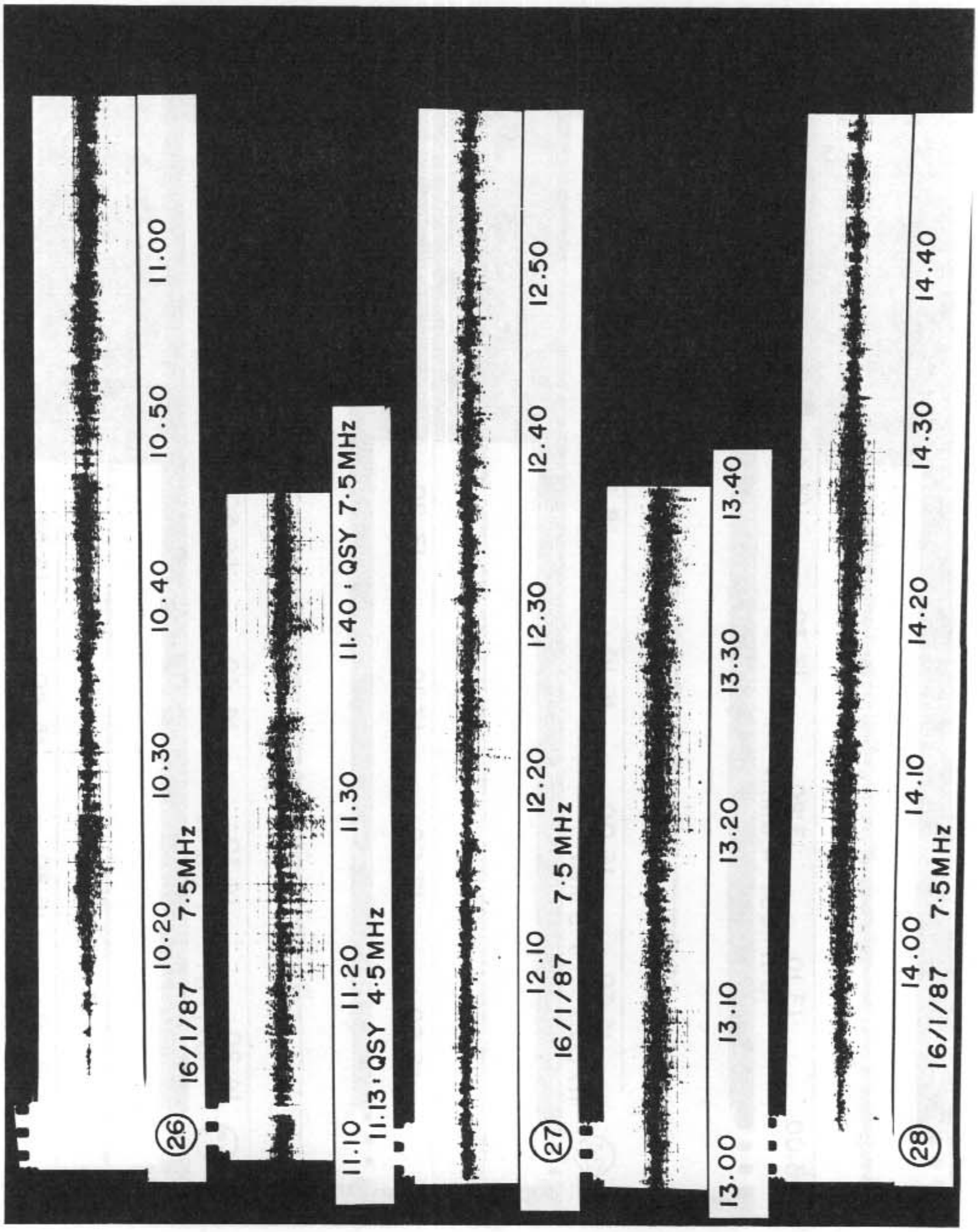

⑱
6/1/87
10.50
7·5MHz
11.00
11.10
11.20
11.30
11.40
11.50
12.00
12.10
12.20
12.30
12.40
12:50
13.00
13:10
13.20
13.30
13.40
13.50
⑲
8/1/87
15:20
7·5MHz
15.30
15:40
15.50
16:00
16.10
16.20
16.30
16.40
16.50
17.00



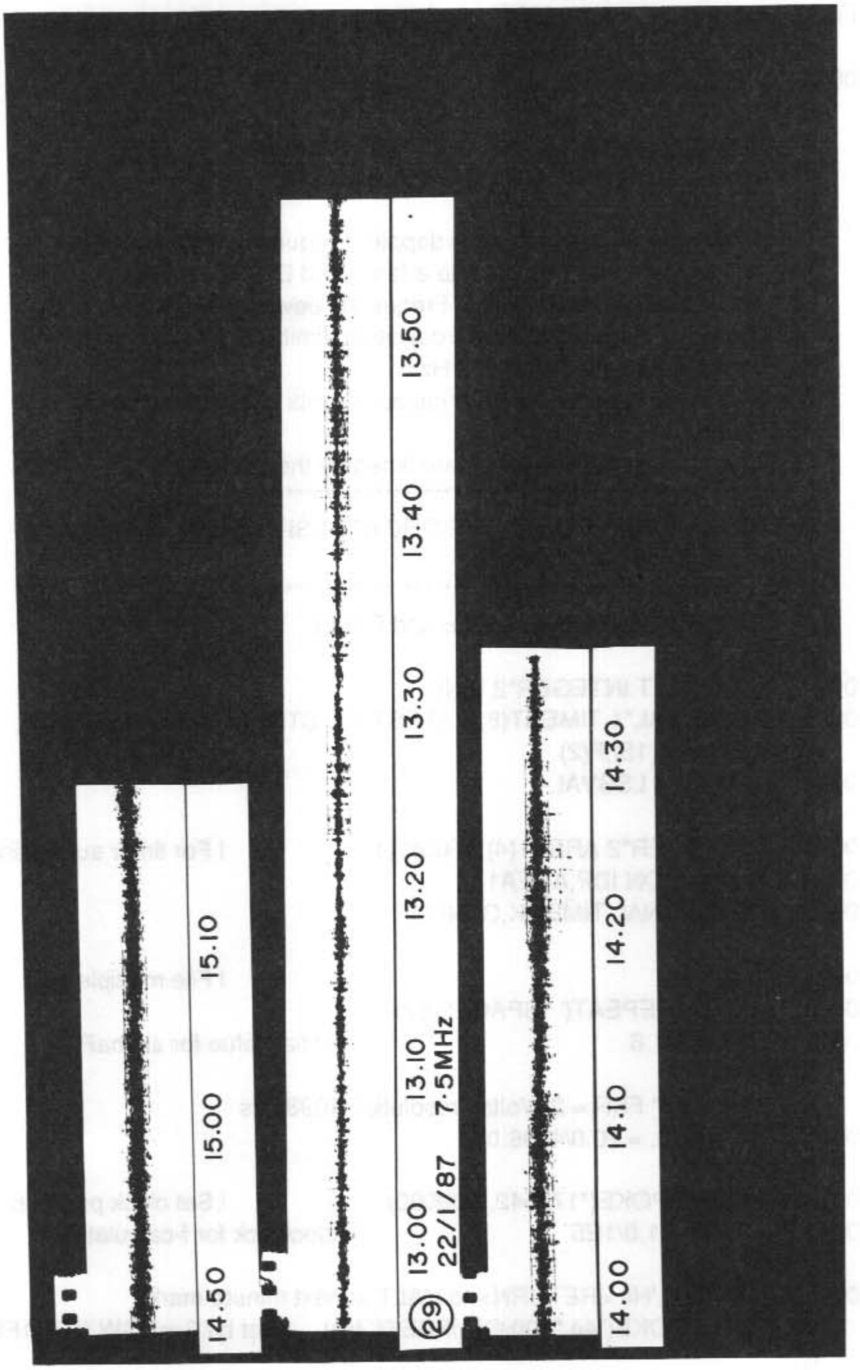
17.10
17.20
17.30
17.40
17.50
18.00
18.10
⑳ 10.20 9/1/87
10.30 7·5 MHz
10.40
10.50
11.00
11.10
11.20
11.30
11.40
11.50
12.00
12.10
12.20
12.30
12.40
12.50
13.00
13.10
13.20
㉑ 13.40 9/1/87
13.50 7·5 MHz
14.00
14.10
14.20
14.30
14.40


14.50
15.00
15.10
15.20
15.30
15.40
15.50
16.00
16.10
16.20
16.30
16.40
16.50
16.28 : QSY 4·5MHz
㉒ 16.30
10/1/87
16.40
4·5MHz
16.50
17.00
17.10
17.20
17.30
17.40
17.50
18.00
18.10
㉓ 12:00
11/1/87
12.10
7·5MHz
12.20
12.30
12.40
12.50


13.00
13.10
13.11 · QSY 4·5MHz
13.20
13.30
13.40
(24)
11/1/87
15.50
7·5MHz
16.00
16.10
16.20
16.30
16.40
16.50
17.00
17.10
17.20
(25)
15/1/87
14.20
14.30
7·5MHz
14.40
14.50
15.00
15.10
15.20
15.30
15.40
15.50



㉖
16/1/87
10.20
7·5MHz
10.30
10.40
10.50
11.00
11.10
11.13·QSY 4·5MHz
11.20
11.30
11.40·QSY 7·5MHz
㉗
16/1/87
12.10
7·5 MHz
12.20
12.30
12.40
12.50
13.00
13.10
13.20
13.30
13.40
㉘
16/1/87
14.00
7·5MHz
14.10
14.20
14.30
14.40


14.50
15.00
15.10
(29)
13.00
22/1/87
13.10
7·5MHz
13.20
13.30
13.40
13.50
14.00
14.10
14.20
14.30

Appendix IV. FORTRAN program TIMFSD for determining Doppler frequency shift values

```
FORTRAN IV          V02.1-1   Sun 28-Dec-86 07:22:56     PAGE 001

0001      PROGRAM TIMFSD
   C
   C   Sjoerd Jongens .... December 1986 onwards
   C   Modified from VNGDAT of Ian Grant, Oct '86
   C
   C   This program measures the doppler Frequency shift and Signal
    C   strength and prints/plots it to a formatted Disk file every minute,
    C   as well as hourly statistics. Frequency deviation limits are set by
    C   previous minute's standard deviation (limit = 5*sigmaF); with
    C   an absolute maximum of 8 Hz.
   C   The clock used for period meausurements is re-started every
    C   cycle.
   C   The minute and hour stats are timed by the LTC.
   C   *****************************************************
   C   THIS VERSION USES THE GRC 600-LSI-11(1030) INSTEAD OF
          DATATRANS DT2758
   C   ******************************************************
   C   To be LINKed with SAMPLE and DTLIB
   C
0002      IMPLICIT INTEGER*2 (I,N)
0003      LOGICAL*1 TIMEST(8),DATEST(10),STRING(84),SPACES(84),
     #  FILNAM(15),F(2)
0004      REAL*4 LSBVAL
   C
0005      INTEGER*2 AREA1(4),AREA2(4)              ! For timer subroutines
0006      COMMON IDP,AREA1
0007      EXTERNAL TIMEMK,GTNAME
   C
0008      IFI = 0                                  ! File multiple tally
0009      CALL REPEAT(' ',SPACES,82)
0010      SDF = 1.6                          ! Initial value for sigmaF
   C
   C        "1030" FSR = 20Volts , resolution 4096 bits
0011      LSBVAL = 20.0/4096.0
   C
0012      CALL IPOKE("172542,"000000)              ! Set clock preset buffer
0013      TICK = 1.0/1E5                     ! clock tick for f-calculations
   C
0014      TYPE *,'Hit <RETURN> to HALT at next minute mark.'
   C   CALL IPOKE("44,"100.OR.IPEEK("44))   ! Set bit 6 in JSW (RT11FB)
   C
```

```
C     Every hour:
C
90    CALL GTNAME(FILNAM)                          ! Obtain new filename
      IF (IERR.EQ.0) GOTO 210
      IF (IERR.EQ.1) GOTO 900
      IF (IERR.EQ.2) GOTO 901
      STOP 'Filename error test failed'
C
C     Create new file:
C
210   OPEN(UNIT=2,NAME=FILNAM,TYPE='NEW',ERR=902)
C     ICHAN = ILUN(2)                              ! Obtain channel#
C
C     Notify the operator:
C
      WRITE (5,650) (FILNAM(K),K=1,14),ICHAN
650   FORMAT(1H ,'Writing to: ',14A1,' on chan: ',I2)
C
      CALL DATE(DATEST)
      WRITE (2,630,ERR=903) (DATEST(K),K=1,9)          ! File the header
630   FORMAT (X,' Av.d-F SigmaF Aver.S SigmaS    ',9A1,
     #'  |<-(-2)-Av.dF-(+2)->|    0<------SigmF--(2)->|',
     # '   0<--AvS--9---->|   0<-SigmS->|')
      IU = 0
      IL = 0
      IH = 0
      HTDEF = 0
      HSIGM = 0
      HTS = 0
      HSIGS = 0
C
      CALL ITIMER(1,0,0,0,AREA2,3,TIMEMK)          ! Set up for 1 hour
C
C     Every minute:
C
100   TOTDEF = 0
      SIGMA = 0
      TOTS = 0
      SIGS = 0
      IM = 0
      SDF = AMIN1(SDF,1.6)                         ! limit to +/- 8 Hz
      PERLO = 1.0/(10.0-5.0*SDF)                   ! period for low f limit
      PERHI = 1.0/(10.0+5.0*SDF)                   ! period for high f limit
```

```
C
      CALL ITIMER(0,1,0,0,AREA1,2,TIMEMK)      ! Set up for 1 minute
C
C     Every second period:
C
C     Wait for leading edge of REQA bit in DRV11 status register
C
200   IF (IPEEK("167770).AND."000200) GOTO 200                ! Wait 0
300   IF (.NOT.(IPEEK("167770).AND."000200)) GOTO 300         ! Wait 1
C
C     Start (restart) the clock
C
      CALL IPOKEB("172540,"21)                 ! Single cycle on 100kHz
C
C     Wait for leading edge of REQA bit in DRV11 status register
C
400   IF (IPEEK("167770).AND."000200) GOTO 400                ! Wait 0
450   IF (.NOT.(IPEEK("167770).AND."000200)) GOTO 450         ! Wait 1
C
C     Read the clock
C
      ICOUNT = IPEEK("172544)                          ! 16 bit binary
C
      CALL IPOKE("172540,"0)                           ! Stop the clock
C
C     Read the signal strength on channel 1 (the 2nd)
C
      CALL SAMPLE(1,1,1,0,3,IS)          ! Use ADAC LABDAC routine
C
      CALL IPOKE("172542,"0)                           ! Clear clock
preset buffer
C
C     Work out the desired results
C
      STRENG = 10*LSBVAL*(IS+2)                ! 0.9 = S9 (2=offset)
      IF (ICOUNT.EQ.0) GOTO 550                ! An overflow has occured
      PERIOD = FLT16(ICOUNT)*TICK              ! Real time (sec's)
C
      IF (PERIOD.LT.PERLO) GOTO 500            ! Limit 10-5*sigmaF [Hz]
550   PERIOD = PERLO
      IL = IL+1                                ! Lower F limit exc'd
500   IF (PERIOD.GT.PERHI) GOTO 600            ! Limit 10+5*sigmaF [Hz]
      PERIOD = PERHI
```

```
      IU = IU+1                                   ! Upper F limit exc'd
  600 DELTAF = 1.0/PERIOD-10.0
      TOTDEF = TOTDEF+DELTAF
      SIGMA = SIGMA+DELTAF**2                     ! Req'd for stand.dev.
C
      TOTS = TOTS+STRENG
      SIGS = SIGS+STRENG**2
C
      IM = IM+1
      IH = IH+1
C
      ID = IDP                                    ! Read time mark
      IDP = 0                                     ! Clear time mark
      IF (ID.EQ.1) STOP 'QUEUE PROBLEMS'          ! Error return
      IF (ID.EQ.2) GOTO 110                       ! Minute is up
      IF (ID.EQ.3) GOTO 120                       ! Hour is up
C
      GOTO 200
C
  110 CALL TIME(TIMEST)                           ! Upon LTC minute
C
C     Prepare hourly stats:
C
      HTDEF = HTDEF+TOTDEF
      HSIGM = HSIGM+SIGMA
      HTS = HTS+TOTS
      HSIGS = HSIGS+SIGS
C
C     Calculate and print stats every minute:
C
      AVDEF = TOTDEF/IM
      SDF = SQRT((SIGMA-IM*AVDEF**2)/(IM-1))
C
      AVS = TOTS/IM
      SDS = SQRT((SIGS-IM*AVS**2)/(IM-1))
C
C     Print/plot routines
C
      CALL SCOPY(SPACES,STRING)                   ! Clear string
      CALL INSERT('|',STRING,11,1)                ! Zero mark Av.dF
      CALL INSERT('*',STRING,11+INT(5*AVDEF+SIGN(0.5,AVDEF)),1)
      CALL INSERT('|',STRING,27,1)                ! Zero mark SigmF
      CALL INSERT('*',STRING,27+INT(10*SDF+0.5),1)          ! SigmF
```

```
         CALL INSERT('|',STRING,52,1)                    ! Zero mark AvS
         CALL INSERT('*',STRING,52+INT(10*AVS+0.5),1)    ! Av.S
         CALL INSERT('|',STRING,72,1)                    ! Zero mark SigS
         CALL INSERT('*',STRING,72+INT(20*SDS+0.5),1)    ! SigS
  C
         WRITE(2,610,ERR=904) AVDEF,SDF,AVS,SDS,
      #  (TIMEST(K),K=1,8),(STRING(K),K=1,82)
   610   FORMAT(X,F7.3,F7.3,F7.3,F7.3,'   ',8A1,'    ',82A1)
  C
  C   Check keyboard entry <CR> for halting the program
  C
         ISTAT = ITTINR()
         IF (ISTAT.GT.0) GOTO 120
  C
         GOTO 100                                       ! Next minute
  C
  C
  C   Print hourly stats:
  C
   120   HTDEF = HTDEF/IH
         HSDF = SQRT((HSIGM-IH*HTDEF**2)/(IH-1))
         HTS = HTS/IH
         HSDS = SQRT((HSIGS-IH*HTS**2)/(IH-1))
         WRITE(2,640,ERR=905) HTDEF,HSDF,HTS,HSDS,IU,IL,IH
  C
   640   FORMAT(X,F7.3,F7.3,F7.3,F7.3,'   Hourly statistics. Freq dev: ',
      #  '# too high = ',I3,' ; # too low = ',I3,' ; # samples = ',I5,
      #  /)
  C
         CLOSE(UNIT=2)                                           ! Close file
  C
         IF (ISTAT.GT.0) STOP 'Operator interrupt'
         GOTO 90                                                 ! Next hour
  C
   900   STOP 'Some problem in encoding extension mod'
   901   STOP 'More than 10 files of the same hour'
   902   STOP 'Some problem in opening the new file'
   903   STOP 'Some problem in filing the header printout'
   904   STOP 'Some problem writing the minute string to file'
   905   STOP 'Some problem writing the hourly statistics to file'
         END
  C
  C   ****************************************************
```

```
Local Variables, .PSECT $DATA, Size = 000604 ( 194. words)

Name    Type  Offset   Name    Type  Offset    Name    Type  Offset

AVDEF   R*4   000462   AVS     R*4   000466    DELTAF  R*4   000454
HSDF    R*4   000500   HSDS    R*4   000504    HSIGM   R*4   000372
HSIGS   R*4   000402   HTDEF   R*4   000366    HTS     R*4   000376
ICHAN   I*2   000356   ICOUNT  I*2   000440    ID      I*2   000460
IERR    I*2   000352   IFI     I*2   000340    IH      I*2   000364
IL      I*2   000362   IM      I*2   000426    IS      I*2   000442
ISTAT   I*2   000476   IU      I*2   000360    K       I*2   000354
LSBVAL  R*4   000334   PERHI   R*4   000434    PERIOD  R*4   000450
PERLO   R*4   000430   SDF     R*4   000342    SDS     R*4   000472
SIGMA   R*4   000412   SIGS    R*4   000422    STRENG  R*4   000444
TICK    R*4   000346   TOTDEF  R*4   000406    TOTS    R*4   000416


COMMON Block /      /, Size = 000012 (   5. words)

Name  Type  Offset   Name  Type  Offset   Name  Type  Offset

IDP   I*2   000000   AREA1 I*2   000002


Local and COMMON Arrays:

Name    Type  Section Offset  ------Size-----  Dimensions
AREA1   I*2   .$$$$.   000002      000010      (   4.) (4)
AREA2   I*2   $DATA    000314      000010      (   4.) (4)
DATEST  L*1   $DATA    000010      000012      (   5.) (10)
F       L*1   $DATA    000311      000002      (   1.) (2)
FILNAM  L*1   $DATA    000272      000017      (   8.) (15)
SPACES  L*1   $DATA    000146      000124      (  42.) (84)
STRING  L*1   $DATA    000022      000124      (  42.) (84)
TIMEST  L*1   $DATA    000000      000010      (   4.) (8)

Subroutines, Functions, Statement and Processor-Defined Functions:

Name    Type  Name    Type  Name    Type  Name    Type  Name    Type

AMIN1   R*4   DATE    R*4   FLT16   R*4   GTNAME  R*4   INSERT  I*2
INT     I*2   IPEEK   I*2   IPOKE   I*2   IPOKEB  I*2   ITIMER  I*2
ITTINR  I*2   REPEAT  R*4   SAMPLE  R*4   SCOPY   R*4   SIGN    R*4
SQRT    R*4   TIME    R*4   TIMEMK  R*4
```

```
0001       SUBROUTINE TIMEMK(ID)
   C    ****************************************************
   C
0002       COMMON IDPASS,IAREA(4)
0003       IDPASS = ID
0004       IF (ID.NE.3) RETURN
0006       CALL ICMKT(0,IAREA)                ! When hour up, cancel minute timer
0007       RETURN
0008       END
   C
   C    *****************************************************
```

```
FORTRAN IV    Storage Map for Program Unit TIMEMK


Local Variables, .PSECT $DATA, Size = 000002 (    1. words)

Name  Type  Offset    Name  Type  Offset    Name  Type  Offset

ID     I*2 @ 000000


COMMON Block /      /, Size = 000012 (    5. words)

Name  Type  Offset    Name  Type  Offset    Name  Type  Offset

IDPASS I*2  000000    IAREA  I*2  000002


Local and COMMON Arrays:

Name     Type   Section Offset ------Size----- Dimensions
IAREA    I*2    .$$$$.    000002      000010          (    4.) (4)

Subroutines, Functions, Statement and Processor-Defined Functions:

Name  Type  Name  Type  Name  Type  Name  Type  Name  Type

ICMKT  I*2
```

```
      SUBROUTINE GTNAME(FILNAM,IERR)
C    ****************************************************
C
      LOGICAL*1 DATEST(10), TIMEST(8), FILNAM(15),
     # TEMP1(3),TEMP2(4),TEMP3(2),FILEN(7)
      INTEGER*2 FILRAD(4)
C
      CALL DATE(DATEST)                     ! Get current date
      CALL TIME(TIMEST)                     ! Get current time
C
      CALL REPEAT(FILNAM,FILNAM,0)          ! Fill with null string
      CALL INSERT('DL1:',FILNAM,1,4)        ! Start off with device
      CALL SUBSTR(DATEST,TEMP1,1,2)         ! Get day
      CALL INSERT(TEMP1,FILNAM,5,2)         ! Enter day in filename
      CALL SUBSTR(DATEST,TEMP2,4,3)         ! Get month
      CALL INSERT(TEMP2,FILNAM,7,3)         ! Add month to filename
      CALL SUBSTR(DATEST,TEMP3,9,1)         ! Get year
      CALL INSERT(TEMP3,FILNAM,10,1)        ! Add year to filename
      CALL INSERT('.H',FILNAM,11,2)         ! Indicate extension
      CALL INSERT(TIMEST,FILNAM,13,2)       ! Add hour to extension
C
C     Check if file exists:
C
C     Make a RADIX-50 version of the file name:
C
      DATA FILRAD/3RDL1,0,0,0/
      CALL SUBSTR(FILNAM,TEMP2,5,3)                 ! Get first half
      CALL IRAD50(6,TEMP2,IFILE)                    ! Convert to RADIX
      FILRAD(2) = IFILE
      CALL SUBSTR(FILNAM,TEMP2,8,3)                 ! Get second half
      CALL IRAD50(6,TEMP2,IFILE)                    ! Convert also
      FILRAD(3) = IFILE
 230  CALL SUBSTR(FILNAM,TEMP2,12,3)                ! Get extension
      CALL IRAD50(3,TEMP2,IFILE)                    ! Convert extension
      FILRAD(4) = IFILE
C
      ICHECK = IGETC()                              ! Get a free channel
      IF(ICHECK.LT.0) STOP 'No free channel available'
      IOLD = LOOKUP(ICHECK,FILRAD)                  ! See if file is there
      IF(IOLD.EQ.-2) GOTO 220                       ! File not found
      IF(IOLD.LE.-3) STOP 'Unable to check old files'
C
```

```
C    File does exist, change extension:
C
      CALL CLOSEC(ICHECK)                              ! Close business
      CALL IFREEC(ICHECK)
C
      ENCODE(1,650,F,ERR=900) IFI              ! Change IFI to ASCII
 650  FORMAT (I1)
      CALL INSERT(F,FILNAM,12,1)               ! Modify extension
      IFI = IFI+1
      IF(IFI.EQ.10) GOTO 901
      GOTO 230                                 ! Try again
C
 220  CALL IFREEC(ICHECK)
      IERR = 0                                 ! Everything's dandy
      RETURN
 900  IERR = 1                                 ! ENCODE error
      RETURN
 901  IERR = 2                                 ! Too many filenames
      RETURN
      END
C
```

```
FORTRAN IV    Storage Map for Program Unit GTNAME

Local Variables, .PSECT $DATA, Size = 000072 (   29. words)

Name   Type  Offset        Name   Type  Offset   Name  Type  Offset

F         R*4  000064   ICHECK I*2  000060    IERR   I*2 @ 000002
IFI       I*2  000070   IFILE    I*2  000056   IOLD   I*2    000062

Local and COMMON Arrays:

Name      Type      Section   Offset  ------Size----- Dimensions
DATEST  L*1       $DATA    000004         000012       (   5.) (10)
FILEN     L*1       $DATA    000037         000007       (   4.) (7)
FILNAM  L*1 @   $DATA    000000         000017       (   8.) (15)
FILRAD   I*2       $DATA    000046         000010       (   4.) (4)
TEMP1    L*1       $DATA    000026         000003       (   2.) (3)
TEMP2    L*1       $DATA    000031         000004       (   2.) (4)
TEMP3    L*1       $DATA    000035         000002       (   1.) (2)
TIMEST   L*1       $DATA    000016         000010       (   4.) (8)


Subroutines, Functions, Statement and Processor-Defined Functions:

Name        Type  Name        Type   Name        Type   Name       Type     Name      Type

CLOSEC  R*4    DATE         R*4     IFREEC    I*2      IGETC       I*2      INSERT  I*2
IRAD50   I*2    LOOKUP   I*2     REPEAT   R*4     SUBSTR  R*4      TIME       R*4
```

Appendix V. FORTRAN program BLEEP3 for determining the modulation and group delay

```
FORTRAN IV          V02.1-1    Fri 23-Jan-87 16:39:39          PAGE 001

0001        PROGRAM BLEEP3
    C
    C     Sjoerd Jongens .... January 1987
    C
    C     This program measures the VNG 1 second "bleep"-delay in relation
    C     to a local (laboratory) clock, which needs to provide a sampling
    C     trigger burst (25kHz) on the DT2758 RTC trigger input of less than
    C    0.5 secs, or until halted by this program's dropping of CSR1 on
    C     a DRV11C parallel interface (could be altered to a D>A channel).
    C      ****************************************************
    C     This version takes one bleep at a time, and stores the corrected
    C     value in a 60 bleep array for smoothing.
    C     The sample buffer is adjusted to keep track of delay drift.
    C     After it has collected the first 60 bleeps, and determined the
    C     average delay, it locks onto that value allowing +/- 1ms only.
    C      *************************************************
0002        IMPLICIT INTEGER*2 (I,N)
0003        DIMENSION DELAY(60), ENVLPS(60)
0004        BYTE CURSOR(5)
    C
0005        INTEGER*2 AREA(4) ,IBUF(4000) ,NUMBER(60) ,IBFTOT(25)
0006        INTEGER*2 NSIGN(25) ,NSPRED(37)
0007        COMMON /TIME/ITEL
0008        EXTERNAL ITOUT
    C
0009        CURSOR(1)="033
0010        CURSOR(2)="133
0011        CURSOR(3)="001
0012        CURSOR(4)="101
0013        CURSOR(5)="200
    C
0014        TYPE *,'An ICOM R71A loudspeaker polarity is assumed (neg.)'
0015        NPOLRC = -1           ! Receiver pol.
    C
0016        TYPE *,'The delay is
           *assessed in a window of 25ms, after at least 3ms of'
0017        TYPE *,'noise determination.'
0018        TYPE *,'Adjustments are made
           *for smooth delay drifts, with allowance for only 1ms'
0019        TYPE *,
           *'either way compared with the smoothed delay (60 bleeps).'
```

```
        TYPE *,' '
  C
        MSEC = 25
        NBUFS = 25*MSEC
        NBUF = NBUFS+80
  C
        CALL IPOKE("44,"100.OR.IPEEK("44))        ! For RT11FB halt test
  C
  C  ***************************************************
  C  Synchronise the program to the external trigger (occuring
       1/second)
  C  Find the gap after the 25kHz burst:
  C  ***************************************************
   C
        CALL IPOKE("167770,"2)                    ! Enable trigger
  C
        IC = 0                                    ! Pre-check one cycle
  C
        TYPE *,'Bleeps here when trigger-burst cycle is found OK:'
        TYPE *,' '
  C
  C  Use LTC for time-out checks:
  C
   90   CALL ISYLTC(1)                            ! Ensure LTC is on
        CALL ITIMER(0,0,1,0,AREA,6,ITOUT)         ! Set up for 1 sec
  C
  100   ICMF = 0
        ITEL = 0
  C
  C  See if trigger burst appears during 1 tick period (1/50th sec):
  C  Get dummy DMA samples using the DT2758 ext. trigger
  C  Ignore DTLIB errors while entering during burst.
  C
        CALL RTS (ITEST,1,,,0,,,,9,ICMF,IBEF)     ! Set up for sampling
        CALL ITIMER (0,0,0,1,AREA,2,ITOUT)        ! Set up for 1 tick
  102   IF (ICMF.NE.0) GOTO 101                   ! Check if completed
        IF (ITEL.EQ.0) GOTO 102                   ! Check if time-out
        CALL RTS(,,,,,,,,-1,,)                    ! Clear RTS
  101   CALL ICMKT (2,AREA)                       ! Clear the ticker
  C
        IF (IBEF.EQ.0) GOTO 150                   ! Trigger or error?
        IF (ICMF.LT.0) GOTO 150                   ! Error = trigger
        IF (ITEL.EQ.2) GOTO 100                   ! Try another tick
```

```
0047       IF (ITEL.EQ.6) STOP 'No trigger pulse seen for 1 sec'
   C
   C
0049 150   CALL ICMKT(0,AREA)                       ! Clear all timers
   C
   C   Find the end (absence) of the burst:
   C
0050       CALL ITIMER(0,0,1,0,AREA,5,ITOUT)        ! Set up for 1 sec
   C
0051 155   ICMF = 0
0052       ITEL = 0
0053       CALL RTS (ITEST,1,,,0,,,,8,ICMF,IBEF)
0054       CALL ITIMER(0,0,0,1,AREA,3,ITOUT)        ! Set up for 1 tick
0055 162   IF (ICMF.NE.0) GOTO 161                  ! Completed?
0057       IF (ITEL.EQ.0) GOTO 162                  ! Time-out?
0059       CALL RTS(,,,,,,,,-1,,)                   ! Clear RTS
0060 161   CALL ICMKT(3,AREA)                       ! Clear the timer
   C
0061       IF (IBEF.EQ.0) GOTO 155                  ! Got a trigger
0063       IF (ICMF.LT.0) GOTO 155                  ! Error = trigger
0065       IF (ITEL.EQ.0) GOTO 155                  ! Ticked off?
0067       IF (ITEL.EQ.5) STOP 'No gap found within 1 second'
   C
0069       CALL ICMKT(0,AREA)                       ! Clear all timers
   C
0070       IF (IC.EQ.10) GOTO 200                   ! After header print
   C
0072       IF (IC.GE.1) GOTO 198                    ! Print header
   C
   C   See if trigger burst arrives again within 1/2 second:
   C
0074       ICMF=0
0075       ITEL=0
   C
0076       CALL RTS (ITEST,1,,,0,,,,9,ICMF,IBEF)
0077       CALL ITIMER (0,0,0,26,AREA,4,ITOUT)      ! Set up for 26 ticks
0078 172   IF (ICMF.NE.0) GOTO 171                  ! Completed?
0080       IF (ITEL.EQ.0) GOTO 172                  ! Time-out?
0082       CALL RTS(,,,,,,,,-1,,)                   ! Clear RTS
0083 171   CALL ICMKT (0,AREA)                      ! Clear all timers
   C
0084       IF (IBEF.NE.0) STOP 'No sample obtained after 1/2 sec gap'
0086       IF (ICMF.LT.0) STOP 'DTLIB error checking the gap'
```

```
      IF (ITEL.EQ.4) STOP 'No trigger pulse seen after 1/2 sec gap'
C
      CALL ITTOUR("007)                              ! Ring bell
      IC = 1                                         ! Indicate test passed OK
      GOTO 90
C
198   TYPE *,''
      TYPE *,'Hit <RETURN> to halt...'
      TYPE *,' '
      TYPE *, 'DELAY ZEROC  PHASE  S/N  No.cycles'
      TYPE *, ' [ms]  [ms]  [deg]  [dB]   #'
33     FORMAT ('+ MISSED one...      error',I2)
20     FORMAT ('+',F4.1,F6.1,I8,F6.1,I4,/,
     # F6.2,F6.2,I7,F7.1,I4)
      GOTO 197
1      CALL PRINT(CURSOR)                            ! After bleeps lost
197   TYPE *, '                         = last bleep'
      TYPE *, '                         = last 60 bleeps'
      CALL PRINT(CURSOR)                             ! Move cursor up
C
      IC = 10                                   ! Indicate header is printed
C
C  *************************************************
C
      DO 5 I = 1,60
      DELAY(I) = 3.
      ENVLPS(I)= 3.
5      NUMBER(I)= 0
      SMOOTH = 0.
      ARRIVE = 0.
      MPHASE = 0
      IPHASE = 1
      IPSIGN = NPOLRC
      MIN = 0
      IV = 0
      NSTART = 1
      NNOISE = 75
      NERR = 0
C
      NBUF = NBUFS+NNOISE+5
      LOCK = NBUF
C
      GOTO 90                                        ! Wait for end of trigger
```

```
C
C   ****************************************************
C   Now take a FAST SWEEP sample series, using the trigger burst:
C   ****************************************************
C
199  WRITE (5,33) NERR
     IF (IV.GT.0) IV = IV+1                   ! Smooth valid?
     IF (IV.GT.61) GOTO 1                     ! Missed >60 bleeps
C
200  CALL ISYLTC(1)                           ! Turn LTC on
     CALL ISLEEP(0,0,0,2)                     ! Allow WRITE to finish
     CALL ISYLTC(0)                           ! Turn LTC off
     ICMF = 0
     CALL RTS (IBUF,NBUF,,,0,,,,5,ICMF,IBEF)  ! Sample FAST SWEEP
     CALL LWAIT (ICMF,0)
C
     IF (ICMF.LT.0) STOP 'DTLIB error while sampling bleep'
C
     CALL IPOKE("167770,"0)                   ! Reset trigger disable
     CALL IPOKE("167770,"2)                   ! Enable next second
C
C   Now output it to the D>A for the CRO:
C
D   DO 300 I=1,250
C
D300 CALL IPOKE("176750,IBUF(I))
C
C   CALL IPOKE("176750,"0)                    ! Zero output
C
C   ****************************************************
C   Determine peak noise level during first 3ms (75 samples) or up:
C   ****************************************************
C
     INOISE = 1                               ! Default value
C
     IF (IV.EQ.0) GOTO 787                    ! Smooth valid?
     NNOISE = INT(25.*(ARRIVE-1.0)+0.5)       ! Allow 1ms gap
     NNOISE = MAX0 (NNOISE,75)                ! Set lower limit
C   NSTART = NNOISE-74                        ! New start point (opt.)
     NBUF = NBUFS+NNOISE+5                    ! New buffer size
     NBUF = MIN0 (NBUF,4000)                  ! Set upper limit
     LOCK = NNOISE+50                         ! Allow 2ms window
C
```

```
787  DO 350 ITALLY= NSTART, NNOISE
350  INOISE = MAX0 ((IABS(IBUF(ITALLY))),INOISE)
C
C    Set threshold:
C
     INOISL = 2*INOISE                              ! Set 6dB S/N ratio
     INOISL = MAX0 (INOISL,10)                      ! Minimum of 0.05V
C
C    ***************************************************
C    Find START of signal after the noise sampling:
C    ***************************************************
C
     NERR = -1                                      ! ERROR "NO BLEEP" NO.1
710  ITALLY = ITALLY+1                              ! Starting @ NNOISE+1
     IF (ITALLY.GT.LOCK) GOTO 199                   ! Start later than 2ms?
C
     IF (IABS(IBUF(ITALLY)).LE.INOISL) GOTO 710 ! Exc'ds noise limit?
     ITALST = ITALLY                                ! When?
     ISR = ISIGN(1,IBUF(ITALLY))                    ! Get polarity
C
C    Go back to where this cycle started:
C
713  ITALLY = ITALLY-1
     NERR = -2                                      !ERROR "NO BLEEP" NO.2
     IF (ITALLY.LE.NNOISE) GOTO 199                 ! Passed the start?
     NERR = -3                                      !ERROR "NO BLEEP" NO.3
     IF ((ITALST-ITALLY).GT.25) GOTO 199            ! More than 1 cycle?
C
     IF (IABS(IBUF(ITALLY)).LT.10) GOTO 715 ! Below minimum noise?
     IS1= ISIGN(IBUF(ITALLY))
     IS2 = ISIGN(IBUF(ITALLY-1))
     IF (IS1.EQ.IS2) GOTO 713                       ! Sign change?
715  IBEGIN = ITALLY-1                              ! When?
C
     ITALLY = ITALST                                ! Back to start
C
C    If already collected 60 bleeps, go to phase matching:
C
     IF (IV.GT.0) GOTO 5000                         ! Smooth valid?
C
C    ********************************************
C    Average all the zero-crossings (when not yet smoothed):
C    ***************************************************
```

```
C
      N = 1
      NC = 1
      TT = 0.
      SMAXST = 0.
C
  730 SMAXST = SMAXST+IMAXS                ! For averaging
      IMAXS = 1
      IPREV = ITALLY
C
C     The phase delay is worked out: (when ITALLY=1 : t=0), crossing=
C     midway ITALLY and ITALLY+1 for every second zero crossing;
C     t=(ITALLY-1+(0.5/2))/25kHz [msec],
C     reduced by number of half cycles:
C
      T = ((ITALLY-0.75)/25.)-(NC*0.5)     ! The time for this crossing
      T = AMAX1(T,3.0)                     ! Minimum= noise window
      TT = TT+T                            ! For the averaging
C
  780 ITALLY = ITALLY+1                    ! Next sample
      IF (ITALLY.GE.NBUF) GOTO 900         ! Reached bleep end?
C
      IMAXS = MAX0 ((IABS(IBUF(ITALLY))),IMAXS)! Get maximum signal
      IS1 = ISIGN(1,IBUF(ITALLY))
      IS2 = ISIGN(1,IBUF(ITALLY+1))
      IF (IS1.EQ.IS2) GOTO 780             ! Sign change?
C
      N = N+1                              ! Increment averaging
tally
      NC = NC+1                            ! Increment half cycle
count
  740 IF((ITALLY-IPREV).LT.23) GOTO 730    ! Missed any crossings?
      NC = NC+1                            ! Correct the 1/2 cycle
count
      IPREV = IPREV+25                     ! Correct the reference
      GOTO 740                             ! Try again
C
  900 PHADEL = TT/N                        ! For this bleep
      SMAXAV = SMAXST/N                    ! Average max signal
      CYCLES = N/2.
      NCYCLS = INT(CYCLES+0.5)    ! Continue with # whole cycles
      GOTO 910
C
```

```
C   ***************************************************
C   Do phase matching when smooth is valid (delay is well averaged)
C   ***************************************************
C
C   Add all the samples for each 1kHz cycle in each bleep (25 smples)
C
5000  SMAXST = 0.
C
      DO 5750 L=1,25
5750  IBFTOT(L) = 0
C
      CN = (NBUF+1-IBEGIN)/25.
      NUMB = INT(CN)                          ! # of 1kHz cycles
      NERR = -4                               ! ERROR "NO BLEEP" NO.4
      IF (NUMB.LT.1) GOTO 199                 ! Enough cycles?
      NCYCLS = 0
C
C   Store samples in new array, starting with this sample:
C
      DO 5720 K= IBEGIN+1, IBEGIN+(NUMB*25), 25       ! Take cycles
      IMAXS = 0
      DO 5730 L= 1, 25                        ! Take samples
      I = K+L-2
      IBFTOT(L) = IBFTOT(L)+IBUF(I)
5730  IMAXS = MAX0 (IABS(IBUF(I)),IMAXS)      ! Peak cycle signal
      IF (IMAXS.GE.NOISL) GOTO 5725           ! Acceptable signal?
      IMAXS = 0                               ! Eliminate if not
      NCYCLS = NCYCLS+1                       ! Tally rejects
5725  SMAXST = SMAXST+IMAXS
5720  CONTINUE
      NCYCLS = NUMB-NCYCLS                    ! Correct # cycles
      NUMB = NCYCLS                           ! Adjust for averaging
      NUMB = MAX0(NUMB,1)                     ! Prevent ./. 0
      SMAXAV = SMAXST/NUMB                    ! Average signal
C
C   Output the averaged cycle to the CRO:
C
D   DO 2500 L=1,25
D   BUFAV = IBFTOT(L)/NUMB                    ! Calculate average
D2500 CALL IPOKE("176750,INT(BUFAV+0.5))      ! To D>A Ch0
D   CALL IPOKE("176750,"0)                    ! Zero when done
C
C   *********************************************
```

```
      C   Determine the zero crossing(s) of the summed samples:
      C   ********************************************
      C
            LOCK = 1
            JUMP = 1
            AVZER1= 0.
            AVZER2= 0.
      C
      5780  IS1 = ISIGN(1,IBFTOT(LOCK))
            IS2 = ISIGN(1,IBFTOT(LOCK+1))
            IF (IS1.NE.IS2) GOTO (5800,5810),JUMP      ! Sign change?
      5805  LOCK = LOCK+1
            IF (LOCK.LT.25) GOTO 5780                  ! Reached cycle end?
            NERR = -5                                  ! ERROR "NO BLEEP" NO.5
            IF (JUMP.EQ.0) GOTO 199                    ! Seen any 0-x at all?
            GOTO 5820
      C
      5800  AVZER1= FLOAT(LOCK)                        ! First zero crossing
            AVZER2= AVZER1                             ! In case no second 0-x
            JUMP = 2                                   ! Change the GOTO addr.
            GOTO 5805
      C
      5810  AVZER2= FLOAT(LOCK)-12.5                   ! Second zero crossing
      C
      C   Take the average of the two 0-x's:
      C
      5820  AVZERA = (AVZER1+AVZER2)/2.
      C
      C   Determine the polarity of the cycles before the 1st 0-x,
      C   which should be same as start cycle of envelope:
      C
            IPSIGN = ISIGN(1,IBUF(1))                  ! Sign of average start
      C
      C   IF (IPSIGN.NE.ISR) CALL ITTOUR("007)         ! What if not?
      C
            PHADEL = (IBEGIN+AVZERA-1.)/25.-0.5        ! Phased delay
      C
      C   **************************************************
      C   Calculate average (smoothed) delays:
      C   **************************************************
      C
      C   Calculate the Signal-to-Noise ratio (dB):
      C
```

```
 910  SNOISE = FLOAT(INOISE)
      SMAXAV = AMAX1(SMAXAV,1.0)              ! Prevent Log errors
      STON = 20.*(ALOG10(SMAXAV/SNOISE))
      NERR = -6                               ! ERROR "NO BLEEP" NO.6
      IF (STON.LT.6.0) GOTO 199               ! Check minimum S/N
C
      MIN = MIN+1
      IF (MIN.LT.61) GOTO 998                 ! Got 60 yet?
      IV = 1                                  ! Smooth is valid
      MIN = 1                                 ! Reset index
C
C
C     Work out the smoothed (sliding average) delays:
C
 998  IF(IV.GT.0) IV= 1                       ! Reset error tally
C
      ENVLPC = IBEGIN/25.                     ! Retraced start.envelope
C
3000  IF (PHADEL.GE.ENVLPC) GOTO 3500 ! Out of phase by n cycles?
      PHADEL = PHADEL+1.0
      GOTO 3000
C
3500  IF ((PHADEL-ENVLPC).LE.0.5) GOTO 3600 ! Within 180 degrees?
      PHADEL= PHADEL-0.5
      GOTO 3500
C
3600  IF (NPOLRC.NE.IPSIGN) GOTO 3700         ! Opposite polarity?
      PHADEL = 2*ENVLPC+1.0-PHADEL            ! Adjust delay to phase
C
3700  NUMBER(MIN) = NCYCLS                    ! Number of cycles seen
      DELAY(MIN) = NCYCLS*PHADEL              ! Weigh this bleep-phase
      ENVLPS(MIN) = ENVLPC                    ! Corrected envelopes
C
      ISNUMB = 0
      SMOOTH = 0.
      ARRIVE = 0.
C
      DO 4000 I= 1,60
      ISNUMB = ISNUMB+NUMBER(I)               ! Total # cycles
      SMOOTH = SMOOTH+DELAY(I)                ! Weighed phase delays
4000  ARRIVE = ARRIVE+ENVLPS(I)               ! Totaling envelopes
C
      SMOOTH = SMOOTH/ISNUMB                  ! Weighed phase delays
```

```
      SMOOTH = AMAX1(SMOOTH,3.0)                 ! Set lower limit
C
      ARRIVE = ARRIVE/60.                        ! Envelope arrival
      ARRIVE = AMAX1(ARRIVE,3.0)                 ! Set minimum
C
      IPHASE = INT(360.*AMOD((PHADEL-ENVLPC),1.))
      MPHASE = INT(360.*AMOD((SMOOTH-ARRIVE),1.))
C
      IF (IV.GT.0) GOTO 6000                     ! Smooth valid?
      SMOOTH = 3.
      ARRIVE = 3.
      MPHASE = 0
C
 6000 IF (IV.EQ.0) GOTO 8000                     ! Smooth valid?
      IF (MIN.GT.1) GOTO 7000                    ! End of minute?
C
C    Place the smoothed phased delay on CH1 of A>D for 1 sec:
C
      ISM = INT((SMOOTH-3.0)*409.6+0.5)
      ISM = MIN0(ISM,2047)
      CALL IPOKE("176752,ISM)
      GOTO 8000
C
C    Place the smoothed arrival on 2nd channel (CH1) of A>D:
C
 7000 ISM = INT ((ARRIVE-3.)*409.6+0.5)          ! 8V = 4ms after noise
      ISM = MIN0 (ISM,2047)
      CALL IPOKE ("176752,ISM)
C
 8000 WRITE(5,20)
     # ENVLPC,PHADEL,IPHASE,STON,NCYCLS,ARRIVE,SMOOTH,MPHASE
C    PRINT *,ISPHAS,AVZER1,AVZER2,AVZERA
      CALL PRINT(CURSOR)                         ! Move cursor up
C
C    **************************************************
C    Indicate end of bleep calculations:
C
      CALL IPOKE("176750,400)                    ! +2V to CRO
      CALL IPOKE("176750,-400)                   ! -2V to CRO
C
      IST = ITTINR()
      IF (IST.GT.0) GOTO 600                     ! Check keyboard
C
```

```
0317      GOTO 200
   C
0318 600  CALL ISYLTC(1)                                    ! Turn LTC on
0319      STOP 'Operator interrupt'
0320      END
   C
   C  ***************************************************
```

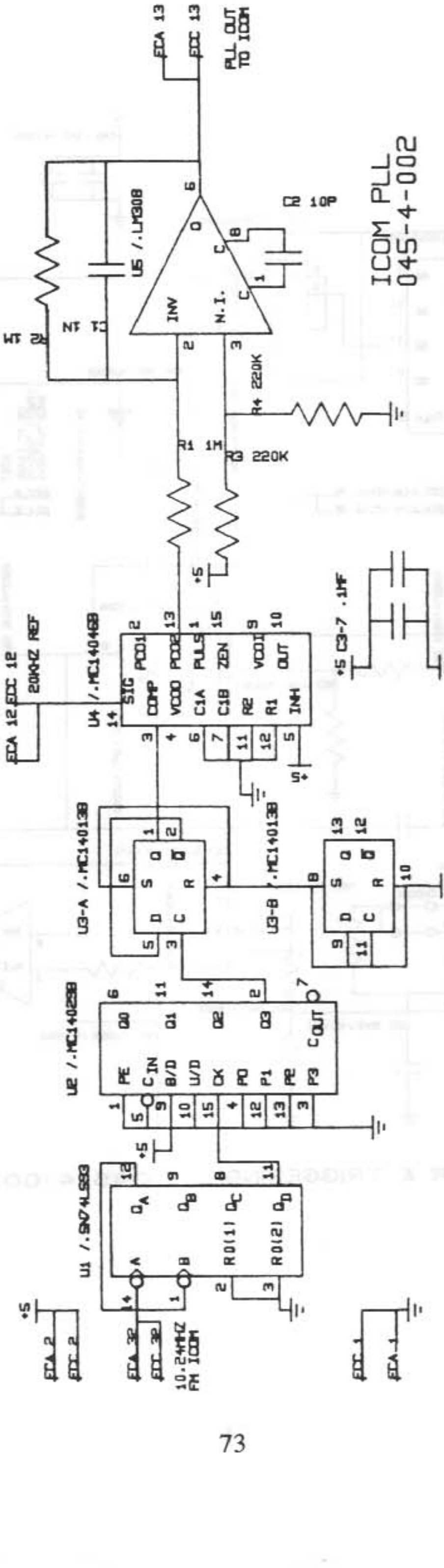
```
FORTRAN IV          Storage Map for Program Unit BLEEP3

Local Variables, .PSECT $DATA, Size = 021442 ( 4497. words)

Name     Type  Offset     Name     Type  Offset     Name     Type  Offset

ARRIVE   R*4   021174     AVZERA   R*4   021332     AVZER1   R*4   021322
AVZER2   R*4   021326     CN       R*4   021306     CYCLES   R*4   021276
ENVLPC   R*4   021346     I        I*2   021166     IBEF     I*2   021164
IBEGIN   I*2   021240     IC       I*2   021156     ICMF     I*2   021160
IMAXS    I*2   021256     INOISE   I*2   021222     INOISL   I*2   021226
IPHASE   I*2   021202     IPREV    I*2   021260     IPSIGN   I*2   021204
ISM      I*2   021354     ISNUMB   I*2   021352     ISR      I*2   021232
IST      I*2   021356     IS1      I*2   021234     IS2      I*2   021236
ITALLY   I*2   021224     ITALST   I*2   021230     ITEST    I*2   021162
IV       I*2   021210     JUMP     I*2   021320     K        I*2   021314
L        I*2   021304     LOCK     I*2   021220     MIN      I*2   021206
MPHASE   I*2   021200     MSEC     I*2   021150     N        I*2   021242
NBUF     I*2   021154     NBUFS    I*2   021152     NC       I*2   021244
NCYCLS   I*2   021302     NERR     I*2   021216     NNOISE   I*2   021214
NOISL    I*2   021316     NPOLRC   I*2   021146     NSTART   I*2   021212
NUMB     I*2   021312     PHADEL   R*4   021266     SMAXAV   R*4   021272
SMAXST   R*4   021252     SMOOTH   R*4   021170     SNOISE   R*4   021336
STON     R*4   021342     T        R*4   021262     TT       R*4   021246


COMMON Block /TIME  /, Size = 000002 (    1. words)

Name     Type  Offset   Name   Type  Offset   Name   Type  Offset

ITEL     I*2   000000
```

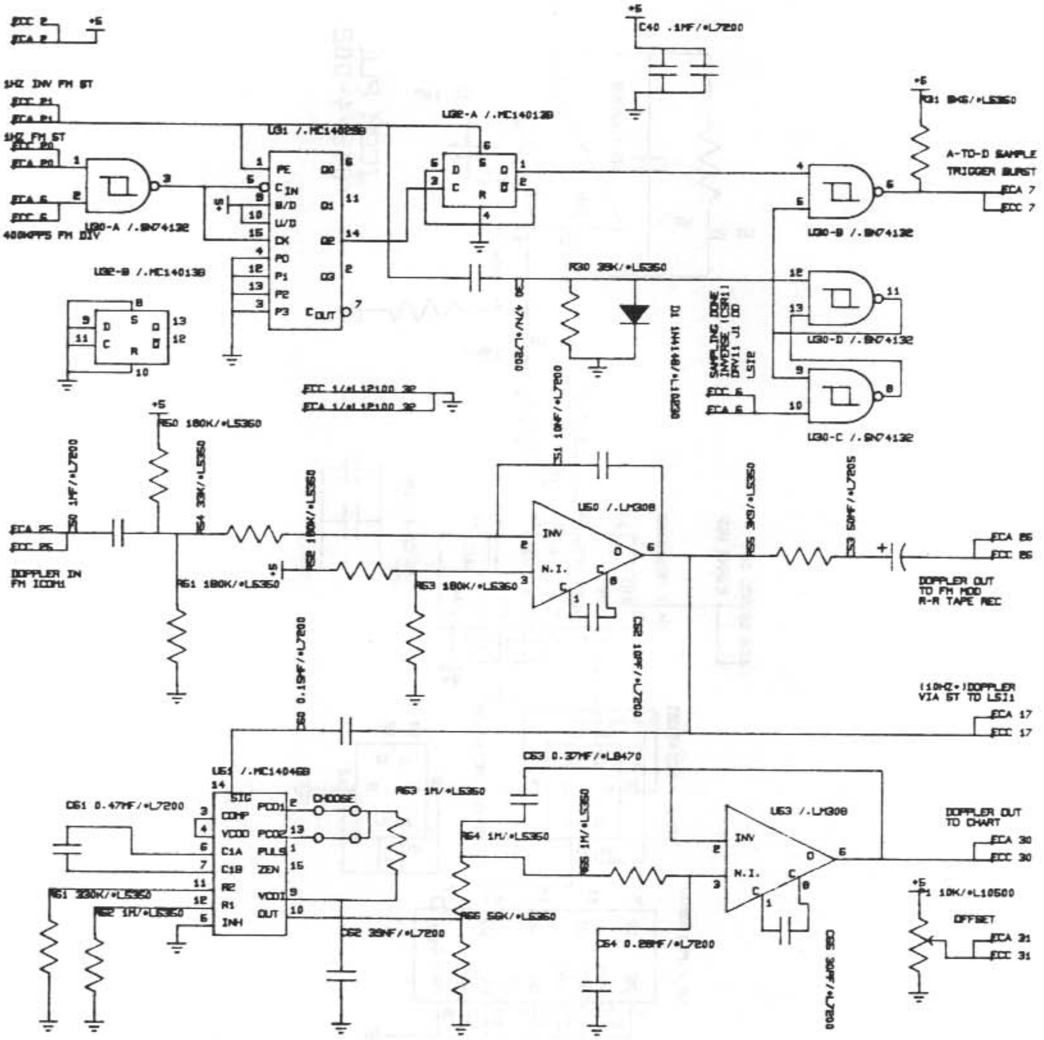
```
Local and COMMON Arrays:

Name    Type  Section   Offset  ------Size-----  Dimensions

AREA    I*2   $DATA     000746            000010   (   4.) (4)
CURSOR  L*1   $DATA     000740            000005   (   3.) (5)
DELAY   R*4   $DATA     000000            000360   ( 120.) (60)
ENVLPS  R*4   $DATA     000360            000360   ( 120.) (60)
IBFTOT  I*2   $DATA     020646            000062   (  25.) (25)
IBUF    I*2   $DATA     000756            017500   (4000.) (4000)
NSIGN   I*2   $DATA     020730            000062   (  25.) (25)
NSPRED  I*2   $DATA     021012            000112   (  37.) (37)
NUMBER  I*2   $DATA     020456            000170   (  60.) (60)

Subroutines, Functions, Statement and Processor-Defined Functions:

Name    Type  Name    Type  Name    Type  Name    Type  Name    Type

ALOG10  R*4   AMAX1   R*4   AMOD    R*4   FLOAT   R*4   IABS    I*2
ICMKT   I*2   INT     I*2   IPEEK   I*2   IPOKE   I*2   ISIGN   I*2
ISLEEP  I*2   ISYLTC  I*2   ITIMER  I*2   ITOUT   I*2   ITTINR  I*2
ITTOUR  I*2   LWAIT   I*2   MAX0    I*2   MIN0    I*2   PRINT   R*4
RTS     R*4
```

```
0001      SUBROUTINE ITOUT(ID)
   C  **************************************************
   C
0002      COMMON /TIME/ITEL
   C
   C
0003      IF (ITEL.NE.0) RETURN                           ! Already done?
0005      ITEL = ID                              ! Tell test loop
0006      RETURN
0007      END
   C  **************************************************
```

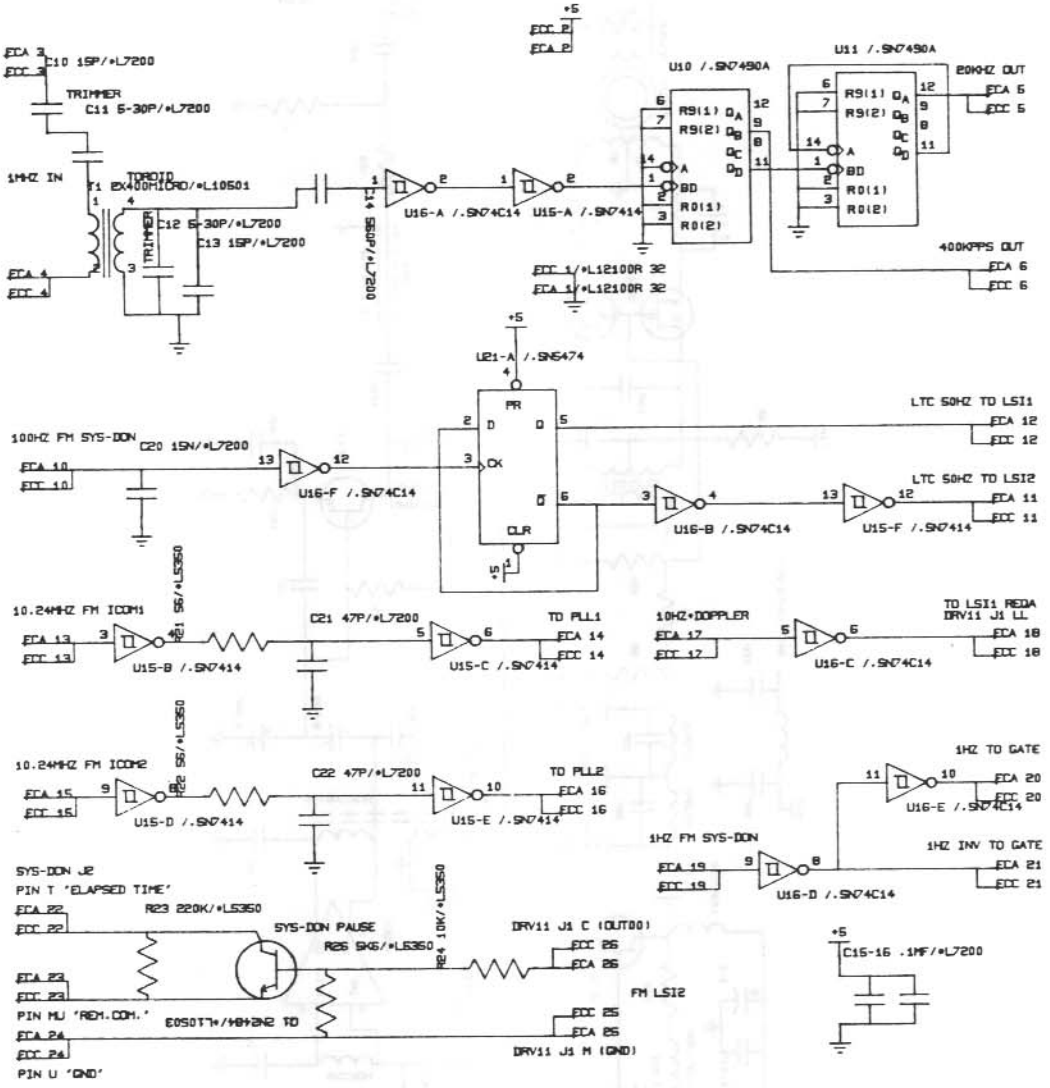
```
FORTRAN IV     Storage Map for Program Unit ITOUT

Local Variables, .PSECT $DATA, Size = 000002 (    1. words)

Name  Type  Offset    Name  Type  Offset    Name  Type  Offset

ID    I*2 @ 000000


COMMON Block /TIME  /, Size = 000002 (    1. words)

Name  Type  Offset    Name  Type  Offset    Name  Type  Offset

ITEL  I*2   000000
```

Appendix VI. Schematic diagrams of the auxilliary circuits required

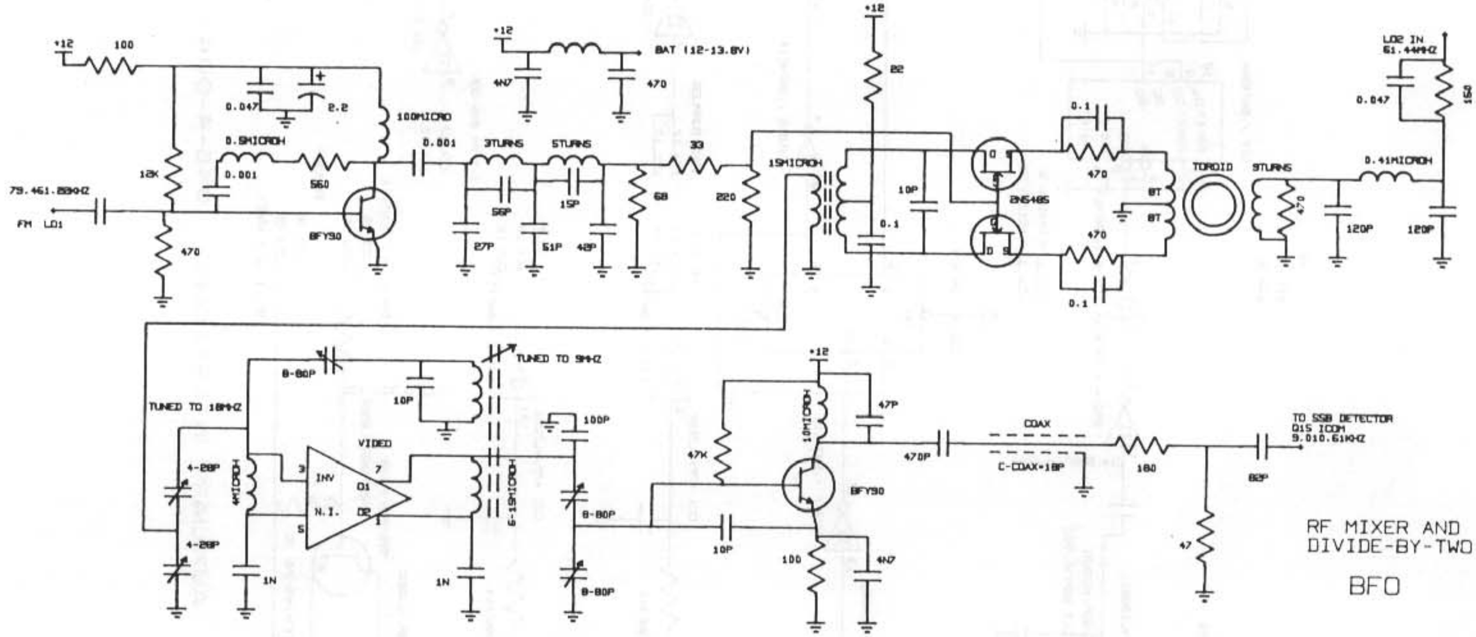




FREQ. METER & TRIGGERING 045-4-003

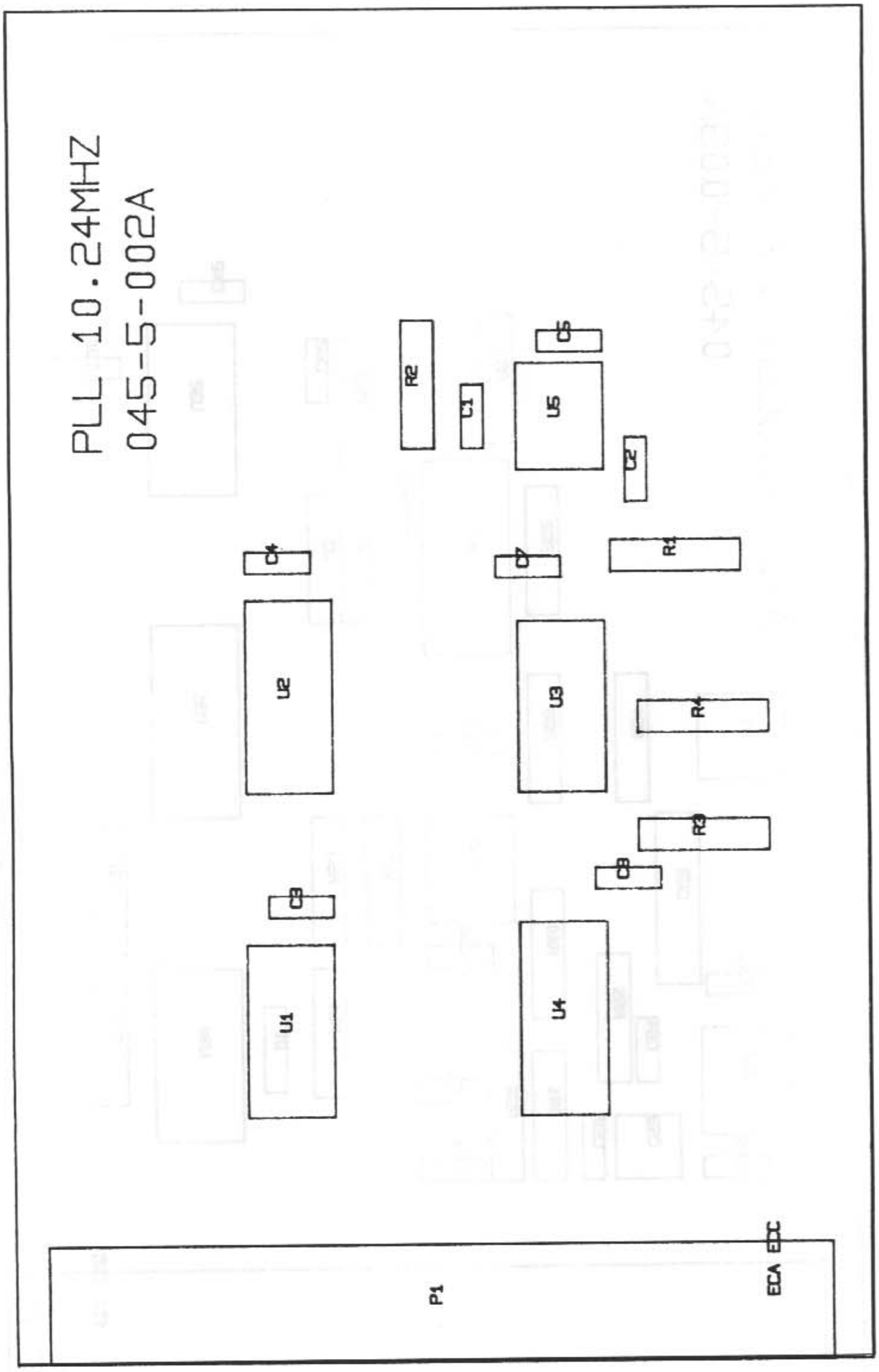


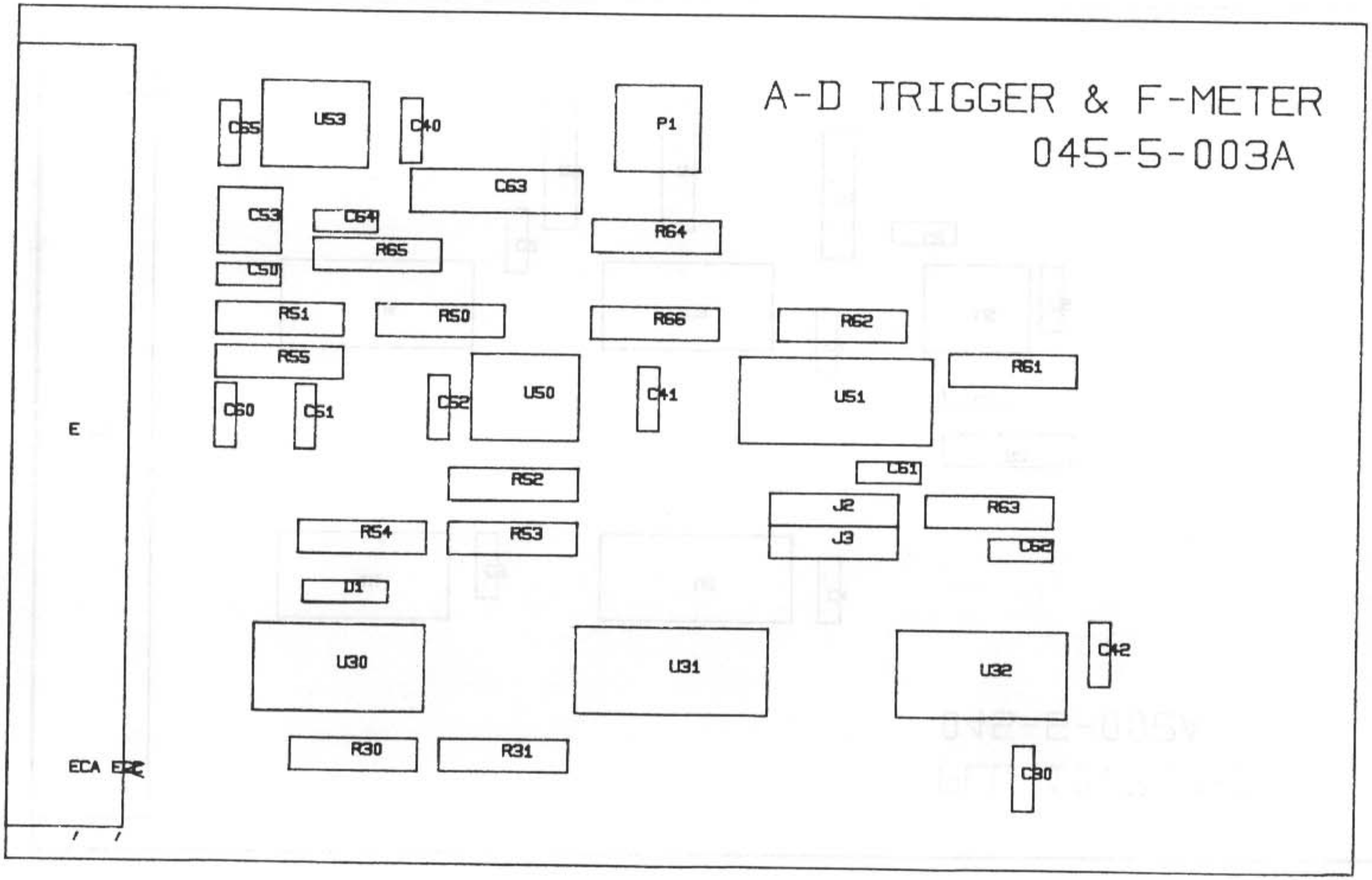
AUXILIARY INTERFACING 045-4-004



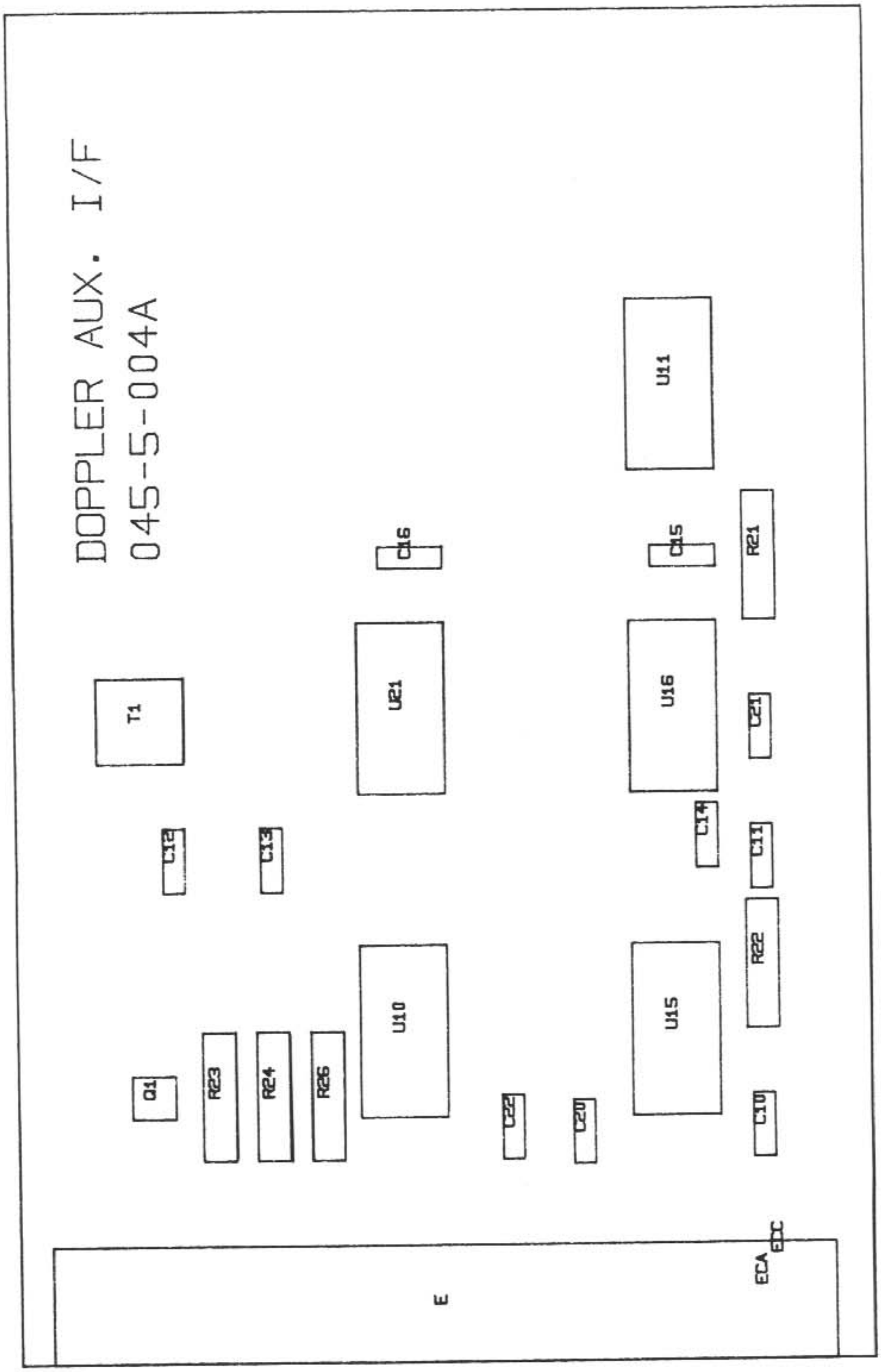
RF MIXER AND
DIVIDE-BY-TWO

BFO


PLL 10.24MHZ
045-5-002A
R2
C1
U5
C5
C2
C4
C7
R1
U2
U3
R4
R3
C8
C3
U1
U4
P1
ECA EDC



A-D TRIGGER & F-METER
045-5-003A


DOPPLER AUX. I/F
045-5-004A
T1
U21
U16
U11
C16
C15
R21
C21
C12
C13
C14
C11
U10
U15
R22
Q1
R23
R24
R26
C22
C20
C10
E
ECA
ECC