0-315-12332-X

National Library of Canada — Bibliothèque nationale du Canada

Canadian Theses Division — Division des thèses canadiennes

Ottawa, Canada
K1A 0N4

60243

PERMISSION TO MICROFILM — AUTORISATION DE MICROFILMER

- Please print or type — Écrire en lettres moulées ou dactylographier

Full Name of Author — Nom complet de l'auteur

LO, WING FAI

Date of Birth — Date de naissance: 24 October 1955

Country of Birth — Lieu de naissance: HONG KONG.

Permanent Address — Résidence fixe

147 TEMPLEMONT PLACE N.E. CALGARY, ALBERTA
T1Y 5A5, CANADA

Title of Thesis — Titre de la thèse

A DIGITAL SIGNAL PROCESSOR FOR A 408 MHz SUPERSYNTHESIS TELESCOPE

University — Université

UNIVERSITY OF ALBERTA.

Degree for which thesis was presented — Grade pour lequel cette thèse fut présentée

M. Sc.

Year this degree conferred — Année d'obtention de ce grade: 1982

Name of Supervisor — Nom du directeur de thèse: D. ROUTLEDGE

Date: 18 August 1982

Signature: Lo Wing Fai

THE UNIVERSITY OF ALBERTA

A DIGITAL SIGNAL PROCESSOR FOR A 408 MHz SUPERSYNTHESIS TELESCOPE

by

Wing F. Lo

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES AND RESEARCH

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE

OF Master of Science

DEPARTMENT OF ELECTRICAL ENGINEERING

EDMONTON, ALBERTA

Fall 1982

THE UNIVERSITY OF ALBERTA

RELEASE FORM

NAME OF AUTHOR Wing F. Lo

TITLE OF THESIS A DIGITAL SIGNAL PROCESSOR FOR A 408 MHz SUPERSYNTHESIS TELESCOPE

DEGREE FOR WHICH THESIS WAS PRESENTED Master of Science

YEAR THIS DEGREE GRANTED Fall 1982

(SIGNED) Lo Wing Fai

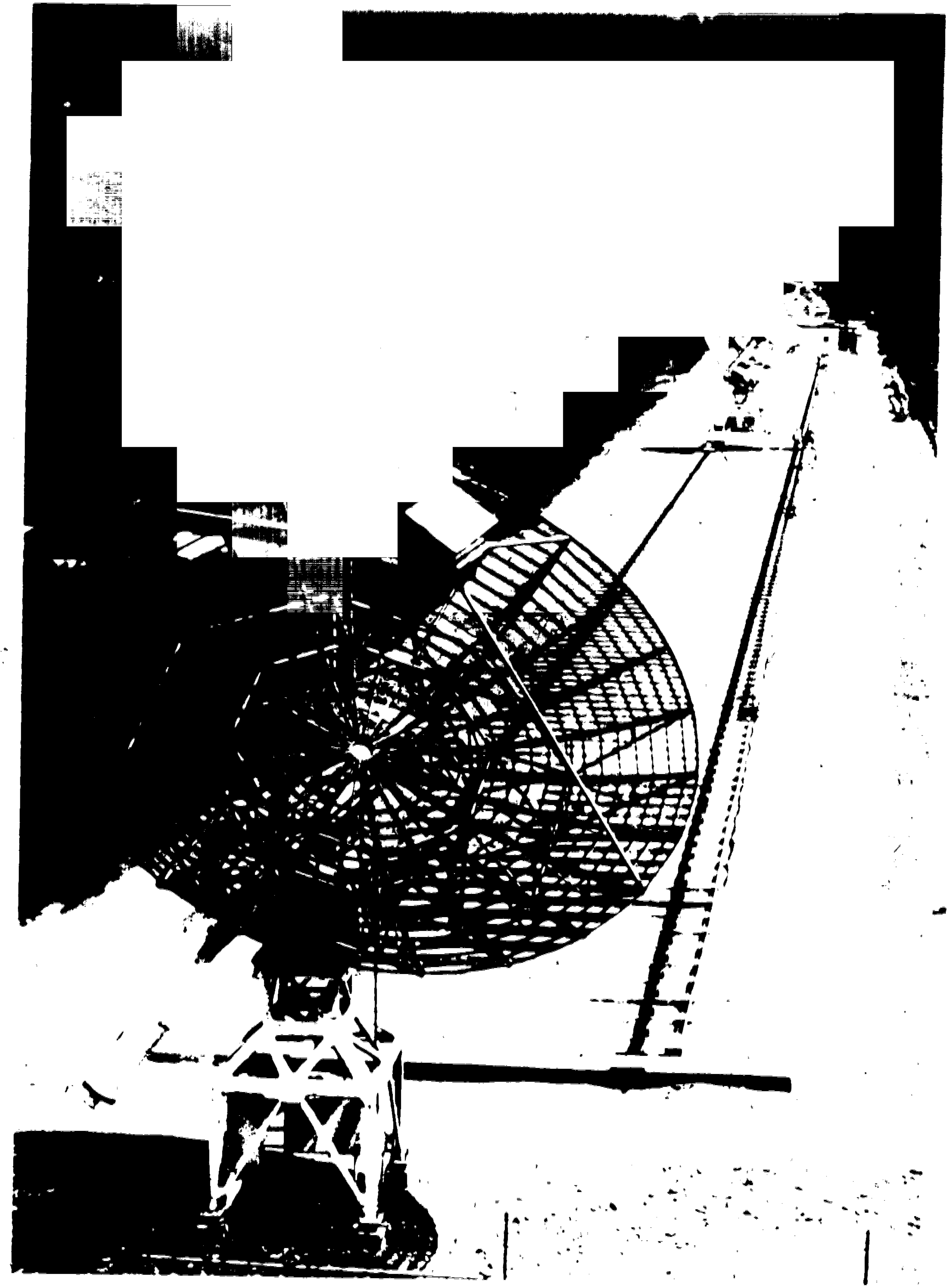
PERMANENT ADDRESS

147 Templemont Place N.E.,

Calgary, Alberta, T1Y 5A5,

CANADA.

DATED 4 August, 1982.



THE UNIVERSITY OF ALBERTA

FACULTY OF GRADUATE STUDIES AND RESEARCH

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research, for acceptance, a thesis entitled A DIGITAL SIGNAL PROCESSOR FOR A 408 MHz SUPERSYNTHESIS TELESCOPE submitted by Wing F. Lo in partial fulfilment of the requirements for the degree of Master of Science.

D. Routledge

Supervisor

Supervisor

Keith A. Stromsmoe

Date 4 August 1982

DEDICATION

To My Parents

ABSTRACT

A real time digital signal processor for a 408 MHz continuum synthesis telescope is described. The system takes in 4 MHz baseband signals, and performs delay equalization, phase derotation and cross-correlation numerically. Real and quadrature channel outputs are also generated without duplication of the correlator. The scheme replaces the variable cable delay and intricate local oscillator phase rotation system, resulting in simpler implementation and lower chances of receiver cross talk.

Path delay compensation is done in a coarse step with a digital delay unit and in a fine step with interpolation of the correlation function. An algorithm which performs interpolation and quadrature channel generation in a single operation is presented. Fringe derotation is performed digitally after correlation. Imperfections of the quadrature channel and ripples caused by fringe derotation after correlation are discussed in detail.

An MC68000 microcomputer is used as the arithmetic processor and controller. A realtime multi-tasking executive, the "Tiny Operating System (TOS)", was specially developed for this application. A monitor program, the "Tiny Operating System Monitor (TOSMON)", was also developed to improve the observability, testability and controllability of the system.

ACKNOWLEDGEMENT

The author expresses his sincere appreciation to his supervisors Dr. D Routledge. Dr. P. E. Dewdney, and Dr. J. F. Vaneldik for their support and advice throughout the project.

The author is also grateful to all the staff of the Dominion Radio Astrophysical Observatory for their help and advice during the development of the 408 MHz digital signal processor. Special thanks must be given to Dr. T. L. Landecker for his continual moral and technical support. Thanks must also be given to Mr. J. H. Dawson for his help with the hardware of the PDP11's. Mr. G. Croes for the modification of the host computer software. Dr. C. Costain for permission to reproduce the map of 3C66 (Figure 6.3.1(a) and 6.3.2). Dr. T. L. Landecker and Dr. P. E. Dewdney for producing the map of 3C66 (Figure 6.3.1(b)) from the visibility functions (Figure 6.3.3), and Mr. B. Oliver for the photographic work involved in the preparation of this manuscript.

The help provided by Mr. P. Haswell of the Department of Electrical Engineering with the 68000 is also acknowledged.

The author wishes to thank the Department of Electrical Engineering and the NSERC for their financial support.

Table of Contents

LIST OF FIGURES

1. INTRODUCTION

1.1 Aperture Synthesis

With Carl Jansky's discovery of galactic radio radiation in 1932, the radio spectrum was opened up to astronomers. Since then, the study of radio radiation from extraterrestrial sources has led to important astronomical discoveries like pulsars and quasars. During the astronomer's quest for ever increasing resolution and sensitivity, the radio telescope, an instrument for receiving radiation from afar, has evolved from Jansky's simple array to the highly sophisticated synthesis telescope.

One of the fundamental limits of angular resolution of telescopes is set by interference to about the ratio of the wavelength to the aperture dimensions. Since wavelengths of the radio spectrum are orders of magnitude longer than the optical wavelengths, radio telescopes require kilometer aperture dimensions, even at the highest operating frequencies, to obtain resolutions comparable to their optical counterparts. Aperture synthesis techniques allow a large aperture to be synthesised with much smaller antenna elements and obtain high angular resolution without having to build a huge filled aperture antenna.

A large aperture may be synthesised by an array of elements suitably connected together. In the study of time invariant radio sources, not all the aperture must be present at the same time. Earth-rotation aperture synthesis[1], or supersynthesis, uses two antenna elements lying some distance apart at the ends of a baseline. When viewed from a distant source, the elements will describe an ellipse with respect to each other as the earth rotates. By changing the distance between the two elements and accumulating the signal in a digital computer, a filled aperture equivalent to the ellipse described by the largest spacing between the elements can be synthesised.

In return for such convenience, the complexity of signal processing is much increased[2]. It requires that a real time signal processing system, either analog or digital, be built into the receiver for forming and controlling the interference fringes. In addition, more data processing, not necessarily in real time, is required to transform the information, or visibility function, gathered over a period of time with different element spacings into a map of the sky intensity.

The aim of this project is to implement the real time signal processor embedded in the receiver with digital techniques. The scheme results in simpler hardware implementation over conventional analog processing systems, with better stability and flexibility.

1.2 Interferometry.

The discussion of interferometry in this section is mainly based on Fomalont and Wright[2]. The basic interferometer consists of two elements separated by a distance, as in figure 1.2.1. By correlating the signals from the elements, a fine structured interference fringe pattern will modulate the main beam of the individual elements.

Assuming the simple case of a point source with monochromatic radiation at frequency ω or wavelength λ, outputs of v_1 and v_2 from the antenna feeds are

$$v_1 = k_1 \sqrt{S} \cos(\omega t)$$

$$v_2 = k_1 \sqrt{S} \cos[\omega(t-\tau)]$$

$$= k_1 \sqrt{S} \cos[\omega t - 2\pi(B/\lambda)\cos\theta] \qquad (1.2.1)$$

where S is the flux density of the source and K is a proportionality constant.

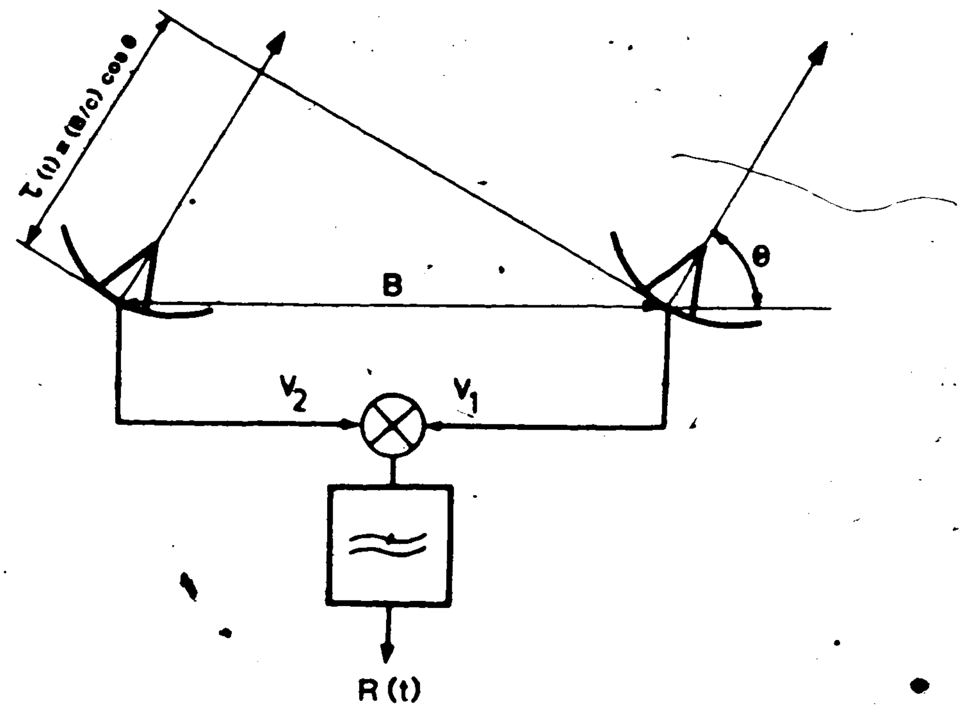


Figure 1.2.1 The Basic Interferometer

The time difference between wavefront arrival at the two antennas, the geometric delay, is a function of θ, the elevation angle. For fixed sources, θ changes with the earth's rotation, hence the geometric delay $\tau(t)$ is a function of time. The response of the basic interferometer is given by

$$R(t) = k_2 \; S \; \cos\left[\frac{2\pi B}{\lambda} \cos\theta \right] \qquad (1.2.2)$$

Figure 1.2.2 shows a schematic working diagram of an interferometer. In addition to the basic interferometer, a variable delay τ_D is inserted on the shorter signal path to equalize the geometric delay at the I.F. A controlled phase shift is injected into the local oscillator signal delivered to one of the antennas. An additional channel, the quadrature channel, is provided by inserting a 90° phase shift into one of the I.F. signals before correlation. Table 1.2.1 shows the expressions for signals at different points of the interferometer, when receiving monochromatic radiation of

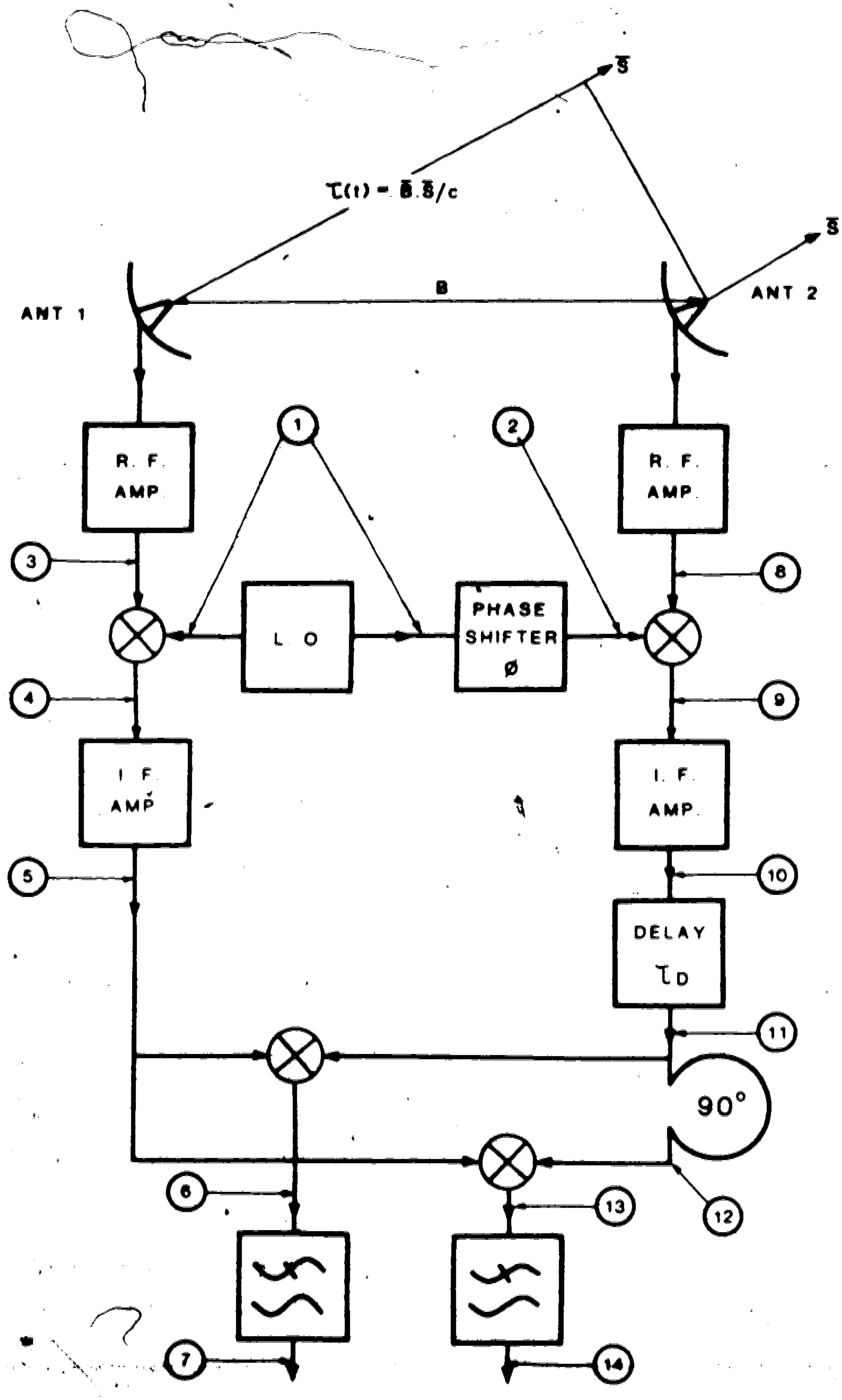


Figure 1.2.2 A schematic working diagram of an interferometer.

(1) $= \cos(\omega_o t)$

(2) $= \cos[\omega_o t + \phi(t)]$

(3) $= \cos \omega[t - \tau(t)]$

(4) $= \cos[\omega t + \omega_o t - \omega\tau(t)] + \cos[\omega t - \omega_o t - \omega\tau(t)]$

(5) $= \cos[(\omega - \omega_o)t - \omega\tau(t)]$

(8) $= \cos(\omega t)$

(9) $= \cos[\omega t + \omega_o t + \phi(t)] + \cos[\omega t - \omega_o t - \phi(t)]$

(10) $= \cos[(\omega - \omega_o)t - \phi(t)]$

(11) $= \cos[(\omega - \omega_o)(t - \tau_D) - \phi(t)]$

(12) $= \cos[(\omega - \omega_o)(t - \tau_D) - \phi(t) - 90°]$

(6) $= \cos\{(\omega - \omega_o)[2t - \tau_D - \tau(t)] - \omega_o\tau(t) - \phi(t)\}$

$+ \cos\{(\omega - \omega_o)[\tau(t) - \tau_D] + \omega_o\tau(t) - \phi(t)\}$

(7) $= \cos\{\omega[\tau(t) - \tau_D] + \omega_o\tau_D - \phi(t)\}$

(13) $= \cos\{(\omega - \omega_o)[2t - \tau_D - \tau(t)] - \omega_o\tau(t) - \phi(t) - 90°\}$

$+ \cos\{(\omega - \omega_o)[\tau(t) - \tau_D] + \omega_o\tau(t) - \phi(t) - 90°\}$

(14) $= \cos\{\omega[\tau(t) - \tau_D] + \omega_o\tau_D - \phi(t) - 90°\}$

Table 1.2.1 Expersions for signals at different points of figure 1.2.1

angular frequency ω. Outputs of the real and quadrature channels are:

$$R_{r_\omega}(t) = \cos\{\omega[\tau(t) - \tau_D] + \omega_o\tau_D - \phi(t)\}$$
$$= \mathrm{Real}\{\exp[i\omega(\tau(t) - \tau_D)]$$
$$\times \exp[i(\omega_o\tau_D - \phi(t))]\} \quad (1.2.3)$$

$$R_{q_\omega}(t) = \sin\{\omega[\tau(t) - \tau_D] + \omega_o\tau_D - \phi(t)\}$$
$$= \mathrm{Im}\{\exp[i\omega(\tau(t) - \tau_D)]$$
$$\times \exp[i(\omega_o\tau_D - \phi(t))]\} \quad (1.2.4)$$

For the broad-band response of the real channel, (1.2.3) is integrated over the passband. Assuming that the pass-band shape is determined by the I.F. system with a shape of $\alpha(\omega)$, the broad-band response is

$$R_r(t) = \int \alpha(\omega - \omega_c)\, e^{i\omega[\tau(t) - \tau_D]}$$
$$\times e^{i[\omega_o\tau_D - \phi(t)]}\, d\omega$$
$$= e^{i[\omega_o\tau_D - \phi(t)]}$$
$$\times \int \alpha(\omega - \omega_c)\, e^{i\omega[\tau(t) - \tau_D]}\, d\omega \quad (1.2.5)$$

where ω_c is the centre of the band.

Substituting $\delta\omega = \omega - \omega_c$

$$R_r(t) = e^{i[\omega_o\tau_D - \phi(t) + \omega_c\delta\tau]}$$
$$\times \int \alpha(\delta\omega) e^{i\delta\omega\delta\tau}\, d(\delta\omega) \quad (1.2.6)$$

where
$$\delta\tau = \tau(t) - \tau_D$$
$$\omega_{IF} = \omega_c - \omega_o$$
$$\beta(\delta\tau) = \int \alpha(\delta\omega)\, e^{i\delta\omega\delta\tau}\, d(\delta\omega) \tag{1.2.7}$$

Thus the broad-band real and quadrature channel responses are

$$R_r(t) = \beta(\delta\tau)\cos[\omega_{IF}\delta\tau + \omega_o\tau_D - \phi(t)] \tag{1.2.8}$$

$$R_q(t) = \beta(\delta\tau)\sin[\omega_{IF}\delta\tau + \omega_o\tau_D - \phi(t)] \tag{1.2.9}$$

The aim of the digital signal processor is to synthesise a real and a quadrature channel output equivalent to (1.2.8) and (1.2.9)

A delay unit has been included in the interferometer in figure 1.2.2 to equalize the signal path difference or geometric delay $\tau(t)$. Equations (1.2.8) and (1.2.9) show the effects of unequalized delay on the response $\beta(\delta\tau)$, the fringe washing function, is the Fourier transform of the passband shape. When the unequalized delay $\delta\tau$ approaches the reciprocal of the bandwidth, $\beta(\delta\tau)$ will be significantly less than unity. Since the geometric delay $\tau(t)$ changes with the diurnal motion, the path compensaton delay τ_D must be variable and track the geometric delay to minimise $\delta\tau$

The variable delay τ_D is sometimes realised by switching into the signal path binary weighted lengths of cables. Since the implementation is discrete in nature, any unequalized delay, $\delta\tau \neq \tau(t) - \tau_D$, will show up in R(t) as phase in the cosine function argument. One of the purposes of the injected phase shift $\phi(t)$ between the local oscillator signals is to cancel out such effects. By controlling the path compensation delay τ_D and phase shift $\phi(t)$ together, the interference fringe pattern could be rotated at will. Fringe derotation or fringe freezing is usually done to counteract the effects of the earth's rotation and fix the fringes on the source. allowing easier instrumentation in correlation and integration.

Without the quadrature channel, the amplitude and phase of the source has to be measured by moving the fringe pattern relative to the source. By adding a 90° phase shifter and a correlator, the quadrature channel allows simultaneous observation of the source with two fringe patterns 90° apart, effectively doubling the amount of information received. The 90° shift could be approximated with a piece of cable at I.F. or realised with a broad-band phase shifter.

2. THE SYNTHESIS TELESCOPE AT DRAO

An earth rotation aperture synthesis telescope has been built at the Dominion Radio Astrophysical Observatory (DRAO) in Penticton B.C., Canada. The instrument is designed primarily for spectroscopic studies of the λ 21 cm line of neutral hydrogen. The telescope combines techniques of aperture synthesis and correlation spectroscopy to provide angular resolution of 1 arc minute over a circular field of view of 2 degrees, and frequency resolution of 128 channels over a variable bandwidth from 0.25 to 4 MHz. A continuum channel of 20 MHz bandwidth is also provided.

In the study of radio sources, the spectral index provides key information concerning the phenomena of emission. Also, with observations made at more than one frequency, it is possible to separate thermal from non-thermal sources in a map[13]. These applications make the addition of a lower frequency channel to the existing synthesis telescope very desirable. It has been proposed by Dr. T. L. Landecker[11] that a 408 MHz continuum channel be added to the existing 1420 MHz synthesis telescope at DRAO. The new continuum channel will give information on spectral index and allow good sensitivity to low surface brightness objects.

Section 2.1 will describe the existing 1420 MHz neutral hydrogen spectroscopic supersynthesis telescope at DRAO. Section 2.2 will briefly describe the 408 MHz continuum channel and explain how it is to integrate with the 1420 MHz system.

2.1 The 1420 MHz System

Discussion of the 1420 MHz system is mainly based on Roger et al.[3] and Dewdney[7]. The digital spectrometer is also included. This section will describe the 1420 MHz system with emphasis on the local oscillator phase rotation system, the switched cable delay and the digital correlators for comparison with the 408 MHz system. Figure 2.1.1 shows the 1420 MHz spectroscopic receiver block diagram for

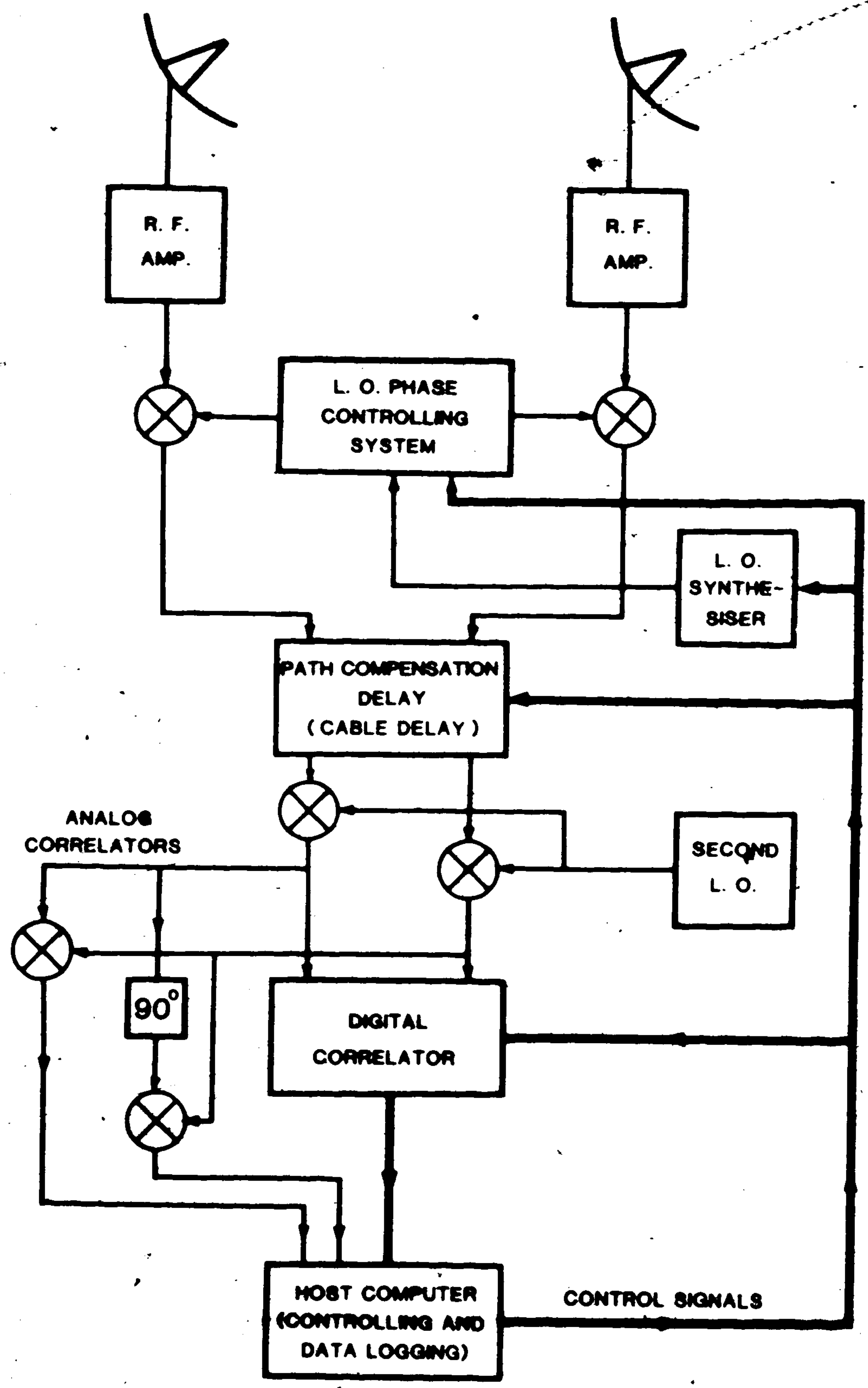


Figure 2.1.1 Receiver block diagram of 1420 MHz system at DRAO

one interferometer.

2.1.1 The Antennas

The antennas of the supersynthesis telescope are four 8.6m paraboloids. These are illuminated with circularly polarised feed horns. The half power beam widths of the antennas are 1.7° and 6° at 1420 and 408 MHz respectively. The two end elements are fixed at 600m distance while the others are movable on a 300m precision track as shown in fig 2.1.2. The antennas are equatorially mounted. The movable elements are supported by three bogies with two resting on the north "master" rail and the third resting on the south "slave" rail. The rails are 6.3m apart. These are lying on concrete ties of 8m length spaced 1.25m apart. Concrete anchors secure the rails at both ends to prevent thermal expansion and contraction. The head of both rails and the south side of the master rail are machined to a tolerance of ±0.5mm. The design of the antenna, supporting structure and precision track are such that the polar axis of the equatorial mount deviates less then 2 arc min at all positions along the track. Thus no adjustment is required after movement of the antennas. The configuration allows four interferometers to be in simultaneous operation between elements 1 and 2, 1 and 3, 2 and 4, and 3 and 4.

2.1.2 The 1420 MHz R.F. System

The incoming signal reflected off the parabolic reflector is collected by the feed horn and amplified by the radio frequency (R.F.) amplifier. The R.F. stage consists of a room temperature parametric amplifier followed by a transistor amplifier with a total excess noise of 54 K. The existing feed horn produces an additional spillover of 25 K. New feed horns with lower spillover and dual frequency capabilities at 1420 and 408 MHz are being developed. For phase stability, the focus boxes which contain the R.F. amplifiers and mixers are held at a constant temperature of 25.0 ±0.2°C with thermal electric heater-coolers.

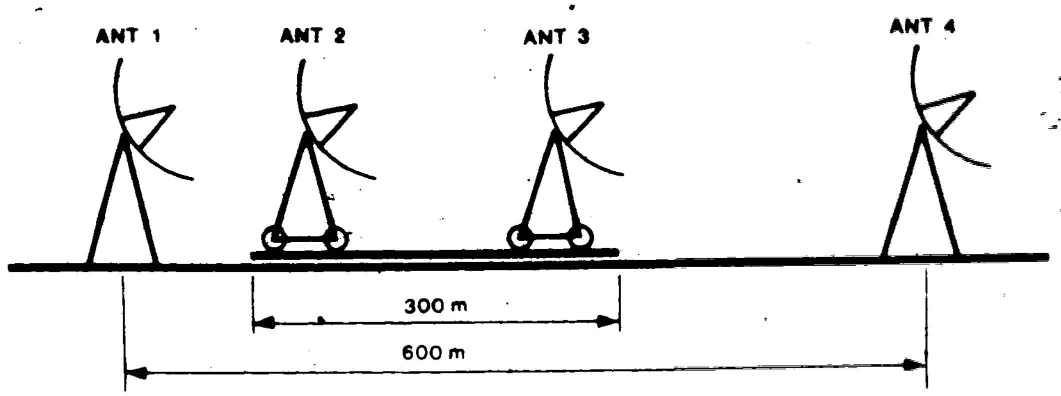


Figure 2.1.2 Antenna element layout. The centre two elements are movable on a precision track. The minimum distance between the left-most two elements is 3 units or 90.7 meters

2.1.3 The 1420 MHz Local Oscillator System.

The first mixers are housed inside the focus boxes of the antennas. Coherent local oscillator signals must be delivered to each of the antennas with high relative phase stablility. Since the electrical length of a coaxial cable is temperature dependent, the high frequency local oscillator signals delivered through long feed cables can give rise to significant phase instability. A closed loop phase controlling system was specially developed to overcome the problem. The closed loop phase controlling system also provides a mechanism of injecting a controlled phase difference ϕ between the L.O. signals delivered to the antennas as required in a conventional interferometer.

Figure 2.1.3 shows the simplified schematics of the local oscillator system. A computer controlled synthesiser generates a master L.O. signal at 695 MHz with a range allowing for Doppler shifts of the Earth's motion and radial velocity of most nearby galaxies. A 2 MHz signal is combined with the master oscillator signal to

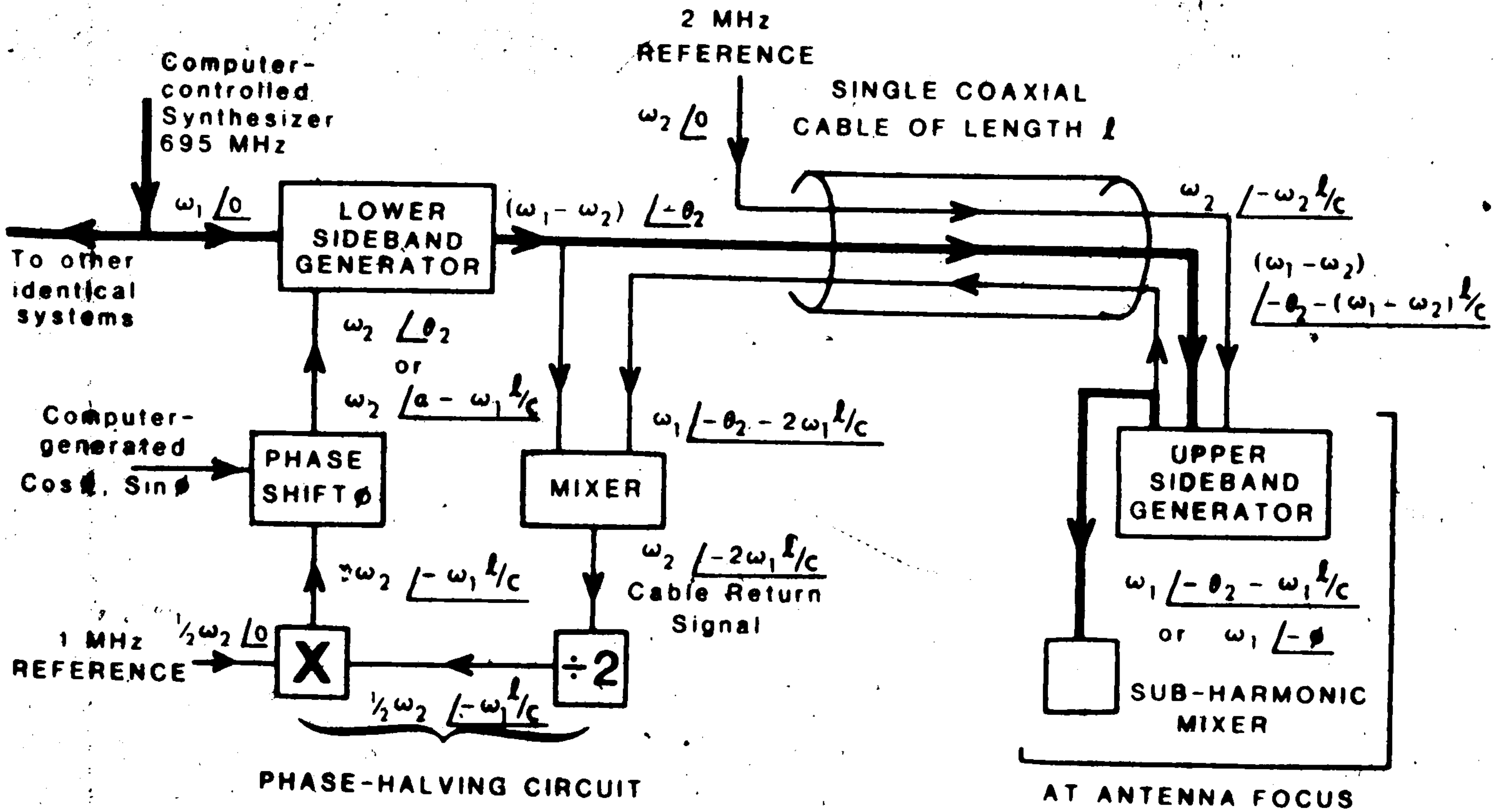


Figure 2.1.3 Local oscillator system of the 1420 MHz receiver

produce a lower sideband that travels down the feed cable together with the 2 MHz signal. The signals are combined in an upper sideband generator in the focus box to generate the 695 MHz L.O. signal for the sub-harmonic mixer. Part of the L.O. signal at the focus box is sent back through the same feed cable. Mixed with the output of the lower sideband generator, this signal produces a 2 MHz signal with a phase shift corresponding to two passages of the L.O. signal through the cable at 695 MHz. Dividing the resultant 2 MHz signal by 2 and mixing it with a 1 MHz reference produces a 2 MHz signal with phase corresponding to a single passage of the 695 MHz signal down the cable. This final 2 MHz signal when used to drive the upper sideband generator, closes the feedback loop and locks the phase of the L.O. signal in the focus box to the master oscillator signal independent of the feed cable.

A phase shift ϕ introduced at the 2 MHz signal entering the lower sideband generator will also appear in the L.O. signal at the focus box. A precision phase shifter made of analog multipliers performs phase shifting using the standard single sideband mixing scheme

$$\cos(\phi + \omega t) = \cos(\omega t)\cos\phi - \sin(\omega t)\sin\phi \qquad (2.1.1)$$

The signals $\sin(\phi)$ and $\cos(\phi)$ are generated by the computer with D/A converters. The precision phase shifter is not compensated in the loop. Any nonlinearity in synthesising equation (2.1.1) will appear as phase errors of the L.O. signal at the focus box. The L.O. signal at the focus box can be controlled to within 1° rms deviation from the desired value.

2.1.4 The Switched Cable Delay

To equalize the geometric delay, a variable delay is inserted in the shorter signal path. Binary weighted lengths of cables are often used to implement a quasi-continuous delay. Double-pole-double-throw diode switches are used to select a length of cable or an equal loss resistive network. Since the electrical length of a cable is a function of temperature, all the delay cables are held at tightly controlled constant temperature. The dispersion, frequency dependent propagation speed, has to be equalized to maintain correct delay over the band. Frequency dependent attenuation is also equalized to preserve the band shape.

The variable delays are inserted at the I.F., which is 20 MHz wide centred on 30 MHz, in the 1420 MHz system. Ten binary weighted lengths of cables from 64 cm to 320 m corresponding to $\lambda_{IF}/16$ to $32\ \lambda_{IF}$ are used. In propagating through the $64\ \lambda_{IF}$ length of cable, the I.F. signal suffers a relative attenuation of 16 db between band edges and an rms phase error of 30°. Equalisers are used for cable sections of $2\ \lambda_{IF}$ and longer, to equalize both frequency dependent attenuation and dispersion.

2.1.5 The Continuum Correlators

Out of the 20 MHz wide I.F. signal, a 4 MHz wide portion at the centre of the band that contains spectral line information is selected for digital spectroscopy. The rest of the spectrum is correlated with analog correlators which form the continuum channel outputs. A broad-band 90° phase shifter is introduced into one arm of the correlator to produce a quadrature, or sine, channel output. The analog correlators are made up of analog multipliers and analog integrators. The integrators are reset every 8 seconds. The output of the integrator is digitised and sampled by the host computer every 8 seconds.

2.1.6 The Digital Spectrometer

The discovery of many radio-frequency lines has made spectral line interferometry necessary. It requires many channels of narrow bandwidth to resolve the complicated frequency (velocity) dependence of the source brightness. The narrow bandwidth channels can be obtained either by filter banks placed across the two input IF lines or by Fourier transformation of the correlation function. With the present digital technology, the digital correlation approach is much preferred. In the 1420 MHz system, a 4 MHz band at the centre of the I.F. signal that contains spectral line information is filtered out and further down mixed to a quasi-baseband signal of

4 to 8 MHz. The quasi-baseband signal is sampled at 16 MHz and quantised into 3 levels. The digital signals are then fed into 256 channel digital crosscorrelators. The output of the digital correlators is sampled every 8 seconds and the data transfered to the host computer. After Fourier transformation, the correlation function will give a 128 point complex power spectral graph of the signal.

2.1.7 The Controller

Like all complicated electronic systems, the synthesis telescope is controlled by a computer. There is, in fact, more than one computer within the supersynthesis telescope, but one of them, the host computer, is responsible for overall system control.

The host computer, a PDP11/23 16-bit minicomputer, controls and coordinates all the operations of the synthesis telescope. The host computer is responsible for calculating the coordinates of the source, and frequency shifts due to motion of the source and the Earth at the beginning of an observation. During the observation, the fringe angle ϕ and path compensation delay are recalculated and updated continuously. The host computer is also responsible for driving the antenna elements and maintaining tracking of the source. Outputs of the 1420 MHz digital spectrometer, the continuum correlators, and the 408 MHz signal processor must be logged by the host computer every 8 seconds. The data are stored on magnetic discs for map production on a larger computer when the series of observations is completed. In the normal mode of operation, the operator's interaction with various subsystems is all done via the host computer. To perform an observation the operator creates a file of a list of instructions and parameters. The host computer interprets the file and carries out the observations automatically according to the list without further operator intervention.

2.2 The New 408 MHz Continuum Channel

As pointed out in the beginning of this chapter, a 4 MHz wide channel at 408 MHz is to be added to the synthesis telescope. The design of the new system is aimed at simplicity and low cost.

2.2.1 The 408 MHz Analog System

The 408 MHz analog system and the dual frequency feeds of the antenna are being developed by B.G. Veidt[14] at the time of writing. The analog system shown in figure 2.2.1 includes the radio frequency amplifier (R.F.Amp), the local oscillator (L.O.) system and intermediate frequency (I.F.) amplifiers. Commercial transistor amplifiers will likely be used in both R.F. and I.F. stages. The local oscillator system being developed by Veidt is much simpler than the 1420 MHz L.O. system. In the 408 MHz system, the L.O. signal and IF signals will be transported via the same coaxial cable in opposite directions. In the system shown in figure 2.2.2, the phases of the L.O. signal and the returned I.F. signal will be compensated and are, to the first order, independent of the cable length. IF signals are further down mixed to baseband for the 408 MHz digital signal processor.

2.2.2 The 408 MHz Digital Signal Processor

The scope of this M.Sc project covers the design and implementation of the 408 MHz digital signal processor (DSP). The DSP accepts 4 MHz baseband signal and produces outputs corresponding to the real and quadrature channels of a conventional interferometer. The functions of delay equalisation and fringe derotation are also performed within the DSP. Figure 2.2.3 shows the simplified signal flow diagram for one interferometer with emphasis on the DSP. Analog baseband signals are sampled at 16 MHz and quantised into 3 levels with the quantiser. A digital delay unit performs coarse delay equalisation. The signals are then cross-correlated digitally, producing a 16 point correlation function. The correlation function is then read into the microcomputer. All further signal processing is done in software. The sampled correlation function is interpolated to obtain the exact delay value for the real channel. The quadrature channel is generated by performing a Hilbert transform on

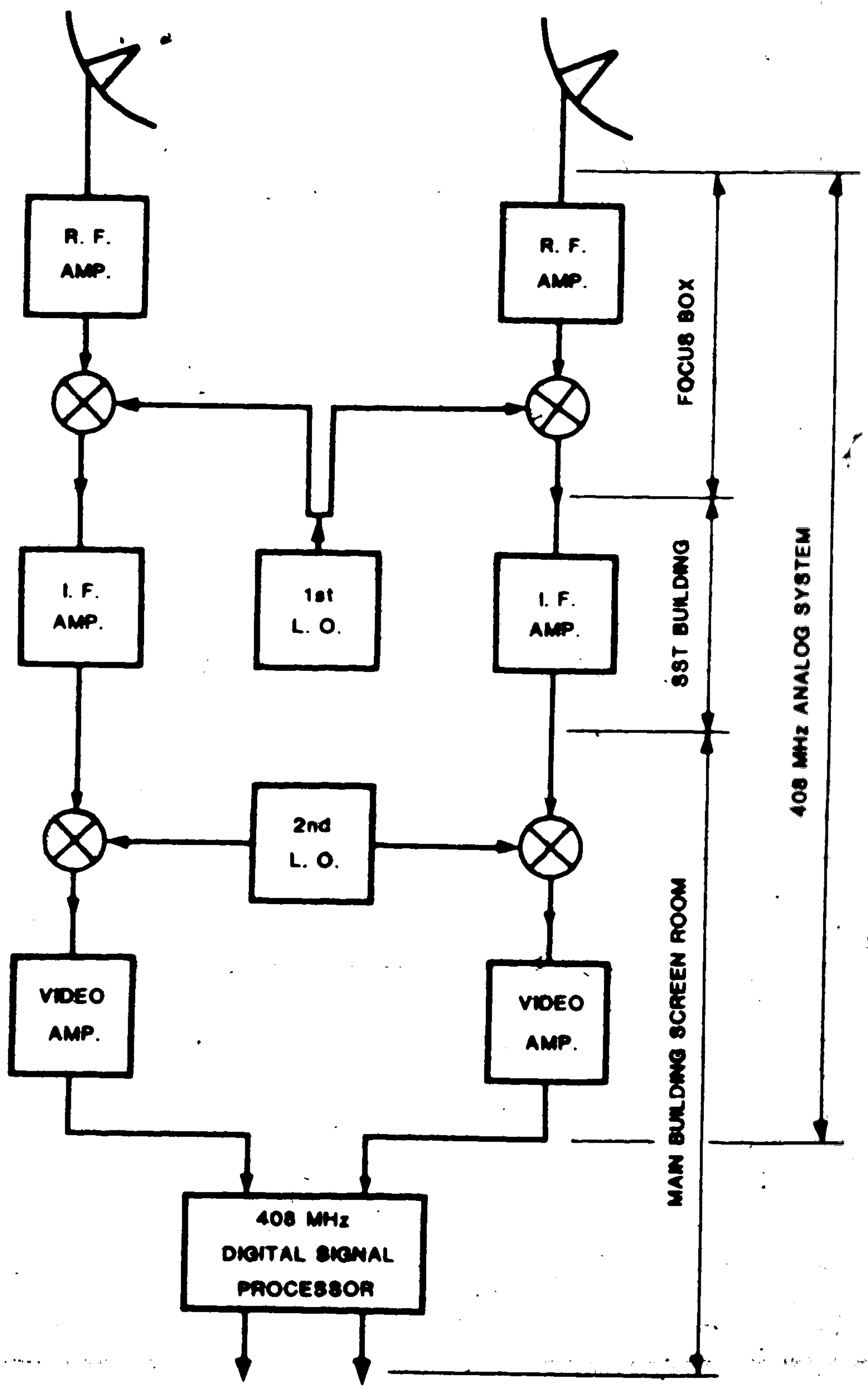


Figure 2.2:1 The 408 MHz Telescope

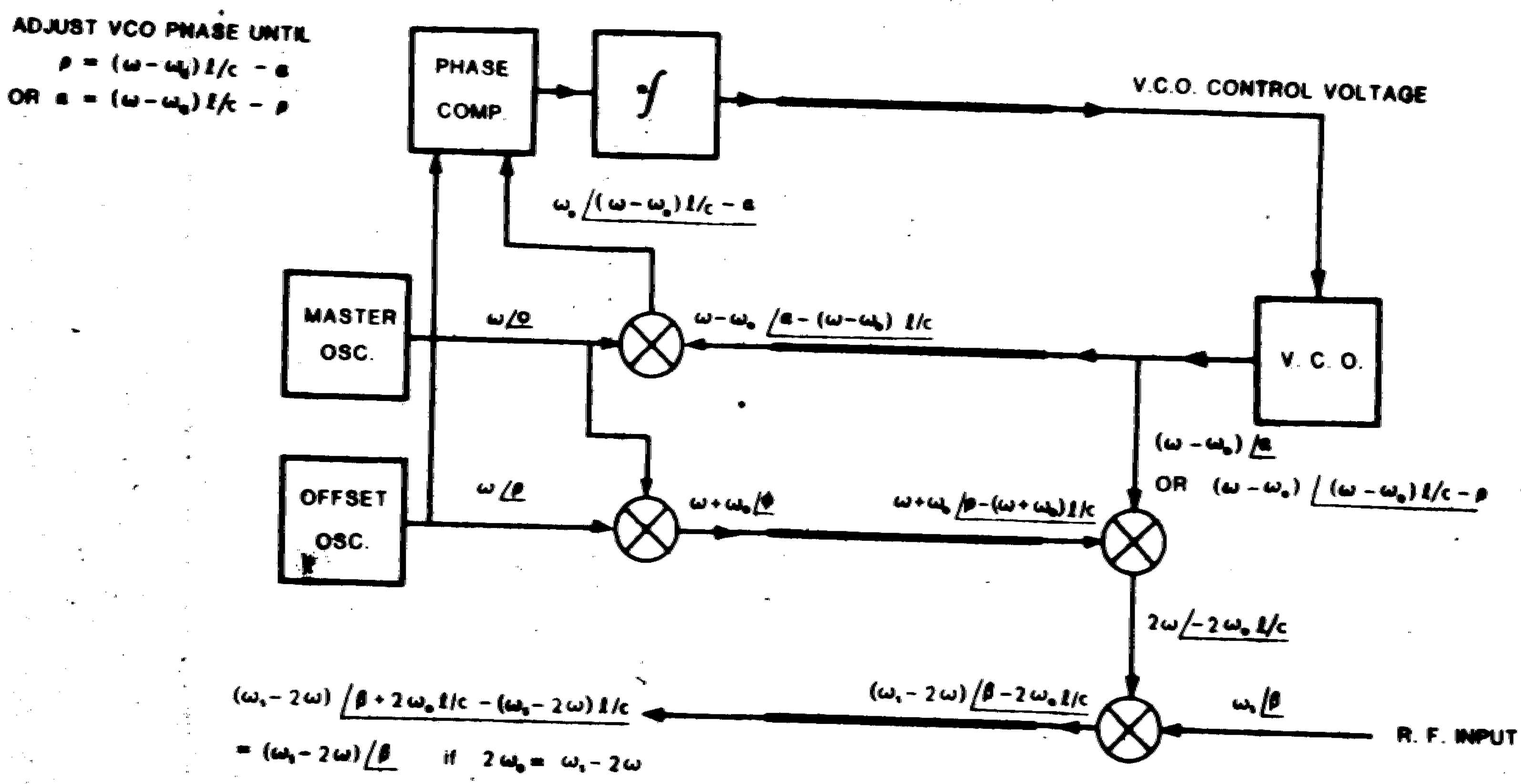


Figure 2.2.2 L.O. system of the 408 MHz receiver system (Landecker and Veidt)

the correlation function. Finally, fringe derotation is done by numerically synthesising the single side band down-mixing equation. The theory of operation of the DSP will be discussed in detail in chapter 3.

One of the main design criteria of the 408 MHz system is to share as many existing subsystems with the 1420 MHz system as possible or to use straight replications to minimise system development time and cost. Eventually the 408 MHz system will integrate with the 1420 MHz system, sharing the antennas, the controlling host computer and the off line map production software.

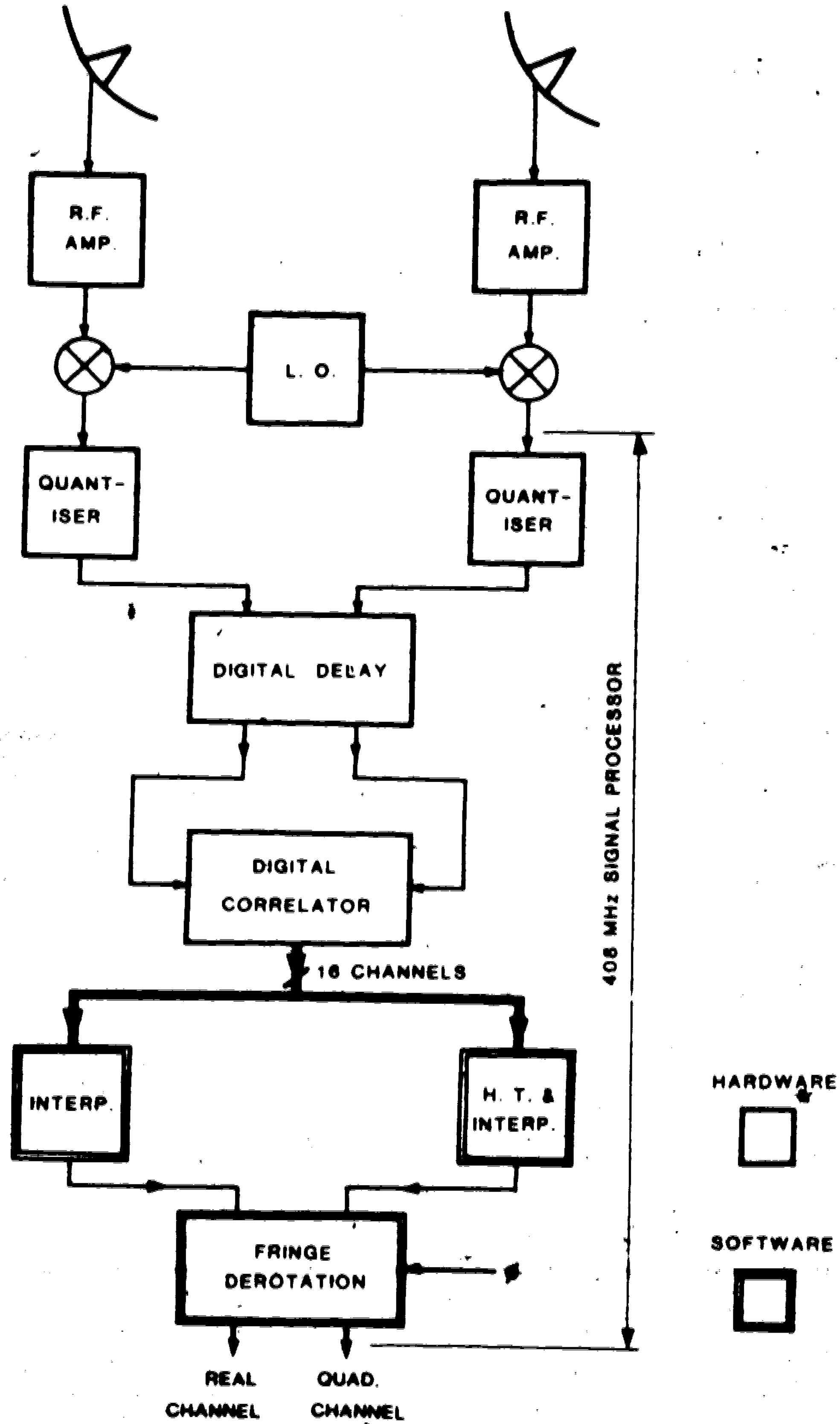


Figure 2.2.3 System block diagram of the digital signal processor.

3. THEORY OF OPERATION OF THE 408 MHz DIGITAL SIGNAL PROCESSOR

3.1 Digital Correlation

The correlation function provides a useful tool in the study of random variables. Mathematically, the crosscorrelation function is defined as

$$Rx_1x_2(\tau) = \overline{x_1(t)\ x_2(t + \tau)} \tag{3.1.1}$$

where $x_1(t)$ and $x_2(t)$ are stationary signals and the bar denotes statistical average. The correlation function can also be expressed as a time average if the signals $x_1(t)$ and $x_2(t)$ are ergodic.

$$Rx_1x_2(\tau) = \lim_{T\to\infty} \frac{1}{T} \int_{-T/2}^{T/2} x_1(t)\ x_2(t + \tau)\ dt \tag{3.1.2}$$

The autocorrelation function can be defined as

$$Rxx(\tau) = \lim_{T\to\infty} \frac{1}{T} \int_{-T/2}^{T/2} x(t)\ x(t + \tau)\ dt \tag{3.1.3}$$

The autocorrelation function Rxx is maximum at $\tau = 0$. A normalised autocorrelation function can be defined as:

$$\rho_x(\tau) = \frac{Rxx(\tau)}{Rxx(0)} \tag{3.1.4}$$

The correlation function $Rx_1x_2(\tau)$, besides being a useful mathematical tool, can be estimated at discrete values of τ with relatively simple hardware, for stationary and ergodic random signals $x_1(t)$ $x_2(t)$. Only a statistical estimate could be obtained since the definition of the correlation function calls for infinite time integration.

Figure 3.1.1 shows the structure of a crosscorrelator. Two streams of signals are cross-multiplied and integrated. The outputs of the integrators are:

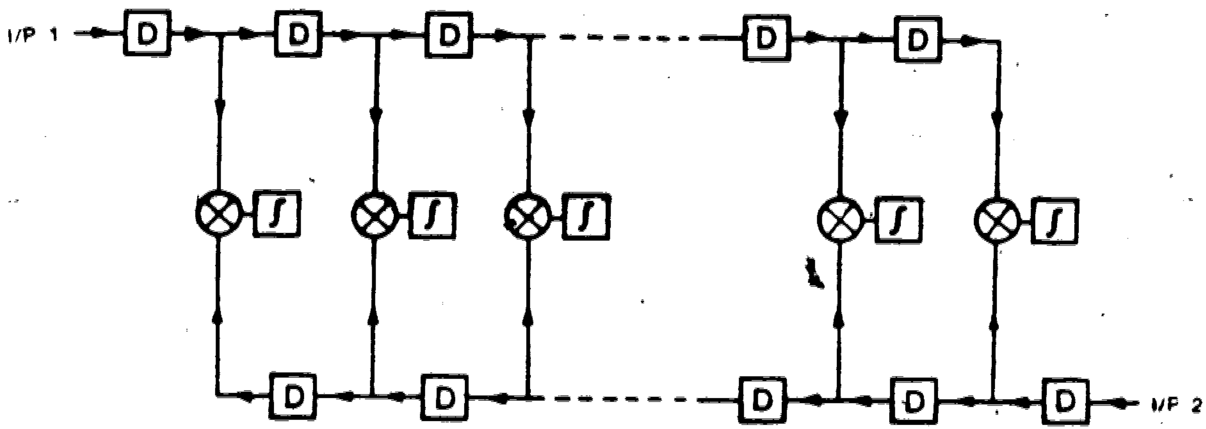


Figure 3.1.1 Structure of a crosscorrelator

$$Rx_1x_2(nD) = \frac{1}{T}\int_0^T x_1(t-nD)\; x_2[t-(N-n+1)D]\; dt \qquad (3.1.5)$$

where D is the delay time of each element

n is the position of the integrator from left to right numbering from 1 to N

A correlation function $Rx_1x_2(nD)$ sampled in the delay domain could be obtained from the correlator. Either digital or analog circuitry can be used to implement the correlator. Analog correlators provide wide bandwidth and higher S/N ratio but require analog delays and high speed precision multipliers that are expensive and less convenient to build in large numbers.

In digital signal processing, the signals have to be sampled in discrete time and quantised into discrete values. Digital correlators are practical because very coarse quantisation can be used which greatly simplifies the multipliers and accumulators. Van Vleck and Middleton[4] have shown that the effect of quantisation of band limited Gaussian noise is to spill part of the energy out of the band limits. Such effects are not very severe, if the S/N ratio is already very low, even at the extreme case of two level, or one bit quantisation. It was further shown[4] that the normalised autocorrelation function of the two level quantised and unquantised signal are related by the Van Vleck relationship:

$$\rho_x(\tau) = \sin [(\pi/2) \rho_x'(\tau)] \tag{3.1.6}$$

where ρx is the normalised autocorrelation function of the band limited Gaussian noise $x(t)$.

$\rho x'$ is the normalised autocorrelation function of the two level quantised signal $x'(t)$

With equation (3.1.6), a one bit, or two level quantisation, correlator can effectively compute the normalised autocorrelation function of the unquantised, band-limited noise. Relations between normalised autocorrelation functions of 3 level quantised noise and unquantised noise are given by Dewdney[7]

In radio astronomy applications, the digital correlator is often used to correlate two signals which are deeply buried in noise. Figure 3.1.2 shows a model of the application. $n_1(t)$ and $n_2(t)$ are Gaussian noise produced by antennas and receivers. $x_1(t)$ and $x_2(t)$ are correlated signals from the same radio source, also with Gaussian distributions but usually much smaller than $n_1(t)$ and $n_2(t)$ in amplitude.

Weinreb[5] has shown that for a finite time extent signal $x(t)$, the variance, or statistical fluctuation, of the one bit correlation function $\rho x'(\tau)$ is larger than the variance of the unquantised correlation function $\rho x(\tau)$. Bowers and Klingler[6] have developed the work to cover digital correlation of weak signals in uncorrelated noise using different schemes. The increase of statistical fluctuation decreases the S/N

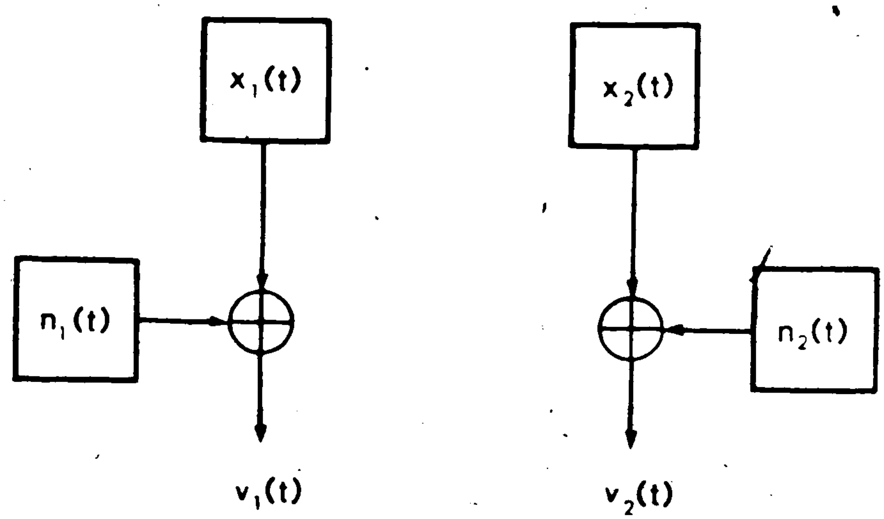


Figure 3.1.2 Model of received astronomical signals

ratio in terms of detecting the small amplitude correlated signal among the large receiver plus antenna noise. A factor of degradation of S/N ratio can be defined as

$$D = \frac{\text{Output S/N Ratio of Perfect Analog Correlator}}{\text{Output S/N Ratio of Digital Correlator}} \tag{3.1.7}$$

Since the S/N ratio is inversely proportional to the square root of the integration time[15], degradation due to coarse quantisation could be compensated with a longer integration period. Bowers and Klingler[6] and Cooper[8] showed that two to five levels of quantisation is a good compromise between hardware complexity and a high degradation factor. The degradation factor for two level and 3 level quantisation is 1.57 and 1.24 respectively. Increasing the sampling rate beyond the Nyquist rate[6] was also shown to reduce the degradation factor. The scheme of 3 level by 3 level quantisation with twice the Nyquist sampling rate is used for the digital correlator in the spectrometer of the 1420 MHz system at DRAO. A modified version of the correlator will be used in the 408 MHz continuum correlation. The degradation factor for the 3 level by 3 level correlator is 1.14 at twice the Nyquist sampling rate. The decision levels of the quantizers are $\pm 0.6\ \sigma$ where σ is the rms noise level. In the 1420 MHz system, the maximum correlation produced by the strongest source in the sky is about 20%. A relation similar to (3.1.6) is given by Dewdney[7] for the 3 level by 3 level correlator. The relationship indicated that at 20% correlation, the nonlinearity produced is negligible, and thus no correction similar to the Van Vleck relation is required.

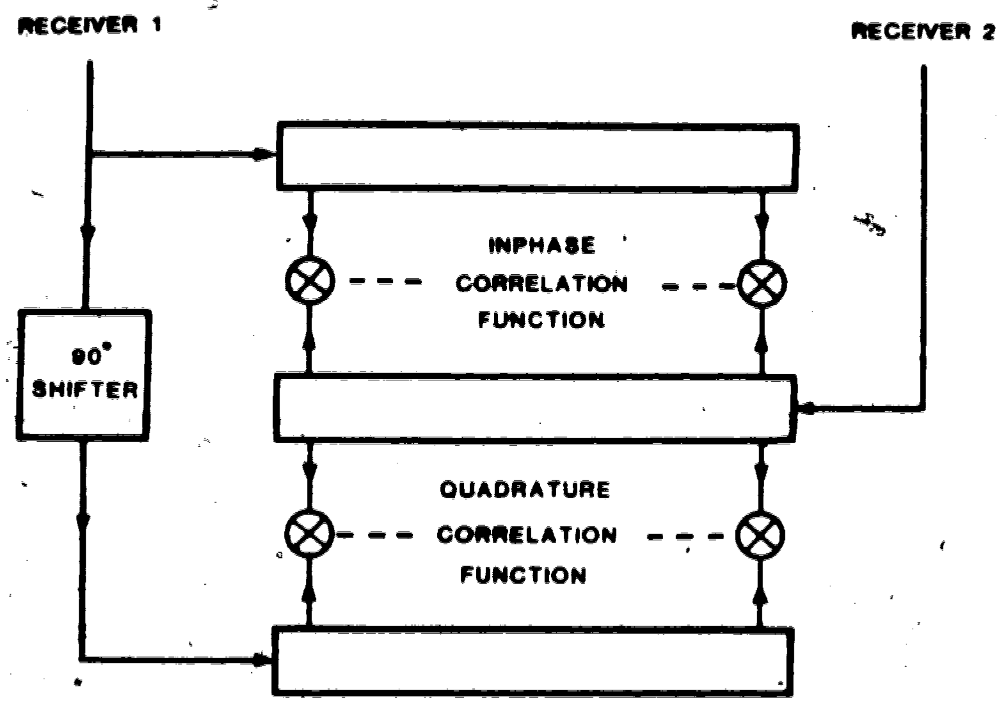
3.2 Quadrature Channel Generation.

The digital signal processor employs digital correlators of similar design to those in the 1420 MHz system although only continuum outputs are required. The output from the correlator provides more information than just spectral shape through the Fourier transform. An important fact is that the quadrature channel could also be generated from the correlation function[10]. The author has developed an algorithm to perform quadrature channel generation and interpolation with a single operation.

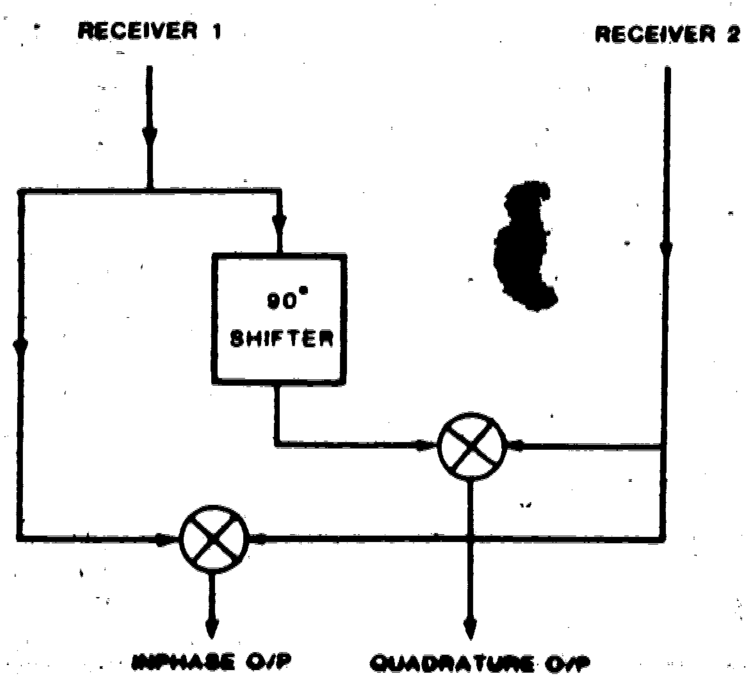
The conventional method of generating the quadrature channel is to insert into one of the signal paths a broadband 90° phase shifter as shown in fig 3.2.1. Mathematically, the 90° phase shifter is represented by a Hilbert transform. The Hilbert transform is defined in the frequency domain as

$$H(f) = \begin{cases} -j, & f < 0, \\ +j, & f > 0 \end{cases} \tag{3.2.1}$$

H(f) operates on the input spectrum by rotating the positive frequency components through −90° and the negative frequency components through +90°. In the time domain, the Hilbert transform is equivalent to convolving the input function with the



(b) DIGITAL CORRELATOR



(a) ANALOG CORRELATOR

Figure 3.2.1 Conventional ways of generating quadrature channel kernel of the Hilbert transform, which is the time domain function h(t) as shown in figure 3.2.2. The Hilbert transform kernel is:

$$h(t) \quad = \quad \mathcal{F}^{-1}\left\{H(f)\right\} = -1/(\pi t) \qquad (3.2.2)$$

From linear system theory the input and output of the Hilbert transform are related in the time domain by the convolution integral:

$$y(t) \quad = \int_{-\infty}^{\infty} x(t-\alpha)\ h(\alpha)\ d\alpha \qquad (3.2.3)$$

where y(t) is the Hilbert transform of x(t). Multiplying the conjugate of (3.2.3) by x(t - τ) and taking the expectation value and reversing the order of integration gives [18]:

$$E[\ y^*(t)\ x(t-\tau)\ ] = \int_{-\infty}^{\infty} E[\ x(t-\tau)\ x^*(t-\alpha)]\ h^*(\alpha)\ d\alpha$$

$$Rxy(\tau) \quad = \int_{-\infty}^{\infty} Rxx(\tau-\alpha)\ h^*(\alpha)\ d\alpha$$

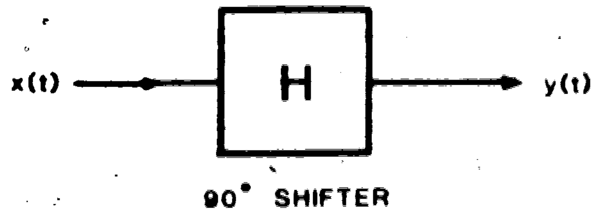
$$= \quad H'\{\ Rxx(\tau)\ \}. \qquad (3.2.4)$$

where Rxy(τ) is the cross-correlation of x and y, Rxx(τ) is the autocorrelation of x, and H' is a system with impulse response $h^*(t)$. Since h(t) is real: $h^*(t)$ = h(t) and H' = H. Therefore (3.2.4) becomes:

$$Rxy(\tau) \quad = H\ \{\ Rxx(\tau)\ \}. \qquad (3.2.5)$$

which states that the quadrature autocorrelation function could be obtained by applying the Hilbert transform to the inphase correlation function.

The next step is to generalise (3.2.5) to the crosscorrelation of received signals. From figure 2.1.1 the received signals $v_1(t)$ and $v_2(t)$ could each be



$$y(t) = \int_{-\infty}^{\infty} x(t-a)\,h(a)\,da$$
$$= H\,\{x(t)\}$$

Figure 3.2.2 Mathematical representation of the Hilbert transform.

decomposed into a source component x(t) and a receiver and sky noise component n(t)

$$v_1(t) = x_1(t) + n_1(t) \tag{3.2.6}$$

$$v_2(t) = x_2(t) + n_2(t) \tag{3.2.7}$$

where $x_1(t)$ and $x_2(t)$ are signals from the same source in the sky. Assuming perfect geometric delay equalisation, $x_1(t) = x_2(t) = x(t)$. The crosscorrelation of v_1 and v_2 is

$$Rv_1v_2(\tau) = Rxx(\tau) + Rxn_1(\tau) + Rxn_2(\tau) + Rn_1n_2(\tau) \tag{3.1.8}$$

From the definition of correlation function, the integration is carried out from negative infinity to infinity in time. Since the receiver noise $n_1(t)$ and $n_2(t)$ are independent of each other and of the signal x(t),

$$Rxn_1(\tau) = Rxn_2(\tau) = Rn_1n_2(\tau) = 0. \tag{3.2.9}$$

Therefore

$$Rv_1v_2(\tau) = Rxx(\tau). \tag{3.2.10}$$

Similarly, the crosscorrelation of v_1 and v_2 is:

$$R\hat{v}_1v_2(\tau) = Rxy(\tau) \tag{3.2.11}$$

where $\hat{v}_1(t)$ is the Hilbert transform of $v_1(t)$ and $y(t)$ is the Hilbert transform of $x(t)$. Combining (3.2.10), (3.2.11) and (3.2.5) gives:

$$R\hat{v}_1v_2(\tau) = H[\,Rv_1v_2(\tau)\,] \tag{3.2.12}$$

which is equivalent to (3.2.5) but includes the receiver noise.

The effects of finite integration period have been pointed out in section 3.1. $Rxn_1(\tau)$, $Rxn_2(\tau)$, and $Rn_1n_2(\tau)$ will not be identically zero and will appear as noise superimposed on the correlation function $Rxx(\tau)$. The effect of errors in geometric delay equalisation is to introduce a shift of $Rxx(\tau)$ in the delay domain to $Rxx(\tau + \delta\tau)$, where $\delta\tau$ is the unequalized geometric delay.

The Hilbert transform could be approximated numerically in many ways. A straightforward approach is to transform into the frequency domain, multiply by $+j$ and $-j$ appropriately, and inverse transform back into the τ domain with the aid of the FFT algorithm. Another approach makes use of the convolution theorem. The input function Rxx is convolved with the kernel, or impulse response h(t), of the Hilbert transform transfer function. In numerical convolution difficulties may arise since h(t) contains a pole and a discontinuity at the origin. A way to get around the difficulty is to band limit the definition of H(f). Since the crosscorrelator can produce only a finite number of output channels, the correlation function $Rxx(\tau)$ is

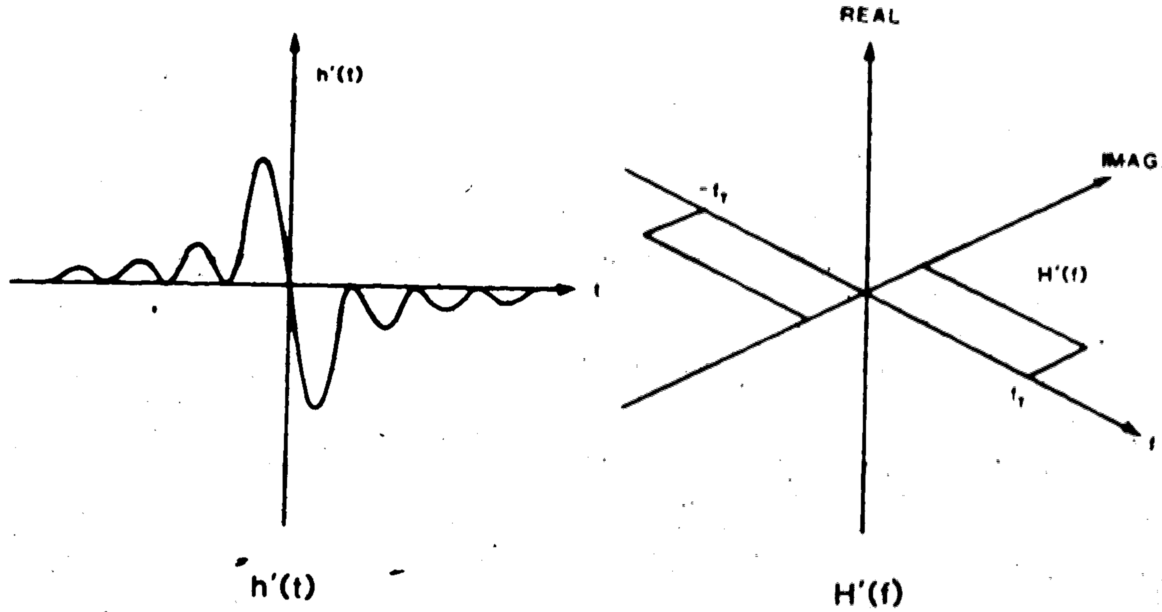


Figure 3.2.3 One of The Modified Hilbert Transforms

sampled in the τ domain and is therefore band limited. Truncation of H(f) to H'(f) above the band limit of the correlation function should therefore not affect the result of the transformation. Figure 3.2.3 shows one way of bandlimiting the Hilbert Transform by simple truncation. In doing so the kernel or impulse response of the modified Hilbert transform h'(t) contains no pole or discontinuity, as shown in figure 3.2.3. Other kinds of band limiting with smooth roll off are discussed in section 3.3.

In this particular application, the interest lies in producing Rxx and Rxy at only one particular value of τ corresponding to the real and quadrature outputs of the continuum correlators as in figure 3.2.1(a). In performing a numerical modified Hilbert transform for one value of τ by convolution, the computation degenerates to just the dot product of two vectors, which is much more economical to compute than the FFT algorithm. Simulation studies of the crosscorrelator with the Amdahl Computer of the University of Alberta have shown that both of the methods produce very satisfactory results. Details of the simulation are described in section 4.1.

3.3 Digital Simulation of Continuous Delay

3.3.1 Interpolation of Correlation Function

In figure 1.2.2 a variable delay τ_D is required in the shorter signal path to equalize the geometric delay $\tau(t)$. Associated with each crosscorrelator is a digital delay unit which could be switched into either one of the signal paths (figure 3.3.1).

The problem arises with the quantisation steps of the digital delay. In conventional implementation, loops of cable are switched in to vary the length of signal path. The increment could be as small as $\lambda_{IF}/16$ as in the 1420 MHz system. With digital delay, the signals are sampled at fixed time intervals in the quantisers, resulting in quantisation of delay time to one sample period. In terms of the correlator output, this means we cannot sample at the exact value of τ, say τ_o, which we want. Interpolation will have to be used to obtain $Rxx(\tau_o)$ and $Rxy(\tau_o)$ from the sampled correlation functions $Rxx(kT)$ and $Rxy(kT)$.

In the generation of Rxy from Rxx by convolution, the modified Hilbert transform performs the Hilbert transform and low pass filtering. If the modified Hilbert transform in figure 3.2.3 is used any frequency components above the truncation frequency f_T will not pass through. If the truncation frequency f_T is chosen to be 1/2T, where T is the sampling period of the correlation function Rxx, then the convolution with $h'(t)$, the truncated Hilbert transform kernel, will perform interpolation as well as the Hilbert transformation. Following the same argument, convolution of $Rxx(kT)$ with $\mathrm{sinc}(t/T)$ will also result in perfect recovery of intermediate values. Simulation results have shown very good recovery of intermediate values of a crosscorrelation function sampled at 16 points at twice the Nyquist rate in the τ domain (see section 4.2).

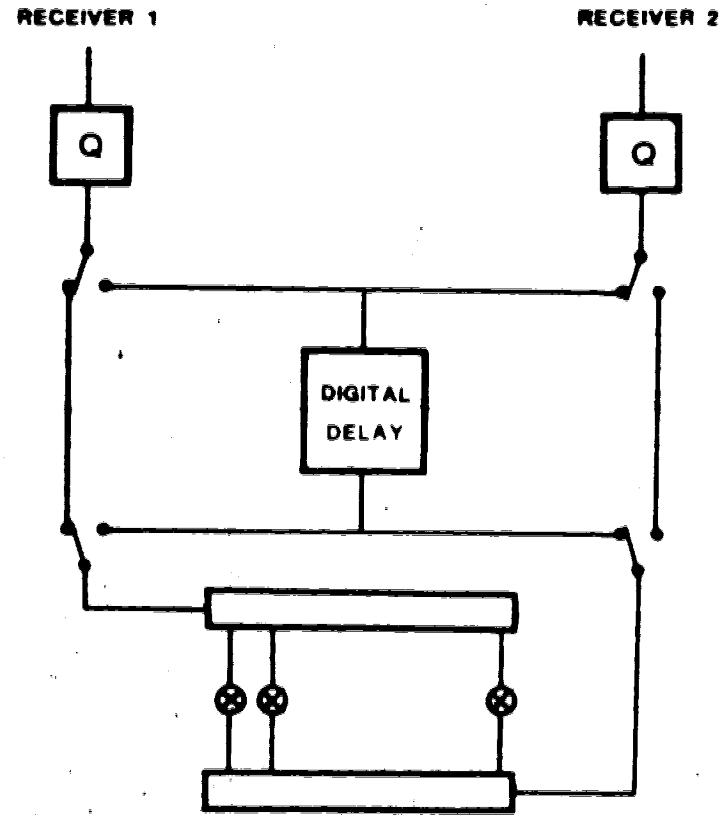


Figure 3.3.1 : The Digital Delay Unit and Crosscorrelator Arrangement

In a conventional interferometer, unequalized geometric delay will produce slight decorrelation, represented by a fringe washing function, and a phase effect at the I.F. as shown in equations (1.2.8) and (1.2.9). The 1420 MHz system has to cancel the phase effect with the controlled local oscillator phase difference $\phi(t)$, whenever a delay switching is done. In the 408 MHz system, the delay is essentially continuous, limited in resolution only by the word length used in interpolation arithmetic. The delay value used is updated every minor integration period resulting in, effectively exact delay equalisation. Thus no phase compensation is required when even the coarse delay is switched.

3.3.2 Choice of Interpolation Functions

From the Wiener-Kinchine Theorem the power spectral density function and the autocorrelation function are related by the Fourier Transform. Sampling the correlation function corresponds to repeating the power spectral density function.

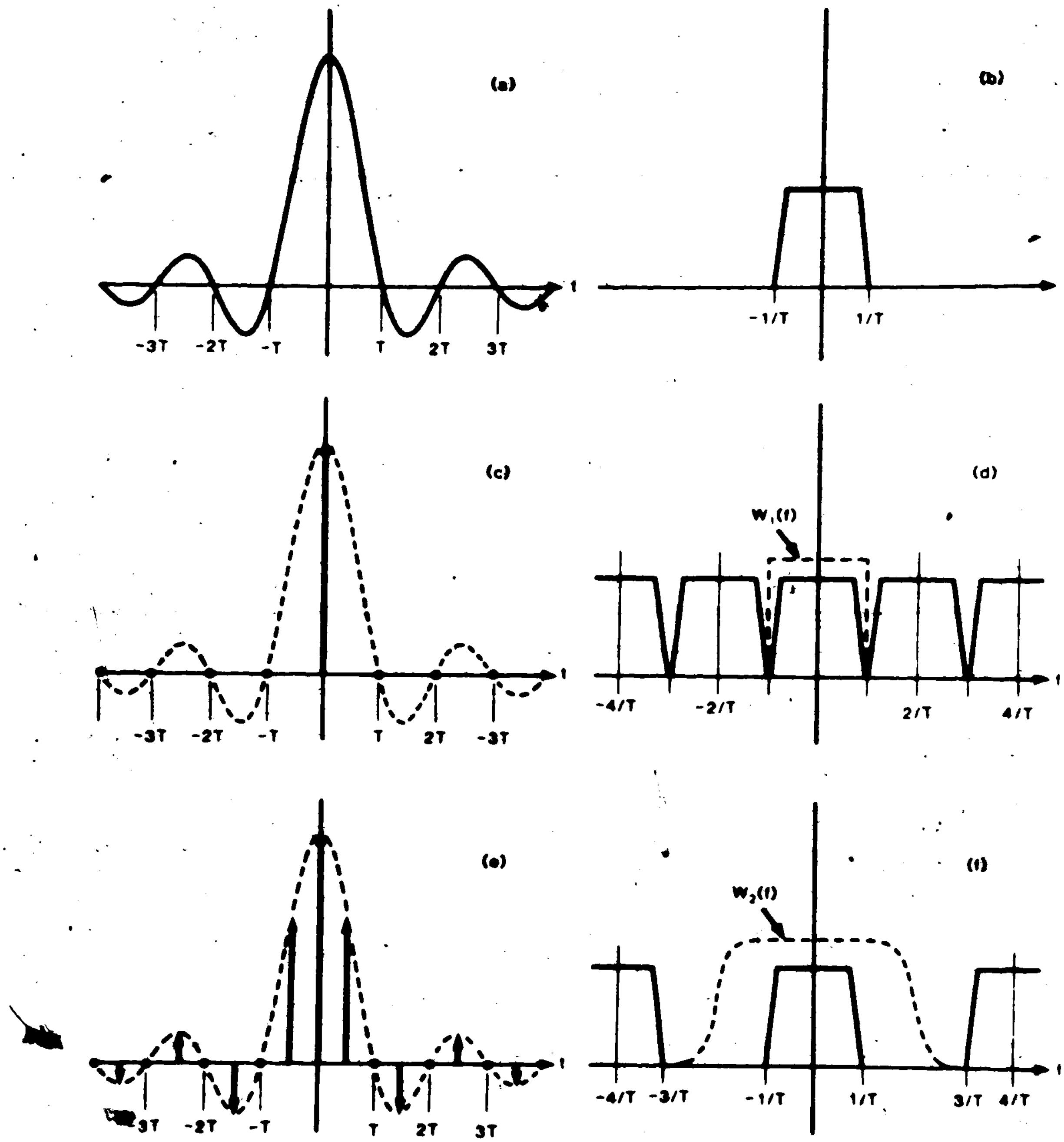


Figure 3.3.2 The choice of window function and correlation function sampling rate.

Interpolating the sampled correlation function by convolution is equivalent to multiplying the repeated power spectral density function with a window function. If the correlation function is sampled in the τ domain at the Nyquist rate, the interpolation function in the frequency domain should be a square window $W_1(f)$ just covering the original power spectral density function, as shown in figure 3.3.2(d). Windows other than the rectangular function will alter the shape of the power spectral density function and fail to recover the unsampled correlation function. The interpolation function in the time domain is:

$$w_1(t) = \mathfrak{F}^{-1}\left[W_1(f)\right] \qquad (3.3.1)$$

If the correlation function is sampled at more than the Nyquist rate, a window with gradual roll off, as in figure 3.3.2(f) could be used. Such a window, $W_2(f)$, results in lower sidelobes of the time domain interpolation function $h_2(t)$. Interpolation functions with quickly decaying sidelobes are preferred because the correlation function is available over a only finite extent of τ limited by the number of channels of the correlator.

To obtain the quadrature channel at a particular value of delay τ_s, the correlation function has to be Hilbert transformed and then interpolated. In the frequency domain, this corresponds to multiplying the power spectral density by $H(f)$ and a window funtion $W(f)$,

$$\begin{aligned} Pxy(f) &= Pxx(f)\, H(f)\, W(f) \\ &= Pxx(f)\, H'(f) \end{aligned} \qquad (3.3.2)$$

where $Pxx(f)$ = Power spectral density of incoming signal $x(t)$

$Pxy(f)$ = Cross Power spectral density of $x(t)$ and $y(t)$

$H'(f)$ = $H(f)\, W(f)$

= bandlimited Hilbert Transform in frequency domain

Since $H'(f)$ is the Hilbert Transform of $W(f)$, $h'(t)$ is the Hilbert transform of $w(t)$, which is the time domain window function. In the time domain,

$$Rxy(\tau) = Rxx(\tau) * h'(t) \tag{3.3.3}$$

$$\text{where } h'(t) = \mathfrak{F}^{-1}\{H'(f)\}$$

$$= \mathfrak{F}^{-1}\{H(f)W(f)\}$$

which states that the operations of Hilbert transformation and interpolation could be performed with a single convolution. Figure 3.3.3 shows the rectangular and 50% raised cosine window functions and their Hilbert transform in both the frequency and the time domain. Analytical expressions of $w_2(t)$ and $h_2(t)$ of figure 3.3.3(e) and 3.3.3(f) are derived in appendix I.

3.3.3 Effects of Finite Correlator Length

The time domain interpolation functions shown in figure 3.3.3(b), (d), (f), (h) are of infinite extent. Since the correlation function is available over only a finite extent of τ, errors will be introduced in interpolation. Fortunately the correlation function decays towards zero with a rate of at least $1/\tau$ outside a central portion.

The correlation function is a superposition of the fringe washing function shifted in τ and modified by the source brightness and antenna response. With a primary half power beam width of 7° at 408 MHz, the maximum difference in delay between the beam centre and beam edge is only 2 delay units [1]. When pointed at an extented source of uniform brightness, the correlation function beyond the maximum delay difference decays towards zero as fast as the fringe washing function. In the absence of a strong source close to the primary antenna lobe, the correlation function could be expected to decay towards zero properly.

---

[1]one delay unit is one 1/16 of a microsecond

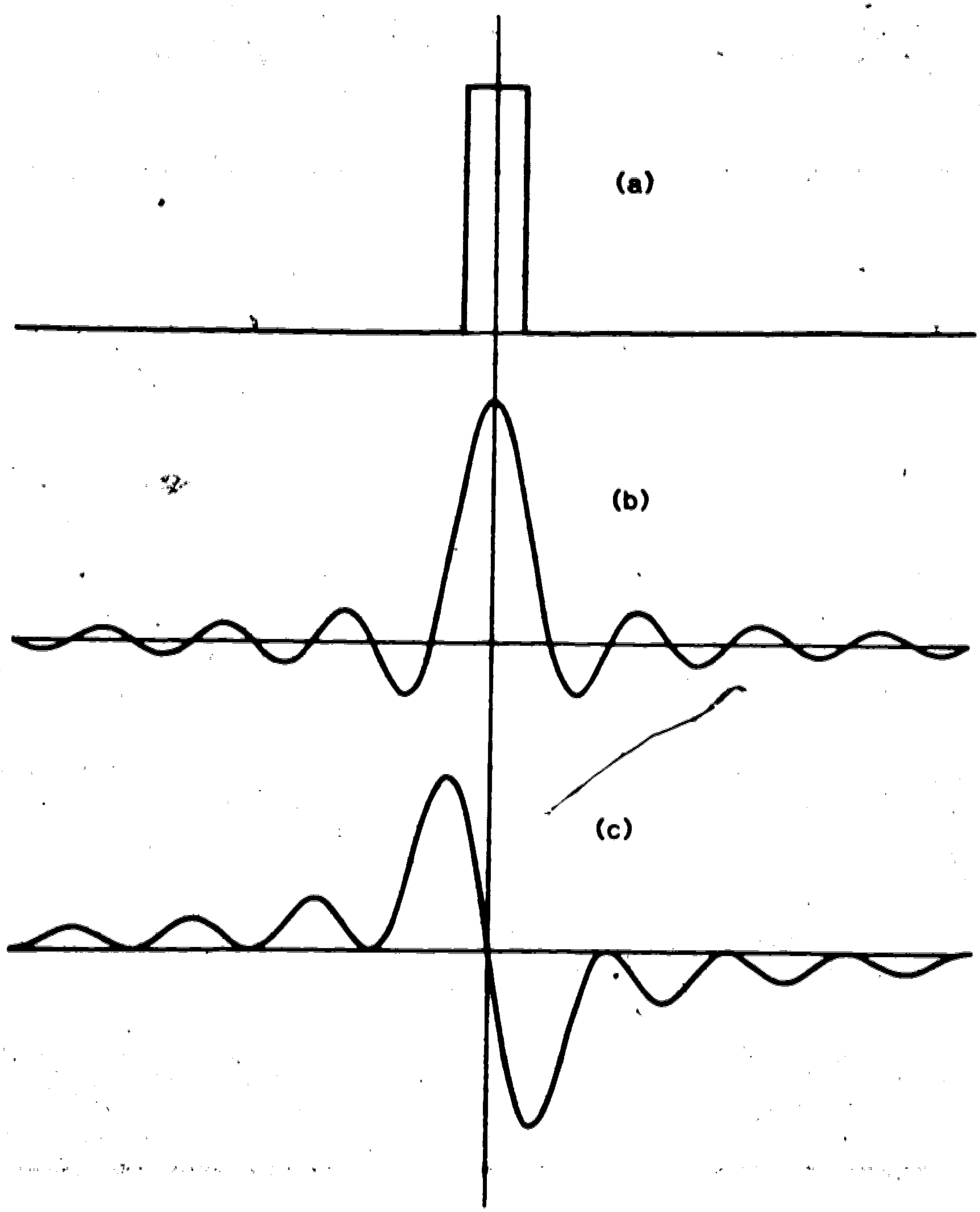


Figure 3.3.3 The frequency domain window functions and time domain interpolation functions. Figure (a) is the rectangular window $W_1(f)$. Figure (b) is the time domain interpolation function $w_1(t)$, which is the Fourier transform of (a). Figure (c) is the modified Hilbert transform kernel $h_1(t)$, which is the Hilbert transform of (b).

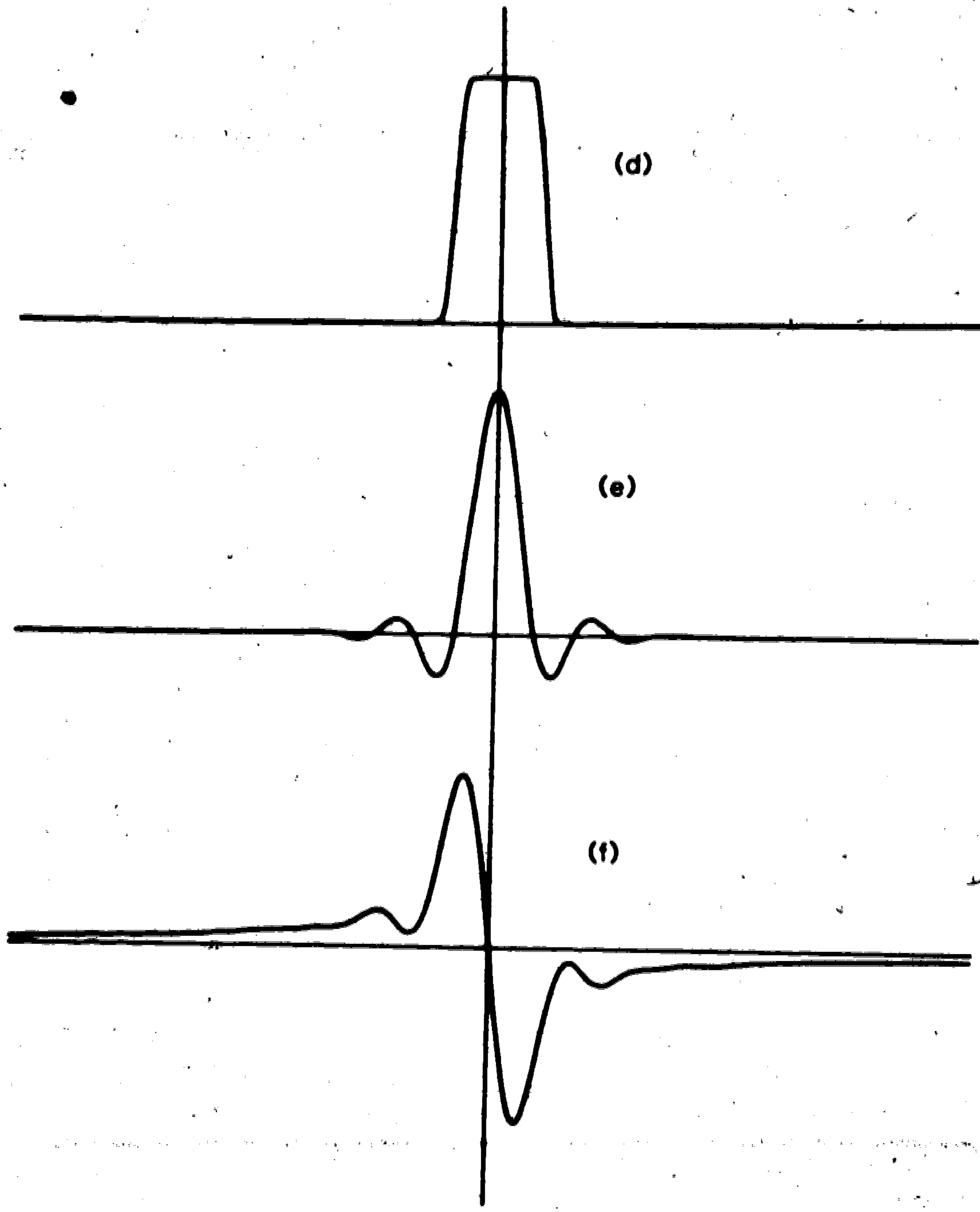


Figure 3.3.3 Continued. Figure (d) is the 50% raised cosine frequency domain window $W_2(f)$. Figure (e) is the time domain interpolation function $w_2(t)$, which is the Fourier transform of (d). Figure (f) is the modified Hilbert transform kernel $h_2(t)$ which is the Hilbert transform of (e).

As shown in figure 3.3.3, although the interpolation function $w_1(t)$, which is the Fourier transform of the 50% raised cosine window, has much lower sidelobes than the sinc function, the Hilbert transform of $w_1(t)$, $h_1(t)$, does not decay towards zero faster than the cosc(t) [1] function. Instead, ripples along the decaying slope are much reduced. With limited correlator length, the ripples on the slope of h'(t) can cause variation of the quadrature channel output as the map centre changes along the delay domain.

Figures 3.3.4.1 to 3.3.4.3 show the result of a simulation in which the gains of the inphase and quadrature channels are plotted against different positions of a single point source along the delay domain. Different correlator lengths, correlation function sampling rates, and interpolation functions were also simulated. The effect of correlation function truncation is to reduce the output to slightly less than unity in most cases. As expected, the output of both the inphase and quadrature channels approaches unity as the correlator length increases. Since the quadrature channel interpolation function decays more slowly than the real channel interpolation function, finite correlator length reduces the quadrature channel to a value less than the inphase channel. Note that all graphs are symmetric about delay value −0.5 because there are an even number of channels (16) available in this simulation.

Figure 3.3.5 shows the ratio of quadrature channel gain to real channel gain for different configurations. The dependence of gain ratio on delay is stronger for shorter correlators. The Fourier transformed 50% raised cosine function gives a lower dependence than the sinc function but requires twice the Nyquist sampling rate of the correlation function in the delay domain. It will be shown in section 3.4, that a difference in inphase and quadrature channel gain will lead to ripples of twice the fringe rate at the output. Figure 3.3.5(d) shows one case using a 16 channel correlator. The ratio of quadrature to real channel gain is rather independent of the source position over the extent of ±1 delay unit for this case. This configuration, sampling the correlation function at twice the Nyquist rate and using functions $w_1(t)$

---

[1] $\mathrm{cosc}(x) = [\cos(\pi x) - 1]/(\pi x)$

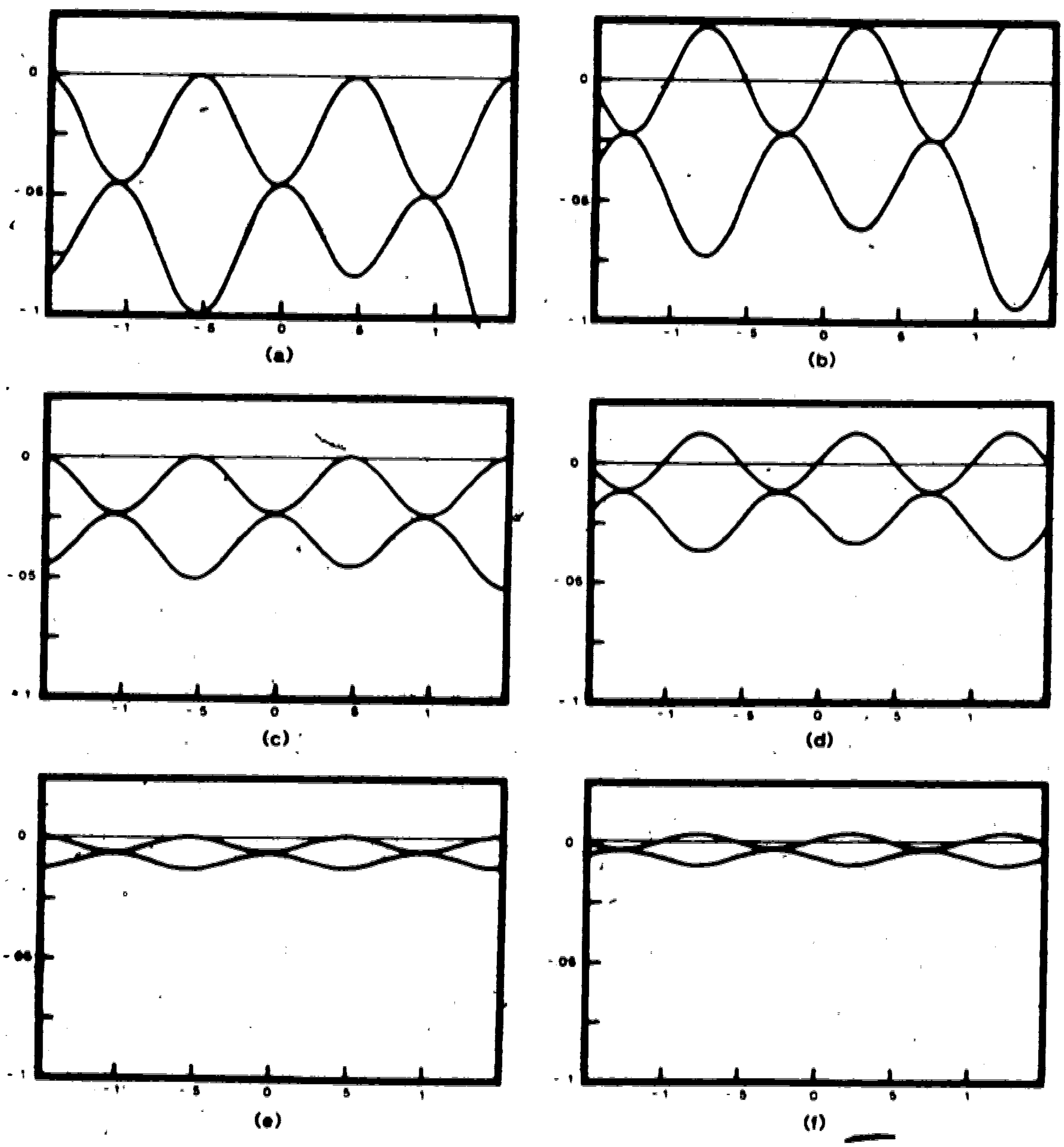


Figure 3.3.4.1 Gain of the inphase and quadrature channel vs delay using sinc function for interpolation and Nyquist rate sampling of the correlation function in delay domain. The first, second, and third rows correspond to correlators of 8, 16 and 64 channels respectively. The left column represents a point source at the phase centre. The vertical scale represents departure from unity. The right column represents a point source 0.5 delay unit off the phase centre. The vertical scale represents departure from the nominal value.

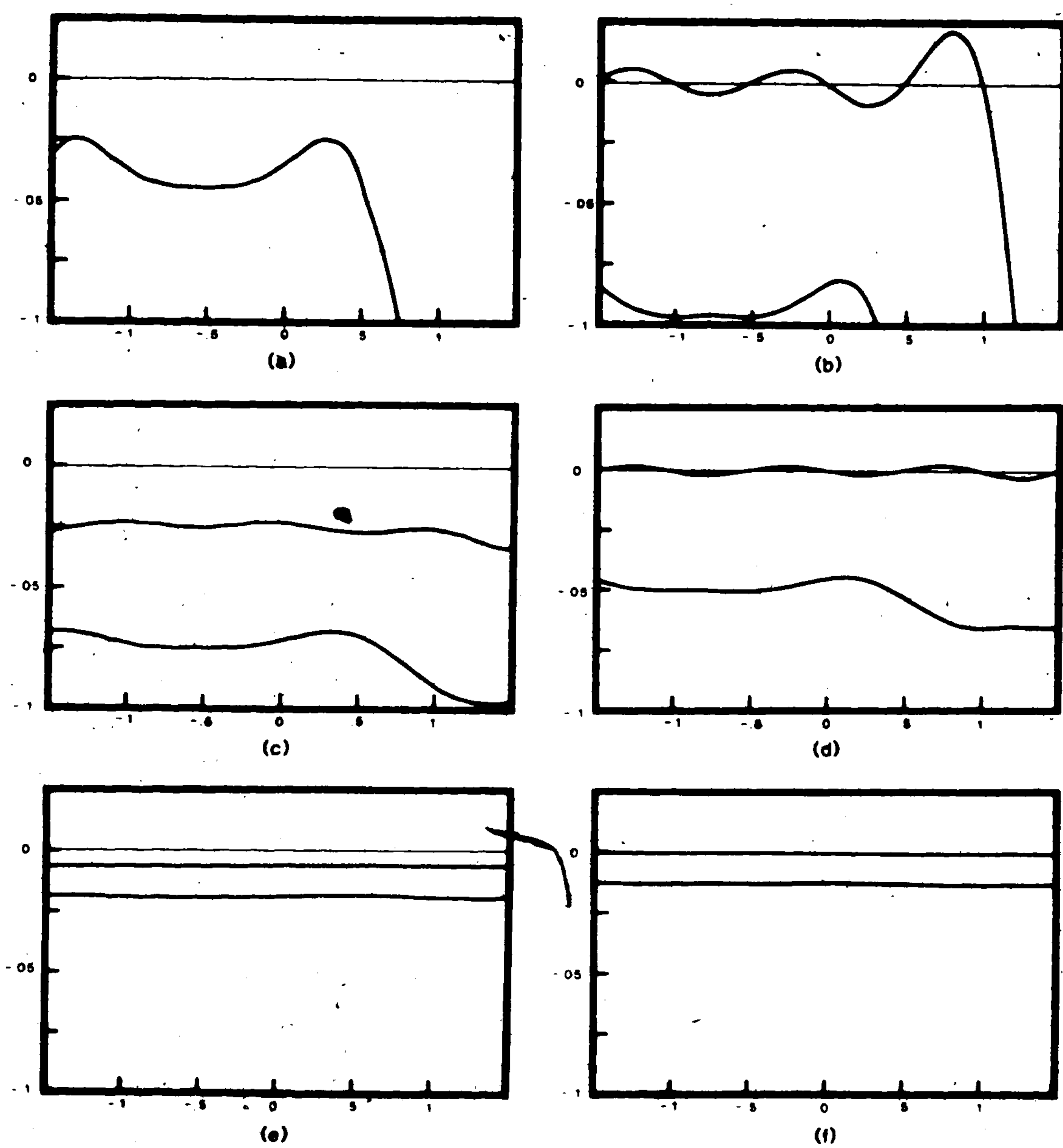


Figure 3.3.4.2 Gain of the inphase and quadrature channel vs delay using sinc function interpolation and *twice* Nyquist sampling rate of the correlation function in delay domain. The first, second, and third rows correspond to correlators of 8, 16 and 64 channels respectively. The left column represents a point source at the phase centre. The vertical scale represents departure from unity. The right column represents a point source 0.5 delay unit off the phase centre. The vertical scale represents departure from the nominal value.

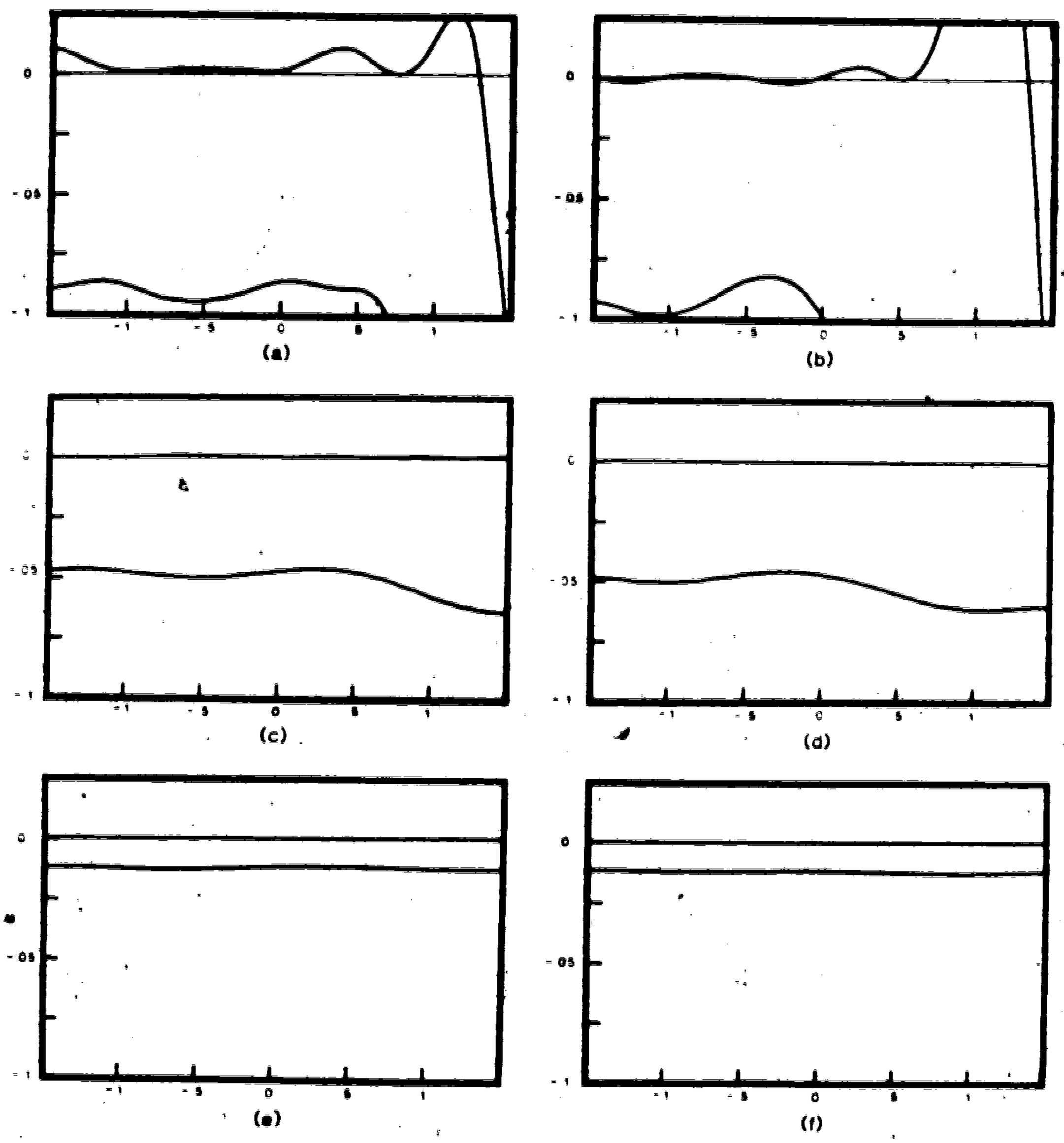


Figure 3.3.4.3 Gain of the real and quadrature channel vs delay using Fourier transform of the 50% raised cosine function for interpolation and sampling the correlation function at *twice* the Nyquist rate in delay domain. The first, second, and third rows correspond to correlators of 8, 16 and 64 channels respectively. The left column represents a point source at the phase centre. The vertical scale represents departure from unity. The right column represents a point source 0.5 delay unit off the phase centre. The vertical scale represents departure from the nominal value.

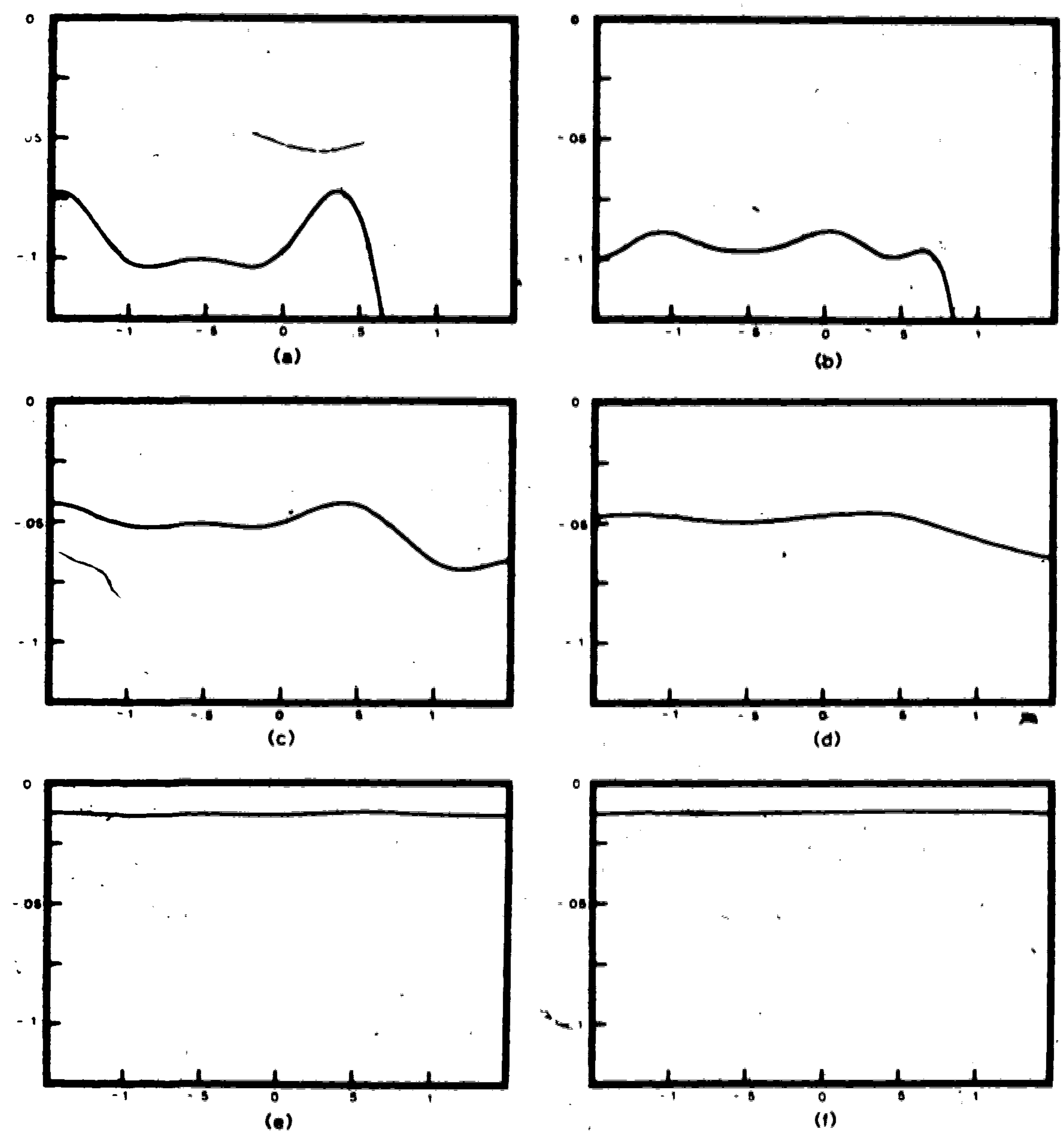


Figure 3.3.5 Ratio of gain of the quadrature channel to the real channel vs delay. The left and the right columns correspond to using sinc function and Fourier transform of 50% raised cosine for interpolation, respectively. The first, second, and third rows correspond to 8, 16 and 64 channel correlators. All cases are for point source at phase centre and twice the Nyquist rate sampling of the correlation function in delay domain.

and $h_2(t)$ for interpolation, is chosen for implementation since a constant correction factor could be used to equalize the quadrature channel gain.

3.4 Fringe Derotation

3.4.1 Theory of Fringe Derotation

In the basic interferometer shown in figure 1.2.1, a fixed pattern of interference fringes is formed on the plane of the sky. As the earth rotates, the fringe pattern follows. The interferometer output when pointing at a point source will vary sinusoidally as described by (1.2.2).

$$R(t) = k_2 \; S \; \cos\left(\frac{2\pi B}{\lambda} \cos\theta \right) \tag{3.4.1}$$

The frequency of R(t) is called the natural fringe rate. For ease of instrumentation and data analysis, it is often desirable to derotate the fringe or freeze the fringe pattern at the source. Stopping the fringes will also stop the sinusoidal variation of R(t), which allows longer integration period in the correlators. Conventionally, fringe derotation is done by applying a controlled differential phase $\phi(t)$ between the L.O. signals sent to the first mixers. The responses of the more sophisticated interferometer of figure 1.2.2 are given by (1.2.8) and (1.2.9).

$$Rr(t) = \beta(\delta\tau) \; \cos[\omega_{IF}\delta\tau + \omega_0\tau(t) - \phi(t)] \tag{3.4.2}$$

$$Rq(t) = \beta(\delta\tau) \; \sin[\omega_{IF}\delta\tau + \omega_0\tau(t) - \phi(t)] \tag{3.4.3}$$

To stop the fringes, the injected phase $\phi(t)$ should be equal to the fringe phase, or,

$$\phi(t) = \omega_{IF}\delta\tau + \omega_0\tau(t) \tag{3.4.4}$$

Expanding the geometric delay $\tau(t)$,

$$\phi(t) = \omega_{IF}\delta\tau + \frac{B\omega_o}{c}\cos(DEC)\sin(HA) \tag{3.4.5}$$

where B = baseline length

DEC = source declination

HA = source hour angle

$\delta\tau$ = delay equalisation error

$= \tau(t) - \tau_D$

Since delay equalisation error is negligible in the 408 MHz system, $\phi(t)$ reduces to

$$\phi(t) = \frac{B\omega_o}{c}\cos(DEC)\sin(HA) \tag{3.4.6}$$

In the schematic working diagram in figure 1.2.2, the local oscillator delivers output to the two mixers with a controlled phase shift $\phi(t)$. The response of the real channel of the interferometer is given by

$$R_r(t) = \beta(\delta\tau)\cos[\omega_{IF}\delta\tau + \omega_o\tau_D - \phi(t)] \tag{3.4.7}$$

where $\delta\tau = \tau(t) - \tau_D$. The operation of the quadrature channel is equivalent to having a 90° phase shifter in one of the paths. Its response is therefore given by

$$R_q(t) = \beta(\delta\tau)\sin[\omega_{IF}\delta\tau + \omega_o\tau_D - \phi(t)] \tag{3.4.8}$$

But in the 408 MHz system there is no provision for injecting $\phi(t)$ into the local oscillator signals. Instead, the correlators are sampled rapidly and fringe derotation is applied to the correlation function. The real and quadrature correlation functions $Rxx(\tau_o)$ and $Rxy(\tau_o)$ are given by:

$$Rxx(\tau_o, t) = \beta(\delta\tau)\cos[\omega_{IF}\delta\tau + \omega_o\tau_D] \tag{3.4.9}$$

$$Rxy(\tau_o, t) = \beta(\delta\tau)\sin[\omega_{IF}\delta\tau + \omega_o\tau_D] \tag{3.4.10}$$

Since $\phi(t)$ is a known injected phase, equations (3.4.7) and (3.4.8) may be synthesised with (3.4.9) and (3.4.10)[12].

$$Rr(t) = Rxx(\tau_o,t)\cos\phi(t) + Rxy(\tau_o,t)\sin\phi(t) \quad (3.4.11)$$

$$Rq(t) = Rxy(\tau_o,t)\cos\phi(t) - Rxx(\tau_o,t)\sin\phi(t) \quad (3.4.12)$$

Conceptually, the derotation is done exactly as single-side-band down-mixing, the only difference being that two output channels are generated instead of one. Figure 3.4.1 shows the schematics of down mixing.

With the given baseline geometry and frequency of operation, the maximum fringe rate is 21.36° per second for the 408 MHz interferometer. Equations (3.4.11) and (3.4.12) have to be synthesised with a new value of $\phi(t)$ every minor integration period. In practice, after every minor integration, all the accumulators of the correlators will have to be read, interpolation and Hilbert transformation performed to obtain $Rxx(\tau_o)$ and $Rxy(\tau_o)$ and the expressions in (3.4.7) and (3.4.8) evaluated for $Rr(t)$ and $Rq(t)$.

3.4.2 Ripples Caused By Derotation After Correlation

In the scheme of derotation after correlation, any crosstalk which exists between the two incoming signals will give rise to ripples of the fringe rate at the output. Crosstalk between the incoming signals will contaminate the correlation functions $Rxx(\tau,t)$ and $Rxy(\tau,t)$ with time invariant terms $\varepsilon_x(\tau)$ and $\varepsilon_y(\tau)$ respectively. After derotation:

$$\begin{aligned} Rr_1 &= (Rxx + \varepsilon_x)\cos(\phi) + (Rxy + \varepsilon_y)\sin(\phi) \\ &= Rxx\cos(\phi) + Rxy\sin(\phi) + \varepsilon_x\cos(\phi) + \varepsilon_y\sin(\phi) \\ &= Rr + \varepsilon\cos(\phi - k) \end{aligned} \quad (3.4.13)$$

$$Rq_1 = (Rxy + \varepsilon_y)\cos(\phi) - (Rxx + \varepsilon_x)\sin(\phi)$$

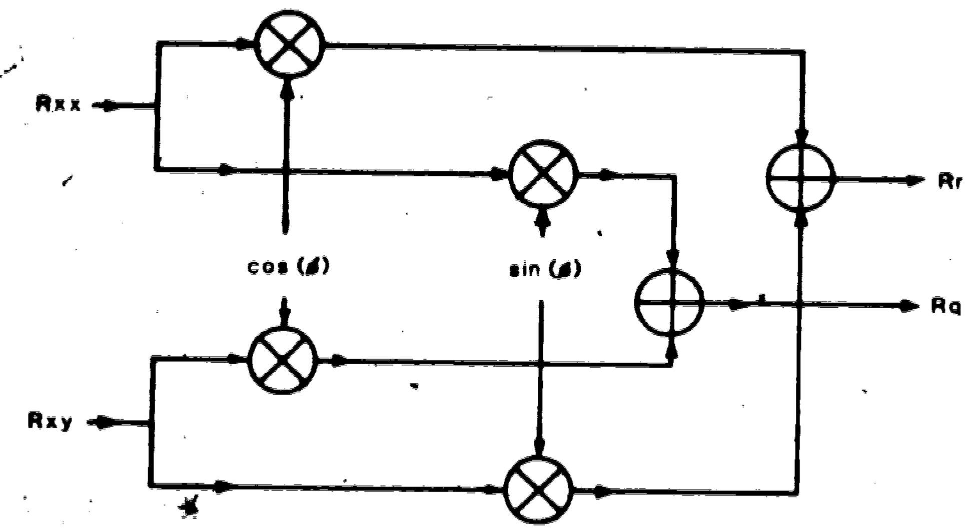


Figure 3.4.1 Schematics of fringe derotation.

$$
\begin{aligned}
&= Rxy\cos(\phi) - Rxx\sin(\phi) + \epsilon_y\cos(\phi) - \epsilon_x\sin(\phi) \\
&= Rq + \epsilon\cos(\phi + 90^\circ - k) \\
&= Rq - \epsilon\sin(\phi - k)
\end{aligned}
\tag{3.4.14}
$$

where $\epsilon^2 = \epsilon_x^2 + \epsilon_y^2$, and $k = \tan^{-1}(\epsilon_y / \epsilon_x)$

The inphase and quadrature channel output are contaminated with a cosine and a sine term of the fringe rate.

As mentioned in section 3.3.2, finite correlator length leads to a slightly different gain in the inphase and quadrature channel outputs. Such slight gain difference will lead to ripples of twice fringe rate at the output. Let the gain of the quadrature channel be modified by $(1 - \delta)$, where δ is positive and less than unity. After fringe derotation:

$$Rr_2(t) = Rxx\cos(\phi) + (1-\delta)Rxy\sin(\phi)$$

$$Rr_2(t) = Rr - \delta Rxy\sin(\phi). \tag{3.4.15}$$

Expressing the correlation functions $Rxx(\tau_s,t)$ and $Rxy(\tau_s,t)$ in terms of the perfectly derotated output $Rr(t)$ and $Rq(t)$ gives:

$$Rxx(\tau,t) = Rr\cos(\phi) - Rq\sin(\phi) \tag{3.4.16}$$

$$Rxy(\tau,t) = Rq\cos(\phi) + Rr\sin(\phi). \tag{3.4.17}$$

The term $Rxy\sin(\phi)$ in equation (3.4.15) can be expresed as

$$\begin{aligned} Rxy(t)\sin(\phi) &= Rq(t)\cos(\phi)\sin(\phi) + Rr(t)\sin^2(\phi) \\ &= \tfrac{1}{2}Rq(t)\sin(2\phi) + \tfrac{1}{2}Rr(t)\,[1 - \cos(2\phi)]. \end{aligned} \tag{3.4.18}$$

Since $Rr(t)$ and $Rq(t)$ are slow time varying functions, ripples of twice fringe rate will appear in the output $Rr_2(t)$. Similarly, for the quadrature channel

$$\begin{aligned} Rq_2(t) &= Rxy\,(1-\delta)\cos(\phi) - Rxx\sin(\phi) \\ &= Rq(t) - \delta\,Rxy\cos(\phi) \\ &= Rq(t) - \delta\,[\,Rq(t)\cos^2(\phi) + Rr(t)\sin(\phi)\cos(\phi)\,] \\ &= Rq(t) - \tfrac{1}{2}\delta\,[\,Rq(t)\,(1 + \cos(2\phi)) + Rr(t)\sin(2\phi)\,] \end{aligned} \tag{3.4.19}$$

which also contains ripples of twice fringe rate.

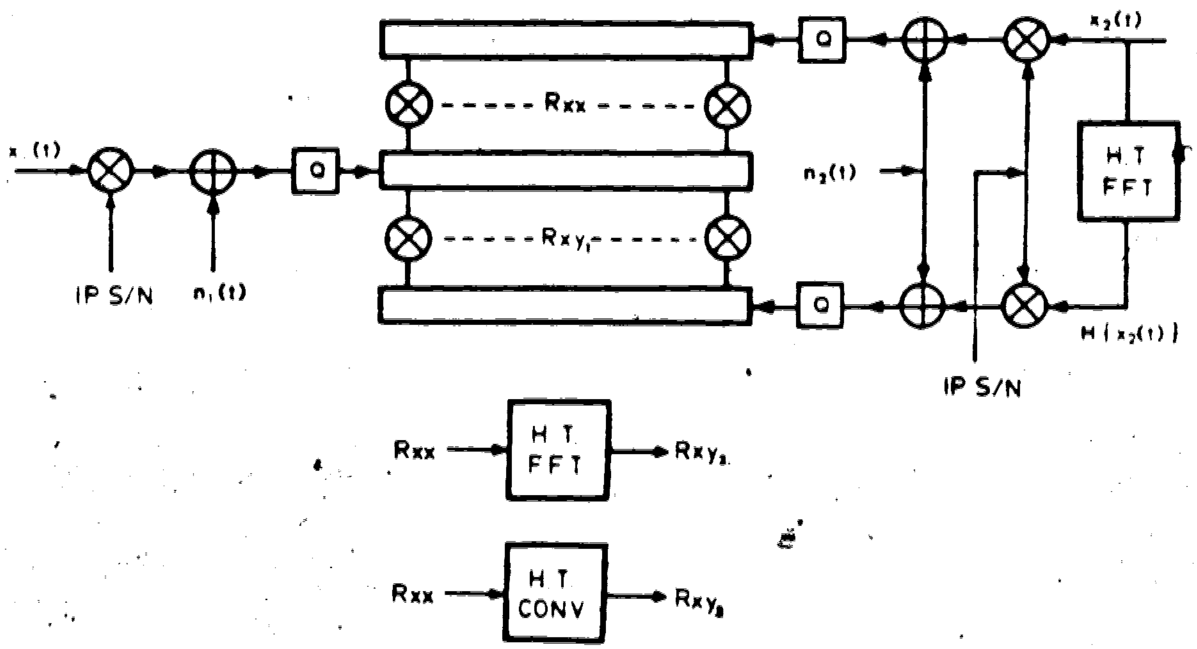
4. The Crosscorrelator and Quadrature Channel Generation Simulation

A simulation study of the 408 MHz DSP has been done with the Amdahl computer of the University of Alberta. The simulation was intended to prove the feasibility of the scheme, especially with finite correlator length. It also served as a design aid for finding a suitable correlator configuration. The simulation of the DSP included crosscorrelation, quadrature channel generation and continuous delay equalisation by interpolation. Different means of quadrature channel generation were simulated for comparison.

4.1 Configuration Of Simulation

The simulation software consists mainly of a noise generator program and a correlator simulator program. The interactive noise generator provides the generation of sampled band-limited noise streams. Any source configuration can be synthesised with the superposition of point sources of different powers at different delays along the correlation function. Outputs of the noise generator are stored in disc files for use by the correlator simulator. A Hilbert transformed or 90° phase shifted version of a noise stream can also be generated for simulation of the conventional method of quadrature channel generation.

The correlator simulator, as shown in figure 4.1.1, simulates the actual process of digital crosscorrelation. Two hypothetical correlators are simulated. One correlates the two inphase inputs $x_1(t)$ and $x_2(t)$ to produce the inphase correlation function Rx_1x_2, or Rxx in short. The other correlates the inphase and the quadrature input $x_1(t)$ and $\hat{x}_2(t)$ to produce the quadrature correlation function $Rx_1\hat{x}_2$, or Rxy_1 for short. The quadrature input $\hat{x}_2(t)$ is produced by Hilbert transforming the input $x_2(t)$ by FFT, the equivalent of a 90° phase shift. These two hypothetical correlators are each 129 channels long, with the input signals sampled at sixteen times the Nyquist rate and the correlation function sampled at eight times the Nyquist rate in the delay domain. The delay of the 129 channels is equivalent to 33 channels of the physical correlator, or 2 microseconds.



$x_1(t)$	= INPUT SIGNAL 1
$x_2(t)$	= INPUT SIGNAL 2
$y_2(t)$	= HILBERT TRANSFORM OF $x_2(t)$
$n_1(t), n_2(t)$	= INDEPENDENT RECEIVER AND SKY NOISE
IP S/N	= INPUT S/N RATIO
	= SOURCE TEMP. / (RECEIVER TEMP. + SKY TEMP.)
Rxx	= CROSSCORRELATION OF $x_1(t)$ AND $x_2(t)$ WITH NOISE.
Rxy_1	= CORSSCORRELATION OF $x_1(t)$ AND $y_2(t)$ WITH NOISE.
Rxy_2	= HILBERT TRANSFORM OF Rxx BY FFT.
Rxy_3	= HILBERT TRANSFORM OF Rxx BY CONVOLUTION.

Figure 4.1.1 Configuration of correlator simulation

The input signals entering the correlator are multiplied by an input S/N ratio which is the source temperature / (receiver temperature + sky temperature). An uncorrelated noise stream representing the receiver and sky noise is added to each of the inputs. Output of the inphase crosscorrelator is then Hilbert transformed using either the FFT or convolution to produce the quadrature correlation functions Rxy_2 and Rxy_3 respectively. These are then plotted on the same graph with Rxy_1 for comparison. A quantiser Q just before the correlator input provides the option of quantising the input into 3 levels for simulation of the coarse quantisation used in digital correlation. Correction of the correlation function analogous to the Van Vleck correction given in equation (3.1.6) is applied to both Rxx and Rxy_i when quantisation is in effect.

To test the interpolation and quadrature channel generation algorithm used in the DSP, sixteen points are sampled from the central half of the inphase correlation function corresponding to the output of the 16-channel correlator used in the DSP. The interpolation algorithm is applied to the 16 samples to recover the intermediate values. The modified Hilbert transform by convolution is also applied to generate the quadrature correlation function with intermediate values. These are plotted on the same graph with Rxx and Rxy_i for comparison. The interpolation function and modified Hilbert transform kernel are based on the 50% raised cosine window function as shown in Appendix I. Appendix IV includes the listing of the noise generator and the correlator simulator programs.

4.2 Results Of Simulation

Simulations were performed for two source configurations a) a single point source at the field center, and b) two point sources with a power ratio of 1.2 lying on opposite edges of the field of view. For each source configuration, simulation was performed with the four combinations of high and low input S/N ratios and with and without 3 level quantisation. The unrealistically high input S/N ratios of 0.5 and 10.0 were used because of the short duration of integration affordable. Every simulation run correlated 8000 samples from each input, corresponding to only 62.5 microseconds of integration. The short duration of integration simulated was limited by the excessive C.P.U time and storage space required for the noise samples.

The results of the eight simulation runs are plotted in figure 4.2.1 to 4.2.8. Plot (b) of every figure is a magnified plot of the central part of plot (a). The x-axis is numbered from 0 to 32 corresponding to a 33 channel physical correlator. The small arrowheads above the lower boundary line indicate the sampling positions of the 16 channel corelator used in the DSP. The y-axis indicates the number accumulated in the correlator after correlation of 8000 input samples.

The quadrature correlation functions Rxy_1, Rxy_2 and Rxy_3 show very good agreement in all cases of high input S/N ratio. The departures of Rxy from Rxy_2 and Rxy_3 in the low input S/N ratio cases are slightly higher than that predicted by statistical fluctuations. Such discrepancies are due to difficulties in generating totally uncorrelated noise streams using pseudo-random noise generation algorithms.

In general, the Hilbert transform of the inphase correlation function by both FFT and convolution agrees well with the quadrature correlation function. The interpolation and modified Hilbert transform by convolution algorithms operating on the 16-sample correlation function produce very satisfactory results in recovering the inphase convolution function and generating the quadrature correlation function. Agreement is particularly good around the centre of the correlation function where the outputs of the continuum channels are derived. The results show that a 3 level by 3 level 16-channel correlator with channel spacing of 1/16 microsecond is sufficient for generating an inphase and quadrature continuum channel output with the algorithm used.

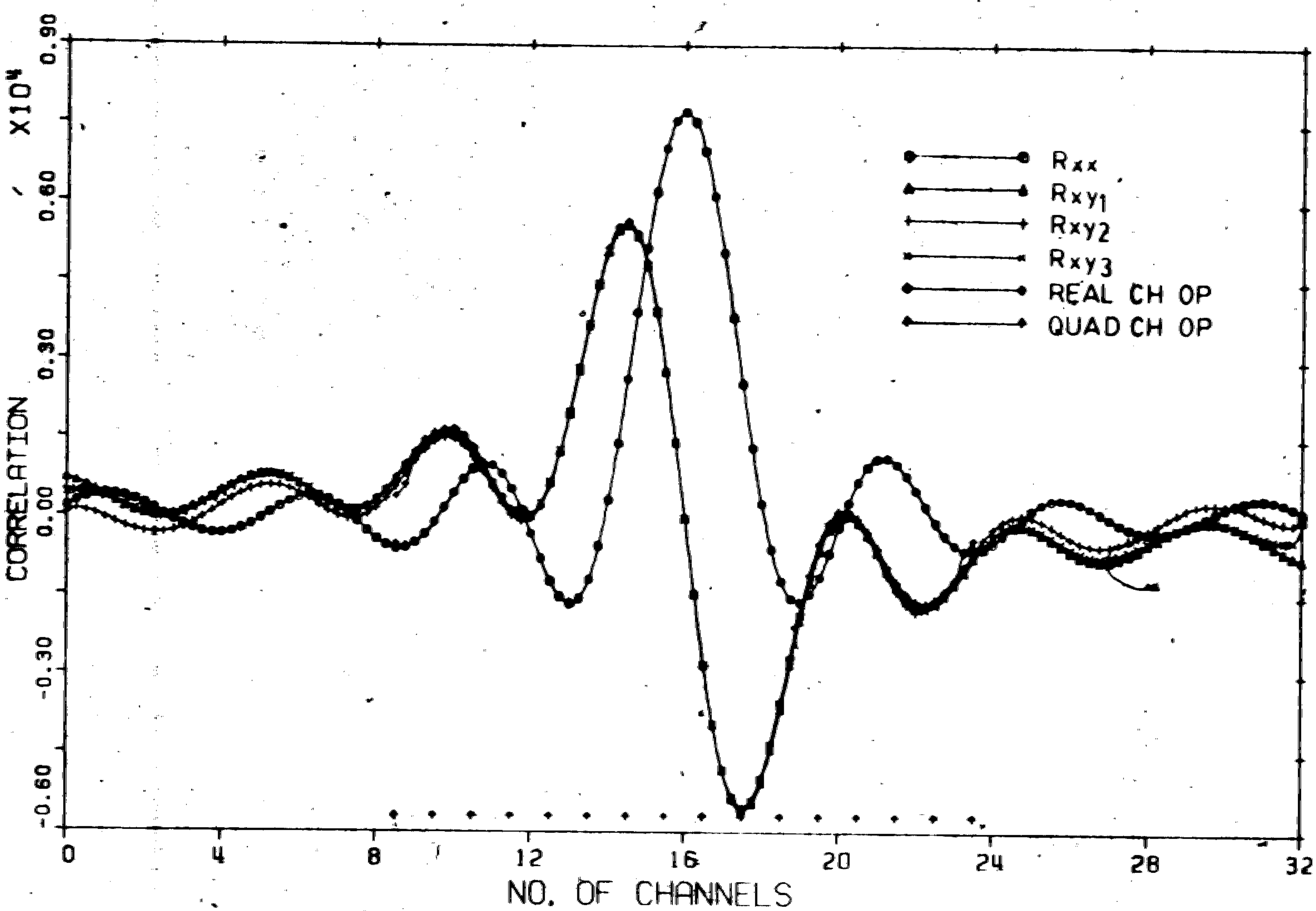


Figure 4.2.1(a) Single point source at centre of field. Quantisation = NO. Input S/N ratio = 10.

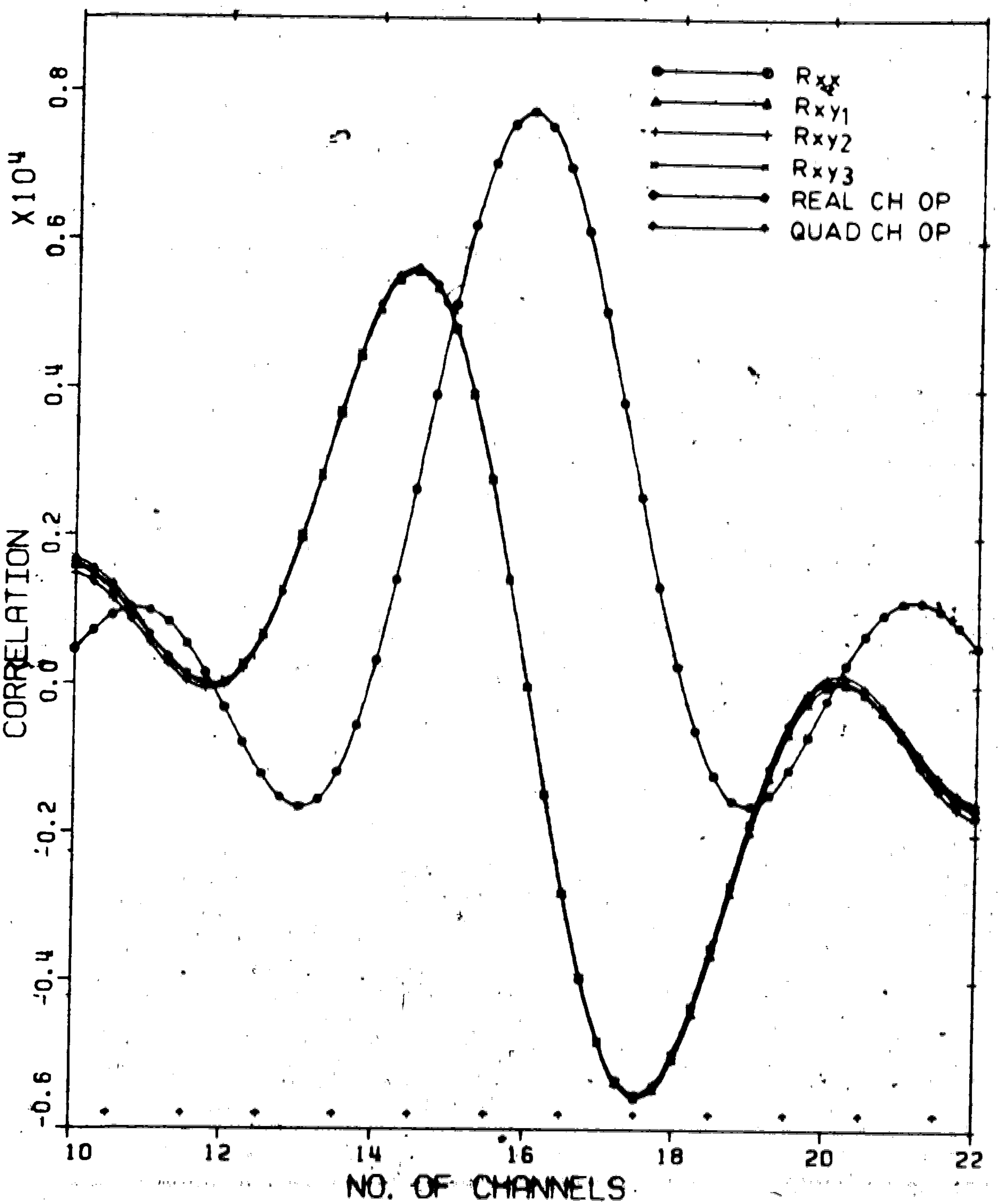


Figure 4.2.1(b) Expanded scale plot of central part of 4.2.1(a)

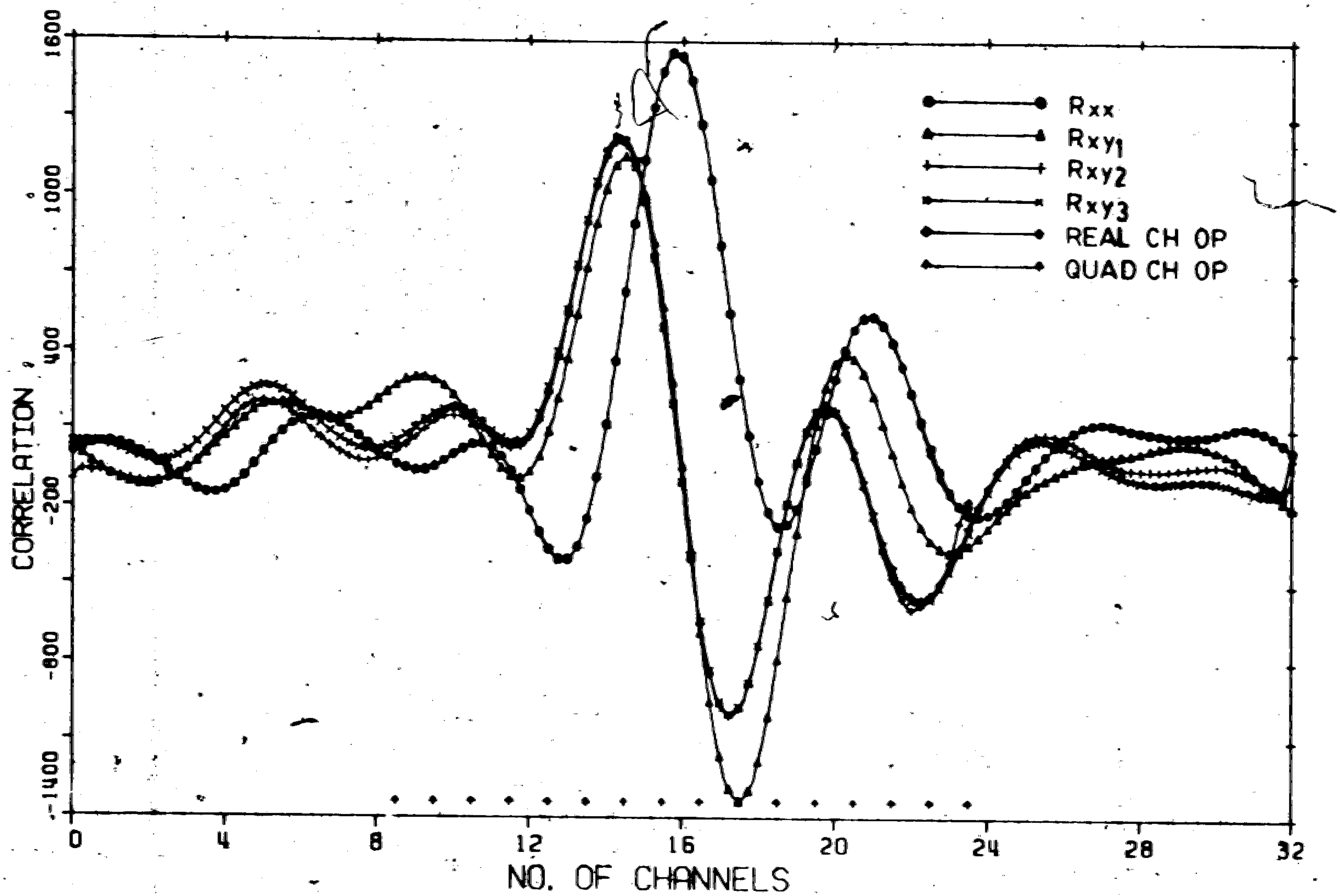


Figure 4.2.2(a) Single point source at centre of field. Quantisation = NO. Input S/N ratio = 0.5.

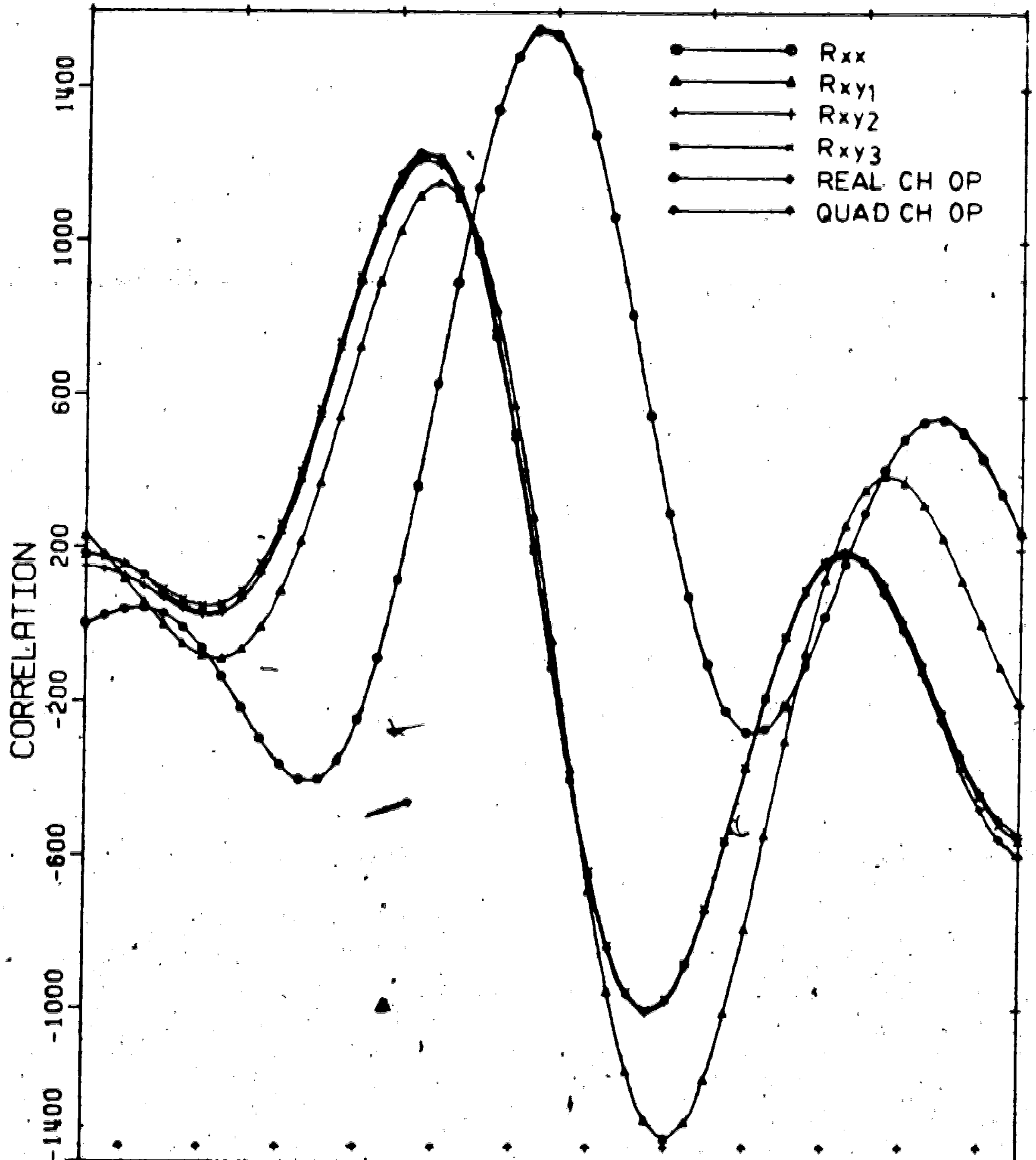


Figure 4.2.2(b) Expanded scale plot of central part of 4.2.2(a)

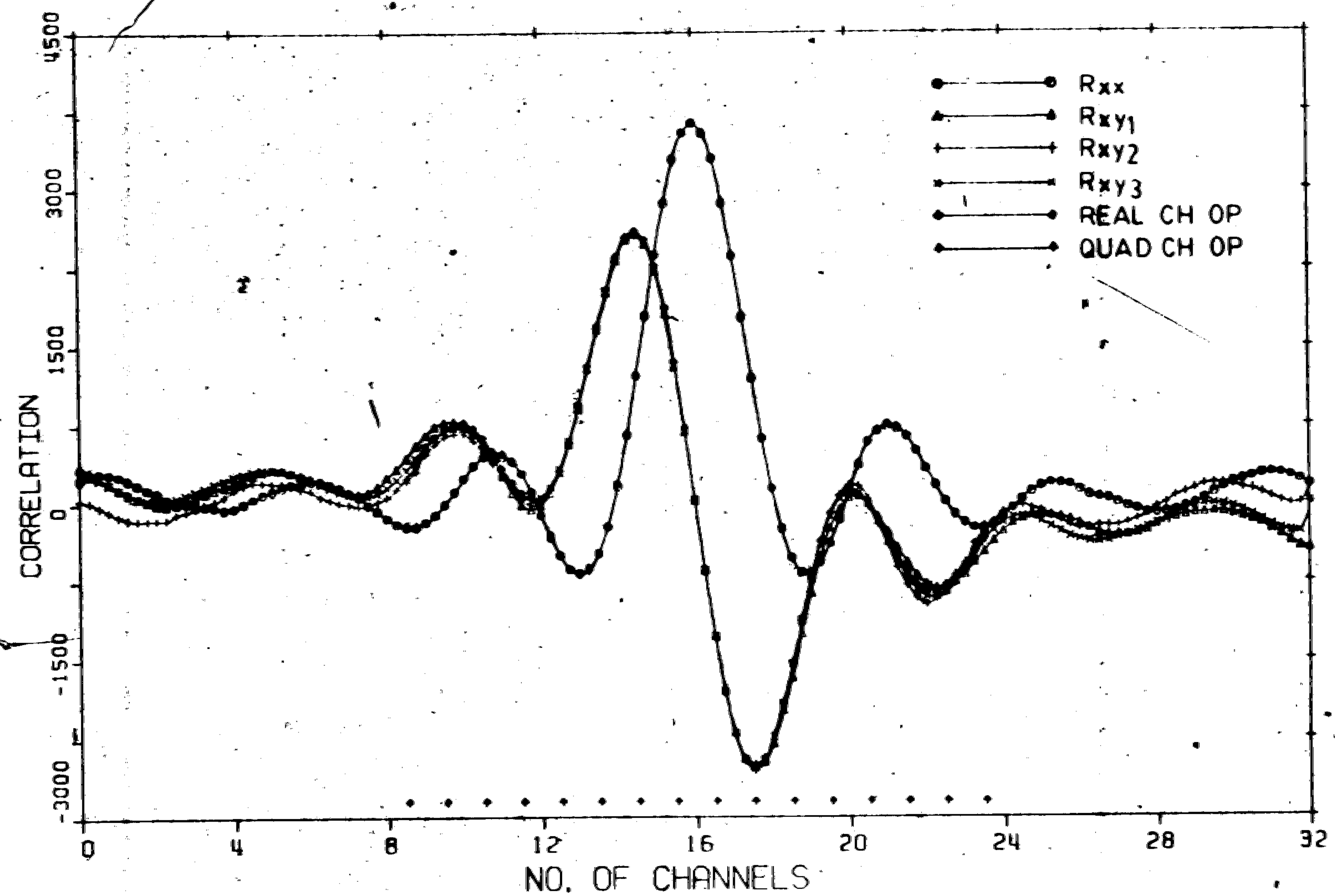


Figure 4.2.3(a) Single point source at centre of field. Quantisation = YES. Input S/N ratio = 10.

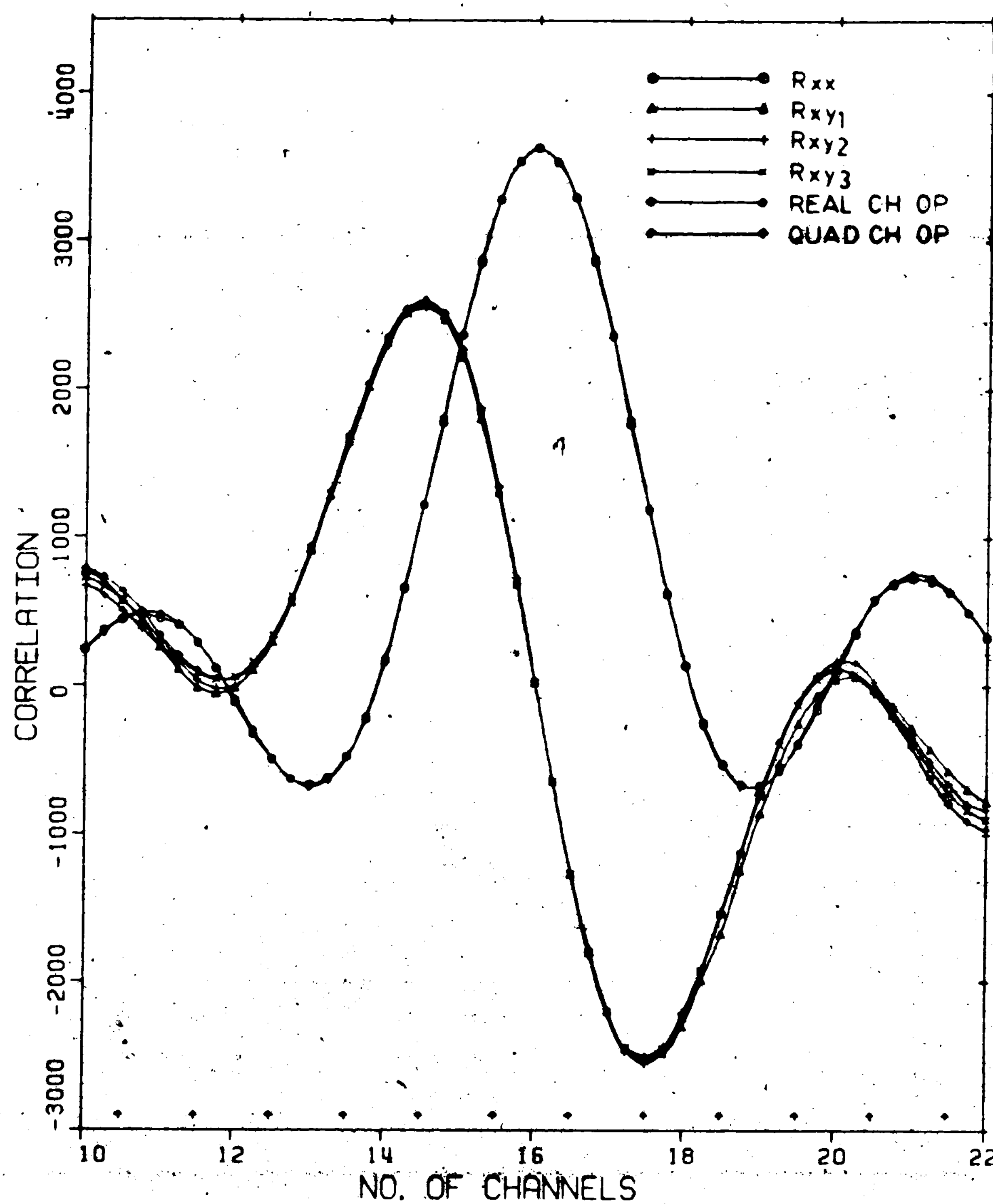


Figure 4.2.3(b) Expanded scale plot of central part of 4.2.3(a)

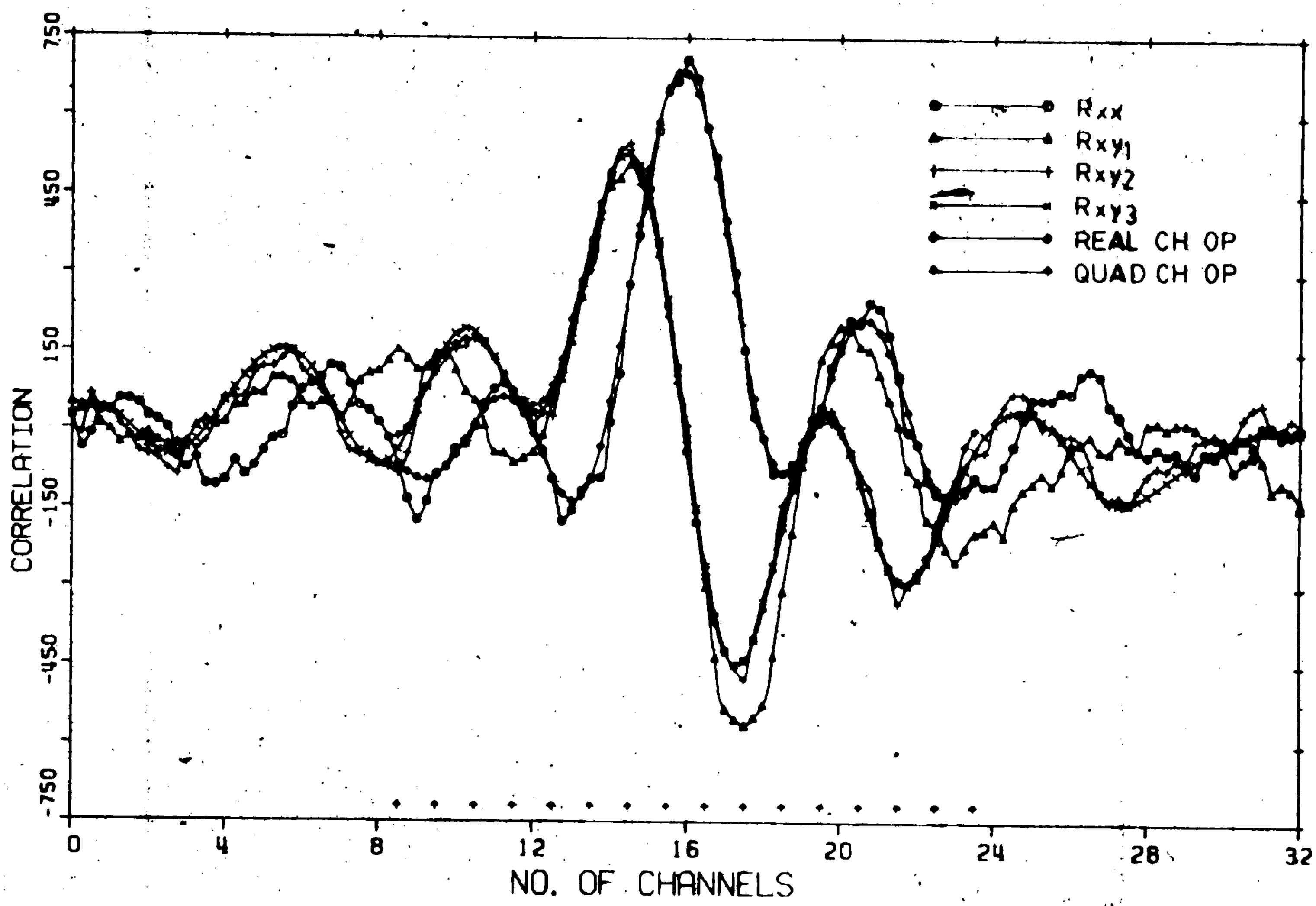


Figure 4.2.4(a) Single point source at centre of field. Quantisation = YES Input S/N ratio = 0.5

7

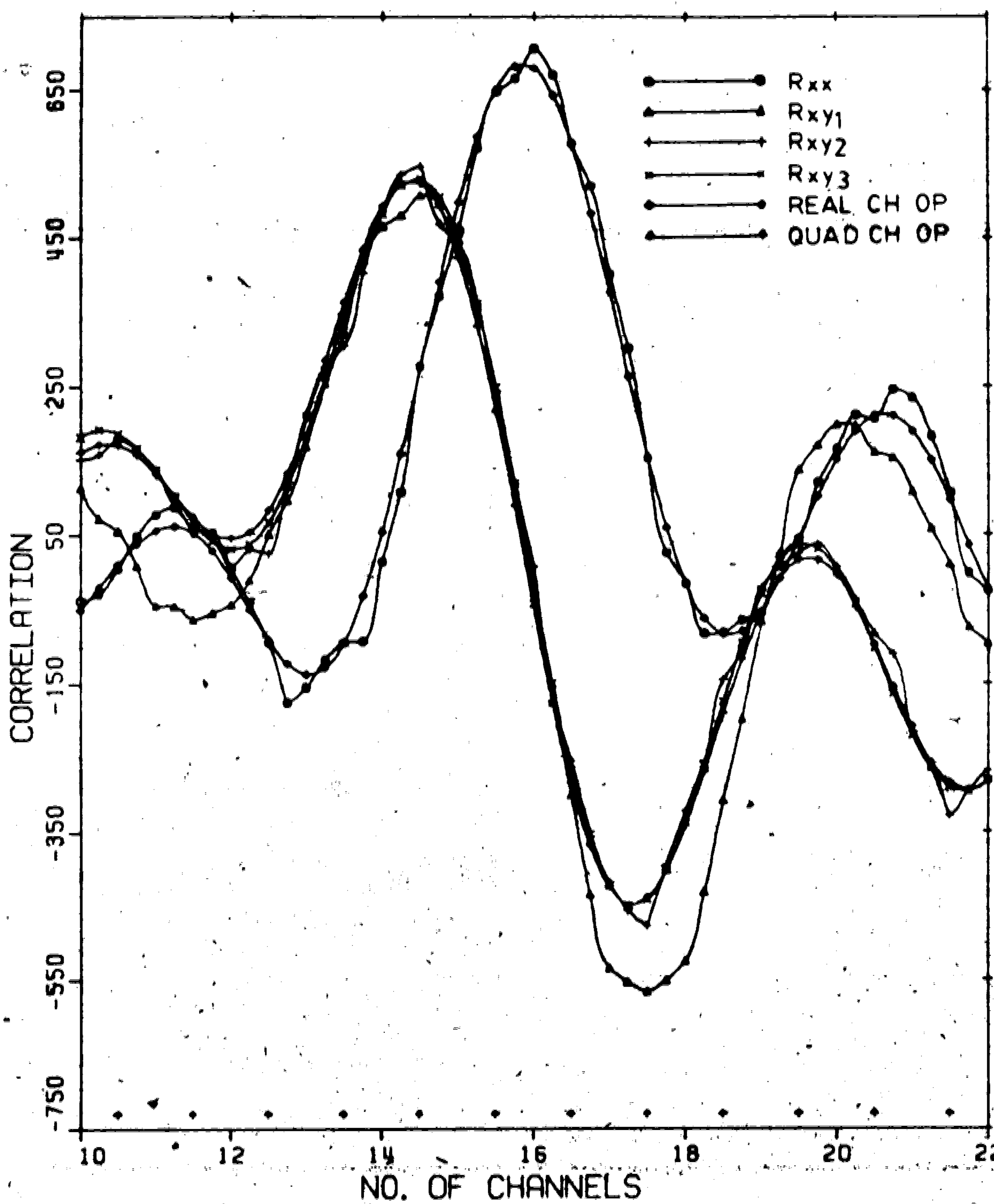


Figure 4.2.4(b) Expanded scale plot of central part of 4.2.4(a)

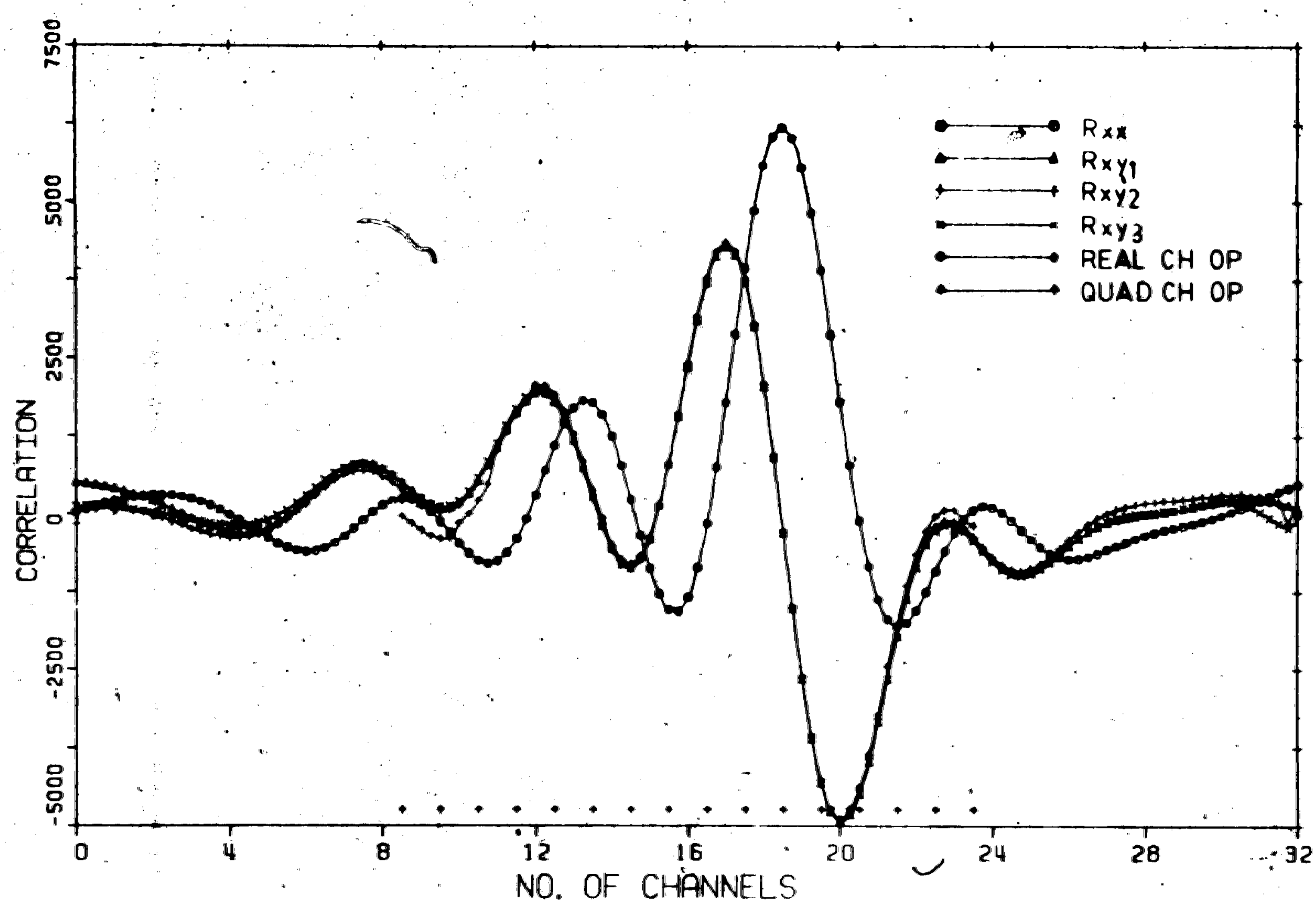


Figure 4.2.5(a) Two point sources: 2.5 channels right weight=2, 2.5 channels left weight=1. Quantisation = NO. Input S/N ratio = 10.

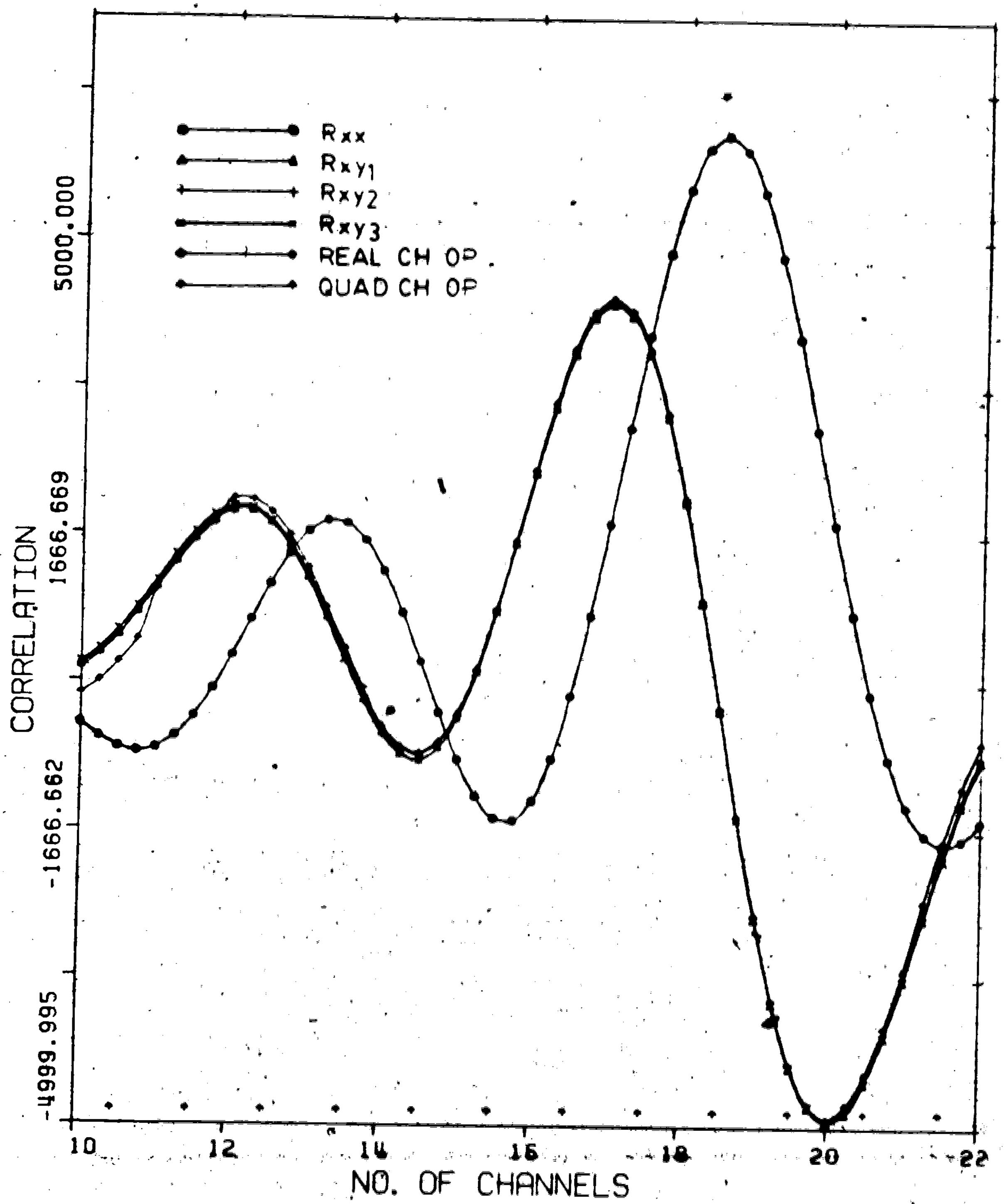


Figure 4.2.5(b) Expanded scale plot of central part of 4.2.5(a)

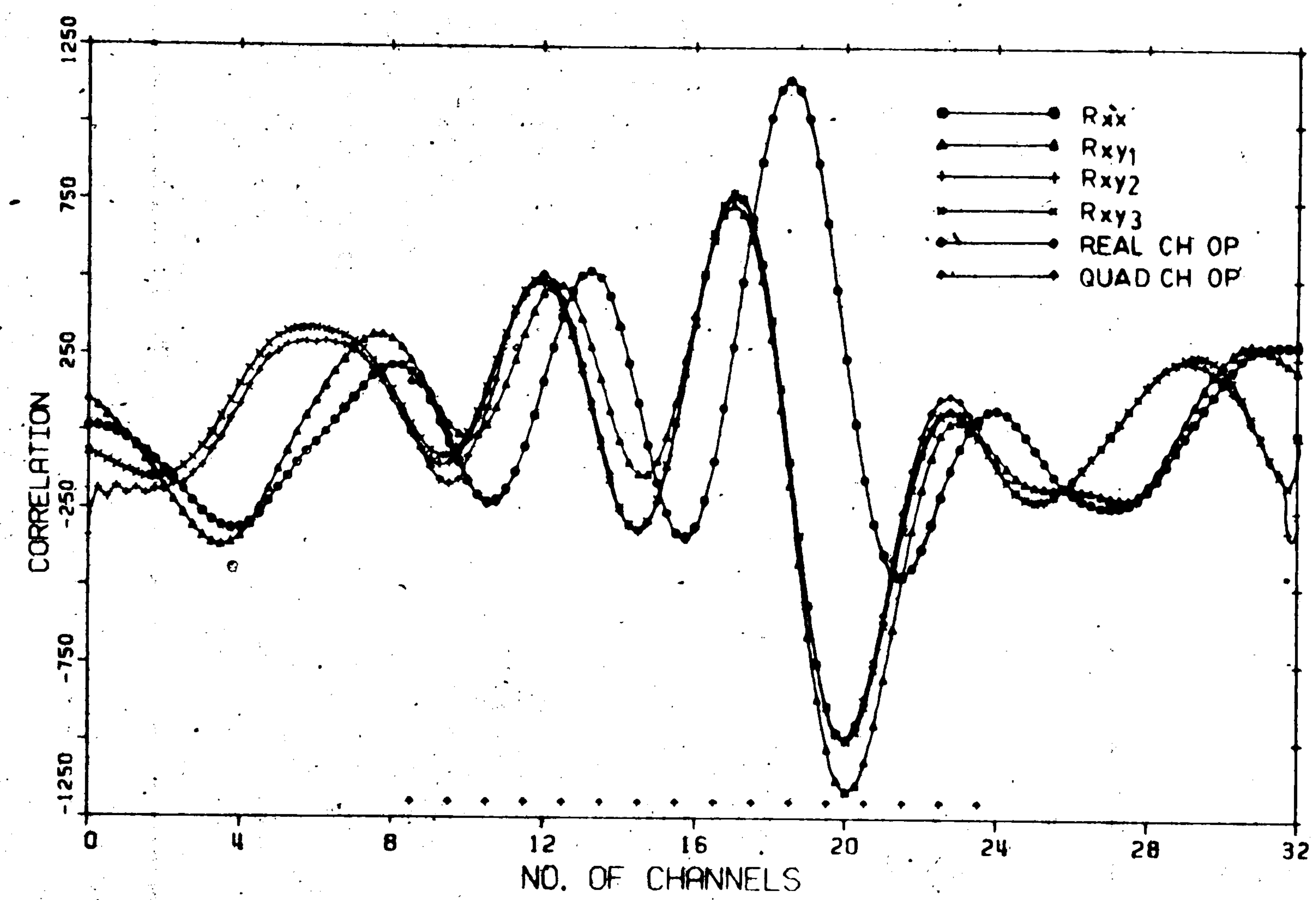


Figure 4.2.6(a) Two point sources 2.5 channels right weight = 2, 2.5 channels left weight = 1. Quantisation = NO. Input S/N ratio = 0.5.

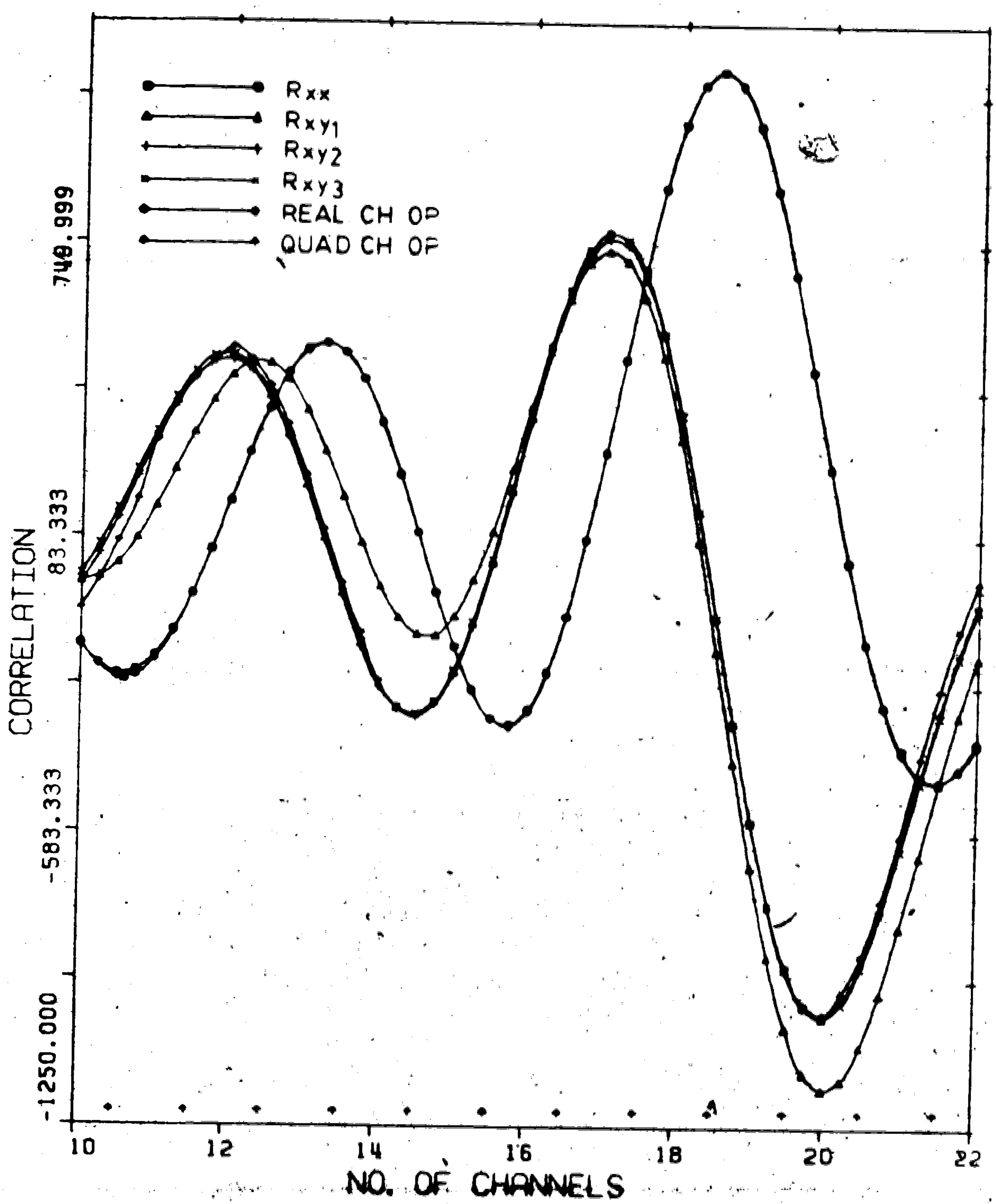


Figure 4.2.6(b) Expanded scale plot of central part of 4.2.6(a)

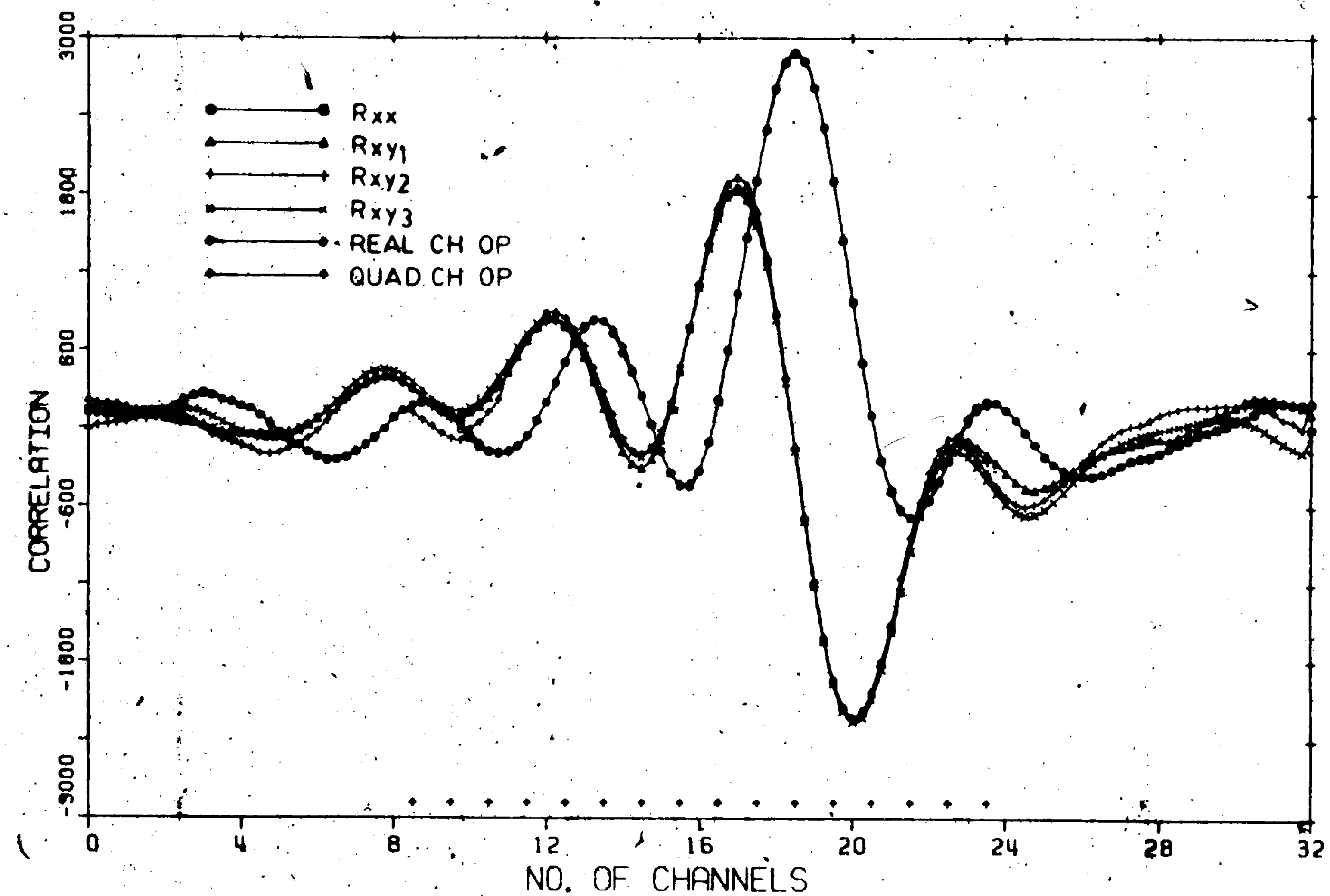


Figure 4.2.7(a) Two point sources 2.5 channels right weight = 2, 2.5 channels left weight = 1. Quantisation = YES. Input S/N ratio = 10.

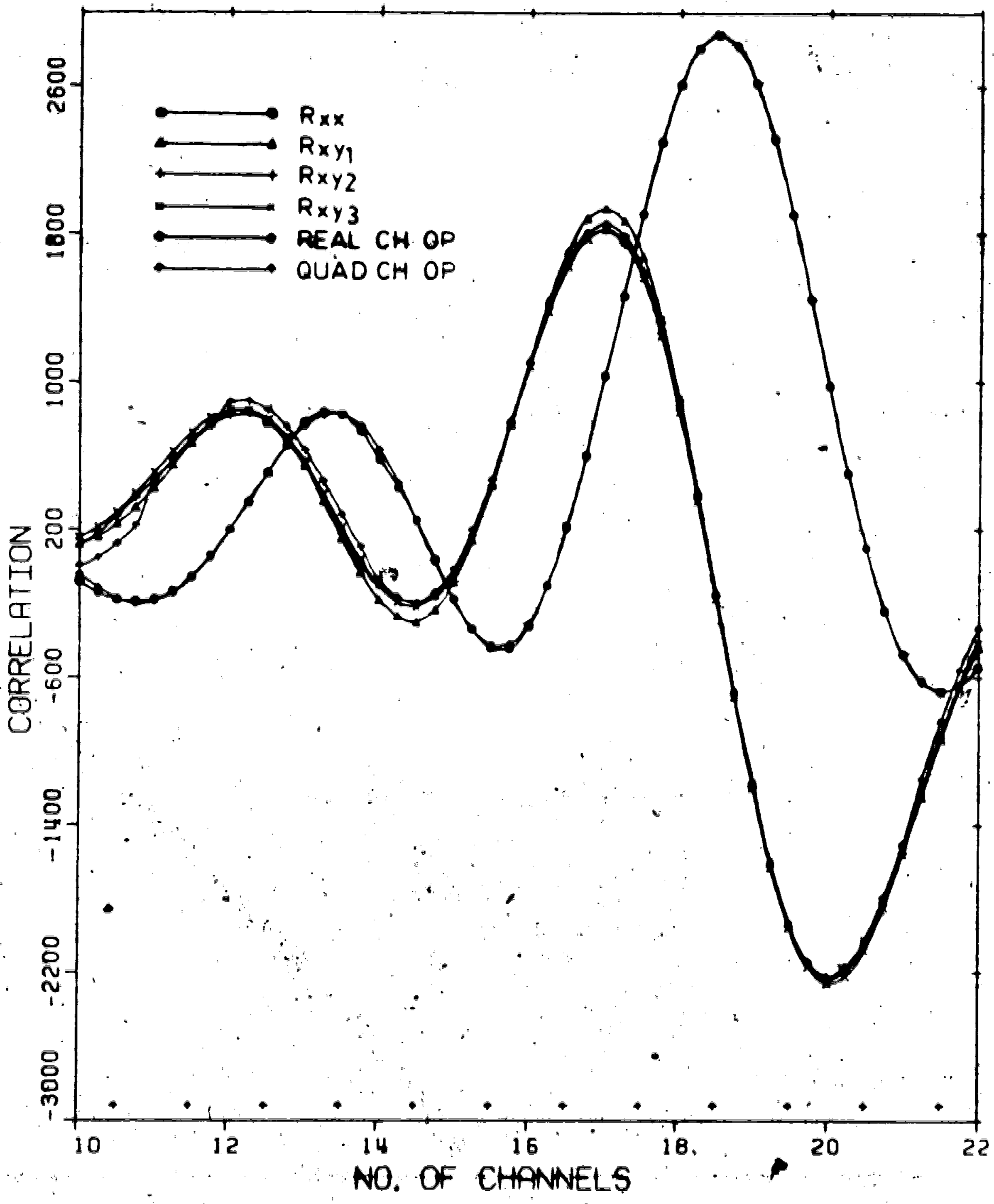


Figure 4.2.7(b) Expanded scale plot of central part of 4.2.7(a)

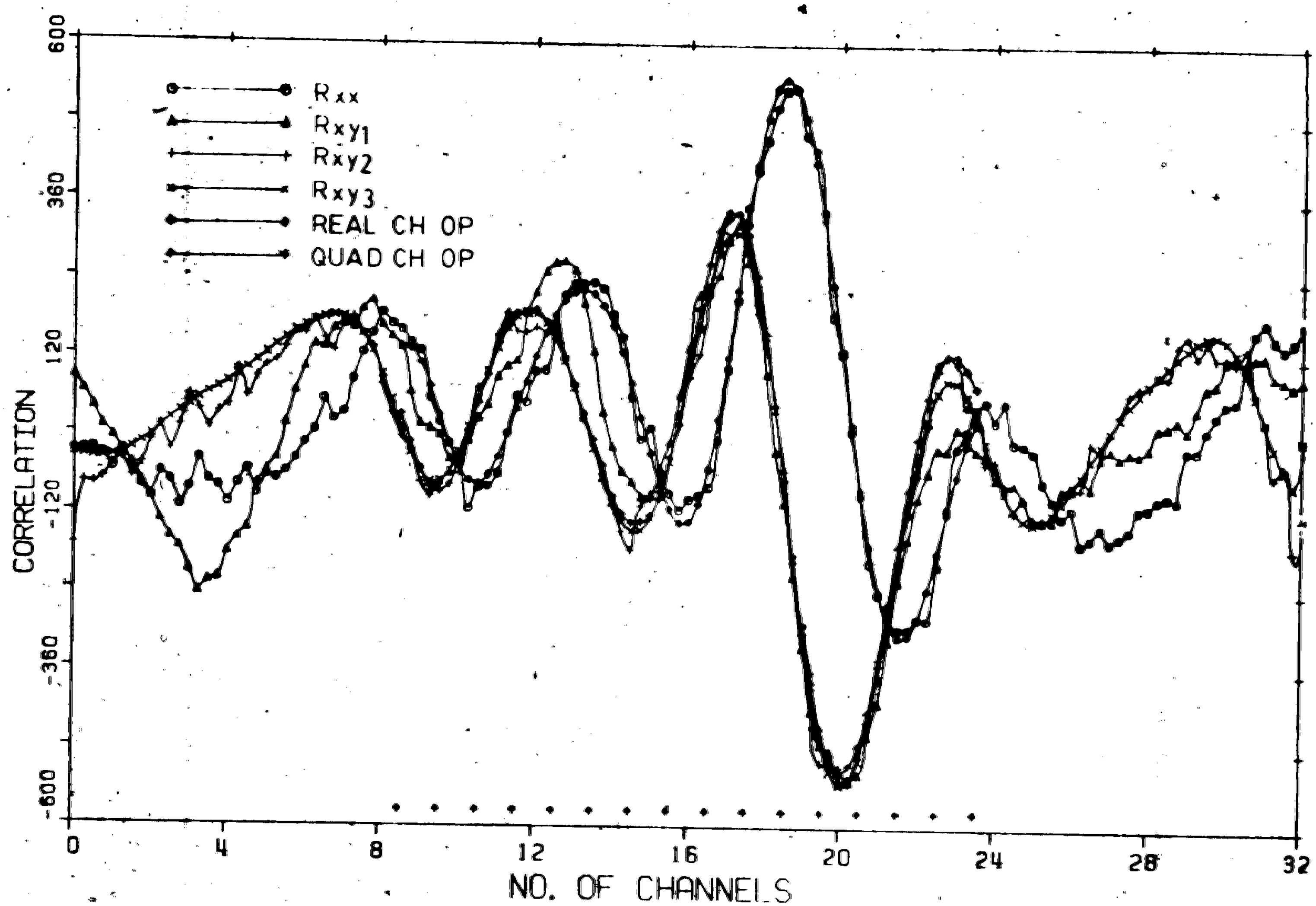


Figure. 4.2.8(a) Two point sources. 2.5 channels right weight = 2, 2.5 channels left weight = 1. Quantisation = YES. Input S/N ratio = 0.5.

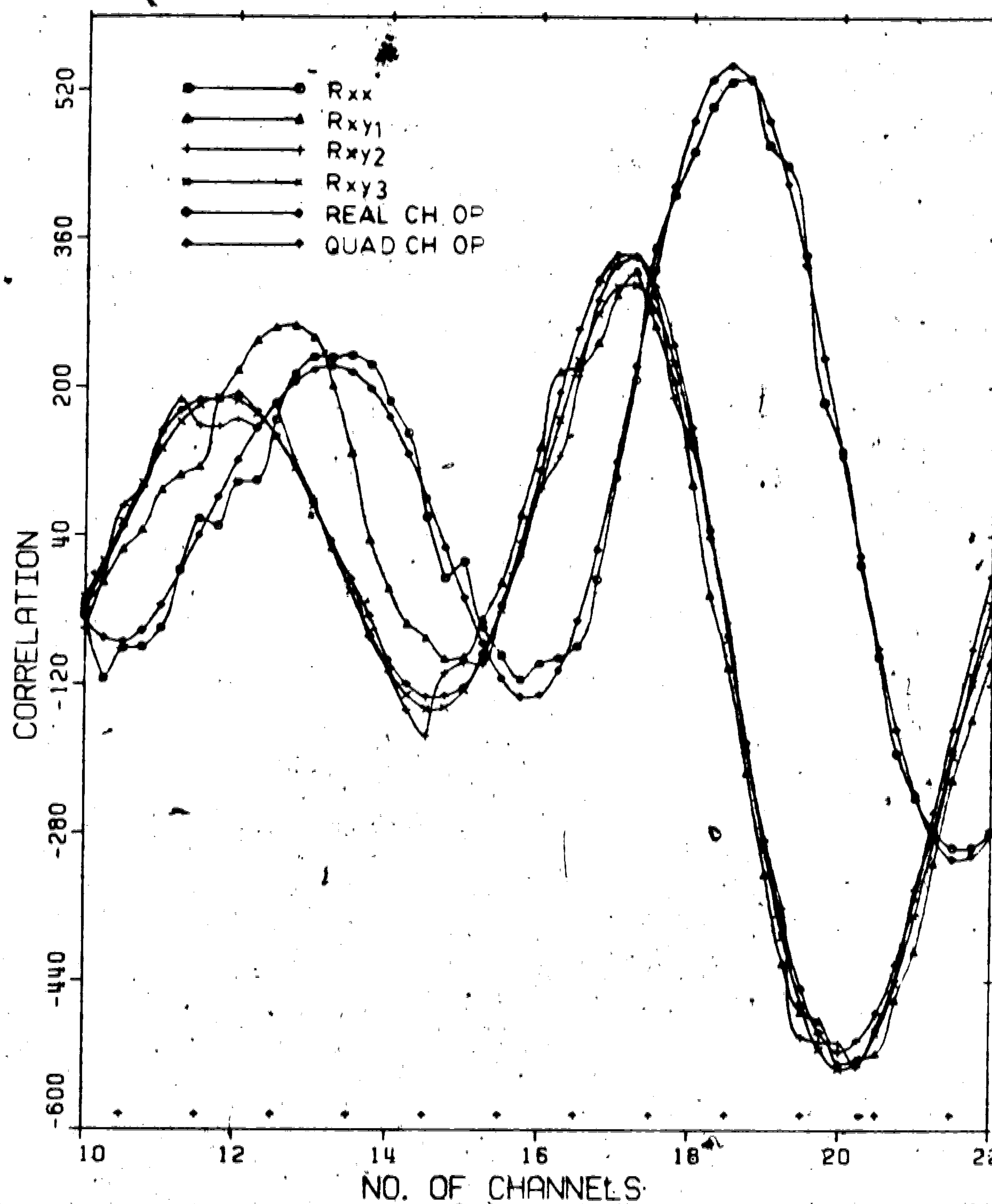


Figure 4.2.8(b) Expanded scale plot of central part of 4.2.8(a).

5. DESIGN AND IMPLEMENTATION OF THE SIGNAL PROCESSOR

The rough scheme and theory of operation has been presented in previous chapters. This chapter will describe in detail the scheme and environment of operation. The design and implementation of the processor will also be presented with emphases on an analysis of major system parameters and a choice of technology.

5.1 System Specification

The main design criterion of the 408 MHz system is to obtain a simple and reliable continuum channel, sharing as much existing hardware and software as possible. The solution chosen is to apply digital signal processing technology to eliminate or replace expensive and less stable analog equipment with digital hardware.

The 408 MHz digital signal processor will be a subsystem of the super-synthesis telescope. When completed, the signal processor will be used routinely for radio astronomical observations. Engineering aspects such as maintainability, servicibility, expandability and human interface are emphasised.

5.1.1 Environment of Operation

The 408 MHz Digital Signal Processor (DSP) will finally integrate with the 1420 MHz system and operate in an embedded mode. The host computer, a PDP 11/23, controls the operation of the whole synthesis telescope. The 408 MHz DSP communicates with the environment via four major ports as shown in figure 5.1.1. A 16 bit parallel bidirectional port links the signal processor with the host computer. A 16 bit parallel input port provides information from the SST control word, which is a common control word issued by the host computer for controlling various subsystems of the synthesis telescope. A parallel sidereal clock interface port provides the signal processor with sidereal time. Finally a serial port links up the processor with the system console terminal.

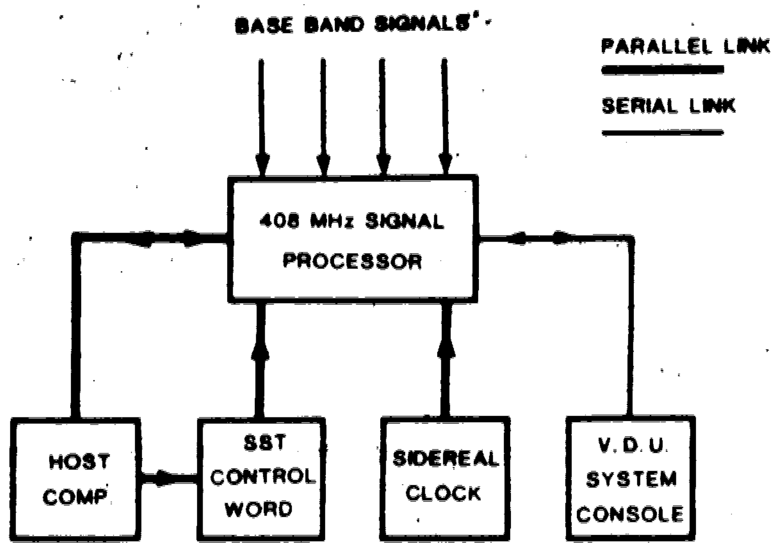


Figure 5.1.1 Communication environment of the 408 MHz signal processor

The host computer and the 408 MHz signal processor communicate in a master/slave mode. They are loosely coupled in terms of communications. Physically they are linked by two 16 bit unidirectional ports. Communication is based on asynchronous full hand shaking protocol. At the beginning of an observation run, the host computer will issue an initialisation command to the slave, followed by a list of observation parameters. The signal processor is expected to return a set of data after every major integration period, which is chosen to be about 8 seconds. Similarly the signal processor is commanded at the end of an observation run to stop various activities. The starting and stopping of observation runs are strictly controlled by the host computer, otherwise the signal processor operates in an autonomous mode.

The interface to the sidereal clock allows the signal processor to read the sidereal time independently. Thus hour angle dependent variables could be calculated to the required accuracy. A serial port to the system console interfaces the operator to the system through a Tiny Operating System Monitor (TOSMON).

The Super Synthesis Telescope (SST) control word is a 16 bit control word issued by the host computer and is distributed throughout the SST system. The SST control word controls the integration timing and crosstalk calibration in the 408 MHz system. The signal processor uses the timing signals to synchronise the resetting of correlator accumulators and the integration period. The de-assertion of the RESET line in the SST word starts a new major integration period.

5.1.2 System Analysis and Design

As shown in figure 2.2.3, the schematic signal flow diagram of the 408 MHz system, the signal path involves cross correlation, Hilbert transformation and interpolation. Throughout the conception of the scheme, a dedicated microcomputer was assumed to be available for the arithmetic operations and control functions.

The incoming baseband signals are digitised and sampled at 16 MHz rate. They are then cross-correlated in a hardware crosscorrelator. The output is a correlation function slowly varying in time. The shape of the correlation function is determined by the structure of the source seen at the particular hour angle and changes as the earth rotates. Since there is no fringe derotation before correlation, the correlation function is further modulated in phase at the fringe rate. To counteract such modulation, or perform fringe derotation, the correlation function must be sampled frequently enough as the shape changes with time. The maximum fringe rate of the 408 MHz system can reach one cycle every 17 sec. To obtain good orthogonality between the inphase and quadrature channels, the correlation function will be sampled after every minor integration period, which is about 100 ms. The Hilbert transform and interpolation are performed on the correlation function after every minor integraton period. The output is derotated to obtain the inphase and quadrature channels. The real and quadrature channels are further integrated up to a major integration period, which is about 8 seconds, before being transferred to to the required accuracy. A serial port to the system console interfaces the operator to the system through a Tiny Operating System Monitor (TOSMON).

The Super Synthesis Telescope (SST) control word is a 16 bit control word issued by the host computer and is distributed throughout the SST system. The SST control word controls the integration timing and crosstalk calibration in the 408 MHz system. The signal processor uses the timing signals to synchronise the resetting of correlator accumulators and the integration period. The de-assertion of the RESET line in the SST word starts a new major integration period

5.1.2 System Analysis and Design

As shown in figure 2.2.3, the schematic signal flow diagram of the 408 MHz system, the signal path involves cross correlation, Hilbert transformation and interpolation. Throughout the conception of the scheme, a dedicated microcomputer was assumed to be available for the arithmetic operations and control functions.

The incoming baseband signals are digitised and sampled at 16 MHz rate. They are then cross-correlated in a hardware crosscorrelator. The output is a correlation function slowly varying in time. The shape of the correlation function is determined by the structure of the source seen at the particular hour angle, and changes as the earth rotates. Since there is no fringe derotation before correlation, the correlation function is further modulated in phase at the fringe rate. To counteract such modulation, or perform fringe derotation, the correlation function must be sampled frequently enough as the shape changes with time. The maximum fringe rate of the 408 MHz system can reach one cycle every 17 sec. To obtain good orthogonality between the inphase and quadrature channels, the correlation function will be sampled after every minor integration period, which is about 100 ms. The Hilbert transform and interpolation are performed on the correlation function after every minor integraton period. The output is derotated to obtain the inphase and quadrature channels. The real and quadrature channels are further integrated up to a major integration period, which is about 8 seconds, before being transferred to the host computer as results.

5.1.3 Choice of Technology

The strategy used throughout the DSP design is to choose a powerful microcomputer and apply advanced[1] software technology. Advantage is taken of the microcomputer's processing power by performing functions in software whenever possible. The complex realtime synchronisation and multitasking software problems are tackled with the implementation of a realtime multitasking executive. To solve the otherwise almost insurmountable software development problem, a high level language is used with cross development on an established host computer.

The functions of reading the correlator, bit transposing the data, interpolation and Hilbert transformation must be done for each interferometer within one minor integration period, which is about 100ms. Rough estimations show that such a load is well beyond the capability of eight bit microprocessors like the 6800 and 8080. The MC68000, one of the most powerful 16 bit microcomputers available at the time of design, was chosen for its enhanced processing capabilities and the availability of high level language for software development.

5.2 Hardware

The 408 MHz signal processor hardware for four interferometers is shown in figure 5.2.1. The incoming signals are sampled at 16 MHz and quantised into 3 levels which are represented by 2 bits in digital form. The pair of signals that form a product are passed through a digital delay and then crosscorrelated in a 16 channel correlator. A system clock generator generates all the clock signals for the digital delays and correlators. The digital delays and correlators are arranged in bus structures and are connected via interfaces to the 68000 microcomputer. The 68000 has direct control over all the hardware subsystems except the quantisers.

---

[1] advanced as applied to microcomputers

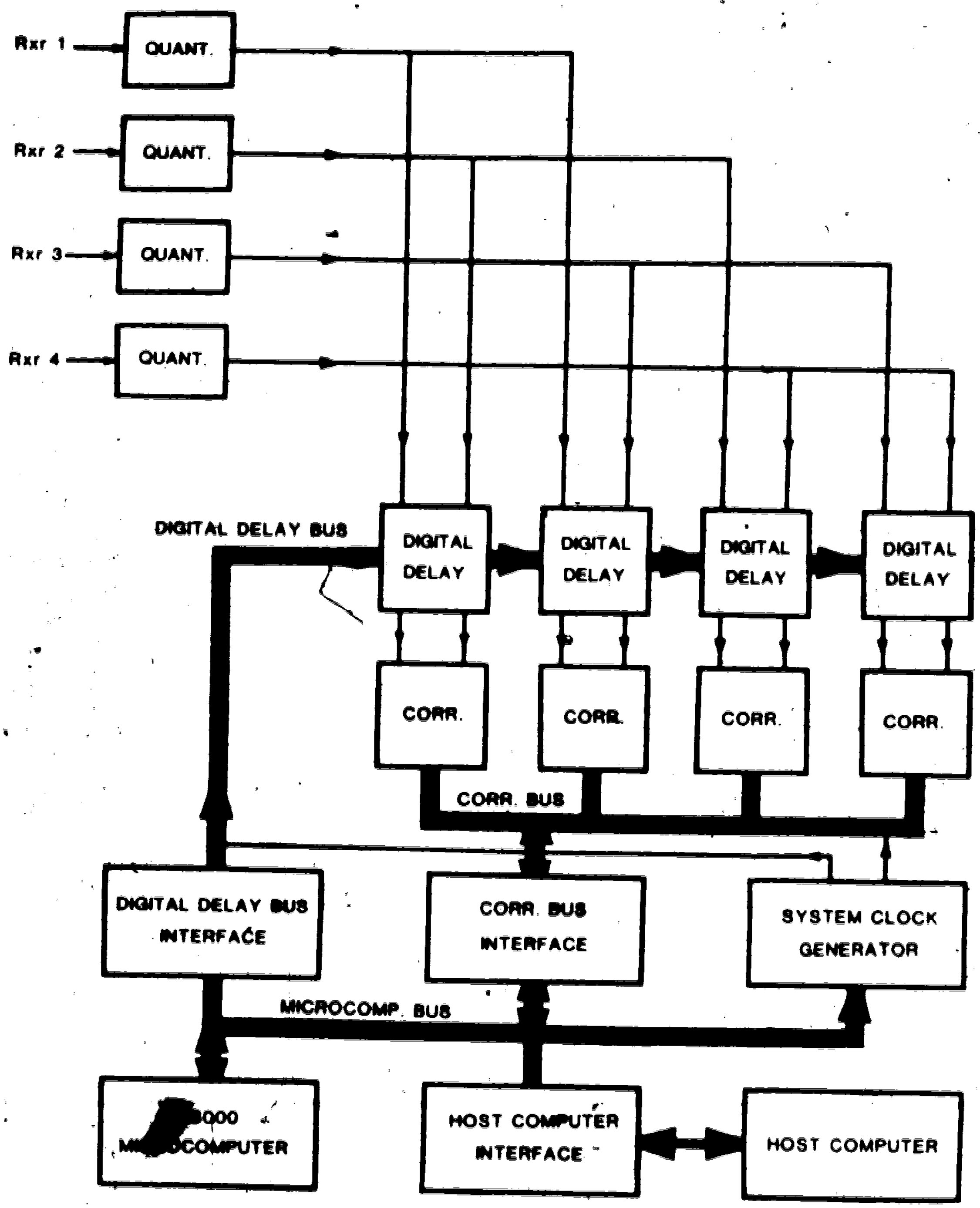


Figure 5.2.1 Hardware configuration of the 408 MHz digital signal processor for four interferometers.

5.2.1 The Quantisers

The quantisers are the interface between the analog system and the digital signal processor. Incoming baseband signals are sampled at 16 MHz and quantised into three levels represented by two bits. The quantisers used in the 408 MHz system are replicas of those used in the 1420 MHz digital spectrometer. Although a 3 level quantiser is simple in concept, the hardware implementation is not trivial. Besides having to work at high speed, the application demands very low crosstalk between individual units. Crosstalk between a pair of quantisers will contaminate the corresponding crosscorrelation function. Care has been taken to ensure high isolation between units by using separate power supplies and signal splitters for common clock signals. A typical radio source produces only 0.1% correlation in the 1420 MHz system. Preferably the level of crosstalk between quantiser units will be much lower.

5.2.2 The Digital Delay

The digital delay and the correlation function interpolation together form the path compensation delay. The digital delay is a coarse delay unit limited in resolution by the sampling rate of the analog signal. The main purpose of the digital delay is to shift the centre of the correlation function to the centre of the correlator, so that the interpolation algorithm can take care of the fractional part of delay value. The digital delay is required to insert from 0 to 32 units [1] of delay into either signal path for delay compensation. It is able to align the centre of the correlation function with the centre of the correlator to within 1 delay unit.

Figure 5.2.2 shows the schematic diagram of a digital delay unit. The digital delay unit is made up of shift registers and multiplexers. Each delay unit is made up of about fifty TTL IC's. Compared with the switched cable delays, the digital delays are small, inexpensive, and most of all stable and reliable.

---

[1]one delay unit = 1 sample period = 1/16 microsecond

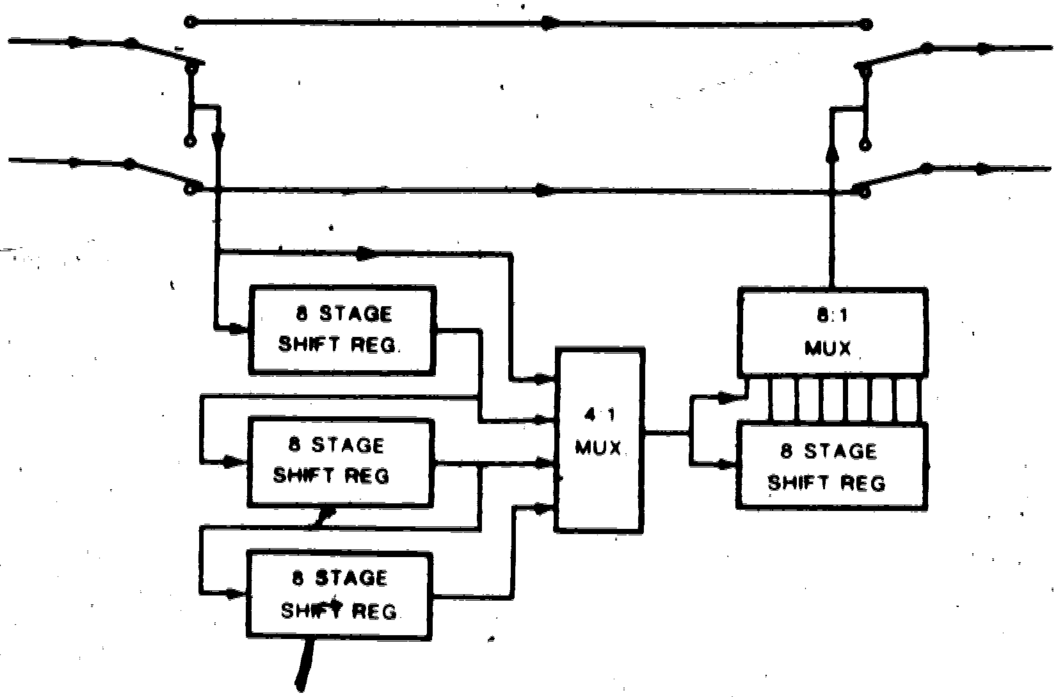


Figure 5.2.2 Schematic of the digital delay unit

5.2.3 The Digital Crosscorrelator

The correlator unit is a modification of the 1420 MHz spectrometer correlator[7]. Input signals of the correlators are quantised into levels of +1, 0, −1 and are represented by the two-bit code of 10, 00 and 01 respectively. There are nine terms in the product space, but the value of the product preserves the trilevel characteristic. The correlator uses a biased accumulation scheme. An offset count rate exists for the zero product. Two trains of count pulses at 16 MHz with 180° offset, controlled by the −1 and +1 product terms, are used to clock a ripple counter chain. A −1 is used to delete an offset count pulse and a +1 is used to gate in an extra count. The advantages are that only up-counting ripple counters need be used and the speed requirement is lower for subsequent stages. The schematic of the multiplier and integrator for one channel is shown in figure 5.2.3.

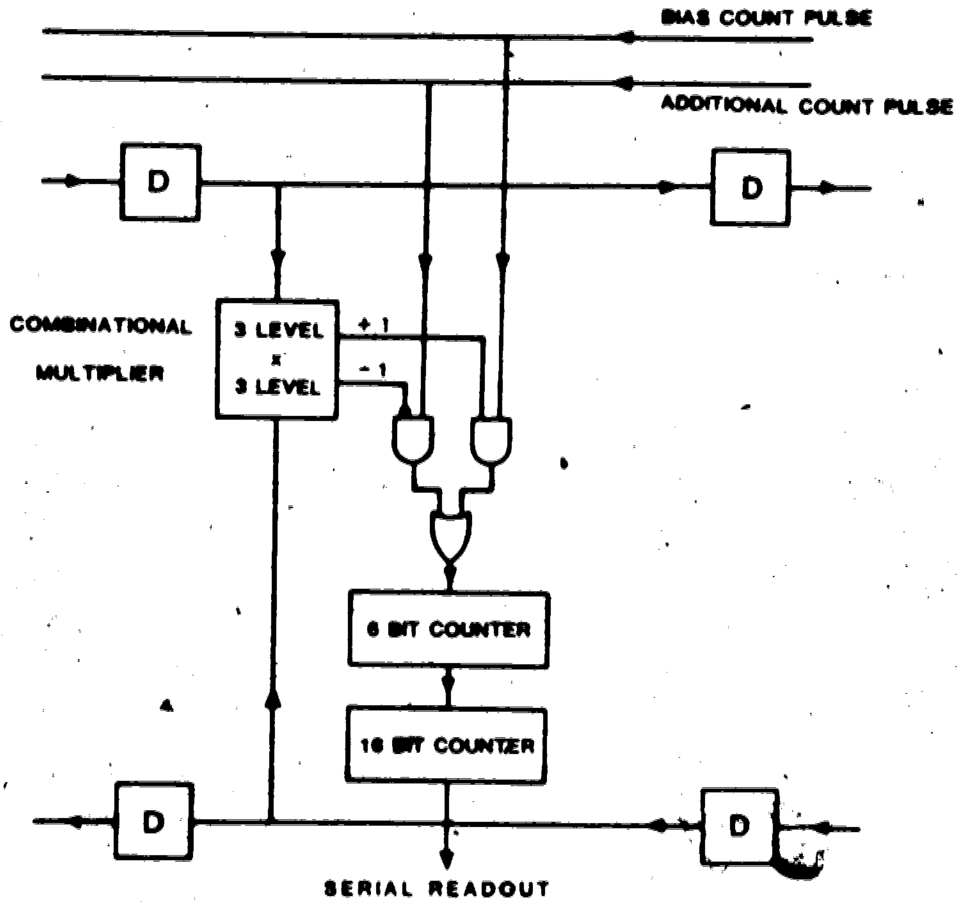


Figure 5.2.3 Schematic of the multiplier and integrator for one channel of the correlator.

Figure 5.2.4 shows the arrangement of the 16 channel crosscorrelator. The spacing between channels is one sample period, or 1/16 microsecond, in the delay domain. The delay units are simple shift registers. Each multiplier is implemented with a few gates for the coding scheme used. The ripple counter chain consists of a TTL divide-by-64 counter followed by a CMOS 16 bit counter. Since the CMOS counters are provided only with serial readout, the output of the correlator is organised in bit-serial channel-parallel form. A 16 bit word from the correlator consists of a bit from every channel. The data must be bit transposed to obtain a 16 bit word for every channel.

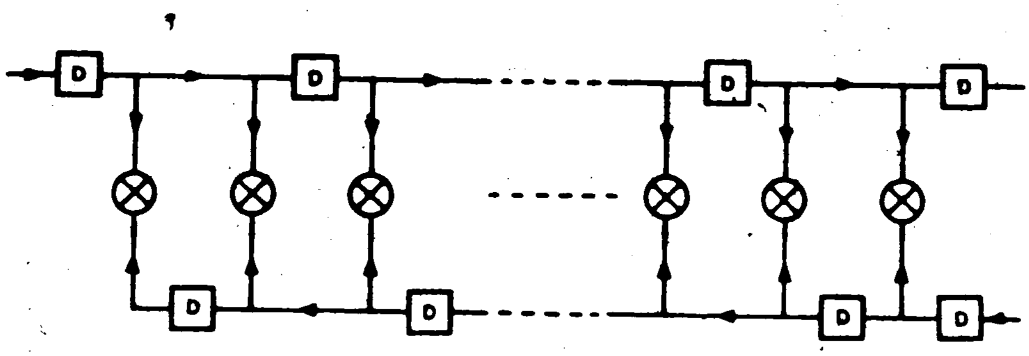


Figure 5.2.4 Arrangement of the 16 channel correlator.

5.2.4 The Microcomputer

The microcomputer hardware consists of a commercial single board computer MC68000 Design Module, an interface board and an erasable programmable read only memory (EPROM) board. The single board computer contains 32K bytes of random access memory (RAM), serial and parallel I/O ports and three 16 bit timers. The serial ports are used for system console communications and host computer communications during software development. The timers are used for generating the sidereal and observation real time clock interrupts. The MC68000 Design Module supports seven levels of interrupt which are used for communications with the environment and various internal timings.

The 68000 processor is a 16 bit general purpose machine. The architecture supports eight 32-bit data registers and eight 32-bit address registers. Most data manipulation instructions can operate on 8 bit, 16 bit or 32 bit data. These include the 16 bit by 16 bit multiplication and 32 bit divide by 16 bit instructions. Other powerful instructions include multiple move that can load and store any combination of the 16 internal registers with a single instruction. The instructions can operate with 14 addressing modes, including autoincrement and autodecrement via address registers. The architecture, the rich arithmetic instruction set, and the long operand length make the 68000 a very powerful arithmetic processor. The processor also supports operating system functions with supervisory mode of execution, privileged instruction set, and trap instructions.

The microcomputer controls various subsystems via parallel interfaces. The digital delays and correlators are organised as bus structures. Each unit is assigned a unique address. For example, the microcomputer programs the digital delays by simply sending a control word containing the direction and value of delay to each unit. Outputs of the correlators are read after every minor integration in a similar fashion.

5.2.5 Hardware Packaging

Figure 5.2.5 shows the equipment layout of the 408 MHz digital signal processor. The hardware is built into 24 inch rack mount modules. Each pair of quantisers is built into a 3.5 inch module. Two digital delay units are fabricated on one printed circuit board with the same dimensions as a correlator board. The correlator boards and digital delay boards are vertically mounted in a 17 slot chassis, which has enough room and power capacity to support 10 interferometers in simultaneous operation. The microcomputer, the EPROM board, the interface board, and the system clock generator are housed in a controller chassis. Because of the large physical size of the microcomputer boards, extender cards used in conventional bus structures for individual card access may produce undesirable effects on the relatively fast bus signals. A flexible bus is adopted instead, which allows the assembly to be opened up like a book. To allow easy access, the controller housing is built into a sophisticated slide and hinge device with 3 degrees of freedom of movement (see figure 5.2.5). The controller and correlator are supplied by a common power supply module. The whole assembly is housed in a screened room to avoid radio frequency interference from the fast digital signals.

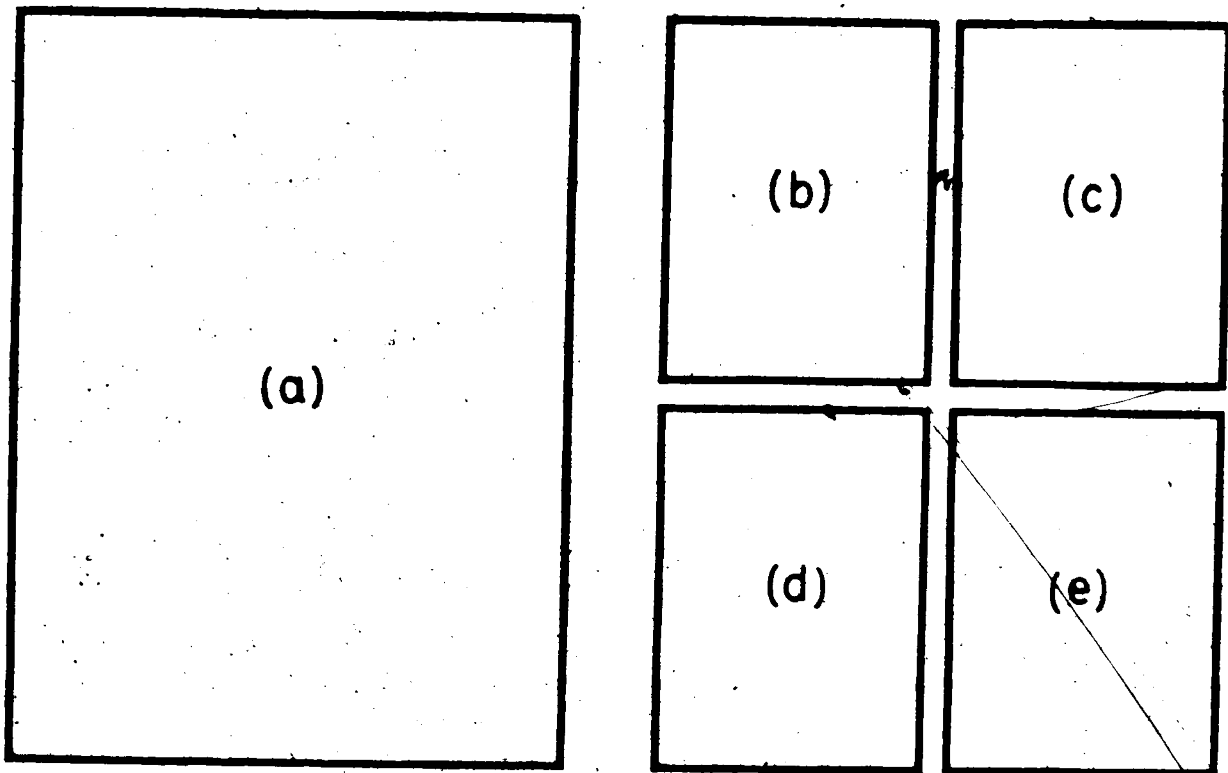
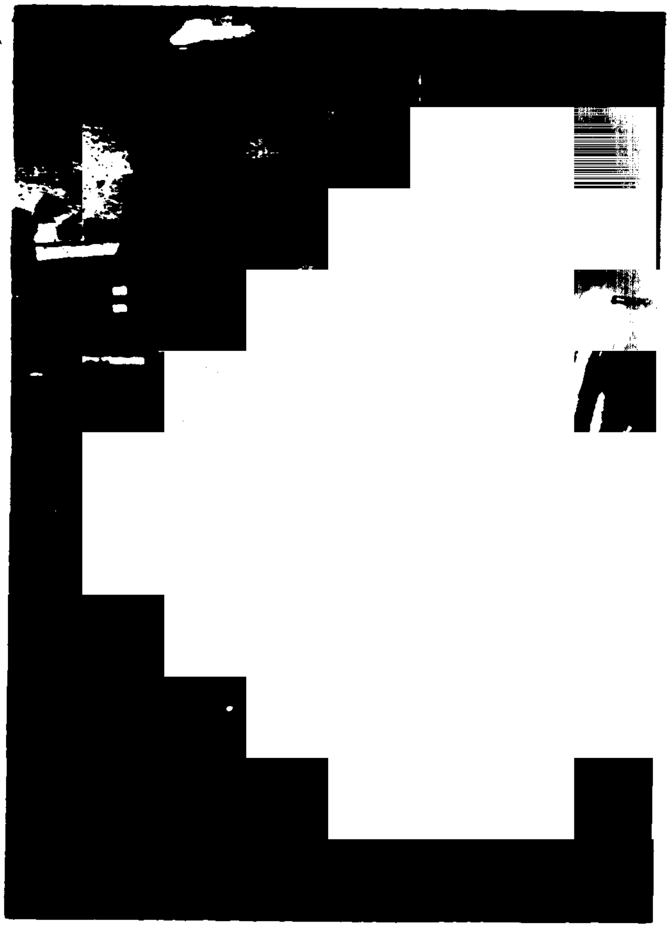
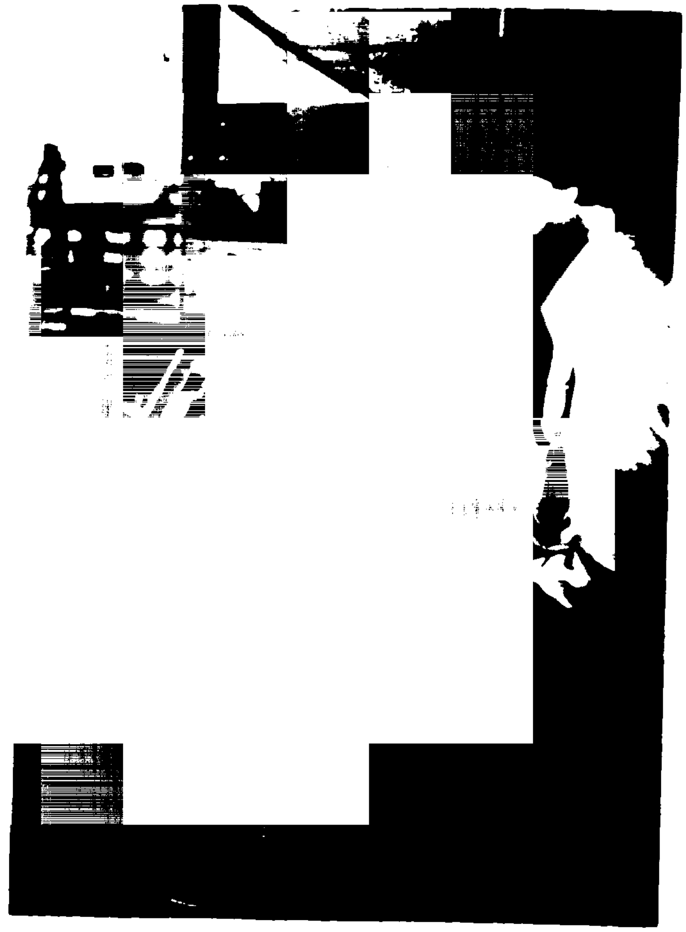
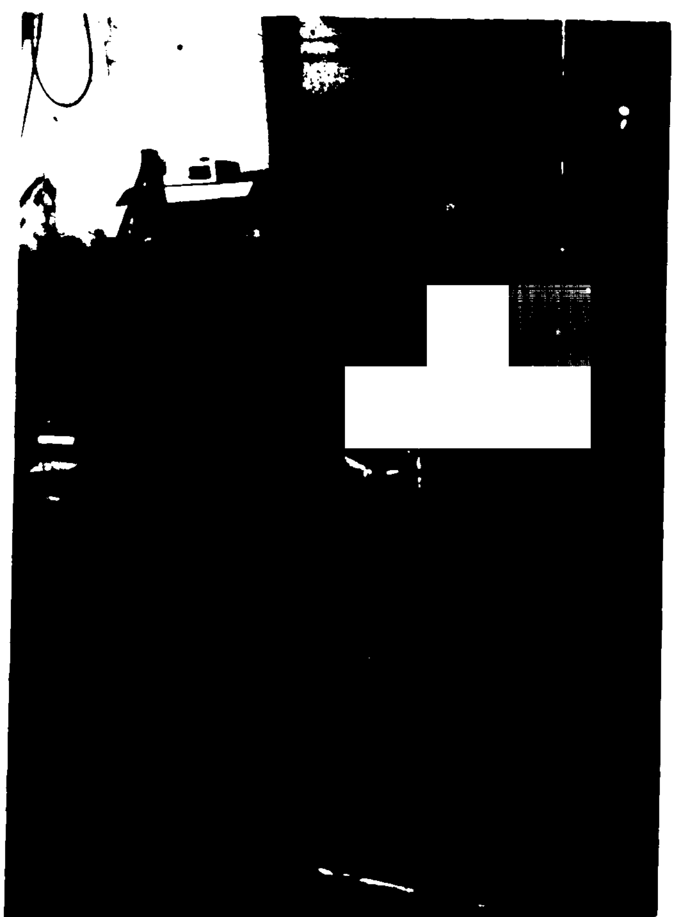
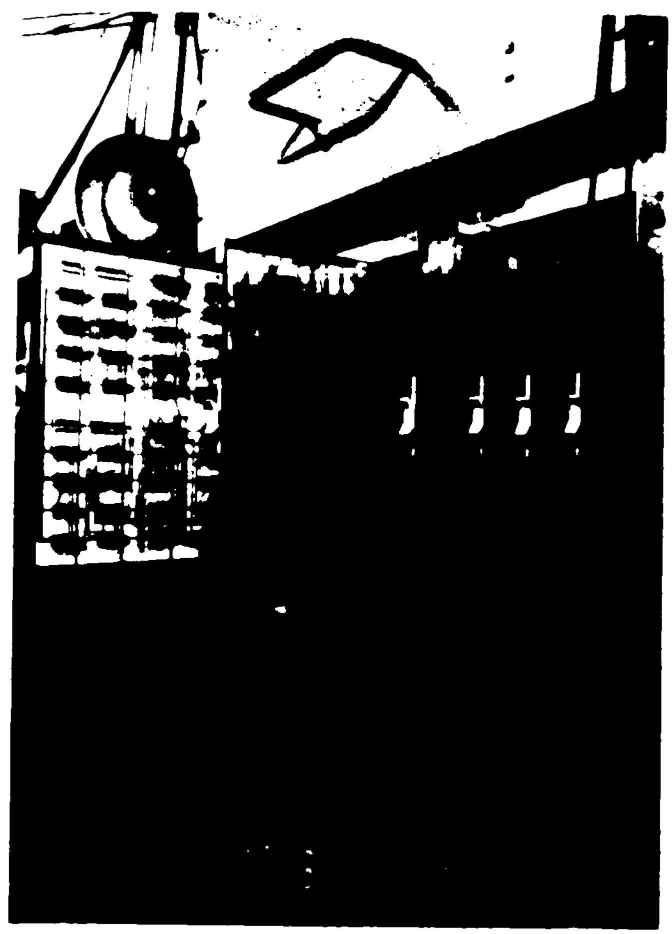


Figure 5.2.5 Physical packaging of hardware

Figure (a) shows the front view of the complete 408 MHz digital signal processor (DSP). The units from the top are the power supply module, the controller module, the correlator module, the video amplifiers and the quantiser modules.

Figures (b), (c), and (d) show the unfolding of the controller chassis that allows easy access to individual boards during operation. The 68000 Design Module single board microcomputer is situated on the top of the stack of boards as shown in (c). Below the microcomputer are the interface board, the EPROM board, and the system clock generator board as shown in (d).

Figure (e) shows the correlator module. The foreground circuits are the correlator boards. The fully extracted board is the digital delay board.

5.3 Software

The 408 MHz signal processor is a software intensive instrument. All of the control and many of the signal processing functions are performed in software. The realtime signal processing functions can be classified as follows:

1 Tasks every minor integration period (approximately 100 ms):

 a. Read correlator accumulators, clear accumulator;

 b. Bit transpose the serial correlator data;

 c. Set up the inphase and quadrature interpolation functions;

 d. Perform inner product on the correlator output with the interpolation functions;

 e. Multiply the inner product with sin(ϕ) and cos(ϕ) to perform fringe derotation;

 f. Accumulate the derotated results;

 g. Update the fringe angle ϕ and delay value with linear extrapolation by adding $\delta\phi$ to ϕ and $\delta\tau$ to τ.

2 Tasks every major integration period (approximately 8 seconds):

 a. Convert the accumulated result to the format required by the host computer;

 b. Send results to host computer;

 c. Calculate the fringe angle ϕ, delay value τ, and their linear extrapolation incremental values $\delta\tau$ and $\delta\phi$;

 d. Send out new values of digital delay.

The above list of functions is not simply performed in straight sequence. Some of the functions are asynchronous and have to be carried out concurrently with others. For example, the sending of results to the host computer is asynchronous, and the host computer is not guaranteed to respond within a certain period. At the same time other pressing jobs like variable updating must be completed within strict time limits. Besides the signal processing functions, other software requirements include communications with the host computer, interpretation of host computer commands, communications with the operator via the system consôle, and keeping track of sidereal time.

Facing the complicated realtime processing requirements, the Tiny Operating System was developed to centralise the functions of process switching and synchronisation with well defined boundaries, so as to alleviate part of the application software complexity. A monitor, the Tiny Operating System Monitor (TOSMON), was developed to improve the observability, controllability and testability of the system. TOSMON also aims at providing a friendly interface to the operator. Details of Tiny Operating System are described in the following sections.

5.3.1 The Tiny Operating System (TOS)

In conventional microcomputer software implementation without operating systems, more than one thread of execution is provided by interrupt servicing. In implementations with a single stack, the order of execution is strictly determined by the last-in-first-out nature of the stack. Such a sequence is too restrictive for the realtime application of the signal processor. The alternative solution is to employ a multitasking executive, the kernel of an operating system, to enable the processor to be shared among more than one thread of execution in any order. Since no realtime multitasking executive was available commercially for the 68000 at the time of software implementation, the Tiny Operating System (TOS) was developed by the author.

TOS is the kernel of an operating system designed to support realtime multitasking operations on the 68000. Many of the design ideas of TOS are based on Lister[16]. TOS provides the basic functions of processor dispatching, realtime task management, semaphore queueing facilities and primitive input and output. By providing these basic functions, TOS frees the observation software from tedious inter-task and realtime synchronisation and provides a private environment for each task.

5.3.1.1 Process and Process Descriptors

A process, sometimes called a task, represents an activity in the system. The execution of a process brings about the progress of an activity. From the system point of view, a process is an independent entity. The processor executes instructions on behalf of a process. From the processor's point of view, a process is a unit of execution, while a program or a piece of code is a series of instructions. Based on these concepts, a process exists independent of programs. Thus a piece of code can be shared by more than one process. Conversely, a process can be regarded as a thread of execution going through sequences of instructions.

In this implementation, each process is represented by a process descriptor in the system. A process descriptor is a data structure for saving all the information concerning a process when it is temporarily suspended. To continue the execution of a temporarily suspended process, the process must be brought back to the state it was in just before suspension. This requires that all the CPU registers and status sometimes called the volatile environment be saved. A process descriptor is made up of storage space for the registers, a field indicating the status of the process, pointers for different kinds of linkage and a name field for identification. A process may also be referred to by its "PROCESSID" which is the address of the corresponding process descriptor. Figure 5.3.2 shows the structure of a process descriptor.

The status of a process, as indicated by the process status field, is always in one of the following states:

1. Running;
2. Runnable;
3. Activated but not runnable; or
4. Deactivated.

A running process is one that the processor is currently executing. A runnable process is a process ready for execution and waiting for the CPU's attention. When the execution of a process comes to a halt because of input or output operations or

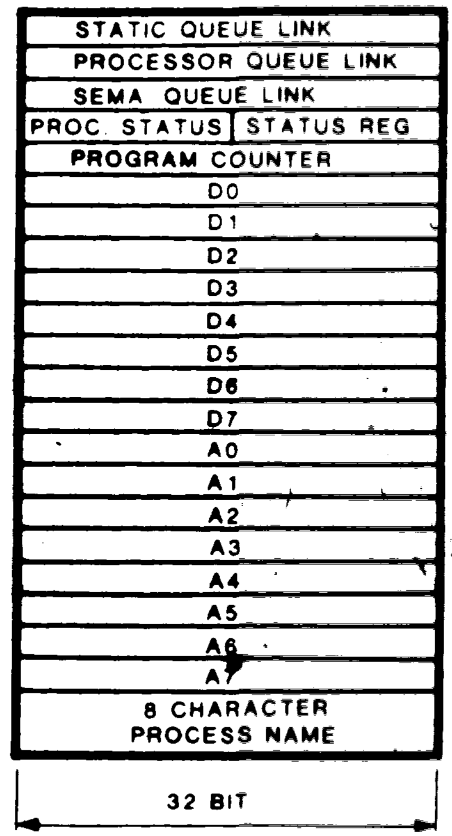


Figure 5.3.2 Structure of a process descriptor

waiting for some condition to occur, the process is marked as activated but not runnable. If the I/O operation is completed or the condition the process has been waiting for has occurred, the status will be changed back to runnable. When the execution of a process comes to a termination, the process will be marked as deactivated. A running or runnable process is always activated while a deactivated process is always unrunnable. Deactivated processes are not removed since most of them are realtime processes and execution will start again as soon as the appropriate time comes.

5.3.1.2 Processor Dispatcher

A Central Table in TOS contains the essential system parameters like realtime clocks and pointers to system data structures. Two of these pointers point to the static queue and the processor queue. Another pointer points to the current process

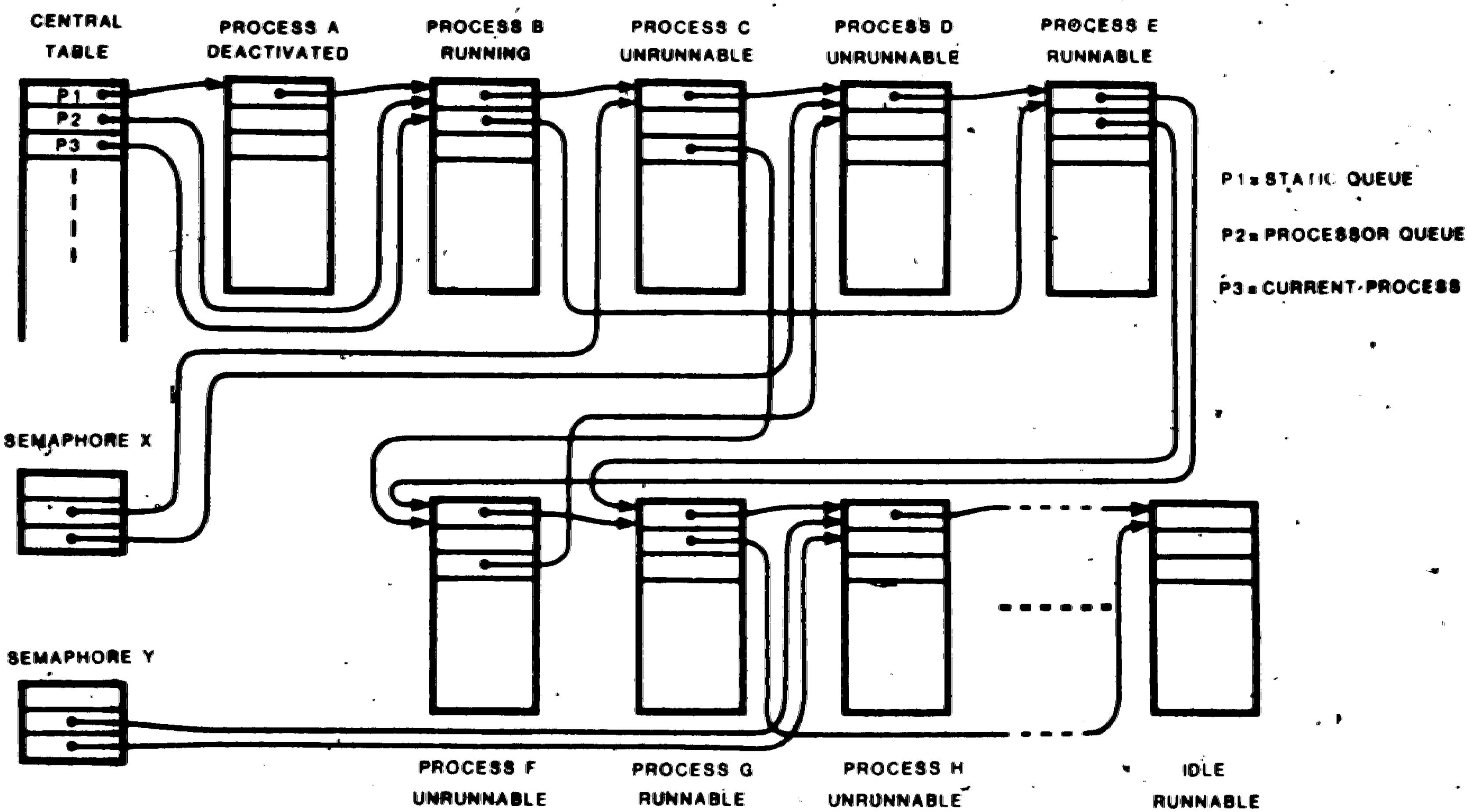


Figure 5.3.3 The central table and process queues. Linkage pointers are all pointing to the begining of process descriptors, or PROCESSID's.

under execution. The static queue links up all process descriptors in the system regardless of their status and determines their priority of execution. The processor queue links up only the runnable processes in descending priority. Processes in the processor queue are waiting for the CPU's execution. Figure 5.3.3 shows the state of the process queues and the Central Table at one instant. As shown in the structure of a process descriptor in figure 5.3.2, the first three fields in a process descriptor are the static queue link, the processor queue link and semaphore queue link. Process B is currently being executed. Processes B, E, G and process IDLE are linked up to form the processor queue and are waiting for execution. Process A has been completed and deactivated. Processes C, F and D are unrunnable and form a semaphore queue, queuing after semaphore X. Similarly, process H is queuing after semaphore Y. The IDLE process is a process executing just an idle loop. IDLE is always runnable and is always sitting at the end of the static and processor queues.

The order of execution is based on strict priority, that is, only the process at the top of the processor queue will receive any CPU attention. The strict priority system simplifies realtime multitasking programming considerations and has the additional advantage of being easy to implement. The processor dispatcher enables the processor to switch between processes. The processor dispatcher is invoked every time an interrupt or trap condition occurs. Process switching is required whenever the current process becomes unrunnable or terminated, or a process of higher priority is made runnable as a result of interrupt or trap operation. On occurrence of process switching, the processor dispatcher saves the context of the current process in the corresponding process descriptor, loads a new context from the process descriptor at the top of the processor queue, sets the new process to the current process, and executes the new current process.

The process descriptor, processor queue and processor dispatcher provide the basis of multitasking operation. Other functions like realtime job management and semaphore signal and wait all rely on the processor being able to switch execution from one process to another.

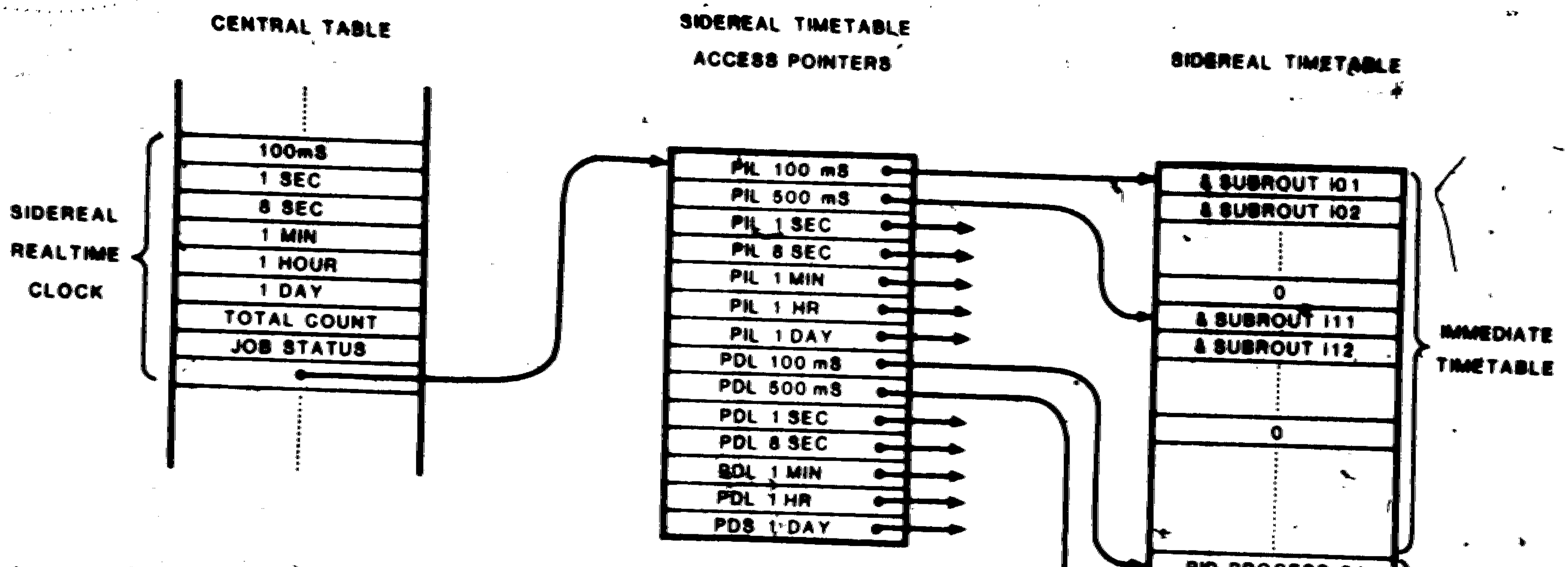
5.3.1.3 Realtime Manager

The realtime manager keeps realtime clocks up to 65536 days with resolution of 100ms, and performs realtime triggered operations. Realtime operations could be classified, into "immediate jobs" and "deferred jobs". Immediate jobs are those operations performed in supervisory environment during realtime clock interrupt service. Deferred jobs are terminated processes re-activated by the realtime manager. When a realtime clock interrupt occurs the realtime manager increments the internal software clock and performs a list of immediate jobs, activates a list of terminated processes, and transfers execution control to the processor dispatcher.

The software realtime clocks are organised as counters for 100ms, 1 second, 8 second, 1 minute, 1 hour and 1 day. Associated with each realtime clock are two time tables, the immediate time table and the deferred time table. The immediate time table contains the starting address of the immediate jobs, organised into lists of every 100ms, 500ms, 1 second, 8 seconds, 1 minute, 1 hour and 1 day. The list of immediate jobs is executed if the clock time is a multiple of the corresponding period. Similarly the deferred time table contains 'PROCESSID' as entries and the list of processes is activated at the appropriate clock tick. This particular implementation provides two real time clocks, the sidereal clock and observation clock which keeps the sidereal time and elapsed observation time respectively.

Figure 5.3.4 shows the structure of the sidereal realtime clock. For illustration, consider a realtime clock interrupt corresponding to the sidereal time of, say, 35 day 15 hr 23 minute and 13.0 second. When the interrupt occurs, the sidereal clock is identified and the realtime manager invoked. The realtime manager must then perform the following actions:

1. Increment the chain of sidereal realtime clock counters accordingly and mark the clock tick as a multiple of 0.1 sec, 0.5 sec. and 1 sec.
2. Access the 100 ms list of the immediate timetable via the timetable access pointers.



PIL = POINTER TO LIST OF IMMEDIATE JOBS
PDL = POINTER TO LIST OF DEFERED JOBS
& = ADDRESS OF
PID = PROCESSID OF

Figure 5.3.4 Structure of the realtime clock and timetable

3. Execute the list of subroutines in the order listed.
4. Repeat point 2 and 3 for the 500 ms and the 1 second list.
5. Access the 100 ms list of the deferred timetable via the timetable access pointers.
6. Access the process status field of each of the process descriptors in the list. If the status is deactivated, i.e. the realtime job is completed in time, the status is changed to runnable. Else the task is not finished within its given time. The process status is left unchanged and a bit is set both in the status word of the process descriptor and in the job status word of the realtime clock to indicate the error.
7. Repeat point 5 and 6 for 500 ms and 1 sec list.
8. Rearrange the processor queue according to the new process status.
9. Transfer control to the processor dispatcher.

The organisation of the timetables is highly flexible. The timetables reside in RAM and can be changed at run time. Splitting the real time functions into immediate and deferred jobs further enhances the flexibility of the system. A typical application is to log data from an external device as an immediate job and process the data in the background as a deferred job.

5.3.1.4 Semaphores and The Signal and Wait Functions

TOS provides semaphore queuing and dequeuing facilities for communication between processes. A semaphore is a data structure with a field of semaphore value, which is a small integer, a pointer to the head of the semaphore queue and a pointer to the end of the semaphore queue. A semaphore queue is a list of processes represented by their descriptors, queuing after a semaphore. Semaphores can only be operated on by the functions signal and wait. The wait function is equivalent to:

```
wait (semaphore)
begin  if (semaphore value is greater than or equal to 1)
                then    decrement semaphore value by 1 and proceed
```

```
            else  put the calling process on the semaphore queue, make it
                  unrunnable and remove it from the processor queue
end
```

The signal function is equivalent to:

```
signal (semaphore)
begin  if (semaphore queue is empty)
            then  increment semaphore value by 1
            else  free the process at the top of the semaphore queue
                  change the status to runnable, and insert the process
                  into the processor queue
end
```

A nonsharable resource can be protected by initialising a semaphore value to one and inserting wait functions in the requesting processes before accessing it.

```
Process A                         Process B
     |                                 |
     |                                 |
wait (writeterminal)                   |
write to terminal                 wait (writeterminal)
     |                            [blocked]
     |                                 |
signal (writeterminal)                 |
     |                            [freed]
     |                            write to terminal
     |                                 |
     |                                 |
     |                            signal (writeterminal)
     |                                 |
```

In the above example, the semaphore writeterminal is used to protect the terminal as a nonsharable resource. If process A is writing to a terminal, while process B executes the wait (writeterminal) function, process B will find that the semaphore value of writeterminal is zero and process B will be blocked by the semaphore. With the execution of signal (writeterminal) in process A, the process at the top of the semaphore queue, process B in this case, will be freed and allowed to access the nonsharable resource.

Process synchronisation with semaphore can be achieved by having one process signaling on a semaphore with the other waiting for the same semaphore. In the case of data processing and sending results to the host computer, the steps are synchronised by initialising the semaphore to 0 and inserting a signal in the data producer and a wait in the data consumer.

```
DATAREDUCTION                 TRANSMITDATA
      |                             |
      |                       wait (newresult)
reduce a set of data          [blocked]
write result in buffer              |
signal (newresult)                  |
      |                       [freed]
      |                       transmit result to host
      |                             |
```

The signal and wait operations on semaphore newresult guarantee TRANSMITDATA will not try to send a result before the result is ready.

The semaphore queuing facilities are provided by the semaphore link field in process descriptors. The queueing and dequeuing are first in first out. Since a process represents only one thread of execution and could only be queuing after one semaphore at any time, a single field in the process descriptor will suffice. Figure 5.3.5 shows the structure of a semaphore and figure 5.3.3 shows how processes

SEMAPHORE VALUE
HEAD OF QUEUE POINTER
END OF QUEUE POINTER

Figure 5.3.5 Structure of a semaphore

are queued after semaphores.

5.3.2 The Tiny Operating System Monitor (TOSMON)

The 408 MHz Digital Signal Processor (DSP) is a software intensive instrument. Signals appear only as data in memory locations once they have entered the microcomputer. The Tiny Operating System Monitor (TOSMON) is developed to monitor, control and test the system. TOSMON enables the operator to probe into various internal parameters during observation via the system console. Functions of TOSMON can be classified into system functions and observation functions.

The observation functions include:

- CD Continuously Display the results
- CO Continue Observation: continue a stopped observation.
- DI Display: display the output of every channel
- OM Observation Mode: set the mode of observation.
- OV Observation Variables: edit observation variables.
- OP Observation Parameters: edit observation parameters.
- SO Stop Observation: stop the current observation.
- HO Host: run under host control.
- LO Local: ignores host computer and generates all timing signals locally.
- AO Analog Output: configure the analog output channel.
- HE Help: print out help messages to the user.

The system functions include:

AC	Active: display all activated processes.
ET	Edit timetable: edit the deferred timetable.
MB	MACSBUG: transfer control to the manufacturer's resident monitor MACSBUG.
PQ	Processor Queue: take a snap shot of and display the processor queue.
PR	Priority: edit the priority, or static queue, of the system.
PS	Process Status: display and change the process status of processes in the static queue.
TI	Time: display and change real time clocks.
SQ	Semaphore Queue: display the semaphore queue.
IC	Incomplete: print out the processes which cannot be completed within their given time.
RU	Run: run a C program or a subroutine in core.

TOSMON is designed to provide a friendly human interface. It is interactive in nature and is intended to be used without a manual. The help command is always available and prints out help messages. When a valid command is received TOSMON prompts the user for parameters. Often, the user is asked to make a choice among several listed commands. The manual in appendix III contains a list of examples of user conversations with TOSMON. TOSMON has been proven to fulfil the design goals. It has greatly enhanced the observability, controllability and testability of the system. At the issuance of a two character command, the continuous display mode will print out the sidereal time, source hour angle, the elapsed time of observation, the interferometer spacing, path compensation delay, fringe angle, and the real and quadrature output of each interferometer every 8 seconds. TOSMON is also partly responsible for the speedy development and commissioning of the digital signal processor.

5.3.3 Observation Processes

The observation processes consist of a collection of processes performing various functions directly or indirectly related to observations. There are three processes handling observations directly: MAJORUPDATE, MINORUPDATE and DATAREDUCTION. MAJORUPDATE and MINORUPDATE are responsible for keeping various hour angle-dependent variables updated. DATAREDUCTION reduces the 16 channel correlator output to the final inphase and quadrature channel outputs. Other servicing processes include the READHOST which reads and interprets host computer commands, SENDHOST which sends the inphase and quadrature channel outputs to the host computer after every major update preiod, and READSIDEREAL which reads the hardware sidereal clock.

DATAREDUCTION and MINORUPDATE are listed in the observation clock deferred timetable 100ms list. MAJORUPDATE is listed in the 8 second list of the same timetable. DATAREDUCTION performs a signal function on semaphore Result after every major integration period which triggers the SENDHOST process to send out the results to the host computer. The READHOST process is always waiting for commands or data from the host computer, and is activated whenever the host sends a word to the DSP. READSIDEREAL is listed in the one minute deferred timetable of the sidereal clock. When activated, READSIDEREAL reads the hardware sidereal clock and sets the internal software sidereal clock to the new value.

5.3.4 Software Development and Implementation

Software development represents a major part of the total development effort. The 408 MHz Digital Signal Processor software is cross developed on a PDP11/45. It was decided that high level language should be used whenever possible for ease of development and maintenance. The high level language C was chosen for the following reasons:

1. Structure: C is a language which uses if-then-else constructs and while, for and repeat loops instead of goto's and do loops.
2. Reentrant: All subroutine local variables are stored in the stack and are

created as the routine is entered, thus allowing a single routine to be shared by more than one process.

3. Relatively few restrictions: C is a relatively unrestrictive language that allows access of addresses of variables and subroutines, bit manipulation, and writing to absolute addresses. These features are valuable in a microcomputer operating environment.
4. Double precision floating point arithmetic: C supports single and double floating point variables and double floating point arithmetic, which is preferred in fringe phase angle calculation.
5. Availability: the C cross-compiler was one of the very few cross-compilers available for the 68000 at the time of software development.

All of the TOS code was written in assembler language, while the whole TOSMON was written in C. The observation processes were written in a mix of C and assembler language. The size of source code breaks down as follows:

	Assembler	C language	
TOS	1500	0	lines
TOSMON	0	1900	lines
Observation Processes	700	400	lines
Common Data Structure	850	0	lines

A total of 5500 lines of source code were written for the 68000. This source code did not include the FORTRAN generated trigonometric and interpolation function tables. The target code for the 68000 was stored in 48 K bytes of Erasable Programmable Read Only Memory (EPROM). Another 2000 lines of C program has also been written to supplement the cross development package on the PDP11 host computer.

During development, source code written in C was edited and stored in the host computer. A C cross-compiler compiled the source code and translated the object code into an S-record file, which is a hexadecimal representation of the object code in ASCII. The S-records were then down loaded into the RAM area of the 68000. The target code was programmed into EPROM as each software module became reasonably stable.

5.4 Scheme of Operation of The 408 MHz DSP

5.4.1 Initialisation and Track Calculations

All the software of the 68000 microcomputer resides in EPROM. At power-up the 68000 microcomputer performs a list of initialisation functions. These include setting up all the system data structures such as realtime clocks, timetables and process descriptors, and initialising all the programmable I/O devices. The 68000 then starts execution of the runnable processes, which are TOSMON and IDLE at power-on time. The signal processor is now ready to receive commands from the system console and the host computer. The starting up of the signal processor is fully automatic, which implies the signal processor can restart automatically after power failure with an observation initialisation from the host computer.

To start an observation, the host computer issues an initialisation command, followed by a list of parameters. The list of global parameters includes the mode of observation, frequency of observation, and source coordinates. For each interferometer, the parameter list consists of the baseline length, track errors in the three dimensions of equatorial coordinates, and collimation error. After receiving these parameters, the microcomputer calculates a list of constants and the track error effects for the observation.

The three components of the track error, x, y and z, are defined in terms of departure of the *East antenna from its supposed position by*:

x towards the east,

y in the equatorial plane at right angles to x, measured positive towards the north, and z below the equatorial plane.

The x-direction track error is just an increase in baseline length. The y-component effectively changes the source hour angle by [17]:

$$\delta HA = -y/B \tag{5.4.1}$$

where δHA is the source hour angle error.

The effect of the z-component track error is to introduce a phase angle independent of hour angle

$$\phi_{z_i} = \frac{2\pi}{\lambda} z \sin(DEC) \tag{5.4.2}$$

where DEC is the source declination.

The effects of track errors are stored for each interferometer and recalled each time the phase angle and the path compensation delay are recalculated.

After the initialisation, the observation will not actually start until the falling edge of the reset pulse of the SST control word arrives which signals the beginning of an integration period.

5.4.2 The Observation Variable Update and Event Timing

During the observation, various hour angle dependent variables have to be updated every minor integration period. Among these are the phase angle and the path compensation delay. Phase angle is given by (3.4.6)

$$\phi(t) = \frac{B\omega_o}{c} \cos(DEC) \sin(HA)$$

$$= \frac{2\pi B}{\lambda_o} \cos(DEC) \sin(HA) \tag{5.4.3}$$

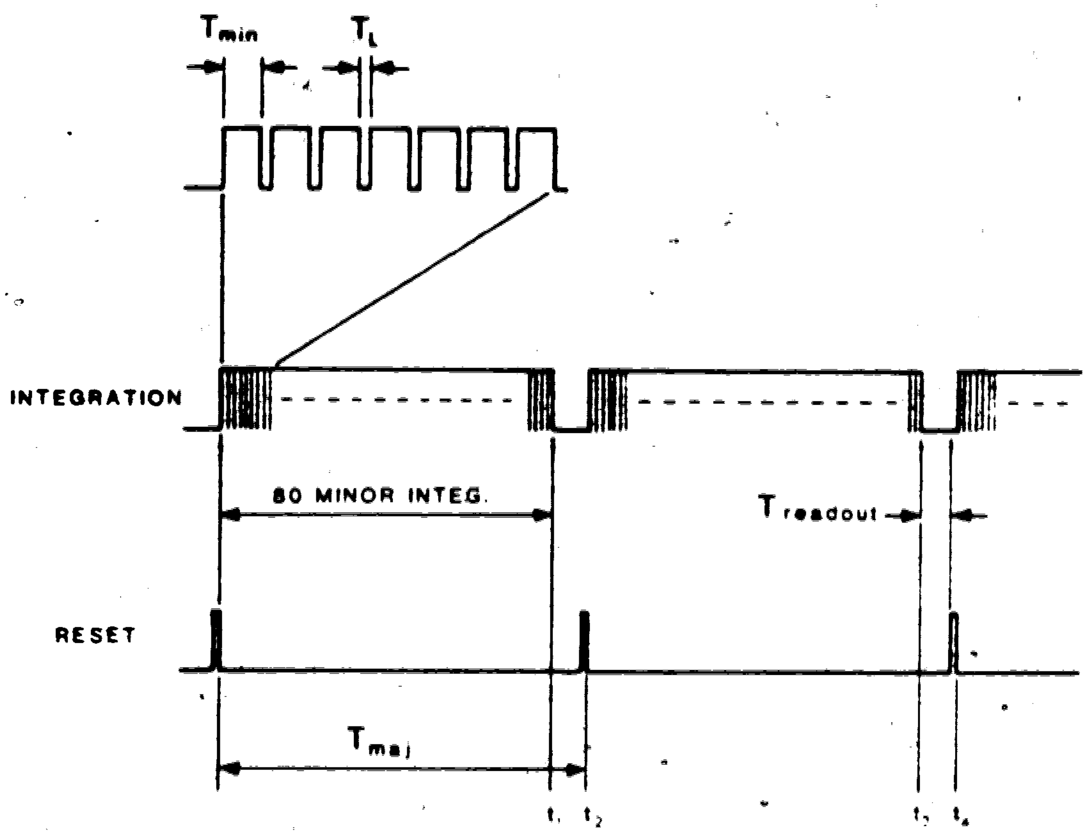


Figure 5.4.1 Timing diagram of various events

The geometric delay expressed in source coordinates and baseline length is

$$\tau(t) = \frac{B}{c} \cos(DEC) \sin(HA) \qquad (5.4.4)$$

The term $2\pi B/\lambda$, is of the order of 10^3 for large spacings. To obtain an accuracy in $\phi(t)$ of the order of $1°$ requires the cos(DEC) and sin(HA) to be evaluated to an accuracy of 1 ppm which implies cosine and sine functions should preferably be evaluated with double precision floating point arithmetic. The microcomputer used does not have a hardware floating point processor and software evaluation of double precision floating point arithmetic is very time consuming. A linear extrapolation scheme is used instead to update the phase angle and path compensation delay

$$\phi(t+Tmin) = \phi(t) + \frac{d\phi}{dt} Tmin$$
$$= \phi(t) + \delta\phi \qquad (5.5.5)$$

$$\tau(t+Tmin) = \tau(t) + \frac{d\tau}{dt} Tmin$$
$$= \tau(t) + \delta\tau \qquad (5.5.6)$$

where Tmin = minor integration period

$$\delta\phi = \frac{d\phi}{dt} Tmin$$

$$\delta\tau = \frac{d\tau}{dt} Tmin$$

For every major integration period ϕ, $\delta\phi$, τ and $\delta\tau$ are calculated for every interferometer. Equations (5.5.5) and (5.5.6) are used to extrapolate the values of ϕ and τ within one major integration period. Appendix II shows that such a scheme provides sufficient accuracy even for operation at 1420 MHz.

Figure 5.4.1 shows the timing of various events during observation. The major integration period is synchronised to the reset signal of the SST control word. The falling edge of the reset pulse signals the start of a major integration period. Integration in the correlators starts immediately at the beginning of a major integration period and lasts through one minor integration period which is determined by a hardware timer. The correlator accumulators are read and another minor integration is started immediately by restarting the timer. While the correlators are integrating, DATAREDUCTION, a software process, processes the previous set of correlator outputs. The MINORUPDATE then updates the phase angle and path compensation delay using linear extrapolation. The outputs of DATAREDUCTION accumulate for one major integration period as the inphase and quadrature channel outputs.

After the 80th minor integration, the timer is not restarted until the next falling edge of the reset pulse. The internal observation realtime clock is driven by the integration timer interrupt. Restarting the timer at the next falling edge of the reset pulse effectively forces synchronisation of the observation clock to the SST control word timing. This period of no integration is devoted to data logging in the SST system. The host computer reads the outputs from various subsystems during this period. For the 408 MHz signal processor, data is ready to be sent to the host a few milliseconds after the last minor integration. The process SENDHOST is responsible for sending data to the host computer. At the same time, t_1 in figure 5.4.1, MAJORUPDATE starts to calculate the phase angle, path compensation delay, and their incremental values for the major integration period starting at t_4. These new values are stored in buffers and are not copied into actual variable locations until t_4. By this means, MAJORUPDATE can safely extend into the integration period starting at t_2 without interfering with DATAREDUCTION.

The updating of hour angle is asynchronous to the major or minor integration period. Each tick of the sidereal clock increments the hour angle by the equivalent of 100 ms sidereal time.

In the normal mode of operation, timing signals come from the SST control word and the hardware sidereal clock. A "local" mode of operation is provided for development and testing. In the local operation mode all timing signals are simulated locally. The sidereal clock runs on an interpolation basis while the SST control word signals are generated locally. The host computer communication responses are also locally simulated. The local mode allows stand-alone operation of the signal processor. In this mode, over 90% of the hardware and software could be developed and tested independently, which greatly improved the testability of the system and significantly reduced the telescope time required for development.

5.4.3 Phase Switching

In the 1420 MHz system, a typical strong source can only produce 0.1% correlation in the correlator. Other sources may be 100 times weaker. In the complicated receiver system, spurious signals may be picked up by both channels of the interferometer, or crosstalk may exist between the two channels in their long signal path from the antenna to the correlators. This undesirable correlation may produce spurious responses in the interferometer.

Phase switching can be applied to suppress such spurious response by a few orders of magnitude. In phase switching, a 180° phase change is applied to the local oscillator signal of one of the antennas. The correlation that exists before the phase switching point will change sign while that contributed after will not. By successively switching the phase and subtracting one response from the other, the time invariant spurious response can be eliminated.

In the 408 MHz system, phase switching will be applied to the first mixers, which will be housed in the focus boxes of the antennas. Since the antennas are physically separated from each other by reasonable distances, chances of crosstalk introduced into the circuitry before the first mixers are very much reduced.

Since the fringe derotation is done after correlation in the 408 MHz system, correlation caused by spurious signals and/or crosstalk will also be derotated. This spurious correlation will effectively change at the fringe rate. The rate of phase switching must be fast enough that the spurious correlation does not change appreciably within successive phase switch half cycles. To eliminate the changing spurious correlation, phase switching is applied to alternate minor integrations. The phase is effectively switched at 5 Hz which is 85 times the maximum fringe rate at 408 MHz.

There are two modes of operation for the SST system, the calibration mode and the observation mode. The observation mode is the normal mode of operation. In the calibration mode, one of the local oscillator signals' phase is switched 90° after every 9 major integration periods. Running the calibration mode on a point source allows easy calibration of system gain and collimation errors. In normal operation, a 12 hour observation is usually preceded and followed by 20 minute calibration runs.

6. TESTING AND OBSERVATION

The 408 MHz digital signal processor is a complex system. To ease development and to ensure system integrity, each subsystem was tested thoroughly before system integration. Section 6.1 describes the testing of individual subsystems, while section 6.2 describes the testing of the integrated system with simulated signals. Since the digital signal processor was developed before the analog system, the final system had to be tested with signals from the 1420 MHz system. Section 6.3 describes a map of 3C66 made with the 1420 MHz front-end and the digital signal processor. The map is compared with another map of the same source made with the 1420 MHz continuum system.

6.1 Testing of Subsystems

6.1.1 Testing of the Correlators and Quantisers

The signal processing algorithms used in the digital signal processor all operate on the crosscorrelation functions of the incoming signals. It is essential that the hardware producing the correlation function, which includes the quantisers, digital delay unit and the digital crosscorrelator, operate properly. The digital delay unit is a relatively simple subsystem; imperfection is more likely to creep into the quantisers or the crosscorrelators. Tests described in this subsection are aimed at testing these two subsystems.

6.1.1.1 Uncorrelated Noise Test

The uncorrelated noise test is intended to check for spurious correlation produced within the digital signal processor itself, especially the quantisers. The uncorrelated noise sources are independent wideband (up to 100 MHz) noise generators followed by a 4 MHz fifth order Butterworth lowpass filter. Four independent noise sources were fed into the four quantisers and four pairs of correlation products were formed. The outputs of the crosscorrelators were accumulated over a period of six hours and the resultant correlation functions are plotted in figure 6.1.1. From figure 6.1.1, any correlation which exists is of the

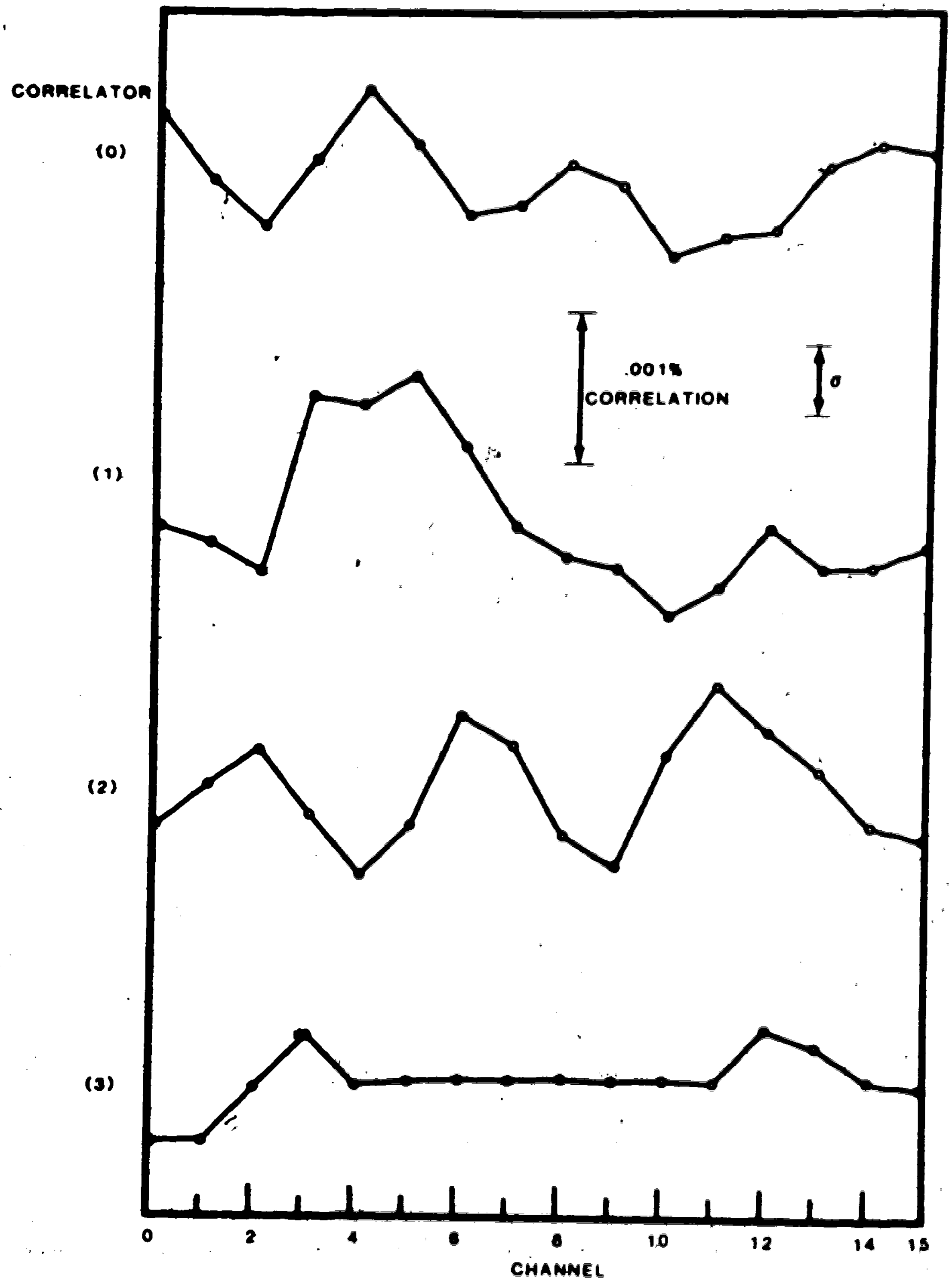


Figure 6.1.1 Uncorrelated noise test of correlators. Each curve is a 16 point crosscorrelation function of two uncorrelated noise streams produced by the digital correlator over a six hour integration.

order of 10^{-5} or less. Suppressed further by phase switching, any spurious correlation produced by the digital signal processor itself will be negligible compared with the receiver noise even for a 12 hour integration of the signal from a point source.

6.1.1.2 Correlated Noise Test

The correlated noise test is aimed at measuring the shape of the correlation function at more than one level of correlation. The test used a configuration similar to the uncorrelated noise test in 6.1.1.1, but applied only to one correlator. A resistive network was placed between the two analog signal paths before the inputs to the quantisers to introduce some crosstalk. Crosstalk levels of 2% and 15% were tried with 10 minute integration each. The resultant scaled correlation functions are plotted in figure 6.1.2.

The spectral shape of the noise sources is effectively determined by the lowpass filters. The shape of the correlation function in figure 6.1.2 agrees well with the Fourier transform of the pass-band shape. The shapes of the correlation function for 2% and 15% crosstalk are very similiar. This confirms that the Van Vleck correction similar to equation (3.1.6) is not required for a 3 level by 3 level correlator up to around 15% correlation.

6.1.2 Testing of Quadrature Channel Generation

The quadrature channel is generated numerically from the crosscorrelation function in the 408 MHz system. Due to the finite correlator length, the quadrature channel gain is less than the real channel and is delay dependent[19]. Since derotation is done after correlation, the orthogonality of the quadrature channel is crucial to fringe derotation[20]. Two tests were performed to ensure the quality of the quadrature channel.

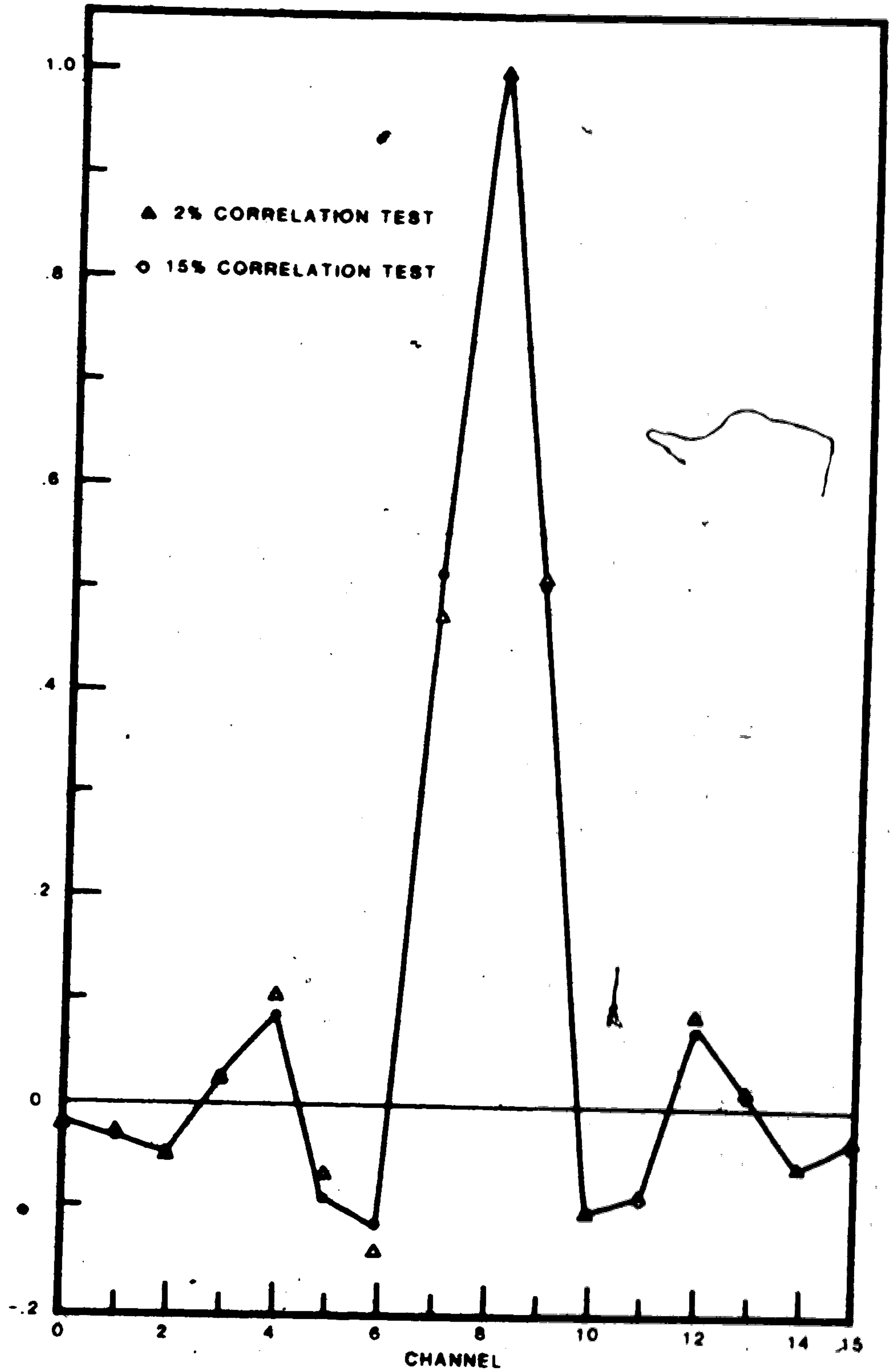


Figure 6.1.2 2% and 15 % correlation test of correlators. Input signals are correlated bandlimited baseband noise of 4 MHz bandwidth. The correlator channels are spaced at 1/16 microsecond.

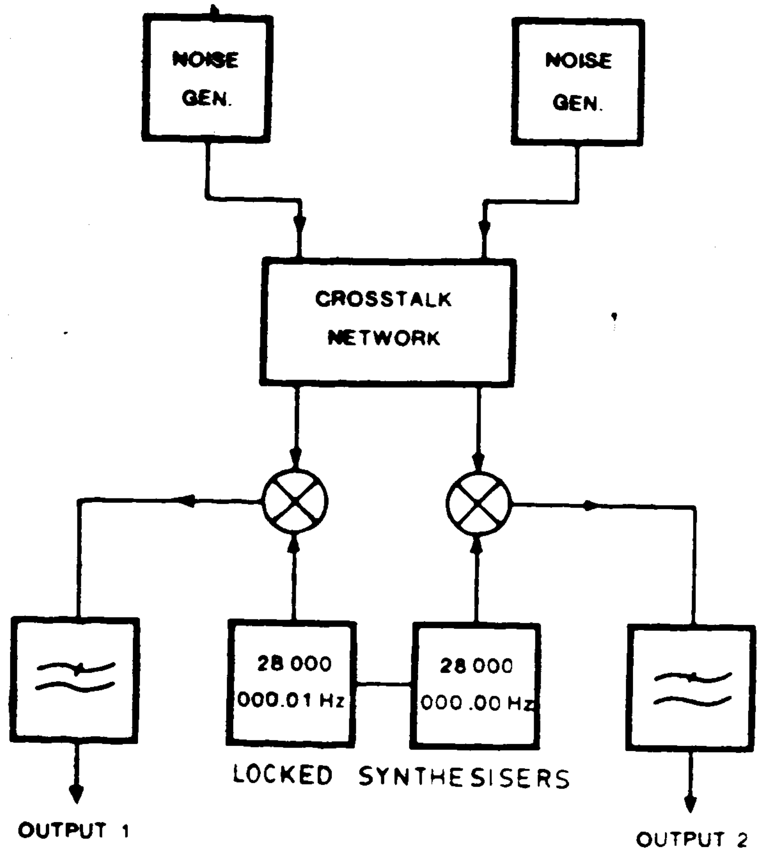


Figure 6.1.3 Simulation of point source with 100 second fringe rate.

6.1.2.1 10 mHz Sine Wave Test

The first test used a simulated point source with 100 second fringe rate. Figure 6.1.3 shows the circuit for producing the simulated source. A resistive crosstalk network introduces correlation into two streams of wide band noise from two noise generators. The correlated noise streams are down mixed in two single side band (SSB) mixers. Two locked synthesisers are used to generate signals at 28.000000000 MHz and 28.000000010 MHz respectively. These signals are used as local oscillator signals for the two SSB down-mixers. The 10 mHz difference in L.O. frequency results in a simulated point source with 100 second fringe rate.

In the 10 mHz sine wave test, the simulated point source signals were fed into the crosscorrelators. Interpolation and quadrature channel generation algorithms

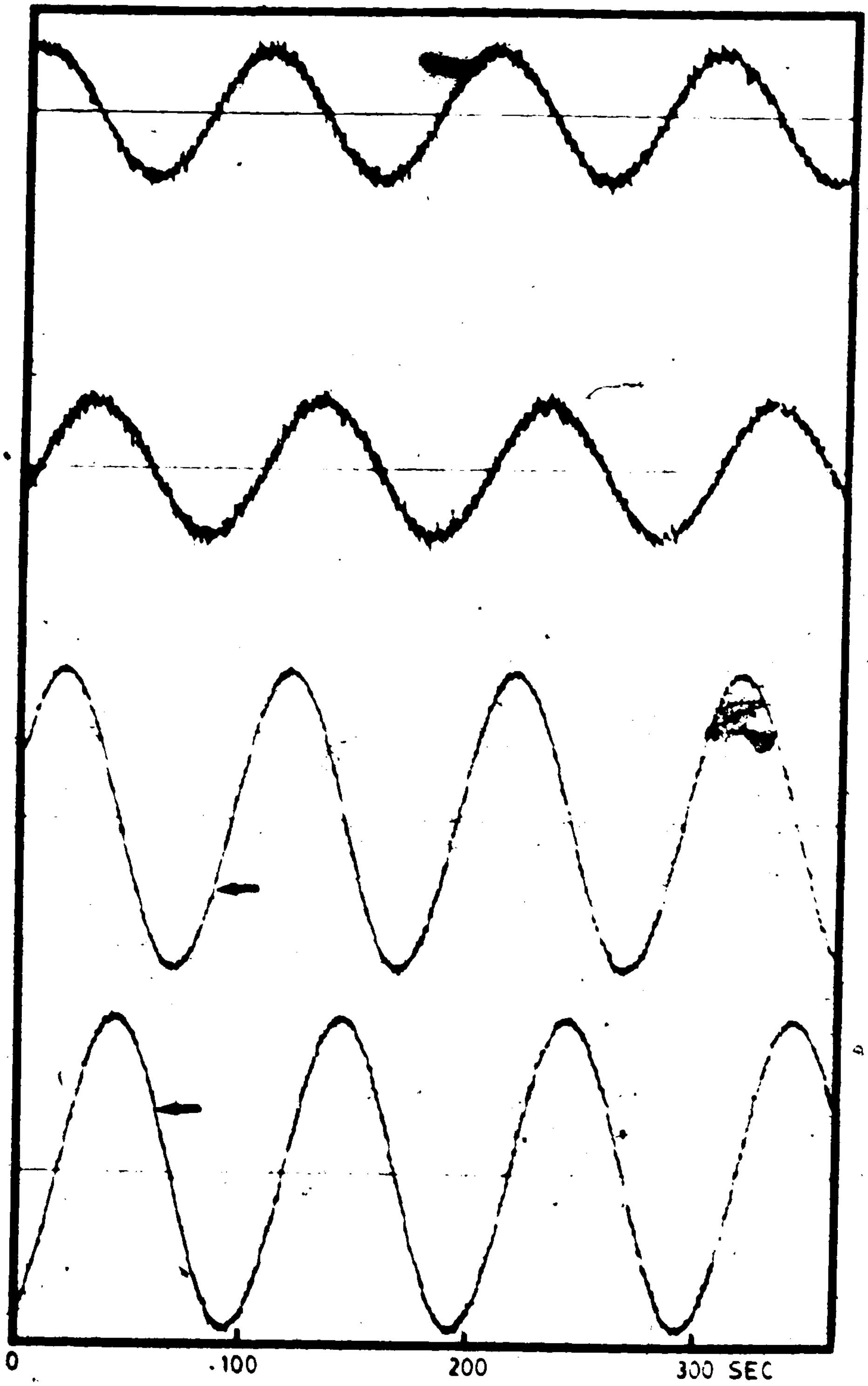


Figure 6.1.4 Response of real and quadrature channels to a simulated point source of 100 second fringe rate with derotation disabled. The real and quadrature channel outputs are sampled every 0.1 second. The top and bottom curves correspond to 2% and 20% correlation respectively. The discontinuities (arrows) in the 20% correlation curves are caused by a half second readout period of which there is no integration.

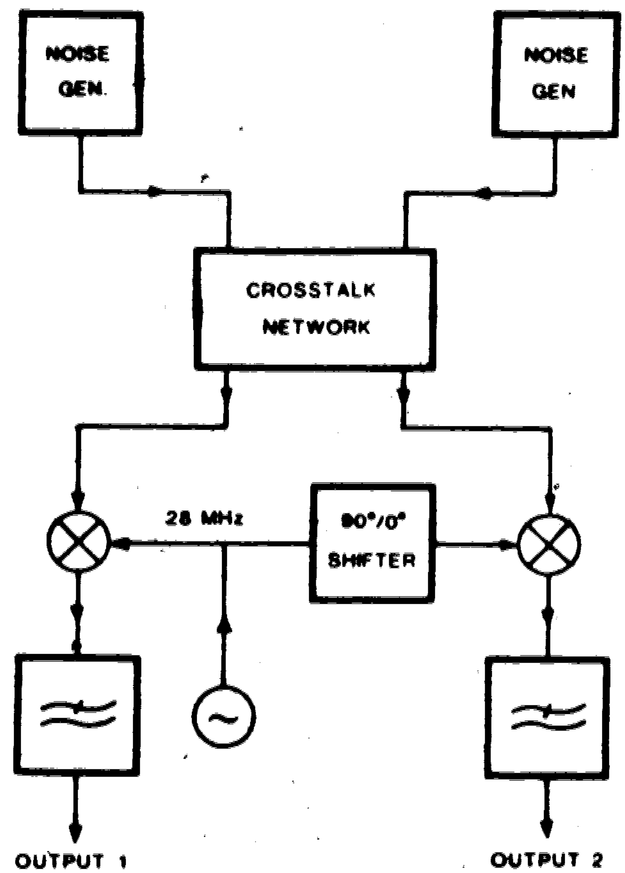


Figure 6.1.5 Correlated noise source with optional 90° phase shift.

were applied to the correlation function to produce the real and the quadrature channel outputs. The two outputs were sampled after every minor integration, or 0.1 second, and the results were plotted in figure 6.1.4.

6.1.2.2 Gain and Orthogonality of The Quadrature Channel vs Delay Test

The quadrature channel gain is less then the real channel gain and is more delay dependent[19]. This test is aimed at testing the dependence of the gain and orthogonality of the quadrature channel on delay. Figure 6.1.5 shows the signal source configuration. The correlated noise sources and SSB down-mixer connections are the same as in figure 6.1.3 but the L.O. signals are different. The two L.O. signals are derived from a common source with a phase shifter inserted into one of the L.O. signal paths. The phase shifter can insert either 90° or 0° phase shift into one of the L.O. signals, allowing the exchange of the real and quadrature channels at the output, except for a sign reversal.

Different lengths of cable up to 0.75 delay units[1] were inserted between the crosstalk network and the quantiser input in one of the signal paths. The interpolation algorithm was used to cancel out the inserted cable delay. Departure of gain and orthogonality from unity and 90° were plotted against delay. The phase departure from 90° includes the error of the 90° phase shifter which is within ±0.5°. A constant gain correction factor has been applied to the quadrature channel to bring the gain close to unity. The results plotted in figure 6.1.6 show a maximum gain error of 1% and a maximum phase error of 1°.

6.1.3 Testing Fringe Derotation

Two tests were performed to test the fringe derotation mechanism. The first test used a simulated point source similar to figure 6.1.3. The second test used the front end of the 1420 MHz system to receive signal from a point source in the sky.

6.1.3.1 The Simulated Point Source Derotation Test

The signal source used in this test was similar to the configuration in figure 6.1.3 except that an additional loop of cable was inserted into one of the signal paths between the crosstalk network and the SSB down-mixers to test the derotation at different values of delay. The fringe rate used was 100 second or 10 mHz. The digital signal processor was given a source coordinate with an equivalent fringe rate and path compensation delay to cancel out the inserted cable delay. Results of the derotated fringes were plotted in figure 6.1.7, in which (a), (b), (c) and (d) show the derotated visibility function plot of a simulated point source with different delay values. The aim of this test was to detect ripples of 50 second and 100 second period which are produced during fringe derotation. The vertical scale of figure 6.1.7 starts from zero. The rms value of the ripple is below 1% of the average value. The general slope shown in figure 6.1.7 (b) and (d) was due to a slight

---

[1]one delay unit = 1/16 microsecond

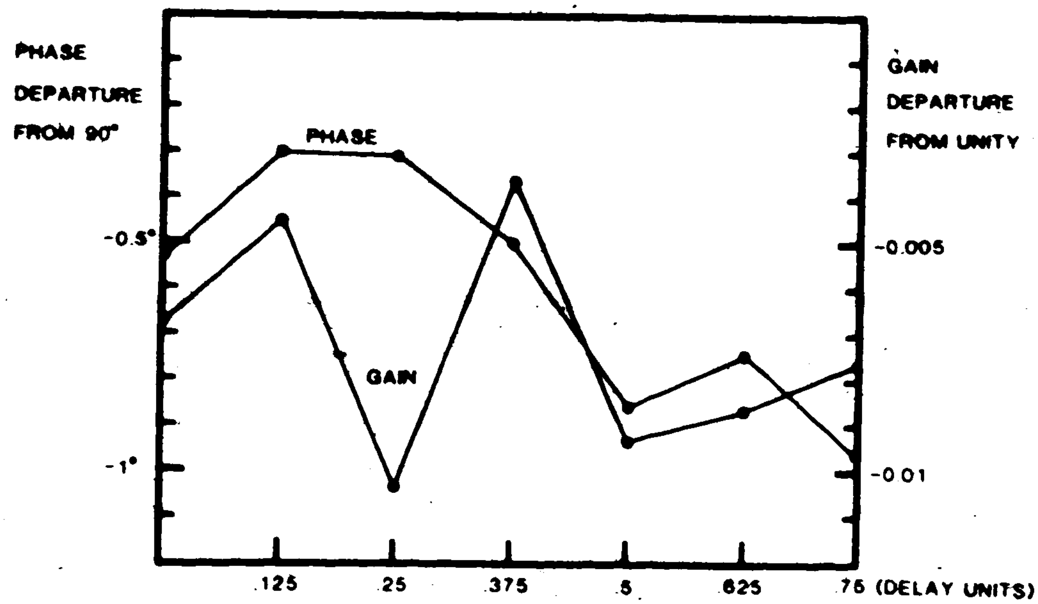
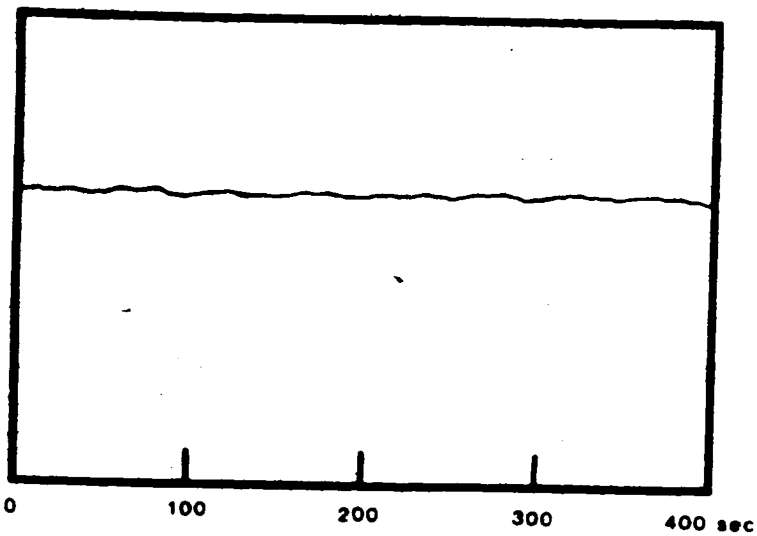


Figure 6.1.6 Gain and orthogonality of the quadrature channel vs delay

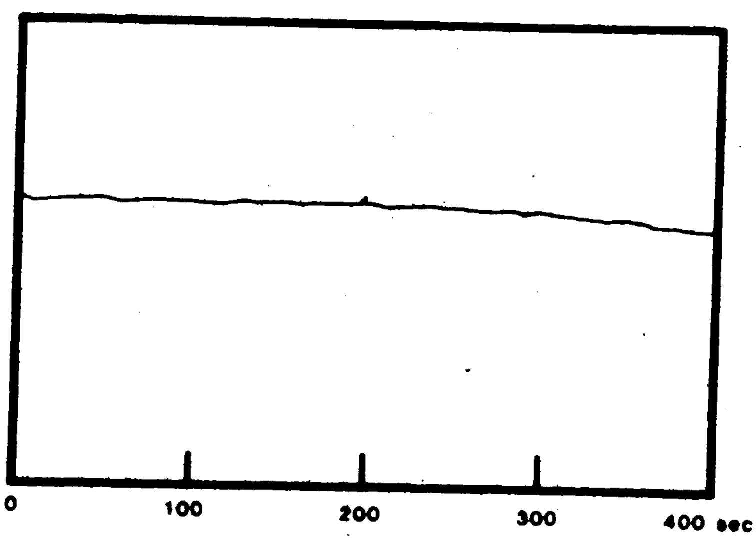
difference between the actual fringe rate and the derotation rate. This error arose because the source was a simulated point source with fixed 100 second fringe rate while the derotation was done assuming a real source whose fringe rate was a function of time.

6.1.3.2 Derotation of a Point Source In The Sky

This is a simple test of the whole digital signal processor. Signals from the 1420 MHz system were down mixed to baseband and fed into the DSP. The signals were band limited to 4 MHz with lowpass filters. The fringe derotation and path delay compensation mechanisms were disabled in the 1420 MHz system. The 1420 MHz analog system was used as four coherent superheterodyne receivers, leaving the fringe derotation and path delay compensation to the DSP.

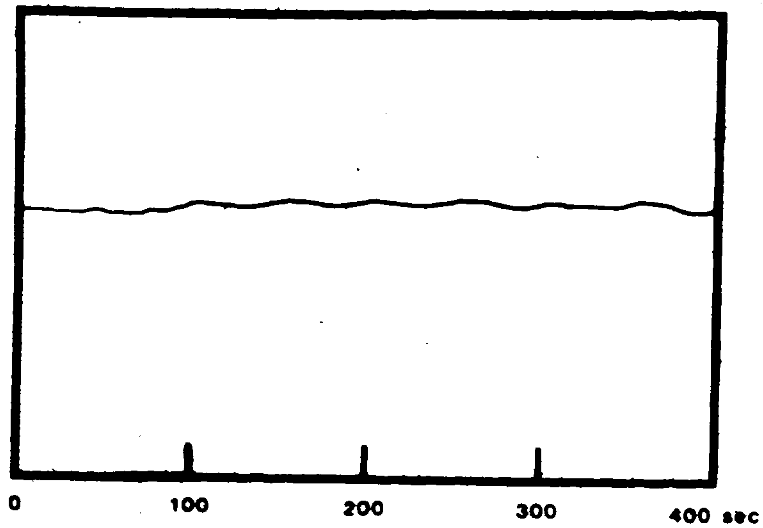


(a)

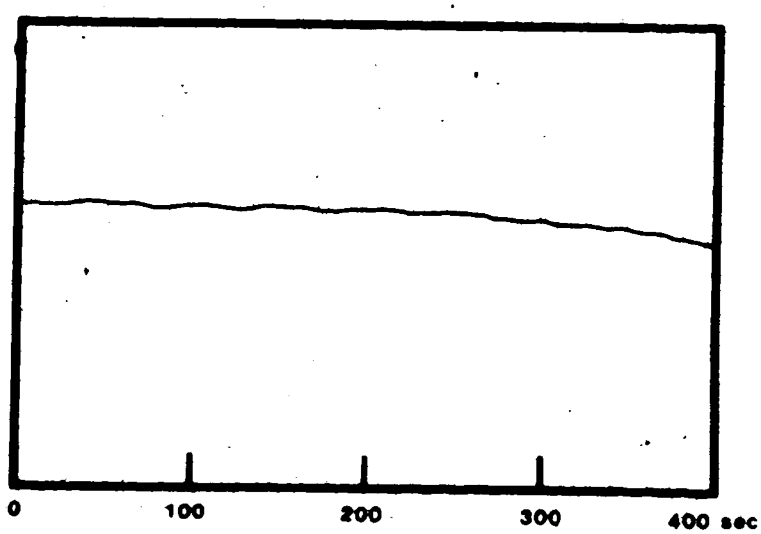


(b)

Figure 6.1.7 Derotation of a simulated point source. The fringe period is 100 second. Delay used in figures (a), (b), (c) and (d) is 0, 0.25, 0.5 and 0.75 delay units respectively.



(c)



(d)

Figure 6.1.7 Continued

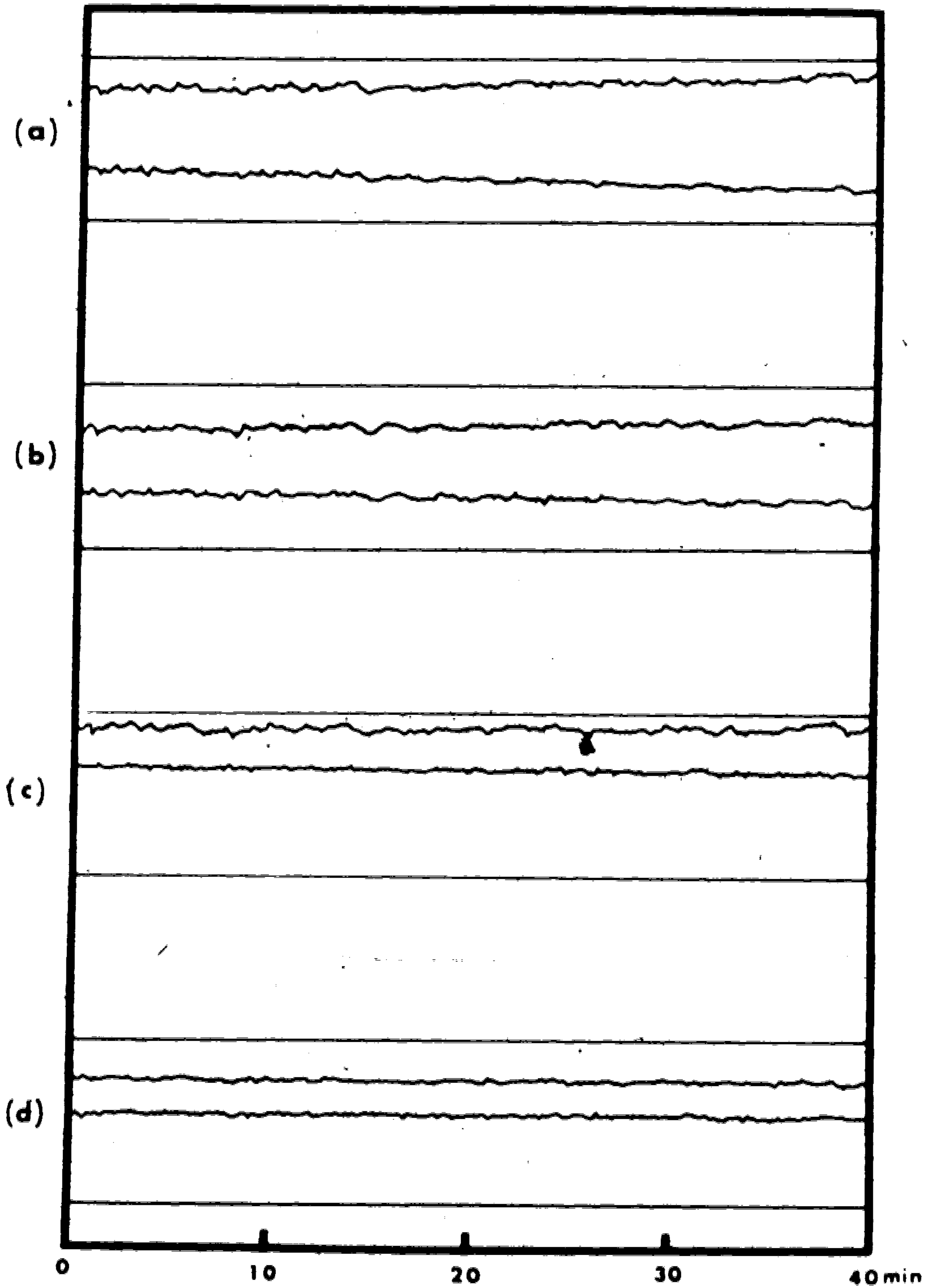


Figure 6.1.8 Visibility plot of 3C405 observed near hour angle of 4 hr 20 min, when it appears as a point source. If derotation had not been performed, there would have been between 15 and 115 turns of phase during the period of observation, depending on the spacing. From top down, the four pairs of visibility plots correspond to interferometers of spacing 90, 126, 54, and 18 units, where one unit = 30/7 meters. The upper curve in each pair is the quadrature channel output while the lower one is the real channel output.

3C405 was chosen as a strong point source when observed near an hour angle of 12 hr 20 min or 0 hr 20 min. Four interferometers with baseline lengths of 18, 54, 90 and 126 baseline units[4] were used simultaneously in the observation. The interferometer response, or visibility functions, were plotted in figure 6.1.8. The results show successful derotation of the point source at all four spacings.

6.2 Testing of The Integrated Digital Signal Processor

This section describes the testing of the DSP after the system has been integrated with the whole 1420 MHz synthesis telescope. The test was to observe a point source for 12 hours and hope to obtain a straight line interferometer response.

The 1420 MHz analog front-end was used in the same way as in the sky point source derotation test in section 6.1.3.2. The visibility function, or interferometer response, of the 12 hour observation of 3C295 is shown in figure 6.2.1. Passing the 12 hour point source test required many subsystems of the digital signal processor to be operating properly. Besides the functions tested in the previous sections, functions tested in the 12 hour point source observation included communications with the environment and track error corrections. Any sidereal clock error or any internal timing error were successfully corrected since these would have appeared as y-direction track errors or a shift of the right ascension of the source.

Communication with the host computer was fully tested in the 12 hour observation. The DSP could be started, initialised and stopped properly. The results could also be passed to the host computer correctly in an asynchronous mode, without missing any data or generating any spurious interrupts over a 12 hour period. The visibility functions shown in figure 6.2.1 are very close to straight lines, except for the receiver noise and atmospheric effects. The lack of general slope or curvature implies most of the x-and-y direction track errors and sidereal timing errors have been successfully corrected. The collimination errors and z-direction track errors are difficult to distinguish, and have not been corrected yet, since the

---

[4] one base line spacing unit = 30/7 meters

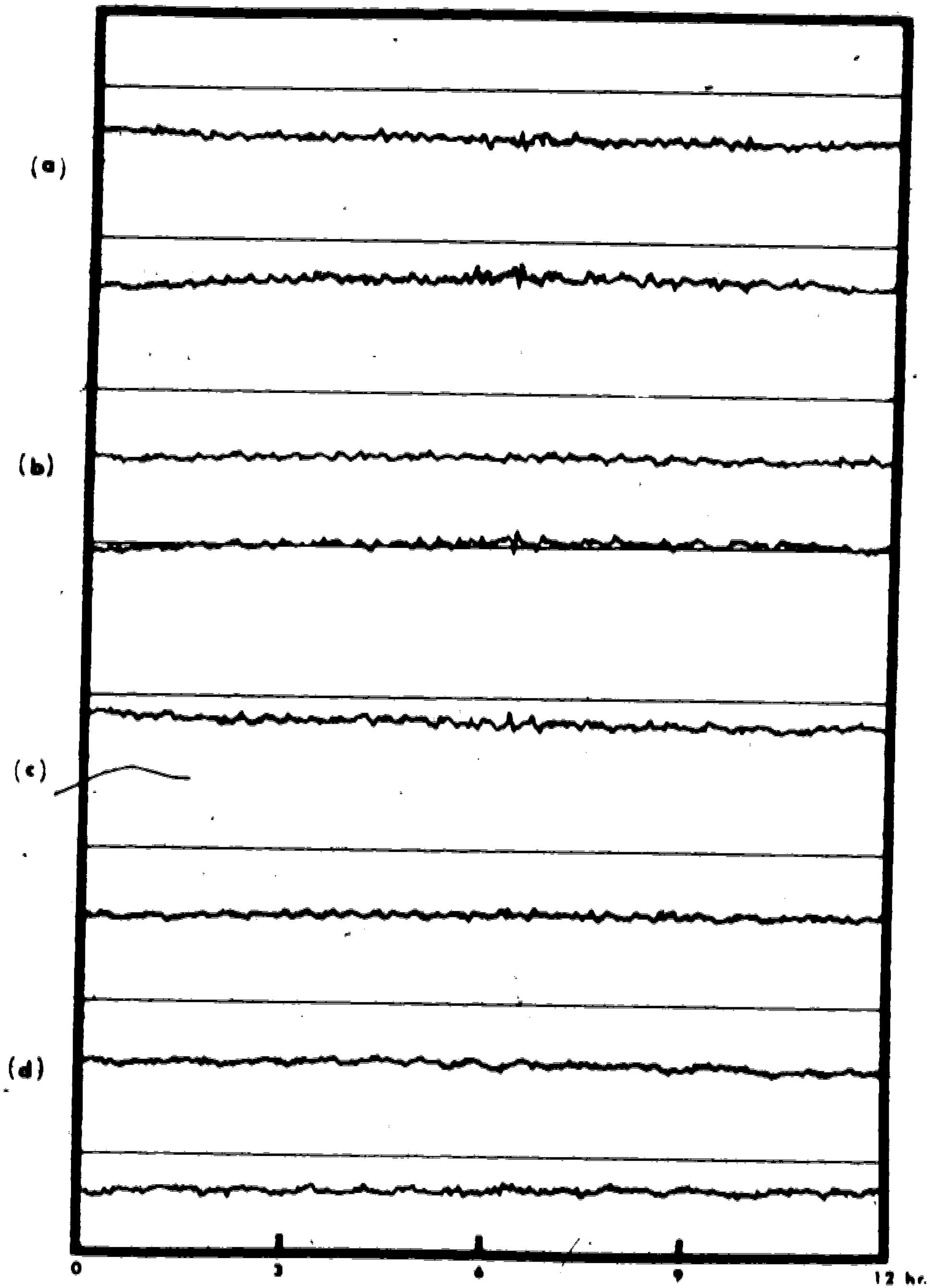


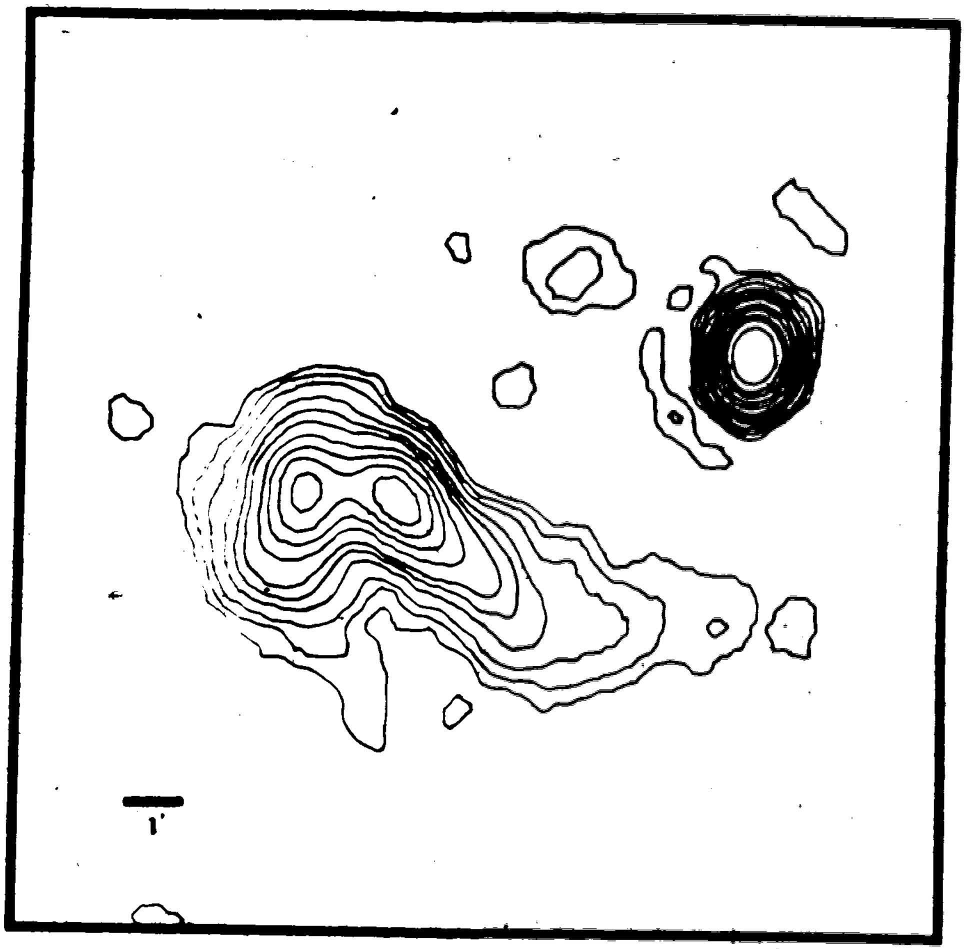
Figure 6.2.1 Visibility plot of the point source 3C295 for a 12 hour observation. The baseline lengths for (a), (b), (c) and (d) are 114, 72, 72 and 30 baseline units, where one unit is 30/7 meters.

new 408 MHz analog system will have a set of totally different collimination errors.

6.3 Observation of 3C66

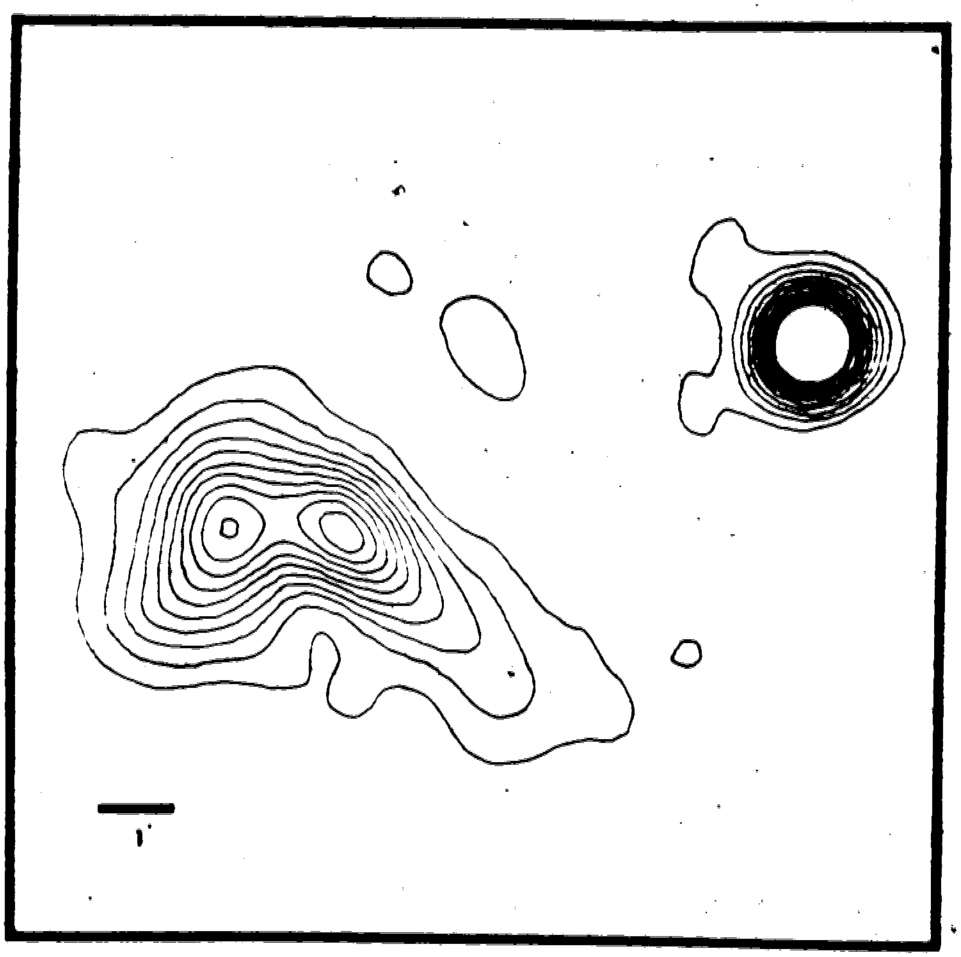
The final test of the DSP was to observe a source of known structure and produce a map from the visibility functions. Figure 6.3.1 shows two maps of 3C66. Map (a) was made with the 1420 MHz continuum system and map (b) was made with the 1420 MHz front end and 408 MHz DSP. The two maps were each made with eight baseline spacings or two 12 hour observations. The antenna spacings used in the observations were identical. The two maps were processed slightly differently during map production and cleaning, resulting in slightly different scales. However, the two maps are very similar, with only minor differences.

Figure 6.3.2 and 6.3.3 show the visibility function plots of 3C66 made with the 1420 MHz continuum system and the DSP respectively. The signal-to noise (S/N) ratio of figure 6.3.2 is about twice that of figure 6.3.3. The difference in S/N ratio is expected since the 1420 MHz continuum system has a bandwidth about four times that of the DSP. The two visibility plots also look very similar except for the collimination errors which are eliminated during the map production.



(a)

Figure 6.3.1 Maps of 3C66. Figure (a) is made from observations with the 1420 MHz continuum system. Figure (b) is made from observations with the 1420 MHz analog front-end and the 408 MHz DSP. The two observations were made with identical antenna spacings but processed differently during map making, resulting in two maps of slightly different scale. Contour levels are 60, 180, 300, 420, 540, 660, 780, 900, 1020, and 1140.



(b)

Figure 6.3.1 Continued

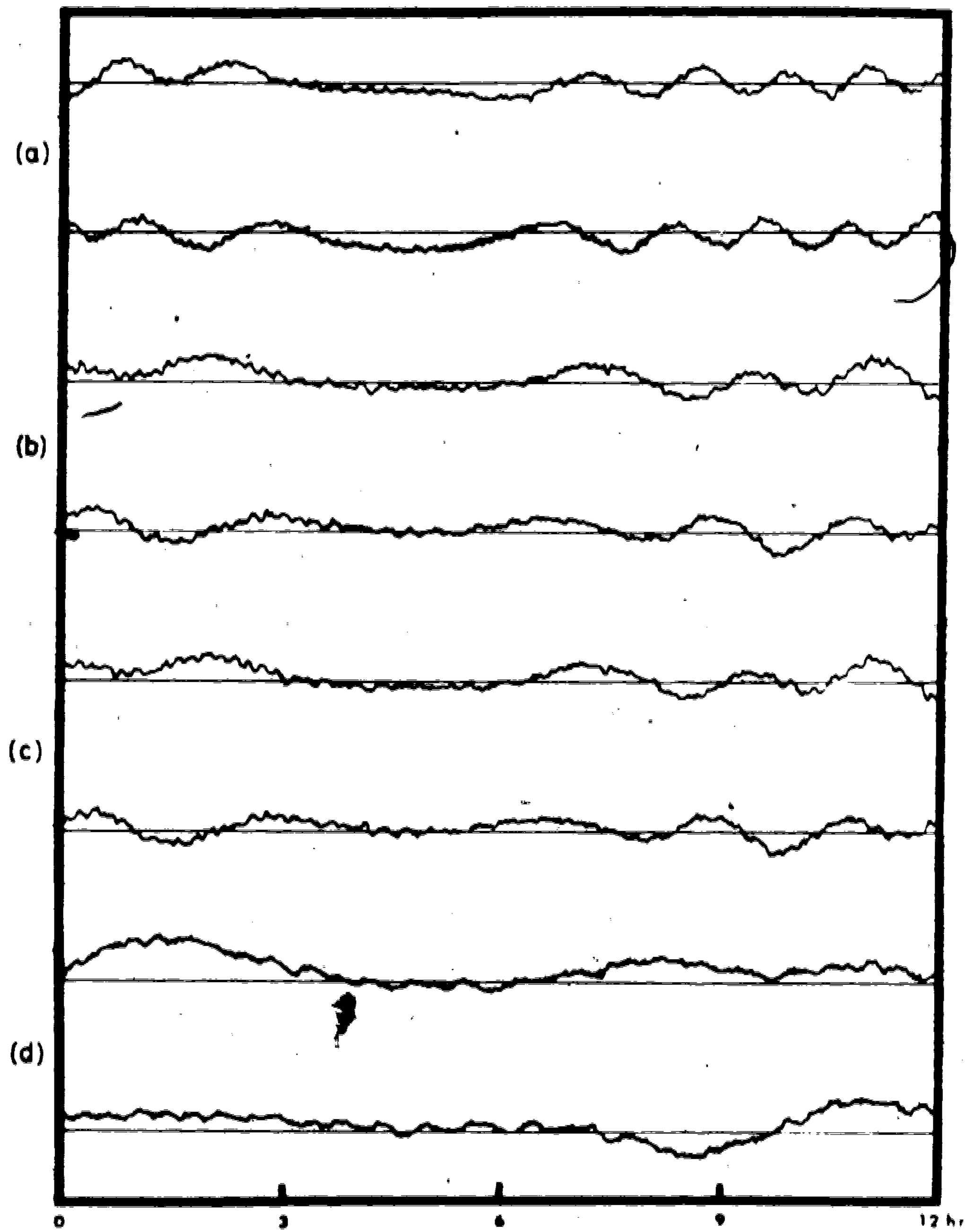


Figure 6.3.2 Visibility plots of 3C66 observed with the 1420 MHz continuum system. The interferometer baselines are 108, 72, 72, 36, 126, 90, 54 and 18 units, for curves (a) to (h), where 1 unit=30/7 meter.

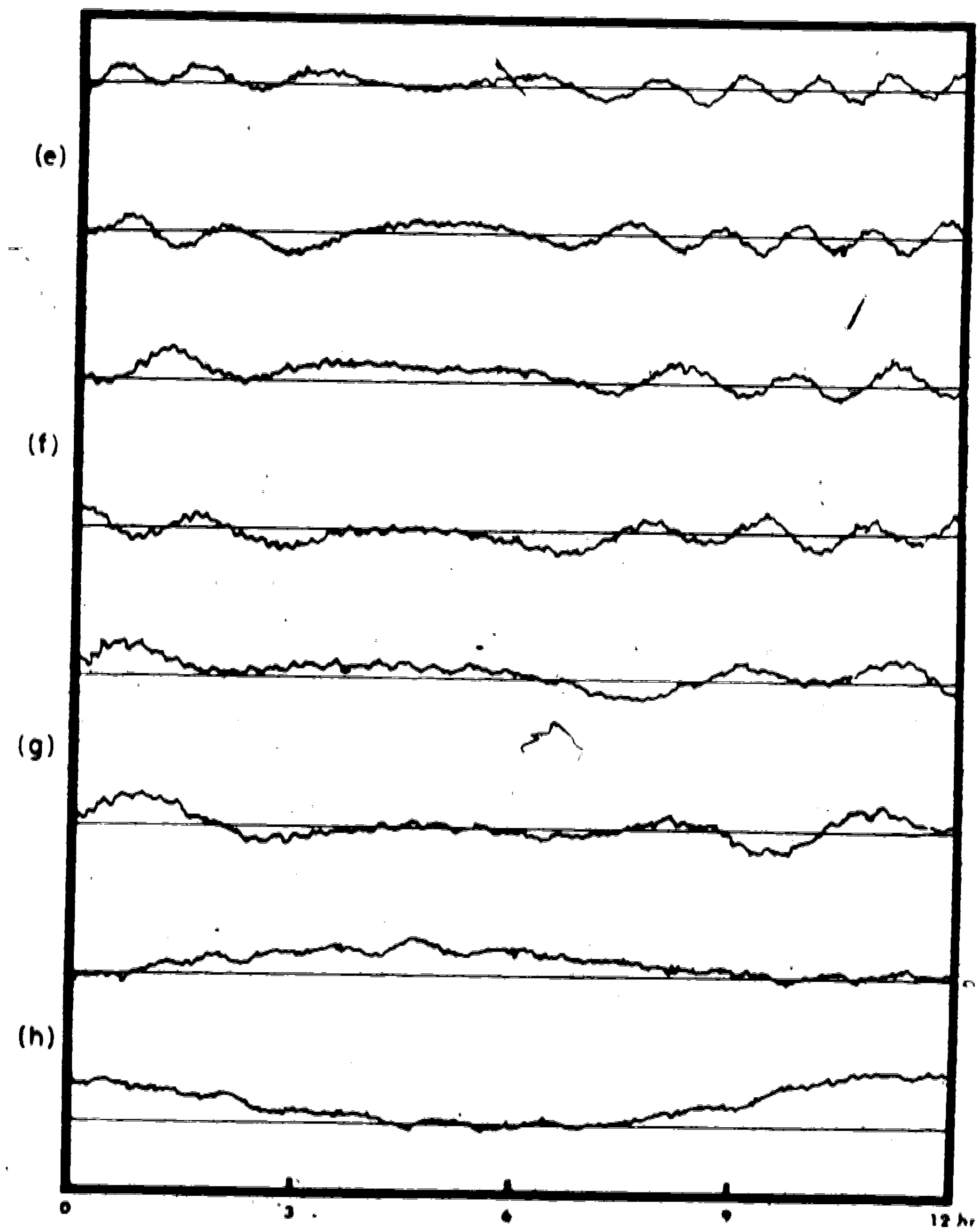


Figure 6.3.2 Continued

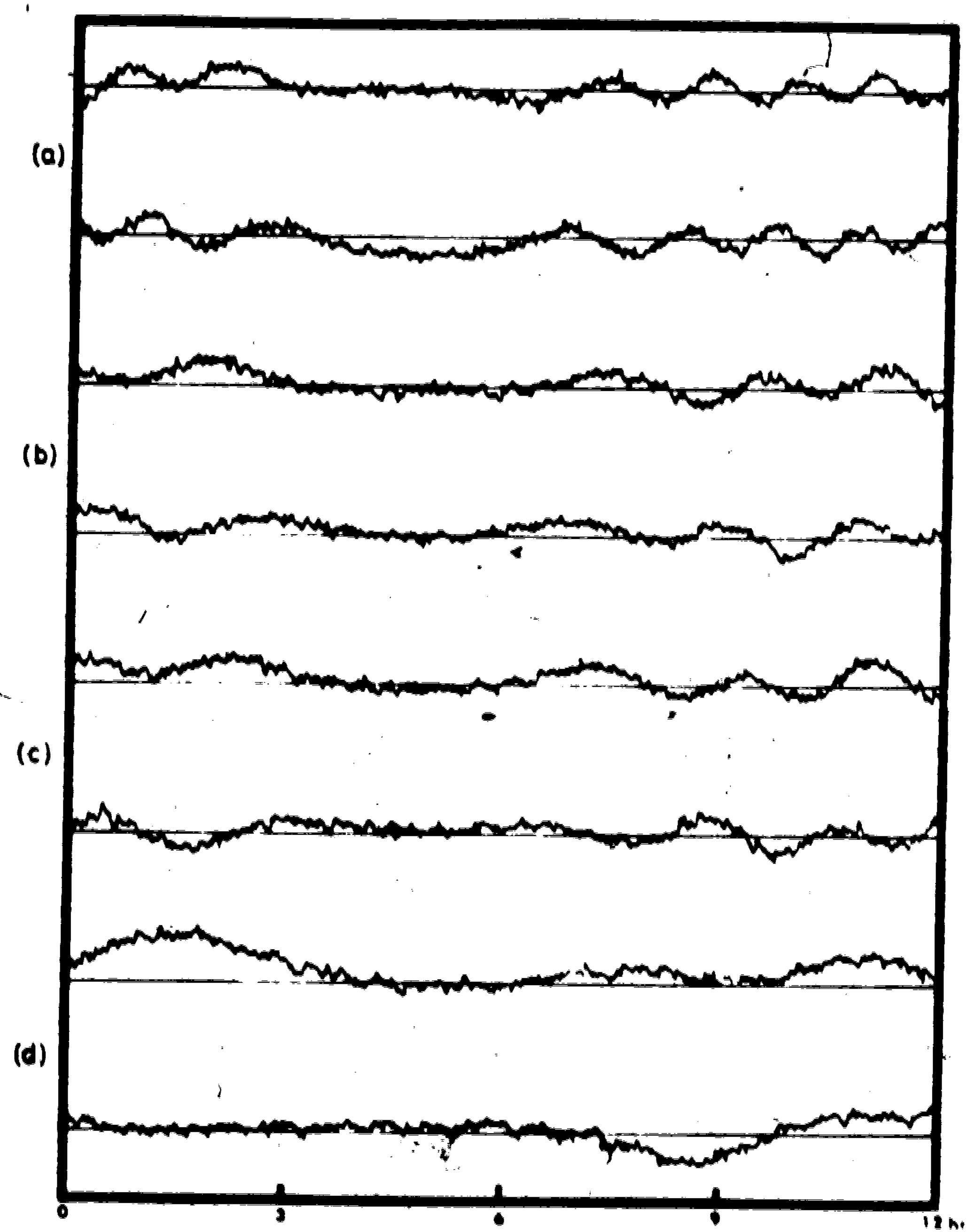


Figure 6.3.3 Visibility plots of 3C68 observed with the 1420 MHz analog front-end and the DSP. Interferometer baselines are the same as those in figure 6.3.2, i.e. 108, 72, 72, 36, 126, 90, 54 and 18 for (a) to (h). The burst of noise between 6 and 9 hr in figure (h) is probably due to the Sun.

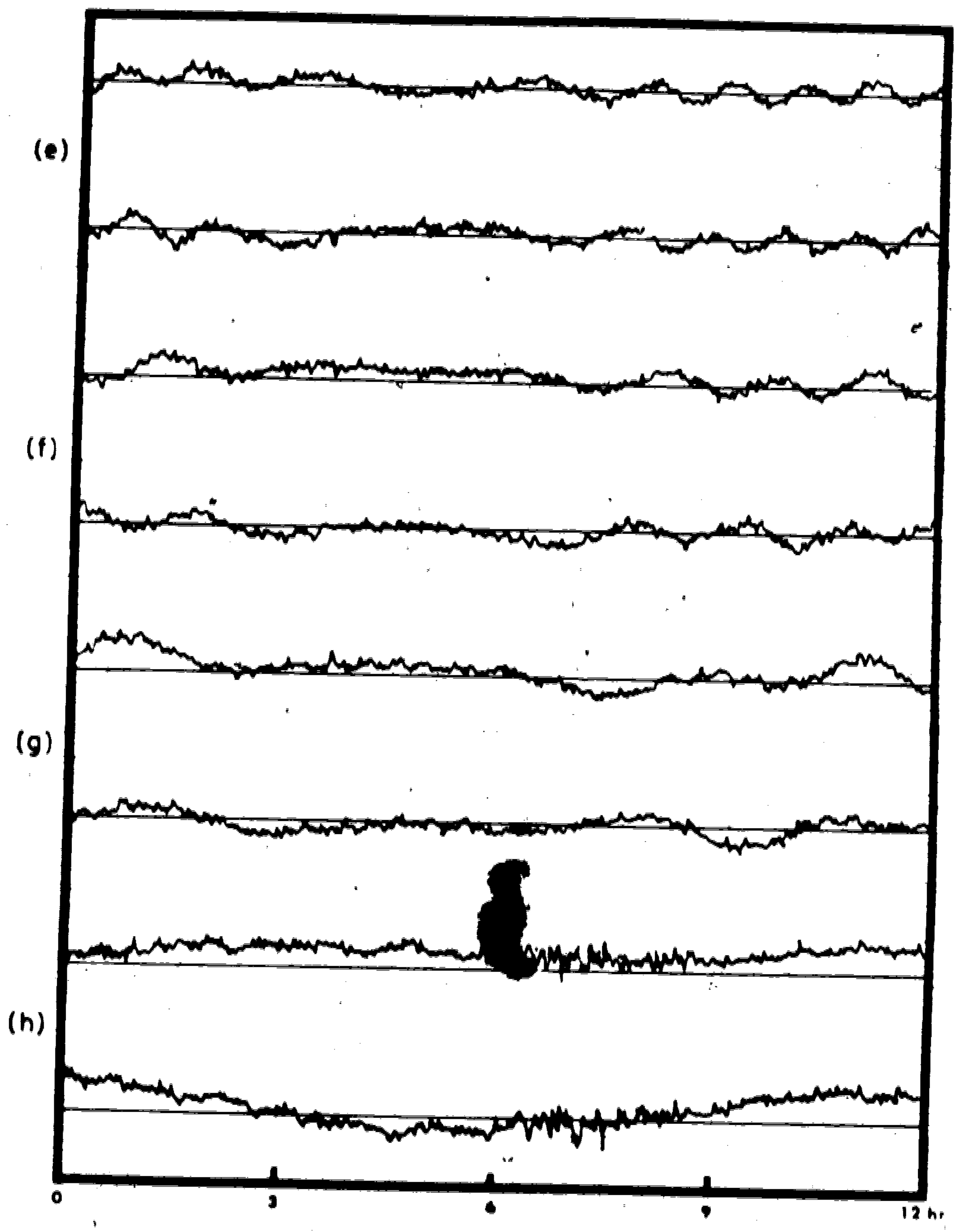


Figure 6.3.3 Continued.

7. SUMMARY AND CONCLUSIONS

7.1 Summary

The aim of this project was to design and implement a digital signal processor (DSP) for a 408 MHz continuum supersynthesis telescope. The processor accepts a 4 MHz baseband signal, performs the functions of path delay compensation, quadrature channel generation and fringe derotation to produce outputs equivalent to the real and quadrature channel output of a conventional interferometer.

The incoming signals are digitised and crosscorrelated in a 16-channel digital correlator. Path delay compensation is performed in two steps. A coarse digital delay unit shifts the phase centre close to the centre of the correlator, then interpolation is used to obtain exact path delay compensation. The quadrature channel is generated numerically by Hilbert transforming the correlation function. Finally, fringe derotation is applied to the real and quadrature correlation functions. For good orthogonality, the correlator is sampled every 100 ms. Interpolation, numerical Hilbert transformation, and fringe derotation by synthesis of the single sideband down-mixing equations are performed for each interferometer every 100 ms.

The whole project was based on applying contemporary digital technology to replace analog subsystems. The digital signal processor replaced the switched cable delay, the intricate local oscillator differential phase controlling system and the quadrature channel generation circuitry. Replacing the analog equipment with digital circuitry reduced the chance of crosstalk between the received signals and enhanced the phase stability. The digital delay scheme totally elimated the delay equalisation error, dispersion and frequency dependent attenuation problems of the cable delay system.

The strategy of design and implementation was to choose a powerful microcomputer and implement functions in software whenever possible. The result was a software-intensive but powerful and flexible instrument. The software synchronisation and timing problems were solved by employing a realtime multi-tasking executive, the Tiny Operating System (TOS), specially developed for this application. High level language C was used for software implementation, whenever possible, to ease software development. A monitor program, the Tiny Operating System Monitor (TOSMON), interfaces the system to the operator and allows the manipulation of system and observation process parameters during observation. The monitor has greatly increased the observability, testibility and controllability of the system.

Physically the DSP for ten interferometers can be fitted into a four-foot vertical space of a 24-inch rack, whereas the analog counterpart would take up a small room. Besides being less expensive and easier to build, the digital system has the additional advantage of being very easy to replicate and expand since a lot of the functions are performed in software and digital circuitry requires no tuning.

7.2 System Performance

Since the DSP was developed before the 408 MHz analog front-end, signals from the 1420 MHz system were used for testing and observation. A 12 hour observation of a point source shows successful operation of delay equalisation, quadrature channel generation, fringe derotation and track error correction. The final integrated DSP was tested with an automatic observation (under the control of the host computer) of 3C66. The map produced shows very good agreement with another map of the same source made with the 1420 MHz continuum system. The visibility plot made with the DSP shows about half the S/N ratio of those made with the 1420 MHz continuum system. The difference in S/N ratio was expected since the 1420 MHz continuum system has a reception bandwidth of 15 MHz compared to the 4 MHz bandwidth of the DSP.

Due to the finite correlator length, imperfections exist in the quadrature correlation function generated numerically. The quadrature channel gain depends slightly on the position of the the phase centre along the correlation function and also on the source structure. Unbalanced gain between the real and the quadrature channels gives rise to ripples of twice fringe rate at the output after derotation. With the 16 channel correlator used, such ripples were kept below 2% for most cases. Spurious correlation gives rise to ripples at the output at the fringe rate after derotation. Derotation after correlation makes spurious correlation slightly more difficult to correct with phase switching. The rate of phase switching should preferably be much higher than the fringe rate. However, these two kinds of ripples are at much higher frequency than the u-v plane sampling rate except near hour angles of 6 and 18 hour when the fringe rate approach zero as given by the derivative of (4.4.3). These ripples can be filtered out easily by postprocessing the visibility functions except near the tangential points.

7.3 Detailed documentation

The DSP is implemented as part of the supersynthesis telescope in the Dominion Radio Astrophysical Observatory (DRAO) in Penticton B.C., Canada. Detailed documentation in the form of hardware schematics and software listing for the DSP are on file at DRAO and can be made avaliable.

7.4 Recommendation For Further Studies And Possible Applications

Unlike most research projects, the design and implementation of the DSP is not an open-ended project. Although far from being perfect, the DSP is successful as far as meeting the design objectives is concerned. However, there is room for improvement in the quadrature channel generation. More simulation studies could be made with different source configurations and interpolation functions to find a better combination of interpolation function and correlator configuration.

The techniques used in the DSP could possibly be extended to correlation spectroscopy, where the use of a digital correlator is natural. But the signal processing overhead will be much increased when it is necessary to produce a multi-channel correlation function output. However, in generating the inphase correlation function only, the interpolation, or fractional part of path compensation delay, could be replaced by a phase shift in the quantiser sampling clock. In correlation spectroscopy, the quadrature channel output is not required except for fringe derotation after correlation. For an N-channel crosscorrelation function, two N-point FFTs and N syntheses of the SSB down-mixing equations are required for every correlation function sample to replace the fringe derotation mechanism in the local oscillator system. Therefore numerical derotation may not be economical for large N.

8. REFERENCES

[1] Fomalont, E.B. "Earth-Rotation Aperture Synthesis", Proc. IEEE vol 61, pp 1211-1218, Sept 1973.

[2] Fomalont, E.B. and Wright, M.C.H. "Galactic and Extra-Galactic Radio Astronomy", Verschuur, G.L. and Kellermann, K.I. eds, Springer-Verlag Inc. New York 1974.

[3] Roger, R.S., Costain, C.H., Lacey, J.D., Landecker, T.L., and Bowers, F.K. "A Supersynthesis Radio Telescope for Neutral Hydrogen Spectroscopy at The Dominion Radio Astrophysical Observatory", Proc IEEE vol 61, pp 1270-1276, 1973.

[4] Van Vleck, J.H and Middleton, D. "The Spectrum of Clipped Noise", Proc IEEE, vol 54, pp 2-19, Jan 1966.

[5] Weinreb, S. "A Digital Spectral Analysis Technique and Its Application To Radio Astronomy", M.I.T. Tech. Rep. 412, 1963.

[6] Bowers, G.K and Klingler, R.J. "Quantisation of Correlation Spectrometers", Astron. Astrophys. suppl. vol 15, pp 373-380, 1974.

[7] Dewdney, P.E. "Product Transition Correlators", Rev. Sci. Instrum. vol 51, pp 17-21, Nov. 1980.

[8] Cooper, B.F.C. "Correlators With Two-Bit Quantisation", Aust. J. Phys. vol 23, pp 521-527, 1970.

[9] Bracewell, R. "The Fourier Transform And Its Applications", McGraw-Hill, New York, 1965.

[10] Dewdney, P.E. "One Dimensional Formulation Of Signal Processing Required For Spectroscopic Synthesis", Dominion Radio Astrophysical Observatory Technical Report, 1980.

[11] Landecker, T.L. "A Proposal For A Low Frequency Continuum Channel For The Synthesis Telescope", Dominion Radio Astrophysical Observatory Technical Report, 1980.

[12] Dewdney, P.E., private communication, 1980.

[13] Sieber, W., Salter, C.J., and Mayer, C.J., "Spectral and Polarization Characteristics of the Supernova Remnant CTA1", Astron. Astrophys., vol. 103, pp 393-404, 1981.

[14] Veidt, B.G. private communication, 1982.

[15] Kraus, J.D., "Radio Astronomy", McGraw-Hill, New York, 1966.

[16] Lister, A.M., "Fundamentals of Operating Systems", Macmillan London, 1975.

[17] Elsmore, B., Kenderdine, S., and Sir Martin Ryle, "The Operation of the Cambridge One-mile Diameter Radio Telescope", Monthly Notices of the Royal Astronomical Society, vol 134, pp 89-95, 1966.

[18] Papoulis, A., "Signal Analysis", McGraw-Hill, New York, 1977.

[19] ibid p 39.

[20] ibid pp 44-48.

9. APPENDIX I : DERIVATION OF REAL AND QUADRATURE CONVOLUTION FUNCTION

9.1 Interpolation Functions Based on Rectangular Frequency Domain Window

The rectangular window in the frequency domain is:

$$W_1(f) = \mathrm{rect}(f)$$

where

$$\mathrm{rect}(f) = \begin{cases} 1, & -0.5 \leqslant f \leqslant 0.5 \\ 0, & \text{elsewhere} \end{cases} \tag{A1.1}$$

The real channel interpolation function is:

$$\begin{aligned} w_1(t) &= \mathfrak{F}^{-1}\left\{W_1(f)\right\} \\ &= \mathrm{sinc}(t). \end{aligned} \tag{A1.2}$$

The quadrature channel interpolation function, or modified Hilbert transform kernel, is the Hilbert transform of the real channel interpolation function.

$$\begin{aligned} h_1(t) &= H\left\{w_1(t)\right\} \\ &= \mathfrak{F}^{-1}\left\{W_1(f)\, H(f)\right\} \end{aligned} \tag{A1.3}$$

where $H(f)$ is the Hilbert transform frequency domain function and

$$H(f) = \begin{cases} +j, & f > 0 \\ -j, & f \leqslant 0 \end{cases}$$

Therefore the time domain quadrature channel interpolation function $h_1(t)$ becomes.

$$
\begin{aligned}
h_1(t) &= \int_{-\infty}^{\infty} W_1(f)\, H(f)\, \exp(j2\pi ft)\, df \\
&= \int_{-.5}^{0} -j\, \exp(j2\pi ft)\, df + \int_{0}^{0.5} j\, \exp(j2\pi ft)\, df \\
&= \frac{\cos(\pi t) - 1}{\pi t} \\
&= \operatorname{cosc}(t)
\end{aligned} \tag{A1.4}
$$

where cosc(t) is defined as $(\cos(\pi t) - 1)/(\pi t)$

9.2 Interpolation Functions Based on 50% Raised Cosine Fuction

The 50% raised cosine function $W_2(f)$ can be defined as the sum of three functions

$$W_2(f) = \tfrac{1}{2}[\, G_1(f) + G_2(f) + G_3(f)\,] \tag{A1.5}$$

$$\text{where } G_1(f) = \begin{cases} -\cos(2\pi f), & -1 \leq f \leq -0.5 \text{ or } 0.5 \leq f \leq 1 \\ 0, & \text{elsewhere} \end{cases} \tag{A1.6}$$

$$G_2(f) = \operatorname{rect}(f/2) \tag{A1.7}$$

$$G_3(f) = \operatorname{rect}(f) \tag{A1.8}$$

The function $G_1(f)$ can be expressed as:

$$G_1(f) = [\, \operatorname{rect}(f) - \operatorname{rect}(f/2)\,] \cos(2\pi f) \tag{A1.9}$$

$G_1(f)$ in the time domain is:

$$
\begin{aligned}
g_1(t) &= \mathfrak{F}^{-1}\left\{ [\, \operatorname{rect}(f) - \operatorname{rect}(f/2)\,] \cos(2\pi f) \right\} \\
&= [\, \operatorname{sinc}(t) - 2\, \operatorname{sinc}(2t)\,] * \tfrac{1}{2}[\, \delta(t+1) + \delta(t-1)\,]
\end{aligned}
$$

$$= \mathrm{sinc}(t+1) + \mathrm{sinc}(t-1) - \mathrm{sinc}[2(t+1)] - \mathrm{sinc}[2(t-1)] \qquad \text{(A1.10)}$$

$G_2(f)$ and $G_3(f)$ in the time domain are:

$$g_2(t) = 2\,\mathrm{sinc}(2t) \qquad \text{(A1.11)}$$

$$g_3(t) = \mathrm{sinc}(t) \qquad \text{(A1.12)}$$

The inverse Fourier transform of the 50% raised cosine fufction is:

$$\begin{aligned} g(t) &= \tfrac{1}{2}[g_1(t) + g_2(t) + g_3(t)] \\ &= \tfrac{1}{2}[\mathrm{sinc}(t) + 2\,\mathrm{sinc}(2t) - 2\,\mathrm{sinc}(2(t+1)) \\ &\qquad -2\,\mathrm{sinc}(2(t-1)) + \mathrm{sinc}\,(t+1) + \mathrm{sinc}\,(t-1)] \end{aligned} \qquad \text{(A1.13)}$$

The quadrature channel interpolation function $h_2(t)$ is the Hilbert transform of $w_2(t)$. Therefore:

$$h_2(t) = H\left\{w_2(t)\right\} \qquad \text{(A1.14)}$$

Since the Hilbert transform of the sinc function is the cosc function, $h_2(t)$ is given by replacing all the sinc functions with cosc function on the RHS of (A1.13).

$$\begin{aligned} h_2(t) &= \tfrac{1}{2}[\,\mathrm{cosc}(t) + 2\,\mathrm{cosc}(2t) - 2\,\mathrm{cosc}(2(t+1)) - 2\,\mathrm{cosc}(2(t-1)) \\ &\qquad + \mathrm{cosc}(t+1) + \mathrm{cosc}(t-1)\,] \end{aligned} \qquad \text{(A1.15)}$$

10. Appendix II: Maximum Fringe Phase Error Introduced By Linear Extrapolation

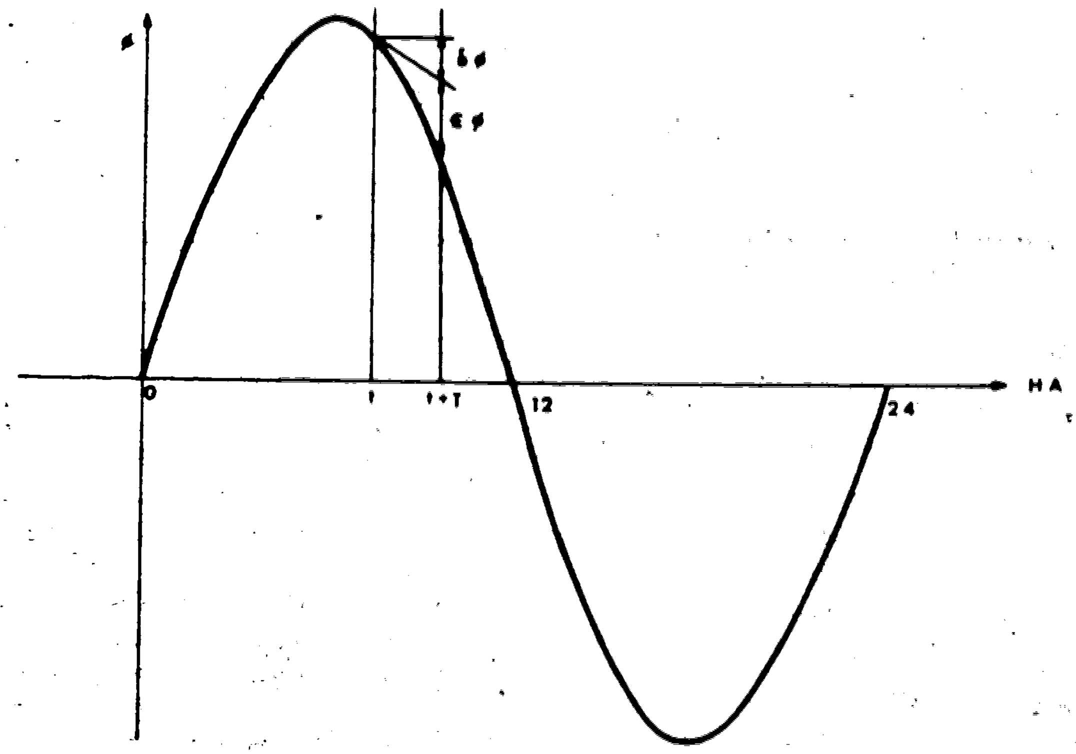


Figure A2.1 Linear extrapolation error for sinusoidal functions

The fringe phase ϕ is given by (5.4.3) as:

$$\phi(t) = \frac{2\pi B}{\lambda_o} \cos(DEC) \sin(HA)$$

$$= K \sin(HA) \qquad (A2.1)$$

where $K = (2\pi B/\lambda_o) \cos(DEC)$. The fringe phase is a sinusoidal function of hour angle as plotted in figure A2.1. Within a major integration period, $\phi(t)$ is calculated with linear extrapolation by:

$$\phi(t + T) = \phi(t) + \frac{d\phi}{dt} T \qquad (A2.2)$$

From figure A2.1, the error introduced in a linear extrapolation over time T is:

$$\varepsilon\phi = \phi(t + T) - \phi(t) + \frac{d\phi}{dt} T$$

$$= K \sin(t+T) - K \sin(t) - KT \cos(t)$$

$$= K [\sin(t) \cos(T) + \sin(T) \cos(t) - \sin(t) - T \cos(t)]$$

$$= K [(\cos(T) - 1) \sin(t) + (\sin(T) - T) \cos(t)] \qquad \text{(A2.3)}$$

The maximum value of T is one major integration period which corresponds to 5.8178×10^{-4} radian. The value of the terms $\cos(T) - 1$ and $\sin(T) - T$ are -1.692×10^{-7} and -3.282×10^{-11} respectively. The value of K is at its maximum for maximum antenna spacing and low source declination. When operating at 1420 MHz under these conditions K has a maximum value of 2840. Subsituting these numerical values into equation (A2.3) gives a maximum error in ϕ of 4.81×10^{-4} radian around $\sin(HA) = 1$. The maximum fringe phase error introduced by linear extrapolation is very small compared with the specification of 1° overall fringe phase accuracy.

11. APPENDIX III: 408 MHz DIGITAL SIGNAL PROCESSOR MANUAL

11.1 Hardware Configuration

The digital signal processor hardware is made up of three modules: the controller, the correlator and the power supply unit. The controller houses a 68000 microcomputer, a memory board, the interfacing circuitry and a system clock generator. The correlator module houses the correlator boards and the programmable digital delay boards.

To set up the system, the user should connect up the system according to the interconnection table. Set the start-up switch on the 68000 board to TOSMON position, connect a terminal of the right baud rate to J18 of the controller and power up the system. The TOSMON prompt should appear on the screen. If not, see 408HARD.DOC for trouble shooting.

11.2 TOSMON Commands

TOSMON is a system monitor with special commands for controlling the observation in progress. TOSMON commands can be classified into observation commands and system commands. Observation commands are used to manipulate and monitor the observation processes. The system commands are used to manipulate and monitor the operations of TOS.

TOSMON is interactive in nature. All commands are two characters long. The third and subsequent characters are ignored. When a command is received, TOSMON prompts the user for arguments. When the user is asked to make a multiple choice selection, the option in square brackets " [ ] " is the default. Similarly, a default answer to a variable update leaves the variable unchanged. In the following subsections the small print shows examples of conversation with TOSMON. The user input is underlined for clarity.

CONNECTOR	CONNECTION	FUNCTION
POWER SUPPLY UNIT:		
J1	TO CONTROLLER J1	power supply to controller
J2	TO CORRELATOR J2	power supply to correlator
J3	TO CORRELAROR J3	power supply to correlator
CONTROLLER UNIT:		
J1	FROM POWER SUPPLY J1	power supply to controller
J2	FROM HOST INTERFACE	parallel host to micro port
J3	TO HOST INTERFACE	parallel micro to host port
J4	FROM SST WORD	parallel input port for SST word
J5	TO CORRELATOR	correlator bus lower word
J6	TO CORRELATOR	correlator bus upper word
J7	DIGITAL DELAYS	digital delay control bus (DDC BUS)
J8,J9	TO CORRELATOR	correlator Shift & Count pulses
J10,J11	TO CORRELATOR	advance & retard test signal
J12	TO QUANTISER 1	16 MHz sampling clock
J13	TO QUANTISER 2	16 MHz sampling clock
J14,J15	TO QUANTISERS	1 sec timing pulse
J16	TO CHART RECORDER	analog output channel
J17	TO 408 RF SYSTEM	phase switching control
J18	TO SYSTEM CONSOLE	RS232 port to system console VDU
J19	TO DEVELOPMENT HOST	RS232 port to development host
J22	TO SIDEREAL CLOCK	parallel input from sidereal clock

Table A3.1 Interconnection Table.

11.2.1 Observation Commands

11.2.1.1 Analog Output (AO)

The AO command selects either the real channel output, the quadrature channel output, or the phase to be connected to the analog channel output.

Subcommands:

- **RE** Connects the analog output to the real channel output
- **QU** Connects the analog output to the quadrature channel output
- **PH** Connects the analog output to the fringe angle ϕ.

```
TOSMON: AO
Which interferometer do you want the Analog Output on?9
Invalid interferometer number.  Enter again.
3
What do you want on the analog output?
REal output, QUadrature output or PHase phi? [RE]
QU
Enter n for 2**n scaling factor. (0 <= n <=6) 3
Connection Made.
TOSMON:
TOSMON: AO
Which interferometer do you want the Analog Output on?1
What do you want on the analog output?
REal output, QUadrature output or PHase phi? [RE]
PH
Connection Made.
TOSMON:
```

11.2.1.2 Continuous Display (CD)

The CD command inserts the CONTINUOUSDISPLAY process into the 8 second list of the Observation Timetable. A set of observation variables and output values is printed after every major integration. Hitting the "RETURN" key leaves the continuous display mode by deleting the CONTINUOUSDISPLAY process from the Observation Timetable.

```
TOSMON: CD

Sidereal time:          64 day  0 hour  55 min  28.2 sec.
Observation period =     0 day  0 hour   7 min  52 sec.
Hour Angle = 19H 43M 28.31S  Declination = 20D 45' 23.29''
              Spacing          Delay       Phi     Real O/P  Quad O/P
Intf 0       11.999720      -2.30938       82       -16       -9
Intf 1       20.000583      -3.84912      240        13        2
Intf 2       45.000280      -8.66054      134       -33       -7
Intf 3      112.000000     -21.55480      275        37      -19
```

```
Sidereal time:             64 day  0 hour  55 min  36.2 sec.
Observation period =       0 day   0 hour  8 min   0 sec.
Hour Angle = 19H 43M 36.41S  Declination = 20D 45' 23.29''
                 Spacing             Delay        Phi    Real O/P   Quad O/P
Intf 0          11.999720          -2.30872      103          7         -7
Intf 1          20.000583          -3.84802      275          4         10
Intf 2          45.000280          -8.65806      212        -23         30
Intf 3         112.000000         -21.54864      112        -18        -35

Sidereal time:             64 day  0 hour  55 min  44.3 sec.
Observation period =       0 day   0 hour  8 min   7 sec.
Hour Angle = 19H 43M 44.41S  Declination = 20D 45' 23.29''
                 Spacing             Delay        Phi    Real O/P   Quad O/P
Intf 0          11.999720          -2.30807      124         15        -20
Intf 1          20.000583          -3.84693      309        -10         32
Intf 2          45.000280          -8.65562      290          3         17
Intf 3         112.000000         -21.54255      306         15         30
-
Continous Display Stopped
TOSMON:
```

11.2.1.3 Continue Observation (CO)

The CO command is the inverse of the SO (stop observation) command. The CO command inserts the DATAREDUCTION and MAJORUPDATE processes into the Observation Timetable

11.2.1.4 Display (DI)

The DI command prints out a set of observation variables and the outputs of all interferometers.

```
TOSMON: DI

Sidereal time:             64 day  0 hour  54 min  33.1 sec.
Observation period =       0 day   0 hour  7 min   0 sec.
Hour Angle = 19H 42M 33.22S  Declination = 20D 45' 23.29''
                 Spacing             Delay        Phi    Real O/P   Quad O/P
Intf 0          11.999720          -2.31393      299          9          2
Intf 1          20.000583          -3.85670        1         -8         13
Intf 2          45.000280          -8.67759      318         -2          5
Intf 3         112.000000         -21.59725       23         -4         25
TOSMON: DI

Sidereal time:             64 day  0 hour  54 min  44.5 sec.
Observation period =       0 day   0 hour  7 min   11 sec.
Hour Angle = 19H 42M 44.62S  Declination = 20D 45' 23.29''
                 Spacing             Delay        Phi    Real O/P   Quad O/P
Intf 0          11.999720          -2.31328      327          5         -4
Intf 1          20.000583          -3.85562       48         14          1
Intf 2          45.000280          -8.67517       63        -19          3
Intf 3         112.000000         -21.59121      284         14        -18
TOSMON:
```

11.2.1.5 Host (HO)

Runs under the timing control of the host computer and the SST control word. The DSP is in HO mode after power on initialisation.

11.2.1.6 Local (LO)

Uses local timing. The host computer communication port and SST control word are ignored. All timing signals and host computer responses are generated locally.

11.2.1.7 Observation Mode (OM)

Selects the mode of Observation.

Subcommands:

- **OB** Observation. The mode for normal observation run
- **CA** Calibration. The mode for observing point source for calibration. The fringe phase is switched 90° after every 1.5 minute for collimation error calibration.

11.2.1.8 Observation Variable (OV)

The OV command prints out the observation variables for each interferometer. The CH subcommand allow access to some variables used in development only.

```
TOSMON: OV
Source HA        =-1.112320e+00
INTF      DELAY          COARSE     FINE     DELTADLY     PHI       DELTAPHI
  0     -2.30148e+00    fffffffe             fd96a0c2      3a3   ff99f675        14
  1     -3.83595e+00    fffffffd             f95016d6      610   ff55ea22        21
  2     -8.63092e+00    fffffff8             faf43fc9      da4   fe817f05        4b
  3     -2.14811e+01    ffffffeb             fc27d21b     21f5   fc482492        bc
Alright? CHange or STop? [ST] Change command is for development use only.
ST
TOSMON:
```

11.2.1.9 Observation Parameters (OP)

The OP command makes a temporary copy of the observation parameters and opens the temporary copy for edit.

Subcommands:

- **CI** Changes Interferometer-specific parameters. Opens the interferometers' specific parameters of baseline length, track errors and collimation errors for edit.

CG Change Global parameters. Opens the global parameters of frequency of observation and source coordinates for edit.

SE Set. Sets the actual parameters to the values of the temporary parameters and reinitialises all associated variables. This command effectively aborts the current observation.

ST STop. Aborts the current edit session and returns to TOSMON.

```
TOSMON: OP
Frequency = 1420505000.00 Hz
Right Ascension = 9H 25M 12.39S  Declination = 30D 9' 13.86''
       SPACING         TRK ERR X   TRK ERR Y   TRK ERR Z       COL ERR
Intf 0    0.0000000          0           0           0         0.00000
Intf 1    0.0000000          0           0           0         0.00000
Intf 2    0.0000000          0           0           0         0.00000
Intf 3    0.0000000          0           0           0         0.00000
OK? STop, SEt, CG change global, CI change interferometer specific par?
CI
Which interferometer?0
Spacing = 0.00000000e+00?
20
Track Error X =         0 tenth mm?
3
Track Error Y =         0 tenth mm?
56
Track Error Z =         0 tenth mm?
24
Collimation Error =     0.000000e+00 degrees?
0.9
Frequency = 1420505000.00 Hz
Right Ascension = 9H 25M 12.39S  Declination = 30D 9' 13.86''
       SPACING         TRK ERR X   TRK ERR Y   TRK ERR Z       COL ERR
Intf 0   20.0000000          3          56          24         0.88959
Intf 1    0.0000000          0           0           0         0.00000
Intf 2    0.0000000          0           0           0         0.00000
Intf 3    0.0000000          0           0           0         0.00000
OK? STop, SEt, CG change global, CI change interferometer specific par?
ST
```

The ST command at the end of the edit aborts all the temporary parameters and returns to TOSMON. On reissuing the OP command, the obseravtion parameters are found to be unchanged.

```
TOSMON: OP
Frequency = 1420505000.00 Hz
Right Ascension = 9H 25M 12.39S  Declination = 30D 9' 13.86''
       SPACING         TRK ERR X   TRK ERR Y   TRK ERR Z       COL ERR
Intf 0    0.0000000          0           0           0         0.00000
Intf 1    0.0000000          0           0           0         0.00000
Intf 2    0.0000000          0           0           0         0.00000
Intf 3    0.0000000          0           0           0         0.00000
OK? STop, SEt, CG change global, CI change interferometer specific par?
CG
Frequency =     1420.50500000 MHz ?
1420.375
Declination = 30D 9' 13.86''
degree ?_
Right Ascension =       9H 25M 12.39S
hr ?_
```

```
Frequency = 1420375000.00 Hz
Right Ascension = 9H 25M 12.39S  Declination = 30D 9' 13.86''
       SPACING          TRK ERR X   TRK ERR Y   TRK ERR Z        COL ERR
Intf 0     0.0000000          0           0           0          0.00000
Intf 1     0.0000000          0           0           0          0.00000
Intf 2     0.0000000          0           0           0          0.00000
Intf 3     0.0000000          0           0           0          0.00000
OK? STop, SEt, CG change global, CI change interferometer specific par?
CG
Frequency =    1420.37500000 MHz ?
_
Declination = 30D 9' 13.86''
degree ? 45
arc min ? 36
arc sec ? 12.89
Right Ascension =      9H 25M 12.39S
hr ? 12
min ? 0
sec ? 25.8

Frequency = 1420375000.00 Hz
Right Ascension = 12H 0M 25.79S  Declination = 45D 36' 12.88''
       SPACING          TRK ERR X   TRK ERR Y   TRK ERR Z        COL ERR
Intf 0     0.0000000          0           0           0          0.00000
Intf 1     0.0000000          0           0           0          0.00000
Intf 2     0.0000000          0           0           0          0.00000
Intf 3     0.0000000          0           0           0          0.00000
OK? STop, SEt, CG change global, CI change interferometer specific par?
CI
Which interferometer?0
Spacing = 0.00000000e+00?
50
Track Error X =        0 tenth mm?
_
Track Error Y =        0 tenth mm?
23
Track Error Z =        0 tenth mm?
_
Collimation Error =    0.000000e+00 degrees?
42
Frequency = 1420375000.00 Hz
Right Ascension = 12H 0M 25.79S  Declination = 45D 36' 12.88''
       SPACING          TRK ERR X   TRK ERR Y   TRK ERR Z        COL ERR
Intf 0    50.0000000          0          23           0         61.99585
Intf 1     0.0000000          0           0           0          0.00000
Intf 2     0.0000000          0           0           0          0.00000
Intf 3     0.0000000          0           0           0          0.00000
OK? STop, SEt, CG change global, CI change interferometer specific par?
SE
SEt will abort  current observation and reinitialise all parameters.
Please confirm. (SEt/STop)SE
```

The SE command at the end of the edit session aborts the current observation and reinitialise all parameters and related variables. On issuing the OP command again, the parameters are found to be changed.

```
TOSMON: OP
Frequency = 1420375000.00 Hz
Right Ascension = 12H 0M 25.79S  Declination = 45D 36' 12.88''
         SPACING        TRK ERR X   TRK ERR Y   TRK ERR Z       COL ERR
Intf 0   50.0000000         0          23          0          61.99585
Intf 1    0.0000000         0           0          0           0.00000
Intf 2    0.0000000         0           0          0           0.00000
Intf 3    0.0000000         0           0          0           0.00000
OK? STop, SEt, CG change global, CI change interferometer specific par?
SI
TOSMON:
```

11.2.1.10 Stop Observation (SO)

The SO command stops the observation by deleting the MAJORUPDATE and DATAREDUCTION from the Observation Timetable. The DSP will not attempt to send any result to the host computer.

11.2.1.11 Help (HE)

Prints out the menu of help messages.

```
TOSMON: HE
What help do you want on? SYstem or OBservation commands? OB
The following OBSERVATION commands are available:
AO      Analog Output
CD      Continous Display
CO      Continue Observation
DI      DIsplay results
TI      TIme
HO      HOst timing
LO      LOcal timing
SO      Stop Observation
OP      edit Observation Parameters
OV      edit Observation Variables
HE      HElp
TOSMON: HE
What help do you want on? SYstem or OBservation commands? SY
The following SYSTEM commands are available:
AC      prints ACtive tasks
ET      Edit Timetable
IC      prints InComplete tasks
PQ      prints Processor Queue
PR      edit PRiority of tasks
PS      edit Process Status
SQ      prints Semaphore Queue
RU      RUn a subroutine or task
HE      HElp
TOSMON:
```

11.2.2 System Commands

11.2.2.1 Edit Timetable (ET)

The ET command initiates the timetable editing mode. Only the deferred timetable, or entries of processes, can be edited.

Subcommands:

DE DElete. Deletes a process from a time table. All entries of the named processes will be deleted from the specified timetable.

DI Display. Displays all lists of deferred jobs.

IN INsert. Inserts a process into a timetable. The user will be asked for the list into which the process is inserted.

```
TOSMON: ET
Which table to edit? SIdereal/OBservation? [OB]
SI
INsert DElete DIsplay or STop? [DI] _
Tenth Second list:
MINUPDAT --->
Half Second list:
One Second list:
Eight Second list:
Minute list:
READSIDE --->
Hour list:
Day list:
INsert DElete DIsplay or STop? [DI] DE
Name of process? READSIDE
INsert DElete DIsplay or STop? [DI] _
Tenth Second list:
MINUPDAT --->
Half Second list:
One Second list:
Eight Second list:
Minute list:
Hour list:
Day list:
INsert DElete DIsplay or STop? [DI] IN
Name of process? READSIDE
Insert into which list?
0 = 1/10 sec, 1 = .5 sec, 2 = 1 sec, 3 = 8 sec, 4 = min, 5 = hour, 6 = day
4
INsert DElete DIsplay or STop? [DI] _
Tenth Second list:
MINUPDAT --->
Half Second list:
One Second list:
Eight Second list:
Minute list:
READSIDE --->
Hour list:
Day list:
INsert DElete DIsplay or STop? [DI] ST
TOSMON:
```

The following example shows how the ET command is used to insert the CONTINUOUSDISPLAY process into the one minute list of the Observation Timetable to force the printing of a DI output every minute.

```
TOSMON: ET
Which table to edit? SIdereal/OBservation? [OB]
SI
INsert DElete DIsplay or STop? [DI]_
Tenth Second list:
MINUPDAT --->
Half Second list:
One Second list:
```

```
Eight Second list:
Minute list:
READSIDE --->
Hour list:
Day list:
INsert DElete DIsplay or STop? [DI] IN
Name of process? CONTDISP
Insert into which list?
0 = 1/10 sec, 1 = .5 sec, 2 = 1 sec, 3 = 8 sec, 4 = min, 5 = hour, 6 = day
4
INsert DElete DIsplay or STop? [DI] _
Tenth Second list:
MINUPDAT --->
Half Second list:
One Second list:
Eight Second list:
Minute list:
READSIDE ---> CONTDISP --->
Hour list:
Day list:
INsert DElete DIsplay or STop? [DI] ST
TOSMON:
Sidereal time:              64 day  2 hour  24 min  0.0 sec.
Observation period =         0 day  0 hour  23 min  45 sec.
Hour Angle = 14H 23M 34.35S  Declination = 45D 36' 12.88''
                 Spacing            Delay      Phi    Real O/P  Quad O/P
Intf 0          50.000000        -4.69038      351      -14      -14
Intf 1          25.000000        -2.34515       27       15       -7
Intf 2          80.000000        -7.50449      226        1       16
Intf 3         112.000000       -10.50629      166        0       43

Sidereal time:              64 day  2 hour  25 min  0.0 sec.
Observation period =         0 day  0 hour  24 min  41 sec.
Hour Angle = 14H 24M 34.34S  Declination = 45D 36' 12.88'
                 Spacing            Delay      Phi    Real O/P  Quad O/P
Intf 0          50.000000        -4.71678      180      -13        7
Intf 1          25.000000        -2.35836      301       -5       14
Intf 2          80.000000        -7.54674      239       23       [illegible]
Intf 3         112.000000       -10.56544      330       63       [illegible]

TOSMON: ET
Which table to edit? SIdereal/OBservation? [OB]
SI
INsert DElete DIsplay or STop? [DI] DE
Name of process? CONTDISP
INsert DElete DIsplay or STop? [DI] _
Tenth Second list:
MINUPDAT --->
Half Second list:
One Second list:
Eight Second list:
Minute list:
READSIDE --->
Hour list:
Day list:
INsert DElete DIsplay or STop? [DI] ST
TOSMON:
```

11.2.2.2 Incomplete (IC)

Prints out the name of the process(es) which cannot be completed within their given time. Applies to jobs listed in defered timetables only.

11.2.2.3 MACSBUG (MB)

Transfers control to the resident Motorola monitor program MACSBUG. All TOS and observation activities are stopped. Control could be transfered back to TOSMON by the GO command if register and memory content have not been disturbed. The P2 command will set MACSBUG to transparent mode which allows the user to communicate directly with the host computer. Control A will leave the transparent mode and resume normal MACSBUG operations. For details of MACSBUG commands, please refer to "MC68000 DESIGN MODULE USER'S GUIDE" published by Motorola.

11.2.2.4 Processor Queue (PQ)

The PQ command takes a snap shot of the processor queue and prints it out. The user is not allowed to change the processor queue. PQ returns to TOSMON automatically.

11.2.2.5 Priority (PR)

The PR command allows the priority of the process, regardless of status, to be edited.

Subcommands:

- **CH** Change. Changes the priority of a process by putting it in front of another process.
- **ST** Stop. Stops the current edit session and returns to TOSMON.

```
TOSMON: PR
PRIORITY:
DATAREDU ---> MINUPDAT ---> MAJUPDAT ---> SENDHOST ---> READHOST --->
READSIDE ---> CONTDISP ---> TOSMON   ---> TEST     ---> REPORT   --->
IDLE     --->
CHange or SToP? [ST] CH
Process to be moved ? READHOST
Place it infornt of ? SENDHOST
PRIORITY:
DATAREDU ---> MINUPDAT ---> MAJUPDAT ---> READHOST ---> SENDHOST --->
READSIDE ---> CONTDISP ---> TOSMON   ---> TEST     ---> REPORT   --->
IDLE     --->
CHange or SToP? [ST] _
TOSMON: PR
PRIORITY:
DATAREDU ---> MINUPDAT ---> MAJUPDAT ---> READHOST ---> SENDHOST --->
READSIDE ---> CONTDISP ---> TOSMON   ---> TEST     ---> REPORT   --->
IDLE     --->
CHange or SToP? [ST] CH
Process to be moved ? IDLE
Thou shall not change the priority or status of IDLE.
```

```
PRIORITY:
DATAREDU ---> MINUPDAT ---> MAJUPDAT ---> READHOST ---> SENDHOST --->
READSIDE ---> CONTDISP ---> TOSMON   ---> TEST     ---> REPORT   --->
IDLE     --->
CHange or STop? [ST] CH
Process to be moved ? CONTINUOUS
No such process!
PRIORITY:
DATAREDU ---> MINUPDAT ---> MAJUPDAT ---> READHOST ---> SENDHOST --->
READSIDE ---> CONTDISP ---> TOSMON   ---> TEST     ---> REPORT   --->
IDLE     --->
CHange or STop? [ST] CONTDISP
Unknown Command!
PRIORITY:
DATAREDU ---> MINUPDAT ---> MAJUPDAT ---> READHOST ---> SENDHOST --->
READSIDE ---> CONTDISP ---> TOSMON   ---> TEST     ---> REPORT   --->
IDLE     --->
CHange or STop? [ST] CH
Process to be moved ? CONTDISP
Place it infornt of ? TOSMON
PRIORITY:
DATAREDU ---> MINUPDAT ---> MAJUPDAT ---> READHOST ---> SENDHOST --->
READSIDE ---> CONTDISP ---> TOSMON   ---> TEST     ---> REPORT   --->
IDLE     --->
CHange or STop? [ST] CH
Process to be moved ? CONTDISP
Place it infornt of ? TEST
PRIORITY:
DATAREDU ---> MINUPDAT ---> MAJUPDAT ---> READHOST ---> SENDHOST --->
READSIDE ---> TOSMON   ---> CONTDISP ---> TEST     ---> REPORT   --->
IDLE     --->
CHange or STop? [ST]_
TOSMON:
```

11.2.2.6 Process Status (PS)

The PS command prints out a list of all processes and their status.

Subcommands

- **CH** Changes the process status by exclusive-or'ing the process status with a user entered mask
- **SE** Sets the process status to the user-entered value.
- **ST** Stop Stops the process status mode and returns to TOSMON.

```
TOSMON: PS
PROCESSID:                 PROCESS STATUS:
DATAREDU                   0000
MINUPDAT                   0000
MAJUPDAT                   0000
READHOST                   0400
SENDHOST                   0400
READSIDE                   0000
TOSMON                     0600
CONTDISP                   0000
TEST                       0000
REPORT                     0000
IDLE                       0600
```

```
SEt, CHange or STop? CH
Process? CONTDISP
Enter mask in hex for EX-OR: 0400
PROCESSID:                       PROCESS STATUS:
DATAREDU                         0000
MINUPDAT                         0000
MAJUPDAT                         0000
READHOST                         0400
SENDHOST                         0400
READSIDE                         0000
TOSMON                           0600
CONTDISP                         0400
TEST                             0000
REPORT                           0000
IDLE                             0600
SEt, CHange or STop? CH
Process? IDLE
Thou shall not change the priority or status of IDLE.
PROCESSID:                       PROCESS STATUS:
DATAREDU                         0000
MINUPDAT                         0000
MAJUPDAT                         0000
READHOST                         0400
SENDHOST                         0400
READSIDE                         0000
TOSMON                           0600
CONTDISP                         0400
TEST                             0000
REPORT                           0000
IDLE                             0600
SEt, CHange or STop? ST
TOSMON:
```

11.2.2.7 RUN (RU)

Runs a C program or a subroutine in memory. The program is represented by process descriptor TEST. Since no memory protection is available, running of test programs during serious observation is NOT recommended.

11.2.2.8 Semaphore Queue (SQ)

The SQ command prints out the queue of the processes blocked by a semaphore. The user is asked to enter the start address of a semaphore. The system checks the semaphore queue and prints the processes out only if they can all be found in the static queue. An error message will be given otherwise. This checking prevents incorrect entry of a semaphore address, and eliminates the possibility of searching through an infinite list.

11.2.2.9 Time (TI)

The TI command can be used to set the Sidereal and Observation clock.

```
TOSMON: TI
Which clock? SIdereal or OBservation? [OB] SI
Day: 64,  Hour: 2,  Min: 26,  Sec(60): 6,  Sec(8): 6
CHange or STop? [ST] CH
Day =  64 ?_
Hour = 2?_
Min  = 26?_
Sec(60) =       31? _
Day: 64,  Hour: 2,  Min: 26,  Sec(60): 33,  Sec(8): 1
CHange or STop? [ST] _
TOSMON: TI
Which clock? SIdereal or OBservation? [OB] _
Day: 0,  Hour: 0,  Min: 28,  Sec(60): 16,  Sec(8): 0
CHange or STop? [ST] CH
Day =  0 ?_
Hour = 0?
2
Min  = 28?
40
Sec(60) =       2?_3
Day: 0,  Hour: 2,  Min: 40,  Sec(60): 3,  Sec(8): 7
CHange or STop? [ST]_
TOSMON:
```

12. APPENDIX IV: SIMULATION SOFTWARE LISTING

This appendix includes the FORTRAN listing of the noise generator and the simulator programs. These programs were developed under the Michigan Terminal System (MTS) of the University of Alberta and call subroutines from the International Mathematics and Statistics Library (IMSL).

```
C-----------------------------------------------------------------
C
C       PROGRAM RANGEN
C
C           FUNCTION: FOR GENERATION OF SAMPLED BAND-LIMITED
C                     GAUSSIAN NOISE SEQUENCES INTERACTIVELY.
C                     COMMANDS AVAILABLE ARE:
C                         GENERATION OF GAUSSIAN NOISE SEQUENCE
C                         LOW-PASS FILTERING
C                         HILBERT TRANSFORMATION
C                         MERGING OF NOISE STREAMS
C                         NORMALISATION OF NOISE SEQUENCE
C                         PRINT A NOISE SEQUENCE
C
C           USE:      $RUN RANGEN.OBJ+*IMSLLIB 11=NOISE1
C                     12=NOISE2 13=NOISE3
C
C           IO ASSIGNMENT
C                     UNIT 11, 12, AND 13 CORRESPOND TO FILE
C                     1, 2, AND 3 RESPECTIVELY
C                     UNIT 19 AND 20 ARE ATTACHED TO THE TERMINAL
C
C           LIBRARY SUPPORT REQUIRED:
C                     INTERNATIONAL MATHEMATICS AND STATISTICAL
C                     LIBRARY (IMSL)
C
C       VERSION 1.3   20 JUNE 1982
C
C-----------------------------------------------------------------



C-----------------------------------------------------------------
C     DEFINE FUNCTION SINC
C-----------------------------------------------------------------
      FUNCTION SINC(X)
C-----------------------------------------------------------------
      IF (ABS(X) .LE. .0001) GOTO 50
      SINC = (SIN(3.1415926*X))/(3.1415926*X)
      RETURN
   50 SINC = 1.0
      RETURN
      END


C     -----------------------------------------------------------
C     DEFINE CONVOLUTION SUBROUTINE
C     SAM     SAMPLE ARRAY DIMENSION LSAM
C     FCN     FUNCTION ARRAY DIMENSION LFCN
C     RESULT  RESULT ARRAY DIMENSION LRES
C     -----------------------------------------------------------
      SUBROUTINE CONVOL(LSAM,SAM,LFCN,FCN,LRES,RESULT)
C     -----------------------------------------------------------
      DIMENSION SAM(LSAM), FCN(LFCN), RESULT(LRES)
      DO 140 I=1,LRES
      XSAM=0.0
      DO 150 J=1,LFCN
      XSAM = XSAM + FCN(LFCN+1-J)*SAM(J+I)
  150 CONTINUE
      RESULT(I) = XSAM
```

```
  140 CONTINUE
      RETURN
      END


C     ----------------------------------------------------------------
C     DEFINE HILBERT TRANSFORM BY FFT
C     SAM      INPUT ARRAY DIMENSION LSAM
C     CSAM     COMPLEX ARRAY WORKSPACE DIMENSION LSAM
C     IWK,WK   WORKSPACE ARRAY DIMENSION LWK
C     ----------------------------------------------------------------
      SUBROUTINE HTFFT(LSAM, SAM, CSAM, LWK, IWK, WK)
C     ----------------------------------------------------------------
C
      COMPLEX CSAM
      DIMENSION SAM(LSAM),CSAM(LSAM),IWK(LWK),WK(LWK)
C
C     SET UP CSAM
C
      DO 100 I=1,LSAM
  100 CSAM(I) = CMPLX(SAM(I),0.0)
C
C     PERFORM FORWARD TRANSFORM, USUALLY REQUIRED TO CONJUGATE
C     THE SAMPLE BUT SINCE THE PRESENT SAMPLE IS REAL, NO NEED
C
      CALL FFTCC(CSAM, LSAM, IWK, WK)
C
C     CONJUGATION AFTER TRANSFORMATION AS WELL
C
      DO 150 I=1,LSAM
          CSAM(I) = CONJG(CSAM(I))
  150 CONTINUE
      LHSAM=LSAM/2
C
C     MULTIPLY POS FREQ COMPONENTS BY J AS IN HILBERT TRANS
C     DEFINED BY BRACEWELL.  POSTIVE FREQ STARTS FORM O TO LSAM/2
C
      DO 200 I= 1, LHSAM
      CSAM(I) = CSAM(I)*(0.0 ,1.0)
  200 CONTINUE
C
C     MULTIPLY NEG FREQ COMPONENTS BY -J AS IN HILBERT TRANS
C     NEG FREQ STARTS FROM LSAM TO LSAM/2 + 1
C
      LHSAM1=LSAM/2 + 2
      DO 300 I=LHSAM1,LSAM
      CSAM(I) = CSAM(I)*(0.0,-1.0)
  300 CONTINUE
C
C     PERFORM INVERSE FORUIER TRANSFORM
C
      CALL FFTCC(CSAM,LSAM,IWK,WK)
CO    WRITE(6,1000)(CSAM(I),I=1,LSAM)
 1000 FORMAT('1AFTER HILBERT TRANSFORM WITH FFT',///,(4F15.5))
      DO 400 I=1,LSAM
  400 SAM(I) = REAL(CSAM(I))
      RETURN
      END
```

```
C     ---------------------------------------------------
C     DEFINE SUBROUTINE READFILE.  READF READS IN A FILE OF
C     LENGTH LONG INTO AN THE ARRAY SAMPLE
C     ---------------------------------------------------
      SUBROUTINE READF(LENGTH,SAMPLE)
C     ---------------------------------------------------
C
      DIMENSION SAMPLE(LENGTH)
      WRITE(20,100)
  100 FORMAT('ENTER THE SOURCE FILE YOU WANT TO READ FORM.',
     C '  ANSWER 1,2 OR 3 ONLY.')
      READ(19,200)IFILE
  200 FORMAT(I1)
      IFILE = IFILE+10
      FIND(IFILE'1000)
      READ(IFILE)(SAMPLE(I),I=1,LENGTH)
      RETURN
      END


C     ---------------------------------------------------
C     DEFINE WRITEFILE.  WRITEF IS THE COMPLEMENT OF READF
C     ---------------------------------------------------
      SUBROUTINE WRITEF(LENGTH,SAMPLE)
C     ---------------------------------------------------
C
      DIMENSION SAMPLE(LENGTH)
      WRITE(20,100)
  100 FORMAT('ENTER THE FILE YOU WANT RESULTS TO BE STORED.',
     C '  ANSWER 1,2 OR 3 ONLY')
      READ(19,200)IFILE
  200 FORMAT(I1)
      IFILE = 10 + IFILE
      FIND(IFILE'1000)
      WRITE(IFILE)(SAMPLE(I),I=1,LENGTH)
      WRITE(20,1040)
 1040 FORMAT(/,'DONE',/)
      RETURN
      END


C     ---------------------------------------------------
C     DEFINE VARMEN.  VARMEN CALCULATES THE VARIANCE AND
C     OF A NOISE SEQUENCE.
C     SAMPLE  THE INPUT NOISE SAMPLE DIMENSION LENGTH
C     AMEAN   THE RETRUNED MEAN
C     VAR     THE RETURNED VARIANCE
C     ---------------------------------------------------
      SUBROUTINE VARMEN(LENGTH,SAMPLE,AMEAN,VAR)
C     ---------------------------------------------------
C
      DIMENSION SAMPLE(LENGTH)
      SUM=0
      VAR=0
      DO 200 J=1,LENGTH
      SAMP=SAMPLE(J)
      SUM=SUM+SAMP
      VAR=VAR+SAMP*SAMP
  200 CONTINUE
      ALENG = LENGTH
```

```
      AMEAN=SUM/ALENG
      VAR=VAR/ALENG
CO    WRITE(6,1000)AMEAN,VAR
C1000 FORMAT(///,' MEAN =',E15.7,'    VARIANCE =',E15.7)
      RETURN
      END


C     ------------------------------------------------------------
C     DEFINNE ANMLS. ANMLS NORMALISE A NOISE SEQUENCE TO
C     UNIT VARIANCE.
C     SAM      THE INPUT SAMPLE DIMENSIONED LSAM
C     ------------------------------------------------------------
      SUBROUTINE ANMLS(LSAM,SAM)
C     ------------------------------------------------------------
C
      DIMENSION SAM(LSAM)
      CALL VARMEN(LSAM,SAM,AMEAN,VAR)
      STDEVA = SQRT(VAR)
      DO 100 I=1,LSAM
  100 SAM(I)=SAM(I)/STDEVA
      RETURN
      END



C-----------------------------------------------------------------
C
C     MAIN PROGRAM
C
C-----------------------------------------------------------------
      DOUBLE PRECISION DSEED
      COMPLEX CSAM
      REAL M,N
      DIMENSION SAMPLE(8500),SAM1(8500),CSAM(8500),IWK(8500),
     C WK(8500),SINCA(500)
      DATA G,F,M,N,H,P,S/'G','F','M','N','H','P','S'/
C
C     SET UP TERMINAL COMMUNICATION LOGICAL UNITS
C
      CALL FTNCMD('ASSIGN 19=*MSOURCE*',19)
      CALL FTNCMD('ASSIGN 20=*MSINK*',17)
      WRITE(20,1000)
 1000 FORMAT(/,'LOW PASSED FILTER NOISE GENERATION PROGRAM.'
     C,/,'  MANIPULATES THREE FILES 1,2 AND 3.'
     C,/,'INPUT THE LENGTH OF NOISE SAMPLE',
     C' WANTED',/,'PUT IN A DECIMAL PT FOR THE STUPID FORMAT')
      READ (19,1500)AL
 1500 FORMAT(G15.5)
      LENGTH = AL
      WRITE(20,1050)LENGTH
 1050 FORMAT(/,'LENGTH OF OPERATION =',I6)
      WRITE(20,1010)
 1010 FORMAT(//,' COMMAND SELECTION:',//,' G = GENERATE WHITE NOISE'
     C,/, ' F = LOWPASS FILTERING',/,' M = MERGING 2 NOISE SEQUENCE'
     C,/, ' N = NORMALISE A NOISE SEQUENCE',
     C/,' H = HILBERT TRANSFORM',/,' P = PRINT',/,' S = STOP')
 2000 CONTINUE
      WRITE(20,1020)
 1020 FORMAT('COMMAND?')
```

```
      READ(19,1510)CMD
 1510 FORMAT(A1)
      IF (CMD .EQ. G) GOTO 2100
      IF (CMD .EQ. F) GOTO 2200
      IF (CMD .EQ. M) GOTO 2300
      IF (CMD .EQ. H) GOTO 2400
      IF (CMD .EQ. P) GOTO 2500
      IF (CMD .EQ. N) GOTO 2600
      IF (CMD .EQ. S) GOTO 9999
      WRITE(20,1030)
 1030 FORMAT(' INVALID COMMAND')
      WRITE(20,1010)
      GOTO 2000
C
C     GNERATION SECTION
C
 2100 WRITE(20,1100)
 1100 FORMAT(' ENTER DSEED')
      READ(19,1600)SEED
 1600 FORMAT(G15.5)
      DSEED=SEED
      WRITE(20,1105)
 1105 FORMAT(' WANT 1) GGNML  OR  2) GGNPM ?  ANS 1 OR 2')
      READ(19,1605)IRSEL
 1605 FORMAT(I1)
      IF (IRSEL .EQ. 2) GOTO 2110
      CALL GGNML(DSEED, LENGTH, SAMPLE)
      GOTO 2120
 2110 CALL GGNPM(DSEED, LENGTH, SAMPLE)
 2120 CONTINUE
      CALL WRITEF(LENGTH,SAMPLE)
      GOTO 2000
C
C     LOW PASS FILTERING SECTION
C
 2200 WRITE(20,1200)
 1200 FORMAT(' LOW PASS FILTERING')
      CALL READF(LENGTH,SAMPLE)
      WRITE(20,1210)
 1210 FORMAT(' ENTER THE WIDTH OF SINC FUNCTION',//,
     C 'PUT IN A DECIMAL PT')
      READ(19,1610)ALSINC
 1610 FORMAT(G15.5)
      LSINC=ALSINC
      WRITE(20,1220)
 1220 FORMAT(' ENTER THE SCALE OF SINC FUNCTION. 2. 4. 8. OR 16.')
      READ(19,1610) SCALE
      MSINC=LSINC/2
      MSINC1=MSINC+1
      DO 2210 I=1,MSINC1
      AI=I
      SINCA(MSINC+I) = SINC((AI-1.)/SCALE)
      SINCA(MSINC+2-I) = SINCA(MSINC+I)
 2210 CONTINUE
C
C     CALL CONVOLUTION SUBROUTINE
C
      CALL CONVOL(LENGTH+LSINC,SAMPLE,LSINC,SINCA,LENGTH,SAM1)
      CALL ANMLS(LENGTH,SAM1)
      CALL WRITEF(LENGTH,SAM1)
```

```
      GOTO 2000
C
C     NOISE SEQUENCE MERGING SECTION
C
 2300 WRITE(20,1300)
 1300 FORMAT(' MERGE TWO NOISE SOURCE TOGETHER WITH A SHIFT',
     C ' AND WEIGHT'
     C ,/,'ENTER NOISE SOURCE 1, UNSHIFTED, UNITY WEIGHT')
      CALL READF(LENGTH, SAMPLE)
      WRITE(20,1305)
 1305 FORMAT(' ENTER NOISE SOURCE 2, SHIFTED AND WEIGHTED.')
      CALL READF(LENGTH,SAM1)
      WRITE(20,1310)
 1310 FORMAT(' ENTER THE NO OF SHIFTS IN SOURCE 2 WITH DECIMAL PT.')
      READ(19,1350)SHIFT
 1350 FORMAT(G15.5)
      ISHIFT=SHIFT
      WRITE(20,1320)
 1320 FORMAT(' ENTER THE WEIGHT OF SOURCE 2')
      READ(19,1350)WEIGHT
      ICOPY= LENGTH-ISHIFT
      DO 2310 I=1,ICOPY
 2310 SAMPLE(I) = SAMPLE(I) + SAM1(I+ISHIFT)*WEIGHT
      CALL ANMLS(LENGTH,SAMPLE)
      CALL WRITEF(LENGTH,SAMPLE)
      GOTO 2000
C
C     HILBERT TRANSFORM SECTION
C
 2400 WRITE(20,1400)
 1400 FORMAT(' HILBERT TRANSFORM.',//,
     C 'PERFORMS HILBERT TRANSFORMATION ON A NOISE SAMPLE BY FFT.')
      CALL READF(LENGTH,SAMPLE)
      CALL HTFFT(LENGTH,SAMPLE,CSAM,5000,IWK,WK)
      CALL WRITEF(LENGTH,SAMPLE)
      GOTO 2000
C
C     PRINT SECTION
C
 2500 WRITE(20,1450)
 1450 FORMAT(//,'PRINTS THE NUMBERS IN A NOISE SEQUENCE.',//,
     C'SELECT THE FILE YOU WANT TO PRINT.')
      CALL READF(LENGTH,SAMPLE)
      WRITE(20,1455)
 1455 FORMAT(/'ENTER THE START PT. PUT IN A DECIMAL PT.')
      READ(19,1950)START
 1950 FORMAT(G15.5)
      ISTART=START
      WRITE(20,1460)
 1460 FORMAT(/'ENTER THE END PT. PUT IN A DECIMAL PT.')
      READ(19,1950)ENDP
      IENDP=ENDP
      WRITE(6,1470)ISTART,IENDP,(SAMPLE(I),I=ISTART,IENDP)
 1470 FORMAT('1',//,'SECTION OF NOISE FROM SAMPLE',I5,
     C'  TO SAMPLE',I5,///,(5G15.7))
      GOTO 2000
C
C     NORMALISE SECTION
C
 2600 WRITE(20,1700)
```

7

```
      1700 FORMAT(' NORMALISE A SEQUENCE OF RANDOM NUMBER')
           CALL READF(LENGTH,SAMPLE)
           CALL ANMLS(LENGTH,SAMPLE)
           CALL WRITEF(LENGTH,SAMPLE)
           GOTO 2000
      C
      C    STOP
      C
      9999 STOP
           END
END OF FILE
```

```
C---------------------------------------------------------------
C
C      PROGRAM SIMUL
C      FUNCTION: TO SIMULATE A DIGITAL CROSSCORRELATOR
C                AND COMPARE DIFFERENT MEANS OF QUADRATURE
C                CHANNEL GENERATION.
C
C      IO ASSIGNMENT:
C                UNIT 19,20 = TERMINAL, UNIT 6 = PRINTER OUTPUT.
C                UNIT 10 = OUTPUT DATA FILE FOR GRAPH PLOTTING.
C                UNIT 11 = NOISEA, 12 = NOISEB, 13 = SIGNALA
C                UNIT 14 = SIGNALB, 15 = SIGNALB HILBERT TRANSFORMED
C
C      LIBRARY SUPPORT REQUIRED:
C                INTERNATIONAL MATHEMATICAL AND STATISTICAL LIBRARY
C
C      VERSION 2.2  20 JUNE 1982.
C
C---------------------------------------------------------------


C      ----------------------------------------------------
C      FUNCTION TO QUANTISE AN ARRAY INTO 3 LEVELS
C      SAMPLE  INPUT NOISE SAMPLE DIMENSION LSAM
C      QLEV1, QLEV2  DECISION LEVELS
C      OUTPUT 3 LEVEL ARRAY OVERWRITES SAMPLE
C      ----------------------------------------------------
       SUBROUTINE QTN3(LSAM,SAMPLE,QLEV1,QLEV2)
C      -----------------------------------
C
       DIMENSION SAMPLE(LSAM)
       DO 170 I=1,LSAM
       IF (SAMPLE(I) .GE. QLEV1) SAMPLE(I)=1.0
       IF (SAMPLE(I).LT.QLEV1 .AND. SAMPLE(I).GT.QLEV2) SAMPLE(I)=0.0
       IF (SAMPLE(I) .LE. QLEV2) SAMPLE(I)=-1.0
   170 CONTINUE
       RETURN
       END


C      ----------------------------------------------------
C      CORSS CORRELATION ROUTINE.
C      SAM1, SAM2, INPUT NOISE SAMPLES DIMENSIONED LSAM
C      XR       CORRELATOR ACCUMULATORS DIMENSIONED LXR
C      ----------------------------------------------------
       SUBROUTINE XCOR(LSAM, SAM1, SAM2, LXR, XR)
C      -----------------------------------
C
       DIMENSION SAM1(LSAM), SAM2(LSAM), XR(LXR)
       LRUN = LSAM-LXR
C
C      INITIALISE XR
C
       DO 210 I=1,LXR
       SUM=0.0
       DO 220 J=1,LRUN
       SUM = SUM+ SAM1(I+J)*SAM2(LXR+1-I+J)
   220 CONTINUE
       XR(I)= SUM
   210 CONTINUE
```

```
C       WRITE(6,1000)(XR(I),I=1,LXR)
C1000 FORMAT('1','CROSSCORRELATION',//,(5F15.7))
        RETURN
        END


C      ---------------------------------------------------------------
C       CONVOLUTION ROUTINE
C       SAM      INPUT NOISE SAMPLE DIMENSION LSAM
C       FCN      FUNCTION TO BE CONVOLVED WITH DIMENSION LFCN
C       RESULT   RESULTING ARRAY DIMENSION LRES
C      ---------------------------------------------------------------
        SUBROUTINE CONVOL(LSAM,SAM,LFCN,FCN,LRES,RESULT)
C      ---------------------------------------------------
C
        DIMENSION SAM(LSAM), FCN(LFCN), RESULT(LRES)
        DO 140 I=1,LRES
        XSAM=0.0
        DO 150 J=1,LFCN
        XSAM = XSAM + FCN(LFCN+1-J)*SAM(J+I)
  150   CONTINUE
        RESULT(I) = XSAM
  140   CONTINUE
        RETURN
        END


C      ---------------------------------------------------------------
C       SUBROUTINE PLOT. PLOTS GRAPH ON A LINE PRINTER
C       GRAPH1, GRAPH2  GRAPHS TO BE PLOTTED, DIMENSION LGRAPH
C      ---------------------------------------------------------------
        SUBROUTINE PLOT(LGRAPH,GRAPH1,GRAPH2)
C      ---------------------------------------
C
        DIMENSION A(200), GRAPH1(LGRAPH), GRAPH2(LGRAPH)
        DATA STAR, CROSS, DOT, BLANK/'*','X','.',' '/
        LA= 121
        ALA = LA
        AMAX =-1.0E30
        AMIN = 1.0E30
        DO 100 I=1,LGRAPH
        IF (AMAX .LT. GRAPH1(I)) AMAX=GRAPH1(I)
        IF (AMIN .GT. GRAPH1(I)) AMIN=GRAPH1(I)
        IF (AMAX .LT. GRAPH2(I)) AMAX=GRAPH2(I)
        IF (AMIN .GT. GRAPH2(I)) AMIN=GRAPH2(I)
  100   CONTINUE
        WRITE(6,1000)AMIN,AMAX
 1000   FORMAT(' MIN=',E15.7,'
     C'                                                  ','MAX=',E15.7)
        DO 110 I=1,LA
  110   A(I)=DOT
        WRITE(6,1010)(A(I),I=1,LA)
 1010   FORMAT('0',200A1)
        RANGE = AMAX-AMIN
        IXAXIS= 1
        IF((AMAX*AMIN) .GT. 0) GOTO 120
        IXAXIS= (-AMIN/RANGE)*(ALA-1) +1.5
  120   CONTINUE
        DO 200 I=1,LA
  200   A(I)= BLANK
```

```
            IY1 = 1
            IY2 = 1
            DO 210 I=1,LGRAPH
            A(IY1)= BLANK
            A(IY2)= BLANK
            A(IXAXIS) = DOT
            IY1 = ((GRAPH1(I) - AMIN)*(ALA-1))/RANGE +1.5
            A(IY1) = STAR
            IY2 = ((GRAPH2(I) - AMIN)*(ALA-1))/RANGE +1.5
            A(IY2) = CROSS
            LWRITE = MAXO(IXAXIS,IY1,IY2)
            WRITE(6,1010)(A(J),J=1,LWRITE)
        210 CONTINUE
            RETURN
            END

      C     -----------------------------------------------------
      C     ROUTINE TO CALCULATE VARIANCE AND MEAN
      C     SAMPLE  INPUT NOISE SEQUENCE DIMENSION LENGTH
      C     AMEAN   RETURNED MEAN VALUE
      C     VAR     RETURNED VARANCE
      C     -----------------------------------------------------
            SUBROUTINE VARMEN(LENGTH,SAMPLE,AMEAN,VAR)
      C     ---------------------------------------
      C
            DIMENSION SAMPLE(LENGTH)
            SUM=0
            VAR=0
            DO 200 J=1,LENGTH
            SAMP=SAMPLE(J)
            SUM=SUM+SAMP
            VAR=VAR+SAMP*SAMP
        200 CONTINUE
            ALENG = LENGTH
            AMEAN=SUM/ALENG
            VAR=VAR/ALENG
      CO    WRITE(6,1000)AMEAN,VAR
      C1000 FORMAT(///,' MEAN =',E15.7,'     VARIANCE =',E15.7)
            RETURN
            END


      C     -----------------------------------------------------
      C     DEFINE SINC FUNCTION
      C     -----------------------------------------------------
      C     SINC FUNCTION
      C     -------------
            FUNCTION SINC(X)
            IF(ABS(X) .LE. 1E-5) GOTO 100
            PIX = 3.1415926535 * X
            SINC = ( SIN(PIX) ) / PIX
            GOTO 110
      100   SINC = 1
      110   RETURN
            END


      C     -----------------------------------------------------
      C     DEFINE COSC FUNCTION, THE HILBERT TRANSFORM OF SINC
      C     COSC FUNCTION (HILBERT TRANSFORM OF SINC)
```

```
C     ---------------------------------------------------------------
      FUNCTION COSC(X)
C     ----------------
      IF (ABS(X) .LE. 1E-5) GOTO 100
      PIX = 3.1415926359 * X
      COSC = ( COS(PIX) - 1.0 ) / PIX
      GOTO 110
100   COSC = 0
110   RETURN
      END


C     ---------------------------------------------------------------
C     TIME DOMAIN INTERPOLATION FUNCTION W1(T) = INVERSE F.T.
C     OF 50% RAISED COSINE FUNCTION.
C     ---------------------------------------------------------------
      FUNCTION W1(T)
C     --------------
      W1 = SINC(T) + 2*SINC(2*T) -SINC(2*(T+1))
     C       +0.5*SINC(T+1.) -SINC(2*(T-1)) +0.5*SINC(T-1)
      W1 = W1/3.
      RETURN
      END


C     ---------------------------------------------------------------
C     H1(X) HILBERT TRANSFORM OF INTERPOLATION FUNCTION W1(T)
C     ---------------------------------------------------------------
      FUNCTION H1(T)
C     --------------
      H1 = COSC(T) + 2*COSC(2*T) -COSC(2*(T+1.))
     C       +0.5*COSC(T+1.) -COSC(2*(T-1.)) +0.5*COSC(T-1.)
      H1 = H1/3.
      RETURN
      END


C     ---------------------------------------------------------------
C     EXTENDED CONVOLUTION PRODUCE A RESULT WITH THE SAME
C     AS THE INPUT SAMPLE
C     SAM     INPUT NOISE SAMPLE DIMENSION LSAMEX
C     RESULT  OUTPUT RESULTS DIMENSION LSAMEX
C     FA      THE FUNCTION TO BE CONVOLVED WITH, DIM LF
C     THE INPUT SAMPLE CONTAINS ONLY LSAMEX - 2*LF USEFUL
C     SAMPLES WHEN ENTERING THE ROUTINE.
C     ---------------------------------------------------------------
      SUBROUTINE EXTCON(LSAMEX,SAM,LF,FA,RESULT)
C     ------------------------------------------
C
      DIMENSION SAM(LSAMEX),FA(LF),RESULT(LSAMEX)
C
C     PAD THE SAMPLE UP WITH ZEROS AT BOTH ENDS
C
      LSAM = LSAMEX - 2*LF
      DO 200 I=1, LSAM
      J= LSAMEX-LF+1-I
  200 SAM(J) = SAM(J-LF)
      DO 210 I=1,LF
  210 SAM(I)=0.0
      IEMPTY= LSAMEX-LF+1
```

```
      DO 220 I=IEMPTY,LSAMEX
  220 SAM(I)=0.0
C
C     CALL CONVOLUTION ROUTINE TO PERFORM H.T.
C
      CALL CONVOL(LSAMEX,SAM,LF,FA,LSAMEX-LF,RESULT)
C
C     COPY RESULT BACK TO SAM AND PAD UP WITH ZEROS
C
      DO 300 I=1,LSAM
  300 SAM(I) = RESULT(I+LF/2)
      LSAM1 = LSAM+1
      DO 310 I= LSAM1,LSAMEX
  310 SAM(I)=0.0
CD    WRITE(6,1040)(SAM(I),I=1,LSAM)
C1040 FORMAT('1AFTER HILBERT TRANSFORM WITH CONV',//,(5F15.7))
      RETURN
      END


C     ------------------------------------------------------------
C     SUBROUTINE HTCON. HILBERT TRANSFORM BY CONVOLUTION
C     SAM      NOISE INPUT SAMPLE DIMENSION LSAMEX
C     RESULT   OUTPUT SEQUENCE DIMENSION LSAMEX
C     FA       WORK SPACE DIMENSIONED LF
C     LSAMEX IS THE EXTENDED LENGTH OF THE SAMPLE IE
C     LSAMEX = LSAM + 2*LF
C     ------------------------------------------------------------
      SUBROUTINE HTCON(LSAMEX,SAM,LF,FA,RESULT)
C     ---------------------------------------
      DIMENSION SAM(LSAMEX),FA(LF),RESULT(LSAMEX)
      AM1F = LF/2+1
      DO 100 I= 1,LF
      AI=I
      FA(I)= H1((AI-AM1F)/8.0)
  100 CONTINUE
C
C     CALL EXTCON TO PERFORM CONVOLUTION WITH EXTENDED LENGTH
C
      CALL EXTCON(LSAMEX,SAM,LF,FA,RESULT)
      SCALE = 3./16.
      DO 200 I=1,LSAMEX
      SAM(I) = SAM(I)*SCALE
  200 CONTINUE
      RETURN
      END


C     ------------------------------------------------------------
C     PERFORM HILBERT TRANSFORM BY FFT
C     SAM      INPUT SAMPLES TO BE TRANSFORMED DIM LSAM
C     CSAM     COMPLEX WORKSPACE DIM LSAM
C     IWK,WK   WORKSPACE DIMENSION LWK
C     ------------------------------------------------------------
      SUBROUTINE HTFFT(LSAM, SAM, CSAM, LWK, IWK, WK)
C     ---------------------------------------
C
      COMPLEX CSAM
      DIMENSION SAM(LSAM),CSAM(LSAM),IWK(LWK),WK(LWK)
C
```

```
C      SET UP CSAM
C
       DO 100 I=1,LSAM
   100 CSAM(I) = CMPLX(SAM(I),0.0)
C
C      PERFORM FORWARD TRANSFORM, USUALLY REQUIRED TO CONJUGATE
C      THE SAMPLE BUT SINCE THE PRESENT SAMPLE IS REAL, NO NEED
C
       CALL FFTCC(CSAM, LSAM, IWK, WK)
C
C      CONJUGATION AFTER TRANSFORMATION AS WELL
C
       DO 150 I=1,LSAM
           CSAM(I) = CONJG(CSAM(I))
  150  CONTINUE
       LHSAM=LSAM/2
C
C      MULTIPLY POS FREQ COMPONENTS BY J AS IN HILBERT TRANS
C      DEFINED IN BRACEWELL.  POSTIVE FREQ COMP STARTS FROM
C      0 TO LSAM/2
C
       DO 200 I= 1, LHSAM
       CSAM(I) = CSAM(I)*(0.0 ,1.0)
   200 CONTINUE
C
C      MULTIPLY NEG FREQ COMPONENTS BY -J AS IN HILBERT TRANS
C      NEG FREQ COMPONENT STARTS FORM LSAM TO LSAM/2 +1
       LHSAM1=LSAM/2 + 2
       DO 300 I=LHSAM1,LSAM
       CSAM(I) = CSAM(I)*(0.0,-1.0)
   300 CONTINUE
C
C      PERFORM INVERSE FORUIER TRANSFORM
C
       CALL FFTCC(CSAM,LSAM,IWK,WK)
C0     WRITE(6,1000)(CSAM(I),I=1,LSAM)
C1000  FORMAT('1AFTER HILBERT TRANSFORM WITH FFT',///,(4F15.5))
       ALSAM = LSAM
       SCALE = 1.0/ALSAM
       DO 400 I=1,LSAM
   400 SAM(I) = REAL(CSAM(I)) * SCALE
       RETURN
       END


C      ----------------------------------------------------
C      INTERPOLATION ROUTINE. PERFORM INTERPOLATIONN BY
C      CONVOLVING THE SAMPLE WITH THE TIME DOMAIN INTERPOLATION
C      FUNCTION
C      SAM     INPUT SAMPLED FUNCTION DIMENSION LSAMEX.
C              NON SAMPLE POSITIONS ARE ALL ZEROS.
C      FARRAY  INTERPOLATION FUNCTION DIMENSION LF
C      RESULT  OUTPUT ARRAY DIMENSION LSAMEX
C      ----------------------------------------------------
       SUBROUTINE AINTER(LSAMEX,SAM,LF,FARRAY,RESULT)
C      ------------------------------------------
C
       DIMENSION SAM(LSAMEX),FARRAY(LF),RESULT(LSAMEX)
       AM1F = LF/2 +1
       DO 100 I=1,LF
```

```
      AI=I
  100 FARRAY(I) = W1((AI-AM1F)/8.0)
C
C     CALL EXTCON TO PERFORM CONVOLUTION WITH EXTENDED LENGTH
C     TO PRESERVE THE LENGTH OF SAMPLES.
C
      CALL EXTCON(LSAMEX,SAM,LF,FARRAY,RESULT)
      SCALE = 3./16.
      DO 200 I=1,LSAMEX
      SAM(I) = SAM(I) * SCALE
  200 CONTINUE
      RETURN
      END


C     --------------------------------------------------
C     NORMALISATION ROUTINE. NORMALISE THE NOISE SEQUENCE TO
C     UNIT VARIANCE.
C     SAM      INPUT NOISE SEQUENCE DIMENSIONED LSAM
C     --------------------------------------------------
      SUBROUTINE ANMLS(LSAM,SAM)
C     ------------------------
C
      DIMENSION SAM(LSAM)
      CALL VARMEN(LSAM,SAM,AMEAN,VAR)
      STDEVA = SQRT(VAR)
      DO 100 I=1,LSAM
  100 SAM(I)=SAM(I)/STDEVA
      RETURN
      END


C     --------------------------------------------------
C     READ YES NO. READS A YES OR NO ANSWER FORM THE TERMINAL.
C     REJECTS INPUTS OTHER THAN YES OR NO.
C     ANS      VALUE TO BE RETURNED.
C     --------------------------------------------------
      FUNCTION READYN(ANS)
C     -------------------
C
      REAL N
      DATA N,Y/'N','Y'/
  100 READ(19,1000)ANS
 1000 FORMAT(A1)
      IF (ANS .EQ. N) GOTO 200
      IF (ANS .EQ. Y) GOTO 200
      WRITE(20,1010)
 1010 FORMAT('INVALID ANSWER, Y OR N ONLY')
      GOTO 100
  200 READYN = ANS
      RETURN
      END


C     --------------------------------------------------
C     LAGRANGE FUNCTION AL
C     XI       VECTOR OF X POSITIONS DIMENSIONED N
C     I        THE ITH LAGRANGE FUNCTION
C     X        THE FUNCITON ARGUEMENT
C     --------------------------------------------------
```

```
      FUNCTION AL(XI,N,I,X)
C     -------------------
C     LAGRANGE FUNCTION AL
      DIMENSION XI(N)
      PROD = 1.0
      DO 1000 J=1,N
        IF(J .EQ. I) GOTO 1100
        PROD  = PROD * (X - XI(J))
 1100   CONTINUE
 1000 CONTINUE
      AL     = PROD
      RETURN
      END


C     --------------------------------------------------------
C     LAGRANGE INTERPOLATION FUNCTION FOR VAN VLECK CORRECTION
C     THE YI'S ARE DECLEARED AS DATA ELEMENTS
C     --------------------------------------------------------
      FUNCTION ALAGR(X)
C     -----------------
      REAL*4 Y(9) /0.3020,0.4045,0.5090,0.6155,0.7252,0.8394,
     C              0.9923,1.065,1.2510/
      REAL*4 XI(9) /0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9,0.94,1.0/
      IF (X .GT. 0.3) GOTO 1000
        ALAGR = X
      GOTO 9999
 1000 SUM = 0.0
      N = 9
      DO 1200 I = 1,9
        SUM = SUM + AL(XI,N,I,X)/AL(XI,N,I,XI(I)) * Y(I)
 1200 CONTINUE
      ALAGR = SUM
 9999 RETURN
      END
C     ------------------------------------------------------
C     VAN VLECK CORRECTION FUNCTION
C     PERFORMS VAN VLECK CORRECTION FOR THE CORRELATION FCN
C     LENGTH  THE NUMBER OF SAMPLES CORRELATED
C     R       THE CORRELATION FUCNTION DIMENSIONED LARC
C     ------------------------------------------------------
      SUBROUTINE CORREC(LENGTH,LARC,R)
C     -----------------------------
      DIMENSION R(LARC)
C
      ALENG = LENGTH
      ALARC = LARC
      CORPK = (ALENG - ALARC) * 0.5566
      DO 1000 I = 1,LARC
      TEMP = ABS(R(I)/CORPK)
      IF (TEMP .LT. 0.3) GOTO 1100
      WRITE(20,500)TEMP
  500 FORMAT(' TEMP = ',G15.5)
      R(I) = R(I) * TEMP/ALAGR(TEMP)
 1100 CONTINUE
 1000 CONTINUE
      RETURN
      END
```

```
C-----------------------------------------------------------
C
C     MAIN PROGRAM SIMUL
C
C-----------------------------------------------------------
      COMPLEX CSAM
      REAL N
      DIMENSION RXIP1(8500), RXIP2(8500), SIG1(8500), SIG2(8500),
     C SIG2HT(8500), WK2(1000), TITLE(90), COR(600),
     C ARC(600), R(600), R1(600),CSAM(8500),IWK(1000),WK(1000)
      DATA N,Y/'N','Y'/
C
C     SET UP WRITE FILES WITH MTS CALLS
C
      CALL FTNCMD('ASSIGN 19=*MSOURCE*',19)
      CALL FTNCMD('ASSIGN 20=*MSINK*',17)
C
C     DEFINE LENGTHS OF ARRAYS FOR CALLING VARIOUS ROUTNES
C
      LIWK = 600
C
C     READ IN PARAMETERS INTERACTIVELY
C
      WRITE(6,1103)
 1103 FORMAT('1DIGITAL CROSS CORRELATOR SIMULATION VERSION 2.0',//)
      WRITE(20,1113)
 1113 FORMAT('INPUT THE STRING OF CHARACTERS FOR TITLE. 80 CHAR')
      READ(19,1114)TITLE
 1114 FORMAT(100A1)
      WRITE(20,1115)TITLE
      WRITE(6,1116)TITLE
 1115 FORMAT('1',100A1)
 1116 FORMAT(' ',100A1)
 2500 WRITE(20,1100)
 1100 FORMAT('ENTER NO OF CORRELATOR CHANNELS IN IDEAL SIMULATION,',
     C' ODD ONLY',/,'PUT IN A DECIMAL PT. PLEASE')
      READ(19,1101)ALARC
      LARC=ALARC
 1101 FORMAT(G15.5)
      WRITE(20,1102)LARC
      WRITE(6,1102)LARC
 1102 FORMAT(//,'NO OF CORRELATOR CHANNELS=',I8,//)
      WRITE(20,1104)
 1104 FORMAT('LENGTH OF RUN?',/,'PUT IN A DECIMAL PT.')
      READ(19,1101)ALENG
      LENGTH=ALENG
      WRITE(20,1105)LENGTH
      WRITE(6,1105)LENGTH
 1105 FORMAT('LENGTH OF RUN =',I8,//)
      WRITE(20,1110)
 1110 FORMAT('S/N RATIO?  PUT IN A DECIMAL PT PLEASE.')
      READ(19,1101)SN
      WRITE(20,1111)SN
      WRITE(6,1111)SN
 1111 FORMAT('S/N RATIO =',G15.5,//)
      WRITE(20,1106)
 1106 FORMAT('WANT QUANTISATION? Y OR N')
      Q=READYN(Q)
      WRITE(20,1107)Q
      WRITE(6,1107)Q
```

```
 1107 FORMAT('QUANTISATION = ',A1)
      WRITE(20,1108)
 1108 FORMAT('WANT NUMERICAL VALUES OF GRAPH PRINT OUT? Y OR N')
      ANUM=READYN(ANUM)
      WRITE(20,1109)
 1109 FORMAT('WANT CORRELATION OF JUST RECEIVER NOISE? Y OR N')
      CN=READYN(CN)
C
C     READ IN THE CORRELATOR INPUT RANDOM NUMBERS
C
      FIND(11'1000)
      READ(11)(RXIP1(I),I=1,LENGTH)
      FIND(12'1000)
      READ(12)(RXIP2(I),I=1,LENGTH)
      IF (CN .EQ. N) GOTO 2000
C
C     CORRELATION OF RECEIVER NOISE ALONE
C
      CALL XCOR(LENGTH,RXIP1,RXIP2,LARC,R)
      IF (ANUM .EQ. N) GOTO 2120
      WRITE(6,1000)(R(I),I=1,LARC)
 1000 FORMAT('1CROSS CORRELATION OF RECEIVER NOISE',//,
     C(5F15.7))
 2120 CONTINUE
      WRITE(6,1115)TITLE
      WRITE(6,1011)
 1011 FORMAT(' CORRELATION OF RECEIVER NOISE')
      WRITE(6,1012)LARC,LENGTH,SN,Q
 1012 FORMAT('NO OF CORRELATOR CH =',I4,'        LENGTH OF RUN =',
     C I5,'        S/N RATIO = ',G13.5,'      QUANTISATION = ',A1,
     C '     +=REAL   *=QUAD')
      CALL PLOT(LARC,R,R)
      WRITE(10)(R(I),I=1,LARC)
 2000 CONTINUE
C
C     INJECT SIGNAL INTO RECEIVER NOISE
C     SN IS SIGNAL TO NOISE RATIO
C
      WRITE(20,1021)
 1021 FORMAT('DO YOU WANT TO GO AHEAD? IT COSTS YOU $'S')
      GOON = READYN(GOON)
      IF (GOON .EQ. N) GOTO 2400
C
C     READ IN THE SIGNALS TO BE CORRELATED
C
      FIND(13'1000)
      READ(13)(SIG1(I),I=1,LENGTH)
      FIND(14'1000)
      READ(14)(SIG2(I),I=1,LENGTH)
      FIND(15'1000)
      READ(15)(SIG2HT(I),I=1,LENGTH)
      DO 300 I=1,LENGTH
      RXIP1(I) = RXIP1(I) + SIG1(I)*SN
      RXIP2(I) = RXIP2(I) + SIG2(I)*SN
  300 CONTINUE
      CALL ANMLS(LENGTH,RXIP1)
      CALL ANMLS(LENGTH,RXIP2)
      IF (Q .EQ. N) GOTO 2100
      CALL QTN3(LENGTH,RXIP1,0.6,-0.6)
      CALL QTN3(LENGTH,RXIP2,0.6,-0.6)
```

```
 2100 CONTINUE
      CALL XCOR(LENGTH,RXIP2,RXIP1,LARC,ARC)
      IF (Q .EQ. N) GOTO 2105
      CALL CORREC(LENGTH,LARC,ARC)
C     CALL CORREC TO PERFORM VAN VLECK CORRECTION
 2105 CONTINUE
      IF (ANUM .EQ. N) GOTO 2130
      WRITE(6,1010)(ARC(I),I=1,LARC)
 1010 FORMAT('1REAL CHANNEL CORRELATOR OUTPUT',//,(5F15.7))
 2130 CONTINUE
      WRITE(10)(ARC(I),I=1,LARC)
C
C     REREAD FILE TO ERESTABLISH RXIP2
C
      FIND(12'1000)
      READ(12)(RXIP2(I),I=1,LENGTH)
      DO 400 I=1,LENGTH
  400 RXIP2(I) = RXIP2(I) + SIG2HT(I)*SN
      CALL ANMLS(LENGTH,RXIP2)
      IF (Q .EQ. N) GOTO 2110
      CALL QTN3(LENGTH,RXIP2,0.6,-0.6)
 2110 CONTINUE
      CALL XCOR(LENGTH,RXIP2,RXIP1,LARC,R1)
      IF (Q .EQ. N) GOTO 2115
      CALL CORREC(LENGTH, LARC, R1)
C     CALL CORREC TO PERFORM VAN VLECK CORRECTION
 2115 CONTINUE
      IF (ANUM .EQ. N) GOTO 2170
      WRITE(6,1020)(R1(I),I=1,LARC)
 1020 FORMAT('1CORRELATOR OP QUAD CHANNEL',//,(5F15.7))
 2170 CONTINUE
      WRITE(6,1115)TITLE
      WRITE(6,1080)
 1080 FORMAT(' CORRELATOR OUTPUT')
      WRITE(6,1012)LARC,LENGTH,SN,Q
      CALL PLOT(LARC,R1,ARC)
      WRITE(10)(R1(I),I=1,LARC)
C     CALL ANMLS(LARC,ARC)
C
C     PERFORM HILBERT TRANSFORM ON THE CORRELATION OUTPUT
C
      WRITE(20,1081)
 1081 FORMAT('DO YOU WANT HILBERT TRANSFORM OF CORRELATION',
     C ' FUNCTION BY FFT? Y OR N')
      GOON = READYN(GOON)
      IF (GOON .EQ. N) GOTO 2175
      DO 100 I=1,LARC
  100 R1(I) = ARC(I)
      CALL HTFFT(LARC-1,R1,CSAM,LIWK,IWK,WK)
C     DO 110 I=1,LARC
C 110 R1(I) = -R1(I)
C     CALL ANMLS(LARC,R1)
      IF ( ANUM .EQ. N) GOTO 2150
      WRITE(6,1091)(R1(I),I=1,LARC)
 1091 FORMAT('1HILBERT TRANSFORM OF REAL CHANNEL BY FFT',//,
     C (5F15.7))
 2150 CONTINUE
      WRITE(6,1115)TITLE
      WRITE(6,1090)
 1090 FORMAT(' HILBERT TRANSFORM OF REAL CHANNEL BY FFT')
```

```
      WRITE(6,1012)LARC,LENGTH,SN,Q
      CALL PLOT(LARC,R1,ARC)
 2175 CONTINUE
      WRITE(10)(R1(I),I=1,LARC)
      WRITE(20,1093)
 1093 FORMAT('DO YOU WANT HILBERT TRANSFORM OF CORRELATION',
     C ' FUNCTION BY CONVOLUTION? Y OR N')
      GOON = READYN(GOON)
      IF (GOON .EQ.N) GOTO 2190
      DO 200 I=1,LARC
  200 R1(I)=ARC(I)
      CALL HTCON((LARC-1)*3,R1,LARC-1,WK,WK2)
C     DO 210 I=1,LARC
C 210 R1(I) = -R1(I)
      IF (ANUM .EQ. N) GOTO 2160
      WRITE(6,1092)(R1(I),I=1,LARC)
 1092 FORMAT('1HILBERT TRANSFORM OF REAL CHANNEL BY CONVOLUTION',
     C //,(5F15.7))
 2160 CONTINUE
      WRITE(6,1115)TITLE
      WRITE(6,1070)
 1070 FORMAT(' HILBERT TRANSFORM OF REAL CHANNEL BY CONVOLUTION')
      WRITE(6,1012)LARC,LENGTH,SN,Q
      CALL PLOT(LARC,R1,ARC)
 2190 CONTINUE
      WRITE(10)(R1(I),I=1,LARC)
      WRITE(20,1210)
 1210 FORMAT('ENTER THE NO OF CORRELATOR CHANNELS IN REAL LIFE ',
     C 'CORRELATOR SIMULATION ','EVEN ONLY. PUT IN A DECIMAL PT ')
      READ(19,1101)ALCOR
      LCOR=ALCOR
      WRITE(20,1220)LCOR
 1220 FORMAT(/,'NO OF CORRELATOR CHANNELS TO BE SIMULATED =',I4)
      LCOREX = 4*LCOR - 3
      DO 850 I=1,LCOREX
  850 COR(I) = 0.0
      DO 800 I=1,LCOREX,4
  800 COR(I) = ARC((LARC-LCOREX)/2 + I)
      WRITE(6,1115)TITLE
      WRITE(6,1225)
 1225 FORMAT(' SAMPLES POINTS AVAILABLE IN REAL LIFE CORRELATOR')
      LTHQTR=LENGTH/4
      WRITE(6,1012)LCOR,LTHQTR,SN,Q
      CALL PLOT(LCOREX,COR,COR)
      DO 500 I=1,LCOREX
  500 R(I) =COR(I)
C
C     CALL INTERPOLATION ROUTINE
C
      CALL AINTER(3*LCOREX,R,LCOREX,WK,WK2)
      DO 600 I=1,LCOREX
  600 R1(I)= COR(I)
C
C     CALL HTCON FOR HILBERT TRANSFORMATION AND INTERPOLATION
C
      CALL HTCON(3*LCOREX,R1,LCOREX,WK,WK2)
C     CALL ANMLS(LCOREX,R)
C     CALL ANMLS(LCOREX,R1)
      DO 1223 I=1,LCOREX
      R(I) = R(I) * 4.
```

```
          R1(I) = R1(I) * 4.
     1223 CONTINUE
          WRITE(10)(R(I),I=1,LCOREX)
          IF (ANUM .EQ. N) GOTO 2200
          WRITE(6,1221)(R(I),I=1,LCOREX)
     1221 FORMAT('1REAL CHANNEL OUTPUT. RESULT OF REAL LIFE CORRELATOR'
        C ,' SIMULATION',//,(5F15.7))
          WRITE(6,1222)(R(I),I=1,LCOREX)
     1222 FORMAT('1QUAD CHANNEL OUTPUT. RESULT OF REAL LIFE CORRELATOR'
        C ,' SIMULATION',//,(5F15.7))
     2200 CONTINUE
          WRITE(6,1115)TITLE
          WRITE(6,1230)
     1230 FORMAT(' RESULT OF INTERPOLATION WITH EVERY FORTH SAMPLE',
        C', AS IN REAL LIFE CORRELATOR')
          WRITE(6,1012)LCOR,LTHOTR,SN,Q
          CALL PLOT(LCOREX,R1,R)
          WRITE(10)(R1(I),I=1,LCOREX)
     2400 WRITE(20,1300)
     1300 FORMAT('DO YOU WANT ANOTHER RUN WITH DIFFERENT S/N OR Q?')
          GOON=READYN(GOON)
          IF (GOON .EQ. Y) GOTO 2500
          WRITE(20,1310)
     1310 FORMAT('DO YOU WANT ANOTHER RUN WITH DIFFERRENT NO OF CHANNELS ',
        C/,'FOR REAL LIFE CORRELATOR?  Y OR N')
          GOON=READYN(GOON)
          IF (GOON .EQ. Y ) GOTO 2190
     9999 STOP
          END
END OF FILE
```