COMPUTER CONTROL OF A BARRY RESEARCH CHIRPSOUNDER

Thesis
Submitted in Fulfilment of the
Requirements for the Degree of
MASTER OF SCIENCE
of Rhodes University

by

GEOFFREY PHILIP EVANS

December 1984

ACKNOWLEDGEMENTS

Firstly , I wish to thank my supervisor , Professor J.A. Gledhill , for his encouragement and patience throughout the project.

I would like to thank Allon Poole who conceived the idea of advanced ionospheric chirpsounding and thereby presented me with a fascinating project. I am grateful for all his theoretical guidance , suggestions and advice.

I am grateful to Barry Guthrie who introduced me to the programming language FORTH which proved invaluable for software development.

The many helpful suggestions regarding presentation given by Ray Haggard , Ian Dore and Janos Csuzda were greatly appreciated.

I would like to thank the Department of Transport and the Co-operative Scientific Programmes division of the Council for Scientific and Industrial Research for financing this project. The project forms part of the Antarctic Upper Atmosphere Research Programme of the Department of Physics and Electronics of Rhodes University.

CONTENTS

ABSTRACT

Page

CHAPTER 1 INTRODUCTION TO IONOSPHERIC OBSERVATIONS

CHAPTER 2 EXTENDING THE CAPABILITIES OF THE VOS-1C VERTICHIRP SOUNDER

CHAPTER 3 CONTROLLER DESIGN CONCEPTS

CHAPTER 4 CONTROLLER HARDWARE

CHAPTER 5 CONTROLLER SOFTWARE

CHAPTER 6 SYSTEM TESTS

APPENDIX A VERTICHIRP CONTROLLER HARDWARE - MANUAL 1

APPENDIX B VERTICHIRP CONTROLLER HARDWARE - MANUAL 2

APPENDIX C VERTICHIRP CONTROLLER SOFTWARE - CHANGES AND ADDITIONS TO FORTH

APPENDIX D VERTICHIRP CONTROLLER SOFTWARE - OPERATORS MANUAL

APPENDIX E VERTICHIRP CONTROLLER SOFTWARE - PROGRAM GLOSSARY

Figure and Table Number | Page Number

ABSTRACT

This thesis describes the design and development of a computer-based controller together with additional hardware that greatly extends the capabilities of a Barry Research VOS-1 Chirpsounder.

The measurement of the virtual height of the ionosphere as a function of frequency using pulse- and frequency-modulated carrier wave (FMCW) techniques is described and the concept of the so called "digital" ionosonde is introduced.

The modifications required for the standard Chirpsounder to perform as a versatile digital chirp ionosonde are discussed. Simplified block diagrams are used to describe the Controller hardware which is fully described in two comprehensive service manuals which have been included as appendices.

Important aspects of the Controller software and data storage formats are described in detail. The emphasis is then placed on system capabilities. An operators' software manual which describes system initialization and operation in terms of system commands is included as an appendix.

Results of tests at both Grahamstown , South Africa , and at the SANAE base in the Antarctic are presented.

CHAPTER 1

INTRODUCTION TO IONOSPHERIC OBSERVATIONS

1.1 The Ionosphere

The ionosphere extends from an altitude of about 60 km to more than 1000 km [Mitra , 1952 ; Kelso , 1964]. In this part of the atmosphere the constituent gases are partially ionised by the action of solar radiation in the ultra-violet and X-ray region of the electromagnetic spectrum.

The ionization per unit volume varies with altitude , latitude , longitude , time of day , season and sunspot cycle among other effects. Ion and electron motions are influenced by mutual interactions , interaction with the neutral gases , the geomagnetic field , electric fields , gravitation and diffusion.

Although the electron concentration in the ionosphere is only about 1% of that of the neutral gases it gives rise to phenomena that are negligible or absent in the lower atmosphere. Of great importance is the ability of the ionosphere to refract and reflect radio waves. Besides finding practical application in long distance radio communications this property is used for the investigation of the physics of the ionosphere itself.

1.2 Ionospheric Observations

In 1925 Breit and Tuve developed the pulse-echo method of ionospheric observation [Mitra , 1964]. This involved starting the sweep of an oscilloscope and then transmitting a radio frequency pulse of a few hundred microseconds duration. After a delay the pulse echoes from the ionosphere were received and displayed on the same oscilloscope sweep. The time between the transmission of the RF pulse and the reception of the first echo was obtained directly from the oscilloscope. By multiplying this delay time Δt by the velocity of an electromagnetic wave in free space the apparent or equivalent path length was obtained.

This distance is equal to the distance that would be travelled in the same time interval by a signal propagating in free space. Division of this distance by two gives the virtual height of the ionosphere at the transmitted frequency.

If $h'(f,t)$ = virtual height of the ionosphere
c = velocity of an electromagnetic wave in free space
Δt = delay time

Then :-

$$h' = c\Delta t/2 \qquad (1.1)$$

The virtual height is thus directly proportional to the signal delay time.

The pulse-echo principle is used in the pulse ionosonde to produce an ionogram which is a plot of virtual height as a function of radio frequency. Ionograms are generally recorded on film. A typical ionogram recorded at Grahamstown , latitude 33^0 18' S , is shown in figure (1.1).

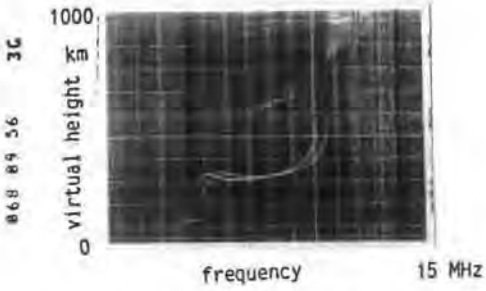


Figure (1.1) An Ionogram

The ionosphere often contains several layers , the maximum plasma frequency of a given layer being called the critical frequency of that layer. The maximum critical frequency of the whole ionosphere is known as the penetration frequency. Frequencies higher than the penetration frequency are not reflected though they may be scattered.

1.3 Ionosonde Improvements

Technological advances over the years have resulted in significant changes to the basic ionosonde together with new methods of data recording and processing. The earliest ionograms were recorded manually and were of poor accuracy when ionospheric characteristics changed rapidly. The time to record an ionogram was ,in general , reduced by automatic frequency-sweep ionosondes. Further improvements resulted when the motor-driven mechanical methods of generating a frequency sweep were replaced by electronic methods. The basic ionosonde became smaller , cheaper and less complex.

M.L.Philips (1974) briefly describes the important characteristics of a number of ionosondes that have been developed. Included are low frequency ionosondes designed to provide additional information on the complex D and E regions and a phase ionosonde which gives much improved resolution in the measurement of virtual height. He also describes equipment which uses coded pulse signals to improve signal-to-noise ratio.

Besides the basic measurement of the virtual height of the ionosphere as a function of frequency researchers devised equipment to separate the two circularly polarised components of the ionospheric echo. Even further assistance in ionogram interpretation was obtained with the determination of the angle of arrival of the returning signal.

1.4 The Digital Ionosonde

About ten years ago the digital computer made possible the so called "digital" or "advanced" ionosonde. In such an ionosonde the entire sounder function is under computer control and numeric rather than analogue analysis of the data is done. The flexibility of the software together with rapid data processing permits not only the recording of the virtual height of the ionosphere but simultaneous measurements of the received signal amplitude, phase and polarization. From these measurements the angle of arrival of the reflected signal and the Doppler velocity of the reflecting layer can be calculated.

At present two important digital ionosondes are the Dynasonde and the Digisonde. The Dynasonde , developed by J. W. Wright , was designed for the investigation of rapid ("dynamic") ionospheric variations [Wright , 1970]. A digital computer controls the system comprising frequency synthesizer, pulse transmitter and receiver and also processes the received data. Besides recording the digitized data on magnetic tape an analogue ionogram can be recorded on film. The system covers the frequency range 0.5 MHz to 32.0 MHz. The Dynasonde can also calculate real-time electron density profiles and electron collision frequencies.

The Digital Integrating Goniometric Ionospheric Sounder or Digisonde developed by K. Bibl concentrates on the improvement of the signal-to-noise ratio by the digital integration of phase-coded pulses [Bibl , 1970]. Ionograms from 0.5 MHz to 16.0 MHz are produced in digital form from tape.

Both sounders described above are based on pulse ionosonde techniques. Digital sounders , being recent innovations , are expensive and as a result there are only a few in general use today.

1.5 The FMCW Ionosonde (Frequency Modulated Continuous Wave) Vertical Incidence Operation

The pulse ionosonde transmits an impulse with a pulse length equal to the desired time delay resolution and a pulse repetition rate low enough to avoid any ambiguities over the expected time-delay range. The major disadvantage in such a system is the difficulty in achieving adequate average power without excessive peak power. High peak powers (in the range 1 to 30 kW) cause interference in nearby HF receivers and require large vacuum tube amplifiers together with expensive antennae and feed systems.

The development of the digitally-synthesized frequency sweep by Dr. G. Barry and Dr. R. Fenwick provided the key to the successful application of linear frequency modulation to ionospheric sounding [Barry Research , 1971]. Linear frequency modulated ionosondes produce a characteristic "chirp" when heard on a normal HF receiver hence the tradename "Chirpsounder" adopted by the Barry Research Corporation.

The purpose of both pulse and chirp ionosondes is to measure propagation-mode time delay (and hence virtual height) and signal amplitude as a function of transmitted frequency. The way in which this is done differs greatly between the two techniques.

The frequency versus time distribution of transmitted and received energy is shown below for both ionosondes [Fenwick , 1973].

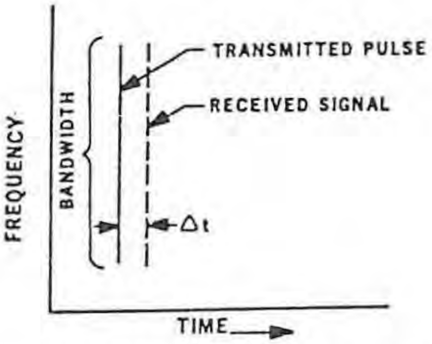


Figure (1.2a) Pulse Sounder - frequency vs time

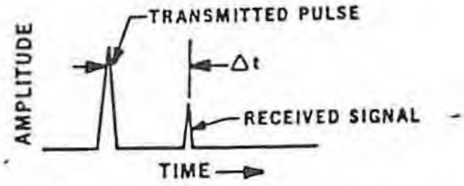


Figure (1.2b) Pulse Sounder - amplitude vs time

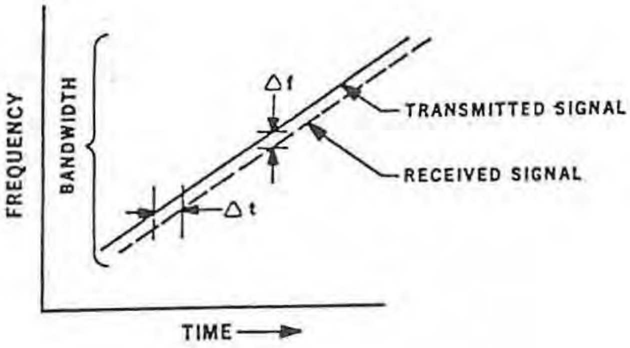


Figure (1.3a) FMCW Sounder - frequency vs time.

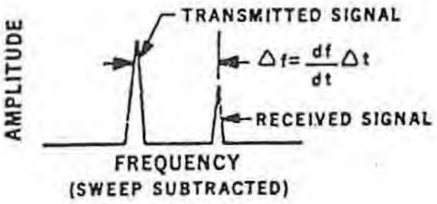


Figure (1.3b) FMCW Sounder - amplitude vs frequency

Figure (1.2a) shows that the pulse transmitter emits energy simultaneously across the entire bandwidth of about 20 kHz. The received signal has the same energy distribution but is delayed by the travel time Δt.

The Chirpsounder transmits a continuous synthesized linear frequency ramp as shown in figure (1.3a). The received signal is a frequency sweep delayed by the travel time Δt. Provided that the total sweep time is long compared to Δt the received signal may be regarded as being offset in frequency by Δf from the transmitted frequency.

This frequency difference Δf is related to the travel time by :-

$$\Delta f = (df/dt)\Delta t \qquad (1.2)$$

where df/dt is the linear sweep rate.

If the received frequency is subtracted from that being simultaneously transmitted , a signal with a particular delay becomes a tone with a frequency proportional to that time delay. This tone is then analysed using an audio spectrum analyser to obtain a time delay display. A simplified FMCW Sounder block diagram (after Fenwick , 1973) is given in figure (1.4) .

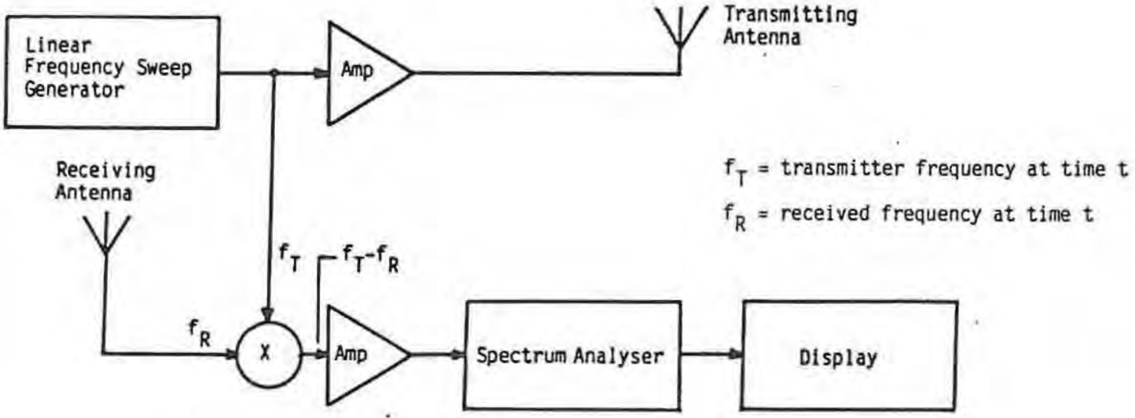


Figure (1.4) Block Diagram - Simplified FMCW Sounder

The frequency range analysed is determined by the linear sweep rate and the desired delay time range. Example :- For a sweep rate of 50 kHz/s and a maximum delay time of 10 ms , the frequency difference, Δf_{max}, would be

$$\Delta f_{max} = (50000)(0.01)$$
$$= 500 \text{ Hz}$$

Maximum ionospheric delay times , except near the critical frequency , are of the order of 5 ms.

The effective bandwidth of the sweeping receiver is thus very small (Eg. 500 Hz). This means that a good signal-to-noise ratio can be obtained with low transmitter output power (Eg. VOS-1 Chirpsounder output = 8W peak). The small bandwidth also reduces the vulnerability of the sweeping receiver to narrow-band interference which is encountered sequentially and can be reduced or eliminated by clipping or gating.

Vertical incidence operation with co-located transmitter and receiver is achieved by using electronic T/R switching which alternately connects the transmitter and then the receiver to a common antenna.

A limitation of the FMCW technique is the ambiguity between time delay and Doppler frequency shifts , the latter being caused by line-of-sight motion of the reflecting point. In certain radar applications this can cause serious problems. For mid-latitude ionospheric sounding at "normal" (50 kHz/s) sweep rates the maximum Doppler frequency shift is only about 1% of the frequency difference Δf and is usually ignored. The Digital Chirpsounder (controlled by the hardware and software described in this thesis) allows evaluation of both virtual height and Doppler shift using the FMCW technique.

Oblique Incidence Operation

Geographic separation of transmitter and receiver allows the echo structure of an obliquely transmitted signal to be obtained. Either pulse or FMCW techniques can be used to obtain oblique ionograms.

In the case of the pulse sounder the pulse repetition rates and the frequencies of transmitter and receiver must be kept synchronised. This is achieved using independent stable crystal oscillators [Davies , 1965].

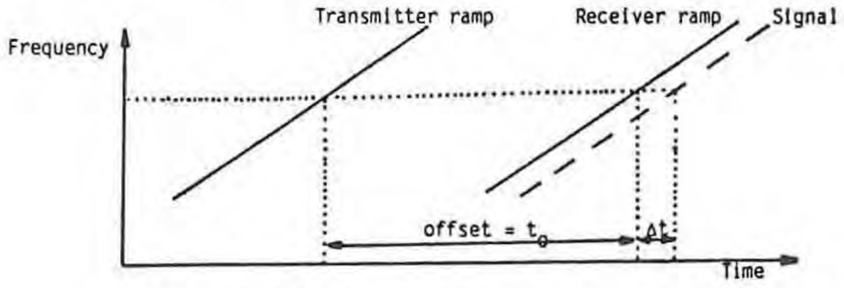


Figure (1.5) FMCW Oblique Incidence Operation

Oblique incidence recordings using FMCW require that the transmitting and receiving chirpsounders generate frequency ramps that are identical except that the receiver ramp is delayed with respect to the transmitter ramp. This delay or offset must be such as to allow the signal to fall within the receiver bandwidth.

Δt , the relative time delay of the signal at a specific frequency with respect to the receiving system clock is obtained from the oblique ionogram. If the offset t_0 is known the total propagation time t can be obtained. Multiplication of t by the velocity of an electromagnetic wave in free space gives the virtual group path length P' of the signal [Davies, 1975].

An oblique ionogram is thus a record of group path P' versus frequency with a range determined by the receiver bandwidth and a scale zero determined by the offset t_0.

1.6 Electromagnetic Waves in an Ionised Medium

To clarify the meanings of the terms phase range , group range and real range it is necessary to discuss the propagation of an electromagnetic wave in a plasma [Ratcliffe ,1959 ; Budden , 1964 ; Ratcliffe 1972].

For an electromagnetic wave in a vacuum the phase velocity (v) is equal to the speed of light (c) in a vacuum.

$$v = c \qquad (1.3)$$

For any other medium the phase velocity is given by

$$v = c/n \qquad (1.4)$$

where n is the phase refractive index of the medium.
If n depends on the wave frequency the medium is said to be dispersive.

Fourier analysis shows that a wave can only be monochromatic if it is of infinite duration. The phase velocity and refractive index mentioned above refer to an unmodulated and therefore monochromatic wave of infinite duration.

If the duration is finite (Δt) the waves must occupy a finite bandwidth ($\Delta f = 1/\Delta t$) for cancellation before and after the time Δt. The sinusoidal amplitude modulation ω_m of a carrier ω_c is identical to a monochromatic carrier ω_c plus two monochromatic sidebands $\omega_c + \omega_m$ and $\omega_c - \omega_m$. If the phase or the frequency is modulated instead of the amplitude , the modulated wave can still be represented as a carrier plus a spectrum of sidebands. When the wave is of finite duration a continuum of sidebands is obtained.

If a propagated wave is also modulated , the velocity of the modulation , the group velocity , is important. The group velocity is not necessarily equal to the phase velocity since each carrier and sideband propagates at the phase velocity appropriate to its frequency. The medium through which propagation takes place will , in general , be dispersive , so that the different components will travel at different speeds and the modulation will travel at some other velocity - the group velocity.

To summarise , a pulse of waves travels at the group velocity while waves within the pulse travel at the phase velocity. Energy and information are carried at the group velocity.

The formula for the refractive index of an ionised medium is given by the Appleton - Hartree equation [Ratcliffe , 1959 pg 37 equation 4.4.1]. If collisions and the geomagnetic field are neglected the refractive index n is given by :-

$$\omega_n^2 = 1 - X$$

$$= 1 - \omega_n^2/\omega^2 \qquad (1.5)$$

where ω_n is proportional to the electron density

and ω is the angular wave frequency

As the wave penetrates the ionosphere the phase refractive index becomes smaller. For ω_n greater than ω the wave cannot propagate since the refractive index becomes imaginary. The energy carried by the radio wave is reflected from the level at which $\omega_n = \omega$, that is where the plasma frequency equals the wave frequency.

A radio pulse travels at the group velocity u.

$$u = c/n_g \qquad (1.6)$$

where c = speed of light in a vacuum , n_g = group refractive index.

If the phase refractive index n is given by :-

$$n = \omega_n{}^2/\omega^2 \qquad (1.7)$$

the group refractive index can be shown to be

$$n_g = 1/n \qquad (1.8)$$

The radio pulse therefore travels more slowly in the ionosphere than light in free space. The calculation of virtual height or group range assumed a constant velocity equal to that of light in a vacuum. Because of group retardation in the ionosphere the virtual height is always greater than the real height. The real range is the actual path travelled by a signal of frequency f to the point of reflection and is denoted h.

Inclusion of the geomagnetic field in the Appleton-Hartree equation gives two values for the refractive index. This gives rise to two components in the vertical-incidence signal reflected by the ionosphere which are called the ordinary wave (o) and the extraordinary wave (x). The names "ordinary" and "extraordinary" are taken over from the phenomenon of double refraction in optics. These waves will , in general , have elliptical polarizations with opposite senses.

1.7 Thesis Scope

This thesis describes the computer controller and associated hardware and software developed to add digital capabilities similar to those of the Dynasonde and Digisonde to an FMCW ionosonde , the Barry Research VOS-1 Vertichirp Sounder.

CHAPTER 2

EXTENDING THE CAPABILITIES OF THE VOS-1C VERTICHIRP SOUNDER

2.1 General Introduction to the VOS-1C Vertichirp Sounder

The Barry Research VOS-1C Vertichirp Sounder is an FMCW or "chirp" ionosonde designed to produce vertical and oblique incidence ionograms in the frequency range 0.5 to 30.0 MHz [Barry Research , 1972].

In the vertical incidence mode the virtual height range is 0 to 1000 km. In the oblique mode range windows of 3 1/3 ms , 5 ms , 6 2/3 ms and 10 ms are available.

Twelve three-position switches allow the automatic selection of a vertical run , an oblique run or no run ("off") at 5 minute intervals in the hour. All oblique runs are either signal receptions or signal transmissions.

Vertical operation requires a single antenna which is used for both transmission and reception. Electronic transmit/receive (T/R) switching reduces the transmitter peak power of 8 W to an average of 3 W. An antenna switch selects a different antenna for oblique operation.

Both vertical and oblique ionograms are recorded on 35mm film. Identification information for each ionogram includes year , place , year-day number and time. Each ionogram is recorded with frequency and range marks.

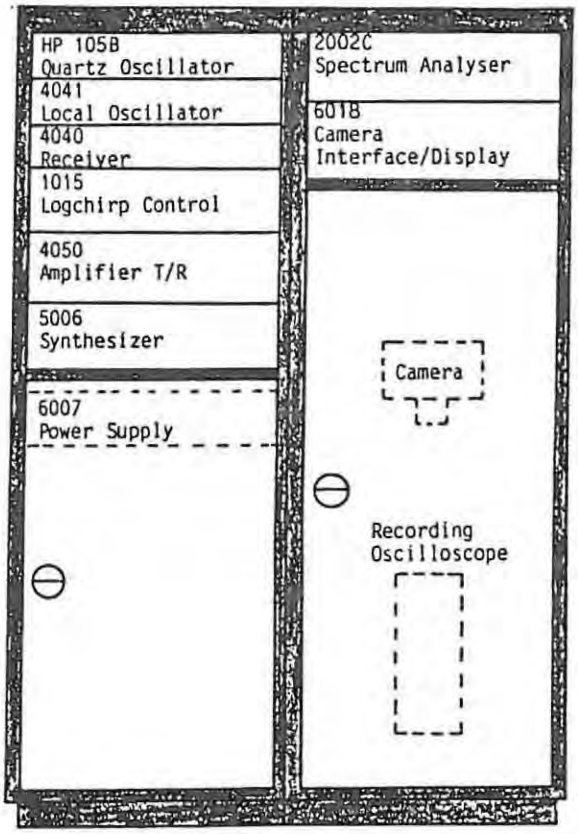


Figure (2.1) shows the physical location of all the units in the system.

Figure (2.1) VOS-1C Vertical/Oblique Chirpsounder

2.2 Vertical Operation

Front panel switches on the 1015 Logchirp Control set the vertical low and high frequency limits and allow the selection of one of fifteen "linear" or "logarithmic" overall sweep rates. These overall frequency versus time curves are approximated by linear segments of half-second duration of a fixed 50 kHz/s basic sweep rate. At the end of each segment the frequency is either advanced or retarded very rapidly so that the segments define the desired overall frequency versus time curve. These half-second segments followed by a high speed correction are called "cells".

Figure (2.2a) shows a linear overall sweep rate with jump forward corrections , necessary because the overall sweep rate is greater than the 50 kHz/s basic rate.

Figure (2.2b) shows a linear sweep rate with jump back corrections since the overall rate is less than the basic rate.

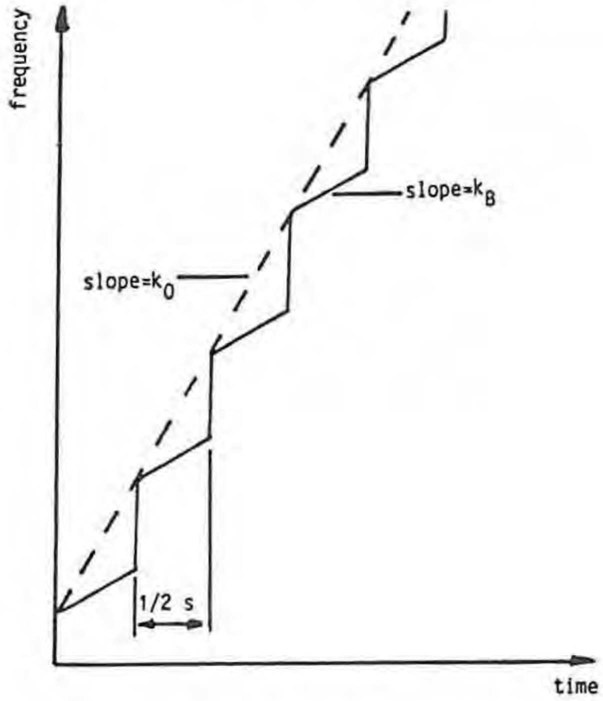


Figure (2.2a) Linear Sweep - jump forward corrections

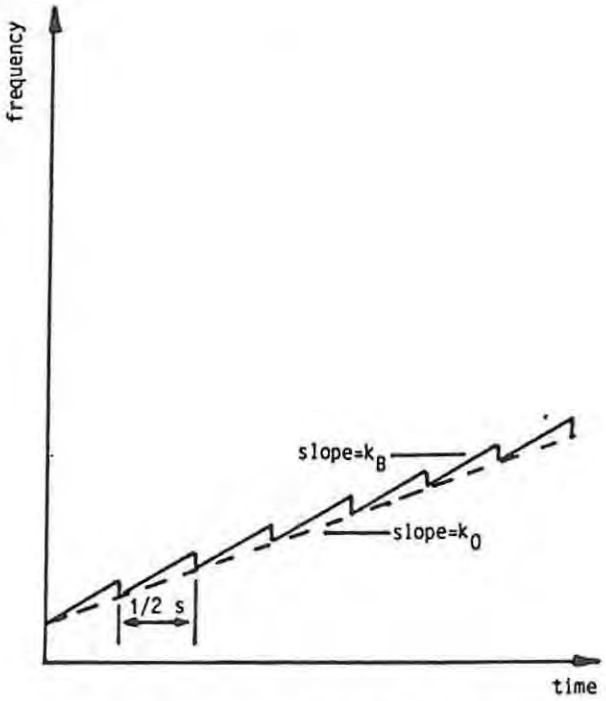


Figure (2.2b) Linear Sweep - jump back corrections

To produce an ionogram with a logarithmic frequency scale an exponential frequency increase with time is approximated as shown (on an exaggerated scale) in figure (2.3).

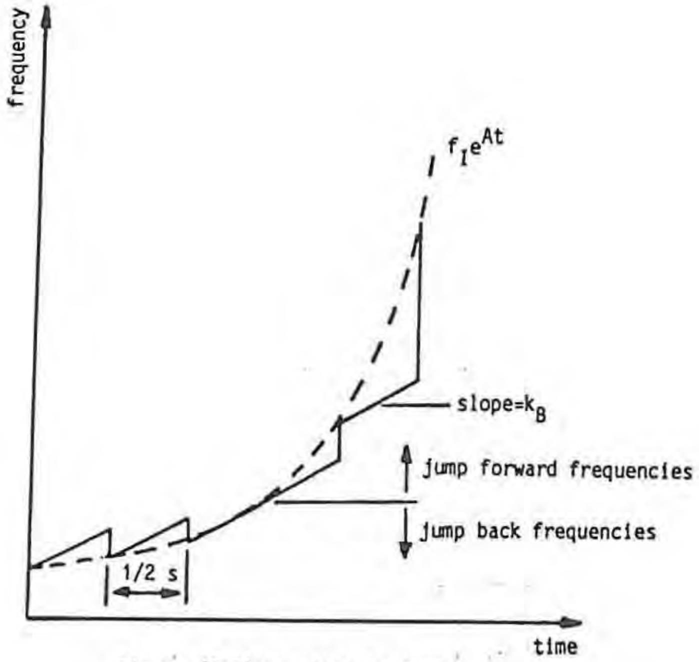


Figure (2.3) Logarithmic Sweep

An exponential frequency increase with time takes the form :-

$$f(t) = k_1 e^{At} \qquad (2.1)$$

where

A = the exponential time constant

t = time

At t = 0 $f(0) = k_1$ so that k_1 is the ionogram start frequency f_I.

Equation (2.1) becomes :-

$$f(t) = f_I e^{At} \qquad (2.2)$$

As stated in section 1.5 a received FMCW signal may be regarded as being frequency-offset by Δf from the transmitted frequency , provided the total sweep time is long compared to the travel (or delay) time.

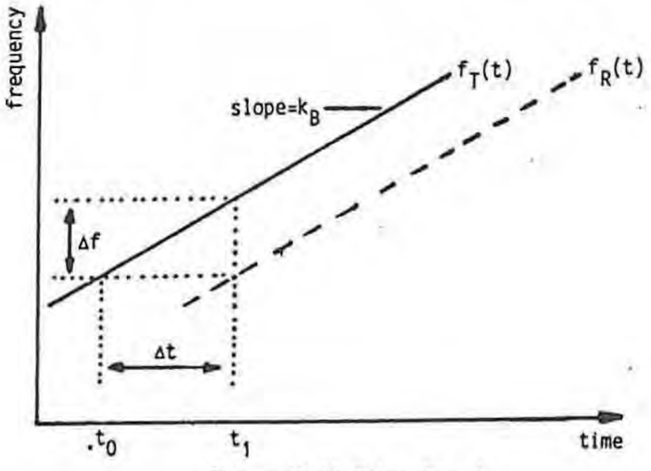


Figure (2.4) Chirp Signal

Let f_T = transmitted frequency at time t_1

and f_R = received frequency at time t_1

then the difference frequency $\Delta f = f_T - f_R$ (2.3)

Equation (1.2) gives the relationship between frequency difference and delay time as :-

$$\Delta f = (df/dt)\Delta t \qquad (2.4)$$

where df/dt is the basic linear sweep rate.

Let $df/dt = k_B$, then

$$\Delta f = k_B \Delta t$$

$$f_T - f_R = k_B \Delta t$$

$$\Delta t = (f_T - f_R)/k_B \qquad (2.5)$$

The virtual height of the ionosphere h' is given by equation (1.1) as :-

$$h' = c\Delta t/2$$

$$= c(f_T - f_R)/2k_B \qquad (2.6)$$

$(f_T - f_R)_{max}$ is equal to the effective bandwidth of the receiver which is 500 Hz for the Vertichirp sounder. The basic rate k_B is fixed at 50 kHz/s.

$$h'_{max} = (3.10^5)(500)/(2)(50.10^3) \text{ km.}$$

$$= 1500 \text{ km.} \qquad (2.7)$$

In the Vertichirp system however the spectrum analyser is set to analyse 0 to 333 Hz giving :-

$$h'_{max} = (3.10^5)(333)/(2)(50.10^3) \text{ km}$$

$$= 1000 \text{ km.} \qquad (2.8)$$

2.3 Oblique Operation

The lower and upper frequency limits for oblique operation are set by front panel switches on Logchirp Control. An uninterrupted linear frequency sweep of either 50 kHz/s or 100 kHz/s is used. With a sweep rate of 100 kHz/s and an analysis range of 333 1/3 Hz the oblique range window has a width of :-

$$\Delta t_{window} = (f_T - f_R)_{max}/k_B \qquad \text{from equn. (2.5)}$$

$$= (333\ 1/3)/100.10^3$$

$$= 3\ 1/3 \text{ ms} \qquad (2.9)$$

For oblique incidence synchronization three timing slip rates of 1 , 10 and 100 ms/s are provided. System timing can either be advanced or retarded at the rate selected.

2.4 "Normal" recording Programmme - Grahamstown and Sanae

Parameter	Vertical	Oblique
Frequency range	0.5 - 15.0 MHz	2.0 - 30.0 MHz
Sweep Rate	50 kHz/s	100 kHz/s
Sweep Duration	4 m 50 s	4 m 40 s
Peak transmitter power	8 W	8 W
Average transmitter power	3 W	8 W
Spectrum Analyser analysis range	0 - 333.3 Hz	0 - 333.3 Hz
Vertical virtual height range	0 - 1000 km	-
Oblique range window	-	3.33 ms

Ionogram Sequencing

Function	Sweep Start Times in the hour
Vertical	00 , 15 , 30 , 45
Oblique	05 , 20 , 35 , 50
Off	10 , 25 , 40 , 55

Vertical incidence recordings were made at both Grahamstown and Sanae at the above times. Oblique incidence records have the Sanae system transmitting and the Grahamstown system receiving. An oblique "calibration" was done each week to determine the offset time t_0 on the Sanae - Grahamstown path.

2.5 The Digital Chirpsounder

The main characteristics of a digital or advanced ionosonde are :-

1. Control of the entire system is by digital computer thereby giving it great flexibility.

2. Numeric analysis of the received data by digital computer either in real time or at a later time from some bulk storage medium such as magnetic tape.

The Barry Research VOS-1C Chirpsounder described in section 2.1 is not a digital ionosonde.

The idea of advanced chirpsounding was conceived by A.W.V.Poole. His PhD. thesis "Advanced Ionospheric Chirpsounding" [Poole , 1983] extends the basic theory of chirpsounding to include measurements of the group range and phase velocity of the reflection point as well as the amplitude , arrival angle and polarization mode of the reflected energy. The thesis shows that all the above measurements can be evaluated from measured phase differences, these being well approximated by differences in the phases of the discrete Fourier frequency components of the difference frequency f_D. Allon Poole's thesis therefore details the theory behind and numeric analysis of the received chirp signal and presents initial results showing system capabilities.

To perform the measurements listed above extensive changes had to be made to the standard Barry Research Chirpsounder:-

1. The Chirpsounder control unit , Logchirp Control , was replaced by an M6809 based microcomputer plus additional hardware. The hardware and software design of this new Controller are the subject of this thesis.

2. Two phase matched receivers were purchased and incorporated into the system to allow simultaneous phase measurements on two different receive antennae. The matched receivers contain digitally switchable attenuators that can vary the gain in 2 dB steps over a total range of 110 dB, allowing the signal amplitude to be measured.

3. A hardware Fast Fourier Transform (FFT) analyser [Fisher , 1979] performs the task of simultaneously sampling and transforming the outputs of the dual receivers. The FFT analyser provides two sets of discrete complex amplitudes from which the phases can be calculated. The analyser also computes the power spectrum. It was designed and built at Rhodes University and is the subject of an MSc. thesis entitled "A Real Time Fast Fourier Transform Analyser" by J.S.Fisher , 1979.

4. A Data Capture System (DCS) based on an M6809 microcomputer receives data from the Controller , receiver AGC and FFT Analyser. Ordered data are stored on magnetic tape for later analysis by mainframe computer. The DCS, also designed and constructed at Rhodes University , is described in detail by Poole , 1983.

A simplified block diagram of the modified Chirpsounder is given in figure (2.5).

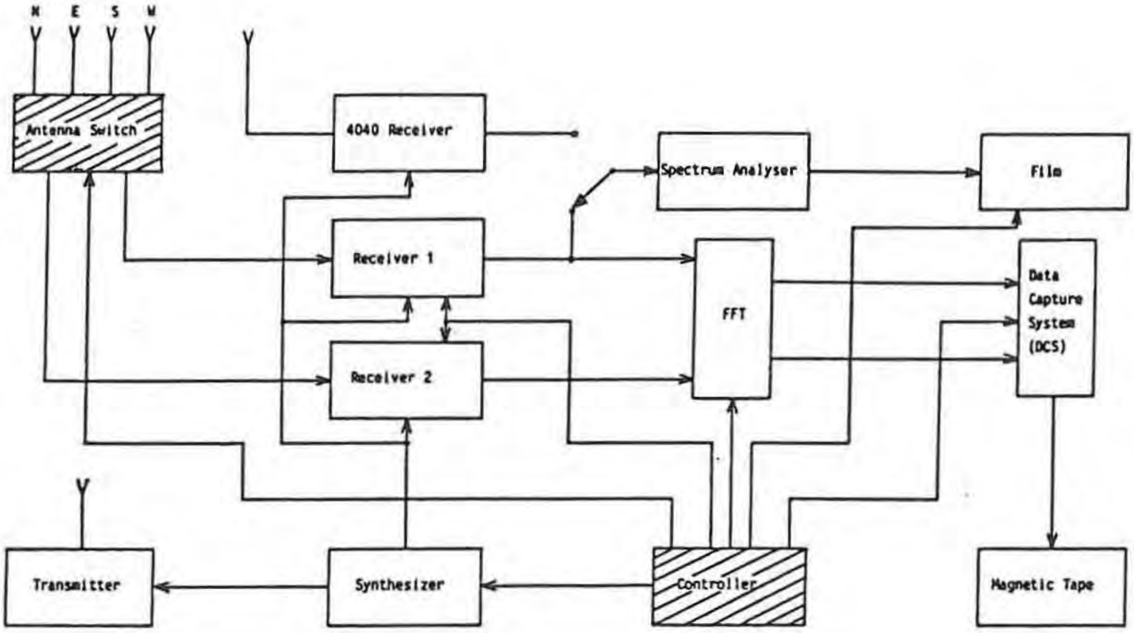


Figure (2.5) Simplified System Block Diagram

The shaded blocks were designed and constructed by the author.

CHAPTER 3

CONTROLLER DESIGN CONCEPTS

3.1 Basic Requirements

The new Controller's basic requirement was to have all the vertical incidence , oblique incidence and programming capabilities of the Barry Research Logchirp Control as described in Chapter 2. In addition to this it was to allow far more sophisticated control of system functions to enable the simultaneous evaluation of group range and phase velocity of the reflection point as well as the three-dimensional angle of arrival and polarization mode of the received signal.

These parameters were to be determined from phase differences obtained by separating the two received signals in time , frequency and orientation. To separate the received signals in time and in frequency a new structure of synthesizer control was required. This structure is described in section 3.2. Separation in space and orientation required the development of an antenna switch that connects the two receivers to the appropriate antennae.

The ability to operate in a chirp mode (non-zero basic rate) with an overall sweep rate of zero was also required to allow the evaluation of all the above parameters at a fixed frequency. This type of measurement has been termed a "Stationary Ionogram".

The system was also required to operate in the so called "Stationary Doppler" mode in which both basic sweep rate and overall sweep rate are zero and a fixed frequency is transmitted. Ionospheric vertical motion is then detected by the Doppler frequency change of the received signal.

The new Controller was to have a minimum of hardware bound restraints with software defining and controlling system operation. A much more flexible system of sequencing ionograms was also required.

Finally , the new Controller was to be integrated into the existing system in such a way that it could be unplugged and easily replaced by the Barry Research Logchirp Control if so desired.

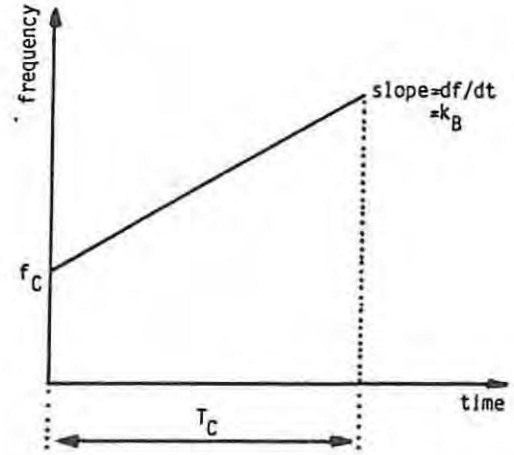
3.2 Ionogram Structure

The frequency versus time structure developed for the Digital Chirpsounder is described below.

A Cell

The "cell" is the fundamental building block of this structure. During a cell the transmitter frequency can increase at some linear rate called the "basic rate" (k_B) or it can remain constant ($k_B = 0$). The following parameters are associated with each cell of period T_C :-

1. A frequency offset Δf which is added to the "sounding" start frequency - see figure (3.2).
2. Film drive either enabled or disabled
3. Dual receiver antennae selection.

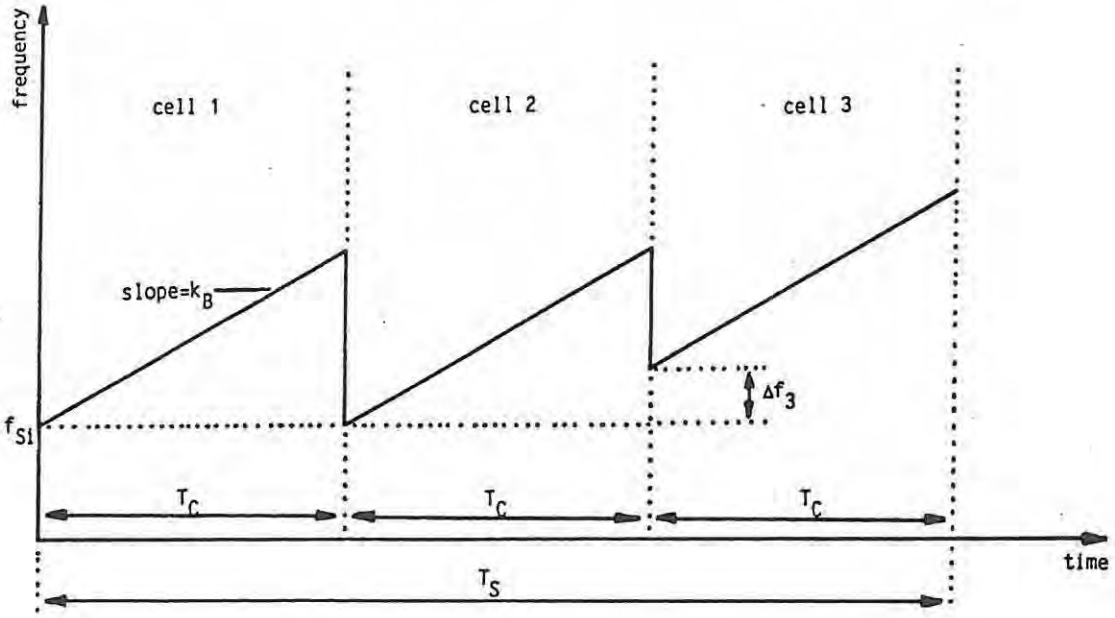


f_C = cell start frequency
T_C = cell period
k_B = basic rate

Figure (3.1) A Cell

A Sounding

A "sounding" consists of a number of adjacent cells which together allow the evaluation of certain ionospheric parameters at a specific (characteristic) frequency. An n-cell sounding consists of n adjacent cells numbered from 1 to n. Cell start frequencies within a sounding are calculated by adding the cell offset frequency to the sounding start frequency.



This example shows a 3-cell sounding (n = 3).

f_{Si} = sounding i start frequency

T_C = cell period

T_S = sounding period

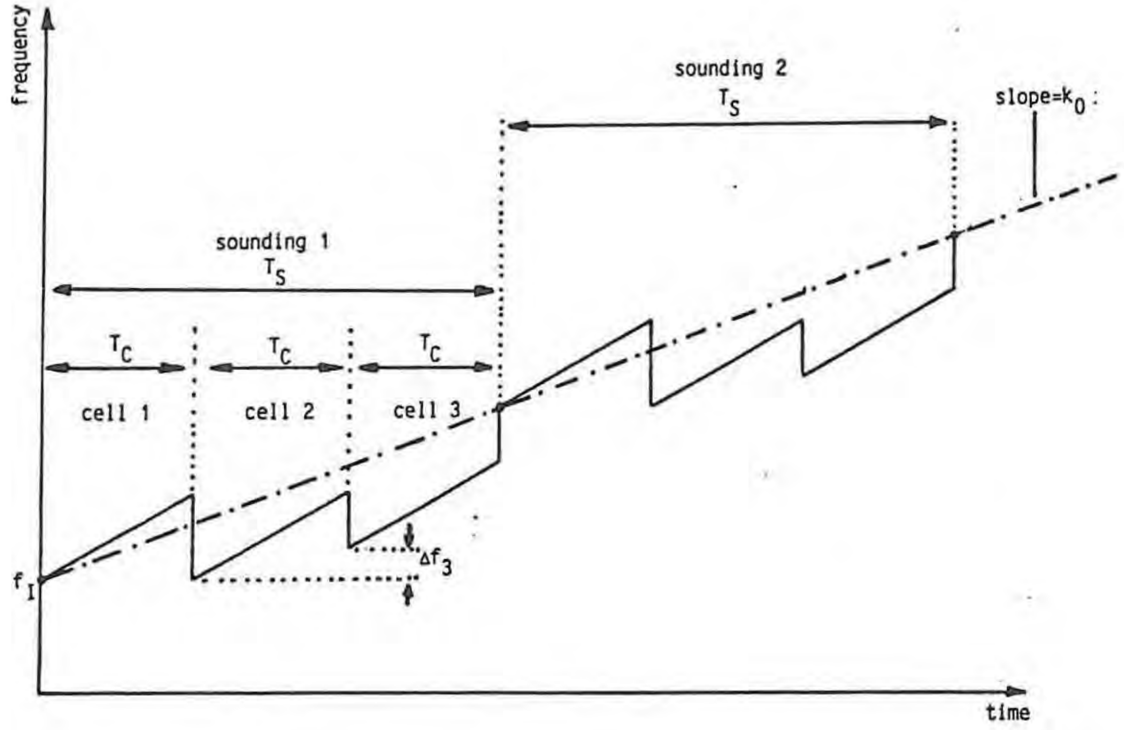
k_B = basic rate

Δf_3 = cell 3 offset frequency

Figure (3.2) A Sounding

An Ionogram

An "ionogram" consists of a number of consecutive soundings whose characteristic (or sounding start) frequencies are related by some linear or logarithmic rate of change.



An ionogram structure having 3-cell soundings related by the linear overall sweep rate k_0.

f_I = ionogram start frequency

T_C = cell period

T_S = sounding period

Δf_3 = cell 3 offset frequency

k_0 = overall sweep rate , linear frequency scale

Figure (3.3) An Ionogram

3.3 Basic Equations

The symbols used in describing an ionogram are listed below.

	Symbol	Units
f_{Cij}	start frequency of cell j in sounding i	Hz
T_C	cell period (invariant for ionogram)	s
k_B	basic sweep rate	Hz/s
n	total number of cells per sounding	-
f_{Si}	start frequency of sounding i	Hz
T_S	sounding period (invariant for ionogram)	s
Δf_j	cell j offset frequency	Hz
f_I	ionogram start frequency	Hz
f_E	ionogram end frequency	Hz
m	total number of soundings per ionogram	-
T_I	ionogram period or duration	s
k_O	overall sweep rate , linear frequency scale	Hz/s
A	overall sweep rate exp time constant , log freq scale	s^{-1}

The sounding period T_S for an n cell sounding is given by :-

$$T_S = nT_C \tag{3.1}$$

The start frequency of cell j in sounding i is obtained by adding cell j offset frequency to sounding i start frequency. That is :-

$$f_{Cij} = f_{Si} + \Delta f_j \tag{3.2}$$

where j = 1 to n for each i from 1 to m

The ionogram start frequency is thus

$$\begin{aligned} f_I &= f_{C11} \\ &= f_{S1} + \Delta f_1 \end{aligned} \qquad (3.3)$$

For a linear overall sweep rate the start frequency of sounding i is given by :-

$$\begin{aligned} f_{Si} &= f_I + k_0 i T_S \\ &= f_I + k_0 i n T_C \end{aligned} \qquad (3.4)$$

Given the start and end frequencies of an ionogram with a linear overall rate the total number of soundings (m) in the ionogram is obtained by setting i = m and solving for m.

$$f_{Sm} = f_I + k_0 m n T_C$$

Since $f_{Sm} = f_E$, the ionogram end frequency , the equation becomes :-

$$\begin{aligned} f_E &= f_I + k_0 m n T_C \\ m &= (f_E - f_I)/k_0 n T_C \end{aligned} \qquad (3.5)$$

For an exponential overall sweep rate the start frequency of sounding i is given by :-

$$f_{Si} = f_I e^{AiT_S} \qquad (3.6)$$

The total number of soundings in an ionogram with an exponential sweep rate is given by :-

$$\begin{aligned} f_{Sm} &= f_I e^{AmT_S} \\ f_E &= f_I e^{AmT_S} \\ e^{AmT_S} &= f_E/f_I \\ m &= \ln(f_E/f_I)/(AT_S) \end{aligned} \qquad (3.7)$$

The duration of an ionogram with either a linear or an exponential sweep rate is simply :-

$$\begin{aligned} T_I &= mT_S \\ &= mnT_C \end{aligned} \qquad (3.8)$$

3.4 Windowing

The effective receiver bandwidth of 500 Hz defines the upper limit of the frequency difference $f_T - f_R$. Equation (2.6) gives the virtual height of the ionosphere as :-

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

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

$$h'_{min} = 0 \text{ km.}$$
$$h'_{max} = c(f_T - f_R)_{max}/2k_B \quad \text{km.}$$

This range could be adjusted to cover any "window" of 500 Hz by subtracting a fixed offset f_0 from the difference frequency.

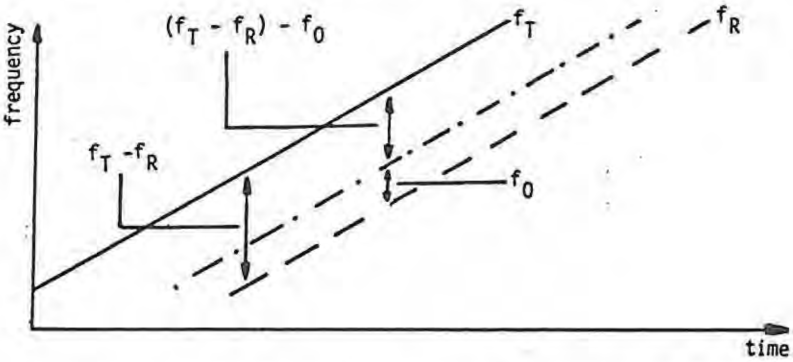


Figure (3.4) Windowing , frequency vs time.

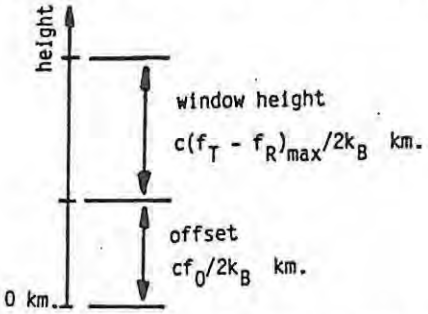


Figure (3.5) Windowing , offset height and window height.

Then
$$h' = c((f_T - f_R) - f_0)/2k_B$$
$$= c(f_T - f_R)/2k_B - cf_0/2k_B \quad (3.9)$$

The maximum value of the first term , that is $c(f_T - f_R)_{max}/2k_B$, defines the window height with $cf_0/2k_B$ km being the offset from 0 km.

Equation (2.6) states that $h' = c(f_T - f_R)/2k_B$ from which h'_{max} was obtained by setting $(f_T - f_R)$ equal to the receiver bandwidth which is fixed.

It follows that doubling the basic rate k_B results in h'_{max} being halved. The 500 Hz baseband signal range is thus spread over half the height range. The frequency change per cell can be kept constant by halving the cell period T_C each time the basic rate is doubled.

Basic rate changes with windowing (equation (3.9)) therefore affect both the height of the window being analysed as well as the height offset from 0 km. It was decided to develop hardware such that the offset term $cf_0/2k_B$ could be set in multiples of half the window height.
The minimum offset frequency f_{0min} is thus

$$(cf_{0min}/2k_B) = c(f_t - f_R)_{max}/4k_B$$

$$f_{0min} = (f_T - f_R)_{max}/2$$

$$f_{0min} = 250 \text{ Hz} \qquad (3.10)$$

This means that the higher the basic rate the smaller the offset step size becomes. Despite the obvious limitations of higher and higher basic rates usually coupled with shorter and shorter cell periods , windowing was incorporated into the system for experimental purposes. The intention was to use windowing for ionospheric E- and F-region studies.

3.5 Stationary Doppler

The frequency versus time curve for a Chirpsounder, figure (1.3a), shows that at any time t the instantaneous transmitter frequency is higher than the frequency of the received signal.

With fixed-frequency operation (basic and overall sweep rates zero) this would occur naturally if the reflection point moved away from the receiver on the ground. Motion of the reflecting layer toward the receiver would not be detected because the 4040 receiver has an image rejection of 60 dB minimum.

To detect downward motion as well , the "no-motion" signal must be positioned in the centre of the range of Doppler frequencies of interest.

3.6 Controller Initial Development

Hardware

To achieve the flexibility and scope of control as outlined in section in section 3.1 it was apparent from the outset that a microcomputer based system was required. Familarity with the Motorola M6800 8-bit microprocessor contributed to the decision to base the Controller on a South West Technical Products Corporation (SWTPC) 6800 computer system.

The basic system consisting of power supply , mother board , processor board and serial I/O port was modified by the author to fit into a standard 19 inch equipment cabinet to allow it to physically replace Logchirp Control. Twelve double sided 28 pin edge connectors were also mounted in the cabinet for the additional hardware that was required.

To allow the fast evaluation of the exponential functions of equations (3.6) and (3.7) a SWTPC calculator board (MPN) was included in the system.

Data and command entry via front panel switches was rejected as being far too limiting and clumsy. Primary communication with the system by VDU was envisaged with hardcopy being available by replacing the VDU with a teletype.

Software

For software development a 5 1/4 inch dual floppy disk drive was used with the M6800 computer. The high level language ABASIC was investigated but proved too unwieldy for control purposes. Programming was therefore initially done in assembler. A number of the hardware interface boards were tested using these programs. A large part of the system program was also written in assembler. Both the writing of the source code in assembler and the assembly process proved to be very time consuming.

Investigation of the programming language FORTH showed that it was an ideal method of software development. Complex programmes can be written quickly in a structured manner in FORTH. The threaded code aspect of the language results in low memory usage but can add up to 20% to the execution time when compared with straight assembler coding. In most situations this overhead presents no problems but if maximum execution speed is required assembler coded routines can be easily incorporated into FORTH.

At this time an M6809 processor board became available for the SWTPC computer. The 6800 system was upgraded to 6809 because the internal architecture of the 6809 microprocessor lends itself to a very efficient implementation of the FORTH language. All the Controller software was subsequently written in FORTH.(See Appendix D)

3.7 Interchangeability - Controller/Logchirp

Special interconnection cables between the Controller and Amplifier T/R and Controller and Camera Interface were designed into the system to allow easy swapping of control units. This proved to be extremely useful during the initial test stages since the system could be run under Logchirp control during modifications to the new Controller resulting in little data loss.

CHAPTER 4

CONTROLLER HARDWARE

4.1 Introduction

Two comprehensive cross-referenced manuals describing the hardware have been prepared by the author. The prime object was to document the system fully for the operator at the South African Antarctic base , Sanae. The two manuals allow the simultaneous reference to a circuit diagram and its corresponding circuit description.

"Vertichirp Controller Hardware - Manual 1 ; Description of Printed Circuit Boards" is included as Appendix A and describes each board in the form :-

1. Introduction
2. Circuit development (if applicable)
3. Circuit description
4. Control data table

"Vertichirp Controller Hardware - Manual 2 ; Circuit Diagrams , Component Location Diagrams and Wire Lists" is included as Appendix B and contains :-

1. Circuit Diagrams
2. Component Location Diagrams
3. Wire Lists

This documentation format devised by the author is used to completely describe each of the printed circuit boards.

This chapter will contain more general information on the hardware.

4.2 Block Diagram of Modified Vertichirp Sounder

Figure (4.1) expands on the simplified block diagram of figure (2.5) by including all the clock and control lines as well as the RF signal routing. All the shaded blocks were designed and constructed by the author.

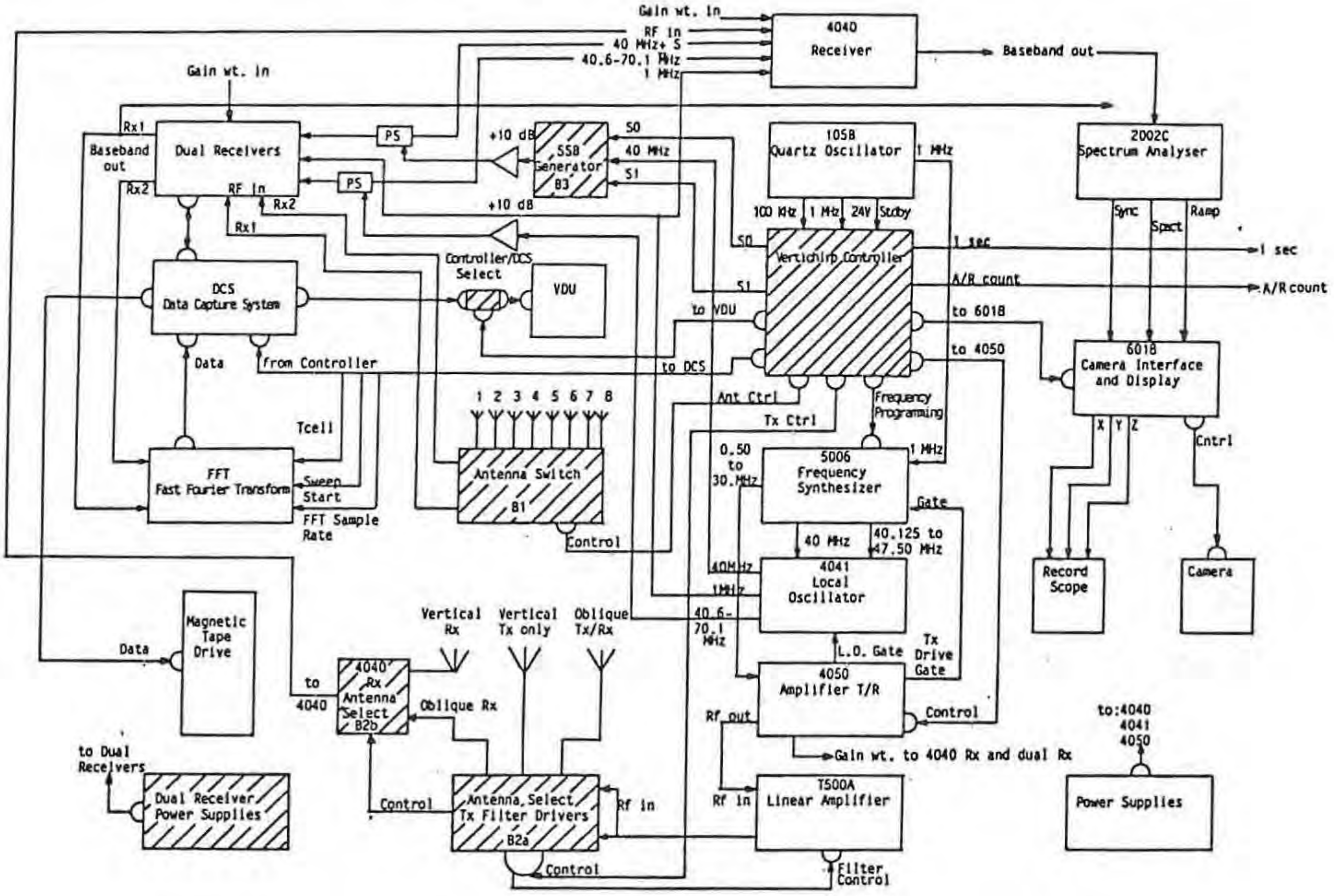


Figure (4.1) Block Diagram - Modified Vertichirp Sounder

The basic clock frequencies of 100 kHz and 1 MHz are generated by the 105B Quartz Oscillator which also provides a battery backup +24 VDC supply.
The new Vertichirp Controller oversees the entire system operation. It provides programming information to the 5006 Frequency Synthesizer and control signals to the 4050 Amplifier T/R , the antenna selection relays , the antenna switching circuitry , the 6018 Camera Interface/Display and the FFT Analyser. It also sends data to the to the Data Capture System computer.

Two independent receiver signal paths exist. The first consists of the 4040 Receiver , 2002 Spectrum Analyser and FRS-1 Film Recording System. Because the T500A Linear Amplifier does not support electronic T/R switching the RF transmitted on the vertical Tx antenna is received on a separate vertical Rx antenna switched through to the 4040 Receiver by B2b , the 4040 Rx antenna select circuit. The receiver baseband output is analysed by the 2002 Spectrum Analyser before being recorded on film.

The second signal path consists of two phase-matched receivers , an FFT Analyser and a Data Capture System (DCS) which stores the data on magnetic tape. B1 Antenna Switch allows the connection of one of eight receive antennae to either of the two receivers.

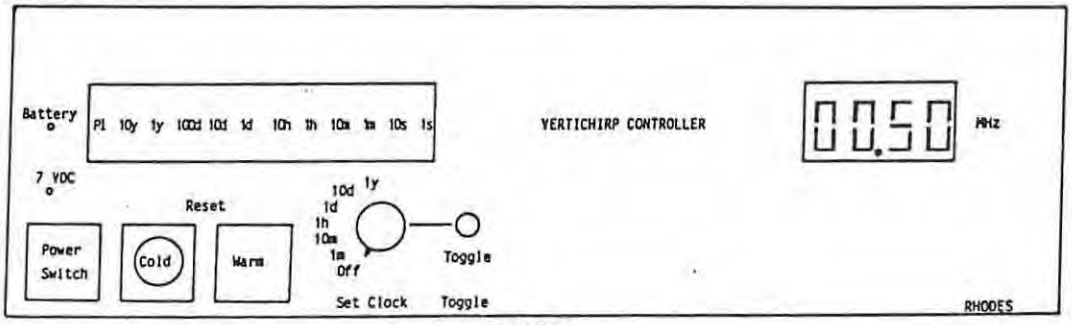
The baseband outputs of the two receivers are analysed by the FFT Analyser and DCS computer which stores partially processed data on magnetic tape for further analysis by mainframe computer ’n Poole , 1983 °.

Note that the output of one of the two phase-matched receivers can be recorded on film by using its baseband output as the input to the 2002 Spectrum Analyser instead of the 4040 Receiver output. The 4040 Receiver is however more sensitive than either of the matched receivers and is therefore preferred for the film record.

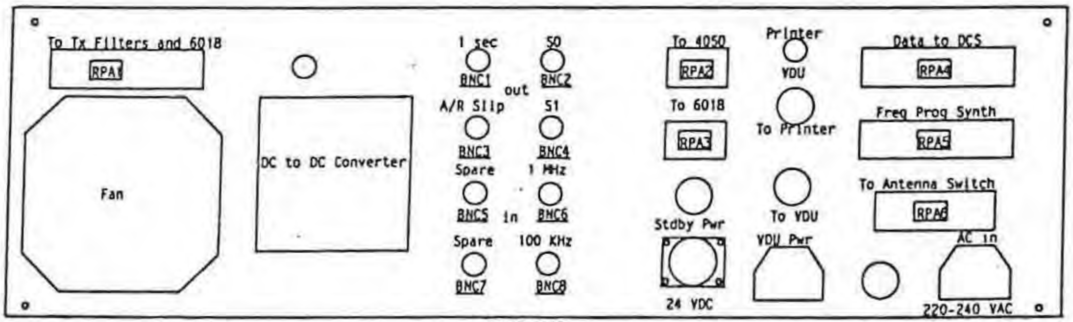
To run the 4040 Receiver and the phase-matched receivers together the 40 MHz and 40.6 to 70.1 MHz local oscillators are amplified before driving in-phase power splitters. The 1 MHz local oscillator signal , being a TTL signal , drives both dual receivers and 4040 Receiver directly. The gain-weight signal from the 4050 Amplifier T/R also drives both receiver sets.

The SSB Generator B3 allows the 40 MHz local oscillator signal to be offset for windowing and stationary-Doppler measurements.

A single VDU is used in the system and can be switched between Controller and Data Capture System computers.



FRONT PANEL



REAR PANEL

RPAn = Rear Panel Amphenol n

Figure· (4.2) Vertichirp Controller Front and Rear Panels

4.3 Location of Vertichirp Controller Circuit Boards

The SWTPC MP-B2 Motherboard , modified for use with the MP-09 Processor board , is mounted on the right hand side of the Controller chassis. The I/O half of this board is horizontal as usual but the processor and memory slots are positioned vertically. This means that the processor, memory and real-time clock boards plug into these slots from the front of the Controller with the component side down. See figure (4.3).

Behind the I/O ports on the rear panel is a cooling fan , the DC to DC converter , the diode PCB and the filter PCB. In front of the CPU and memory boards and secured from the top is the frequency display board , FD.

The front left side of the chassis carries the MP-P power supply. The power switch , "cold" and "warm" reset buttons and the real-time clock display , RTCD are also on the left.

Controller hardware PC board +5V regulators with corresponding edge connectors occupy the remaining space. The Controller PC board slots are numbered from the rear from PC1 to PC12.

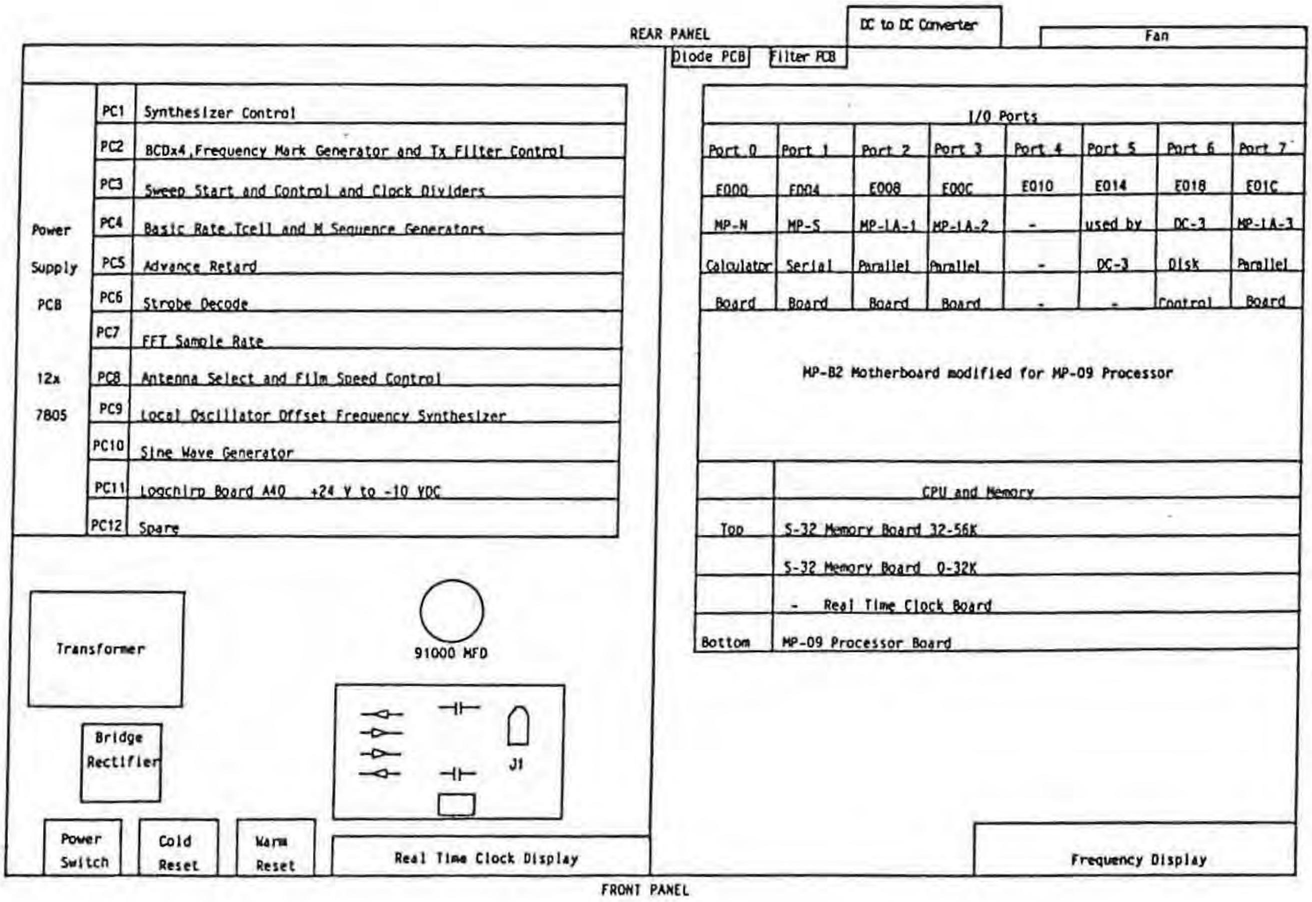


Figure (4.3) Location of Controller Circuit Boards

4.4 Controller Power Supplies

The MP-P power supply provides unregulated 7 to 8 VDC to both the computer and the Controller hardware. Each SWTPC board has its own +5V regulator. Controller edge connectors PC1 to PC12 each have their own chassis-mounted +5V regulator. A +24V battery is used to provide a failsafe +5VDC and -10VDC supply. -12V is regulated to -5VDC for the sine wave generator PC10.

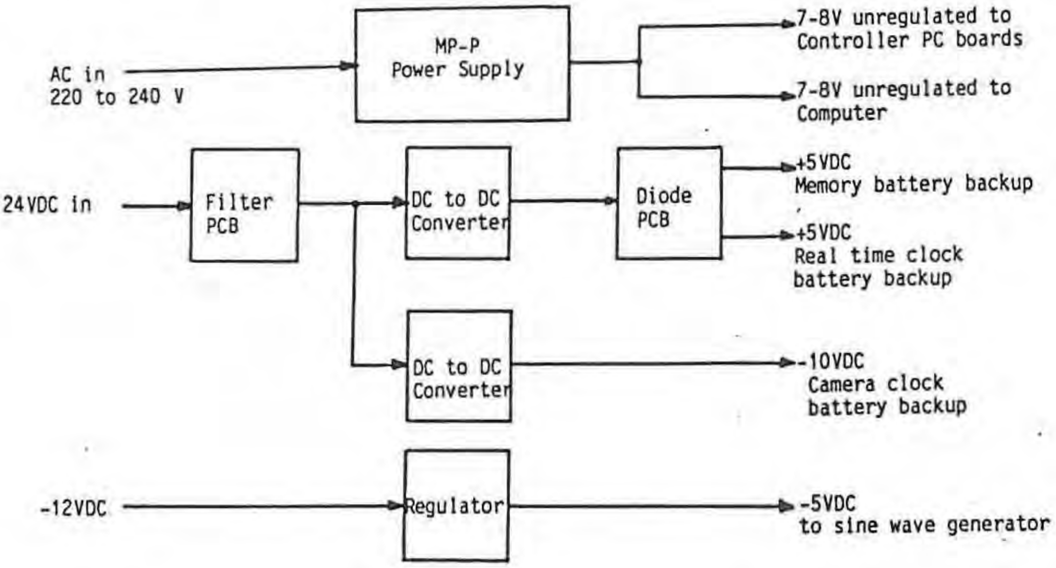
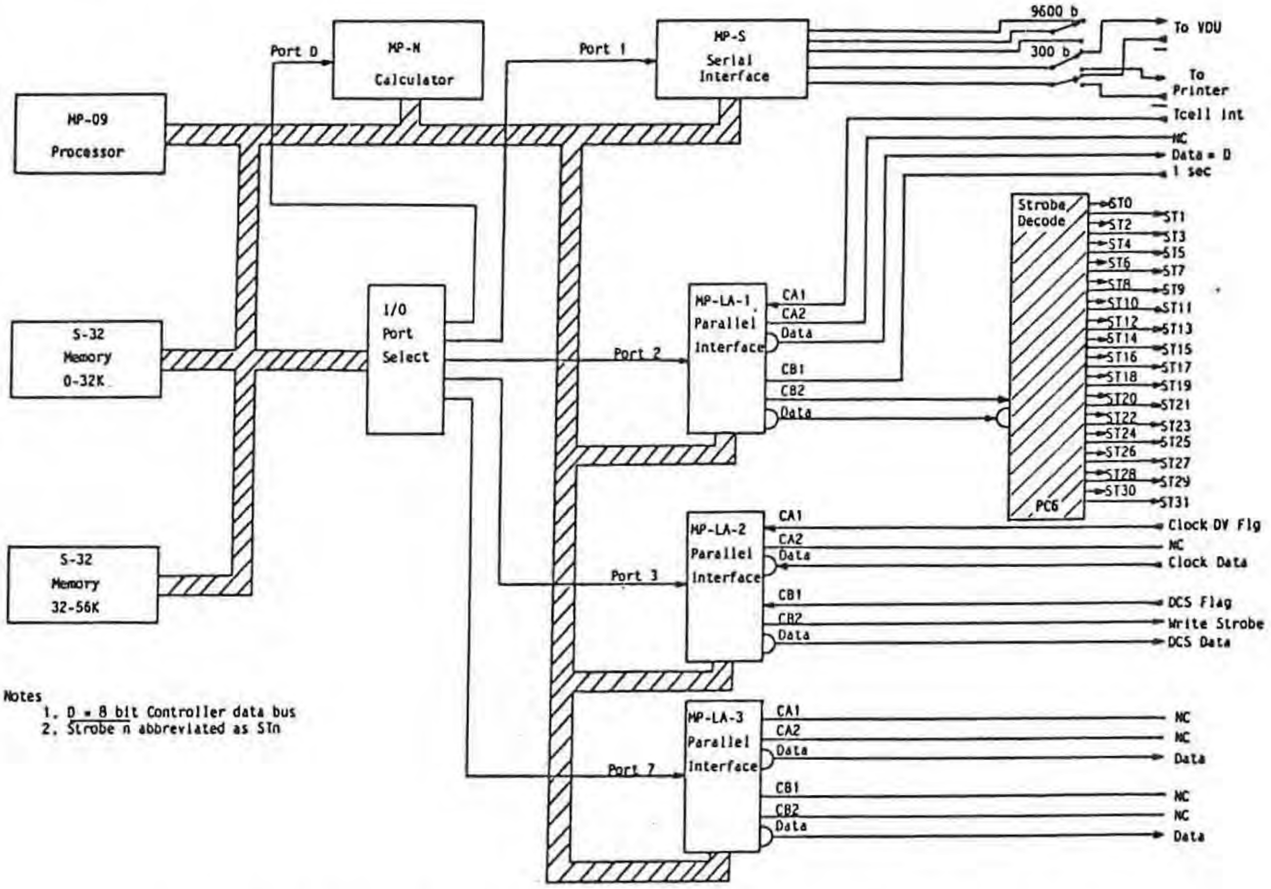


Figure (4.4) Block Diagram - Controller Power Supplies

4.5 Vertichirp Controller Block Diagram 1 - Computer and Strobe Decode

Figure (4.5) shows the commercial SWTPC computer system together with the Strobe Decode board (PC6) designed by the author. A three-pole change-over switch connects the MP-S Serial Interface output and input lines to either a VDU at 9600b or to a printer at 300b.

Parallel Interface Adapter (PIA) MP-LA-1 handles the Tcell pulse interrupt on its CA1 input line and the real time clock interrupt on its CB1 line. The A side data outputs of MP-LA-1 drive the Controller 8-bit data bus which is called "D" in the circuit diagrams. The B side data outputs of MP-LA-1 together with the CB1 output configured as a "write strobe with E restore" (WRE) are used on the Strobe Decode board PC6 to generate all the strobe signals used in the Controller.



Notes
1. D = 8 bit Controller data bus
2. Strobe n abbreviated as STn

Figure (4.5) Vertichirp Controller Block Diagram 1 - Computer and Strobe Decode (PC6)

The A side of parallel interface MP-LA-2 is used to input the real-time clock data with the data valid flag on the CA1 input. The B side of MP-LA-2 is used to output ionogram control parameters to the Data Capture System (DCS). The CB1 line is used as an acknowledge flag from the data capture system. MP-LA-3 is a spare parallel interface.

Computer Modifications

One of the S-32 memory boards was modified to operate in the address space from 32K to 56K. Both memory boards were provided with a battery supply.

Controller Data Bus(D) and Strobe Decode (PC6)

The Controller hardware was designed in such a way that it requires that data from the computer be latched to select one of a number of possible outputs of a particular controlled parameter.

The method used for parameter selection makes use of both sides of a PIA. Data written to the A side are latched by the PIA and distributed to the Controller hardware. The Controller 8-bit data bus is designated "D" in the system block diagrams. Data written to the PIA B side together with the CB2 output configured as a "write strobe with E restore" are used by the Strobe Decode board PC6 to generate the strobes required for data latching and system control. All the strobe signals are active low but have been called STn for clarity in the circuit diagrams. The strobe number n is decimal unless otherwise specified.

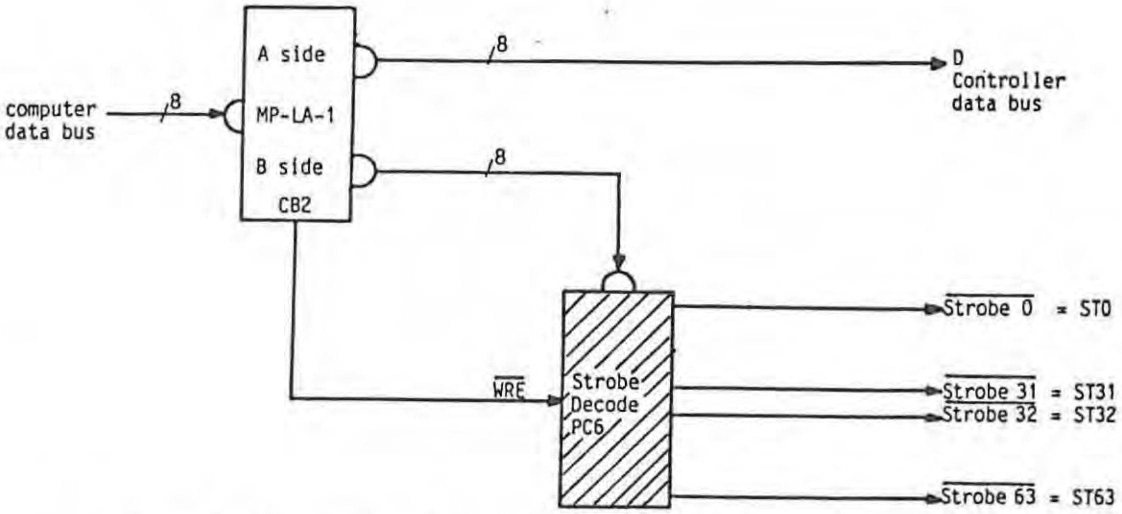


Figure (4.6) Block Diagram - Controller Data Bus (D) and Strobe Decode (PC6)

4.6 Vertichirp Controller Block Diagram 2 - Interface Circuitry

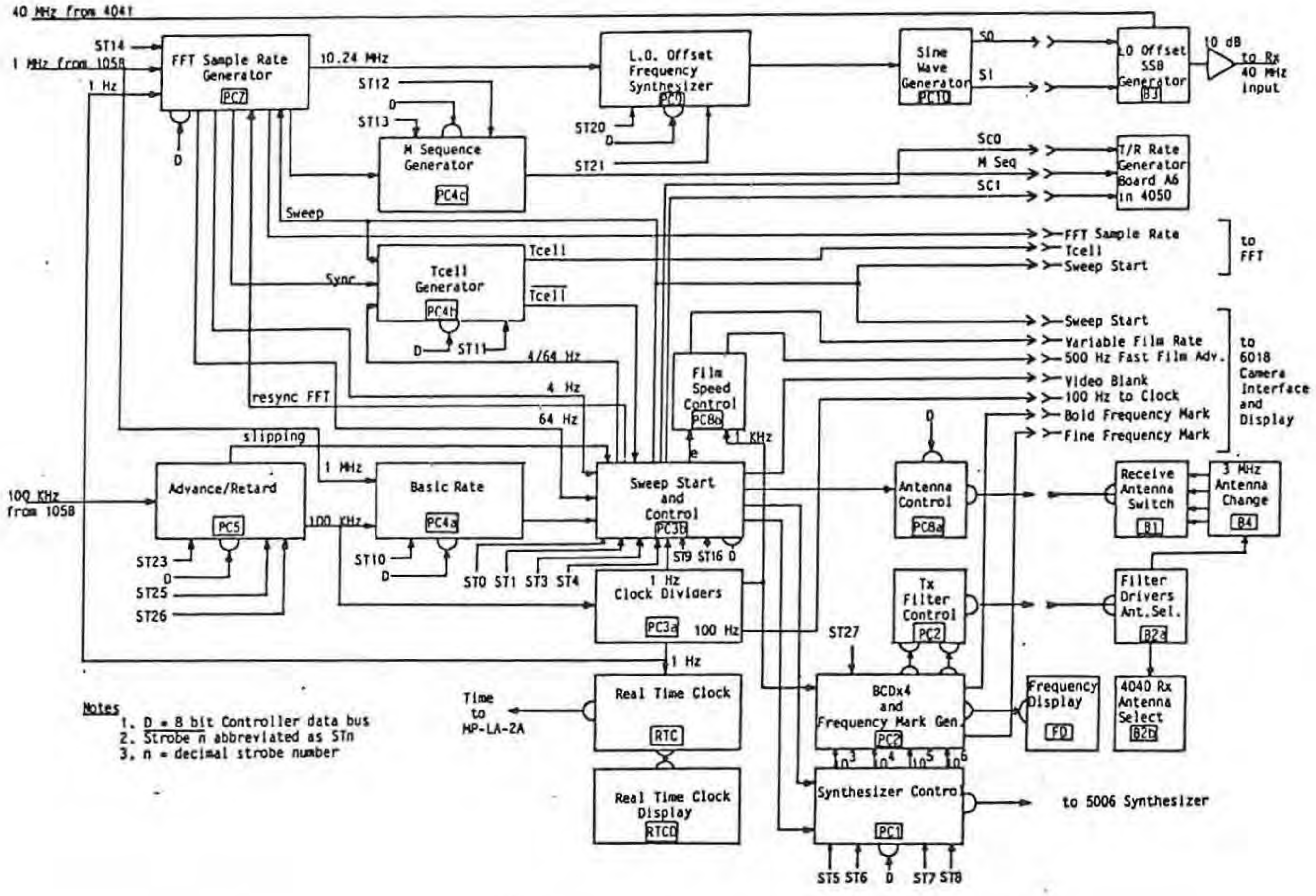


Figure (4.7) Vertichirp Controller Block Diagram 2 - Interface Circuitry

A very brief description of the block diagram of figure (4.7) is given below.

The 100 kHz signal from the 105B Quartz Oscillator can have pulses inserted or deleted by the Advance/Retard board (PC5). This allows the system timing to be changed in steps of 10 microseconds. The Clock Dividers (PC3a) divide the 100 kHz input signal down to 100 Hz for the film recording system and to 1 Hz for the Real-Time Clock (RTC) which displays the system time on the front panel display (RTCD). The real-time clock can also be read by the computer via parallel port MP-LA-2A. The advance/retard , clock divider and real-time clock circuits are all provided with a battery supply.

The Basic Rate board (PC4a) generates a number of different basic rate clock signals. One of these is selected to clock the Synthesizer Control board (PC1) which provides BCD programming information to the 5006 Frequency Synthesizer.

The FFT Sample-Rate Generator (PC7) outputs 10.24 MHz to the L.O. Offset Frequency Synthesizer (PC9) which generates the clock for the Sine Wave Generator (PC10). PC10 outputs two sine waves S0 and S1 which have a fixed phase difference of either 90^0 lag or 90^0 lead between them at frequency f. L.O. Offset SSB Generator (B3) outputs either the lower sideband or upper sideband of the 40 MHz local oscillator signal offset by frequency f.

2048 Hz from the FFT Sample-Rate Generator (PC7) clocks the M-sequence Generator (PC4c). The output M-sequence which has a programmable duration is modified by the T/R Rate Generator board to produce the signals used in transmit/receive switching.

The FFT Analyser sample-rate signal is selected from eight signals ranging from 1024 Hz to 8 Hz generated on PC7.

Either 4 Hz or 64 Hz from PC7 is selected to go to the Tcell Generator board (PC4b). The time between Tcell pulses determines the cell period T_C. The choice of input frequency to the Tcell Generator board gives two overlapping sets of cell periods , one set being for normal multi-cell vertical ionograms , the other for Stationary-Doppler measurements.

Circuitry is provided to synchronize the FFT rate generator with the 1 Hz signal from the Clock Dividers (PC3a).

The Sweep-Start and Control board (PC3b) synchronizes ionogram start times with the falling edge of the 1 Hz system clock. PC3b also contains two 8-bit registers , Control Register A and Control Register B , which define system operation as well as "power-on reset" circuitry.

PC2 multiplies the synthesizer BCD programming information by four to obtain the transmitter frequency which is then displayed on the front panel Frequency Display board (FD). PC2 also generates bold and fine frequency mark signals for the camera system and selects the correct transmitter filters via the Filter Drivers in box B2a. Box B2b contains two co-axial relays with associated driver circuitry used for the selection of either a vertical or an oblique antenna as the input to the 4040 receiver.

PC8a provides the control signals to the Receive Antenna Switch (B1) which allows the selection of any of eight receive antennae to either of the two phase-matched receivers. The Film Speed Control (PC8b) generates a film-rate signal that can be changed by the computer.

The following sections explain why each of the functional blocks shown in figure (4.7) were necessary. The circuitry of each function is described in greater detail in block diagram form. All the interface circuits , except for the Real-time Clock and Real-time-clock Display , were designed and constructed by the author. Allon Poole designed and built the clock.

4.7 System Timing Circuits

The four boards used for system timing are discussed here.

Advance/Retard Circuit (PC5)

System timing is derived from the 100 kHz signal from the HP 105B Quartz Oscillator. The Advance/Retard board (PC5) allows the system timing to be changed in 10 microsecond steps by the addition or deletion of pulses from this 100 kHz signal.

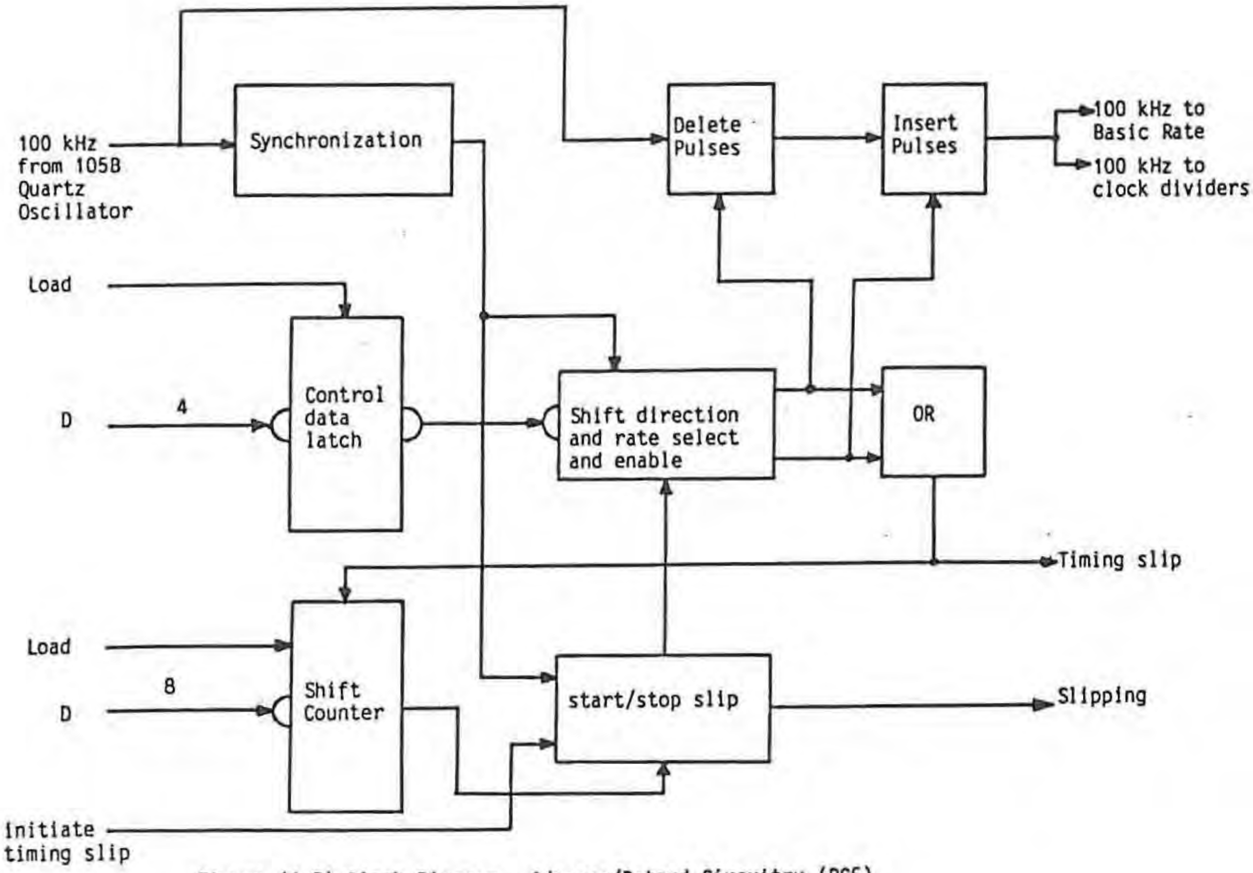


Figure (4.8) Block Diagram - Advance/Retard Circuitry (PC5)

The ability to change system timing is of prime importance for oblique incidence synchronization which depends on the relative stability of the transmitting and receiving ionosonde Quartz Oscillators. Different rates of change of timing are useful in searching for the oblique signal. Timing changes must affect both the ionogram start times as well as the instantaneous frequency of the system if done during a sweep. Both requirements are met by deriving the system timing and the basic rate clock signal from the output of the Advance/Retard board.

Block Diagram - Advance/Retard Circuit (PC5)

Four latched bits from the Controller data bus (D) determine shift direction (advance or retard) and shift rate (1 , 10 or 100 ms/s). An 8-bit number loaded into the counters is decremented by either the advance or the retard signal which adds or deletes pulses from the 100 kHz square wave when shifting is initiated. When the counters reach zero, shifting is disabled.

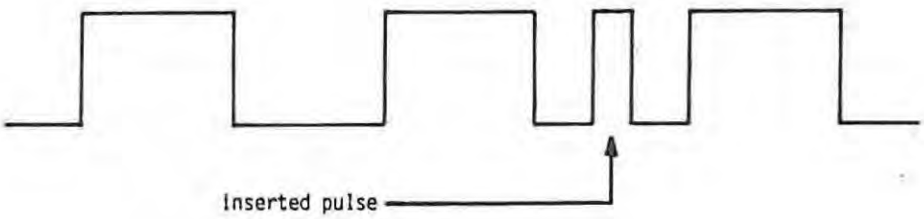


Figure (4.9a) Advance Waveform

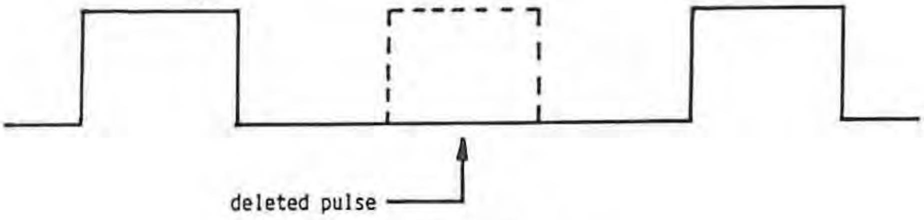


Figure (4.9b) Retard Waveform

Clock Dividers (PC3a)

The 100 kHz signal from the Advance/Retard circuit board (PC5) is divided to produce 1 kHz for the film-speed board (PC8a) , 100 Hz for the clock in the 6018 Camera Interface/Display and 1 Hz for the system real-time clock and sweep-start circuitry.

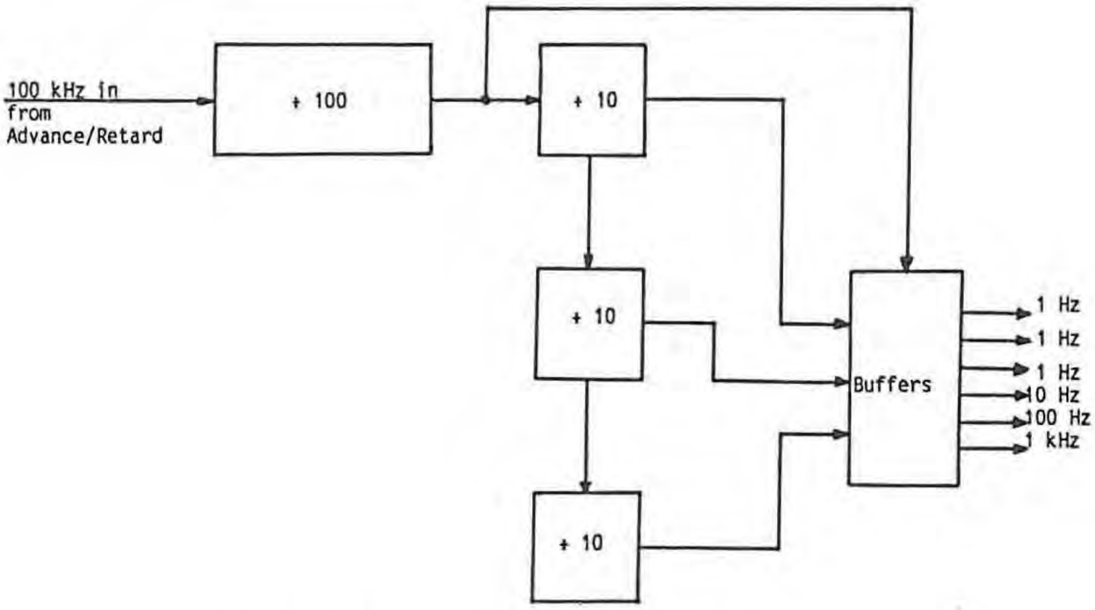


Figure (4.10) Block Diagram - Clock Dividers (PC3a)

The standby battery supplies +5V to all the clock dividers which produce 1 kHz , 100 Hz , 10 Hz and 1 Hz from the 100 kHz input signal.

Real Time Clock (RTC)

The real time clock (designed by A. Poole) provides place , year , year day-number , hours , minutes and seconds to both the clock display (RTCD) and also to the control computer. It consists of a battery-supplied clock multiplexer board clocked by 1 Hz from the clock dividers.

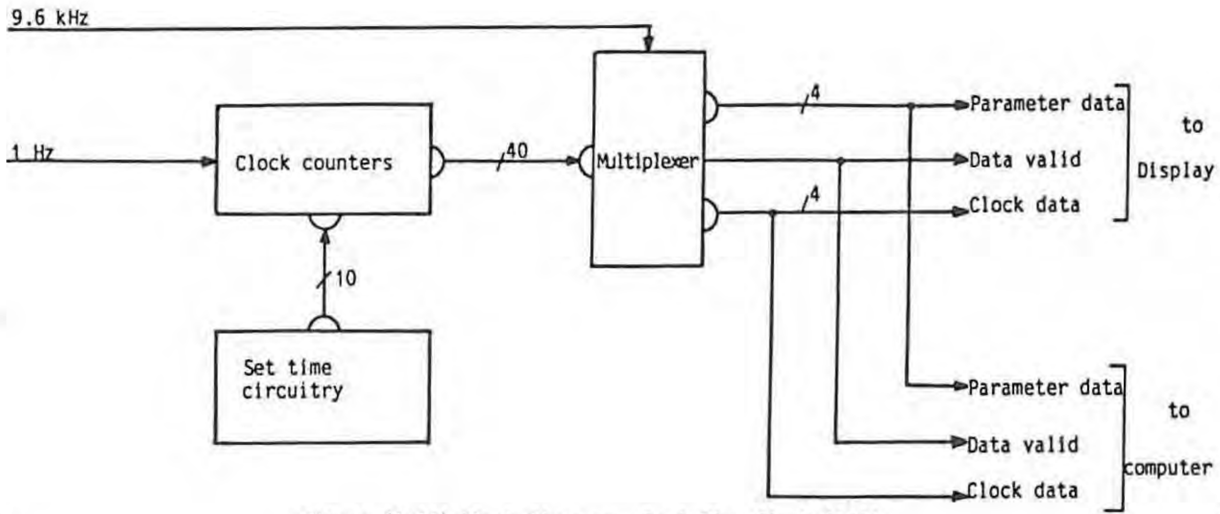


Figure (4.11) Block Diagram - Real Time Clock (RTC)

1 Hz from the clock dividers increments the time in the clock counters which is initially set using the switches on the Controller front panel. The 4-bit BCD clock data are associated with a 4-bit parameter number that ranges from 0 to 9. The data-valid pulse is used for demultiplexing.

Parameter number	0	1	2	3	4	5	6	7	8	9
Clock data	S1	S10	M1	M10	H1	H10	D1	D10	D100	Y1

Real Time Clock Display (RTCD)

This board (designed by A. Poole) demultiplexes and displays the 10-digit clock data from the real-time clock board. Place identification and a tens-of-years digit are added to the display using a switch array on this board.

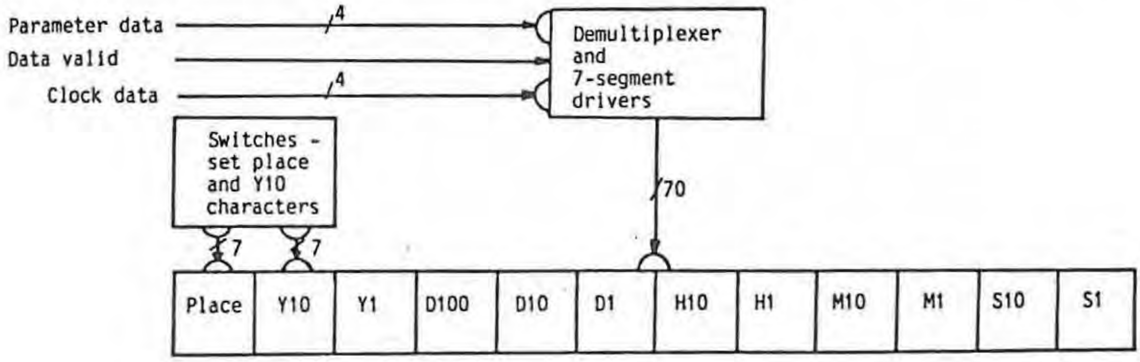


Figure (4.12) Block Diagram - Real Time Clock Display (RTCD)

4.8 Synthesizer Control

The Basic-Rate and Synthesizer Control boards are described in this section.

Basic-Rate Circuitry - (PC4a)

The original system under Logchirp Control provided one fixed basic rate of 50 kHz/s for vertical operation and a choice of either 50 or 100 kHz/s basic rate for oblique operation. Equation (2.6) gives the virtual height of the ionosphere as :- $h' = c(f_T - f_R)_{max}/2k_B$

Note that h' is inversely proportional to the basic sweep rate k_B.

The Basic-Rate board (PC4a) provides a number of different basic rates besides 50 and 100 kHz/s. They are intended for experimental vertical and oblique operation with the most important set being 25 , 50 ,100 , 200 , 400 and 800 kHz/s. In generating these signals two other basic-rate sets were produced. These are 40 and 80 kHz/s and 250 , 500 , 1000 , 2000 and 4000 kHz/s , the latter set probably not being of much practical value. A basic rate of zero is used for Stationary-Doppler recordings.

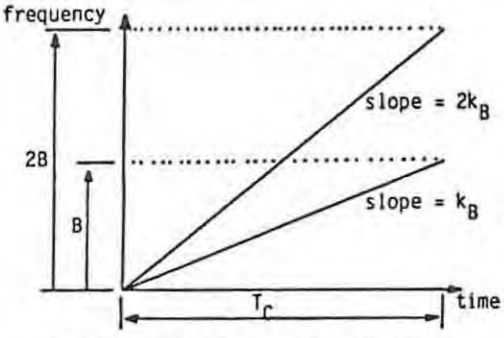


Figure (4.13) Chirp Signal Bandwidth

Note that doubling the basic rate doubles the chirp bandwidth B for a fixed cell period T_C , fig.(4.13). This will affect height resolution particularly near a critical frequency where the electron density changes rapidly. The same frequency change per cell can be obtained by halving the cell period but this results in less energy being transmitted for that measurement.

The 5006 Frequency Synthesizer is directly programmable from 40.125 to 47.5 MHz in 1 Hz increments by supplying it with a 7-decade BCD number. The 0.5 to 30.0 MHz range is obtained by multiplying this signal by 4 and subtracting 160.0 MHz. If the programmed frequency f_P is given by :-

$$f_P = 40.0 \text{ MHz} + \text{BCD number} \qquad (4.1)$$

the transmitted signal f_T can be written as :-

$$f_T = 4f_P - 160.0 \text{ MHz}$$

$$f_T = 4(40.0 + \text{BCD number}) - 160.0 \text{ MHz}$$

$$f_T = 4(\text{BCD number}) \qquad (4.2)$$

The 0.5 to 30.0 MHz range is therefore covered in 4 Hz increments.

To generate a frequency ramp or "chirp" signal with a slope of k_B kHz/s the synthesizer programming counters on PC1 must be clocked at $k_B/4$ kHz. This frequency is generated by the Basic-Rate board (PC4a). Latched data selects the input frequency and the output basic-rate clock.

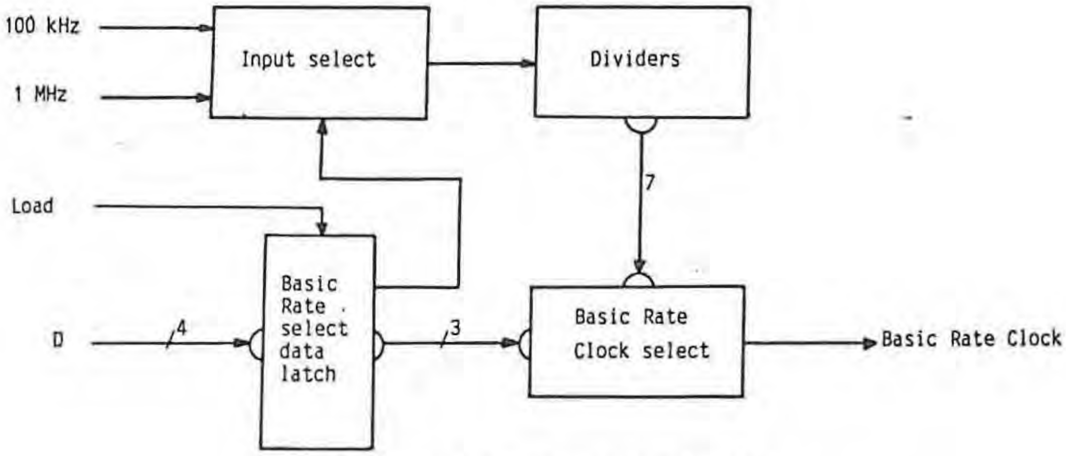


Figure (4.14) Block Diagram - Basic Rate (PC4a)

Synthesizer Control - (PC1)

The 5006 Frequency Synthesizer produces three swept-frequency outputs (40.125 - 47.5 MHz , 0.5 - 30.0 MHz and 0.125 - 7.5 MHz) and four fixed frequency outputs (100 kHz , 1 MHz , 5 MHz and 40 MHz).

Frequencies in the range 40.125 - 47.5 MHz are programmed with a resolution of 1 Hz by PC1 which supplies the synthesizer with a 7-decade BCD number. Subtraction of 40 MHz from this signal gives the 0.125 - 7.5 MHz output. The transmitted signal f_T (0.5 - 30.0 MHz) is related to the BCD number according to equation (4.2) , $f_T = 4($ BCD number $)$, and has a frequency resolution of 4 Hz.

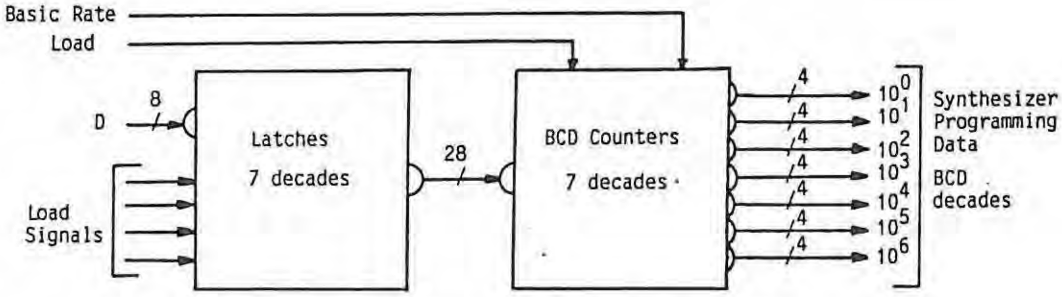


Figure (4.15) Block Diagram - Synthesizer Control (PC1)

Latches on the Synthesizer Control board (PC1) are sequentially loaded with 7 decades of programming data from the Controller bus. These 28 bits are parallel loaded into the BCD counters which program the synthesizer. The counters can be incremented by the basic-rate clock.

4.9 FFT Sample-Rate Generator (PC7)

The Fast Fourier Transform (FFT) Analyser [Fisher , 1979] was designed to sample and simultaneously transform two time-varying signals band-limited to 500 Hz to provide two sets of discrete complex amplitudes from which signal phases could be calculated.

The analyser performs a 1024 point transform and outputs 512 cosine , 512 sine and 512 power spectral amplitudes for each channel as there are 512 positive frequency points in the transform.

A sample rate of 1024 Hz satisfies the minimum sample frequency condition of the Sampling Theorem and also gives a resolution of 1 Hz in the frequency domain. A higher sample rate would allow the faster accumulation of 1024 points but would reduce frequency domain resolution. The FFT Sample-Rate generator was designed with 1024 Hz as the maximum sample rate.

To reduce sampling time Poole added a "stop sampling" signal to the FFT Analyser to cause it to transform the accumulated data which was made up to 1024 points by the inclusion of zeros. This reduces frequency resolution but was necessary to enable a multi-cell ionogram to be recorded in a reasonable time (e.g. 5 minutes).

Line-of-sight motion of the ionosphere can be detected by setting the transmitter to a specific frequency and analysing the Doppler frequency shift on the received signal to produce a so-called "Stationary-Doppler Ionogram". The FFT Sample-Rate generator was therefore designed to allow the selection of sample rates from 2^{10} = 1024 Hz down to 2^3 = 8 Hz. Note that the input signal to the FFT Analyser must be band-limited to half the sample rate to prevent aliasing.

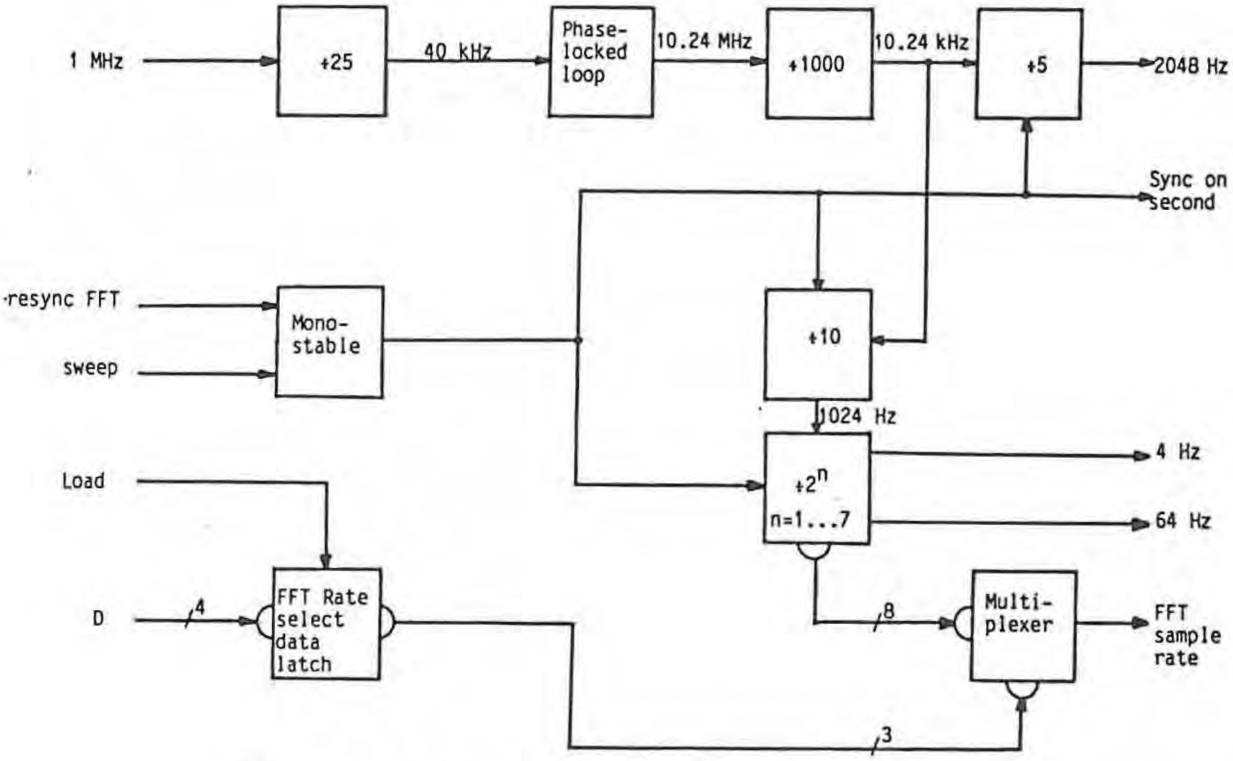


Figure (4.16) Block Diagram - FFT Sample Rate Generator (PC7)

1 MHz from the 105B Quartz Oscillator is divided by 25 to produce a 40 kHz reference signal for the phase-locked loop which multiplies it by 256 to provide 10.24 MHz output.

The 10.24 MHz signal is divided by 10^3 to 10.24 kHz. Division by 5 gives 2048 Hz which is used to clock the M-sequence generator.

Division of 10.24 kHz by 10 gives 1024 Hz which is divided to 512 , 256 , 128 , 64 , 32 , 16 , 8 and 4 Hz. One of the 8 frequencies from 1024 Hz to 8 Hz can be selected as the FFT sample-rate signal.

4 Hz and 64 Hz are output with either one or the other being selected as the Tcell-rate clock. Because the cell period is derived from a generator based on a phase-locked loop , provision is made for synchronization with the system 1 Hz clock signal at the begining of every ionogram. "Re-sync FFT" is required for resynchronization of the Tcell pulse to the system clock during oblique signal searching.

4.10 Windowing and Stationary-Doppler Recordings

To perform both windowing and stationary-Doppler recordings as described in section 3.4 and 3.5 a fixed frequency offset must be introduced into the received signal path. For windowing the ionosphere the instantaneous frequency difference between the transmitted and the received frequency must be reduced. For stationary Doppler recordings the "no motion" signal must be positioned within the analysis range by creating a fixed frequency difference between transmitted and received frequency.

The obvious method of achieving these offsets would be to use two frequency synthesizers , one to generate the signal to be transmitted and the other to generate the receiver local oscillator frequency ramp. This approach was not used because of the cost of a second synthesizer.

The second method considered involved programming the synthesizer from two sets of counters with the transmit/receive (T/R) signal being used to select between them. Presetting the counters with different initial values before clocking would introduce the required offset. The first problem with this idea is that the minimum bit period of the T/R waveform is 488 microseconds whereas the synthesizer takes about 1 millisecond for the frequency to settle to its new value. The second problem is that consecutive transmit or receive signal segments will , in general , be discontinuous in phase. The resulting signal would therefore not be the continuous linear ramp required for chirpsounding , and parameter evaluation from signal phase would not be possible.

A block diagram of the Chirpsounder receiver is given in figure (4.17). The received signal is mixed with the swept 40.6 to 70.1 MHz local oscillator signal to give a 40.1 MHz first IF. Mixing this with 40.0 MHz gives the second IF of 100 kHz. A 100 kHz bandpass crystal filter determines the ultimate bandwidth of 530 Hz centred about 100275 Hz. This signal is then demodulated to get the baseband signal.

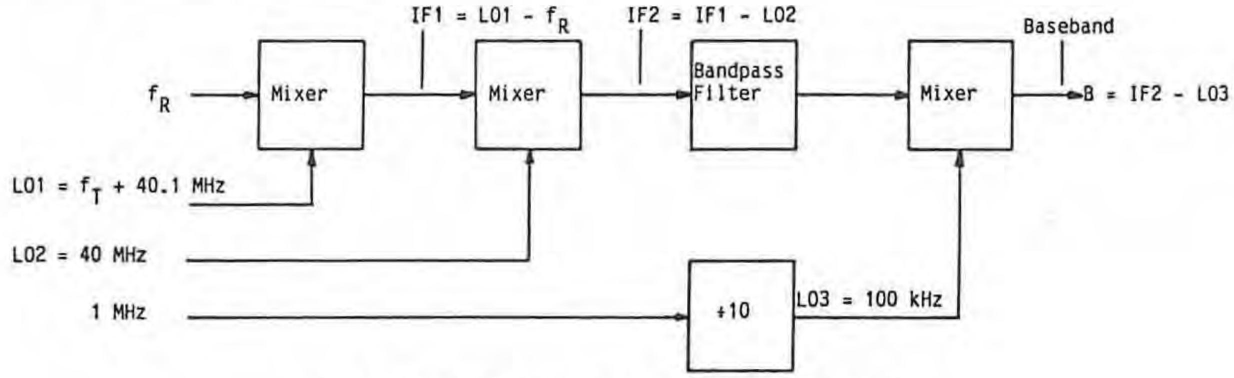


Figure (4.17) Block Diagram - Chirpsounder Receiver

Equation (2.3) gives the chirpsounding difference frequency as :-

$$\Delta f = f_T - f_R$$

$$\text{thus } f_R = f_T - \Delta f$$

The frequency of the swept local oscillator , L01 , is given by :-

$$L01 = f_T + 40.1 \text{ MHz} \tag{4.3}$$

The first intermediate frequency , IF1 is :-

$$IF1 = L01 - f_R$$

Substituting for LO1 and f_R yeilds :-

$$IF1 = f_T + 40.1 - (f_T - \Delta f)$$

$$= 40.1 + \Delta f \qquad (4.4)$$

The second local oscillator , LO2 , is :-

$$LO2 = 40 \text{ MHz} \qquad (4.5)$$

The second intermediate frequency , IF2 is :-

$$IF2 = IF1 - LO2$$

Substituting for IF1 and LO2 gives :-

$$IF2 = 40.1 + \Delta f - 40$$

$$= 100 \text{ kHz} + \Delta f \qquad (4.6)$$

The third local oscillator , LO3 , is :-

$$LO3 = 100 \text{ kHz} \qquad (4.7)$$

and the baseband signal , B , is thus :-

$$B = IF2 - LO3$$

$$= 100 + \Delta f - 100$$

$$= \Delta f \qquad (4.8)$$

Windowing (section 3.4) requires that the difference frequency Δf be reduced by some fixed offset f_0 before the bandpass filter. This can be achieved by increasing the frequency of LO2 by f_0.

That is :- $$LO2_{window} = 40 \text{ MHz} + f_0 \qquad (4.9)$$

then $$IF2 = IF1 - LO2$$

$$= 40.1 + \Delta f - (40 + f_0)$$

$$= 100 \text{ kHz} + \Delta f - f_0 \qquad (4.10)$$

and the baseband signal is :- $$B = IF2 - LO3$$

$$= 100 + \Delta f - f_0 - 100$$

$$= \Delta f - f_0 \qquad (4.11)$$

Equation (3.10) gives f_{0min} = 250 Hz. For ease of synthesis of both windowing and Doppler offset frequencies f_{0min} was chosen to be 256 Hz.

For stationary Doppler recordings (section 3.5), no line-of sight component of motion of the reflection point results in f = 0 Hz. This "no motion" signal can be positioned within the analysis range by reducing the frequency of LO2 by some fixed offset f_{OD}.

$$LO2_{Doppler} = 40 \text{ MHz} - f_{OD} \qquad (4.12)$$

The second intermediate frequency becomes :-

$$\begin{aligned} IF2 &= IF1 - LO2 \\ &= 40.1 + 0 - (40 - f_{OD}) \text{ as } \Delta f = 0 \\ &= 100 \text{ kHz} + f_{OD} \qquad (4.13) \end{aligned}$$

The baseband is then :-

$$\begin{aligned} B &= IF2 - LO3 \\ &= 100 + f_{OD} - 100 \\ &= f_{OD} \qquad (4.14) \end{aligned}$$

The "no motion" signal is thus positioned at f_{OD} Hz enabling both upward and downward motion of the reflection point to be observed. The choice of f_{OD} is governed by both the maximum expected vertical component of velocity of the reflection point and the frequency of transmission.

If an observer moves at velocity u away from a source of electromagnetic radiation of frequency f the observer receives frequency f' given by (Halliday and Resnick , 1970) :-

$$f' = f(1 - u/c)/\sqrt{1 - (u/c)^2} \qquad (4.15)$$

where c = velocity of an electromagnetic wave in free space.

In an ionospheric context equation (4.15) gives the frequency observed at the moving reflection point. Reflection of this signal back to the source introduces a second Doppler shift so that the frequency observed at the receiver is :-

$$f'' = f'(1 - u/c)/\sqrt{1 - (u/c)^2}$$

Substituting for f' from equation (4.15) gives :-

$$f'' = f(1 - u/c)^2/(1 - (u/c)^2)$$

$$= f(1 - 2u/c + (u/c)^2)/(1 - (u/c)^2)$$

Since u is very much smaller than c , $(u/c)^2$ can be approximated as 0.

Then $$f'' = f - 2uf/c$$

or $$f - f'' = - 2uf/c$$

$$\Delta f_{\text{Doppler reflected}} = -2uf/c \qquad (4.16)$$

An upper limit of $f_{\text{Doppler Reflected}}$ can be calculated by taking a sounding frequency of 10 MHz and the maximum velocity of the reflection point as being about +60 m/s (ie. upwards).

Then $$\Delta f_{\text{Doppler reflected}} = -2.60.10^7/3.10^8$$

$$= - 4 \text{ Hz}$$

The system was designed to have convenient local oscillator offsets for Doppler measurements of f_{OD} = 2 , 4 , 8 , 16 , 32 or 64 Hz.

$LO2_{window}$ (equation (4.9)) can be obtained by selecting the upper sideband of LO2 modulated by the windowing offset frequency f_0.

$LO2_{Doppler}$ (equation (4.12)) can be obtained by selecting the lower sideband of LO2 modulated by the Doppler offset frequency f_{OD}.

The Local Oscillator Offset Frequency Synthesizer (PC9) , Sine Wave Generator (PC10) and SSB Generator (B3) were designed to generate $LO2_{window}$ and $LO2_{Doppler}$.

4.11 Offsetting the 40 MHz Local Oscillator

The circuitry described below was designed to offset the 40 MHz local oscillator by a programmed amount for the purpose of ionospheric windowing and stationary-Doppler measurements.

Local Oscillator Offset Frequency Synthesizer (PC9)

A clocking frequency of 256f is required by PC10 to produce two sine waves of frequency f. PC9 synthesizes 256x2 , 256x4 , 256x8 , 256x16 , 256x 32 , and 256x64 for stationary-Doppler measurements and 256(256x1) through to 256(256x99) for windowing.

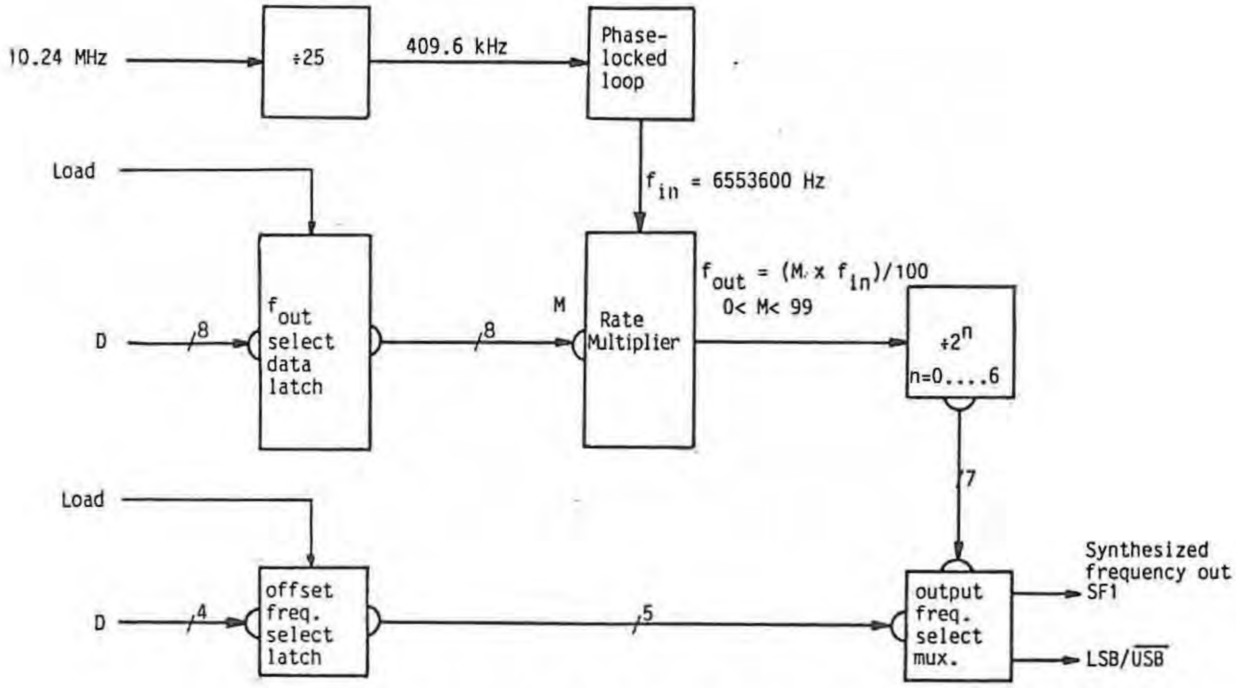


Figure (4.18) Block Diagram - Local Oscillator Offset frequency Synthesizer (PC9)

10.24 MHz from the FFT Sample-Rate Generator (PC7) is divided by 25 to provide a 409600 Hz reference signal for the phase-locked loop which outputs 6553600 Hz. Note that 6553600 Hz can be written as $2^{18} \times 5^2$ which is equal to $2^{16} \times 100$. The output frequency of the rate multiplier f_{out} is related to the input frequency f_{in} by $f_{out} = Mf_{in}/100$ for M greater than 0 but less than or equal to 99. Therefore :-

$$\begin{aligned} f_{out} &= M.2^{16}.100/100 \\ &= M.2^8.2^8 \\ &= 256.256.M \qquad (4.17) \end{aligned}$$

The output frequency select multiplexer selects $f_{out}/2^0$ as SF1 , the clocking frequency for PC10 for generating all the windowing offset frequencies. The speed of the 2716 EPROMS on PC10 limits the maximum allowable value of M to 30.

To generate the frequencies for stationary-Doppler the rate multiplication factor M is set to 1 so that $f_{out} = 256.256$. Division of this frequency by 2^n for n ranging from 1 to 6 gives the six clocking frequencies used in Doppler measurements.

The output signal SF1 can also be set to a low TTL logic level when no local oscillator offset is required.

Sine Wave Generator (PC10)

The Sine Wave Generator (PC10) generates two sine waves S0 and S1 of the same frequency f. The phase of S1 can be selected to lead or lag that of S0 by 90^0. These two signals are used in the SSB (Single Sideband) Generator (B3) to offset the 40 MHz local oscillator by the frequency f. B3 uses the phase-shift method of generating a single sideband signal.

The minimum offset frequency required is 2 Hz for stationary Doppler. The maximum offset frequency used for windowing is about 7.5 kHz. To ensure a constant 90^0 phase shift over such a wide range of frequencies the sine wave generator (PC10) had to be directly coupled to the following stage (B3).

Sine Wave Generator Circuit Development

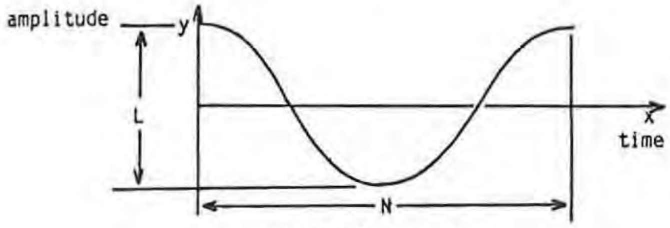


Figure (4.19) Cosine Wave

Let one cycle of a cosine wave (for convenience in later analysis) be sampled N times along the x-axis with the y-axis being divided into L quantized levels. The N quantized y values are stored in an EPROM at addresses 0 to N-1. A synchronous binary counter clocked at Nf addresses the EPROM as shown in figure (4.20). The output data are latched and drive a digital to analogue converter.

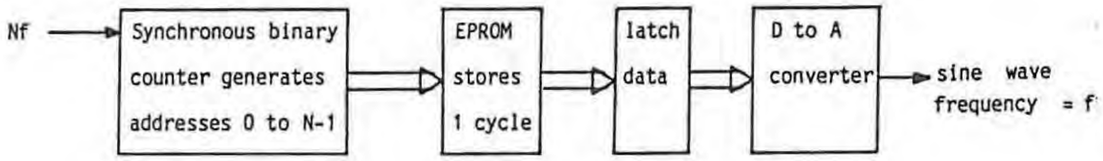


Figure (4.20) Block Diagram - Digital Sine Wave Generator

The D to A converter output is a stepped cosine wave of frequency f with L quantized levels. Figure (4.21) shows a cosine wave with L = N = 32.

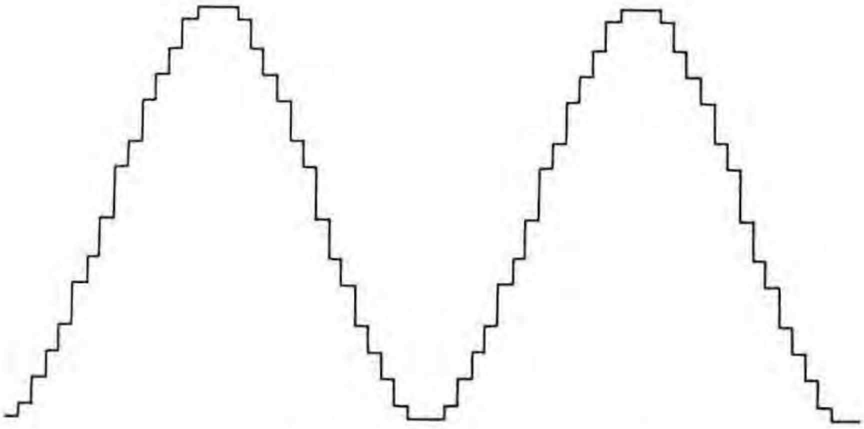


Figure (4.21) Digitally Generated Sine Wave

Harmonic Content of the Output Cosine Wave

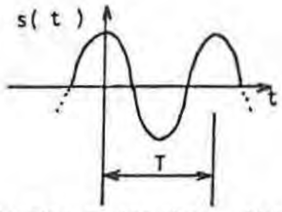


Figure (4.22a) Cosine Wave - time domain

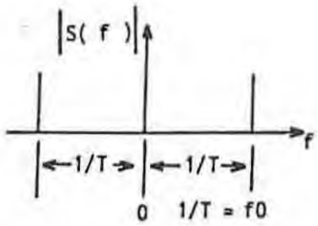


Figure (4.22b) Cosine Wave - frequency domain

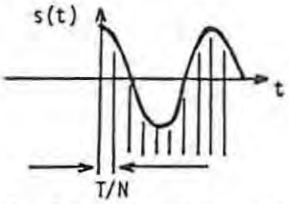


Figure (4.23a) Sampled Cosine Wave - time domain

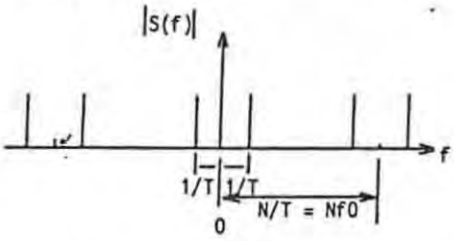


Figure (4.23b) Sampled Cosine Wave - frequency domain

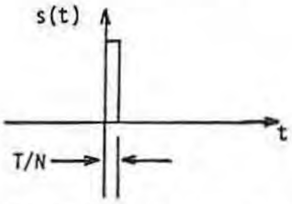


Figure (4.24a) Top Hat - time domain

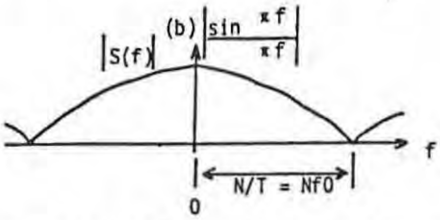


Figure (4.24b) Top Hat - frequency domain

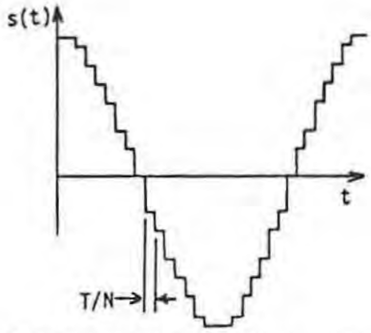


Figure (4.25a) Quantized Cosine Wave - time domain

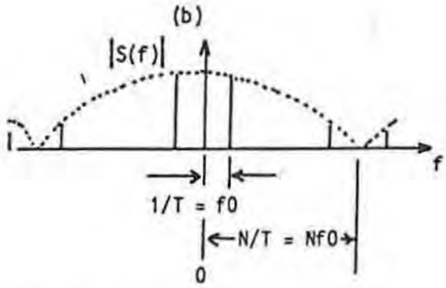


Figure (4.25b) Quantized Cosine Wave - frequency domain

$$s(\ t\ ) = \int_{-\infty}^{+\infty} S(\ f\ )\ e^{+j2\pi ft}dt$$

$$S(\ f\ ) = \int_{-\infty}^{+\infty} s(\ t\ )\ e^{-j2\pi ft}dt$$

Figure (4.22a) shows a cosine wave s(t) with frequency f and period T. Figure (4.22b) shows the modulus of the frequency spectrum S(f) which consists of only the fundamental frequency f.

The effect of sampling the cosine wave N times per cycle , figure (4.23a) , is shown in the frequency domain in figure (4.23b). Harmonics are present at (N - 1)f and (N + 1)f , (2N - 1)f and (2N + 1)f etc.

The modulus of the frequency spectrum of a pulse of width T/N is shown in figure (4.24a). This is the modulus of the sinc function , that is $\left|\frac{\sin \pi f}{\pi f}\right|$.

The modulus of the frequency spectrum of the quantized cosine wave of figure (4.25a) is shown in figure (4.25b) and is the convolution of (4.23a) and (4.24a). The quantized cosine wave has harmonics at :-

f , (N - 1)f , (N + 1)f , (2N - 1)f , (2N + 1)f , ...

with amplitudes of :-

1 , 1/(N - 1) , 1/(N + 1) , 1/(2N - 1) , 1/(2N + 1) , ...

The number of samples of one cycle of the cosine wave was choosen to be 256. This allows the use of an 8-bit binary counter to address the EPROM. The clocking frequency , 256f , is generated by PC9.

By choosing L = 256 levels on the y-axis the full 8-bits of EPROM data can be used to drive an 8-bit digital-to-analogue converter.

The output sine wave has therefore a first harmonic of amplitude 1/(N - 1) of the amplitude of the fundamental at frequency (N - 1)f = 255f.

The EPROM data were calculated as follows :-

N = total number of x-axis samples for 1 cosine cycle = 256.

L = upper limit of 256 quantized levels from 0 to 255 = 255

n = n th sample , $0 \leqq n \leqq 255$

S_n = EPROM data , the n th sample. $0 \leqq S_n \leqq L$

$$S_n = (\cos(360/N)n+1)(L/2)$$

$$= (\cos\ (360/256))n+1)(255/2) \qquad (4.18)$$

for $0 \leqq n \leqq 255$

The quadrature sine wave generator designed for the system is shown below in figure (4.26).

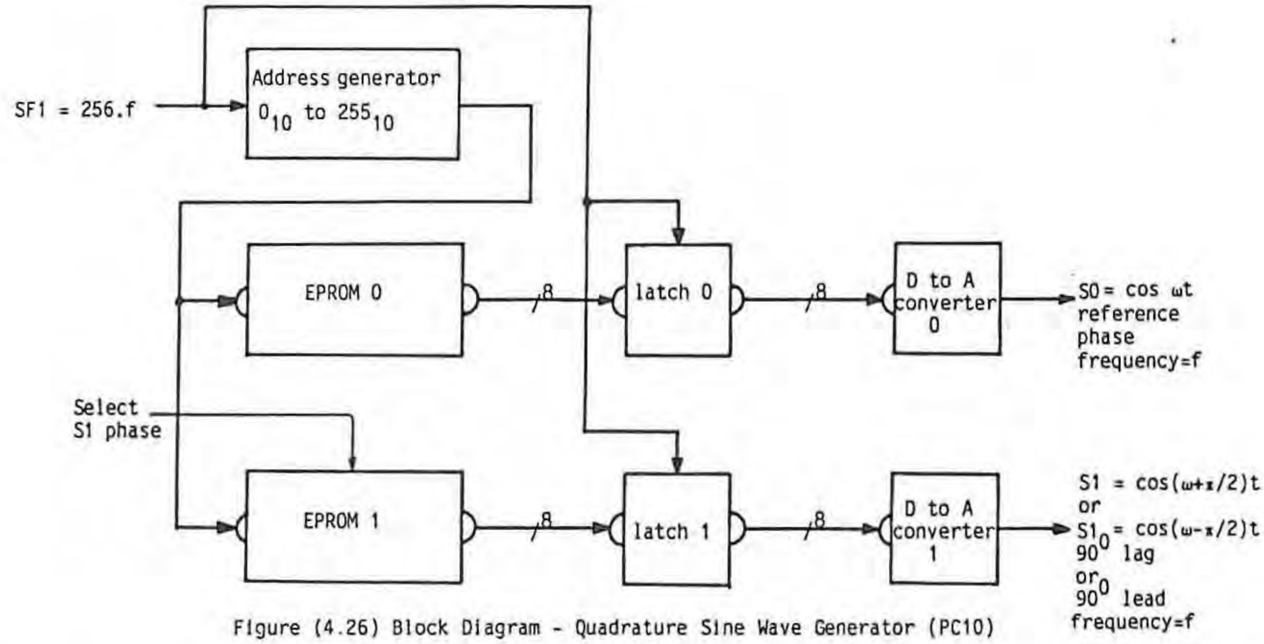


Figure (4.26) Block Diagram - Quadrature Sine Wave Generator (PC10)

A synchronous binary counter clocked at 256f addresses the two EPROMS. EPROM 0 contains one cycle of cosine wave data at addresses 0 to 255. These data generate the reference signal via latch 0 and D-to-A converter 0. The output signal can be written as :-

$$S0 = E_m \cos \omega_m t \qquad (4.19)$$

where E_m = maximum signal amplitude , $\omega = 2\pi f$ and t = time.

EPROM 1 contains two sets of data, the first at addresses 0 to 255 being the EPROM 0 data shifted by $+90^0$ (i.e. $\pi/2$ radians). When selected this data generates

$$S1 = E_m \cos(\omega_m + \pi/2)t \qquad (4.20)$$

The second set of data stored at addresses 256 to 511 is the EPROM 0 data shifted by -90^0 (i.e. $-\pi/2$ radians). When selected this set gives :-

$$S1 = E_m \cos(\omega_m - \pi/2)t \qquad (4.21)$$

The signals S0 and S1 drive the SSB Generator (B3) directly.

SSB Generator Circuit (B3)

The SSB generator circuit is used to offset the 40 MHz local oscillator by the frequency f generated on PC10. The lower sideband (40 - f) is used for stationary-Doppler and the upper sideband (40 + f) for windowing. For normal vertical- and oblique-incidence ionograms the 40 MHz signal is not shifted.

The phase-shift SSB circuit [Schwartz, 1970] is given in figure (4.27)

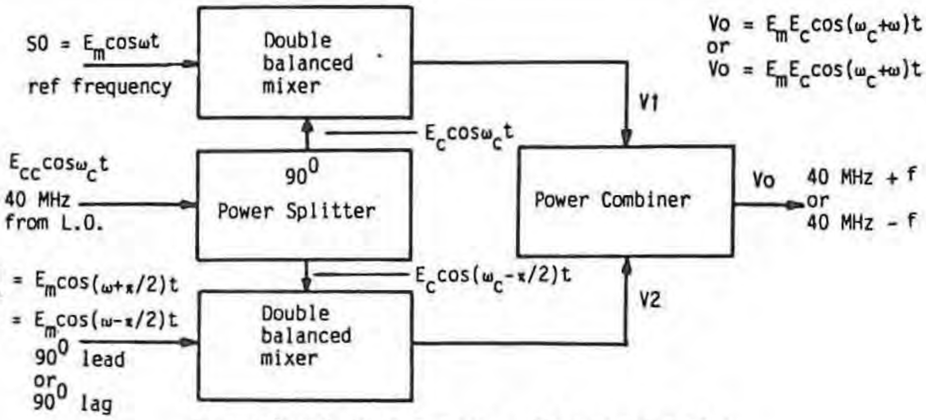


Figure (4.27) Block Diagram - SSB Generator (B3)

The 40 MHz carrier , $E_{cc}\cos\omega_c t$ is split into two signals $E_c\cos\omega_c t$ and $E_c\cos(\omega_c-\pi/2)t$ which are mixed with S0 and S1 respectively. The signal V1 is therefore :-

$$V1 = (E_c\cos\omega_c t)(E_m\cos\omega t)$$

Using the identity (cosA)(cosB) = 1/2cos(A+B) + 1/2cos(A-B)

$$V1 = 1/2E_cE_m\cos(\omega_c+\omega)t + 1/2E_cE_m\cos(\omega_c-\omega)t \qquad (4.22)$$

Similarly , choosing $S1 = E_m\cos(\omega+\pi/2)t$, the signal V2 is :-

$$\begin{aligned} V2 &= (E_c\cos(\omega_c-\pi/2)t)(E_m\cos(\omega+\pi/2)t) \\ &= 1/2E_cE_m\cos(\omega_c-\pi/2+\omega+\pi/2)t + 1/2E_cE_m\cos(\omega_c-\pi/2-\omega-\pi/2)t \\ &= 1/2E_cE_m\cos(\omega_c+\omega)t + 1/2E_cE_m\cos(\omega_c-\omega-\pi)t \end{aligned}$$

Since $\cos(A-\pi) = -\cos A$, V2 can be written :-

$$V2 = 1/2E_cE_m\cos(\omega_c+\omega)t - 1/2E_cE_m\cos(\omega_c-\omega)t \qquad (4.23)$$

The power combiner sums V1 and V2 to give :-

$$\begin{aligned} Vo &= V1 + V2 \\ &= 1/2E_cE_m\cos(\omega_c+\omega)t + 1/2E_cE_m\cos(\omega_c-\omega)t \\ &\quad + 1/2E_cE_m\cos(\omega_c+\omega)t - 1/2E_cE_m\cos(\omega_c-\omega)t \\ &= E_cE_m\cos(\omega_c+\omega)t \qquad (4.24) \end{aligned}$$

This is the upper sideband as required for windowing (equation (4.9)).

By choosing $S1 = E_m\cos(\omega-\pi/2)t$ the lower sideband is obtained . That is :-

$$Vo = E_cE_m\cos(\omega_c-\omega)t \qquad (4.25)$$

This signal is used for stationary-Doppler measurements (equation (4.12)).

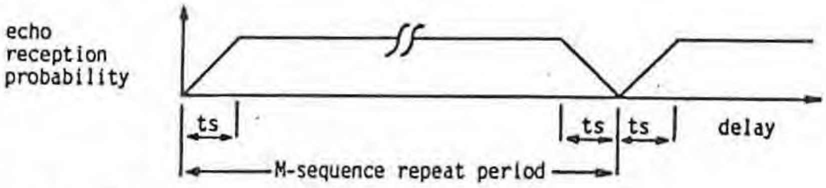
4.12 Transmit/Receive Switch Control

Programmable M-Sequence Generator

The minimum ionospheric virtual height h'_{min} is approximately 90 km. The corresponding minimum delay time Δt_{min} can be obtained from equation (1.1).

$$\begin{aligned}\Delta t_{min} &= 2h'_{min}/c \\ &= 2.90/3.10^5 \\ &= 600\,\mu s \qquad (4.26)\end{aligned}$$

Since the minimum target delay is short compared with the duration of a cell it is necessary to employ transmit/receive (T/R) switching. A maximum-length sequence or M-sequence , also known as a pseudo-random binary sequence [MacWilliams and Sloane , 1976] , was chosen as the switching waveform because the echo reception probability (auto-correlation function) rises linearly during the first bit period to a constant value. The probability remains constant for all delay times up to one bit period less than the M-sequence repeat period when it falls to zero. The M-sequence repeat period was chosen to be equal to the cell period which is very much greater than the maximum delay of interest.



ts = single bit period

Figure (4.28) M-sequence - reception probability vs delay time

600 μs as calculated in equation (4.26) is thus the maximum single bit period allowed before reduced echo reception probability occurs at the

delays of interest. The minimum single-bit period is determined by the maximum rate of T/R switching possible which is about 200μs.
An M-sequence can be generated using a shift register with appropriate feedback. Certain bits are modulo-two summed (exclusive-OR'ed) and fed back into the shift register. An n-bit shift register can have 2^n possible states. An M-sequence of order n can have 2^n-1 possible states before repeating since the all-zero state is invalid.

An 11-bit shift register has $2^{11} = 2048$ possible states. Choosing a clock frequency of 2048 Hz results in a single-bit period of 1/2048 = 488.28 μs. Relating this time to height using equation (1.1) :-

$$\begin{aligned} h' &= (488.28\times10^{-6})(3\times10^5)/2 \\ &= 73.24 \text{ km} \end{aligned}$$

The echo reception probability reaches its constant value for reflections from above 73 km. E-region reflections which are from about 90 km are therefore received with maximum possible probability.

An 11-bit M-sequence can be clocked $2^{11}-1$ times before it repeats. Clocked at 2048 Hz the sequence would complete in 2047 clock pulses and would begin again on the 2048th clock pulse. For $T_C = 1$ s the start of the M-sequence relative to the Tcell pulse would precess by 1 clock pulse (488.28 μs) each second. By resetting the M-sequence at the start of each cell there would be no precession and the code would be identical for each cell.

A clock frequency of 2048 Hz implies 1024 pulses in half a second. Because $2^{10} = 1024$ an M-sequence of order 10 would repeat every 1023 clock pulses. Thus for different cell periods all related by a factor of 2 , it is only necessary to change the M-sequence order to obtain a code with a repeat period equal to the cell period.

The Programmable M-sequence Generator (PC4c) allows the selection of M-sequences of order 6 through 16 to correspond to cell periods of 1/32 s through to 32 s. To obtain M-sequences of order 6,7,8,9,10,11,12,13,14,15 and 16 a 16 bit shift register is required. A computer program was run to list all the possible feedback points for an M-sequence of a particular order. It was discovered that any one of the M-sequences required could be obtained by using only 4 feedback points. Shift register bits 2 and 3 were common to all the orders required.

The shift register bits are designated b1 through to b16. The 4 feedback points are designated F1 , F2 , F3 and F4 and the modulo-2 addition of the data selected by these points is called F.

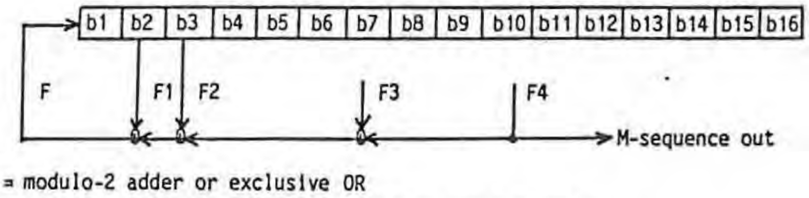


Figure (4.29) M-sequence Generator

Depending on the order required the selection of one of b4 , b5 , b8 or b9 as F3 gives an M-sequence. The fourth feedback point F4 must select data from the bit number equal to the M-sequence order required as this determines the shift register length. For example , F4 must be connected to b10 for an M-sequence of order 10. Being connected to the end of the shift register this point is also the output of the M-sequence.

If the data selected by F1 , F2 , F3 and F4 is A , B , C and D respectively the feedback F is :-

$$F = A \oplus (B \oplus (C \oplus D)) \qquad (4.27)$$

The reset on the shift registers used clears all the bits to zero. This is the only invalid bit combination in an M-sequence. If inverted it becomes the valid all-ones code.

Let the shift register contain inverted data. Then the four feedback points select the inverted data A , B , C and D. The feedback F after successive exclusive ORing is :-

$$\begin{aligned} F &= \overline{A} \oplus (\overline{B} \oplus (\overline{C} \oplus \overline{D})) \\ &= \overline{A} \oplus (\overline{B} \oplus (C \oplus D)) && \text{as } \overline{X} \oplus \overline{Y} = X \oplus Y \\ &= \overline{A} \oplus (\overline{B \oplus C \oplus D}) && \text{as } \overline{X} \oplus Y = \overline{X \oplus Y} \\ &= A \oplus B \oplus C \oplus D && \text{as } \overline{X} \oplus \overline{Y} = X \oplus Y \\ F &= A \oplus B \oplus C \oplus D && (4.28) \end{aligned}$$

The feedback is exactly the same as was obtained with non-inverted data. Because the shift register contains inverted data F must be inverted to get the correct feedback.

$$\overline{F} = \overline{A \oplus B \oplus C \oplus D}$$

The non-inverted M-sequence is obtained by inverting the data selected by F4.

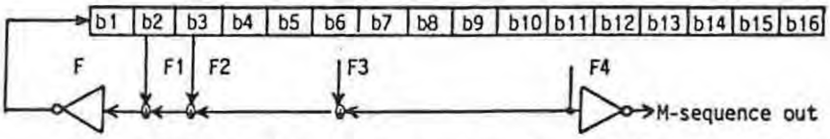


Figure (4.30) M-sequence generator - inverted data

The XOR gate and inverter can be replaced by an XNOR gate. If the other two XOR gates are also replaced with XNOR gates (shown here as XOR gates followed by an inverter) the correct feedback is still obtained as shown below in figure (4.31)

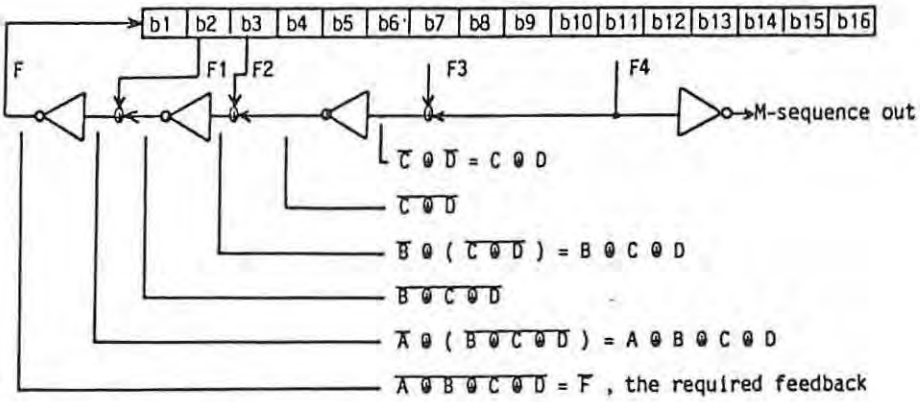


Figure (4.31) M-sequence generator - XNOR implementation

Figure (4.32) shows the block diagram of the programmable M-sequence generator. Data from the computer is latched and then used to select the third and fourth feedback points thereby determining the order of the M-sequence. The Tcell pulse resets the shift register to an all-zero state at the beginning of each cell. The shift register is clocked at 2048 Hz.

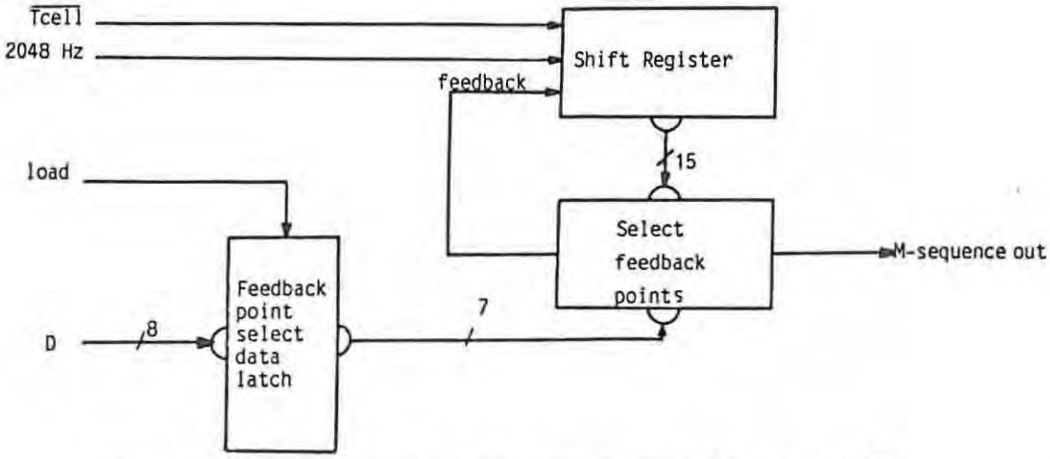


Figure (4.32) Block Diagram - Programmable M-sequence Generator (PC4c)

T/R Gain Weight Generator - replaces 4050 Amplifier T/R board A6

The original Barry Research T/R signal was based on a 16-bit code , the period of which was changed at a 1 Hz rate to eliminate the blind range that would otherwise occur. The T/R rate generator board produced this signal and converted it to three signals, the T/R pulse train, the Rx gain-weight pulse train and the Tx gain-weight pulse train. These signals are all the same except for the Rx gain-weight pulse train which rises 180 s before the other two.

The T/R Gain-Weight Generator board is shown as a block diagram in figure (4.33). It generates the three signals from the M-sequence produced on PC4c and plugs directly into slot A6 in the 4050 Amplifier T/R. The two control lines allow the output signals to be set to one of three possible states. These states are Tx only or Rx only for oblique operation or T/R switching for vertical operation.

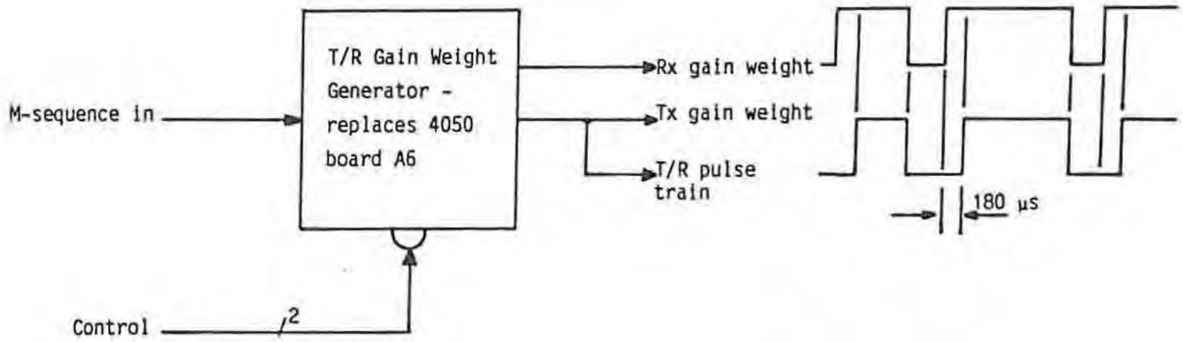


Figure (4.33) Block Diagram - T/R Gain Weight Generator (replaces 4050 board A6)

4.13 System Control

Tcell Pulse Generator (PC4b)

The cell, as described in section 3.2, is the fundamental building block of the synthesizer control system. During a cell the ionosonde transmitter frequency can increase at some linear rate, the basic rate, or it can remain constant. The Tcell Pulse Generator (PC4b) outputs the pulses that determine the cell period T_C.

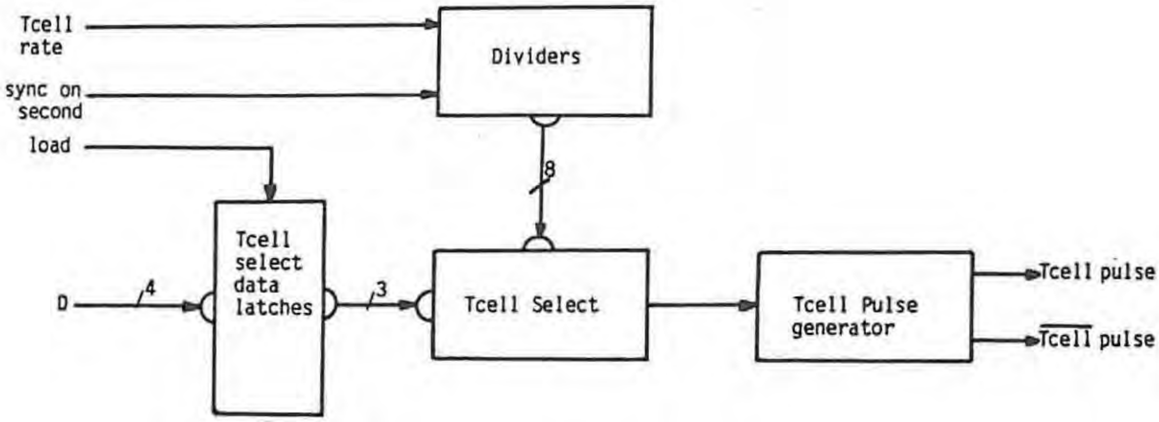


Figure (4.34) Block Diagram - Tcell Pulse Generator (PC4b)

Either 64 or 4 Hz can be selected as the Tcell rate. This provides two overlapping sets of cell periods, one set being for normal multi-cell vertical or oblique ionograms and the other for stationary-Doppler measurements. The latter set includes cell periods up to 64 s for the integration and analysis of small Doppler frequency shifts.

The dividers can be cleared by the "sync on second" pulse. Latched data select the required cell period. The output 100 ns long pulse, "Tcell pulse", signals the beginning of a new cell.

Gating on PC3b allows the Tcell pulse to be used to interrupt the computer , load the synthesizer counters and latch antenna-select data.

Sweep Start and Control (PC3b)

The sweep start circuitry was designed to start an ionogram synchronous with the falling edge of the 1 Hz clock signal. At any time less than 1 s before an ionogram is due to start the circuit can be armed by a strobe signal. The signal "sweep" goes high on the next falling edge of the 1 Hz clock , resynchronizes the FFT Rate Generator output and enables the system.

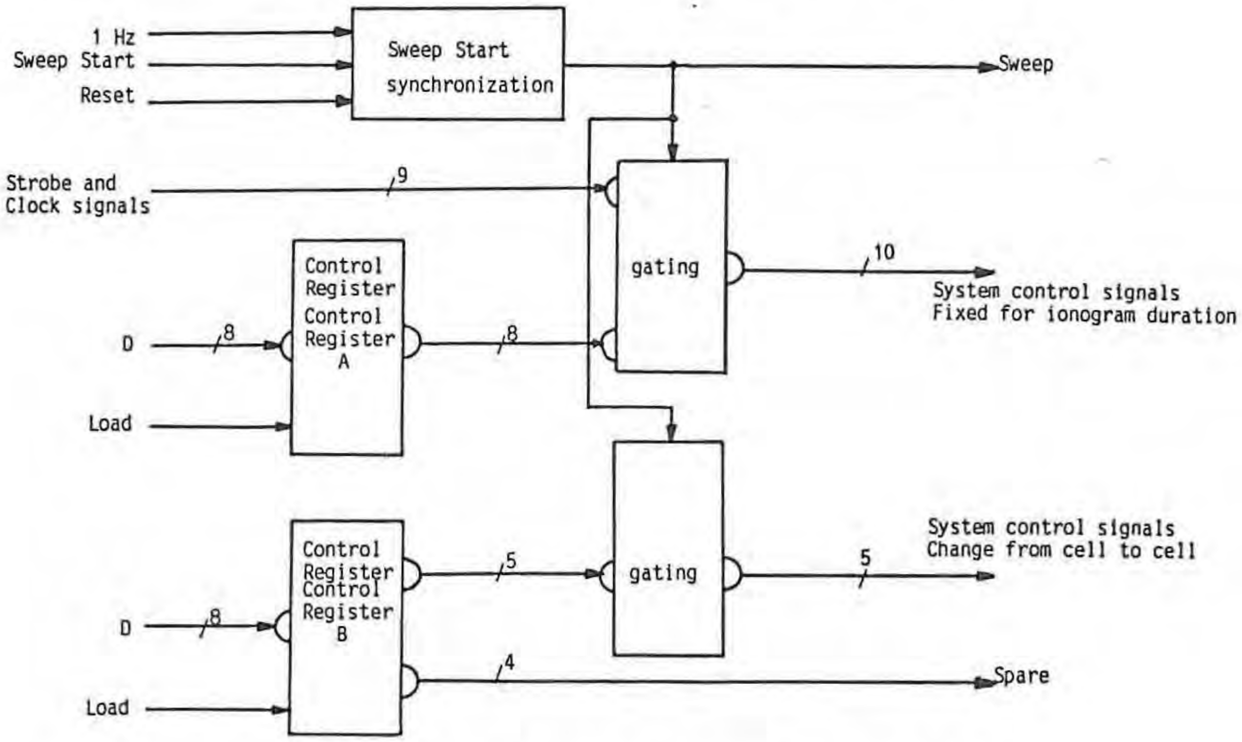


Figure (4.35) Block Diagram - Sweep Start and Control (PC3b)

Ionogram execution is controlled by data written to two 8-bit registers , Control Register A and Control Register B.

The data latched in Control Register A determine the type of recording required (e.g. vertical , oblique , stationary Doppler , etc.) and do not change during execution. Table (4.1) lists the function of each of the eight bits.

Controller Data Bus Bit		Function
Bit No.	State	-
D7	0 1	4 Hz - Stationary Doppler 64 Hz - Ionograms Tcell pulse gen rate select
D6	0 1	disable Sweep signal to FFT enable Sweep signal to FFT
D5 , D4	0 0 0 1 1 0 1 1	Tx only Tx only Rx only T/R switching T/R Control. D5 also selects oblique or vertical transmit antenna.
D3	0 1	disable Tcell interrupt to CPU enable Tcell interrupt to CPU
D2	0 1	disable antenna load by Tcell pulse enable antenna load by Tcell pulse
D1	0 1	disable synth counter load by Tcell pulse enable synth counter load by Tcell pulse
D0	0 1	disable synth counter basic rate clock enable synth counter basic rate clock

Table (4.1) Control Register A

As shown in Table (4.1) above,bits D4 and D5 select either Tx-only or Rx-only modes for oblique operation or T/R switching for vertical operation. Bit D5 is used to select either the oblique or the vertical transmit antenna. These bits also generate the control signal used by the 4040 Rx Antenna Select (B2b) circuit which connects the 4040 Rx to either the oblique antenna or to a vertical incidence receive antenna.

Control Register B contains data that may change from cell to cell during a recording (e.g. vertical , oblique etc.). Table (4.2) shows the function of each bit.

Controller Data bus bit		Function
Bit No.	State	
D7	0 1	spare control bit B7 available on PCB only
D6	0 1	spare control bit B6
D5	0 1	spare control bit B5
D4	0 1	spare control bit B4
D3	0 1	disable fast film advance enable fast film advance
D2	0 1	disable transmitter enable transmitter
D1	0 1	Internal Rx AGC External Tx AGC
D0	0 1	disable film drive enable film drive

Table (4.2) Control Register B

4.14 BCD x 4 , Frequency Mark Generator and Tx Filter Control (PC2)

The simplified block diagram of this board is given below in figure (4.36).

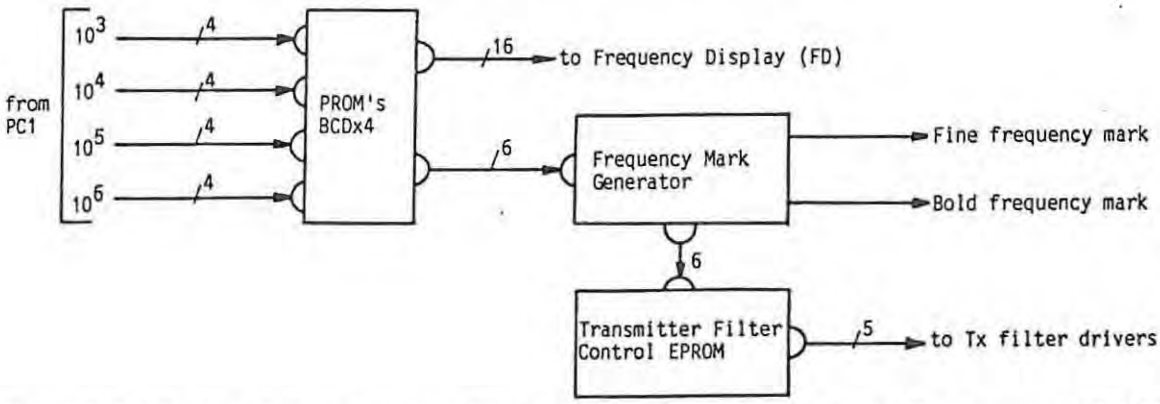


Figure (4.36) Simplified Block Diagram - BCD x 4 , Frequency Mark Generator and Tx Filter Control (PC2)

BCD x 4 Circuitry

Equation (4.2) gives the relationship between the transmitted frequency and the synthesizer BCD programming data as :-

$$f_T = 4(\text{BCD programming number})$$

Circuit board PC2 digitally multiplies the synthesizer programming data by 4 (using PROM's) to obtain the instantaneous transmitter frequency which is displayed on the Controller front panel by the Frequency Display board (FD).

The maximum BCD number from any of the programming decades is 9. Multiplication by 4 gives 9 x 4 = 36 , that is six units with a carry of 3. Including a possible carry of 3 from a previous stage gives 36 + 3 = 39. The maximum carry out is therefore 3 which requires two bits for binary representation. Table (4.3) shows the results in binary of multiplying the BCD numbers 0 through 9 by 4 for Cin (carry in) = 0 , 1 , 2 and 3.

		BCD x 4							
Decimal	BCD	Cin = 00		Cin = 01		Cin = 10		Cin = 11	
	2^3 2^2 2^1 2^0	Cout	BCDx4	Cout	BCDx4	Cout	BCDx4	Cout	BCDx4
0	0 0 0 0	0 0	0 0 0 0	0 0	0 0 0 1	0 0	0 0 1 0	0 0	0 0 1 1
1	0 0 0 1	0 0	0 1 0 0	0 0	0 1 0 1	0 0	0 1 1 0	0 0	0 1 1 1
2	0 0 1 0	0 0	1 0 0 0	0 0	1 0 0 1	0 1	0 0 0 0	0 1	0 0 0 1
3	0 0 1 1	0 1	0 0 1 0	0 1	0 0 1 1	0 1	0 1 0 0	0 1	0 1 0 1
4	0 1 0 0	0 1	0 1 1 0	0 1	0 1 1 1	0 1	1 0 0 0	0 1	1 0 0 1
5	0 1 0 1	1 0	0 0 0 0	1 0	0 0 0 1	1 0	0 0 1 0	1 0	0 0 1 1
6	0 1 1 0	1 0	0 1 0 0	1 0	0 1 0 1	1 0	0 1 1 0	1 0	0 1 1 1
7	0 1 1 1	1 0	1 0 0 0	1 0	1 0 0 1	1 1	0 0 0 0	1 1	0 0 0 1
8	1 0 0 0	1 1	0 0 1 0	1 1	0 0 1 1	1 1	0 1 0 0	1 1	0 1 1 1
9	1 0 0 1	1 1	0 1 1 0	1 1	0 1 1 1	1 1	1 0 0 0	1 1	1 0 0 1

Table (4.3) BCD x 4 - Binary Representation

From the table it is clear that the carry-in LSB determines the LSB of (BCD x 4 + Cin). The carry-in MSB defines two sets of BCD x 4 data :-

Set 1 = (BCD x 4)

Set 2 = (BCD x 4) + 2

The carry-in LSB therefore determines the LSB (2^0) of the multiplication. PROM address lines A0 to A3 are driven by the BCD number to be multiplied by 4. The carry-in MSB selects the correct data set via address line A4.

Address	Data
0 - 9	set 1
10 - 15	not used
16 - 25	set 2
26 - 31	not used

Five bits of PROM output data are used , the first three being the bits 2^1 , 2^2 and 2^3 of the result and the next two being the carry out LSB and carry out MSB.

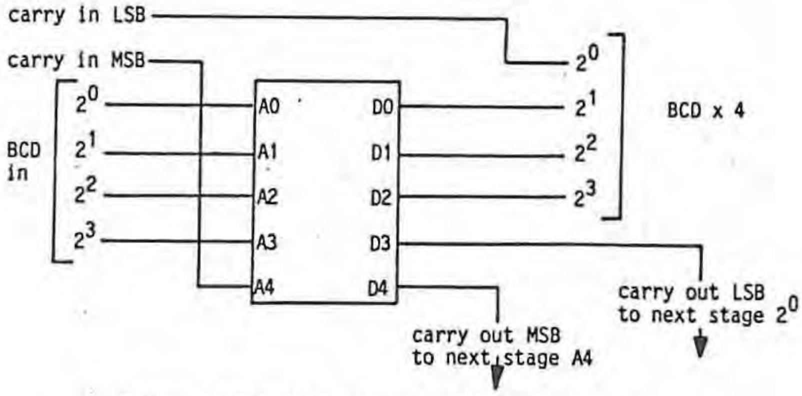


Figure (4.37) PROM Address and Data Organization

Frequency Mark Generator

Frequency marks were required on film at the following frequencies:-

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 25 26 27 28 29 30 MHz

Bold frequency marks were required at 0 , 10 , 20 and 30 MHz.

Only the first transition of a frequency was to trigger a frequency mark with subsequent transitions being ignored. This condition is necessary because a particular frequency can be passed repeatedly at the low end of a logarithmic overall sweep or when using a "jumpback" linear overall rate.

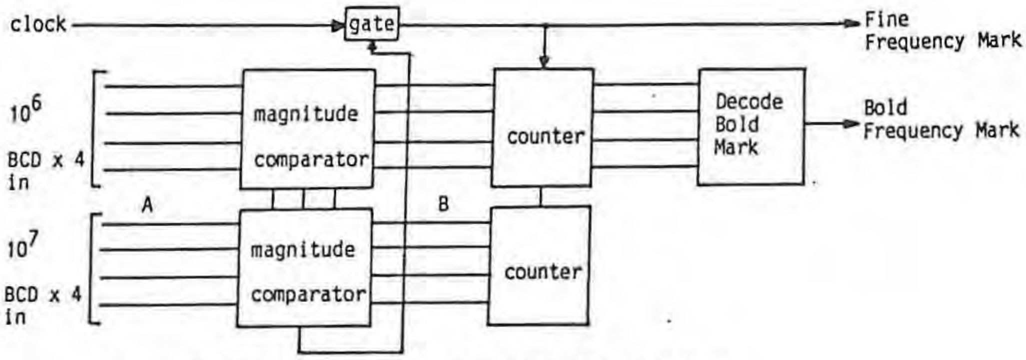


Figure (4.38) Block Diagram - Frequency Mark Generator

Two 4-bit magnitude comparators compare the 10^6 and 10^7 BCD x 4 decades , "A" , with the number "B" output by two BCD counters. The clock to the counters is disabled when B is greater than A and enabled when B is less than or equal to A.

If A = x then , for the clock to be disabled B = x+1. When A changes to x+1 the clock is enabled , the next clock pulse increments B to x+2 and the comparator disables the gate. The gated clock pulse signals the x+1 MHz fine frequency mark once only as required.

Because the BCD counters are one count ahead of the actual frequency bold frequency marks must be triggered when the counters contain 1 , 11 , 21 and 31. Bold frequency marks are obtained by fully decoding the LSB counter to produce a pulse when it contains 0 0 0 1.

Transmitter Filter Control

The T500A linear power amplifier that was added to the system (figure(4.1)) has five filters on the RF output. These filters cover the ranges 2-3 , 3-5, 5-8 , 8-15 and 15-30 MHz and are controlled by an EPROM. Using the frequency-mark-generator BCD counters to address the EPROM ensures that only the first transition of a switching frequency is acted upon.

The 3 MHz filter control signal is also used to switch from the low frequency receive antenna set to the high frequency set.

Frequency Display (FD)

The four most significant decades of the BCD x 4 data are displayed on the frequency display board (FD) in the form "xx.xx" MHz.

4.15 Antenna Switching

Antenna Select , Tx Filter Drivers and 4040 RX Antenna Select (B2)

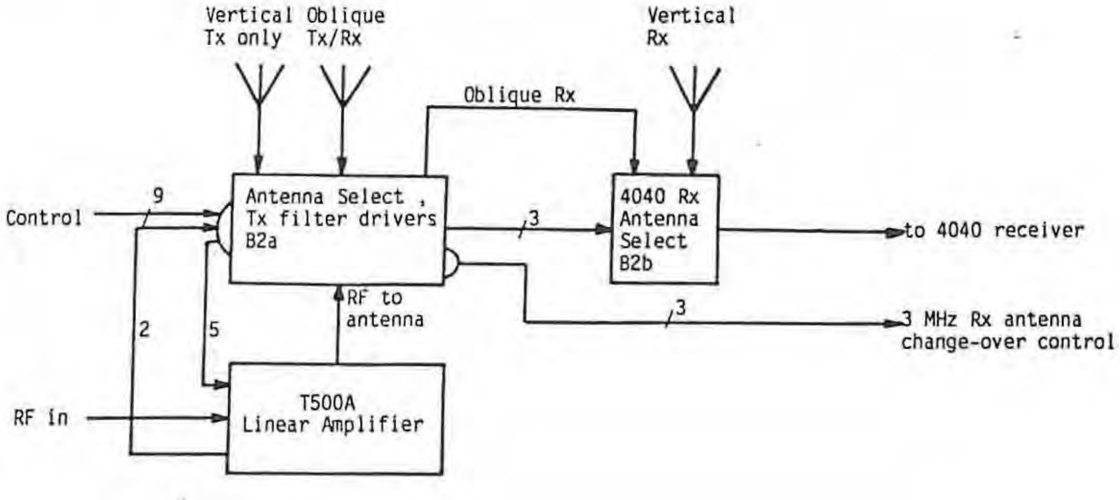


Figure (4.39) Block Diagram - Vertical/Oblique Antenna selection

RF from the 4050 Amplifier T/R drives the T500A linear amplifier which boosts the output power by about 10dB. Co-axial relays in B2a connect the T500A output to either a vertical transmit antenna or to an oblique antenna. With the transmitter connected to the vertical Tx antenna the oblique antenna is switched through to B2b. Here either the oblique antenna or a vertical incidence receive antenna can be selected to go to the 4040 receiver.

B2a also contains the T500A output filter relay driver circuitry. The T500A was modified by the addition of protection diodes across the filter relays. A 22.5 VDC supply mounted in the T500A supplies power to the co-axial relays in B2a and B2b. B2a outputs the 3 MHz change-over control signal to B4.

Antenna Select Board (PC8a)

The antenna control logic (PC8a) together with the antenna switch (B1) allows the selection of one of eight receive antennae to either of two receivers. Unused antennae are kept grounded.

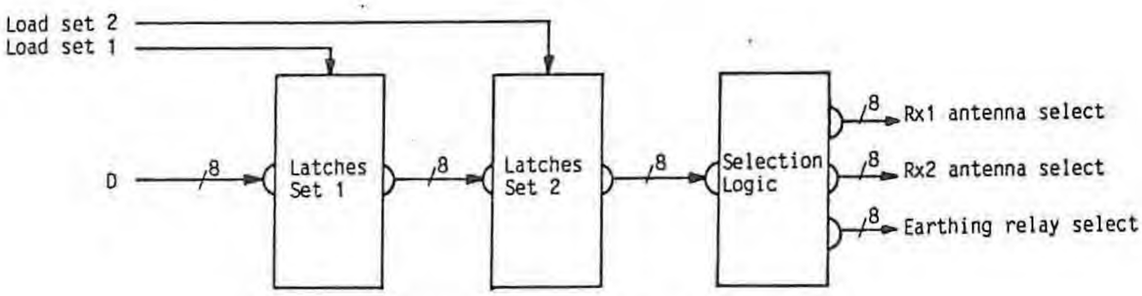


Figure (4.40) Block Diagram - Antenna Select (PC8a)

Double latching allows antennae switching to be controlled by the Tcell pulse which loads the second set of latches from the first.

3 MHz Antenna Change-over (B4) and Receive Antenna Switch (B1)

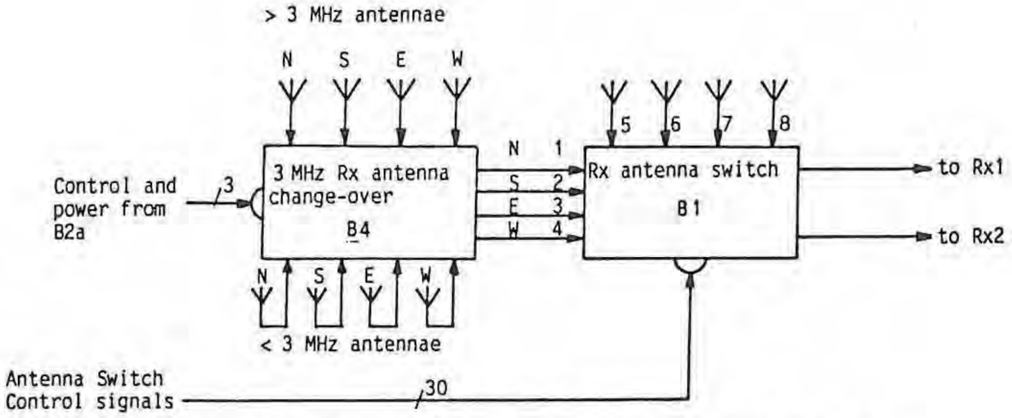


Figure (4.41) Block Diagram - Dual Receiver Antenna Switching

Electrically-short receive antennae were designed for the system by Poole , 1983. To improve receiver low frequency response two sets of four antennae were used with a change-over frequency of 3 MHz. The output signals of B4 are applied to four of the eight available inputs on the receive antenna switch (B1). B1 contains the switching circuitry that allows the selection of one of eight receive antennae to either of two receivers.

4.16 Film Speed Control (PC8b)

The film speed control circuitry generates two signals , a 500 Hz fast-film-advance signal and a variable-film-speed signal. The variable- film-speed signal can be enabled or disabled by computer.

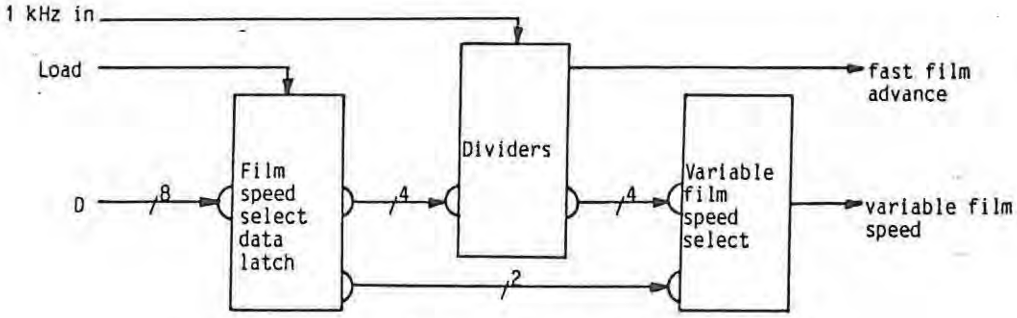


Figure (4.42) Block Diagram - Film Speed Control (PC8b)

4.17 Digitized Chirpsounder - Front and Rear Views

Figure (4.43) shows the front view of the digitized VOS-1C Vertical/Oblique Chirpsounder system.

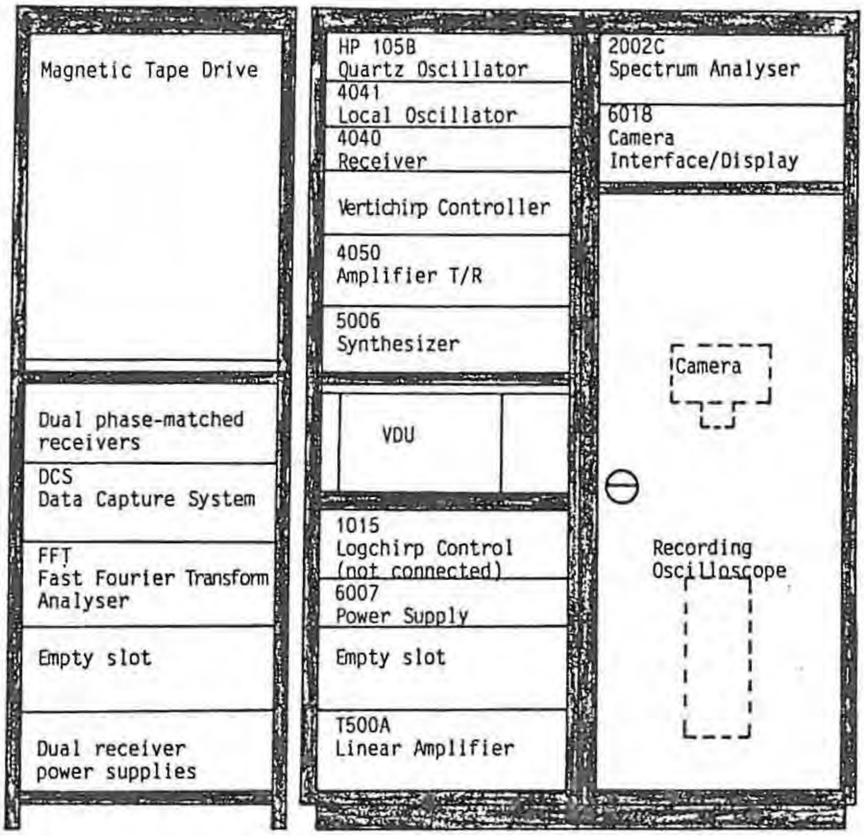
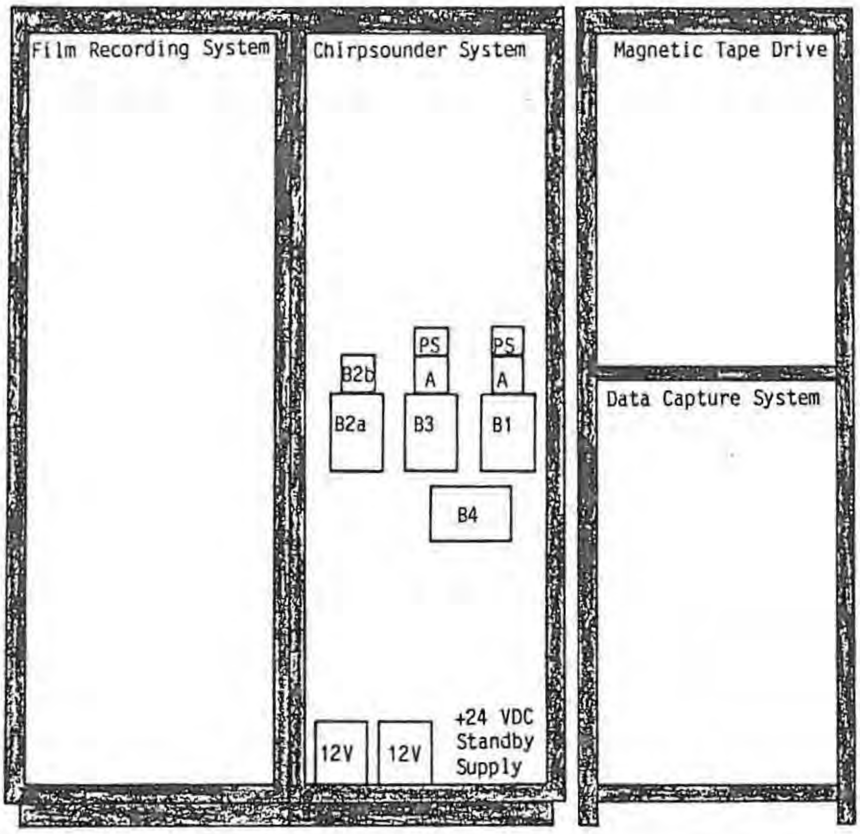


Figure (4.43) Digitized VOS-1C Vertical/Oblique Chirpsounder - Front View

Figure (4.44) shows the rear view of the digitized VOS-1C Vertical/Oblique Chirpsounder system.



PS = 0^0 Power Splitter
A = 10dB RF Amplifier
B1 = Receive Antenna Switch
B2a = Antenna Select , Tx Filter Drivers
B2b = 4040 Rx Antenna Select
B3 = SSB Generator
B4 = 3 MHz Antenna Change-over

Figure (4.44) Digitized VOS-1C Vertical/Oblique Chirpsounder - Rear View

CHAPTER 5

CONTROLLER SOFTWARE

5.1 Introduction

The initial software development and the choice of FORTH as a programming language has already been described in section 3.6. Subsequent software development was done on the Controller computer system itself with a dual 5 1/4 inch disk drive being used for storage. This arrangement allowed immediate testing of the software with the hardware. The final Controller program was stored in EPROM and the disk drive was removed from the system.

A block diagram of the major components of the system software is given in section 5.2 . More detailed information is given on each functional block in the following sections.

The structure of FORTH is described as an introduction to the modifications made to the standard FORTH memory map and dictionary. The changes briefly discussed here are fully documented in Appendix C.

The manual "Vertichirp Controller Software - Operators Manual" is included as Appendix D and describes system initialization together with all the major commands used for system control , data entry , data display and editing and function execution. This appendix also contains the Controller program listing in the form of 86 screens of FORTH word definitions.

Appendix E - "Vertichirp Controller Software - Program Glossary" gives detailed information on the operation of each defined FORTH word. The state of the FORTH parameter stack is shown both before and after word execution. Where relevant the numeric calculator (MPN) stack contents are also shown.

5.2 Vertichirp Controller Program Structure

The major functional areas of the Vertichirp Controller program are shown in figure (5.1). Each block is described in detail in the following sections.

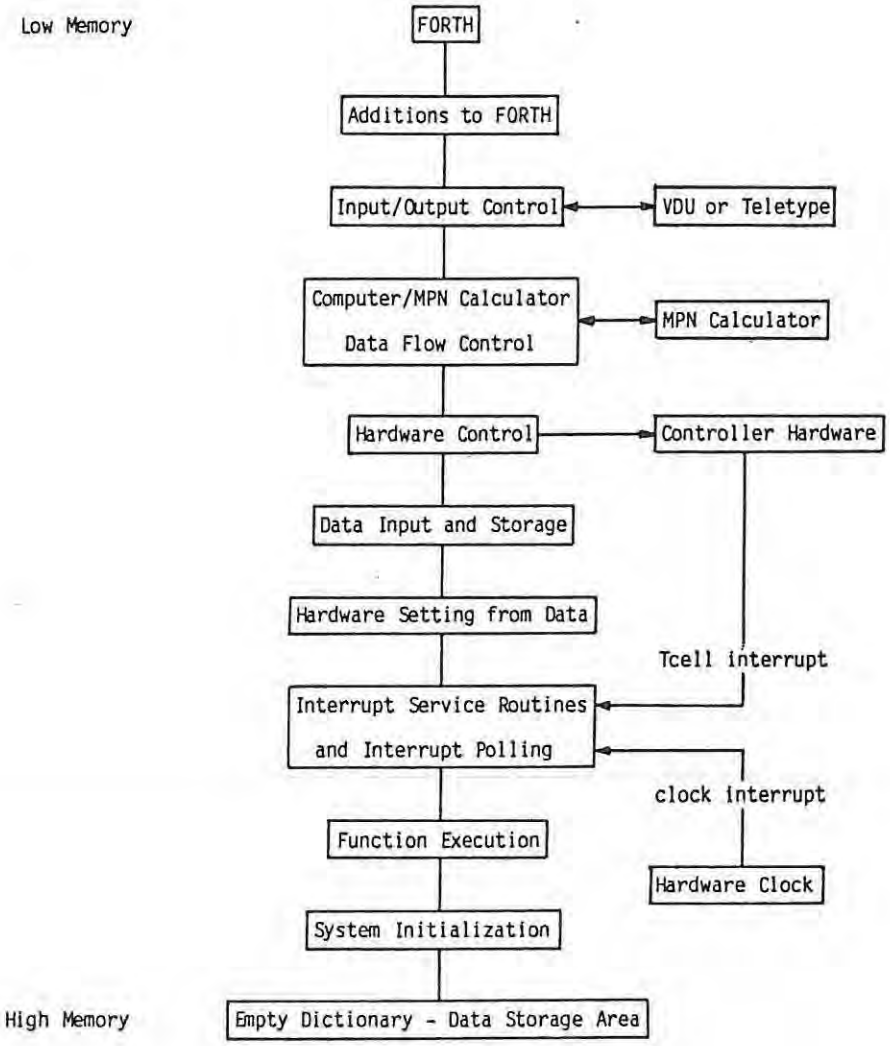


Figure (5.1) Program Structure

5.3 FORTH - An Introduction

FORTH [Rather and Moore , 1976] is a combined operating system , interpreter and compiler. It features an extensible command set which leads naturally to the development of an application oriented vocabulary. FORTH combines computational flexibility with compact code and high speed and was specifically designed as a control language.

The FORTH executive coordinates a large number of small modular routines which are defined as "words" in a dictionary. The dictionary is therefore a threaded list of variable-length entries. Programming in FORTH consists of defining new words in terms of previously defined words. Once defined a new word can be tested immediately by typing it at the terminal.

FORTH is a stack language and makes use of two last-in first-out (LIFO) stacks. The variable or parameter stack is used for calculations (using reverse Polish notation) and the return stack is used for program control.

5.4 Additions and Changes to FORTH

In developing the Controller software it became apparent that certain additions to the FORTH kernel were required. By relocating the disk buffer which was situated after the kernel at \$1BF0 , 1040_{10} bytes of memory were made available for these additions.

The program FORTHAD was written to allow primitives (words coded in assembly language) and FDB- (form double byte) coded words to be added to the FORTH kernel. FORTHAD was used in developing both Controller and DCS

(Data Capture System) software.

Seven different sets of FORTH words were added to the kernel and each set is documented in Appendix C in the following form :-

1. Assembler listing of primitive- or FDB-coded words
2. Glossary describing the operation of each word and its action on the parameter stack.

Fourteen double-precision operators were added to facilitate memory , stack, arithmetic and logical manipulation of 32-bit numbers.

The original 68'FORTH for 6809 did not include words for interrupt processing. The words RTI (return from interrupt) , CLI (clear interrupt mask) , SEIF (set I and F interrupt mask bits) and IRQ (vector FORTH execution to an interrupt service FORTH word) were added to the kernel as primitives. An example of using interrupts is given in Appendix C.

A SWTPC Numeric Calculator board (MPN) was included in the system (I/0 port 0) to provide a fast means of evaluating system parameters. The calculator functions multiply , divide , add , reciprocal , exponential and natural log proved invaluable for 8-digit BCD calculations. Being BCD the calculator output required no processing except for the packing of 2 BCD digits per byte prior to outputting to the frequency synthesizer control hardware on PC1. The calculator is used in scientific mode only. Primitive TOMPN sends data or instructions to the calculator board. Primitive ANS transfers the calculator x register to computer memory.

The primitive VECTOR (by A. Poole and documented by the author) was included to allow vectored FORTH word execution to simplify certain procedures.

Program development was done on drive 1 which was not initialized at power on. The primitive RESTORE1 causes disk drive 1 to do a "seek track 00" when executed. This resets the track counter and ensures that subsequent disk accesses are performed correctly.

A CASE structure for FORTH [Eaker , 1980] was initially written using colon definitions (the usual method of writing FORTH code). The words defined were CASE , OF , ENDOF and ENDCASE. In use the colon definition OF generated 10 bytes of code. The case structure was made more memory efficient by defining a run time primitive (OF) so that each OF would only generate 4 bytes. The entire CASE structure was added to the FORTH kernal.

Comparison words < and > which operate on signed numbers were used for checking the limits of the dictionary and parameter stack. Unsigned comparison words U< and U> were defined to allow positioning of the stack and dictionary above 8000_{16} (32 K).

The file CHANGES documents all the changes made to the original version of FORTH. These are :-

1. FORTH words changed for operation above 32 K.
2. FORTH memory map changes.
3. Correction to the word +LOOP
4. Changes to allow EPROMing of FORTH

The assembled binary file is used as an overlay to effect the changes.

The procedure for generating the Controller version of FORTH is then detailed together with the resultant expanded FORTH memory map. The overall system memory map is also given. The system monitor , S-BUG , was modified to set the dynamic address translator so that physical addresses were equal to logical addresses. S-BUG was also changed to vector to the FORTH warm start address at power on if FORTH was in EPROM. If FORTH was not found the monitor program was executed. The non-maskable interrupt (NMI) vector was changed to vector to the FORTH cold start address.

The procedure for saving the EPROM version of the Controller program is given in detail. The final memory map shows the EPROM/RAM allocations for the Controller system.

5.5 Input/Output Control

The Controller software was written primarily for a Heath VDU. The direct cursor addressing facility of the VDU is used to maintain 3 separate areas on the screen as shown in figure (5.2)

Lines 1 to 6	System Status Area
Lines 7 to 18	Data Display Area
Lines 20 to 24	Data and Command Input Area

Figure (5.2) Heath VDU Screen Areas

The system status area occupies 7 lines and displays place name , year, year day-number , the current programme being run , the current and next function to be executed in that programme and the system timing position.
The major part of the screen , lines 8 to 18 , is used for data display and

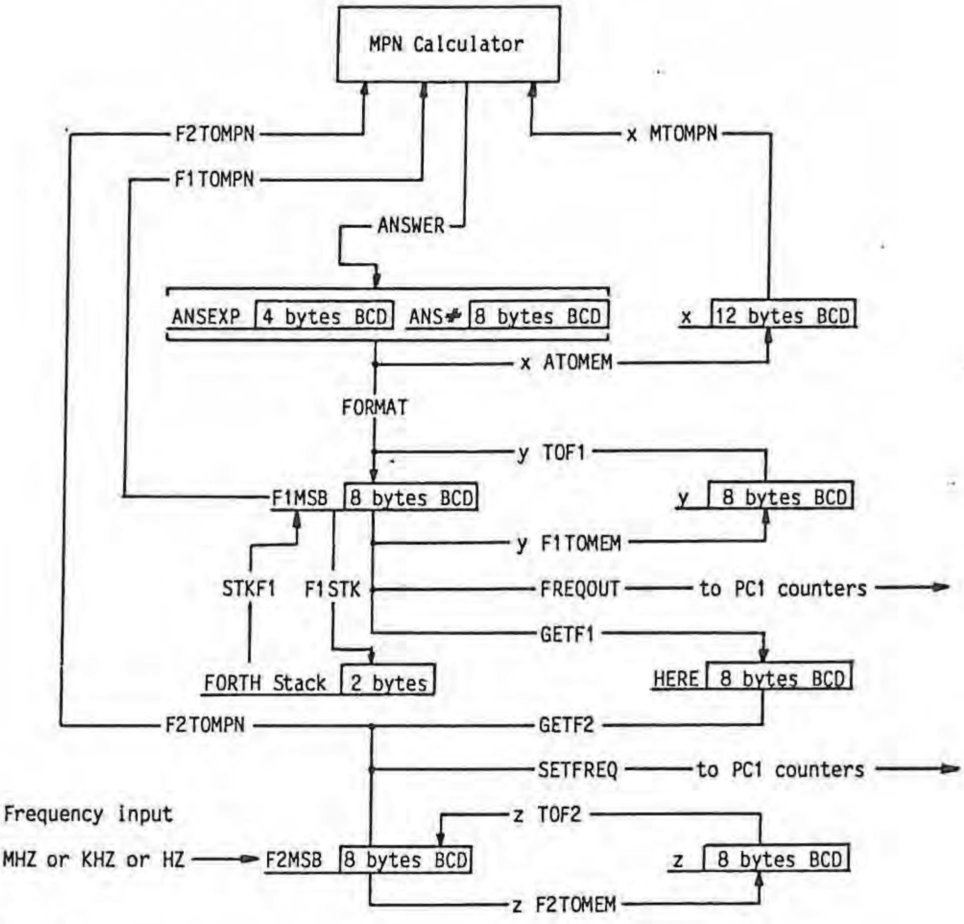
editing. Lines 20 to 24 are used for command and data entry.

Input/output routines that do not use direct cursor addressing were included to allow the use of any VDU or teletype. In the so called "Printer" mode all input and output data and commands are listed sequentially. Commands are provided to enable and disable the printing of system status.

5.6 Computer/MPN Data Flow Control

Answers from the MPN calculator x-register are stored in 12 bytes of computer memory beginning at address ANSEXP. The first two bytes contain the exponent , the next two the signs of exponent and mantissa and the last 8 , beginning at address ANS#, contain the mantissa.

Figure (5.3) shows computer memory locations ANSEXP , ANS#, F1MSB , F2MSB , x , y , z , HERE and the FORTH stack together with the FORTH words defined to move numbers between them. The mantissa and exponent of the answer buffer are formatted into an 8 digit BCD number in Hz which is stored in frequency buffer 1 , F1MSB. F1MSB is used during ionogram setup and execution. Frequency buffer 2 , F2MSB , is used for frequency input. Having two buffers allows data editing while an ionogram is executing.



Notes

1. Computer memory storage depicted as follows :- address | n bytes data |
2. The FORTH words associated with all data transfers are given in capitals

Figure (5.3) Computer/MPN Data Flow Control

5.7 Hardware Control

The hardware control words input data and then convert them into codes suitable for use by the hardware. These codes are listed after each printed circuit board description in Appendix A.

Ionogram execution is controlled by data written to two 8 bit registers , Control Register A and Control Register B on PC3. Control Register A is set according to the type of ionogram required such as Vertical , Oblique receive , Stationary Doppler etc. Once set the data in this register are not changed for the duration of the ionogram. Control Register B is used for data that may change during ionogram execution such as enabling/disabling the film drive or transmitter. Control Register B software is written to facilitate the changing of individual control bits.

The parameter setting software checks for valid input data. If the data are valid conversion to code takes place and the result is output to the hardware. This software makes hardware setting and testing very easy :-

Command	Function
50 KHZ/SEC SETRATE	Set basic rate (k_B)to 50 kHz/s
4 DOP SETDOP	Set Doppler offset frequency to 4 Hz.
1/ TCELL SETCELL	Set cell Period (T_C) to 1 s
1/2 MSEQ SETMSEQ	Set M-sequence period to 1/2 s
1024 FFT SETFFT	Set FFT sample rate to 1024 Hz.
3 4 ANTENNA SETANTENNA	Rx1 to antenna 3 and Rx2 to antenna 4
5 MM/MIN SPEED SETSPEED	Selects a film speed as close to 5 mm/min as possible.

The advance/retard software allows system timing changes to be made in ten microsecond increments with the initial and final positions being displayed on the screen. Three advance slip rates and three retard slip rates are provided. After every millisecond of shift the terminal input buffer is checked for operator break-in. Shift commands take the following form :-

Command Line	Function
10.25 MSEC A@1	10.25 ms advanced at a rate of 1 ms/s
15.0 MSEC A@10	15.0 ms advanced at a rate of 10 ms/s
7.5 MSEC A@20	7.5 ms advanced at a rate of 20 ms/s
12.5 MSEC R@1	12.35 ms retarded at a rate of 1 ms/s
5.0 MSEC R@10	5.0 ms retarded at a rate of 10 ms/s
50.5 MSEC R@20	50.5 ms retarted at a rate of 20 ms/s

In the SEARCH mode all that is required to sweep the timing is one of the shift commands A@1 through to R@20. The SEARCH mode is used for locating the oblique incidence signal.

5.8 Data Input and Storage

Data are stored in the FORTH dictionary as a list of hardware codes headed by a unique name. To facilitate handling each data set was assigned a "datatype" code and a "subtype" code.

Datatype 1 - SOUNDING data (a subset of ionogram data)
Subtypes - 0 = Sounding for normal ionogram.
1 = Sounding for Stationary Doppler ionogram.

Datatype 2 - IONOGRAM data (the sounding that the ionogram uses must exist before the ionogram is defined).

Subtypes - Control register A data in the range 0 to 255_{10}
e.g. 255 = Vertical ionogram
124 = Stationary-Doppler ionogram

Datatype 3 - TIMING Slip Data

Subtypes - 0 = Advance
1 = Retard

Datatype 4 - FORTHWORD Data (The parameter field address (pfa) of a previously defined FORTH word is all the data that is stored after the data type code. This system "FORTH Word" was created to allow new operations to be defined and executed under programme control.

Subtypes - None , set to 0

Datatype 5 - PROGRAMME (A prioritized list of times of execution and functions to be executed)
Note :- A "programme" of events is distinguished from a computer "program" by the difference in spelling.

Subtypes - None , set to 0.

Sounding Input - Type 1 Data

Command :- SOUNDING x where x = a unique user-assigned data name.
In practice sounding names were given the prefix "S" e.g. S1 .
Figure (5.4) shows the flowchart for inputting sounding data named S1 .

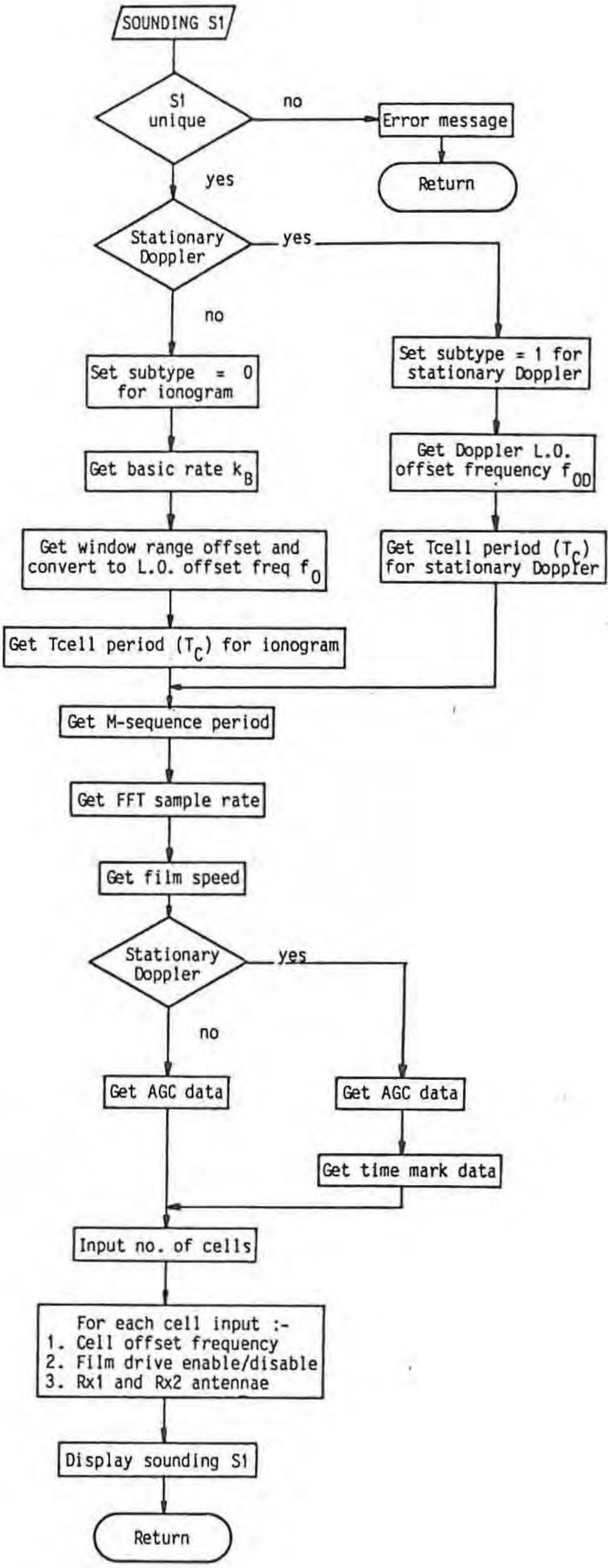


Figure (5.4) Input Sounding Data

Ionogram Input - Type 2 Data

Command :- IONOGRAM x where x = unique user-assigned data name.

Figure (5.5) gives the ionogram input flowchart. The data are named I1.

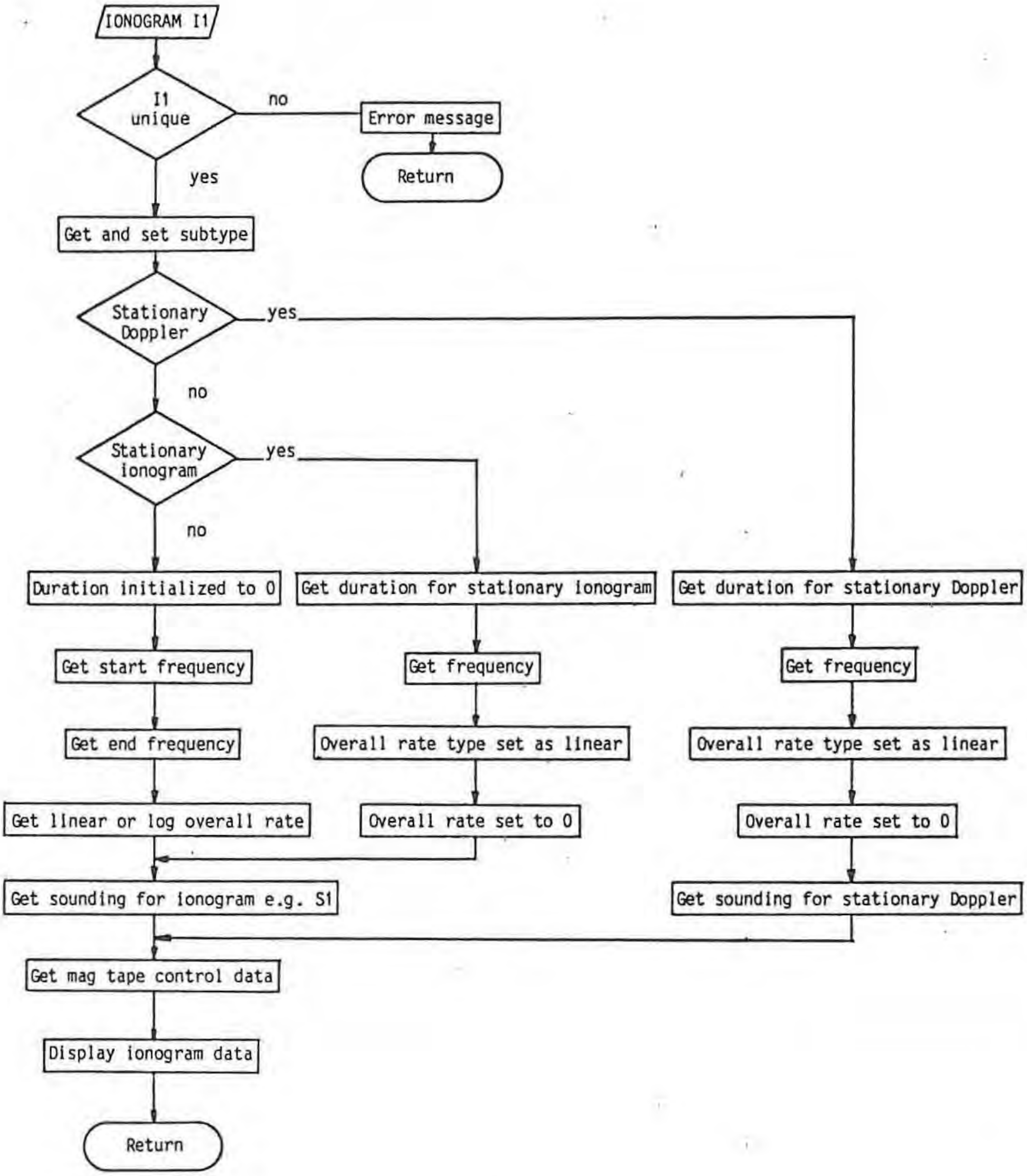


Figure (5.5) Input Ionogram Data

Timing Slip Input - Type 3 Data

Command :- TIMING x where x = unique user-assigned data name

Figure (5.6) shows the flowchart for entering timing slip data named T1.

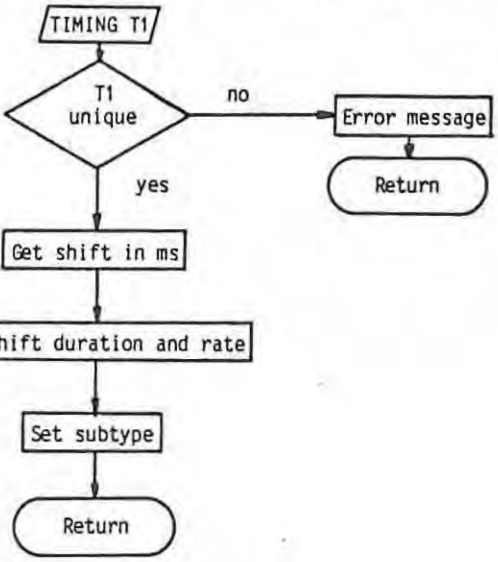


Figure (5.6) Input Timing Slip Data

FORTH Word Input - Type 4 Data

Command :- FORTHWORD x where x = unique user-assigned data name.

Figure (5.7) shows the flowchart for FORTH Word input , the defined system Word being called F1.

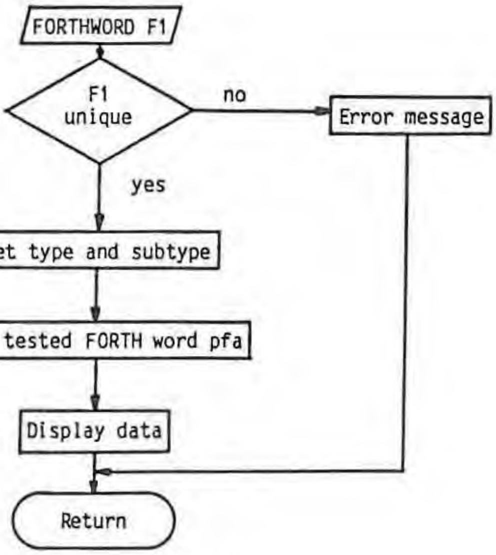


Figure (5.7) Input FORTH Word data

Programme Input - Type 5 Data

Command :- PROGRAMME x where x = a unique user-assigned data name.

Figure (5.8) shows that all entries are edited into initialized programme space.

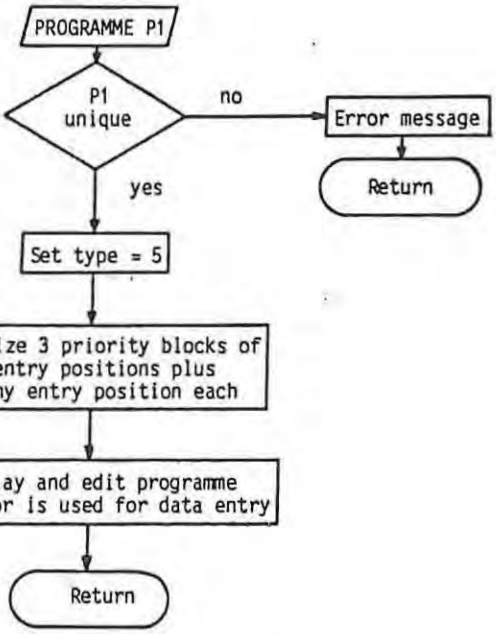


Figure (5.8) Input Programme Data

Data Editing

Command :- EDIT x where x = a data name entered using the above commands.

Any of the input data can be edited except for the number of cells per sounding.

E.g. EDIT S1 , EDIT I1 , EDIT T1 , EDIT F1 , EDIT P1

Delete Data

Command :- DLIST

DLIST lists all the entered data names and allows their erasure. See Appendix C for a full description of DLIST.

5.9 Hardware Setting from Stored Data

Data types 2 , 3 , 4 and 5 can all be "set" and executed as "functions". Sounding data , being a subset of ionogram data , cannot be executed as a function but is used to set the system hardware prior to ionogram execution.

Set Sounding

Routine SSET of figure (5.9) requires the parameter field address (pfa) of the data (e.g. pfa of S1) on the stack and leaves a true flag if setting is successful.

Set Ionogram

Routine ISET (figure 5.10) requires the pfa of ionogram data on the stack (e.g. pfa of I1). This routine uses SSET and then calculates ionogram control parameters.

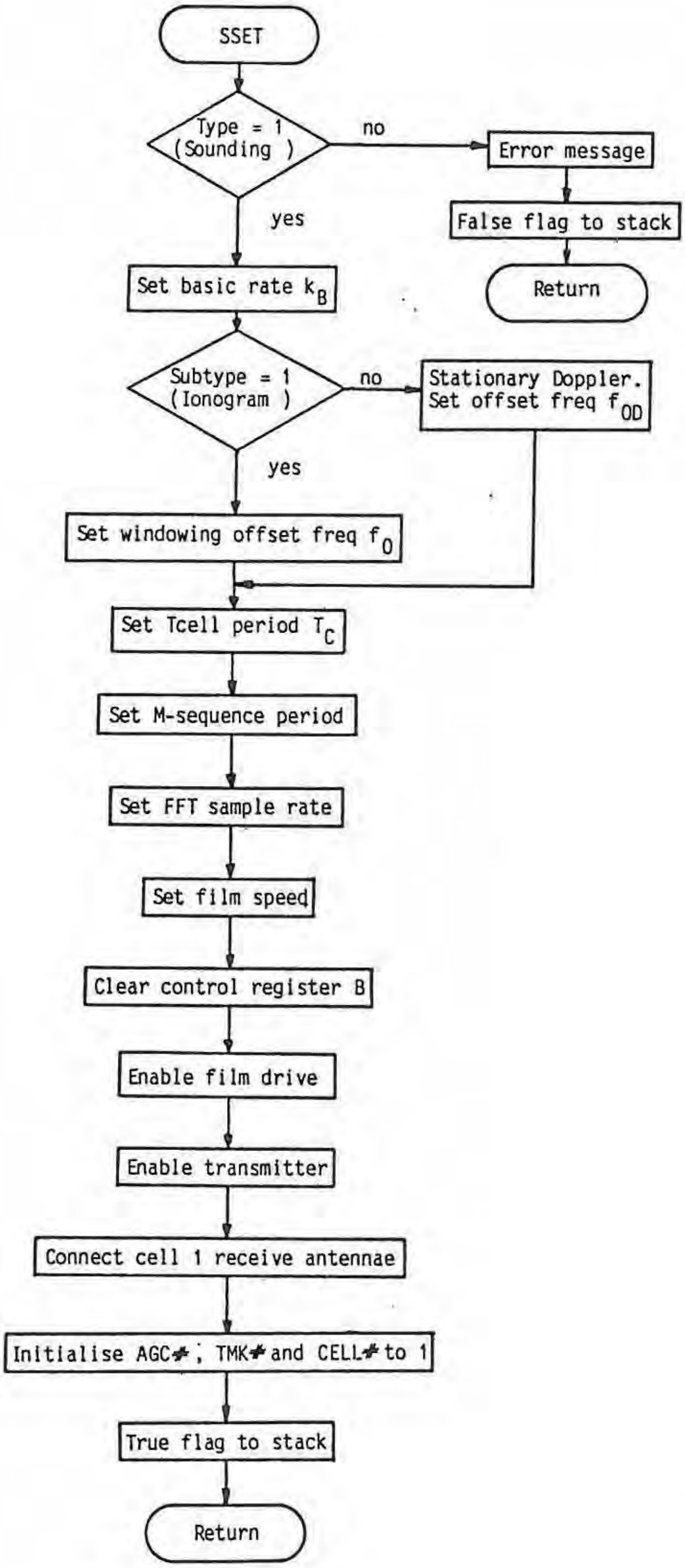


Figure (5.9) Set Sounding

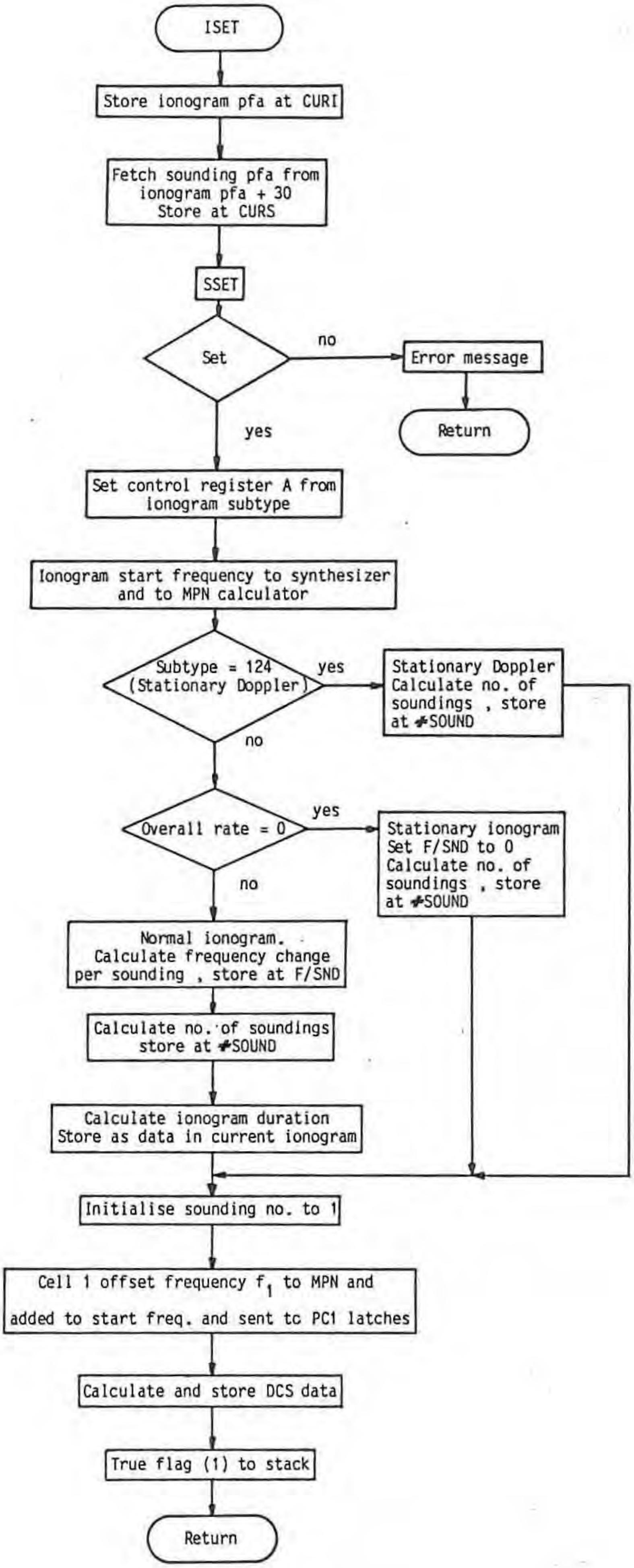


Figure (5.10) Set Ionogram

Set Timing Slip

Figure (5.11) shows the routine TSET which requires the pfa of timing slip data on the stack prior to execution.

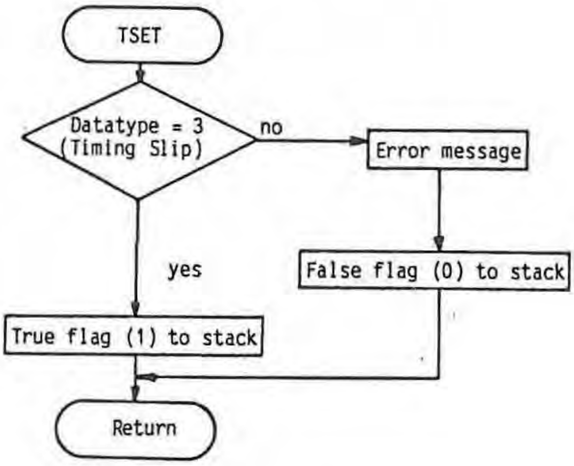


Figure (5.11) Set Timing Slip

Set FORTH Word

The flowchart of routine FSET is given in figure (5.12). The pfa of a system defined FORTH Word is required on the stack before FSET is executed.

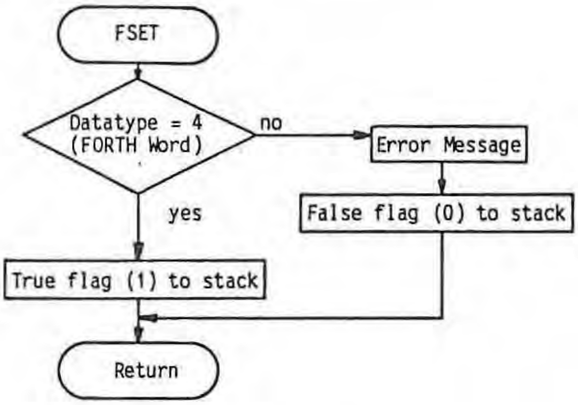


Figure (5.12) Set System defined FORTH Word

Set Programme

The flowchart of PSET is given in figure (5.13) and requires a pfa on the stack prior to execution.

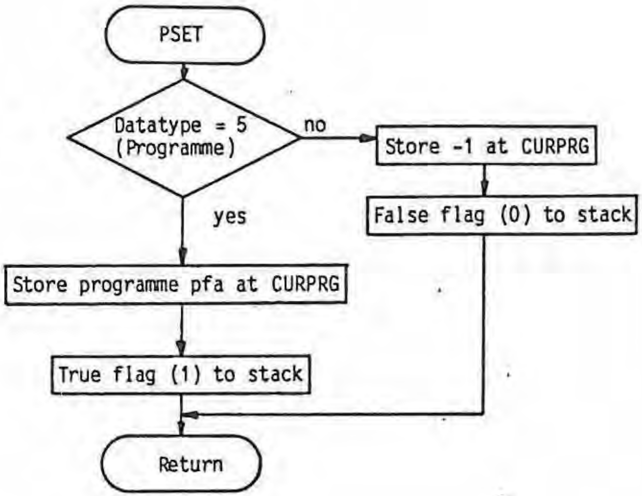


Figure (5.13) Set Programme

5.10 Interrupt Service Routines and Interrupt Polling

There are two signals that can generate interrupts , the 1 Hz clock signal negative edge and the Tcell pulse positive edge. The Tcell pulse only occurs when a sweep is in progress.

Clock Interrupt Service Routine

This routine called CLOCK is flowcharted in figure (5.14).

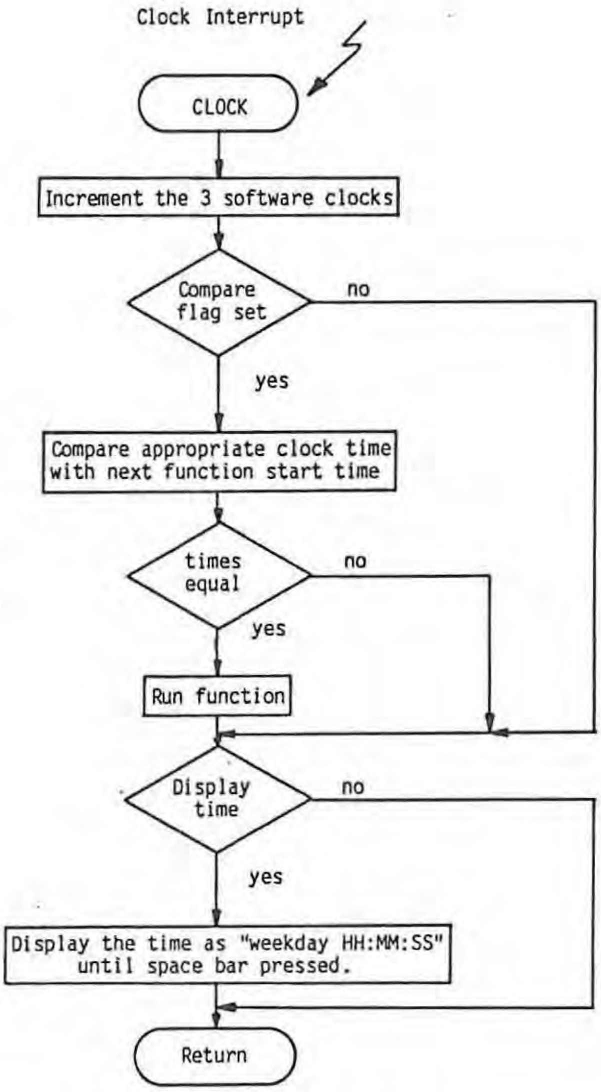


Figure (5.14) Clock Interrupt Service Routine

Tcell Interrupt Service Routine

This routine , NCELL , controls ionogram execution as shown in figure (5.15).

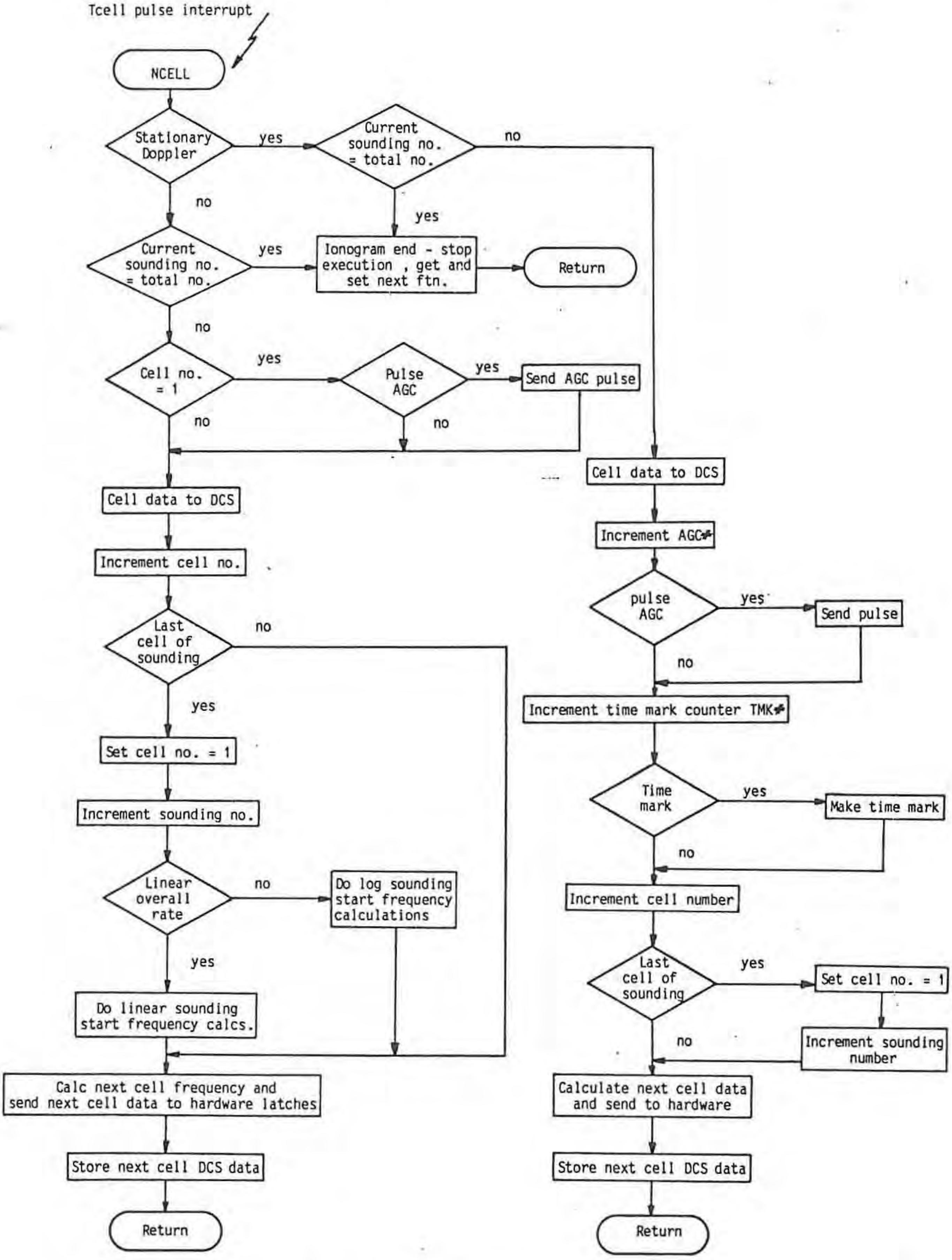


Figure (5.15) Tcell Interrupt Service Routine.

Interrupt Polling

Both the Tcell interrupt flag and the clock interrupt flag are checked whenever there is an interrupt. Tcell interrupts are given the highest priority as shown in figure (5.16).

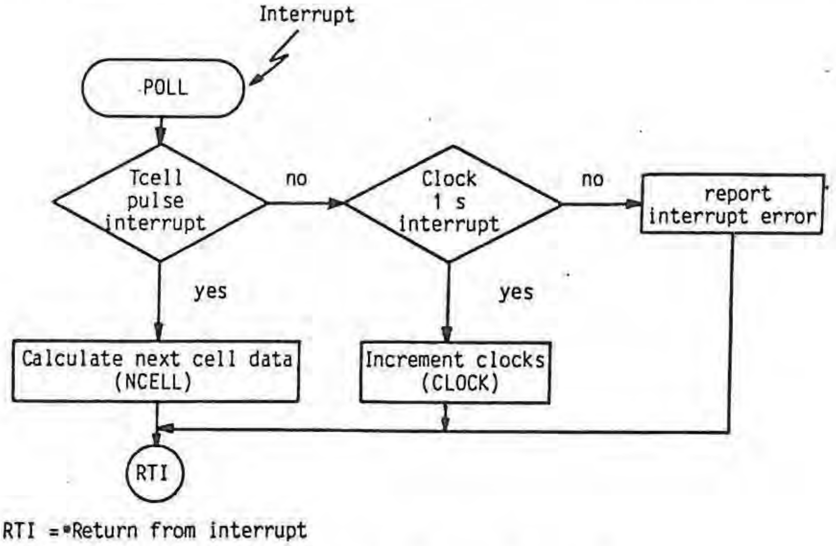


Figure (5.16) Interrupt Polling Routine

5.11 Function Execution

Ionogram , Timing Slip , FORTH Word and Programme data can all be executed as functions. Sounding data , being a subset of the ionogram data , cannot be executed.

Run Sounding

The routine RS of figure (5.17) is included for completness only. It expects a pfa on the stack which is dropped and an error message is output.

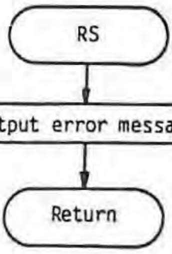


Figure (5.17) Run Sounding

Run Ionogram

RI expects the pfa of ionogram data on the stack and initiates an ionogram sweep. Prior to executing RI the routine ISET must be executed to set the hardware and calculate run parameters. RI is flowcharted in figure (5.18).

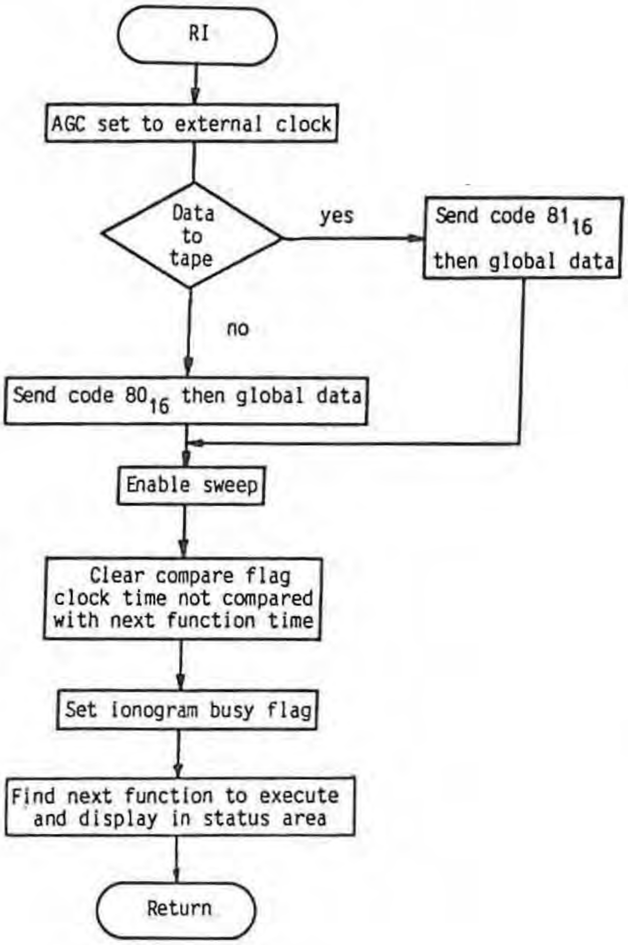


Figure (5.18) Run Ionogram

Run Timing Slip

RTS expects the pfa of timing-slip data on the stack and executes the slip with operator intervention disallowed.

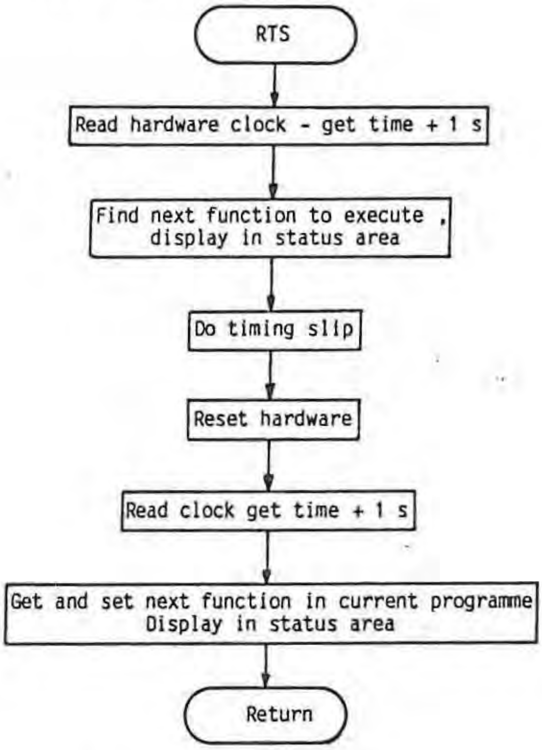


Figure (5.19) Run Timing Slip

Run FORTH Word

RFW expects the pfa of a system-defined FORTH Word on the stack and executes the normally defined word that was stored as data (see FORTHWORD x). RFW is flowcharted in figure (5.20).

Run Programme

RPRG expects the pfa of a programme on the stack. The programme is searched for the next execution time and the corresponding function is set. See figure (5.21)

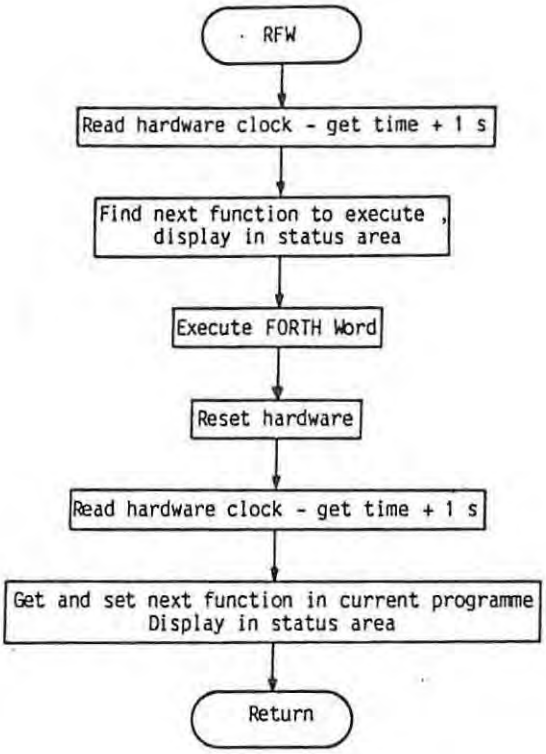


Figure (5.20) Run FORTH Word

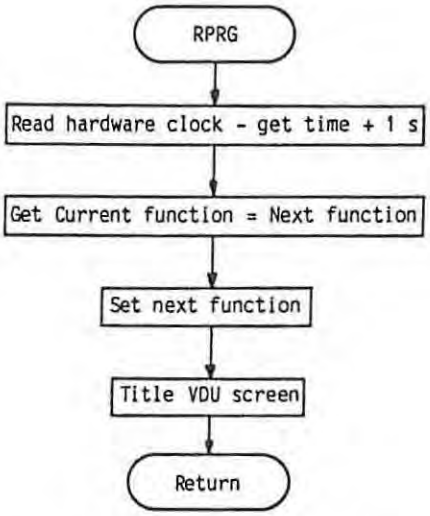


Figure (5.21) Run Programme

The RUN Command

Command :- RUN x where x can be a sounding , ionogram , timing slip , FORTH Word or programme data name.

Note that RUN S1 where S1 is a sounding name gives an error message only.

In the flowchart of figure (5.22) SETFTN and RFTN use the primitive definition VECTOR to execute the correct routine according to data type.

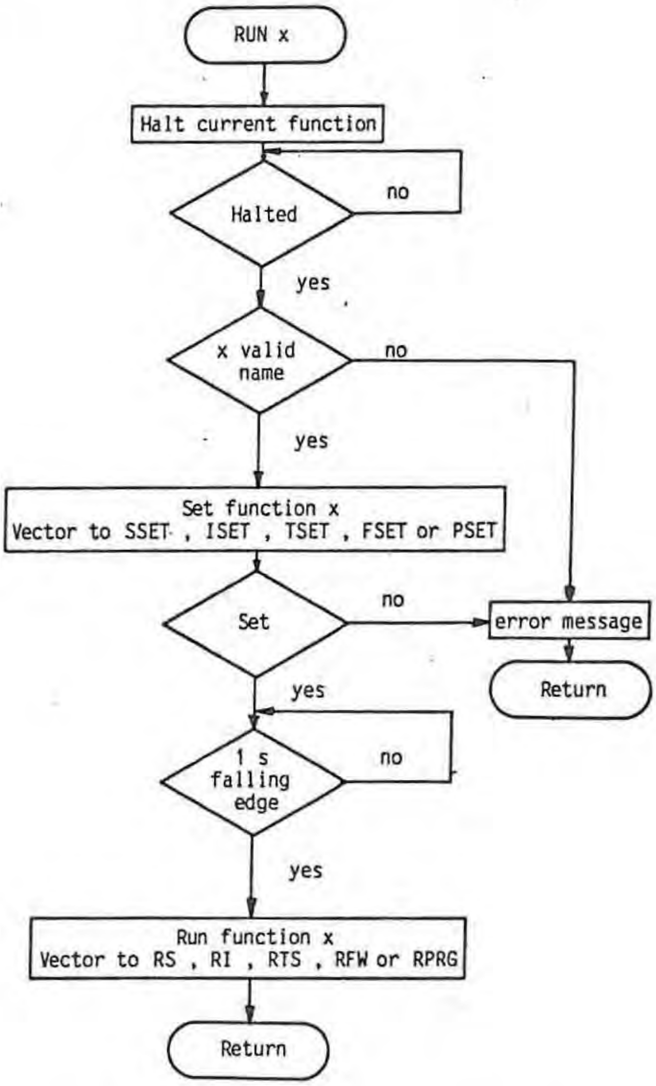


Figure (5.22) The RUN Command

Programme Execution

When a programme is run (e.g. RUN P1) it is made the current programme and the data it contains are checked to find the next function to execute. The flowchart of the routine that does this (SETNXT) is shown in figure (5.23)

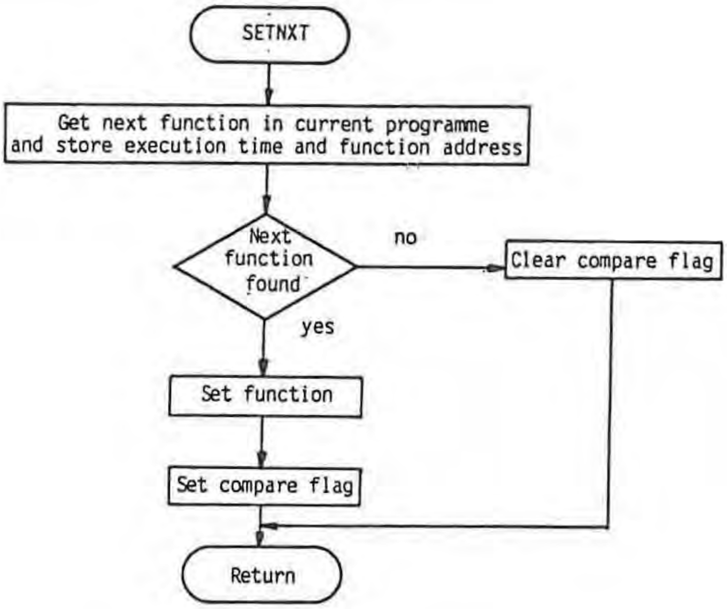


Figure (5.23) Routine SETNXT

Once the next function has been found the time at which it is to be executed is stored and compared with the appropriate software clock every second. All three software clocks are 1 s in advance of the actual time. Equality is therefore detected 1 s early and the function is enabled for execution. Execution takes place synchronous with the next clock pulse.

5.12 Function Execution Timing

Figure (5.24) shows the timing diagram for ionogram execution under programme control.

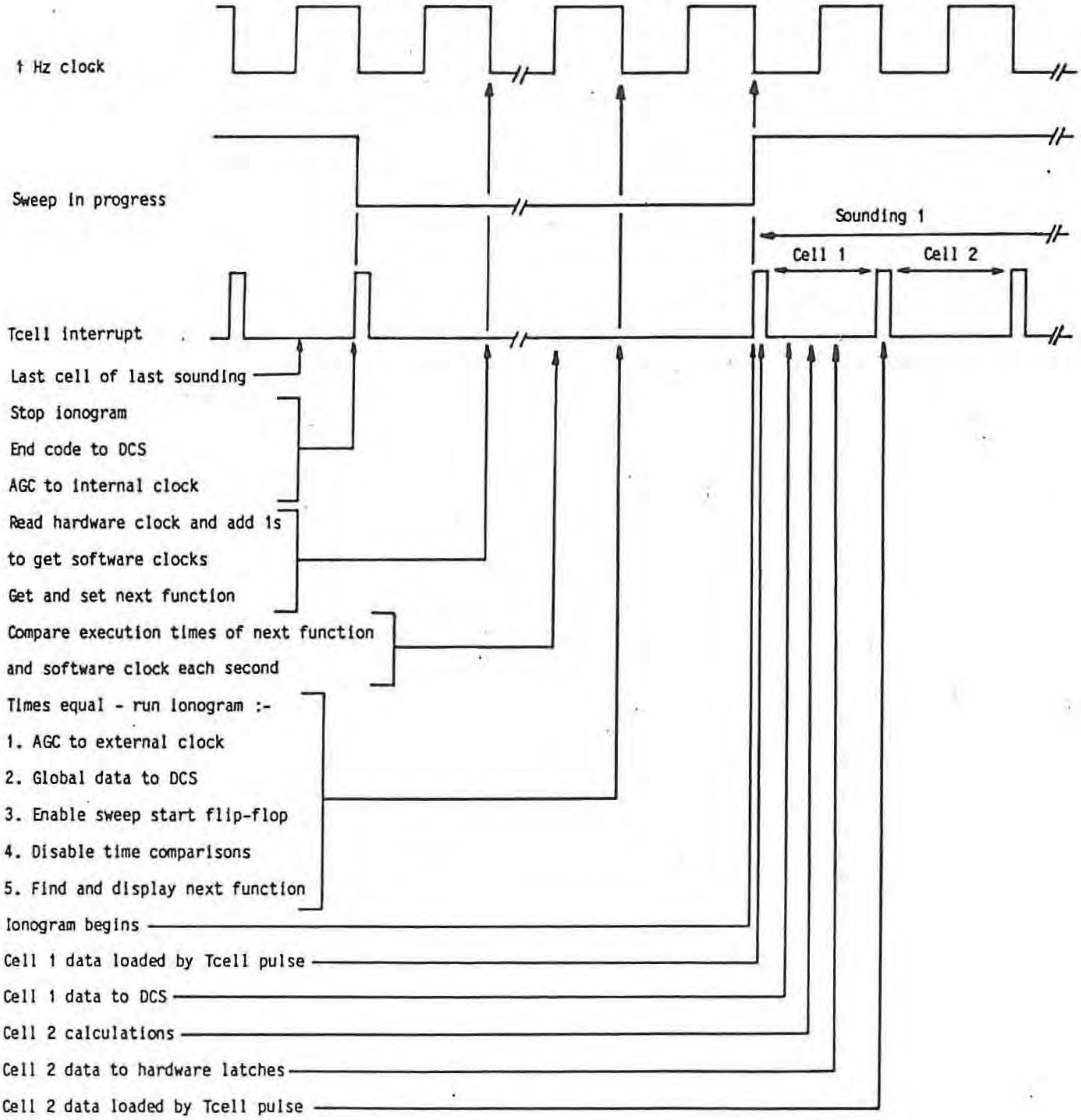


Figure (5.24) Ionogram execution

An ionogram stops once the calculated total number of soundings have been completed. The real-time hardware clock is read on the next falling edge of the 1 Hz signal and immediately incremented to get the three software clock times which are all 1 s in advance of the real time.

The next ionogram in the current programme is then found and set and its execution time is compared with the appropriate software clock each second. When the times are equal ionogram execution is enabled and global data are sent to the Data Capture System. The ionogram begins at the programmed time on the next negative-going edge of the 1 Hz signal. Time comparisons are disabled and the ionogram continues to completion. Th current and next functions are displayed in the status area.

The Tcell pulse signal is gated on by sweep in progress going high. The Tcell interrupt service routine sends the current cell data to the DCS before calculating data for the next cell. Next cell start frequency and antennae connections are output to the hardware and latched ready to be loaded by the next Tcell pulse.

Execution of a timing slip , a system FORTH Word or a programme under programme control is similar to that of an ionogram. Because equality of programmed time and software clock is detected 1 s early the software waits until the next 1 Hz signal falling edge before executing the function.

If the function is a programme it is made the "current" programme and the next function to be executed in the new programme is found and set.

CHAPTER 6

SYSTEM TESTS

6.1 Introduction

During 1983 the different stages of the modified Chirpsounder (figure (4.1)) were tested. Initial tests involved only the replacement of the 1015 Logchirp Control with the new computer-based Controller with ionograms being recorded on film. A number of these tests are reported here.

The testing then progressed to include a new receive antenna system designed by Poole who also modified the dual phase-matched receivers. Phase calibration of the signal path from the antenna cables through the receivers was done [Poole ,1983] using the single sideband circuits (PC9 , PC10 and B3) designed by the author. Poole designed the hardware and software of the data capture system (DCS) which stores data on magnetic tape for later offline analysis.

Since digital recordings were the prime object of the modified system numerous tests and experiments were executed and analysed by Allon Poole to determine system performance [Poole , 1983 , Chapter 5]. An example of the digital capabilities of the system is presented.

Ionosonde operation on the S. A. Agulhas is described with the emphasis being on Controller programming versatility.

6.2 Controller Initial Tests

The first requirement of the Controller was that it should be a direct functional replacement for the 1015 Logchirp Control (see Chapter 3). Initial tests involved using the Controller to produce vertical- and oblique-incidence ionograms on film according to the "normal" recording schedule as given in section 2.4. Examples of 1015 Logchirp Control and Controller produced vertical-incidence ionograms are given in figures (6.1) and (6.2).

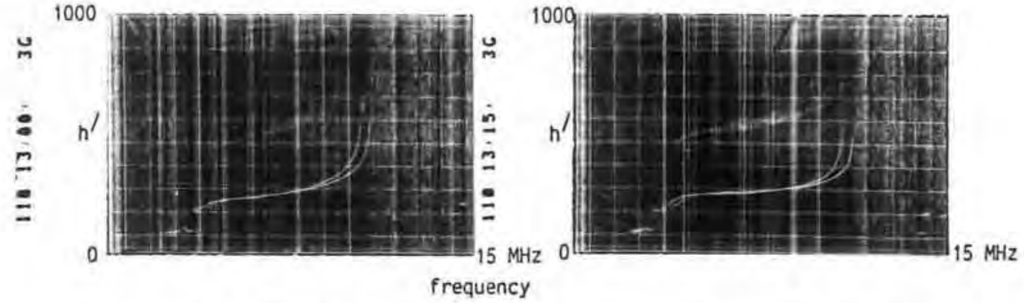


Figure (6.1) Vertical Incidence Ionograms - 1015 Logchirp Control

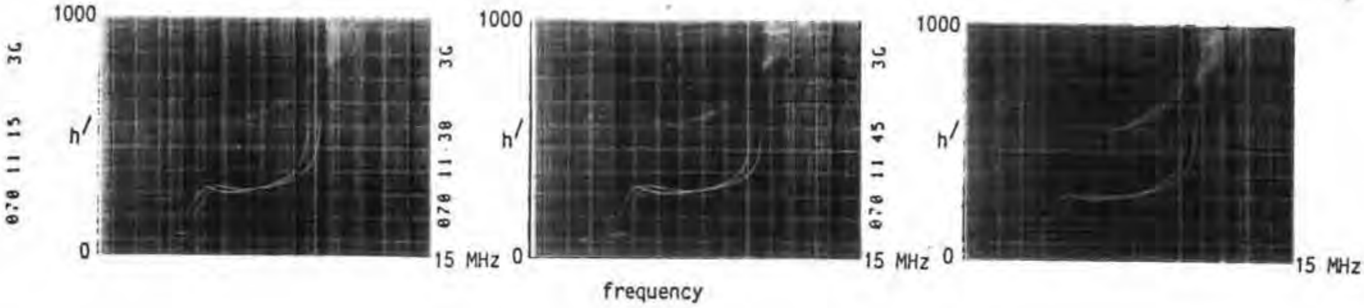


Figure (6.2) Vertical Incidence Ionograms - Computer Controller

The Controller-produced vertical-incidence ionograms compare excellently with those obtained using the 1015 Logchirp Control unit.

Figure (6.3) shows 2 SANAE - Grahamstown oblique-incidence ionograms recorded using the 1015 Logchirp Control unit.

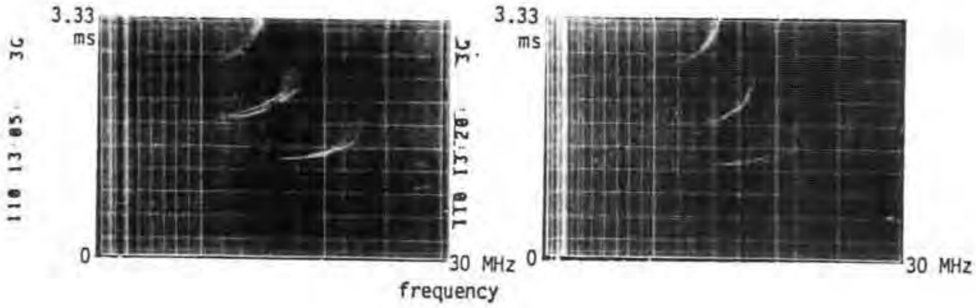


Figure (6.3) Oblique Incidence Ionograms - 1015 Logchirp Control

Figure (6.4) shows a sequence of oblique-incidence ionograms for the same path recorded using the Controller.

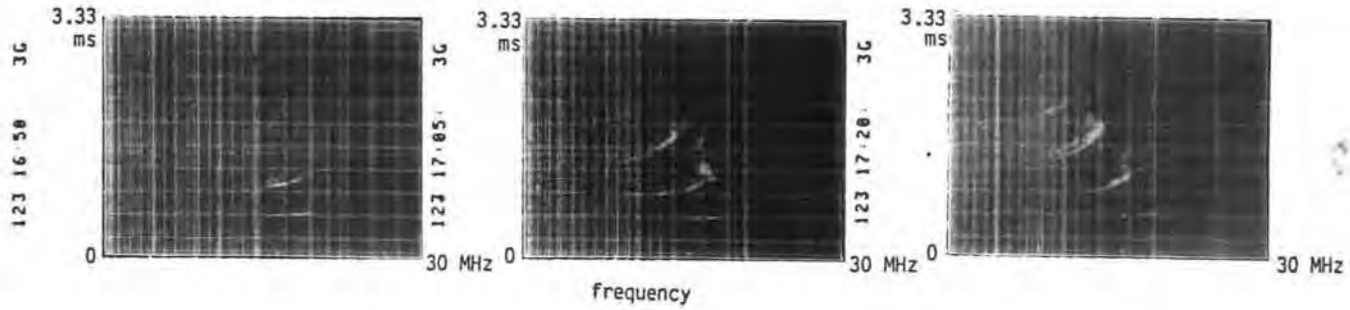


Figure (6.4) Oblique Incidence Ionograms - Computer Controller

The ionograms shown above verify that the Controller operates successfully in the oblique-receive mode. The oblique-transmit mode was also tested with the signal being received at Sanae.

Besides functionally replacing the Logchirp Control unit the Controller provided far more flexible control of the ionosonde. For example ; the extended programming capabilities of the Controller were used to perform an automatic oblique calibration sequence once a week between 14h05 and 15h55 , a task that was previously done manually. The Advance/Retard hardware and software proved most efficient for system timing changes and oblique signal synchronization.

6.3 Basic Rate Tests

Figure (6.5a) shows a vertical-incidence ionogram with basic and overall sweep-rate equal to 50 kHz/s. Doubling the basic rate doubles the frequency difference for a given delay (equation (2.5)) and halves the ionogram virtual height range (equation (2.6)) as shown in figure (6.5b). Examples of ionograms with basic rates of 100 kHz/s (6.5b) , 200 kHz/s (6.5c) and 400 kHz/s (6.5d) are shown. Note that different frequency ranges are swept.

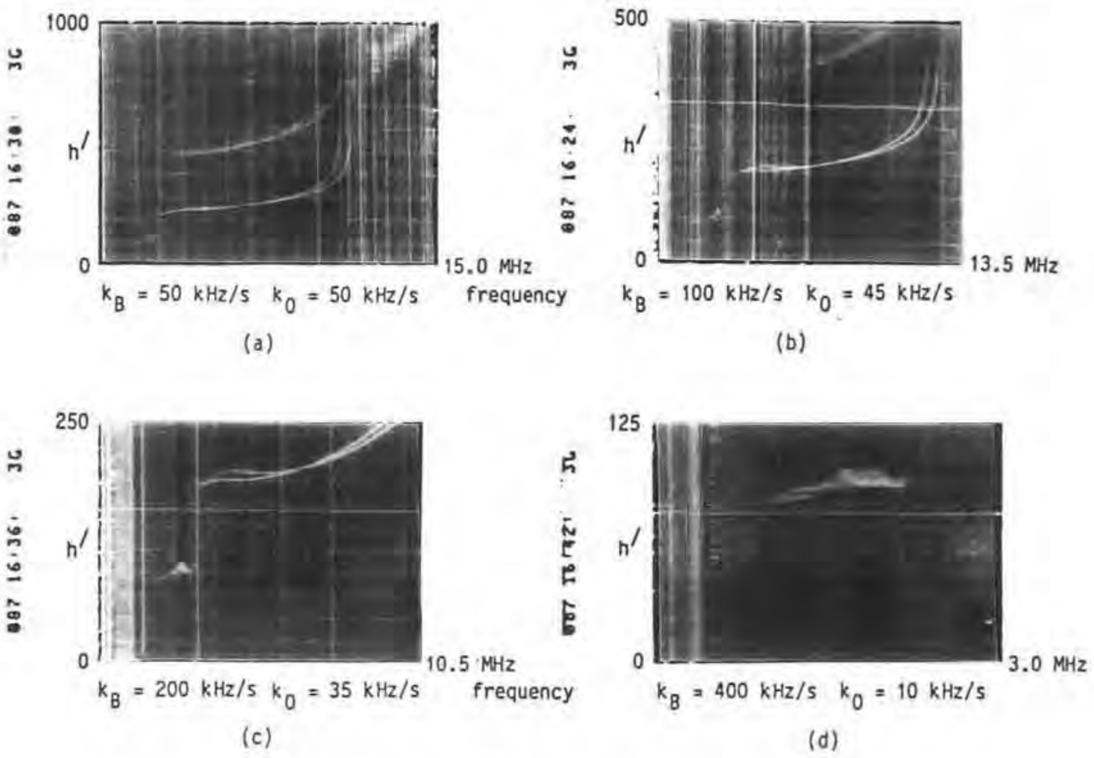


Figure (6.5) Vertical Incidence Ionograms - Basic Rate changes

6.4 40 MHz Local Oscillator Offsetting

Stationary Doppler

For stationary-Doppler recordings a small offset f_{OD} is subtracted from the 40 MHz local oscillator. PC9 and PC10 together with the SSB generator circuit (B3) perform the frequency shift. This circuitry operated very well as the test stationary-Doppler ionogram indicates.

Figure (6.6a) shows a normal vertical ionogram ($k_B = k_0 = 50$ kHz/s) recorded at 12h00. The stationary-Doppler ionogram of figure (6.6b) was recorded at 6.5 MHz with the 40 MHz local oscillator offset by 4 Hz. An anti-aliasing filter set at 12.5 Hz was included between receiver and spectrum analyser which was set to analyse 10 Hz. The maximum observed Doppler shift of about 1/4 Hz corresponds to a line-of-sight velocity given by equation (4.16) :-

$$u = (1/4)(3)(10^8)/2(6.5)(10^6) = 5.7 \text{ m/s}$$

The preceding 5 quarter-hourly ionograms confirm that the reflecting layer at 6.5 MHz remains at about 300 km.

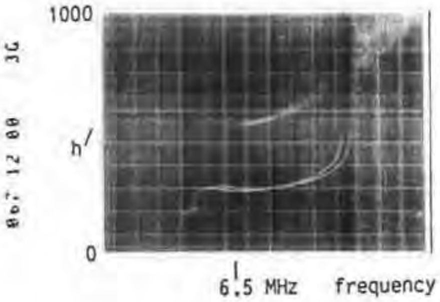


Figure (6.6a) Vertical Incidence Ionogram

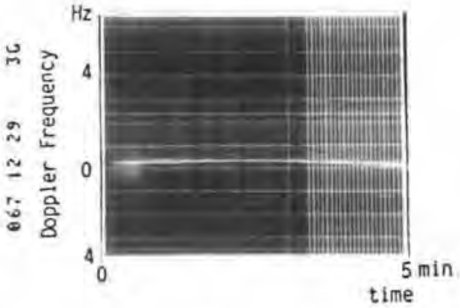


Figure (6.6b) Stationary Doppler Ionogram at 6.5 MHz

Windowing

$L02_{window}$ is obtained by selecting the upper sideband of L02 modulated by the windowing offset frequency f_0 (see chapter 4). The first three ionograms shown below were recorded with different basic rates and no range offset. The fourth has a range offset of 192 km.

Figure (6.7a) Vertical, $k_B = k_0 = 50$ kHz/s

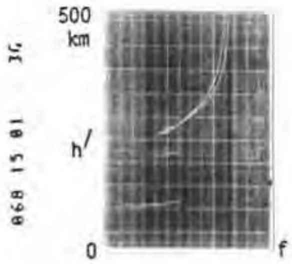


Figure (6.7b) Vertical, $k_B = k_0 = 100$ kHz/s

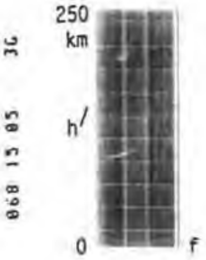


Figure (6.7c) Vertical, $k_B = k_0 = 200$ kHz/s

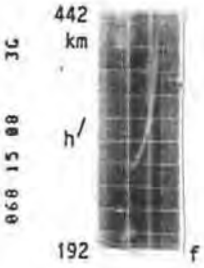
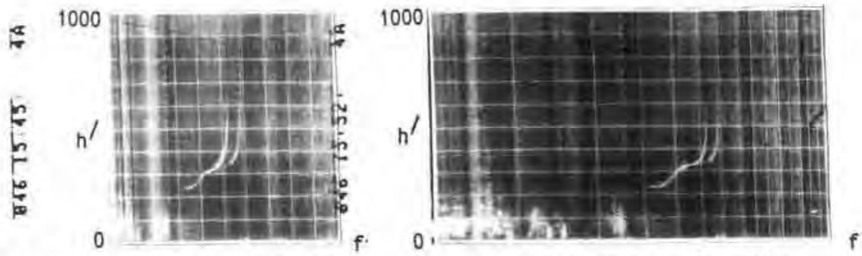


Figure (6.7d) Vertical, $k_B = k_0 = 200$ kHz/s,
L.O offset = 256 Hz $h'_{min} = 192$ km

In Figure (6.7c) the second order E-region reflection is just visible at a virtual height of 212 km. The same basic rate was used in figure (6.7d) but the 40 MHz local oscillator was offset by 256 Hz to give a window from 192 to 442 km. The second order E-region reflection appears at the correct height on the new range scale and the F-region trace is once again visible.

6.5 Logarithmic Sweep Rate Test

Figure (6.8) shows an ionogram recorded at Sanae with linear overall rate of 50 kHz/s followed by an ionogram with a logarithmic frequency scale recorded using a log overall rate of .01 oct/s.



Linear overall rate = 50 kHz/s

Log overall rate = .01 oct/s

Figure (6.8) Linear and Log Overall sweep rate

6.6 Stationary Ionogram Test

Figure (6.9) shows two vertical-incidence ionograms before and after a six-minute Stationary Ionogram which was recorded at 6 MHz , a frequency chosen because of good separation between the o- and x-mode reflections.

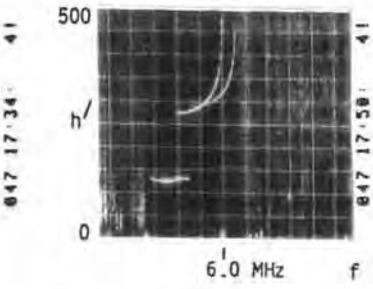


Figure (6.9a)

Vertical, k_B = 100 kHz/s

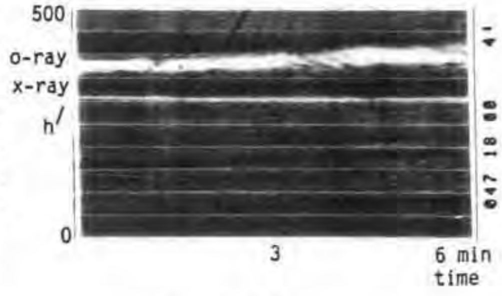


Figure (6.9b)

Stationary Ionogram at 6.0 MHz

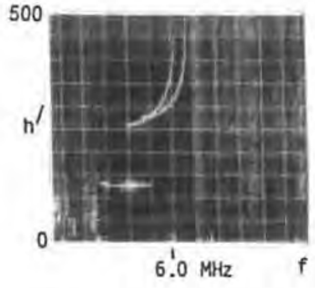


Figure (6.9c)

Vertical , k_B = 100 kHz/s

6.7 Operation as a Digital Chirpsounder

Besides having the capabilities as tested above the Controller was designed to allow the simultaneous evaluation of group range and phase velocity of the reflection point and the three-dimensional angle of arrival and polarization mode of the return signal.

To perform these measurements the Controller :-

1. sets the dual receiver AGC mode and signals when the AGC may change.
2. outputs a stop-sampling pulse to the FFT analyser at the end of each cell
3. sends global and cell information to the Data Capture System computer
4. selects the 2 receive antennae to be used during each cell.

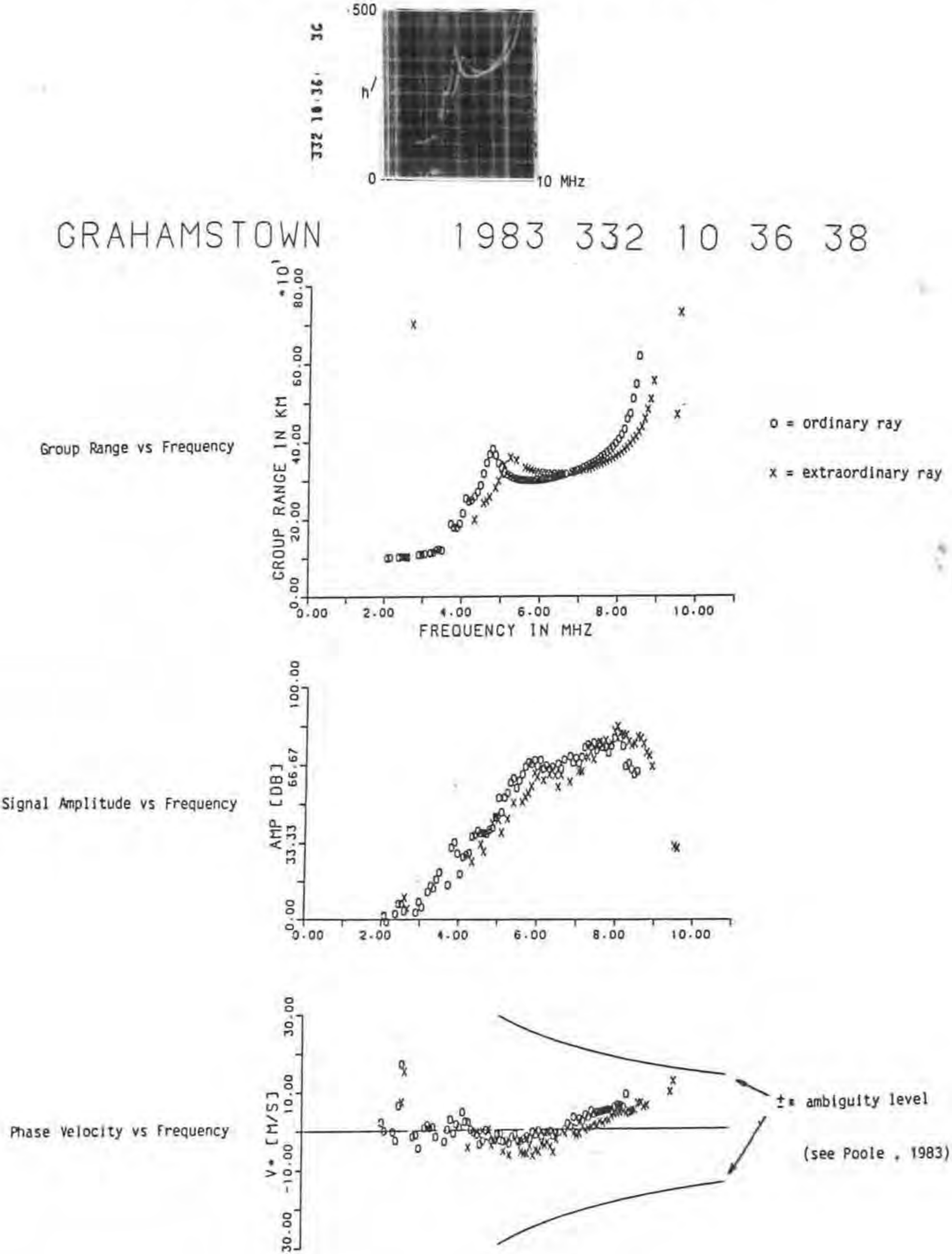
The DCS stores system and echo data on magnetic tape for later offline processing by mainframe computer. The film record provides a convenient means of selecting interesting ionograms for digital processing.

The LOG command was used to list ionogram data SYNOP/512 and associated sounding data S3/512. Note that digital data were to be recorded on magnetic tape and that the AGC was to be clocked once per sounding. The virtual height range given assumes analysis of the full 500 Hz baseband signal. Analysis of only 333.3 Hz reduces the maximum virtual height to 500 km.

```
LOG SYNOP/512
                              SYNOP/512 - IONOGRAM ( VERTICAL )

Ionogram Duration =  0 :  4 : 50
Data to Tape      = Yes

Start Frequency   = 00.500000 MHz
End   Frequency   = 15.000000 MHz
Overall Rate      = 50 kHz/sec

Sounding          = S3/512

                             S3/512 - SOUNDING ( for IONOGRAM )

Basic Rate  = 100 kHz/sec
L.O. Offset = 0 Hz
Height Range= 0 to 750 Km
Tcell       = 1/2 sec
Mseq        = 1/2 sec
FFT Rate    = 1024 Hz
Film Speed  = 6 mm/min
Rx AGC      = Clk/Snd
Time Marks  = Off

No of Cells = 3

             Offset        Film Drive      Rx1 Antenna      Rx2 Antenna

Cell 1    00.000000 MHz        Y                3                4
Cell 2    00.000000 MHz        Y                1                4
Cell 3    00.005000 MHz        Y                1                2  OK
```

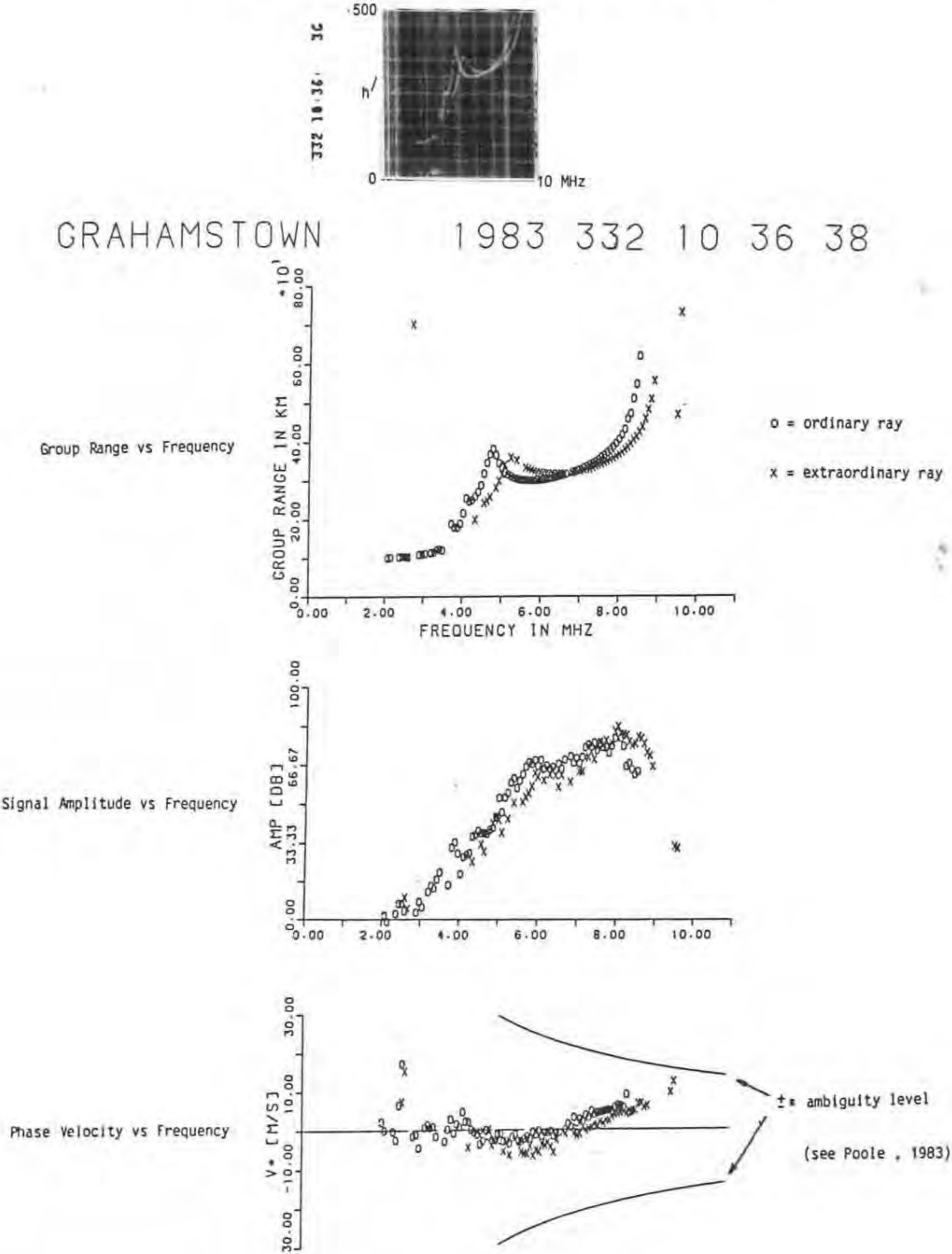
The film record of figure (6.10) is followed by a typical presentation of the digital data produced using the mainframe analysis programs written by Allon Poole.

Figure (6.10) Vertical-Incidence Ionogram - Film and Digital Records

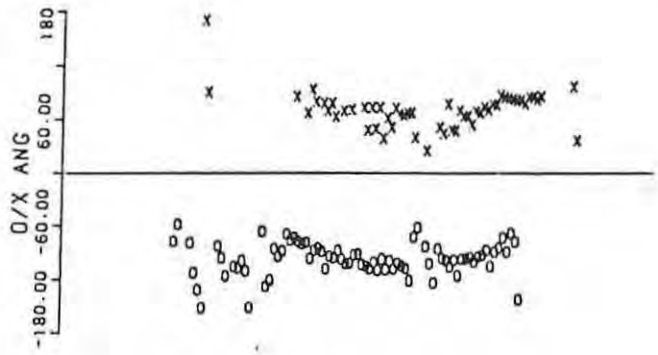


The film record of figure (6.10) is followed by a typical presentation of the digital data produced using the mainframe analysis programs written by Allon Poole.

Figure (6.10) Vertical-Incidence Ionogram - Film and Digital Records

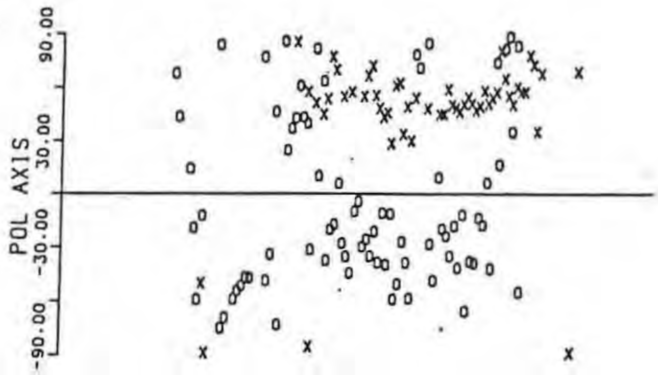


Polarisation angle vs Frequency



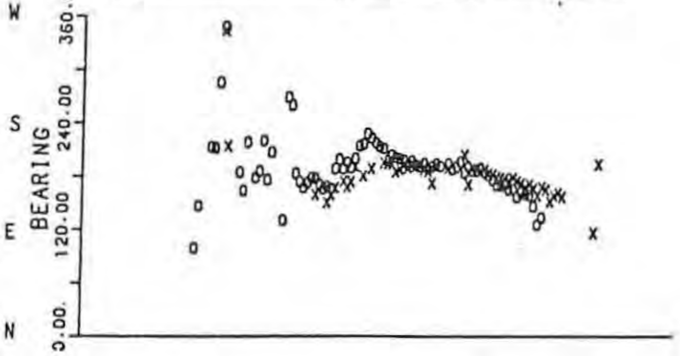
Inclination of the major semi-axis

of the pclarisation ellipse vs frequency

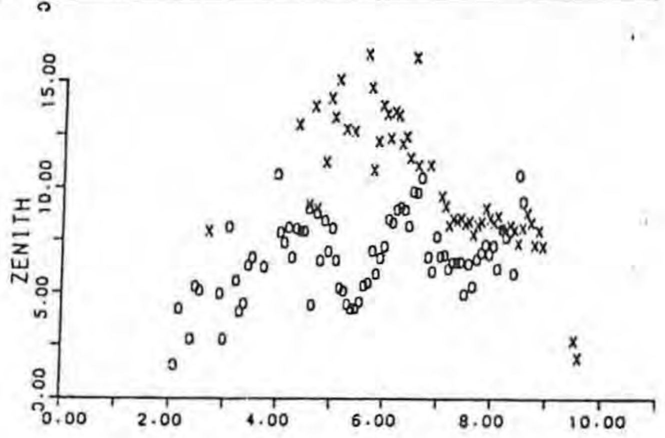


Angle of arrival data - presentation method 1

Bearing vs frequency



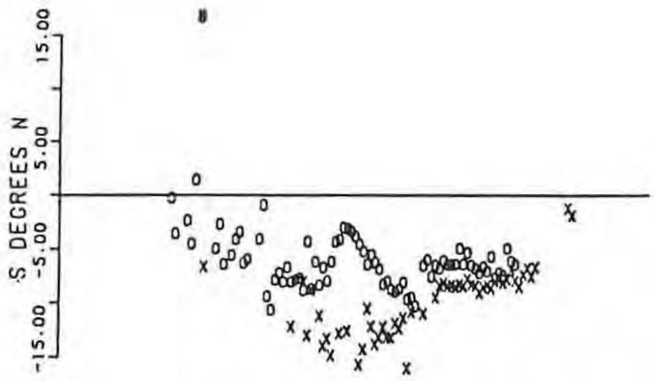
Zenith Angle vs Frequency



The bearing , zenith angle and group range vector give the orientation of the returning echo

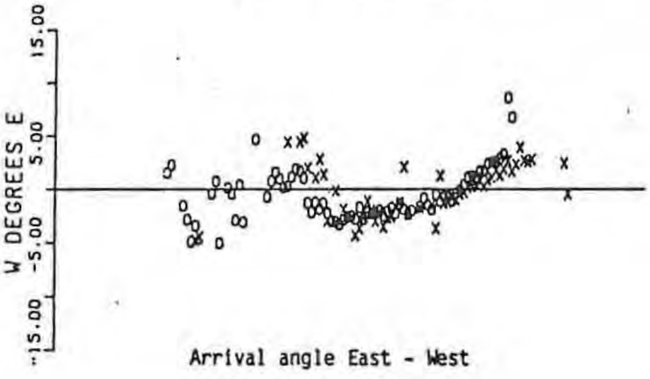
Angle of arrival data - presentation method 2

Arrival angle in the North-South plane



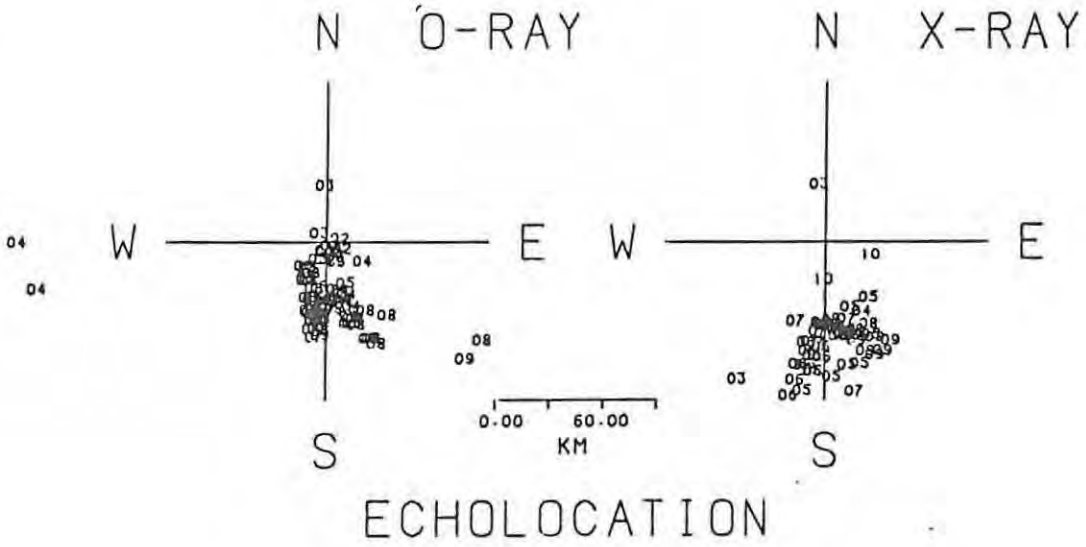
Arrival angle North - South

Arrival angle in the East-West plane



Arrival angle East - West

Angle of arrival data - presentation method 3



Echolocation - the point of the group range vector is projected onto a horizontal plane

6.8 Project ISAAC - Controller Programming Example

The modified Controller-operated Chirpsounder (without the digital data recording system) was installed on the S.A. Agulhas in June 1983 to investigate the ionosphere in the region between South Africa and Brazil in the so called "South Atlantic Anomaly" (ISAAC = International South Atlantic Anomaly Campaign).

The Controller operated very sucessfully throughout the voyage of 23 days and recorded 2128 quarter-hourly ionograms. A practical example of Controller programming capabilities as used on the ship is given below.

Programme P1 ran during the day and and swapped to P2 at 20h55. P2 swapped back to P1 at 06h55 the following morning.

```
LOG P1
                                P1 - PROGRAMME

Priority Block #1       Day          Time                Function
Weekly Data
            Entry  1   --------- -- : -- : --    ---------

Priority Block #2       Day          Time                Function
Daily  Data
            Entry  1   --------- 09 : 55 : 00    I1 - IONOGRAM ( VERTICAL )
            Entry  2   --------- 10 : 00 : 00    NO-IONO - FORTH WORD
            Entry  3   --------- 10 : 15 : 00    NO-IONO - FORTH WORD
            Entry  4   --------- 10 : 20 : 00    I1 - IONOGRAM ( VERTICAL )
            Entry  5   --------- 14 : 40 : 00    I1 - IONOGRAM ( VERTICAL )
            Entry  6   --------- 14 : 45 : 00    NO-IONO - FORTH WORD
            Entry  7   --------- 15 : 00 : 00    NO-IONO - FORTH WORD
            Entry  8   --------- 15 : 05 : 00    I1 - IONOGRAM ( VERTICAL )
            Entry  9   --------- 20 : 55 : 00    P2 - PROGRAMME
            Entry 10   --------- -- : -- : --    ---------
            Entry 11   --------- -- : -- : --    ---------

Priority Block #3       Day          Time                Function
Hourly Data
            Entry  1   --------- -- : 00 : 00    I1 - IONOGRAM ( VERTICAL )
            Entry  2   --------- -- : 15 : 00    I1 - IONOGRAM ( VERTICAL )
            Entry  3   --------- -- : 30 : 00    I1 - IONOGRAM ( VERTICAL )
            Entry  4   --------- -- : 45 : 00    I1 - IONOGRAM ( VERTICAL )
            Entry  5   --------- -- : -- : --    ---------
OK
```

```
LOG NO-IONO
                           NO-IONO - FORTH WORD

                           FORTH Word = MAP  OK
```

```
FORTH Colon Definition of MAP

: MAP ;

No operation.
```

The priority block 2 entries in programme P1 produce two twenty minute off periods from 10h00 to 10h20 and 14h45 to 15h05 for weather map reception. Ionograms that should have been run at 10h00 and 10h15 are moved to 09h55 and 10h20 respectively. The 14h45 and 15h00 ionograms are similarly moved to 14h40 and 15h05. When not overidden by higher priority block entries I1 is executed every 15 minutes at 00 , 15 , 30 and 45. Note that I1 sweeps to 15 MHz for daytime ionogram recording.

Ionogram data I1 is shown with an example from film in figure (6.11)

```
LOG I1
                          I1 - IONOGRAM ( VERTICAL )

Ionogram Duration =  0 :  4 : 50
Data to Tape      = No

Start Frequency   = 00.500000 MHz
End   Frequency   = 15.000000 MHz
Overall Rate      = 50 kHz/sec

Sounding          = S1

                         S1 - SOUNDING ( for IONOGRAM )

Basic Rate  = 50 kHz/sec
L.O. Offset = 0 Hz
Height Range= 0 to 1500 Km
Tcell       = 1/ sec
Mseq        = 1/ sec
FFT Rate    = 8 Hz
Film Speed  = 6 mm/min
Rx AGC      = Clk/Snd
Time Marks  = Off

No of Cells = 1

           Offset        Film Drive      Rx1 Antenna      Rx2 Antenna

Cell 1   00.000000 MHz       Y               1                2  OK
```

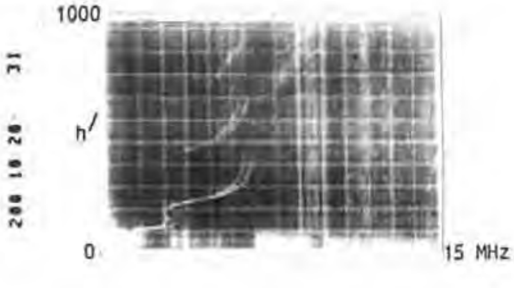


Figure (6.11) Vertical Ionogram I1 - Data and Film Record

Programme P2 executes two ionograms per available 5-minute slot that sweep to 7.5 MHz only as the penetration frequency at night is lower than that during the day. Ionogram I7.5@50 has a basic rate of 50 kHz/s and I7.5@100 a basic rate of 100 kHz/s to give an expanded virtual height scale. Ionogram data and film records are shown in figures (6.12a) and (6.12b).

```
LOG P2
                          P2 - PROGRAMME

Priority Block #1      Day          Time            Function
Weekly Data
          Entry  1   --------- -- : -- : --   ---------

Priority Block #2      Day          Time            Function
Daily  Data
          Entry  1   --------- 06 : 55 : 00   P1 - PROGRAMME
          Entry  2   --------- -- : -- : --   ---------
          Entry  3   --------- -- : -- : --   ---------

Priority Block #3      Day          Time            Function
Hourly Data
          Entry  1   --------- -- : 00 : 00   I7.5@50 - IONOGRAM ( VERTICAL )
          Entry  2   --------- -- : 02 : 30   I7.5@100 - IONOGRAM ( VERTICAL )
          Entry  3   --------- -- : 15 : 00   I7.5@50 - IONOGRAM ( VERTICAL )
          Entry  4   --------- -- : 17 : 30   I7.5@100 - IONOGRAM ( VERTICAL )
          Entry  5   --------- -- : 30 : 00   I7.5@50 - IONOGRAM ( VERTICAL )
          Entry  6   --------- -- : 32 : 30   I7.5@100 - IONOGRAM ( VERTICAL )
          Entry  7   --------- -- : 45 : 00   I7.5@50 - IONOGRAM ( VERTICAL )
          Entry  8   --------- -- : 47 : 30   I7.5@100 - IONOGRAM ( VERTICAL )
          Entry  9   --------- -- : -- : --   ---------
 OK
```

```
LOG I7.5@50
                        I7.5@50 - IONOGRAM ( VERTICAL )

Ionogram Duration =  0 :  2 : 20
Data to Tape        = No

Start Frequency     = 00.500000 MHz
End   Frequency     = 07.500000 MHz
Overall Rate        = 50 kHz/sec

Sounding            = S1

                       S1 - SOUNDING ( for IONOGRAM )

Basic Rate  = 50 kHz/sec
L.O. Offset = 0 Hz
Height Range= 0 to 1500 Km
Tcell       = 1/ sec
Mseq        = 1/ sec
FFT Rate    = 8 Hz
Film Speed  = 6 mm/min
Rx AGC      = Clk/Snd
Time Marks  = Off

No of Cells = 1

           Offset       Film Drive      Rx1 Antenna      Rx2 Antenna

Cell 1   00.000000 MHz      Y                1                2  OK
```

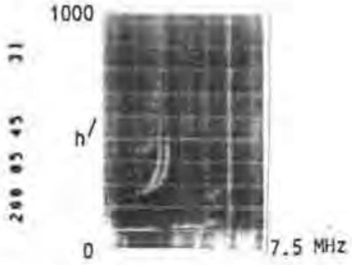


Figure (6.12a) Vertical Ionogram 17.5@50 - Data and Film Record

```
LOG 17.5@100
                          17.5@100 - IONOGRAM ( VERTICAL )

Ionogram Duration =  0 :  2 : 20
Data to Tape       = No

Start Frequency   = 00.500000 MHz
End   Frequency   = 07.500000 MHz
Overall Rate      = 50 kHz/sec

Sounding          = S2

                         S2 - SOUNDING ( for IONOGRAM )

Basic Rate  = 100 kHz/sec
L.O. Offset = 0 Hz
Height Range= 0 to 750 Km
Tcell       = 1/ sec
Mseq        = 1/ sec
FFT Rate    = 1024 Hz
Film Speed  = 6 mm/min
Rx AGC      = Clk/Snd
Time Marks  = Off

No of Cells = 1

            Offset       Film Drive      Rx1 Antenna      Rx2 Antenna

Cell 1    00.000000 MHz      Y              1               2  OK
```

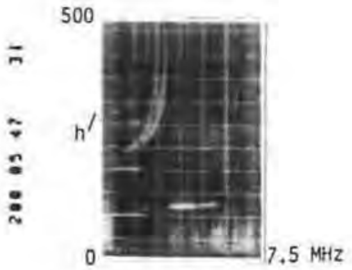


Figure (6.12b) Vertical Ionogram 17.5@100 - Data and Film Record

6.9 Digital Chirpsounder Experiments

The digital Chirpsounder was used for a number of experiments in 1983 :-

1. Digital vertical-incidence recordings were done during sunrise to collected data for a student's honours project.

2. Execution of vertical-incidence ionograms of 80-second duration from 3 to 11 MHz provided useful data for the investigation of travelling ionospheric disturbances.

3. Stationary ionogram data revealed that the Doppler velocity (V^*) of the reflection point exhibits cyclic behaviour during periods of Pc3 micropulsation activity. A paper titled "Capabilities of an FMCW (chirp) Ionosonde applied to the observation of Low Latitude PC3 pulsations" by A.W.V. Poole , P.R. Sutcliffe and J.A. Gledhill was presented at the Florence Assembly of the International Union of Radio Science in August and is presently being prepared for publication.

At the beginning of 1984 the digital Chirpsounder system was installed by the author at the South African Antarctic base , SANAE. A low frequency and a high frequency set of receive antennae were erected , the change-over frequency being 3 MHz. The "normal" recording schedule (section 2.4) with ionogram storage on film has been executed during the year by the Controller. The system has been used in "digital" mode for stationary-ionogram recording during micropulsation activity.

6.10 Commercial Digital Chirpsounders

The National Institute for Telecommunications Research (NITR) of the Council for Scientific and Industrial Research (CSIR) has purchased a digital ionosonde from the manufacturers , BR Communications (formerly Barry Research Corp.). In drawing up the system specifications the NITR consulted the author on aspects of Controller operation. Subsequent to this BR Communiucations requested details of the programmable M-sequence generator circuitry. This circuit is included in the digital Chirpsounder they now manufacture.

6.11 Conclusion

A suggested improvement is that additional system data be stored on film. The ability to write a number of lines of alphanumeric data to film has been catered for as outlined in Appendix E page E69.

The Controller hardware and software form an integral part of the digital Chirpsounder system which has proved to be extremely versatile. The full capabilities of the system have yet to be realized.

REFERENCES

Barry Research. (1971). Chirpsounder Systems. Palo Alto , California.

Barry Research. (1972). Vertichirp Sounder VOS -1C , Operating and Service Manuals. Palo Alto , California.

Bibl, K., Patenaude, J.A., Reinsch, B.W. and Vargas-Vila, R. (1970). Report AFCRL-71-0001 , USAF Cambridge Research Labs.

Budden, K.G. (1964). Lectures on Magnetionic Theory. Blackie and Son Ltd., London and Glasgow.

Davies, K. (1965). Ionospheric Radio Propagation. NBS Monograph 80. U.S. Dept. of Commerce , Washington D.C.

Eaker, C. (1980). "Just in Case" , FORTH Dimensions , II/3 , Sept/Oct.

Fenwick, R.B. (1973). Oblique Chirpsounders - the HF Communications Test Set Tech. Note 2 , Rev A. Barry Research Corp., Palo Alto, California.

Fisher, J.S. (1979). "A Real Time Fast Fourier Transform Analyser". M.Sc. Thesis , Rhodes University, Grahamstown.

Halliday, P. and Resnick, R. (1970). Fundamentals of Physics. John Wiley and Sons Inc., New York.

Kelso, J.M. (1964). Radio Ray Propagation in the Ionosphere. McGraw-Hill Book Co., New York.

Mac Williams, F.J. and Sloane, N.J.A. (1976). "Pseudo- Random Sequences and Arrays". Proc. IEEE. 64 , 1715 - 1728.

Mitra, S.K. (1952). The Upper Atmosphere. 2nd Edition , The Asiatic Society Monograph Series, Calcutta.

Philips, M.L. (1974). "Ground Based Vertical Incidence Ionograms". IEEE. Transactions on Antennas and Propagation , AP-22 , 785 - 795.

Poole, A.W.V. (1983). "Advanced Ionospheric Chirpsounding". PhD. Thesis , Rhodes University, Grahamstown.

Ratcliffe, J.A. (1959). The magneto-ionic theory and its application to the ionosphere. Cambridge University Press , Cambridge.

Ratcliffe, J.A. (1972). An introduction to the ionosphere and magnetosphere. Cambridge University Press , Cambridge.

Rather, E.D. and Moore , C.H. (1976). "The FORTH Approach to Operating Systems". Proc. Annual Conference Association for Computing Machinery Inc.

Schwartz M. (1970). Information Transmission, Modulation and Noise. McGraw-Hill Kogakusha Ltd., Tokyo.

Wright, J.W. (1970). "A Comparative Discussion of Four Modern Approaches to Ionosonde Design". U.S. Dept of Commerce , Environmental Research Lab., Boulder , Colorado. Correspondence to A.H. Shapley.

APPENDIX A

VERTICHIRP CONTROLLER HARDWARE - MANUAL 1

DESCRIPTION OF PRINTED CIRCUIT BOARD OPERATION

G. P. EVANS

ANTARCTIC RESEARCH - RHODES UNIVERSITY

DECEMBER 1983

Vertichirp Controller Hardware - Manual 1

Description of Printed Circuit Board Operation

Vertichirp Sounder System Block Diagram

The simplified system block diagram shows the major components of the computer controlled ionosonde. The following block diagram shows all the Vertichirp Controller input and output signal paths as well as the RF signal routing. The Controller , together with its interconnection cables , directly replaces the 1015 Logchirp Control.

The VDU can be switched to either the Controller computer or to the Data Capture system computer. The Vertichirp Controller Software Manual describes all programming options.

Two independant receiver/analysis signal paths exist . The first signal path consists of the 4040 Receiver , 2002 Spectrum Analyser and FRS-1 Film Recording System. Because the T500A Linear Amplifier does not allow T/R switching, the RF transmitted on the vertical Tx antenna is received by a separate vertical Rx antenna switched through to the 4040 Receiver by B2a and B2b. The receiver baseband output is analysed by the 2002 Spectrum Analyser before being recorded on film.

The second signal path consists of two phase-matched receivers , an FFT Analysis System and a Data Capture System which stores the data on magnetic tape. The antenna switch B1 allows the selection of one of eight receive antennae to either of the two receivers. The baseband outputs of both receivers are analysed by the FFT circuitry and DCS computer and the partially processed data are stored on magnetic tape for further processing in a mainframe computer.

Note that the output of one of the two matched receivers can be stored on film by using it as the input to the 2002 Spectrum Analyser instead of the 4040 Receiver baseband signal. The 4040 Receiver is however more sensitive than either of the matched receivers and is therefore preferred for the film record.

To run the 4040 Receiver and the matched receivers together the 40 MHz and 40.6 - 70.1 MHz Local Oscillator (LO) injection frequencies are amplified before driving in-phase power splitters. The 1 MHz LO signal , being a TTL signal, drives both the dual receivers and the 4040 receiver directly. The Gain-Weight signal from the 4050 Amplifier T/R also drives both receiver sets.

Vertichirp Controller Block Diagram 1

This block diagram shows the microcomputer system together with the controller Strobe Decode Board , PC6.

A three pole change-over switch connects the MP-S Serial Interface to either a VDU at 9600b or to a printer at 300b.

Parallel Interface MP-LA-1 handles the Tcell-pulse interrupt on its CA1 input line and the Real-Time Clock one second interrupt on its CB1 line.

MP-LA-1 A side data outputs drive the Controller hardware data bus. The B side data together with the CB1 output,configured as a write strobe with E restore ($\overline{\text{WRE}}$), are used to generate all the strobe signals on the Strobe Decode Board , PC6.

The A side of Parallel Interface MP-LA-2 is used to input the Real - Time Clock Data with the data valid flag on the CA1 input. The B side of MP-LA-2 is used to output ionogram control parameters to the Data Capture System. The CB1 line is used as an acknowledge flag from the Data Capture System.

MP-LA-3 is a spare Parallel Interface.

Vertichirp Controller Block Diagram 2

The 100 kHz signal from the 105B Quartz Oscillator can have pulses deleted or inserted by Advance/Retard Board,PC5 before going to the Clock Dividers on PC3a and the Basic Rate circuitry on PC4a.

The FFT Sample Rate Generator , PC7 , uses 1 MHz as an input signal and generates 10.24 MHz for the LO Offset Frequency Synthesizer , PC9 , 2048 Hz for the M-Sequence Generator , PC4c , the FFT Sample Rate signal , 4 and 64 Hz for Tcell pulse generation and a sync on second pulse.

40 MHz from the 4041 Local Oscillator goes to brass box B3 where it can be offset by the frequency generated by the Sine Wave Generator , PC10.

The Sweep Start and Control Board PC3b contains two control registers , Control Register A and Control Register B , which define system operation.

The Controller hardware data bus is called 'D' in the block diagram. Data on this bus is latched on the different boards by strobe signals generated by the Strobe Decode board , PC6. All the strobe signals are active low , the nth strobe being abbreviated as STn.

Vertichirp Controller Power Supplies

The SWTPC microprocessor system power supply , MP-P , powers both the computer and the control hardware with unregulated 7-8 VDC via connector J1. All the SWTPC boards have on-board +5V regulators. The controller boards PC1 to PC12 have chassis mounted +5V regulators.

The +24V battery supply from the 105B Quartz Oscillator is taken via a Filter PCB on the rear panel to a DC to DC converter which supplies +5VDC to the circuits which must remain active when main power failures. Six diodes connect this supply to the two S-32 Memory Boards and also to the Clock Dividers on PC3a.

Logchirp Board A40 in slot PC12 converts +24V to -10V which is used in the 6018 Camera Interface to power its clock circuitry.

-12V from the MP-B2 mother board is regulated to -5V to supply the Sine Wave Generator on PC10.

Location of Vertichirp Controller Circuit Boards

The SWTPC MP-B2 Motherboard modified for use with the MP-09 Processor Board is mounted on the right hand side of the Controller chassis. The I/O half of this board is horizontal as usual but the CPU and Memory slots are vertically mounted. This means that the CPU and Memory boards plug into these slots from the front of the Controller with the component side down.

Behind the I/O ports on the rear panel is a cooling fan , the DC to DC converter , the Diode PCB and the Filter PCB. In front of the CPU and Memory boards and secured from the top is the Frequency Display Board , FD.

The front left of the chassis carries the MP-P power supply. The power switch , Cold and Warm Reset buttons and Real Time Clock Display , RTCD are also on the left.

Controller hardware PC board +5V regulators with corresponding edge connectors occupy the remaining space. The Controller PC board slots are numbered from the rear beginning with PC1 and ending with PC12.

Vertichirp Controller Printed Circuit Boards

The PC Boards are discussed in ascending order from PC1 to PC12. Additional circuitry associated with a particular PC board is discussed immediately after it. For example , the description of the Antenna Switch in brass box B1 immediately follows that of the Antenna Select and Film Speed Control printed circuit board , PC8.

Each circuit description in this manual (Vertichirp Controller Hardware 1) consists of :

1. Introduction
2. Circuit Development (if applicable)
3. Circuit Description
4. Control / Output Selection Data

The second manual , Vertichirp Controller Hardware 2 , contains :-

1. Circuit Diagram
2. Component Location Diagram
3. Wire Lists

Synthesizer Control , PC1

Introduction

The Barry Research 5006 Frequency Synthesizer provides 3 swept frequency outputs (40.125 - 47.5 MHz , 0.5 - 30.0 MHz and 0.125 - 7.5 MHz) and 4 fixed frequency outputs (100 kHz , 1 MHz , 5 MHz and 40 MHz).

The 0.5 to 30.0 MHz transmit signal f_t is derived from the 40.125 to 47.5 MHz signal f_o as is the first local oscillator injection frequency. Seven decades of BCD programming data to the synthesizer allow the synthesis of f_o in 1Hz increments. The 0.5 to 30.0 MHz output f_t is derived directly from f_o as follows :-

$$f_t = 4f_o - 160 \text{ MHz}$$

The 0.5 to 30.0 MHz frequency range is therefore covered in 4 Hz increments. The output frequency is equal to the synthesizer programming BCD number multiplied by 4.

A frequency ramp or ' chirp ' signal can be generated under digital control by incrementing the synthesizer programming BCD number at one quarter of the required chirp rate. To generate a frequency ramp with a rate of change of frequency of 50 kHz/s the BCD programming data must be incremented at 12.5 kHz.

The Synthesizer Control Board , PC1 , outputs BCD to the synthesizer.

Synthesizer Control , PC1

Circuit Description

Latches U1 and U5 are loaded from the Controller data bus by strobe ST5, U9 and U13 by ST6 , U4 and U8 by ST7 and U12 by ST8. Synchronous BCD counters U2,U6,U10,U14,U3,U7,and U11 are all parallel loaded from the latches by the synthesizer counter load pulse generated on the Sweep Start and Control board PC3b. The outputs of these counters drive the synthesizer directly via a rear panel 50 way Amphenol connector.

Standard TTL synchronous counters are used on this board as Low Power Schottky TTL does not drive the synthesizer reliably.

BCDx4 , Frequency Mark Generator and Tx Filter Control , PC2

BCDx4 Circuitry

Introduction

The 5006 Frequency Synthesizer is directly programmable over the range 40.125 to 47.5 MHz. The 0.5 to 30.0 MHz range is obtained by multiplying this signal by 4 and subtracting 160.0 MHz. Thus the transmit sweep f_t is :-

$$f_t = 4\ f_o - 160\ \text{MHz}$$

where f_o is the programmed frequency sweep.

The relationship between the transmit sweep , f_t , and the BCD programming data sent to the synthesizer can be simplified to :-

$$f_t = (\text{ BCD programming data }) \times 4\ \text{MHz}$$

for programming data from 00125000 to 07500000.

The instantaneous transmitter frequency is displayed on the front panel and is driven by the BCD programming data multiplied by 4.

Circuit Develompent

The maximum BCD number from any one of the control decades is 9. Multiplication by 4 gives 9 x 4 = 36 , that is six units with a carry out of 3. Including a possible carry in of three from a previous stage gives 36 + 3 = 39. The maximum carry out is therefore decimal 3 which requires two bits for binary representation.

The table below shows the results of multiplying BCD numbers 0 through 9 by 4 for the 4 possible values of carry in ; 0 , 1 , 2 and 3.

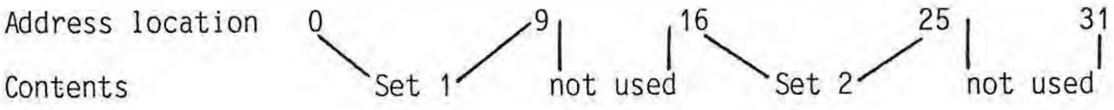
		BCD x 4							
Decimal	BCD	Cin = 00		Cin = 01		Cin = 10		Cin = 11	
	2^3 2^2 2^1 2^0	Cout	BCDx4	Cout	BCDx4	Cout	BCDx4	Cout	BCDx4
0	0 0 0 0	0 0	0 0 0 0	0 0	0 0 0 1	0 0	0 0 1 0	0 0	0 0 1 1
1	0 0 0 1	0 0	0 1 0 0	0 0	0 1 0 1	0 0	0 1 1 0	0 0	0 1 1 1
2	0 0 1 0	0 0	1 0 0 0	0 0	1 0 0 1	0 1	0 0 0 0	0 1	0 0 0 1
3	0 0 1 1	0 1	0 0 1 0	0 1	0 0 1 1	0 1	0 1 0 0	0 1	0 1 0 1
4	0 1 0 0	0 1	0 1 1 0	0 1	0 1 1 1	0 1	1 0 0 0	0 1	1 0 0 1
5	0 1 0 1	1 0	0 0 0 0	1 0	0 0 0 1	1 0	0 0 1 0	1 0	0 0 1 1
6	0 1 1 0	1 0	0 1 0 0	1 0	0 1 0 1	1 0	0 1 1 0	1 0	0 1 1 1
7	0 1 1 1	1 0	1 0 0 0	1 0	1 0 0 1	1 1	0 0 0 0	1 1	0 0 0 1
8	1 0 0 0	1 1	0 0 1 0	1 1	0 0 1 1	1 1	0 1 0 0	1 1	0 1 1 1
9	1 0 0 1	1 1	0 1 1 0	1 1	0 1 1 1	1 1	1 0 0 0	1 1	1 0 0 1

From the table it is clear that the carry in LSB determines the LSB of (BCD x 4 + carry in). The carry in MSB defines two sets of BCD x 4 data.

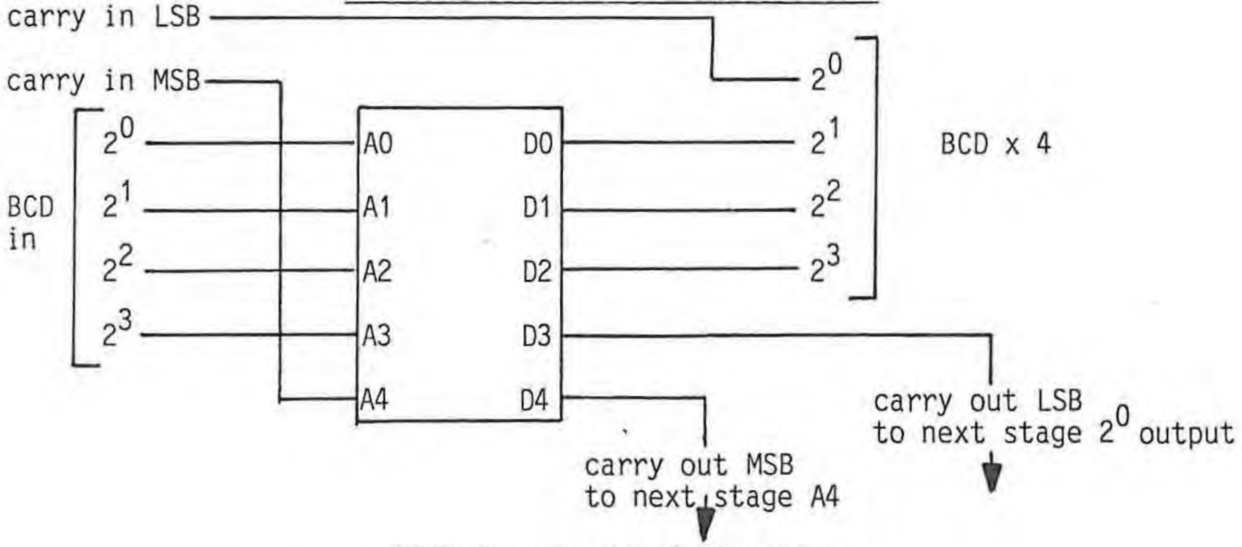
Set one = BCD x 4

Set two = (BCD x 4) + 2

The carry in MSB is used to select the correct data set stored in a 32 x 8 PROM via address input line A4. Address input lines A0 , A1 , A2 and A3 are driven by the BCD number to be multiplied by 4.



ROM Address and Data Organisation



PROM Programming Information

Set 1		
Decimal Addr.	Hex Addr.	Hex Data
0	0 0	0 0
1	0 1	0 2
2	0 2	0 4
3	0 3	0 9
4	0 4	0 B
5	0 5	1 0
6	0 6	1 2
7	0 7	1 4
8	0 8	1 9
9	0 9	1 B
10	0 A	-
11	0 B	-
12	0 C	-
13	0 D	-
14	0 E	-
15	0 F	-

Set 2		
Decimal Addr.	Hex Addr.	Hex Data
16	1 0	0 1
17	1 1	0 3
18	1 2	0 8
19	1 3	0 A
20	1 4	0 C
21	1 5	1 1
22	1 6	1 3
23	1 7	1 8
24	1 8	1 A
25	1 9	1 C
26	1 A	-
27	1 B	-
28	1 C	-
29	1 D	-
30	1 E	-
31	1 F	-

Circuit Description

74S288 PROM's U10 , U11 , U12 , and U13 each contain the two data sets and are addressed by BCD decades 10^3 to 10^6 respectively. The 10^3 BCD input to U10 is included to generate the two carry bits for the 10^4 decade which is the least significant decade of BCD x 4 that is displayed.

BCD x 4 decades 10^7 , 10^6 , 10^5 and 10^4 go to the Frequency Display board which displays the transmit frequency.

Frequency Mark Generator

Introduction

Frequency marks are generated on film at certain fixed frequencies. These frequencies are :-

0 1 2 3 4 5 6 7 8 9 10

11 12 13 14 15 16 17 18 19 20

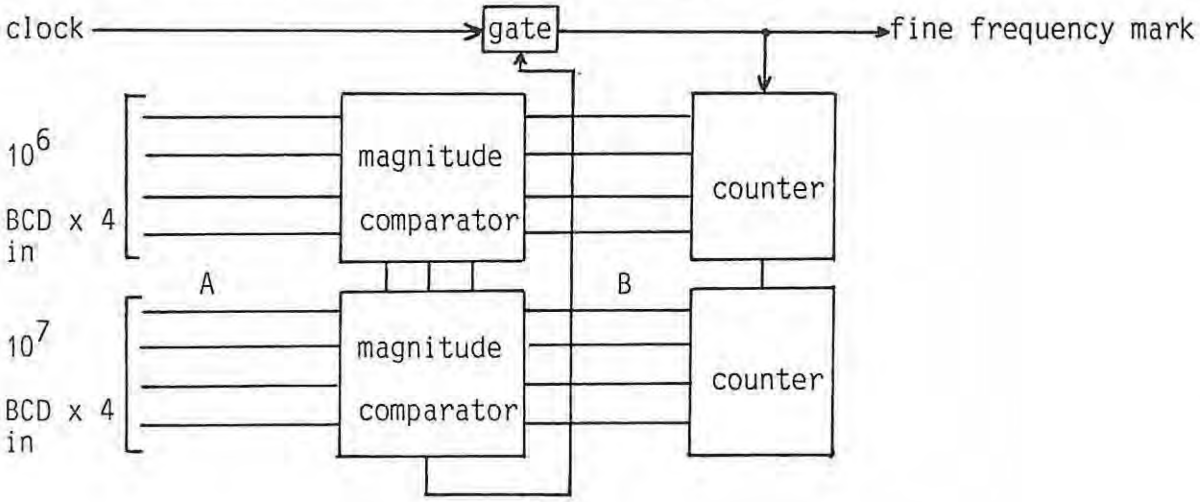
21 22 23 25 25 26 27 28 29 30 MHz

All frequency marks are fine except for those at 1 , 10 , 20 and 30 MHz which are bold.

The first transition of a frequency at which a mark is required must trigger the mark. Subsequent transitions of the frequency must be ignored. This is because a particular frequency could be passed more than once at the low end of a logarithmic sweep or when using a jumpback linear rate.

Circuit Development

In the circuit below two 4 bit magnitude comparators are used to compare the 10^6 and 10^7 BCD x 4 decades (A) with the number (B) on the output of two BCD counters. The clock to the counters is disabled when B is greater than A and enabled when B is less than or equal to A.



Let the number A equal x and B equal x+1. Since B is greater than A the clock to the counters is disabled. When A changes to x+1 the gate is enabled and the next clock pulse increments B to x+2. This disables the gate since B is greater than A. The gated clock pulse signals the x+1 MHz fine frequency mark. Only one mark is generated at a particular frequency no matter how many times the A number transitions that frequency.

As an example let A change from 5 to 6. B will be incremented by the gated clock from 6 to 7 generating the 6 MHz frequency mark.

Bold frequency marks are required at 0 , 10 , 20 and 30 MHz. Because the counters are one count ahead of the actual frequency the corresponding outputs are 1 , 11 , 21 and 31. Bold frequency marks are obtained by ignoring the MSB counter and fully decoding the LSB counter to produce a pulse when its binary output is 0001 (MSB--LSB).

Circuit Description

4 bit magnitude comparators U8 and U9 compare data from the BCDx4 PROM's U12 and U13 , with the output of counters U4 and U5. Data to the comparator A side is from the BCDx4 PROM's and data to the B side is from the counters.

At the start of an ionogram counters U4 and U5 are cleared by a pulse on the Clear Freq Mark input. Clock pulses from the Basic Rate Clock increment the counters (B) until B is greater than A and the A less than B output of U9 goes high. As an ionogram proceeds and A is incremented the A less than B output of U9 goes low. This signal , inverted by U7B , enables gate U6D. Basic Rate Clock pulses inverted by U7C are allowed to trigger the monostable consisting of U6A , C5 , R2 and U6C. The 5 microsecond output pulse clocks counters U4 and U5 as well as triggering a positive going fine frequency mark via C7 , R4 , U6B and U7A.

With SW1 open frequency marks are generated from 1 to 30 MHz. Closing SW1 inhibits fine frequency marks above 15 MHz. U3 , an AND-OR-INVERT gate , disables U6B when the counter output (which is 1 ahead of the frequency) is 16 or greater.

The outputs of U4 , the LSB , are called Q_0 , Q_1 , Q_2 and Q_3.

The outputs of U5 , the MSB , are called Q'_0 , Q'_1 , Q'_2 and Q'_3.

Decimal No.	Q'_1	Q'_0	Q_3	Q_2	Q_1	Q_0	
16	0	1	0	1	1	0	$Q'_0 \cdot Q_2 \cdot Q_1$
17	0	1	0	1	1	1	
18	0	1	1	0	0	0	$Q'_0 \cdot Q_3$
19	0	1	1	0	0	1	
20	1	0	0	0	0	0	Q'_1
↓	↓	↓	↓	↓	↓	↓	
30	1	1	0	0	0	0	

Frequency marks are inhibited when :-

$Q'_0 \cdot Q_2 \cdot Q_1 + Q'_0 \cdot Q_3 + Q'_1$ is true.

The output of U3 (74LS54) goes low to inhibit the frequency marks.

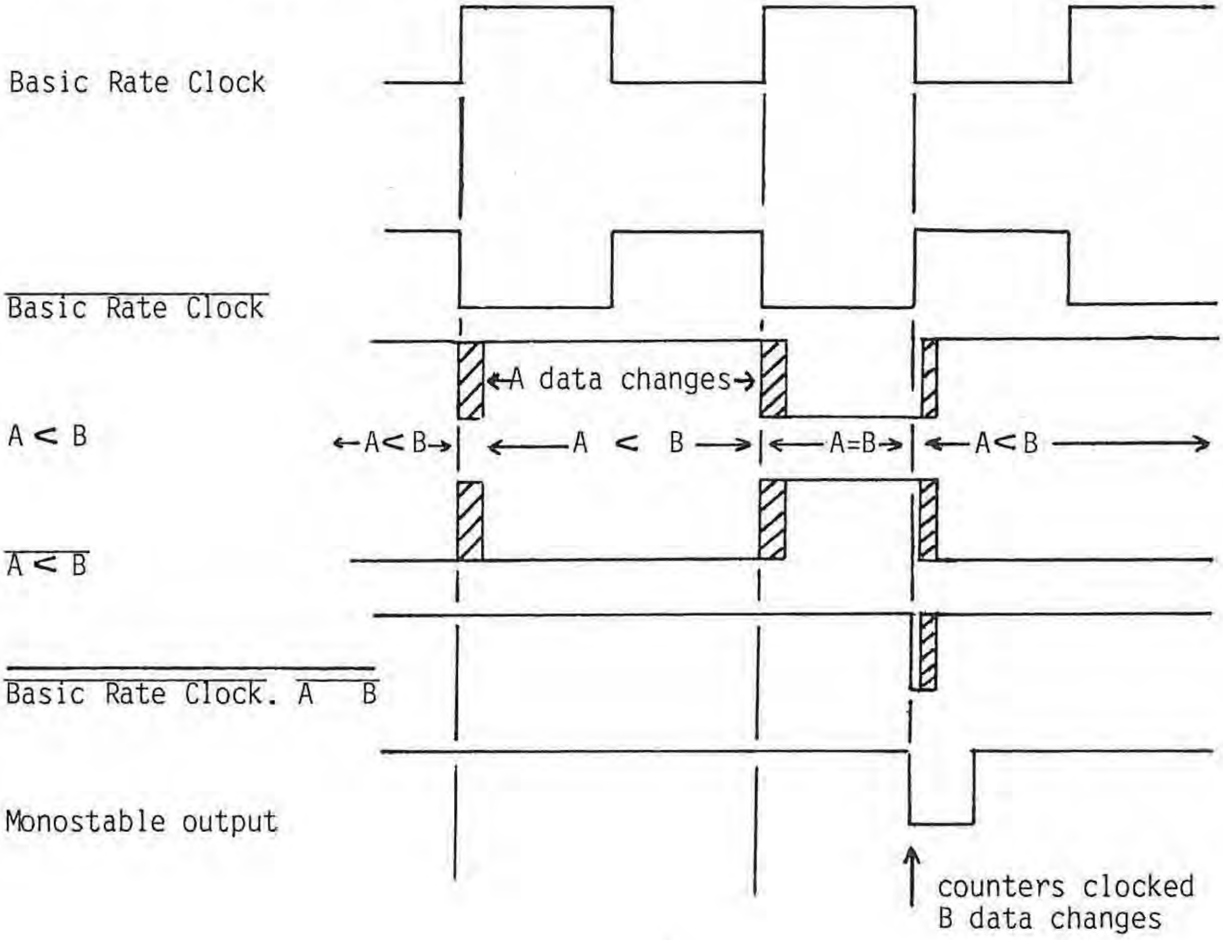
Bold frequency marks are required at 0 , 10 , 20 and 30 MHz. The corresponding counter outputs are 1 , 11 , 21 and 31. Counter U4 output equal to 0 0 0 1 (MSB--LSB) is detected by U7D , U7E , U7F , U1A , U1B and U1C and differentiated by C3 and R1 to produce a short positive going trigger pulse on the output of U2C.

Pin 9 of U2C can be strobed low by the computer to produce a Bold mark on film to indicate time on Stationary Doppler recordings.

Timing Diagram

PROM propagation delays to the A inputs of the magnitude comparators together with ripple counting delays on the B inputs could lead to false frequency mark triggering and Tx Filter switching at incorrect frequencies. The timing diagram below shows how the use of the Basic Rate Clock together with a monostable eliminates these potential problems.

The BCD data to the synthesizer changes synchronously on the rising edge of the basic rate clock.



Tx Filter Control

Introduction

The T500A Linear Power Amplifier has five filters on the RF output. These filters numbered F1 , F2 , F3 , F4 and F5 cover the ranges 2 - 3 , 3 - 5 , 5 - 8 , 8 - 15 , and 15 - 30 MHz.

Circuit Description

Counters U4 and U5 address 2716 EPROM U14 thus ensuring that only the first transition of a filter change frequency is acted upon. The EPROM has eight output data bits. Five of these are used , one per filter. At any one time only one of the five filters is selected by a low on its control line.

As BCD data from counters U4 and U5 address the EPROM hex locations A to F , 1A to 1F and 2A to 2F are never addressed. These locations are however programmed with the current filter select data. The filter select data is stored as a FLEX file and is called FILTER.

The filter select data goes to the Antenna Select/Tx Filter driver brass box , B2 , via rear panel 50 way amphenol connector RPA1.

```
ANTARCTIC RESEARCH ( RHODES )          2-8-83 TSC ASSEMBLER    PAGE    1
TX FILTER DRIVER DATA


                         ***    TRANSMITTER FILTER DRIVER DATA     ***
                         *
                         *  THE FREQUENCY MARK COUNTERS ARE USED TO ADDRESS AN EPROM .
                         *  THESE COUNTERS ONLY COUNT UP AND ARE 1 MHz AHEAD OF
                         *  THE ACTUAL FREQUENCY BEING TRANSMITTED .
                         *  THIS INSURES THAT THE FIRST TRANSITION OF A FREQUENCY
                         *  SWITCHES TO THE NEXT FILTER .
                         *  AS BCD IS USED ON THE ADDRESS LINES HEX LOCATIONS
                         *  A TO F , 1A TO 1F AND 2A TO 2F ARE NEVER ADDRESSED .
                         *  THEY ARE HOWEVER PROGRAMMED WITH THE CURRENT FILTER DATA .
                         *  5 OF THE 8 OUTPUT DATA LINES ARE USED .
                         *  A FILTER IS SELECTED WHEN IT'S CONTROL LINE GOES LOW .
                         *  THE 5 FILTERS COVER THE FREQUENCY RANGE AS FOLLOWS
                         *     2 - 3 , 3 - 5 , 5 - 8 , 9 - 15 , 15 - 30 MHz .
                         *
                         *     FILTER DATA
                         *
 0000                          ORG    $0000
 0000 FE FE FE FE        BAND1 FCB    $FE,$FE,$FE,$FE
 0004 FD FD              BAND2 FCB    $FD,$FD
 0006 FB FB FB           BAND3 FCB    $FB,$FB,$FB
 0009 F7 F7 F7 F7        BAND4 FCB    $F7,$F7,$F7,$F7
 000D F7 F7 F7 F7              FCB    $F7,$F7,$F7,$F7
 0011 F7 F7 F7 F7              FCB    $F7,$F7,$F7,$F7
 0015 F7                       FCB    $F7
 0016 EF EF EF EF        BAND5 FCB    $EF,$EF,$EF,$EF
 001A EF EF EF EF              FCB    $EF,$EF,$EF,$EF
 001E EF EF EF EF              FCB    $EF,$EF,$EF,$EF
 0022 EF EF EF EF              FCB    $EF,$EF,$EF,$EF
 0026 EF EF EF EF              FCB    $EF,$EF,$EF,$EF
 002A EF EF EF EF              FCB    $EF,$EF,$EF,$EF
 002E EF EF EF EF              FCB    $EF,$EF,$EF,$EF
                               END

0 ERROR(S) DETECTED


ANTARCTIC RESEARCH ( RHODES )          2-8-83 TSC ASSEMBLER    PAGE    2
TX FILTER DRIVER DATA


SYMBOL TABLE:

BAND1  0000   BAND2  0004   BAND3  0006   BAND4  0009   BAND5  0016


+++
```

Frequency Display ,FD

Introduction

The Frequency Display board displays the frequency being transmitted as XX.XX MHz.

Circuit Description

Four decades of BCD x 4 data from PC2 go to 4 seven segment decoder drivers U1 , U2 , U3 and U4. These drive two dual displays DIS1 and DIS2 through 300 ohm current limiting resistors.

The Frequency Display board , FD , has its own 5V regulator U5 powering the drivers and the display.

Antenna Select and Tx Filter Drivers , B2a

Vertical/Oblique Tx Antenna Switching and Relay Drivers for Tx Filter Control

Introduction

The T500A linear power amplifier has 5 filters on the Rf output. The selected filter is switched in by two relays drawing 140 mA between them. Each relay driver in the relay driver brass box , B2a , controls a set of filter switching relays.

The Vertical/Oblique antenna switching relays connect either the vertical Tx antenna or the oblique Tx/Rx antenna to the T500A Amplifier output. When the vertical Tx antenna is connected to the T500A the oblique antenna is switched to the Rx Antenna Select box , B2b.

Circuit Description

Both the filter drivers and the co-axial relay drivers are identical. If the control line to filter 1 driver , F1 , is at a logic 1 then Q1 is on , Q2 is off and the filter relays are not activated. When F1 changes to a logic 0 , Q1 switches off , Q2 switches on and filter 1 relays in the T500A amplifier are activated. Protection diodes , one per set of filter relays , have been added to the T500A amplifier circuitry.

The $V/\overline{O}$ Select driver obtains +22.5VDC from a power supply mounted in the T500A Linear Amplifier. When $V/\overline{O}$ Select is high the relays are not activated and the transmitter is connected to the Vertical Tx antenna and the Oblique antenna is connected to the Rx Antenna Select box.

When $V/\overline{O}$ is low the relays are energised to switch the transmitter output from the vertical Tx antenna to the oblique Tx antenna. These relays draw 180 mA when supplied with 22.5VDC.

The 14 way Amphenol connector that plugs into the Antenna Select and Tx Filter box , B2a , has two cables coming from it . Cable 1 goes to the Controller connector RPA1 and cable 2 goes to the T500A Linear Amplifier.

Subminiature co-axial cable , cable 3 , goes from RPA1 pin 8 to the subminiature connector on the Antenna Select and Tx Filter box , B2a. This cable carries the signal Vertical Rx/$\overline{\text{Oblique Rx}}$ to the 4040 Rx Antenna Select box B2b which is mounted on top of B2a.

The 3 MHz Antennae Change-over Switching box obtains its control signal together with 5V and gnd. from B2a via connector AC1 (Ant. Change-over 1). The control signal used is F1 , the 2-3 MHz filter select , which remains low for frequencies below 3 MHz.

4040 Rx Antenna Select - B2b

Introduction

This box contains a co-axial relay plus driver circuitry and selects either a vertical Rx antenna or the oblique antenna to the 4040 Receiver. It is mounted on top of Antenna Select and Tx Filter box , B2a.

Circuit Description

The relay driver circuitry is identical to that used in the Antenna Select and Tx Filter box , B2a. Relay driver power , +5 and +22.5 VDC , is obtained from B2a.

Cable 3 carries the control signal from the Controller rear panel Amphenol connector to the Rx Antenna Select box B2b via B2a. When the control signal is high the relay is not activated and the vertical Rx antenna is connected to the 4040A Receiver. Activating the relay with a logic low on the control line connects the oblique antenna to the 4040A Receiver.

Trans World Electronics T500A Linear Amplifier Modifications

Introduction

The power amplifier has 5 relay switched filters on the RF output. Remote filter switching is possible via a rear panel connector.

Additions to Circuitry

To select a filter its control line must be grounded. The Tx Filter Box , B2a , contains 5 relay drivers which do this. The addition of 5 1N4007 diodes across the filter control lines in the T500A protect the driver transistors. The filters can be bypassed by connecting a co-axial cable jumper between two BNC connectors mounted on the T500A rear panel.

A 22.5 VDC supply using an LM317 voltage regulator supplies power to the antenna select co-axial relays in brass box B2a and also to the 4040 receiver antenna select co-axial relay in small brass box B2b.

Note:

The T500A Linear Amplifier has a Press to Talk (PTT) feature requiring that pin 10 on the remote control socket S03 be grounded to transmit. The mating plug has pin 10 grounded with the result that it must be plugged in to enable the transmitter.

3 MHz Antennae Change-over , B4

Introduction

The frequency range of a vertical ionogram is from 0.5 Mhz to 15.0 MHz. The 4 electrically short receive antennae that were used worked well above 3.0 MHz. To improve the low frequency response a second set of 4 longer receive antennae were introduced.

The 3 MHz Antenna Change-over box switches between LF and HF antennae sets.

Circuit Description

100 ohm twin core shielded cable is used from the antenna boxes to the 3.0 MHz Antenna Change-over box. The 8 baluns , BL1 through BL8 , transform the 100 ohm line impedance to 50 ohm.

Grounding relays GRL1 , GRL2 , GRL3 and GRL4 earth the HF signals when the control signal is low ; that is for frequencies up to 3 MHz. Select relays SRLY1 , SRLY2 , SRLY3 and SRLY4 then connect the LF antennae to the output.

The control signal goes high at 3 MHz to ground the LF antenna set and connect the HF antennae to the output connectors.

Clock Dividers ,PC3a

Introduction

A number of frequencies are required for system operation . The Time/Date (T/D) clock in the 6018 Camera Interface/Display requires 100 Hz and the Real Time Clock , RTC , requires 1 Hz. A battery supply to the clock dividers ensures that these signals are not affected by loss of power.

Circuit Description

The input 100 kHz square wave is divided by U6 , U2 , U1 , U5 and U10 to provide outputs of 1 kHz , 100 Hz , 10 Hz and 1 Hz which are buffered by U15. An unbuffered 1 Hz signal drives the Real Time Clock , RTC.

U3 monitors the supplies to PC3 , PC4 , PC5 , PC6 , PC7 , PC8 , PC9 and PC10 via a Scotchflex connector and illuminates front panel LED "PC3 to PC10 +5V Supply" if all supplies are present.

Sweep Start and Control , PC3b

Introduction

The Sweep Start circuitry is required to start an ionogram synchronous with the 1 Hz clock signal falling edge. This is important for oblique synchronization. Ionogram execution is controlled by data written to two 8-bit registers , Control Register A and Control Register B.

System Power on clear circuitry is also included on this board.

Circuit Description

Control Registers

Control Register A

Control Register A consists of two 4-bit latches U14 and U18 which can be loaded from the controller data bus. The data written to this register does not change during an ionogram. The function of each bit is :-

Controller Data Bus Bit		Function
Bit No.	State	-
D7	0 1	4 Hz - Stationary Doppler 64 Hz - Ionograms Tcell pulse gen rate select
D6	0 1	disable Sweep signal to FFT enable Sweep signal to FFT
D5 , D4	0 0 0 1 1 0 1 1	Tx only Tx only Rx only T/R switching T/R Control. D5 also selects oblique or vertical transmit antenna.
D3	0 1	disable Tcell interrupt to CPU enable Tcell interrupt to CPU
D2	0 1	disable antenna load by Tcell pulse enable antenna load by Tcell pulse
D1	0 1	disable synth counter load by Tcell pulse enable synth counter load by Tcell pulse
D0	0 1	disable synth counter basic rate clock enable synth counter basic rate clock

Note:

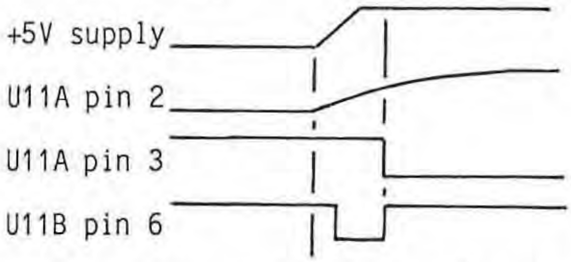
D4 and D5 , once latched , are called SC0 and SC1 respectively.

Control Register B

Control Register B consists of latches U9 and U4. This control register is intended for data that may change during an ionogram. Assignment table :-

Controller Data bus bit		Function
Bit No.	State	
D7	0 1	spare control bit B7 available on PCB only
D6	0 1	spare control bit B6
D5	0 1	spare control bit B5
D4	0 1	spare control bit B4
D3	0 1	disable fast film advance enable fast film advance
D2	0 1	disable transmitter enable transmitter
D1	0 1	Internal Rx AGC External Tx AGC
D0	0 1	disable film drive enable film drive

Power on Clear



At switch on the +5V supply charges C5 through R2. The output of battery supplied Schmitt trigger NAND gate U11A remains high for about 64 ms until C5 charges to the negative going input threshold of 1.6 V when it goes low. U11B inverts this signal so that $\overline{\text{POC}}$ remains low at switch on resetting the sweep start , the Advance/Retard and film drive flip-flops. STO to U11B pin 5 allows the computer to generate a $\overline{\text{POC}}$ signal.

Sweep Start

Flip-flops U8A and U8B are initially cleared. Just less than 1 second before an ionogram is due to start strobe ST22 clocks U8A , setting it. The ionogram begins when U8B is set on the next falling edge of the 1 Hz clock signal.

The Sweep start signal enables the Tcell pulse generator and the 6018 Film recording system and also synchronises the output of the FFT rate generator with the 1Hz clock signal.

T/R Control and Antenna Selection

In Control Register A bits D5 and D4 select either Tx only , Rx only or T/R switching. T/R switching is used for vertical incidence ionograms and Tx only and Rx only for oblique ionograms. When bit D5 is low the transmitter is switched to the Oblique antenna.

Vertical incidence operation is characterised by T/R switching which is selected when SC0 = SC1 = 1. U11D and U11C detect this condition and generate the control signal Vertical Rx/$\overline{\text{Oblique Rx}}$ used by the 4040 Rx Antenna Select relay box.

4/64 Hz Select

Control Register A bit D7 together NAND gates U12A , U12B , U12C and U12D select either 4 Hz or 64 Hz as the input frequency to be used by the Tcell pulse generator board.

Video Blank and Film Drive Inhibit

Control Register B bit D0 controls this function by providing the data inputs to flip-flop U7B which is clocked by the Tcell pulse. This allows control of the film recording system synchronous with the Tcell period.

Real Time Clock , RTC

Introduction

The real time clock provides place , year , year day number , hours , minutes and seconds to both the user and to the controlling computer. It consists of a battery backed-up clock multiplexer board and a display unit.

Circuit Description

The clock is driven by 1 Hz from controller board PC3b. This signal clocks dual up counters U11 , U12 , U13 , U14 and U15 which are interconnected to count seconds , minutes , hours , days and unit years. The 1 Hz pulse falling edge also triggers 1 microsecond monostable U22 which resets both counters in dual up counter U23.

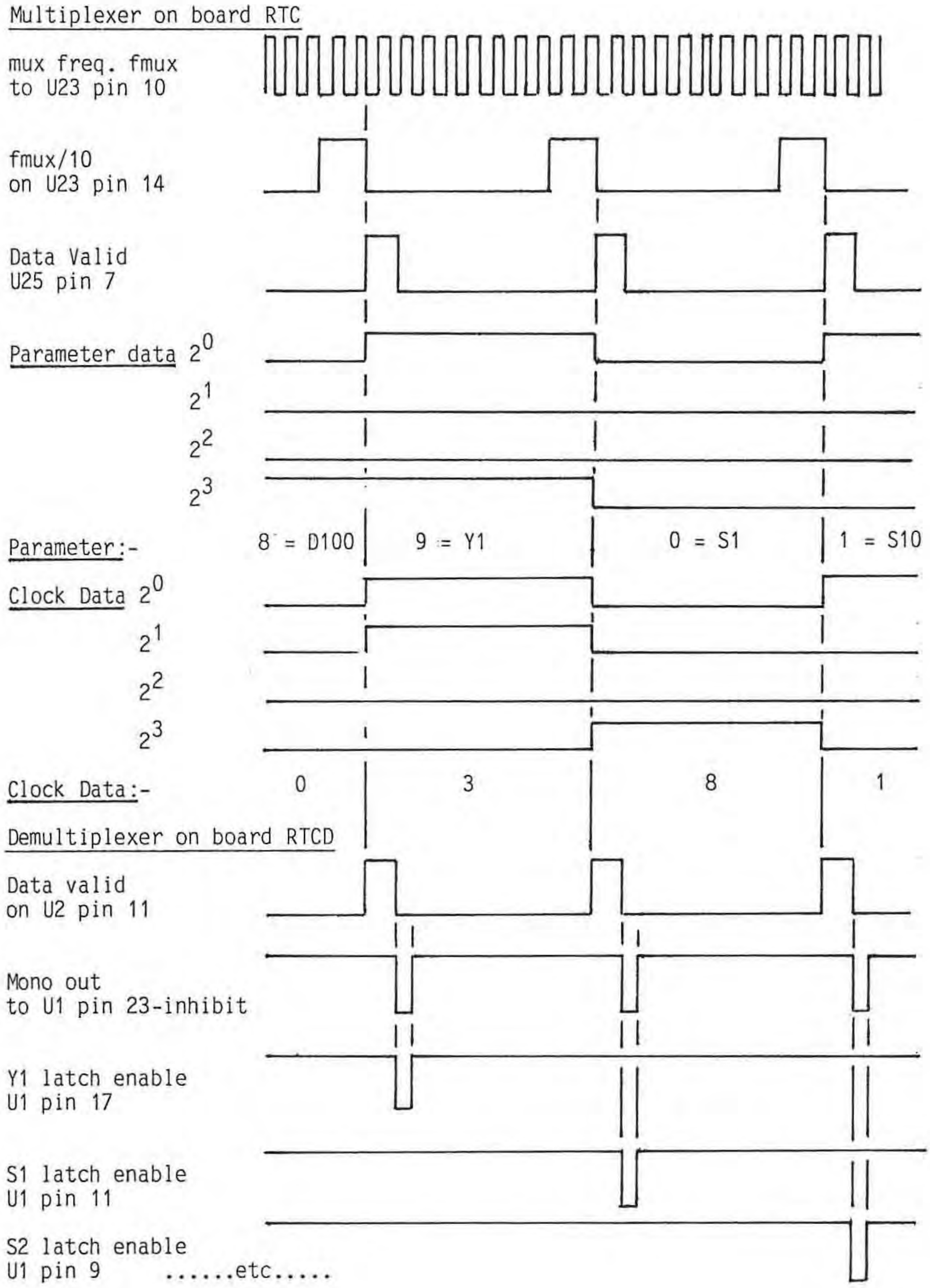
The multiplexing frequency of 9.6 kHz is obtained from the computer 600b clock line. This frequency is divided by 10 by half of dual up counter U23. The output clocks the other half as well as triggering the other monostable in U22. The monostable output on pin 6 , a 10 microsecond pulse , is buffered by U25. This pulse, the data valid pulse , is used by the demultiplexing circuitry.

The outputs of the counter that is clocked at 0.96 kHz address 4 to 16 line decoder U24 which enables one of the ten 3-state buffers U1 through U10 , putting the corresponding BCD clock data onto the clock data bus. The outputs of this counter are also buffered by U25 and provide parameter information.

Parameter	0	1	2	3	4	5	6	7	8	9
Clock Data	S1	S10	M1	M10	H1	H10	D1	D10	D100	Y1

Q1 , R8 , R9 and R10 disable 3-state buffers U1 through U10 when the main supply fails. The front panel rotary switch is used to to selectthe digit to be changed. The toggle switch increments the selected digit by 1 each time it is pressed. U26D and U26C debounce the toggle switch.

Timing Diagrams



Real Time Clock Display , RTCD

Introduction

This board demultiplexes the BCD clock data using the 4 bit parameter data and displays it in the form Y1 D100 D10 D1 H10 H1 M10 M1 S10 S1. Preceding this data is a place identifier and one digit for tens of years. These two digits are set using DIL switches.

Circuit Description

The BCD clock data bus goes to BCD-to-7 segment latch/decoder/drivers U3 through U12 all of which drive MAN 54 7 segment displays via 150 ohm resistors.

The parameter data to U1 selects one of the BCD-to-seven segment latch/decoder/drivers to latch the data on the clock data bus. Monostable U2 , triggered by the data valid pulse falling edge inhibits 4 to 16 line decoder U1 for 5 microseconds to ensure that the clock data is stable before it is latched.

Basic Rate , PC4a

Introduction

The Basic Rate is the linear rate at which the transmitter frequency is increased during a cell. This circuit allows the selection of a number of a number of different basic rate clocks and hence basic rates.

Circuit Description

ST10 clocks U14 to latch data from the Controller data bus. Either 100 kHz or 1 MHz can be selected by output Q3 of U14 and U1 to go to U4 and U5. U4 divides the input by 2 , 4 , 8 and 16 and U5 divides by 5 and 10. All 6 frequencies together with the input frequency are applied to the inputs of 8-input multiplexer U9. One of these rates is selected by U14 outputs Q0 , Q1 and Q2 which address U9. The selected frequency is the basic rate clock used to clock the BCD counters which drive the synthesizer. Because the programmed synthesized sweep is multiplied by 4 a clocking frequency of 12.5 kHz produces a 50 kHz ramp or Basic Rate.

Basic Rate Select Data

Controller Data Bus Utilization

D7	D6	D5	D4	D3	D2	D1	D0
not used				Basic Rate Select			

Basic Rate Strobe

Decimal ST10

Hex ST0A

Basic Rate Select Table

Binary Data				Hex Data	Basic Rate Clock	Basic Rate
D3	D2	D1	D0	Hex	kHz	KHz/s
0	0	0	0	00	6,25	25
0	0	0	1	01	10	40
0	0	1	0	02	12,5	50
0	0	1	1	03	20	80
0	1	0	0	04	25	100
0	1	0	1	05	50	200
0	1	1	0	06	100	400
0	1	1	1	07	-	-
1	0	0	0	08	62.5	250
1	0	0	1	09	100	400
1	0	1	0	0A	125	500
1	0	1	1	0B	200	800
1	1	0	0	0C	250	1000
1	1	0	1	0D	500	2000
1	1	1	0	0E	1000	4000
1	1	1	1	0F	-	-

Tcell Pulse Generator , PC4b

Introduction

The cell is the fundamental building block of the system. During a cell the ionosonde frequency can increase at some linear rate called the Basic Rate or it can remain fixed. The cell period is called Tcell. The Tcell circuitry on PC4b generates the Tcell pulse from one of two sets of eight different clock rates.

Circuit Description

The input frequency used by the Tcell pulse generator board is selected by D7 of Control Register A on PC3b. D7 = 0 selects the Doppler Tcell rate of 4 Hz and D7 = 1 selects the ionogram Tcell rate of 64 Hz.

U10 and U11 divide this input by 2 , 4 , 8 , 16 , 32 , 64 , 128 and 256. Data latched by ST11 in U15 selects one of these 8 rates to trigger monostable U16 if the sweep is high. The monostable output pulse signals the beginning of a cell.

An Ionogram begins when Sweep changes from low to high. This signal triggers monostable U9 on PC7 which generates a sync on second pulse clearing Tcell dividers U10 and U6 and setting U11 output low. If U16 Pins 3 and 4 are initially low , sweep going high triggers the first Tcell pulse. If U16 pins 3 and 4 are initially high and Sweep goes high the first Tcell pulse is generated when the sync on second pulse clears the Tcell dividers causing a high to low transition on U16 pins 3 and 4.

Tcell Select Data

Controller Data Bus Utilization

D7	D6	D5	D4	D3	D2	D1	D0
not used					Tcell Select		

Tcell Strobe

Decimal = ST11

Hex = ST0B

Tcell Period Select Table

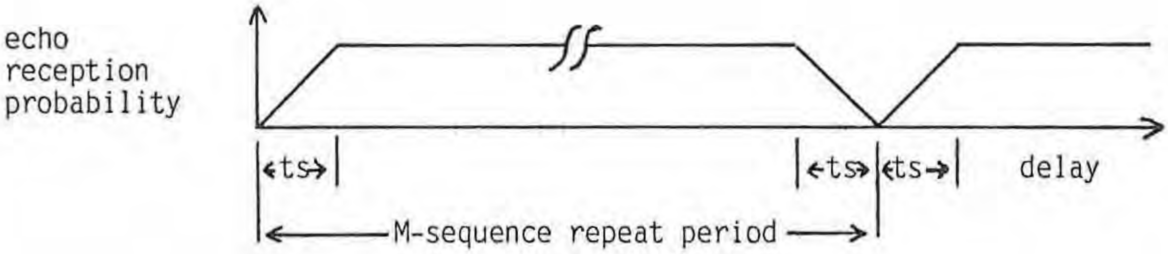
Binary Data			Hex Data	Tcell Period (s)	
D2	D1	D0	-	64 Hz in	4 Hz in
0	0	0	00	1/32	1/2
0	0	1	01	1/16	1
0	1	0	02	1/8	2
0	1	1	03	1/4	4
1	0	0	04	1/2	8
1	0	1	05	1	16
1	1	0	06	2	32
1	1	1	07	4	64

Programmable M-Sequence Generator , PC4c

Introduction

Since the minimum target delay is short compared to the duration of a cell it is necessary to employ T/R switching.

A maximum length sequence or M-sequence (also known as a pseudo-random binary sequence) was choosen as the switching waveform as the echo reception probability versus delay (auto-correlation function) is constant except for delays from zero to a delay equal to a single bit period where it rises linearly. At integral multiples of the M-sequence repeat period it also falls to zero then rises to the constant value again. The M-sequences used have repeat periods much greater than the maximum delay of interest.



ts = single bit period

In ionospheric measurements the minimum target range or virtual height vh is approximately 90 Km.

The corresponding delay $T = (2 * vh) / c$, where $c = 3*10^8$ m/s

$$= (2 * 90 * 10^3) / 3*10^8 \text{ s}$$

$$= 600 \text{ us}$$

tsmax = 600 us is thus the maximum single bit period allowed before reduced echo reception probability occurs at the delays of interest.

The minimum single bit period is determined by the maximum rate of transmit/receive switching possible and is about 200 us.

An M-sequence can be generated by using a shift register with appropriate feedback. Certain bits are modulo-two summed (exclusive-OR'ed) and fed back into the shift register. An n bit shift register can have 2^n possible states. An M-sequence of order n can have 2^n-1 possible states before repeating since the all zero state is invalid .

Circuit Design

An 11 bit shift register has $2^{11} = 2048$ possible states.Choosing a clock frequency of 2048 Hz results in a single bit period of 1/2048 = 488.28 us. Related to height , $vh = (t * c) / 2$

$$= 488.28 * 10^{-6} * 3 * 10^{8}$$

$$= 73.2 \text{ Km}$$

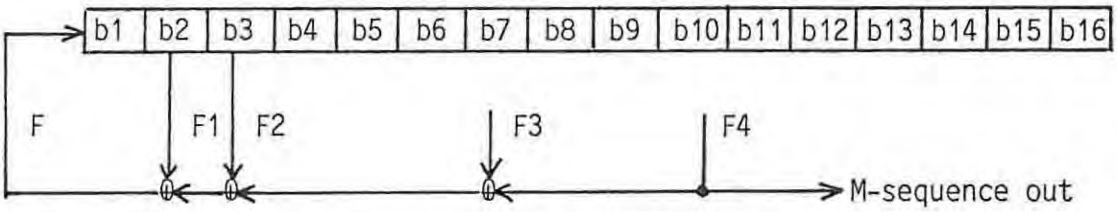
73 Km is thus the half width of the notch near zero delay in the autocorrelation function and is well below the E-region.

An 11 bit M-sequence can be clocked $2^{11}-1$ times before it repeats. Clocked at 2048 Hz the M-sequence will complete in 2047 clock pulses and will begin again on the 2048th clock pulse. For Tcell = 1 s the start of the M-sequence relative to Tcell will precess by 1 clock pulse (488.28 us) each second . If the M-sequence is reset at the start of each cell by the Tcell pulse the code will not precess and will be identical for each cell.

A clock frequency of 2048 Hz implies 1024 pulses in half a second . $2^{10} = 1024$, so that an M-sequence of order 10 would repeat every 1023 clock pulses . Thus , for a number of Tcell periods all related by a factor of 2 (ie 4 s , 2 s , 1 s , 1/2 s ...) , it is only necessary to change the order of the M-sequence. The programmable M-sequence generator allows the selection of sequences of order 6 through 16 to correspond to Tcell periods of 1/32 s through to 32 s.

To obtain M-sequences of order 6,7,8,9,10,11,12,13,14,15 and 16 a 16 bit shift register is required. A computer program was run to list all possible feedback points that would yeild an M-sequence of a particular order. It was discovered that any one of the M-sequences required could be obtained by using 4 feedback points only . Two feedback points , namely those from shift register bits 2 and 3 , are common to all the required orders.

The shift register bits are designated b1 through b16. The 4 feedback points are designated F1 , F2 , F3 and F4 and the modulo-2 addition of the data selected by these points F.



⊕ = modulo-2 adder or exclusive OR

Depending on the order, selection of one of b4 , b5 , b8 or b9 by F3 yeilds an M-sequence. The fourth feedback point must select data from bits b6 through b16 for M-sequences of order 6 through 16. Being the last bit in the shift register this is also the output of the M-sequence.

Let the data selected by F1 , F2 , F3 and F4 be A , B , C and D respectively. The feedback equation is then A ⊕ (B ⊕ (C ⊕ D)).

The reset on the shift registers used clears all the bits to zero. This is the only invalid bit combination in an M-sequence. If inverted , however , it becomes the valid all one's state.

Let the shift register contain inverted data. F1 , F2 , F3 and F4 then select inverted data $\overline{A}$, $\overline{B}$, $\overline{C}$ and $\overline{D}$.

Successive XORing of this data yeilds :-

$$F = \overline{A} \oplus (\overline{B} \oplus (\overline{C} \oplus \overline{D}))$$

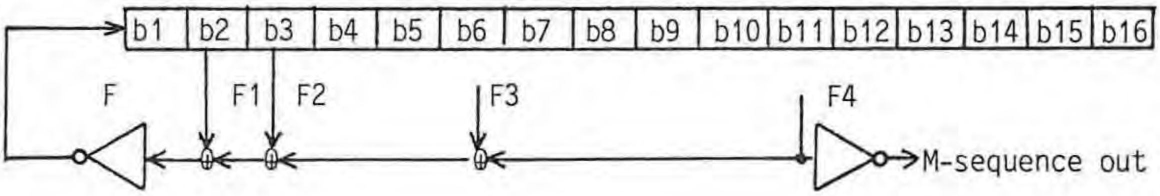
$$= \overline{A} \oplus (\overline{B} \oplus (C \oplus D)) \quad \text{applying } \overline{X} \oplus \overline{Y} = X \oplus Y$$

$$= \overline{A} \oplus (\overline{B \oplus C \oplus D}) \quad \text{applying } \overline{X} \oplus Y = \overline{X \oplus Y}$$

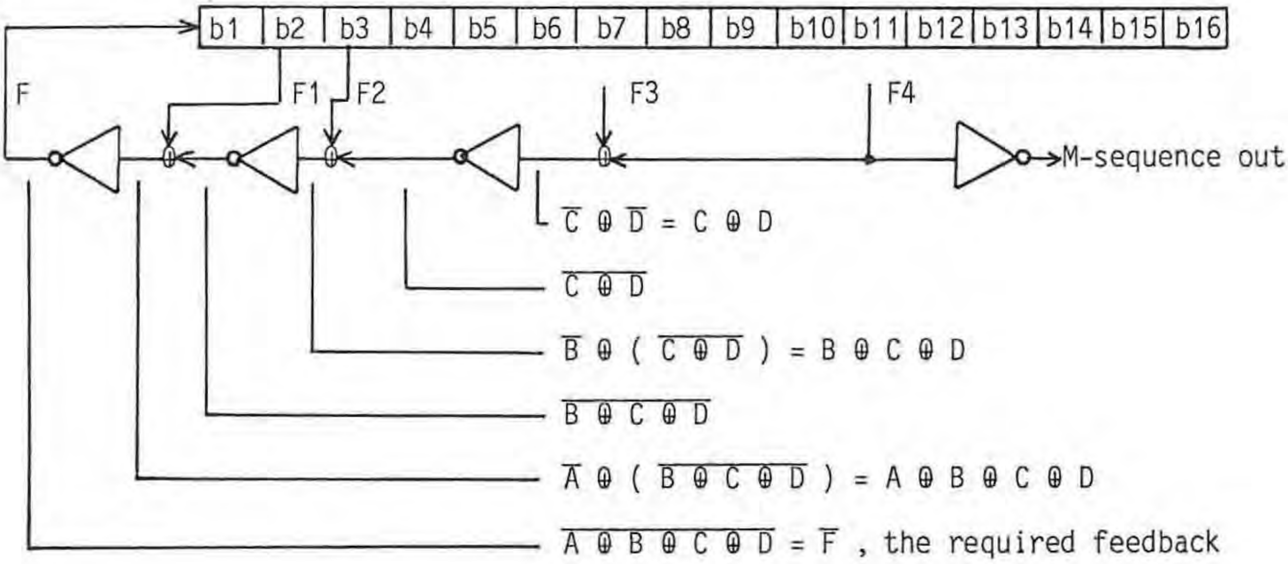
$$= A \oplus B \oplus C \oplus D \quad \text{applying } \overline{X} \oplus \overline{Y} = X \oplus Y$$

This feedback equation is the same as that obtained with non-inverted data. Because the shift register contains inverted data the feedback F must be inverted after XORing to give :-

$$\overline{F} = \overline{A \oplus B \oplus C \oplus D}$$



The XOR gate and inverter can be replaced by an XNOR gate. Replacing the 2 remaining XOR gates with XNOR gates gives the following :-



Circuit Description

U2D inverts the 2048 Hz clock signal to U7 and U8 which make up a 16 bit shift register. The first and second feedback points F1 and F2 are common to all orders of M-sequence and are taken from b2 and b3. The data from these points is fed to XNOR gates U2A pin 1 and U2B pin 5. For the third feedback point one of the four bits b4 , b5 , b8 or b9 can be selected via multiplexer U3 to U2C pin 8. Data bus bits D0 and D1 are used for this selection. The fourth feedback point is similarily selected from one of the bits b6 through b16 by data bus bits D2 , D3 , D4 , D5 and D6 addressing 2 multiplexers U12 and U13. The M-sequence of the selected order is obtained from U13 pin 6 , the Z output of the multiplexer. The Tcell pulse resets the shift register to an all zero state at the beginning of each cell.

M-sequence Select Data

Controller Data Bus Utilization

D7	D6	D5	D4	D3	D2	D1	D0
not used	M-sequence Select						

M-sequence Strobe

Decimal = ST12

Hex = ST0C

M-sequence Select Table

Binary Data							Hex Data	Feedback				M-sequence Order	Repetition Rate
D6	D5	D4	D3	D2	D1	D0	-	F1	F2	F3	F4	No.shift reg bits	seconds
0	0	0	0	0	0	1	01	b2	b3	b5	b6	6	1/32
0	0	0	0	1	0	0	04	b2	b3	b4	b7	7	1/16
0	0	0	1	0	0	0	08	b2	b3	b4	b8	8	1/8
0	0	0	1	1	0	1	0D	b2	b3	b5	b9	9	1/4
0	0	1	0	0	0	1	11	b2	b3	b5	b10	10	1/2
0	1	0	0	0	0	1	21	b2	b3	b5	b11	11	1
0	1	1	0	0	1	1	33	b2	b3	b9	b12	12	2
1	0	0	0	0	1	0	42	b2	b3	b8	b13	13	4
1	0	1	0	0	1	0	52	b2	b3	b8	b14	14	8
1	1	0	0	0	0	1	61	b2	b3	b5	b15	15	16
1	1	1	0	0	0	1	71	b2	b3	b5	b16	16	32

T/R Gain Weight Generator - replaces Amplifier T/R board A6

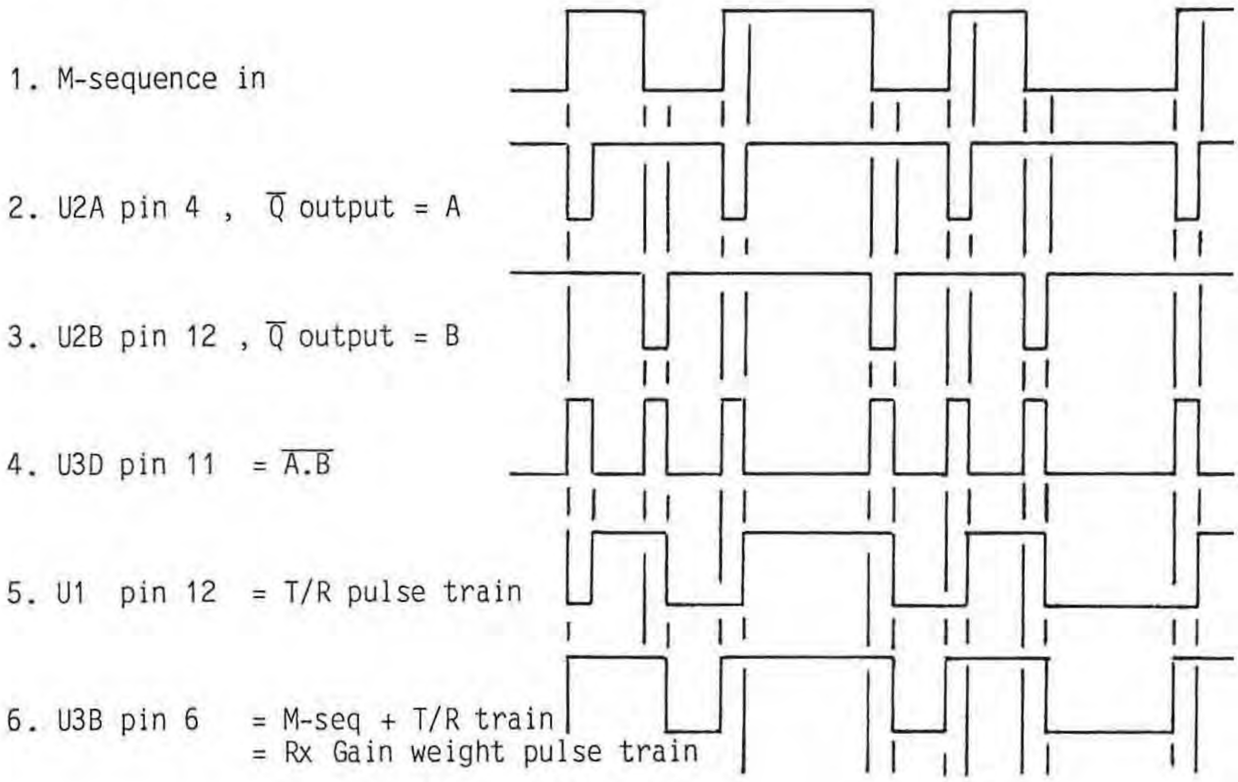
Introduction

The original Barry Research T/R rate generator board produces three pulse trains. These are the T/R pulse train , the Rx gain weight and the Tx gain weight pulse train. The pulse trains are derived from a common clock frequency which can be modulated at a 1 Hz rate to eliminate the blind range associoated with the Barry Research 16 bit T/R code. The pulse trains are all the same except for the Rx gain weight pulse train which rises 180 microseconds before the other two.

The T/R Gain Weight Generator board generates the three pulse trains from the M-sequence generated on PC4c and plugs directly into the A6 slot in the 4050 Amplifier T/R.

Circuit Description

Timing Diagram



The M-sequence rising edge triggers monostable U2A producing a 150 microsecond low going pulse. Because U3D pin 13 is high a high going pulse is obtained on pin 11. The falling edge of this pulse clocks flip-flop U1 which has the M-sequence pulse train applied to the J input. Because this is at a logic 1 level the output of U1 on pin 12 goes high.

The M-sequence falling edge triggers monostable U2B producing a 150 microsecond low going pulse. Because U3D pin 12 is high a high going pulse is obtained on pin 11. The falling edge of this pulse clocks flip-flop U1 changing the output from high to low as the M-sequence pulse train on the J input is low.

If both monostables have equal periods the waveform obtained on U1 pin 12 is the input M-sequence delayed by 150 microseconds. This delayed M-sequence is then the T/R pulse train. U3A , U3C and U3B OR the input M-sequence with the delayed M-sequence to produce the receive gain weight pulse train which rises 150 microseconds before the T/R pulse train.

Control lines SC0 and SC1 from Control Register A on PC3b together with U4A , U4B , U4C and U4D are used to select Tx only , Rx only or T/R switching. The table below the control data , function selected and the states of the output lines.

Control Lines		Function	Output Pulse Train		
SC1	SC0	-	Rx Gain Wt.	Tx Gain wt.	T/R Pulse Train
0	0	Tx only	1	1	1
0	1	Tx only	1	1	1
1	0	Rx only	0	0	0
1	1	T/R switch	waveform 6	waveform 5	waveform 5

Advance Retard , PC5

Introduction

For oblique synchronization the timing of the receiving ionosonde relative to the transmitting ionosonde must be adjustable. This timing change must affect both the ionogram start times as well as the instantaneous frequency of the ramp during an oblique ionogram.

The 100 kHz square wave is divided down to 1 Hz for the hardware Real Time Clock and is also used to generate the basic rates used in producing oblique ionograms. By adding in or deleting pulses from the 100 kHz square wave both requirements can be met.

Circuit Description

The 100 kHz , 1Vrms output of the 105B Quartz Oscillator is squared by Q2. Dividers U1 , U2 and U3 provide the correct intervals at which to add in or delete pulses from the 100 kHz signal. Each pulse is equivalent to $1/10^5$ seconds , that is 10 microseconds.

U1 divides 100 kHz by 10 to provide 10^4 pulses per second. Adding in or deleting a pulse from the 100 kHz square wave at 10^4 pulses per second results in a timing slip rate of 10^4 x 10 microseconds = 100 ms/s.

The output of U2 gives 10^3 pulses per second for a slip rate of 10^3 x 10 microseconds = 10 ms/s and the output of U3 gives 10^2 pulses per second for a slip rate of 10^2 x 10 microseconds = 1 ms/s.

U13 is loaded with shift direction and rate data.

Controller Data bus	Logic 0	Logic 1
D0	Advance	Retard
D1	disabled	1 ms/s rate
D2	disabled	10 ms/s rate
D3	disabled	100 ms/s rate

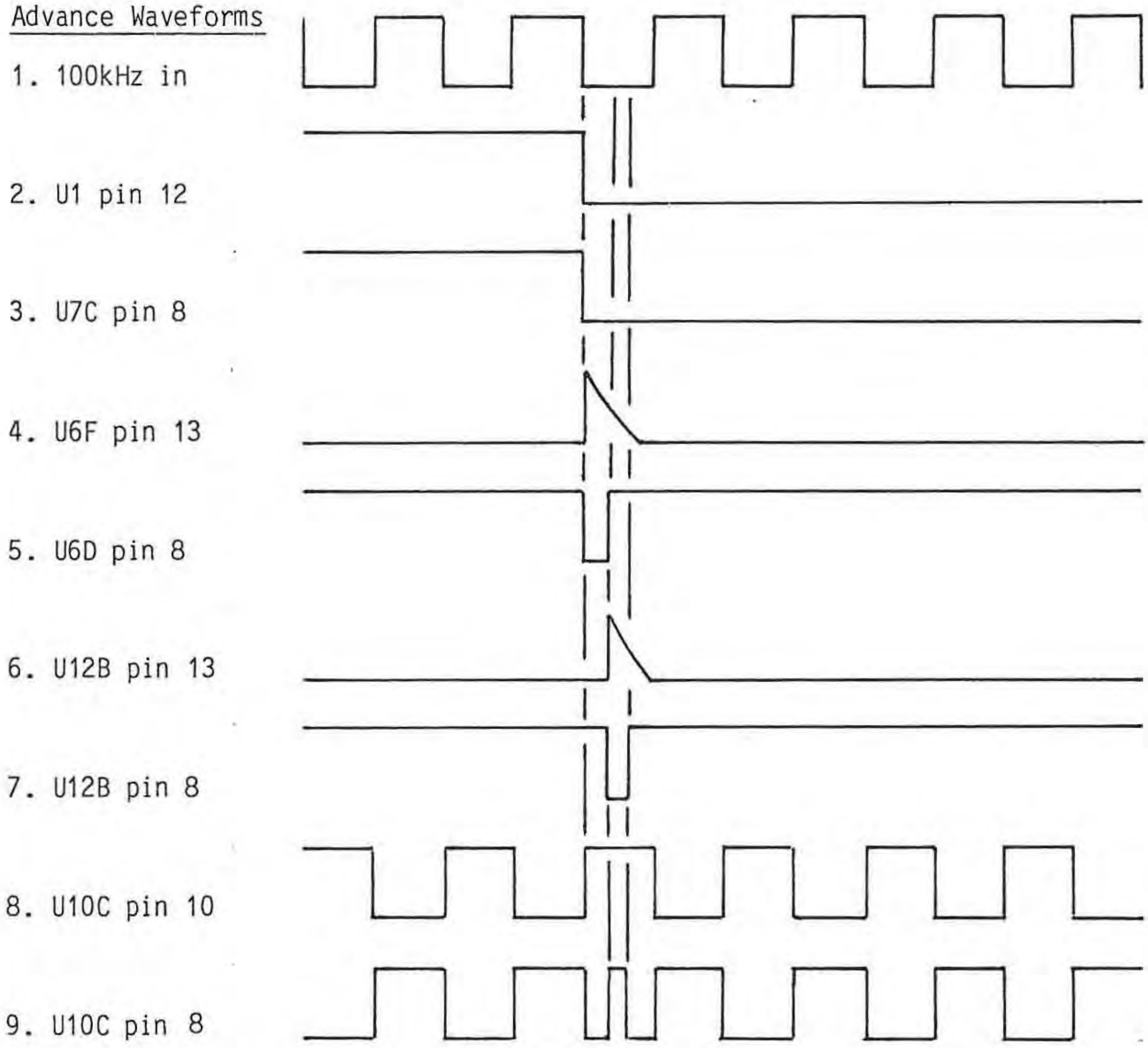
Note :- Only one of the three rates must be enabled at any time.

With the control latch U13 set the required shift is parallel loaded into two four bit binary counters , U9 and U14. Strobe 26 (ST26) initiates the timing slip by setting flip-flop U4A. The selected rate clocks flip-flop U4B enabling either U12A or U12B. With retard selected U6A , U12A and U10D delete pulses from the 100 kHz waveform retarding the timing. To advance the timing U6B , U6F , U6E , U6D , U12B and U10C add pulses to the 100 kHz square wave.

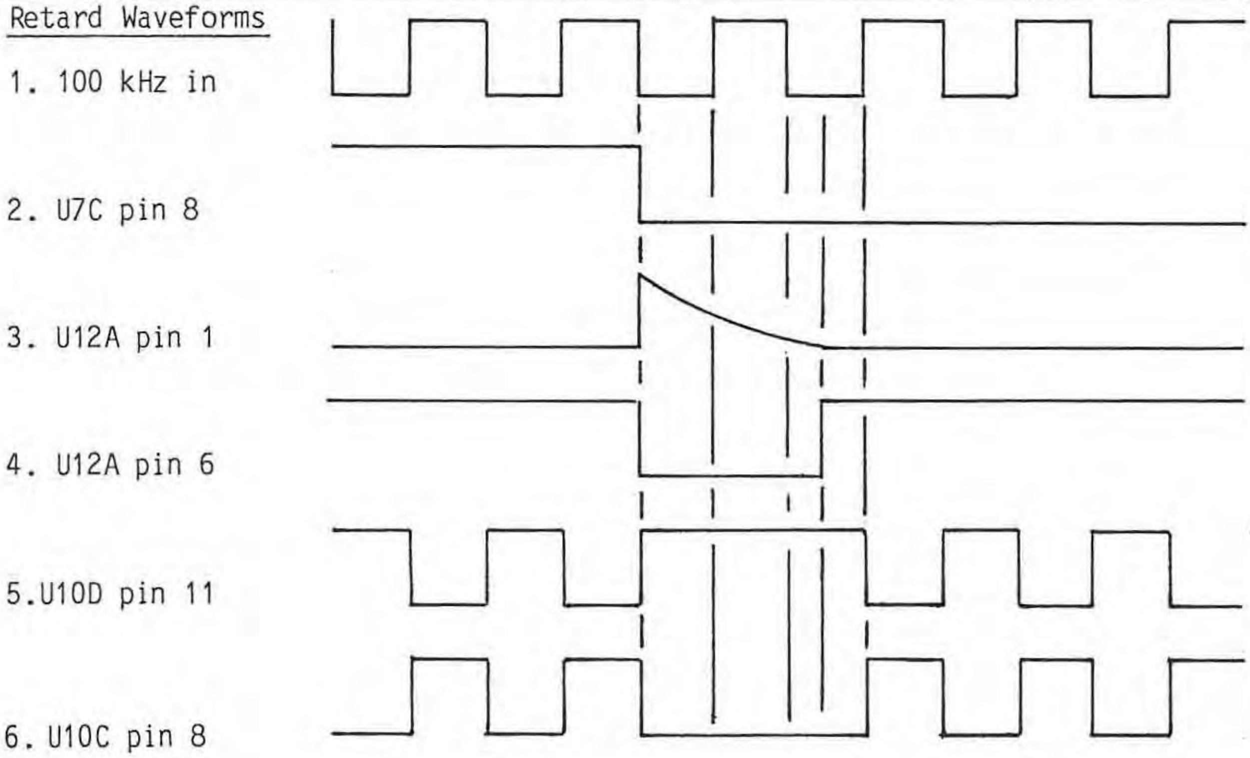
Either the pulses that added to the 100 kHz signal or those that are used for pulse deletion decrement counters U9 and U14. When zero is reached flip-flops U4A and U4B are cleared and the timing slip stops.

The software executes a required shift by loading and slipping in multiples of 1 ms followed by a remainder. The smallest possible shift is 10 microseconds.

Advance Waveforms



Retard Waveforms



Advance Retard Select data

Controller Data Bus Utilization

Control Data

Data bit	Function
D7 to D4	not used
D3	0 - disable 100 ms/s 1 - enable 100 ms/s
D2	0 - disable 10 ms/s 1 - enable 10 ms/s
D1	0 - disable 1 ms/s 1 - enable 1 ms/s
D0	0 - Advance 1 - Retard

Advance Retard Control Data Strobe

Decimal = ST23

Hex = ST17

Shift Data

D7	D6	D5	D4	D3	D2	D1	D0
Shift Data							

Advance Retard Shift Data Strobe

Decimal = ST25

Hex = ST19

Strobe Decode , C6

Introduction

The Controller hardware requires that data from the computer be latched to select one of a number of possible outputs of a particular controlled parameter. For example , selection of a basic rate clock frequency from one of the ten that are generated on PC4a.

The method used for parameter selection makes use of both sides of a PIA with the A side outputting the data and the B side generating a strobe pulse via suitable circuitry.

Data written to the PIA A side is latched by the PIA and distributed to the controller hardware. Data written to the PIA B side together with the CB2 output configured as a write strobe with E restore are used by the Strobe Decode board C6 to generate the strobes required for data latching and system control.

Circuit Operation

PIA MP-LA-1 , A side

The A side latches the data written to it by the computer. This data is distributed to the controller hardware on the controller data bus.

PIA MP-LA-1 , B side

The B side data output lines together with the CB2 output are decoded on the Strobe Decode board , PC6. Data bits D6 and D7 are not used.

Bits D5 , D4 and D3 address one of 8 decoder U5 which selects one 8 other decoders for the duration of the $\overline{\text{WRE}}$ (CB2 output) pulse. These bits thus select a set of eight strobe lines.

Bits D2 , D1 and D0 select , via the enabled decoder , one of the eight output lines which pulses low. The duration of the strobe pulse thus generated is approximately 1 microsecond with the microprocessor running at 1 MHz.

This arrangement allows for up to 64 strobe lines which can be used for data latching or clocking logic. Strobe line signal names have been abbreviated from $\overline{\text{Strobe n}}$ to STn. At present the system uses all the strobe lines from ST0 to ST23 and ST25 to ST28. The 4 spare strobe lines available on the chassis edge connector are ST24 , ST29 , ST30 and ST31. The upper edge connector is not used.

Strobe Signal Functions

Strobe Signals		Function
Decimal	Hex	-
ST0	ST00	Clear sweep , A/R and film drive flip-flop
ST1	ST01	Clear sweep flip-flop
ST2	ST02	Clear A/R flip-flop
ST3	ST03	Load Control Register A
ST4	ST04	Load Control Register B
ST5	ST05	Synth latch 10^0 , 10^1
ST6	ST06	Synth latch 10^2 , 10^3
ST7	ST07	Synth latch 10^4 , 10^5
ST8	ST08	Synth latch 10^6 , 10^7
ST9	ST09	Synth counter initial load
ST10	ST0A	Basic Rate data load
ST11	ST0B	Tcell pulse period data load
ST12	ST0C	M-sequence data load
ST13	ST0D	spare
ST14	ST0E	FFT data load
ST15	ST0F	load antenna latches U1 , U2

Strobe Signal Functions

Strobe Signal		Function
Decimal	Hex	-
ST16	ST10	Antenna data initial load
ST17	ST11	Load film speed data
ST18	ST12	Rx AGC ext clock , data to DCS
ST19	ST13	Rx Phase sync , data to DCS
ST20	ST14	Load LO offset select data
ST21	ST15	Load LO offset data
ST22	ST16	Sweep Start
ST23	ST17	Load A/R control data
ST24	ST18	Spare
ST25	ST19	Load A/R shift data
ST26	ST1A	A/R shift start
ST27	ST1B	Time mark for stationary Doppler
ST28	ST1C	Baseband filter control (unused)
ST29	ST1D	Spare
ST30	ST1E	Spare
ST31	ST1F	Spare

FFT Sample Rate , PC7

Introduction

The FFT unit was designed to analyse the 512 Hz band limited dual channel receiver baseband signal. A sample rate clock of 1024 Hz is required to collect 1024 samples for the transform.

Circuit Description

The 1 MHz sine wave from the 105B Quartz oscillator is buffered by Q2 before being divided by 5 by U1 to 200 kHz. U6 divides the 200 kHz by 5 to 40 kHz which is used as the reference frequency for phase-frequency detector U10. The phase detector drives VCO U2 at 10.24 MHz. U7 and U11 divide 10.24 MHz by 256 to give 40 kHz at the phase detector variable input. This is locked onto the reference 40 Khz by the phase-locked loop.

The 10.24 MHz signal is divided by 1000 by U3 , U4 and U5 to give 10.24 kHz. U12 divides this by 5 to give the 2048 Hz clock used by the M-sequence generator. 10.24 kHz is also divided by 10 by U8 yeilding 1024 Hz. U13 and U14 are binary counters which produce 512 , 256 , 128 , 64 , 32 , 16 , 8 and 4 Hz.

Data latched in U15 addresses multiplexer U16 selecting sample rates ranging from 1024 Hz through to 8 Hz. 4 Hz and 64 Hz are taken off the board to drive the Tcell pulse generator.

U9 produces a sync on second pulse at the beginning of an ionogram when the sweep signal goes from low to high. This synchronises the FFT sample rate with the 1 second pulse from the system clock on PC3a.

FFT Rate Select Data

Controller Data Bus Utillization

D7	D6	D5	D4	D3	D2	D1	D0
not used					FFT Rate Select		

FFT Sample Rate Strobe

Decimal = ST14

Hex = ST0E

FFT Sample Rate Select Table

Binary Data	Hex Data	FFT Rate
D2 D1 D0	-	Hz
0 0 0	00	8
0 0 1	01	16
0 1 0	02	32
0 1 1	03	64
1 0 0	04	128
1 0 1	05	256
1 1 0	06	512
1 1 1	07	1024

Antenna Select , PC8a

Introduction

The antenna select logic together with the antenna switch box select one of 8 antennae to either of two receivers. Unused antennae are kept grounded. Antenna selection data can be changed directly by computer or by Tcell pulse.

Circuit Description

Rx1 antenna select bits D0 , D1 and D2 together with enable bit D3 are latched in U1 and Rx2 antenna select bits D4 , D5 and D6 plus enable bit D7 are latched in U2 by computer strobe ST15.

The antenna load pulse generated on PC3b transfers Rx1 and Rx2 data to U5 and U6 respectively. This pulse is either a computer strobe (ST16) or Tcell pulse. The outputs of U5 address 1 of 8 decoder/demultiplexer U9 selecting the antenna for Rx1 and the outputs of U6 address 1 of 8 decoder/demultiplexer U10 to select the Rx2 antenna. The outputs of U9 and U10 are active low , that is they all remain high except for the selected line.

All 8 antenna have a normally high grounding control line driven by an AND gate. Selection of an antenna to either Rx1 or Rx2 enables that antenna by driving its grounding control line low.

Antenna Select Data

Controller Data Bus Utilization

D7	D6	D5	D4	D3	D2	D1	D0
enable Rx2	Rx2 antenna select			enable Rx1	Rx1 antenna select		

Antenna Select Strobe

Decimal = ST15

Hex = ST0F

Antenna Select Data Table

All data in hex.

		Rx1 Antenna								
Antenna No.		Off	1	2	3	4	5	6	7	8
Rx 2 Antenna	Off	00	08	09	0A	0B	0C	0D	0E	0F
	1	80	88	89	8A	8B	8C	8D	8E	8F
	2	90	98	99	9A	9B	9C	9D	9E	9F
	3	A0	A8	A9	AA	AB	AC	AD	AE	AF
	4	B0	B8	B9	BA	BB	BC	BD	BE	BF
	5	C0	C8	C9	CA	CB	CC	CD	CE	CF
	6	D0	D8	D9	DA	DB	DC	DD	DE	DF
	7	E0	E8	E9	EA	EB	EC	ED	EE	EF
	8	F0	F8	F9	FA	FB	FC	FD	FE	FF

Film Speed Control , PC8b

Introduction

The film speed control board generates two signals , a 500 Hz fast film advance signal and a variable film rate signal. The variable film rate signal can be enabled or disabled by the computer.

Circuit Description

U3 divides the 1 kHz input signal by 2 to 500 Hz which is used in the 6018 Camera Interface for fast film advance.

Film speed select data is latched by U11 and U12. Counter U7 is clocked down at 500 Hz. $\overline{\text{TCD}}$ goes low when zero is reached and clocks U4 as well as parallel loading U7 from U11. U7 therefore divides the 500 Hz clock by the number on its parallel inputs which is set by bits D0 , D1 , D2 and D3 and latched by U11.

The resulting pulse train is divided by 2 , 4 , 8 and 16 by U4 and one of these is selected by 4 input multiplexer U8 addressed by latched data bits D4 and D5. Data bits D6 and D7 are not used.

The usual film speed is 6 mm/min which corresponds to hex code 2C and a frequency of 5.2 Hz.

Film Speed Select Data

Controller Data bus Utillization

D7	D6	D5	D4	D3	D2	D1	D0
not used		second division		first division			

Film Speed Select Strobe

Decimal = ST17 Hex = ST11

Film Speed Select Table

500 Hz corresponds to a film speed of 600 mm/min.

First Division Data bits D3,D2,D1,D0		Second Division Data bits D5 , D4							
Hex Data	Division by	Data 0		1		2		3	
		/2		/4		/8		/16	
—	—	Hz	mm/min	Hz	mm/min	Hz	mm/min	Hz	mm/min
0	-	-	-	-	-	-	-	-	-
1	1	250	300	125	150	62.5	75	31.3	37.5
2	2	125	150	62.5	75	31.3	37.5	15.6	18.8
3	3	83.3	100	41.6	50	20.8	25.0	10.4	12.5
4	4	62.5	75	31.3	37.5	15.6	18.8	7.8	9.4
5	5	50.0	60	25.0	30.0	12.5	15.0	6.2	7.5
6	6	41.6	50	20.8	25.0	10.4	12.5	5.2	6.2
7	7	35.7	42.8	17.9	21.4	9.0	10.2	4.5	5.1
8	8	31.2	37.5	15.6	18.8	7.8	9.4	3.9	4.7
9	9	27.8	33.3	13.9	16.7	6.9	8.3	3.4	4.2
A	10	25.0	30.0	12.5	15.0	6.2	7.5	3.1	3.8
B	11	22.7	27.3	11.4	13.6	5.7	6.8	2.8	3.4
C	12	20.8	25.0	10.4	12.5	5.2	6.2	2.6	3.1
D	13	19.2	23.1	9.6	11.5	4.8	5.8	2.4	2.9
E	14	17.9	21.4	8.9	10.7	4.4	5.4	2.2	2.7
F	15	16.7	20	8.3	10	4.2	5	2.1	2.5

Antenna Switch Brass Box , B1

Introduction

The antenna switch brass box , driven by the antenna select logic on PC8a , selects one of eight antenna to either of two receivers. Unused antenna are kept grounded.

Circuit Description

Normally open DIP reed relays TRL1 through to TRL8 select the antenna to Rx1 and normally open DIP reed relays BRL1 through BRL8 select the antenna to Rx2. Normally closed DIP reed relays ERL1 through ERL8 keep all the antennae grounded except for those selected.

Note that the normally open reed relays are either 831A-4 which have an internal protection diode or 831A-3 together with an external protection diode.

All control and supply lines in the antenna switch brass box are looped through ferite beads to reduce interference on the inputs to the receivers.

Local Oscillator Offset Frequency Synthesizer , PC9

Introduction

To obtain a sine wave of frequency f from the Sine Wave Generator on PC10 a clocking frequency of 256 x f is required. The Local Oscillator Offset Frequency Synthesizer on PC9 synthesizes two clocking frequency ranges , one for stationary Doppler measurements and the other for windowing.

The clock frequencies for stationary Doppler measurements are 256 x 2 , 256 x 4 , 256 x 8 , 256 x 16 , 256 x 32 and 256 x 64 and those for windowing are 256 x (256 x 1) through to 256 x (256 x 99). The choice of sine wave frequencies is discussed in the introduction to the SSB Generator.

Circuit Description

U10 divides 10.24 MHz from the FFT Sample Rate board , PC7 , by 5 to 2.048 MHz. A second division by 5 by U5 reduces this to 409600 Hz which is used as the reference frequency to phase-frequency detector U1. The phase detector drives VCO U6 at 6553600 Hz. U11 divides this frequency by 16 to give 409600 Hz at the phase detector variable input.

6553600 Hz can be written as $2^{18} \times 5^2$. This frequency drives two synchronous decade rate multipliers U7 and U8. The rate multipliers generate an output frequency f_{out} which is related to the input frequency f_{in} by :-

$f_{out} = (M \times f_{in})/100$ where M is greater than 0 and less than or equal to 99 and is latched by U12 and U13.

$$
\begin{aligned}
f_{out} &= (M \times 2^{18} \times 5^2) /100 \\
&= (M \times 2^{16} \times 2^2 \times 5^2)/100 \\
&= M \times 2^{16} \\
&= M \times 2^8 \times 2^8 \\
&= 256 (256 \times M)
\end{aligned}
$$

for M greater than 0 and less than or equal to 99.

These frequencies are used for windowing.

f_{out} is gated to SF1 by U2A , U2B and U2C when Q3 of U14 is high.

To generate the frequencies for stationary Doppler M is set to 1 to obtain $f_{out} = 256 \times 256$. When Q3 of U14 is low , $\overline{Q3}$ is high enabling U3 and U4 which divide f_{out} by 2 , 4 , 8 , 16 , 32 , 64 and 128.

Multiplexer U9 addressed by data latched in U14 selects the required stationary Doppler offset frequency.

$\overline{Q3}$ of U14 which selects either Doppler or windowing frequencies is also taken to the Sine Wave Generator to select upper sideband for windowing and lower sideband for Doppler.

Local Oscillator Offset Frequency Synthesizer Select Data

Controller Data Bus Utilization

The four controller data bus bits D0 , D1 , D2 and D3 are used for the control of the final output frequency. Their latching strobe is :-

Decimal = ST20

Hex = ST14

The full 8 bit width of the controller data bus is used as two BCD numbers to select the rate multiplier output frequency. The latching strobe is :-

Decimal = ST21

Hex = ST15

Local Oscillator Offset Frequency Synthesizer Select Table

Control data-U14		Freq. Data-U12,U13	f_{out}	SF1	Function
Binary	Hex	BCD	Hz	Hz	-
0 0 0 0	0	00	0 (low)	0 (low)	Off
0 0 0 0	0	01	256x256x1	0 (low)	Off
0 0 0 1	1	01	256x256x1	256x2	Doppler
0 0 1 0	2	01	256x256x1	256x4	Doppler
0 0 1 1	3	01	256x256x1	256x8	Doppler
0 1 0 0	4	01	256x256x1	256x16	Doppler
0 1 0 1	5	01	256x256x1	256x32	Doppler
0 1 1 0	6	01	256x256x1	256x64	Doppler
0 1 1 1	7	01	256x256x1	0 (low)	Off
1 0 0 0	8	01	256x256x1	256x256x1	Windowing
1 0 0 0	8	02	256x256x2	256x256x2	Windowing
1 0 0 0	8	03	256x256x3	256x256x3	Windowing
1 0 0 0	8	l	l	l	l
1 0 0 0	8	99	259x256x99	256x256x99	Windowing

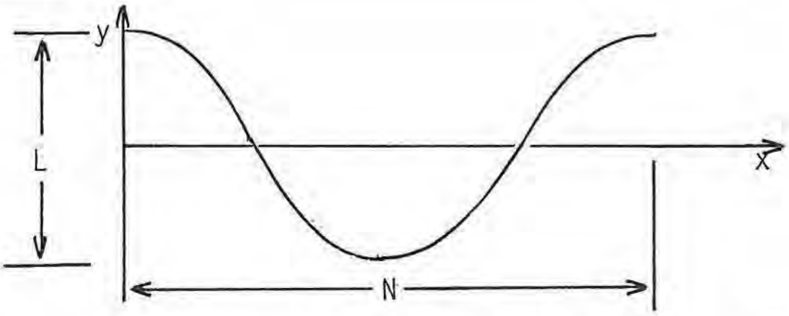
Sine Wave Generator , PC10

Introduction

This board generates two sine waves of the same frequency called S0 and S1. S1 can be selected to either lead S0 by 90^0 or lag S0 by 90^0. These signals are used in the SSB Generator brass box which generates a single sideband signal using a phase shift technique.

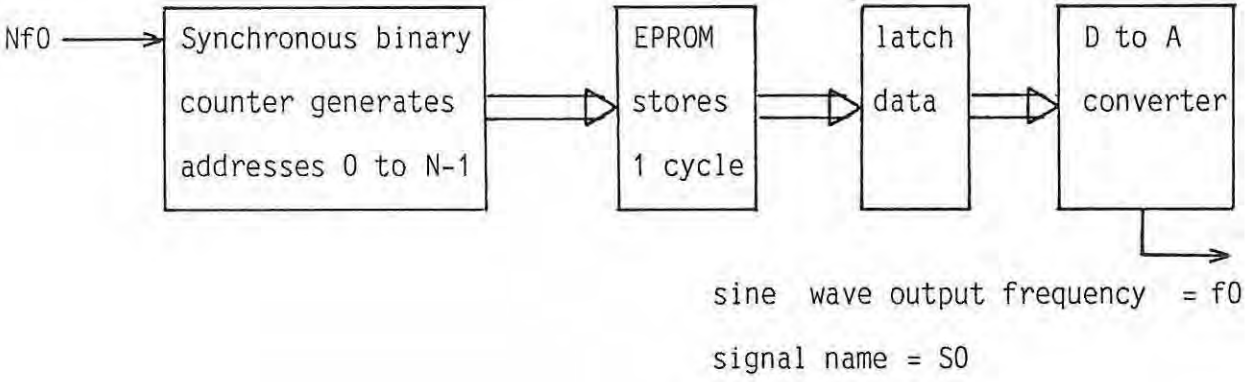
To ensure a constant 90^0 phase shift over a wide frequency range the sine wave generator must be directly coupled to the SSB generator circuitry.

Circuit Development



One cycle of a sine wave is sampled N times along the x axis. The y axis is divided into L quantised levels.

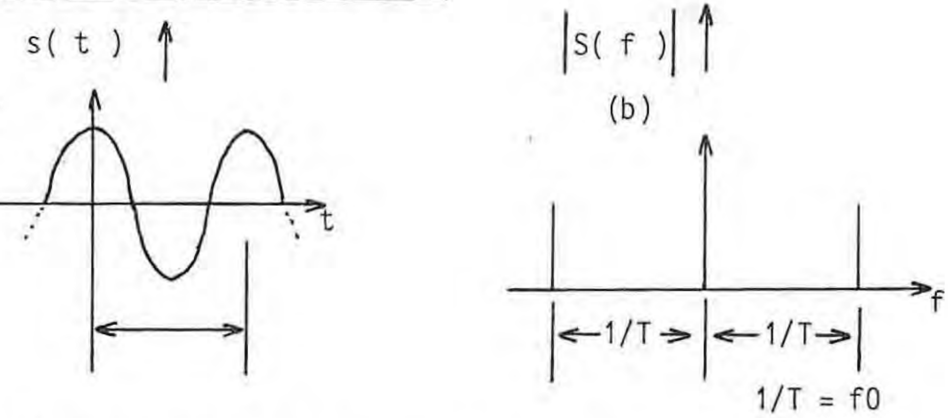
The N quantised y values are stored in an EPROM from address 0 to N-1.



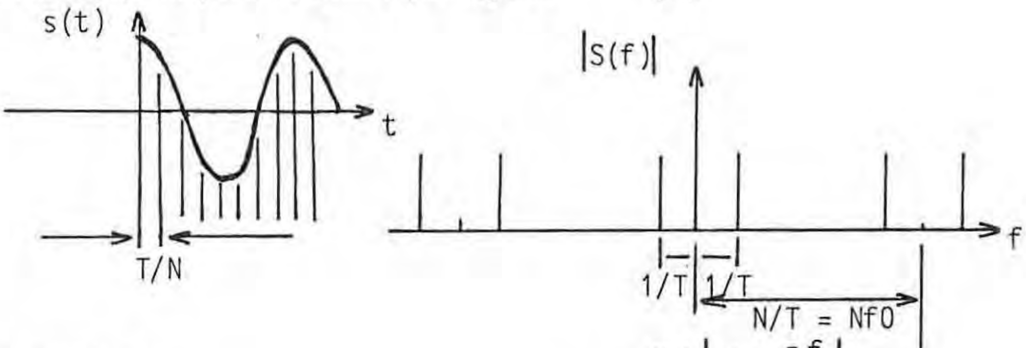
A synchronous binary counter clocked at Nf0 addresses the EPROM , the data at each address being latched before driving a digital to analogue converter. The D to A output is a sine wave of frequency f0 having L quantised steps.

The Harmonic Content of the Output Sine Wave

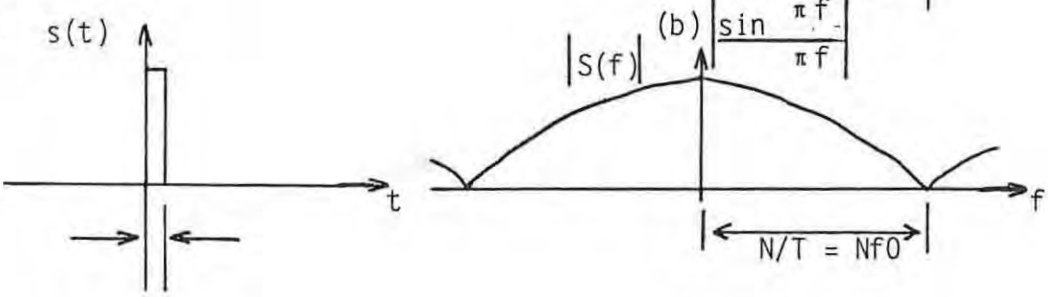
1(a) Cosine wave



2(a). Cosine wave 1(a) sampled N times per cycle

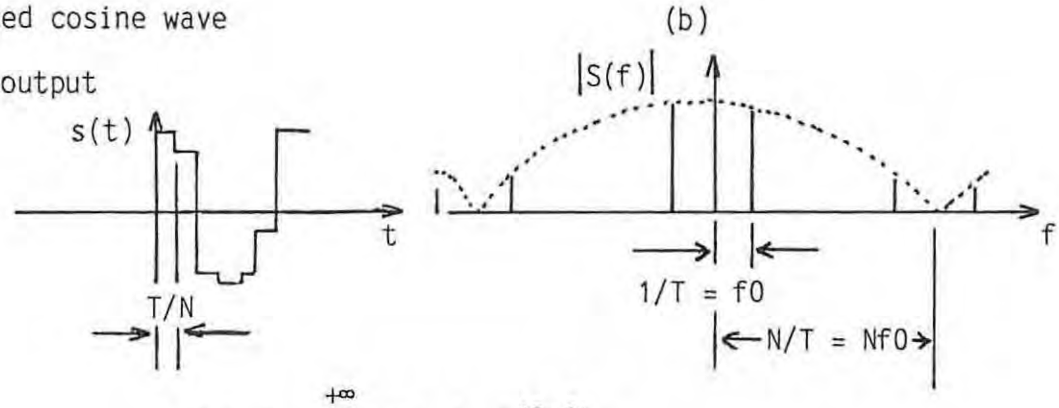


3(a). Top hat



4(a). Quantised cosine wave

D to A output



$$S(\ f\ ) = \int_{-\infty}^{+\infty} s(\ t\ )\ e^{-j2\pi ft}dt$$

$$s(\ t\ ) = \int_{-\infty}^{+\infty} S(\ f\ )\ e^{+j2\pi ft}dt$$

Figure 1(a) shows a cosine wave s(t) of frequency f0 and period T. Figure 1(b) shows the modulus of the frequency spectrum S(f) which consists of only the fundamental frequency f0.

In figure 2(b) the effect of sampling the cosine wave N times per cycle is shown in the frequency domain. Harmonics are present at (N - 1)f0 and (N + 1)f0 followed by (2N - 1)f0 , (2N + 1)f0 and so on.

The mod of the frequency spectrum of a pulse of width T/N is shown in figure 3(b). This is the mod of the sinc function , that is $\left|\frac{\sin \pi x}{\pi x}\right|$

The mod of the frequency spectrum of the quantised cosine wave of figure 4(a) is shown in 4(b) and is a convolution of 2(a) and 3(a).

The quantised cosine wave has harmonics at

f0 , (N-1)f0 , (N+1)f0 , (2N-1)f0 , (2N+1)f0 , ...

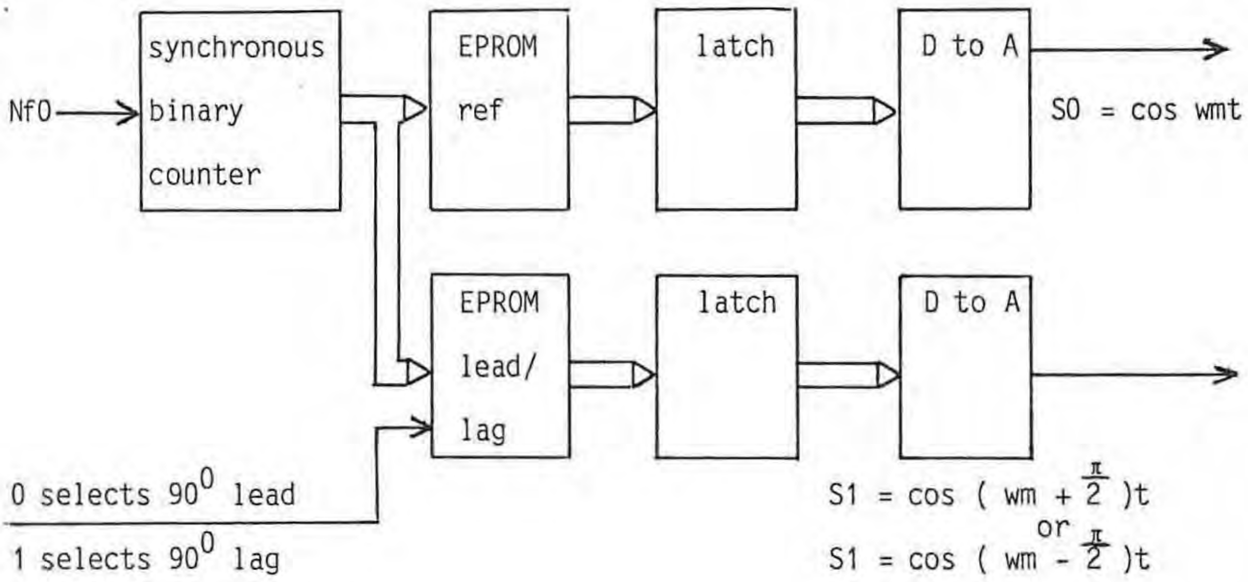
with amplitudes of

1 , 1/(N-1) , 1/(N+1) , 1/(2N-1) , 1/(2N+1) , ...

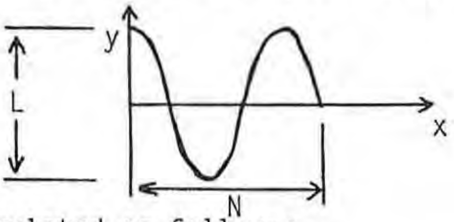
The number of samples of 1 cycle of cosine wave was chosen to be N = 256. This allows the use of an 8 bit binary counter to address the EPROM. The clocking frequency , 256 x f0 , is generated by PC9.

By chosing L = 256 levels on the y axis the full 8 bits of EPROM data are used to drive an 8 bit D to A converter. The output sine wave has therefore a first harmonic of amplitude 1/(N-1) = 1/255 of the amplitude of the fundamental at frequency (N-1)f0 = 255 x f0.

Quadrature Sine Wave Generator



To generate two sine waves having a specified phase relationship a second EPROM , latch and D to A are driven by the synchronous binary counter. Two sets of data are stored in this EPROM , the first set from addresses 0 to 255 and the second from 256 to 511. Set 1 is a $+90^0$ shifted version of the reference wave data which , when selected , gives $S1 = \cos (wm + \frac{\pi}{2})t$. Set 2 , when selected , gives $S1 = \cos (wm - \frac{\pi}{2})t$.



The EPROM data is calculated as follows:-

N = total number of x axis samples for 1 cosine cycle = 256

L = total number of steps along y axis = 256

n = n th sample

Sn = the EPROM data for the n th sampled value $0 \leq Sn \leq L$

For one cycle of cosine wave $Sn = (\cos(360^0/N)n + 1)L/2$

for $L \leq N \leq 256$ and $0 \leq n \leq 256$

The reference cosine wave data followed by the cosine plus 90^0 and cosine minus 90^0 data tables are listed below. Their FLEX file names are COSINE and COSLL (cosine lead lag) respectively.

```
                                 *** COSINE WAVE DATA TABLES ***
                                 *
                                 * X AXIS - 1 CYCLE HAS 256 SAMPLES
                                 * Y AXIS - 1 CYCLE HAS 256 QUANTISED STEPS
                                 *
                                 * COSINE TABLE
0000                                     ORG    $0000
0000 FF FF FF FF                         FCB    255,255,255,255,254,254,254,253
0004 FE FE FE FD
0008 FD FC FB FA                         FCB    253,252,251,250,250,249,248,246
000C FA F9 F8 F6
0010 F5 F4 F3 F1                         FCB    245,244,243,241,240,238,237,235
0014 F0 EE ED EB
0018 EA E8 E6 E4                         FCB    234,232,230,228,226,224,222,220
001C E2 E0 DE DC
0020 DA D7 D5 D3                         FCB    218,215,213,211,208,206,203,201
0024 D0 CE CB C9
0028 C6 C4 C1 BE                         FCB    198,196,193,190,188,185,182,179
002C BC B9 B6 B3
0030 B0 AD AA A7                         FCB    176,173,170,167,165,162,158,155
0034 A5 A2 9E 9B
0038 98 95 92 8F                         FCB    152,149,146,143,140,137,134,131
003C 8C 89 86 83
0040 80 7C 79 76                         FCB    128,124,121,118,115,112,109,106
0044 73 70 6D 6A
0048 67 64 61 5D                         FCB    103,100,97,93,90,88,85,82
004C 5A 58 55 52
0050 4F 4C 49 46                         FCB    79,76,73,70,67,65,62,59
0054 43 41 3E 3B
0058 39 36 34 31                         FCB    57,54,52,49,47,44,42,40
005C 2F 2C 2A 28
0060 25 23 21 1F                         FCB    37,35,33,31,29,27,25,23
0064 1D 1B 19 17
0068 15 14 12 11                         FCB    21,20,18,17,15,14,12,11
006C 0F 0E 0C 0B
0070 0A 09 07 06                         FCB    10,9,7,6,5,5,4,3
0074 05 05 04 03
0078 02 02 01 01                         FCB    2,2,1,1,1,0,0,0
007C 01 00 00 00
0080 00 00 00 00                         FCB    0,0,0,0,1,1,1,2
0084 01 01 01 02
0088 02 03 04 05                         FCB    2,3,4,5,5,6,7,9
008C 05 06 07 09
0090 0A 0B 0C 0E                         FCB    10,11,12,14,15,17,18,20
0094 0F 11 12 14
0098 15 17 19 1B                         FCB    21,23,25,27,29,31,33,35
009C 1D 1F 21 23
00A0 25 28 2A 2C                         FCB    37,40,42,44,47,49,52,54
00A4 2F 31 34 36
00A8 39 3B 3E 41                         FCB    57,59,62,65,67,70,73,76
00AC 43 46 49 4C
00B0 4F 52 55 58                         FCB    79,82,85,88,90,93,97,100
```

```
ANTARCTIC RESEARCH ( RHODES )        10-12-82 TSC ASSEMBLER   PAGE    2
PROM DATA

 00B4 5A 5D 61 64
 00B8 67 6A 6D 70          FCB    103,106,109,112,115,118,121,124
 00BC 73 76 79 7C
 00C0 80 83 86 89          FCB    128,131,134,137,140,143,146,149
 00C4 8C 8F 92 95
 00C8 98 9B 9E A2          FCB    152,155,158,162,165,167,170,173
 00CC A5 A7 AA AD
 00D0 B0 B3 B6 B9          FCB    176,179,182,185,188,190,193,196
 00D4 BC BE C1 C4
 00D8 C6 C9 CB CE          FCB    198,201,203,206,208,211,213,215
 00DC D0 D3 D5 D7
 00E0 DA DC DE E0          FCB    218,220,222,224,226,228,230,232
 00E4 E2 E4 E6 E8
 00E8 EA EB ED EE          FCB    234,235,237,238,240,241,243,244
 00EC F0 F1 F3 F4
 00F0 F5 F6 F8 F9          FCB    245,246,248,249,250,250,251,252
 00F4 FA FA FB FC
 00F8 FD FD FE FE          FCB    253,253,254,254,254,255,255,255
 00FC FE FF FF FF
                           END

0 ERROR(S) DETECTED
```

```
ANTARCTIC RESEARCH ( RHODES )        10-12-82 TSC ASSEMBLER   PAGE    3
PROM DATA


SYMBOL TABLE:



+++
```

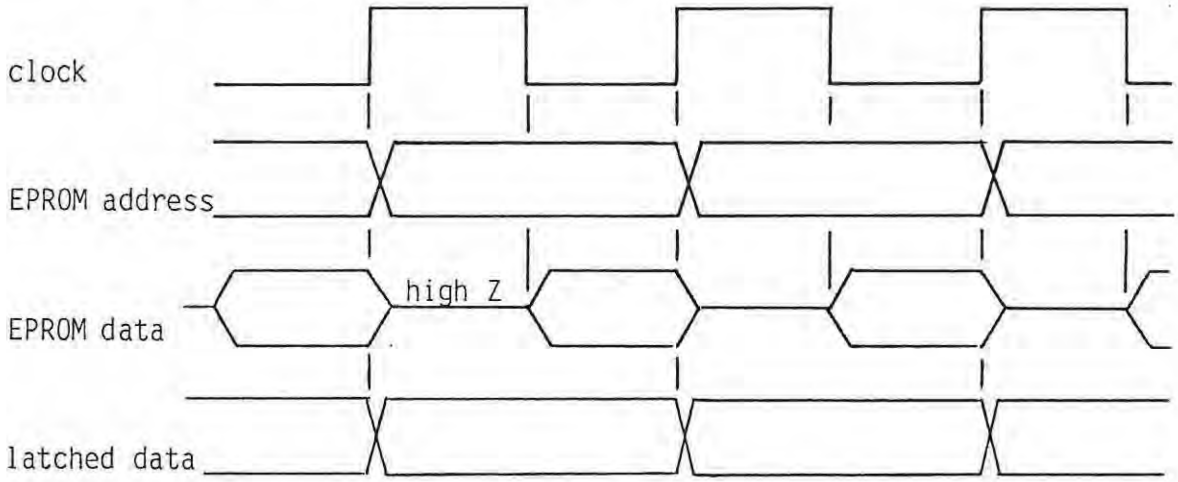
```
                         *** COSINE PLUS 90 AND COSINE MINUS 90 DATA TABLES ***
                         *
                         * X AXIS - 1 CYCLE HAS 256 SAMPLES
                         * Y AXIS - 1 CYCLE HAS 256 QUANTISED STEPS
                         *
                         *** COSINE PLUS 90 TABLE ***
                         *
0000                             ORG     $0000
0000 80 7C 79 76                 FCB     128,124,121,118,115,112,109,106
0004 73 70 6D 6A
0008 67 64 61 5D                 FCB     103,100,97,93,90,88,85,82
000C 5A 58 55 52
0010 4F 4C 49 46                 FCB     79,76,73,70,67,65,62,59
0014 43 41 3E 3B
0018 39 36 34 31                 FCB     57,54,52,49,47,44,42,40
001C 2F 2C 2A 28
0020 25 23 21 1F                 FCB     37,35,33,31,29,27,25,23
0024 1D 1B 19 17
0028 15 14 12 11                 FCB     21,20,18,17,15,14,12,11
002C 0F 0E 0C 0B
0030 0A 09 07 06                 FCB     10,9,7,6,5,5,4,3
0034 05 05 04 03
0038 02 02 01 01                 FCB     2,2,1,1,1,0,0,0
003C 01 00 00 00
0040 00 00 00 00                 FCB     0,0,0,0,1,1,1,2
0044 01 01 01 02
0048 02 03 04 05                 FCB     2,3,4,5,5,6,7,9
004C 05 06 07 09
0050 0A 0B 0C 0E                 FCB     10,11,12,14,15,17,18,20
0054 0F 11 12 14
0058 15 17 19 1B                 FCB     21,23,25,27,29,31,33,35
005C 1D 1F 21 23
0060 25 28 2A 2C                 FCB     37,40,42,44,47,49,52,54
0064 2F 31 34 36
0068 39 3B 3E 41                 FCB     57,59,62,65,67,70,73,76
006C 43 46 49 4C
0070 4F 52 55 58                 FCB     79,82,85,88,90,93,97,100
0074 5A 5D 61 64
0078 67 6A 6D 70                 FCB     103,106,109,112,115,118,121,124
007C 73 76 79 7C
0080 80 83 86 89                 FCB     128,131,134,137,140,143,146,149
0084 8C 8F 92 95
0088 98 9B 9E A2                 FCB     152,155,158,162,165,167,170,173
008C A5 A7 AA AD
0090 B0 B3 B6 B9                 FCB     176,179,182,185,188,190,193,196
0094 BC BE C1 C4
0098 C6 C9 CB CE                 FCB     198,201,203,206,208,211,213,215
009C D0 D3 D5 D7
00A0 DA DC DE E0                 FCB     218,220,222,224,226,228,230,232
00A4 E2 E4 E6 E8
00A8 EA EB ED EE                 FCB     234,235,237,238,240,241,243,244
00AC F0 F1 F3 F4
```

```
 00B0 F5 F6 F8 F9           FCB    245,246,248,249,250,250,251,252
 00B4 FA FA FB FC
 00B8 FD FD FE FE           FCB    253,253,254,254,254,255,255,255
 00BC FE FF FF FF
 00C0 FF FF FF FF           FCB    255,255,255,255,254,254,254,253
 00C4 FE FE FE FD
 00C8 FD FC FB FA           FCB    253,252,251,250,250,249,248,246
 00CC FA F9 F8 F6
 00D0 F5 F4 F3 F1           FCB    245,244,243,241,240,238,237,235
 00D4 F0 EE ED EB
 00D8 EA E8 E6 E4           FCB    234,232,230,228,226,224,222,220
 00DC E2 E0 DE DC
 00E0 DA D7 D5 D3           FCB    218,215,213,211,208,206,203,201
 00E4 D0 CE CB C9
 00E8 C6 C4 C1 BE           FCB    198,196,193,190,188,185,182,179
 00EC BC B9 B6 B3
 00F0 B0 AD AA A7           FCB    176,173,170,167,165,162,158,155
 00F4 A5 A2 9E 9B
 00F8 98 95 92 8F           FCB    152,149,146,143,140,137,134,131
 00FC 8C 89 86 83
                      *
                      *** COSINE MINUS 90 TABLE ***
                      *
                      * X AXIS - 1 CYCLE HAS 256 SAMPLES
                      * Y AXIS - 1 CYCLE HAS 256 QUANTISED STEPS
                      *
 0100 80 83 86 89           FCB    128,131,134,137,140,143,146,149
 0104 8C 8F 92 95
 0108 98 9B 9E A2           FCB    152,155,158,162,165,167,170,173
 010C A5 A7 AA AD
 0110 B0 B3 B6 B9           FCB    176,179,182,185,188,190,193,196
 0114 BC BE C1 C4
 0118 C6 C9 CB CE           FCB    198,201,203,206,208,211,213,215
 011C D0 D3 D5 D7
 0120 DA DC DE E0           FCB    218,220,222,224,226,228,230,232
 0124 E2 E4 E6 E8
 0128 EA EB ED EE           FCB    234,235,237,238,240,241,243,244
 012C F0 F1 F3 F4
 0130 F5 F6 F8 F9           FCB    245,246,248,249,250,250,251,252
 0134 FA FA FB FC
 0138 FD FD FE FE           FCB    253,253,254,254,254,255,255,255
 013C FE FF FF FF
 0140 FF FF FF FF           FCB    255,255,255,255,254,254,254,253
 0144 FE FE FE FD
 0148 FD FC FB FA           FCB    253,252,251,250,250,249,248,246
 014C FA F9 F8 F6
 0150 F5 F4 F3 F1           FCB    245,244,243,241,240,238,237,235
 0154 F0 EE ED EB
 0158 EA E8 E6 E4           FCB    234,232,230,228,226,224,222,220
 015C E2 E0 DE DC
 0160 DA D7 D5 D3           FCB    218,215,213,211,208,206,203.201
 0164 D0 CE CB C9
 0168 C6 C4 C1 BE           FCB    198,196,193,190,188,185,182,179
```

```
  016C BC B9 B6 B3
  0170 B0 AD AA A7            FCB     176,173,170,167,165,162,158,155
  0174 A5 A2 9E 9B
  0178 98 95 92 8F            FCB     152,149,146,143,140,137,134,131
  017C 8C 89 86 83
  0180 80 7C 79 76            FCB     128,124,121,118,115,112,109,106
  0184 73 70 6D 6A
  0188 67 64 61 5D            FCB     103,100,97,93,90,88,85,82
  018C 5A 58 55 52
  0190 4F 4C 49 46            FCB     79,76,73,70,67,65,62,59
  0194 43 41 3E 3B
  0198 39 36 34 31            FCB     57,54,52,49,47,44,42,40
  019C 2F 2C 2A 28
  01A0 25 23 21 1F            FCB     37,35,33,31,29,27,25,23
  01A4 1D 1B 19 17
  01A8 15 14 12 11            FCB     21,20,18,17,15,14,12,11
  01AC 0F 0E 0C 0B
  01B0 0A 09 07 06            FCB     10,9,7,6,5,5,4,3
  01B4 05 05 04 03
  01B8 02 02 01 01            FCB     2,2,1,1,1,0,0,0
  01BC 01 00 00 00
  01C0 00 00 00 00            FCB     0,0,0,0,1,1,1,2
  01C4 01 01 01 02
  01C8 02 03 04 05            FCB     2,3,4,5,5,6,7,9
  01CC 05 06 07 09
  01D0 0A 0B 0C 0E            FCB     10,11,12,14,15,17,18,20
  01D4 0F 11 12 14
  01D8 15 17 19 1B            FCB     21,23,25,27,29,31,33,35
  01DC 1D 1F 21 23
  01E0 25 28 2A 2C            FCB     37,40,42,44,47,49,52,54
  01E4 2F 31 34 36
  01E8 39 3B 3E 41            FCB     57,59,62,65,67,70,73,76
  01EC 43 46 49 4C
  01F0 4F 52 55 58            FCB     79,82,85,88,90,93,97,100
  01F4 5A 5D 61 64
  01F8 67 6A 6D 70            FCB     103,106,109,112,115,118,121,124
  01FC 73 76 79 7C
                              END

0 ERROR(S) DETECTED


ANTARCTIC RESEARCH ( RHODES )          10-12-82 TSC ASSEMBLER    PAGE    4
PROM DATA


SYMBOL TABLE:



+++
```

Circuit Description

An 8 bit synchronous binary counter consisting of U2 and U3 is clocked at 256 times the required output frequency. The counter outputs address EPROM U7 which contains the reference cosine wave data. The same addresses to EPROM U6 select either cosine wave data that leads the reference by 90^0 or lags the reference by 90^0. The LSB/$\overline{\text{USB}}$ signal to U6 pin 23 selects 90^0 lead data if 0 and 90^0 lag data if at a logic 1.

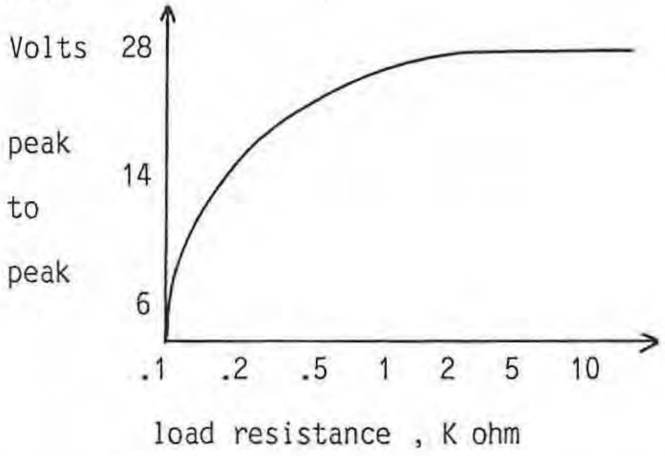


The EPROM addresses change on the rising edge of the clock. The clock signal also drives the EPROM $\overline{\text{OE}}$ (output enable) which means that when the clock signal is high the EPROM data outputs are in a high impedance state. When the clock goes low the addressed data appears on the output bus. Latches U8 , U4 , U5 and U1 latch this data on the clock rising edge.

Latches U8 and U4 drive D to A converter U10 which , together with operational amplifier U11A , is connected for symmetrical offset binary operation. It is important that the two 2K7 resistors be within .05 % of each other for the output to be symmetrical about ground.

VR2 allows adjustment of the output of U11A so that U11B is not overdriven. U11B is used as a buffer/driver into a 50 ohm load. The output voltage swing of U11B is proportional to its load impedance.

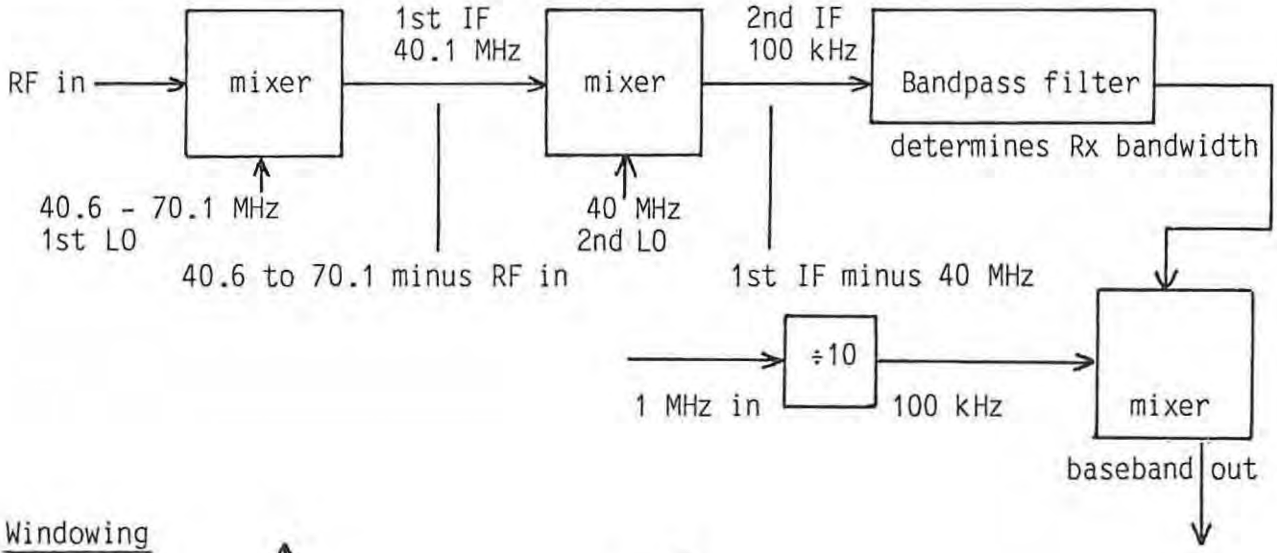
The graph below shows the peak to peak voltage swing variation for differing load resistances.



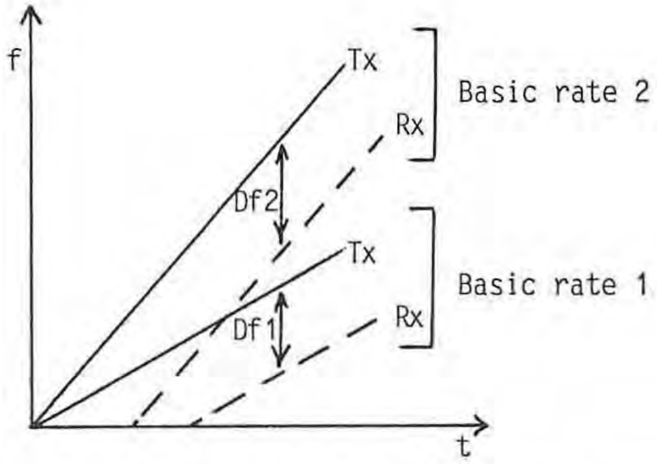
With a 50 ohm resistor in series with a mixer having a 50 ohm input impedance , that is a total load of 100 ohm , the output voltage swing is about 6 V peak to peak before clipping begins. The mixers used operate below this limiting value.

40 MHz Offsetting for Doppler and Windowing

Receiver Block Diagram



Windowing



At higher basic rates the frequency difference between the instantaneous Tx frequency and the received signal is higher. The Rx bandwidth is limited to 500 Hz. To be able to receive signals which would normally be outside the Rx bandwidth the frequency difference Df must reduced by a known amount.

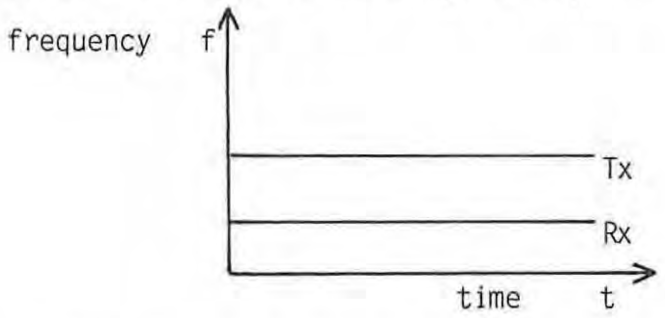
The RF in is mixed with a swept 40.6 to 70.1 MHz signal giving a 40.1 MHz first IF. The second IF of 100 kHz is obtained by mixing with 40 MHz.

At higher sweep rates the difference between the received Rf in and the first injection frequency will be higher. To obtain the 100 kHz IF the 40 MHz signal must be increased in frequency to 40 + Df MHz. Windowing therefore requires the USB of 40 MHz modulated by the offset Df.

Doppler

Operation is at a fixed frequency. For reception of a signal the instantaneous received RF must be at a lower frequency than the instantaneous transmitted frequency. This occurs naturally when the reflecting layer moves upward away from the receiver on the ground . Downward motion of the reflecting layer would not be detected.

To detect downward motion as well the ' no motion ' signal must be positioned in the centre of the range of Doppler frequencies of interest.



Normal chirp sounding reception occurs when Tx_{inst} is greater than RF_{in}. That is the 40.6 to 70.1 first local oscillator must be at a higher frequency than (RF_{in} + 40.1) MHz. If both these frequencies are the same the 1 st IF = 40.1 MHz exactly , the 2 nd IF = 100 kHz exactly and the trace appears at 0 Km.

Because the 40.0 MHz LO signal is subtracted from the first IF signal it must be reduced by a fixed frequency to increase the 100 kHz signal by that frequency and so offset the zero position.

For the detection of both upward and downward motion the LSB of 40 MHz modulated by the offset Df is required.

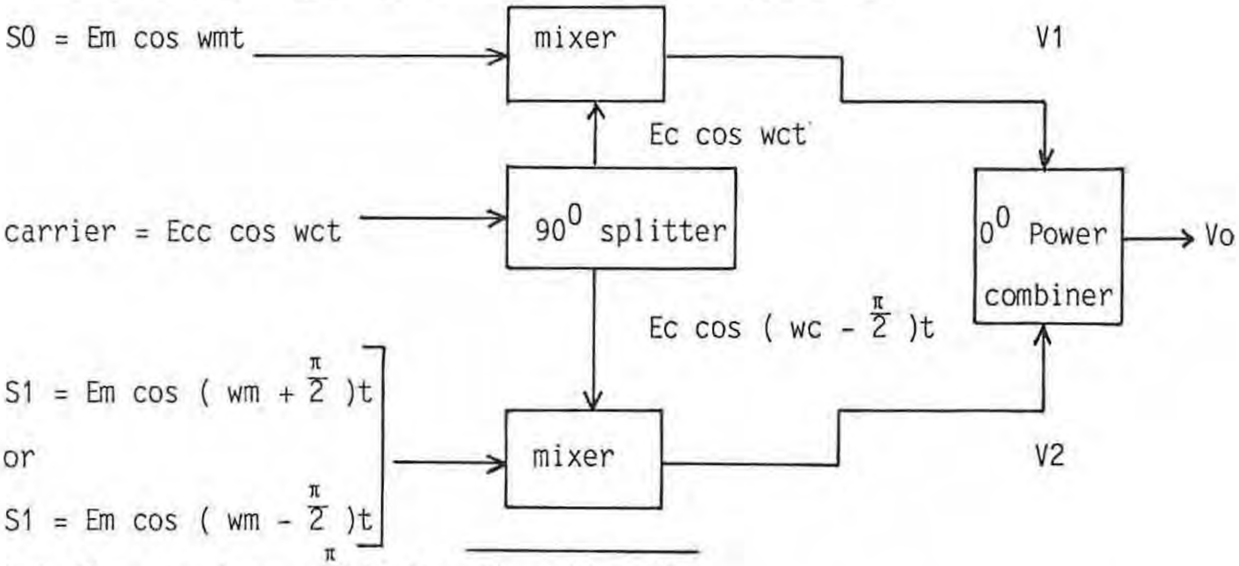
SSB Generator Brass box , B3

Introduction

The SSB Generator is used to offset the 40 MHz local oscillator signal by the frequency generated by the sine wave generator on PC10. The lower sideband (LSB) is used for stationary Doppler reception and the upper sideband (USB) for ionogram windowing. For normal vertical or oblique Rx ionograms the 40 MHz is not shifted.

The SSB generator uses the phase shift technique of generating a single sideband signal. Four passive RF components are used in the circuit.

Circuit Development - Phase Shift SSB Signal Generation



Let $S1 = \cos (wm + \frac{\pi}{2})t$ then $Vo = V1 + V2$

$$Vo = (Em \cos wmt)(Ec \cos wct) + (Em \cos (wm + \frac{\pi}{2})t)(Ec \cos (wc - \frac{\pi}{2})t$$

applying $(\cos A)(\cos B) = 1/2 \cos (A+B) + 1/2 \cos (A-B)$ yeilds :-

$$Vo = EmEc/2 \cos (wc+wm)t + EmEc/2 \cos (wc - wm)t$$

$$+ EmEc/2 \cos (wc-\frac{\pi}{2}+wm+\frac{\pi}{2})t + EmEc/2 \cos (wc-\frac{\pi}{2}-wm-\frac{\pi}{2})t$$

$$= EmEc \cos (wc + wm)t + EmEc/2 \cos (wc-wm)t + EmEc/2 \cos (wc-wm-\pi)t$$

applying $\cos (A-\pi) = -\cos A$ yeilds :-

$Vo = EmEc \cos (wc+wm)t$, which is the upper sideband.

Choosing $S1 = \cos (wm-\frac{\pi}{2})t$ gives the lower sideband.

Circuit Description

The 40 MHz local oscillator signal is split by 90^0 splitter RF2. Its reference output is mixed with the reference sine wave S0 by mixer RF1. Mixer RF4 mixes two phase shifted signals , that from RF2 with S1. The mixer outputs are combined by in-phase power combiner RF3 to give the required single sideband signal with an amplitude of 100 mV peak to peak.

Computer Modifications

1. S-32 Memory Board modifications for use from 32K to 56K

Hex address	Address bus MSB's						
	A15	A14	A13	A12			
8xxx	1	0	0	0	8K	lower 16K	32K
9xxx	1	0	0	1			
Axxx	1	0	1	0	8K		
Bxxx	1	0	1	1			
Cxxx	1	1	0	0	8K	upper 16K	
Dxxx	1	1	0	1			
Exxx	1	1	1	0	8K		
Fxxx	1	1	1	1			

The upper 32K of the address space is selected when A15 = 1.

A14 = 0 selects the lower 16K block.

A14 = 1 selects the upper 16K block.

Hex addresses from E000 to FFFF are asigned to the I/O Ports and the Monitor (S-Bug) and therefore cannot be used for memory. These addresses correspond to the upper 8K of the upper 16K block of memory on the S-32 Memory Board. This board has gating allowing the enabling/disabling of memory in two 16K blocks only. To exclude the upper 8K block, addresses E000 to FFFF, the data bus buffers must be disabled when A15 = A14 = A13 = 1. This is achieved using IC21, a 1 of 8 decoder , which is not used in this particular microprocessor configuration. The changes are shown on the partial circuit diagram in thick black ink.

Circuit Logic

The jumper on the output of IC24 determines the S-32 Memory Board location and is in the 32K to 64K position. The upper 16K is selected when A15 = 1,A14 = 1 and $\overline{E}$ goes low resulting in a low on IC23 pin 8 which is jumpered to IC18 and IC19 by the ' Enable Upper 16K ' jumper.

The circuit change involves gating the signal from IC23 pin 8 with A13 such that when A15 = A14 = A13 = 1 the data bus buffers are disabled. Printed circuit tracks to IC21 pins 1 , 2 ,3 ,5 and 6 are cut and pin 1 is connected to A13 , pin 2 to IC23 pin 8 , pins 3 and 5 to ground and pin 6 to +5V. IC21 output $\overline{0}$ on pin 15 is connected directly to the enables on IC18 and IC19 and the jumper block is omitted. Thus when A13 = 0 the ' upper 16K' data bus buffers are enabled and when A13 = 1 they are disabled. This allows memory up to DFFF and reserves locations E000 to FFFF for I/O and the S-Bug monitor.

2. S-32 Memory Board Battery Backup.

Both S-32 Memory Boards have EPROM and RAM integrated circuits on them. The contents in RAM must be retained during power failures. The +24V battery supply from the 105B Quartz Oscillator is converted to +5VDC and supplies both S-32 boards via the Diode PCB.

The signal Pon from Advance/Retard PC5 drives IC23 pin 12. This sets the memory R/W control line into read mode when the power fails by going from high to low.

6018 Camera Interface/Display Modifications

Linear Variable Oscillator / 500 Hz Oscillator

This board is unplugged when the Controller is used. The Controller provides the variable film rate and 500 Hz fast film advance signals.

PC board A4-V jumpered to A4-W connects A5-F to pin 9 of U3 on A5 when A4 is plugged in (Logchirp operation).

Video Amplifier/Drive Logic , A5

This section describes the changes to PCB A5 for Controller operation.

For an oblique transmission the sounding set up in the Controller must have the film drive set to 'off' to inhibit the variable film rate signal to the 6018. The video signal must also be blanked.

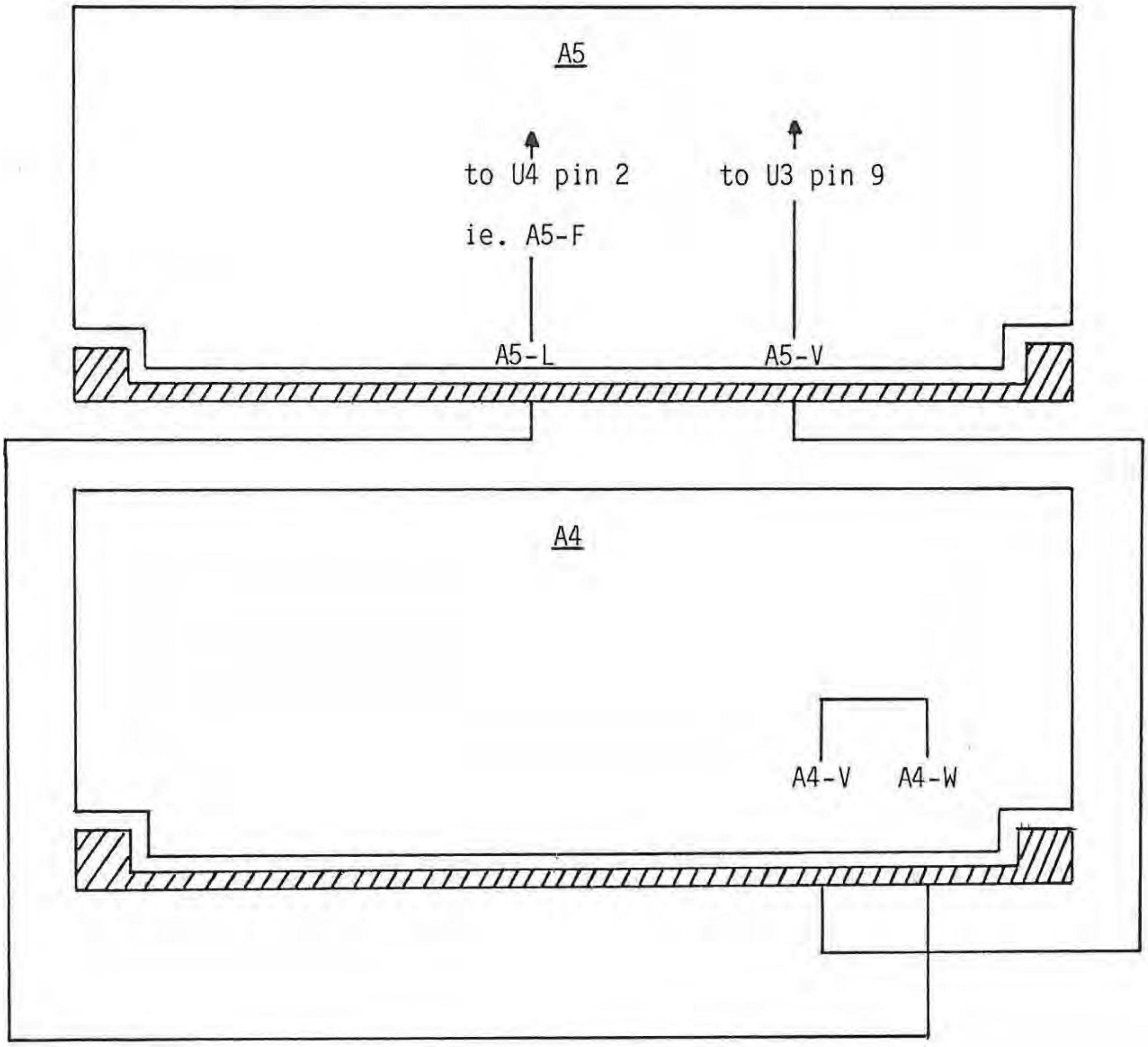
A multi-cell sounding programmed with film drive on for some cells and off for others must have the film drive signal inhibited and the video blanked for the 'off' cells.

Controller board PC3 generates these two signals called 'film drive inhibit' and 'video blank' respectively. 6018 board A5 is modified to utilise the 'video blank' signal as follows.

1. U1 pin 9 connected to U1 pin 10.
2. U1 pin 1 disconnected from U1 pin 2.
3. A5-F connected to U1 pin 2.
4. Track to U3 pin 9 disconnected.
5. U3 pin 9 pulled high by a 4K7 resistor.
6. U3 pin 9 connected to A5-V.
7. A5-F connected to A5-L on PCB A5.
8. Edge connector A5-V to A4-V.
9. Edge connector A5-L to A4-W.
10. A4-W connected to A4-V on PCB A4.
11. 'Video blank' signal from J2-4 to A5-F.

When Logchirp control is used A4 is plugged in connecting A5-L to A5-V so that the 'OV Tx oblique' signal goes to U3 pin 9 on board A5.

Using the Controller requires that A4 be unplugged. This results in U3 pin 9 being set to a logic 1.



System Timing Diagrams

All of the system timing signals are derived from the 100 kHz from the 105B Quartz Oscillator. This 100 kHz signal , called Q0 , undergoes three inversions through the Advance retard circuitry on PC5 , to become $\overline{Q0}$.

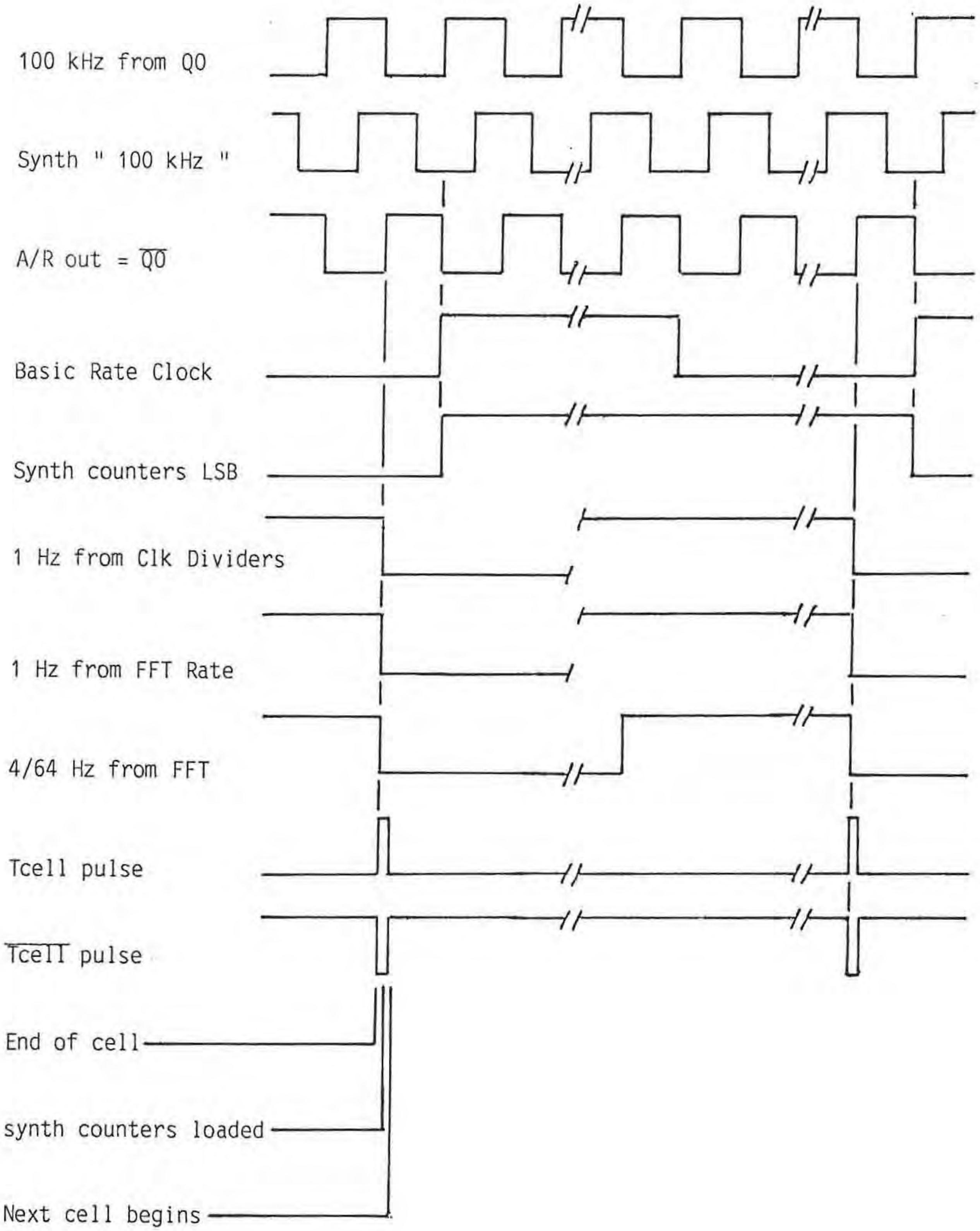
The synthesizer " 100 kHz " is the timing signal used by the synthesizer and is the same as Q0 except for a 90^0 phase shift. Programming data to the synthesizer must be stable for the 3 microseconds after this signal has been high for 1 microsecond.

The counters that drive the synthesizer change on the $\overline{Q0}$ falling edge which occurs 3.5 microseconds before the programming data must be stable.

The 1 Hz from the FFT Rate Generator ,PC7, is synchronised with the 1 Hz from the Clock Dividers on PC3a at the start of each ionogram. The falling edge of the 4/64 Hz signal (either 4 or 64 Hz from the FFT Rate Generator is selected on PC3b) clocks the Tcell dividers which trigger , on the falling edge , the 100 nanosecond Tcell pulse.

This pulse signals the beginning of a cell by interrupting the computer. The inverse of the Tcell pulse , called $\overline{\text{Tcell}}$, transfers latched data to the synthesizer counters , transfers receive antenna select data from the holding latches to the select latches , resets the M-sequence generator and clocks the film drive flip-flop. $\overline{\text{Tcell}}$ is also output to the hardware FFT ' stop sampling ' input via Controller connector RPA4.

System Timing Diagrams



Conversion from Logchirp Control to Vertichirp Controller

1. Switch off AC power to the system.
2. Unplug all the cables from Logchirp Control.

These are:-

   a. AC in
   b. STBY PWR.
   c. .125 - 7.5 MHz from synth.
   d. " 100 kHz " from synth.
   e. 100 kHz from 105B Quartz Oscillator.
   f. 5 MHz from synth.
   g. Freq Prog synthesizer 50 way cable.
   h. Cable to 4050 Amplifier T/R.
   i. Cable to 6018 Camera Interface/Display.

3. Remove board A40 (+24 VDC to -10 VDC Converter) from Logchirp Control.
4. Plug A40 into Controller slot PC12.
5. Remove board A6 from 4050 Amplifier T/R.
6. Plug the Controller T/R Gain Weight Generator board into the A6 slot in the 4050 Amplifier T/R.
7. Remove board A4 from the 6018 Camera Interface/Display.

   NB: Slot A4 is left unoccupied when the 6018 is used with the Controller.
8. Plug the following cables into the Controller rear panel:-

   a. AC in.
   b. STBY PWR - Check that front panel " Battery " LED illuminates.
   c. 100 kHz from 105B Quartz Oscillator (QO) to BNC8.

      NB: 100 kHz from <u>Quartz Oscillator</u>
   d. 1 MHz from 105B QO to BNC6.
   e. Freq Prog Synthesizer - the cable from the synthesizer.

f. Special cable between Controller and 4050 Amplifier T/R.

g. Special cable between Controller and 6018 Camera Interface/Display.

h. Cable between Controller and Rx Antenna Switch , B1.

i. Cable between Controller and Data Capture System , DCS.

j. Cable between Controller and Tx Antenna select , B2a.

The other cable from B2a goes to the T500A Linear Amplifier.

k. VDU Pwr.

l. Cable between Controller and VDU.

NB: Check that the toggle switch mounted under the VDU is switched to the " Controller " position.

9. RF Signal Paths - Tx.

A. Controller using 8W to an antenna which is also used for Rx.

a. 4050 " TO ANT " to antenna.

b. 4050 " RF TO REC " to the 4040 Rx.

B. Controller using T500A Linear Amplifier.

Note:- Separate Tx and Rx antennae must be used.

a. 4050 " TO ANT " to T500A " RF in ".

b. T500A " TO ANT " to Antenna Select Box , B2a.

This box connects the transmitter to either the vertical Tx antenna or to the oblique antenna. If the T500A is tripped or switched off it is automatically bypassed.

10. RF Signal Paths - Rx.

A. Controller using 4040 Rx.

a. 1 MHz from 4041 LO to 4040 Rx.

b. 40 MHz from 4041 LO to 4040 Rx.

c. 40.6 - 70.1 from 4041 LO to 4040 Rx.

d. Gain wt from 4050 Amplifier T/R to 4040 Rx.

- e. 4040 Rx RF in:-
  - i. If connected as in 6A RF in from 4050 "RF TO REC ".
  - ii. If connected as in 6B RF in from Antenna Switch , B1
    or from 4040 Rx Antenna select , B2b
    or directly in from a vertical Rx antenna.
- f. 4040 Rx baseband out to 2002C Spectrum Analyser video input.

B. Controller using dual Rx.

- a. 1 MHz from 4041 LO to dual Rx 1 MHz in.
- b. 40 MHz from 4041 LO to dual Rx 2nd LO in.
- c. 40.6 - 70.1 MHz to 10 dB RF amplifier.
- d. 10 dB amplifier output to dual Rx 1st LO in.
- e. Gain wt from 4050 Amplifier T/R to dual Rx Gain wt in.
- f. Dual Rx RF in from Antenna Switch , B1.
- g. Either RX1 or Rx2 baseband output to 2002C Spectrum Analyser video input via the attenuating series resistor for the film record.
- h. Rx1 and Rx2 baseband outputs to FFT inputs.

C. Controller using dual Rx and 4040 Rx.

- a. 1 MHz from 4041 LO to 4040 Rx.
- b. 40 MHz from 4041 LO to power splitter PS1.
- c. Power splitter outputs to 4040 Rx and dual Rx.
- d. 40.6 - 70.1 MHz to 10dB RF amplifier.
- e. 10 dB RF amp output to power splitter PS2.
- f. Power splitter outputs to 4040 Rx and dual Rx.
- g. Gain wt from 4050 Amp T/R to 4040 Rx and dual Rx.
- h. 4040 Rx baseband out to 2002C Spectrum Analyser video - film record.
- i. Dual Rx outputs from Rx1 , Rx2 to FFT.

APPENDIX B

VERTICHIRP CONTROLLER HARDWARE - MANUAL 2

CIRCUIT DIAGRAMS , COMPONENT LOCATION DIAGRAMS AND WIRE LISTS

G. P. EVANS

ANTARCTIC RESEARCH - RHODES UNIVERSITY

DECEMBER 1983

Vertichirp Controller Hardware - Manual 2

Circuit Diagrams , Component Location Diagrams and Wire Lists

Documentation Notes

1. Block Diagrams

Single signal lines are drawn as :-

Multiple signal lines are drawn as :-

2. All edge connector numbers are typed in a rectangular box on the circuit diagrams. Component side connectors are prefixed " T " (Top) and solder side connectors are prefixed " B " (Bottom). Connector numbering of both sides is from left to right the board being viewed from the component side. Signal names are typed alongside each box.

Input signal :- synth counter clock $\boxed{\text{T6}}$———

Output signal :- ———$\boxed{\text{T22}}$ Sweep

3. The following prefixes are used in component numbering :-

U - Integrated Circuit
R - Resistor
Q - Transistor
C - Capacitor
D - Diode
J - Jumper

4. Components are numbered in horizontal rows from left to right with numbering beginning in the top left corner on the component side of the printed circuit board.

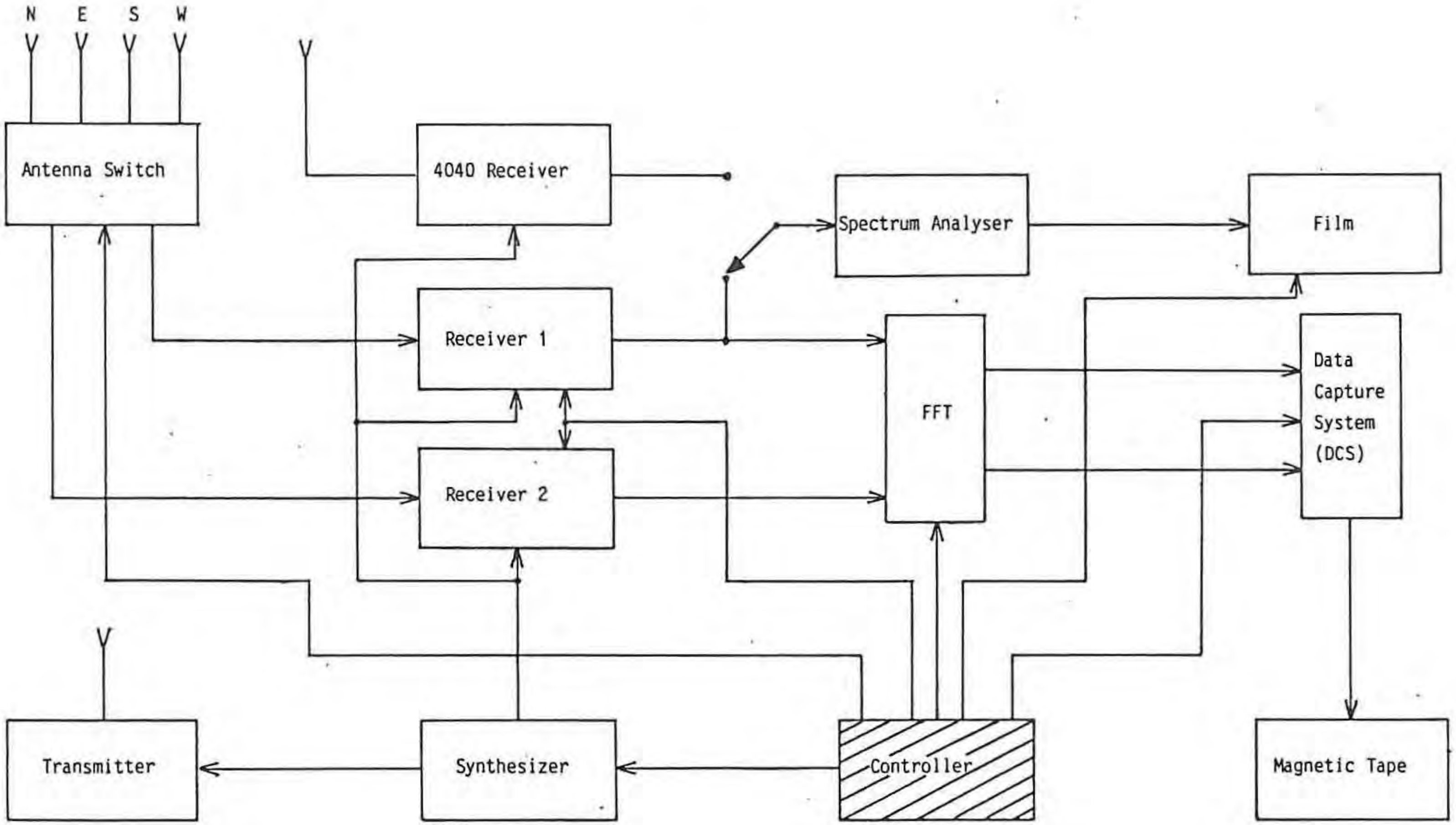
5. In the wire lists the direction arrow has the following meaning :-

Arrow to the left - input to the board being described.

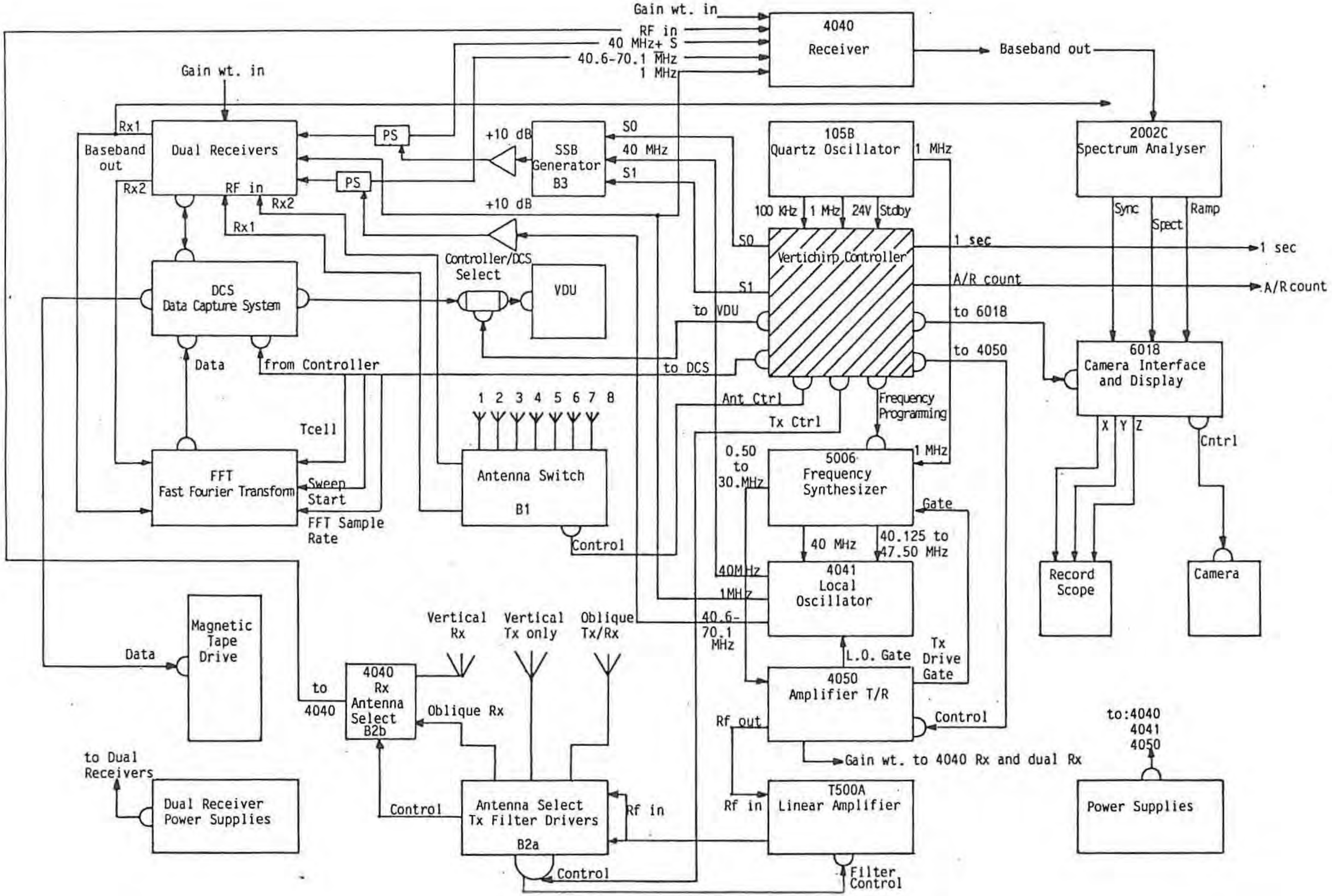
Arrow to the right - output from the board being described.

6. Example of signal tracing :-

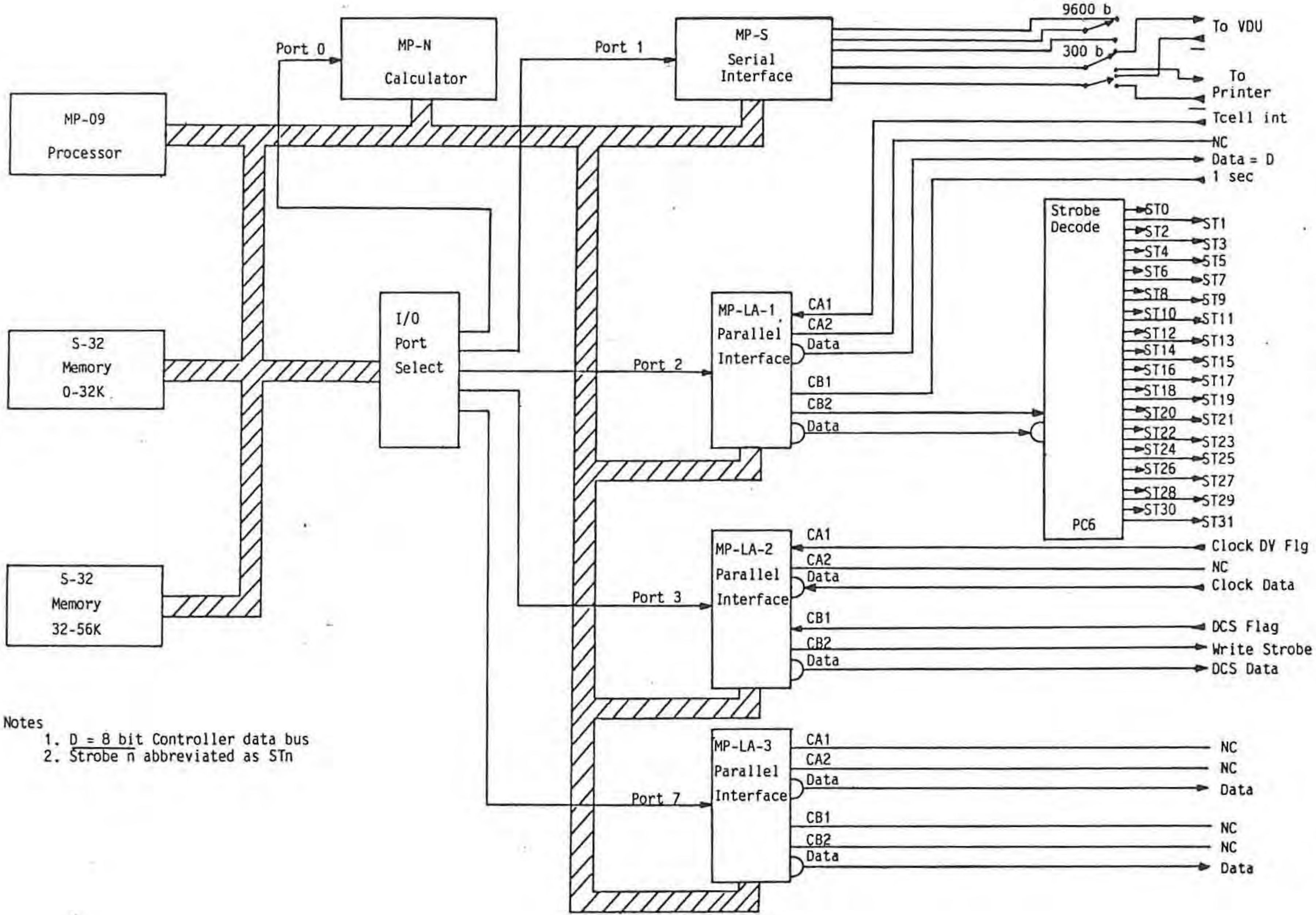
PC1 circuit diagram shows " synth counter clock " signal on T6 going to U2 pin 5. The component location diagram shows the position of U2 on the printed circuit board. The wire list shows that the signal is an input to PC1 and that it comes from PC3 connector T21.



Simplified System Block Diagram

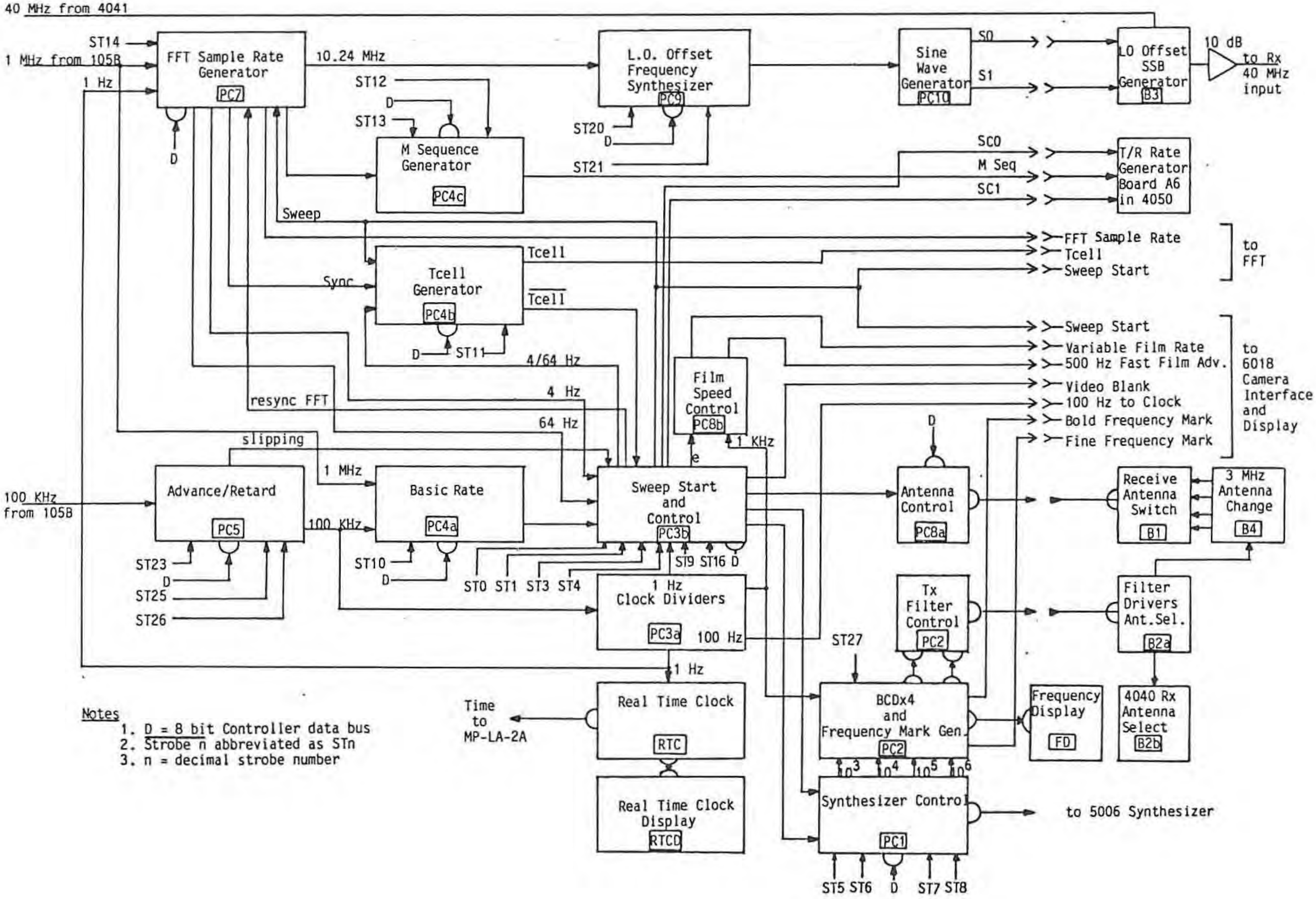


Sounder System Block Diagram



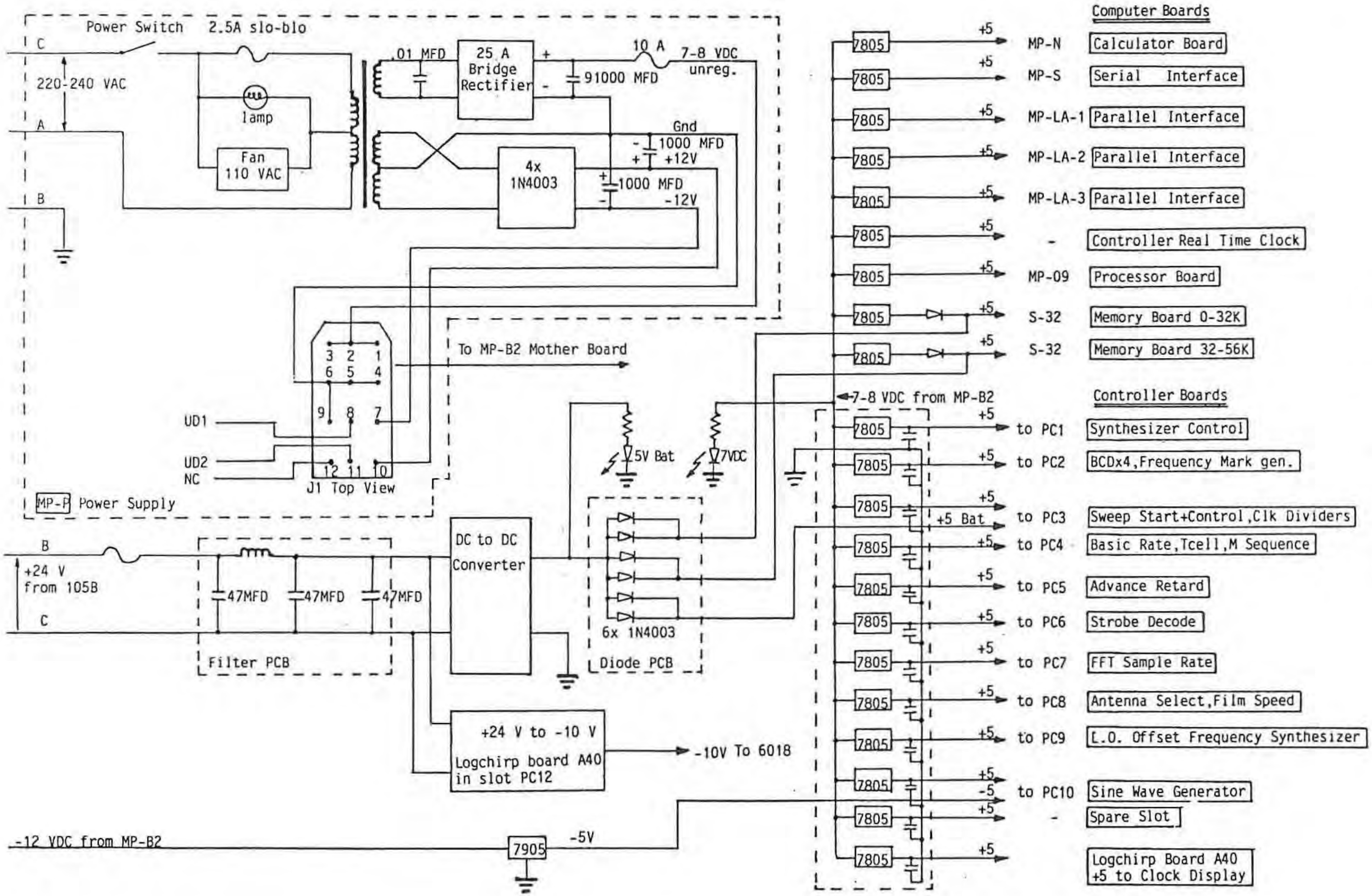
Notes
1. $\overline{D}$ = 8 bit Controller data bus
2. Strobe $\overline{n}$ abbreviated as STn

Vertichirp Controller Block Diagram 1



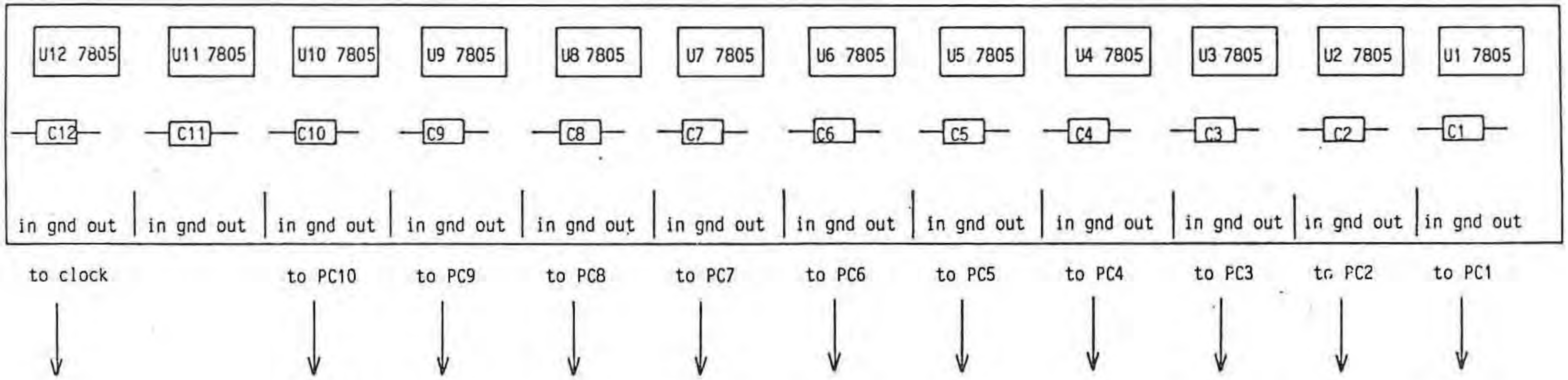
Notes
1. D = 8 bit Controller data bus
2. Strobe n abbreviated as STn
3. n = decimal strobe number

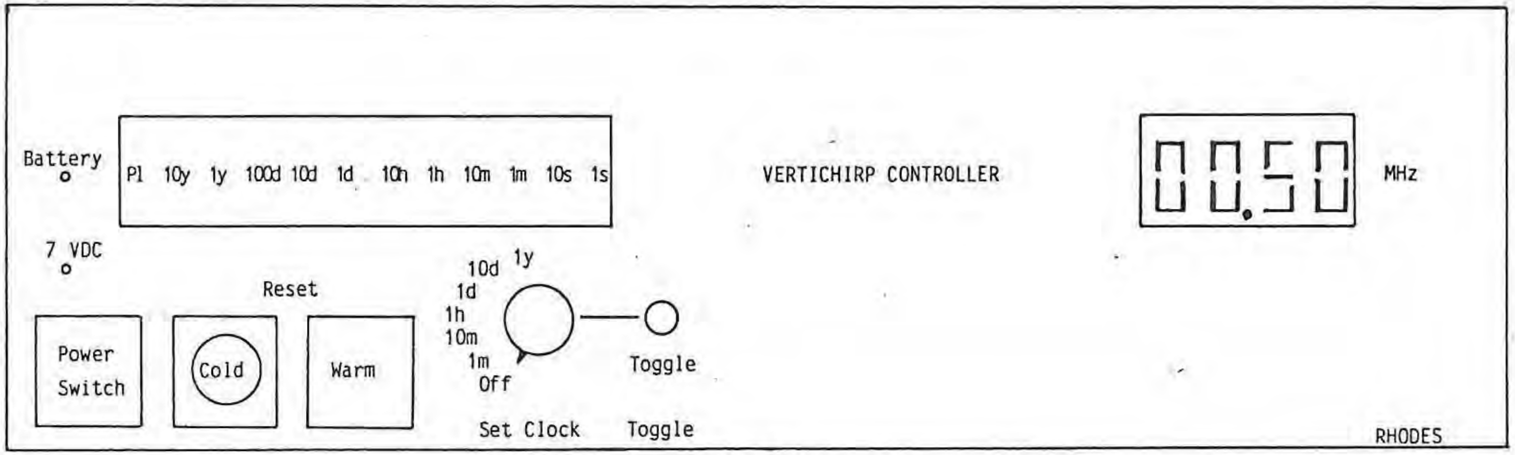
Vertichirp Controller Block Diagram 2



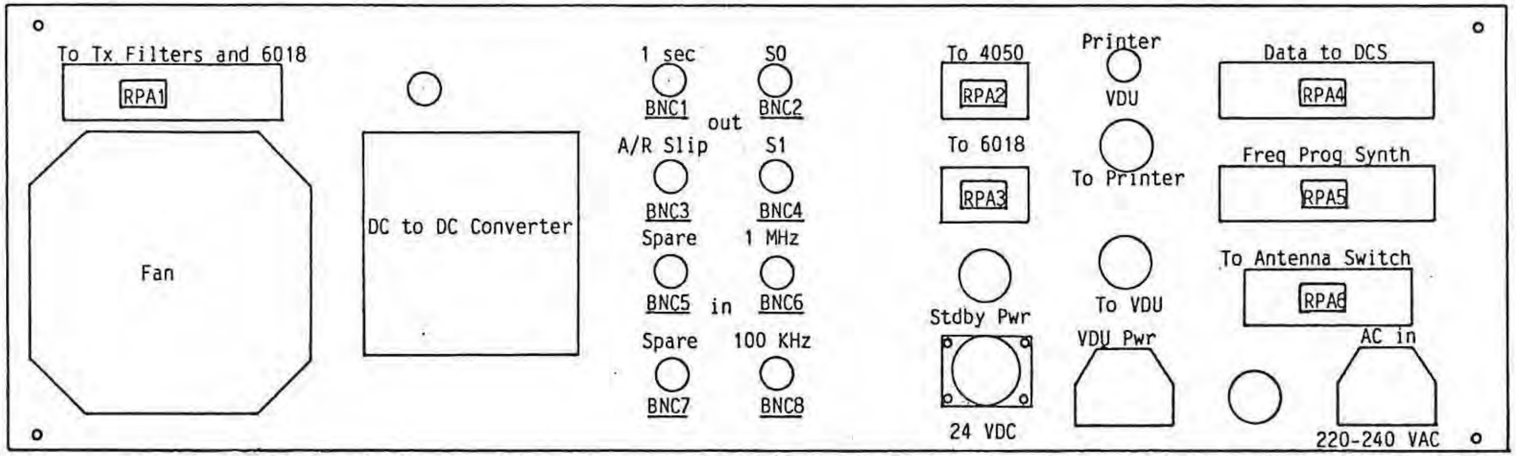
Vertichirp Controller Power Supplies

Controller Hardware 5VDC Power Supplies





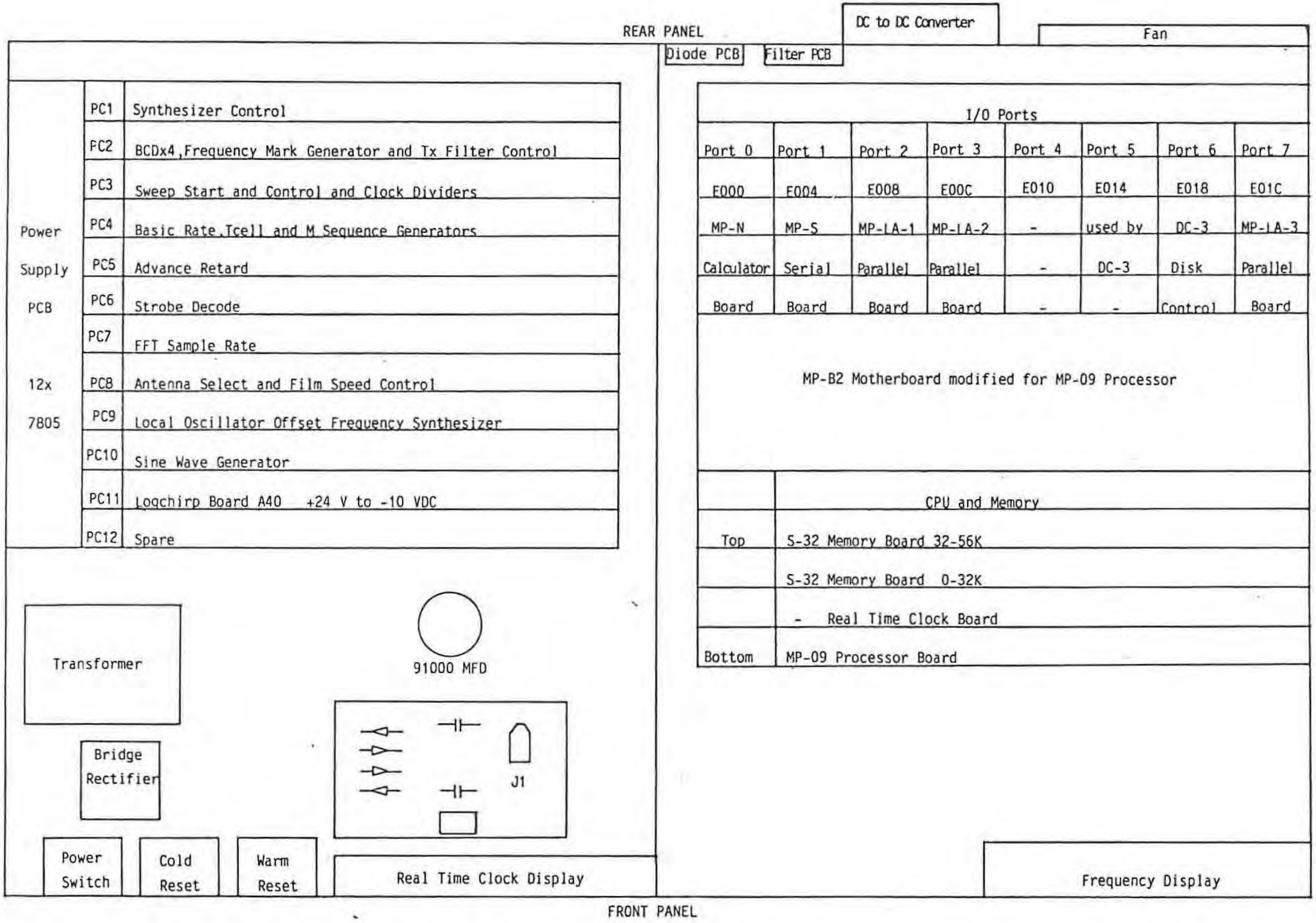
FRONT PANEL



REAR PANEL

RPAn = Rear Panel Amphenol n

Vertichirp Controller Front and Rear Panels



PC1	Synthesizer Control
PC2	BCDx4,Frequency Mark Generator and Tx Filter Control
PC3	Sweep Start and Control and Clock Dividers
PC4	Basic Rate,Tcell and M Sequence Generators
PC5	Advance Retard
PC6	Strobe Decode
PC7	FFT Sample Rate
PC8	Antenna Select and Film Speed Control
PC9	Local Oscillator Offset Frequency Synthesizer
PC10	Sine Wave Generator
PC11	Logchirp Board A40 +24 V to -10 VDC
PC12	Spare

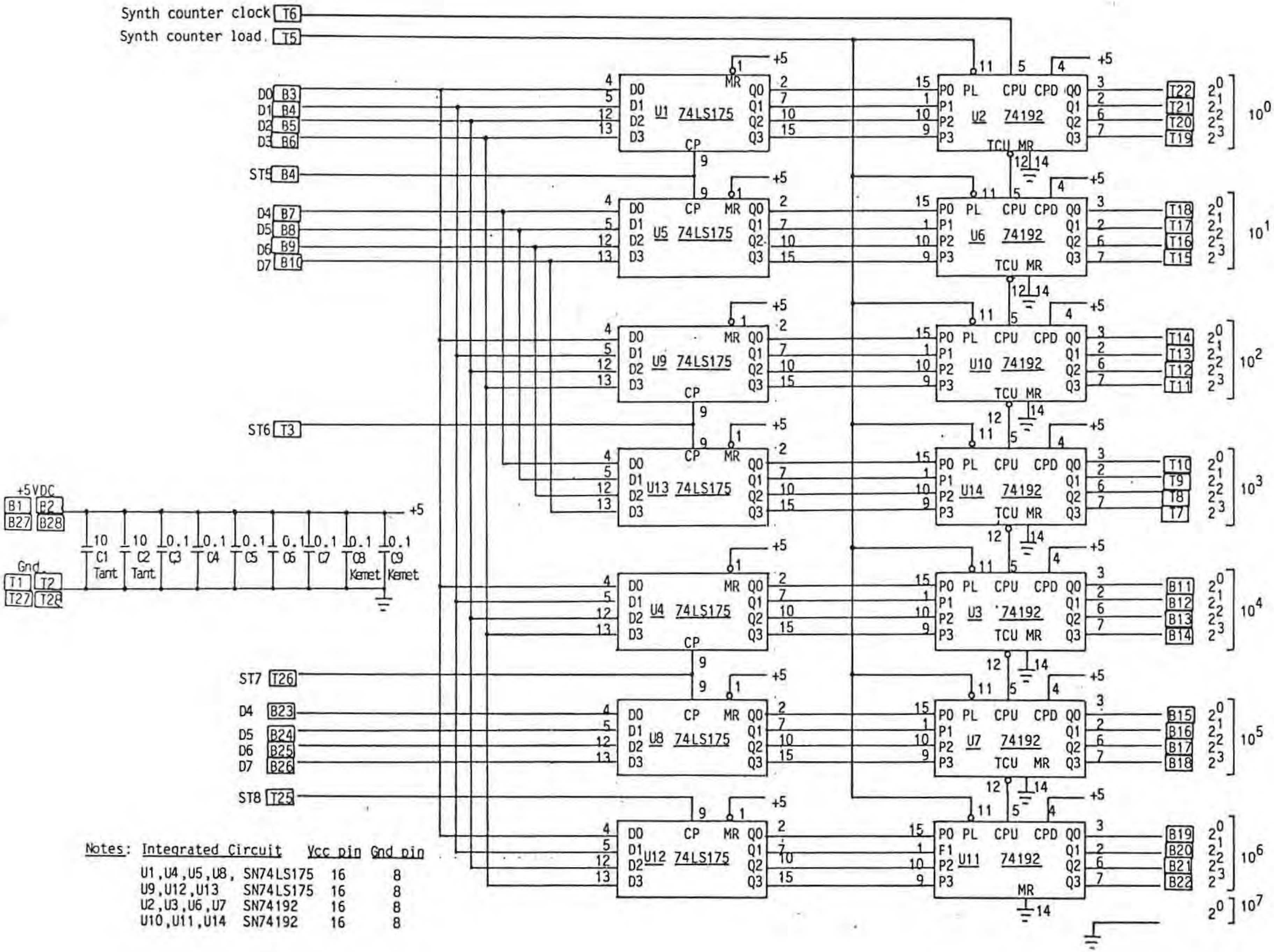
I/O Ports

Port 0	Port 1	Port 2	Port 3	Port 4	Port 5	Port 6	Port 7
E000	E004	E008	E00C	E010	E014	E018	E01C
MP-N	MP-S	MP-LA-1	MP-LA-2	-	used by	DC-3	MP-LA-3
Calculator	Serial	Parallel	Parallel	-	DC-3	Disk	Parallel
Board	Board	Board	Board	-	-	Control	Board

	CPU and Memory
Top	S-32 Memory Board 32-56K
	S-32 Memory Board 0-32K
	- Real Time Clock Board
Bottom	MP-09 Processor Board

Location of Vertichirp Controller Circuit Boards

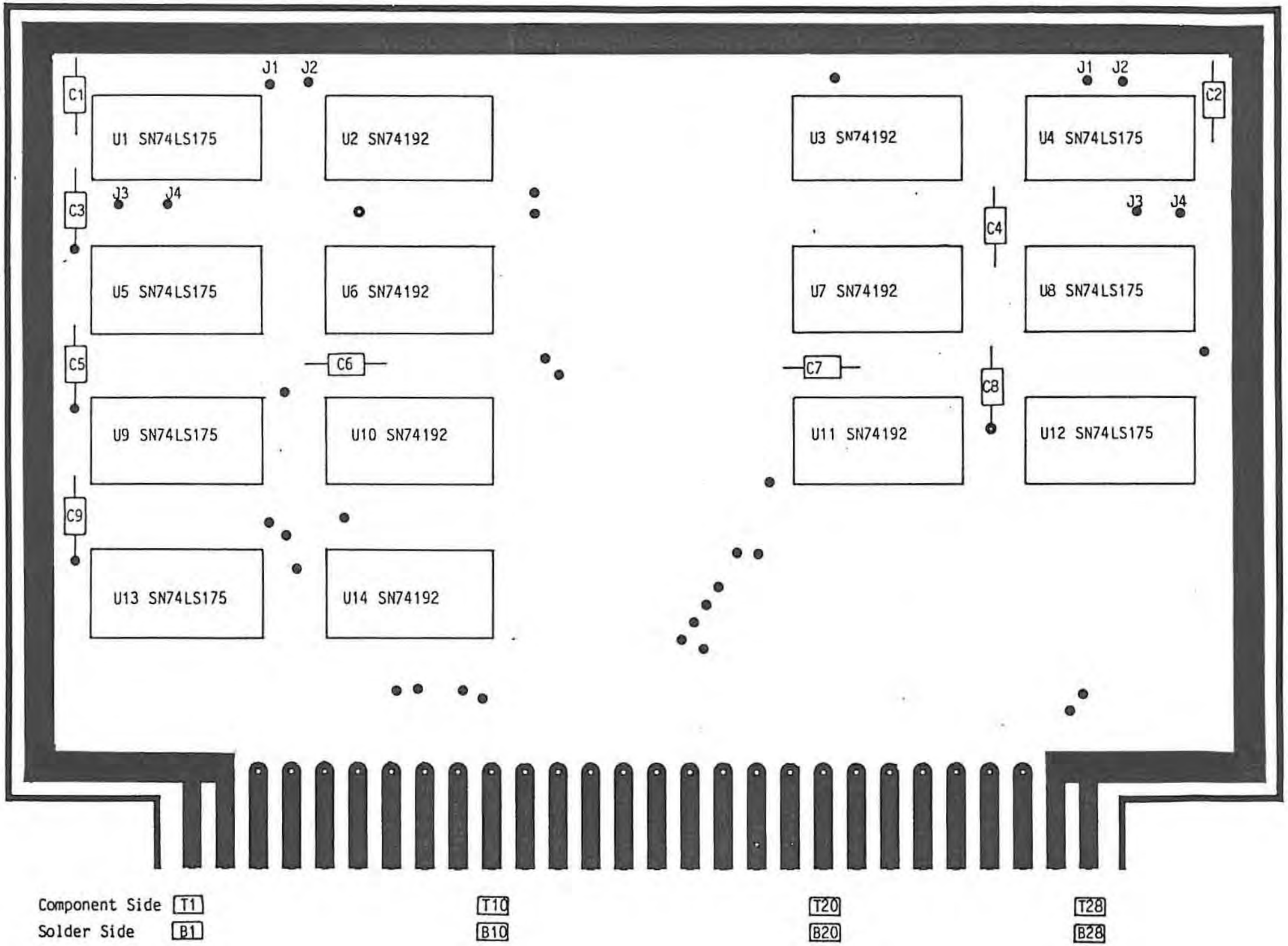
Synthesizer Control , PC1



Notes:

Integrated Circuit		Vcc pin	Gnd pin
U1,U4,U5,U8,	SN74LS175	16	8
U9,U12,U13	SN74LS175	16	8
U2,U3,U6,U7	SN74192	16	8
U10,U11,U14	SN74192	16	8

Synthesizer Control , PC1



Notes:

● = Feedthrough

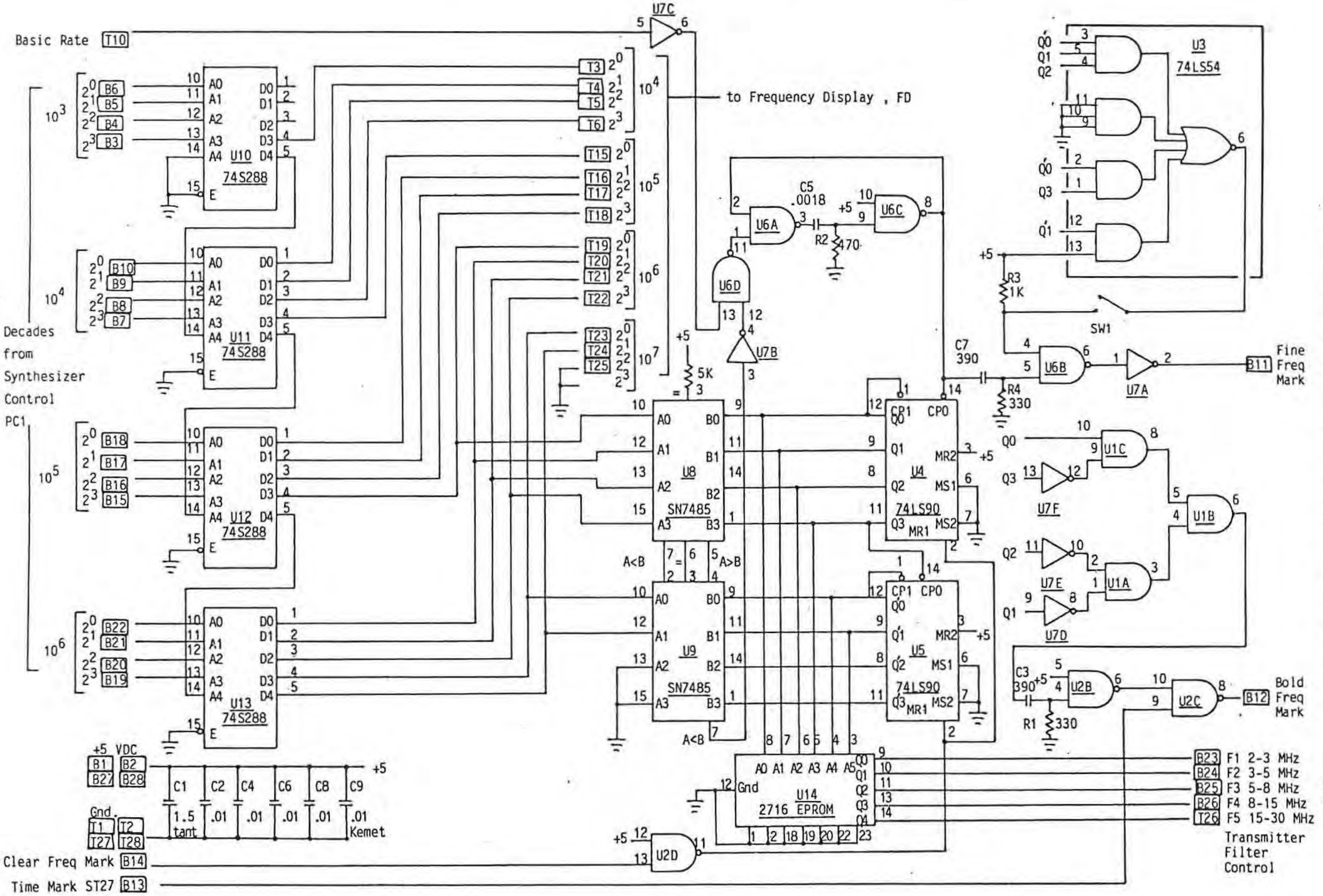
Jn Jn = Jumper n

Synthesizer Control , PC1 - Component Side (T)

Pin No.	Signal Name and Function	Direction	From/to	Pin No.
T1,T2	gnd.	<--	chassis	-
T3	ST6 latch 10^2 and 10^3 decades	<--	PC6	T4
T4	ST5 latch 10^0 and 10^1 decades	<--	PC6	T5
T5	synth counter load	<--	PC3	T18
T6	synth counter clock	<--	PC3	T21
T7	2^3 10^3 out	--> -->	RPA5 PC2	33 B3
T8	2^2	--> -->	RPA5 PC2	32 B4
T9	2^1	--> -->	RPA5 PC2	8 B5
T10	2^0	--> -->	RPA5 PC2	7 B6
T11	2^3 10^2 out	-->	RPA5	31
T12	2^2	-->	RPA5	30
T13	2^1	-->	RPA5	6
T14	2^0	-->	RPA5	5
T15	2^3 10^1 out	-->	RPA5	29
T16	2^2	-->	RPA5	28
T17	2^1	-->	RPA5	4
T18	2^0	-->	RPA5	3
T19	2^3 10^0 out	-->	RPA5	27
T20	2^2	-->	RPA5	26
T21	2^1	-->	RPA5	2
T22	2^0	-->	RPA5	1
T23	-	-	-	-
T24	-	-	-	-
T25	ST8 latch 10^6 decade	<--	PC6	B3
T26	ST7 latch 10^4 and 10^5 decades	<--	PC6	T3
T27,T28	gnd	<--	chassis	-

Synthesizer Control , PC1 - Solder Side (B)

Pin No.	Signal Name and Function	Direction	From/to	Pin No.
B1,B2	+5VDC supply	<--	chassis	-
B3	D0	<--	MP-LA-1A	A0
B4	D1	<--	MP-LA-1A	A1
B5	D2 Controller	<--	MP-LA-1A	A2
B6	D3 hardware	<--	MP-LA-1A	A3
B7	D4 data	<--	MP-LA-1A	A4
B8	D5 bus	<--	MP-LA-1A	A5
B9	D6	<--	MP-LA-1A	A6
B10	D7	<--	MP-LA-1A	A7
B11	2^0 10^4 out	--> -->	RPA5 PC2	9 B10
B12	2^1	--> -->	RPA5 PC2	8 B9
B13	2^2	--> -->	RPA5 PC2	34 B8
B14	2^3	--> -->	RPA5 PC2	33 B7
B15	2^0 10^5 out	--> -->	RPA5 PC2	11 B18
B16	2^1	--> -->	RPA5 PC2	12 B17
B17	2^2	--> -->	RPA5 PC2	36 B16
B18	2^3	--> -->	RPA5 PC2	37 B15
B19	2^0 10^6 out	--> -->	RPA5 PC2	13 B22
B20	2^1	--> -->	RPA5 PC2	14 B21
B21	2^2	--> -->	RPA5 PC2	38 B20
B22	2^3	--> -->	RPA5 PC2	39 B19
B23	D4 Controller	<--	MP-LA-1A	A4
B24	D5 hardware	<--	MP-LA-1A	A5
B25	D6 data	<--	MP-LA-1A	A6
B26	D7 bus	<--	MP-LA-1A	A7
B27,B28	+5VDC supply	<--	chassis	-



BCDx4, Frequency Mark Generator and Tx Filter Control, PC2

BCD x 4 Frequency Mark Generator and Tx Filter Control , PC2

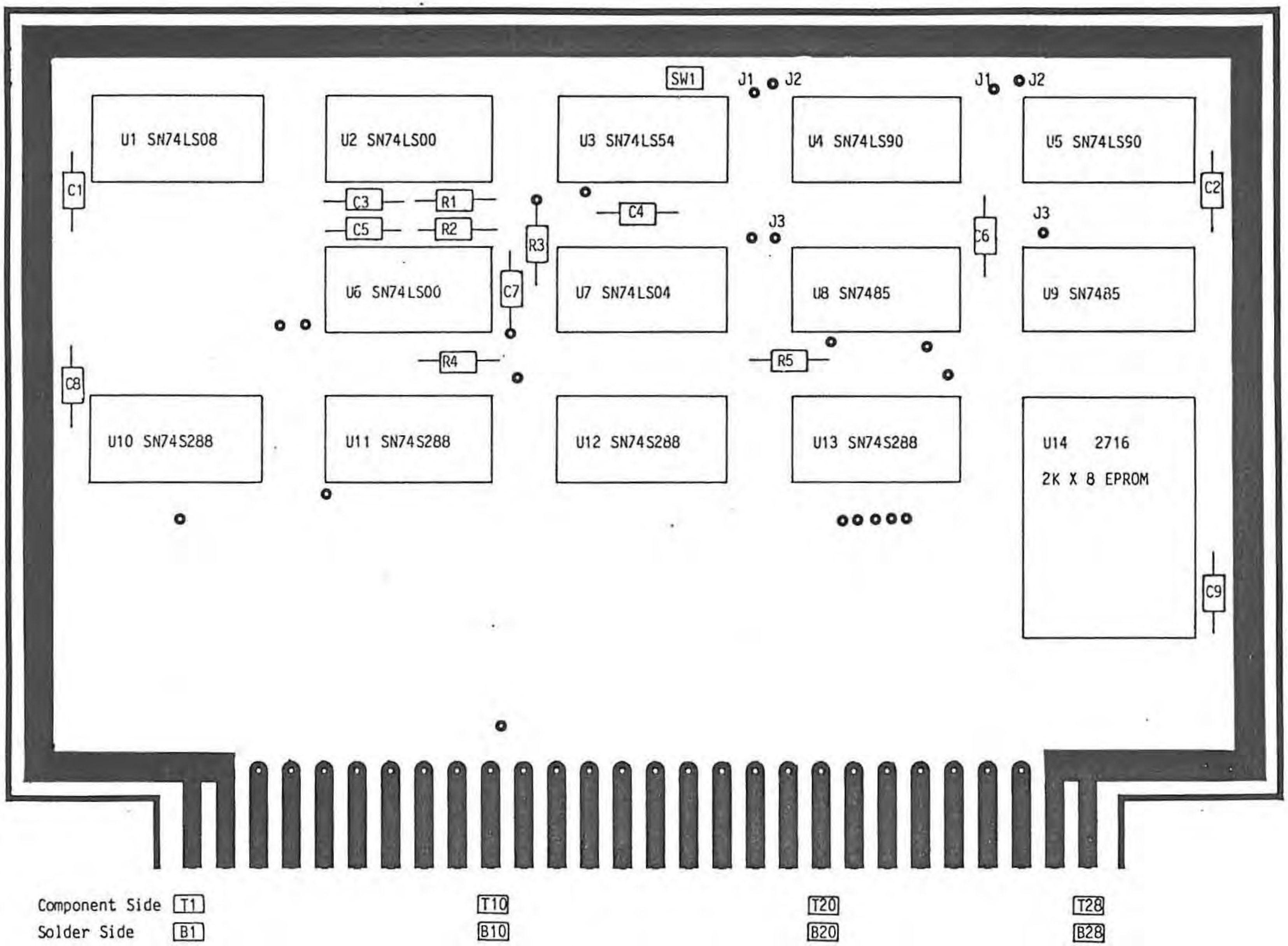
Notes:

Integrated Circuit		Vcc pin	Gnd. pin
U1	SN74LS08	14	7
U2 , U6	SN74LS00	14	7
U3	SN74LS54	14	7
U4 , U5	SN74LS90	5	10
U7	SN74LS04	14	7
U8 , U9	SN7485	16	8
U10 , U11 , U12 , U13	SN74S288	16	8
U14	2716 EPROM	24	12

All resistors in ohms.

All capacitors in microfarads unless otherwise stated.

BCDx4 Frequency Mark Generator and Tx Filter Control PC2



Notes:
● = Feedthrough
Jn Jn = Jumper n

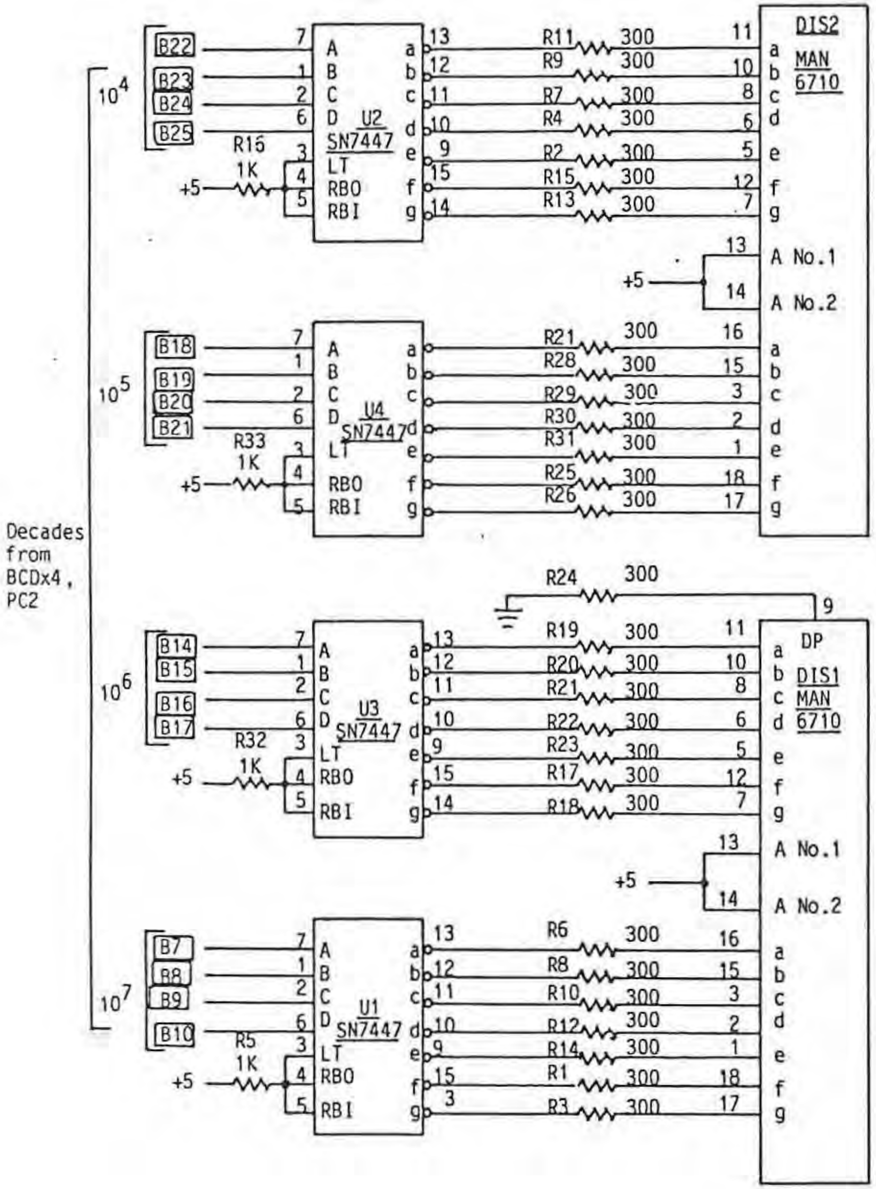
BCDx4,Frequency Mark Generator & Tx filter Control PC2 - Component Side(T)

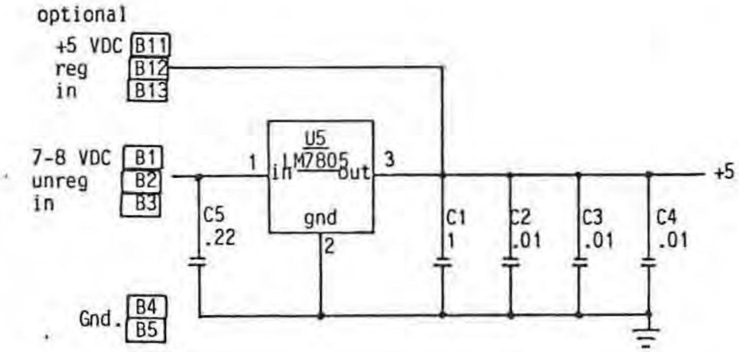
Pin No.	Signal Name and Function	Direction	From/to	Pin No.
T1,T2	gnd.	<--	chassis	-
T3	2^0 10^4 BCDx4 data	-->	FD	B22
T4	2^1	-->	FD	B23
T5	2^2	-->	FD	B24
T6	2^3	-->	FD	B25
T7	-	-	-	-
T8	-	-	-	-
T9	-	-	-	-
T10	Basic Rate Clk. to freq mark gen	<--	PC4	B13
T11	-	-	-	-
T12	-	-	-	-
T13	-	-	-	-
T14	-	-	-	-
T15	2^0 10^5 BCDx4 data	-->	FD	B18
T16	2^1	-->	FD	B19
T17	2^2	-->	FD	B20
T18	2^3	-->	FD	B21
T19	2^0 10^6 BCDx4 data	-->	FD	B14
T20	2^1	-->	FD	B15
T21	2^2	-->	FD	B16
T22	2^3	-->	FD	B17
T23	2^0 10^7 BCDx4 data	-->	FD	B7
T24	2^1	-->	FD	B8
T25	2^2 2^3	--> -->	FD FD	B9 B10
T26	F5 15 to 30 MHz filter select	-->	RPA1	5
T27,T28	gnd.	<--	chassis	-

BCDx4, Frequency Mark Generator & Tx Filter Control PC2 - Solder Side (B)

Pin No.	Signal Name and Function	Direction	From/to	Pin No.
B1,B2	+5VDC supply	<--	chassis	-
B3	2^3 10^3 synth data	<--	PC1	T7
B4	2^2	<--	PC1	T8
B5	2^1	<--	PC1	T9
B6	2^0	<--	PC1	T10
B7	2^3 10^4 synth data	<--	PC1	B14
B8	2^2	<--	PC1	B13
B9	2^1	<--	PC1	B12
B10	2^0	<--	PC1	B11
B11	Fine Freq Mark	-->	RPA3	11
B12	Bold Freq Mark	-->	RPA3	10
B13	ST27 Time Mark	<--	PC6	B22
B14	Clear frequency mark counters	<--	PC7	T12
B15	2^3 10^5 synth data	<--	PC1	B18
B16	2^2	<--	PC1	B17
B17	2^1	<--	PC1	B16
B18	2^0	<--	PC1	B15
B19	2^3 10^6	<--	PC1	B22
B20	2^2	<--	PC1	B21
B21	2^1	<--	PC1	B19
B22	2^0	<--	PC1	B18
B23	F1 2-3 MHz filter	-->Brn	RPA1	1
B24	F2 3-5 MHz filter	-->Red	RPA1	2
B25	F3 5-8 MHz filter	-->Org	RPA1	3
B26	F4 8-15 MHz filter	-->Yel	RPA1	4
B27,B28	+5VDC supply	<--	chassis	-

Frequency Display , FD

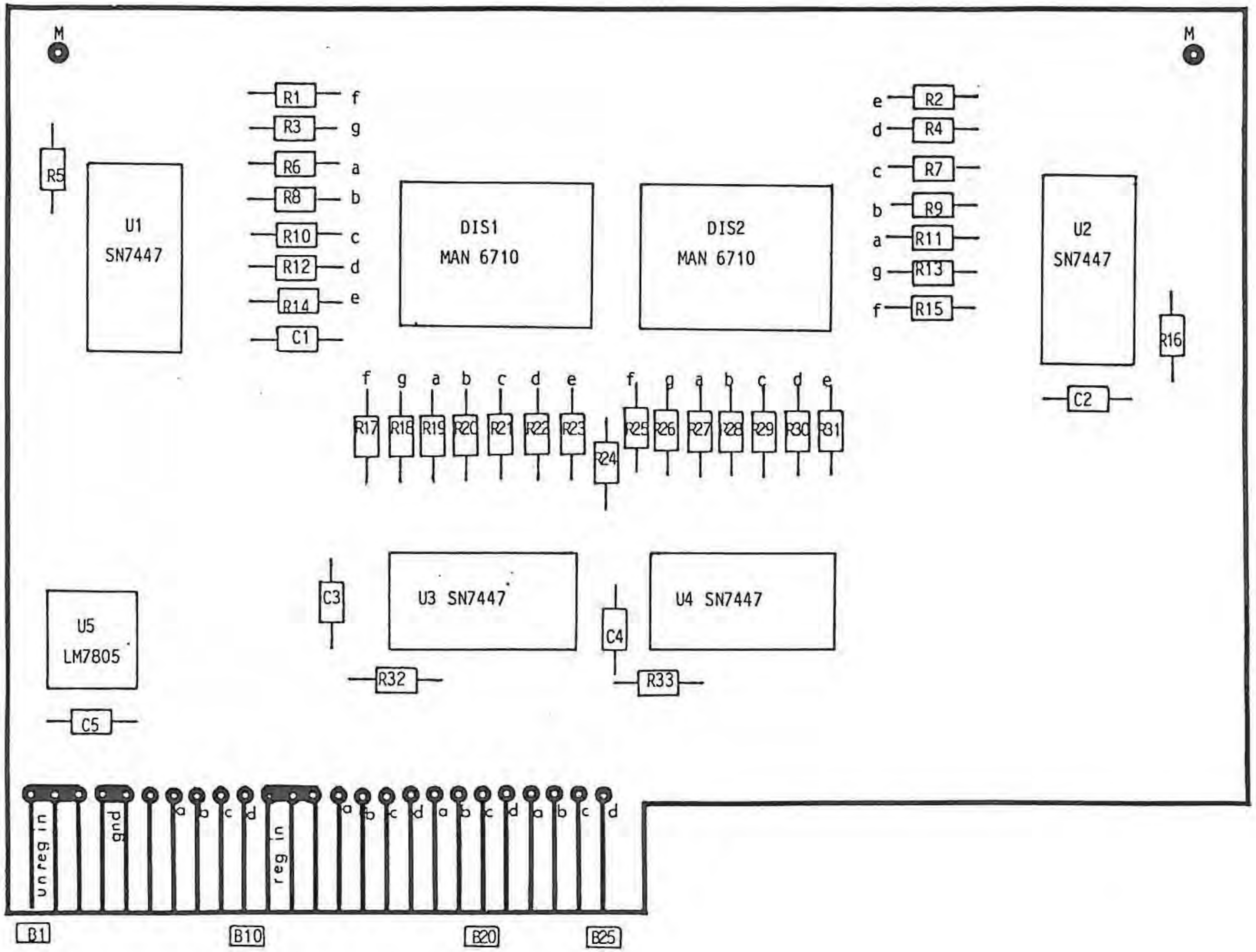




Notes:

Integrated Circuit		Vcc pin	Gnd. pin
U1,U2	SN7447	16	8
U3,U4	SN7447	16	8
DIS1	MAB6710	13,14	-
DIS2	MAN6710	13,14	-
U5	LM7805	-	2

Frequency Display and Drivers FD



<u>Notes</u>:

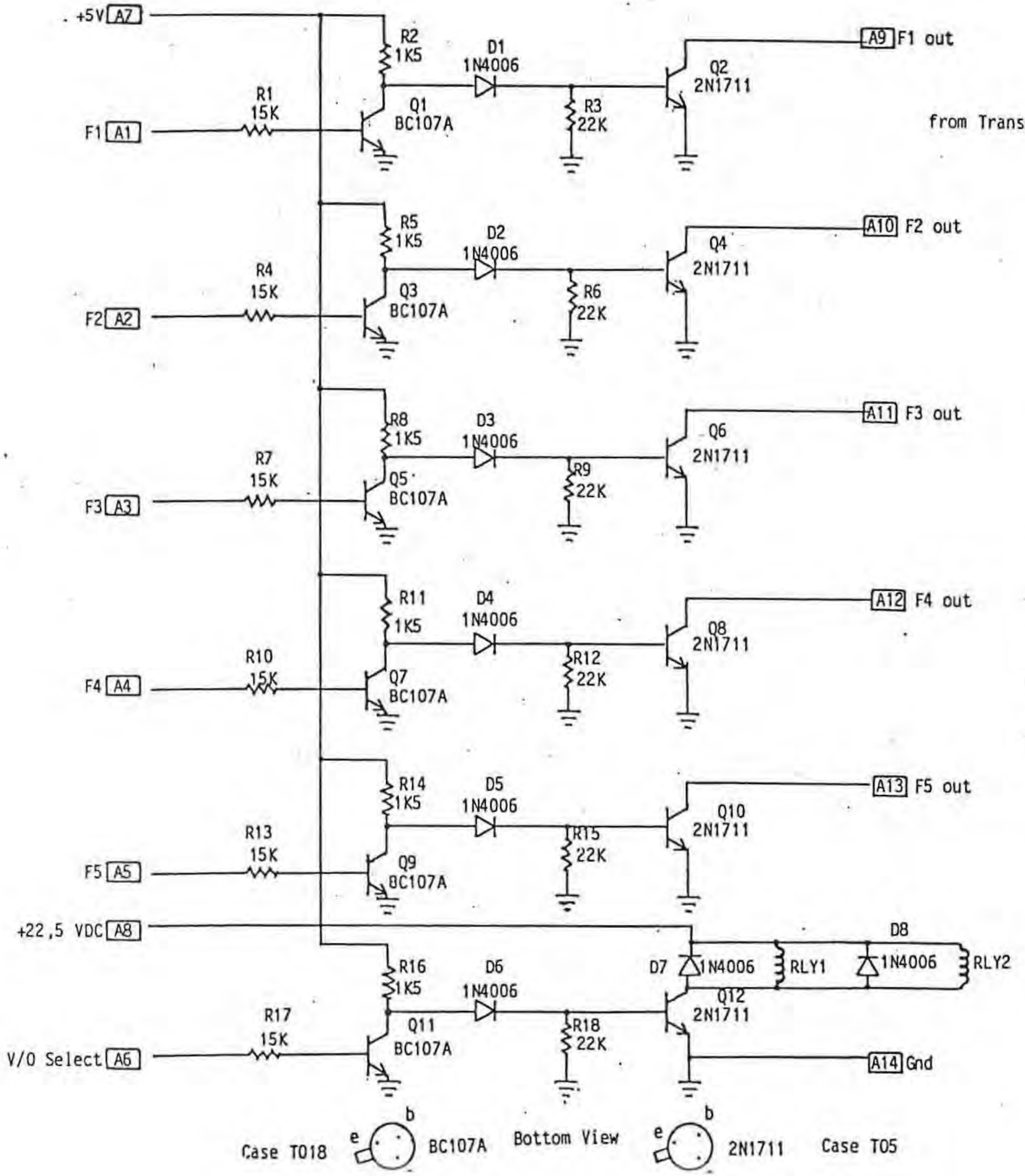
M = Mounting hole

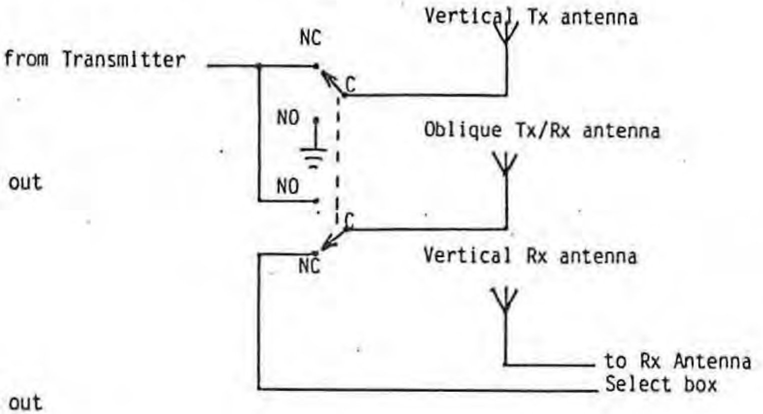
Frequency Display , FD - Solder Side (B)

Pin No.	Signal Name and Function	Direction	From/to	Pin No.
B1,B2,B3	7 - 8 VDC unregulated	<--	chassis	-
B4	gnd	<--	chassis	-
B5	gnd	<--	chassis	-
B6	-	-	-	-
B7	2^0 10^7 BCDx4 data	<--	PC2	T23
B8	2^1	<--	PC2	T24
B9	2^2	<--	PC2	T25
B10	2^3	<--	PC2	T25
B11	optional +5V regulated in	<--	-	-
B12	optional +5V regulated in	<--	-	-
B13	optional +5V regulated in	<--	-	-
B14	2^0 10^6 BCDx4 data	<--	PC2	T19
B15	2^1	<--	PC2	T20
B16	2^2	<--	PC2	T21
B17	2^3	<--	PC2	T22
B18	2^0 10^5 BCDx4 data	<--	PC2	T15
B19	2^1	<--	PC2	T16
B20	2^2	<--	PC2	T17
B21	2^3	<--	PC2	T18
B22	2^0 10^4 BCDx4 data	<--	PC2	T3
B23	2^1	<--	PC2	T4
B24	2^2	<--	PC2	T5
B25	2^3	<--	PC2	T6

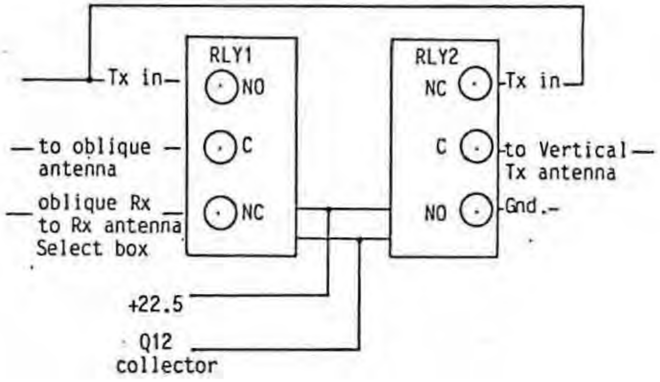
Antenna Select and Tx Filter Drivers

Relay Drivers for Tx Filter Control and Vertical/Oblique Antenna Switching





RF Signal Paths



Notes:

An = Amphenol 14 pin socket pin n

Filter control drivers switch 140 mA

Vertical/Oblique driver switches 180 mA

NO = Normally open

C = Common

NC = Normally closed

(·) = BNC connector

Antenna Select and Transmitter Filter Control Signals , RPA1

Signals to 50 way rear panel amphenol connector , RPA1

Pin No.	Signal Name	Colour	Pin No.	Signal Name	Colour
1	F1 2-3 MHz	Brn	26		
2	F2 3-5 MHz	Red	27		
3	F3 5-8 MHz	Orn	28		
4	F4 8-15 MHz	Yel	29		
5	F5 15-30 MHz	Grn	30		
6	V/O Select	Grn/Wht	31		
7	+5VDC	Red	32		
8	Vertical Rx/$\overline{\text{Oblique Rx}}$		33		
9			34		
10			35		
11			36		
12			37		
13			38		
14			39		
15			40		
16			41		
17			42		
18			43		
19			44		
20			45		
21			46		
22			47		
23			48		
24			49		
25	gnd.	Blk	50	gnd	Blk

Antenna Select and Tx Filter Box , B2a

14 way Amphenol socket connections.

Pin No.	Signal Name	Colour	Pin No.	Signal Name	Colour
A1	F1 2-3 MHz ctrl in	Brn	A8	+22.5 V	Blu
A2	F2 3-5 MHz ctrl in	Red	A9	F1 driver out	Brn/Wht
A3	F3 5-8 MHz ctrl in	Orn	A10	F2 driver out	Red/Wht
A4	F4 8-15 MHz ctrl in	Yel	A11	F3 driver out	Orn/Wht
A5	F5 15-30 MHz ctrl in	Grn	A12	F4 driver out	Yel/Wht
A6	V/O Select	Grn/Wht	A13	F5 driver out	Grn/Wht
A7	+5V	Red	A14	gnd.	Blk

14 way Amphenol plug and cable connections

Pin no.	Signal Name	Colour	Pin No.	Signal Name	Colour
A1	F1 ctrl	Red	A8	+22.5 V	Blu
A2	F2 ctrl	Orn	A9	F1 driver out	Brn/Wht
A3	F3 ctrl	Yel	A10	F2 driver out	Red/Wht
A4	F4 ctrl	Grn	A11	F3 driver out	Orn/Wht
A5	F5 ctrl	Blu	A12	F4 driver out	Yel/Wht
A6	V/O Select	Vio	A13	F5 driver out	Grn/Wht
A7	+5 V	Blk	A14	gnd.	Blk
	Cable 1 to RPA1			Cable 2 to T500A Amp	

Note :- RPA1 ground on pins 25 and 50 is connected to Cable 1 shield braid.

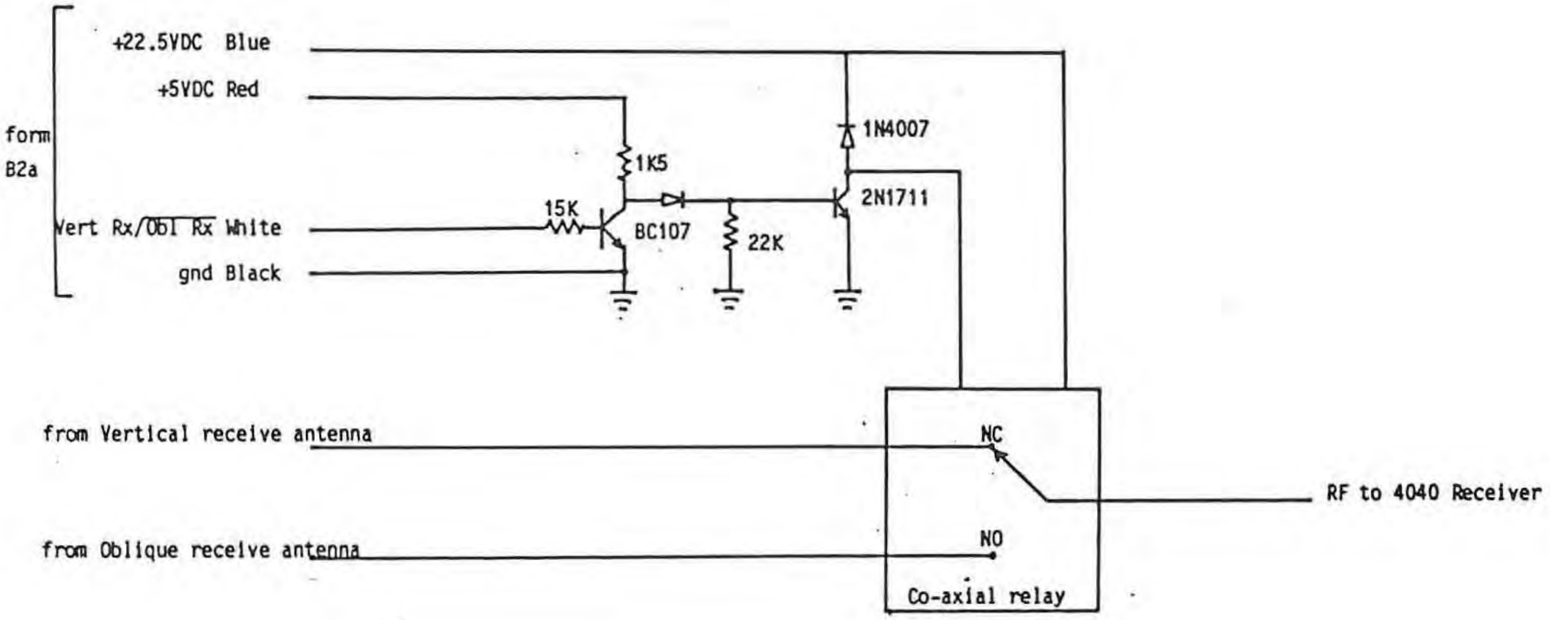
Subminiature Co-axial Connector on B2a

Centre Conductor - Vertical Rx/$\overline{\text{Oblique Rx}}$ for 4040 Rx Antenna Select.

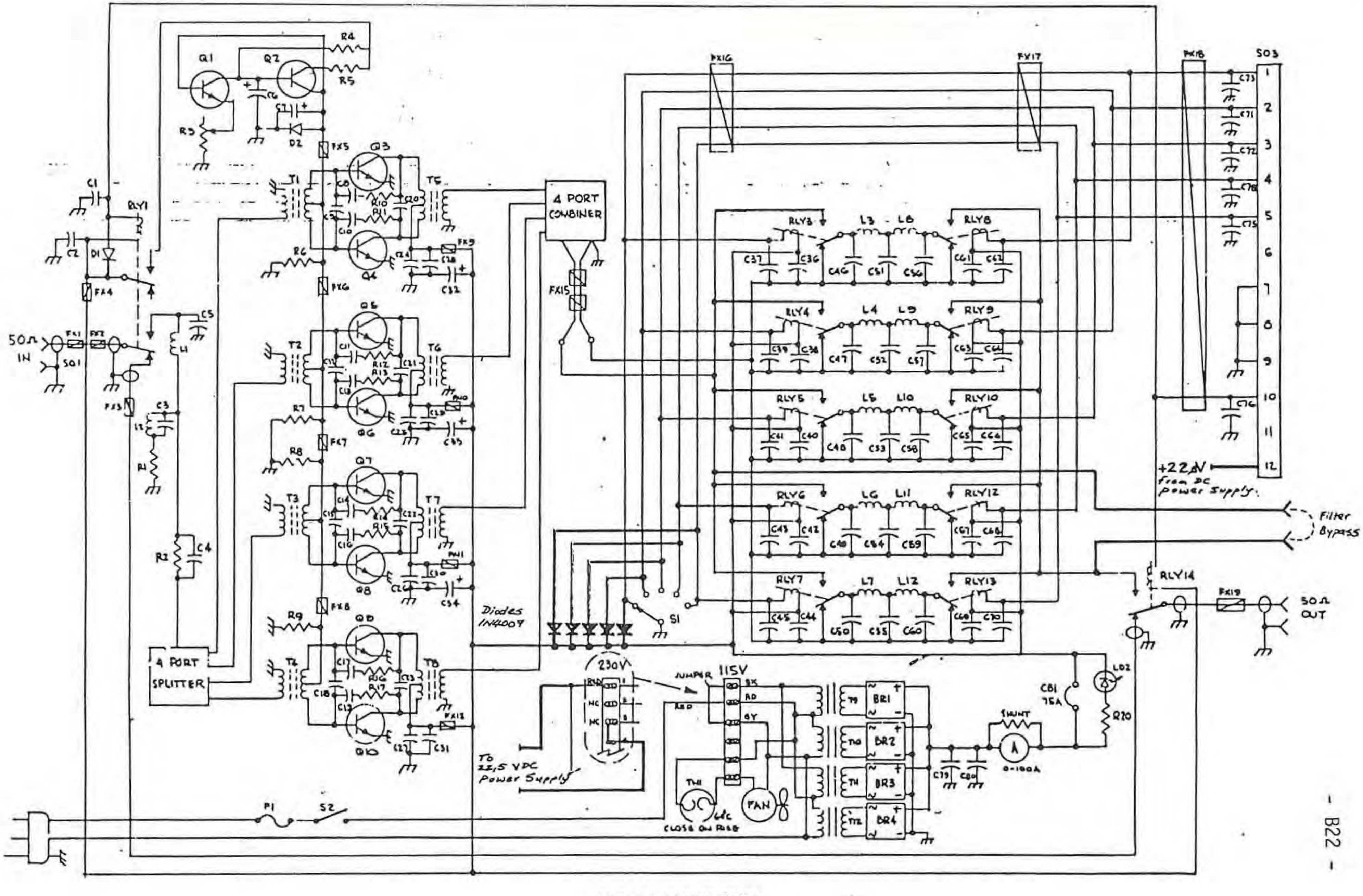
Shield - gnd.

Note :- This signal goes through B2a to the 4040 Rx Antenna Select Box B2b on a white wire. Power from B2a to B2b is +22.5 V (blue) and +5V (red).

AC1 pin connections:- 1 = F1 ctrl signal , 2 = gnd , 3 = +5VDC from B2a.

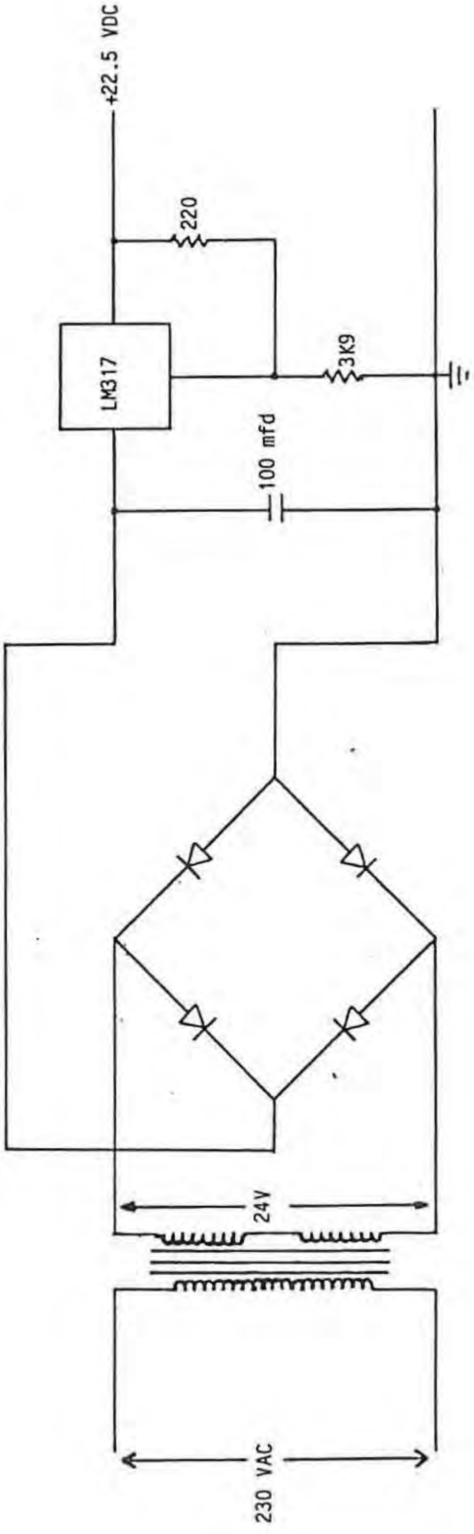


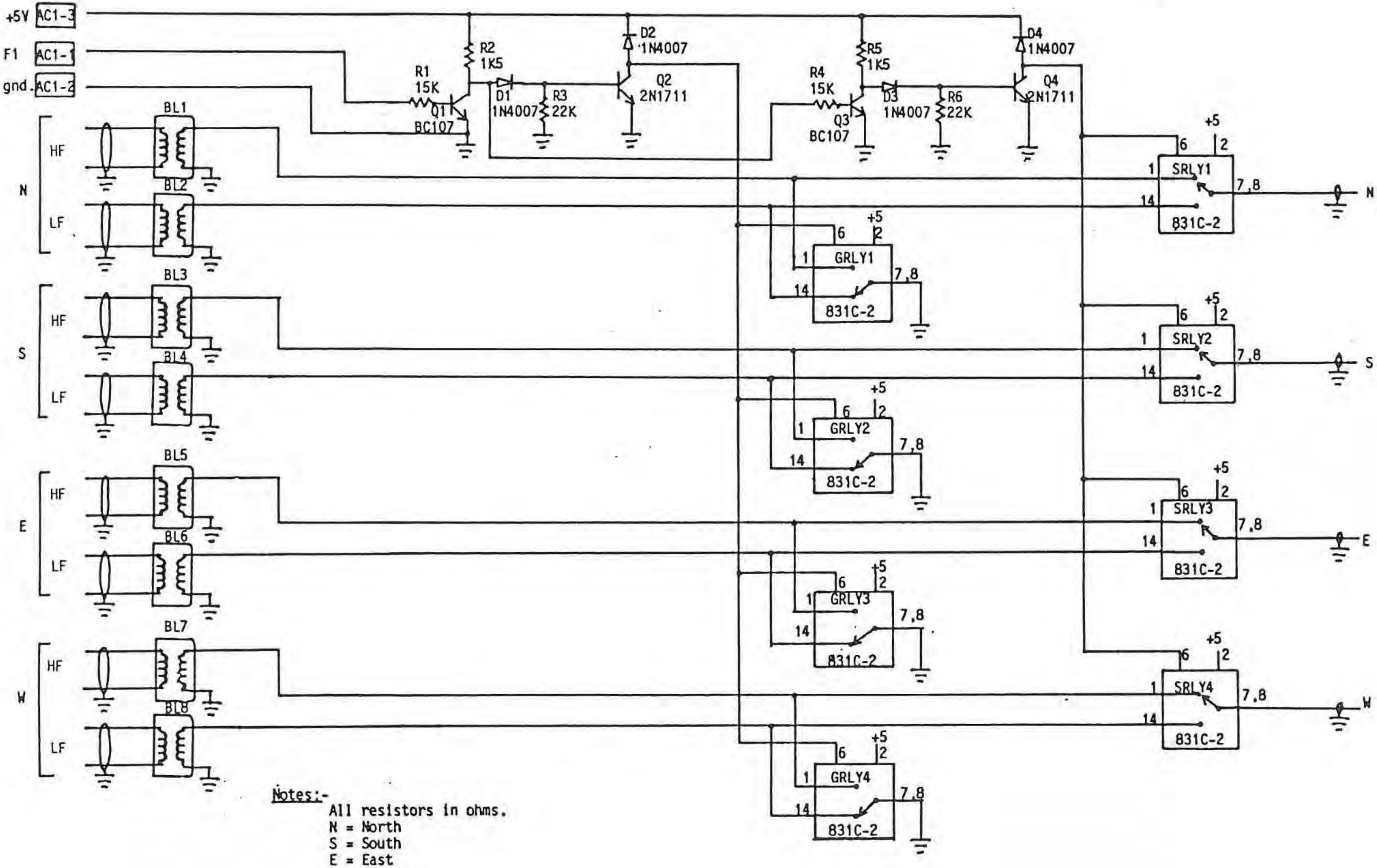
4040 Rx Antenna Select , B2b



SCHEMATIC TWE MOD T600A
(mods Sep 1982)

+22.5 VDC Power Supply mounted in TWE Power Amplifier

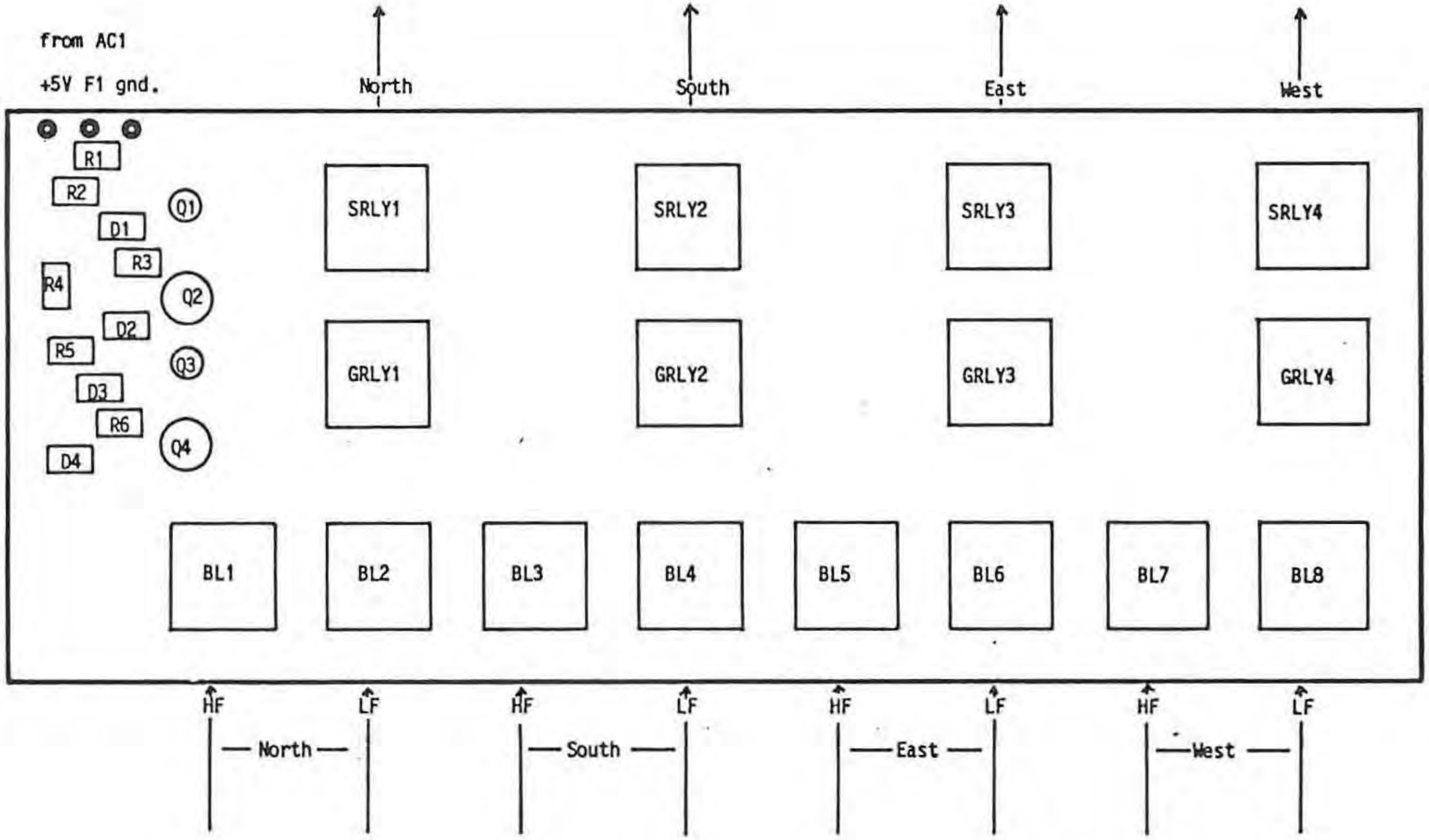




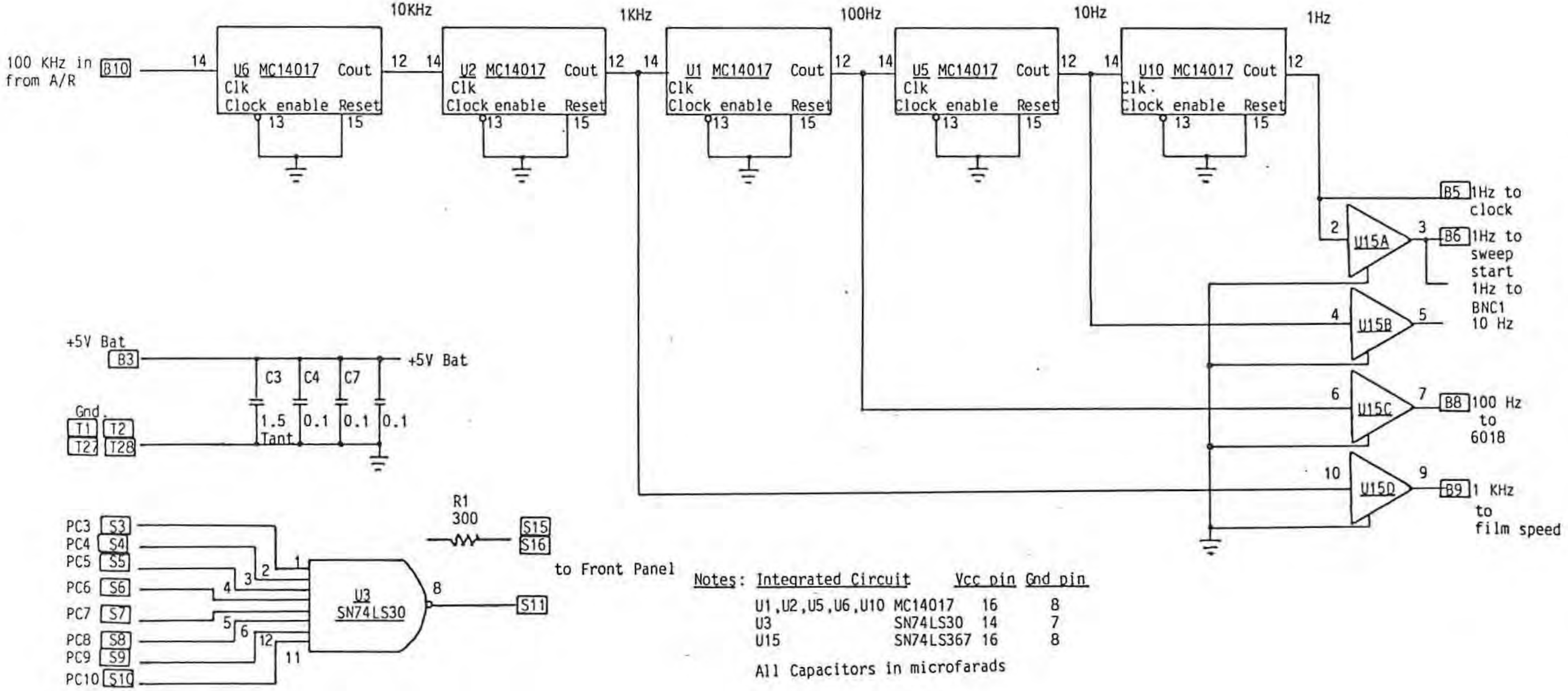
Notes:-

All resistors in ohms.
N = North
S = South
E = East
W = West
LF = Low Frequency
HF = High Frequency

3 MHz Antennae Change-over , B4



3 MHz Antennae Change-over , B4

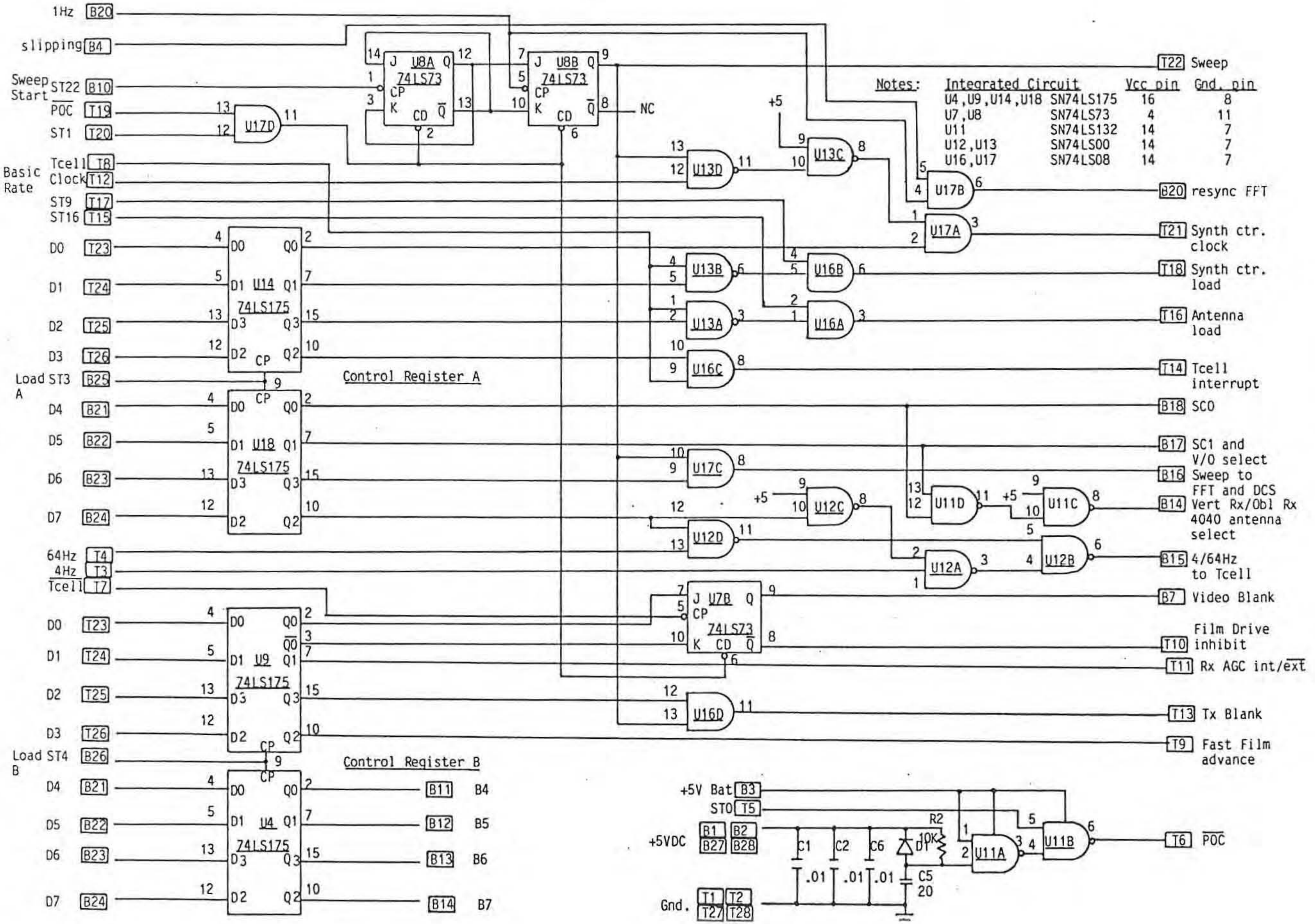


Notes:

Integrated Circuit		Vcc pin	Gnd pin
U1,U2,U5,U6,U10	MC14017	16	8
U3	SN74LS30	14	7
U15	SN74LS367	16	8

All Capacitors in microfarads

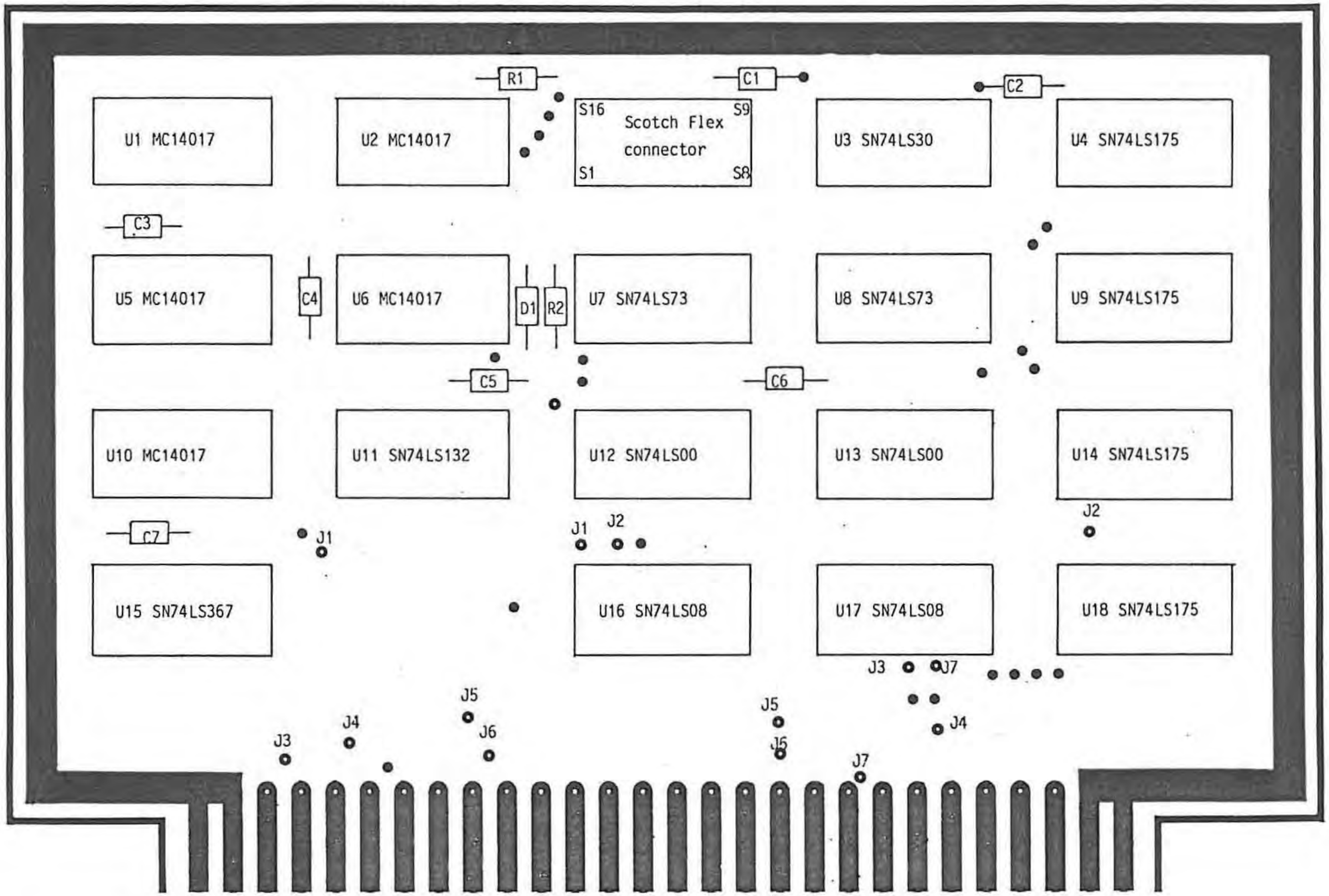
Clock Dividers , PC3a



Notes:

Integrated Circuit		Vcc pin	Gnd. pin
U4, U9, U14, U18	SN74LS175	16	8
U7, U8	SN74LS73	4	11
U11	SN74LS132	14	7
U12, U13	SN74LS00	14	7
U16, U17	SN74LS08	14	7

Sweep Start and Control, PC3b



Component Side T1 T10 T20 T28
Solder Side B1 B10 B20 B28

Notes:
○ = Feedthrough
Jn Jn = Jumper n

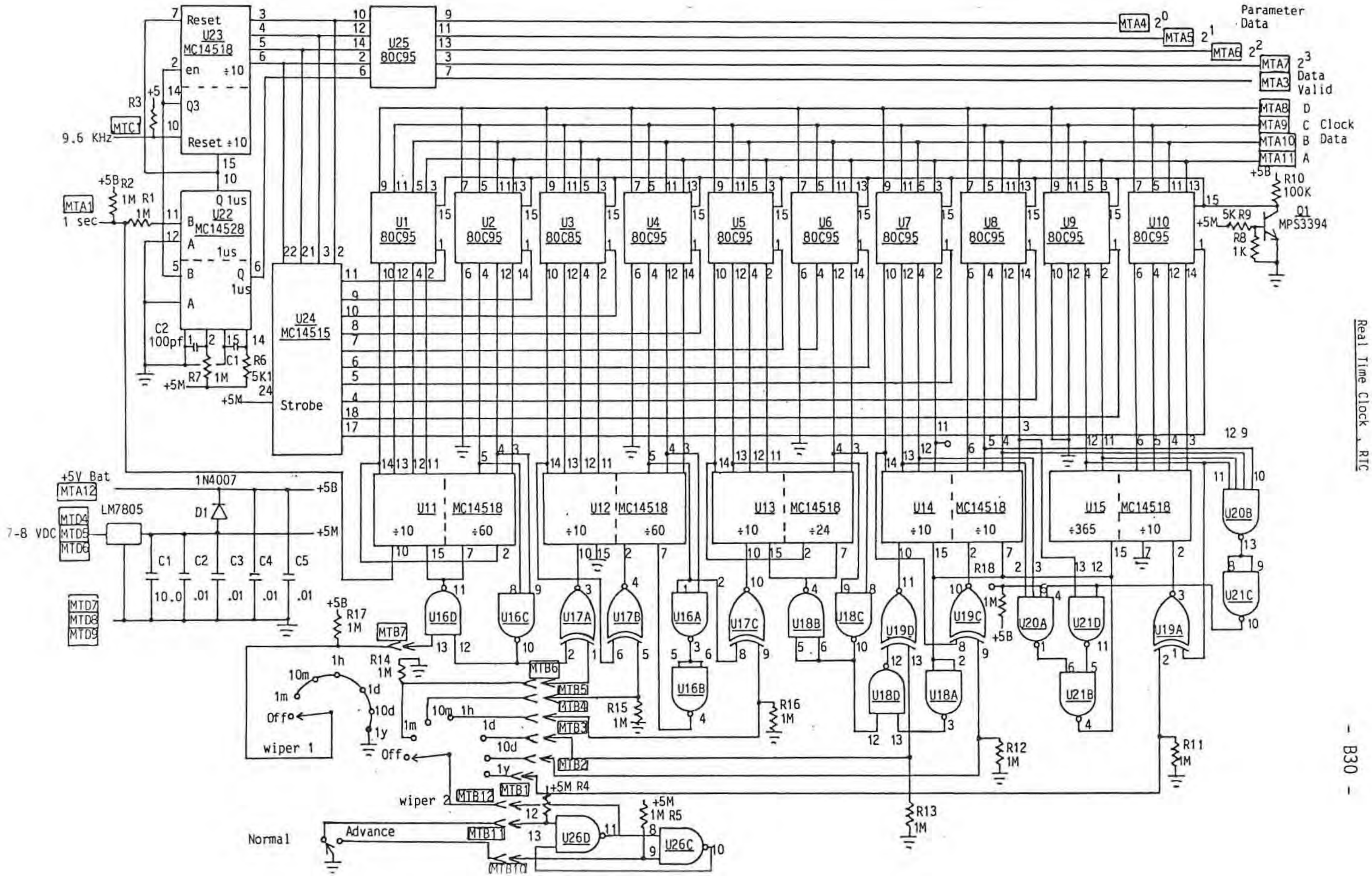
Clock Dividers and Sweep Start and Control , PC3

Clock and Sweep Start and Control , PC3 - Component Side (T)

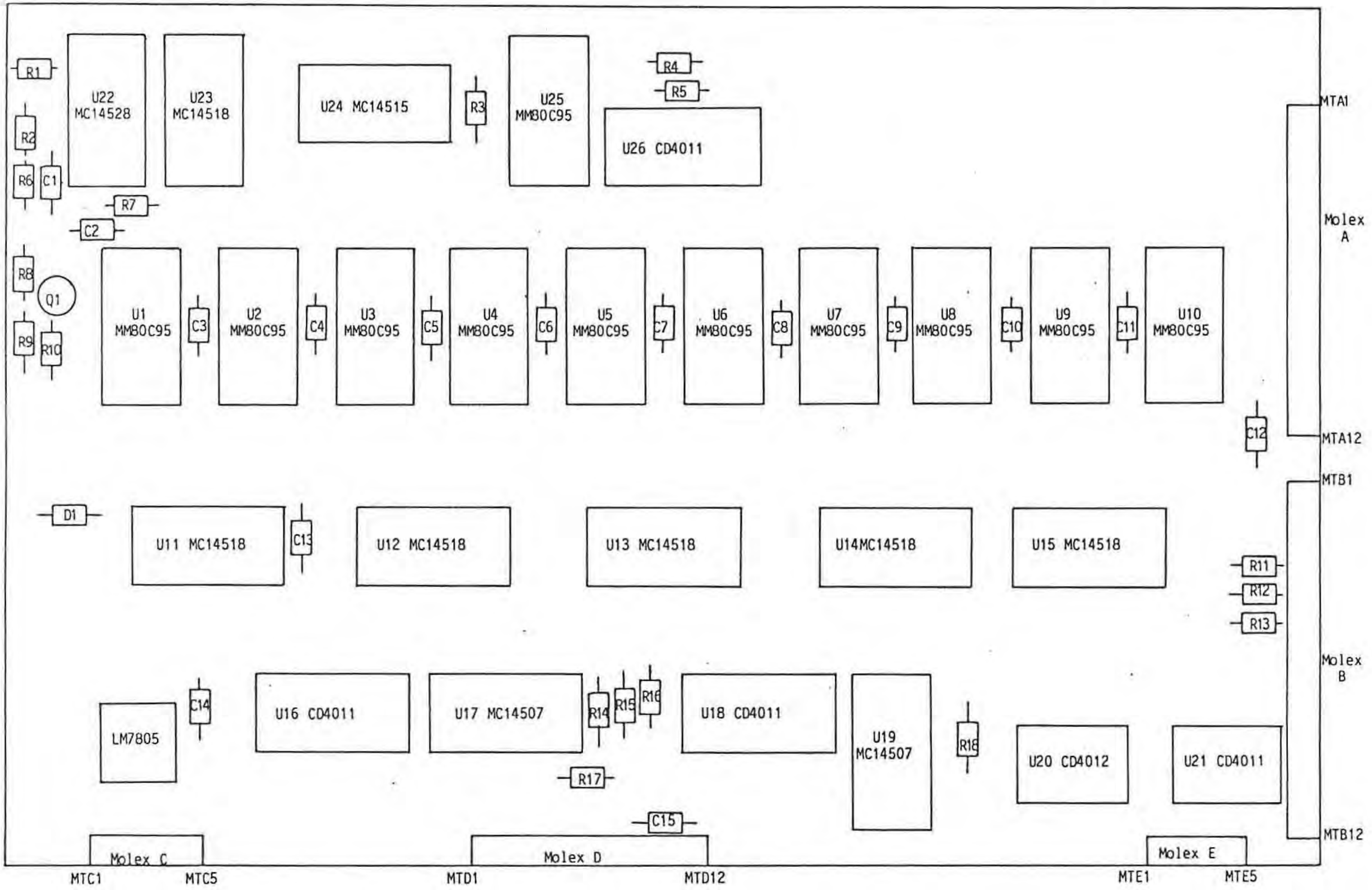
Pin No.	Signal Name and Function	Direction	From/to	Pin No.
T1,T2	gnd	<--	chassis	-
T3	4 Hz	<--	PC7	B18
T4	64 Hz	<--	PC7	B15
T5	ST0 Clear sweep A/R & film dr f/f	<--	PC6	T10
T6	$\overline{\text{POC}}$ Power on clear	--> -->	PC3 PC5	T19 T26
T7	$\overline{\text{Tcell}}$ pulse	<--	PC4	T12
T8	Tcell pulse	<--	PC4	B12
T9	Fast Film advance	-->	RPA3	7
T10	Film Drive inhibit	-->	PC8	B16
T11	Rx AGC int/$\overline{\text{ext}}$ (data to DCS)	-->	RPA4	13
T12	Basic Rate Clock	<--	PC4	B13
T13	Tx Blank	-->	RPA2	2
T14	Tcell interrupt to CPU	-->	MP-LA-1A	CA1
T15	ST16 Antenna data initial load	<--	PC6	T11
T16	Antenna load	-->	PC8	B12
T17	ST9 Synth cntr initial load	<--	PC6	B4
T18	Synth cntr load	-->	PC1	T5
T19	$\overline{\text{POC}}$ Power on clear	<--	PC3	T6
T20	ST1 Clear sweep f/f	<--	PC6	T9
T21	Synth cntr clock	-->	PC1	T6
T22	Sweep To Tcell pulse gen & camera	--> -->	PC4 RPA3	B11 3
T23	D0 Controller data bus	<--		
T24	D1	<--		
T25	D2	<--		
T26	D3	<--		
T27,T28	gnd.	<--	chassis	-

Clock and Sweep Start and Control , PC3 - Solder Side (B)

Pin No.	Signal Name and Function	Direction	From/to	Pin No.
B1,B2	+5VDC Supply	<--	chassis	-
B3	+5V Bat Battery Supply	<--	diode PCB	-
B4	slipping Timing change	<--	PC5	T19
B5	1 Hz	-->	RTC	MTA1
B6	1 Hz buffered	-->	PC3 BNC1	B20 -
B7	Video Blank	-->	RPA3	4
B8	100 Hz buffered	-->	RPA3	8
B9	1 kHz buffered	-->	PC8	B14
B10	100 kHz in	<--	PC5	B7
B11	B4 Spare latched data	-->	-	-
B12	B5 Note that B7 is available	-->	-	-
B13	B6 on the printed circuit board.	-->	-	-
B14	Vertical Rx/$\overline{\text{Oblique Rx}}$ 4040 ant.sel	-->	RPA1	8
B15	4/64 Hz out to Tcell generator	-->	PC4	B8
B16	Sweep , gated , to FFT and DCS	--> --> -->	PC4 PC7 RPA4	B11 B14 37
B17	SC1 and V/0 select Tx , Rx and	--> -->	RPA1 RPA2	6 9
B18	SC0 T/R control	-->	RPA2	8
B19	ST22 Sweep start pulse in	<--	PC6	T17
B20	resync FFT gen	-->	PC7	B11
B21	D4 Controller data bus	<--		
B22	D5	<--		
B23	D6	<--		
B24	D7	<--		
B25	ST3 load Control Register A	<--	PC6	T7
B26	ST4 load Control register B	<--	PC6	T6
B27,B28	+5VDC Supply	<--	chassis	-



Real Time Clock , RTC



Real Time Clock , RTC

Real Time Clock , RTC

Molex Connector A (MTA)

Pin No.	Signal Name and Function	Direction	From/to	Pin No.
MTA1	1 s 1 second square wave	<--	PC3	B5
MTA2	-	-	-	-
MTA3	Data Valid	--> -->	RTCD MP-LA-2A	MB9 CA1
MTA4	2^0 Parameter Data	--> -->	RTCD MP-LA-2A	MB8 A0
MTA5	2^1	--> -->	RTCD MP-LA-2A	MB7 A1
MTA6	2^2	--> -->	RTCD MP-LA-2A	MB6 A2
MTA7	2^3	--> -->	RTCD MP-LA-2A	MB5 A3
MTA8	D MSB Clock Data	--> -->	RTCD MP-LA-2A	MB4 A7
MTA9	C	--> -->	RTCD MP-LA-2A	MB3 A6
MTA10	B	--> -->	RTCD MP-LA-2A	MB2 A5
MTA11	A	--> -->	RTCD MP-LA-2A	MB1 A4
MT12	+5V Bat Battery supply	<--	Diode PCB	-

Molex Connector B (MTB)

Pin No.	Signal Name and Function	Direction	From/to	Pin No.
MTB1	1y	<--	Rot switch	-
MTB2	10d	<--	Rot switch	-
MTB3	1d	<--	Rot switch	-
MTB4	1h	<--	Rot switch	-
MTB5	10m	<--	Rot switch	-
MTB6	1m	<--	Rot switch	-
MTB7	SW1 zero seconds	<--	Wiper 1	-
MTB8	+5 main supply	-->	RTCD	MB12
MTB9	gnd.	-->	RTCD	MB11
MTB10	Advance pulse advances clock	<--	toggle N.O.	-
MTB11	Normal normal posn of switch	<--	toggle N.C.	-
MTB12	SW2 Select parameter	<--	wiper 2	-

Real Time Clock , RTC

Molex Connector C (MTC)

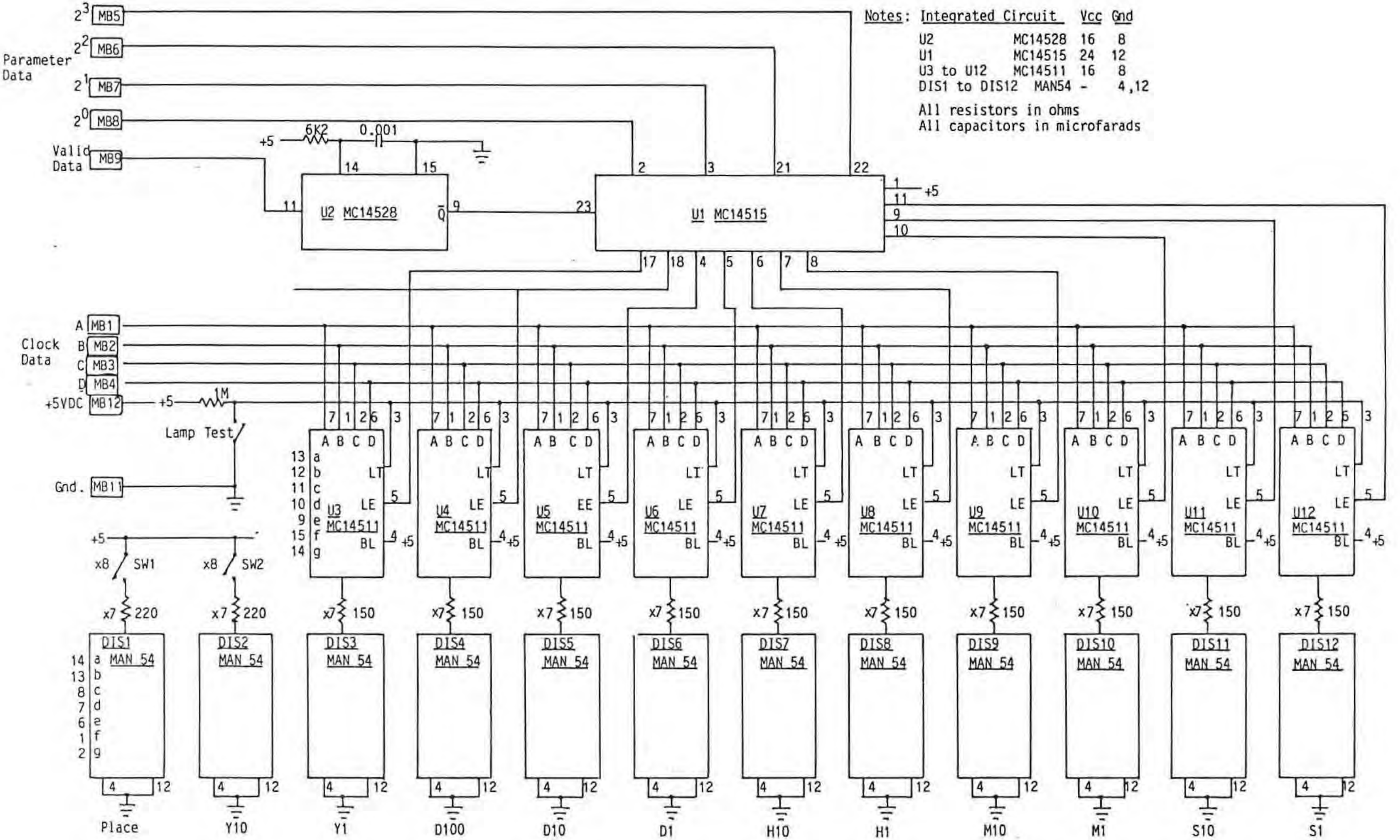
Pin No.	Signal Name and Function	Direction	From/to	Pin No.
MTC1	9.6 kHz Multiplexing freq in	<--	MPB2	1
MTC2	-	-	-	-
MTC3	-	-	-	-
MTC4	-	-	-	-
MTC5	-	-	-	-

Note: The multiplexing frequency of 9.6 kHz is obtained from the computer motherboard 600b line.

Molex Connector D (MTD)

Pin No.	Signal Name and Function	Direction	From/to	Pin No.
MTD1	-	-	-	-
MTD2	-	-	-	-
MTD3	-	-	-	-
MTD4	7 - 8 VDC	<--	MPB2	7-8V
MTD5	7 - 8 VDC	<--	MPB2	7-8V
MTD6	7 - 8 VDC	<--	MPB2	7-8V
MTD7	gnd.	<--	MPB2	gnd.
MTD8	gnd	<--	MPB2	gnd
MTD9	gnd	<--	MPB2	gnd
MTD10	-	-	-	-
MTD11	-	-	-	-
MTD12	-	-	-	-

Note: Molex connector E is for mechanical support only.

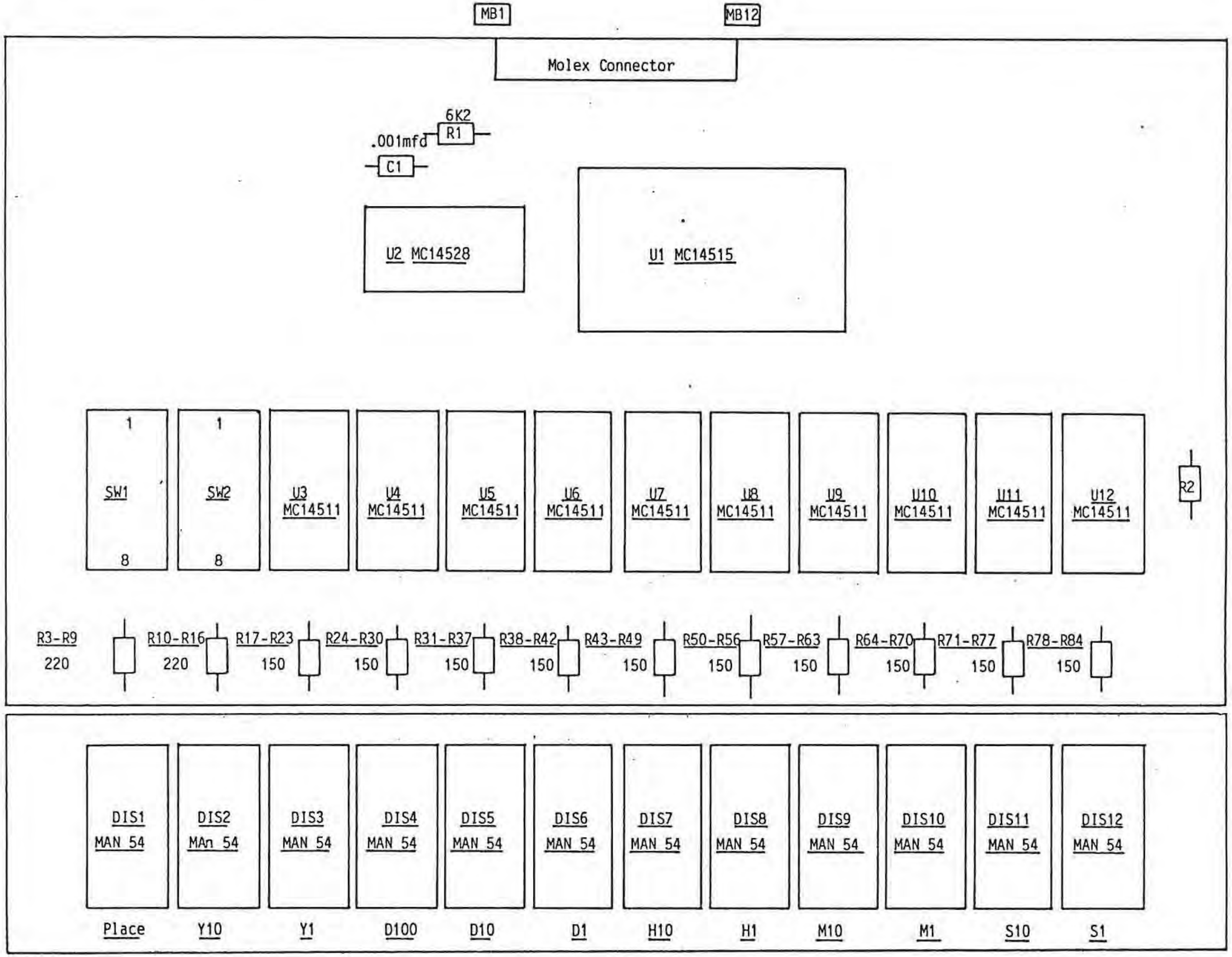


Notes:

Integrated Circuit		Vcc	Gnd
U2	MC14528	16	8
U1	MC14515	24	12
U3 to U12	MC14511	16	8
DIS1 to DIS12	MAN54	-	4,12

All resistors in ohms
All capacitors in microfarads

Real Time Clock Display , RTCD



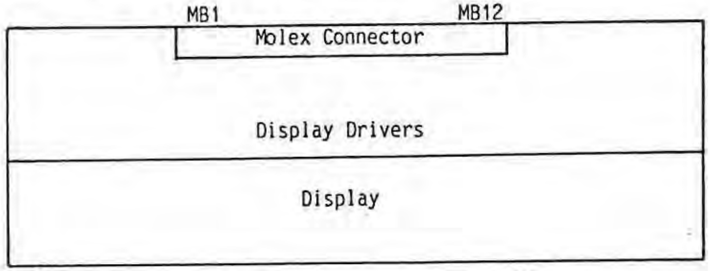
Real Time Clock Display , RTCD

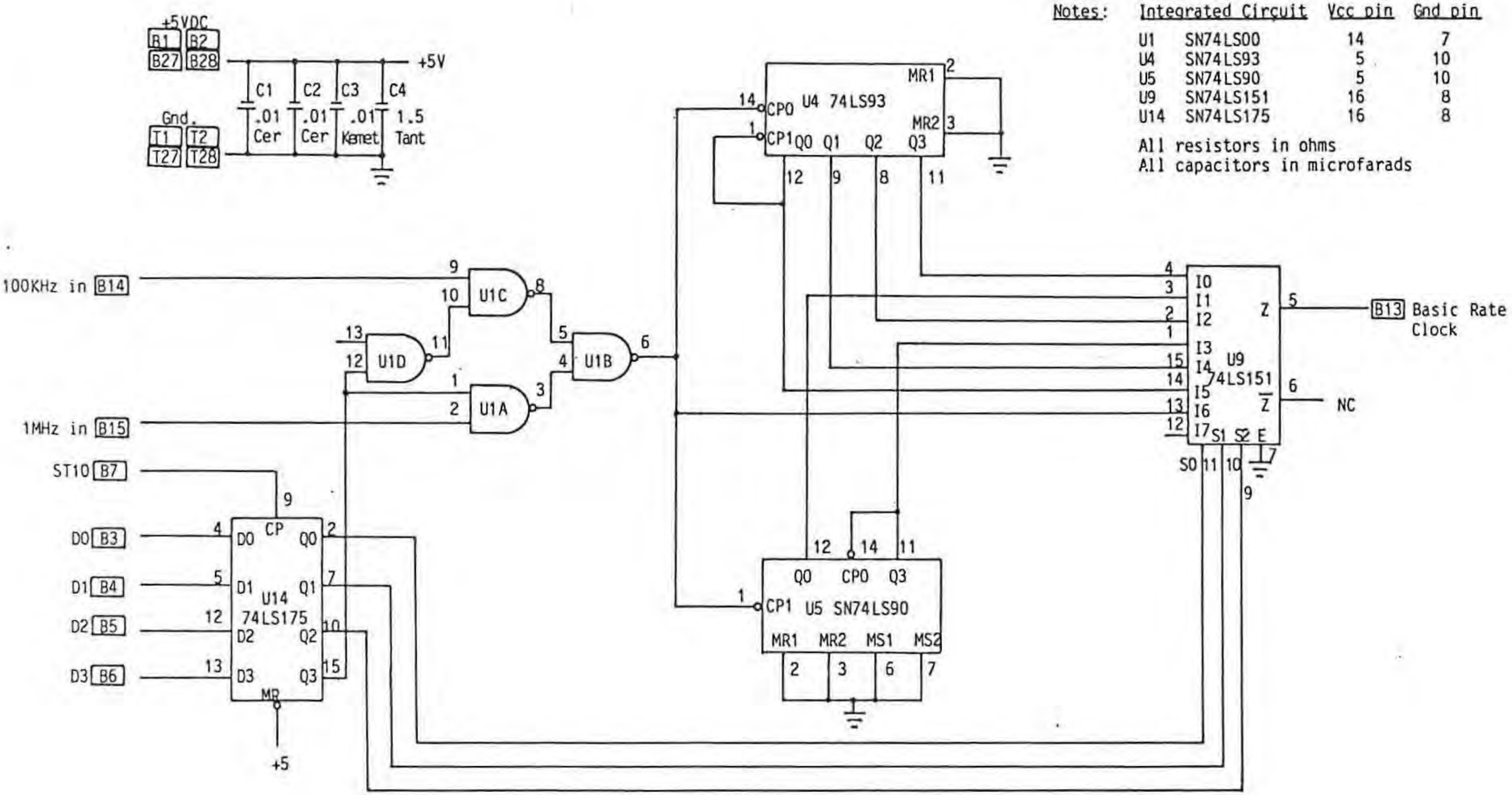
Real Time Clock Display , RTCD

Solder Side Molex Connector (MB)

Pin No.	Signal Name and Function		Direction	From/to	Pin No.
MB1	A	LSB Clock Data	<--	RTC	MTA11
MB2	B		<--	RTC	MTA10
MB3	C		<--	RTC	MTA9
MB4	D	MSB	<--	RTC	MTA8
MB5	2^3	Parameter data	<--	RTC	MTA7
MB6	2^2		<--	RTC	MTA6
MB7	2^1		<--	RTC	MTA5
MB8	2^0		<--	RTC	MTA4
MB9	Data valid		<--	RTC3	MTA3
MB10	-		-	-	-
MB11	gnd.		<--	RTC	MTB9
MB12	+5 VDC		<--	RTC	MTB8

Connector Location



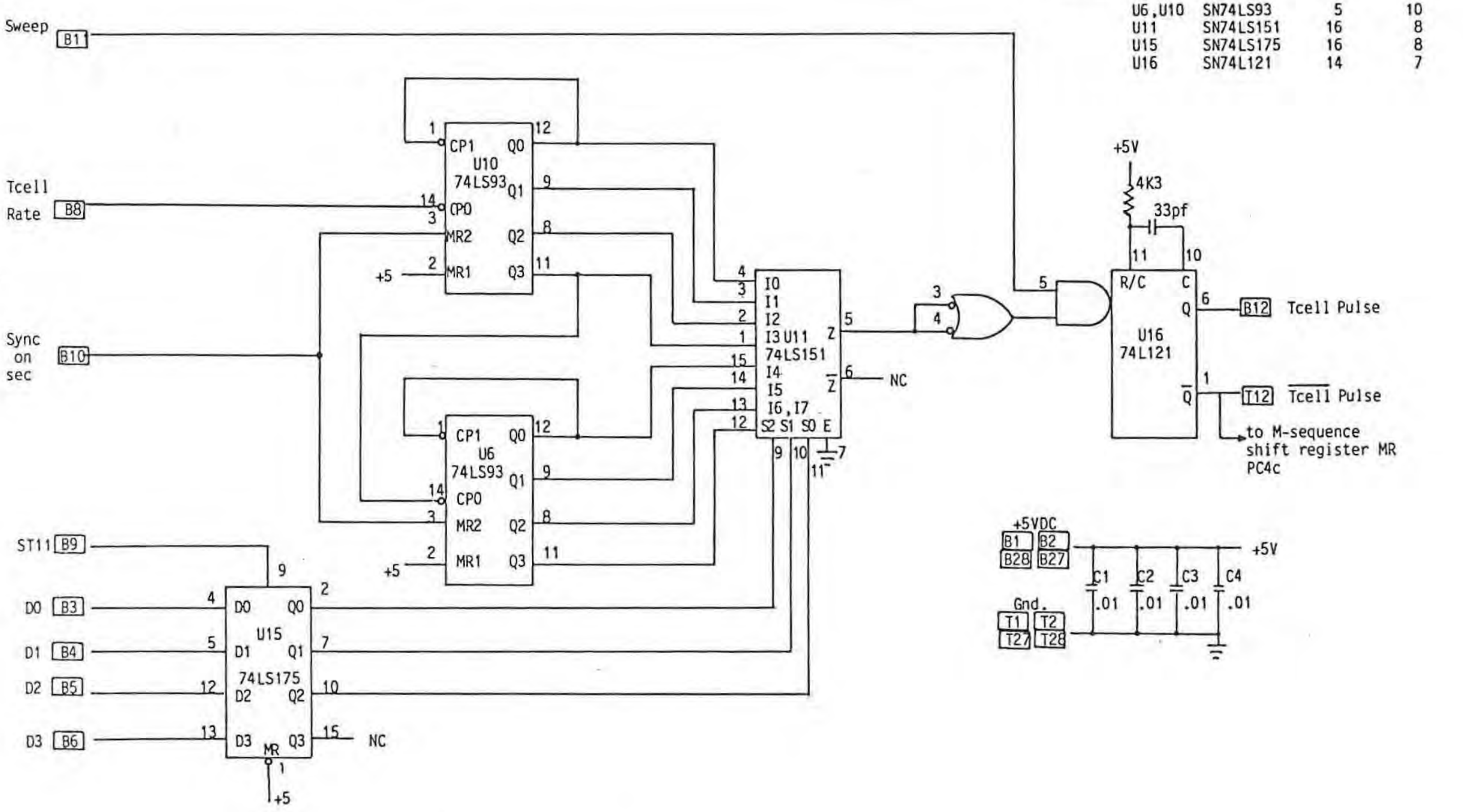


Notes:

Integrated Circuit		Vcc pin	Gnd pin
U1	SN74LS00	14	7
U4	SN74LS93	5	10
U5	SN74LS90	5	10
U9	SN74LS151	16	8
U14	SN74LS175	16	8

All resistors in ohms
All capacitors in microfarads

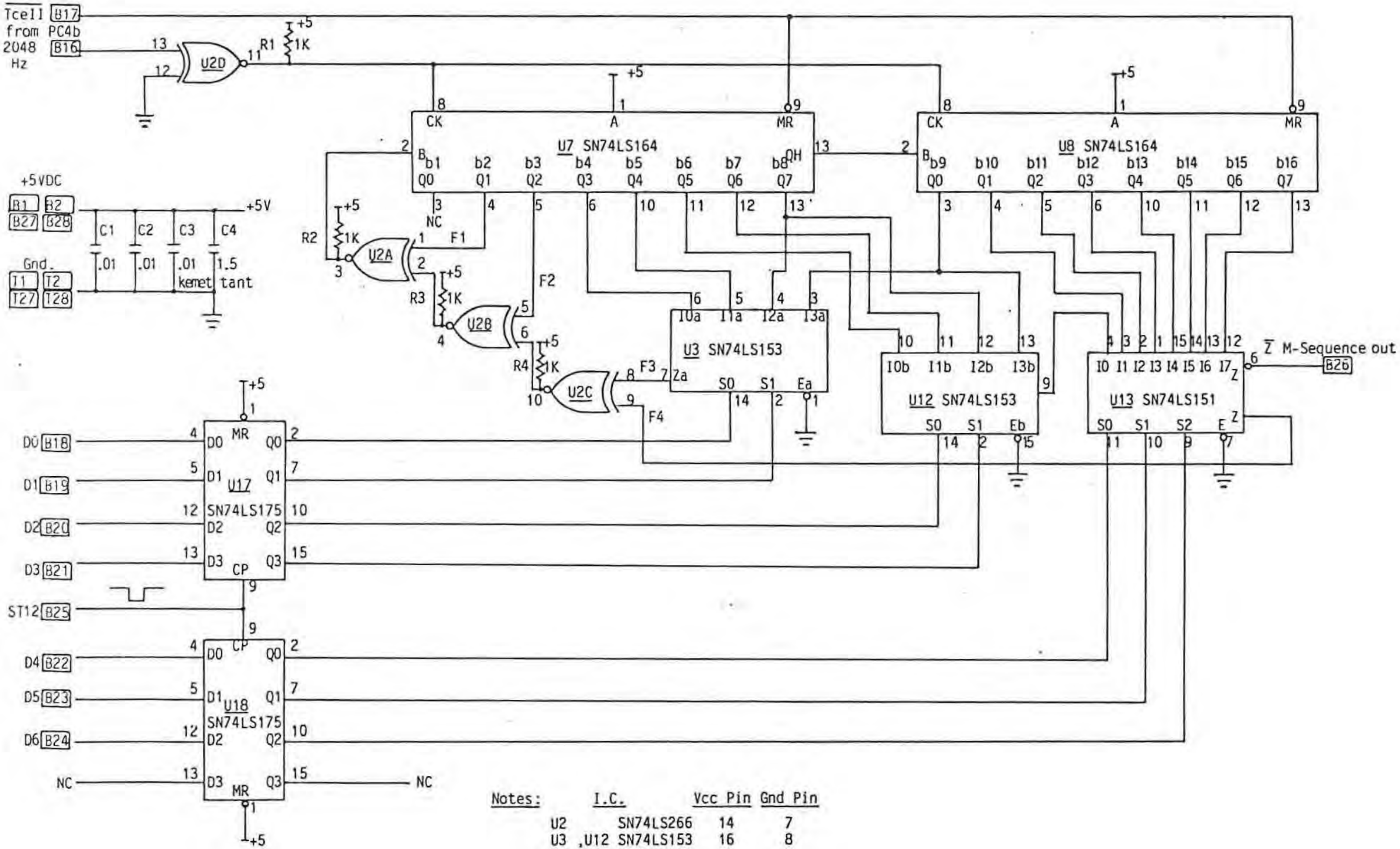
Basic Rate , PC4a



Notes:

Integrated Circuit		Vcc pin	Gnd pin
U6,U10	SN74LS93	5	10
U11	SN74LS151	16	8
U15	SN74LS175	16	8
U16	SN74L121	14	7

Tcell , PC4b



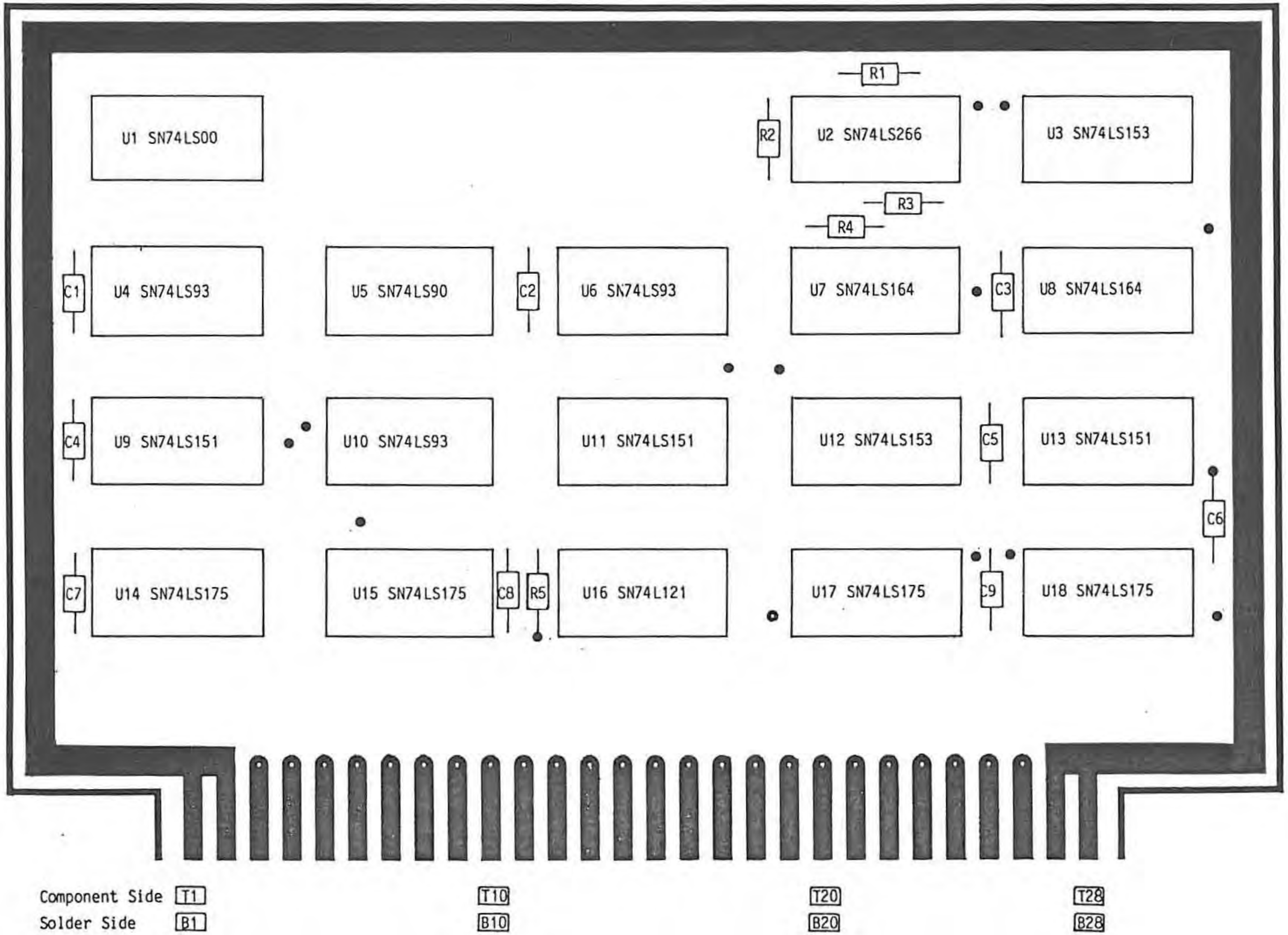
Notes:

I.C.		Vcc Pin	Gnd Pin
U2	SN74LS266	14	7
U3 ,U12	SN74LS153	16	8
U7 ,U8	SN74LS164	14	7
U13	SN74LS151	16	8
U17,U18	SN74LS175	16	8

All resistors in ohms
All capacitors in microfarads
Ceramic capacitors unless otherwise specified

Programmable M-Sequence Generator , PC4c

Basic Rate , Tcell and M-sequence Generator , PC4



Notes:

○ = Feedthrough

Jn Jn = Jumper n

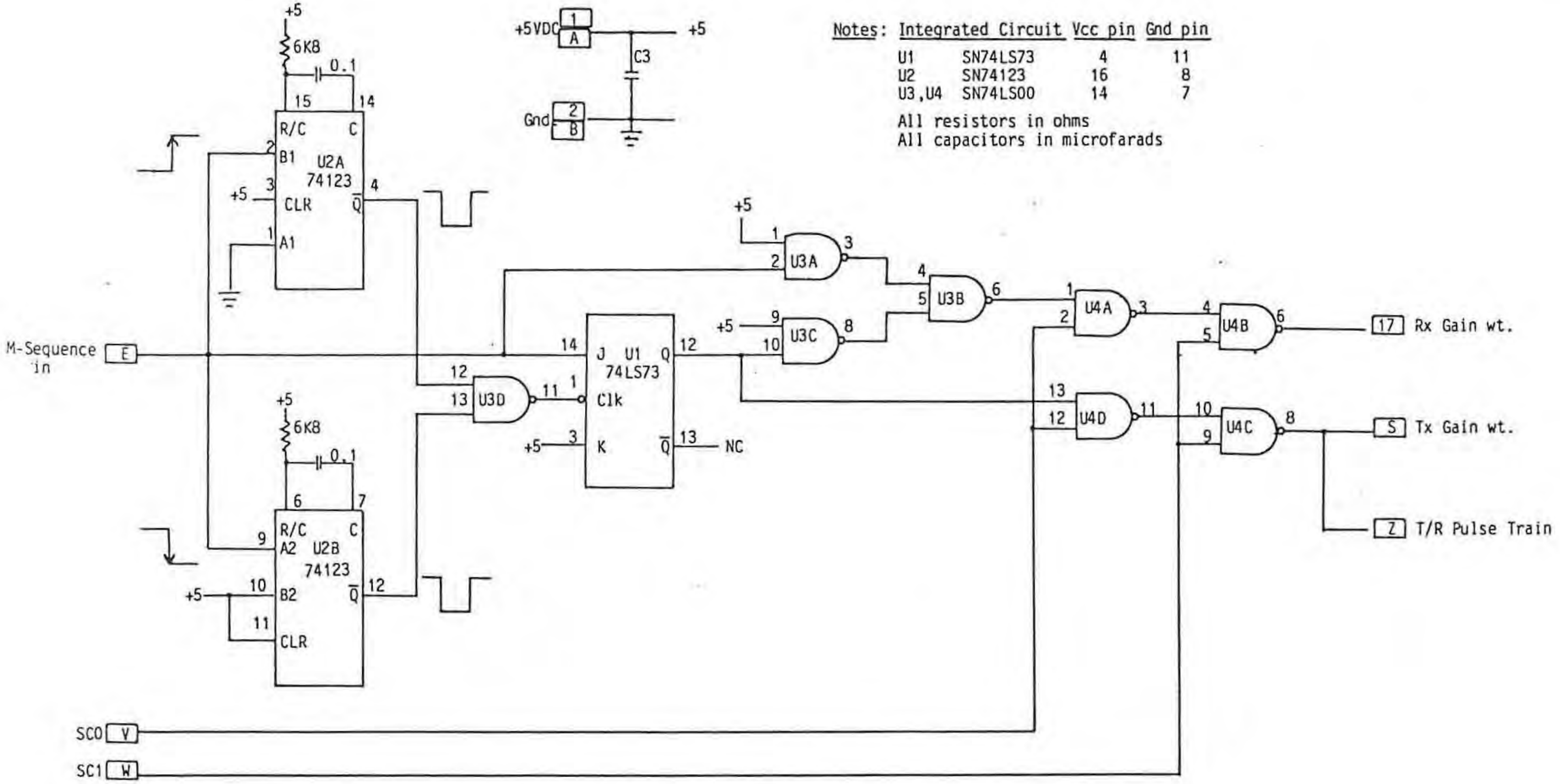
Basic Rate , Tcell and M-Sequence Generators , PC4 - Component Side (T)

Pin No.	Signal Name and Function	Direction	From/to	Pin No.
T1,T2	gnd.	<--	chassis	-
T3	-	-	-	-
T4	-	-	-	-
T5	-	-	-	-
T6	-	-	-	-
T7	-	-	-	-
T8	-	-	-	-
T9	-	-	-	-
T10	-	-	-	-
T11	-	-	-	-
T12	$\overline{\text{Tcell}}$ pulse	--> -->	RPA4 PC3	39 T7
T13	-	-	-	-
T14	-	-	-	-
T15	-	-	-	-
T16	-	-	-	-
T17	-	-	-	-
T18	-	-	-	-
T19	-	-	-	-
T20	-	-	-	-
T21	-	-	-	-
T22	-	-	-	-
T23	-	-	-	-
T24	-	-	-	-
T25	-	-	-	-
T26	-	-	-	-
T27,T28	gnd.	<--	chassis	-

Basic Rate , Tcell and M-Sequence Generators , PC4 - Solder Side (B)

Pin No.	Signal Name and Function	Direction	From/to	Pin No.
B1,B2	+5 VDC supply	<--	chassis	-
B3	D0 Controller data bus	<--		
B4	D1	<--		
B5	D2	<--		
B6	D3	<--		
B7	ST10 Basic Rate load	<--	PC6	B5
B8	4/64 Hz from Sweep Start	<--	PC3	B15
B9	ST11 Tcell period load	<--	PC6	B6
B10	sync on second	<--	PC7	T12
B11	Sweep	<--	PC3	T22
B12	Tcell pulse	-->	PC3	T8
B13	Basic Rate Clock	--> -->	PC3 PC2	T12 T10
B14	100 kHz in	<--	PC5	B8
B15	1 MHz in	<--	PC7	T4
B16	2048 Hz in	<--	PC7	B13
B17	$\overline{\text{Tcell}}$ signal already on PC4	-	-	-
B18	D0 Controller data bus	<--		
B19	D1	<--		
B20	D2	<--		
B21	D3	<--		
B22	D4	<--		
B23	D5	<--		
B24	D6	<--		
B25	ST12 M-Sequence load	<--	PC6	B7
B26	M-sequence out	-->	RPA2	10
B27,B28	+5VDC supply	<--	chassis	-

T/R Gain Weight Generator - replaces Amplifier T/R board A6

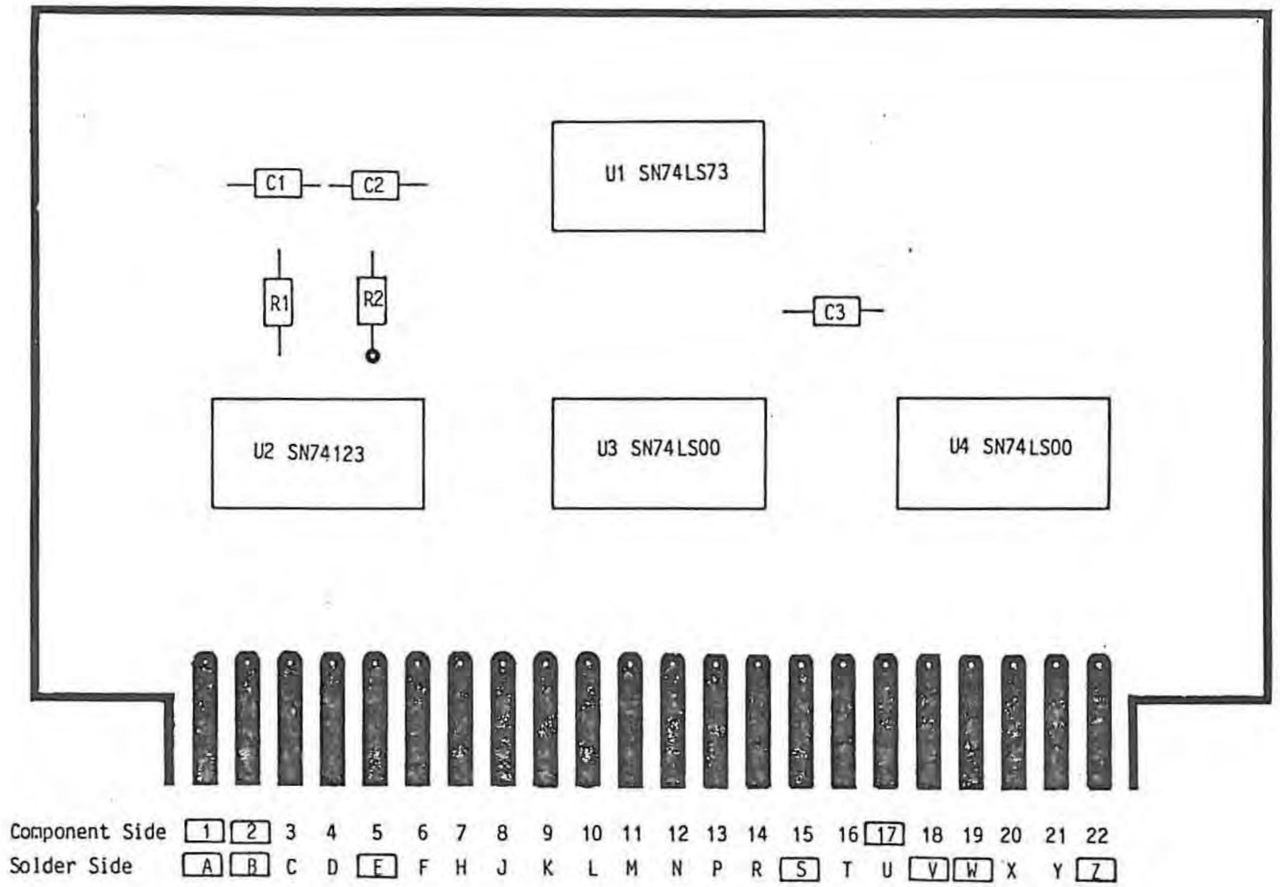


Notes:

Integrated Circuit		Vcc pin	Gnd pin
U1	SN74LS73	4	11
U2	SN74123	16	8
U3,U4	SN74LS00	14	7

All resistors in ohms
All capacitors in microfarads

I/R Gain Weight Generator - replaces Amplifier I/R board A6



Notes:

= Feedthrough

Jn Jn = Jumper n

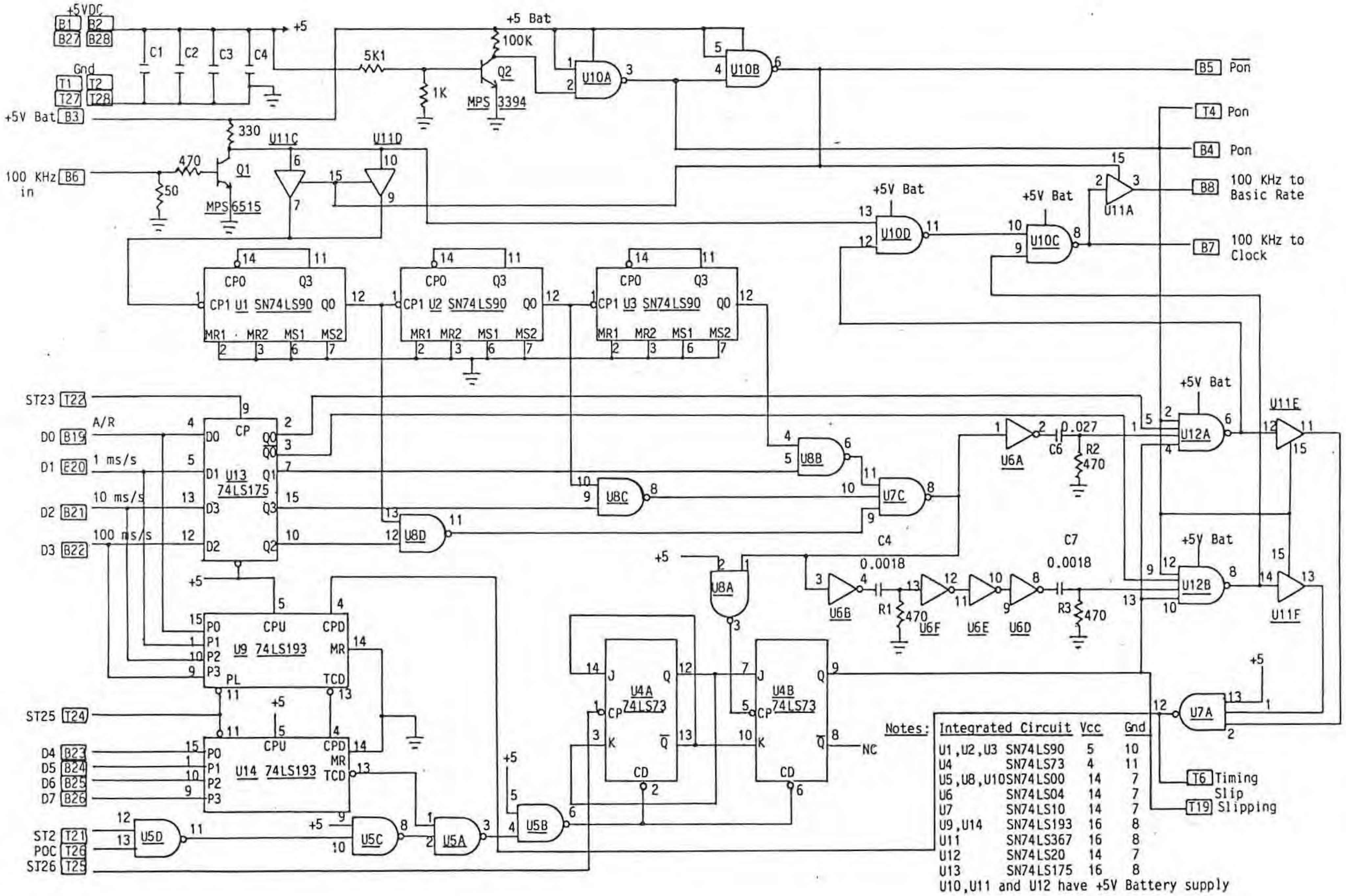
T/R Gain Weight Generator - replaces Amplifier T/R board A6 - Component Side

Pin No.	Signal Name and Function	Direction	From/to	Pin No.
1	+5VDC supply	<--	chassis	-
2	gnd.	<--	chassis	-
3	-	-	-	-
4	-	-	-	-
5	-	-	-	-
6	-	-	-	-
7	-	-	-	-
8	-	-	-	-
9	-	-	-	-
10	-	-	-	-
11	-	-	-	-
12	-	-	-	-
13	-	-	-	-
14	-	-	-	-
15	-	-	-	-
16	-	-	-	-
17	Rx Gain Weight out	-->	A7	J
18	-	-	-	-
19	-	-	-	-
20	-	-	-	-
21	-	-	-	-
22	-	-	-	-

T/R Gain Weight Generator - replaces Amplifier T/R board A6 - Solder Side

Pin No.	Signal Name and Function	Direction	From/to	Pin No.
A	+5VDC supply	<--	chassis	-
B	gnd.	<--	chassis	-
C	-	-	-	-
D	-	-	-	-
E	M-sequence in	<--	J7	10
F	-	-	-	-
H	-	-	-	-
J	-	-	-	-
K	-	-	-	-
L	-	-	-	-
M	-	-	-	-
N	-	-	-	-
P	-	-	-	-
R	-	-	-	-
S	Tx Gain weight	-->	A5	C
T	-	-	-	-
U	-	-	-	-
V	SC0 Tx , Rx and T/R control	<--	J7	8
W	SC1	<--	J7	9
X	-	-	-	-
Y	-	-	-	-
Z	T/R pulse train	-->	A7	D

Advance Retard . PC5

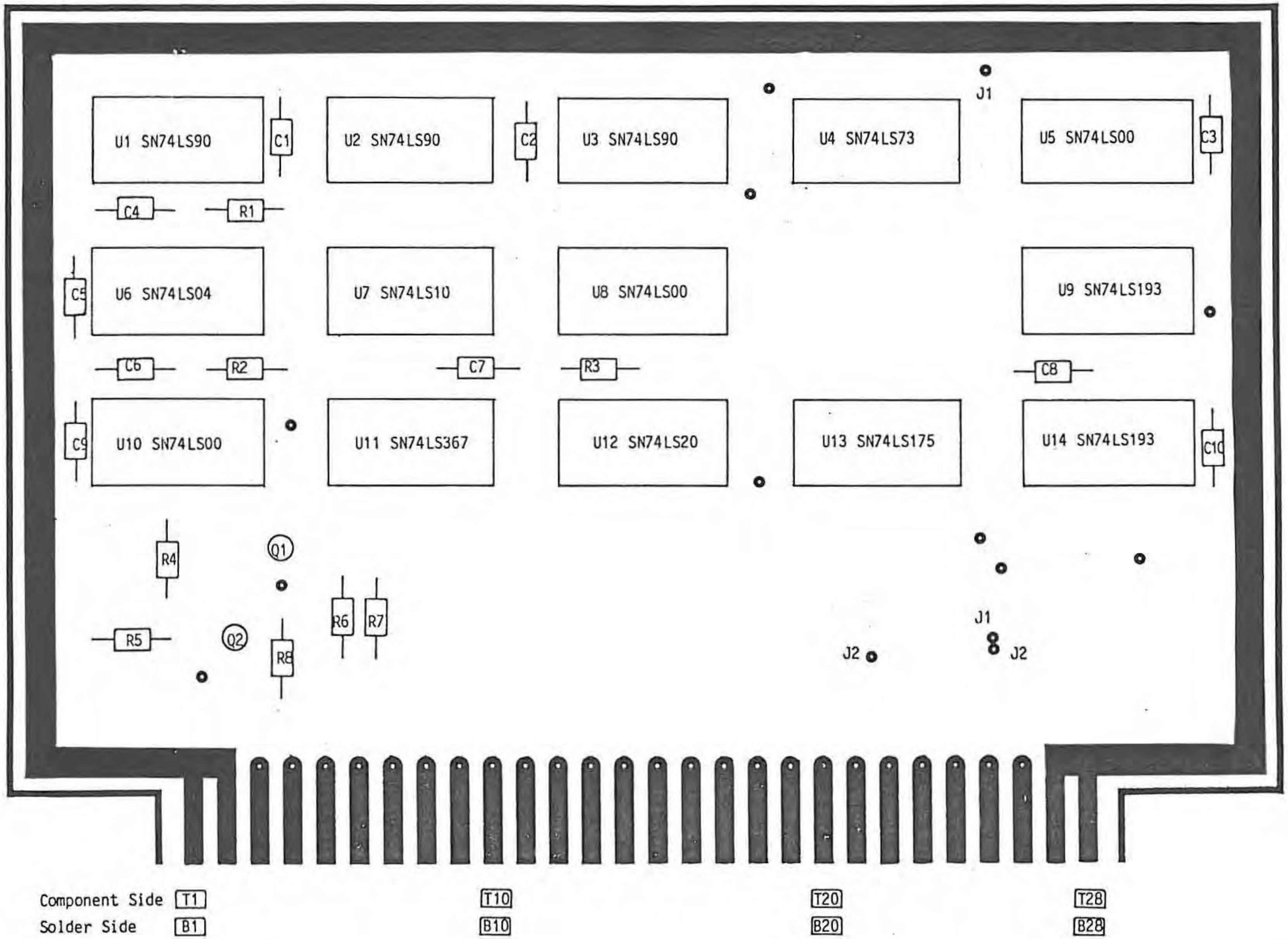


Notes:

Integrated Circuit		Vcc	Gnd
U1,U2,U3	SN74LS90	5	10
U4	SN74LS73	4	11
U5,U8,U10	SN74LS00	14	7
U6	SN74LS04	14	7
U7	SN74LS10	14	7
U9,U14	SN74LS193	16	8
U11	SN74LS367	16	8
U12	SN74LS20	14	7
U13	SN74LS175	16	8

U10,U11 and U12 have +5V Battery supply

Advance Retard , PC5



Notes:

● = Feedthrough

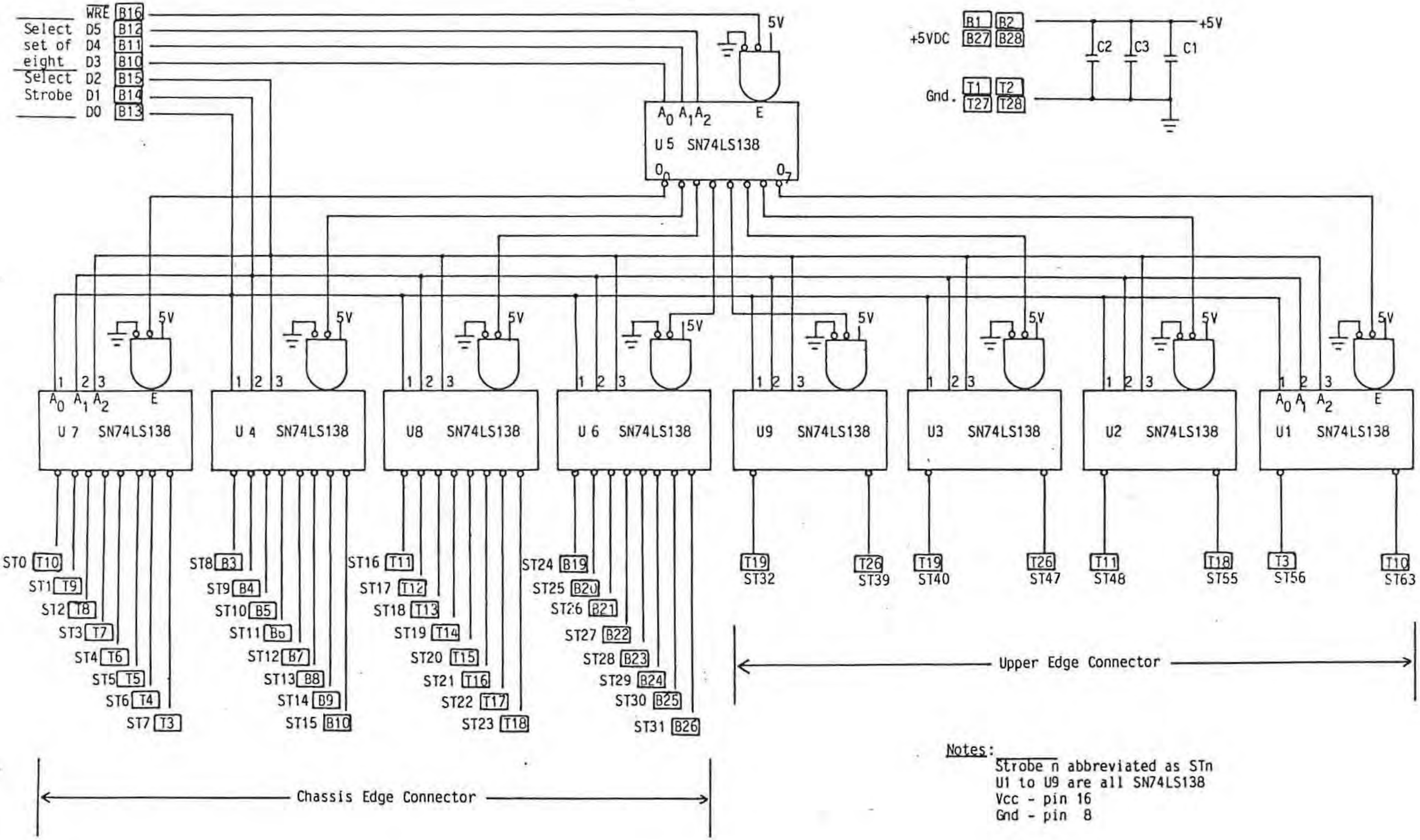
Jn Jn = Jumper n

Advance Retard , PC5 - Component Side (T)

Pin No.	Signal Name and Function	Direction	From/to	Pin No.
T1,T2	gnd	<--	chassis	-
T3	-	-	-	-
T4	Pon	<--	PC5	B4
T5	-	-	-	-
T6	Timing Slip pulses adv. or rtd.	-->	BNC3	-
T7	-	-	-	-
T8	-	-	-	-
T9	-	-	-	-
T10	-	-	-	-
T11	-	-	-	-
T12	-	-	-	-
T13	-	-	-	-
T14	-	-	-	-
T15	-	-	-	-
T16	-	-	-	-
T17	-	-	-	-
T18	-	-	-	-
T19	slipping	-->	PC3	B4
T20	-	-	-	-
T21	ST2 Clear A/R flip-flop	<--	PC6	T8
T22	ST23 load A/R control data	<--	PC6	T18
T23	-	-	-	-
T24	ST25 load A/R shift data	<--	PC6	B20
T25	ST26 A/R shift start	<--	PC6	B21
T26	$\overline{\text{POC}}$	<--	PC3	T6
T27,T28	gnd	<--	chassis	-

Advance/Retard , PC5 - Solder Side (B)

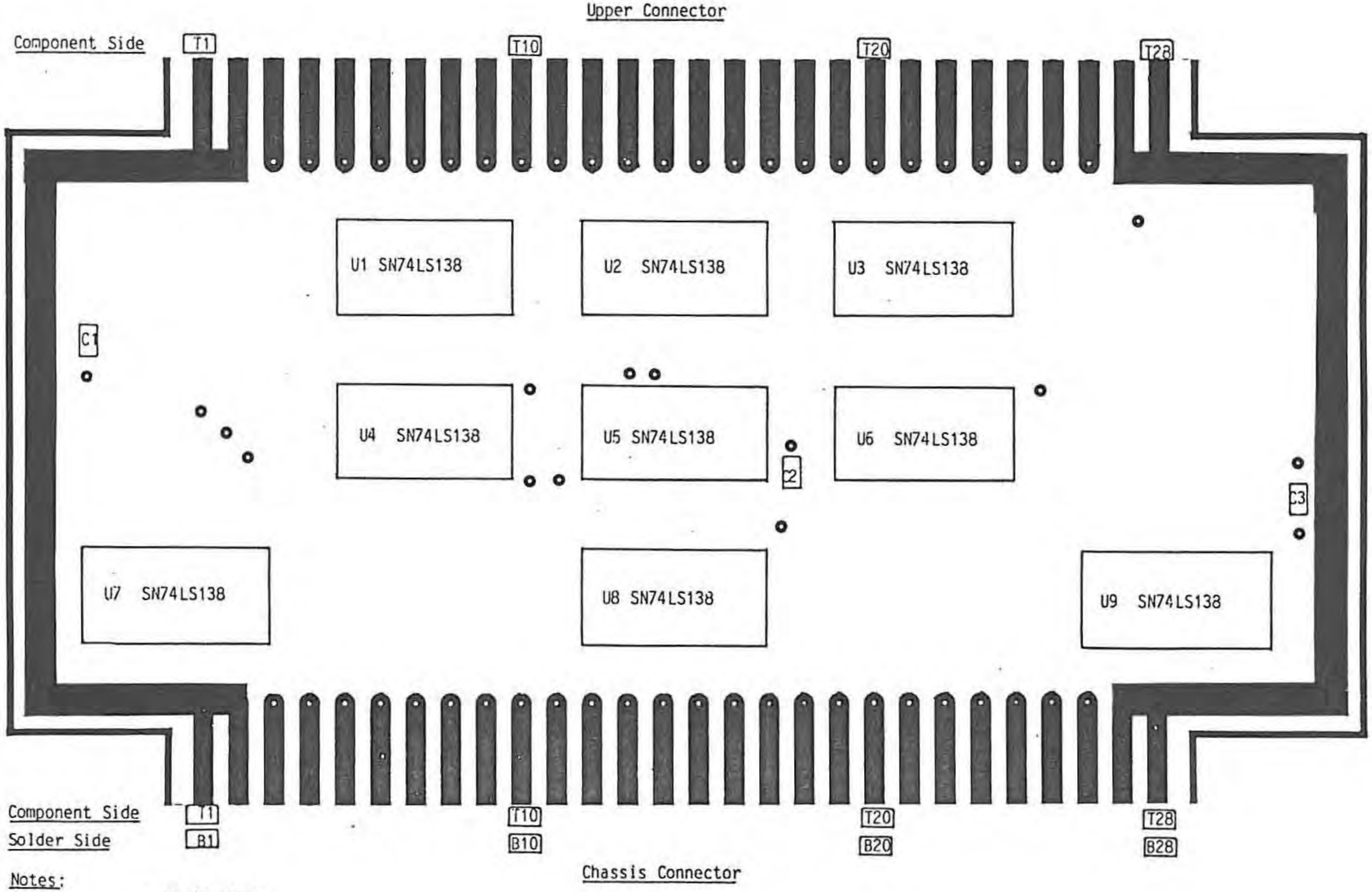
Pin No.	Signal Name and Function	Direction	From/to	Pin No.
B1,B2	+5VDC supply	<--	chassis	-
B3	+5V Bat Battery supply	<--	Diode PCB	-
B4	Pon	--> -->	PC5 S-32 memory	T4 -
B5	$\overline{\text{Pon}}$	-->	-	-
B6	100 kHz in from Q.O.	<--	BNC8	-
B7	100 kHz to clock	-->	PC3	B10
B8	100 kHz to Basic Rate	-->	PC4	B14
B9	-	-	-	-
B10	-	-	-	-
B11	-	-	-	-
B12	-	-	-	-
B13	-	-	-	-
B14	-	-	-	-
B15	-	-	-	-
B16	-	-	-	-
B17	-	-	-	-
B18	-	-	-	-
B19	D0 Controller data bus	<--		
B20	D1	<--		
B21	D2	<--		
B22	D3	<--		
B23	D4	<--		
B24	D5	<--		
B25	D6	<--		
B26	D7	<--		
B27,B27	+5VDC supply	<--	chassis	-



Notes:
Strobe n abbreviated as STn
U1 to U9 are all SN74LS138
Vcc - pin 16
Gnd - pin 8

Strobe Decode . PC6

Strobe Decode , PC6



Notes:

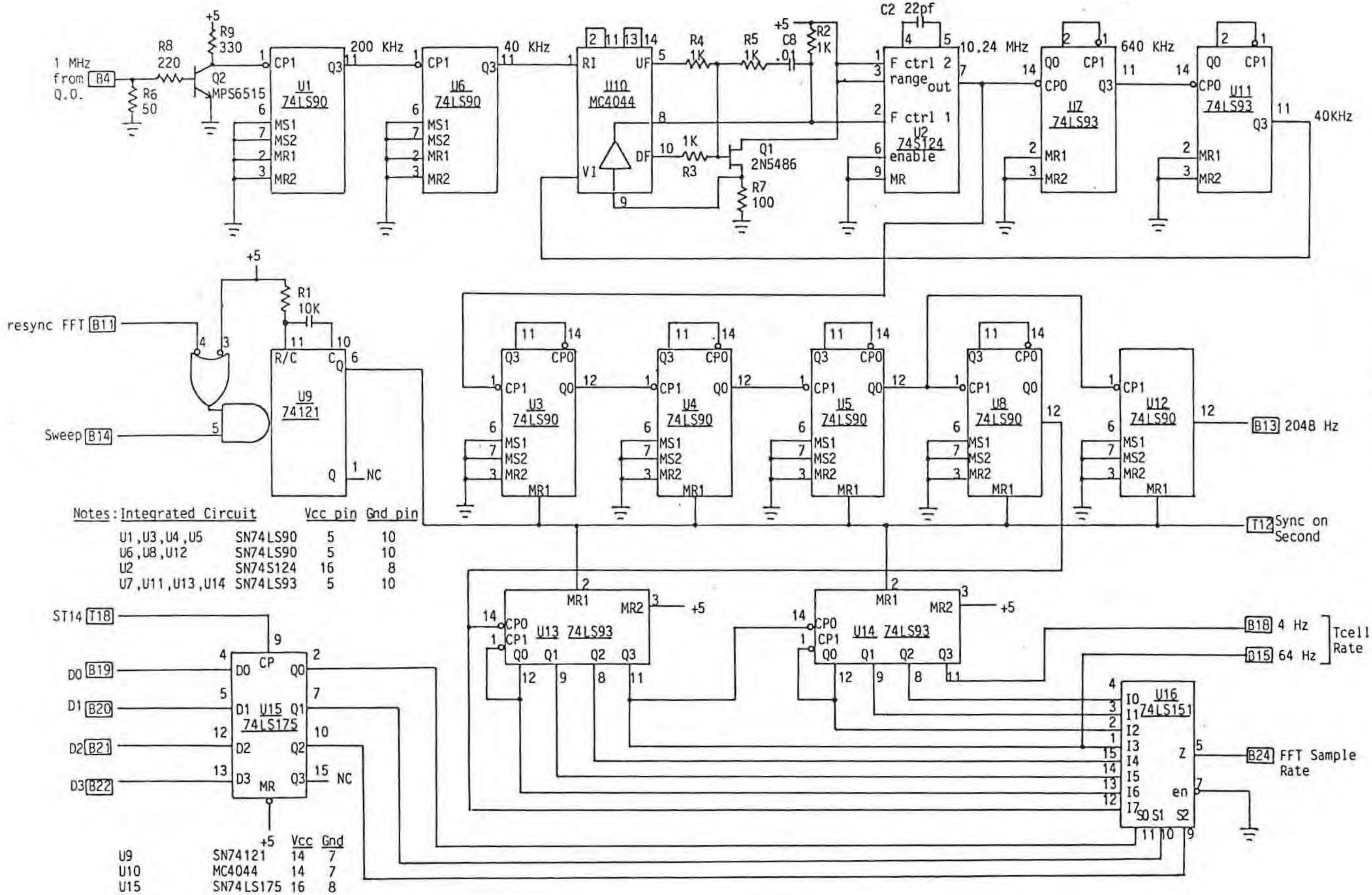
= Feedthrough
Jn Jn = Jumper n

Strobe Decode , PC6 - Component Side (T)

Pin No.	Signal	Name and Function	Direction	From/to	Pin No.
T1,T2	gnd		<--	chassis	-
T3	ST7	synth latch 10^4,10^5	-->	PC1	T26
T4	ST6	synth latch 10^2,10^3	-->	PC1	T3
T5	ST5	synth latch 10^0,10^1	-->	PC1	T4
T6	ST4	load Control Register B	-->	PC3	B26
T7	ST3	load Control Register A	-->	PC3	B25
T8	ST2	clear A/R flip-flop	-->	PC5	T21
T9	ST1	clear sweep flip-flop	-->	PC3	T20
T10	ST0	clear sweep,A/R & film dr f/f	-->	PC3	T5
T11	ST16	antenna data initial load	-->	PC3	T15
T12	ST17	load film speed data	-->	PC8	B21
T13	ST18	Rx AGC ext clock,data to DCS	-->	RPA4	14
T14	ST19	Rx Phase sync. ,data to DCS	-->	RPA4	12
T15	ST20	load LO offset select data	-->	PC9	T26
T16	ST21	load LO offset data	-->	PC9	T17
T17	ST22	Sweep Start	-->	PC3	B19
T18	ST23	load A/R control data	-->	PC5	T22
T19	ST32	spare	-->		
T20	ST33	spare	-->		
T21	ST34	spare	-->		
T22	ST35	spare	-->		
T23	ST36	spare	-->		
T24	ST37	spare	-->		
T25	ST38	spare	-->		
T26	ST39	spare	-->		
T27,T28	gnd		<--	chassis	-

Strobe Decode , PC6 - Solder Side (B)

Pin No.	Signal	Name and Function	Direction	From/to	Pin No.
B1,B2	+5VDC supply		<--	chassis	-
B3	ST8	synth latch 10^6,10^7	-->	PC1	T25
B4	ST9	synth counter initial load	-->	PC3	T17
B5	ST10	Basic Rate data load	-->	PC4	B7
B6	ST11	Tcell pulse period data load	-->	PC4	B9
B7	ST12	M-Sequence data load	-->	PC4	B25
B8	ST13	not used	-->		
B9	ST14	FFT data load	-->	PC7	T18
B10	ST15	load antenna latches U1,U2	-->	PC8	B11
B11	D3	Strobe Select data	<--	MP-LA-1B	B3
B12	D4		<--	MP-LA-1B	B4
B13	D5		<--	MP-LA-1B	B5
B14	D0		<--	MP-LA-1B	B0
B15	D1		<--	MP-LA-1B	B1
B16	D2		<--	MP-LA-1B	B2
B17	WRE	write strobe with E restore	<--	MP-LA-1B	CB2
B18	-		-	-	-
B19	ST24	spare	-->		
B20	ST25	load A/R shift data	-->	PC5	T24
B21	ST26	A/R shift start	-->	PC5	T25
B22	ST27	Time Mark for stat Doppler	-->	PC2	B13
B23	ST28	baseband filter ctrl (unused)	-->		
B24	ST29	spare	-->		
B25	ST30	spare	-->		
B26	ST31	spare	-->		
B27,B28	+5VDC supply		<--	chassis	-

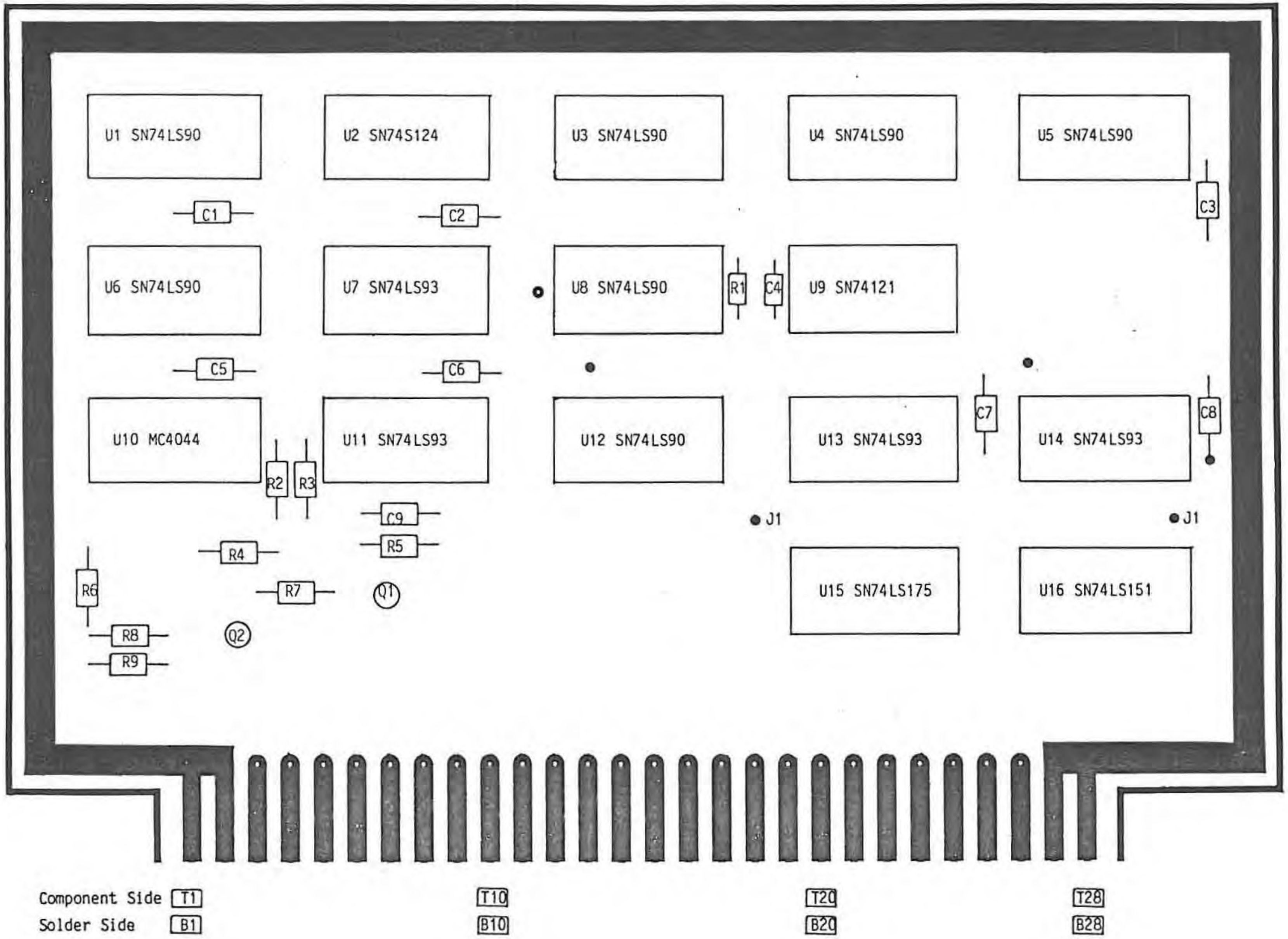


Notes:

Integrated Circuit		Vcc pin	Gnd pin
U1,U3,U4,U5	SN74LS90	5	10
U6,U8,U12	SN74LS90	5	10
U2	SN74S124	16	8
U7,U11,U13,U14	SN74LS93	5	10

		Vcc	Gnd
U9	SN74121	14	7
U10	MC4044	14	7
U15	SN74LS175	16	8
U16	SN74LS151	16	8

FFT Sample Rate , PC7



Notes:

● = Feedthrough

Jn Jn = Jumper n

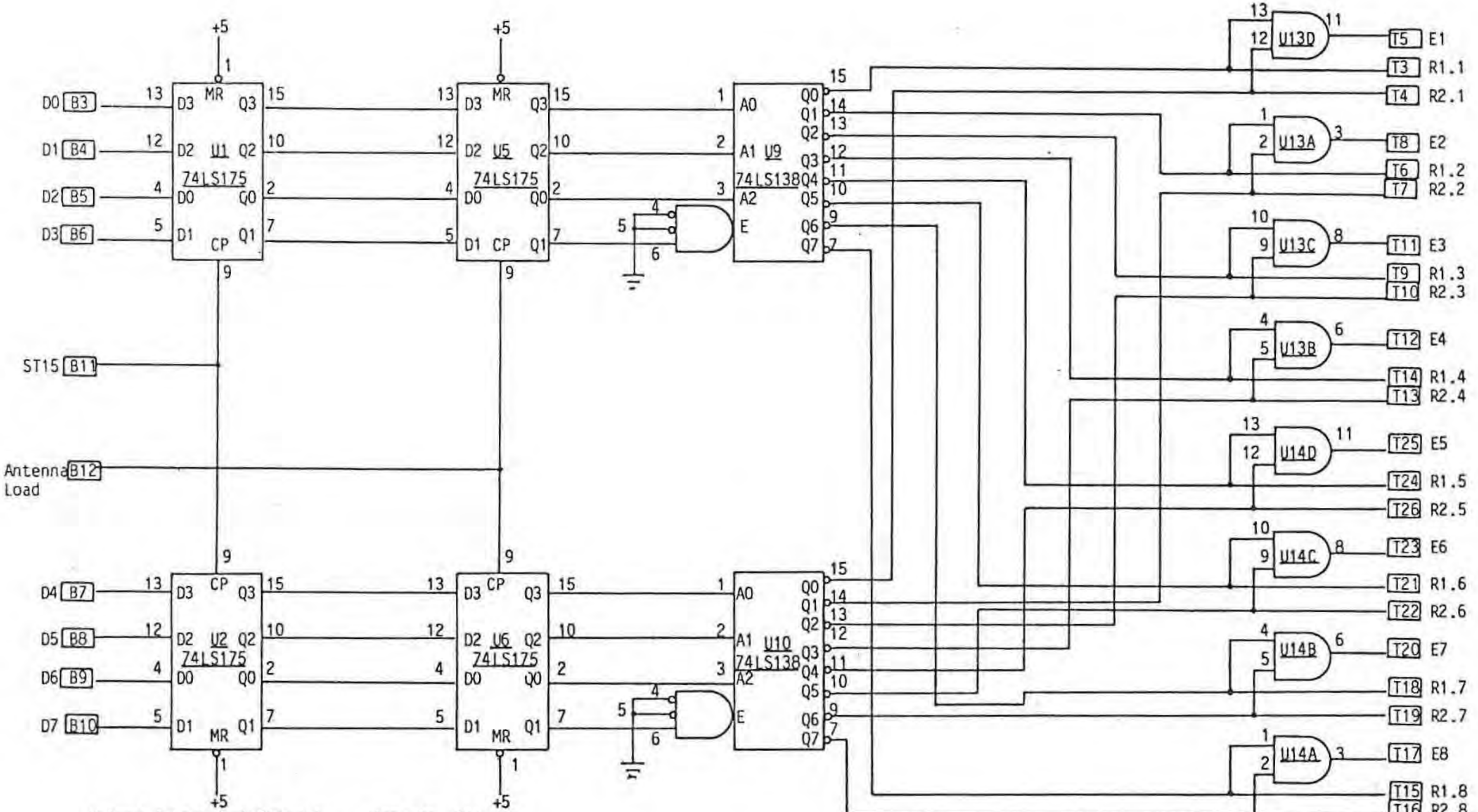
FFT Sample Rate , PC7

FFT Sample Rate , PC7 - Component Side (T)

Pin No.	Signal Name and Function	Direction	From/to	Pin No.
T1,T2	gnd	<--	chassis	-
T3	-	-	-	-
T4	1 MHz to Basic Rate	-->	PC4	B13
T5	-	-	-	-
T6	-	-	-	-
T7	-	-	-	-
T8	-	-	-	-
T9	-	-	-	-
T10	10.24 MHz	-->	PC9	B3
T11	-	-	-	-
T12	Sync on second	--> -->	PC4 PC2	B10 B14
T13	-	-	-	-
T14	-	-	-	-
T15	-	-	-	-
T16	-	-	-	-
T17	-	-	-	-
T18	ST14 FFT data load	<--	PC6	B9
T19	-	-	-	-
T20	-	-	-	-
T21	-	-	-	-
T22	-	-	-	-
T23	-	-	-	-
T24	-	-	-	-
T25	-	-	-	-
T26	-	-	-	-
T27,T28	gnd	<--	chassis	-

FFT Sample Rate , PC7 - Solder Side (B)

Pin No.	Signal Name and Function	Direction	From/to	Pin No.
B1,B2	+5VDC supply	<--Red	chassis	-
B3	-	-	-	-
B4	1 MHz sine wave from 105B Q.O.	<--	BNC6	-
B5	-	-	-	-
B6	-	-	-	-
B7	-	-	-	-
B8	-	-	-	-
B9	-	-	-	-
B10	-	-	-	-
B11	resync FFT gen	<--	PC3	B20
B12	-	-	-	-
B13	2048 Hz	-->	PC4	B16
B14	Sweep	<--	PC3	B16
B15	64 Hz	-->	PC3	T4
B16	128 Hz	---		
B17	256 Hz	---		
B18	4 Hz	-->	PC3	T3
B19	D0	<--		
B20	D1	<--		
B21	D2	<--		
B22	D3	<--		
B23	-	-	-	-
B24	FFT Sample Rate	-->	RPA4	38
B25	-	-	-	-
B26	-	-	-	-
B27,B28	+5VDC supply	<--Red	chassis	-

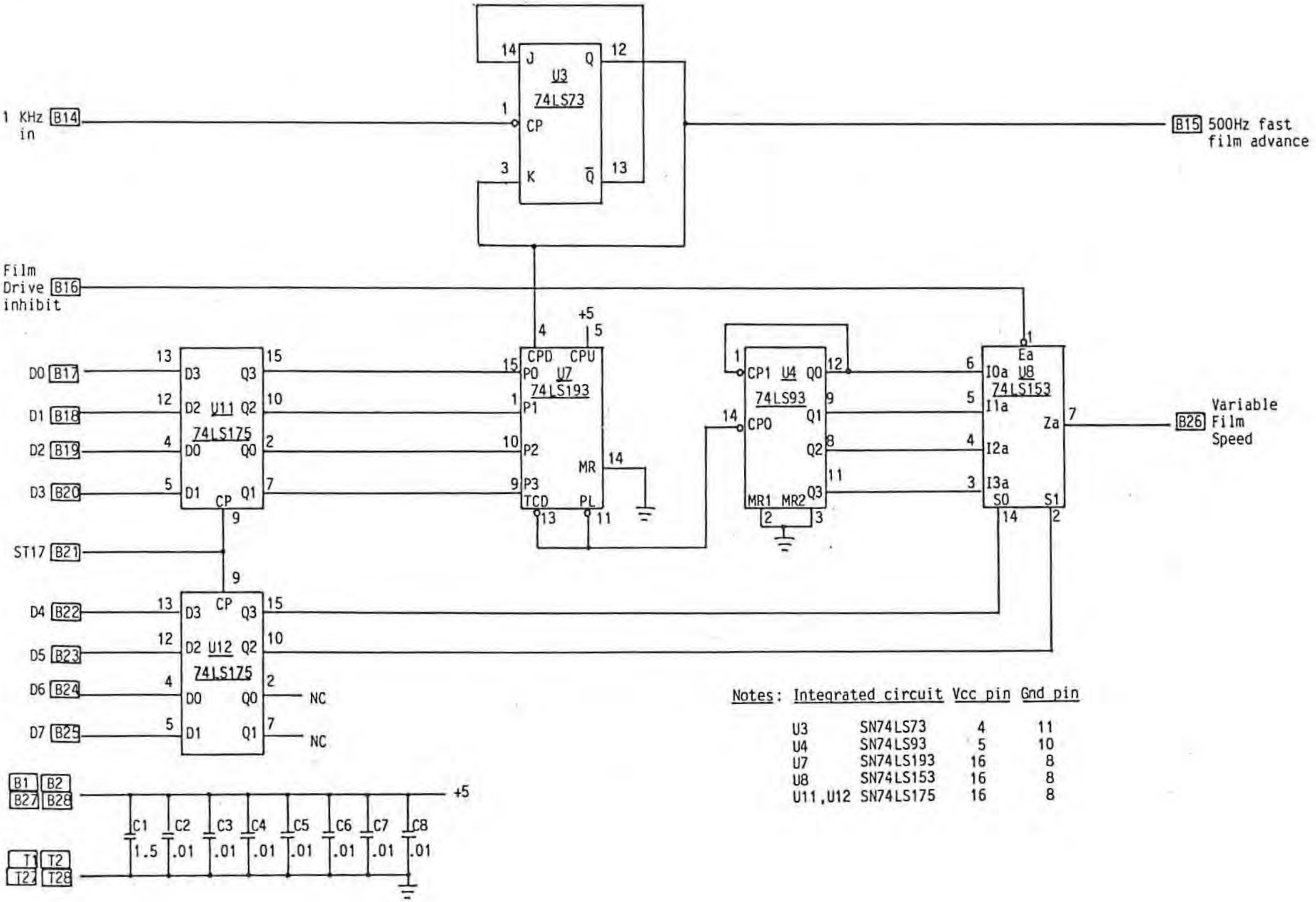


Antenna Select , PC8a

Notes:

Integrated Circuit		Vcc pin	Gnd pin
U1,U2,U5,U6	SN74LS175	16	8
U9,U10	SN74LS138	16	8
U13,U14	SN74LS08	14	7

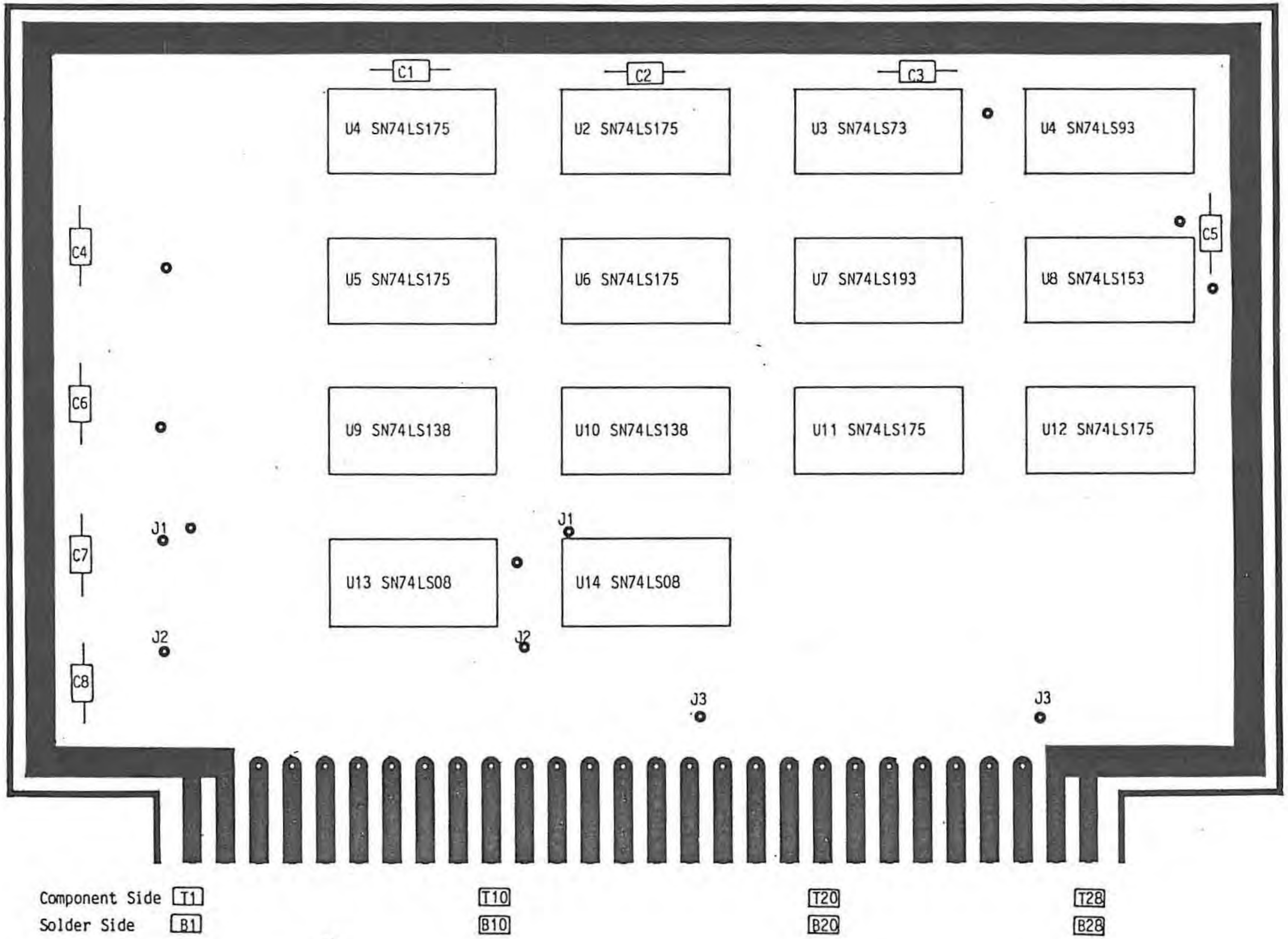
E1 to E8,R1.1 to R1.8 and R2.1 to R2.8 are active low signals
En = enable antenna n
R1.n = antenna n to Rx1
R2.n = antenna n to Rx2



Notes:

Integrated circuit		Vcc pin	Gnd pin
U3	SN74LS73	4	11
U4	SN74LS93	5	10
U7	SN74LS193	16	8
U8	SN74LS153	16	8
U11,U12	SN74LS175	16	8

Film Speed Control , PCBb



Notes:

◎ = Feedthrough

Jn Jn = Jumper n

Antenna Select and Film Speed Control , PCB

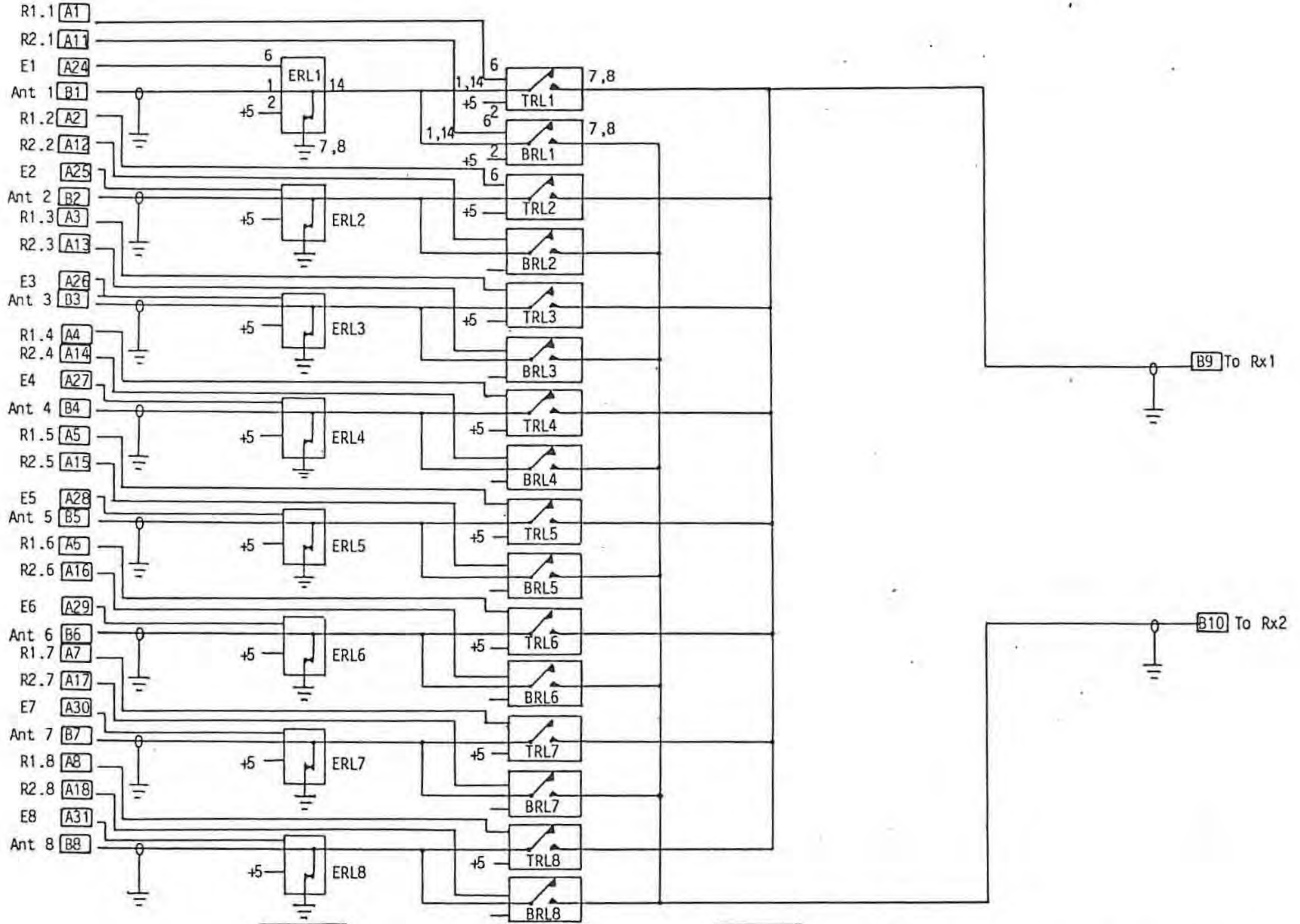
Antenna Select and Film Speed Control , PC8 - Component Side (T)

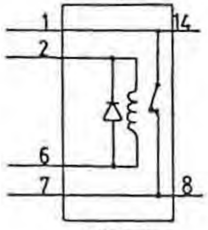
Pin No.	Signal Name and Function	Direction	From/to	Pin No.
T1,T2	gnd	<--Blk	chassis	-
T3	R1.1 Rx1 to ant 1	-->Blk	RPA6	1
T4	R2.1 Rx2 to ant 1	-->Blk	RPA6	11
T5	E1 enable ant 1	-->Blk	RPA6	24
T6	R1.2 Rx1 to ant 2	-->Brn	RPA6	2
T7	R2.2 Rx2 to ant 2	-->Brn	RPA6	12
T8	E2 Enable ant 2	-->Brn	RPA6	25
T9	R1.3 Rx1 to ant 3	-->Red	RPA6	3
T10	R2.3 Rx2 to ant 3	-->Red	RPA6	13
T11	E3 Enable ant 3	-->Red	RPA6	26
T12	E4 Enable ant 4	-->Orn	RPA6	27
T13	R2.4 Rx2 to ant 4	-->Orn	RPA6	14
T14	R1.4 Rx1 to ant 4	-->Orn	RPA6	4
T15	R1.8 Rx1 to ant 8	-->Vio	RPA6	8
T16	R2.8 Rx2 to ant 8	-->Vio	RPA6	18
T17	E8 Enable ant 8	-->Vio	RPA6	31
T18	R1.7 Rx1 to ant 7	-->Blu	RPA6	7
T19	R2.7 Rx2 to ant 7	-->Blu	RPA6	17
T20	E7 Enable ant 7	-->Blu	RPA6	30
T21	R1.6 Rx1 to ant 6	-->Grn	RPA6	6
T22	R2.6 Rx2 to ant 6	-->Grn	RPA6	16
T23	E6 Enable ant 6	-->Grn	RPA6	29
T24	R1.5 Rx1 to ant 5	-->Yel	RPA6	5
T25	E5 Enable ant 5	-->Yel	RPA6	28
T26	R2.5 Rx2 to ant 5	-->Yel	RPA6	15
T27,T28	gnd	<--Blk	chassis	-

Antenna Select and Film Speed Control , PC8 - Solder Side (B)

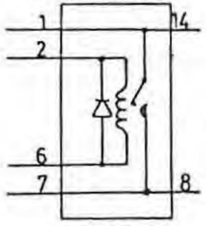
Pin No.	Signal Name and Function	Direction	From/to	Pin No.
B1,B2	+5VDC supply	<--Red	chassis	-
B3	D0 Controller data bus	<--		
B4	D1	<--		
B5	D2	<--		
B6	D3	<--		
B7	D4	<--		
B8	D5	<--		
B9	D6	<--		
B10	D7	<--		
B11	ST15 load ant latches U1,U2	<--	PC6	B10
B12	Antenna load loads U5,U6	<--	PC3	T16
B13	-	-	-	-
B14	1 kHz in	<--	PC3	B9
B15	500 Hz fast film advanve	-->	RPA3	1
B16	Film drive inhibit	<--	PC3	T10
B17	D0 Controller data bus	<--		
B18	D1	<--		
B19	D2	<--		
B20	D3	<--		
B21	ST17 load film speed data	<--	PC6	T12
B22	D4 Controller data bus	<--		
B23	D5	<--		
B24	D6	<--		
B25	D7	<--		
B26	Variable Film Speed	-->	RPA3	2
B27,B28	+5VDC supply	<--Red	chassis	-

Antenna Switch



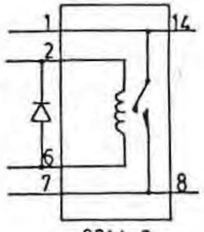


831B-4



831A-4

or



831A-3

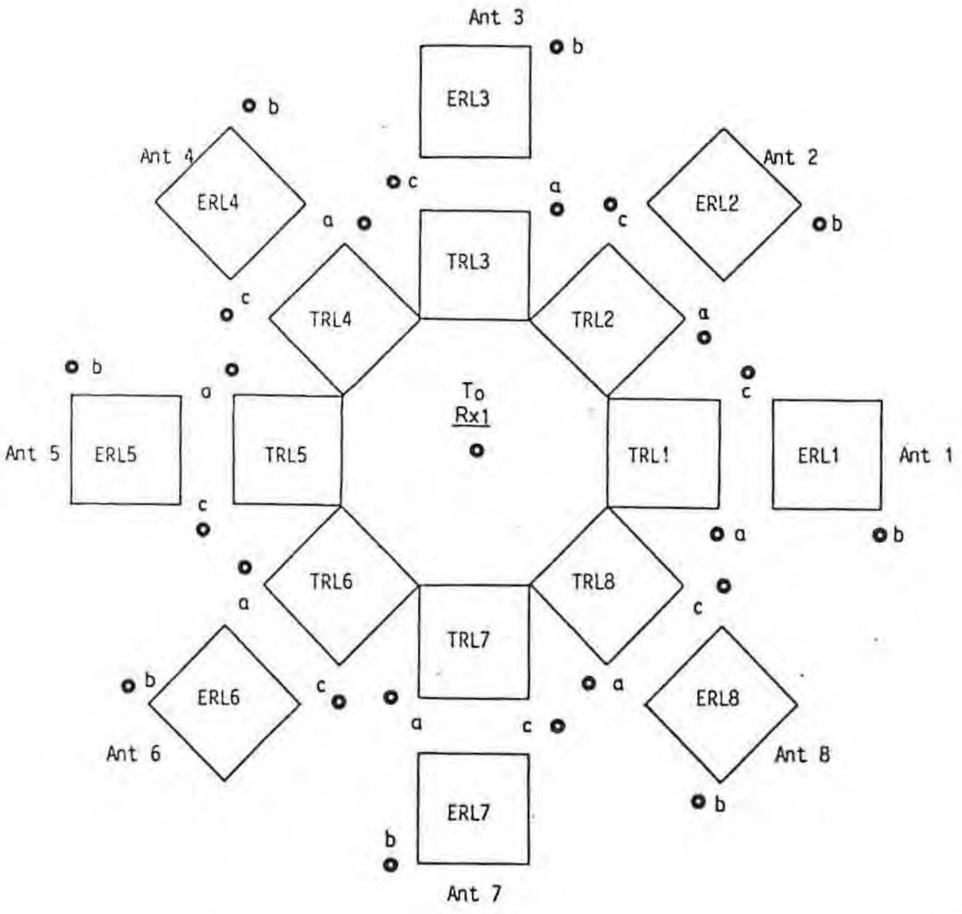
Note:

Normally open relays are either 831A-4 or 831A-3 with an external diode.

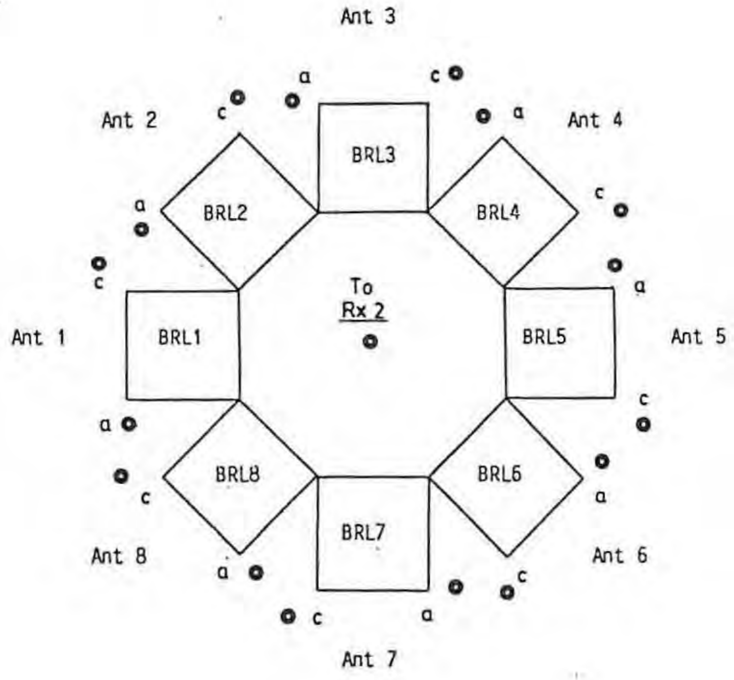
A1 to A31 are amphenol socket pin numbers

B1 to B10 are BNC connectors

Antenna Switch - Top and Bottom PC Boards



Antenna Switch top board - Signal Switching to Rx1 and Antenna Earth Control

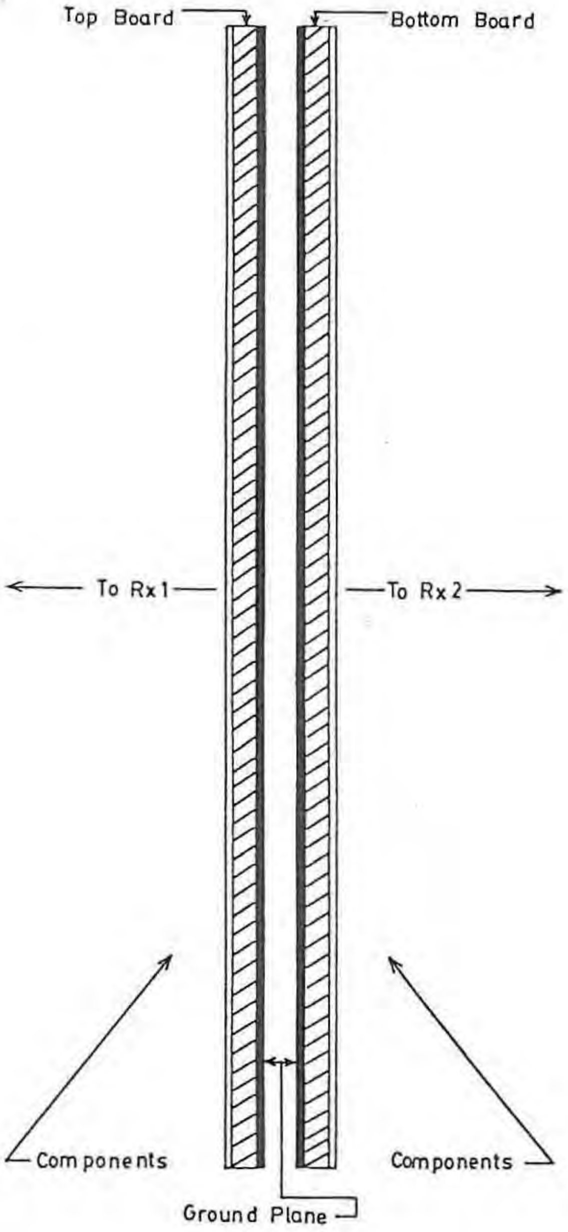


Antenna Switch bottom board - Signal Switching to Rx2

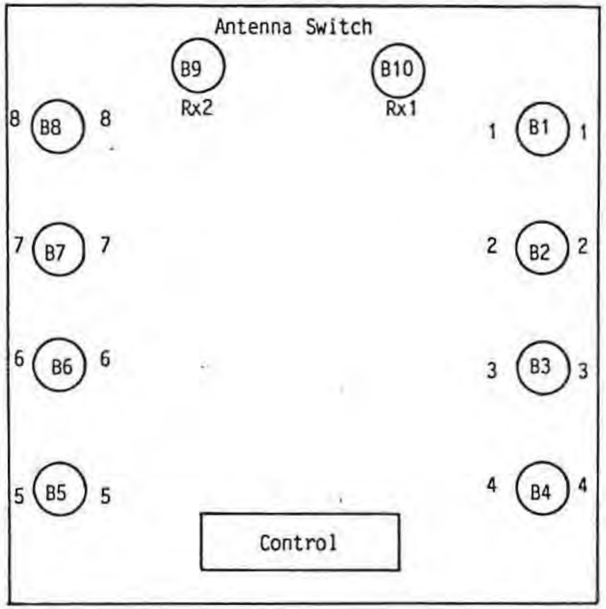
Notes:

a = Antenna select control
b = Antenna earthing control
c = Signal from antenna
TRLn and BRLn relays are 831A-4 or 831A-3 with ext diode.
ERLn relays are all 831B-4
= Feedthrough

Antenna Switch PC Board and Front Panel



Antenna Switch PC Board Locations



B1 to B10 are BNC connectors

Control = cable from RPA6

Antenna Switch Brass Box

Antenna Switch Control Signals

Rear Panel 36 way Amphenol connector RPA6 and Antenna Switch Brass Box , B1

Pin No.	Signal Name	Colour	Pin No.	Signal Name	Colour
1	R1.1	Blk	19	gnd	Blk
2	R1.2	Brn	20	gnd	Blk
3	R1.3	Red	21	gnd	Blk
4	R1.4	Orn	22	-	-
5	R1.5	Yel	23	-	-
6	R1.6	Grn	24	E1	Blk
7	R1.7	Blu	25	E2	Brn
8	R1.8	Vio	26	E3	Red
9	-	-	27	E4	Orn
10	-	-	28	E5	Yel
11	R2.1	Blk	29	E6	Grn
12	R2.2	Brn	30	E7	Blu
13	R2.3	Red	31	E8	Vio
14	R2.4	Orn	32	-	-
15	R2.5	Yel	33	-	-
16	R2.6	Grn	34	+5VDC	Red
17	R2.7	Blu	35	+5VDC	Red
18	R2.8	Vio	36	+5VDC	Red

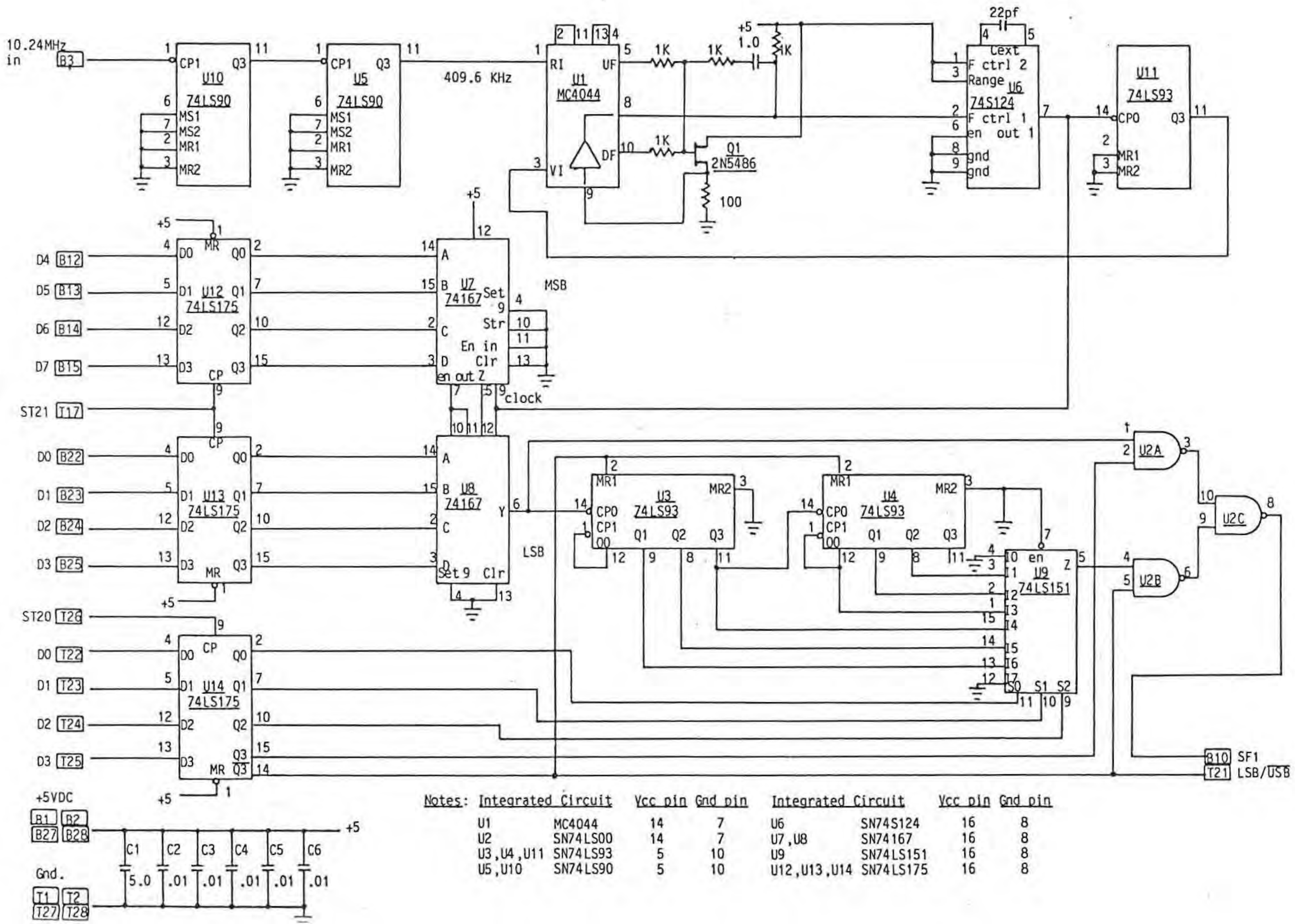
Notes:

The colours listed above are used from

(a) Antenna Select , PC8a to Rear Panel Amphenol RPA6.

(b) Cable between RPA6 and Antenna Switch box B1.

(c) Antenna Switch B1 amphenol connector to antenna switch PC boards.

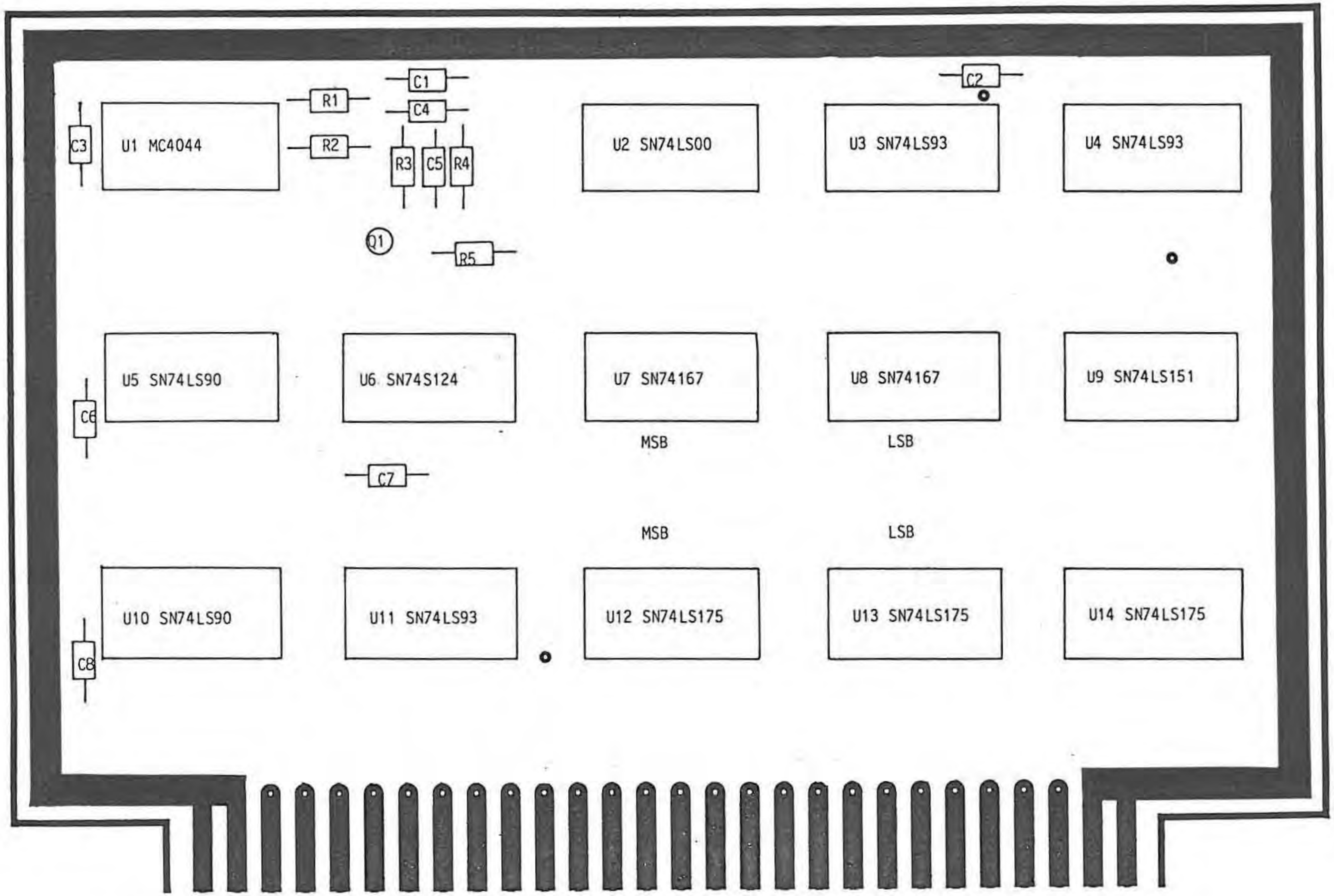


Notes:

Integrated Circuit		Vcc pin	Gnd pin	Integrated Circuit		Vcc pin	Gnd pin
U1	MC4044	14	7	U6	SN74S124	16	8
U2	SN74LS00	14	7	U7,U8	SN74167	16	8
U3,U4,U11	SN74LS93	5	10	U9	SN74LS151	16	8
U5,U10	SN74LS90	5	10	U12,U13,U14	SN74LS175	16	8

Local Oscillator Offset Frequency Synthesizer , PC9

Local Oscillator Offset Frequency Synthesizer , PC9



Component Side	T1	T10	T20	T28
Solder Side	B1	B10	B20	B28

Notes:

● = Feedthrough

Jn Jn = Jumper n

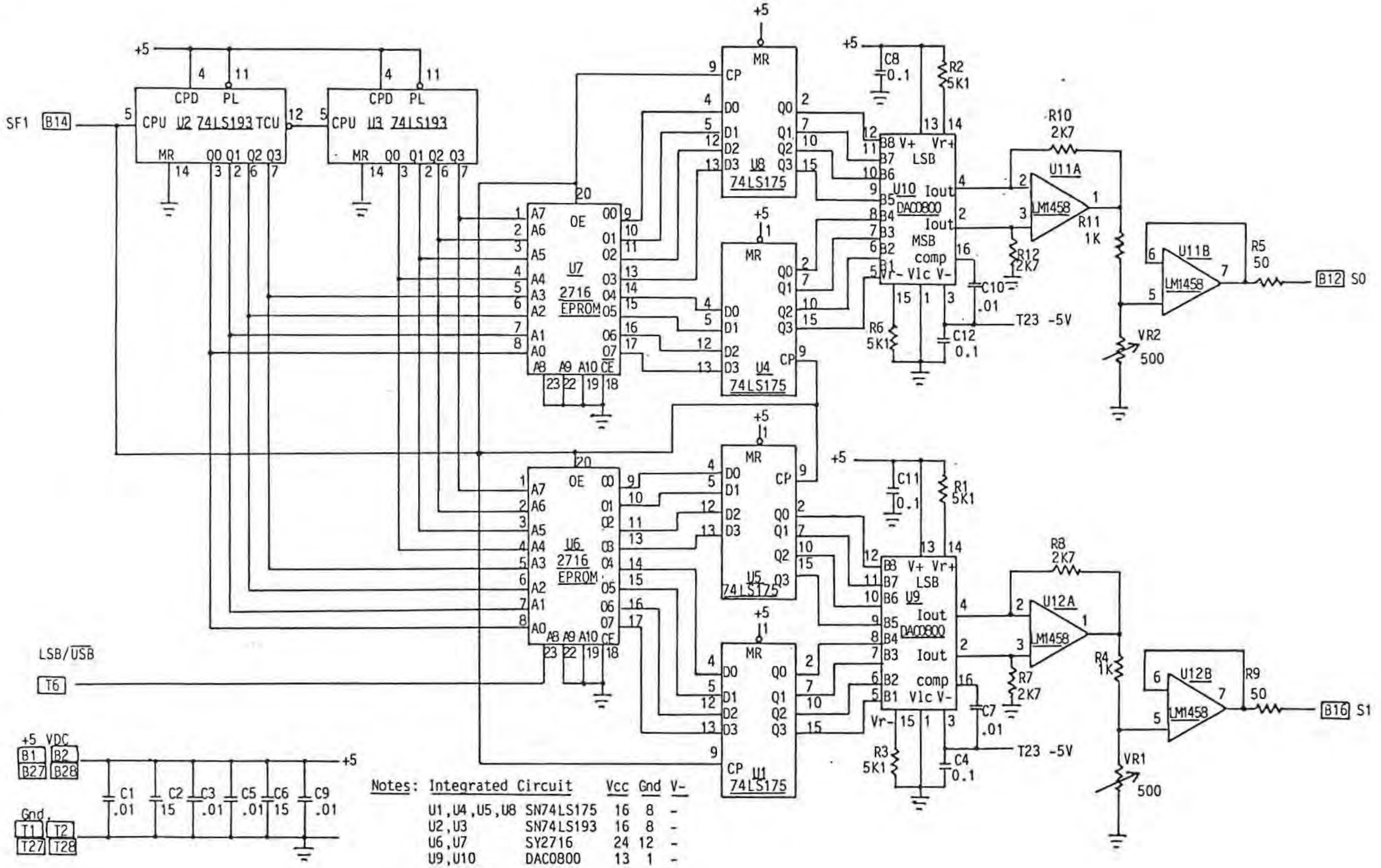
Local Oscillator Offset Frequency Synthesizer , PC9 - Component Side (T)

Pin No.	Signal	Name and Function	Direction	From/to	Pin No.
T1,T2	gnd		<--Blk	chassis	-
T3	-		-	-	-
T4	-		-	-	-
T5	-		-	-	-
T6	-		-	-	-
T7	-		-	-	-
T8	-		-	-	-
T9	-		-	-	-
T10	-		-	-	-
T11	-		-	-	-
T12	-		-	-	-
T13	-		-	-	-
T14	-		-	-	-
T15	-		-	-	-
T16	-		-	-	-
T17	ST21	load LO offset data	<--	PC6	T16
T18	-		-	-	-
T19	-		-	-	-
T20	-		-	-	-
T21	LSB/$\overline{\text{USB}}$	Sideband select	-->	PC10	T6
T22	D0	Controller data bus	<--		
T23	D1		<--		
T24	D2		<--		
T25	D3		<--		
T26	ST20	load lo offset select data	<--	PC6	T15
T27,T28	gnd		<--Blk	chassis	-

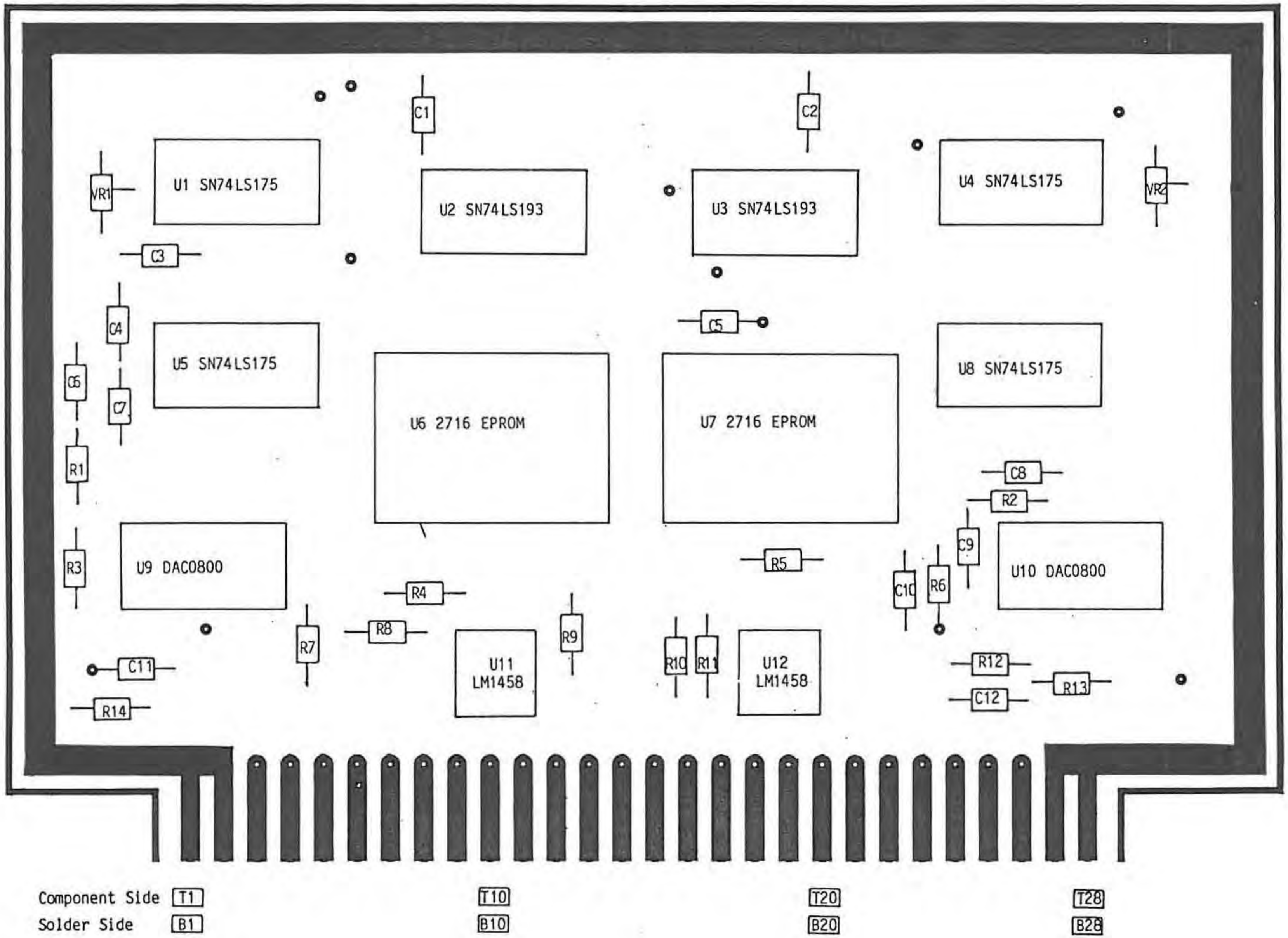
Local Oscillator Offset Frequency Synthesizer , PC9 - Solder Side (B)

Pin No.	Signal	Name and Function	Direction	From/to	Pin No.
B1,B2	+5VDC supply		<--Red	chassis	-
B3	10.24 MHz		<--	PC7	T10
B4	-		-	-	-
B5	-		-	-	-
B6	-		-	-	-
B7	-		-	-	-
B8	-		-	-	-
B9	-		-	-	-
B10	SF1	synthesizer freq 1 = 256 x LO offset freq	-->	PC10	B14
B11	-		-	-	-
B12	D4	Controller data bus	<--		
B13	D5		<--		
B14	D6		<--		
B15	D7		<--		
B16	-		-	-	-
B17	-		-	-	-
B18	-		-	-	-
B19	-		-	-	-
B20	-		-	-	-
B21	-		-	-	-
B22	D0	Controller data bus	<--		
B23	D1		<--		
B24	D2		<--		
B25	D3		<--		
B26	-		-	-	-
B27,B28	+5VDC supply		<--Red	chassis	-

Sine Wave Generator , PC10



Notes: Integrated Circuit		Vcc	Gnd	V-
U1,U4,U5,U8	SN74LS175	16	8	-
U2,U3	SN74LS193	16	8	-
U6,U7	SY2716	24	12	-
U9,U10	DAC0800	13	1	-
U11,U12	LM1458	8	-	4



Notes:

● = Feedthrough

Jn Jn = Jumper n

Sine Wave Generator , PC10

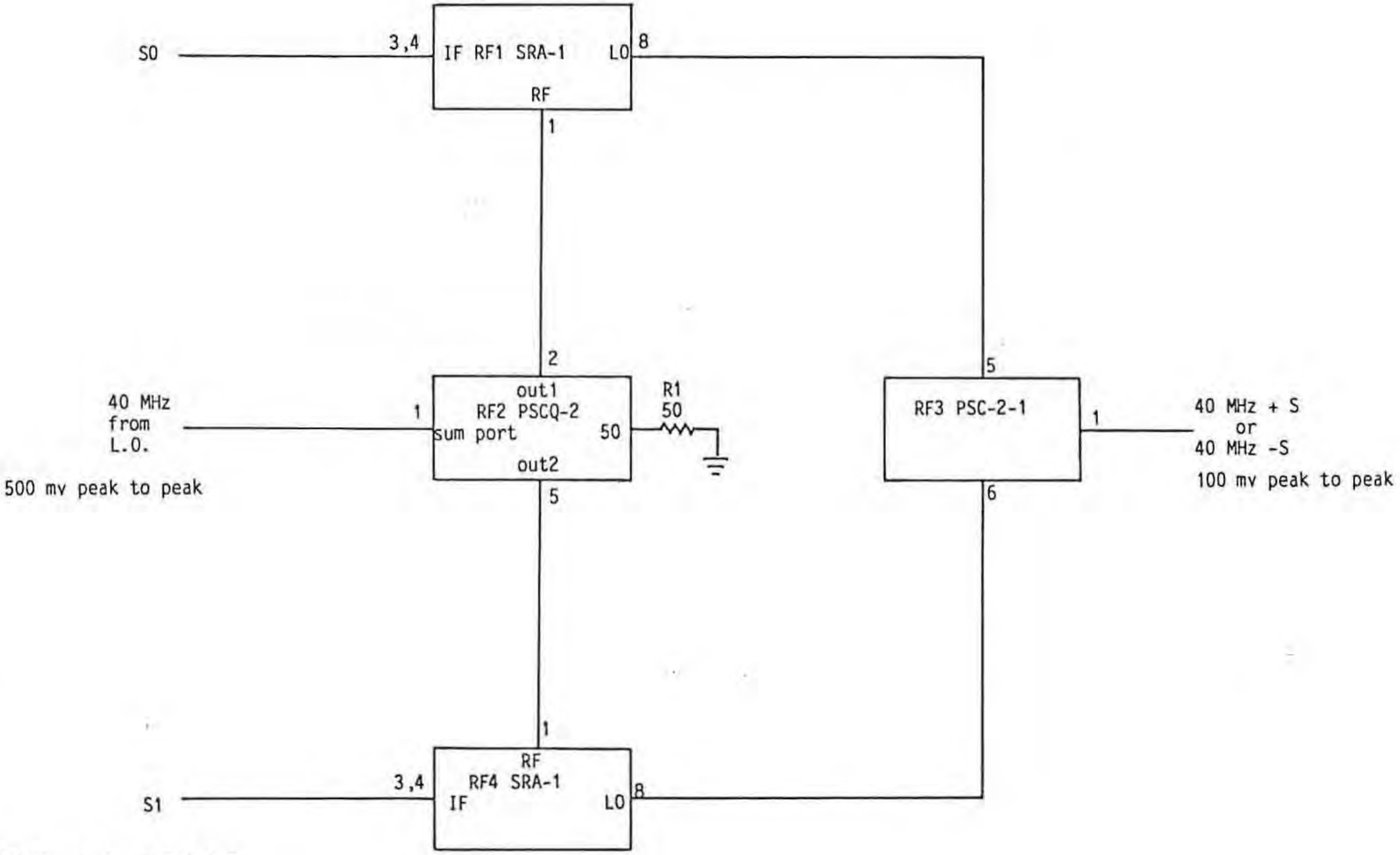
Sine Wave Generator , PC10 - Component Side (T)

Pin No.	Signal Name and Function	Direction	From/to	Pin No.
T1,T2	gnd	<--Blk	chassis	-
T3	-	-	-	-
T4	-	-	-	-
T5	-	-	-	-
T6	LSB/$\overline{\text{USB}}$	<--	PC9	T21
T7	-	-	-	-
T8	-	-	-	-
T9	-	-	-	-
T10	-	-	-	-
T11	-	-	-	-
T12	-	-	-	-
T13	-	-	-	-
T14	-	-	-	-
T15	-	-	-	-
T16	-	-	-	-
T17	-	-	-	-
T18	-	-	-	-
T19	-	-	-	-
T20	-	-	-	-
T21	-	-	-	-
T22	-	-	-	-
T23	-5VDC	<--	MP-B2	-
T24	-	-	-	-
T25	-	-	-	-
T26	-	-	-	-
T27,T28	gnd	<--Blk	chassis	-

Sine Wave Generator , PC10 - Solder Side (B)

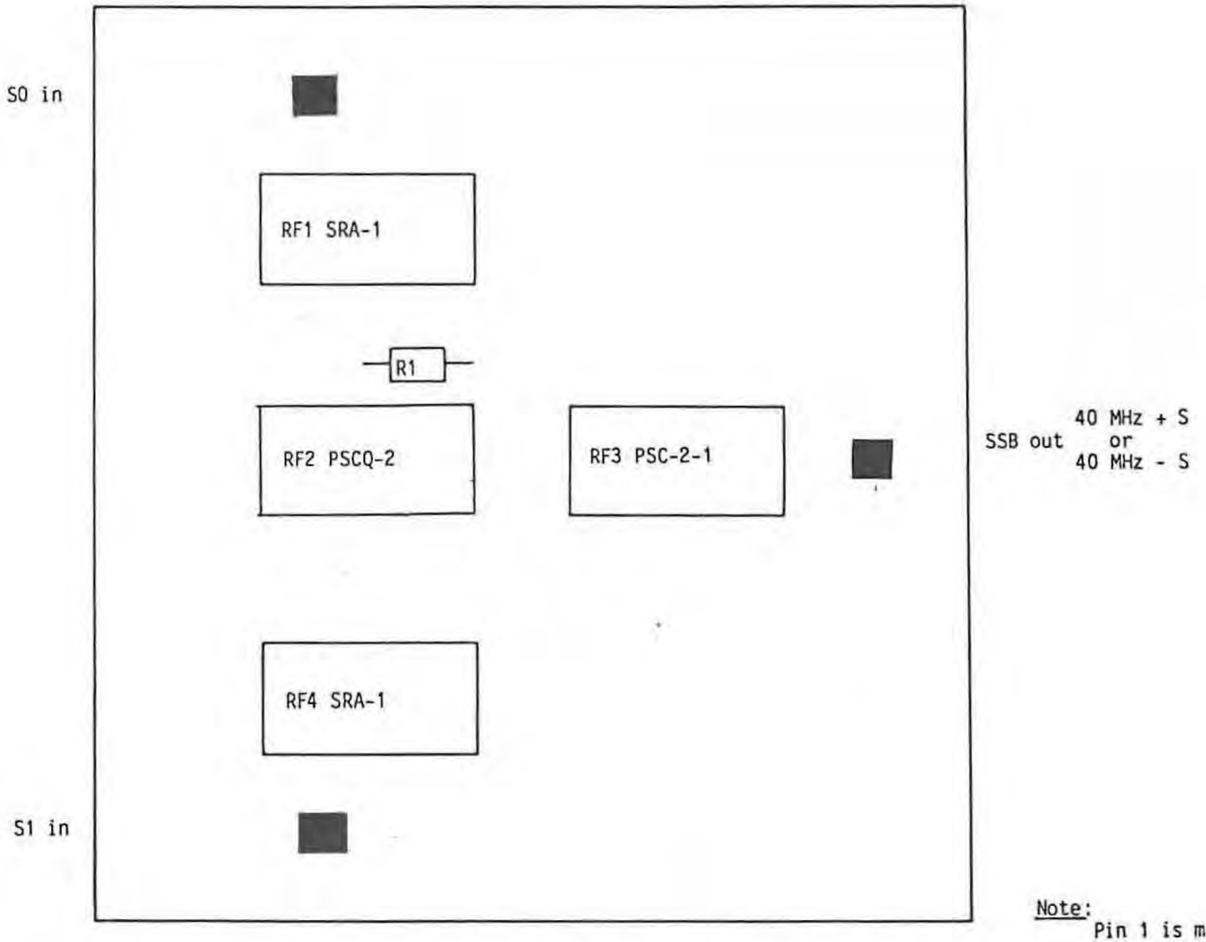
Pin No.	Signal Name and Function	Direction	From/to	Pin No.
B1,B2	+5VDC supply	<--Red	chassis	-
B3	-	-	-	-
B4	-	-	-	-
B5	-	-	-	-
B6	-	-	-	-
B7	-	-	-	-
B8	-	-	-	-
B9	-	-	-	-
B10	-	-	-	-
B11	-	-	-	-
B12	S1 $+90^0$ or -90^0 phase shift rel S0	-->	BNC4	-
B13	-	-	-	-
B14	SF1 256 x LO offset frequency	<--	PC9	B10
B15	-	-	-	-
B16	S0 reference LO offset frequency	-->	BNC2	-
B17	-	-	-	-
B18	-	-	-	-
B19	-	-	-	-
B20	-	-	-	-
B21	-	-	-	-
B22	-	-	-	-
B23	-	-	-	-
B24	-	-	-	-
B25	-	-	-	-
B26	-	-	-	-
B27,B28	+5VDC supply	<--Red	chassis	-

Single Side-band Generator

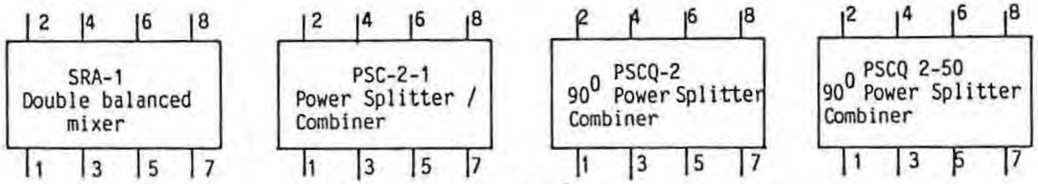


Notes:
SRA-1 gnd. on 2,5,6,7
PSCQ-2 gnd. on 3,4,7,8
PSC-2-1 gnd. on 2,3,4,7,8

Single Side-band Generator



Note:
Pin 1 is marked with a blue bead



Top Views

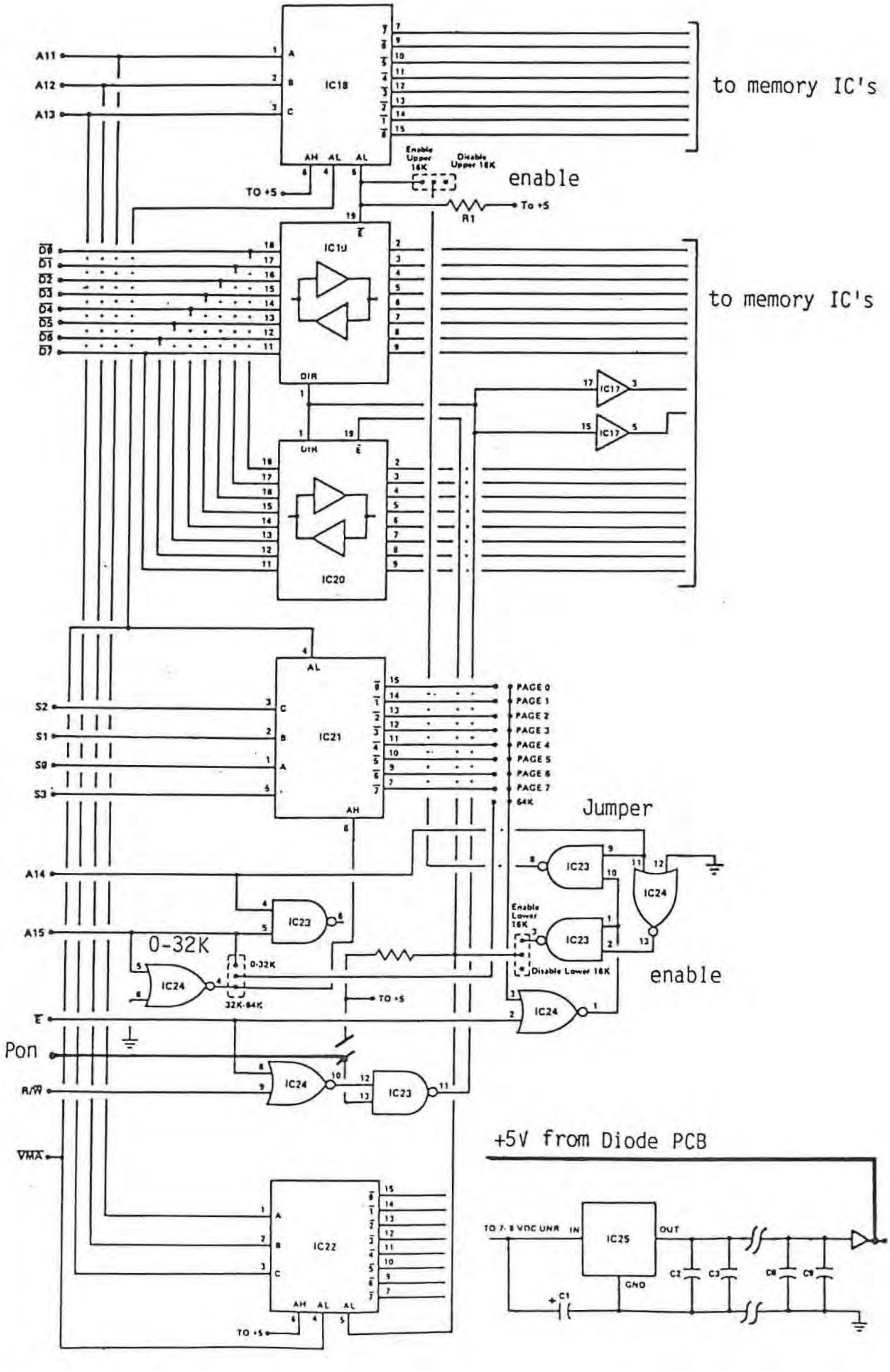
Logchirp board A40 , +24V to -10VDC Converter , PC12 - Component Side (T)

Pin No.	A40 Pin No.	Signal Name	Direction	From/to	Pin No.
T1,T2	-	-	-	-	-
T3	-	-	-	-	-
T4	-	-	-	-	-
T5	-	-	-	-	-
T6	-	-	-	-	-
T7	1	-	-	-	-
T8	2	gnd.	<--	chassis	-
T9	3	-	-	-	-
T10	4	-	-	-	-
T11	5	-	-	-	-
T12	6	-	-	-	-
T13	7	-10VDC	-->	RPA3	6
T14	8	-	-	-	-
T15	9	-	-	-	-
T16	10	-	-	-	-
T17	11	-	-	-	-
T18	12	-	-	-	-
T19	13	-	-	-	-
T20	14	-	-	-	-
T21	15	-	-	-	-
T22	16	-	-	-	-
T23	17	-	-	-	-
T24	18	-	-	-	-
T25	19	-	-	-	-
T26	20	-	-	-	-
T27,T28	21,22	-	-	-	-

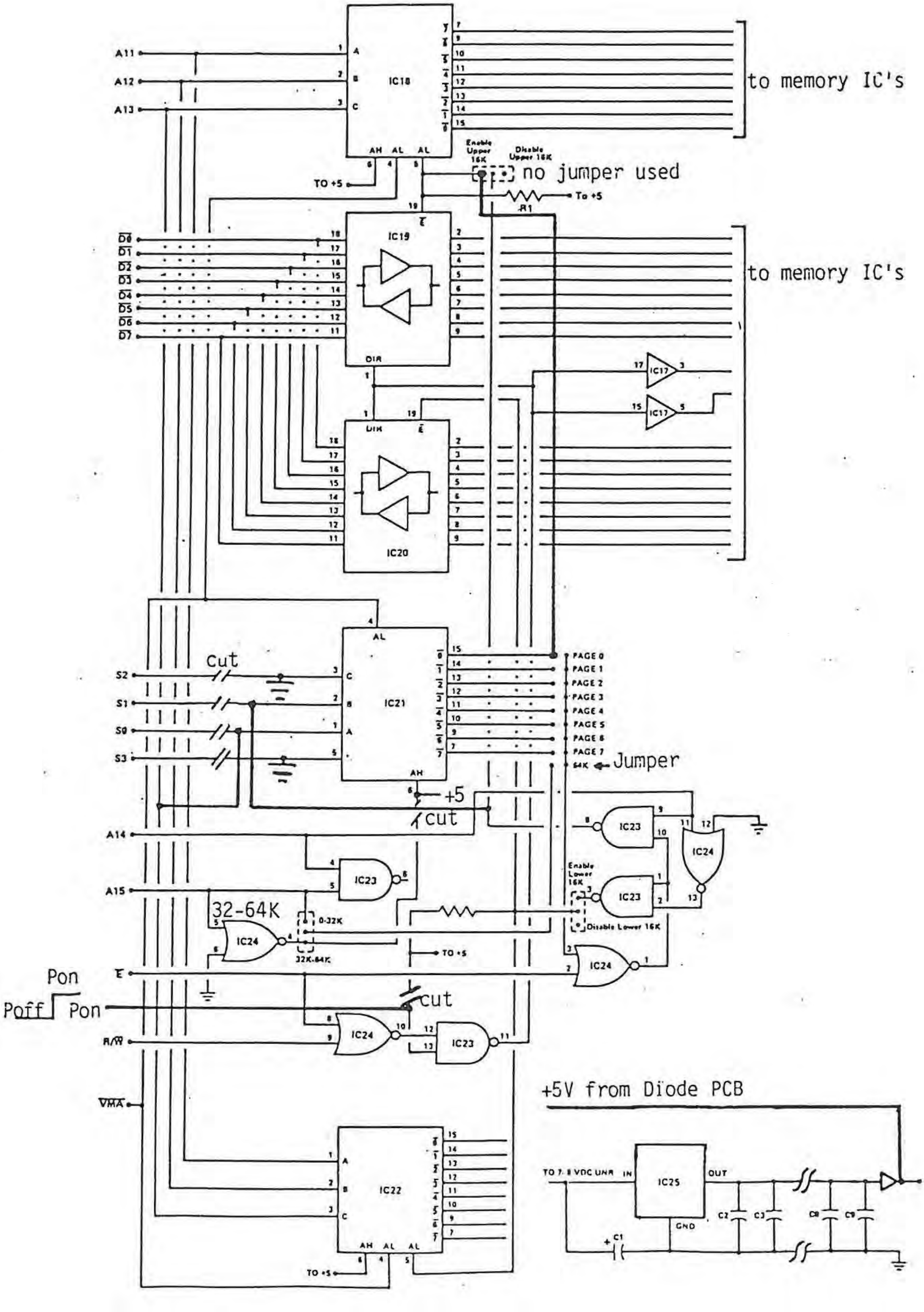
Logchirp board A40 , +24V to -10VDC Converter , PC12 - Solder Side (B)

Pin No.	A40 Pin No.	Signal Name	Direction	From/to	Pin No.
B1,B2	-	-	-	-	-
B3	-	-	-	-	-
B4	-	-	-	-	-
B5	-	-	-	-	-
B6	-	-	-	-	-
B7	A	-	-	-	-
B8	B	gnd	<--	chassis	-
B9	C	-	-	-	-
B10	D	+24VDC	<--	Filter PCB	-
B11	E	-	-	-	-
B12	F	-	-	-	-
B13	H	-	-	-	-
B14	J	-	-	-	-
B15	K	-	-	-	-
B16	L	-	-	-	-
B17	M	-	-	-	-
B18	N	-	-	-	-
B19	P	-	-	-	-
B20	R	-	-	-	-
B21	S	-	-	-	-
B22	T	-	-	-	-
B23	U	-	-	-	-
B24	V	-	-	-	-
B25	W	-	-	-	-
B26	X	-	-	-	-
B27,B28	Y,Z	-	-	-	-

S-32 Memory board — 0 to 32K



S-32 Memory Board — 32K to 64K



Control Signals to 4050 Amplifier T/R , RPA2

RPA2 = Controller Rear Panel Amphenol socket number 2.

Pin No.	Signal Name and Function	Direction	From/to	Pin No.
1	-	-	-	-
2	Tx Blank	<--Wht	PC3	T13
3	-	-	-	-
4	-	-	-	-
5	-	-	-	-
6	-	-	-	-
7	-	-	-	-
8	SC0 Tx , Rx and T/R control	<--Blk	PC3	B18
9	SC1	<--Brn	PC3	B17
10	M-Sequence out	<--Red	PC4	B26
11	-	-	-	-
12	-	-	-	-
13	gnd	<--Blk	chassis	-
14	gnd	<--Blk	chassis	-

Additions to 4050 Amplifier T/R rear panel Amphenol connector ,J7

Pin No.	Signal Name and Function	Direction	From/to	Pin No.
8	SC0 Tx , Rx and T/R control	-->Blk	A6	V
9	SC1	-->Brn	A6	W
10	M-Sequence out	-->Red	A6	E

Control Signals to 6018 Camera Interface/Display , RPA3

RPA3 = Controller Rear Panel Amphenol Socket number 3

Pin No.	Signal Name and Function	Direction	From/to	Pin No.
1	500 Hz fast film advance	<--Co-ax	PC8	B15
2	Variable film speed	<--Co-ax	PC8	B26
3	Sweep	<--Red	PC3	T22
4	Video Blank	<--Blk/Wht	PC3	B7
5	+5V Bat battery supply	<--	Diode PCB	-
6	-10VDC battery supply	<--	PC12(A40)	T13(7)
7	Fast Film advance	<--	PC3	T9
8	100 Hz to 6018 clock	<--	PC3	B8
9	100 Hz to 6018 clock	<--	PC3	B8
10	Bold frequency mark	<--Yel	PC2	B12
11	Fine frequency mark	<--Grn	PC2	B11
12	gnd	<--Blk	-	-
13	gnd	<--Blk	-	-
14	gnd	<--Blk	-	-

Additions to 6018 Camera Interface/Display rear panel Amphenol connector , J2

Pin No.	Signal Name and Function	Direction	From/to	Pin No.
4	Video Blank	-->	A5	F

Signals to and from the Data Capture System , RPA4

RPA4 = Controller Rear Panel Amphenol Connector number 4. Pins 1 to 25.

Pin No.	Signal Name and Function	Direction	From/to	Pin No.
1	B0 Data to DCS	<--Blk	MP-LA-2B	B0
2	B1	<--Brn	MP-LA-2B	B1
3	B2	<--Red	MP-LA-2B	B2
4	B3	<--Orn	MP-LA-2B	B3
5	B4	<--Yel	MP-LA-2B	B4
6	B5	<--Grn	MP-LA-2B	B5
7	B6	<--Blu	MP-LA-2B	B6
8	B7	<--Vio	MP-LA-2B	B7
9	CB1 flag , negative transition	<--Gry	MP-LA-2B	CB1
10	WRE write strobe with E restore	<--Wht	MP-LA-2B	CB2
11	gnd	<--Blk	chassis	-
12	Rx Phase sync	<--Orn	PC6	T14
13	Rx $\overline{\text{Int}}$/ext clock select	<--Wht	PC3	T11
14	Rx AGC ext clock	<--Red	PC6	T13
15	gnd	<--Blk	-	-
16	-	-	-	-
17	-	-	-	-
18	-	-	-	-
19	-	-	-	-
20	-	-	-	-
21	-	-	-	-
22	-	-	-	-
23	-	-	-	-
24	-	-	-	-
25	-	-	-	-

Signals to and from the Data Capture System , RPA4

RPA4 = Controller Rear Panel Amphenol Connector number 4. Pins 26 to 50.

Pin No.	Signal Name and Function	Direction	From/to	Pin No.
26	-	-	-	-
27	-	-	-	-
28	-	-	-	-
29	-	-	-	-
30	-	-	-	-
31	-	-	-	-
32	-	-	-	-
33	-	-	-	-
34	-	-	-	-
35	-	-	-	-
36	gnd	<--Blk	chassis	-
37	Sweep to FFT	<--Blk	PC3	B16
38	FFT Sample Rate	<--Yel	PC7	B24
39	$\overline{\text{Tcell}}$ to FFT	<--Brn	PC4	T12
40	gnd	<--Blk	chassis	-
41	-	-	-	-
42	-	-	-	-
43	-	-	-	-
44	-	-	-	-
45	-	-	-	-
46	-	-	-	-
47	-	-	-	-
48	-	-	-	-
49	-	-	-	-
50	-	-	-	-

APPENDIX C

VERTICHIRP CONTROLLER SOFTWARE

CHANGES AND ADDITIONS TO FORTH

G.P.EVANS

ANTARCTIC RESEARCH - RHODES UNIVERSITY

DECEMBER 1984

APPENDIX C

VERTICHIRP CONTROLLER SOFTWARE - CHANGES AND ADDITIONS TO FORTH

Symbol	Meaning
addr	16 bit memory address
b	8 bit byte , high 8 bits zero
c	7 bit Ascii character , high 9 bits zero
d	32 bit signed double number , MSB on top of stack
f	boolean flag , 0 = false , non-zero = true
ff	boolean false flag = 0
tf	boolean true flag = non-zero
n	16 bit signed integer number
u	16 bit unsigned integer number

The first line of each glossary entry gives the FORTH word being described followed by a symbolic description of the action of the word on the parameter stack. The symbols indicate the order in which the input parameters have been placed on the stack. Three dashes "---" indicate the execution point and any parameters left on the stack are then listed. The top of the stack (most accessible parameter) is to the right in this notation.

* Original FORTH Memory Map *

$0000 --> $1BD0 = FORTH kernel

$1BD1 --> $1BEF = empty

$1BF0 --> $1FFF = disk buffer

$2000 --> $204F = FORTH registers and pointers
and user variables.

$2050 = "FORTH" (a word)

$207E = "TASK" (a word)

$2089 --> = beginning of dictionary (grows up)

$2F30 = data stack (grows down)

$2F30 --> $2FB4 = TIB (terminal input buffer)

$3000 = RETURN stack base (grows down)

+++

```
+++LIST FORTHAD
       TTL   ANTARCTIC RESEARCH (RHODES)
       STTL  ADDITIONS TO FORTH  KERNEL
       OPT   PAG,NOC,MAC,EXP
*
* This program allows Primitives or FDB coded words
* to be added to the FORTH Kernel.
* PROCEDURE
* Create files of primitives to be assembled (FLEX EDITor)
* These files have the form :-
*
* Function of Primitive as a Comment
* Any other Comments
*
* The MACRO for creating dictionary headers has the form :-
* WORDM  word char count,word except last char,last char,IMMEDIATE
* For primitives
*       WORDM  7,EXAMPL,E
*EXAM   FDB    *+2
*       Assembler Mneumonics
*       JMP    NEXT        MUST end with a jump to NEXT
*
* For DOCOL Words :-
*      WORDM  2,E,G
*EG    FDB    DOCOL
*      FDB's defining the word EG
*      FDB    SEMIS              MUST end with an FDB SEMIS
*
* Assemble FORTHAD into named file (ASMB,FORTHAD,NAME)
* The command  is ASMB,FORTHAD,NAME,+P2,+PRMA,PRMB,PRMC
* Where
* NAME - name of binary file
* P2   - Assembled listing to start from page 2
* PRMA - HEX equal to last Kernal name field addr.(NFA)
* PRMB - HEX equal to next free memory location in Kernel
* PRMC - Name of text file containing Primitives
*
* These parameters can be obtained as follows from FORTH
* '(Word before 'FORTH' in dict) NFA . = PRMA (HEX)
*  PRMA 30 DUMP
*  The first free memory location in the Kernel
*  is the first $20 in the string of $20's
*  This address = PRMB
*
* Load FORTH
* GET NAME
*
DOCOL  EQU  $0073
NEXT   EQU  $0077
NEXT3  EQU  $0079
SEMIS  EQU  $0080
*
* Set LASTNM equal to last Kernel name field address (NFA)
*
LASTNM SET  $&A
*
       PAG
```

```
*
*** MACRO for creating dictionary headers ***
*
WORDM  MACRO
NEXTNM SET    *
       IFC    &4,IMMEDIATE
       FCB    &1+$C0
* 1st byte is no of char with sign and immed bit on if IMMEDIATE
       ELSE
       FCB    &1+$80
       ENDIF
       IFNC   &1,1
       FCC    '&2'
       ENDIF
* if more than one char,then all but last in here
* then last has sign bit set
       FCB    $80+'&3
       FDB    LASTNM
LASTNM SET    NEXTNM
       ENDM
*
* Set ORG equal to next free memory location in Kernel
* and declare the file containing FORTH Primitives
* for assembly using a LIB directive.
*
       PAG
*
       ORG  $&B
*
       LIB  &C
*
* The next code sets the link-back for this version
*
       ORG  $1196
       FDB  LASTNM
       ORG  $2806
       FDB  LASTNM
       END
*

+++
```

* Primitive and colon definition additions to FORTH Kernel *

FLEX Text files assembled via the programme FORTHAD.

1. Double Precision Operators

```
FLEX Filename = DPRIMS.TXT
Command line  = ASMB,FORTHAD,DPRIMS,+P2,+1A34,1BE0,DPRIMS
Binary File   = DPRIMS.BIN
Begin address = $1BE0
End   address = $1CBA
```

2. Interrupt Handling in FORTH

```
FLEX Filename = INTPRIMS.TXT
Command line  = ASMB,FORTHAD,INTPRIMS,+P2,+1CAC,1CBB,INTPRIMS
Binary File   = INTPRIMS.BIN
Begin address = $1CBB
End   address = $1CED
```

3. MPN Calculator Drivers

```
FLEX Filename = CALPRIMS.TXT
Command line  = ASMB,FORTHAD,CALPRIMS,+P2,+1CDF,1CEE,CALPRIMS
Binary File   = CALPRIMS.BIN
Begin address = $1CEE
End   address = $1D6C
```

4. Primitive FORTH Word VECTOR

```
FLEX Filename = VECPRIM.TXT
Command line  = ASMB,FORTHAD,VECPRIM,+P2,+1D17,1D6D,VECPRIM
Binary File   = VECPRIM.BIN
Begin address = $1D6D
End   address = $1D92
```

5. Primitive FORTH Word RESTORE1

```
FLEX Filename = RESTORE1.TXT
Command line  = ASMB,FORTHAD,RESTORE,+P2,+1D6D,1D93,RESTORE
Binary File   = RESTORE.BIN
Begin address = $1D93
End   address = $1DAE
```

6. CASE Statement

```
FLEX Filename = CASE.TXT
Command line  = ASMB,FORTHAD,CASE,+P2,+1D93,1DAF,CASE
Binary File   = CASE.BIN
Begin address = $1DAF
End   address = $1E52
```

7. Primitives U< and U>

```
FLEX Filename = UNSIGNED.TXT
Command line  = ASMB,FORTHAD,UNSIGNED,+P2,+1E21,1E53,UNSIGNED
Binary File   = UNSIGNED.BIN
Begin address = $1E53
End   address = $1E7C
```

```
+++ASMB,FORTHAD,DPRIMS,+P2,+1A34,1BE0,DPRIMS
DELETE OLD BINARY (Y-N)? Y


ANTARCTIC RESEARCH (RHODES)          9-30-81 TSC ASSEMBLER   PAGE    2
ADDITIONS TO FORTH  KERNEL


                         *
 1BE0                           ORG    $1BE0
                         *
                         *
                         *              Double Precision FORTH Operatives
                         *
                         * Single Precision No:- 2 Bytes
                         *
                         *          Stacks    Memory
                         * addr+1   LSB       LSB
                         * addr     MSB       MSB
                         *       Top of Stack
                         *
                         * Double Precision No:- 4 Bytes
                         *
                         *          Stacks    Memory
                         * addr+3   LSB       LSB
                         * addr+2   LSB .     LSB
                         * addr+1   MSB       MSB
                         * addr     MSB       MSB
                         *       Top of Stack
                         *
                         ***Double Precision Memory and Stack Operators***
                         *
                         * Primitives
                         *
                         * Primitive D@
                         *
 1BE0                           WORDM  2,D,@
               1BE0 NEXTNM      SET    *
 1BE0 82                        FCB    2+$80
 1BE1 44                        FCC    'D'
 1BE2 C0                        FCB    $80+'@
 1BE3 1A34                      FDB    LASTNM
               1BE0 LASTNM      SET    NEXTNM
                                ENDM
 1BE5 1BE7               DAT    FDB    *+2
 1BE7 37   10                   PULU   X          addr to X
 1BE9 EC   84                   LDD    ,X         MSBs from mem to D
 1BEB AE   02                   LDX    2,X        LSBs from mem to X
 1BED 20   57                   BRA    PSHXD      X then D to stack,NEXT
                         *
                         * Primitive D!
                         *
 1BEF                           WORDM  2,D,!
               1BEF NEXTNM      SET    *
 1BEF 82                        FCB    2+$80
 1BF0 44                        FCC    'D'
 1BF1 A1                        FCB    $80+'!
 1BF2 1BE0                      FDB    LASTNM
               1BEF LASTNM      SET    NEXTNM
                                ENDM
```

```
 1BF4 1BF6          DSTORE  FDB    *+2
 1BF6 37    10              PULU   X           addr to X
 1BF8 37    06              PULU   D           MSBs to D
 1BFA ED    84              STD    ,X          MSBs to addr
 1BFC 37    06              PULU   D           LSBs to D
 1BFE ED    02              STD    2,X         LSBs to addr+2
 1C00 0E    77              JMP    NEXT
                    *
                    * Primitive DDUP
                    *
 1C02                       WORDM  4,DDU,P
               1C02 NEXTNM  SET    *
 1C02 84                    FCB    4+$80
 1C03 44 44 55              FCC    'DDU'
 1C06 D0                    FCB    $80+'P
 1C07 1BEF                  FDB    LASTNM
               1C02 LASTNM  SET    NEXTNM
                            ENDM
 1C09 1C0B          DDUP    FDB    *+2
 1C0B EC    C4              LDD    ,U          MSBs to D
 1C0D AE    42              LDX    2,U         LSBs to X
 1C0F 20    35              BRA    PSHXD
                    *
                    * Primitive DOVER
                    *
 1C11                       WORDM  5,DOVE,R
               1C11 NEXTNM  SET    *
 1C11 85                    FCB    5+$80
 1C12 44 4F 56 45           FCC    'DOVE
 1C16 D2                    FCB    $80+'R
 1C17 1C02                  FDB    LASTNM
               1C11 LASTNM  SET    NEXTNM
                            ENDM
 1C19 1C1B          DOVER   FDB    *+2
 1C1B EC    44              LDD    4,U         MSBs to D
 1C1D AE    46              LDX    6,U         LSBs to X
 1C1F 20    25              BRA    PSHXD
                    *
                    * Primitive D>R
                    *
 1C21                       WORDM  3,D>,R
               1C21 NEXTNM  SET    *
 1C21 83                    FCB    3+$80
 1C22 44 3E                 FCC    'D>'
 1C24 D2                    FCB    $80+'R
 1C25 1C11                  FDB    LASTNM
               1C21 LASTNM  SET    NEXTNM
                            ENDM
 1C27 1C29          DTOR    FDB    *+2
 1C29 37    16              PULU   D,X         MSBs to D, LSBs to X
 1C2B 34    16              PSHS   X,D         LSBs, MSBs to S
 1C2D 0E    77              JMP    NEXT
                    *
```

```
ANTARCTIC RESEARCH (RHODES)          9-30-81 TSC ASSEMBLER   PAGE    4
ADDITIONS TO FORTH  KERNEL

                        * Primitive DR>
                        *
 1C2F                           WORDM   3,DR,>
                 1C2F   NEXTNM  SET     *
 1C2F 83                        FCB     3+$80
 1C30 44 52                     FCC     'DR'
 1C32 BE                        FCB     $80+'>
 1C33 1C21                      FDB     LASTNM
                 1C2F   LASTNM  SET     NEXTNM
                                ENDM
 1C35 1C37              DFROMR  FDB     *+2
 1C37 35    16                  PULS    D,X         MSBs to D, LSBs to X
 1C39 20    0B                  BRA     PSHXD
                        *
                        * Primitive DR
                        *
 1C3B                           WORDM   2,D,R
                 1C3B   NEXTNM  SET     *
 1C3B 82                        FCB     2+$80
 1C3C 44                        FCC     'D'
 1C3D D2                        FCB     $80+'R
 1C3E 1C2F                      FDB     LASTNM
                 1C3B   LASTNM  SET     NEXTNM
                                ENDM
 1C40 1C42              DR      FDB     *+2
 1C42 EC    E4                  LDD     ,S          Copy MSBs to D
 1C44 AE    62                  LDX     2,S         Copy LSBs to X
 1C46 36    16          PSHXD   PSHU    X,D
 1C48 0E    77                  JMP     NEXT
                        *
                        * Colon Definitions
                        *
                        * : DDROP DROP DROP ;
                        *
                 066B   DROP    EQU     $066B
 1C4A                           WORDM   5,DDRO,P
                 1C4A   NEXTNM  SET     *
 1C4A 85                        FCB     5+$80
 1C4B 44 44 52 4F               FCC     'DDRO'
 1C4F D0                        FCB     $80+'P
 1C50 1C3B                      FDB     LASTNM
                 1C4A   LASTNM  SET     NEXTNM
                                ENDM
 1C52 0073 066B         DDROP   FDB     DOCOL,DROP,DROP,SEMIS
 1C56 066B 0080
                        *
                        * : DSWAP ROT >R ROT R> ;
                        *
                 0C86   ROT     EQU     $0C86
                 0639   TOR     EQU     $0639
                 0647   FROMR   EQU     $0647
 1C5A                           WORDM   5,DSWA,P
                 1C5A   NEXTNM  SET     *
```

```
1C5A 85                     FCB    5+$80
1C5B 44 53 57 41            FCC    'DSWA'
1C5F D0                     FCB    $80+'P
1C60 1C4A                   FDB    LASTNM
              1C5A  LASTNM  SET    NEXTNM
                            ENDM
1C62 0073 0C86      DSWAP   FDB    DOCOL,ROT,TOR,ROT,FROMR,SEMIS
1C66 0639 0C86
1C6A 0647 0080
                    *
                    * : DROT D>R DSWAP DR> DSWAP ;
                    *
1C6E                        WORDM  4,DRO,T
              1C6E  NEXTNM  SET    *
1C6E 84                     FCB    4+$80
1C6F 44 52 4F               FCC    'DRO'
1C72 D4                     FCB    $80+'T
1C73 1C5A                   FDB    LASTNM
              1C6E  LASTNM  SET    NEXTNM
                            ENDM
1C75 0073 1C27      DROT    FDB    DOCOL,DTOR,DSWAP,DFROMR,DSWAP,SEMIS
1C79 1C62 1C35
1C7D 1C62 0080
                    *
                    * Double Precision Arithmetic and Logical
                    *
                    * Colon Definitions
                    *
                    * : D- DMINUS D+ ;
                    *
              0461  DMINUS  EQU    $0461
              042B  DPLUS   EQU    $042B
1C81                        WORDM  2,D,-
              1C81  NEXTNM  SET    *
1C81 82                     FCB    2+$80
1C82 44                     FCC    'D'
1C83 AD                     FCB    $80+'-
1C84 1C6E                   FDB    LASTNM
              1C81  LASTNM  SET    NEXTNM
                            ENDM
1C86 0073 0461      DSUB    FDB    DOCOL,DMINUS,DPLUS,SEMIS
1C8A 042B 0080
                    *
                    * Double Precision Comparison
                    *
                    * Colon definitions
                    *
                    * : D< D- SWAP DROP 0< ;
                    *
              0679  SWAP    EQU    $0679
              0611  ZLESS   EQU    $0611
1C8E                        WORDM  2,D,<
              1C8E  NEXTNM  SET    *
```

```
 1C8E 82                     FCB    2+$80
 1C8F 44                     FCC    'D'
 1C90 BC                     FCB    $80+'<
 1C91 1C81                   FDB    LASTNM
               1C8E  LASTNM  SET    NEXTNM
                             ENDM
 1C93 0073 1C86      DLESS   FDB    DOCOL,DSUB,SWAP,DROP,ZLESS,SEMIS
 1C97 0679 066B
 1C9B 0611 0080
                     *
                     * : D> DSWAP D< ;
                     *
 1C9F                        WORDM  2,D,>
               1C9F  NEXTNM  SET    *
 1C9F 82                     FCB    2+$80
 1CA0 44                     FCC    'D'
 1CA1 BE                     FCB    $80+'>
 1CA2 1C8E                   FDB    LASTNM
               1C9F  LASTNM  SET    NEXTNM
                             ENDM
 1CA4 0073 1C62      DGREAT  FDB    DOCOL,DSWAP,DLESS,SEMIS
 1CA8 1C93 0080
                     *
                     * : D= D- OR 0= ;
                     *
               03FF  OR      EQU    $03FF
               05FE  ZEQU    EQU    $05FE
 1CAC                        WORDM  2,D,=
               1CAC  NEXTNM  SET    *
 1CAC 82                     FCB    2+$80
 1CAD 44                     FCC    'D'
 1CAE BD                     FCB    $80+'=
 1CAF 1C9F                   FDB    LASTNM
               1CAC  LASTNM  SET    NEXTNM
                             ENDM
 1CB1 0073 1C86      DEQUAL  FDB    DOCOL,DSUB,OR,ZEQU,SEMIS
 1CB5 03FF 05FE
 1CB9 0080
                     *
                     *
                     * The next code sets the link-back for this version
                     *
 1196                        ORG    $1196
 1196 1CAC                   FDB    LASTNM
 2806                        ORG    $2806
 2806 1CAC                   FDB    LASTNM
                             END

0 ERROR(S) DETECTED
```

Double Precision - Memory and Stack

D@	addr --- d Leave the 32 bit contents of the address
D!	d addr --- Store 32 bit d at the address
DDUP	d --- d d Duplicate the double number on the stack
DOVER	d1 d2 --- d1 d2 d1 Copy the second stack double value , placing it as the new top value
D>R	d --- Remove d from the data stack and place as most accessible on the return stack. Use should be balanced with DR> in the same definition
DR>	--- d Remove the top value d from the return stack and leave it on the data stack. See D>R and DR
DR	--- d Copy the top of the return stack to the data stack
DDROP	d --- Drop the double number from the stack.
DSWAP	d1 d2 --- d2 d1 Exchange the top two double numbers on the stack.
DROT	d1 d2 d3 --- d2 d3 d1 Rotate the top three double numbers on the stack bringing the third to the top

Double Precision - Arithmetic and Logical.

D-	d1 d2 --- diff Leave the difference of d1-d2

Double Precision - Comparison.

D<	d1 d2 --- f Leave a true flag if d1 is less than d2 ; otherwise leave a false flag
D>	d1 d2 --- f Leave a true flag if d1 is greater than d2 ; otherwise leave a false flag.
D=	d1 d2 --- f Leave a true flag if d1 = d2 ; otherwise leave a false flag.

```
+++ASMB,FORTHAD,INTPRIMS,+P2,+1CAC,1CBB.INTPRIMS
DELETE OLD BINARY (Y-N)? Y


ANTARCTIC RESEARCH (RHODES)          5-17-82 TSC ASSEMBLER    PAGE    2
ADDITIONS TO FORTH  KERNEL


                     *
  1CBB                         ORG    $1CBB
                     *
                     *
                     ***   Interrupt handling in FORTH    ***
                     *
                     * Primitive RTI - Return from Interrupt
                     *
  1CBB                         WORDM  3,RT,I
               1CBB  NEXTNM    SET    *
  1CBB 83                      FCB    3+$80
  1CBC 52 54                   FCC    'RT'
  1CBE C9                      FCB    $80+'I
  1CBF 1CAC                    FDB    LASTNM
               1CBB  LASTNM    SET    NEXTNM
                               ENDM
  1CC1 1CC3          RTI       FDB    *+2
  1CC3 35    40                PULS   U         Remove value DOCOL pushed to S
  1CC5 3B                      RTI              Return from Interrupt
                     *        JMP    NEXT       Never executed
                     *
                     * Primitive CLI - Clear Interrupt Mask
                     *
  1CC6                         WORDM  3,CL,I
               1CC6  NEXTNM    SET    *
  1CC6 83                      FCB    3+$80
  1CC7 43 4C                   FCC    'CL'
  1CC9 C9                      FCB    $80+'I
  1CCA 1CBB                    FDB    LASTNM
               1CC6  LASTNM    SET    NEXTNM
                               ENDM
  1CCC 1CCE          CLI       FDB    *+2
  1CCE 1C    EF                ANDCC  #$EF      11101111
  1CD0 0E    77                JMP    NEXT
                     *
                     * Primitive SEIF - Set I and F Masks
                     *
  1CD2                         WORDM  4,SEI,F
               1CD2  NEXTNM    SET    *
  1CD2 84                      FCB    4+$80
  1CD3 53 45 49                FCC    'SEI'
  1CD6 C6                      FCB    $80+'F
  1CD7 1CC6                    FDB    LASTNM
               1CD2  LASTNM    SET    NEXTNM
                               ENDM
  1CD9 1CDB          SEIF      FDB    *+2
  1CDB 1A    50                ORCC   #$50      01010000
  1CDD 0E    77                JMP    NEXT
                     *
                     * Primitive IRQ - Vectors FORTH to an Interrupt Word
                     *
  1CDF                         WORDM  3,IR,Q
```

```
ANTARCTIC RESEARCH (RHODES)          5-17-82 TSC ASSEMBLER   PAGE    3
ADDITIONS TO FORTH  KERNEL

               1CDF  NEXTNM  SET    *
 1CDF 83                     FCB    3+$80
 1CE0 49 52                  FCC    'IR'
 1CE2 D1                     FCB    $80+'Q
 1CE3 1CD2                   FDB    LASTNM
               1CDF  LASTNM  SET    NEXTNM
                             ENDM
 1CE5 1CE7                   FDB    *+2        Header shows where
 1CE7 0E   77                JMP    NEXT       IRQ code is stored
 1CE9 BE   23E6      IRQ     LDX    $23E6      $23E6 contains ADDR of CFA of Interrupt Word
 1CEC 0E   79                JMP    NEXT3      JMP indirect to code pointed to by X
                     *
                     *
                     * The next code sets the link-back for this version
                     *
 1196                        ORG    $1196
 1196 1CDF                   FDB    LASTNM
 2806                        ORG    $2806
 2806 1CDF                   FDB    LASTNM
                             END

0 ERROR(S) DETECTED
```

* Interrupts in FORTH *

RTI ---

Return from interrupt. The interrupt service FORTH word must end with this word.

CLI ---

Clear interrupt mask (I)

SEIF ---

Set I and F interrupt masks

IRQ ---

Vectors FORTH execution to an interrupt service FORTH word. Vectoring is via SBUG IRQ vector at $DFC8 which is loaded with the address of the machine code of the FORTH IRQ word. ($1CE9)
This vectors FORTH execution to the FORTH word whose CFA is stored at $23E6

Using Interrupts

1. The interrupt service FORTH word must be defined and must end with RTI.

   Eg.
```
    : POLL ---forth words--- RTI ;
```

2. Set the vector at $23E6 to point to the interrupt service FORTH word CFA.

   Eg.
```
HEX
    : STORECFA LIT POLL 23E6 ! ;
```

3. Set SBUG IRQ vector at $DFC8 to point to machine code in IRQ word

   Eg.
```
HEX
     : SETVEC 1CE9 DFC8 ! ;
```

4. Combining these
```
     : SETIRQ STORECFA SETVEC ;
                   or
     : SETIRQ LIT POLL 23E6 ! 1CE9 DFC8 ! ;
```

```
+++ASMB,FORTHAD,CALPRIMS,+P2,+1CDF,1CEE,CALPRIMS
DELETE OLD BINARY (Y-N)? Y


ANTARCTIC RESEARCH (RHODES)          9-13-82 TSC ASSEMBLER   PAGE    2
ADDITIONS TO FORTH  KERNEL


                          *
 1CEE                           ORG    $1CEE
                          *
                          *        MPN Calculator Drivers
                          *
                          * TOMPN Sends data or instructions to the calculator
                          * ANS   Terminates number entry and outputs result
                          *
                    E000  PORT   EQU    $E000      CALCULATOR PORT ADDRESS
 1CEE                            WORDM  5,TOMP,N
                    1CEE  NEXTNM SET    *
 1CEE 85                         FCB    5+$80
 1CEF 54 4F 4D 50                FCC    'TOMP'
 1CF3 CE                         FCB    $80+'N
 1CF4 1CDF                       FDB    LASTNM
                    1CEE  LASTNM SET    NEXTNM
                                 ENDM
 1CF6 1CF8                TOMPN  FDB    *+2
 1CF8 8E   E000                  LDX    #PORT
 1CFB 37   06                    PULU   D
 1CFD 1E   89                    EXG    A,B
 1CFF E6   01             OUTINS LDAB   1,X        WAIT FOR READY
 1D01 2A   FC                    BPL    OUTINS
 1D03 A7   84                    STAA   0,X        FORWARD INSTRUCTION TO CALC.
 1D05 E6   84                    LDAB   0,X        CLEAR FLAG BIT
 1D07 C6   3C                    LDAB   #$3C       LOW HOLD-NEG READY
 1D09 E7   01                    STAB   1,X        BRING HOLD LINE LOW
 1D0B E6   01             WAIT1  LDAB   1,X
 1D0D 2A   FC                    BPL    WAIT1      LOOK FOR READY LOW
 1D0F E6   84                    LDAB   0,X        CLEAR FLAG BIT
 1D11 C6   36                    LDAB   #$36       HIGH HOLD-POS READY
 1D13 E7   01                    STAB   1,X        RETURN HOLD LINE HIGH
 1D15 0E   77                    JMP    NEXT
                          *
                    2060  ANSEXP EQU    $2060      ANSWER BUFFER
 1D17                            WORDM  3,AN,S
                    1D17  NEXTNM SET    *
 1D17 83                         FCB    3+$80
 1D18 41 4E                      FCC    'AN'
 1D1A D3                         FCB    $80+'S
 1D1B 1CEE                       FDB    LASTNM
                    1D17  LASTNM SET    NEXTNM
                                 ENDM
 1D1D 1D1F                ANS    FDB    *+2        POINTER TO CODE
 1D1F 34   20                    PSHS   Y          SAVE Y
 1D21 108E 2060                  LDY    #ANSEXP    POINT Y TO BUFFER
 1D25 8E   E000                  LDX    #PORT      SET X = CALC PORT ADDRESS
 1D28 E6   01             OUTANS LDAB   1,X
 1D2A 2A   FC                    BPL    OUTANS
 1D2C A6   84                    LDAA   0,X        CLEAR FLAG BIT
 1D2E 86   16                    LDAA   #$16       SEND AN OUT
 1D30 A7   84                    STAA   0,X
```

```
ANTARCTIC RESEARCH (RHODES)           9-13-82 TSC ASSEMBLER   PAGE    3
ADDITIONS TO FORTH  KERNEL

 1D32 C6    3E               LDAB   #$3E      LOW HOLD-POS READY
 1D34 E7    01               STAB   1,X       BRING HOLD LINE LOW
 1D36 E6    01        WAIT2  LDAB   1,X       WAIT FOR SECOND READY
 1D38 2A    FC               BPL    WAIT2
 1D3A E6    84               LDAB   0,X       CLEAR FLAG BIT
 1D3C 86    0F               LDAA   #$0F
 1D3E A7    84               STAA   0,X       SEND A NOP
 1D40 E6    03        WAIT3  LDAB   3,X       LOOK FOR R/W STROBE
 1D42 2B    06               BMI    OUTDIG    TRANSFER CALC DATA TO MEMORY
 1D44 E6    01               LDAB   1,X       LOOK FOR READY STROBE
 1D46 2B    13               BMI    CONFLG    PRINT MEMORY CONTENTS
 1D48 20    F6               BRA    WAIT3
 1D4A A6    02        OUTDIG LDAA   2,X       LOAD OUT DATA INTO A
 1D4C 1F    894D             TAB
 1D4F 84    0F               ANDA   #$0F      ELIMINATE UPPER 4 BITS
 1D51 8A    30               ORAA   #$30      CONVERT TO ASCII DATA
 1D53 54                     LSRB
 1D54 54                     LSRB
 1D55 54                     LSRB
 1D56 54                     LSRB
 1D57 A7    A5               STAA   B,Y
 1D59 20    E5               BRA    WAIT3
 1D5B 86    36        CONFLG LDAA   #$36      HIGH HOLD-POS READY
 1D5D A7    01               STAA   1,X       BRING HOLD LINE HIGH
 1D5F A6    84               LDAA   0,X       CLEAR FLAG BIT
 1D61 84    80               ANDA   #$80      GET ERROR FLAG
 1D63 C6    00               LDAB   #0
 1D65 1E    89               EXG    A,B
 1D67 36    06               PSHU   D
 1D69 35    20               PULS   Y         RESTORE Y
 1D6B 0E    77               JMP    NEXT
                      *
                      * The next code sets the link-back for this version
                      *
 1196                        ORG    $1196
 1196 1D17                   FDB    LASTNM
 2806                        ORG    $2806
 2806 1D17                   FDB    LASTNM
                             END

0 ERROR(S) DETECTED
```

```
LIST,MPN
        * MPN Calculator Driver Words *

   The calculator is used in SCIENTIFIC MODE only

   FORTH word "MPN" initialises calculator and
   sets scientific mode

   Primitive Definitions

   TOMPN    n ---
            Sends data or instructions to the calculator

   ANS        --- f
            Makes use of the calculator 'OUT' instruction
            which 'ENTERS' and 'OUTPUTS' the number sent
            using "TOMPN".
            "ANS" can therefore be used for termination of
            number entry as well as for output of an
            answer after calculation.
            "ANS" leaves a flag on the stack for error checking
             false flag ( 0 ) = no error
             true  flag ( 1 ) = error
            "ANS" stores the calculator X register in memory
            in BCD as follows

ANSEXP      0  0  1  1  MS digit of exponent
            0  0  1  1  LS digit of exponent
            0  0  1  1  Sm  0  0  Se
            not used
ANS#        0  0  1  1  MS digit of mantissa
            0  0  1  1  BCD digit
            0  0  1  1  BCD digit
            0  0  1  1  BCD digit
            0  0  1  1  BCD digit
            0  0  1  1  BCD digit
            0  0  1  1  BCD digit
            0  0  1  1  LS digit of mantissa

            Sm is the sign of the mantissa 0 = positive
            Se is the sign of the exponent 0 = positive


+++
```

```
+++ASMB,FORTHAD,VECPRIM.+P2.+1D17,1D6D,VECPRIM
DELETE OLD BINARY (Y-N)? Y


ANTARCTIC RESEARCH (RHODES)          5-17-82 TSC ASSEMBLER   PAGE    2
ADDITIONS TO FORTH  KERNEL


                    *
 1D6D                       ORG     $1D6D
                    *
                    *
                    ***   Vectored FORTH Word Execution   ***
                    *
                    * Primitive VECTOR
                    *
                    * Used in a colon definition only
                    * Expects a Number n on the Stack
                    * Loads X from Y ( IP ) advancing Y by 2
                    * It will perform a JMP [0,X]
                    * IF X = addr of QSTACK
                    *  Occurs if VECTOR typed outside a colon definition
                    *  Then QSTACK only is executed
                    * IF X = SEMIS
                    *  Occurs if number n > number of words in vector list
                    *  Then SEMIS is executed
                    * IF X = NNNN
                    *  Where NNNN is the nth address in the vector list
                    *  Then the nth FORTH word is executed
                    *
                    *SEMIS  EQU     $0080    Supplied by FORTHAD
              0C5D  QSTACK  EQU     $0C5D
              0136  CRE     EQU     $0136    Address containing $0080
                    *
 1D6D                       WORDM   6,VECTO,R
              1D6D  NEXTNM  SET     *
 1D6D 86                    FCB     6+$80
 1D6E 56 45 43 54           FCC     'VECTO'
 1D72 4F
 1D73 D2                    FCB     $80+'R
 1D74 1D17                  FDB     LASTNM
              1D6D  LASTNM  SET     NEXTNM
                            ENDM
 1D76 1D78          VECTOR  FDB     *+2      Pointer to Code
 1D78 EC   C1               LDD     0,U++    Get offset from Stack,drop Stack
 1D7A AE   A1       LOOP    LDX     0,Y++    'NEXT' first instruction
 1D7C 8C   0C5D             CMPX    #QSTACK  First check
 1D7F 27   10               BEQ     OUT2
 1D81 8C   0080             CMPX    #SEMIS   Second check
 1D84 27   0B               BEQ     OUT2
 1D86 83   0001             SUBD    #1       Decrement offset counter
 1D89 27   02               BEQ     OUT1     If = 0, X points at required word
 1D8B 20   ED               BRA     LOOP     If not = 0, loop
                    *
 1D8D 108E 0136     OUT1    LDY     #CRE     Adjust return address
 1D91 6E   94       OUT2    JMP     [0,X]    'NEXT' second instruction
                    *
                    *
                    * The next code sets the link-back for this version
                    *
```

```
ANTARCTIC RESEARCH (RHODES)          5-17-82 TSC ASSEMBLER   PAGE    3
ADDITIONS TO FORTH  KERNEL

 1196                   ORG    $1196
 1196 1D6D              FDB    LASTNM
 2806                   ORG    $2806
 2806 1D6D              FDB    LASTNM
                        END

0 ERROR(S) DETECTED
```

* Vectored execution in FORTH *

VECTOR   n ---

Used in a colon definition in the form

```
: EXECNTH VECTOR ( word 1 ) --- ( word n ) --- ( last word) ;
```

If EXECNTH is executed with a number n on the stack only the nth FORTH word following VECTOR will execute.

Example

Definitions

```
: MOND ." Monday" ;  : TUES ." Tuesday" ;
: PRINTDAY VECTOR MOND TUES ;
```

Execution

1 PRINTDAY will print Monday
2 PRINTDAY will print Tuesday
3 PRINTDAY will not do anything !

```
+++ASMB,FORTHAD,RESTORE,+P2,+1D6D,1D93,RESTORE
DELETE OLD BINARY (Y-N)? Y


ANTARCTIC RESEARCH (RHODES)          5-17-82 TSC ASSEMBLER   PAGE    2
ADDITIONS TO FORTH  KERNEL


                           *
 1D93                              ORG     $1D93
                           *
                           *
                           * Drive 1 Head Restore ( Seek Track 00)
                           *
                  1A7B     DOSFCB  EQU     $1A7B      Contains Address of Flex FCB ( $C840 )
                  DE0C     DRSEL   EQU     $DE0C      Drive Select
                  DE09     DRREST  EQU     $DE09      Restore
                           *
 1D93                              WORDM   8,RESTORE,1
                  1D93     NEXTNM  SET     *
 1D93 88                           FCB     8+$80
 1D94 52 45 53 54                  FCC     'RESTORE'
 1D98 4F 52 45
 1D9B B1                           FCB     $80+'1
 1D9C 1D6D                         FDB     LASTNM
                  1D93     LASTNM  SET     NEXTNM
                                   ENDM
 1D9E 1DA0                 REST1   FDB     *+2
 1DA0 BE   1A7B                    LDX     DOSFCB     Set X = Flex FCB
 1DA3 86   01                      LDA     #1         Drive 1
 1DA5 A7   03                      STA     3,X        Drive Byte of FCB
 1DA7 BD   DE0C                    JSR     DRSEL      Select Drive 1
 1DAA BD   DE09                    JSR     DRREST     Restore Drive 1
 1DAD 0E   77                      JMP     NEXT
                           *
                           *
                           * The next code sets the link-back for this version
                           *
 1196                              ORG     $1196
 1196 1D93                         FDB     LASTNM
 2806                              ORG     $2806
 2806 1D93                         FDB     LASTNM
                                   END

0 ERROR(S) DETECTED
```

```
+++ASMB,FORTHAD.CASE,+P2,+1D93,1DAF,CASE
DELETE OLD BINARY (Y-N)? Y


ANTARCTIC RESEARCH (RHODES)          8-13-82 TSC ASSEMBLER   PAGE    2
ADDITIONS TO FORTH  KERNEL


                     *
 1DAF                        ORG    $1DAF
                     *
                     *     CASE Statement for FORTH
                     *
                     *   Equates used in definitions of CASE , (OF) , OF , ENDOF , ENDCASE
                     *
               0A3A  QCOMP    EQU    $0A3A
               08AC  CSP      EQU    $08AC
               06A7  AT       EQU    $06A7
               0A0D  SCSP     EQU    $0A0D
               01E7  LIT      EQU    $01E7
               01FF  BRAN     EQU    $01FF
               020B  ZBRAN    EQU    $020B
               0211  ZBYES    EQU    $0211
               021A  ZBNO     EQU    $021A
               0A67  QPAIRS   EQU    $0A67
               0AAE  COMPIL   EQU    $0AAE
               08D8  HERE     EQU    $08D8
               076B  ZERO     EQU    $076B
               08F4  COMMA    EQU    $08F4
               0679  SWAP     EQU    $0679
               077B  TWO      EQU    $077B
               0FDD  BCOMP    EQU    $0FDD
               149F  ENDIF    EQU    $149F
               066B  DROP     EQU    $066B
               148D  BEGIN    EQU    $148D
               01BB  SPAT     EQU    $01BB
               0921  EQUAL    EQU    $0921
               05FE  ZEQU     EQU    $05FE
               158F  WHILE    EQU    $158F
               153D  REPEAT   EQU    $153D
               06BF  STORE    EQU    $06BF
                     *
                     *
                     * Colon Definition of CASE
                     *
                     * : CASE ?COMP CSP @ !CSP 4 ; IMMEDIATE
                     *
 1DAF                         WORDM  4,CAS,E,IMMEDIATE
               1DAF  NEXTNM   SET    *
 1DAF C4                      FCB    4+$C0
 1DB0 43 41 53                FCC    'CAS'
 1DB3 C5                      FCB    $80+'E
 1DB4 1D93                    FDB    LASTNM
               1DAF  LASTNM   SET    NEXTNM
                              ENDM
 1DB6 0073 0A3A      CASE     FDB    DOCOL,QCOMP,CSP,AT,SCSP,LIT,$0004,SEMIS
 1DBA 08AC 06A7
 1DBE 0A0D 01E7
 1DC2 0004 0080
                     *
```

```
                   *
                   * Primitive Definition of (OF)
                   *
1DC6                       WORDM  4,(OF,)
             1DC6  NEXTNM  SET    *
1DC6 C4                    FCB    4+$C0
1DC7 28 4F 46              FCC    '(OF'
1DCA A9                    FCB    $80+')
1DCB 1DAF                  FDB    LASTNM
             1DC6  LASTNM  SET    NEXTNM
                           ENDM
1DCD 1DCF          POF     FDB    *+2
1DCF 37    06              PULU   D          Get value that preceeded OF
1DD1 10A3 C4               CMPD   U          Compare with select value
1DD4 26    05              BNE    NXTOF      Not equal- continue to next
1DD6 33    42              LEAU   2,U        Equal - drop select value
1DD8 7E    021A            JMP    ZBNO       Skip branch , execute required words
1DDB 7E    0211    NXTOF   JMP    ZBYES      Branch to next value
                   *      JMP    NEXT     Never Executed
                   *
                   *
                   * Colon Definition of OF
                   *
                   * : OF 4 ?PAIRS COMPILE (OF) HERE 0 , 5 ; IMMEDIATE
                   *
                   * Note : 1 OF is an IMMEDIATE
                   *         2 When OF executes COMPILE compiles CFA of Primitive (OF)
                   *           into the FORTH word being defined.
                   *
1DDE                       WORDM  2,O,F,IMMEDIATE
             1DDE  NEXTNM  SET    *
1DDE C2                    FCB    2+$C0
1DDF 4F                    FCC    'O'
1DE0 C6                    FCB    $80+'F
1DE1 1DC6                  FDB    LASTNM
             1DDE  LASTNM  SET    NEXTNM
                           ENDM
1DE3 0073 01E7     OF      FDB    DOCOL,LIT,$0004,QPAIRS,COMPIL,POF,HERE
1DE7 0004 0A67
1DEB 0AAE 1DCD
1DEF 08D8
1DF1 076B 08F4             FDB    ZERO,COMMA,LIT,$0005,SEMIS
1DF5 01E7 0005
1DF9 0080
                   *
                   *
                   * Colon Definition of ENDOF
                   *
                   * : ENDOF 5 ?PAIRS COMPILE BRANCH HERE 0 ,
                   *          SWAP 2 [COMPILE] ENDIF 4 ; IMMEDIATE
                   *
                   * Note : ENDOF is ' DEFINED ' into the dictionary not Compiled
                   *         Thus [COMPILE] ENDIF becomes FDB   ENDIF
```

```
                         *
 1DFB                           WORDM  5,ENDO,F,IMMEDIATE
               1DFB  NEXTNM     SET    *
 1DFB C5                        FCB    5+$C0
 1DFC 45 4E 44 4F               FCC    'ENDO'
 1E00 C6                        FCB    $80+'F
 1E01 1DDE                      FDB    LASTNM
               1DFB  LASTNM     SET    NEXTNM
                                ENDM
 1E03 0073 01E7      ENDOF      FDB    DOCOL,LIT,$0005,QPAIRS,COMPIL,BRAN
 1E07 0005 0A67
 1E0B 0AAE 01FF
 1E0F 08D8 076B                 FDB    HERE,ZERO,COMMA,SWAP,TWO
 1E13 08F4 0679
 1E17 077B
 1E19 149F 01E7                 FDB    ENDIF,LIT,$0004,SEMIS
 1E1D 0004 0080
                         *
                         *
                         * Colon Definition of ENDCASE
                         *
                         * : ENDCASE 4 ?PAIRS COMPILE DROP BEGIN SP@ CSP @ = 0=
                         *   WHILE 2 [COMPILE] ENDIF REPEAT CSP ! ; IMMEDIATE
                         *
                         * Note : In an FDB definition the BEGIN WHILE REPEAT structure
                         *        becomes ZBRAN,(offset) together with BRAN,(offset)
                         *        In an FDB definition [COMPILE] ENDIF becomes FDB  ENDIF
                         *
 1E21                           WORDM  7,ENDCAS,E,IMMEDIATE
               1E21  NEXTNM     SET    *
 1E21 C7                        FCB    7+$C0
 1E22 45 4E 44 43               FCC    'ENDCAS'
 1E26 41 53
 1E28 C5                        FCB    $80+'E
 1E29 1DFB                      FDB    LASTNM
               1E21  LASTNM     SET    NEXTNM
                                ENDM
 1E2B 0073 01E7      ENDCAS     FDB    DOCOL,LIT,$0004,QPAIRS,COMPIL,DROP
 1E2F 0004 0A67
 1E33 0AAE 066B
 1E37 01BB 08AC                 FDB    SPAT,CSP,AT,EQUAL,ZEQU,ZBRAN,$000A
 1E3B 06A7 0921
 1E3F 05FE 020B
 1E43 000A
 1E45 077B 149F                 FDB    TWO,ENDIF,BRAN,$FFEC
 1E49 01FF FFEC
 1E4D 08AC 06BF                 FDB    CSP,STORE,SEMIS
 1E51 0080
                         *
                         *
                         * The next code sets the link-back for this version
                         *
 1196                           ORG    $1196
```

```
ANTARCTIC RESEARCH (RHODES)          8-13-82 TSC ASSEMBLER   PAGE    5
ADDITIONS TO FORTH  KERNEL

  1196 1E21                    FDB    LASTNM
  2806                         ORG    $2806
  2806 1E21                    FDB    LASTNM
                               END

0 ERROR(S) DETECTED
```

* CASE Statement for FORTH *

CASE --- addr n (compiling)
Used in a colon definition of the form
CASE ... OF ... ENDOF ... ENDCASE
OF .. ENDOF pairs may be repeated as necessary

At compile time CASE saves the current value of CSP and resets it to the current position of the stack. This information is used by ENDCASE to resolve forward references left on the stack by any ENDOF's before it. n is left for subsequent error checking.

CASE has no run time effects.

OF --- addr n (compiling)
n1 n2 --- n1 (if no match)
n1 n2 --- (if n1 = n2)
Used in a colon definition of the form
CASE ... OF ... ENDOF ... ENDCASE

At compile time OF emplaces (OF) and reserves space for an offset at addr. addr is used by ENDOF to resolve the offset. n is used for error checking .

At run time,OF checks n1 and n2 for equality. If equal, n1 and n2 are both dropped from the stack,and execution continues to the next ENDOF. If not equal,n2 is dropped and execution jumps to whatever follows the next ENDOF.

ENDOF addr1 n1 --- addr2 n2 (compiling)
Used in a colon definition of the form
CASE ... OF ... ENDOF ... ENDCASE

At compile time ENDOF emplaces BRANCH reserving a branch offset saves addr2 on the stack and also n2 for error checking. ENDOF also resolves the pending forward branches from OF by calculating the offset from addr1 to here and storing it at addr1.

At run time ENDOF transfers control to the code following the next ENDCASE provided there was a match at the last OF. If there was no match at the last OF , ENDOF is the location to which execution will branch.

```
ENDCASE addr1 ... addrn n --- ( compiling )
                        n --- ( if no match )
                          --- ( if match found )
```

Used in a colon definition of the form
CASE ... OF ... ENDOF ... ENDCASE

At compile time ENDCASE compiles a drop then computes forward branch offsets until all addresses left by previous ENDOF's have been resolved. Finally the value of CSP saved by CASE is restored. n is used for error checking.

At run time ENDCASE drops the select value if it does not equal any case values. ENDCASE then serves as the destination of forward branches from all previous ENDOF's.

Example

```
: PRINTDAY CASE            1 OF ."Monday"     ENDOF
  2 OF ." Tuesday"  ENDOF 3 OF ." Wednesday" ENDOF
  4 OF ." Thursday" ENDOF 5 OF ." Friday"    ENDOF ENDCASE ;
```

1 PRINTDAY will print Monday
4 PRINTDAY will print Thursday
9 PRINTDAY will not do anything !

+++

```
   ASMB,FORTHAD,UNSIGNED,+P2,+1E21,1E53,UNSIGNED
DELETE OLD BINARY (Y-N)? Y


ANTARCTIC RESEARCH (RHODES)         11-11-82 TSC ASSEMBLER   PAGE    2
ADDITIONS TO FORTH  KERNEL


                        *
  1E53                          ORG    $1E53
                        *
                        *
                        *** Unsigned Comparison Operatives ***
                        *
                        * u1 u2 - - - f
                        * True Flag if u1 < u2
                        *
                        *** Primitive U< ***
                        *
  1E53                          WORDM  2,U,<
                  1E53  NEXTNM  SET    *
  1E53 82                       FCB    2+$80
  1E54 55                       FCC    'U'
  1E55 BC                       FCB    $80+'<
  1E56 1E21                     FDB    LASTNM
                  1E53  LASTNM  SET    NEXTNM
                                ENDM
  1E58 1E5A             UNLESS  FDB    *+2
  1E5A 37    06                 PULU   D         Get u2 to A and B
  1E5C A1    C4                 CMPA   0,U       Compare MSB's
  1E5E 22    09                 BHI    ULESST    Branch if MSB u1 < MSB u2
  1E60 26    04                 BNE    ULESSF    Branch if MSB u1 # MSB u2
  1E62 E1    41                 CMPB   1,U       Compare LSB's
  1E64 22    03                 BHI    ULESST    Branch if LSB u1 < LSB u2
  1E66 5F               ULESSF  CLRB             u1 not less than u2
  1E67 20    02                 BRA    ULESSX
  1E69 C6    01         ULESST  LDB    #1        u1 less than u2
  1E6B 4F               ULESSX  CLRA
  1E6C ED    C4                 STD    ,U
  1E6E 0E    77                 JMP    NEXT
                        *
                        *** Colon Definition U> ***
                        *
                        * u1 u2 - - - f
                        * True Flag if u1 > u2
                        *
                        * : U> SWAP U< ;
                        *
                  0679  SWAP    EQU    $0679
                        *
  1E70                          WORDM  2,U,>
                  1E70  NEXTNM  SET    *
  1E70 82                       FCB    2+$80
  1E71 55                       FCC    'U'
  1E72 BE                       FCB    $80+'>
  1E73 1E53                     FDB    LASTNM
                  1E70  LASTNM  SET    NEXTNM
                                ENDM
  1E75 0073 0679        UGREAT  FDB    DOCOL,SWAP,UNLESS,SEMIS
  1E79 1E58 0080
```

```
ANTARCTIC RESEARCH (RHODES)        11-11-82 TSC ASSEMBLER   PAGE    3
ADDITIONS TO FORTH  KERNEL

                         *
                         *
                         * The next code sets the link-back for this version
                         *
 1196                           ORG     $1196
 1196 1E70                      FDB     LASTNM
 2806                           ORG     $2806
 2806 1E70                      FDB     LASTNM
                                END

0 ERROR(S) DETECTED
```

*   Single Precision Unsigned Comparison FORTH Words   *

FORTH uses the comparison words "<" and ">" for checking the limits of the Dictionary and Data Stack.
"<" and ">" operate on signed numbers only. For operation above $7FFF the unsigned operators "U<" and "U>" were defined.

U<   u1 u2 --- f

True if u1 less than u2

u1 and u2 are 16 bit numbers (ie unsigned)

U>   u1 u2 -- f
True if u1 is greater than u2

u1 and u2 are 16 bit numbers (ie unsigned)

```
+++ASMB,CHANGES
DELETE OLD BINARY (Y-N)? Y
                         *
                                   TTL    ANTARCTIC RESEARCH (RHODES)
                                   STTL   CHANGES TO FORTH KERNEL
                                   OPT    PAG
                                   PAG


ANTARCTIC RESEARCH (RHODES)        2-22-83 TSC ASSEMBLER   PAGE    1
CHANGES TO FORTH KERNEL


                         *
                         ***                   Changes to                     ***
                         ***   68'FORTH for 6809 Version 01.01 (Talbot Microsystems) ***
                         ***                    used for                      ***
                         ***          Vertichirp Controller Programming       ***
                         *
                         **    Operation above 32K     ***
                         *
                         * FORTH uses comparison words < and > which operate on signed numbers
                         * for checking the limits of the Dictionary and Stack
                         * < and > are used by ?STACK and FORGET
                         * To be able to use Dictionary and Stack memory above $8000
                         * unsigned comparison operators U< and U> have been defined
                         *
                         * Change to ?STACK
                         *
 0C69                              ORG    $0C69
 0C69 1E58                         FDB    $1E58    was $05A3 ie < , changed to U<
 0C78                              ORG    $0C78
 0C78 1E58                         FDB    $1E58    was $05A3 ie < , changed to U<
                         *
                         * Change to FORGET
                         *
 144A                              ORG    $144A
 144A 1E58                         FDB    $1E58    was $05A3 ie < , changed to U<
 1459                              ORG    $1459
 1459 1E75                         FDB    $1E75    was $092D ie > , changed to U>
                         *
                                   PAG
```

```
                        *
                        ***   Changes to FORTH Memory Map   ***
                        *
 0023                         ORG    $0023
 0023 282E              FENCIN  FDB    $282E      was $207E - initial fence at TASK
 0025 2839              DPINIT  FDB    $2839      was $2089 - cold start for DP
 0027 2810              VOCINT  FDB    $2810      was $2060 - cold start for VOC-LINK
 0033                         ORG    $0033
 0033 23EF              XVIRBG  FDB    $23EF      was $1BF0 - disk buffer beginning
 0035 27FF              XVIRED  FDB    $27FF      was $2000 - disk buffer end
 0037 C000              XDSMBG  FDB    $C000      was $3000 - sim disk,not used
 0039 C000              XDSMED  FDB    $C000      was $4000 - sim disk,not used
 003B BF30              SINIT   FDB    $BF30      was $2F30 - initial top of Data Stack
 003D BF30              TIBINT  FDB    $BF30      was $2F30 - terminal input buffer
 003F C000              RINIT   FDB    $C000      was $3000 - initial top of return stack
 115F                         ORG    $115F
 115F 2808                      FDB    $2808      was $2058 - reference to FORTH
 119E                         ORG    $119E
 119E 282E                      FDB    $282E      was $207E - VOC-LINK in copied code
 11C3                         ORG    $11C3
 11C3 2800                      FDB    $2800      was $2050 - TASK backlink in copied code
                        *
                        ***   Change to FORTH Word +LOOP    ***
                        *
                        * The code for +LOOP includes the incorrect instruction ANDCC #$1
                        * This clears the condition code register retaining
                        * only the carry flag if there is one .
                        * The Interrupt Mask , if set , is cleared when +LOOP is used
                        * The correct instruction should be ORCC #$1
                        *
 0244                         ORG    $0244
 0244 1A    01                ORCC   #$1        was ANDCC #$1
                        *
                              PAG
```

```
                  *
                  ***  Changes to FORTH Kernel to enable EPROMING   ***
                  *
                  * To inhibit loading of screen 43 during COLD start
                  *
 0138                     ORG    $0138
 0138 00          IFCOLD  FCB    $00        Was $FF
 0151                     ORG    $0151
 0151 00                  FCB    $00        Was $FF
                  *
                  * To move the system variables XUSE and XPREV
                  * from $004B,$004D to RAM at $23EA,$23EC
                  *
 23EA                     ORG    $23EA
 23EA             XUSE    RMB    2          XUSE  was $004B
 23EC             XPREV   RMB    2          XPREV was $004D
                  *
                  * Set references to XUSE,XPREV in COLD to NOPs
                  *
 015F                     ORG    $015F
 015F 1212                FDB    $1212
 0161 1212                FDB    $1212
                  *
                  * Change references to XUSE,XPREV in Disk Primitives
                  *
 11E2                     ORG    $11E2
 11E2 23EA                FDB    XUSE
 11ED                     ORG    $11ED
 11ED 23EC                FDB    XPREV
                  *
                  * To print error number only , inhibiting
                  * disk read for error messages
                  *
 0049                     ORG    $0049
 0049 0000        WRNINT  FDB    $0000      0 = No Disk
                  *
                  * When using disks error messages can be obtained
                  * by executing the following FORTH Word
                  * : DISK BASE @ HEX 1 2030 ! BASE ! ;
                  *
                  * To move the location of NUMTRY to RAM
                  * NUMTRY contains the number of trys at reading a disk
                  *
 23EE                     ORG    $23EE
 23EE             NUMTRY  RMB    1          NUMTRY was $1B12
                  *
 1B20                     ORG    $1B20
 1B20 23EE                FDB    NUMTRY
 1B32                     ORG    $1B32
 1B32 23EE                FDB    NUMTRY
 1B45                     ORG    $1B45
 1B45 23EE                FDB    NUMTRY
```

CHANGES TO FORTH KERNEL

```
                     *
                     *** Special Initialisation Subroutine called from COLD ***
                     *
                     * Subroutine Call in COLD
                     *
 01AF                        ORG    $01AF
 01AF BD    1FEA     INTSPC  JSR    SPINIT
                     *
                     * Equates used in Subroutine
                     *
            2038     XSCR    EQU    $2038
            2040     XSTATE  EQU    $2040
            204E     IOSTAT  EQU    $204E
                     *
                     * Subroutine situated at end of FORTH Kernel
                     *
                     * Initialise XUSE,XPREV
                     *
 1FEA                        ORG    $1FEA
 1FEA 8E    23EF     SPINIT  LDX    #$23EF    $23EF = XVIRBG
 1FED BF    23EA             STX    XUSE
 1FF0 BF    23EC             STX    XPREV
                     *
                     * Initialise important user variables
                     *
 1FF3 8E    0000             LDX    #$0000
 1FF6 BF    2038             STX    XSCR      Disk screen being accessed ( 0 = Terminal )
 1FF9 BF    2040             STX    XSTATE    Flag for interpret or compile modes
 1FFC BF    204E             STX    IOSTAT    Last ACIA status from write/read
 1FFF 39                     RTS
                             END

0 ERROR(S) DETECTED
```

* FORTH Memory Map expansion Procedure *

The following binary files are required on the working drive

1. CHANGES.BIN
2. DPRIMS.BIN
3. INTPRIMS.BIN
4. CALPRIMS.BIN
5. VECPRIMS.BIN
6. RESTORE1.BIN
7. CASE.BIN
8. UNSIGNED.BIN

The FLEX command FORTH loads and runs FORTH Using the FORTH word "_" (underline) together with the FLEX command "GET" all the above files are loaded into memory.

ie.

```
_ GET CHANGES
_ GET DPRIMS
_ GET INTPRIMS
_ GET CALPRIMS
_ GET VECPRIMS
_ GET RESTORE1
- GET CASE
_ GET UNSIGNED
```

NB All the primitives in DPRIMS through to UNSIGNED must be loaded in the order above thus ensuring that the dictionary backlinks are set up correctly.

Typing the FORTH command "COLD" will set up the system using the addresses overlayed by CHANGES.BIN
Type VLIST to check that all the new primitives are in the dictionary.
Check the operation of FORTH and the added primitives before saving the modified version on disk.
This expanded Memory Map FORTH is saved to disk by

```
_ SAVE 1.FORTH.CMD,0000,2000,0
```

+++

```
 *   Expanded FORTH Memory Map   *

$0000 --> $1BD0 = FORTH Kernel

$1BD1 --> $1BDF = empty

$1BE0 --> $1CBA = double precision words
$1CBB --> $1CED = interrupt Words
$1CEE --> $1D6C = MPN Calculator words
$1D6D --> $1D92 = the word VECTOR
$1D93 --> $1DAE = the word RESTORE1
$1DAB --> $1DAF = CASE statement words
$1E53 --> $1E7C = the words U< and U>
$1E7D --> $1FE9 = empty
$1FEA --> $1FFF = init. subroutine added by CHANGES

$2000 --> $204F = FORTH registers and pointers
                  and user variables.

$2050 --> $23E5 = empty

$23E6 and $23E7 = IRQ vector

$23E8 and $23E9 = empty

$23EA and $23EB = XUSE

$23EC and $23ED = XPREV

$23EE           = NUMTRY

$23EF           = empty

$23F0 --> $27FF = disk buffer

$2800 --> $2838 = "FORTH" and "TASK"

$2839 -->       = beginning of dictionary ( grows up )


$BF30           = data stack ( grows down )

$BF30 --> $BFB4 = TIB ( terminal input buffer )

$BFFF           = RETURN stack base ( grows down )
----------------------------------------------------

$C000 --> $DFFF = RAM for FLEX 9 ( DOS )
```

```
*    Overall Memory Map for 6809     *

$0000 --> $BFFF = FORTH ( expanded version )

$C000 --> $DFFF = for FLEX Disk operating system

$E000           =  I/O address beginning

$E000 --> $E003 = MPN Calculator Board

$E004 --> $E007 = MPS Serial interface board

$E008 --> $E00B = MPLA1 // port -  hardware control

$E00C --> $E00F = MPLA2 // port -  clock and DCS

$E010 --> $E013 = not used

$E014 --> $E017 = jumpered for disk controller

$E018 --> $E01B = DC3 disk controller board

$E01C --> $E01F = MPLA3 // port - film I.D.

$E020 -->$EFFF = empty

$F000 --> $F7FF = empty

$F800 --> $FFFF = SBUG MONITOR
```

S-BUG Monitor Modification

The SWTPC 6809 monitor was modified as follows for use in the Vertichirp Controller system.

Interrupt Vectors

Locations	Name	Old vectoring address	New vectoring address	Function
FFFE,FFFF	Restart	FF00	FF00	Set DAT FORTH warm start
FFFC,FFFD	NMI	FF00	0000	FORTH cold start

The MP-09 processor board contains an on board Dynamic Address Translator (DAT) that divides the computer's 64K address space into 16 4K blocks. Each memory location has therefore a physical address and a logical address. The S-BUG monitor sets the DAT to position the highest physical 4K block of RAM at logical address D000-DFFF. The next 4K block is positioned at C000-CFFF with subsequent blocks being logically assigned addresses from 0000 up to BFFF.

In the Vertichirp Controller system both FORTH and the Controller programmes are stored in EPROM in low memory form 0000. To prevent rearrangement of memory the S-BUG monitor was modified to set up the DAT to always assign addresses so that logical address = physical address.

SET DAT

```
FF0E                          ORG   $FF0E
FF0E 8E   DFD0                LDX   #$DFD0   X=ADDRESS OF DAT COPY
FF11 86   0F                  LDA   #$0F     SET PHYSICAL ADDRESS
FF13 A7   80   LOOP1          STA   0,X+     EQUAL TO LOGICAL ADDRESS
FF15 4A                       DECA           IN COPY
FF16 26   FB                  BNE   LOOP1
FF18 86   F0                  LDA   #$F0
FF1A A7   84                  STA   0,X      STORE $F0 AT $DFDF
FF1C 47                       INCA
FF1D 87   82                  STA   0,-X     STORE $F1 AT $DFDE
FF1F 8E   FFF0                LDX   #$FFF0
FF22 10   DFD0                LDY   #$DFD0
FF26 C6                       LDB   #$10
FF28 A6         LOOP2         LDA   0,Y+     COPY TABLE TO DAT
FF2A A7                       STA   0,X+
FF2C 5A                       DECB
FF2D 26   F9                  BNE   LOOP2
FF2F 53                       COMB
FF30 F7   DFE2                STB   $DFF2
FF33 10CE DEC0                LDS   #$DFC0    SET HARDWARE STACK
FF37 16   F8DA                LBR   $F814
```

Call to code checking if FORTH is in EPROM.

```
F85E                       ORG    $F85E
F85E 17    067F            LBSR   $FEE0
```

The following code checks locations 0000 , 0001 , 0002 and 0003 during power on initialization to see if the system has FORTH in EPROM. If it is a jump to FORTH "warm start" is executed. If not execution continues to S-BUG "next command" routine.

RUN S-BUG OR FORTH

```
FEE0 17    FECB            LBSR   $FDAE      PRINT "K "CRLF
FEE3 CC    1601            LDD    #$1601     FORTH 1ST 2 BYTES
FEE6 10    9300            CMPD   $00        FORTH ?
FEE9 27    01              BEQ    CHECK      YES,CHECK NEXT 2 BYTES
FEEB 39                    RTS               NO,GO TO S-BUG "NXT CMD"'
FEEC CC    3F16            LDD    #$3F16     FORTH BYTES 3 AND 4
FEEF 10    9302            CMPD   $02        FORTH?
FEF2 27    01              BEQ    FORTH!
FEF4 39                    RTS
FEF5 7E    0003 FORTH!     JMP    $0003      GO TO FORTH WARM
```

NOTE: BOTH FORTH COLD AND WARM STARTS RE-POSITION THE STACKS.

S-BUG MONITOR MODIFICATIONS

1. S-BUG OCCUPIES memory from $F800-$FFFF.It is saved on disk as a binary file using the FLEX command "SAVE".
2. The FLEX command "FIX" is used to change the binary file to the code given above.
3. The modified version called S-BUG FTH is used to program the new EPROM.

Execution of Controller Programs

The FORTH word ABORT is modified to execute the word COLD? which checks for prior system initialization and executes either VCOLD or VWARM before continuing with the displaced word QUIT which is the outer interpreter.

* Vertichirp Controller Programme EPROMMING Procedure *

Load expanded memory map version of FORTH

```
Set FORTH Vocabulary        - FORTH DEFINITIONS
Forget the word "TASK"      - FORGET TASK
Select drive 1              - DR1
DR1 head restore            - RESTORE1
Empty disk buffers          - EMPTY-BUFFERS
Load Controller Programmes  - 1 LOAD
```

The last screen (SCR 86) executes the following

```
HEX 982E 0023 ! 9839 0025 ! 9810 0027 ! 9808 115F !
' COLD? CFA 1177 ! ' SETTIME NFA 1196 !
    982E 119E ! 9800 11C3 !
```

This modifies the Kernel to move "FORTH" and "TASK"
above the Controller programmes and redirect startup
to Controller warm start (VWARM)
To check these changes either press the COLD Reset button
or type the FORTH command "COLD".
"FORTH" and "TASK" will be swapped to RAM beginning at $9800
Execution will then proceed to the Controller word "COLD?"
which will prompt for a Controller cold start.
"VLIST" should show "TASK" then "FORTH" at the top of
the dictionary followed by the last Controller word "SETTIME"
and all the words that preceeded it.
The first Controller word should be preceded by "FORTH"
"U>" "U<" ..... etc

If all the above is correct and the programme is operational
it is saved as a binary file from FORTH as follows

```
Set IRQ interrupt mask bit - SEIF
Reset I/O ports            - RESET
```

Place FLEX system disk in drive 0
Place disk with at least 157 empty sectors in drive 1

```
_ CAT,1
```

will show the catalog of drive 1 followed
by the number of remaining sectors on the disk.

```
_ SAVE 1.FORTH.ROM,0000,9800,0
```

The file FORTH.ROM is used for programming the EPROM's.

```
  *   VERTICHIRP CONTROLLER MEMORY MAP   *

$0000 --> $1BD0 = FORTH Kernel

$1BD1 --> $1BDF = empty                            8K EPROM

$1BE0 --> $1FFF = Kernel additions using the
                  programme FORTHAD go here.

$2000 --> $204F = FORTH registers and pointers
                  and user variables.

$2050 --> $23E5 = temporary storage and flags

$23E6 and $23E7 = IRQ vector

$23E8 and $23E9 = empty
                                                   2K RAM
$23EA and $23EB = XUSE

$23EC and $23ED = XPREV

$23EE           = NUMTRY

$23EF           = empty

$23F0 --> $27FF = disk buffer


$2800 --> $97FF =  CONTROLLER dictionary          28K EPROM

$9800 -->       = "FORTH" and "TASK"

$9839 -->       = beginning of dictionary

$BF30           = data stack ( grows down )       10K RAM

$BF30 --> $BFB4 = TIB ( term. input buffer )

$BFFF           = RETURN stack ( grows down )

--------------------------------------------

$C000 --> $DFFF = RAM for Flex 9 ( DOS )           8K RAM

$E000 --> $E01F = I/O Addresses
$E020 --> $F7FF = empty
$F800 --> $FFFF = modified S-BUG , S-BUG FTH      2K EPROM
```

APPENDIX D

VERTICHIRP CONTROLLER SOFTWARE

OPERATORS MANUAL

G.P.EVANS

ANTARCTIC RESEARCH - RHODES UNIVERSITY

DECEMBER 1983

APPENDIX D

VERTICHIRP_CONTROLLER_SOFTWARE

OPERATORS_MANUAL

G.P.EVANS

ANTARCTIC RESEARCH - RHODES UNIVERSITY

DECEMBER 1983

Vertichirp Controller Software

Operators Manual

Contents — Page Number

Vertichirp Controller Software - Operators Manual

Introduction

This manual describes the initialization of the Vertichirp Controller System and the commands used for system control , data entry , data display and modification and function execution.

System Initialization

The conversion from Logchirp Control to the Vertichirp Controller is detailed on page 85 of Vertichirp Controller Hardware - Manual 1.

Important :- Check that the toggle switch mounted under the VDU is in the " Controller " position.

When connected as required the AC power to the system can be switched on. The terminal should respond with :-

```
68' FORTH - 09 VERS #1.1

Heath VDU (Y or N)?
```

If this prompt does not appear press the WARM reset followed by the COLD reset on the Vertichirp Controller front panel.

The response to any prompt takes the form of valid data or a command followed by a carriage return (CR). If the VDU was a Heath the response Y (CR) would begin the initialization procedure by dividing the screen into three areas :-

Lines 1 to 6 = System Status Area
Lines 8 to 18 = Data Display Area
Lines 20 to 24 = Command and Data entry area

Any other VDU or teletype can be used by answering N (CR) to the first prompt. Separate areas are not allocated and input and output data and commands are listed one after another. The VDU used must be set up so that the screen scrolls up when the bottom line is reached.

Initialization continues with requests for Place Name , Year and Day of the Week. The final instruction is to set the Real Time Clock (RTC) using the switches on the Controller front panel.

Set Real Time Clock

The clock data is displayed on the Controller front panel as :-

```
        PL   Y10  Y1   D100 D10  D1   H10  H1   M10  M1   S10  S1
```

The place identifier , PL , and tens of years , Y10 , are set using the DIL switches on the Real Time Clock Display board , RTCD.

All the other digits except seconds are changed by using the front panel rotary switch to select the digit to be incremented each time the toggle switch is pressed. The rotary switch inhibits the clock and sets the seconds to zero. Because a carry increments the next significant digit it is easier to set the clock by beginning with unit minutes and ending with unit years.

The clock should be set ahead of the reference time and then enabled when the reference time is judged to be equal to the set time by moving the rotary switch to ' Off '. The final timing adjustments are described under the heading ' Set Timing Position '.

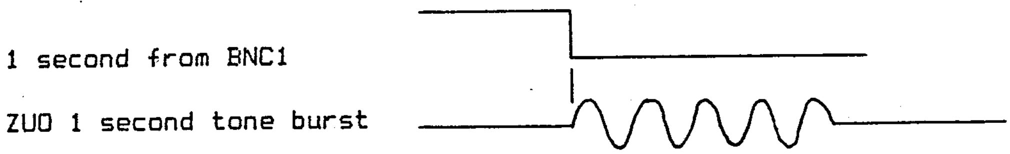
The VDU space bar can then be pressed to signal the computer to read the time. Once read , the screen status area is titled as follows :-

```
ANTARCTIC RESEARCH           VERTICHIRP CONTROLLER        RHODES UNIVERSITY
                                  GRAHAMSTOWN 1983

Day Number          :- 349

Current Programme :- Not Set!

Timing Position     :- 0000.00 ms Advanced --->
```

The time can be checked using the command TIME which causes the time to be displayed in the status area.

Set Timing Position

All that now remains is to set the system timing relative to some standard time signal such as ZUO. This is achieved by displaying both the Controller 1 second signal from connector BNC1 and the ZUO signal on a dual channel oscilloscope. The falling edge of the Controller 1 second square wave from BNC1 must be moved to coincide with the beginning of the reference signals one second marker tone burst. This is shown below :-



The Vertichirp Controller timing is adjusted using the following command format :-

(shift in ms in the form xxx.xx) space (MSEC) space (Adv or Rtd Command)

Note. The number specifying the shift must include a decimal point .

The Advance and Retard commands are :-

A@1 = advance at 1 ms/s

A@10 = advance at 10 ms/s

A@20 = advance at 20 ms/s

R@1 = retard at 1 ms/s

R@10 = retard at 10 ms/s

R@20 = retard at 20 ms/s

Examples

Coarse timing adjustment (see SEARCH for an alternative method):-

60.0 MSEC A@20 Advances 60 ms at a rate of 20 ms/s

Fine timing adjustment :-

0.25 MSEC R@1 Retards .25 ms at a rate of 1 ms/s

NB. Check that the Controller timing is adjusted in the correct direction.

When the Controller 1 second falling edge coincides with the beginning of the reference time signal the Timing Position , which is displayed in the status area , is set to zero by the command CLRPOSN. Subsequent timing adjustments displayed in the status area are relative to the position at which the command CLRPOSN was issued.

Note that the 105B Quartz Oscillator which provides the clock signals for the system will drift relative to the radio time signal (eg. ZUO). This means that after a number of days with the timing position = 0000.00 ms the actual position of the system in time will be slightly advanced or retarded relative to the radio time signal.

This drift can be minimised by adjusting the 105B Quartz Oscillator Coarse or Fine frequency adjust controls.

Abbreviated Command List

HELP displays the following abbreviated list of commands together with the function each performs.

```
HELP

                     -Command-                 -Function-
System               COLD                      Cold Start
                     WARM                      Warm Start
                     VDU                       VDU I/O
                     PRNT                      Printer or VDU I/O
                     TITLE                     Shows Status
                     STATUSOFF                 No Updates
                     TIME                      Shows Time
                     SETTIME                   Sets Time
Data Entry           SOUNDING  x               x = User assigned Data Name
                     IONOGRAM  x               x = User assigned Data Name
                     TIMING    x               x = User assigned Data Name
                     FORTHWORD x               x = User assigned Data Name
                     PROGRAMME x               x = User assigned Data Name
Data Display/Modify  EDIT    x                 x = Defined Data Name
                     DISPLAY x                 x = Defined Data Name
                     PRINT   x                 x = Defined Data Name
                     LOG     x                 x = Defined Data Name
                     DLIST                     List Data Names
Execution            SET x                     x = Defined Data Name
                     RUN x                     x = Defined Data Name
                     HALT                      Stop Ionogram

Press SPACE BAR to exit
```

These commands are described in more detail below. Related commands not in the list are also included.

System Commands

HELP

Display abbreviated command list until space bar is pressed.

COLD

Cold start. All previously entered data are lost. The system prompts for initialization.

WARM

Warm start. All previously entered data are retained. The screen is titled showing the status of the system before the warm start.

VDU

Flag VDU? is set to 1. Data are positioned on the Heath VDU screen using cursor addressing. The Heath screen is divided into 3 different areas which are used for system status display , data display and command or data entry respectively.

PRINTER (or PRNT)

Flag VDU? is cleared to 0. Cursor addressing is replaced by carriage return only. This command selects printer or VDU I/O with commands , prompts and data being listed line by line. The VDU must be set up to scroll up a line when the last line has been filled. The top line is lost but a new line is created at the bottom of the screen.

CLRVDU

VDU :- The screen is cleared and the cursor placed at the top left hand corner.

PRINTER :- A Carriage return only is output.

TITLE

VDU :- The screen is cleared then titled and the current status displayed

PRINTER :- The title followed by the current status is printed.

STATUSOFF

Flag STAT is cleared to 0. The status of the current and next functions in the current programme are not updated on either VDU or printer.

STATUS

Flag STAT set to 1. The status of the current and next functions in the current programme are updated on either VDU or printer.

TIME

VDU :- The words " Current Programme " in the status area are replaced by the words " Current Time " followed by the time in the form :-

day of week : hours : minutes : seconds. The time is updated every second.

PRINTER :- The current time is printed once only , immediately after the the unit seconds change.

SETTIME

This command instructs the user to set the hardware real time clock. Once set pressing the space bar signals the computer to read the hardware clock.

SETPL

Prompts for and sets the place name.

SETYR

Prompts for and sets the year.

SETDY

Prompts for and sets the day of the week.

PAUSE

Pauses the execution of the current programme until the space bar is pressed. After a PAUSE the next function in the current programme is set up for execution. PAUSE halts an executing Ionogram.

CANCEL

The current programme is cancelled. The status area shows :-

" Current Programme :- Not Set ! "

Oblique Synchronization

The following commands are used for oblique ionogram synchronization. The timing position relative to the position at which the command CLRPOSN was used is displayed in the status area in the form :-

Timing Position :- (last position in ms) ---> (present position in ms)

Commands

1. Specifying the shift required in milliseconds.

The command format is :-

(shift in ms in the form xxx.xx) space (MSEC) space (Adv or Rtd Command)

Note : The number specifying the shift must include a decimal point.

Advance Commands

A@1 = Advance at 1 ms/s

A@10 = Advance at 10 ms/s

A@20 = Advance at 20 ms/s

Retard Commands

R@1 = Retard at 1 ms/s

R@10 = Retard at 10 ms/s

R@20 = Retard at 20 ms/s

Example

29.0 MSEC R@10 ie: retard 29.0 milliseconds at a rate of 10 ms/s.

The status area would show :-

Timing Position :- 0000.00 ms Advanced ---> 29.00 ms Retarded

2. The SEARCH Command

The SEARCH command prompts only for a shift direction and rate. The changing timing position is shown in the status area. Slipping is terminated by pressing the space bar. The SEARCH mode is exited by typing EXIT. The timing position prior to the use of SEARCH can be easily returned to using the command LAST.

3. The LAST Command

LAST slips the timing at 20 ms/s to the last position.

Eg. Timing Position :- 0000.00 ms Advanced --> 29.00 ms Retarded

After typing LAST the timing position status would be :-

Timing Position :- 29.00 ms Retarded ---> 0000.00 ms Advanced

Data Entry Commands

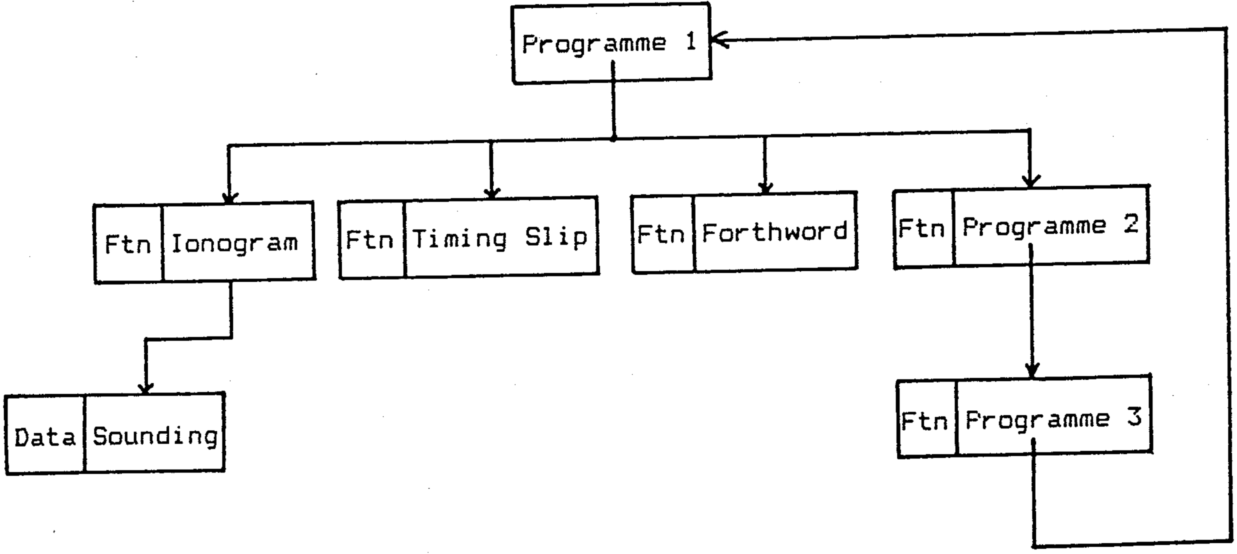
Introduction

There are five different types of data. They are Soundings , Ionograms , Timing Changes , FORTHWORDS and Programmes.

Each entry of a particular data type is given a unique descriptive name by the user. The computer then prompts for all the data required by that particular data type. Once all the prompts have been answered all the entered data is displayed . Any errors during entry can be corrected using the EDIT command.

Of the five data types four can be entered as functions in a Programme. The functions are Ionograms , Timing Changes , Forthwords and Programmes. These can all be executed using the RUN command.

A Sounding contains data used by an Ionogram and must therefore be defined before the Ionogram. A sounding cannot be RUN.



SOUNDING x

x is a user assigned Sounding data name.

Example

The command SOUNDING S1 will prompt for all the data necessary to define a Sounding with the name S1.

The data contained in a Sounding are the " microscopic " details of an Ionogram like basic rate , cell period and FFT sample rate. Once a Sounding has been defined the Ionogram that uses it may be defined.

An example of programming a Sounding is given below. The I/O mode was set to PRINTER to make this listing. User responses , all followed by (CR) , are highlighted by rectangular boxes.

```
SOUNDING S1
Stationary Doppler (Y or N)?N
Valid Data :- 25 40 50 80 100 200 400 250 500 800 1000 2000 4000 kHz/sec
Basic Rate  =50
Valid Data :- 0 to 950 Km
Height Range= 0 to 1500 Km
New Minimum Height Range=0


New Height Range= 0 to 1500 Km
Valid Data :- 1/32 1/16 1/8 1/4 1/2 1/ 2/ 4/ Sec
Tcell        =1/
Valid Data :- 1/32 1/16 1/8 1/4 1/2 1/ 2/ 4/ 8/ 16/ 32/ Sec
Mseq         =1/
Valid Data :- 8 16 32 64 128 256 512 1024 Hz
FFT Rate     = 1024
Valid Data :- Between 2 and 150 mm/min
Film Speed  =6
Closest Speed = 6 mm/min
Fixed Receiver Gain (Y or N)?N
Number of Cells =1
Cell No = 1
Valid Data :- xxxx Hz, xxx.xxx kHz or xx.xxx MHz
Offset from Origin ( Number only ) =0 Units =HZ
Film Drive (Y or N)?Y
Valid Data :- 1 2 3 4 5 6 7 8
Rx1 Antenna =1
Rx2 Antenna =2
```

IONOGRAM x

x is a user assigned Ionogram data name.

Example

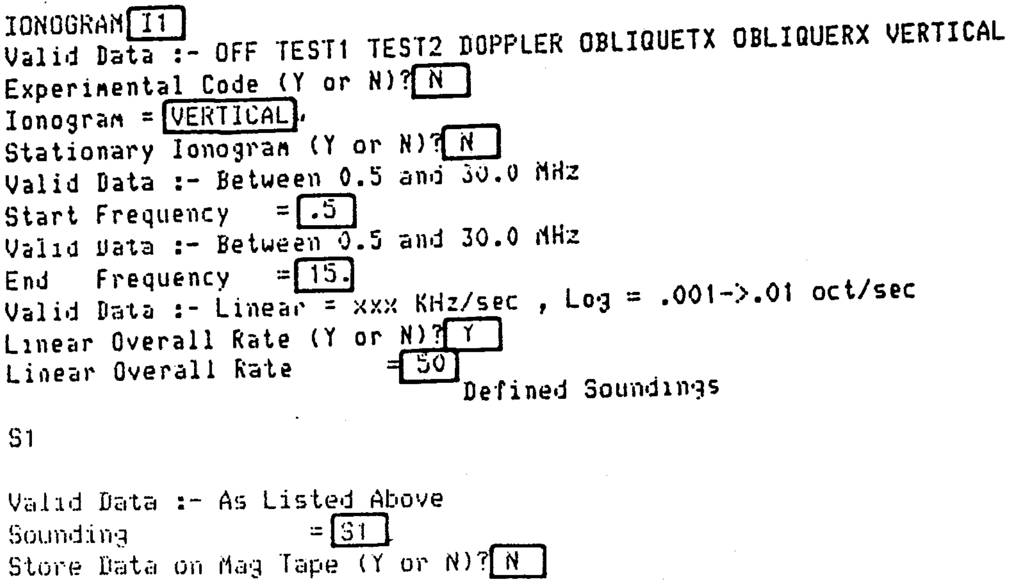
IONOGRAM I1 prompts for all the data necessary to define an Ionogram with the name I1.

Note

One of the requests in the defining of an Ionogram is for a previously defined compatible sounding. Make certain that one exists before entering Ionogram data.

The data contained in an Ionogram are the " Macroscopic " details of the Ionogram function such as Ionogram type (eg VERTICAL) , start frequency and end frequency.

An example of programming an Ionogram is given below. The listing was made with the I/O mode set to PRINTER. User responses , all followed by a carriage return (CR) , are highlighted by rectangular boxes.



```
IONOGRAM I1
Valid Data :- OFF TEST1 TEST2 DOPPLER OBLIQUETX OBLIQUERX VERTICAL
Experimental Code (Y or N)?N
Ionogram =VERTICAL
Stationary Ionogram (Y or N)?N
Valid Data :- Between 0.5 and 30.0 MHz
Start Frequency   =.5
Valid Data :- Between 0.5 and 30.0 MHz
End   Frequency   =15.
Valid Data :- Linear = xxx KHz/sec , Log = .001->.01 oct/sec
Linear Overall Rate (Y or N)?Y
Linear Overall Rate       =50
                                Defined Soundings

S1

Valid Data :- As Listed Above
Sounding                  =S1
Store Data on Mag Tape (Y or N)?N
```

TIMING x

x is a user assigned Timing Slip data name.

Example

TIMING T1 prompts for a required shift in milliseconds followed by a shift direction and rate to define a Timing Slip called T1.

An example of programming a Timing Slip is given below. User responses , all followed by (CR) , are highlighted by boxes.

```
TIMING T1
Valid Data :-xxx.xx ms
Shift = 29.0
Valid Data:- A@1 A@10 A@20 R@1 R@10 R@20
Shift Direction and Rate = A@20

                         T1 - TIMING SLIP ( ADVANCE )
Shift = 29.0 ms
Shift Direction and Rate = A@20 ms/s
OK
```

FORTHWORD x

x is a user assigned Forthword data name.

Example

FORTHWORD F1 prompts for a previously defined and tested FORTH colon definition which is assigned the Forthword data name F1. Forthwords so defined can be executed using the RUN command or can be entered in a Programme. This function must be used with caution.

An example of the use of FORTHWORD is given below.

1. FORTH colon definition:-

```
: CHKDY DYNO @ 300 = IF CLRDIS CR CR ." The day number is 300 !" HOME ENDIF ;
```

When executed CHKDY fetches the day number and checks to see if it is 300. If it is the display area is cleared and the message :-

' The day number is 300 ! ' is printed on the third line.

2. Data/Function definition using Controller command FORTHWORD.

```
FORTHWORD CHECKDAY

Valid Data :- TESTED FORTH Word !

FORTH Word = CHKDY
                                  CHECKDAY - FORTHWORD

                                  FORTH Word = CHKDY
```

PROGRAMME x (or PROGRAM x)

x is the user assigned Programme data name.

A Programme is a list of times and functions arranged in three priority blocks. Functions that can be executed under programme control are Ionograms , Timing Slips , Forthwords and Programmes. Each of the three priority blocks can have up to 24 entries. Each entry consists of a time associated with a function.

Note :- A Programme can call another Programme at a specific time adding great flexibility to Controller programming.

Priority Block #1 - Weekly Data

In this block week day , hours , minutes and seconds must all be specified together with the function to be executed. If no other function is running the specified function will be executed when the Real Time Clock and entry times are equal.

A function specified in this priority block will run once a week.

Priority Block #2 - Daily Data

In this priority block hours , minutes and seconds must all be specified together with the function to be executed. If no other function is running and there is no priority block #1 entry at the same time , the specified function will be executed when the real time clock and entry time are equal.

A function specified in priority block #2 will run once a day unless a priority block #1 entry overrides it.

Priority Block #3 - Hourly Data

In this block minutes and seconds must be specified together with the function to be executed. If no other function is running and there are no priority block #1 or priority block #2 entries at the same time , the specified function will be executed when the real time clock and entry times are equal.

A function specified in priority block #3 will run once an hour unless overridden by either a priority block #2 or priority block #1 entry.

Programme_Entry

PROGRAMME P1 assigns three empty priority blocks to the Programme name P1 and displays the first 8 empty locations of priority block #1. Times and functions can the be entered in any priority block using the programme editor. Allow a minimum of 5 seconds between functions for housekeeping.

An example of a programme is given below. The LOG x command prints all entries plus one empty entry per priority block. The example shows 4 entries each of the function I1 , a vertical Ionogram , and the function OBR , an oblique receive Ionogram , in priority block #3. These entries execute every hour except on Wednesdays between 14:10:00 and 14:55:00 when priority block #1 entries are executed and oblique receive ionograms alternate with oblique transmissions.

```
LOG P1
                              P1 - PROGRAMME

Priority Block #1       Day          Time           Function
Weekly Data
         Entry  1   Wednesday 14 : 10 : 00   OBT - IONOGRAM ( OBLIQUE TX )
         Entry  2   Wednesday 14 : 15 : 00   OBR - IONOGRAM ( OBLIQUE RX )
         Entry  3   Wednesday 14 : 20 : 00   OBT - IONOGRAM ( OBLIQUE TX )
         Entry  4   Wednesday 14 : 25 : 00   OBR - IONOGRAM ( OBLIQUE RX )
         Entry  5   Wednesday 14 : 30 : 00   OBT - IONOGRAM ( OBLIQUE TX )
         Entry  6   Wednesday 14 : 35 : 00   OBR - IONOGRAM ( OBLIQUE RX )
         Entry  7   Wednesday 14 : 40 : 00   OBT - IONOGRAM ( OBLIQUE TX )
         Entry  8   Wednesday 14 : 45 : 00   OBR - IONOGRAM ( OBLIQUE RX )
         Entry  9   Wednesday 14 : 50 : 00   OBT - IONOGRAM ( OBLIQUE TX )
         Entry 10   Wednesday 14 : 55 : 00   OBR - IONOGRAM ( OBLIQUE RX )
         Entry 11   --------- -- : -- : --   ---------

Priority Block #2       Day          Time           Function
Daily  Data
         Entry  1   --------- -- : -- : --   ---------

Priority Block #3       Day          Time           Function
Hourly Data
         Entry  1   --------- -- : 00 : 00   I1 - IONOGRAM ( VERTICAL )
         Entry  2   --------- -- : 05 : 00   OBR - IONOGRAM ( OBLIQUE RX )
         Entry  3   --------- -- : 15 : 00   I1 - IONOGRAM ( VERTICAL )
         Entry  4   --------- -- : 20 : 00   OBR - IONOGRAM ( OBLIQUE RX )
         Entry  5   --------- -- : 30 : 00   I1 - IONOGRAM ( VERTICAL )
         Entry  6   --------- -- : 35 : 00   OBR - IONOGRAM ( OBLIQUE RX )
         Entry  7   --------- -- : 45 : 00   I1 - IONOGRAM ( VERTICAL )
         Entry  8   --------- -- : 50 : 00   OBR - IONOGRAM ( OBLIQUE RX )
         Entry  9   --------- -- : -- : --   ---------
```

With STATUS selected the Current and Next Functions in the Current Programme are displayed and updated in the status area.

Data Display/Modify Commands

Introduction

These commands allow previously entered data to be displayed and changed.

EDIT x

x is the user assigned data name and may be a Sounding name , an Ionogram name , a Timing slip name , a Forthword name or a Programme name.

The EDIT command displays the data type x and the subtype as a heading. The user can then either change a parameter or , in the case of a Sounding or Programme , display other areas of data , or exit the editor. Selecting change results in a list of parameters that can be changed being printed. Upon selecting one of these the user is prompted for valid input.

DISPLAY x

x is the user assigned data name and may be one of the 5 different data types.

The DISPLAY command must be used on the Heath VDU only because it switches the I/O mode to VDU.

Sounding and Programme data are displayed in the EDIT format to allow the selection of different areas of data (Eg. Selection of different priority blocks in a Programme). Ionogram , Timing Slip and Forthword data are listed in the display area. I/O is left in the VDU mode.

PRINT x

x is the user assigned data name and may be one of the 5 different data types.

PRINT first switches the I/O mode to PRINTER. This command can be used with either a printer or with any VDU. If x is a valid name the data associated with it are printed. If x is a Programme name all entries plus one empty entry per priority block are printed.

I/O is left in the PRINTER mode.

LOG x

x is the user assigned data name and may be one of the 5 different data types.

LOG switches the I/O mode to PRINTER before printing the specified data. LOG is the same as PRINT except when x is the name of an Ionogram. In this case LOG prints the Ionogram data followed by the Sounding data referred to by the Ionogram.

<u>Example</u>

```
LOG I1
                          I1 - IONOGRAM ( VERTICAL )

Ionogram Duration =  0 :  4 : 50
Data to Tape      = No

Start Frequency   = 00.500000 MHz
End   Frequency   = 15.000000 MHz
Overall Rate      = 50 kHz/sec

Sounding          = S1
                        S1 - SOUNDING ( for IONOGRAM )

Basic Rate   = 50 kHz/sec
L.O. Offset = 0 Hz
Height Range= 0 to 1500 Km
Tcell        = 1/ sec
Mseq         = 1/ sec
FFT Rate     = 1024 Hz
Film Speed   = 6 mm/min
Rx AGC       = Clk/Snd
Time Marks   = Off

No of Cells = 1

             Offset       Film Drive      Rx1 Antenna       Rx2 Antenna

Cell 1    00.000000 MHz      Y                1                 2  OK
```

DLIST

Lists all the names entered since COLD reset , the most recent entry heading the list. The names listed include the data entered using the Controller commands SOUNDING , IONOGRAM , TIMING , FORTHWORD and PROGRAMME as well as any FORTH colon definitions.

The user can either exit DLIST or erase all names up to and including a particular name. This function must be used with caution because functions entered after a Programme and referred to by the Programme may be deleted. If this is done the invalid Programme entries will display the message :- " Invalid Forward Reference - DELETE ! ". All invalid entries must be deleted from the Programme.

```
DLIST
                              Defined Functions
        STAT/512          STAT/1024        SYNOP/512         S3/512
        S3/1024           P1               OI1               OS1
        I1                S1
Valid Data :- Function Name
Erase Function(Y or N)? N
 OK
```

SHOWS

SHOWS = Show Soundings

This command lists all the defined Sounding names , beginning with the most recent.

SHOWI

SHOWI = Show Ionograms

This command lists all the defined Ionogram names , beginning with the most recent.

SHOWT

SHOWT = Show Timing Slip

This command list all the defined Timing Slip names , beginning with the most recent.

SHOWF

SHOWF = Show Forthwords

This command lists all the Forthword names defined for execution in a Programme. The most recent definition heads the list.

Note: These words (datatype 4) are executed either by using the Controller command RUN or by entering them in a Programme.

SHOWP

SHOWP = Show Programmes

This command lists all the defined Programme names beginning with the most recent.

Function Execution

SET x

x is the user assigned data name and may be one of the 5 different data types.

In the case of Soundings and Ionograms the Controller hardware and software is set up according to the data referred to. Setting a programme makes it the Current Programme. Setting a Timing Slip or a Forthword only results in a check for an invalid forward reference.

After this command is used (Eg. when checking that the oblique Tx / oblique Rx antenna relay is switching) the Programme required as the Current Programme should be reset using the command RUN (Eg. RUN P1)

RUN x

x is the user assigned data name and may be an Ionogram , a Timing Slip , a Forthword or a Programme.

For x = an Ionogram name the Controller hardware is first set as required and the initial data is sent to the DCS. The Ionogram begins on the next falling edge of the 1 Hz clock signal.

If the DCS is not connected or does not acknowledge the data , the Controller VDU bell will sound.

For x = a Timing Slip name , the specified slip is excecuted beginning on the 1 Hz clock falling edge.

For x = a Programme name , the specified Programme is made the Current Programme and the next function to be executed in this Programme is found and set. The Current Programme name and the Next Function name are displayed in the status area.

If an Ionogram is executing and RUN x is used the Ionogram will be halted before the function x is executed.

HALT

This command is used to terminate an Ionogram before its end frequency is reached. The current sounding is completed and the Ionogram end code is sent to the DCS.

```
ANTARCTIC RESEARCH ( RHODES )        VERTICHIRP CONTROL PROGRAMS

                         * SCREEN INDEX *

  1 ( Reserved Memory )   ( *** Verticchirp Control Programs   *** )
  2 ( Reserved Memory )
  3 ( Reserved Memory )
  4 ( Reserved Memory )
  5 ( Reserved Memory )
  6 ( Reserved Memory )
  7 ( Reserved Memory )
  8 ( Reserved Memory )
  9 ( Reserved Memory )
 10 ( VDU / Printer Control )                                  HEX
 11 ( VDU / Printer Control )                                  HEX
 12 ( VDU / Printer Control )                              DECIMAL
 13 ( Number and Word Input )                                  HEX
 14 ( MPN Calculator Control )                                 HEX
 15 ( Port Initialisation )                                    HEX
 16 ( Synthesizer Control - Input MHZ , KHZ or HZ )            HEX
 17 ( Synthesizer Control - Print MHz/4 or MHz . Freq to MPN ) HEX
 18 ( Synthesizer Control - Freq to Synth.ANSEXP to Memory )   HEX
 19 ( Hardware Control - Control Register A )                  HEX
 20 ( Hardware Control - Control Register B )                  HEX
 21 ( Hardware Control - Basic Rate and Doppler )          DECIMAL
 22 ( Hardware Control - Tcell and M Sequence )                HEX
 23 ( Hardware Control - FFT Sample Rate and Antenna Switch )  HEX
 24 ( Hardware Control - Film Speed , Windowing and Sync's )   HEX
 25 ( Hardware Control - Advance Retard Control )              HEX
 26 ( Hardware Control - Advance Retard Control )              HEX
 27 ( Hardware Control - Advance Retard Search )           DECIMAL
 28 ( Datatype Handling )                                  DECIMAL
 29 ( Sounding Parameters )                                DECIMAL
 30 ( Sounding Parameters )                                DECIMAL
 31 ( Sounding Parameters )                                DECIMAL
 32 ( Sounding Parameters )                                DECIMAL
 33 ( Build Sounding )                                     DECIMAL
 34 ( Display Sounding )                                   DECIMAL
 35 ( Display Sounding )                                   DECIMAL
 36 ( Display Sounding - Cell Data )                       DECIMAL
 37 ( Display Sounding )                                   DECIMAL
 38 ( Edit Sounding - Prompts )                            DECIMAL
 39 ( Edit Sounding Parameter )                            DECIMAL
 40 ( Edit Sounding Parameter )                            DECIMAL
 41 ( Display and Edit Sounding )                          DECIMAL
 42 ( Set Sounding )                                       DECIMAL
 43 ( Ionogram Parameters )                                DECIMAL
 44 ( Ionogram Parameters )                                DECIMAL
 45 ( Ionogram Parameters )                                DECIMAL
 46 ( Build Ionogram )                                     DECIMAL
 47 ( Display Ionogram )                                   DECIMAL
 48 ( Display Ionogram )                                   DECIMAL
 49 ( Edit Ionogram - Prompts )                            DECIMAL
 50 ( Edit Ionogram Parameter )                            DECIMAL
 51 ( Display and Edit Ionogram )                          DECIMAL
 52 ( Ionogram Calculations )                              DECIMAL
 53 ( Ionogram Setting )                                   DECIMAL
 54 ( Ionogram Setting )                                   DECIMAL OK
```

```
ANTARCTIC RESEARCH ( RHODES )        VERTICHIRP CONTROL PROGRAMS

                          * SCREEN INDEX *

 55 ( Timing Slip Parameters - Build Timing Slip )          DECIMAL
 56 ( Display Timing Slip )                                 DECIMAL
 57 ( Edit Timing Slip , Set Timing Slip )                  DECIMAL
 58 ( FORTH Word Parameters - Build FORTH Word )            DECIMAL
 59 ( Display and Edit FORTH Word )                         DECIMAL
 60 ( Set Place , Year , Weekday Number )                       HEX
 61 ( Read Real Time Clock )                                    HEX
 62 ( Increment Real Time Clock - Get Time + 1 sec )        DECIMAL
 63 ( Display Day , Display Time )                          DECIMAL
 64 ( Allot Programme )                                     DECIMAL
 65 ( Programme Parameters )                                DECIMAL
 66 ( Display Programme )                                   DECIMAL
 67 ( Display Programme )                                   DECIMAL
 68 ( Display and Edit Programme )                          DECIMAL
 69 ( Programme Display Control )                           DECIMAL
 70 ( Display and Edit Programme )                          DECIMAL
 71 ( Programme Lookahead )                                 DECIMAL
 72 ( Programme Lookahead )                                     HEX
 73 ( Data to Film )                                            HEX
 74 ( Data to DCS )                                             HEX
 75 ( Data to DCS )                                             HEX
 76 ( Print Title and Display Status )                      DECIMAL
 77 ( Display Datatype. Edit Datatype )                     DECIMAL
 78 ( Set Datatype )                                        DECIMAL
 79 ( Show and Delete Datatype )                            DECIMAL
 80 ( Sounding and Cell Start Frequencies. Function Stop )  DECIMAL
 81 ( Tcell Interrupt Routine )                             DECIMAL
 82 ( Run Function )                                        DECIMAL
 83 ( Clock Service Routine )                               DECIMAL
 84 ( Interrupt Polling )                                       HEX
 85 ( Command List )                                        DECIMAL
 86 ( System Initialisation )                               DECIMAL OK
```

```
SCR # 1
  0         ( ****** Vertichirp Control Programmes ****** )          <
  1                                                                  <
  2 ( [N] = Hex Number of Bytes required by the Flag or Variable   ) <
  3   HEX                   ( *** Input / Output Control ***       ) <
  4   2050 CONSTANT STAT    ( 0 = Status not Displayed             ) <
  5                         ( 1 = Status Displayed             [2] ) <
  6   2052 CONSTANT VDU?    ( 0 = Carriage Return used - Printer   ) <
  7                         ( 1 = Cursor Addressing used - VDU [2] ) <
  8   2054 CONSTANT TFTN    ( 0 = Ftn Execution from Keyboard      ) <
  9                         ( 1 = Ftn Execution by Programme   [2] ) <
 10 ( 2056 CONSTANT IO1 ) (         Spare                      [2] ) <
 11 ( 2058 CONSTANT IO2 ) (         Spare                      [2] ) <
 12 ( 205A CONSTANT IO3 ) (         Spare                      [2] ) <
 13 ( 205C CONSTANT IO4 ) (         Spare                      [2] ) <
 14 ( 205E CONSTANT IO5 ) (         Spare                      [2] ) <
 15                                                             --> <


SCR # 2
  0 ( Reserved Memory )                                              <
  1   HEX                     ( *** Calculator Control ***         ) <
  2   2060 CONSTANT ANSEXP    ( MPN Exponent and Mantissa      [C] ) <
  3   2064 CONSTANT ANS#      ( MPN Mantissa                       ) <
  4   206C CONSTANT F1MSB     ( Formatted Answer               [8] ) <
  5   2074 CONSTANT F2MSB     ( Frequency Input Buffer         [8] ) <
  6                                                                  <
  7                           ( *** Control Registers ***          ) <
  8   207C CONSTANT CONTROLA  ( Control Register A             [2] ) <
  9   207E CONSTANT CONTROLB  ( Control Register B             [2] ) <
 10                                                                  <
 11                           ( *** Advance / Retard Control ***   ) <
 12   2080 CONSTANT SHIFT     ( Current Position  1/100 ms     [4] ) <
 13   2084 CONSTANT LPOSN     ( Last Position     1/100 ms     [4] ) <
 14   2088 CONSTANT SRCH      ( Flag , Set in Search Mode      [2] ) <
 15                                                             --> <


SCR # 3
  0 ( Reserved Memory )                                              <
  1   HEX                   ( *** Sounding Storage ***             ) <
  2   208A CONSTANT #4C     ( No. of Sets of 4 Cells           [2] ) <
  3   208C CONSTANT 4C#     ( No. of Current Set of 4 Cells    [2] ) <
  4   208E CONSTANT CELL#   ( Current Sounding Current Cell no.[2] ) <
  5   2090 CONSTANT AGC#    ( Flag and Count                   [2] ) <
  6                         ( 0 = Fixed Gain                       ) <
  7                         ( 1 = AGC Pulsed each Sounding         ) <
  8                         ( n = AGC Pulsed every n Tcells        ) <
  9   2092 CONSTANT TMK#    ( Counter for Time Marks           [2] ) <
 10                                                             --> <
 11                                                                  <
 12                                                                  <
 13                                                                  <
 14                                                                  <
 15                                                                  <
 OK
```

```
SCR # 4
  0 ( Reserved Memory )                                               <
  1   HEX                       ( *** Ionogram Storage ***          ) <
  2   2094 CONSTANT CURI        ( Current Ionogram Pfa          [2] ) <
  3   2096 CONSTANT CURS        ( Current Sounding Pfa          [2] ) <
  4   2098 CONSTANT #SOUND      ( No. of  Soundings in Ionogram [2] ) <
  5   209A CONSTANT SOUND#      ( Current Sounding no.          [2] ) <
  6   209C CONSTANT F/SND       ( Frequency change per Sounding [8] ) <
  7   20A4 CONSTANT EATS        ( e**ATS for Log Rates          [C] ) <
  8   20B0 CONSTANT CFREQ       ( Cell Frequency for DCS        [8] ) <
  9   20B8 CONSTANT CANT        ( Cell Antennas  for DCS        [2] ) <
 10                                                               --> <
 11                                                                   <
 12                                                                   <
 13                                                                   <
 14                                                                   <
 15                                                                   <


SCR # 5
  0 ( Reserved Memory )                                               <
  1   HEX                       ( *** Real Time Clock Storage ***   ) <
  2   20BA CONSTANT PLACE       ( Place Name 16 Ascii Characters [10] ) <
  3   20CA CONSTANT YEAR        ( Year 4 Ascii Characters       [4] ) <
  4   20CE CONSTANT LEAP        ( Set if Leap Year              [2] ) <
  5   20D0 CONSTANT DYNO        ( Year Day Number               [2] ) <
  6   20D2 CONSTANT WDAY        ( Week Day Number Sun=0 Mon=1 ... [2] ) <
  7   20D4 CONSTANT CDHMS       ( Clock     DHMS                [4] ) <
  8   20D8 CONSTANT CHMS        ( Clock      HMS                [4] ) <
  9   20DC CONSTANT CMS         ( Clock       MS                [4] ) <
 10   20E0 CONSTANT DHMS        ( Lookahead DHMS                [4] ) <
 11   20E4 CONSTANT HMS         ( Lookahead  HMS                [4] ) <
 12   20E8 CONSTANT MS          ( Lookahead   MS                [4] ) <
 13                                                               --> <
 14                                                                   <
 15                                                                   <


SCR # 6
  0 ( Reserved Memory )                                               <
  1   HEX                      ( *** Programme Entry ***            ) <
  2   20EC CONSTANT TP1        ( Addr of Top of Priority Block 1 [2] ) <
  3   20EE CONSTANT TP2        ( Addr of Top of Priority Block 2 [2] ) <
  4   20F0 CONSTANT TP3        ( Addr of Top of Priority Block 3 [2] ) <
  5   20F2 CONSTANT PB         ( Current Priority Block Number   [2] ) <
  6                            ( *** Programme Lookahead ***        ) <
  7   20F4 CONSTANT CURPRG     ( Addr of Current Programme       [2] ) <
  8   20F6 CONSTANT WC         ( Seconds until Next Weekly Iono  [4] ) <
  9   20FA CONSTANT WEA        ( Pfa of Next Weekly Ionogram     [2] ) <
 10   20FC CONSTANT DC         ( Seconds until Next Daily  Iono  [4] ) <
 11   2100 CONSTANT DEA        ( Pfa of Next Daily  Ionogram     [2] ) <
 12   2102 CONSTANT HC         ( Seconds until Next Hourly Iono  [4] ) <
 13   2106 CONSTANT HEA        ( Pfa of Next Hourly Ionogram     [2] ) <
 14                                                              --> <
 15                                                                  <
 OK
```

```
SCR # 7
 ( Reserved Memory )                                                    <
   HEX                    ( *** Programme Lookahead ***              ) <
   2108 CONSTANT NXTIME   ( Next Function Start Time in sec [4] ) <
   210C CONSTANT NXTFTN   ( Next Function Pfa                   [2] ) <
   210E CONSTANT NPB      ( Next Function Priority Block no.[2] ) <
   2110 CONSTANT CURTIME  ( Current Function Start Time in s[4] ) <
   2114 CONSTANT CURFTN   ( Current Function Pfa                [2] ) <
   2116 CONSTANT CPB      ( Current Function Priority Block [2] ) <
   2118 CONSTANT NXTSET   ( 0 = Next Ftn not set                [2] ) <
                          ( 1 = Next Ftn set                        ) <
   211A CONSTANT COMPARE  ( 0 = No time comparison-Iono exec[2] ) <
                          ( 1 = Compare RTC with execution time ) <
                          (     of next Function                [2] ) <
                                                                   --> <
                                                                       <
                                                                       <

SCR # 8
 ( Reserved Memory )                                                    <
   HEX                    ( *** Data Capture System ***             ) <
   211C CONSTANT DCS?     ( 0 = Inhibits Printing of DCS Data       ) <
                          ( 1 = Print Data being sent to DCS[2] ) <
   211E CONSTANT DCSTAT   ( 0 = DCS Acknowledging data                <
                          ( 1 = DCS not Acknowledging data    [2] ) <
                                                                       <
                          ( *** Flag in Interrupt Routine ***       ) <
   2120 CONSTANT TON      ( 0 = Time Display off                    ) <
                          ( 1 = Display Time every second     [2] ) <
                                                                       <
                          ( *** Ionogram Busy Flag ***              ) <
   2122 CONSTANT ION      ( 0 = Ionogram not running                ) <
                          ( 1 = Ionogram running                [2] ) <
                                                                   --> <
                                                                       <

SCR # 9
 ( Reserved Memory )                                                    <
   HEX                    ( *** Datatype Handling ***               ) <
   2124 CONSTANT BDATA    ( Beginning of Data Address           [2] ) <
                                                                   --> <
                                                                       <
                                                                       <
                                                                       <
                                                                       <
                                                                       <
                                                                       <
                                                                       <
                                                                       <
                                                                       <
                                                                       <
                                                                       <
                                                                       <
```

```
SCR # 10
  0 ( VDU / Printer Control )                                    HEX <
  1 : STATUSOFF 0 STAT ! ; ( Controller Status not Displayed )       <
  2 : STATUS    1 STAT ! ; ( Controller Status Displayed     )       <
  3 : S@          STAT @ ; ( Fetch Controller Status Flag    )       <
  4 : PRNT  0 VDU? ! ;     ( Carriage Return used - Printer  )       <
  5 : VDU   1 VDU? ! ;     ( Cursor Addressing used - VDU    )       <
  6 : V@      VDU? @ ;     ( Fetch VDU Flag     ) (    --> f    )    <
  7 : NEWCR   VDU? @ IF 0A EMIT 0D EMIT 8 OUT ! ELSE CR ENDIF ;      <
  8 : CR      NEWCR   ;    ( If VDU inhibit ?TERMINAL in CR  )       <
  9 : ?TERM  TFTN @ DUP IF DROP ?TERMINAL ENDIF ; ( f --> f    )     <
 10 : ESC     1B EMIT ;      ( Output Escape      )                  <
 11 : BELL    7  EMIT ;      ( Ring   Bell        )                  <
 12 : CLRVDU V@ IF ( VDU ) ESC 45 EMIT ELSE ( Printer ) CR ENDIF ;   <
 13 : SAVEXY V@ IF ( VDU ) ESC 6A EMIT ENDIF ; ( VDU saves X,Y )     <
 14 : GOXY   V@ IF ( VDU ) ESC 6B EMIT ENDIF ; ( Cursor to X,Y )     <
 15                                                              --> <


SCR # 11
  0 ( VDU / Printer Control )                                    HEX <
  1 : CURPOS V@ IF ESC 6E EMIT KEY DROP KEY DROP                     <
  2          KEY 20 - KEY 20 - ELSE 0 0 ENDIF ;   ( --> ln# col#   ) <
  3 : GTOXY  ESC 59 EMIT SWAP 0 MAX 17 MIN 20 + EMIT                 <
  4          0 MAX 4B MIN 20 + EMIT ;             ( ln#  col# --> ) <
  5 : GOTOXY V@ IF GTOXY ELSE DDROP CR ENDIF ;    ( ln#  col# --> ) <
  6 : XYONLY V@ IF GTOXY ELSE DDROP    ENDIF ;    ( ln#  col# --> ) <
  7 : CLRLI  0 GTOXY ESC 6C EMIT ;                (        ln# --> ) <
  8 : CLRLN  V@ IF CLRLI ELSE DROP CR ENDIF ;     (        ln# --> ) <
  9 : CLRLIS 1+ SWAP DO I CLRLI LOOP ;            ( ln#1 ln#2 --> ) <
 10 : CLRLNS  V@ IF CLRLIS ELSE DROP DROP ENDIF ; ( ln#1 ln#2 --> ) <
 11 : CEOL    V@ IF 57 OUT @ - SPACES ENDIF ;     ( Clr end of ln ) <
 12 : CLRSTAT V@ IF  0  6 CLRLIS  0 0 GTOXY ELSE CR ENDIF ; (      ) <
 13 : CLRDIS  V@ IF  8 12 CLRLIS  8 0 GTOXY ELSE CR ENDIF ; (      ) <
 14 : CLRD+1  V@ IF  9 12 CLRLIS  9 0 GTOXY ELSE CR ENDIF ; (      ) <
 15 : CLRIN   V@ IF 15 17 CLRLIS 15 0 GTOXY ELSE CR ENDIF ;      --> <


SCR # 12
  0 ( VDU / Printer Control )                                DECIMAL <
  1 : HOME S@ IF V@ IF 20 23 CLRLIS 19 0 GTOXY ELSE CR ENDIF ENDIF ;<
  2                                                    (     )       <
  3 : H+1   S@ IF V@ IF 20 CLRLI ELSE CR ENDIF ENDIF ; (     )       <
  4 : DSHLN V@ IF     0 GTOXY                                        <
  5         79 0 DO ." -" LOOP ELSE DROP ENDIF ;  ( ln# --> )        <
  6 : PTRCR V@ 0= IF CR ENDIF ;                   (         )        <
  7 : ->    SPACES ;                              ( n -->   )        <
  8 : TAB   OUT @ 8 - - SPACES ;                  ( n -->   )        <
  9 : 1SEC  10000 0 DO LOOP ;                     (         )        <
 10 : B1S   BELL 1SEC ;                                              <
 11 : OK?   DUP 0= IF B1S ENDIF ;                 ( f --> f )        <
 12 : W?    ." What ?" ;                          (         )        <
 13 : W?B   W? B1S ;                              (         )        <
 14                                                              --> <
 15                                                                  <
 OK
```

```
SCR # 13
  0 ( Number and Word Input )                                     HEX <
  1 : INPT   QUERY 20 WORD 0 HERE C@ 0 DO DROP HERE 1+ I + C@ DUP   <
  2          39 > OVER 30 < + 0= SWAP 2E = OR DUP 0= IF LEAVE ENDIF <
  3          LOOP DUP IF HERE NUMBER ROT ENDIF ;         ( --> d   ) <
  4 : BLNK   V@ IF IN @ SPACES ENDIF ;                                <
  5 : INPUT  SAVEXY BEGIN GOXY BLNK GOXY INPT UNTIL ;   ( --> d   ) <
  6 : INWORD SAVEXY BEGIN GOXY BLNK GOXY QUERY -FIND                  <
  7          IF C1 = IF DROP 0 ELSE 1 ENDIF ELSE 0 ENDIF UNTIL ;     <
  8                                                      ( --> pfa ) <
  9 : EXEC   IF CFA EXECUTE 1 ELSE DROP GOXY 0 ENDIF ;                <
 10                                                  ( pfa f --> f ) <
 11          1 CONSTANT Y       0 CONSTANT N                          <
 12 : YORN   ." (Y or N)? " BEGIN INWORD DUP ' Y = OVER ' N = OR      <
 13          IF CFA EXECUTE 1 ELSE DROP 0 ENDIF UNTIL ; ( --> f   ) <
 14 : KEYCR  KEY DUP EMIT BEGIN KEY D = UNTIL ;          ( --> c   ) <
 15                                                              --> <


SCR # 14
  0 ( MPN Calculator Control )                                    HEX <
  1                                                                   <
  2 : MPN  7F E000 C! 36 E001 C! 0 E002 C! 34 E003 C!                 <
  3        E002 C@ DROP 2F TOMPN 22 TOMPN ;   ( Initialise Calc )    <
  4                                                                   <
  5 : MPNDP  0A TOMPN ;  : MPNEE  0B TOMPN ;  : MPNCS   0C TOMPN ; <
  6 : MPNXEM 1B TOMPN ;  : MPNMS  1C TOMPN ;  : MPNMR   1D TOMPN ; <
  7 : ENTER  21 TOMPN ;  : MPNROL 23 TOMPN ;  : MPNECLR 2B TOMPN ; <
  8 : MPNXEY 30 TOMPN ;  : MPNEX  31 TOMPN ;  : MPNLN   35 TOMPN ; <
  9 : MPN1/X 37 TOMPN ;  : MPNYX  38 TOMPN ;  : MPN+    39 TOMPN ; <
 10 : MPN-   3A TOMPN ;  : MPN*   3B TOMPN ;  : MPN/    3C TOMPN ; <
 11                                                                   <
 12 : ANSWER ANS IF ." MPN Error" MPNECLR ENDIF ;       (         ) <
 13                                                              --> <
 14                                                                   <
 15                                                                   <


SCR # 15
  0 ( Port Initialisation )                                       HEX <
  1 : MPS   3 E004 C! 11 E004 C! ;                      (         ) <
  2 : STORE SWAP E008 C! E00A C! ;                      ( n1 n2 --> ) <
  3 : MPLA1 FF FF STORE 07 E009 C! 2D E00B C! ;         (         ) <
  4 : MPLA2 00 E00C C!  3C E00D C! FF E00E C! 2C E00F C! ; (      ) <
  5 : MPLA3 FF E01C C!  04 E01D C! FF E01E C! 04 E01F C! ; (      ) <
  6 : PORTS MPN MPS MPLA1 MPLA2 MPLA3 ;                    (      ) <
  7                                                                   <
  8 : A1OFF 00 00 STORE 00 E009 C! 00 E00B C! ;            (      ) <
  9 : A2OFF 00 E00C C!  00 E00D C! 00 E00E C! 00 E00F C! ; (      ) <
 10 : A3OFF 00 E01C C!  00 E01D C! 00 E01E C! 00 E01F C! ; (      ) <
 11 : RESET A1OFF A2OFF A3OFF MPS ;                        (      ) <
 12                                                              --> <
 13                                                                   <
 14                                                                   <
 15                                                                   <
 OK
```

```
SCR # 16
  0 ( Synthesizer Control - Input MHZ , KHZ or HZ )             HEX <
  1 : SETDPL3 DPL @ DUP 3 = 0= IF 3 SWAP - 0 DO A * LOOP ELSE DROP  <
  2                               ENDIF ;        ( n --> n*10**3-DPL ) <
  3 : CONSTORE DUP 0< 0=                                            <
  4   IF 4 /MOD 4 0 DO A /MOD LOOP 5 0 DO 30 OR F2MSB I + C! LOOP   <
  5      3E8 * + 4 / A /MOD A /MOD 8 5 DO 30 OR F2MSB I + C! LOOP 1 <
  6      ELSE DDROP CR ." Freq Error" 0 ENDIF ;          ( r q --> f ) <
  7 : MHZ DPL @ 4 < IF SWAP SETDPL3 ELSE DPL @ 6 =                  <
  8        IF 3E8 M/ ELSE DDROP CR ." Format ?" 0 -1 ENDIF          <
  9                  ENDIF CONSTORE ;                ( d -->  f   ) <
 10 : DRQ DPL @ DUP IF 1 SWAP 0 DO A * LOOP M/ SWAP SETDPL3 SWAP    <
 11                  ELSE DROP SWAP ENDIF ;          ( d --> r q ) <
 12 : KHZ DPL @ 3 >                                                 <
 13       IF DDROP 0 -1 ELSE DRQ ENDIF CONSTORE ;    ( d -->  f   ) <
 14 : HZ  DUP 0 < OVER FA0 > + 0= IF 0 CONSTORE                     <
 15       ELSE DROP CR W? 0 ENDIF ;              ( d -->  f  )    --> <


SCR # 17
  0 ( Synthesizer Control - Print MHz/4 or MHz . Freq to MPN )  HEX <
  1 : PRINTBUF  DUP 2 TYPE ." ." 2 + 6 TYPE ."  MHz/4" ;            <
  2 : PRINTBUF1 F1MSB PRINTBUF ;                                    <
  3 : PRINTBUF2 F2MSB PRINTBUF ;                                    <
  4 : TO#       0 SWAP 0 DO OVER I - C@ F AND I                     <
  5             IF I 0 DO A * LOOP + ELSE + ENDIF LOOP SWAP DROP ;  <
  6 : PRINTF    DUP 4 + 4 TO# 4 * SWAP 7 + 3 TO# 4 * 3E8 /MOD ROT + <
  7   0 <# # # # 2E HOLD # # #> TYPE 0 <# # # # #> TYPE ."  MHz" ;  <
  8 : PRINTF1 F1MSB PRINTF ;           : PRINTF2 F2MSB PRINTF ;     <
  9 : DENT 0 DO DUP I + C@ F AND TOMPN LOOP DROP ;                  <
 10 : FTOMPN  8 DENT ;                 : FENTER  FTOMPN ENTER ;     <
 11 : F1TOMPN F1MSB FTOMPN ;           : F2TOMPN F2MSB FTOMPN ;     <
 12 : F1TOMEM F1MSB SWAP 8 CMOVE ;     : F2TOMEM F2MSB SWAP 8 CMOVE ; <
 13 : GETF1 HERE F1TOMEM 8 ALLOT ;     : GETF2 HERE F2TOMEM 8 ALLOT ; <
 14 : TOF1  F1MSB 8 CMOVE ;            : TOF2  F2MSB 8 CMOVE ;    --> <
 15                                                                 <


SCR # 18
  0 ( Synthesizer Control - Freq to Synth.ANSEXP to Memory )    HEX <
  1 : LZERO  1+ DUP 8 SWAP - DUP DUP                                <
  2          IF F1MSB SWAP 30 FILL ELSE DROP ENDIF ;                <
  3 : XFR    F1MSB + ANS# SWAP ROT CMOVE ;                          <
  4 : FORMAT ANSEXP @ F AND DUP 8 <                                 <
  5          IF LZERO XFR ELSE ." Freq over range" DROP ENDIF ;     <
  6 : NEWFREQ  ANSWER FORMAT ;                                      <
  7 : PACK     DUP C@ SWAP 1+ C@ F AND SWAP 10 * OR ;               <
  8 : TOLATCH  8 0 DO DUP I + PACK SWAP 2 +LOOP DROP                <
  9            9 5 DO I STORE LOOP ; ( Sends 8 bytes to Latches   ) <
 10 : LOADFREQ 0 9 STORE ;           ( Load Counters from Latches ) <
 11 : FREQOUT  F1MSB TOLATCH LOADFREQ ; ( F1MSB to counters       ) <
 12 : SETFREQ  F2MSB TOLATCH LOADFREQ ; ( F2MSB to counters       ) <
 13 : ATOMEM   ANSWER ANSEXP SWAP C CMOVE ;                         <
 14 : MTOMPN   DUP 4 + 1 DENT MPNDP DUP 5 + 7 DENT MPNEE 2 DENT ;   <
 15                                                              --> <
 OK
```

```
SCR # 19
 0 ( Hardware Control - Control Register A )                   HEX <
 1 : STROBE0 0 0 STORE ; ( Clears Sweep f/f,A/R f/f Film f/f )     <
 2 : STROBE1 0 1 STORE ; ( Clears Sweep f/f )                      <
 3 : STROBE2 0 2 STORE ; ( Clears A/R   f/f )                      <
 4   A0 CONSTANT OFF              F1 CONSTANT TEST1                <
 5   F3 CONSTANT TEST2            7C CONSTANT DOPPLER              <
 6   9F CONSTANT OBLIQUETX        EF CONSTANT OBLIQUERX            <
 7   FF CONSTANT VERTICAL                                          <
 8 : SETA DUP CONTROLA ! 3 STORE ; ( Example :- VERTICAL SETA )    <
 9 : A@   CONTROLA @ ;                                             <
10 : A?                                                            <
11      CASE A0 OF ." OFF"        ENDOF F1 OF ." TEST1"      ENDOF <
12           F3 OF ." TEST2"      ENDOF 7C OF ." DOPPLER"    ENDOF <
13           9F OF ." OBLIQUE TX" ENDOF EF OF ." OBLIQUE RX" ENDOF <
14           FF OF ." VERTICAL"   ENDOF                            <
15                 ." Code " . 0  ENDCASE ;                    --> <


SCR # 20
 0 ( Hardware Control - Control Register B )                   HEX <
 1 : SETB    DUP CONTROLB C! 4 STORE ;                             <
 2 : CLEARB  0  SETB ;            : B@      CONTROLB C@ ;          <
 3 : ORSETB  OR SETB ;            : ANDSETB AND SETB ;             <
 4 : FILMON  B@ 01 ORSETB  ;      : FILMOFF B@  FE ANDSETB ;       <
 5 : AGCEXT  B@ 02 ORSETB  ;      : AGCINT  B@  FD ANDSETB ;       <
 6 : TXON    B@ 04 ORSETB  ;      : TXOFF   B@  FB ANDSETB ;       <
 7 : B3ON    B@ 08 ORSETB  ;      : B3OFF   B@  F7 ANDSETB ;       <
 8 : B4ON    B@ 10 ORSETB  ;      : B4OFF   B@  EF ANDSETB ;       <
 9 : B5ON    B@ 20 ORSETB  ;      : B5OFF   B@  DF ANDSETB ;       <
10 : B6ON    B@ 40 ORSETB  ;      : B6OFF   B@  BF ANDSETB ;       <
11 : B7ON    B@ 80 ORSETB  ;      : B7OFF   B@  7F ANDSETB ;       <
12 : BLIST   CR ."  7    6    5    4    3    TX  AGC  FILM"  CR  ; <
13 : B?      BLIST B@ 9 1 DO DUP 80 AND IF ."  1" ELSE ."  0" ENDIF<
14           3 SPACES 2 * LOOP DROP ;                              <
15                                                             --> <


SCR # 21
 0 ( Hardware Control - Basic Rate and Doppler )           DECIMAL <
 1 : TABLE <BUILDS 0 DO , LOOP DOES> SWAP 2 * + @ ; ( Build Table) <
 2   0 4000 2000 1000 800 500  0 250                               <
 3   0  400  200  100  80  50 40  25   16 TABLE RATE ( Basic rates) <
 4 : RATE?   RATE . ." kHz/sec  " ;                                <
 5 : KHZ/SEC DUP 0= IF DROP 1 ENDIF                                <
 6           0 16 0 DO DROP DUP I RATE = IF LEAVE ENDIF I LOOP     <
 7           SWAP DROP DUP 15 = IF DROP W? 0 ELSE 1 ENDIF ;        <
 8 : SETRATE IF 10 STORE ENDIF ; ( Example :- 50 KHZ/SEC SETRATE ) <
 9 : DOPR    DUP                                                   <
10           IF 1+ 2 SWAP 0 DO 2 * LOOP 4 / ENDIF ; ( Code to Hz ) <
11 : DOP     0 8 0 DO DROP DUP I DOPR = IF LEAVE ENDIF I LOOP      <
12           SWAP DROP DUP 6 > IF DROP W? 0 ELSE 1 ENDIF ;         <
13 : DOP.    DOPR . ;                                              <
14 : DOP?    DOP. ." Hz " ;                                        <
15 : SETDOP  IF 20 STORE 1 21 STORE ENDIF ; ( Eg 4 DOP SETDOP) --> <
```

```
SCR # 22
  0 ( Hardware Control - Tcell and M Sequence )                 HEX <
  1 00 CONSTANT 1/32     01 CONSTANT 1/16     02 CONSTANT 1/8       <
  2 03 CONSTANT 1/4      04 CONSTANT 1/2      05 CONSTANT 1/        <
  3 06 CONSTANT 2/       07 CONSTANT 4/       08 CONSTANT 8/        <
  4 09 CONSTANT 16/      0A CONSTANT 32/      0B CONSTANT 64/       <
  5 : TCELL   DUP 0 < OVER 7 > + IF DROP ." Invalid Tcell " ENDIF ; <
  6 : CODE#   5 - DUP ABS 1+ 1 SWAP 0 DO 2 * LOOP 2 / SWAP 0< ;     <
  7 : CELL?   CODE# IF ." 1/" . ELSE . 8 EMIT ." / " ENDIF ;        <
  8 : TCELL?  CELL? ." sec " ;                                      <
  9 : SETCELL B STORE ;             ( Example :- 1/ TCELL SETCELL ) <
 10           71 61 52 42 33 21 11 D 8 4 1 B TABLE MSEQNCE          <
 11 : MSEQ    DUP 0 < OVER B > + 0=                                 <
 12           IF MSEQNCE ELSE DROP ." Invalid Mseq" ENDIF ;         <
 13 : MSEQ?   0 B 0 DO DROP DUP I MSEQ =                            <
 14           IF LEAVE ENDIF I LOOP SWAP DROP TCELL? ;              <
 15 : SETMSEQ C STORE 0 D STORE ;    ( Eg. :- 1/ MSEQ SETMSEQ ) --> <


SCR # 23
  0 ( Hardware Control - FFT Sample Rate and Antenna Switch )   HEX <
  1 : FFTR   1+ 4 SWAP 0 DO 2 * LOOP ;                              <
  2 : FFT    0 9 0 DO DROP DUP I FFTR = IF LEAVE ENDIF I LOOP       <
  3          SWAP DROP DUP 8 = IF DROP W? 0 ELSE 1 ENDIF ;          <
  4 : FFT.   FFTR . ;                                               <
  5 : FFT?   FFT.  ." Hz " ;                                        <
  6 : SETFFT IF 0E STORE ENDIF ;     ( Example :- 1024 FFT SETFFT ) <
  7 : CHKA   DUP 1 < SWAP 8 > + 0= ;                                <
  8 : ANTENNA OVER CHKA OVER CHKA AND                               <
  9           IF 10 * OR 77 + 1 ELSE DDROP W? 0 ENDIF ;             <
 10 : SETANTENNA IF F STORE 0 10 STORE ENDIF ;                      <
 11 ( Example :- 1 2 ANTENNA SETANTENNA )                           <
 12 : ANTS?      DUP F0 AND 10 / 7 - SWAP F AND 7 - ;               <
 13 : ANTENNA?   ANTS? ." Rx1 Antenna = " . CR                      <
 14                    ." Rx2 Antenna = " . ;                       <
 15                                                              --> <


SCR # 24
  0 ( Hardware Control - Film Speed , Windowing and Sync's )    HEX <
  1 : FILMSPEED? 258  OVER 30 AND 10 / 1+ 0 DO 2 / LOOP             <
  2              SWAP F AND DUP IF / ELSE DROP 1 / ENDIF ;          <
  3 : MM/MIN     CR 1+ DUP 3 < OVER 12C > + 0=                      <
  4              IF 40 0 DO DUP I FILMSPEED? DUP ROT <              <
  5               IF ." Closest Speed = " . ." mm/min " 1SEC        <
  6               DROP I LEAVE 1 ELSE DROP ENDIF LOOP               <
  7              ELSE W? DROP 0 ENDIF ;                             <
  8 : SPEED?   FILMSPEED? . ." mm/min " ;                           <
  9 : SETSPEED IF 11 STORE ENDIF ;      ( Eg :- 5 MM/MIN SETSPEED ) <
 10 : KMSTEP DUP 3B7 < IF M* 96 M/ SWAP DROP 100 / A /MOD 10 * OR 1 <
 11          ELSE W? DDROP 0 ENDIF ; ( Rate,min Km to 256 Hz steps) <
 12 : SETWIND IF 8 14 STORE 15 STORE ENDIF ; ( LO Offset-Windowing) <
 13                                                                 <
 14 : RXPULSE           0 12 STORE ;     : RXSYNC  0 13 STORE ;     <
 15 : SWEEP      RXSYNC 0 16 STORE ;     : DOPTIME 0 1B STORE ; --> <
 OK
```

```
SCR # 25
  0 ( Hardware Control - Advance Retard Control )                  HEX <
  1 : MSEC     DRQ ;                         ( d --> r q )              <
  2 : +COUNT   SHIFT D@ D+ SHIFT D! ;        (           )              <
  3 : SLIP     DUP 19 STORE 0 1A STORE ; ( n --> n    )                 <
  4 : WAITCOUNT SWAP DUP 0 > IF DUP 0 DO LOOP SWAP 0 +COUNT              <
  5              ELSE DUP ABS 0 DO LOOP SWAP 0 DMINUS +COUNT ENDIF ;    <
  6 : POSN       SHIFT D@ DDUP DABS <# # # 2E HOLD # # # # #> TYPE      <
  7   ." ms " 0. D< IF ." Retarded " ELSE ." Advanced " ENDIF ;         <
  8 : ENPOS    S@ IF V@ IF 6 2C GOTOXY ENDIF POSN ENDIF ;               <
  9 : CLRPOSN 0. SHIFT D! ENPOS HOME ;                                  <
 10 : MSPOS    V@ IF ENPOS ENDIF ;                                      <
 11 : INPOS    S@ IF 6 14 GOTOXY POSN ." ---> " V@ IF ENPOS ENDIF       <
 12               ENDIF BELL ;                                          <
 13 : DOSLIP  SRCH @ 0= IF INPOS ENDIF SWAP DUP IF 0 DO 64 SLIP         <
 14   WAITCOUNT ?TERM IF LEAVE ENDIF MSPOS LOOP ELSE DROP ENDIF         <
 15   SWAP A / SLIP WAITCOUNT DROP ENPOS ;                          --> <


SCR # 26
  0 ( Hardware Control - Advance Retard Control )                  HEX <
  1 : SETA/R SRCH @ 0= IF SHIFT D@ LPOSN D! ENDIF                       <
  2           17 STORE DOSLIP HOME ;                                    <
  3 : A@1    S@ V@ AND IF  3800 ELSE  3A00 ENDIF 2 SETA/R ;             <
  4 : A@10   S@ V@ AND IF  330  ELSE  600  ENDIF 4 SETA/R ;             <
  5 : A@20   S@ V@ AND IF  1    ELSE  100  ENDIF 8 SETA/R ;             <
  6 : R@1    S@ V@ AND IF -3800 ELSE -3A00 ENDIF 3 SETA/R ;             <
  7 : R@10   S@ V@ AND IF -330  ELSE -600  ENDIF 5 SETA/R ;             <
  8 : R@20   S@ V@ AND IF -1    ELSE -100  ENDIF 9 SETA/R ;             <
  9 : EXIT   ;                                                          <
 10 : LAST   SHIFT D@ LPOSN D@ D- DDUP DABS 2 DPL ! MSEC                <
 11          DSWAP 0. D> IF R@20 ELSE A@20 ENDIF ;                  --> <
 12                                                                     <
 13                                                                     <
 14                                                                     <
 15


SCR # 27
  0 ( Hardware Control - Advance Retard Search )               DECIMAL <
  1 : VAL    CLRIN ." Valid Data :- " ;                                 <
  2 : NPR    NFA ID. ;                                                  <
  3 : DRP    VAL ' A@1 NPR ' A@10 NPR ' A@20 NPR                        <
  4              ' R@1 NPR ' R@10 NPR ' R@20 NPR     ;                  <
  5 : SDRP   DRP ' EXIT NPR  H+1 ." >" ;                                <
  6 : SEARCH INPOS SHIFT D@ LPOSN D! 1 SRCH ! SDRP                      <
  7          BEGIN INWORD DUP ' EXIT = 0= WHILE                         <
  8          CLRIN ."Press SPACE BAR to stop"                           <
  9          CASE ' A@1   OF   0 10000 A@1  ENDOF                       <
 10               ' A@10  OF   0 10000 A@10 ENDOF                       <
 11               ' A@20  OF   0 10000 A@20 ENDOF                       <
 12               ' R@1   OF   0 10000 R@1  ENDOF                       <
 13               ' R@10  OF   0 10000 R@10 ENDOF                       <
 14               ' R@20  OF   0 10000 R@20 ENDOF SPACE W?B ENDCASE     <
 15 V@ IF SDRP ELSE ." >" ENDIF REPEAT DROP 0 SRCH ! HOME ;         --> <
 OK
```

```
SCR # 28
 ( Datatype Handling )                                        DECIMAL  <
                                                                       <
 : NAME? IN @ -FIND IF 193 =                                           <
        IF DROP W? ELSE NPR ."  Exists !" ENDIF DROP IN @ 0 BIS        <
                      ELSE 1 ENDIF SWAP IN ! ;                         <
 : TYPE?   DUP 2+ @ 256 /    ; ( Upper 8 bits of n = Datatype     )    <
 : SUBTYPE DUP 2+ @ 256 MOD ; ( Lower 8 bits of n = Subtype       )    <
 : GETNAME -FIND IF 193 = IF DROP W?B 0 ELSE 1 ENDIF                   <
                ELSE W?B 0 ENDIF ;                                     <
 : RPFA    OVER + ;               ( pfa n      --> pfa  pfa+n     )    <
 : RPFA@   RPFA @ ;               ( pfa n1     --> pfa  n2        )    <
 : RPFA!   >R OVER R> + ! ;       ( pfa n1 n2 --> pfa             )    <
 : GENCR     OUT @ COLUMNS @ 10 - > IF CR 0 OUT ! 7 -> ENDIF ;         <
 : LID.      16 OVER C@ 31 AND - SWAP ID. SPACES ; ( nfa -->      )    <
 : LNPR      NFA LID. ;                            ( pfa -->      )    <
                                                                  -->  <
                                                                       <


SCR # 29
 ( Sounding Parameters )                                      DECIMAL  <
 : BR=      ." Basic Rate   = " ;                         ( Basic Rate ) <
 : RP       VAL 16 0 DO I RATE DUP IF . ELSE DROP ENDIF LOOP           <
            ." kHz/sec" CR BR= ;                                       <
 : HR=      ." Height Range= " ;                          ( Windowing  ) <
 : HZKM     150 M* ROT M/ SWAP DROP ;                                  <
 : HMAX     75000. ROT M/ SWAP DROP ;                                  <
 : RANGE    OVER SWAP HR=                                              <
            HZKM DUP . SWAP ." to " HMAX + . ." Km " ;                 <
 : GETRATE BEGIN RP INPUT DROP KHZ/SEC OK? UNTIL ; ( Get Rate   )      <
 : NMH      ." New Minimum " HR= ;                                     <
 : HTP      VAL ." 0 to 950 Km" CR DUP 0 RANGE CR NMH ;                <
 : GETWIND BEGIN DUP HTP INPUT DROP KMSTEP OK? UNTIL 23 CLRLN 22       <
   CLRLN    PTRCR ." New " DUP DUP 16 / 6 * - 256 *                    <
            ROT OVER RANGE 1SEC SWAP 8 ;                  ( Windowing ) <
                                                                  -->  <


SCR # 30
 ( Sounding Parameters )                                      DECIMAL  <
 : DOP=      ." Doppler Freq= " ;                 ( Doppler Freq    )  <
 : DOPP      VAL 7 0 DO I DOP. LOOP ." Hz" CR DOP= ;                   <
 : GETDOP    BEGIN DOPP INPUT DROP DOP OK? UNTIL DUP DOPR 1 ROT ;      <
 : P         VAL DO I CELL? LOOP ." Sec" CR ; ( Tcell and Mseq  )      <
 : CKT       DUP ' 1/32 < OVER ' 4/  > + 0= ; ( Check Iono Tcell)      <
 : CKD       DUP ' 1/2  < OVER ' 64/ > + 0= ; ( Check Dopp Tcell)      <
 : CKM       DUP ' 1/32 < OVER ' 32/ > + 0= ; ( Check Mseq       )     <
 : TC=       ." Tcell        = " ;                                     <
 : GETTCELL 8  0 P TC= BEGIN INWORD CKT EXEC UNTIL ;       ( Iono)     <
 : GETDCELL 12 4 P TC= BEGIN INWORD CKD EXEC UNTIL 4 - ; ( Dopp)       <
 : DOP??     CLRIN  ." Stationary Doppler " YORN 0= ;                  <
 : MS=       ." Mseq         = " ;                    ( M Sequence)    <
 : GETMSEQ  11 0 P MS= BEGIN INWORD CKM EXEC UNTIL MSEQ ;              <
                                                                  -->  <
                                                                       <

 OK
```

```
SCR # 31
  0 ( Sounding Parameters )                                DECIMAL <
  1 : FT=       ." FFT Rate    = " ;                 ( FFT Rate   ) <
  2 : FFTP      VAL 8 0 DO I FFT. LOOP ." Hz" CR FT= ;               <
  3 : GETFFT    BEGIN FFTP INPUT DROP FFT OK? UNTIL ;                <
  4 : FS=       ." Film Speed  = " ;                 ( Film Speed ) <
  5 : FLMP      VAL ." Between 2 and 150 mm/min" CR FS= ;            <
  6 : GETSPEED BEGIN FLMP INPUT DROP MM/MIN OK? UNTIL ;              <
  7 : AG=       ." Rx AGC      = " ;                 ( Rx AGC     ) <
  8 : RXG       CLRIN ." Fixed Receiver Gain " YORN 0= ;             <
  9 : G  IF CR ." Multiples of Tcell = " INPUT DROP ELSE 0 ENDIF ;  <
 10 : TM=       ." Time Marks  = " ;                 ( Time Marks ) <
 11 : TMKS    CLRIN  TM= YORN G ;                                    <
 12                                                              --> <
 13                                                                  <
 14                                                                  <
 15                                                                  <


SCR # 32
  0 ( Sounding Parameters )                                DECIMAL <
  1 : #CELLS CLRIN ." Number of Cells = " INPUT DROP -DUP 0=         <
  2          IF 1 ENDIF ;                            ( No of Cells ) <
  3 : OFPRM  VAL ." xxxx Hz, xxx.xxx kHz or xx.xxx MHz" CR           <
  4          ." Offset from Origin ( Number only ) = " ;  ( Freq   ) <
  5 : CHKWD  DUP ' MHZ < OVER ' HZ > + 0= ;           ( Check units ) <
  6 : GETOFF BEGIN OFPRM   INPUT                                     <
  7          ."  Units = " INWORD DUP ' HZ = IF SWAP DROP ENDIF      <
  8          CHKWD EXEC 0= IF DDROP CR W? 0 ENDIF OK? UNTIL ;        <
  9 : FILMDR CLRIN ." Film Drive " YORN ;             ( Film Drive  ) <
 10 : ANTP   VAL 9 1 DO I . LOOP CR ;                 ( Antennas    ) <
 11 : GETANT BEGIN ANTP                                              <
 12          ." Rx1 Antenna = " INPUT DROP CR                        <
 13          ." Rx2 Antenna = " INPUT DROP ANTENNA OK? UNTIL ;       <
 14                                                              --> <
 15                                                                  <


SCR # 33
  0 ( Build Sounding )                                     DECIMAL <
  1 : TYPE1    256 + ;                     ( Datatype 1 - Sounding ) <
  2 : SOUNDING NAME?                                                 <
  3   IF <BUILDS 1SEC DOP?? DUP DUP       TYPE1 ,                    <
  4      IF   GETRATE DUP , RATE GETWIND , , , GETTCELL ,            <
  5      ELSE      15     ,       GETDOP   , , , GETDCELL , ENDIF    <
  6           GETMSEQ     ,       GETFFT   ,      GETSPEED ,         <
  7      IF     RXG ,          0 , ELSE    RXG G ,     TMKS , ENDIF  <
  8      #CELLS DUP , 0 DO 20 25 GOTOXY ." Cell No = " I 1+ .        <
  9            GETOFF GETF2 FILMDR , GETANT , LOOP DOES> DROP        <
 10   ENDIF HOME ;                                ( Build Sounding ) <
 11                                                                  <
 12 : SUB1.       ." - SOUNDING ( for "                              <
 13          IF   ." IONOGRAM )"                                     <
 14          ELSE ." STATIONARY DOPPLER )" ENDIF ; ( Print Subtype ) <
 15                                                              --> <
 OK
```

```
SCR # 34
  0 ( Display Sounding )                                   DECIMAL <
  1 : BR.   SUBTYPE                                                <
  2         IF 9 0 GOTOXY BR= 4 RPFA@ RATE? ENDIF ;                <
  3                                            ( Basic Rate Print) <
  4 : LO=   ." L.O. Offset = " ;                                    <
  5 : LOHR. 9 28 GOTOXY 10 RPFA@ LO= . ." Hz  " ( LO. Offset Print) <
  6         SUBTYPE                                                <
  7         IF 9 54 GOTOXY 10 RPFA@ SWAP                           <
  8            4 RPFA@ RATE ROT RANGE ENDIF ;  ( Range      Print ) <
  9 : TC.   10 0 GOTOXY TC=                                        <
 10         SUBTYPE                                                <
 11         SWAP 12 RPFA@ ROT 0=                                   <
 12         IF 4 + ENDIF TCELL? ;              ( Tcell      Print ) <
 13 : MS.   10 28 GOTOXY MS= 14 RPFA@ MSEQ? ;  ( M Sequence Print ) <
 14 : FT.   10 54 GOTOXY FT= 16 RPFA@ FFT? ;   ( FFT Rate   Print ) <
 15                                                            --> <


SCR # 35
  0 ( Display Sounding )                                   DECIMAL <
  1 : FS. 11  0 GOTOXY FS= 18 RPFA@ SPEED? ;   ( Film Speed Print ) <
  2 : AG. 11 28 GOTOXY AG= 20 RPFA@ -DUP                           <
  3              IF   ." Clk/"        SWAP SUBTYPE ROT SWAP        <
  4              IF   ." Snd  "       DROP ELSE  . ." Cells" ENDIF <
  5              ELSE ." Fixed       "     ENDIF ;                 <
  6 : TM. 11 54 GOTOXY TM= 22 RPFA@ -DUP                           <
  7              IF   ." /"   . ." Cells"  ELSE ." Off       " ENDIF ; <
  8 : NC. 12 28 GOTOXY ." No of Cells = "  24 RPFA@ . CR PTRCR ;   <
  9                                                            --> <
 10                                                                <
 11                                                                <
 12                                                                <
 13                                                                <
 14                                                                <
 15                                                                <


SCR # 36
  0 ( Display Sounding - Cell Data )                       DECIMAL <
  1 : CNUM 4C# @ 1 - 4 * + ;                       (    n1 --> n2 ) <
  2 : VECC CNUM 1 - 12 * + ;                       ( n1 n2 --> n3 ) <
  3 : PCEL 14 18 CLRLNS 13 0 GOTOXY                                <
  4               5 1 DO CR ." Cell "                              <
  5        24        RPFA@ I CNUM DUP . = IF LEAVE ENDIF 10 TAB    <
  6        26 I VECC RPFA  PRINTF                        29 TAB    <
  7        34 I VECC RPFA@ IF ." Y" ELSE ." N" ENDIF     46 TAB    <
  8        36 I VECC RPFA@ ANTS? .                       63 TAB .  <
  9        LOOP ;                                                  <
 10 : CELLDATA 13 TAB ." Offset"       25 TAB ." Film Drive"       <
 11            41 TAB ." Rx1 Antenna"  58 TAB ." Rx2 Antenna"      <
 12            24 RPFA@ 4 /MOD SWAP IF 1+ ENDIF #4C !              <
 13            V@ IF PCEL ELSE                                     <
 14            BEGIN PCEL 1 4C# +! 4C# @ #4C @ >                   <
 15            UNTIL 1 4C# ! ENDIF HOME ;                      --> <
 OK
```

```
SCR # 37
  0 ( Display Sounding )                                      DECIMAL <
  1 : DISS   TYPE? 1 =                                                <
  2          IF DUP CLRDIS 23 -> NPR SUBTYPE SUB1. CR                 <
  3          BR. LOHR.     TC. MS. FT.      FS. AG. TM.    PTRCR      <
  4          NC. 1 4C# ! CELLDATA                                     <
  5          ELSE W?B ENDIF ;   ( Display Sounding ) ( pfa --> pfa ) <
  6 : SDISP DISS DROP ;         ( Display Sounding ) ( pfa -->     ) <
  7 : SDISPLAY GETNAME IF SDISP ENDIF ;            ( Print Sounding ) <
  8 : SHOWS  CLRD+1 30 -> ." Defined Soundings" CR CR 7 SPACES        <
  9          CONTEXT @ @ BEGIN GENCR PFA TYPE? 1 =                    <
 10          IF SUBTYPE 2 < IF DUP LNPR ENDIF ENDIF                   <
 11          LFA @ DUP BDATA @ < UNTIL DROP ;      ( Show Soundings)<
 12                                                                --> <
 13                                                                    <
 14                                                                    <
 15                                                                    <


SCR # 38
  0 ( Edit Sounding - Prompts )                               DECIMAL <
  1 : BR     1 ;          : LO    2 ;           : TC    3 ;           <
  2 : MSQ    4 ;          : FT    5 ;           : FS    6 ;           <
  3 : AGC    7 ;          : TM    8 ;           : CELL  9 ;           <
  4 : OF=    ." Offset/Range= " ;                                     <
  5 : CL=    ." Cells        = " ;                                    <
  6 : ESP    VAL                                                      <
  7          BR= ' BR  NPR 5 -> OF= ' LO  NPR 5 -> CL= ' CELL NPR CR <
  8   14 -> TC= ' TC  NPR 5 -> MS= ' MSQ NPR 4 -> FT= ' FT   NPR CR <
  9   14 -> FS= ' FS  NPR 5 -> AG= ' AGC NPR 4 -> TM= ' TM   NPR    <
 10   20  2 GOTOXY ;                         ( Edit Sounding Prompt ) <
 11 : PARA  ESP ."  - Change >>"                                      <
 12          BEGIN INWORD DUP ' BR < OVER ' CELL > + 0= EXEC UNTIL ; <
 13                                          ( Parameter Request    ) <
 14                                                                --> <
 15                                                                    <


SCR # 39
  0 ( Edit Sounding Parameter )                               DECIMAL <
  1 : ELO    SUBTYPE IF 4 RPFA@ RATE GETWIND ELSE GETDOP ENDIF        <
  2                       >R >R 10 RPFA! R> 8 RPFA! R> 6 RPFA! LOHR. ;  <
  3 : EBR    SUBTYPE IF GETRATE 4 RPFA! BR. ELO ELSE W?B ENDIF ;      <
  4 : ETC    SUBTYPE IF GETTCELL ELSE GETDCELL ENDIF 12 RPFA! TC. ;   <
  5 : EMQ                 GETMSEQ  14 RPFA! MS. ;                     <
  6 : EFT                 GETFFT   16 RPFA! FT. ;                     <
  7 : EFS                 GETSPEED 18 RPFA! FS. ;                     <
  8 : EAG    SUBTYPE IF RXG ELSE RXG G ENDIF 20 RPFA! AG. ;           <
  9 : ETM    SUBTYPE IF W?B ELSE TMKS 22 RPFA! ENDIF  TM. ;           <
 10                                                                --> <
 11                                                                    <
 12                                                                    <
 13                                                                    <
 14                                                                    <
 15                                                                    <
 OK
```

```
SCR # 40
  0 ( Edit Sounding Parameter )                           DECIMAL <
  1 : CP    VAL 24 RPFA @ 1+ 1 DO I . LOOP CR ;                   <
  2 : GC#   CP ." Cell Number = " BEGIN 24 RPFA @ INPUT DROP DUP  <
  3         ROT > OVER 1 < + 0= IF 1 ELSE DROP W?B 0 ENDIF UNTIL ; <
  4 : EC#   GC#  1 - 12 * 26 + OVER + DUP GETOFF                  <
  5         F2TOMEM FILMDR OVER 8 + ! GETANT SWAP 10 + ! CELLDATA ; <
  6 : PVEC  VECTOR EBR ELO ETC EMQ EFT EFS EAG ETM EC# ;          <
  7 : EDITS PARA PVEC ;                                       --> <
  8                                                               <
  9                                                               <
 10                                                               <
 11                                                               <
 12                                                               <
 13                                                               <
 14                                                               <
 15                                                               <


SCR # 41
  0 ( Display and Edit Sounding )                         DECIMAL <
  1 : PDES VAL                                                    <
  2  ." N =   Next   4 Cells      C = Change      E = Exit" CR   <
  3  14 -> ." P = Previous 4 Cells" H+1 ." >" ;                   <
  4 : DES DISS                                                    <
  5      BEGIN PDES KEYCR DUP 69 = 0= WHILE                       <
  6      CASE  80 ( P ) OF 4C# @ 1 -   1   MAX 4C# ! PCEL ENDOF   <
  7             78 ( N ) OF 4C# @ 1+  #4C @ MIN 4C# ! PCEL ENDOF   <
  8             67 ( C ) OF EDITS                         ENDOF   <
  9             SPACE W?B ENDCASE REPEAT DROP DROP HOME ;         <
 10 : SDISP    V@ IF DES ELSE SDISP ENDIF ;          ( pfa -->   ) <
 11 : SEDIT    GETNAME IF DES ENDIF ;                (     -->   ) <
 12 : SDISPLAY GETNAME IF SDISP ENDIF ;              (     -->   ) <
 13 : SOUNDING DECIMAL HERE [COMPILE] SOUNDING HERE OVER = IF DROP <
 14          ELSE PFA SDISP ENDIF ; ( Program , Display Sounding) <
 15                                                           --> <


SCR # 42
  0 ( Set Sounding )                                      DECIMAL <
  1 : IFR         ." Invalid Forward Ref. - " ;                   <
  2 : SSET        TYPE? 1 =                         ( pfa -->  f  ) <
  3        IF   4 RPFA@ 1 SETRATE SUBTYPE                         <
  4         IF  8 RPFA@ 1 SETWIND ELSE 6 RPFA@ 1 SETDOP ENDIF     <
  5            12 RPFA@   SETCELL                                 <
  6            14 RPFA@   SETMSEQ                                 <
  7            16 RPFA@ 1 SETFFT                                  <
  8            18 RPFA@ 1 SETSPEED                                <
  9                       CLEARB                                  <
 10            34 RPFA@   IF FILMON  ENDIF TXON                   <
 11            36 +    @ 1 SETANTENNA                             <
 12                      1 DUP DUP AGC# ! TMK# ! CELL# ! 1        <
 13                        ELSE    DROP IFR ." CHANGE "  0 ENDIF ; <
 14 : USSET   GETNAME IF SSET DROP ENDIF ; ( User Set Sounding) --> <
 15                                                               <
 OK
```

```
SCR # 43
( Ionogram Parameters )                                    DECIMAL <
: TYPEP  VAL ' OFF       NPR  ' TEST1     NPR  ' TEST2     NPR   <
             ' DOPPLER   NPR  ' OBLIQUETX NPR  ' OBLIQUERX NPR   <
             ' VERTICAL  NPR    CR                               <
            ." Experimental Code "  YORN CR ." Ionogram " ;      <
: GETYPE TYPEP IF ." Decimal Code = " INPUT DROP ELSE            <
         ." = " BEGIN INWORD DUP ' OFF < OVER ' VERTICAL > + 0= <
         EXEC UNTIL ENDIF ;                          ( Type    ) <
: ID=    ." Ionogram Duration = " ;                              <
: MORS   0 BEGIN DROP INPUT DROP DUP 60 < UNTIL 3 -> ;           <
: GETDUR VAL ." Hours , Minutes , Seconds"  CR                   <
             ." Doppler " ID= 8 EMIT ." >"  CR                   <
             ." Hours = "   INPUT DROP                           <
             ." Minutes = " MORS                                 <
             ." Seconds = " MORS SWAP ROT ;          ( Duration ) <
                                                             --> <


SCR # 44
( Ionogram Parameters )                                    DECIMAL <
: F=    ." Frequency   = " ;                                     <
: SF=   ." Start " F= ;                                          <
: EF=   ." End   " F= ;                                          <
: FP    VAL ." Between 0.5 and 30.0 MHz" CR ;                    <
: INF   BEGIN INPUT MHZ OK? UNTIL ;                              <
: GETF  F=     INF ;                    ( Get Frequency       )  <
: GETSF FP SF= INF ;                    ( Get Start Frequency )  <
: GETEF FP EF= INF ;                    ( Get End   Frequency )  <
: ORA   ." Overall Rate " ;             ( Get Overall Rate    )  <
: OR=   ORA ."       = " ;                                       <
: LIN   ." Linear " ;                                            <
: LN    ." Log    " ;                                            <
: ORP   VAL LIN ." = xxx KHz/sec , Log = .001->.01 oct/sec" CR   <
        LIN ORA YORN DUP CR IF LIN OR= ELSE LN OR= ENDIF ;       <
: GETOR ORP DUP INPUT DROP SWAP 0= IF SETDPL3 ENDIF SWAP ;   --> <


SCR # 45
( Ionogram Parameters )                                    DECIMAL <
: SN=     ." Sounding          = " ;                             <
: VALA    VAL ." As Listed Above" CR ;                           <
: GETSND  CLRDIS SHOWS VALA                                      <
          SN= INWORD SUBTYPE ;         ( Get Sounding        )   <
: CHKSND  IF DROP ." Incompatible Sounding !" 0 B1S              <
          ELSE 1 ENDIF ;               ( Check Sounding      )   <
: GETISND BEGIN GETSND 0= CHKSND UNTIL ; ( Ionogram Sounding )   <
: GETDSND BEGIN GETSND    CHKSND UNTIL ; ( Doppler Sounding  )   <
: TT=     ." Data to Tape      = " ;                             <
: TOMT?   CLRIN ." Store Data on Mag Tape " YORN ;               <
: STATI?  CLRIN ." Stationary Ionogram " YORN ;                  <
                                                             --> <
                                                                 <
                                                                 <
                                                                 <
```

```
SCR # 46
( Build Ionogram )                                          DECIMAL <
: TYPE2  512 + ;                          ( Datatype 2 - Ionogram ) <
: IONOGRAM NAME?                                                    <
  IF <BUILDS 1SEC GETYPE DUP     TYPE2  , 124 =                     <
     IF GETDUR , , , FP GETF GETF2 GETF2 1 , 0 , GETDSND ,          <
     ELSE STATI?                                                    <
          IF GETDUR , , , FP GETF GETF2     GETF2    1 , 0 ,        <
          ELSE 0 , 0 , 0 , GETSF GETF2 GETEF GETF2 GETOR  , ,       <
          ENDIF GETISND ,                                           <
     ENDIF TOMT? , DOES> DROP                                       <
  ENDIF HOME ;                                   ( Build Ionogram ) <
                                                                    <
: SUB2.  ." - IONOGRAM ( " A? ."  )" ;           ( Print Subtype  ) <
                                                                    <
: .C 2 .R ."  : " ;                              ( Print xx :     ) <
                                                                --> <

```

```
SCR # 47
( Display Ionogram )                                         DECIMAL <
: IDR. 10 0 GOTOXY ID=        4 RPFA@ .C 6 RPFA@ .C 8 RPFA@ . ; <
                                                 ( Duration Print ) <
: TT.  11 0 GOTOXY TT=       32  RPFA@                               <
                      IF ." Yes" ELSE ." No " ENDIF ; ( To Tape ?) <
: SF.  13 0 GOTOXY SF=       10  RPFA   PRINTF ; ( Start Freq ) <
: EF.  14 0 GOTOXY EF=       18  RPFA   PRINTF ; ( End   Freq ) <
: OR.  15 0 GOTOXY OR=       26  RPFA@                               <
                      IF     28  RPFA@ . ." kHz/sec  "               <
                      ELSE   28  RPFA@ 10 /MOD 10 /MOD               <
                      ." ." 1 .R 1 .R 1 .R ."  oct/sec " ENDIF ;     <
: SN.  17 0 GOTOXY SN=       30  RPFA@ TYPE? 1 =                     <
                        IF NPR ELSE DROP IFR ." CHANGE " ENDIF ;     <
: VEL. SUBTYPE 124 = IF 18 0 GOTOXY ." Maximum Velocity  = "         <
       14 RPFA 4 TOM >R 30 RPFA@ 10 + @ 100 * 375 M*                 <
                        R> M/ SWAP DROP . ." m/s" ENDIF ;       --> <
```

```
SCR # 48
( Display Ionogram )                                         DECIMAL <
: DISI DUP CLRDIS 25 -> NPR SUBTYPE SUB2. PTRCR                      <
       IDR.   TT.       PTRCR                                        <
        SF.   EF. OR.   PTRCR                                        <
        SN.  VEL. HOME PTRCR ; ( Display Iono ) ( pfa --> pfa ) <
: IDISP DISI DROP ;          ( Display Ionogram ) ( pfa -->      ) <
: IDISPLAY GETNAME IF IDISP ENDIF ;            ( Display Ionogram ) <
: SHOWI CLRD+1 30 -> ." Defined Ionograms" CR CR 7 SPACES            <
        CONTEXT @ @                                                  <
        BEGIN GENCR PFA TYPE? 2 = IF DUP LNPR ENDIF                  <
        LFA @ DUP BDATA @ < UNTIL DROP ;    ( Show Ionograms    ) <
: IONOGRAM DECIMAL HERE [COMPILE] IONOGRAM                           <
           HERE OVER = IF DROP ELSE PFA IDISP ENDIF ;                <
                              ( Program and Display Ionogram ) <
                                                                --> <
                                                                     <
```

```
SCR # 49
( Edit Ionogram - Prompts )                                DECIMAL <
: DUR   1 ;         : DTT 2 ;                                      <
: SF    3 ;         : EF  4 ;        : ORT 5 ;                     <
: SOUND 6 ;                                                        <
: EIP VAL                                                          <
           ID=  ' DUR NPR  5 -> TT= ' DTT   NPR CR                 <
     14 -> SF=  ' SF  NPR  6 -> EF= ' EF    NPR CR                 <
     14 -> OR=  ' ORT NPR  5 -> SN= ' SOUND NPR                    <
     20  2 GOTOXY ;                 ( Edit Ionogram Prompt )       <
: PARI EIP ."  - Change >>"                                        <
      BEGIN INWORD DUP ' DUR < OVER ' SOUND > + 0= EXEC UNTIL ;<
                                    ( Parameter Rrquest    )       <
                                                               --> <
                                                                   <
                                                                   <
                                                                   <
```

```
SCR # 50
( Edit Ionogram Parameter )                                DECIMAL <
: EDR  GETDUR SWAP ROT >R >R 4 RPFA! R> 6 RPFA! R> 8 RPFA!         <
       IDR. ;                                                      <
: ETT  TOMT?   32 RPFA! TT. ;                                      <
: DOPF FP GETF 10 RPFA F2TOMEM 18 RPFA F2TOMEM ;                   <
: ESF  SUBTYPE 124 =                                               <
       IF DOPF EF. VEL. ELSE GETSF 10 RPFA F2TOMEM ENDIF SF. ;     <
: EEF  SUBTYPE 124 =                                               <
       IF DOPF SF. VEL. ELSE GETEF 18 RPFA F2TOMEM ENDIF EF. ;     <
: EOR  SUBTYPE 124 = IF W?B                                        <
       ELSE GETOR SWAP >R 26 RPFA! R> 28 RPFA! OR. 28 RPFA@ 0=     <
       IF 1 26 RPFA! EDR ENDIF ENDIF ;                             <
: ESN  SUBTYPE 124 = IF   GETDSND                                  <
                     ELSE GETISND ENDIF 30 RPFA! DISI ;            <
: IVEC  VECTOR EDR ETT ESF EEF EOR ESN ;                           <
: EDITI PARI IVEC ;                                            --> <
```

```
SCR # 51
( Display and Edit Ionogram )                              DECIMAL <
: PDEI  VAL ."       C = Change       E = Exit"  H+1 ." >" ;       <
: DEI   DISI                                                       <
        BEGIN PDEI KEYCR DUP 69 = 0=                               <
        WHILE 67 ( C ) = IF EDITI ELSE SPACE W?B ENDIF HOME        <
        REPEAT DROP DROP HOME ;                                    <
: IEDIT GETNAME IF DEI ENDIF ;                                     <
                                                               --> <
                                                                   <
                                                                   <
                                                                   <
                                                                   <
                                                                   <
                                                                   <
                                                                   <
                                                                   <
```

```
SCR # 52
( Ionogram Calculations )                               DECIMAL <
: STKF1   7 0 DO 10 /MOD LOOP                                   <
          8 0 DO 48 OR F1MSB I + C! LOOP ; ( Stk to F1 in BCD ) <
: STKMPN  STKF1 F1TOMPN ;                  ( Stk to MPN       ) <
: ENTR    STKMPN ENTER ;                   ( Stk to MPN,Enter ) <
: F1STK   F1MSB 7 + 5 TO# ;                ( F1 to Stack [ n ]) <
: ANSR    NEWFREQ F1STK ;                  ( MPN answer to Stk) <
: CELMPN  CODE# SWAP STKMPN IF MPN1/X ELSE ENTER ENDIF ;        <
: SEC/SND CURS @ 12 RPFA@ OVER SUBTYPE SWAP DROP 0=             <
          IF 4 + ENDIF CELMPN 24 + @ STKMPN MPN* ;              <
: ORTOM   28 + @ ENTR ;                ( Overall Rate to MPN  ) <
: LINOR   ORTOM 250  STKMPN MPN* ;     ( Linear Overall Rate  ) <
: LNOR    ORTOM 1000 STKMPN MPN/ ;     ( Log Overall Rate ) --> <
                                                                <
                                                                <
                                                                <
```

```
SCR # 53
( Ionogram Setting )                                    DECIMAL <
:  SETDOPL SEC/SND 4 RPFA@ ENTR 3600 STKMPN MPN*                <
                   6 RPFA@ 60 *      STKMPN MPN+                <
                   8  +  @           STKMPN MPN+                <
                   MPNXEY MPN/ ANSR #SOUND ! ; ( Set Doppler  ) <
: SETIONO 18 RPFA FENTER 10 RPFA FTOMPN 26 RPFA@                <
  IF MPN- SEC/SND LINOR MPN* NEWFREQ F/SND F1TOMEM              <
  ELSE MPN/ MPNLN SEC/SND LNOR MPN* ENTER MPNEX EATS ATOMEM     <
  MPNROL ENDIF MPN/ ANSR #SOUND ! SEC/SND MPN* ANSR             <
  60 /MOD 60 /MOD CURI @ 4 + ! CURI @ 6 + ! CURI @ 8 + ! ;      <
                                             ( Set Ionogram   ) <
: STRCDAT 4 TOMPN MPN* NEWFREQ CFREQ F1TOMEM ( Cell Frequency ) <
          CURS @ 24 + CELL# @ 12 * + @ ANTS? ( Cell Antennas  ) <
          48 OR CANT C! 48 OR CANT 1+ C! ;   ( Store Cell Data) <
                                                            --> <
                                                                <
```

```
SCR # 54
( Ionogram Setting )                                    DECIMAL <
: ISET DUP CURI ! 30 RPFA@ DUP CURS ! SSET                      <
  IF SUBTYPE SETA 10 RPFA TOF1 FREQOUT F1TOMPN MPNMS A@ 124 =   <
   IF SETDOPL ELSE CURI @ 28 + @                                <
                   IF SETIONO ELSE F/SND 8 48 FILL SETDOPL ENDIF<
              ENDIF 1 SOUND# !                                  <
   CURS @ 26 + FTOMPN MPNMR MPN+ NEWFREQ F1MSB TOLATCH STRCDAT 1<
  ELSE NPR 0 ENDIF ;             ( Set Ionogram ) ( pfa --> f) <
                                                                <
: UISET GETNAME IF ISET DROP ENDIF ; ( User Set Ionogram      ) <
                                                            --> <
                                                                <
                                                                <
                                                                <
                                                                <
                                                                <
```

```
SCR # 55
( Timing Slip Parameters - Build Timing Slip )          DECIMAL <
: SHIFT=    ." Shift = " ;                            ( Shift    ) <
: SDR=      ." Shift Direction and Rate = " ;                    <
: GETSHIFT VAL ." xxx.xx ms" CR SHIFT= INPUT MSEC ;              <
: GETSDR    DRP CR SDR=                                          <
            BEGIN INWORD DUP ' A@1 < OVER ' R@20 > + 0= UNTIL    <
            DUP ' A@20 > ;          ( Direction and Rate       ) <
: TYPE3     768 + ;                 ( Datatype 3 - Timing Slip ) <
: TIMING    NAME? IF <BUILDS 1SEC GETSHIFT GETSDR                <
                 TYPE3   ( Type   ) ,                            <
                 ROT ROT ( Shift ) , ,                           <
                         ( d+r    ) ,   DOES> DROP ENDIF HOME ;  <
                                                                 <
: SUB3.     ." - TIMING SLIP ( " IF  ." RETARD )"                <
            ELSE ." ADVANCE )" ENDIF ;          ( Print Subtype ) <
                                                             --> <


SCR # 56
( Display Timing Slip )                                  DECIMAL <
: SHIFT. 11 0 GOTOXY SHIFT=                                      <
         4 RPFA@ 3 .R ." ." 6 RPFA@ . ." ms/s" ; ( Shift Print) <
: SDR.   13 0 GOTOXY                                             <
               SDR= 8 RPFA@ NPR ." ms/s" ;      ( Dir,Rate Print) <
: DIST   DUP CLRDIS 25 -> NPR SUBTYPE SUB3.                      <
         SHIFT. SDR. HOME PTRCR ;       ( Slip ) ( pfa --> pfa ) <
: TDISP  DIST DROP ;                    ( Slip ) ( pfa -->     ) <
: TDISPLAY GETNAME IF TDISP ENDIF ;     (   Display Timing Slip) <
: SHOWT     CLRD+1 30 -> ." Defined Timing Slip" CR CR 7 SPACES  <
            CONTEXT @ @                                          <
            BEGIN GENCR PFA TYPE? 3 = IF DUP LNPR ENDIF          <
            LFA @ DUP BDATA @ < UNTIL DROP ; ( Show Timing Slip) <
: TIMING DECIMAL HERE [COMPILE] TIMING HERE OVER = IF DROP       <
         ELSE PFA TDISP ENDIF ;  ( Program , Display Timing    ) <
                                                             --> <


SCR # 57
( Edit Timing Slip , Set Timing Slip )                   DECIMAL <
              : S   0 ;             : SDR    1 ;                 <
: ETP  VAL SHIFT= ' S NPR CR 14 -> SDR= ' SDR NPR 20 2 GOTOXY ; <
: PART  ETP ." - Change >>"                                      <
        BEGIN INWORD DUP ' S < OVER ' SDR > + 0= EXEC UNTIL ;    <
: ES    GETSHIFT      >R 6 RPFA! R> 4 RPFA! ;                    <
: ESDR  GETSDR TYPE3 >R 8 RPFA! R> 2 RPFA! ;                     <
: EDITT PART IF ESDR ELSE ES ENDIF DIST ;                        <
: PDET  VAL ." C = Change          E = Exit" H+1 ." >" ;         <
: DET   DIST BEGIN PDET KEYCR DUP 69 = 0= WHILE 67 =             <
        IF EDITT ELSE SPACE W?B ENDIF REPEAT DROP DROP HOME ;    <
: TEDIT GETNAME IF DET ENDIF ;                                   <
: TSET  TYPE? SWAP DROP 3 = IF 1 ELSE IFR ." CHANGE" 0 ENDIF ;   <
: DOT   4 RPFA  D@ ROT 8 + @ CFA EXECUTE ;                       <
: UDOT  GETNAME IF DOT ENDIF HOME ;                              <
: PDOT  0 TFTN !   DOT    1 TFTN ! ;                        -->  <
OK
```

```
SCR # 58
  0 ( FORTH Word Parameters - Build FORTH Word )            DECIMAL <
  1 : FW=       ." FORTH Word = " ;                                 <
  2 : GETFW     VAL ." TESTED FORTH Word!" CR                       <
  3             FW= INWORD ;            ( Get FORTH Word          ) <
  4 : TYPE4     1024 ;                  ( Data Type 4 - FORTH Word ) <
  5 : FORTHWORD NAME? IF <BUILDS 1SEC TYPE4 , GETFW , DOES> DROP    <
  6                   ENDIF HOME ;             ( Build FORTH Word ) <
  7 : SUB4.    DROP ." - FORTH WORD  " ;       ( Print Type       ) <
  8 : FW.      FW= 4 RPFA@ NPR ;               ( FORTH Word Print ) <
  9 : DISF     DUP CLRDIS 25 -> NPR SUBTYPE SUB4. CR CR             <
 10            25 -> FW. HOME ;   ( Display Word ) ( pfa --> pfa  ) <
 11 : FDISP    DISF DROP ;        ( Display Word ) ( pfa -->      ) <
 12 : FDISPLAY GETNAME IF FDISP ENDIF ;              ( Display Word ) <
 13                                                             --> <
 14                                                                 <
 15                                                                 <


SCR # 59
  0 ( Display and Edit FORTH Word )                         DECIMAL <
  1 : SHOWF CLRD+1 25 -> ." FORTH Words for Exec in Prog" CR CR     <
  2        7 SPACES CONTEXT @ @                                     <
  3        BEGIN GENCR PFA TYPE? 4 = IF DUP LNPR ENDIF LFA @ DUP    <
  4        BDATA @ < UNTIL DROP ;             ( Show FORTH Words ) <
  5 : FORTHWORD DECIMAL HERE [COMPILE] FORTHWORD HERE OVER =        <
  6                IF DROP ELSE PFA FDISP ENDIF ; ( Prog , Disp Word ) <
  7 : EFW    GETFW OVER OVER >                ( Edit FORTH Word   ) <
  8          IF 4 RPFA! DISF ELSE DROP W?B ENDIF ;                  <
  9 : DEF    DISF BEGIN PDET KEYCR DUP 69 = 0= WHILE 67 =           <
 10          IF ." - Change Word" EFW ELSE SPACE W?B ENDIF          <
 11          REPEAT DROP DROP HOME ;                                <
 12 : FEDIT  GETNAME IF DEF ENDIF ;                                 <
 13 : FSET   TYPE? SWAP DROP 4 = IF 1 ELSE IFR ." CHANGE" 0 ENDIF ; <
 14 : DOFWD  4 + @ CFA EXECUTE ; ( Execute FORTH Word ) ( pfa --> ) <
 15 : UDOFWD GETNAME IF DOFWD ENDIF ; ( User Exec FORTH Word)   --> <


SCR # 60
  0 ( Set Place , Year , Weekday Number )                       HEX <
  1 : SETPL VAL PLACE 10 20 FILL ." Place Name" CR   ( Set Place  ) <
  2        ." Place = " PLACE 10 EXPECT                             <
  3        10 0 DO PLACE I + DUP C@ 0=                              <
  4        IF 20 SWAP C! ELSE DROP ENDIF LOOP ;                     <
  5 : PL.  10 0 DO PLACE I + C@ EMIT LOOP ;          ( Print Place) <
  6 : SETYR VAL ." 4 Digits" CR                      ( Set    Year ) <
  7        ." Year  = " YEAR 5 EXPECT                               <
  8        YEAR 2+ PACK FF AND 4 /MOD DROP 0= LEAP ! ;              <
  9 : YR.  4 0 DO YEAR  I + C@ EMIT LOOP ;           ( Print  Year) <
 10 : CLRDY 0 WDAY ! ;                     ( Clear Weekday number ) <
 11 : DYP   ." Sun=0  Mon=1  Tue=2  Wed=3  Thu=4  Fri=5  Sat=6" ;  <
 12 : SETDY VAL DYP CR ." Day of Week Number = "                    <
 13   BEGIN INPUT DROP DUP 0 < OVER 6 > + 0= GOXY UNTIL WDAY ! ;    <
 14                                        ( Set   Weekday number ) <
 15                                                             --> <
 OK
```

```
SCR # 61
  0 ( Read Real Time Clock )                                    HEX <
  1 : CLKIOF 2C E00B C! ;          ( Clock Interrupt off ie Masked ) <
  2 : CLKION 2D E00B C! ;          ( Clock Interrupt  on ie Enabled) <
  3 : RSTCLK E00A C@ DROP ;        ( Reset Clock 1 sec Flag        ) <
  4 : SEC?   BEGIN E00B C@ 80 AND UNTIL ; ( Wait for sec -ve edge ) <
  5 : RSTDV  E00C C@ DROP ;                ( Reset Data Valid Flag ) <
  6 : DV?    BEGIN E00D C@ 80 AND UNTIL ; ( Wait for DV -ve edge  ) <
  7 : RDCLK  RSTCLK SEC? RSTCLK BEGIN DV? E00C C@ F0 AND 90 = UNTIL <
  8          A 0 DO DV? E00C C@ F AND LOOP ; ( Read Clock to Stack) <
  9 : STYR   30 OR YEAR 3 + C!  ;               ( Store Years      ) <
 10 : STDYNO A * + A * + DYNO ! ;               ( Store Weekday no  ) <
 11 : STSEC  A * + >R A * + 3C * SWAP A * + + 0 DDUP CMS    D!      <
 12          R> -DUP IF 0 DO E10. D+ LOOP ENDIF DDUP CHMS  D!        <
 13   WDAY @ -DUP IF 0 DO  15180. D+ LOOP ENDIF      CDHMS D! ;      <
 14 : CPYCLK CMS D@ MS D! CHMS D@ HMS D! CDHMS D@ DHMS D! ;          <
 15 : GETIME CLKIOF RDCLK STYR STDYNO STSEC CPYCLK CLKION ;      --> <


SCR # 62
  0 ( Increment Real Time Clock - Get Time + 1 sec )        DECIMAL <
  1 : INCLK CMS 2+ @ 1+ DUP 3600 = IF DROP 0 ENDIF CMS 2+ !         <
  2   CHMS D@ 1. D+ DDUP 86400. D=                                   <
  3   IF DDROP 0. 1 WDAY +! DYNO @ 1+ DUP 365 >                      <
  4   IF DROP LEAP C@ IF 366 ELSE 1 ENDIF ENDIF DYNO !               <
  5   ENDIF CHMS D!                                                  <
  6   CDHMS D@ 1. D+ DDUP 604800. D=                                 <
  7   IF DDROP 0. 0 WDAY ! ENDIF CDHMS D! ;      ( Increment CLock  ) <
  8 : GETIME+1 GETIME INCLK ;                    ( Get Time + 1 sec ) <
  9                                                              --> <
 10                                                                  <
 11                                                                  <
 12                                                                  <
 13                                                                  <
 14                                                                  <
 15                                                                  <


SCR # 63
  0 ( Display Day , Display Time )                          DECIMAL <
  1 : TODHMS  60 M/ 60 /MOD 24 /MOD ;    ( Convert sec to D H M S  ) <
  2 : SU ." Sun"    4 ;   : MO ." Mon"   4 ;  : TU ." Tues" 3 ;      <
  3 : WE ." Wednes" 1 ;   : TH ." Thurs" 2 ;  : FR ." Fri"  4 ;      <
  4 : SA ." Satur"  2 ;                                              <
  5 : WDAYS VECTOR SU MO TU WE TH FR SA ;                            <
  6 : DAY.  1+ WDAYS ." day" SPACES ;    ( Print Day of Week no     ) <
  7                                                                  <
  8 : 2CON   0 <# # # #> TYPE  ;         ( Convert n to 2 Digits   ) <
  9 : MSC.   2CON ."  : " 2CON ;         ( Print MM : SS           ) <
 10 : HMS.   2CON ."  : " MSC. ;         ( Print HH : MM : SS      ) <
 11 : DHMS.  DAY. HMS. ;                 ( Print Day: HH : MM : SS ) <
 12 : CDHMS. CDHMS D@ 1. D- DDUP 0. D<                               <
 13          IF DDROP 604799. ENDIF TODHMS DHMS. ;                   <
 14                               ( Fetch and Print Weekday and Time) <
 15                                                              --> <
 OK
```

```
SCR # 64
  0 ( Allot Programme )                                       DECIMAL <
  1   1280   CONSTANT TYPE5                 ( Data Type 5 - Program ) <
  2 : ALLOTP NAME? IF IN @ 1 ELSE 0 ENDIF DUP                          <
  3          IF <BUILDS 1SEC TYPE5 , 75 0 DO -1. , , 0 , LOOP          <
  4          DOES> DROP ENDIF HOME ;         ( Allot Programme Space) <
  5 : SUB5.  DROP ." - PROGRAMME" 15 -> ;  ( Print Subtype          ) <
  6 : SHOWP  CLRD+1 30 -> ." Defined Programmes" CR CR 7 SPACES        <
  7          CONTEXT @ @ BEGIN GENCR PFA TYPE? 5 = IF DUP LNPR ENDIF<
  8          LFA @ DUP BDATA @ < UNTIL DROP ; ( Show Programmes   ) <
  9 : SHOVEC VECTOR SHOWI SHOWT SHOWF SHOWP ; ( Show Functions    ) <
 10 : SHOP VAL CR 14 -> ." Ionograms   = 1       Timing Changes = 2"<
 11             CR 14 -> ." FORTH Words = 3       Programmes     = 4"<
 12         20 11 GOTOXY ." - Show Function # >>"  0 BEGIN DROP       <
 13         INPUT DROP DUP 1 < OVER 4 > + 0= UNTIL SHOVEC VALA ;      <
 14 : SHFTNS ." Show Functions " YORN                                 <
 15          IF SHOP ELSE 23 CLRLN ENDIF ;                         --> <


SCR # 65
  0 ( Programme Parameters )                                  DECIMAL <
  1 : FUD      ."  Function Data" ;                                   <
  2 : PHF      VAL ." Hourly" FUD ;                                   <
  3 : GETMS ." Minutes = " MORS ." Seconds = " MORS SWAP 60 * + 0 ; <
  4 : PDF      VAL ." Daily" FUD ;                                    <
  5 : GTH      0. ." Hours = " 0 BEGIN DROP INPUT DROP DUP 24 < UNTIL<
  6            -DUP IF 0 DO 3600. D+ LOOP ENDIF 3 -> ;                <
  7 : GETHMS  GTH GETMS D+ ;                                          <
  8 : PWF      VAL ." Weekly Data (" DYP ." )" ;                      <
  9 : GTD      0. ." Weekday No. = " 0 BEGIN DROP INPUT DROP DUP 7 < <
 10            UNTIL -DUP IF 0 DO 86400. D+ LOOP ENDIF 3 -> ;          <
 11 : GETDHMS GTD GETHMS D+ ;                                         <
 12 : CHKTYP  DUP 2 < SWAP 5 > + 0= ;                                 <
 13 : GETFTN  23 0 GOTOXY SHFTNS ." Defined Function = "              <
 14            BEGIN INWORD TYPE? CHKTYP                               <
 15            IF 1 ELSE DROP 0 GOXY ENDIF UNTIL ;                 --> <


SCR # 66
  0 ( Display Programme )                                     DECIMAL <
  1 : EMTY?    DUP D@ DDUP -1.  D=    ; ( No Entry?) ( a --> a t f ) <
  2 : ENTRY?   DUP D@       -1.  D= 0= ; ( Entry ?  ) ( a --> a f    ) <
  3 : DSH      0  DO ." -" LOOP SPACE ; ( n Dashes ) ( n -->         ) <
  4 : 2DSH:    2 DSH ." : " ;              ( 2 Dashes , space , colon ) <
  5 : DASH     DDROP 9 DSH 2DSH: 2DSH: 2 DSH 2 SPACES 9 DSH 6 +        <
  6            87 OUT @ - SPACES CR ;   ( Dashes for Time and Ftn  ) <
  7 : TYPEVEC VECTOR SUB1. SUB2. SUB3. SUB4. SUB5. ;                  <
  8 : FTN.     TYPE? CHKTYP IF DUP NPR 2+ @ 256 /MOD TYPEVEC           <
  9            ELSE DROP IFR ." DELETE!" ENDIF CEOL ;  ( Print Ftn ) <
 10 : FTN@.    3 SPACES 4 + DUP @ FTN. 2+ CR ; ( Fetch , Print Ftn ) <
 11 : DSP1     TODHMS                      DHMS. FTN@. ; ( Display PB1) <
 12 : DSP2     TODHMS  DROP 9 DSH           HMS. FTN@. ; ( Display PB2) <
 13 : DSP3     TODHMS DDROP 9 DSH 2DSH:  MSC. FTN@. ; ( Display PB3) <
 14 : VEC123  VECTOR DSP1 DSP2 DSP3 ;                                 <
 15 : DSP123  EMTY? IF DASH 1 ELSE PB @ VEC123 0 ENDIF  ;          --> <
 OK
```

```
SCR # 67
  0 ( Display Programme )                                     DECIMAL <
  1 : E#.    6 / 1+ 9 ->  ." Entry " 2 .R 3 -> ; ( Print Entry #  ) <
  2 : VECTPB VECTOR TP1 TP2 TP3 ;                                     <
  3 : 8DISP  8 0 DO DUP PB @ VECTPB @ - E#. DSP123 DROP LOOP ;        <
  4 : ADJA   DUP PB @ < IF PB ! 6 - ELSE                              <
  5          DUP PB @ > IF PB ! 6 + ELSE PB ! ENDIF ENDIF PB @ ;      <
  6 : PB# DUP ." Priority Block #" TP1 @ - DUP 144 <                  <
  7   IF DROP 1 ELSE 294 < IF 2 ELSE 3 ENDIF ENDIF DUP . ADJA ;       <
  8 : DTF   5 -> ." Day" 9 -> ." Time" 12 -> ." Function" CR ;        <
  9 : PBDSP 9 0 GOTOXY PB# DTF                                        <
 10   CASE 1 OF ."   Weekly Data" ENDOF 2 OF ."   Daily  Data" ENDOF<
 11         3 OF ."   Hourly Data" ENDOF ENDCASE CR 8DISP 48 - ;      <
 12 : PEP VAL 3 ->                                                    <
 13   ." N = Next 8      1 = Top PB1    I = Insert   E = Exit"        <
 14   CR 17 -> ." P = Previous 8  2 = Top PB2    D = Delete"          <
 15   CR 33 ->                   ." 3 = Top PB3" H+1 ." >" ;      --> <


SCR # 68
  0 ( Display and Edit Programme )                            DECIMAL <
  1 : FNDPBT VECTPB @ ;                   ( Find Priority Block Top ) <
  2 : INE#B  0 BEGIN DROP INPUT DROP DUP 25 < UNTIL 1 - 6 * ;         <
  3 : MVDWN  DUP 138 = 0=                                             <
  4          IF OVER + DUP 6 - ROT 132 +                              <
  5          DO I DUP 6 + 6 CMOVE -6 +LOOP                            <
  6          ELSE + ENDIF DUP ;          ( Move Entries down 1 posn) <
  7 : INSERT VAL PB# CR FNDPBT ." Insert at Entry No. = "             <
  8          INE#B MVDWN PB @ CASE 1 OF PWF CR GETDHMS ENDOF          <
  9                    2 OF PDF CR GETHMS  ENDOF                      <
 10                    3 OF PHF CR GETMS   ENDOF ENDCASE              <
 11                    ROT  D! 4 + GETFTN SWAP ! ; ( Insert Entry ) <
 12 : DELENT VAL PB# CR FNDPBT ." Delete Entry No. = " INE#B          <
 13          OVER + DUP 6 + SWAP ROT 150 + OVER 6 + - CMOVE ;         <
 14                                                 ( Delete Entry ) <
 15                                                                --> <


SCR # 69
  0 ( Programme Display Control )                             DECIMAL <
  1 : DEPINIT DUP CLRDIS 30 -> NPR SUBTYPE SUB5.          CR          <
  2          4 + DUP TP1 ! DUP 150 + TP2 ! DUP 300 + TP3 ! 1 PB ! ; <
  3 : DEP    DEPINIT PBDSP                                            <
  4          BEGIN PEP KEYCR DUP 69 = 0= WHILE                        <
  5          CASE 80 ( P ) OF 48 - TP1 @        MAX PBDSP ENDOF       <
  6               78 ( N ) OF 48 + TP3 @ 96   + MIN PBDSP ENDOF       <
  7               49 ( 1 ) OF DROP 1 PB ! TP1 @    PBDSP ENDOF        <
  8               50 ( 2 ) OF DROP 2 PB ! TP2 @    PBDSP ENDOF        <
  9               51 ( 3 ) OF DROP 3 PB ! TP3 @    PBDSP ENDOF        <
 10               73 ( I ) OF ." - Insert" INSERT  PBDSP ENDOF        <
 11               68 ( D ) OF ." - Delete" DELENT  PBDSP ENDOF        <
 12               SPACE W?B ENDCASE                                   <
 13               REPEAT DROP DROP HOME ; ( Display and Edit Prog ) <
 14                                                                --> <
 15                                                                    <
 OK
```

```
SCR # 70
 ( Display and Edit Programme )                           DECIMAL <
 : PRTP DEPINIT CR PB# PB ! DTF           ." Weekly Data" CR    <
        24 0 DO DUP TP1 @ - E#. DSP123 IF LEAVE ENDIF LOOP CR   <
        DROP TP2 @ 2 PB ! PB# DTF DROP     ." Daily  Data" CR   <
        24 0 DO DUP TP2 @ - E#. DSP123 IF LEAVE ENDIF LOOP CR   <
        DROP TP3 @ 3 PB ! PB# DTF DROP     ." Hourly Data" CR   <
       24 0 DO DUP TP3 @ - E#. DSP123 IF LEAVE ENDIF LOOP DROP ; <
 : PROGRAMME DECIMAL ALLOTP IF IN ! [COMPILE] ' DEP ENDIF HOME ; <
 : PROGRAM   PROGRAMME ;                  ( Alternative command ) <
 : PDISP     V@ IF DEP ELSE PRTP ENDIF   ;         ( pfa -->    ) <
 : PEDIT     GETNAME IF DEP   ENDIF HOME ;         (     -->    ) <
 : PSET      TYPE? 5 =                             ( pfa --> f  ) <
             IF CURPRG ! 1 ELSE DROP -1 CURPRG ! 0 ENDIF ;        <
 : UPSET     GETNAME IF PSET DROP ENDIF HOME ;    (     -->     ) <
 : CPRG. 3 0 GOTOXY ." Current Programme:- " CURPRG @ DUP -1 =   <
             IF DROP ." Not Set!" ELSE NPR 6 -> ENDIF ;       --> <

SCR # 71
 ( Programme Lookahead )                                  DECIMAL <
 : CHKW DUP DUP D@ DHMS D@ D- DDUP 1. D<                          <
        IF 604800. D+ ENDIF DDUP WC D@ D<                         <
        IF WC D! WEA ! ELSE DDROP DROP ENDIF 6 + ; ( Check PB1 ) <
 : NXTW 900000. WC D! CURPRG @ 4 + DUP WEA ! ENTRY?               <
        IF BEGIN CHKW ENTRY? 0= UNTIL ENDIF DROP ; ( Next  PB1 ) <
 : CHKD DUP DUP D@ HMS D@ D- DDUP 1. D<                           <
        IF 86400. D+ ENDIF DDUP DC D@ D<                          <
        IF DC D! DEA ! ELSE DDROP DROP ENDIF 6 + ; ( Check PB2 ) <
 : NXTD 900000. DC D! CURPRG @ 154 + ENTRY?                       <
        IF BEGIN CHKD ENTRY? 0= UNTIL ENDIF DROP ; ( Next  PB2 ) <
 : CHKH DUP DUP D@ MS D@ D- DDUP 1. D<                            <
        IF 3600. D+ ENDIF DDUP HC D@ D<                           <
        IF HC D! HEA ! ELSE DDROP DROP ENDIF 6 + ; ( Check PB3 ) <
 : NXTH 900000. HC D! CURPRG @ 304 + ENTRY?        ( Next  PB3 ) <
        IF BEGIN CHKH ENTRY? 0= UNTIL ENDIF DROP ;             --> <

SCR # 72
 ( Programme Lookahead )                                      HEX <
 : NTOC NXTIME D@ CURTIME D! NXTFTN @ CURFTN ! NPB @ CPB ! ;     <
              ( Moves Next Time and NPB to Current Time and CPB ) <
 : EATTI DUP D@ NXTIME D! 4 + @ NXTFTN ! NPB ! ;                  <
              ( Time to NXTIME , PB to NPB) ( PB ent-addr -->   ) <
 : GETNXT CURPRG @ -1 = 0= IF NXTW NXTD NXTH WC D@ DC D@ D>       <
            IF 2 DEA @ DC D@ ELSE 1 WEA @ WC D@ ENDIF HC D@ D>    <
  IF DROP DROP 3 HEA @ ENDIF NTOC EATTI ELSE 0 NXTFTN ! ENDIF ; <
 ( Select smallest of WC DC HC if Programme Set )   (    -->    ) <
 : GETC=N GETNXT NTOC ;     ( Get Next Ftn and copy to Current  ) <
 : PBPRT  VEC123 DROP ;     ( Print Priority Block              ) <
 : CNDIS CURPRG @ -1 = 0= IF CR 4 0 GOTOXY                        <
 ." Current Function :- " CURFTN @ 0= IF ." Empty Programme" CEOL<
   CR ELSE CURTIME DUP D@ CPB @ PBPRT ENDIF                        <
 ." Next    Function :- " NXTFTN @ 0= IF ." Empty Programme" CEOL<
   ELSE NXTIME DUP D@ NPB @ PBPRT ENDIF ENDIF ;                --> <
```

```
SCR # 73
( Data to Film )                                                HEX <
-->                                                                 <
                                                                    <
                                                                    <
                                                                    <
                                                                    <
                                                                    <
                                                                    <
                                                                    <
                                                                    <
                                                                    <
                                                                    <
                                                                    <
                                                                    <
                                                                    <
                                                                    <
```

```
SCR # 74
( Data to DCS )                                                 HEX <
: RSTDCS  0 DCSTAT ! ;                    ( Clear DCS Status Flag ) <
: CLRACK  E00E C@ DROP ;                  ( Clear Acknowledge Flag) <
: DCSACK? 0                                                         <
          A 0 DO DROP I E00F C@ 80 AND IF LEAVE ENDIF LOOP          <
          9 = IF 1 DCSTAT ! BELL ENDIF CLRACK ;                     <
: TODCS   DCS?  @           ( Acsii to DCS ) ( Char -->           ) <
          IF DUP EMIT ENDIF                      ( Print DCS Data ) <
          DCSTAT @                                                  <
          IF DROP ELSE E00E C! DCSACK? ENDIF ; ( Send to DCS      ) <
: ATODCS  30 OR TODCS ;     ( Convert n , 0 to 9 , to Ascii       ) <
: NTODCS  DUP >R 1 - 0 DO A /MOD      LOOP                          <
             R>     0 DO 30 OR TODCS LOOP ; ( n1 n2 --->          ) <
: DCSCR   DCS? @ IF CR ENDIF ;                                      <
: DCSTAB  DCS? @ IF OUT @ 4E > IF CR ENDIF ENDIF ;                  <
: DCSDATA CLRVDU 0 20 GOTOXY ." DCS Data " CR 1 DCS? ! ;    -->     <
```

```
SCR # 75
( Data to DCS )                                                 HEX <
: PLDCS 10 0 DO PLACE    I + C@ TODCS LOOP ;    ( Place to DCS ) <
: YRDCS  2 0 DO YEAR  2+ I + C@ TODCS LOOP ;    ( Year  to DCS ) <
: DHMSDCS DYNO @ 3 NTODCS                       ( Time  to DCS ) <
          CDHMS D@ TODHMS DROP 3 0 DO 2 NTODCS LOOP ;            <
: DATDCS  CURS @ DUP 04 + @ RATE 4 NTODCS  DUP 18 + @  2 NTODCS <
                 DUP 0C + @ OVER 2+ @ 0= IF  4 + ENDIF 2 NTODCS <
                 DUP 10 + @ 3 +  2 NTODCS    A + @     5 NTODCS <
                 SHIFT D@ DDUP 0. D< IF 2D ELSE 2B ENDIF TODCS  <
  DABS <# # # # # # # #> OVER + SWAP DO I C@ TODCS LOOP ;        <
: IDAT    TODCS  PLDCS YRDCS DHMSDCS DATDCS 3F TODCS DCSCR ;    <
: IDATDCS RSTDCS 80 IDAT ;    ( Data to DCS only               ) <
: IDATMT  RSTDCS 81 IDAT ;    ( Data to DCS and Magnetic Tape ) <
: CDATDCS RSTDCS 82 TODCS     ( Cell Data to DCS               ) <
          CFREQ A + CFREQ DO I C@ TODCS LOOP 3F TODCS DCSTAB ;  <
: ENDCS   RSTDCS 83 TODCS 3F TODCS ;  ( End code to DCS )   --> <
```

```
SCR # 76
( Print Title and Display Status )                          DECIMAL <
: HDING CLRVDU    7 DSHLN 19 DSHLN                                  <
        0  0 GOTOXY ." ANTARCTIC RESEARCH"               10 SPACES<
                    ." VERTICHIRP CONTROLLER"            10 SPACES<
                    ." RHODES UNIVERSITY" ;                         <
: PLYR. 1 28 GOTOXY    PL. SPACE YR.       ;                        <
: DYNO. 2  0 GOTOXY ." Day Number       :- " DYNO   @   .  ;       <
: TPOSN. 6 0 GOTOXY ." Timing  Position :- " POSN ." --->" ;       <
                                                                    <
: TITLE DECIMAL HDING PLYR. DYNO. CPRG. CNDIS TPOSN.                <
        12 25 GOTOXY ." Type HELP for Command List " HOME ;         <
                                                               --> <
                                                                    <
                                                                    <
                                                                    <
                                                                    <
```

```
SCR # 77
( Display Datatype. Edit Datatype )                          DECIMAL <
: PRINTER PRNT ;                                                     <
: DISFTN DECIMAL TYPE?                                               <
         VECTOR SDISP ( Sounding      ) IDISP ( Ionogram    )        <
                TDISP ( Timing Change ) FDISP ( FORTH Word )         <
                PDISP ( Program       ) ;                            <
: DISPLAY VDU  TITLE GETNAME IF DISFTN ENDIF HOME ; ( To VDU   ) <
: PRINT   PRNT GETNAME IF DISFTN ENDIF HOME ; ( Printer or VDU) <
: LOG     PRNT GETNAME IF TYPE? 2 =           ( Print for Log ) <
          IF   DUP DISFTN 30 + @ ENDIF DISFTN ENDIF HOME ;           <
                                                                     <
: EDFTN   DECIMAL TYPE?                                              <
          VECTOR  DES ( Sounding      ) DEI ( Ionogram    )          <
                  DET ( Timing Change ) DEF ( FORTH Word )           <
                  DEP ( Programme     ) ;                            <
: EDIT GETNAME IF EDFTN ENDIF HOME ;  ( Edit Datatype     )  --> <
```

```
SCR # 78
( Set Datatype )                                             DECIMAL <
: SETFTN  TYPE? VECTOR SSET ( Sounding  )  ISET ( Ionogram     ) <
                       TSET ( Timing    )  FSET ( FORTH Word   ) <
                       PSET ( Programme ) ;                          <
: SETNXT GETNXT NXTFTN @ DUP                                         <
         IF SETFTN ENDIF DUP NXTSET ! COMPARE ! ;                    <
: SET    GETNAME IF CR ." Busy --> " SETFTN                          <
                 IF ." Set !" ENDIF ENDIF B1S 1SEC HOME ;            <
                                              ( Set Function ) <
                                                               --> <
                                                                     <
                                                                     <
                                                                     <
                                                                     <
                                                                     <
                                                                     <
```

```
SCR # 79
( Show and Delete Datatype )                            DECIMAL <
: SHOWTYP CLRDIS 28 -> ." Defined Functions " CR                <
          7 SPACES CONTEXT @ @ BEGIN GENCR PFA                  <
          DUP LNPR LFA @ DUP BDATA @ U< UNTIL DROP ;            <
                                                  ( Show Type ) <
: DLIST   SHOWTYP VAL ." Function Name" CR                      <
                      ." Erase Function" YORN CR                <
          IF ." Erase up to " INWORD                            <
             DUP CURPRG @ 1+ U< IF -1 CURPRG ! ENDIF            <
             BDATA @ U<                                         <
             IF U?B ELSE 0 IN ! FORGET TITLE SHOWTYP ENDIF      <
          ENDIF HOME ;                            ( List Data ) <
                                                            --> <
                                                                <
                                                                <
                                                                <
```

```
SCR # 80
( Sounding and Cell Start Frequencies. Function Stop )  DECIMAL <
: DISCN  S@ IF CNDIS H+1 ENDIF ;                                <
: SETUP  .5 1 DPL ! MHZ DROP SETFREQ     ( Freq to .5 MHz     ) <
  AGCINT GETIME+1 SETNXT DISCN 0 ION ! ; ( AGC to int,set,disp) <
: FNDNXT CPYCLK GETNXT DISCN ;           ( Find,Disp next Ftn ) <
: SSFLIN F/SND FTOMPN MPNMR MPN+ MPNMS ; ( Snd start f,Linear ) <
: SSFLN  EATS  MTOMPN MPNMR MPN* MPNMS ; ( Snd start f,Log    ) <
: CSFN   CURS @ 26 + CELL# @ 1 - 12 * +                         <
         DUP FTOMPN MPNMR MPN+ NEWFREQ F1MSB TOLATCH            <
         DUP 8 + @ IF FILMON ELSE FILMOFF ENDIF                 <
            10 + @ 15 STORE  STRCDAT ;   ( Cell n start freq  ) <
: STO,1 STROBE0 STROBE1 ;                                       <
: STOP  STO,1 DCS? @ IF 0 DCS? ! S@ IF TITLE ENDIF ENDIF        <
        ENDCS SETUP ;                             ( Stop Iono ) <
: HALT  COMPARE @ 0= IF #SOUND @ SOUND# ! ENDIF ; ( End  Iono ) <
: HALTED? BEGIN ION @ 0= UNTIL ;                            --> <
```

```
SCR # 81
( Tcell Interrupt Routine )                             DECIMAL <
: NCELL      CURI @ 2+ @ 256 MOD 124 =                          <
  IF SOUND# @ #SOUND @ > IF STOP H+1 ELSE                       <
     CDATDCS CURS @ DUP 20 + @ -DUP                             <
     IF 1 AGC# +! AGC# @ < IF RXPULSE 1 AGC# ! ENDIF ENDIF      <
                        22 + @ -DUP                             <
     IF 1 TMK# +! TMK# @ < IF DOPTIME 1 TMK# ! ENDIF ENDIF      <
       1 CELL# +! CELL# @ CURS @ 24 + @ >                       <
       IF 1 CELL# ! 1 SOUND# +! ENDIF CSFN ENDIF                <
  ELSE SOUND# @ #SOUND @ > IF STOP H+1 ELSE                     <
       CELL# @ 1 = IF CURS @ 20 + @ IF RXPULSE ENDIF ENDIF      <
       CDATDCS 1 CELL# +! CELL# @ CURS @ 24 + @ >               <
          IF 1 CELL# ! 1 SOUND# +! CURI @ 26 + @                <
             IF SSFLIN ELSE SSFLN ENDIF                         <
          ENDIF CSFN       ENDIF                                <
  ENDIF ;                                                   --> <
```

```
SCR # 82
  0 ( Run Function )                                       DECIMAL <
  1 : RS   DROP W?B 1SEC HOME ;      ( Run Sounding - not allowed ) <
  2 : RI   AGCEXT 32 + @ IF IDATMT ELSE IDATDCS ENDIF               <
  3        SWEEP 0 COMPARE ! 1 ION ! FNDNXT ;   ( Run Ionogram    ) <
  4 : RTS  GETIME+1 FNDNXT PDOT  STO,1 SETUP ;  ( Run Timing Slip ) <
  5 : RFW  GETIME+1 FNDNXT DOFWD STO,1 SETUP ;  ( Run FORTH Word  ) <
  6 : RPRG DROP GETIME+1 GETC=N SETNXT TITLE ;  ( Run Programme   ) <
  7 : RFTN TYPE? VECTOR    RS ( Sounding    ) RI   ( Ionogram )     <
  8   RTS  ( Timing Slip ) RFW ( FORTH Word ) RPRG ( Program  ) ;   <
  9 : RUN  HALT HALTED? GETNAME IF DUP SETFTN                       <
 10        IF CMS D@ BEGIN DDUP CMS D@ D= 0= UNTIL DDROP RFTN       <
 11        ELSE DROP ENDIF ENDIF ;                 ( Run Function ) <
 12 : AOFF   S@ STATUSOFF HALT HALTED? -1 CURPRG ! SETNXT STAT ! ;  <
 13 : CANCEL AOFF TITLE ;                          ( Cancel Prog. ) <
 14 : PAUSE  CURPRG @ AOFF CLRVDU 12 20 GOTOXY      ." PAUSE - Press<
 15  Space Bar to continue" BEGIN ?TERMINAL UNTIL PSET RPRG ;   --> <


SCR # 83
  0 ( Clock Service Routine )                              DECIMAL <
  1 : PBTIME VECTOR CDHMS CHMS CMS ;                                <
  2 : TIME   1 TON !    ."  Press Space Bar to stop"                <
  3          3 0 GOTOXY ." Current Time     :- "                    <
  4          BEGIN ?TERMINAL UNTIL 0 TON ! 3 CLRLN CPRG. HOME ;     <
  5 : CLOCK  INCLK  COMPARE @                                       <
  6          IF NPB @ PBTIME D@ NXTIME D@ D=                        <
  7             IF NXTFTN @ RFTN ENDIF                              <
  8          ENDIF                                                  <
  9          TON @                                                  <
 10          IF 3 20 GOTOXY V@ 0=                                   <
 11             IF 0 TON ! ENDIF CDHMS.                             <
 12          ENDIF ;                                            --> <
 13                                                                  <
 14                                                                  <
 15                                                                  <


SCR # 84
  0 ( Interrupt Polling )                                      HEX <
  1 : POLL   E009 C@ 80 AND IF E008 C@ DROP NCELL ELSE              <
  2          E00B C@ 80 AND IF E00A C@ DROP CLOCK ELSE              <
  3          ." Interrupt Error " ENDIF ENDIF RTI ;                 <
  4 : SETIRQ LIT POLL 23E6 ! 1CE9 DFC8 ! ;                      --> <
  5                                                                  <
  6                                                                  <
  7                                                                  <
  8                                                                  <
  9                                                                  <
 10                                                                  <
 11                                                                  <
 12                                                                  <
 13                                                                  <
 14                                                                  <
 15                                                                  <
```

```
SCR # 85
( Command List )                                             DECIMAL <
: T1 20 TAB ;  : HD CLRVDU CR T1 ;  : R1 CR T1 ;  : T2 45 TAB ; <
: U1 ." x" T2 ." x = User assigned Data Name" ; : U U1 R1 ;     <
: W ." x" T2 ." x = Defined Data Name" R1 ;                     <
: HELP  HD ." -Command-" T2 ." -Function-" CR ." System"     T1 <
." COLD" T2 ." Cold Start" R1 ." WARM" T2 ." Warm Start"     R1 <
." VDU" T2 ." VDU I/O" R1 ." PRNT" T2 ." Printer or VDU I/O" R1 <
." TITLE" T2 ." Shows Status" R1 ." STATUSOFF" T2 ." No Updates"<
 R1 ." TIME" T2 ." Shows Time" R1 ." SETTIME" T2 ." Sets Time"  <
CR ." Data Entry " T1 ." SOUNDING  " U ." IONOGRAM  " U         <
   ." TIMING    " U   ." FORTHWORD " U ." PROGRAMME " U1        <
CR ." Data Display/Modify"   T1 ." EDIT    " W ." DISPLAY " W   <
  ." PRINT   " W ." LOG     " W ." DLIST" T2 ." List Data Names"<
 CR ." Execution" T1                                            <
     ." SET " W ." RUN " W ." HALT" T2 ." Stop Ionogram" CR CR  <
." Press SPACE BAR to exit" BEGIN ?TERMINAL UNTIL TITLE ; -->   <


SCR # 86
( System Initialisation )                                    DECIMAL <
: SETTM VAL CR ." Set time then press SPACE BAR "               <
        BEGIN ?TERMINAL UNTIL ;                     ( Set Time ) <
: VWARM SEIF SETIRQ CR  ." Busy !"                              <
        1 TFTN ! 1 STAT ! 1 NPB !                               <
        0 DCS? ! 0 TON  ! 0 SRCH !                              <
        PORTS STROBE0 STROBE1 CLI SETUP GETNXT TITLE ; ( Warm ) <
: VCOLD SEIF CR CR ." Heath VDU " YORN VDU? ! HDING             <
        8 30 GOTOXY ." Initialization"                          <
        TPOSN. CLRPOSN SETPL SETYR SETDY SETTM                  <
        -1 CURPRG ! 0 COMPARE ! -26567 BDATA ! VWARM ; ( Cold ) <
: COLD? HERE 37 @ ( initial value for DP ie DPINIT ) =          <
        IF VCOLD ELSE VWARM ENDIF QUIT ; ( Cold or Warm Start?) <
: SETTIME SETTM SETUP GETNXT TITLE CR TIME ;        ( Set Time ) <
HEX 982E 23 ! 9839 25 ! 9810 27 ! 9808 115F ! ´ COLD? CFA 1177 !<
 ´ SETTIME NFA 1196 ! 982E 119E ! 9800 11C3 !   DECIMAL      ;S <
```

FORTH HANDY REFERENCE

Stack inputs and outputs are shown; top of stack on right.
This card follows usage of the Forth Interest Group (S.F. Bay Area) usage aligned with the *Forth 78* International Standard

For more info.
Forth Interest Group of Southern Africa
P.O. Box 29452
0132 Sunnyside

Operand key:

n, n1, . . .	16-bit signed numbers
d, d1, . . .	32-bit signed numbers
u	16-bit unsigned number
addr	address
b	8-bit byte
c	7-bit ascii character value
f	boolean flag

STACK MANIPULATION

DUP	(n → n n)	Duplicate top of stack.
DROP	(n →)	Throw away top of stack
SWAP	(n1 n2 → n2 n1)	Reverse top two stack items.
OVER	(n1 n2 → n1 n2 n1)	Make copy of second item on top.
ROT	(n1 n2 n3 → n2 n3 n1)	Rotate third item to top.
-DUP	(n → n ?)	Duplicate only if non-zero.
>R	(n →)	Move top item to "return stack" for temporary storage (use caution).
R>	(→ n)	Retrieve item from return stack.
R	(→ n)	Copy top of return stack onto stack.

NUMBER BASES

DECIMAL	(→)	Set decimal base.
HEX	(→)	Set hexadecimal base.
BASE	(→ addr)	System variable containing number base.

ARITHMETIC AND LOGICAL

+	(n1 n2 → sum)	Add.
D+	(d1 d2 → sum)	Add double-precision numbers.
-	(n1 n2 → diff)	Subtract (n1−n2).
*	(n1 n2 → prod)	Multiply.
/	(n1 n2 → quot)	Divide (n1/n2).
MOD	(n1 n2 → rem)	Modulo (*i.e.* remainder from division).
/MOD	(n1 n2 → rem quot)	Divide, giving remainder and quotient.
*/MOD	(n1 n2 n3 → rem quot)	Multiply, then divide (n1*n2/n3), with double-precision intermediate.
*/	(n1 n2 n3 → quot)	Like */MOD, but give quotient only.
MAX	(n1 n2 → max)	Maximum.
MIN	(n1 n2 → min)	Minimum.
ABS	(n → absolute)	Absolute value.
DABS	(d → absolute)	Absolute value of double-precision number.
MINUS	(n → -n)	Change sign.
DMINUS	(d → -d)	Change sign of double-precision number.
AND	(n1 n2 → and)	Logical AND (bitwise).
OR	(n1 n2 → or)	Logical OR (bitwise).
XOR	(n1 n2 → xor)	Logical exclusive OR (bitwise).

COMPARISON

<	(n1 n2 → f)	True if n1 less than n2.
>	(n1 n2 → f)	True if n1 greater than n2.
=	(n1 n2 → f)	True if top two numbers are equal.
0<	(n → f)	True if top number negative.
0=	(n → f)	True if top number zero (*i.e.*, reverses truth value).

MEMORY

@	(addr → n)	Replace word address by contents.
!	(n addr →)	Store second word at address on top.
C@	(addr → b)	Fetch one byte only.
C!	(b addr →)	Store one byte only.
?	(addr →)	Print contents of address.
+!	(n addr →)	Add second number on stack to contents of address on top
CMOVE	(from to u →)	Move u bytes in memory
FILL	(addr u b →)	Fill u bytes in memory with b, beginning at address.
ERASE	(addr u →)	Fill u bytes in memory with zeroes, beginning at address.
BLANKS	(addr u →)	Fill u bytes in memory with blanks, beginning at address.

CONTROL STRUCTURES

DO LOOP	do (end+1 start →)	Set up loop, given index range
I	(→ index)	Place current index value on stack
LEAVE	(→)	Terminate loop at next LOOP or +LOOP.
DO +LOOP	do (end+1 start →) +loop (n →)	Like DO LOOP, but adds stack value (instead of always '1') to index
IF (true) ENDIF	if (f →)	If top of stack true (non-zero), execute. [Note: *Forth 78* uses IF THEN]
IF (true) ELSE (false) ENDIF	if (f →)	Same, but if false, execute ELSE clause. [Note: *Forth 78* uses IF ELSE THEN]
BEGIN UNTIL	until (f →)	Loop back to BEGIN until true at UNTIL. [Note: *Forth 78* uses BEGIN END]
BEGIN WHILE REPEAT	while (f →)	Loop while true at WHILE. REPEAT loops unconditionally to BEGIN [Note: *Forth 78* uses BEGIN IF AGAIN]

TERMINAL INPUT-OUTPUT

	(n →)	Print number.
R	(n fieldwidth →)	Print number, right-justified in field
D.	(d →)	Print double-precision number.
D.R	(d fieldwidth →)	Print double-precision number, right-justified in field.
CR	(→)	Do a carriage return.
SPACE	(→)	Type one space.
SPACES	(n →)	Type n spaces.
."	(→)	Print message (terminated by ").
DUMP	(addr u →)	Dump u words starting at address.
TYPE	(addr u →)	Type string of u characters starting at address.
COUNT	(addr → addr+1 u)	Change length-byte string to TYPE form.
?TERMINAL	(→ f)	True if terminal break request present.
KEY	(→ c)	Read key, put ascii value on stack.
EMIT	(c →)	Type ascii value from stack
EXPECT	(addr n →)	Read n characters (or until carriage return) from input to address
WORD	(c →)	Read one word from input stream, using given character (usually blank) as delimiter.

INPUT-OUTPUT FORMATTING

NUMBER	(addr → d)	Convert string at address to double-precision number.
<#	(→)	Start output string.
#	(d → d)	Convert next digit of double-precision number and add character to output string.
#S	(d → 0 0)	Convert all significant digits of double-precision number to output string.
SIGN	(n d → d)	Insert sign of n into output string.
#>	(d → addr u)	Terminate output string (ready for TYPE).
HOLD	(c →)	Insert ascii character into output string.

DISK HANDLING

LIST	(screen →)	List a disk screen.
LOAD	(screen →)	Load disk screen (compile or execute).
BLOCK	(block → addr)	Read disk block to memory address.
B/BUF	(→ n)	System constant giving disk block size in bytes.
BLK	(→ addr)	System variable containing current block number.
SCR	(→ addr)	System variable containing current screen number.
UPDATE	(→)	Mark last buffer accessed as updated.
FLUSH	(→)	Write all updated buffers to disk.
EMPTY-BUFFERS	(→)	Erase all buffers.

DEFINING WORDS

: xxx	(→)	Begin colon definition of xxx.
;	(→)	End colon definition.
VARIABLE xxx	(n →) xxx: (→ addr)	Create a variable named xxx with initial value n; returns address when executed.
CONSTANT xxx	(n →) xxx: (→ n)	Create a constant named xxx with value n; returns value when executed.
CODE xxx	(→)	Begin definition of assembly-language primitive operation named xxx.
;CODE	(→)	Used to create a new defining word, with execution-time "code routine" for this data type in assembly.
<BUILDS . . . DOES>	does: (→ addr)	Used to create a new defining word, with execution-time routine for this data t higher-level Forth.

VOCABULARIES

CONTEXT	(→ addr)	Returns address of pointer to context vocabulary (searched first)
CURRENT	(→ addr)	Returns address of pointer to current vocabulary (where new definitions are put)
FORTH	(→)	Main Forth vocabulary (execution of FORTH sets CONTEXT vocabulary)
EDITOR	(→)	Editor vocabulary; sets CONTEXT
ASSEMBLER	(→)	Assembler vocabulary; sets CONTEXT.
DEFINITIONS	(→)	Sets CURRENT vocabulary to CONTEXT
VOCABULARY xxx	(→)	Create new vocabulary named xxx
VLIST	(→)	Print names of all words in CONTEXT vocabulary

MISCELLANEOUS AND SYSTEM

(	(→)	Begin comment, terminated by right paren on same line; space after (
FORGET xxx	(→)	Forget all definitions back to and including xxx
ABORT	(→)	Error termination of operation
' xxx	(→ addr)	Find the address of xxx in the dictionary, if used in definition, compile address
HERE	(→ addr)	Returns address of next unused byte in the dictionary
PAD	(→ addr)	Returns address of scratch area (usually 68 bytes beyond HERE)
IN	(→ addr)	System variable containing offset into input buffer, used, e.g., by WORD
SP@	(→ addr)	Returns address of top stack item
ALLOT	(n →)	Leave a gap of n bytes in the dictionary
,	(n →)	Compile a number into the dictionary

FORTH Error Messages

```
Error No   Message

err # 1 EMPTY STACK
err # 2 DICTIONARY FULL
err # 3 HAS INCORRECT ADDRESS MODE
err # 4 ISN'T UNIQUE
err # 5
err # 6 DISC RANGE ?
err # 7 FULL STACK
err # 8 DISC ERROR !
err # 9
err # 10
err # 11
err # 12
err # 13
err # 14
err # 15 68'FORTH  Talbot Microsystems       Ver 1.0  JAN 1980
err # 16 ( ERROR MESSAGES )
err # 17 COMPILATION ONLY,  USE IN DEFINITION
err # 18 EXECUTION ONLY
err # 19 CONDITIONALS NOT PAIRED
err # 20 DEFINITION NOT FINISHED
err # 21 IN PROTECTED DICTIONARY
err # 22 USE ONLY WHEN LOADING
err # 23 OFF CURRENT EDITING SCREEN
err # 24 DECLARE VOCABULARY
err # 25
err # 26
err # 27
err # 28
err # 29
err # 30
 OK
```

APPENDIX E

VERTICHIRP CONTROLLER SOFTWARE

PROGRAM GLOSSARY

G. P. EVANS

ANTARCTIC RESEARCH - RHODES UNIVERSITY

DECEMBER 1984

APPENDIX E

VERTICHIRP CONTROLLER SOFTWARE - PROGRAM GLOSSARY

Contents — Page Number

Screen No.

APPENDIX E

VERTICHIRP CONTROLLER SOFTWARE - PROGRAMME GLOSSARY

Introduction

This glossary is designed to be used in conjunction with the programme listings in the Vertichirp Controller Software - Operators Manual.

The following symbols are used in describing numbers on the FORTH parameter stack.

Symbol	Meaning
addr	16 bit memory address
b	8 bit byte , high 8 bits zero
c	7 bit Ascii character , high 9 bits zero
d	32 bit signed double number , MSB on top of stack
f	boolean flag , 0 = false , non-zero = true
ff	boolean false flag = 0
tf	boolean true flag = non-zero
n	16 bit signed integer number
u	16 bit unsigned integer number

The first line of each glossary entry gives the FORTH word being described followed by a symbolic description of the action of the word on the parameter stack. The symbols indicate the order in which the input parameters have been placed on the stack. Three dashes "---" indicate the execution point and any parameters left on the stack are then listed. The top of the stack (most accessible parameter) is to the right in this notation.

Note :- Unless otherwise specified named stack numbers are all 16 bit signed integers (n). Named stack integers are enclosed in brackets if more than 1 word long. E.g. (basic rate code) --- (basic rate)

Screens 1 to 9   Reserved Memory

These screens allocate memory for flags and storage using the FORTH word CONSTANT. Subsequent use of the name given pushes the address specified as the constant onto the stack allowing variables to be fetched or stored from that location.

The FORTH word VARIABLE was not used as it allocates named variable storage within the dictionary making programmme EPROMming impossible.

Screen 10   VDU/Printer Control

Note :- The software was designed to support either a Heath VDU using cursor addressing or any other VDU or printer. Unless otherwise specified the system Heath VDU will be assumed to be the I/O device.

STATUSOFF	---	Flag STAT cleared , controller status not displayed.
STATUS	---	Flag STAT set , controller status displayed.
S@	--- f	Fetch controller status flag from STAT.
PRNT	---	Flag VDU? cleared , cursor addressing not used.
VDU	---	Flag VDU? set , use Heath VDU cursor addressing.
V@	--- f	Fetch VDU flag from VDU?
NEWCR	---	New Carriage Return. Defined to disallow break-in when listing to Heath VDU , allow break-in when listing to printer.

CR	---	Carriage Return. Redefinition using NEWCR.
?TERM	--- f	Flag TFTN defines operation of ?TERM. TFTN = 1 ?TERM executes ?TERMINAL to obtain a flag , true if a key has been pressed. Allows user initiated functions to be terminated from the keyboard whereas program initiated functions must continue to completion. (Timing Shifts)
ESC	---	Outputs escape character.
BELL	---	Rings I/O device bell.
CLRVDU	---	Clear VDU screen if VDU? = 1 else CR.
SAVEXY	---	VDU saves cursor posn. if VDU? = 1
GOXY	---	VDU cursor goes to previously saved posn. if VDU? = 1

Screen 11 VDU/Printer Control

CURPOS	--- (line no.) (column no.)	VDU sends cursor posn. if VDU? = 1 else 0 0 left on stack. Line no. range 0 to 23 , Column no. range 0 to 75
GTOXY	(line no.) (column no.) ---	Moves cursor to posn. specified.
GOTOXY	(line no.) (column no.) ---	Cursor to posn. specified if VDU? = 1 else CR

XYONLY	(line no.) (column no.) ---	Cursor to posn. specified if VDU? =1
CLRLI	(line no.) ---	Clears VDU line specified.
CLRLN	(line no.) ---	Clears specified line if VDU? = 1 else CR
CLRLIS	(start line no.) (end line no.) ---	Clears VDU from start line no. to end line no.
CLRLNS	(start line no.) (end line no.) ---	Clears VDU from start line no. to end line no. if VDU? = 1 else CR
CEOL	---	Clear to end of current line if VDU? = 1
CLRSTAT	---	Clear status area , lines 0 to 6 , if VDU? = 1 else CR
CLRDIS	---	Clear display area , lines 8 to 18 , if VDU? = 1 else CR
CLRD+1	---	Clear display area lines 9 to 18 if VDU? = 1 else CR
CLRIN	---	Clear input area , lines 21 to 23 , if VDU? = 1 else CR

Screen_12 VDU/Printer_Control

HOME	---	Cursor to input area , line 20 , if STAT = VDU? = 1 else CR
H+1	---	Cursor to line 21 if STAT = VDU? = 1 else CR

DSHLN	n --- 80 dashes written to line n if VDU? = 1
PTRCR	--- Printer carriage return. CR output if VDU? = 0
->	n --- Type n spaces. Short form of SPACES
TAB	n --- Move to column n on current line
1SEC	--- Loops for approx 1 second
B1S	--- Ring bell , wait 1 s
OK?	f --- f If f = 0 ring bell wait 1 s
W?	--- "What ?"
W?B	--- "What ?" , bell , wait 1 s

Screen 13 Number and Word Input

INPT

--- ff not numeric input

--- d tf numeric input

QUERY gets either 80 characters or up to CR , stores them at the addr. stored in TIB.

20_{16} WORD gets string from TIB to HERE (dictionary buffer) WORD leaves char count in first byte then chars ending with 2 or more spaces. The loop fetches each char in turn and if not a number or " . " the loop is exited leaving 0 on the stack else it loops through all char then converts the string to d using NUMBER leaving d tf on the stack.

This word was written because the standard FORTH number conversion executes QUIT if conversion fails allowing the program to continue. INPT signals conversion failure with ff only to allow re-prompting for data.

BLNK	--- IN spaces output if VDU? = 1. Used to clear non-numeric input data off screen.
INPUT	--- d Loops until true flag put on stack by INPT. Used for inputting positive n and d only.
INWORD	--- pfa Loops until a valid dictionary word is input leaving the word's parameter field address on the stack
EXEC	pfa ff --- ff pfa tf --- tf Execute word designated by pfa if true flag on stack. Note: Execution of certain words leaves data on the stack.
YORN	--- f "(Y or N)?" Loops , Y = tf , N = ff
KEYCR	--- c Accepts char c followed by CR

Screen_14 MPN_Calculator_Control

MPN ---

Initializes MPN calculator board in port 0

Hex_address	Hex_Data	Comments
E000	7F	Initialize A side
E001	36	High hold POS ready
E002	00	Initialize B side
E003	34	Negative R/W

Note :- E002 is read to clear R/W flag.

MPNDP ---

Decimal point

MPNEE ---

Enter exponent

MPNCS ---

Change sign of MPN x register

MPNXEM ---

Exchange MPN x register and memory.

MPNMS ---

Memory store , x register stored in calculator memory.

MPNMR ---

Memory recall , memory to calculator x register

ENTER ---

Terminate number entry

MPNROL ---

Rotate MPN registers. x to y , y to z , z to t , t to x

MPNECLR ---

Reset calculator

MPNXEY	--- Exchange MPN x and y registers
MPNEX	--- e^x
MPNLN	--- $\ln x$
MPN1/X	--- 1/x i.e. reciprocal
MPNYX	--- y^x
MPN+	--- Add registers , y + x
MPN-	--- Subtract registers , y - x
MPN*	--- Multiply registers , yx
MPN/	--- Divide registers , y/x
ANSWER	--- Uses primitive ANS to transfer calculator x register to computer memory at ANSEXP. If calculator error flag is set an error is reported and the calculator error flag cleared.

Screen_15 Port_Initialization

MPS	--- Initialize serial port. $03 = master reset then set for 8 bits plus 1 stop bit , $11.

STORE n1 n2 ---

n1 to A side MP-LA-1. Data to controller data bus.

n2 to B side MP-LA-1. Data to Strobe Decode ,PC6.

Use : (data) (strobe) STORE

MPLA1 ---

Initializes parallel interface 1 (MP-LA-1) in port 2.

Addr.	Reg.	Data	Description
E008	PRA	FF	A = outputs,data to controller data bus
E009	CRA	07	Tcell int., CA1 positive edge.
E00A	PRB	FF	B = outputs,data to Strobe Decode,PC6
E00B	CRB	2D	Clock int., CB1 negative edge. CB2 = write strobe with E restore.

MPLA2 ---

Initializes parallel interface 2 (MP-LA-2) in port 3.

Addr.	Reg.	Data	Description
E00C	PRA	00	A = inputs , clock data
E00D	CRA	3C	Clock data valid flag , CA1 neg edge.
E00E	PRB	FF	B = outputs . Data to DCS.
E00F	CRB	2C	DCS flag , CB1 negative edge. CB2 = write strobe with E restore.

MPLA3 ---

Initializes parallel interface 3 (MP-LA-3) in port 7.

Addr.	Reg.	Data	Description
E01C	PRA	FF	A side = all outputs
E01D	CRA	04	CA1,CA2 not defined
E01E	PRB	FF	B side = all outputs
E01F	CRB	04	CB1,CB2 not defined

PORTS ---

Initializes MPN , MPS , MPLA1 , MPLA2 and MPLA3.

A1OFF ---

Clear parallel port MP-LA-1

A2OFF ---

Clear parallel port MP-LA-2

A3OFF ---

Clear parallel port MP-LA-3

RESET ---

Clear all parallel ports , initialise serial port.

Screen 16 Synthesizer Control - Input MHz , kHz or Hz.

SETDPL3 n --- $(n.10^{3-DPL})$

Sets the number of digits after the decimal point to 3.

DPL = no. of digits input after decimal point.

CONSTORE r q --- f

r (remainder) and q (quotient) are the results of a /MOD or M/ operation. CONSTORE checks to see that q is positive , if so q is divided by 4 to produce 5 ASCII numbers that are stored at F2MSB to F2MSB + 4. The remainder of this division is multiplied by 1000 and added to r and the result is divided by 4 to produce 3 ASCII numbers stored at F2MSB + 5 , F@MSB + 6 and F2MSB + 7. If CONSTORE is executed successfully a true flag is left on the stack else a false flag. The number stored at F2MSB is called F2.

MHZ d ---

Converts d to d/4 (freq/4 for synthesizer programming) which is stored at F2MSB.

	d can be entered in the following forms:- xx. , xx.x , xx.xx , xx.xxx or xx.xxxxxx
DRQ	d --- r q Converts double no. d = q.r to two single numbers , a 3 digit remainder r and a 3 digit quotient q
KHZ	d --- Converts d to d/4 (freq/4 for synthesizer programming) which is stored at F2MSB. d can be entered in the following forms:- xx. , xx.x , xx.xx or xx.xxx
HZ	n --- f If n is less than 3999 it is converted to n/4 and stored at F2MSB and a true flag left on the stack else a false flag.
Screen 17	Synthesizer Control - Print MHz/4 , or MHz. Freq to MPN.
PRINTBUF	addr --- Prints 8 ASCII digits from addr (MHz/4)
PRINTBUF1	--- Prints F1 from F1MSB , freq/4
PRINTBUF2	--- Prints F2 from F2MSB , freq/4
TO#	(LSB addr) n1 --- (sum = n2) Converts n1 BCD digits from (LSB addr) into a single number n2 on the FORTH stack. n2 must be less than 32000.
PRINTF	(MSB addr) --- 8 BCD numbers from (MSB addr) are converted to a double number on the stack. This is multiplied by 4 and printed in the form:- xx.xxxxxx MHz

PRINTF1	---	Prints F1 x 4 MHz from F1MSB
PRINTF2	---	Prints F2 x 4 MHz from F2MSB
DENT	(MSB addr) n ---	Digit Enter. n digits from (MSB addr) sent to MPN calculator.
FTOMPN	(MSB addr) ---	Frequency to MPN - 8 digits from (MSB addr) to calculator.
FENTER	(MSB addr) ---	Frequency Enter - 8 digits from (MSB addr) to MPN and entered
F1TOMPN	---	F1 to MPN
F2TOMPN	---	F2 to MPN
F1TOMEM	addr ---	8 BCD digits from F1MSB to addr
F2TOMEM	addr ---	8 BCD digits from F2MSB to addr
GETF1	---	8 BCD digits from F1MSB to dictionary at HERE. DP advanced 8.
GETF2	---	8 BCD digits from F2MSB to dictionary at HERE. DP advsnced 8.
TOF1	addr ---	Moves 8 bcd digits from addr to F1MSB
TOF2	addr ---	Moves 8 BCD digits from addr to F2MSB

Screen_18 Synthesizer_Control-Frequency_to_Synthesizer._ANSEXP_to_memory

LZERO	exp --- (exp + 1) (no. of leading zeros)	Generates leading zeros depending on the value of the exponent Converts MPN scientific mode answer to an 8 digit number. exp = 0 , 7 leading zeros exp = 7 , 0 leading zeros
XFR	(exp + 1) (no. of leading zeros) ---	8-(no. of leading zeros) digits transferred from ANS# to F1MSB
FORMAT	---	Check that exp less than 8. If less uses LZERO and XFR to convert and transfer MPN answer to F1MSB else error message.
NEWFREQ	---	Executes ANSWER then FORMAT
PACK	addr --- n	2 BCD bytes form addr and (addr + 1) packed into n
TOLATCH	addr ---	8 bytes from addr , packed and output to synth. latches , PC1
LOADFREQ	---	Latched data loaded into synthesizer counters
FREQOUT	---	F1 sent to synth latches and counters on PC1.
SETFREQ	---	F2 sent to synth latches and counters on PC1.
ATOMEM	addr ---	Answer to memory. Executes ANSWER then moves 12 bytes from ANSEXP (exp , signs and mantissa) to addr.

MTOMPN addr ---

Memory to MPN. Data stored using ATOMEM to MPN. Not entered.

Screen 19 Hardware Control - Control Register A

STROBE0 ---

Clears sweep f/f , advance/retard f/f and film drive f/f

STROBE1 ---

Clears sweep flip-flop.

STROBE2 ---

Clears advance/retard flip-flop

Constants:-

Name	Hex Code (sets control register A)
OFF	A0
TEST1	F1
TEST2	F3
DOPPLER	7C
OBLIQUETX	9F
OBLIQUERX	EF
VERTICAL	FF

SETA n ---

n is entered using the above names or as a number. A copy is kept at CONTROLA and n is output to the hardware control register A on PC3. This data determines the type of recording and does not change during recording execution.

A@ --- n

Fetch contents of CONTROLA

A? n ---

n = named code print name else "Decimal code = n"

Screen_20 Hardware_Control_-_Control_Register_B

SETB	n --- n copied to CONTROLB and output to control register b on PC3 CONTROLB data may change from cell to cell.
CLEARB	--- Clear control register B
B@	--- n Contents of CONTROLB fetched to stack
ORSETB	n1 n2 --- n1 and n2 logically ORed. Result sent to control register B This allows individual bits to be set.
ANDSETB	n1 n2 --- n1 and n2 logically ANDed. Result sent to control register B This allows individual bits to be cleared.

FILMON	--- b0 = 1 enables film dr	FILMOFF	--- b0 = 0 disables film drive
AGCEXT	--- b1 =1 ext AGC clock	AGCINT	--- b1 = 0 int AGC clock
TXON	--- b2 = 1 Tx enabled	TXOFF	--- b2 = 0 Tx disabled
B3ON	--- b3 = 1 fast film drive	B3OFF	--- b3 = 0 fast drive disabled
B4ON	--- b4 = 1 spare control	B4OFF	--- b4 = 0 spare control
B5ON	--- b5 = 1 spare	B5OFF	--- b5 = 0 spare

B6ON	--- b6 = 1 spare	B6OFF	--- b6 = 0 spare
B7ON	--- b7 = 1 spare	B7OFF	--- b7 = 0 spare
BLIST	--- Heading for control reg B status		
B?	--- Fetch and print control register B data		

Screen_21 Hardware_Control_-_Basic_Rate_and_Doppler

TABLE	--- Used to generate a lookup table.	
RATE	Uses TABLE to generate a table of basic rates. Data are listed in reverse order with the last no. = no. elements in list	
RATE?	(basic rate code) --- (basic rate , kHz/s) Code required by PC4 hardware converted to basic rate in kHz/s and printed.	
KHZ/SEC	(basic rate , kHz/s) --- ff (basic rate , kHz/s) --- (basic rate code) tf	Valid rate Invalid rate
SETRATE	(basic rate code) f --- Set hardware if f = 1 Use:- 50 KHZ/SEC SETRATE	
DOPR	(Doppler code = n) --- 2^n Converts n to 2^n for n not equal to zero. For n = 0 returns 0. Converts code for setting stationary Doppler offset frequency , PC9 , to an offset frequency in Hz.	
DOP	(offset freq = n) --- ff (offset freq = n) --- (Doppler code) tf n = offset freq in Hz for stationary Doppler recordings.	

n out of range ff only else converted to Doppler code and tf.

DOP. (Doppler code) ---

Converts Doppler code to an offset freq in Hz and prints it.

DOP? (Doppler code) ---

Executes DOP. "Hz"

SETDOP (Doppler code) f ---

If true flag L.O. offset synthesizer , PC9 , set to required freq in stationary Doppler mode.

Use :- 4 DOP SETDOP Sets stationary Doppler freq to 4 Hz

Screen 22 Hardware Control - Tcell and M-sequence

The constants listed below are named codes used for setting up both cell and M-sequence periods.

Constants :-

Name	Hex Code	Name	Hex Code
1/32	00	4/	07
1/16	01	8/	08
1/8	02	16/	09
1/4	03	32/	0A
1/2	04	64/	0B
1/	05		

TCELL (Tcell code) --- Invalid

(Tcell code) --- (Tcell code) Valid

Checks if (Tcell code) is greater than or equal to 0 , less than or equal to 7. Invalid,prints error message, clears stack

CODE# code --- time f

Converts code to time in seconds , flag false for times equal to or greater than 1 s otherwise true.

CELL?	code --- Uses CODE# , prints time/ for times greater than 1s else prints 1/time
TCELL?	code --- Uses CELL? and prints "sec" after time.
SETCELL	code --- Set Tcell pulse generator , PC4b , to give selected period. Use :- 1/2 TCELL SETCELL Sets cell period to 1/2 s
MSQNCE :-	Uses TABLE to set up a table of M-sequence period codes.
MSEQ	n --- Invalid n --- code Valid n checked , if valid uses MSQNCE to convert it to code else error message printed.
MSEQ?	code --- Uses lookup table to find which element corresponds to code then uses TCELL? to convert the entry number to a time period
SETMSEQ	code f --- Set M-sequence gen on PC4c to the period determined by code if the flag is true. Use :- 1/2 MSEQ SETMSEQ Sets M-sequence period to 1/2 s
Screen_23	Hardware_Control_-_FFT_Sample_Rate_and_Antenna_Switch
FFTR	code --- (8 x 2^n) Converts FFT rate code to fequency in Hz
FFT	n --- ff Invalid n --- code tf Valid If n greater than or equal to 8 , less than or equal to 1024 it is converted to an FFT sample rate code with true flag

FFT.	code --- code converted to frequency and printed
FFT?	code --- FFT. followed by "Hz"
SETFFT	code f --- FFT Sample rate , PC7 , set with code if true flag Use :- 1024 FFT SETFFT Sets FFT sample rate to 1024 Hz.
CHKA	n --- f True flag if n greater than or equal to 1 less than or equal to 8
ANTENNA	n1 n2 --- ff n1 n2 --- code tf n1 = Rx1 antenna , n2 = Rx2 antenna. code obtained by shifting n2 to upper 4 bytes (n2 x 16) logically ORing with n1 and adding 77_{16}
ANTS?	code --- (Rx1 ant) (RX2 ant) Converts antenna select code to Rx1 ant no. and Rx2 ant no.
ANTENNA?	code --- Converts antenna select code to two numbers and prints them.

Screen_24 Hardware_Control_-_Film_Speed_,_Windowing_and_Syncs

Film_Speed

Input frequency = 500 Hz i.e. 10 mm/s = 600 mm/min

Two hardware divisions give 60 film speeds from 300 mm/min to 2.5 mm/min

FILMSPEED?	code --- speed Film speed code converted to speed in range 2.5 to 300 mm/min

MM/MIN speed --- ff

speed --- code tf

Speed out of range , ff only. Speed valid , converted to code that is closest to the required speed plus tf.

SPEED? code ---

Code converted to a speed and printed (xxx mm/min)

SETSPEED code f ---

If true flag film speed is set using code.

Use :- 5 MM/MIN SETSPEED

Windowing

Equation (3.9) gives the height offset as $h_0 = cf_0/2k_B$ km. when a fixed offset f_0 is subtracted from the difference frequency. The offset f_0 is therefore given by :- $f_0 = h_0k_B/(c/2)$

For h_0 in km and k_B in kHz f_0 in Hz is given by :-

$$f_0 = h_0k_B/150 \text{ Hz}$$

The local oscillator offset frequency synthesizer was designed with f_{0min} = 256 Hz (see Chapter 4 , Windowing and Stationary Doppler) with higher offset frequencies being multiples of this.

Number of 256 Hz steps = $f_0/256 = h_0k_B/(150)(256)$

If 0 km offset is required the code for no offset (00) is output.

KMSTEP k_B h_0 --- ff Invalid h_0

k_B h_0 --- n tf Valid h_0

The offset height in km , if less than 951 , is multiplied by the basic rate k_B and the result divided by 150 and 256 to give the number of 256 Hz steps. This number is converted to a two digit packed BCD number n and a true flag . If the offset height is greater than 951 km error message output and ff.

SETWIND n f ---

If true flag the L.O. offset synthesizer , PC9 , is set to the required frequency in the windowing mode.

E.g. k_B h_0 KMSTEP SETWIND

RXPULSE ---

Clocks dual Rx AGC if set to EXT mode in control register B.

RXSYNC ---

Synchronizes dual Rx output signal phases.

SWEEP ---

Sync receivers , enable sweep start flip-flop. A sweep (recording) begins on the next falling edge of the 1 s clock.

DOPTIME ---

Doppler Time mark. Sends a pulse to PC2 to produce a time mark on film for stationary Doppler recordings.

Screen 25 Hardware Control - Advance/Retard Control

The advance/retard software was designed to provide the following functions

1. Set timing position counter to zero
2. Advance or retard timing by a given amount with the capability of halting the shift at any time.
3. Display resulting position on VDU (Note :- This is not an absolute position as system timing drifts relative to a time signal e.g. ZUO)

   Display takes the form :-

   Timing Position : 0010.00 ms Advanced ---> 0005.00 ms Retarded
4. "SEARCH" mode used for locating the oblique signal. Prompts only for direction and rate of shift. Shift stopped when required by operator. Right hand count continuously updated.
5. On exiting from SEARCH mode the command LAST allows return to timing posn

prior to SEARCH.

6. Execution of a timing slip by controller programme , operator intervention disallowed.

MSEC	d --- microseconds milliseconds Redefinition of DRQ . Converts double number d = shift in milliseconds to two 3 digit single numbers. E.g. 5.67 --- 67 5
+COUNT	d --- Adds slipped amount d to SHIFT which contains timing position
SLIP	n --- n n = required slip . Sent to advance/retard board PC5 together with the shift enable strobe hex 1A. n left on stack.
WAITCOUNT	wait count --- wait wait determines duration of a loop while the hardware shifts the timing at the required rate. Hardware timing changes in increments of 10 microseconds. Software sends 100 to hardware to produce timing changes in 1 ms increments. The sign of wait is + for advance , - for retard and is used to determine whether to add or subtract count to or from SHIFT.
POSN	--- Prints contents of SHIFT as xxx.xx ms Advanced(or Retarded)
ENPOS	--- End Position. STAT = 0 , does nothing STAT = 1 , VDU = 0 Prints end position VDU = 1 Prints end posn in correct screen area
CLRPOSN	--- Clear Position. SHIFT set to 0 , posn printed

MSPOSN	--- VDU? = 1 , ENPOS prints position each ms slipped
INPOS	--- Initial Position printed.
DOSLIP	microseconds milliseconds wait --- Do Timing Slip. Prints initial and final positions if in printer mode or timing position each ms slipped if in VDU mode Flag TFTN checked , 0 allows keyboard termination of shift.

Screen 26 Hardware Control - Advance Retard Control

SETAR	microseconds milliseconds wait control --- SRCH = 0 , not searching , current posn stored as last posn at LPOSN . control is the number sent to the advance retard board , PC5 , control data latch and determines shift direction and rate. DOSLIP uses remaining numbers.
A@1	microseconds milliseconds --- Advance at 1 ms/s by shift numbers left by MSEC Use :- 10.5 MSEC A@1 Advances 10.5 ms at 1 ms/s
A@10	microseconds milliseconds --- Advance at 10 ms/s by shift numbers left by MSEC Use :- 5.5 MSEC A@10
A@20	microseconds milliseconds --- Advance at 20 ms/s by shift numbers left by MSEC Note :- Each millisecond of shift is done by the hardware at a rate of 100 ms/s but software overhead results in a net shift rate of 20 ms/s hence the command A@20. Use :- 8.5 MSEC A@20

R@1 microseconds milliseconds ---

Retard at 1 ms/s by shift numbers left by MSEC

Use :- 15.0 MSEC R@1

R@10 microseconds milliseconds ---

Retard at 10 ms/s by shift numbers left by MSEC

Use :- 20.5 MSEC R@10

R@20 microseconds milliseconds ---

Retard at 20 ms/s by shift numbers left by MSEC

The hardware shifts at 100 ms/s but net rate is only 20 ms/s

Use :- 12.5 MSEC R@20

EXIT ---

Exit SEARCH mode.

LAST ---

Return timing to previously stored position (LPOSN) at 20 ms/s

In the SEARCH mode last position not updated allowing timing to be reset to the position prior to SEARCH by typing LAST

Command	Display
10. MSEC A@1	Timing Position :- 10.00 ms Advanced ---> 20.50 ms Advanced
LAST	Timing Position :- 20.50 ms Advanced ---> 10.00 ms Advanced
SEARCH	Timing Position :- 10.00 ms Advanced ---> xxx.xx ms Retarded
EXIT	Timing Position :- 10.00 ms Advanced ---> xxx.xx ms Retarded
LAST	Timing POsition :- xxx.xx ms Retarded ---> 10.00 ms Advanced

Screen 27 Hardware Control - Advance Retard Search

VAL ---

Clears lines 21 to 23 and prints "Valid Data :-"

NPR pfa ---

Name Print. Prints name of word given parameter field address

DPR ---

Direction Rate Prompt

SDRP ---

Search Direction and Rate Prompt

SEARCH ---

Current position stored at LPOSN , prompts with SDRP. Typing one of A@1 , A@10 , A@20 , R@1 , R@10 , R@20 followed by CR shifts in selected direction at rate given. Pressing space bar stops shift. Right hand digits updated as timing posn changes EXIT exits from SEARCH mode.

Screen_28 Datatype_Handling

The <BUILDS ..... DOES> structure was used to define FORTH words for data storage. Each data set is identified by the first single number stored as data. The upper byte is the data "type" and the lower byte the data "subtype". FORTH word data entries have the following form :-

Address	Bytes	Contents
nfa = name field address	1	number of letters in assigned name
nfa + 1	x	1 byte per character of assigned name
lfa = link field address	2	Pointer to previous word
cfa = code field address	2	Addr of DODOES = 0700 hex
pfa = parameter field addr	2	Addr of DOES part of structure
pfa + 2	2	Type and subtype of following data

Note :-

Type , subtype and data are "built" into the word being defined and in use the address of the does part points to the word DROP so that attempted execution of the word containing data results in no apparent action. Attempted execution puts pfa on stack , this is dropped by DROP

NAME?	--- f Saves IN. Checks TIB (terminal input buffer) for a word for subsequent use. If no word in TIB prints "What ?". If there is a word but it already exists prints word then " - Exists!". In both cases the saved value of in is dropped and the new value saved to jump over "no word" or existing word in TIB. A false flag is left on the stack. If there is a unique word the saved value of IN is stored at IN so that the unique word can be used by <BUILDS. A true flag is left on the stack.
TYPE?	pfa --- pfa type Returns pfa and type (upper 8 bits) of type/subtype at pfa+2
SUBTYPE	pfa --- pfa subtype Returns pfa and subtype (lower 8 bits) of type/subtype at pfa+2
GETNAME	--- ff no name in TIB or undefined name --- pfa tf pfa of name in TIB left on stack
RPFA	pfa n -- pfa (pfa+n) Relative to pfa. n is offset of data relative to pfa.
RPFA@	pfa n --- pfa data Relative to pfa fetch. Fetches data from address pfa+n
RPFA!	pfa n1 n2 --- pfa Relative to pfa store. Stores n1 at address pfa+n2
GENCR	--- Generate carriage return 10 spaces from end of line.
LID.	nfa --- Print word name from nfa , left justified in 31 column field
LNPR	pfa --- Print word name from pfa , left justified in 31 column field

Screen 29 Sounding Parameters

BR=	--- "Basic Rate = "
RP	--- Rate Prompt. Prints "Valid Data :-" followed by 25 40 50 80 100 200 400 250 500 800 1000 2000 4000 kHz/s
HR=	--- "Height Range = " (for windowing)
HZKM	k_B f_0 --- h_0 Hertz per km. Uses equation $h_0 = (150f_0)/k_B$
HMAX	k_B --- h_{max} Height maximum for given basic rate. From section 3.4 :- $h'_{max} = c(f_T - f_R)_{max}/2k_B$ $(f_T - f_R)_{max}$ = Rx bandwidth = 500 Hz $h'_{max} = 75000/k_B$ km , k_B in kHz.
RANGE	k_B f_0 --- Height Range. Calculates and prints the virtual height range for a given basic rate k_B and local oscillator offset f_0 Printed as :- "Height Range = xxx to xxx km".
GETRATE	--- (basic rate code) Prompts for basic rate , checks input , if valid leaves basic rate code on stack else prompts until valid.
NMH	--- "New Minimum Height Range = "
HTP	(basic rate) --- Height Prompt. Prompts "Valid Data :- 0 to 950 Km" then "Height Range = 0 to xxx km" (xxx depends on basic rate) "New Minimum Height Range = "

GETWIND (basic rate) --- freq step (control = 8)

Uses HTP to get new minimum height range which is converted to the number of 256 Hz steps using KMSTEP. Loops until valid data obtained then uses RANGE to print new height range.

8 = control code for windowing

step = no. of 256 Hz steps stored as packed BCD

freq = (no. of steps) x 256 Hz (single number , binary)

Screen_30 Sounding_Parameters

DOP= ---

"Doppler Freq = "

DOPP ---

Doppler Prompt.

"Valid data :- 0 2 4 8 16 32 64 Hz"

GETDOP --- freq 1 (Doppler code)

Prompts for Doppler offset frequency , if valid converts to code else loops to prompt again. Stack data :-

Doppler code = code to set hardware on PC9 to required freq.

1 = sets f_{out} from rate multiplier to 256 Hz

freq = Doppler offset frequency in Hz (single no.)

P n1 n2 ---

Tcell and M-seq Prompt. Expects two numbers n1 and n2 between 1 and 12 on the stack. Prints corresponding time period FORTH words using CELL? as valid data for Tcell or M-seq prompts.

CKT --- f

Check if Tcell period in correct range , if so tf.

CKD --- f

Check if Doppler Tcell period in correct range , if so tf.

CKM	--- f Check if M-sequence period in correct range , if so tf.
TC=	--- "Tcell = "
GETTCELL	--- (Tcell code) Get Tcell period for ionogram. Prompts and loops until valid word input. Word executed to leave code on stack.
GETDCELL	--- (stationary Doppler Tcell code) Get Tcell period for stationary Doppler. Prompts and loops until valid word input. Word executed to leave code.
DOP??	--- f "Stationary Doppler (Y or N)?" Y = ff , N = tf.
MS=	--- "M-seq = "
GETMSEQ	--- (M-seq code) Get M-sequence period . Prompts and loops until valid word input. Executes word to leave M-seq code on stack.

<u>Screen 31</u> <u>Sounding Parameters</u>

FT=	--- "FFT Rate = "
FFTP	--- FFT Prompt. "Valid Data :- 8 16 32 64 128 256 512 1024 Hz" "FFT Rate = "
GETFFT	--- (FFT rate code) Prompts for FFT rate , looping until valid code obtained.

FS=	--- "Film Speed = "
FLMP	--- Film speed Prompt. "Valid Data :- Between 2 and 150 mm/min"
GETSPEED	--- (film speed code) Prompts for film speed , looping until valid data input. Leaves code of closest speed on stack.
AG=	--- "RX AGC = "
RXG	--- "Fixed Receiver Gain (Y or N)? " Y = ff , N = tf
G	ff --- ff Invalid tf --- n Prompts "Multiples of Tcell = " and inputs n
TM=	--- "Time Marks = "
TMKS	--- n "Time Marks = (Y or N)? " then uses G to get number n.

Screen_32 Sounding_Parameters

#CELLS	--- n "Number of cells = " then inputs n. If 0 input 1 put on stack
OFPRM	--- "Valid Data :- xxxx Hz , xxx.xxx kHz or xx.xxx MHz " "Offset from Origin (number only) = " Origin = sounding start frequency
CHKWD	pfa --- f Check Word. tf if pfa is that of HZ , KHZ or MHZ else ff.

GETOFF ---

Get cell offset frequency. Requests numbers only , car ret , then units , HZ , KHZ or MHZ. Loops until valid no. in F2MSB.

FILMDR --- f

"Film Drive (Y or N)? " Y = tf , N = ff

ANTP ---

"Valid Data :- 1 2 3 4 5 6 7 8 "

GETANT --- (antenna code)

Get Antennae connections. Prompt for Rx1 ant , Rx2 ant then input both and convert to code , looping until data valid.

<u>Screen 33</u> **<u>Build Sounding</u>**

TYPE1 subtype --- (256 + subtype)

Sets datatype = upper byte to 1 by adding 256 to subtype.

SOUNDING ---

Input Sounding data. Use :- SOUNDING x where x is the user assigned data name given to the sounding. If x is unique SOUNDING prompts , gets and stores data as shown below.

Address	Data
pfa+2	Type (upper byte) = 1 , SOUNDING Subtype (lower byte) = 0 , for STATIONARY DOPPLER Subtype (lower byte) = 1 , for IONOGRAM
pfa+4	Basic rate (k_B) code
pfa+6	L.O. offset hardware control code
pfa+8	L.O offset freq as packed BCD for hardware
pfa+10	L.O. offset freq as FORTH single no. in Hz.
pfa+12	Tcell period (T_C) code

Address	Data
pfa+14	M-seq period code
pfa+16	FFT sample rate code
pfa+18	Film speed code
pfa+20	Rx AGC data. B reg = AGCINT before run , AGCEXT during run. Stat Doppler :- Data = 0 = fixed AGC Data = n = pulse per n Tcells Ionogram :- Data = 0 = fixed AGC :- Data = 1 = pulse per sounding
pfa+22	Time mark data. 0 = off , n = mark per n Tcells
pfa+24	Number of cells (n)
pfa+26	Cell 1 offset frequency relative to sounding start frequency , 8 bytes.
pfa+34	Cell 1 film drive enable or disable
pfa+36	Cell 1 antenna connection code
pfa+38	Cell 2 offset frequency ....etc , 12 bytes per cell for as many cells specified (n).

Note:- Attempted execution of data word x results in no apparent action.

SUB1. n ---

Prints subtype from number n

n = 0 "- (SOUNDING for STATIONARY DOPPLER)"

n = 1 "- (SOUNDING for IONOGRAM)"

Screen_34 Display_Sounding

BR. pfa --- pfa

Basic Rate Print from code if subtype = 1 = ionogram.

LO= ---

"L.O. Offset = "

LOHR. pfa --- pfa

Local Oscillator and Height Range Print from stored code.

TC. pfa --- pfa

Tcell period Print from code. Subtype used to differentiate between stationary Doppler Tcell and ionogram Tcell periods.

MS. pfa --- pfa

M-sequence period Print from stored code.

FT. pfa --- pfa

FFT sample rate Print from stored code.

Screen_35 Display_Sounding

FS. pfa --- pfa

Film Speed Print from stored code

AG. pfa --- pfa

AGC control data Print from stored code.

TM. pfa --- pfa

Time mark data Print from stored data.

NC. pfa --- pfa

Number of Cells Print from stored number.

Screen_36 Display_Sounding_-_Cell_Data

CNUM n1 --- n2

Converts n1 between 1 and 4 together with the contents of 4C# (four cell number) to n2 , the entry number.

VECC n1 n2 --- n3

Vector to current cell in sounding data

n1 = offset to cell 1 offset freq , film dr or antenna data

n2 = number between 1 and 4 in current cell block

n3 = offset relative to sounding pfa of current cell data

PCELL pfa ---

Print cell data. Clears cell display area , prints up to 4 sets of cell data in current cell block.

CELLDATA pfa --- pfa

Prints headings , calculates no. of sets of 4 cells (#4C)

VDU - prints up to 4 sets of cell data in display area

Printer - prints all cell data

Screen_37 Display_Sounding

DISS pfa --- pfa

Display Sounding. If type = 1 displays sounding data name , subtype and all the data else error message.

SDISP pfa ---

Same as DISS , drops pfa

SDISPLAY ---

Sounding Display. Use :- SDISPLAY x , x = sounding data name

SHOWS ---

Show Soundings. All sounding data names listed

Screen_38 Edit_Sounding_-_Prompts

BR --- 1
LO --- 2
TC --- 3
MSQ --- 4
FT --- 5
FS --- 6
AGC --- 7
TM --- 8
CELL --- 9

OF=	---	"Offset/Range = "
CL=	---	"Cells = "
ESP	---	Edit Sounding Prompt , uses above short forms e.g. BR
PARA	--- n	Parameter prompt using ESP , e.g. If BR typed n = 1

Screen_39 Edit_Sounding_Parameters

ELO	pfa --- pfa	Edit L.O. offset frequency / height range
EBR	pfa --- pfa	Edit Basic Rate if subtype = 1 = sounding for ionogram
ETC	pfa --- pfa	Edit Tcell period according to subtype
EMQ	pfa --- pfa	Edit M-sequence period
EFT	pfa --- pfa	Edit FFT sample rate
EFS	pfa --- pfa	Edit Film Speed
EAG	pfa --- pfa	Edit AGC data according to subtype

Screen_40 Edit_Sounding_Parameters

CP	pfa --- pfa	Cell Prompt. Lists all valid cell numbers

GC#	pfa --- pfa n Get Cell Number. Prompts , inputs no. (n) , loops until valid.
EC#	pfa --- pfa Edit Cell no. n freq offset,film drive and antennae selection
PVEC	pfa n --- pfa Parameter Vector. Vectors to edit word n and executes it.
EDITS	pfa --- pfa Edit Sounding. Prompts for parameter to edit then edits it.
Screen_41	Display_and_Edit_Sounding
PDES	--- Prompt , for sounding display and edit. N = Next 4 cells C = Change E = Exit P = Previous 4 cells
DES	pfa --- Display and Edit Sounding . Prompts using PDES. N - increments 4C# , MIN of 4C# and #4C , stored at 4C# P - decrements 4C# , MAX of 4C# and 1 , stored at 4C# C - executes EDITS E - Exit
SDISP	pfa --- Sounding Display. Displayed in edit mode on VDU.
SEDIT	--- Sounding Edit. Use :- SEDIT x , x = user asigned sounding data name
SDISPLAY	--- Sounding Display. Use :- SDISPLAY x ; x = sounding data name

SOUNDING	--- Redefinition of SOUNDING incorporating SDISP to display all sounding data once it has been entered.
<u>Screen_42</u>	<u>Set_Sounding</u>
IFR	--- "Invalid Forward Reference - "
SSET	pfa --- f Set Sounding. If datatype = 1 the data is used to set the hardware. The Tx is enabled and AGC# , TMK# , CELL# initialized
USSET	--- User Sounding Set. Use :- USSET x ; x = sounding data name
<u>Screen_43</u>	<u>Ionogram_Parameters</u>
TYPEP	--- f Type of recording Prompt. Choice of 7 named types or code. ff = ionogram , tf = experimental code
GETYPE	--- (control register A code) Gets Type of recording. Stack number used to set ctrl reg A
ID=	--- "Ionogram Duration = "
MORS	--- n Minutes or Seconds input routine. Loops until valid n input.
GETDUR	--- seconds minutes hours Get Duration of stationary Doppler ionograms.

Screen 44 Ionogram Parameters

F= ---

"Frequency = "

SF= ---

"Start Frequency = "

EF ---

"End Frequency = "

FP ---

Frequency Prompt. "Valid data :- Between 0.5 and 30.0 MHz

INF ---

Input Frequency to F2MSB , looping until valid data obtained

GETF ---

Gets Frequency. Prompts for then stores frequency in F2MSB

GETSF ---

Get Start Frequency. Prompt , input then store at F2MSB

GETEF ---

Get End Frequency. Prompt , input then store at F2MSB

ORA ---

"Overall Rate"

OR= ---

"Overall Rate = "

LIN ---

"Linear "

LN ---

"Log "

ORP --- f

Overall Rate Prompt.

"Valid data :- Linear = xxx kHz/s , Log = .001 -->.01 oct/s"

		Queries if Linear or Log. Log = ff = 0 , Linear = tf = 1
GETOR	--- (overall rate) (lin/log flag)	Get Overall Rate with lin/log flag.

Screen 45 Ionogram Parameters

SN=	---	"Sounding = "
VALA	---	"Valid Data :- As listed above"
GETSND	--- pfa subtype	Get Sounding. Lists names , inputs and leaves pfa and subtype
CHKSND	f --- f	Check Sounding. If tf prints "Incompatible Sounding" leaves ff
GETISND	--- pfa	Get Ionogram Sounding. Loops until compatible sounding input
GETDSND	--- pfa	Get stat Doppler Sounding. Loops until compatible sounding in.
TT=	---	"Data to Tape = "
TOMT?	--- f	"Store Data on Mag Tape (Y or N)? " , Y = 1 , N = 0
STATI?	--- f	"Stationary Ionogram (Y or N)? " , Y = 1 , N = 0

Screen 46 Build Ionogram

TYPE2	subtype --- (512 + subtype)	Sets datatype = upper byte to 2 by adding 512 to subtype.

IONOGRAM ---

Input Ionogram Data. Use :- IONOGRAM x where x is the user assigned data name given to the ionogram. If x is unique the ionogram parameter prompts are used to get and store data at the following addresses.

Address	Data
pfa+2	Type (upper byte) = 2 , IONOGRAM
	Subtype (lower byte) , Control register A data
pfa+4	Hours
pfa+6	Minutes Ionogram/Stat Doppler duration
pfa+8	seconds
pfa+10	Start frequency , 8 bytes
pfa+18	End frequency , 8 bytes
pfa+26	0 = log , 1 = linear
pfa+28	Overall rate.
	0 = stat Doppler , n kHz/s = normal ionogram
pfa+30	Sounding pfa
pfa+32	Mag tape flag. 0 = not to tape , 1 = to tape.

Note :- If subtype = 120 the recording is a stationary Doppler and start frequency = end frequency , lin/log = lin and overall rate = 0. The stat Doppler duration is prompted for and stored at pfa+4 , pfa+6 and pfa+8. Ionogram durations are calculated when the ionogram is set. This time is then written into the ionogram data word.

SUB2. n ---

Prints subtype from number n

e.g. If n = 120 " - IONOGRAM (DOPPLER)"

e.g. if n = 255 " - IONOGRAM (VERTICAL)"

.C n ---

Prints n right justified in a 2 column field then space:space

Screen_47 Display_Ionogram

IDR. pfa --- pfa

Ionogram Duration Print.

TT. pfa --- pfa

To Tape? data Print.

SF. pfa --- pfa

Start Frequency Print.

EF. pfa --- pfa

End Frequency Print.

OR. pfa --- pfa

Overall Rate Print

SN. pfa --- pfa

Sounding Name Print.

VEL. pfa --- pfa

maximum observable Velocity of reflection point Print.

Uses equation (4.16) in the form $u_{max} = f_{OD}c/2f$

f_{OD} = stat Doppler L.O. offset frequency

f = stat Doppler transmitter frequency

"Maximum Velocity = xxx m/s"

Screen_48 Display_Ionogram

DISI pfa --- pfa

Display Ionogram. Displays ionogram data name , type and subtype followed by the ionogram data.

IDISP pfa ---

Ionogram Display. Same as DISI but drops pfa.

IDISPLAY ---

Ionogram Display. Use :- IDISPLAY x , x = ionogram data name

SHOWI ---

Show Ionograms. All ionogram data names listed.

IONOGRAM ---

Redefinition of IONOGRAM that inputs then displays data.

Use :- IONOGRAM x , x = user assigned name

Screen_49 Edit_Ionogram_-_Prompts

DUR --- 1 DTT --- 2

SF --- 3 EF --- 4 ORT --- 5

SOUND --- 6

EIP ---

Edit Ionogram Prompt. E.g. "DUR = Ionogram Duration" etc.

PARI --- n

Parameters Ionogram. EIP to prompts for parameters to change.

No exit until valid input given , leaves n between 1 and 6

Screen_50 Edit_Ionogram_Parameters

EDR pfa --- pfa

Edit duration data.

ETT pfa --- pfa

Edit To mag Tape flag

DOPF ---

Doppler Frequency edit. New freq = start freq = end freq.

ESF	pfa --- pfa	Edit Start Frequency. If subtype = Doppler uses DOPF to edit start freq = end freq , recalculates maximum velocity.
EEF	pfa --- pfa	Edit End Frequency. If subtype = Doppler uses DOPF to edit start freq = end freq , recalculates maximum velocity.
EOR	pfa --- pfa	Edit Overall Rate if subtype is not stationary Doppler.
ESN	pfa --- pfa	Edit sounding Name. If ionogram subtype = Doppler get Doppler sounding else get ionogram sounding.
IVEC	n ---	Ionogram edit Vector. Execute ionogram edit word n.
EDITI	pfa --- pfa	Edit Ionogram. Prompts for parameter then edits it.

Screen_51	Display_and_Edit_Ionogram

PDEI	---	Prompt , Display and Edit Ionogram. "C = Change" "E = Exit"
DEI	pfa ---	Display and Edit Ionogram. Prompts as above for C or E. If C uses EDITI to edit ionogram , E exits edit mode.
IEDIT	---	Ionogram Edit. Use :- IEDIT x , x = ionogram data name

Screen_52 Ionogram_Calculations

STKF1	n ---	Stack to F1MSB. Single number n to BCD and stored at F1MSB.
STKMPN	n ---	Stack to MPN calculator via F1MSB.
ENTR	n ---	Enter. n via F1MSB to MPN and entered.
F1STK	--- n	F1 to Stack. Note :- n must be less than 32000
ANSR	--- n	Answer from MPN calculator put on stack.
CELMPN	(Tcell code) ---	Tcell to MPN. Tcell code converted to a number with true flag if period is less than 1 and number sent to MPN. If true flag MPN1/X is executed. Cell period therefore entered in MPN.
SEC/SND	---	Seconds per Sounding. Current Sounding pfa stored in CURS. This is used to obtain Tcell code and number of soundings in current sounding. These are sent to MPN as numbers and not code and are multiplied to give the sounding period. Equation (3.1) :- $T_S = nT_C$
ORTOM	(ionogram pfa) ---	Overall Rate to MPN.
LINOR	(ionogram pfa) ---	Linear Overall Rate. MPN used to multiply rate in kHz/s by 250 as all calcs done on 1/4 the actual Tx freq. (250 = 1000/4)

LNOR (ionogram pfa) ---

Log Overall Rate (oct/s). Entered log rate is multiplied by 1000 then stored. Calculator divides stored log rate by 1000 to get actual rate.

Screen_53 Ionogram_Setting

SETDOPL ---

Set stationary Doppler. SEC/SND used to calculate sounding period. Doppler ionogram duration sent to MPN and number of soundings are calculated and stored at #SOUND

FORTH_Stack	FORTH_Word	MPN_Calculator_Stacks			
-	-	x	y	z	t
-	SEC/SND	sec/snd = T_S	-	-	-
hrs	ENTR	hrs	hrs	T_S	-
3600	STKMPN	3600	hrs	T_S	-
-	MPN*	3600 x hrs = H s	T_S	-	-
min*60 = M s	STKMPN	M seconds	H seconds	T_S	-
-	MPN+	H + M seconds	T_S	-	-
S seconds	STKMPN	S seconds	H + M seconds	T_S	-
-	MPN+	H + M + S seconds	T_S	-	-
-	MPNXEY	T_S	H + M + S sec	-	-
-	MPN/	No. of soundings	-	-	-
No. soundings	ANSR	-	-	-	-

SETIONO (ionogram pfa) ---

Set Ionogram. If overall rate is linear calculations are based on equation (3.5) which gives total soundings in ionogram , m

$m = (f_E - f_I)/k_0 n T_C$

First calculated is $k_0 n T_C = k_0 T_S$ which gives the frequency change per sounding. This is stored at F/SND. Next m is calculated and stored at #SOUND. Finally the ionogram duration $T_I = mT_S$ is calculated and stored in current ionogram.

FORTH Stack	FORTH Word	MPN Calculator Stack			
-	-	x	y	z	t
addr of f_E	FENTER	f_E	f_E	-	-
addr of f_I	FTOMPN	f_I	f_E	-	-
If linear overall rate :-					
-	MPN-	$f_E - f_I$	-	-	-
-	SEC/SND	sec/snd = T_S	$f_E - f_I$	-	-
-	LINOR	k_0	T_S	$f_E - f_I$	-
-	MPN*	k_0T_S	$f_E - f_I$	-	-
-	NEWFREQ copies k_0T_S to F1MSB				
-	F/SND F1TOMEM transfers k_0T_S to F/SND				
-	-	k_0T_S	$f_E - f_I$	-	-
-	MPN/	$(f_E - f_I)/k_0T_S = m$	-	-	-
-	ANSR #SOUND ! stores no. of soundings m at #SOUND				
-	-	m	-	-	-
-	SEC/SND	T_S	m	-	-
-	MPN*	$mT_S = T_I$ seconds	-	-	-
T_I seconds	ANSR	T_I seconds	-	-	-

T_I seconds are converted to hours , minutes and seconds and are stored in the current ionogram at pfa+4 , pfa+6 and pfa+8 (ionogram duration).

If sweep rate is logarithmic the calculations are based on equation (3.7) which gives the total number of soundings in the ionogram as :-

$$m = \ln(f_E/f_I)/AnT_C$$

First calculated is $\ln(f_E/f_I)$ then $AnT_C = AT_S$.

e^{AT_S} is calculated and stored in memory at EATS. m is then calculated and stored at #SOUND. Finally the ionogram duration is calculated and stored within the current ionogram.

FORTH_Stack	FORTH_Word	MPN_Calculator_Stack			
-	-	x	y	z	t
addr of f_E	FENTER	f_E	f_E	-	-
addr of f_I	FTOMPN	f_I	f_E	-	-
-	MPN/	f_E/f_I	-	-	-
-	MPNLN	$\ln(f_E/f_I)$	-	-	-
-	SEC/SND	sec/snd = T_S	$\ln(f_E/f_I)$	-	-
-	LNOR	A	T_S	$\ln(f_E/f_I)$	-
-	MPN*	AT_S	$\ln(f_E/f_I)$	-	-
-	ENTER	AT_S	AT_S	$\ln(f_E/f_I)$	-
-	MPNEX	e^{AT_S}	AT_S	$\ln(f_E/f_I)$	-
-	EATS ATOMEM copies exponent , signs and mantissa from answer buffer to location EATS				
-	-	e^{AT_S}	AT_S	$\ln(f_E/f_I)$	-
-	MPNROL	AT_S	$\ln(f_E/f_I)$	-	e^{AT_S}
-	MPN/	$\ln(f_E/f_I)/AT_S = m$	-	-	e^{AT_S}
m	ANSR	m	-	-	e^{AT_S}
-	#SOUND ! stores no. of soundings m at #SOUND				
-	SEC/SND	T_S	m	-	e^{AT_S}
-	MPN*	$mT_S = T_I$	-	-	e^{AT_S}
T_I	ANSR	T_I	-	-	e^{AT_S}

T_I is converted to hours , minutes and seconds and is stored in the current ionogram at pfa+4 , pfa+6 and pfa+8 (ionogram duration).

STRCDAT ---

Store Cell Data of current cell of current sounding. Cell frequency in MPN is multiplied by 4 to get transmitter frequency which is stored at CFREQ (= cell frequency) and antenna code of same cell is converted to two ASCII characters Rx1 antenna no. stored at CANT , Rx2 ant no. stored at CANT+1

<u>Screen_54</u> <u>Ionogram_Setting</u>

ISET (ionogram pfa) --- f

The current ionogram pfa is stored at CURI. The sounding pfa , stored as ionogram data at ionogram pfa + 30 , is fetched , duplicated and stored at CURS. Duplicate is used by SSET (sounding set) which sets basic rate , cell period , M-sequence period , FFT sample rate , film speed , and L.O. offset frequency. Cell 1 film drive data and antenna selections are also set.

The dual Rx AGC is set to internal and the transmitter enable set to enabled

If the sounding is set successfully the ionogram subtype is used to set control register A. The ionogram start frequency is sent to the synthesizer counters on PC1 and also to the MPN x register from which it is stored in the calculators memory using FORTH word MPNMS . If subtype = Doppler SETDOPL is executed else the current ionogram overall rate is fetched from pfa+28. If the overall rate is non-zero SETIONO is executed else the ionogram is a stationary ionogram (zero overall rate therefore chirp at fixed frequency) and F/SND is set to zero and SETDOPL executed to calculate the sounding period and the number of soundings.

SOUND# contains the sounding number , this is initialized to 1.

The cell 1 offset frequency , stored at current sounding pfa + 26 is sent to MPN and added to the ionogram start frequency. The result = ionogram start frequency plus cell 1 offset frequency is sent to the synthesizer latches on PC1. STRCDAT converts and stores cell information for later transmission to

DCS. If all is executed successfully a true flag is left on the stack.

UISET ---

User Ionogram Set. Use :- UISET x , x = ionogram data name

Screen_55 Timing_Slip_Parameters_-_Build_Timing_Slip

SHIFT= ---

"Shift = "

SDR= ---

"Shift Direction and Rate = "

GETSHIFT --- n1 n2

Prompts for shift in the form xxx.xx ms , inputs number and converts it to n1 microseconds and n2 milliseconds.

GETSDR --- pfa f

Get Shift Direction and Rate

Prompts for and loops until one of A@1 ...... R@20 is input.

pfa = pfa of input word (e.g.A@1) f = 0 for adv. , 1 for rtd.

TYPE3 subtype --- (768 + subtype)

Sets datatype = upper byte to 3 by adding 768 to subtype.

TIMING ---

Input and store timing shift data.

Use :- TIMING x , x = user assigned timing shift data name

Timing data word x is built in the form :-

Address	Data
pfa+2	Type (upper byte) = 3 , Timing Slip Subtype (lower byte) = 0 for advance Subtype (lower byte) = 1 for retard
pfa+4	Shift , n2 milliseconds
pfa+6	Shift , n1 microseconds
pfa+8	pfa of shift direction and rate word

SUB3. n ---

n = 0 " - TIMING SLIP (ADVANCE)"

n = 1 " - TIMING SLIP (RETARD)"

Screen_56 Display_Timing_Slip

SHIFT. pfa --- pfa

Shift Print. "Shift = xx.xxx ms"

SDR. pfa --- pfa

Shift Direction/Rate Print. Eg "Shift Direction and Rate = A@1"

DIST pfa --- pfa

Display Timing. Prints name , type , subtype and data.

TDISP pfa ---

Timing Display. Same as DIST but drops pfa

TDISPLAY ---

Timing Display. Use :- TDISPLAY x , x = timing slip data name

SHOWT ---

Show Timing Slip. Lists all Timing Slip data names

TIMING ---

Redefinition of TIMING. Use :- TIMING x , x = data name

Prompts for then displays entered timing slip.

Screen_57 Edit_Timing_Slip_,_Set_Timing_Slip

S --- 0 SDR --- 1

ETP ---

Edit Timing Prompt. "Shift = S"

"Shift direction and rate = SDR"

PART	--- n	Parameters Timing. Uses ETP to prompt for parameter to be changed. No exit until valid input given. n = either 0 or 1.
ES	pfa --- pfa	Edit shift. New shift input and stored.
ESDR	pfa --- pfa	Edit Shift Direction and Rate.
EDITT	pfa --- pfa	Edit Timing. Prompts for data then edits timing.
PDET	---	Prompt for Display and Edit Timing. "C = Change , E = Exit".
DET	pfa ---	Display and edit timing. Uses PDET , if C uses EDITT to edit timing , if E exits edit mode.
TEDIT	---	Timing data Edit. Use :- TEDIT x x = user assigned timing slip data name.
TSET	pfa --- f	Timing Set. If datatype associated with pfa = 3 a true flag is left on the stack else reports invalid forward reference , ff.
DOT	pfa ---	Do Timing Slip
UDOT	---	User DO Timing Slip. Use :- UDOT x , x = timing slip data name Timing slips using UDOT allow keyboard break-in to stop slip.
PDOT	pfa ---	Programme Do Timing Slip. Keyboard interruption of timing slip disallowed by TFTN = 0 , DOT executed then TFTN set back to 1

Screen_58 FORTH_Word_Parameters_-_Build_FORTH_Word

FW= ---

"FORTH Word = "

GETFW --- pfa

Get Forth Word. " TESTED FORTH Word = " , inputs a previously defined and tested FORTH word and leaves the pfa on the stack

TYPE4 subtype --- (1024 + subtype)

Sets datatype = upper byte to 4 by adding 1024 to the subtype.

FORTHWORD ---

Use :- FORTHWORD x , x = user assigned FORTHWORD data name for use in controller programming . If x is unique FORTHWORD prompts inputs and stores the data as follows :-

Address	Data
pfa+2	Type (MS byte) = 4 , FORTH Word
	Subtype (LS byte) = 0
pfa+4	pfa of previously defined and tested
	FORTH Word , y.

SUB4. n ---

Subtype 4 Print. No subtypes associated with FORTH words. n is dropped and only " - FORTH WORD" is printed.

FW. pfa --- pfa

FORTH Word Print. "FORTH Word = x"

DISF pfa --- pfa

Display FORTH word. Prints the following :-

" x - FORTH WORD "

" FORTH Word = y"

x = Controller FORTH word , used with EDIT , SET , RUN etc.

y = Normally defined FORTH word - colon definition.

FDISP	pfa ---	FORTH Word Display. Uses DISF then drops pfa.
FDISPLAY	---	Use :- FDISPLAY x x = name assigned to Controller FORTH word

Screen_59 Display_and_Edit_FORTH_Word

SHOWF	---	Show FORTH words. Lists all datatype 4 names.
FORTHWORD	--- ,	Redefinition of FORTHWORD. Prompts for , stores then displays entered data.
EFW	pfa ---	Edit FORTH Word. Allows the data word y to be changed.
DEF	pfa ---	Display and Edit FORTH word. C for Change , E for Exit.
FEDIT	---	FORTH word Edit. Use :- FEDIT x x = FORTHWORD data name.
FSET	pfa --- f	FORTHWORD Set. If datatype associated with pfa = 4 a true flag is left on the stack else reports invalid forward ref. , ff.
DOFWD	pfa ---	Do FORTH Word. Data word y stored under name x is executed.
UDOFWD	---	User Do FORTH Word. Use :- UDOFWD x . Executes x data = word y

Screen 60 Set Place , Year , Weekday Number

SETPL	--- Set Place. Location PLACE cleared to ASCII spaces. Prompts for place name which is input and stored at PLACE
PL.	--- Place Print. Place name fetched from PLACE and printed.
SETYR	--- Set Year. Prompts for year which is input and stored at YEAR LEAP = 0 if not a leap year. LEAP = 1 if year is a leap year.
YR.	--- Year Print. Year fetched from YEAR and printed.
CLRDY	--- Clear weekday number location WDAY.
DYP	--- Weekday number prompt.
SETDY	--- Prompt for and store valid weekday number at WDAY

Screen 61 Read Real Time Clock

CLKIOF	--- Clock Interrupt Off. Masked on parallel port MP-LA-1.
CLKION	--- Clock Interrupt On. Enabled on parallel port MP-LA-1.
RSTCLK	--- Reset Clock 1 s interrupt flag by reading MP-LA-2 , PRA
SEC?	--- 1 Hz clock falling edge ? Loops checking MP-LA-1 CRB b7

DV? ---

Data Valid flag falling edge ? Loops checking MP-LA-2 CRA b7

RDCLK --- 1S 10S 1M 10M 1H 10H 1D 10D 100D 1Y

Read time from hardware Clock. Resets 1 Hz flag and waits for next falling edge. Resets 1Hz flag and loops on DV flag checking until parameter data = 9. Once equal to 9 the next multiplexing cycle is read onto the stack with the parameter data masked.

Clock Mux Data :- MS 4 bits = BCD parameter data 0 9
LS 4 bits = BCD clock data

STYR 1Y ---

Store unit years at YEAR+3 in ASCII

STDYNO 1D 10D 100D ---

Store year day number as a single number n at DYNO

STSEC 1S 10S 1M 10M 1H 10H ---

Store time in seconds in the three real time software clocks which use locations CMS , CHMS , CDHMS

1M and 10M are converted to seconds and added to 1S 10S and the result Clock Minutes and Seconds is stored at CMS as a double number. 1H and 10H are converted to seconds and added to CMS to give Clock Hours Minutes and seconds in seconds which is stored at CHMS as a double number. Location WDAY contains the weekday number as previously set. This is multiplied by 8600. to obtain the number of seconds per day and the result is added to CHMS to give Clock Days Hours Minutes and Seconds in seconds which is stored at CDHMS as a double number. The three different software clocks CMS , CHMS and CDHMS are therefore initialized from the hardware clock.

CPYCLK ---

Copy Clock to locations used during programme lookahead.

CMS copied to MS

CHMS copied to HMS

CDHMS copied to DHMS

GETIME ---

Get Time. Clock interrupt off , clock read , data stored and also copied to program lookahead locations. Clock interrupt enabled. The time taken to read convert and store the clock data was measured as 55 ms.

Screen 62 Increment Real Time Clock. Get Time + 1 s

INCLK ---

Increment the three real time clocks as follows.

```
Inc CMS , if = 3600. s (1 hr) CMS = 0. else CMS = CMS + 1.
Inc CHMS , If = 86400. s (1 day) ,
              Inc WDAY and DYNO ,
                  If DYNO is greater than 365 and it is a leap
                     year DYNO = 366 else  DYNO = 1
                  else DYNO = DYNO + 1
           else CHMS = CHMS + 1
Inc CDHMS , if = 604800. , CDHMS = 0. and WDAY = 0.
            else CDHMS = CDHMS + 1.
```

GETIME+1 ---

Get time plus 1 second. Reads and stores the hardware clock time as three software clocks which are all immediately incremented by 1 s . Software clocks therefore 1 s ahead.

Screen 63 Display Day , Display Time

TODHMS	d --- S M H D To Days Hours Minutes and Seconds. Double no. to 4 single nos.
DAY.	n --- week Day number Print. Left justifiedin a 10 column field.
2CON	n --- Two convert. Single no. n converted to 2 digits and printed.
MSC.	S M --- Minutes Seconds Convert Print. Form :- MM : SS
HMS.	S M H --- Hours Minutes Seconds Print. Form :- HH : MM : SS
DHMS.	S M H D --- Day Hours Minutes Seconds Print. Form :- Weekday HH : MM : SS
CDHMS.	--- Clock Day Hours Minutes Print. Fetches the time from CDHMS , subtracts 1 s to correct time , prints :- Weekday HH : MM : SS

Screen 64 Allot Programme

TYPE5	--- 1280 Defined as a constant to push 1280 onto the stack. Used to set the datatype (upper byte of single number) to 5
ALLOTP	--- Allot Programme space. Initializes three priority blocks of 25 entries. Each entry consists of 4 bytes for storing the time double number followed by 2 bytes for storing the pfa of the function to be executed at that time. 24 entries are avaliable for data , the 25th being a dummy entry which is always set to the initialized values of -1. and 0

Address		Data
pfa+2	-	Type (MS byte) = 5
	-	Subtype (LS byte) = 0

Priority_Block_#1

Address	Entry_No.	Time_(4_bytes)	Function_pfa_(2_bytes)	
pfa+4	1	-1.	0	
pfa+10	2	-1.	0	
! V	! V	! V	! V	
pfa+142	24	-1.	0	
pfa+148	25	-1.	0	Dummy entry

Priority_Block_#2

pfa+154	1	-1.	0	
pfa+160	2	-1.	0	
! V	! V	! V	! V	
pfa+292	24	-1.	0	
pfa+298	25	-1.	0	Dummy entry

Priority_Block_#3

pfa+304	1	-1.	0	
pfa+310	2	-1.	0	
! V	! V	! V	! V	
pfa+442	24	-1.	0	
pfa+448	25	-1.	0	Dummy entry

SUB5. n ---

Subtype 5 Print. Prints " - PROGRAMME". No associated subtype.

SHOWP ---

Show Programmes. Lists all programme data names.

SHOVEC	n ---	Show functions vectoring according to n. n = 1 , SHOWI ; n = 2 , SHOWT ; n = 3 , SHOWF ; n = 4 , SHOWP
SHOP	---	Show functions Prompt.
SHFTNS	---	Show Functions. If required all defined names of one of the 4 functions can be displayed.
Screen_65	Programme_Parameters	
FUD	---	" Function Data"
PHF	---	Prompt for Hourly Function data. For priority block 3.
GETMS	--- d	Get Minutes and Seconds as seconds.
PDF	---	Prompt for Daily Function data. For priority block 2.
GTH	--- d	Get Hours , convert to seconds.
GTHMS	--- d	Get Hours minutes and Seconds in seconds.
PWF	---	Prompt for Weekly Function data. For priority block 1.
GTD	--- d	Get Day. Prompts for and gets weekday no. , converted to s.
GETDHMS	--- d	Get Day Hour Minute and Second. Uses GTD and GTHMS and adds.

CHKTYP	datatype --- f Check Datatype. tf if datatype = 2 , 3 , 4 or 5
GETFTN	--- pfa Get Function. Loops until function name with valid datatype is input then exits leaving function pfa on stack.
Screen_66	Display_Programme
EMTY?	addr --- addr time ftn Empty entry ? True flag if time = -1. i.e. No Entry
ENTRY?	addr --- addr f Entry ? True flag if there is an entry. Time not equal to -1.
DSH	n --- Dash. Prints n dashes followed by 1 space.
2DSH:	--- Two dashes plus a colon. " -- :"
DASH	--- "---------- -- : -- : -- ----------"
TYPEVEC	n --- Type vector. Prints type and subtype from subtype number n.
FTN.	pfa --- Function name , type and subtype Print. Checks for forward ref
FTN@.	(entry x addr) --- (entry x addr + 6) Function Fetch and Print. The entry addr is the addr at which the time is stored. The function pfa is fetched from entry x addr + 4 and its name , type and subtype are printed. Entry x addr + 6 = entry (x + 1) addr is left on the stack

DSP1	(entry addr) (time in s) --- (entry addr + 6) Display a priority block 1 entry. "Weekday HH : MM : SS Function name - Type (Subtype)"
DSP2	(entry addr) (time in s) --- (entry addr + 6) Display a priority block 2 entry. "--------- HH : MM : SS Function name - Type (Subtype)"
DSP3	(entry addr) (time in s) --- (entry addr + 6) Display a priority block 3 entry. "--------- -- : MM : SS Function name - Type (Subtype)
VEC123	(entry addr) (time in s) n --- (entry addr + 6) Vector to DSP1 , DSP2 or DSP3 according to n. Inc entry addr.
DSP123	(entry addr) --- (entry addr) f Display PB#1 , PB#2 or PB#3 entry data from given entry addr. If empty entry prints all dashes , leaves a true flag else prints entry data using priority block no. from location PB and VEC123 and leaves a false flag.

Screen_67 Display_Programme

E#.	(current addr - addr of top of current PB) --- Entry number print. "Entry x" where x is calculated using the supplied stack number divided by 6 then adding 1 .
VECTPB	n --- (addr of location containing PB#n addr) Vector to priority block address storage locations :- TP1 contains addr of top of PB#1 of current programme TP2 contains addr of top of PB#2 of current programme TP3 contains addr of top of PB#3 of current programme

8DISP	(entry x addr) --- (entry x addr + 48) Eight Display. Displays entry no. time and function for 8 entries and leaves addr of entry (x + 8) on the stack.
ADJA	(entry addr) PB#n --- (adjusted addr) PB#n Adjust address. Takes dummy entry no. 25 into account. Entry addr in previous block , adjusted addr = entry addr - 6 Entry addr in current block , adjusted addr = entry addr Entry addr in next block , adjusted addr = entry addr + 6
PB#	(entry addr) --- (adjusted addr) PB# Priority Block number calculated from entry addr. ADJA used to adjust address on priority block change. " Priority Block #n"
DTF	--- Day Time Function Headings.
PBDSP	(entry addr) --- (entry addr) Priority Block Display. Prints block number , headings and 8 entries of time and function. The address is incremented by 48 by 8DISP therefore 48 is subtracted to leave (entry addr).
PEP	--- Prompt for display and Edit Programme. N = Next 8 1 = top PB1 I = Insert E = Exit P = Previous 8 2 = top PB2 D = Delete 3 = top PB3

Screen_68 Display_and_Edit_Programme

FNDPBT	n --- (addr of top of PBn) Find Priority Block Top.

INE#B	--- (entry n addr offset from block top) Input entry number convert to offset relative to block top. Inputs no. between 1 and 24. Converts to an address offset relative to block top using 6(E# - 1).
MVDWN	(entry n addr offset) --- (entry n addr) (entry n addr) Move all entries below and including n down 1 position. Two copies of entry n addr are left on stack. Entry 24 is lost.
INSERT	(entry addr) --- (entry addr) Insert entry in priority block. The priority block number is first calculated. Prompts "Insert at Entry No. = " and inputs number n. All entries below and including entry n are moved down 1 position , entry 24 being lost. Prompts for and inputs entry n execution time depending on priority block being edited. Entry n function then prompted for , input and stored.
DELENT	(entry addr) --- (entry addr) Delete entry from priority block. The priority block number is first calculated from (entry addr). Prompts :- " Delete Entry No = " and inputs entry no. n. All entries from n+1 to 25 (which always contains time = -1. , ftn = 0) are moved up 1 position. This overwrites entry n and sets entry 24 to an empty entry.

Screen_69 Programme_Display_Control

DEPINIT	pfa --- pfa+4 Display and Edit Programme - Initialization. Prints programme name and type. Sets TP1 to pfa+4 , TP2 to pfa+154 , TP3 to pfa+304 and PB to 1. Addr of PB1 entry 1 left on stack (pfa+4)

DEP	pfa --- Display and Edit Programme. Initializes using DEPINIT then prompts for a single character followed by return. P - Previous 8 entries. If at top of PB1 display first 8. N - Next 8 entries . If at bottom of PB3 display last 8. 1 - Display first 8 entries of PB1 2 - Display first 8 entries of PB2 3 - Display first 8 entries of PB3 I - Insert entry D - Delete entry E - Exit edit mode

Screen 70 Display and Edit Programme

PRTP	pfa --- Print Programme on printer. Prints programme name and type and initializes display addresses. Prints headings followed by all entries plus 1 empty entry (a line of dashes) for each priority block.
PROGRAMME	--- Redefinition of PROGRAMME which allots programme space then displays the empty programme. Data can then be entered using the edit functions. Use :- PROGRAMME x , x = prog data name
PROGRAM	--- Alternative command for PROGRAMME. Use :- PROGRAM x , x = program data name
PDISP	pfa --- Programme Display. If VDU? = 1 use DEP else use PTRTP

PEDIT ---

Programme Edit. Use :- PEDIT x , x = programme data name

PSET pfa --- f

Programme Set. If datatype = 5 the pfa is stored at CURPRG and a true flag left else pfa dropped , -1 stored at CURPRG and a false flag is left on the stack.

UPSET ---

User Programme Set. Use :- UPSET x , x = programme data name

CPRG. ---

Current Programme Print.

"Current Programme :- Not Set !" if programme not set.

"Current Programme :- x" if programme x set

Screen_71 Programme_Lookahead

A programme can contain up to 24 entries per priority block. All empty entries have execution time = -1. and function = 0 . "Dummy" entry 25 always has time = -1. and function = 0

Priority_Block_#1 Weekly_Data

EDHMS = Execution time with weekDay , Hours , Minutes and Seconds specified. When the copied clock time DHMS = EDHMS the associated function (e.g. ftn1) is executed. Ftn1 is therefore executed once a week.

Priority_Block_#2 Daily_Data

EHMS = Execution time with Hours , Minutes and Seconds specified. When the copied clock time HMS = EDHMS the associated function (e.g. ftn2) is executed. Ftn2 is therefore executed once every day.

Priority_Block#3 Hourly_Data

EMS = Execution time with Minutes and Seconds specified. When the copied clock time MS = EMS the associated function (e.g. ftn3) is executed. Ftn3 is therefore executed once every hour.

The programme lookahead software calculates the difference between the programmed execution times and the copied clock times for each entry of each priority block. The minimum difference obtained in each priority block together with the corresponding entry address is retained. This gives the next function to be executed in each priority block. The function associated with the minimum time difference is the one set up as the next function to be executed. Equality of time differences is resolved by giving the priority block #1 difference the highest priority and the priority block #3 difference the lowest priority.

Let the programmed execution time be T and the hardware clock time C. The number of seconds before the function at T is to be executed is given by T - C for T greater than C. For T less than C the difference T - C is negative and the time before execution is given by $T - C + C_{max}$ where C_{max} is the maximum count applicable to the priority block being processed.

Priority Block #1	C_{Wmax} = 604800.	(no. of seconds in one week)
Priority Block #2	C_{Dmax} = 86400.	(no. of seconds in one day)
Priority Block #3	C_{Hmax} = 3600.	(no. of seconds in one hour)

CHKW (entry n addr) --- (entry n addr + 6)

Check Weekly data. Calculates $T_W - C_W$

If positive , time before entry n , $T_D = T_W - C_W$

If negative , time before entry n , $T_D = T_W - C_W + C_{Wmax}$

T_D is compared with the contents of WC (= Weekly Count). If less T_D is stored at WC and the address of entry n is stored at WEA (Weekly Entry Address) else T_D and entry n address are dropped. Finally the address is incremented by 6 to point at entry n+1.

NXTW ---

Next Weekly function. Initialises WC to a large number. Gets pfa of current programme (from CURPRG) and adds 4 to point at first entry of PB#1. This is stored at WEA. All entries are checked using CHKW which leaves the time before the next function in WC with the corresponding entry address in WEA.

CHKD (entry n addr) --- (entry n address + 6)

Check Daily data. Similar to CHKW but returns the minimum daily count in DC with corresponding address in DEA for entry n compared with previous contents of DC.

NXTD ---

Next Daily function. Similar to NXTW but returns time to next PB#2 function in DC with corresponding entry addr in DEA.

CHKH (entry n) --- (entry n +6)

Check Hourly data. Similar to CHKW but returns the minimum hourly count in HC with corresponding entry address in HEA for entry n compared with the previous contents of HC.

NXTH ---

Next Hourly function. Similar to NXTW but returns the time to the next PB#3 function in HC with entry addr in HEA.

Screen_72	Programme_Lookahead
NTOC	---
	Next time/ftn data TO Current time/function storage. Contents of NXTIME to CURTIME — execution time in s Contents of NXTFTN to CURFTN — Function pfa Contents of NPB to CPB — Priority Block number.
EATTI	PB#n (entry addr) ---
	Entry Address To Time. Execution time copied to NXTIME Function pfa copied to NXTFTN and PB# stored at NPB
GETNXT	---
	Get next function to execute. Selects smallest of WC , DC and HC in current programme. Any equalities are easily resolved in the order PB#1 = highest priority , PB#3 = lowest priority. NTOC transfers previous data to current function storage and the new data is stored as the next function. If the current programme has not been set CURPRG = -1 and NXTFTN = 0
GETC=N	---
	Get Current function data = Next function data.
PBPRT	(time in s) PB#n ---
	Priority Block n time and function Print.
CNDIS	---
	Current and Next Display. Current function time format from CPB , function from CURFTN Next function time format from NPB , function from NXTFTN If an programme empty :- "Empty Programme"

Screen_73 Data_to_Film

Screen 73 was reserved for words to allow the CPU to write alphanumeric data to film using a DE/210 Vacuum Flourescent Display which is 10 characters wide.

Proposed use :- Control register B bit 3 was to control fast film advance.

Simple modification to the 6018 Camera Interface/Display would be necessary.

FORTH definition to advance film by 1 text line :-

```
: NEXTLINE B3ON 100 0 DO LOOP B3OFF ;
```

The data listed below would be written to film each time any ionogram except an oblique Tx was executed.

Line 1	Place (P) , Year (Y) , Day no. (D) in the form :-	-P-YY-DDD-
Line 2	Time (H,M) and Type (VER , OBL etc) in the form :-	HH-MM--TTT
Line 3	Start f (S MHz) , End f (E MHz) , Window Ht (W Km) :-	SS-EE--WWW
Line 4	Message (M) , up to 10 characters long :-	MMMMMMMMMM

Each time a programme was set the programme name was to be written to film.

Screen_74 Data_to_DCS

8 bit ASCII alphanumeric data is sent to the DCS. When the eighth bit is set the other 7 bits are interpreted by the DCS as a control code :-

80_{16}	Global data to DCS , no not write to tape.
81_{16}	Global data to DCS , write to magnetic tape.
82_{16}	Preceeds each set of cell data.
83_{16}	Signals ionogram end.

Global Data - Code 80_{16} or 81_{16}

Name	No. of bytes	Data
Place	16	Alphanumeric
Year	2	Numeric
Time	9	Numeric - DDDHHMMSS
Basic Rate (k_B)	4	numeric kHz/s
No of cells (n)	2	Numeric
Cell Period (T_C)	2	code = x , $T_C = (2^x)(1/32)$ s
FFT Sample Rate	2	code = y , FFT Rate = 2^y Hz
L.O. Offset	5	numeric
A/R shift	7	sign + 6 ASCII , tens of microseconds

Cell Data - Code 82_{16}

Name	No. of Bytes	Data
Cell start freq	8	Hz
Rx1 antenna	1	numeric
Rx2 antenna	1	numeric

RSTDCS ---

Restore DCS. Clears DCS status flag called DCSTAT.

CLRACK ---

Clear DCS Acknowledge flag. Flag set by falling edge on CB1

DCSACK? ---

DCS Acknowledge ? After sending a character to the DCS the acknowledge flag is checked. If set the character was received and the loop is exited. If not set after 10 checks DCSTAT is set to signal that the DCS is not acknowledging.

TODCS	c --- To DCS. If flag DCS? = 1 c is output to the VDU and sent to the DCS. If DCS? = 0 character c is just sent to the DCS. If DCSTAT = 1 the DCS is not acknowledging , subsequent data is not sent until DCSTAT =0.
ATODCS	n --- n in the range 0 to 9 is converted to ASCII and sent to DCS.
NTODCS	n1 n2 --- Numbers to DCS. n2 must be greater than 2. n1 is the stack number to be converted to n2 ASCII digits and sent to DCS.
DCSCR	--- DCS Carriage Return. Used in listing DCS data to VDU.
DCSTAB	--- DCS Tabulate. Tabulate data listed to VDU.
DCSDATA	--- Clears VDU and sets DCS? = 1
Screen_75	Data_to_DCS
PLDCS	--- Place to DCS. 16 ASCII characters from PLACE
YRDCS	--- Year to DCS. 2 ASCII characters from YEAR
DHMSDCS	--- Days , Hours , Minutes , seconds to DCS. From DYNO and CDHMS
DATDCS	--- Current ionogram global Data to DCS. Basic rate from current sounding pfa (stored in CURS) + 4 No. of cells from current sounding pfa + 24

Cell period from current sounding pfa + 12

FFT sample rate from current sounding pfa + 16

L.O. offset frequency from current sounding pfa + 10

A/R position from location SHIFT

IDAT	n --- Initial Data to DCS. n = control code , either 80_{16} or 81_{16} Sends global data - place , year , time , DATDCS and ? to DCS.
IDATDCS	--- Initial Data to DCS , control code 80_{16} , not to mag tape
IDATMT	--- Initial Data to DCS and Mag Tape , control code 81_{16}.
CDATDCS	--- Cell Data to DCS , control code 82_{16}. Sends data from CFREQ = cell frequency and CANT = cell antennae followed by ?.
ENDCS	--- End code to DCS. Sends 83_{16} followed by ? to DCS.
<u>Screen 76</u>	<u>Print Title and Display Status</u>
HDING	--- Heading. Clears screen and prints title on top line.
PLYR.	--- Place Year Print.
DYNO.	--- Year Day Number Print.
TPOSN.	--- Timing Position Print.
TITLE	--- Title the screen . Includes current and next function status.

Screen 77 Display Datatype. Edit Datatype

PRINTER ---

Set I/O device flag to printer.

DISFTN pfa ---

Display Function data. Checks datatype and vectors to corresponding display routine.

DISPLAY ---

Display data on VDU. Use :- DISPLAY x where x can be a Sounding , an Ionogram , a Timing Slip , a FORTH Word or a Programme data name.

PRINT ---

Display data - printer format. Use :- PRINT x , x = data name

LOG ---

Log data on printer. Use :- LOG x , x = any of the data names If x = an Ionogram data name its Sounding is also listed.

EDFTN pfa ---

Edit Function Data. Checks datatype and vectors to corresponding edit/display routine.

EDIT ---

Edit datatype. Use :- EDIT x , x = S,I,T,F or P data name.

Screen 78 Set Datatype

SETFTN pfa --- f

Set Function. Checks datatype and vectors to corresponding set routine.

SETNXT ---

Set Next function. First scans the current programme to find the next function. If not an empty programme it is set and the flag generated by setting is stored at NXTSET and COMPARE

SET ---

Set function. Use :- SET x , x can be any of the 5 datatype names. If correctly set prints "Set !".

Screen_79 Show_and_Delete_Datatype

SHOWTYP ---

Shows all defined datatype names since cold start.

DLIST ---

Data List. Lists all defined datatype names and allows erasure of data entries from the dictionary. NB :- Erasure of name x erases x plus all names defined after x. The controller program reports invalid references which may occur if DLIST is used unwisely.

Screen_80 Sounding_and_Cell_Start_Frequencies._Function_Stop.

DISCN ---

Display Current and Next functions if flag STAT = 1

SETUP ---

Sets frequency to 0.5 MHz , AGC to internal , sets next function , displays current and next time and function if STAT = 1 and clears the ionogram busy flag ION.

FNDNXT ---

Find Next function. Makes a copy of the three real time clocks then gets the next function in the current programme and displays the current and next function data is STAT = 1

SSFLIN ---

Sounding Start Frequency , Linear overall rate.

Equation (3.4) gives the start frequency of sounding i as

$f_{Si} = f_I + k_0 i n T_C$

$k_0 n T_C = f_{snd}$ the frequency change per sounding which is calculated when the ionogram is set and is stored at F/SND

Sounding start frequency $f_{Si} = f_I + i f_{snd}$

The calculator memory is initialized to f_I. Consecutive sounding start frequencies are obtained by adding f_{snd} to the contents of the calculator memory.

i.e. New sounding start frequency = old start frequency + f_{snd}

SSFLN ---

Sounding Start Frequency , Log overall rate.

Equation (3.6) gives the start frequency of sounding i as :-

$$f_{Si} = f_I e^{AiT_S}$$
$$= f_I e^{AT_S(i - 1)} e^{AT_S}$$

e^{AT_S} is calculated when the ionogram is set and is stored at EATS. The calculator memory is initialised to f_I. Consecutive sounding start frequencies are obtained by multiplying the calculator memory contents by e^{AT_S}.

i.e. New sounding start freq = (old start freq)(e^{AT_S})

CSFN ---

Cell n Start Frequency. (Linear or Log). Equation (3.2) gives cell j start freq in sounding i as :- $f_{Cij} = f_{Si} + f_j$

f_{Si} is stored in calculator memory (see above). To this is added the cell j offset freq f_j. The result is sent to the synthesizer latches (PC1) to be loaded by the next Tcell pulse Film drive and antennae data also latched for Tcell pulse load

STO,1 ---

Generates STROBE0 followed by STROBE1. Clears all flip-flops.

STOP ---

Stop Ionogram execution. Hardware and software reset , end code sent to DCS , next function found and set.

HALT ---

Halt ionogram execution by setting the current sounding number equal to the total no. of soundings in the ionogram.

HALTED? ---

Loop that checks ionogram busy flag until it is cleared.

Screen_81 Tcell_Interrupt_Routine

NCELL ---
Next Cell.

If subtype of current ionogram is stationary Doppler then :-

Compares SOUND# (sounding no.) with #SOUND (total no. of soundings).

If SOUND# is the greater then ionogram end has been reached - STOP

else send current cell data (calculated during previous cell) to DCS

increment AGC# and compare with AGC data of current sounding. If greater pulse receiver AGC and reset AGC# to 1

If film marks required increment TMK# and compare with current sounding time mark data. If greater make time mark and reset TMK# to 1

increment CELL# , compare with no. of cells in current sounding. If greater reset CELL# to 1 and increment SOUND#

Finally execute CSFN to calculate and latch next cell data.

```
If subtype of current ionogram is NOT stationary Doppler then :-
Compare SOUND# (sounding no.) with #SOUND ( total no. of soundings).
If SOUND# is the greater then ionogram end has been reached - STOP
else if cell no. = 1 and if sounding AGC data = not fixed , pulse AGC
     send current cell data (calculated during previous cell) to DCS
     increment CELL# and compare with no. of cells in current sounding
     If greater reset CELL# to 1 and increment SOUND#
     If linear overall rate execute SSFLIN else log rate execute SSFLN
Finally execute CSFN to calculate and latch next cell data.
```

Screen_82	Run_Function
RS	pfa --- Run Sounding. Not allowed therefore issues error message.
RI	pfa --- Run Ionogram. AGC set to external. If ionogram has data to tape = yes IDATMT executed else IDATDCS. The sweep flip-flop is then enabled , COMPARE flag set to 0 to inhibit comparison of next function start time with clock during an ionogram and the ionogram busy flag is set to 1. The next function to be executed is found and displayed with its execution time.
RTS	pfa --- Run Timing Slip. Reads hardware clock then finds and displays next function in current programme. Does timing slip then sets the next function to be executed.
RFW	pfa --- Run FORTH Word. Reads hardware clock then finds and displays next function in current programme. FORTH Word executed then the next function is set.

RPRG	pfa --- Run Programme. pfa is dropped and the hardware clock is read. Gets current function = next function in the current programme. The next function is set and the screen titled.
RFTN	pfa --- Run Function. Checks datatype and vectors to correct run word
RUN	--- Run Function. Use :- RUN x x = function name If typed during the execution of an ionogram the executing ionogram is halted , the specified function (x) is set and then run on the negative edge of the 1 Hz clock signal.
AOFF	--- Automatic operation Off. Status flag set to off , current ionogram halted and -1 stored in CURPRG. SETNXT then executed and status set to former state.
CANCEL	--- Cancel programme. Sets CURPRG to -1 using AOFF.
PAUSE	--- Pause programme execution. CURPRG saved then set to -1 using AOFF to inhibit automatic sequencing of functions. Automatic operation resumed by pressing VDU space bar.

Screen_83 Clock_Service_Routine

PBTIME	n --- Vectors to CDHMS , CHMS or CMS according to n.
TIME	--- Set time display flag TON until VDU space bar pressed.

CLOCK ---

Increments the three software clocks. If flag COMPARE is set the priority block number of the next function is obtained from NPB and the corresponding clock time is fetched and compared with the function execution time stored in NXTIME. If equal the next function is run. NB :- Ionogram execution. Since the clock is 1 s ahead of the real time the run command initialises certain parameters (see RI) then enables the sweep start flip-flop. Execution begins on the next falling edge of the 1 Hz clock which occurs at the programmed time.

If TON = 1 the time is displayed with updates each second.

Screen_84 Interrupt_Polling

POLL ---

Interrupt Polling routine. After any interrupt :-

Tcell pulse interrupt flag checked , if set executes NCELL

Clock interrupt flag checked , if set executes CLOCK

If neither Tcell pulse nor clock , "Interrupt error"

Return from interrupt.

SETIRQ ---

Set Interrupt Request vectors.

pfa of POLL stored at $23E6_{16}$

S-BUG vector to primitive IRQ set by storing $1CE9_{16}$ at $DFC8_{16}$

Screen_85 Command_List

The first 7 words listed are used in defining the word HELP

HELP ---

Displays an abbreviated command list.

Screen_86 System_Initialization

SETTM	--- Set hardware clock Time. " Set time then press SPACE BAR".
VWARM	--- Vertichirp controller warm start. Interrupts off , interrupt vectors set , flags initialized , ports initialized , controller hardware reset , interrupts enabled , hardware clock read and next function found and set and screen titled.
VCOLD	--- Vertichirp controller cold start. Queries "Heath VDU (Y or N) ?" , prints "Initialization" then prompts for place name , year and weekday number. Hardware clock must then be set. CURPRG set to -1 = programme not set. COMPARE set to 0. BDATA (beginning of data) initialized. VWARM executed.
COLD?	--- Executed by kernal word ABORT in place of QUIT. Checks to see if any data has been entered. If no data executes VCOLD else executes VWARM. QUIT (loop that checks and inputs from keyboard) then executed.

SETTIME ---

Set hardware clock Time , read into computer and display.

The last two lines of FORTH execute to assign addresses as follows :-

Hex Address	Hex Data	Function
0023	982E	initial fence set to new location of TASK
0025	9839	new cold start value for dictionary pointer
0027	9810	new cold start value for vocabulary link
115F	9808	reference to FORTH in kernal word ABORT
cfa of COLD?	1177	ABORT to execute COLD? before QUIT
nfa of SETTIME	1196	set backlink from word FORTH to SETTIME
119E	982E	new nfa of word TASK
11C3	9800	new nfa of word FORTH