APPLICATIONS OF SIGNAL PROCESSING TECHNIQUES IN MICROPROCESSOR BASED INSTRUMENTATION

by

MADHUKAR DUGGIRALA

B.S., College of Engineering, Guindy, Madras, India, 1982

A MASTER'S THESIS

Submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Electrical Engineering

KANSAS STATE UNIVERSITY

Manhattan, Kansas

1984

Approved by

Major Professor

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Signal processing techniques have definite applications in the area of microprocessor or computer-based instrumentation. A new approach has been evolved wherein a microprocessor-based system can accomplish data acquisition with subsequent processing by a computer utilising suitable processing algorithms. The algorithms used depend on what exactly needs to be measured or determined - a specific example considered here being the measurement of phase. Such a system possesses the inherent advantages of speed, automation, flexibility in type of algorithm used for the specific problem, and enormous computing and processing capabilities coupled with all the advantages which discrete time signal processing can offer.

A microprocessor based phasemeter has been developed which acquires data samples of the original signals and sends them to a computer for subsequent processing by suitable algorithms. These algorithms are based on Adaptive Filtering, FFT, Correlation and Numerical methods. The system was implemented on a North Star microcomputer in conjunction with an MC 6802 interface and HP Multiprogrammer; the whole forming an automated test system. The MC 6802 interface serves as a communication link between the North Star (NS) and HP Multiprogrammer (HPM). The sampling of the signals is initiated by a command from the NS software which invokes an assembly routine in the MC 6802 interface and uses the High Speed Analog to Digital converter (ADC) card on the HPM. These samples are then sent serially to the NS and subsequent

processing takes place by the algorithms stated above to determine the phase difference. The HP 9845B Desktop computer was used to study theoretical simulation results and for the development of algorithms.

Conventional phasemeters operate on the principle of measuring the time difference between the zero crossings of the two signals by "squaring up" the signals. Complex and sophisticated hardware consisting of amplifiers, precision reference generators, filters and networks are normally used. Also, manual controls and adjustments are required for making measurements. The automated microprocessor based system, however, employs more software and less sophisticated hardware. Though this system may not boast of the same accuracy obtainable with the dedicated and sophisticated phasemeter, it still has certain advantages and is capable of further development and improvement. The relative merits and demerits are discussed later.

Other applications can also be found for the system discussed above, examples being spectral estimation (PDS), frequency domain analyses, filtering, combating noise and determination of RMS value etc. Thus this project represents an initial step in a novel direction and the author is optimistic that further development and sophistication is possible in future.

CHAPTER 2

SOME PRELIMINARY CONSIDERATIONS

The specific application of signal processing techniques in phase measurement has been selected. There are some questions to be answered (before we proceed further) such as (a) Why measure phase? (b) What are the conventional methods of measurement? (c) How are the proposed methods different and what advantages do they have over the conventional methods.

The measurement of phase is accomplished by a phasemeter. Such measurements find applications in impedance meters, the measurement of amplifier phase shift, the transfer functions of networks, the measurement of group delay, the measurement of distance and directivity, and the testing and adjusting of filter networks.

The principle of phase measurement used in modern phase instrumentation is illustrated in fig. 2.1. through fig. 2.4. If "t" is the time interval between analogous zero crossings, and "T" is the wave period, then phase angle P=360*(t/T) degrees. The signals are then squared up as shown in fig. 2.2. Most phasemeters use the squared off waves to generate the waveshape shown in fig. 2.3. The Dranetz 305 is one of a series of sophisticated phasemeters available. This phasemeter, after generating the waveform in fig. 2.3. standardizes the waveform amplitude and then filters the waveform to yield the DC value.

The filtering in the simplest case is accomplished by a low pass filter ie. an RC network. The time constant of this filter must be large enough to provide an accurate measurement.

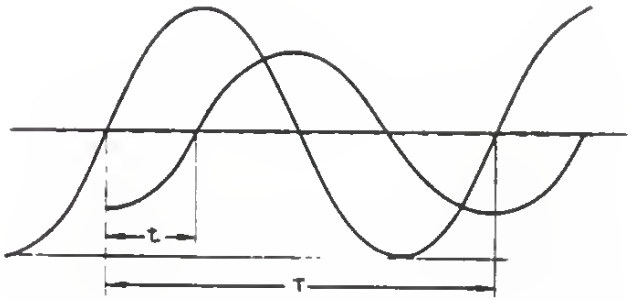


Fig.2.1. Signals with a certain phase difference.

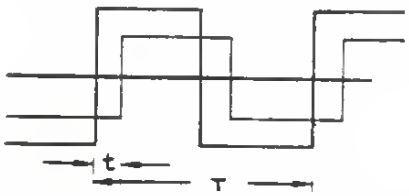


Fig.2.2. Clipping or squaring up the signals.

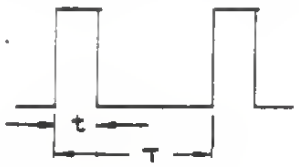


Fig.2.3. Triggering on and off at the zero crossings.

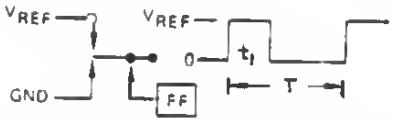


Fig.2.4. Standardizing the waveform amplitude.

Fig.2.1...2.4. Principle of phase measurement.

Vref, and all DC amplifiers following this circuit and other

circuitry must be stable and accurate with time and temperature. The speed of response is limited by the time constant and hence in practice it is made variable by automatic switching of filters with different time constants.

The presence of noise, interference, high harmonic content or various forms of signal modulation may impair accuracy. Also the input signals may vary appreciably in magnitude. This variation is counteracted by switching attenuators. Thus in the Dranetz system it is found that a great degree of complexity and sophistication in circuitry is used to combat all the above mentioned problems.

Having considered the conventional method of phase measurement attention is now focussed on the proposed techniques.Here the analysis of signals is carried out in the discrete time (DT) domain where we work with "samples" of the original signals as shown in fig. 2.5.

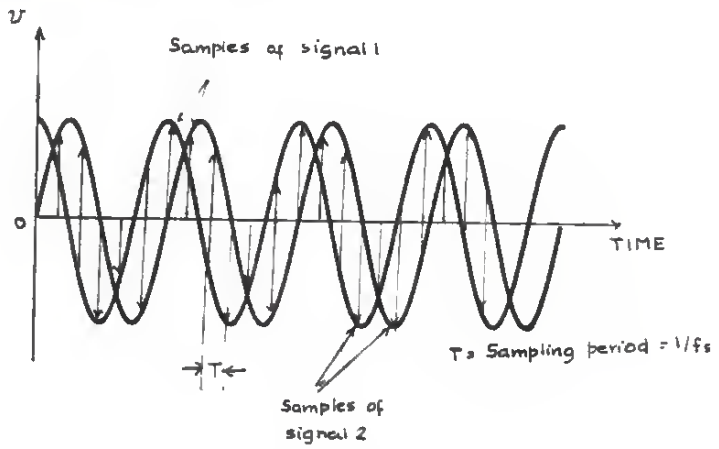


Fig.2.5. Samples of the signals in the DT domain.

It is necessary according to the Nyquist criterion that if

all information in the original signals has to be preserved then the sampling frequency "f_s" must be greater than the frequency of the signals "f_o" that is $f_s > 2*f_o$. Once the signals are characterized by the "samples" then the next step is to develop some algorithm to process these samples to obtain the required characteristic (for example the phase) of the original signals. These samples may be obtained by using the Analog to Digital Converter (ADC).

The ADC produces a digital output proportional to the analog input. The parameters of importance here are the conversion time of the ADC (how fast it can sample) and its resolution (how well or accurately it represents the analog quantity in digital form). Some ADCs that are available have 8 or 12 bits resolution and are capable of moderate sampling rates. The "Flash" type of ADC is available with a high sampling rate (10 Mhz.) and 8 bits resolution. A more detailed discussion on ADCs is found later.

It is necessary at this stage to emphasize the fact that once the signals are "sampled" or "digitized" and the resulting samples are stored in memory then they are absolutely unaffected by any time or temperature changes. Hence the only caution that must be exercised is prior to and during digitizing when the effects of noise, temperature changes, offsets and all the problems associated with hardware exist (as in the conventional phase meter). Thus the single greatest advantage is due to the use of software algorithms which are unaffected by the problems that affect hardware. A computer can then obtain the samples

stored in memory and then process them with the pertinent algorithm and obtain the required result. All the advantages of software such as flexibility, use of statistical methods, computing capabilities become apparent.

The techniques discussed use Adaptive Filtering which uses digital filters (in contrast to the analog RC filters in the conventional phasemeter). Here the coefficients of the filters can be programmed and changes can be rapidly made without changing circuit components. Performance is not affected by wear and aging and is more reliable. The response characteristics can be easily changed by manipulating the coefficients which in turn leads to more flexibility. However, the finite resolution of the ADC, finite precision or word length of the computer, truncation and roundoff errors and overflow/underflow problems are associated with software processing. Nevertheless, by proper hardware/software tradeoffs, it is possible to obtain a suitable operating system.

Thus with these preliminary considerations we now proceed to describe the proposed algorithms.

CHAPTER 3

THE ADAPTIVE FILTERING ALGORITHM

3.1 INTRODUCTION

This method uses a signal processing technique familiarly known as the Widrow's LMS (least mean square) algorithm in the cancellation mode. The schematic is shown in fig 3.1. The two signals y(k) and x(k) form the primary and reference inputs to the adaptive filter configuration. Here y(k) and x(k) are derived by sampling the CT (continuous time) signals x(t) and y(t) and "k" is the discrete time index which represents the sample of the original signals at time t=k. The error e(k) that is generated by comparing g(k) and y(k) is used to change the weights of the discrete time (DT) digital filter W such that the square of the mean error e(k) is a minimum. Thus as the processing continues the error dynamically changes the weighting coefficients of the filter so that the error reduces in magnitude. After a sufficient number of processing steps the error tends to a small value, the weights of the filter are then used to calculate the phase difference between y(k) and x(k). Here y(k) is assumed to lead x(k).

3.2 THEORETICAL ASPECTS

Consider the adaptive scheme in fig. 3.1. It is seen that y(k) and x(k) form the primary and reference inputs and W_k is a M+1 weight filter. The error e(k) obtained by comparing y(k) with the filter output g(k) is used to update the filter weights or coefficients. This process is carried on until the error reduces

to a small value.

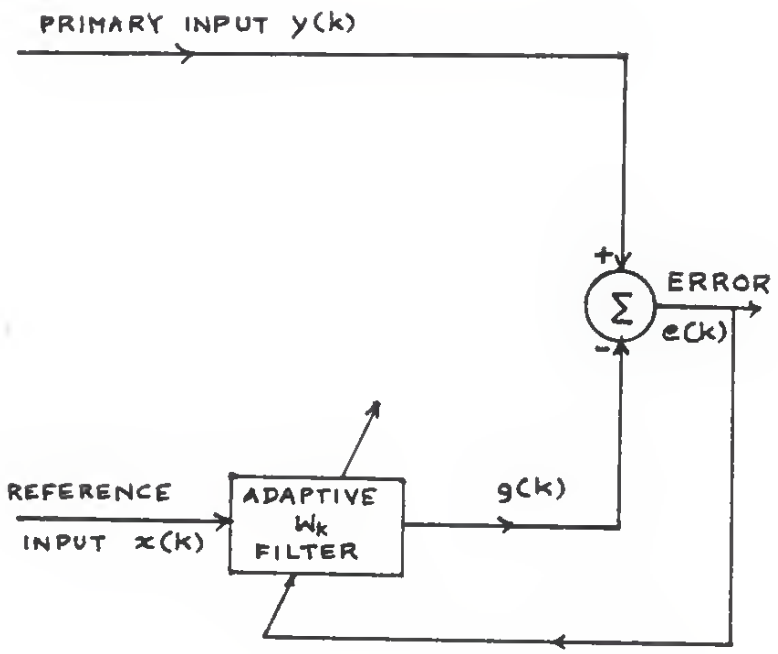


Fig.3.1. Adaptive filtering scheme in the cancellation mode.

Hence we have the following, in vector form:

$W' = (w_o\ w_1\ w_2\ \ldots\ w_M)$ considering M+1 weights.

$X_k' = (x(k)\ x(k-1)\ x(k-2)\ \ldots\ x(k-M))$

where $x(k-1)\ldots x(k-M)$ represent the M past values of $x(k)$.

$x(k) = \sin(w_o kT)$ and $y(k) = \sin(w_o kT+P)$

Now $g(k)= W'X_k = w_o x(k)+w_1 x(k-1)+\ldots w_M x(k-M)$

$e(k) = y(k)-g(k)$

When the error e(k) tends to a small value as a result of the adaptive process then g(k) tends to cancel y(k) at the summing device. Hence g(k) "looks like" y(k). At this stage taking the Z transform to obtain the filter transfer function we have the following:

$G(z) = X(z)[w_o+w_1 z^{-1}+\ldots+w_M z^{-M}]$

The transfer function H(z) is then

$H(z)=G(z)/X(z)$ and since $z=e^{jw_o T}$ where $w_o=2\pi f_o$

f_o being the frequency of the signals and T being the sampling period.

Hence the value of transfer function at the frequency of the signals is given by

$H(w_o) = G(w_o)/X(w_o) = w_o+w_1 e^{-jw_o T}+\ldots+w_M e^{-jMw_o T}$

The phase transfer function $B(w_o)=\arctan[\ ImH(w_o)/ReH(w_o)]$

$ImH(w_o)=[w_1 \sin w_o T+w_2 \sin 2w_o T+\ldots+w_M \sin Mw_o T]$

$ReH(w_o)=[w_o+w_1 \cos w_o T+w_2 \cos 2w_o T+\ldots+w_M \cos Mw_o T]$

The value of B at w_o gives the required phase difference "P" between the two signals x(k) and y(k).

Thus from the above discussion it is apparent that after a

sufficient number of processing steps (or iterations) when the error decreases the phase difference "P" can be directly determined from the coefficients of the fiter W. It must be noted that in the above expression for B it is required to know the values of f_o and T for calculating the phase P.

3.3 SIMULATION AND VERIFICATION

As a first step, software development for the method was done on the North Star microcomputer system. The two signals x(k) and y(k) were generated as $x(k)=\sin(w_o kT)$ and $y(k)=\sin(w_o kT+P)$ where $w=2\pi f_o$, f_o being the frequency of the signals, T the sampling period, k the time index and P the phase difference in radians. The Widrow algorithm was implemented in software. The flow chart and BASIC program listing are found in Appendix 1a and 1b respectively.

The procedure followed is essentially iterative in nature and the weights are updated as follows for a M+1 weight filter.

$w_m(k+1)=w_m(k)+ve(k)x(k-m)$; $\quad 0<=m<=M+1$

where v is the convergence parameter and this parameter is dynamically updated as follows:

$v=\alpha/\sigma_x^2(k)$; $\quad \sigma_x^2(k)=\beta\sigma_x^2(k-1)+(1-\beta)x^2(k)$; $\quad \sigma_y^2(0)=0$

The parameter α must be chosen with care for proper convergence and the phase is calculated as P given by the formula for B in section 3.2.

The most important parameters that are involved in this method are the following:

a. the frequency of the signals (f_o)

b. the sampling frequency (f_s=1/T)

c. the convergence parameter (α)

d. the phase difference involved (P)

e. the number of iterations (K)

f. the ratio (f_s/f_o)

Some of the important results obtained after considerable experimentation with the parameters stated above are as follows.

1. The theoretical value of sampling frequency $f_s>2f_o$, but it was found that $f_s>=3f_o$ for proper working of the method.

2. For a 3 weight filter and with just 30 iterations convergence was established when the value of sampling frequency was equal to four times the signal frequency ie. $f_s=4f_o$ and the calculated phase agreed very closely with the actual phase (absolute error less than 0.2 degrees.)

3. However with a 16 weight filter convergence was easily obtained for a wide range of f_s/f_o.

4. One of the theoretical aspects pertaining to convergence is that a greater number of weights leads to faster convergence. Also the convergence time is also dependent on the convergence parameter as $\tau=1/\alpha=1/(1-\beta)$.

5. Also the faster the convergence the greater is the residual error. Too fast a convergence can overdrive the filter W causing large residual errors. (ie. magnitude of e(k)).

6. Hence the number of weights and the convergence parameter must be chosen carefully to obtain a small residual error and

hence a small error in the calculated phase.

7. The convergence also depends on the type of input.

8. The method works for any two zero mean signals.

9. Since the filter by itself cannot distinguish a phase shift of more than 90°, suitable modifications were made in the software which fixes the quadrant with respect to the sign of the numerator and denominator in the equation for phase as already shown. A phase of 360° could then be easily distinguished.

10. The above results were established assuming y(k) leads x(k).

Analysing the working of the method in the frequency domain, it can be said that the adaptive filter merely adjusts the amplitude and phase of all the components in its input signal x(k) to cancel those in y(k). This is a characteristic of the adaptive process.

3.4 DETERMINATION OF PHASE SIGN

In the above discussion it was assumed that y(t) leads x(t) by a phase difference P. When this is not known we resort to the following method to determine which is the leading or lagging waveform. If during the period x(t) is positive there occurs a negative to positive transition of y(t) then x(t) leads y(t) and if there occurs a positive to negative transition of y(t) then y(t) leads x(t).

However the sign is automatically reflected in the present software. If y(k) and x(k) are interchanged (assuming y(k) leads x(k)) then the calculated phase shows a value of 360°-P where P is the actual phase difference. Numerical methods presented later

can be used to determine to a moderate degree of accuracy the zero crossing from which the sign could be determined.

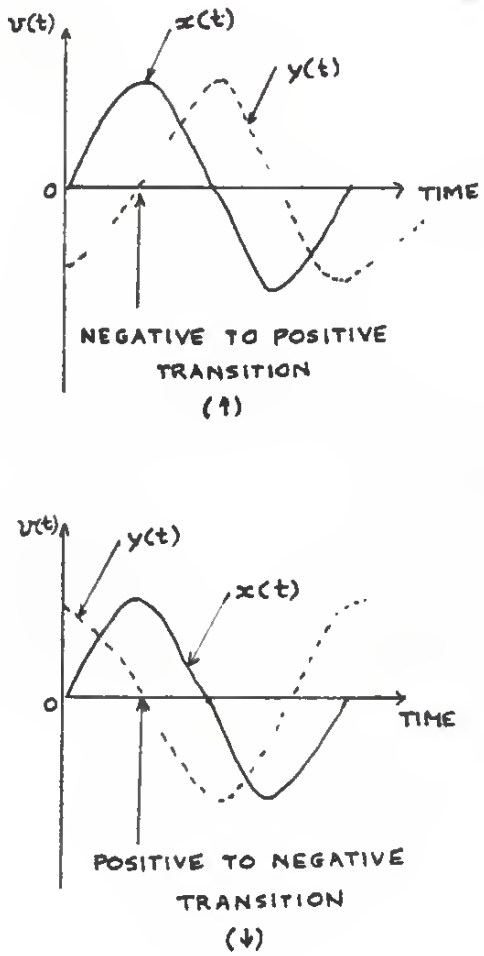


Fig.3.2. Determination of phase sign.

3.5 PRACTICAL ASPECTS AND SYSTEM IMPLEMENTATION

After having established the validity of the method in theory, it was extended to a practical situation. However, as is always the case , the transition from theory to practice is not smooth and many constraints had to be given due importance.

Before we discuss the practical system we must bear in mind the following aspects of the theoretical simulation carried out.

1. The samples generated are correct to 7 decimals.
2. f_o and $T=1/f_s$ are quite accurate since they are just values given by us.
3. No noise is included in the signals.
4. Phase shift is constant with time.

However in a practical system, the situation is a bit different. The finite resolution of the ADC imposes a constraint on the accuracy with which a sample is represented. For example with a 12 bit ADC the accuracy is limited to around 3 decimal places. The frequency of the signals must be determined by some external means and the sampling frequency must be accurately known. The signals may be derived from a function generator which has some noise in its output and may also have some voltage offsets. The phase of the waveform generated may drift with time. Hence all these must be considered in a practical system.

The basic configuration required for system implementation of the method is shown in fig.3.3.

The ADCs must be controlled to sample both the signals x(t) and y(t) simultaneously. These samples may then be stored in memory. Once stored in memory the samples can be accessed by a

computer or microcomputer for processing the samples with the adaptive algorithm. The CRT terminal acts as an Input/Output device to provide interaction between the user and the system.

There are different ways of realizing the above system but the primary efforts were directed to utilise the facilities available. A practical system using the North Star microcomputer in conjunction with the MC 6802 interface and HP Multiprogrammer (HPM) being used as an automated test system. A brief discussion of the HPM and the interface follows.(fig.3.3.a.)

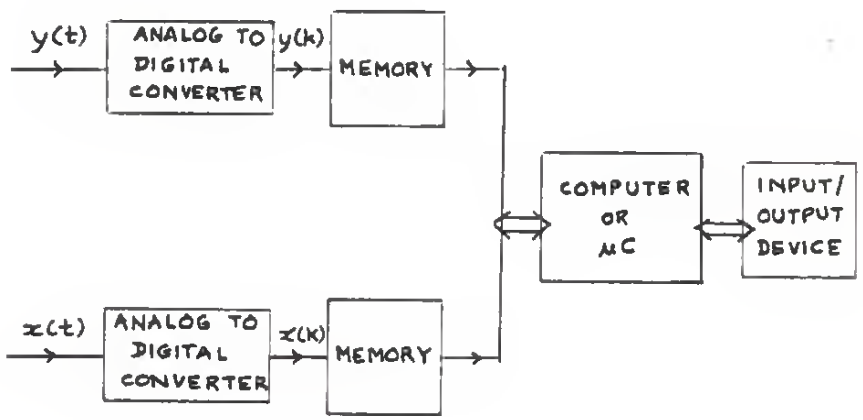


Fig.3.3. System implementation of the adaptive method.

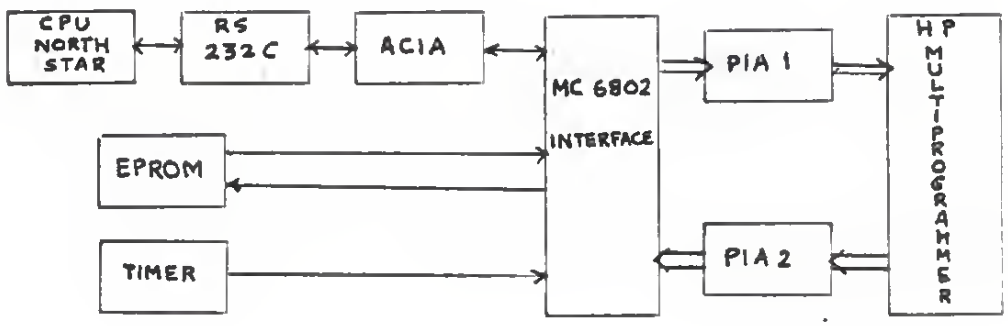


Fig.3.3.a. Schematic of the test system.

The HPM is a master control unit for bidirectional systems. It can be controlled using a computer. It contains many types of accessory cards which can be used depending on the requirement. For example the V/F card in the HPM can be programmed to make voltage or meaurements on the device under test (DUT). The HPM also contains a High Speed Analog to Digital Converter card (ADC). Each card can be accessed by addressing the particular slot in which it resides. The various communication protocols are provided by the following four types of words - control word, output data word, input address word and return data word.

The MC 6802 interface acts as a communication link between the North Star (NS) and the HPM. There exists an assembly routine in the interface EPROM that can address the ADC in the HPM, obtain the samples and store them in the memory available in the interface RAM (1K). The procedure is to first set-up the HPM in the input mode. Then the ADC card is accessed by the proper slot address. The conversions are initiated by a gate signal and the end of conversion is indicated by a flag. The routine makes use of the above procedure to obtain the samples.

The ADC card is capable of 22K conversions per second. However due to the inherent limitations posed by the software assembly routine the present sampling frequency is 9130 samples per second. The number of samples that can be taken is fixed at 512. The ADC has three operating ranges (±10V, ±1V, ±100mV). The interface accomplishes sampling of the signal and stores in memory. It then sends the data samples to the NS using the serial port. Thus the sampling is initiated by a command in the BASIC

program which invokes the assembly routine in the interface EPROM and the samples are then sent to the NS memory. The schematic of the interface and connections are shown in Appendix 2.

A look at fig. 3.3 shows that the two signals have to be sampled simultaneously requiring two ADCs. Since one ADC card is available on the HPM the following method was adopted to enable sampling of the two signals by one ADC and yet make it look like they were sampled simultaneously. The principle is illustrated in fig.3.4 and 3.5.

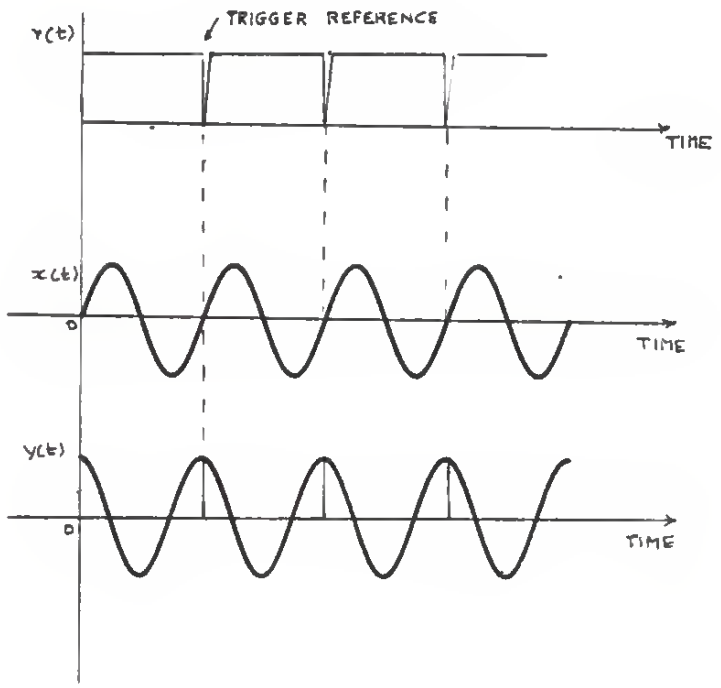


Fig.3.4. Generation of a trigger reference.

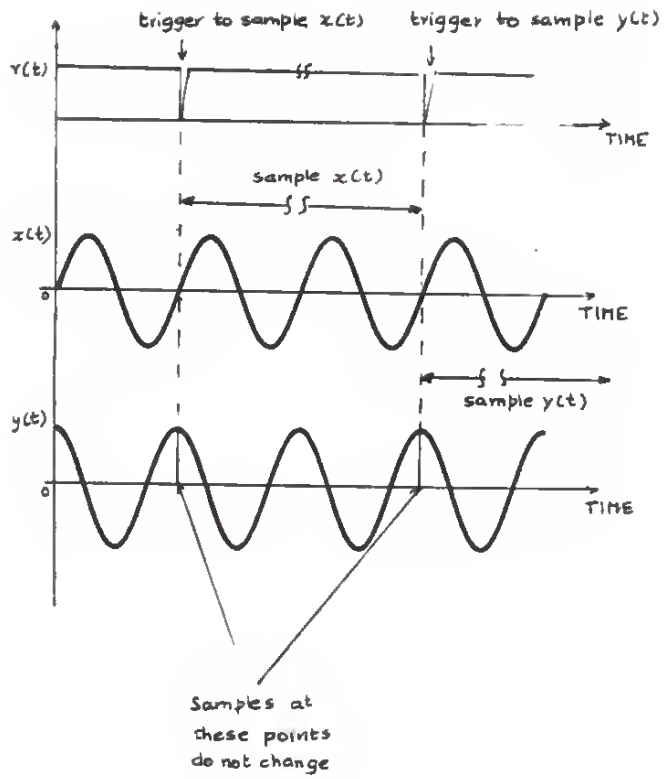


Fig.3.5. Use of trigger reference to sample x(t) and y(t).

The property of the two signals that is exploited in this method is that x(t) and y(t) continuous time signals and are periodic. Hence the signals by themselves do not change with time and with respect to themselves, a trigger reference may be used to initiate the sampling of the two signals.

In order to obtain the trigger reference a zero crossing detector was designed and built.(fig.3.6).

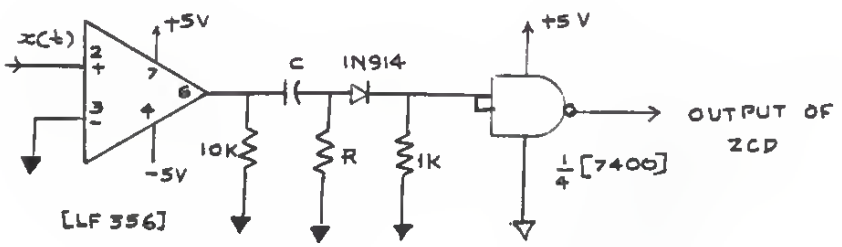


Fig.3.6. Circuit diagram of the zero crossing detector.

The output of the zero crossing detector (ZCD) was used to initiate the sampling of x(t) and y(t). In the assembly routine just before the sampling is actually initiated, a loop was interposed as shown below.

```
ADC assembly routine
.....
.....
BACK    LDAA CRB2
BPL BACK
LDAA DDRB2
.....
.....
.....
Sampling initiated
```

We shall now discuss the operation of the above loop. A glance at the schematic of the interface (Appendix 2) shows that the CB2 terminal (control register B, [pin 18]) of PIA 2 in the interface can be used as an input terminal to which the negative pulses of the ZCD are given. The PIA 2 is initialized such that in the control register CRB2 bit 7 is set by negative transitions (↓) on the input terminal. Thus the output of the ZCD (the transitions) triggers the sampling as the execution of the loop is terminated when bit 7 is set by a negative transition. The MPU read of data register B clears bit 7 and the execution of the sampling process is done.

However, this method was later found to have certain

disadvantages since it introduced some errors. The MPU clock cycle time was found to be 1.1 us since it operates at 920.8 KHz. There is an ambiguity involved since the trigger pulse could arrive at any instant with respect to the two instructions forming the loop. An analysis will reveal that there could be a maximum error of 4 clock cycles (4.4 us) which at a signal frequency of around 1000 Hz. gives rise to a phase error of around 2.5 degrees.

Hence in order to get around this problem an interrupt service routine was used in the assembly program and the properties of the Wait For Interrupt (WAI) were utilised. When this instruction is executed then the uP waits indefinitely until as interrupt occurs and then proceeds to the interrupt service routine without any delay after which the sampling is initiated. Thus the source of ambiguity and error were removed totally. The assembly listing is given in Appendix 3.

The signals were derived using a function generator. In order to know the frequency of the signals a digital frequency counter was used. The ADC was switched to operate in the ±10 V range keeping in view the effects of noise. It can be seen that if a sinewave has to be digitized properly then the amplitude of the signal must be adequate to cover the entire operating range of the ADC. This is discussed in a later chapter.

The signals x(t) and y(t) were derived using a simple RC circuit (fig.3.8). The phase shift of this circuit is given by $\arctan(1/2\pi f RC)$. This circuit also produces attenuation besides

the phase shift. Hence when y(t) is sampled the input is increased to a sufficient value so that the amplitude of y(t) covers the operating range of the ADC.

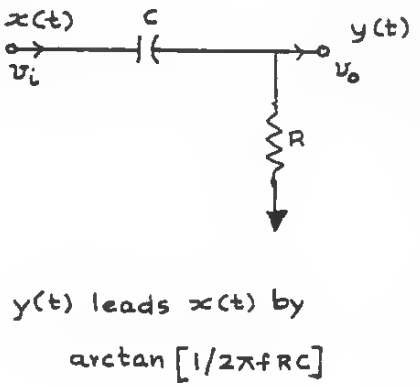
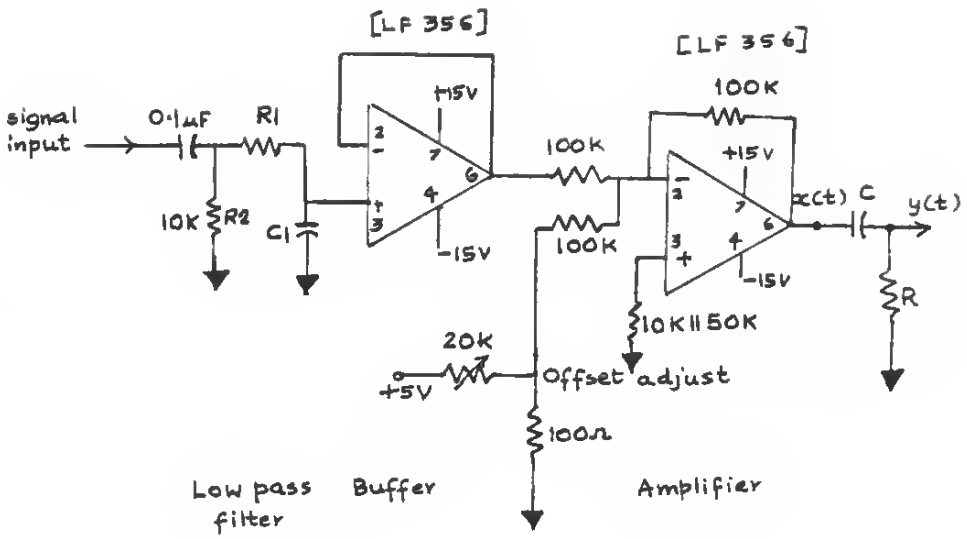


Fig.3.8. Use of RC circuit to generate x(t) and y(t).

Other considerations are that the HP3311 (function generator) produces a maximum sine output of ±5 V and hence an amplifier using an OPAMP was used. Also since the output of the HP3311 was noisy a low pass filter (LPF) was used at the input stage. Offset adjustment was provided for the amplifier, since the method requires that x(t) and y(t) be zero mean signals. The circuit diagram is shown in fig.3.9.



R1 and C1 chosen for desired cutoff; $f_1=1/(2\pi R1C1)$

Fig.3.9. Noise reduction and signal conditioning circuit.

Accomplishing all of the above enabled a working system to be developed which uses the adaptive algorithm. A block diagram of the system implementation is shown in fig. 3.10.

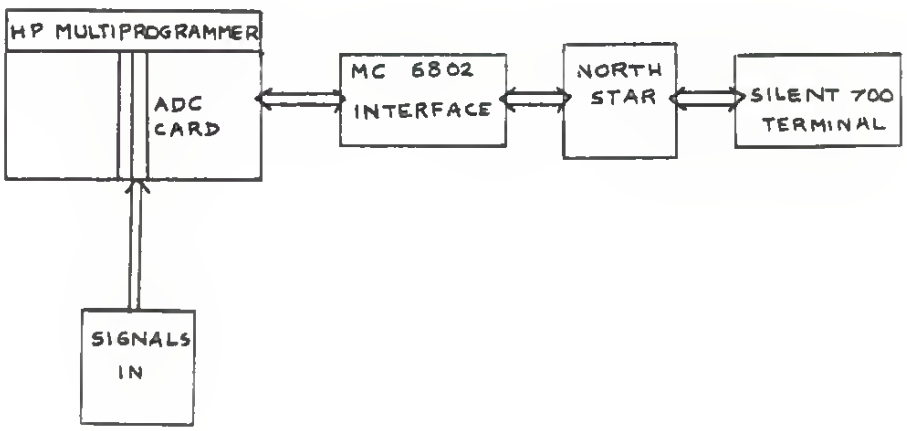


Fig.3.10. Block diagram of the test system.

3.6 WORKING OF THE METHOD AND THE TEST SYSTEM

A schematic of the test system is shown in fig. 3.11.

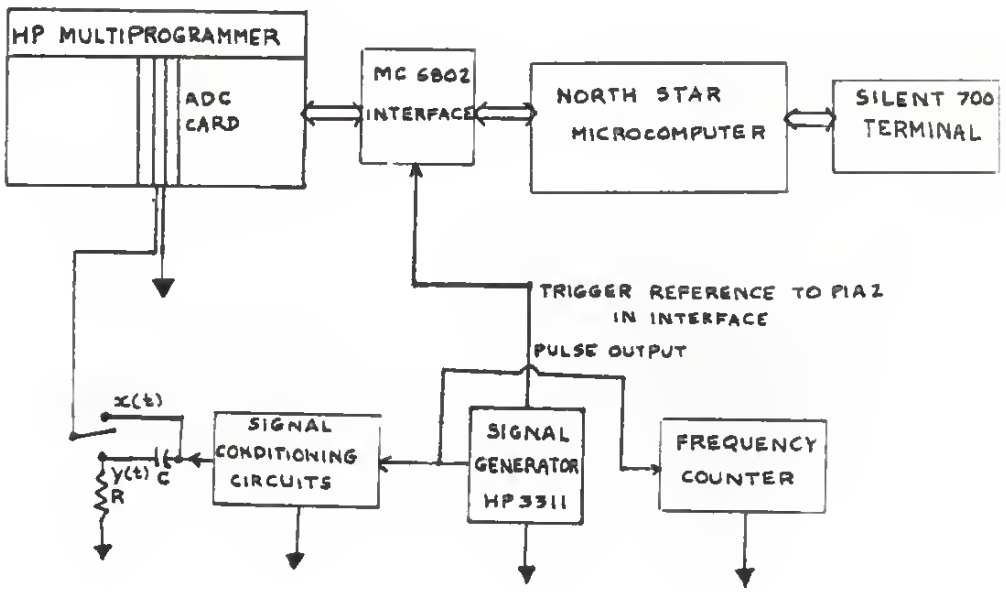


Fig.3.11. Configuration of the test system.

The Silent 700 terminal acts as the communicating device between the operator and the system. The entire operation is controlled by the software in BASIC. A command is sent via the serial port of the NS to the MC 6802 Interface . This consists of the character "A". The main assembly routine in the interface recognises this as a command to invoke the ADC routine and performs the sampling using this routine. The samples are then sent back via the serial port to the NS. The signals y(t) and x(t) are sampled successively. Before the processing starts the frequency of the signals indicated by the frequency is input to the NS by the use of the INPUT statement. The processing is then done and the results are displayed on the Silent terminal. The CRO was used to monitor the amplitudes of the signals to suit the range of the ADC.

A small modification that was carried out at a later stage was that although the ZCD was designed and used, the performance of the comparator was not up to the mark (probably due to noise) and hence a more accurate trigger reference was derived from the pulse output terminals of the HP3311. (this being generated internally in the HP3311).

After considerable experimentation, it was found that the ratio f_s/f_o was very significant in determining convergence. The sampling frequency being fixed at 9130 Hz., the input signal frequency was varied to study the effects of the ratio over convergence and the calculated value of phase. This was done by comparing the phase obtained with the actual phase obtained by the formula $\arctan(1/2\pi f_o RC)$ the values of R and C being

determined to some degree of accuracy by a digital resistance/capacitance meter.(Digibridge).

As discussed earlier in the section 3.2, the two parameters N (the number of weights), and the convergence parameter played a significant role on the convergence (K) and the calculated phase (P). The table below summarises the values of N, α , and K (the number of iterations needed for convergence) that enabled proper working of the method. These were incorporated in the software to cover the required frequency range.

TABLE SHOWING THE VALUES OF THE NUMBER OF FILTER WEIGHTS CONVERGENCE PARAMETER AND ITERATIONS FOR THE COVERED FREQUENCY RANGE

FREQUENCY	WEIGHTS	CONVERGENCE PARAMETER	ITERATIONS
F	N	V1	K
200	8	0.006	200
300	8	0.006	200
400	8	0.006	200
500	8	0.006	200
600	8	0.006	200
700	4	0.004	300
800	4	0.004	300
900	4	0.003	400
1000	4	0.003	400
2000	3	0.003	400

Thus the proper values of N, α, and K enable the filter to converge smoothly to the actual phase. The range covered in the table is from 2000 Hz. to 100 Hz. and the obtained readings are shown in the Appendix 4.

Frequencies less than 100 Hz. could not be considered since considerable attenuation was produced by the RC circuit and the output amplitude was too low to fit the range of ADC. This thus provides a frequency range f_s/f_o from 9130/2000 to 9130/100. Hence if all the frequencies outside this range are to be covered then the sampling frequency must be made variable.

Some of the highlights of system performance are presented below:

1. The sampling frequency must be at least four times the frequency of the signals.
2. The convergence parameter and number of weights determine the number of iterations for convergence, as a function of the ratio f_s/f_o.
3. The working of the method depends to a great extent on the performance of the ADC.
4. The amplitude of the input to the ADC must be properly scaled to suit the operating range of the ADC for proper digitizing.
5. The trigger reference must be as accurate as possible.
6. The data acquisition time for each signal is around 30 sec. and the processing time around 100 sec. Hence in order to decrease the time for one phase measurement the two times must be minimized.
7. The sampling frequency must be made variable to accomodate the

entire frequency range.

8. A 12 bit ADC has been used; a lower resolution would increase the number of iterations for convergence.

9. There is always some residual error due to the fact that the method does not converge to the Wiener solution.

The performance of the adaptive method was quite satisfactory. The main sources of error may be due to the characteristics of the input signal (noise, phase drift etc.) the requirement of an accurate trigger reference not being adequately met, shifts in frequency of function generator with changes in amplitude, accuracy of sampling frequency and finally the performance and repeatability of the ADC.

It may be concluded that this method is capable of being implemented and also capable of reasonable accuracy assuming allthe sources of error are rooted out or minimized. A suitable system for implementation of this method is discussed in a later chapter.

CHAPTER 4

THE FFT AND CORRELATION ALGORITHMS

4.1 INTRODUCTION

The Fast Fourier Transform (FFT) and Correlation algorithms represent other suitable methods for the determination of phase. The FFT forms a very powerful tool in the frequency domain analysis of a sampled signal. It is possible to investigate certain characteristics of the original signal by studying the amplitude and phase spectra which the FFT makes possible. The Correlation technique is more or less used along with the FFT and is also discussed. It is shown that the FFT provides valuable information about the sampled signal which can be utilised to describe system performance and other characteristics of the signal.

4.2 THEORY OF THE FFT

Consider the Continuous time signal x(t) sampled at the Nyquist rate to generate the sequence x(m). The discrete Fourier transform (DFT) is a Fourier representation of the given sequence x(m) $0<=m<=N-1$ and is defined as $X(n)= \sum_{m=0}^{N-1} x(m)W^{nm}$, $0<=n<=N-1$, where N is finite and $W=e^{-j2\pi/N}$ and X(n) is the nth DFT coefficient. It is seen that the DFT coefficients are representative of the coefficients obtained by expanding the given signal in a Fourier series. This is shown in fig.4.1.

Thus the DFT coefficients X(n) represent the sampled signal in the frequency domain as follows.

$$X(n)= \sum_{m=0}^{N-1} x(m)W^{nm} \qquad 0<=n<=N-1$$

$$W=e^{-j2\pi/N}$$

$$X(nw_n) = X(n)$$

which implies that the DFT coefficients yield the Fourier transform of a given sequence at a set of points that are $w_n = 2\pi/NT$ radians/sec apart, or $f_n = 1/NT$ Hz. apart.

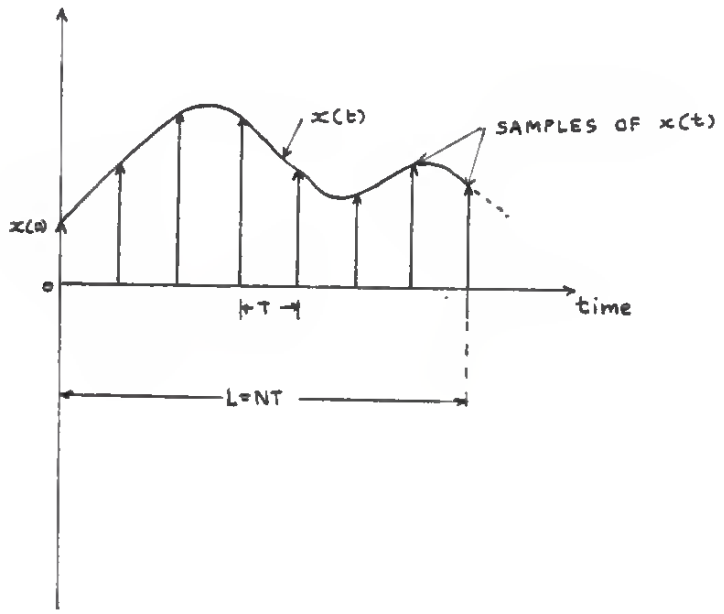


Fig.4.1. An N point DFT of a periodic DT signal.

Thus considering the DFT coefficients X(n) we can see that "n" is the frequency index and that X(n) is in general a complex quantity having a real and imaginary part. Thus we define the amplitude spectrum as $p_n = |X(n)|$

and the phase spectrum as $\gamma_n = \arctan[Im(X(n))/Re(X(n))]$

Thus we see that in the frequency domain, X(n) shows the frequency content of the original signal x(t), except that this information is not available as a continuous function of frequency but only at discrete points given by $f_n = 1/NT$ Hz.

The FFT algorithm is merely an efficient and fast method developed to compute the DFT. Thus a N point FFT of a sequence x(m) is obtained by running an FFT of N data points obtained by sampling x(t). If the sampling frequency is made numerically equal to the value of N then we have $f_n = 1/NT = f_s/N$ since $T = 1/f_s$ ie. $f_n = 1$ Hz.

Thus for n=0 we have |X(0)| representing the DC component

for n=1 we have |X(1)| representing the amplitude of the 1 Hz. component and so on.

Similiarly for the phase spectrum the phase of the different frequency components of the signal is obtained at n=0,1,2.. Hz.

Now let us consider two signals x(t) and y(t) where y(t) is the leading signal. If x(t) and y(t) are sine waves of some frequency f_o then by running a FFT of the samples of x(t) and y(t) we have access to their phase spectra. We determine the phase of the frequency component f_o for x(t) and that for y(t) and the difference yields the phase difference between x(t) and

y(t).

4.3 SIMULATION AND VERIFICATION

The simulation was carried out on the HP 9845B Desktop computer. Two sine waves of frequency equal to 24 Hz. were generated as follows and sampled at 128 Hz. ie.f_s=128 Hz. 128 samples (N=128) were taken for running the FFT.

$x(k) = 0.5 \sin(w_o kT)$

$y(k) = 2 \sin(w_o kT+P)$

P=50 degrees and its equivalent in radians in the expression for y(k).

A 128 point FFT was run on the two sampled signals using the AC waveform analysis package available on the HP 9845B. The amplitude and phase spectra were obtained on the plots shown. Also a printout of the FFT coefficients was obtained. The plots and the printout are shown figs.4.2 through 4.6.

Let us consider the different spectra in turn. A look at the amplitude spectrum of x(k) shows that there is a single frequency component at 24 Hz. with a magnitude of 0.5 which is justified since $x(t) = \sin(2\pi*24*t)(.5)$

A look at the amplitude spectrum of y(k) shows a peak at 24 Hz. again with an amplitude of 2 which is justified since $y(t) = 2\sin(2\pi*24*t+P)$.

By looking at the printout in fig.4.6 we see that the phase of the 24 Hz. component for x(k) is -22.5 degrees and that for y(k) is 27.5 degrees . Hence the phase difference is given by 27.5-(-22.5) = 50 degrees which is exactly the phase difference

between x(t) and y(t).

Thus the algorithm was verified by using the FFT package on the HP 9845B.

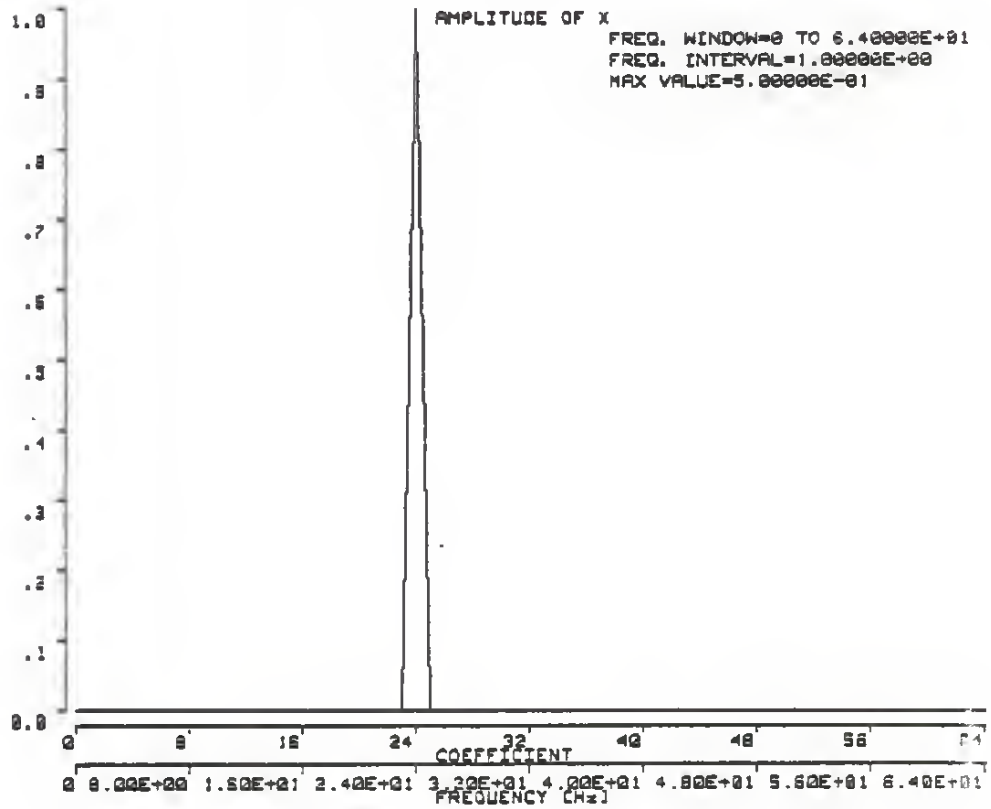


Fig.4.2. Amplitude spectrum of X.

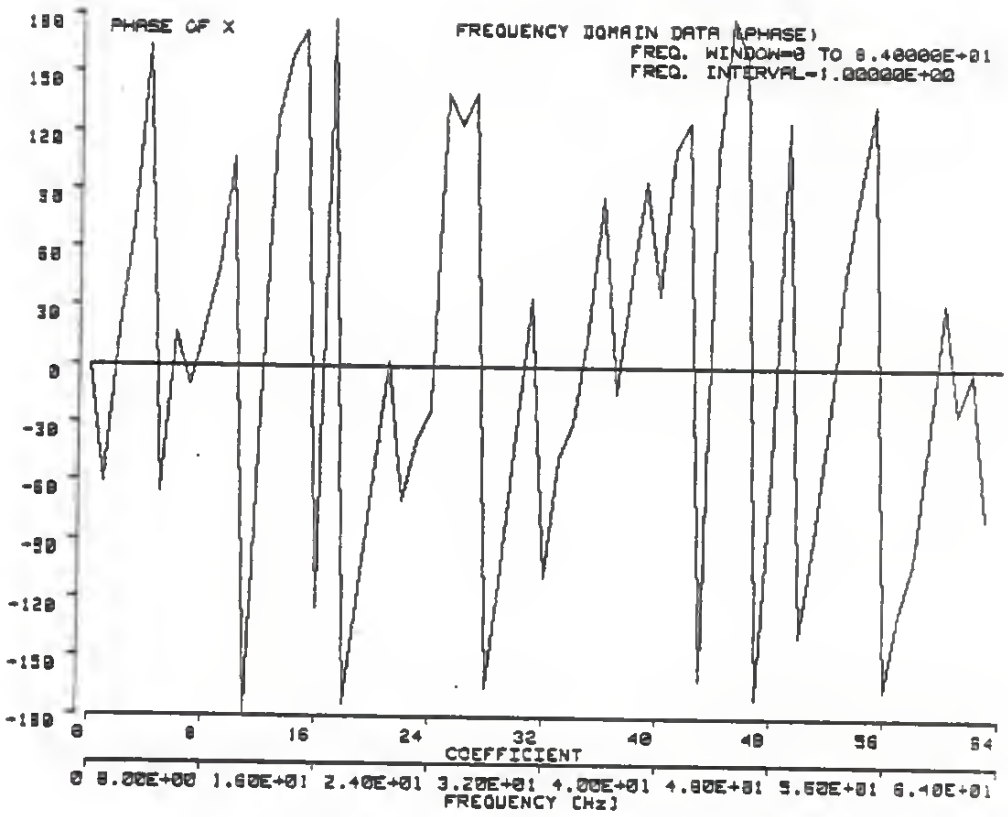


Fig.4.3. Phase spectrum of X.

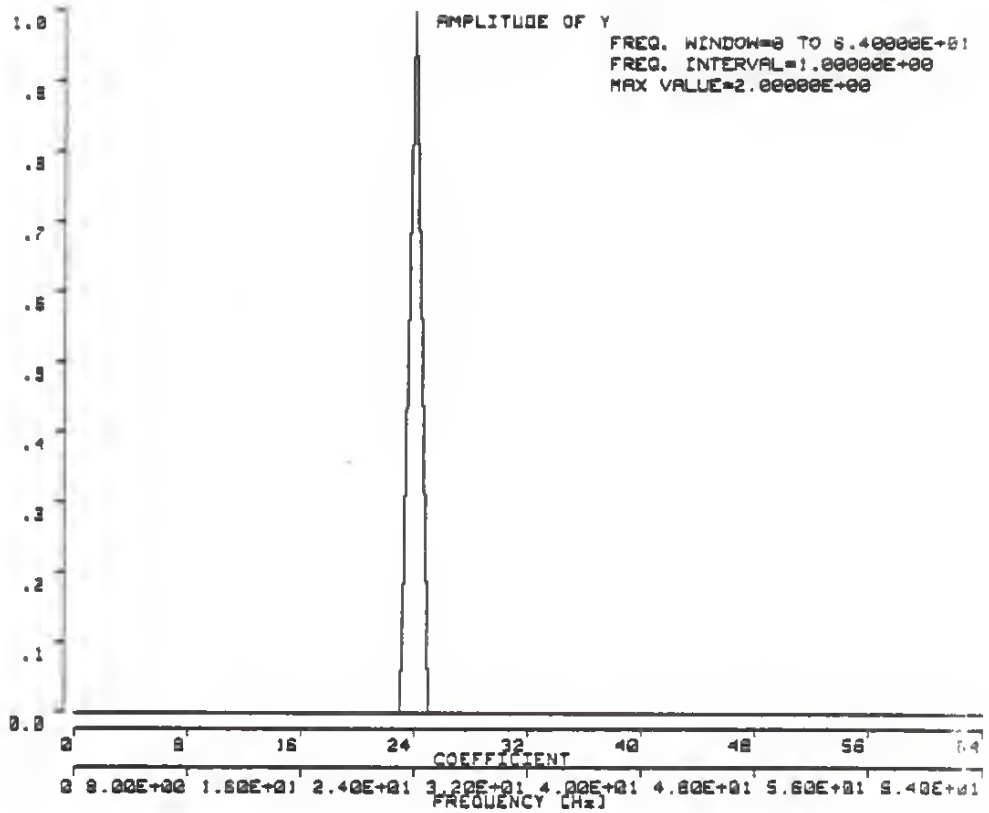


Fig.4.4. Amplitude spectrum of Y.

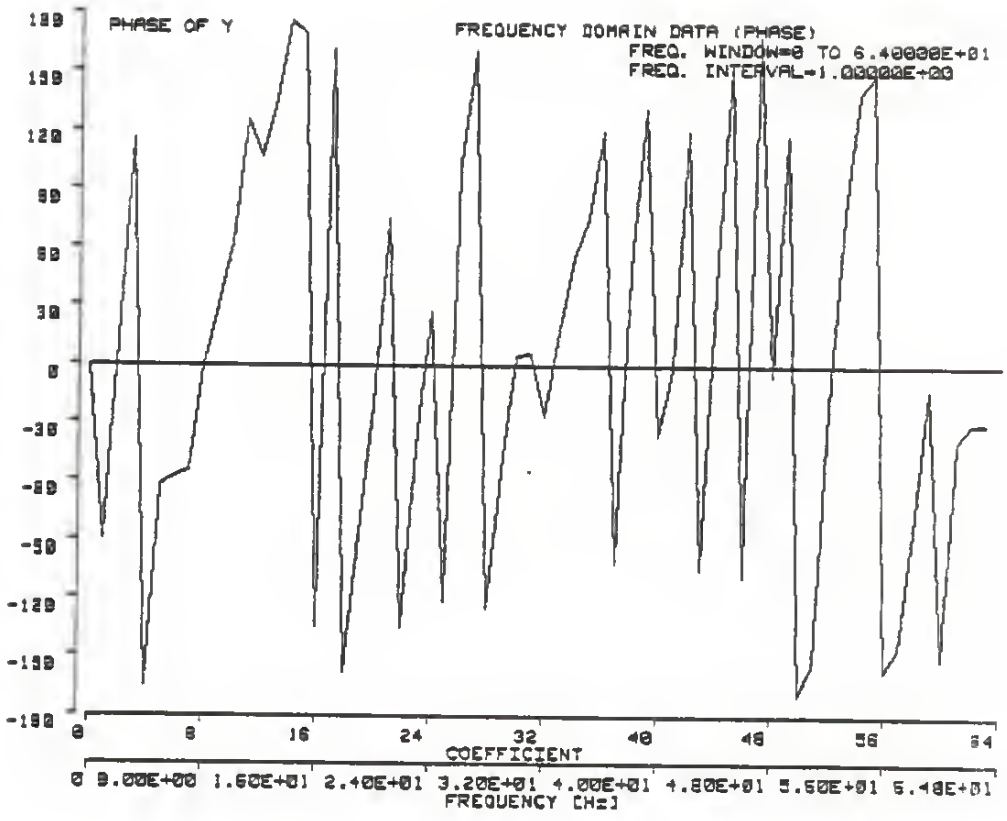


Fig.4.5. Phase spectrum of Y.

FREQUENCY DOMAIN DATA

FREQUENCY WINDOW=0 TO 6.40000E+01[Hz]
FREQUENCY INTERVAL= 1.00000E+00[Hz]

COEFF.	FREQUENCY[Hz]	REAL	IMAG	MAGNITUDE	PHASE[DEG]
DC TERM	0.00000E+00	-1.06250E-11			
MAX FREQ.	6.40000E+01	6.09375E-12			
1	1.00000E+00	1.03122E-11	-1.85450E-11	2.12193E-11	-60.92
2	2.00000E+00	1.41345E-11	4.47469E-12	1.48258E-11	17.57
3	3.00000E+00	3.49895E-12	1.09691E-11	1.15136E-11	72.31
4	4.00000E+00	-1.09111E-11	3.22410E-12	1.13774E-11	163.54
5	5.00000E+00	7.58508E-12	-1.62487E-11	1.79319E-11	-64.98
6	6.00000E+00	1.00908E-11	2.99170E-12	1.05154E-11	16.53
7	7.00000E+00	9.72328E-12	-1.81103E-12	9.89050E-12	-10.55
8	8.00000E+00	1.88410E-11	6.43141E-12	1.99085E-11	18.85
9	9.00000E+00	1.81196E-11	2.07836E-11	2.75731E-11	48.92
10	1.00000E+01	-1.21749E-11	4.06082E-11	4.23941E-11	106.69
11	1.10000E+01	-3.30581E-11	-1.96895E-13	3.30587E-11	-179.66
12	1.20000E+01	1.01196E-11	-4.26504E-12	1.09817E-11	-22.85
13	1.30000E+01	-9.59942E-12	1.28573E-11	1.60453E-11	126.75
14	1.40000E+01	-1.89773E-11	7.21818E-12	2.03037E-11	159.18
15	1.50000E+01	-2.13964E-11	3.01387E-12	2.16076E-11	171.98
16	1.60000E+01	-9.97835E-12	-1.43867E-11	1.75084E-11	-124.74
17	1.70000E+01	-3.95310E-12	2.12129E-13	3.95879E-12	176.93
18	1.80000E+01	-2.69206E-11	-2.62418E-12	2.70482E-11	-174.43
19	1.90000E+01	-1.66766E-11	-3.78865E-11	4.13944E-11	-113.76
20	2.00000E+01	2.25984E-11	-3.42549E-11	4.10376E-11	-56.59
21	2.10000E+01	2.06825E-11	1.81549E-12	2.07074E-11	2.81
22	2.20000E+01	3.40701E-12	-9.18170E-12	9.79344E-12	-69.64
23	2.30000E+01	2.96175E-11	-2.41783E-11	3.82333E-11	-39.23
24	2.40000E+01	4.61940E-01	-1.91342E-01	5.00000E-01	-22.50
25	2.50000E+01	-1.77575E-11	1.45107E-11	2.29323E-11	140.75
26	2.60000E+01	-6.15491E-12	8.93955E-12	1.08533E-11	124.55
27	2.70000E+01	-2.23421E-11	1.82666E-11	2.88589E-11	140.73
28	2.80000E+01	-3.96842E-11	-1.06410E-11	4.10861E-11	-164.99
29	2.90000E+01	-8.60685E-12	-3.87762E-11	3.97199E-11	-102.51
30	3.00000E+01	2.52041E-11	-1.73751E-11	3.06128E-11	-34.58
31	3.10000E+01	9.07410E-12	6.26559E-12	1.10271E-11	34.62
32	3.20000E+01	-4.21875E-12	-1.26563E-11	1.33409E-11	-108.43
33	3.30000E+01	1.07805E-11	-1.16101E-11	1.58434E-11	-47.12
34	3.40000E+01	1.90111E-11	-1.05227E-11	2.17290E-11	-28.96
35	3.50000E+01	2.63512E-11	9.26184E-12	2.79315E-11	19.37
36	3.60000E+01	9.84564E-13	1.73759E-11	1.74038E-11	86.76
37	3.70000E+01	8.84819E-12	-2.09727E-12	9.09335E-12	-13.33
38	3.80000E+01	1.61156E-11	1.60721E-11	2.27601E-11	44.92
39	3.90000E+01	-2.11871E-12	2.23652E-11	2.24654E-11	95.41
40	4.00000E+01	5.00000E-11	3.75000E-11	6.25000E-11	36.87
41	4.10000E+01	-1.72640E-12	4.13274E-12	4.47884E-12	112.67
42	4.20000E+01	-9.29969E-12	1.26525E-11	1.57025E-11	126.32
43	4.30000E+01	-1.46773E-11	-5.17347E-12	1.55624E-11	-160.58
44	4.40000E+01	-1.22805E-12	3.33848E-12	3.55719E-12	110.20
45	4.50000E+01	-7.00515E-12	1.22428E-13	7.00622E-12	179.00
46	4.60000E+01	-3.53890E-12	1.34160E-12	3.78467E-12	159.24
47	4.70000E+01	-1.36030E-11	-2.24751E-12	1.37874E-11	-170.62
48	4.80000E+01	7.47835E-12	-9.69916E-12	1.22474E-11	-52.37
49	4.90000E+01	-6.54820E-12	9.04351E-12	1.11653E-11	125.91
50	5.00000E+01	-9.90145E-12	-8.83301E-12	1.32688E-11	-138.26
51	5.10000E+01	1.78526E-13	-9.34736E-12	9.34907E-12	-88.91
52	5.20000E+01	1.07659E-11	-6.07360E-12	1.23610E-11	-29.43
53	5.30000E+01	1.11360E-11	1.16109E-11	1.60880E-11	46.20
54	5.40000E+01	-4.07135E-13	1.28334E-11	1.28398E-11	91.82
55	5.50000E+01	-1.81793E-11	1.80911E-11	2.56471E-11	135.14
56	5.60000E+01	-5.11848E-11	-1.29436E-11	5.27960E-11	-165.81
57	5.70000E+01	-1.45457E-11	-2.07093E-11	2.53072E-11	-125.08
58	5.80000E+01	-5.81462E-12	-3.33620E-11	3.38649E-11	-99.89
59	5.90000E+01	2.48705E-11	-1.98252E-11	3.18053E-11	-38.56
60	6.00000E+01	8.60474E-12	5.58911E-12	1.02606E-11	33.01
61	6.10000E+01	1.19537E-11	-5.27146E-12	1.30690E-11	-23.79
62	6.20000E+01	1.12365E-11	-3.16946E-13	1.13409E-11	-1.62
63	6.30000E+01	1.55675E-12	-7.03644E-12	7.20659E-12	-77.52

Fig.4.6. Printout of FFT data for X.

FREQUENCY DOMAIN DATA

FREQUENCY WINDOW=0 TO 6.40000E+01[Hz]
FREQUENCY INTERVAL= 1.00000E+00[Hz]

COEFF.	FREQUENCY[Hz]	REAL	IMAG	MAGNITUDE	PHASE[DEG]
DC TERM	0.00000E+00	-1.80469E-11			
MAX FREQ.	6.40000E+01	2.91406E-11			
1	1.00000E+00	3.68841E-13	-2.47676E-11	2.47704E-11	-89.15
2	2.00000E+00	2.24861E-11	4.97397E-12	2.30296E-11	12.47
3	3.00000E+00	-1.50707E-11	3.07509E-11	3.42453E-11	116.11
4	4.00000E+00	-4.57583E-11	-1.18837E-11	4.72763E-11	-165.44
5	5.00000E+00	2.94318E-11	-5.25806E-11	6.11319E-11	-61.22
6	6.00000E+00	7.92958E-12	-1.21028E-11	1.44691E-11	-56.77
7	7.00000E+00	4.00659E-11	-5.63646E-11	6.91538E-11	-54.59
8	8.00000E+00	1.04142E-10	3.59250E-13	1.04142E-10	.20
9	9.00000E+00	7.49475E-11	4.47302E-11	8.72807E-11	30.83
10	1.00000E+01	6.25474E-11	1.20704E-10	1.35947E-10	62.61
11	1.10000E+01	-6.64133E-11	9.20721E-11	1.13525E-10	125.80
12	1.20000E+01	-6.87476E-12	2.27892E-11	2.38035E-11	106.79
13	1.30000E+01	-7.84951E-11	7.32328E-11	1.07352E-10	136.99
14	1.40000E+01	-8.49952E-11	5.97237E-12	8.52047E-11	175.98
15	1.50000E+01	-4.53150E-11	8.25440E-12	4.60606E-11	169.68
16	1.60000E+01	-3.44826E-11	-3.56039E-11	4.95650E-11	-134.08
17	1.70000E+01	-2.99080E-11	9.60078E-12	3.14112E-11	162.20
18	1.80000E+01	-1.24332E-10	-4.95360E-11	1.33837E-10	-158.28
19	1.90000E+01	8.22449E-12	-1.56238E-10	1.56454E-10	-86.99
20	2.00000E+01	1.10939E-10	-4.29784E-11	1.18973E-10	-21.18
21	2.10000E+01	1.49763E-11	5.70855E-11	5.90173E-11	75.30
22	2.20000E+01	-3.94353E-11	-4.07295E-11	5.66925E-11	-134.08
23	2.30000E+01	9.42521E-11	-8.66410E-11	1.28024E-10	-42.59
24	2.40000E+01	1.77402E+00	9.23497E-01	2.00000E+00	27.50
25	2.50000E+01	-2.22918E-11	-3.72059E-11	4.33729E-11	-120.93
26	2.60000E+01	-1.62640E-12	7.01844E-12	7.20443E-12	103.05
27	2.70000E+01	-8.92625E-11	2.97624E-11	9.40935E-11	161.56
28	2.80000E+01	-8.51247E-11	-1.24549E-10	1.50860E-10	-124.35
29	2.90000E+01	9.52150E-11	-1.30260E-10	1.61349E-10	-53.83
30	3.00000E+01	1.01482E-10	9.83270E-12	1.01957E-10	5.53
31	3.10000E+01	4.60154E-11	4.54929E-12	4.62397E-11	5.65
32	3.20000E+01	3.67188E-11	-1.84375E-11	4.10878E-11	-26.66
33	3.30000E+01	7.54806E-11	2.51156E-11	7.95495E-11	18.40
34	3.40000E+01	3.63407E-11	5.40154E-11	6.51023E-11	56.07
35	3.50000E+01	1.24373E-11	4.89966E-11	5.05505E-11	75.76
36	3.60000E+01	-3.10289E-11	5.19379E-11	6.05007E-11	120.86
37	3.70000E+01	-2.04997E-12	-1.08416E-11	1.10337E-11	-100.71
38	3.80000E+01	3.65730E-11	4.25438E-11	5.61031E-11	49.32
39	3.90000E+01	-4.68920E-11	5.23645E-11	7.02915E-11	131.84
40	4.00000E+01	2.10938E-10	-1.53125E-10	2.60657E-10	-35.98
41	4.10000E+01	3.55658E-11	-6.04535E-13	3.55710E-11	-.97
42	4.20000E+01	-2.76509E-11	4.72836E-11	5.47751E-11	120.32
43	4.30000E+01	-1.57932E-11	-5.87581E-11	6.08436E-11	-105.04
44	4.40000E+01	4.41818E-11	3.67949E-11	5.74969E-11	39.79
45	4.50000E+01	-4.85676E-11	2.17144E-11	5.32008E-11	155.91
46	4.60000E+01	-4.66132E-12	-1.42780E-11	1.50196E-11	-108.08
47	4.70000E+01	-1.65805E-11	3.18693E-12	1.68840E-11	169.12
48	4.80000E+01	3.66701E-11	-3.72891E-12	3.68592E-11	-5.81
49	4.90000E+01	-3.36364E-11	6.39559E-11	7.22617E-11	117.74
50	5.00000E+01	-4.44793E-11	-8.50738E-12	4.52556E-11	-169.17
51	5.10000E+01	-1.71285E-11	-8.58689E-12	1.91604E-11	-153.37
52	5.20000E+01	2.57403E-11	-9.45246E-12	2.74211E-11	-20.16
53	5.30000E+01	8.41300E-12	4.17708E-11	4.26096E-11	78.61
54	5.40000E+01	-3.49198E-11	2.85754E-11	4.51214E-11	140.71
55	5.50000E+01	-5.68347E-11	3.41659E-11	6.63136E-11	148.99
56	5.60000E+01	-1.24454E-10	-5.27637E-11	1.35178E-10	-157.02
57	5.70000E+01	-1.30833E-10	-9.04259E-11	1.59041E-10	-145.35
58	5.80000E+01	1.84022E-11	-1.88539E-10	1.89435E-10	-84.43
59	5.90000E+01	1.12401E-10	-2.53973E-11	1.15234E-10	-12.73
60	6.00000E+01	-2.20745E-11	-1.24119E-11	2.53246E-11	-150.65
61	6.10000E+01	4.33981E-11	-3.48041E-11	5.56302E-11	-38.73
62	6.20000E+01	5.63394E-11	-3.24009E-11	6.49919E-11	-29.90
63	6.30000E+01	6.38797E-11	-3.74012E-11	7.40254E-11	-30.35

Fig.4.6.a. Printout of FFT data for Y.

4.4 PRACTICAL ASPECTS AND WORKING OF THE METHOD

In order to demonstrate the working of this method in a practical situation, use was made of the system consisting of the North Star microcomputer, the MC 6802 interface and the HP Multiprogrammer discussed in chapter 3. The same system was used to obtain samples of x(t) and y(t) generated using the function generator and the RC circuit. These samples were then keyed into the HP 9845B computer to form the data on which the FFT was to be run using the AC waveform analysis package.

The package is a standard one available on the HP 9845B. It has two modes of data input - manual and from disc storage. The simulation experiment was carried out by storing 128 samples of data of x(t) and y(t) on disc and then made use of by the FFT program in the package. However, when the actual data samples were taken using the NS system, the manual mode was used and data was input to the HP 9845B sample by sample in this fashion and finally stored on disc.

Certain aspects had to be given due importance and consideration. The sampling frequency is 9130 Hz. and just 128 samples were used which means that the frequency index "n" in the FFT components represents a frequency separation of f_n=1/(128*[1/9130])= 71.33 Hz. Thus if the input signal frequency was to figure in the plots then some integral multiple of f_n had to be chosen as the value for the signal frequency. Hence utmost care was taken to accurately adjust the signal frequency to this

value before sampling was done.

The results are shown in Appendix 5. The phase difference by the FFT method was compared with the actual phase difference obtained by the formula for the phase shift of the RC circuit at that frequency.

These plots can be compared with those obtained by simulation. It can be seen in the amplitude plots for the simulation case there is a distinct peak at the signal frequency and the amplitude is zero at all points but this is not the case in the plots obtained by using actual data samples. These plots in fact show that the sampled signal contains a small amount of other frequencies as well. These effects are considered later in this chapter. It can also be seen that the error in the case of simulation was zero but in this case there is a finite phase error (obtained as the difference between the obtained phase and the actual phase of RC circuit). This may be attributed to the following factors.

1. Finite precision of ADC (12 bits)
2. Just 128 samples were used for the FFT. Since the sampling frequency is 9130 Hz. at least as many samples would have to be used for a very good frequency resolution which would minimise phase errors.
3. Presence of noise in the signals.
4. Inaccuracies in the trigger reference and the characteristics of the function generator.

However in spite of the above drawbacks the error was only

0.99 degrees as can be seen from the plots and improvements in the above stated factors would minimise this error further.

The FFT is a very powerful tool for analysis in the frequency domain. It is capable of a high resolution in frequency provided enough samples are used. As seen from the plots one can clearly see the noise that exists, the presence of harmonics (due to harmonic distortion of signal from generator etc.) and also any dc offsets that may be present. Thus apart from finding the phase the amplitude plots provide extra information as to the performance of the ADC, noise characteristics etc.

In the earlier discussion, it was mentioned that for a resolution of 1 Hz. the sampling frequency must be equal to the number of data points or samples. Thus in a practical system, the sampling frequency must be made variable to cover the entire frequency range. For example, the sampling frequency could be 256 Hz. to cover frequencies from 1 Hz. to 128 Hz. and a 128 point FFT is required. Similarly the value of sampling frequency could be stepped up to say 4096 Hz. and 4096 points are necessary. This means a memory of 4K would be required in hardware and the maximum signal frequency that can be analysed using the FFT would be 2048 Hz. If a memory of 16K could be afforded then f_s would be approximately 16 Khz. and the maximum signal frequency would then be 8 Khz. If such huge memory requirements cannot be met and assuming a 4K memory is available then the frequency range is limited to 2048 Hz.(assuming a resolution of 1 Hz. is necessary). In order to effectively increase the range the following approach is recommended.(fig.4.7).

In this schematic, the signals are heterodyned with a oscillator to provide an intermediate frequency of <=2048 Hz. and

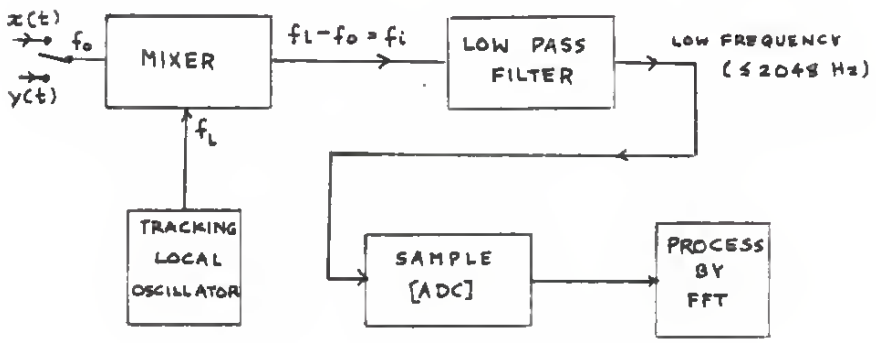


Fig.4.7. Scheme for increasing effective frequency range.

the resulting output is fed to a low pass filter. Once this frequency translation takes place the output of the LPF still contains all the phase information of either x(t) or y(t) and hence can be sampled and processed by the FFT algorithm. The mixer may be realised using an analog multiplier.

With the existing system the sampling frequency is fixed at 9130 Hz. and yet can be used to obtain the phase response of a RC filter from 0 Hz. to say 9130/2 =4500 Hz. The only restraint is that the signal frequencies must be raised in steps from 0 Hz. as integral multiples of f_n=1/NT Hz. Here f_n = 71.33 Hz. and the phase response of the filter can be determined.

It may be noted in passing that FFT chips are available to accomplish a very fast Fourier transform of the data samples. Also this may be achieved by an assembly routine. These may also be considered as possible methods in a practical system.

Thus the FFT method is another practical method that can be used in the determination of phase. As mentioned earlier a lot of extra information is also provided by this method and may be used to advantage.

4.5 THE CORRELATION TECHNIQUE

Efforts were also directed to obtain a suitable method for phase measurement using Correlation techniques. We first consider an attempt to use cross correlation in attaining our goal. By using the FFT along with this method a suitable method for phase

determination is described.

Theory of Correlation:

Consider two sequences $x(k)=V_x\sin(w\ kT)$ and $y(k)=V_y\sin(w\ kT+P)$.

The cross correlation sequence is given by the following:

$$C(mT)=[1/N]\sum_{k=0}^{N-1} x(k)*y(k+m) \qquad 0<=m<=N-1;$$

and with m=0 we have

$$C(0) = (V_x V_y/N)\sum_{k=0}^{N-1} \sin(w_o kT)*\sin(w_o kT+P)$$

$$= (V_x V_y/2N)\sum_{k=0}^{N-1} [\cos P - \cos(2w_o kT+P)]$$

It can be shown that $\sum_{k=0}^{N-1}\cos(2w_o kT+P)$ will go to zero if N is chosen to cover one full period or integral multiple of the period of x(k) or y(k).

Hence $C(0) =(V_x V_y/2)\ \cos P$ or $\cos P = 2C(0)/ V_x V_y$

If V and V are known accurately along with C(0) then the phase P can be determined using the above relation.

Attempts were made to utilise this relation and simulation was done on the HP 9845B. After the generation of x(k) and y(k) the two sequences were multiplied to obtain the cross correlation sequence C . This was then averaged over a period of x(k) and C(0) was computed. Some unreliable results were obtained (for eg. value of cosP>1) and the results were inconclusive and this approach was abandoned.

A slightly different approach produced encouraging results. Consider the signals $x(t)=V_x\sin(w_o t)$ and $y(t)=V_y\sin(w_o t+P)$.

The cross correlation between x(t) and y(t) is x(t)*y(t)

$$C_{xy}(t)= V_x V_y \sin(w_o t)\sin(w_o t+P)$$

$$=[V_x V_y/2]\ (\cos P - \cos(2w_o t+P))$$

$=[V_x V_y/2]\cos P - [V_x V_y/2]\cos(2w_o t+P)$

In the frequency domain the first term represents a dc term independent of frequency and the other term is a harmonic($2w_o$) term. Now if an FFT is run on the cross correlation $C_{xy}(t)$ then the amplitude spectrum will show a dc term which is equal to that given by $[V_x V_y \cos P]/2$. An FFT of x(k) and y(k) in turn would enable V_x, V_y, to be determined from the amplitude plots.
Thus the dc term ie. $C(0), V_x, V_y$, enable cosP and hence P to be determined.

4.6 SIMULATION AND VERIFICATION

Using the same example as was used for the verification of FFT by simulation the following were generated.

x(k)=0.5 sin(2π*24*k*T)

y(k)=2 sin(2π*24*k*T+P)

128 samples of x(k) and y(k) were used to generate the cross correlated sequence. An FFT of this sequence was run and the amplitude plot is shown in fig.4.8. From the printout (Appendix 6) the DC term is found (3.21394E-1) which gives C(0). The values of V and V were obtained from the amplitude plots (0.5 and 2). Thus cosP=2*C(0)/[0.5*2] and from this P was calculated and found to be 50 degrees and this was exactly the phase difference between the two signals x(t) and y(t). Thus the algorithm was verified.

4.7 PRACTICAL ASPECTS

All the practical aspects discussed for the FFT method apply here. Other considerations are the presence of harmonics in

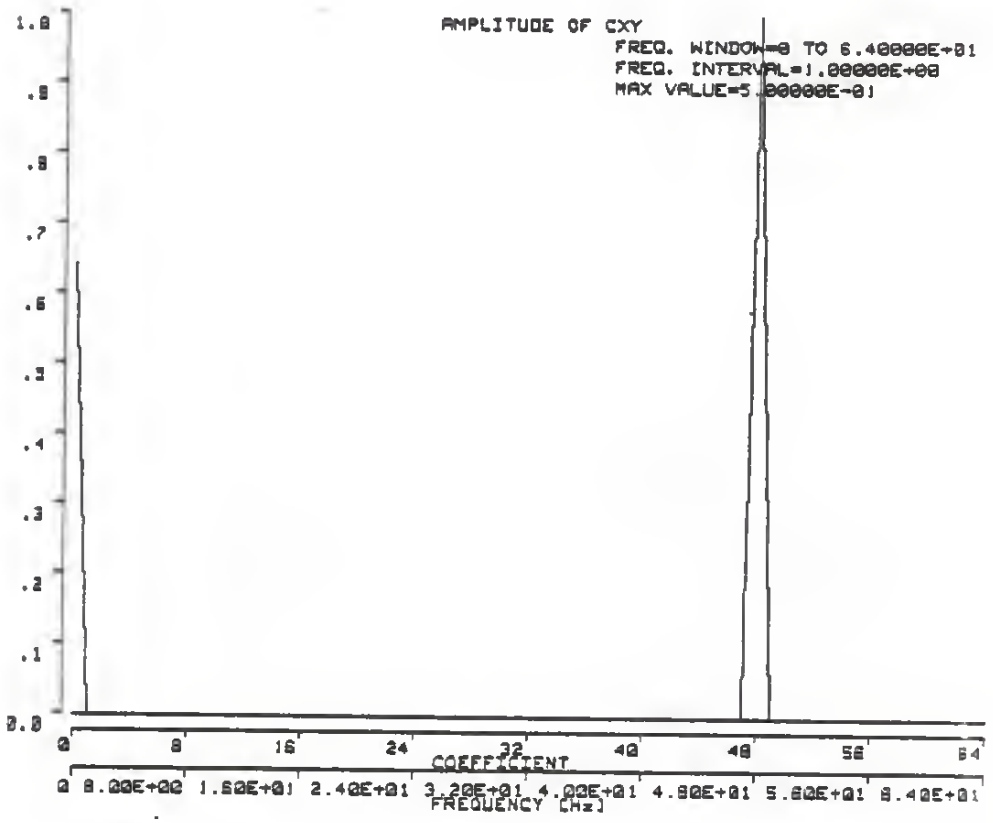


Fig.4.8. Amplitude spectrum of the correlated sequence.

the signals x(t) and y(t) due to the harmonic distortion present in the function generator output. The presence of harmonics will cause errors in the value of the DC term C(0). Also additional frequencies will be introduced as a result of the mixing action produced by the cross correlation (x(t)*y(t)). In order to overcome this problem filtering has to be done. Use of analog filters is not practical and hence the use of digital filtering is recommended.

A simple low pass digital filter can be easily implemented. The digital filters have many advantages such as flexibility, reliability and ease with which response can be changed. Thus a LP digital filter can get rid of the harmonic content in x(k) and y(k) after which the correlation sequence may be formed and analysed using the FFT.

CHAPTER 5

NUMERICAL METHODS

5.1 INTRODUCTION

Numerical and statistical methods may also be used in the determination of phase. This method relies to a great extent on an accurate determination of zero crossings of the signals. This method outlines how the determination of zero crossings of the signals can be used to calculate the phase difference . Numerical methods like curve fitting and Newton's method for finding the root of a polynomial are described.

5.2 THEORY OF THE METHOD

The method uses the fact that the accurate determination of the zero crossings of the two signals enables the determination of phase difference. Consider the situation shown in fig. 5.1. Let us focus our attention to the zero crossing zone; we shall consider the positive to negative zero crossing for the sake of uniformity for both signals. Consider the samples being taken at times t_1, t_2, t_i, t_{i+1}...t_n spaced by an amount T where T is the sampling period and $T=1/f_s$ where f_s is the sampling frequency.

Consider a straight line joining $v(t_i)$ and $v(t_{i+1})$ making intercepts x and y on the reference axis as shown. It is required to obtain the zero crossing time t . Here x+y=T, and using the property of similiar triangles it can be shown that :

$$t_x = t_i + T[1/(1+y/x)] = t_i + T[1/(1+v(t_{i+1})/v(t_i)]$$

If $v(t_{i+1})$ and $v(t_i)$ are known then t_x can be evaluated. Similiarly t_y the time of zero crossing for the second signal

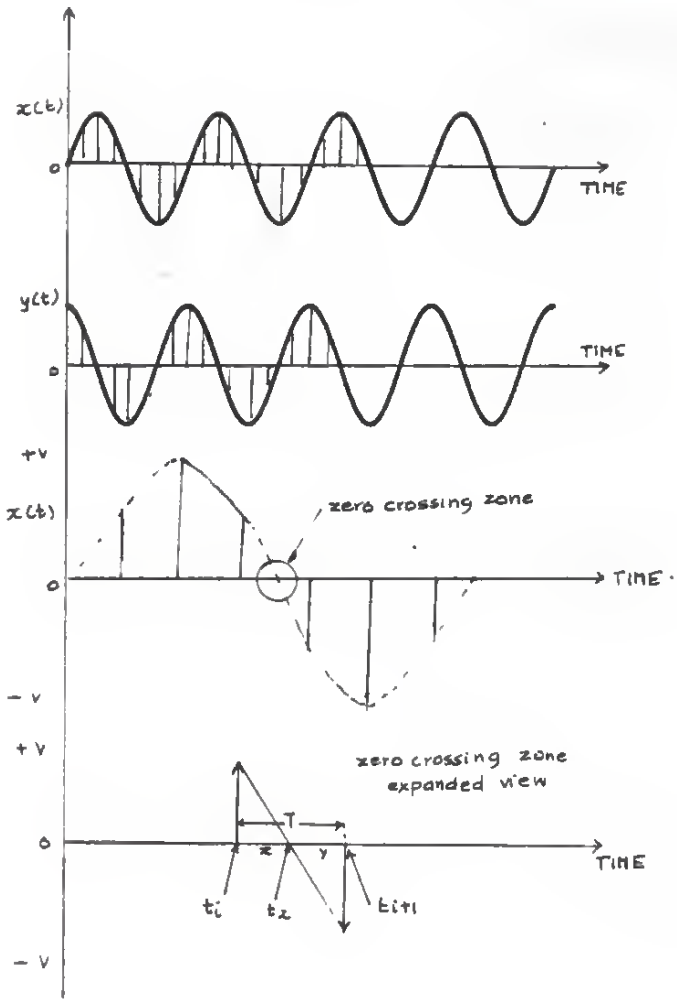


Fig.5.1. Use of zero crossings of signals to find phase.

y(t) can be determined. Once t_x and t_y are known then the difference of time between zero crossings can be determined ie. $\Delta t = t_x - t_y$. Once Δt is known the phase difference can be calculated as follows:

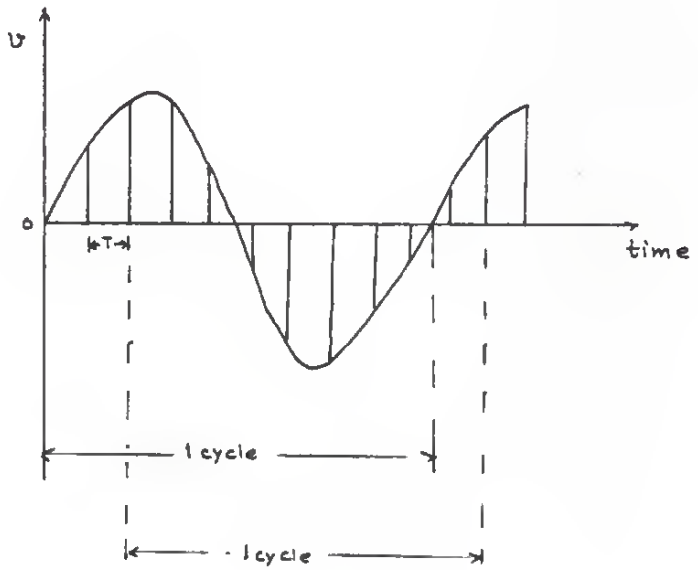


Fig.5.2. Phase determination for a periodic sine wave.

Since a sine wave is periodic the starting point is relatively unimportant as can be seen from fig.5.2. If the ratio of f_s/f_o is say 6 then there would be 6 samples per cycle of the waveform. Hence 6T would represent 1 full cycle or a phase of 360° degrees. Hence by computing $t_x-t_y=\Delta t$, the phase would be given by $P=\Delta t(f_o/f_s)*360$ in degrees. Thus the phase difference could be evaluated.

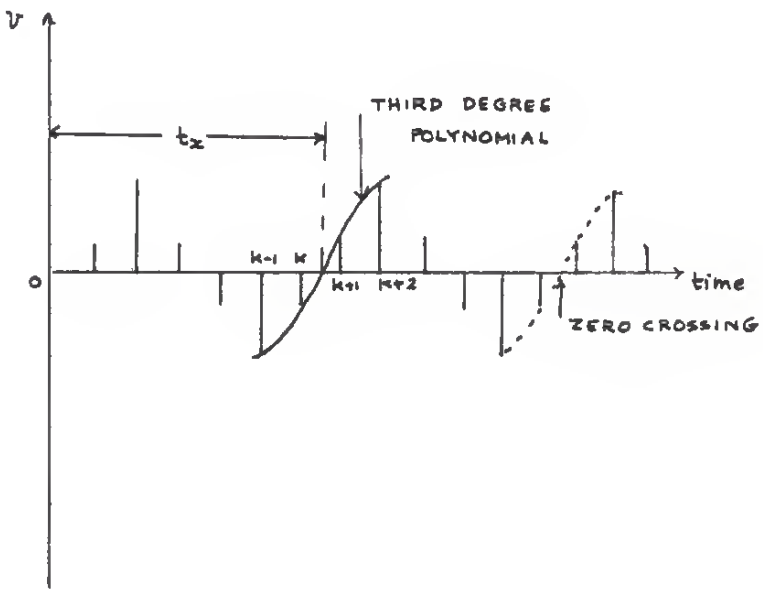
It can be seen from fig.5.1 that this method represents a straight line fit for the sine wave at the zero crossing. Thus this is an approximate fit for the sinewave at the zero crossing. Simulation was done to study the working of this method. It was found that the method produced large phase errors at some points possibly due to certain discontinuities. This may be due to the fact that the straight line fit is an approximate fit for the sine wave. Also further investigation of the expression for t_x reveals that if the ratio $v(t_{i+1})/v(t_i)$ tends to -1 then the denominator becomes zero and this causes a discontinuity. Hence this method was not considered further.

A slightly modified approach was then tried out. Since the straight line fit is approximate a higher degree polynomial is a more accurate representation of a sine wave. Hence a third degree polynomial was used to fit a curve to the sampled data at the zero crossing. The following procedure outlines the method.

1. Obtain samples of x(k) and y(k).
2. Find the values of k when there is a negative to positive transition (zero crossing) of x(k) (this can be determined from a

sign change of the successive samples from -ve to +ve). Repeat this for y(k).

3. Fit a third degree polynomial to the obtained samples at the zero crossings. This requires two data points before and two data points after the zero crossing point under consideration. Fig.5.3 illustrates this.



$p_3(x)=f(a_i)$; $0<=i<=3$, a_i are coefficients of the polynomial.

Fig.5.3. Fitting a polynomial at the zero crossings.

The third degree polynomial is fitted at each zero crossing using two data points to the left and to the right of the crossing. Repeat for y(k).

The polynomial fit is done using the divided difference table as shown in fig.5.4.

THE DIVIDED DIFFERENCE TABLE

k	x(k)			
1	x(1)			
		x'(1)=x(2)-x(1)		
2	x(2)		x"(1)=x'(2)-x'(1)/2	
		x'(2)=x(3)-x(2)		x"(1)= x"(2)-x"(1) 3
3	x(3)		x"(2)=x'(3)-x'(2)/2	
		x'(3)=x(4)-x(3)		
4	x(4)			

$p_3(x)=x(1)+x'(1)(x-1)+x"(1)(x-1)(x-2)+x"(1)(x-1)(x-2)(x-3)$

which reduces to $a_o+a_1x+a_2x^2+a_3x^3$ where the coefficients are given by a_i; $0<=i<=3$.

Fig.5.4. The divided difference table.

The coefficients of the polynomial are determined from the table and the polynomial is given by the expression shown. The same procedure is adopted for the signal y(k) to obtain the polynomial at the zero crossings.

Once this is obtained then the Newton's method is used to obtain the root of this polynomial, in other words the zero crossing time. An iterative scheme is used to determine the root using the Newton's Method.

Thus the zero crossing of x(k) and y(k) are obtained and the value of Δt is used to calculate the phase.

5.3 SIMULATION

Simulation was done for this method and it was found to work reasonably well. The software used is shown in Appendix 7a and the results in Appendix 7b.

Certain very important results were deduced from this. The ratio f_s/f_o had to be fixed at 6 for the best results (least phase error). Any decrease or increase in this ratio increases the phase error. Also when random noise is injected into the samples the error increases as shown in the results. A phase of 360 degrees could be distinguished by suitable modifications in the software. This method is very sensitive to noise and the ratio of sampling frequency to the frequency of the signals is to be fixed at 6, since the polynomial represents the best fit when this condition is satisfied. However an averaging effect is produced by determining the phase at several crossings and then

obtaining the statistical average.

5.4 WORKING OF THE METHOD

When actual data samples were used (obtained using the system discussed in chapter 3) and processed using this method (the ratio f_s/f_o was fixed at 6) the error obtained was around 4 degrees. However by running a predictor of the signals (discussed in chapter 6) which acts as a signal enhancer the error reduced to around 2 degrees.

As a conclusion this method has a constraint on the frequency ratio being fixed at 6 and also the method is sensitive to noise and needs a more accurate representation of the signals. Hence the method may not be very practical. It may be possible to use higher degree polynomials but this might require an increase in the frequency ratio (>6) thus requiring a higher sampling frequency and also limits the useful frequency range of operation of this method.

CHAPTER 6

USE OF PREDICTOR AND NOISE CONSIDERATIONS

6.1 INTRODUCTION

In this chapter the effects of noise on the signals that are used for sampling are studied. A suitable method for combating noise by the use of predictor is presented and also other effects like finite resolution of ADC, finite word length of computer, truncation etc. are also considered and studied.

6.2 USE OF A PREDICTOR

Sufficient care must be taken to ensure that there is no noise in the original signals. This can be done by the use of analog filters like low pass filters etc. In spite of this there still tends to be some noise which also gets "digitized" along with the signal. This noise is assumed have a zero mean and to be random. It is now shown that the adaptive predictor can be used as a signal enhancer and gets rid of the noise.

The schematic of the predictor is shown in fig.6.1. This scheme is similiar to the adaptive scheme in the cancellation mode (mentioned in chapter 3) except that the reference input to the filter W is a delayed version of the input x(k). In other words =1 denotes that the input to the filter at time k is x(k-1) ie. the past value. In vector form for a M weight filter ;

$X_k' = [x(k-1)\ x(k-2)...x(k-M)]$

$g(k) = W'X_k$

$e(k) = x(k)+n(k)-g(k)$

Minimizing the mean square error e(k) causes g(k) to look

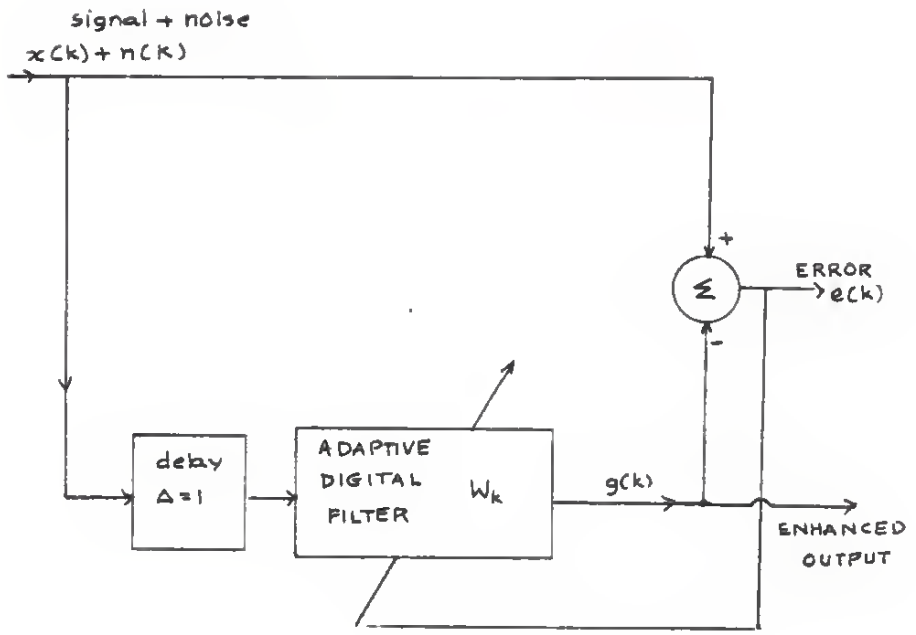


Fig.6.1. Schematic of the adaptive predictor.

like x(k) or in other words the error e(k) tends to look like the noise n(k). Thus the filter W removes as much of the correlated energy from x(k) and the enhanced output is available as g(k). Since the filter takes a few iterations to adapt itself the output g(k) must be taken after a few iterations for subsequent processing.

The predictor can also be implemented as an iterative scheme. Thus the predictor accomplishes enhacement of the signal corrupted by noise. An adaptive predictor with a 10 weight filter was used to clean up signals and the software is shown in Appendix 8. The predictor helped to enhance the signals used especially in the Numerical method and was instrumental in reducing the phase error.

6.3 OTHER CONSIDERATIONS

Certain considerations must be given due importance when dealing with implementation of DT (discrete time) digital systems. The following are the points to be noted.

1. Input quantization error is introduced due to a finite number of bits used to encode or digitize the signal x(t) to form x(k).
2. Coefficient inaccuracy error that results from using a finite number of bits to represent each coefficient w_k.
3. Round off errors that occur each time a product w_k*x(k-i) is rounded to match a prescribed word length. (Truncation errors are worse than roundoff errors.)
4. Certain types of undesired oscillations that may occur due to limited accuracy and adder overflows.

These are the general considerations and we must take into account only the pertinent ones. Basically there are two types of arithmetic:- fixed point and floating point. It can be shown that if a satisfactory representation is required, the number of bits M required for an accuracy of D decimal places is given by M>=3.3D.

Input quantization error: This error occurs when an ADC with a finite resolution is used to sample the signal x(t). If the encoded output is of B bits and the ADC has a step resolution of $q=V_{ref}/(2B)$ then there is an ambiguity of q/2 referred to as the quantization error. Hence we represent the quantized value $x^Q(n)=x(n) + e(n)$ where e(n) represents the error and is a random quantity which is uniformly distributed over (-q/2, q/2). The overall effect of e(n) is assessed in terms of its variance or power which is denoted by σ_e^2. It canbe shown that the value of this variance is equal to $q^2/12$ or $k*2^{-2B}/12$. Hence as B tends to 0 the value of the variance or power of this noise tends to zero. When $x^Q(n)$ forms an input to a DT filter then the output noise is modified depending on the transfer function H(z).

It can be shown that the effects of product roundoff errors, coefficient inaccuracies and adder overflows all contribute some amount of noise power to the overall signal content. These are given due consideration especially in special purpose hardware implementation of DT syatems. However in a software implemented system only the pertinent errors need be considered.

CHAPTER 7

DESIGN CONSIDERATIONS

After having discussed the different algorithms it is essential to consider the design aspects of a suitable microprocessor based system which will accomplish data acquisition in the most efficient manner and also process the data with speed and accuracy. A discussion of the existing system is followed by the design considerations for a more efficient system.

In the existing system the configuration being indicated in fig.7.1 it can be seen that the microprocessor MC 6802 essentially acts as a communication link between the data acquisition device HP Multiprogrammer (HPM) and the data processing device North Star (NS). This important role of the microprocessor is elaborated later in this chapter. An estimation of the data acquisition time and the data processing time will enable us to make further improvements in the existing system. It can be seen that the data acquisition is initiated by a command from the NS via the serial port to the ACIA in the MC6802 interface. The main assembly routine recognises this command ("A") and branches to the ADC subroutine. The procedure is as follows. There are two PIAs (Peripheral Interface Adapter) in the interface each with two 8 bit ports available.(refer to the schematic in Appendix 9). The PIA1 is used to handle all the addressing to the HPM (16 bits) and the PIA2 is used to handle the data from the HPM (12 bits) by utilising the two ports of 8 bits each. When the data is from the HPM (12 bits) the remaining

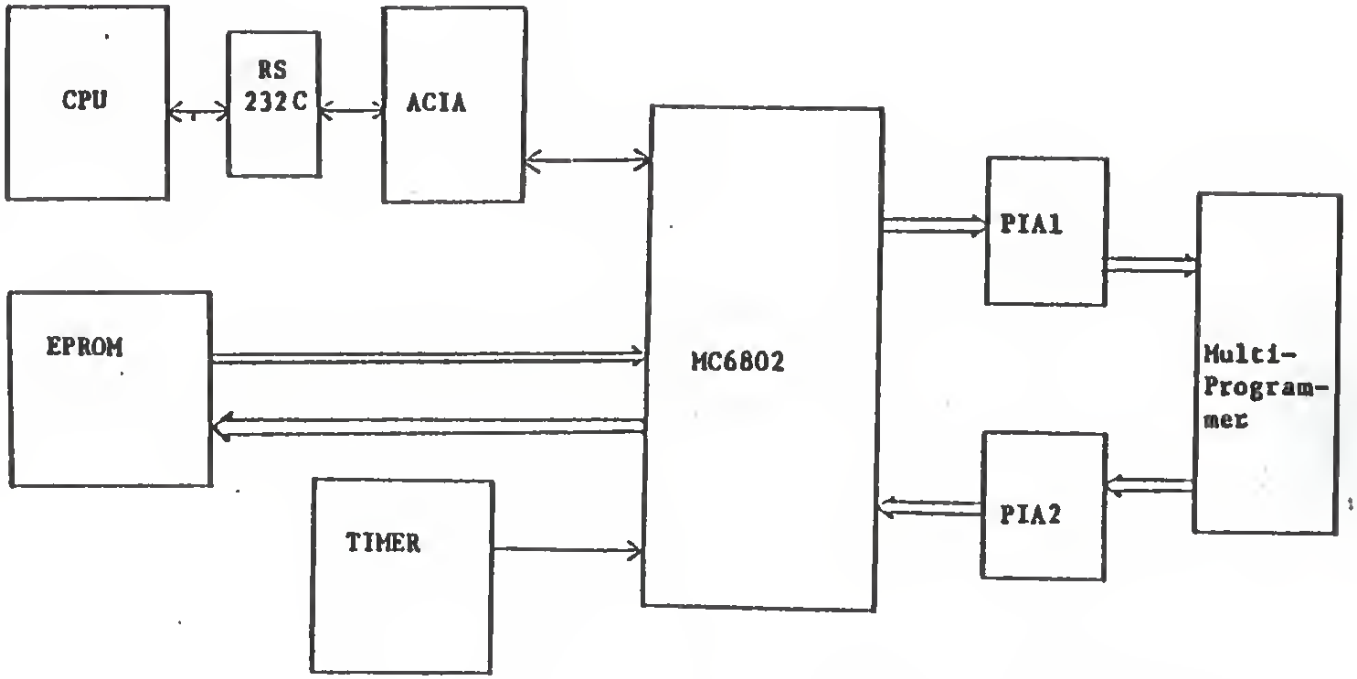


fig 7.1

INTERFACE USING THE MC6802 MICROPROCESSOR

4 bits are masked off.

The ADC subroutine causes a control word and the slot address to be input to the HPM using PIA1 (address 2000) setting up the ADC card in the HPM for input. A gate signal is then given to the ADC to initiate conversion and at the end of the conversion signal indicated by the flag, the data in PIA2 data registers is stored in a convenient memory location. This process is continued until all the samples (512) are taken and stored in memory. These samples are then sent to the NS via the serial port using the ACIA in the interface. It was experimentally found that the time taken for the whole process is around 30 secs. Hence for sampling both signals the time taken is 60 secs. It has been found that the data processing time is around 100 secs. and hence the total time for a phase measurement is around 160 secs. A method to decrease the data acquisition time is now discussed.

It can be seen that the ADC samples at 9130 Hz. and hence the time for taking 512 samples is given by 512/9130 = 0.5 sec. Hence for sampling the two signals it is around 1 sec. Thus it is apparent that the use of serial transmission is the main disadvantage since data is sent in series. Hence if the data acquisition time is to be minimised the parallel transfer of data is to be done. This may be accomplished by using the parallel port of the NS as shown in fig.7.2.

The data processing time however cannot be reduced unless a faster processing machine is used. Attention is now focussed on the data acquisition part and the errors generated and the improvements suggested for the same. It is shown that if an ADC

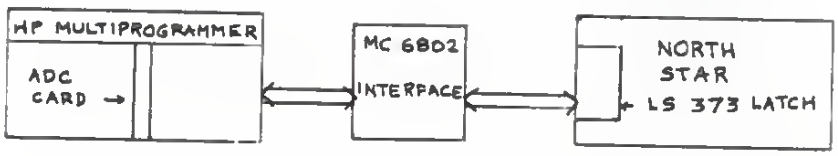


Fig.7.2. Use of the parallel port of the North Star.

has an operating range of ±1V then the signal amplitude must also be of this range to enable proper digitizing of the signal. This is shown in fig.7.3.

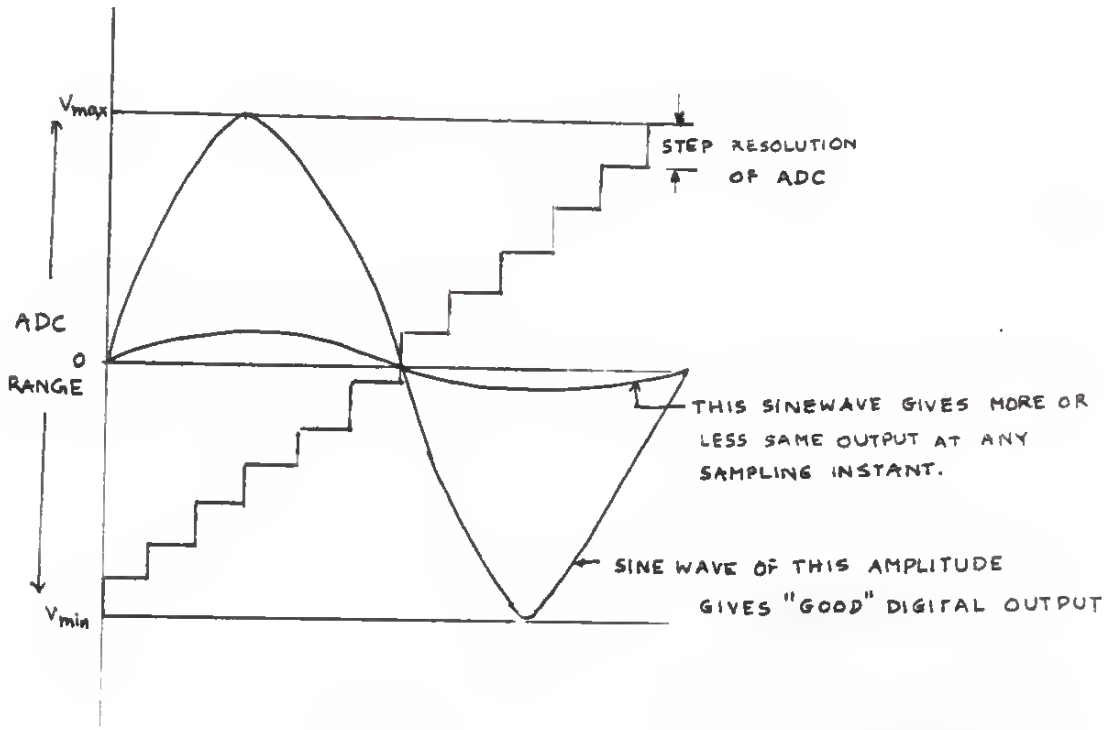


Fig.7.3. Effect of signal amplitude on "digitizing".

Hence in order to enable the "dynamic" adjusting of the signal amplitude for sampling by the ADC of a given operating range the following is suggested.(fig.7.4)

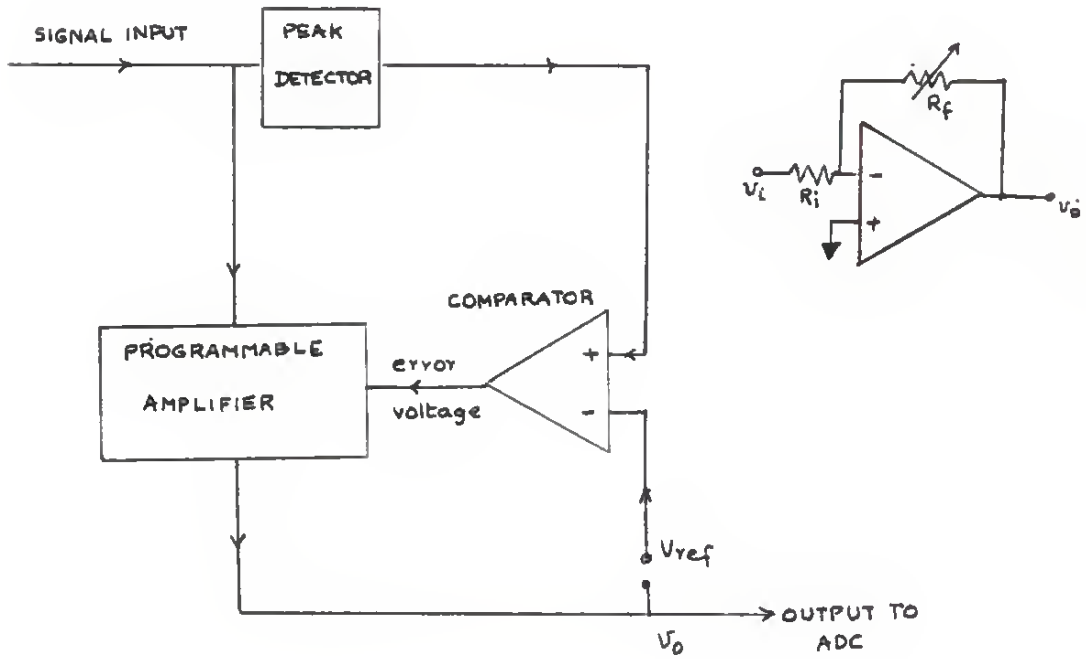


Fig.7.4. Dynamic adjusting of signal amplitude to suit ADC range.

The operation is as follows. The peak detector obtains a measure of the amplitude of the incoming signal. This is then compared with a standard reference V_{ref} and the resulting error voltage is used to control the gain of a programmable amplifier so that the output of the amplifier is now suitable for the operating range of the ADC. If instead of using a separate reference voltage the V_{ref} point is connected to V_o and derived from the output then we have a feedback control system. The programmable amplifier may be realized by using a digital attenuator or variable resistance as one of the resistances in a closed loop OPAMP circuit as shown. Thus irrespective of the signal amplitudes the output V_o is always scaled to suit the ADC range.

Errors due to delays in triggering are now considered. Since the output of the zero crossing detector or some suitable trigger reference is used it is of primary importance that these be very stable and accurate with respect to time. Fig. 7.5 illustrates the errors involved.

Let t1 be the time after the zero crossing of the signal x(k) when the trigger pulse initiates the sampling process for x(k) and that for y(k) be t2. If t2 is different from t1 then this leads to phase error given by p=(t2-t1)*2*Π*f where f represents the frequency of the signals. For a difference of 4us between t1 and t2 and a signal frequency of around 1 KHz. the phase error can be as high as 2.5 degrees. This difference may arise due to the comparator characteristics in the ZCD or more possibly due to the noise at the zero crossings. Hence in order

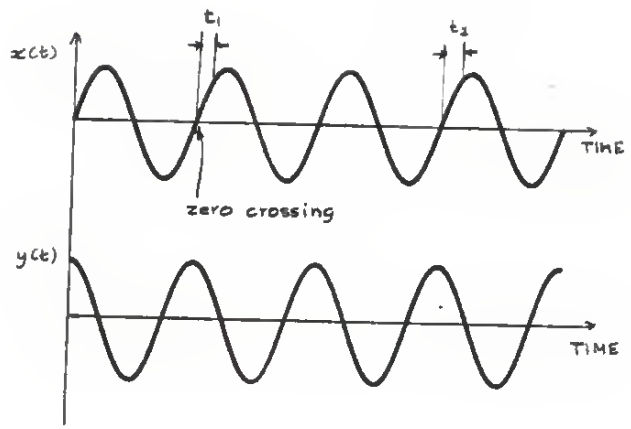


Fig.7.5. Phase errors due to inaccuracies in trigger reference.

to overcome this problem synchronous sampling is suggested using two ADCs unless of course a very stable and accurate trigger reference is provided.(fig. 7.6)

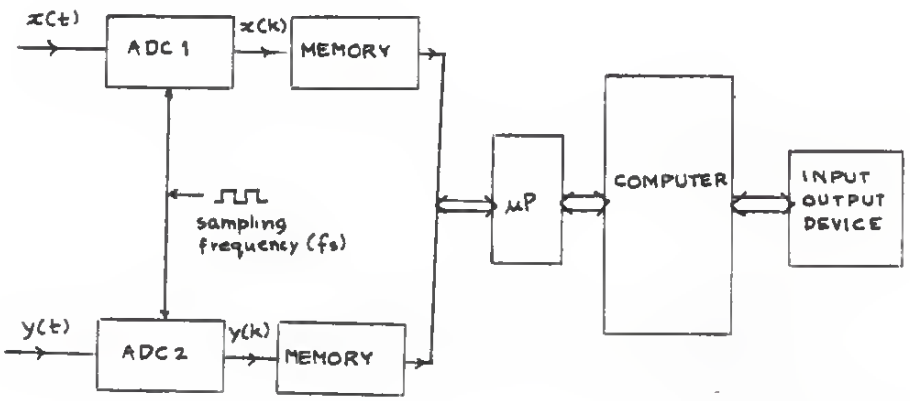


Fig.7.6. Synchronous sampling using two ADCs.

Also the use of an ADC with lower resolution (say 8 bits) would increase the convergence time and the phase error to a certain extent. In fig.7.6 the two signals x(k) and y(k) are simultaneously sampled and the samples are stored in memory 1 and 2 respectively which are then accessed by the microprocessor. The synchronous sampling would reduce the phase error to great extent.

Having considered all of the factors discussed a description of a suitable system for data acquisition and processing as follows. The schematic is shown in fig.7.7.

The two signals are initially conditioned. This includes reduction of noise and automatic magnitude scaling of signals to suit the ADC range. The control circuits and logic control the simultaneous sampling of the ADCs and storing of these samples in memory. The microprocessor then obtains these samples and sends them to the computer for processing. Interaction is provided by the Input/Output device. The most important part of this system is the portion dealing with the acquisition of samples and storing them in memory. This portion is now discussed.

There are basically two methods of controlling the data acquisition :

1. By using the microprocessor
2. Direct Memory Accessing (DMA)

The choice of the method depends on the sampling frequency required. At nominal values of sampling frequency the first method is possible but at higher sampling rates the second method

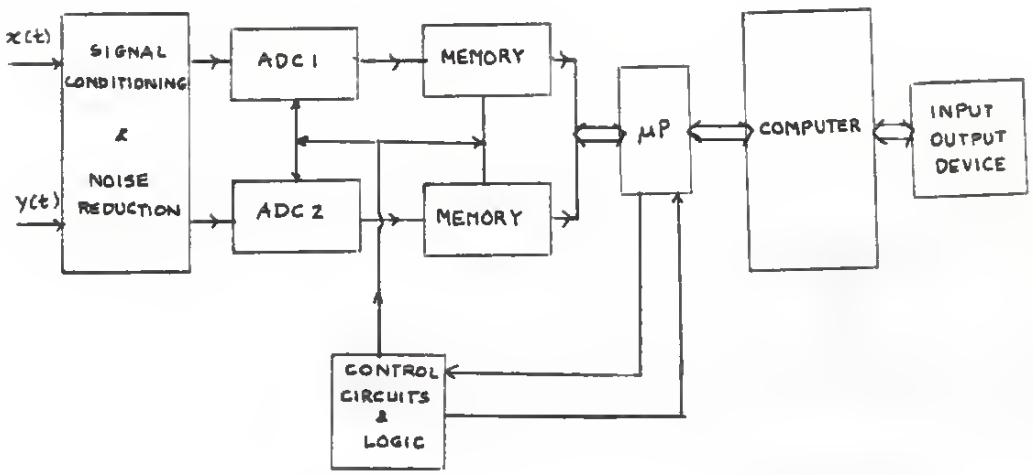


Fig.7.7. Suitable system for implementing the proposed methods.

is desirable. The first method has certain limitations which are now elaborated.

If it is assumed that the microprocessor operates with a clock of 1 MHz. (for the MC6802) then the MPU cycle time is 1 us. This is the time taken for one cycle and each assembly instruction takes a certain number of MPU cycles. The minimum software required to obtain the data samples is:

```
AGAIN   LDAA    SAMPLE      4 CYCLES
        STAA    SOMEWHERE   4 CYCLES
        INX                 4 CYCLES
        CPX  #  FIN ADDR.   3 CYCLES
        BNE     AGAIN       4 CYCLES
TOTAL NUMBER OF CYCLES      19 CYCLES
```

We shall assume that the total number of cycles is approximately 20 which means that it takes a minimum time of 20us for executing these instructions. If the control signals (gate and flag) for initiating ADC conversion and checking end of conversion are also to be incorporated in software then a minimum of another 20 us may be necessary (20 cycles) for execution and hence the total execution time is 40 us. To this must be added the ADC conversion time which from the manual is given to be 50 us. Hence the overall time is 100 us. Thus the sampling frequency is the reciprocal of this time ie. 1/100us = 10 KHz. The value of sampling frequency experimentally determined is 9130 Hz. which is approximately the value obtained in theory. Thus higher rates of sampling are not possible with this method.

The DMA method which uses direct memory accessing is capable

of a much higher sampling frequency. Fig.7.8 shows the schematic.

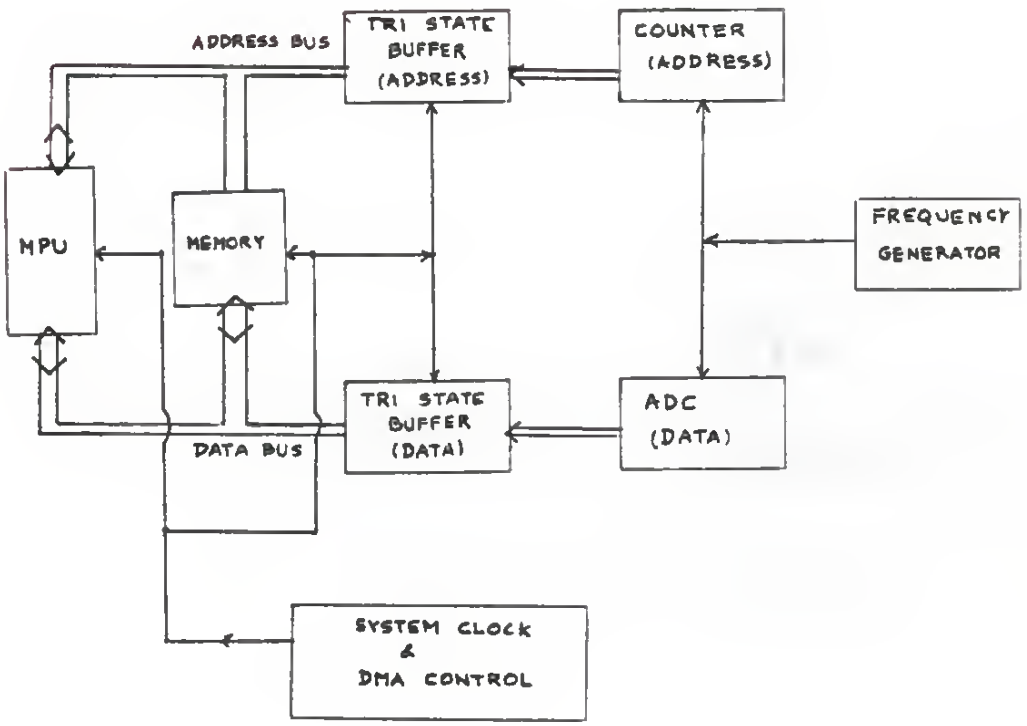


Fig.7.8. Direct Memory Accessing (DMA).

This method is especially suitable when using ADCs which can handle high sampling rates eg. the Flash ADC. Its sampling rate is around 10 MHz. However it is necessary at the same time to have a memory with a short enough access time so that it can take in the data samples given by the ADC at such a high rate.

In the schematic shown two 3-state buffers one for address and one for data are used to access the memory by the external devices. Here a high speed counter is used to provide the address to the memory. The counter is to be synchronized properly with the ADC conversion signals and the other buffer (DB) for data must also be simultaneously used to access the data bus. Thus the microprocessor must essentially disconnect itself from the address and data bus. This may be done by HALTing the uP or by other means depending on the uP used.(example cycle stealing, TSC control etc.) . The DMAC (direct memory access controller) could also be used for achieving DMA.

In order to make the sampling frequency variable a programmable frequency generator (VCO) may be used. A counter may be used to determine the signal frequency which in turn can be used to program the VCO to obtain the sampling frequency. It is essential for proper working of this system all the different parts must be properly synchronised and this requires suitable control logic.

However at high sampling rates the access time of the memory available is a very important aspect to be considered and poses the greatest problem in design and implementation. Also the rise and fall times, the propagation delays and the logic switching

delays must also be taken into account for the system to operate well. It is essential to study different types of memories available CMOS, Schottky TTL, ECL etc. and compare these with respect to speed, cost, amount of memory available, power dissipation etc.

In the system discussed the microprocessor acts as a control/communicating device. The uP is well suited for manipulation at the bit level and hence can do some scaling, bit manipulation, sign insertion etc. The data can then be transferred to a computer for processing and probably storage on disk for future processing.

As far as the processing portion of the system is considered there are again two methods 1. using assembly language 2. high level language. While the first method can handle only highly specific and dedicated algorithms the later has a lot of flexibility. The high level language algorithms in the Computer can be easily written and changed to suit the type of data involved and the processing algorithm itself may be changed for adifferent problem.

Thus by taking into account all these factors discussed the final design may be implemented.

CHAPTER 8

CONCLUSIONS

Before we attempt to conclude, we must make a comparative study of the four methods developed. The overall performance is not only dependent on the algorithm itself but also on the performance of the system used to implement it. However with the existing system the Adaptive method and the FFT method seem to perform well as compared to the other two methods. A comparison of all the methods with respect to a few features is now dealt with.

As far as simulation is concerned the Adaptive, FFT and Correlation methods performed very well. The numerical method had a larger phase error compared to the other three. When actual data samples were used the adaptive and FFT methods worked the best. The effects of noise induced large errors in the numerical method and had to be combated by the use of predictor. Harmonic distortion produced errors in the Correlation method but did not seem to affect the Adaptive and FFT methods. The FFT method has an advantage that it provides other valuable information regarding the signal. The adaptive method has an advantage that it automatically cancels the noise since it uses the cancellation mode. All the methods require that the sampling frequency be variable in order to cover the entire frequency range and be known accurately and also require that the frequency of the signals be known accurately. Having considered all of the above features it seems that the adaptive and FFT methods have the greatest advantages and accuracy. Hence the Adaptive and FFT

methods are recommended for an implementation for the measurement of phase.

CHAPTER 9

OTHER APPLICATIONS

There are several other applications of signal processing techniques in instrumentation. The use of a suitable system for data acquisition enables a variety of algorithms to be used for processing. The use of microprocessors in such applications is increasing and some new dedicated chips called "signal processor" chips are now making their appearance. A brief discussion of other possible applications follows.

Besides the phase and the FFT, the Power Density Spectrum (PDS) of the signal can also be obtained. Also digital filters can be used to shape the signal, alter its characteristics etc. The predictor can be used for signal enhancement and the combating of noise. The adaptive algorithm is a very powerful processing tool. A method for obtaining the amplitude of a sinusoidal signal is suggested as shown in fig.9.1.

It is required to find the amplitude of x(k). Let r(k) be the reference input generated within the computer at the same frequency as that of x(k). The reference input r(k) could also be derived by sampling a 1 volt reference signal of same frequency as that of x(k).

As before when the error e(k) decreases in magnitude to a sufficiently small value then r(k) is transformed in amplitude and phase by W to cancel x(k). We obtain the amplitude transfer function as:

$$H(w_0)=w_0+w_1e^{-jw_oT}+\ldots+w_Me^{-jw_oMT}$$

The magnitude of the transfer function is equal to the square

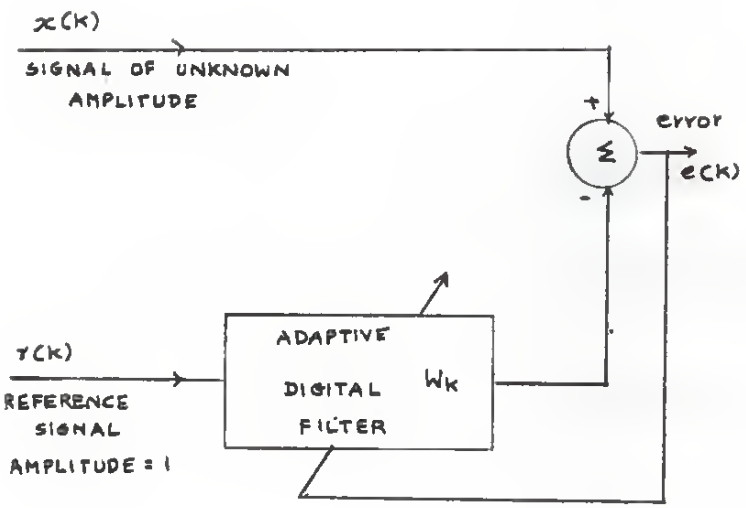


Fig.9.1.A method to find the RMS value of a sine wave.

root of $Re(H(w_o))^2 + Im(H(w_o))^2$.

Since g(k) cancels x(k) at the summing device the magnitude of x(k) is given by the magnitude of the transfer function.

This actually gives the maximum value of the signal. So the maximum value divided by a factor of root 2 would yield the RMS value of x(k). So this is one possible method to obtain the RMS value of a sinusoidal signal given its samples.

Other applications would be in other measurements and in transducer applications. The output of a transducer can be sampled and analysed in the frequency domain to obtain certain important characteristics like frequency content etc. Also after processing the sampled signal, the digital output can be reconverted to analog form by the use of DAC and used. Real time signal processing is now in the limelight.

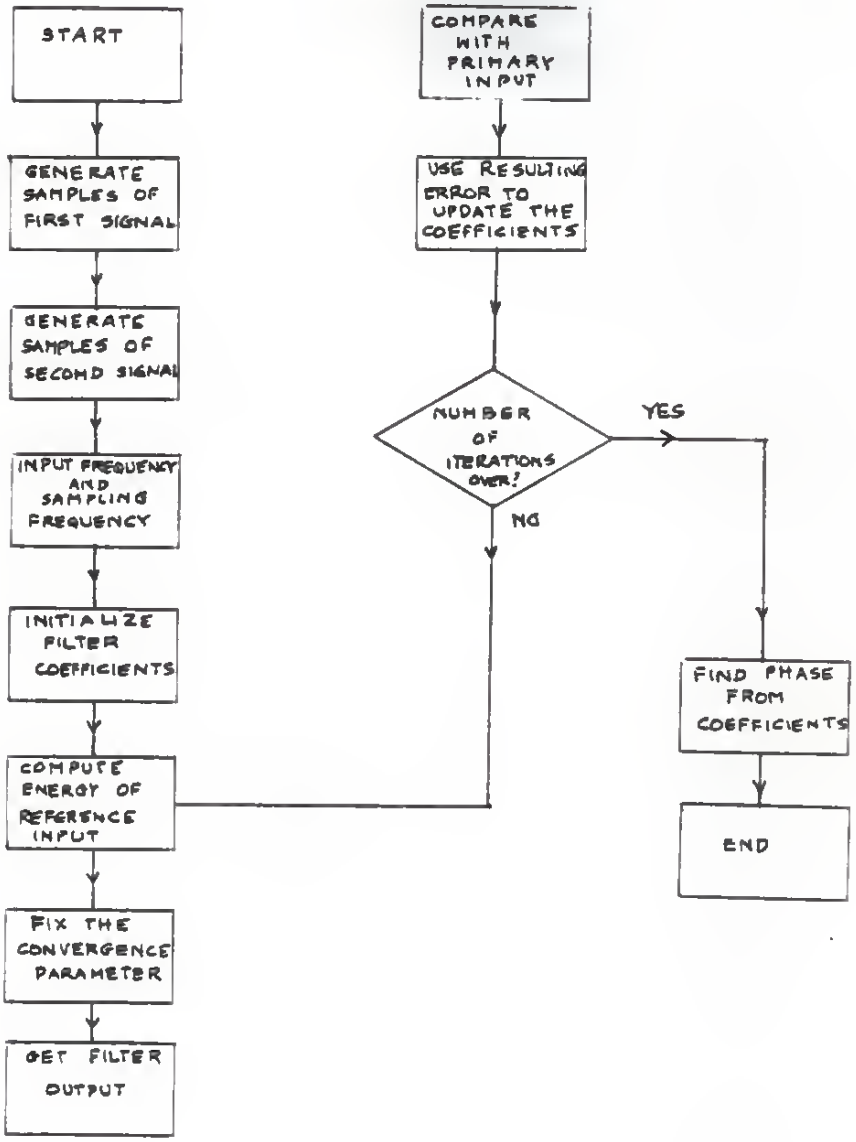
CHAPTER 10

PLOT PACKAGE

A plot package was developed for the HP 9845B which could be useful in plotting data. This data may be available on a floppy disk and the program can be modified to access this data from the disk for subsequent plotting of the data. The program may also be included in the main program as a subroutine and accessed whenever a plot is required. The program listing is given in Appendix 9. The program automatically finds the maximum and minimum values in the data block and plots the data and labels the axes etc. It can be used for plotting random data also. It was used extensively used to plot data obtained from simulation analysis of the different algorithms. It is capable of further modifications and improvements to achieve greater sophistication.

APPENDIX 1 a.

Flow chart for BASIC program. (Adaptive algorithm)



APPENDIX 1 b.

BASIC program listing. (Simulation)

```
10    REM ************************************************************
20    REM THIS IS A SIMULATION PROGRAM WHICH GENERATES TWO SINE
30    REM WAVES WITH A KNOWN PHASE DIFFERENCE. THE FREQUENCY OF
40    REM THE SIGNALS AND THE SAMPLING FREQUENCY CAN BE INPUT TO
50    REM THE PROGRAM AND IT USES THE ADAPTIVE FILTERING SCHEME
60    REM (WIDROW'S LMS ALGORITHM) TO COMPUTE THE PHASE DIFFERENCE.
70    REM THIS VALUE CAN BE COMPARED WITH THE ACTUAL PHASE INPUT TO
80    REM THE PROGRAM.THE AMPLITUDE OF THE SIGNALS CAN BE CHANGED IF
90    REM NECESSARY. ALSO THE AMPLITUDE TRANSFER FUNCTION AT THE
100   REM SIGNAL FREQUENCY IS ALSO DETERMINED.
110   REM ************************************************************
120   REM FORM THE ARRAYS FOR THE FILTER INPUT X(*) AND THE FILTER
130   REM WEIGHTS W(*)
140   REM ************************************************************
150   DIM X(16),W(16)
160   REM ************************************************************
170   REM INPUT THE VALUES OF THE SIGNAL FREQUENCY(F),THE SAMPLING
180   REM FREQUENCY(F1) AND THE REQUIRED PHASE DIFFERENCE(P) BETWEEN
190   REM THE GENERATED SIGNALS
200   REM ************************************************************
210   INPUT F,F1,P
220   F0=2*PI*F
230   T=1/F1
240   P0=P*PI/180
250   REM ************************************************************
260   REM INITIALISE COEFFS. OF THE FILTER
270   REM ************************************************************
280   REM ************************************************************
290   REM FIX THE NUMBER OF WEIGHTS(COEFFICIENTS OF THE FILTER)(N1
300   REM ************************************************************
310   N=10
320   FOR I=1 TO N
330   W(I)=0
340   NEXT I
350   REM ************************************************************
360   REM FIX THE NUMBER OF ITERATIONS
370   REM ************************************************************
380   N1=50
390   REM ************************************************************
400   REM FIX THE VALUE OF CONVERGENCE PARAMETER (V1)
410   REM ************************************************************
420   V1=.002
430   REM ************************************************************
440   REM START PROCESSING
450   REM ************************************************************
460   FOR K=1 TO N1
470   REM ************************************************************
480   REM GEN Y THE PRIMARY INPUT TO THE FILTER
490   REM ************************************************************
500   Y=SIN(F0*K*T+P0)
510   REM ************************************************************
520   REM GEN X1 THE REFERENCE INPUT TO THE FILTER
530   REM ************************************************************
540   X1=SIN(F0*K*T)
550   REM ************************************************************
560   REM ESTIMATE THE INPUT SIGNAL EMERGY
570   REM TO UPDATE    THE CONVERGENCE PARAMETER V
580   REM ************************************************************
590   V2=V2*(1-V1)+V1*X1*X1
600   IF V2<.001 THEN 630
```

```
610   V=V1/V2
620   GOTO 650
630   V=0
640   REM ************************************************************
650   REM FIND THE N PAST VALUES OF X1 TO FORM X THE INPUT TO THE FILTER
660   REM TO FIND THE FILTER OUTPUT G
670   REM ************************************************************
680   FOR J=1 TO N
690   M=J-1
700   X(J)=SIN(F0*T*(K-M))
710   NEXT J
720   G=0
730   REM ************************************************************
740   REM FIND THE FILTER OUTPUT G
750   REM ************************************************************
760   FOR L=1 TO N
770   G1=W(L)*X(L)
780   G=G+G1
790   NEXT L
800   REM ************************************************************
810   REM COMP ERROR BETWEEN THE PRIMARY OUTPUT AND THE FILTER OUTPUT
820   REM ************************************************************
830   E=Y-G
840   PRINT E,
850   REM ************************************************************
860   REM UPDATE COEFFICIENTS OF THE FILTER USING THE ERROR E
870   REM ************************************************************
880   FOR I=1 TO N
890   W(I)=W(I)+V*E*X(I)
900   NEXT I
910   NEXT K
920   PRINT " "
930   PRINT "COEFFICIENTS"
940   FOR I=1 TO N
950   REM PRINT COEFFS.
960   PRINT W(I),
970   NEXT I
980   REM ************************************************************
990   REM FIND PHASE FROM COEFFS. OF THE FILTER. S IS THE REAL PART OF
1000  REM THE TRANSFER FUNCTION AND Z IS THE IMAGINARY PART.
1010  REM ************************************************************
1020  S=0
1030  FOR I=1 TO N
1040  S=S+W(I)*COS(F0*T*(I-1))
1050  NEXT I
1060  Z=0
1070  FOR J=1 TO N
1080  Z=Z+W(J)*SIN(F0*T*(J-1))
1090  NEXT J
1100  Z=-Z
1110  R=Z/S
1120  PRINT "RESULT ",R
1130  C=ATN(R)
1140  C=C*180/PI
1150  REM ************************************************************
1160  REM ADJUST THE VALUE OF THE OBTAINED PHASE TO COVER THE PHASE RANGE
1170  REM FROM 0 THRO 360 USING THE INFORMATION OF THE SIGN OF S AND Z
1180  REM ************************************************************
1190  IF (S<0) AND (Z)0) THEN C1=180+C
1200  IF (S<0) AND (Z<0) THEN C1=180+C

1210  IF (S)0) AND (Z<0) THEN C1=360+C
1220  IF (S)0) AND (Z)0) THEN C1=C
1230  REM ************************************************************
1240  REM PRINT THE VALUE OF PHASE ON THE TERMINAL
1250  REM ************************************************************
1260  PRINT "PHASE=",C
1270  REM ************************************************************
1280  REM PRINT THE VALUE OF ACTUAL PHASE TO BE COMPARED WITH THE
1290  REM CALCULATED VALUE
1300  REM ************************************************************
1310  PRINT "THE ACTUAL PHASE DIFFERENCE IS",P
1320  REM ************************************************************
1330  REM FIND THE AMPLITUDE TRANSFER FUNCTION
1340  REM ************************************************************
1350  Ampl=SQR(Z*Z+S*S)
1360  PRINT "AMPLITUDE TFN=",Ampl
1370  END
```

APPENDIX 1 c.

BASIC program listing (Actual data samples)

```
10 REM ************************************************************
20 REM THIS PROGRAM DETERMINES THE PHASE DIFFERENCE BETWEEN TWO
30 REM SIGNALS BY USING THE ADAPTIVE FILTER ALGORITHM.
40 REM ACTUAL SAMPLES OF THE TWO SIGNALS ARE TAKEN USING THE
50 REM ADC CARD IN THE HP MULTIPROGRAMMER AND SENT TO THE
60 REM NORTH STAR FOR PROCESSING BY THE ALGORITHM
70 REM THE SAMPLES OF THE FIRST SIGNAL ARE TAKEN INDICATED
80 REM BY THE FLICKERING OF LEDS ON THE HPM PANEL
90 REM THE SAMPLES ARE THEN SENT TO THE NS IN APPROX 30 SECS
100 REM AFTER WHICH THE SAMPLES OF THE SECOND SIGNAL ARE TAKEN
110 REM INDICATED BY THE LEDS ON THE HPM PANEL.
120 REM AFTER PROCESSING BY THE ALGORITHM THE RESULTS ARE DISPLAYED
130 REM ON THE SILENT TERMINAL.
140 REM IT IS ASSUMED THE FIRST SIGNAL LEADS THE SECOND SIGNAL
150 REM IN PHASE... THE RESULTS SHOW A VALUE OF 360-P WHERE
160 REM P IS THE PHASE DIFFERENCE IF THE SIGNALS ARE INTERCHANGED
170 REM ************************************************************
180 REM FORM THE ARRAYS FOR STORING THE SAMPLES OF THE TWO SIGNALS
190 REM************************************************************
200 REM Y AND X1 ARE USED FOR THIS
210 REM X USED INITIALLY FOR STORING SAMPLES AND THEN THIS IS
220 REM STORED IN X1, X IS THEN USED FOR CURRENT PROCESSING
230 DIM X(512),Y(512),W(16),X1(512)
240 REM************************************************************
250 REM CLEAR THE ACIA IN INTERFACE(MC 6802)
260 REM************************************************************
270 FOR I=1 TO 10
280 INPUT#1,D$
290NEXT I
300 REM************************************************************
310 REM SEND COMMAND TO INVOKE THE ADC SUBROUTINE IN INTERFACE
320 REM************************************************************
330 PRINT#1,"A"
340 REM************************************************************
350 REM GET THE 512 SAMPLES AND STORE IN Y(*)
360 REM************************************************************
370 FOR I=1 TO 512
380 INPUT#1,D$
390Y(I)=VAL(D$)
400 NEXT I
410 REM************************************************************
420 REM CLEAR THE ACIA AGAIN
430 REM************************************************************
440  FOR I=1 TO 10
450  INPUT#1,D$
```

```
460  NEXT I
470 REM*****************************************************************
480 REM ISSUE COMMAND TO SAMPLE THE SECOND SIGNAL
490 REM*****************************************************************
500  PRINT#1,"A"
510 REM*****************************************************************
520 REM GET 512 SAMPLES OF SECOND SIGNAL
530 REM*****************************************************************
540  FOR I= 1 TO 512
550  INPUT#1,D$
560  X(I)=VAL(D$)
570  NEXT I
580 REM*****************************************************************
590 REM SCALING THE SAMPLES TO OBTAIN THE ACTUAL DECIMAL VALUES
600 REM FOR THE FIRST SIGNAL
610 REM*****************************************************************
620FOR L=1 TO 512
630 IF Y(L)(2048 THEN GOTO 650
640 Y(L)=Y(L)-4096
650 Y(L)=Y(L)*.005
660 NEXT L
670 REM*****************************************************************
680 REM PRINT ABOUT 25 SAMPLES OF FIRST SIGNAL
690 REM*****************************************************************
700 FOR L=1 TO 25
710 ! Y(L),
720 NEXT L
730 PRINT"/"
740 REM*****************************************************************
750 REM  SCALING FOR THE SECOND SIGNAL
760 REM*****************************************************************
770 FOR  I=1 TO 512
780 IF X(I)(2048 THEN GOTO 800
790 X(I)=X(I)-4096
800 X(I)=X(I)*.005
810 NEXT I
820 REM*****************************************************************
830 REM PRINT 25 SAMPLES OF THE SECOND SIGNAL
840 REM*****************************************************************
850 FOR I=1 TO 25
860 ! X(I),
870 NEXT I
880 REM*****************************************************************
890 REMSCALE THE AMPLITUDES OF THE SIGNALS TO 1V
900 REM*****************************************************************
910 FOR I= 1 TO 512
```

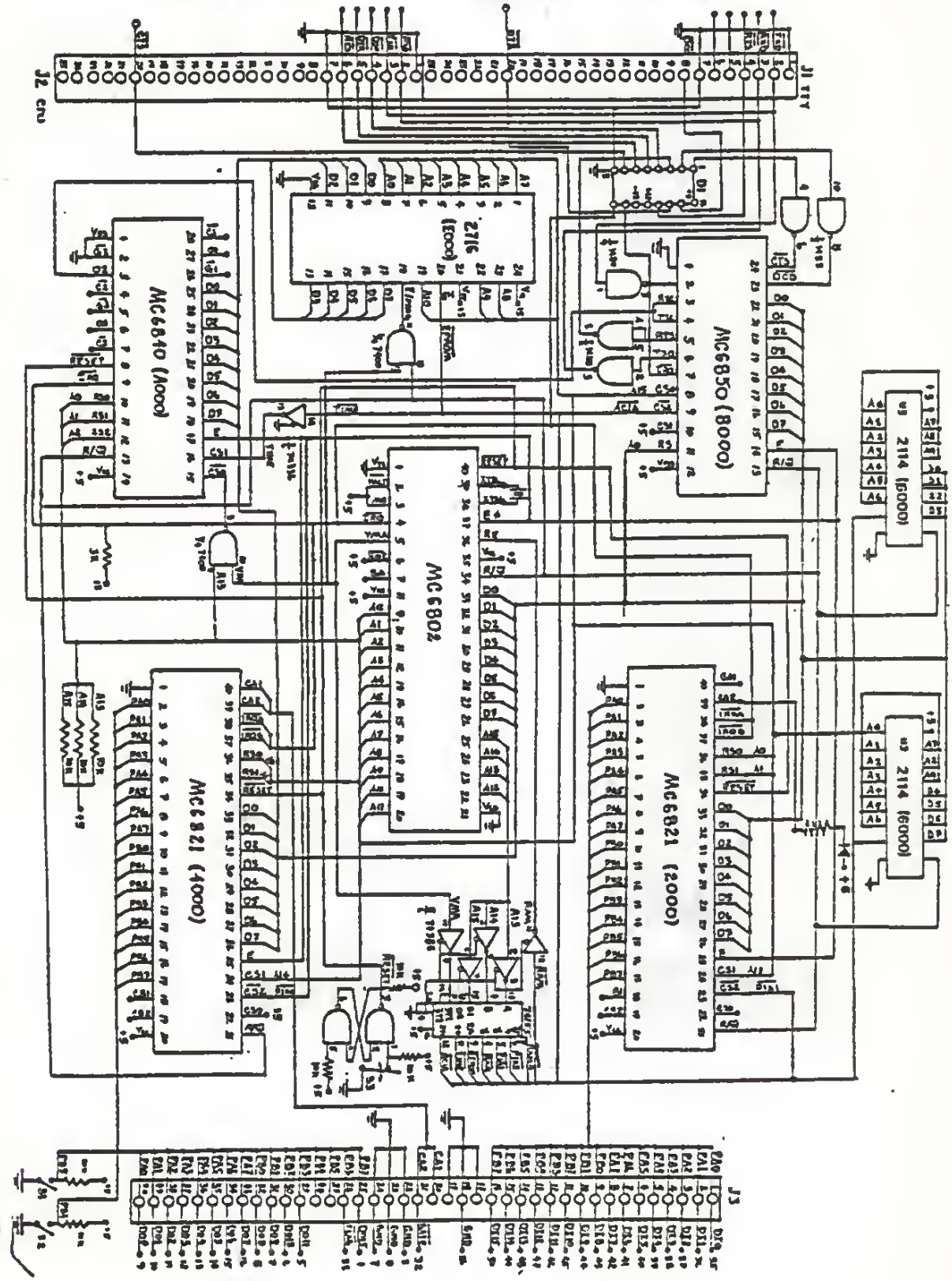
```
920Y(I)=.1*Y(I)
930 X1(I)=X(I)*.1
940NEXT I
950 :" "
960 REM*****************************************************************
970 REM GET THE FREQUENCY OF THE SIGNALS
980 REM*****************************************************************
990 :"FREQ",
1000INPUT F
1010 REM*****************************************************************
1020 REM START PROCESSING
1030 REM*****************************************************************
1040 F0=2*(22/7)*F
1050 F1=9130
1060 T=1/9.13E3
1070 REM*****************************************************************
1080 REM FIX THE VALUES FOR THE NUMBER OF WEIGHTS  FOR THE
1090 REM DIGITAL FILTER
1100 REM FIND THE RATIO OF SAMPLING FREQUENCY TO THE SIGNAL FREQUENCY
1110 REM*****************************************************************
1120 IF F1/F (=15 THEN   GOSUB 1160
1130 IF(F1/F (=11 AND F1/F)15) THEN GOSUB 1200
1140 IF (F1/F(=9 AND F1/F)11) THEN GOSUB 1240
1150 IF (F1/F(=5 AND F1/F)9) THEN GOSUB 1280
1160 N=8
1170 V1=.006
1180 N1=200
1190 RETURN
1200 N=4
1210 V1=.004
1220 N1=300
1230 RETURN
1240 N=4
1250 V1=.003
1260 N1=400
1270 RETURN
1280 N=3
1290 V1=.003
1300 N1=400
1310 RETURN
1320REM*****************************************************************
1330 REM START PROCESSING USING THE DETERMINED VALUES OF
1340 REM THE WEIGHTS(N), CONVERGENCE PARAMETER(V1),AND
1350 REM THE NUMBER OF ITERATIONS(N1)
1360REM*****************************************************************
1370 INITIALIZE THE WEIGHTS(COEFFICIENTS) OF THE FILTER
```

```
1380REM*****************************************************************
1390 FOR I= 1 TO 16
1400 W(I) = 0
1410 NEXT I
1420 FOR K=N TO N1
1430REM*****************************************************************
1440 REM FIND INPUT SIGNAL ENERGY TO UPDATE THE
1450 REM CONVERGENCE PARAMETER
1460REM*****************************************************************
1470 V2=V2*(1-V1)+V1*X1(K)*X1(K)
1480 IF V2<.001 THEN 1510
1490 V=V1/V2
1500 GOTO 1530
1510 V=0
1520REM*****************************************************************
1530REM GEN.X
1540REM*****************************************************************
1550  FOR J=1 TO N
1560  M=J-1
1570 X(J)=X1(K-M)
1580  NEXT J
1590  G=0
1600REM*****************************************************************
1610 REM FIND THE OUTPUT OF THE FILTER G
1620REM*****************************************************************
1630  FOR L= 1 TO N
1640  G1=W(L)*X(L)
1650  G=G+G1
1660  NEXT L
1670REM*****************************************************************
1680 REM COMP. ERROR BETWEEN PRIMARY INPUT & FILTER OUTPUT
1690REM*****************************************************************
1700 E=Y(K)-G
17.0  FOR I=1  TO N
1720REM*****************************************************************
1730 REM UPDATE COEFFS.
1740REM*****************************************************************
1750  W(I)=W(I)+V*E*X(I)
1760  NEXT I
1770  NEXT K
1780  ! "/"
1790  !"COEFFICIENTS"
1800 FOR I=1  TO N
1810 REM PRINT COEFFS.
1820 ! W(I).
1830 NEXT I
```

```
1840REM*************************************************************
1850 REM FIND PHASE FROM COEFFS.
1860REM*************************************************************
1870 REM S IS REAL PART OF THE TRANSFER FUNCTION
1880REM*************************************************************
1890 S=0
1900 FOR I=1  TO N
1910 S=S+W(I)*COS(F0*T*(I-1))
1920 NEXT I
1930REM*************************************************************
1940 REM Z IS THE IMAGINARY PART OF THE TRANSFER FUNCTION
1950REM*************************************************************
1960 Z=0
1970 FOR J=1  TO N
1980 Z=Z+W(J)*SIN(F0*T*(J-1))
1990 NEXT J
2000 Z=-Z
2010 R=Z/S
2020 ! "RESULT", R
2030 C=ATN(R)
2040 C=C*180/(22/7)
2050REM*************************************************************
2060 REM ADJUST THE VALUE OF THE OBTAINED PHASE TO
2070 REM COVER THE ENTIRE RANGE (0 THRO 360) USING THE
2080 REM INFORMATION FROM S AND Z SINCE THE FILTER CAN
2090 REM CAN DISTINGUISH A RANGE (0 THRO 90 ONLY)
2100 REM PRINT THE OBTAINED PHASE AFTER ADJUSTING ON THE SILENT
2110 REM TERMINAL
2120REM*************************************************************
2130 IF S<0 AND Z>0 THEN ! "THE PHASE DIFFERENCE IS",180+C
2140 IF S<0 AND Z<0 THEN ! "THE PHASE DIFFERENCE IS",C+180
2150 IF S>0 AND Z<0 THEN ! "THE PHASE DIFFERENCE IS",360+C
2160 IF S>0 AND Z>0 THEN ! "THE PHASE DIFFERENCE IS",C
2170REM*************************************************************
2180 REM PRINT THE ACTUAL VALUE OF PHASE FROM RC CKT.
2190 REM HERE THE VALUES OF R AND C ARE FIXED
2200REM*************************************************************
2210 !ATN(1/(F0*101.15E3*4.394E-9))*180/(22/7)
2220REM*************************************************************
2230 REM THE ACTUAL PHASE CAN BE COMPARED WITH THE OBTAINED PHASE
2240REM*************************************************************
2250 END
```

APPENDIX 2

Schematic diagram of the M6802 Based Interface



APPENDIX 3

6802 Monitor Program Listing.

```
                    MOTOROLA M68SAM CROSS-ASSMBLER          PAGE  1


          M68SAM 1S THE PROPERTY OF MOTOROLA SPO, INC.
                 COPYRIGHT 1974 BY MOTOROLA INC

          MOTOROLA M6800 CROSS ASSEMBLER, RELEASE 1.1

00001                              NAM     MICRO1
00002      0050     STAKPT EQU     $50
00003      A000     PTMC1  EQU     $A000
00004      A004     PTMB   EQU     $A004
00005      A001     PTMC2  EQU     $A001
00006      0000     INDL   EQU     $00
00007      00E0     INDH   EQU     $E0
00008      0011     MEM    EQU     $11
00009      0012     MEM1   EQU     $12
00010      2001     CRA1   EQU     $2001
00011      2003     CRB1   EQU     $2003
00012      2000     DDRA1  EQU     $2000
00013      2002     DDRB1  EQU     $2002
00014      4001     CRA2   EQU     $4001
00015      4003     CRB2   EQU     $4003
00016      4000     DDRA2  EQU     $4000
00017      4002     DDRB2  EQU     $4002
00018      8000     ACIAS  EQU     $8000
00019      8000     ACIAC  EQU     ACIAS
00020      8001     ACIARD EQU     $8001
00021      8001     ACIAT  EQU     ACIARD
00022      0010     CP     EQU     $10
00023      000E     VALL   EQU     $0E
00024      000F     VALH   EQU     $0F
00025      000C     TMPL   EQU     $0C
00026      0000     TMPH   EQU     $00
00027      0008     KNT    EQU     $08
00028      000A     MARK   EQU     $0A
00029      0006     LSD3   EQU     $06
00030      0007     LSD2   EQU     $07
00031      0008     LSD1   EQU     $08
00032      0009     LSD    EQU     $09
00033      0001     SUBHI  EQU     $01
00034      0000     SUBLO  EQU     $00
00035      0003     MINH1  EQU     $03
00036      0002     MINLO  EQU     $02
00037      0004     NEG    EQU     $04
00038      0005     CARRY  EQU     $05
00039      0014     TEMP   EQU     $14
00040      0016     TYPE   EQU     $16
00041      0017     TVALL  EQU     $17
00042      0018     TVALH  ECU     $18
00043      0019     FFLAG  EQU     $19
00044 E000                         ORG     $E000
00045 E000 00                      FCB     $00,$5F,$00,$17,$00,$05,$00,$02
      E001 5F
      E002 00
      E003 17
      E004 00
      E005 05
```

```
      E006 00
      E007 02
00046                 ********************************************************************
00047                 *                                                                  *
00048                 *          PIA,ACIA, AND PTM INITIALIZATION                        *
00049                 *                                                                  *
00050                 ********************************************************************
00051 E008 8E 0050          LDS    #STAKPT
00052 E00B 86 0C            LDA A  #$00
00053 E00D B7 2001          STA A  CRA1       POINT CRA1 AT DDRA1
00054 E010 B7 2003          STA A  CRB1
00055 E013 86 FF            LDA A  #$FF
00056 E015 B7 2000          STA A  DDRA1
00057 E018 B7 2002          STA A  DDRB1
00058 E01B 86 04            LDA A  #$04
00059 E01D B7 2003          STA A  CRB1
00060 E020 86 3C            LDA A  #$3C
00061 E022 B7 2001          STA A  CRA1
00062 E025 86 00            LDA A  #$00
00063 E027 B7 4001          STA A  CRA2
00064 E02A B7 4003          STA A  CRB2
00065 E02D B7 4000          STA A  DDRA2
00066 E030 B7 4002          STA A  DDRB2
00067 E033 86 04            LDA A  #$04
00068 E035 B7 4003          STA A  CRB2
00069 E038 86 3C            LDA A  #$3C
00070 E03A B7 4001          STA A  CRA2
00071 E03D 86 03            LDA A  #$03
00072 E03F B7 8000          STA A  ACIAC
00073 E042 86 01            LDA A  #$01
00074 E044 B7 8000          STA A  ACIAS
00075 E047 B6 4002          LDA A  DDRB2
00076 E04A 84 30            AND A  #$30
00077 E04C 44               LSR A
00078 E04D 44               LSR A
00079 E04E 44               LSR A
00080 E04F 8B 00            ADD A  #INDL
00081 E051 97 12            STA A  MEM1
00082 E053 B6 E0            LDA A  #INDH
00083 E055 89 00            ADC A  #$00
00084 E057 97 11            STA A  MEM
00085 E059 DE 11            LDX    MEM
00086 E05B EE 00            LDX    0,X
00087 E05D 86 93            LDA A  #$93
00088 E05F B7 A001          STA A  PTMC2
00089 E062 86 00            LDA A  #$00
00090 E064 B7 A000          STA A  PTMC1
00091 E067 FF A004          STX    PTMB
00092 E06A 86 83            LDA A  #$83
00093 E06C B7 A001          STA A  PTMC2
00094                 *
00095                 *
00096                 *
00097                 *
00098                 *
```

```
00099                   ********************************************************************
00100                   *                                                                  *
00101                   *        MAIN ROUTINE TO CHECK FOR VARIOUS COMMANDS $,?,#,%, ,'A'  *
00102                   *                                                                  *
00103                   ********************************************************************
00104 E06P 80 E18P LOOKCO JSR     GETONE    GET THE NEXT CHAR FROM NS WHEN SENT
00105 E0T2 81 26          CMP A   #$26
00106 E074 26 05          BNE     CKQM
00107 E076 80 E100        JSR     DOAT
00108 E079 20 F4          BRA     LOOKCO
00109 E078 81 3F    CKQM  CMP A   #$3F
00110 E07D 26 05          BNE     CKLB      IF NOT ? THEN CHECK IF I7 IS #
00111 E07F 8D E200        JSR     RONSNO    READ  MP   AND SEND  RESULT TO N.S7AR
00112 E082 20 E8          BRA     LOOKCO
00113 E084 81 23    CKLB  CMP A   #$23      CHECK IF CHAR EQUALS #
00114 E086 26 05          BNE     CKREL     IF NOT "#" THEN CHECK IF RELAY COM "%"
00115 E088 80 E400        JSR     TIMER     INI7IALIZE V/F CARD & TIME I7
00116 E08B 20 E2          BRA     LOOKCO
00117 E08D 81 25    CKREL CMP A   #$25      CHECK IF IT IS A RELAY COMMAND
00118 E08F 26 05          BNE     CKVCO     IF NOT  RELAY THEN CHECK IF IT IS VCO COM
00119 E091 8D E500        JSR     RELAY
00120 E094 20 09          BRA     LOOKCO
00121 E096 81 21    CKVCO CMP A   #$21      CHECK IF VCO COMMAND IE " "
00122 E098 26 05          BNE     CADC      IF NO7 VCO COMMAND THEN CHECK IF AOC COM
00123 E09A 80 E600        JSR     VCO
00124 E090 20 00          BRA     LOOKCO
00125 E09P 81 41    CAOC  CMP A   #$41      CHECK IF A/O CONVERTER COMMAND. IE "A"
00126 E0A1 26 CC          BNE     LOOKCO
00127 E0A3 80 E198        JSR     ADC
00128 E0A6 20 CT          BRA     LOOKCO
00129                   ********************************************************************
00130                   *                                                                  *
00131                   *                E N O   OF   MAIN  ROUTINE                        *
00132                   *                                                                  *
00133                   ********************************************************************
00134                   *
00135                   *
00136                   *
00137                   *
00138                   *
00139                   *
00140                   *
00141                   *
00142                   ********************************************************************
00143                   *
00144                   *     DOA7 SUBROU7INE  - TO ACCEPT CHARAC7ERS FROM N.STAR UNTIL $  *
00145                   *                        CONVERT 7O BINARY, AND SEND RESULT TO MP  *
00146                   *                                                                  *
00147                   ********************************************************************
00148 E100                      ORG     $E100
00149 E100 80 E108 DOAT   JSR     NSREAD    GET CHARS FROM N.S7AR UNTIL "$"
00150 E103 96 10          LDA A   CP        IF NOTHING SENT THEN EXIT
00151 E105 27 03          BEQ     XDOAT
00152 E107 80 E168        JSR     STOMP     SEND VALUE IN VALL & VALH TO MP
00153 E10A 39             XDOAT   RTS
```

```
00154                ********************************************************************
00155                *                                                                  *
00156                *     NSREAD - SUBROUTINE TO READ N.STAR & CONVERT TO BINARY       *
00157                *  THIS ROUTINE WILL RECEIVE ASCII DIGITS FROM THE N.STAR UNTIL A="$" *
00158                *  CONVERT  TO 16 BIT BINARY & STORE RESULT IN VALL & VALH.        *
00159                *                                                                  *
00160                ********************************************************************
00161 E108 86 00    NSREAD LDA A  #$00
00162 E10D 97 10           STA A  CP
00163 E10F 97 0E           STA A  VALL
00164 E111 97 0F           STA A  VALH
00165 E113 97 0C           STA A  TMPL
00166 E115 97 00           STA A  TMPH
00167 E117 97 08           STA A  KNT
00168 E119 4C              INC A
00169 E11A 97 0A           STA A  MARK
0016A E11C BD E18F LOOKC2  JSR    GETONE   GET THE NEXT CHAR FROM NS WHEN SENT
0016B E11F 81 24           CMP A  #$24
0016C E121 27 0B           BEQ    CHECK    IF "$" THEN GET OUT
0016D E123 80 30           SUB A  #$30
0016E E125 36              PSH A
0016F E126 7C 0010         INC    CP
00170 E129 20 F1           BRA    LOOKC2
00171 E12B 96 10    CHECK  LDA A  CP
00172 E12D 27 38           BEQ    XNSRD
00173 E12F 32       NEXT   PUL A
00174 E130 D6 08           LDA B  KNT
00175 E132 26 00           BNE    KNTPOS
00176 E134 97 0E           STA A  VALL
00177 E136 7C 0008         INC    KNT
00178 E139 96 10           LDA A  CP
00179 E13B 91 08           CMP A  KNT
0017A E13D 27 28           BEQ    XNSRD
0017B E13F 20 EE           BRA    NEXT
0017C E141 5F       KNTPOS CLR B
0017D E142 0C       X10    CLC
0017E E143 48              ASL A
0017F E144 59              ROL B
00180 E145 97 0C           STA A  TMPL
00181 E147 D7 00           STA B  TMPH
00182 E149 48              ASL A
00183 E14A 59              ROL B
00184 E14B 48              ASL A
00185 E14C 59              ROL B
00186 E14D 0C              CLC
00187 E14E 9B 0C           ADD A  TMPL
00188 E150 D9 00           ADC B  TMPH
00189 E152 7A 0008         DEC    KNT
0018A E155 26 EB           BNE    X10
0018B E157 9B 0E           ADD A  VALL
0018C E159 D9 0F           ADC B  VALH
0018D E15B 97 0E           STA A  VALL
0018E E15D D7 0F           STA B  VALH
0018F E15F 7C 000A         INC    MARK
00190 E162 96 0A           LDA A  MARK
```

```
00209 E164 9T 08          STA A  KNT
00210 E166 91 10          CMP A  CP
00211 E168 26 C5          BNE    NEXT
00212 E16A 39       XNSRD RTS
00213                     ********************************************************************
00214                     *      STOHP - SEND TO HP SUBROUTINE                               *
00215                     *  THIS SUBROUTINE WILL SEND 2 BYTES STORED IN VALL & VALH TO       *
00216                     *  THE 16 BIT IN PORT OF HP - VALH TO HIGHER BYTE & VALL TO LOWER... *
00217                     ********************************************************************
00218 E168 96 0E    STOHP LDA A  VALL
00219 E160 87 2000        STA A  DDRA1
00220 E170 96 0F          LDA A  VALH
00221 E172 8T 2002        STA A  DDRB1
00222 E175 02             NOP
00223 E176 02             NOP
00224 E177 86 4001        LDA A  CRA2
00225 E17A 84 F7          AND A  #$F7
00226 E17C 8T 4001        STA A  CRA2
00227 E17F 86 4002  LOOKC3 LDA A  DDRB2
00228 E182 84 40          AND A  #$40      CHECK FOR FLAG FROM MULTIPROGRAMMER
00229 E184 2T F9          BEQ    LOOKC3
00230 E186 86 4001        LDA A  CRA2
00231 E189 8A 08          ORA A  #$08
00232 E188 8T 4001        STA A  CRA2      WHEN FOUND, RESET THE GATE
00233 E18E 39       XSTOHP RTS
00234                     ********************************************************************
00235                     *
00236                     *   GETONE - SUBROUTINE TO GET 1 CHAR (IN REG A) FROM NS, WHEN SENT  *
00237                     *                                                                    *
00238                     ********************************************************************
00239 E18F B6 8000  GETONE LDA A  ACIAS
00240 E192 4T             ASR A
00241 E193 24 FA          BCC    GETONE    WAIT UNTIL A CHRACTER IS SENT
00242 E195 B6 8001        LDA A  ACIARD    READ IT INTO REG A WHEN SENT
00243 E198 39       XGETON RTS
00244                     ********************************************************************
00245                     *
00246                     *      ADC - SUBROUTINE TO COLLECT 512 (1/2 K ) SAMPLES AND        *
00247                     *      SEND THEM TO THE NORTH STAR.   THE ADC CARD IS ASSUMED TO    *
00248                     *      BE IN SLOT 405 OF THE MULTIPROGRAMMER                        *
00249                     *                                                                   *
00250                     ********************************************************************
00251 E198                      ORG    $E198
00252 E198 CE B0F0  ADC   LDX    #$B0F0   SEND CONTROL WORD  *1111 0000 1011 0000*
00253 E19B 0F 0E          STX    VALL
00254 E1A0 B0 E168        JSR    STOHP
00255 E1A3 CE 0050        LDX    #$0050   ADC CARD IS IN SLOT 405
00256 E1A6 DF 0E          STX    VALL
00257 E1A8 CE 6000        LDX    #$6000 <-- MODIFICATIONS START HERE.
00258 E1AB B0 E168 ANEXT  JSR    STOHP    ADDRESS SLOT 5 WITH GATE SIGNAL
00259 E1AE B6 4000        LDA A  DDRA2    GET THE SAMPLE
00260 E1B1 A7 00          STA A  0,X      STORE IT AT LOCATIONS 6000 ONWARDS
00261 E1B3 08             INX
00262 E1B4 B6 4002        LDA A  DDRB2    GET THE UPPER BYTE OF THE SAMPLE
00263 E1B7 A7 00          STA A  0,X
```

```
00264 E189 08              INX
00265 E18A 8C 6400         CPX    #$6400   COLLECT 512 SAMPLES IN ALL.  IE 1K BYTES
00266 E18D 26 EC           BNE    ANEXT
00267                *
00268                *
00269                *
00270 E18F CE 6000         LDX    #$6000   GET READY TO SEND THE SANPLES TO NORTH STAR
00271 E1C2 A6 00   AAGAIN  LDA A  0,X       GET THE SANPLE STORED IN MEMORY
00272 E1C4 43              COM A            TAKE 1'S CONPLEMENT OF THE LOWER BYTE
00273 E1C5 97 00           STA A  00
00274 E1C7 08              INX
00275 E1C8 A6 00           LDA A  0,X       GET THE HIGHER BYTE OF THE SAMPLE
00276 E1CA 43              COM A
00277 E1CB 84 0F           AND A  #$0F      MASK OFF THE HIGHER 4 BITS
00278 E1CD 97 01           STA A  01
00279 E1CF 08              INX
00280 E1D0 0F 15           STX    $15       SAVE THE SAMPLE POINTER
00281 E1D2 CE 003F         LDX    #$003F
00282 E1D5 BD E4TE         JSR    TIMOUT    WAIT FOR ACIA TO GET READY
00283 E1D8 BD E224         JSR    WSEND     SEND THE SANPLE
00284 E1DB BD E209         JSR    LOOKC1    SEND A CARRIAGE RETURN
00285 E1DE DE 15           LDX    $15
00286 E1E0 8C 6400         CPX    #$6400
00287 E1E3 26 00           BNE    AAGAIN    REPEAT UNTIL ALL 512 SAMPLES ARE SENT
00288 E1E5 39      XAOC    RTS
00289 E200                 ORG    $E200
00290                ********************************************************
00291                *                                                      *
00292                *     RONSND - READ AND SEND TO N.STAR SUBROUTINE      *
00293                *    THIS ROUTINE WILL READ HP ADDRESSED PORT AND SEND THE CORRESP  *
00294                *  DECINAL VALUE -I IN ASCII)- TO THE NORTH STAR FOLLOWED BY "CR"   *
00295                *                                                      *
00296                ********************************************************
00297 E200 CE 0034 RONSND LDX    #$0034   GET A DELAY COUNT
00298 E203 BD E4TE         JSR    TIMOUT    GIVE N.STAR HANDICAP OF 1/10 SEC
00299 E206 BD E216         JSR    IOAT      READ HP AND SEND TO N.STAR
00300 E209 B6 8000 LOOKC1  LDA A  ACIAS
00301 E20C 46              ROR A
00302 E20D 46              ROR A
00303 E20E 24 F9           BCC    LOOKC1
00304 E210 B6 00           LDA A  #$00
00305 E212 B7 8001         STA A  ACIAT
00306 E215 39      XRONSN  RTS
00307                ********************************************************
00308                *                                                      *
00309                *   ROUTINE TO READ 16 BITS OF HP, CONVERT 12 BIT BINARY TO BCD TO  *
00310                *    ASCII, AND SEND IT SERIALLY TO NORTH STAR         *
00311                *                                                      *
00312                ********************************************************
00313 E216 B6 4000 IOAT    LDA A  ODRA2     READ LOWER BYTE OF MULTIPROGRAMMER
00314 E219 43              COM A            1'S COMPLEMENT OF LOW BYTE
00315 E21A 97 00           STA A  SUBLO     LOW BYTE OF SUBTRAHEND
00316 E21C B6 4002         LDA A  ODRB2     READ UPPER BYTE OF MULTIPROGRAMMER
00317 E21F 43              COM A            1'S COMPLEMENT OF HIGH BYTE
00318 E220 84 0F           AND A  #$0F      MASK OFF UPPER 4 BITS
```

```
00319 E222 97 01          STA A  SUBHI    HIGH BYTE OF SUBTRAHEND
00320 E224 CE 0006 WSEND  LDX    #LS03
00321 E227 86 03          LDA A  #$03
00322 E229 C6 00          LDA B  #$00
00323 E22B E7 00    INC   STA B  0,X      CLEAR CURRENT DIGIT
00324 E22D 08             INX
00325 E22E 4A             DEC A
00326 E22F 2A FA          BPL    INC
00327 E231 86 E8          LDA A  #$E8
00328 E233 97 02          STA A  MINLO
00329 E235 86 03          LDA A  #$03
00330 E237 97 03          STA A  MINMI    LOAD 1000 INTO MINUEND
00331 E239 BD E30D POS1   JSR    SUB16    BRANCH TO 16 BIT SUBTRACTION ROUTINE
00332 E23C 96 04          LDA A  NEG      CHECK TO SEE IF SUBTRACTION RESULT IS NEGATIVE
00333 E23E 81 0C          CMP A  #$00
00334 E240 26 05          BNE    NEGR1
00335 E242 7C 0006        INC    LSD3
00336 E245 20 F2          BRA    POS1
00337 E247 96 00    NEGR1 LDA A  SUBLO    RESULT WAS NEGATIVE, ADD 1000 BACK
00338 E249 8B E8          ADD A  #$E8
00339 E24B 97 00          STA A  SUBLO
00340 E24D 96 01          LDA A  SUBHI
00341 E24F 89 03          ADC A  #$03
00342 E251 97 01          STA A  SUBHI
00343 E253 86 64          LDA A  #$64
00344 E255 97 02          STA A  MINLO
00345 E257 86 0C          LDA A  #$00
00346 E259 97 03          STA A  MINMI    LOAD 100 INTO MINUEND
00347 E25B BD E300 POS2   JSR    SUB16    BRANCH TO 16 BIT SUBTRACT ROUTINE
00348 E25E 96 04          LDA A  NEG
00349 E260 81 00          CMP A  #$00
00350 E262 26 05          BNE    NEGR2
00351 E264 7C 0007        INC    LSD2
00352 E267 20 F2          BRA    POS2     RESULT WAS POSITIVE, GO SUBTRACT ANOTHER 100
00353 E269 86 64    NEGR2 LDA A  #$64      RESULT WAS NEGATIVE, ADD 100 BACK
00354 E26B 9B 00          ADD A  SUBLO
00355 E26D 97 00          STA A  SUBLO
00356 E26F 96 01          LDA A  SUBHI
00357 E271 89 00          ADC A  #$00
00358 E273 97 01          STA A  SUBHI
00359 E275 86 0A          LDA A  #$0A      LOAD 10 INTO MINUEND
00360 E277 97 02          STA A  MINLO
00361 E279 7F 0003        CLR    MINMI
00362 E27C BD E300 POS3   JSR    SUB16    BRANCH TO 16 BIT SUBTRACT ROUTINE
00363 E27F 96 04          LDA A  NEG
00364 E281 81 00          CMP A  #$00      CHECK TO SEE IF THE RESULT WAS NEGATIVE
00365 E283 26 05          BNE    NEGR3
00366 E285 7C 0008        INC    LSD1
00367 E288 20 F2          BRA    POS3     RESULT WAS POSITIVE, SUBTRACT ANOTHER 10
00368 E28A 96 00    NEGR3 LDA A  SUBLO
00369 E28C 8B 0A          ADD A  #$0A      RESULT WAS NEGATIVE, ADD 10 BACK
00370 E28E 97 09          STA A  LSD       RESULT IS LSD
00371 E290 C6 04          LDA B  #$04
00372 E292 09       ASC1I DEX
00373 E293 86 30          LDA A  #$30
```

```
00374 E295 A8 00          ADD A  0,X      BCD TO ASCII
00375 E297 A7 00          STA A  0,X
00376 E299 5A             DEC B
00377 E29A 26 F6          BNE    ASCII
00378 E29C CE 0006        LDX    #LSD3    GET READY TO SEND IT BACK
00379 E29F C6 04          LDA B  #$04
00380 E2A1 D7 08          STA B  KNT      INITALIZE COUNTER KNT
00381 E2A3 A6 00    NEXT1 LDA A  0,X      COMPARE THE CHARACTER WITH "0"
00382 E2A5 81 30          CMP A  #$30
00383 E2A7 27 11          BEQ    ZERO
00384 E2A9 F6 8000 LOOKC4 LDA B  ACIAS    READ ACIA STATUS REG
00385 E2AC 56             ROR B
00386 E2AD 56             ROR B
00387 E2AE 24 F9          BCC    LOOKC4
00388 E2B0 B7 8001        STA A  ACIAT    WRITE CHARACTER TO ACIA
00389 E2B3 08             INX
00390 E2B4 7A 0008        DEC    KNT
00391 E2B7 26 11          BNE    PATCH    NO MORE ZEROS CAN BE SUPPRESSED
00392 E2B9 39       XIOAT RTS
00393 E2BA D6 08    ZERO  LDA B  KNT
00394 E2BC C1 01          CMP B  #$01
00395 E2BE 26 04          BNE    NOT
00396 E2C0 86 30          LDA A  #$30
00397 E2C2 20 E5          BRA    LOOKC4
00398 E2C4 7A 0008 NOT    DEC    KNT
00399 E2C7 08             INX
00400 E2C8 20 D9          BRA    NEXT1    ZEROS STILL BEING SUPPRESSED
00401 E2CA A6 00    PATCH LDA A  0,X      NO FURTHER ZERO SUPPRESSION
00402 E2CC 20 D8          BRA    LOOKC4
00403 E300                ORG    $E300
00404                     ********************************************************************
00405                     *                                                                  *
00406                     *      16 BIT SUBTRACTION SUBROUTINE                               *
00407                     *                                                                  *
00408                     ********************************************************************
00409 E300 96 03    SUB16 LDA A  MINHI
00410 E302 36             PSH A
00411 E303 96 02          LDA A  MINLO
00412 E305 36             PSH A           PUT HIGH AND LOW BYTE OF MINUEND ON THE STACK
00413 E306 7F 0005        CLR    CARRY
00414 E309 7F 0004        CLR    NEG
00415 E30C 43             COM A
00416 E30D 0C             CLC
00417 E30E 8B 01          ADD A  #$01     TWOS'S COMPLIMENT OF MINLO
00418 E310 24 03          BCC    L1
00419 E312 7C 0005        INC    CARRY
00420 E315 D6 03    L1    LDA B  MINHI
00421 E317 53             COM B
00422 E318 DB 05          ADD B  CARRY    TWO'S COMPLIMENT OF MINHI
00423 E31A 97 02          STA A  MINLO
00424 E31C D7 03          STA B  MINHI
00425 E31E 7F 0005        CLR    CARRY
00426 E321 0C             CLC
00427 E322 96 00          LDA A  SUBLO
00428 E324 98 02          ADD A  MINLO    SUBTRACT THE LOW BYTES
```

```
00429 E326 24 03          BCC    L2
00430 E328 TC 0005        INC    CARRY
00431 E328 06 01    L2    LDA B  SUBHI
00432 E320 08 03          ADD B  MINHI     SUBTRACT THE HIGH BYTES
00433 E32F 08 05          ADD B  CARRY
00434 E331 97 0C          STA A  SUBLO
00435 E333 0T 01          STA B  SUBHI     STORE THE RESULT
00436 E335 2A 03          BPL    L3
0043T E33T T3 0C04        COM    NEG       SET NEG TO FF IF THE RESULT IS NEGATIVE
00438 E33A 32       L3    PUL A
00439 E338 9T 02          STA A  MINLO
00440 E33D 32             PUL A
00441 E33E 9T 03          STA A  MINHI     RETRIEVE THE ORIGINAL MINUEND
00442 E340 39       XSUB16 RTS
00443 E400                ORG    $E400
00444                     ********************************************************************
00445                     *                                                                  *
00446                     *     ROUTINE TO MONITOR V/F CARD FOR SPECIFIED TIME  AND READ  IT *
00447                     *     BACK, AND SEND THE RESULT BACK TO THE NORTH STAR              *
00448                     *                                                                  *
00449                     ********************************************************************
00450 E400 7F 0016 TIMER  CLR    TYPE
00451 E403 B0 E18F        JSR    GETONE    GET THE NEXT CHAR FROM NS WHEN SENT
00452 E406 B0 30          SUB A  #$30      GET THE HEX EQUIVALENT
00453 E408 48             ASL A            AND MULTIPLY IT BY 2
00454 E409 CE E6EE        LDX    #CA00-2   GET INTO X ADDRESS TO 1ST COUNT MINUS 2
00455 E40C 0F 14          STX    TEMP
00456 E40E 9B 15          ADD A  TEMP+1
0045T E410 9T 15          STA A  TEMP+1
0C45B                  *
00459 E412 B0 E18F        JSR    GETONE    GET THE NEXT CHAR FROM NS WHEN SENT
0C460 E415 81 30          CMP A  #$30      CHECK IF IT EQUALS ZERO
00461 E41T 2T 04          BEQ    ARND
00462 E419 86 20          LDA A  #$20
0G463 E41B 9T 16          STA A  TYPE
00464 E41D CE 60F0 ARND   LDX    #$60F0    GET THE CONTROL WORD
00465 E420 0F 0E          STX    VALL
00466 E422 BD E16B        JSR    STDHP     SEND VALUE IN VALL & VALH TO HP
0046T                  *
00468 E425 86 00          LDA A  #$00      OD = 13 = SLOT ADDRESS FOR V/F CARD
00469 E42T 9T 0F          STA A  VALH
0C470 E429 86 05          LDA A  #$05      SEND A  05 TO  V/F CARD
004T1 E42B BD E4T6        JSR    SEND
004T2                  *
0G4T3 E42E 86 01          LDA A  #$01      SEND A 01 TO V/F CARD
004T4 E430 B0 E4T6        JSR    SEND
0C475                  *
00476 E433 86 09          LDA A  #$09      SEND A 09 TO V/F CARD
004T7 E435 BD E476        JSR    SEND
004TB                  *
00479 E438 0E 14          LDX    TEMP      GET THE POINTER TO THE COUNT
0C4B0 E43A EE 00          LDX    0,X       GET THE CORRESPONDING COUNT IN X
00481 E43C BD E4TE        JSR    TIMOUT
0C482                  *
00483 E43F C6 05          LDA B  #$05      7060 COUNTER TO BE READ 5 TIMES
```

```
00484 E441 07 14          STA B  TEMP
00485                *
00486 E443 B6 00    AGAIN LOA A  #$00      SEND A 13 TO V/F CARD
00487 E445 80 E476        JSR    SEND
00488                *
00489 E448 86 0F          LOA A  #$0F
00490 E44A 80 E476        JSR    SEND
00491                *
00492 E440 CE ACF0        LDX    #$A0F0    SEND ISL AND SYE FOR READING
00493 E450 0F 0E          STX    VALL
00494 E452 80 E168        JSR    STOHP     SEND VALUE IN VALL & VALH TO HP
00495                *
00496 E455 CE 0000        LDX    #$0000    TC TRIGGER  INPUT FROM V/F CARD
00497 E458 0F 0E          STX    VALL
00498 E45A B0 E168        JSR    STOHP     SEND A 9 TO THE V/F SLOT
00499                *
00500 E450 B0 E466        JSR    READ
00501 E460 7A 0014        DEC    TEMP
00502 E463 26 DE          BNE    AGAIN
00503 E465 39       XTIMER RTS
00504                ********************************************************************
00505                *                                                                  *
00506                *    ROUTINE USED BY TIMER TO READ A BYTE FROM V/F CARD & SEND TO NS *
00507                *                                                                  *
00508                ********************************************************************
00509 E466 B0 E200 READ   JSR    RDNSND    TIMEOUT, READ HP AND SEND TO NSTAR
00510 E469 CE 6CF0        LDX    #$60F0
00511 E46C 0F 0E          STX    VALL
00512 E46E B0 E168        JSR    STOHP     SEND VALUE IN VALL & VALH TO HP
00513 E471 B6 0C          LOA A  #$00
00514 E473 97 0F          STA A  VALH
00515 E475 39       XREAD RTS
00516                ********************************************************************
00517                *                                                                  *
00518                *      ROUTINE USED BY TIMER TO ADDRESS THE V/F SLOT               *
00519                *                                                                  *
00520                ********************************************************************
00521 E476 9A 16    SEND   ORA A  TYPE
00522 E478 97 0E          STA A  VALL
00523 E47A BD E168        JSR    STOHP     SEND A 5 TO THE V/F SLOT
00524 E470 39       XSEND RTS
00525                ********************************************************************
00526                *      TIMER SUBROUTINE TO TIMEOUT FOR 1/10, 1, OR  10 SEC DEPENDING *
00527                *                           ON THE CONTENTS OF REG X               *
00528                *                                                                  *
00529                ********************************************************************
00530 E47E 86 50    TIMOUT LDA A  #$50
00531 E480 4A       BACK   DEC A
00532 E481 26 F0          BNE    BACK
00533 E483 09             DEX
00534 E484 26 F8          BNE    TIMOUT
00535 E486 39       XTIMOU RTS
00536                *
00537                *
00538                *
```

```
00539                    *
00540                    ********************************************************************
00541                    *
00542                    *     RELAY  SUBROUTINE                                            *
00543                    *  THIS ROUTINE WILL ESTABLISH(WRITE) -OR- READ BACK RELAY POSITIONS *
00544                    * TO WRITE :- IT ACCEPTS NEW POSITION FROM N.STAR ,CONVERTS TO       *
00545                    *  BINARY, AND SENDS IT OUT ON SLOT A  (0001) OF HP INPUT PORT       *
00546                    *  TO  READ:- FETCHES CONTENTS OF SLOT A OF HP AND SENDS IT TO NS    *
00547                    *
00548                    ********************************************************************
00549 E500                          ORG     $E500
00550 E500 BD E18F       RELAY      JSR     GETONE     GET THE NEXT CHAR FROM NS WHEN SENT
00551 E503 81 57                    CMP A   #$57       IF CHARACTER IS NOT "W" THEN READ RELAY
00552 E505 26 20                    BNE     RDREL
00553 E507 BD E108                  JSR     NSREAD     ELSE GET NEW RELAY POSITIONS  FROM   NS
00554 E50A DE 0E                    LDX     VALL       SAVE THE RELAY POSITIONS FOR LATER USE
00555 E50C DF 17                    STX     TVALL      IF NOTHING SENT THEN OPEN ALL RELAYS
00556                    *
00557 E50E CE 70F0                  LDX     #$70F0     SEND OTE SYE AND TME TO HP
00558 E511 DF 0E                    STX     VALL
00559 E513 BD E168                  JSR     STOHF      SEND VALUE IN VALL & VALH TO HP
00560                    *
00561 E516 DE 17                    LDX     TVALL      GET THE SAVED RELAY POSITIONS
00562 E518 DF 0E                    STX     VALL
00563 E51A 96 0F                    LDA A   VALH
00564 E51C 84 0F                    AND A   #$0F
00565 E51E 8A 10                    ORA A   #$10       ATTACH SLOT A ADDRESS TO DATA
00566 E520 97 0F                    STA A   VALH
00567 E522 BD E168                  JSR     STOHF      SEND VALUE IN VALL & VALH TO HP
00568 E525 20 13                    BRA     XREL       DONE - SO EXIT FROM HERE
00569                    *
00570 E527 CE A0F0       RDREL      LDX     #$A0F0     SEND ISL AND SYE TO TRIGGER HP FOR INPUT
00571 E52A DF 0E                    STX     VALL
00572 E52C BD E168                  JSR     STOHF      SEND VALUE IN VALL & VALH TO HP
00573                    *
00574 E52F CE 0010                  LDX     #$0010     READ SLCT A (0001-CONTROL)
00575 E532 DF 0E                    STX     VALL
00576 E534 BD E168                  JSR     STOHP      SEND VALUE IN VALL & VALH TO HP
00577 E537 BD E200                  JSR     RDNSNO     READ RELAY AND SEND  RESULT TO N.STAR
00578 E53A 39            XREL       RTS
00579 E600                          ORG     $E600
00580                    ********************************************************************
00581                    *                        V C O   SUBROUTINE                        *
00582                    *  THIS SUBROUTINE WILL GET THE RANGE, FREQUENCY, AND AMPLITUDE      *
00583                    *  FROM THE NORTH STAR AND OUTPUT CONTROL TO THE VCO CARD IN SLOT K  *
00584                    ********************************************************************
00585 E600 BD E108       VCO        JSR     NSREAD     PERFORM GET CHARACTERS SUBROUTINE
00586 E603 96 0E                    LDA A   VALL       IF RANGE NOT BETWEEN 0 AND 3
00587 E605 84 03                    AND A   #$03       THEN RETAIN ONLY THE LEAST 2 BITS
00588 E607 48                       ASL A
00589 E608 48                       ASL A
00590 E609 8A 80                    ORA A   #$80       POSITION RANGE AND COMBINE IT WITH SLOT ADDR
00591 E60B 97 16                    STA A   TYPE       SAVE IT FOR LATER USE
00592 E60D 7F 0019                  CLR     FFLAG      FLAG INDICATING NO CHANGE IN FREQUENCY
00593 E610 BD E108       GETFRQ     JSR     NSREAD     GET THE MAGNITUDE OF FREQ IN BINARY
```

```
00594 E613 96 10          LDA A  CP
00595 E615 26 03          BNE    AROUN     IF NOTHING SENT THEN ASSUME
00596 E617 7C 0C19        INC    FFLAG     NO CHANGE IN FREQUENCY OF VCO
00597 E61A 96 0E    AROUN LDA A  VALL
00598 E61C 97 17          STA A  TVALL     SAVE IT FOR NOW AND
00599 E61E BD E108  GETAMP JSR   NSREAD    GET THE AMPLITUDE FROM NORTH STAR
00600 E621 96 10          LDA A  CP        IF NOTHING SENT THEN
00601 E623 27 1F          BEQ    NOCHNG    DO NOT CHANGE THE AMPLITUDE OF VCO OUTPUT
00602 E625 06 0E          LDA B  VALL      SAVE ONLY LAST 8 BITS OF AMPLITUDE
00603 E627 37             PSH B            ON STACK
00604 E628 CE 60F0        LDX    #$60F0    SEND  SYE AND DTE TO HP
00605 E628 0F 0E          STX    VALL
00606 E620 BD E168        JSR    STOHP     SEND VALUE IN VALL & VALH TO HP
00607               *
00608 E630 33             PUL B            GET AMPLITUDE IN VALL
00609 E631 02             NOP              MAY REQUIRE A COM B HERE
00610 E632 D7 0E          STA B  VALL
00611 E634 96 16          LDA A  TYPE      GET SLOT ADDRESS WITH RANGE
00612 E636 8A 01          ORA A  #$01      MAKE WR(AMP) LOW
00613 E638 97 0F          STA A  VALH
00614 E63A BD E168        JSR    STOHP     SEND VALUE IN VALL & VALH TO HP
00615               *
00616 E630 96 16          LDA A  TYPE
00617 E63F 97 0F          STA A  VALH      HOLD AMPLITUDE DATA
00618 E641 BD E168        JSR    STOHP     SEND VALUE IN VALL & VALH TO HP
00619               *
00620 E644 96 19    NOCHNG LDA A  FFLAG
00621 E646 26 10          BNE    XVCO      IF NO CHANGE IN FRQ THEN QUIT
00622 E648 CE 60F0        LDX    #$60F0
00623 E648 0F 0E          STX    VALL
00624 E640 BD E168        JSR    STOHP     SEND VALUE IN VALL & VALH TO HP
00625               *
00626 E650 96 17          LDA A  TVALL     GET THE FREQUENCY MAGNITUDE
00627 E652 43             COM A
00628 E653 97 0E          STA A  VALL
00629 E655 96 16          LDA A  TYPE      GET SLOT ADDRESS AND RANGEE
00630 E657 8A 02          ORA A  #$02      MAKE WR(FRQ) LOW
00631 E659 97 0F          STA A  VALH
00632 E658 BD E168        JSR    STOHP     SEND VALUE IN VALL & VALH TO HP
00633               *
00634 E65E 96 16          LDA A  TYPE
00635 E660 97 0F          STA A  VALH      HOLD FREQUENCY DATA
00636 E662 BD E168        JSR    STOHP     SEND VALUE IN VALL & VALH TO HP
00637 E665 39       XVCO  RTS
00638 E6F0                ORG    $E6F0
00639 E6F0 0288     CADD  FDB    $0288
00640 E6F2 1958           FDB    $1958
00641 E6F4 F070           FCB    $F070
00642 E7FE                ORG    $E7FE
00643 E7FE E0             FCB    $E0,$00
      E7FF 00
00644                     END
```

SYMBOL TABLE

STAKPT	0050	PTMC1	A000	PTMB	A004	PTMC2	A001	INOL	0000	INOH	00E0	MEM	0011	MEM1	C012	CRA1	2001
CRB1	2003	ORA1	2000	DDRB1	2002	CRA2	4001	LRB2	4003	ORA2	4000	ORB2	4002	ACIAS	8000	ACIAC	8000
ACIARC	8001	ACIAT	8001	CP	0010	VALL	000E	VALH	000F	TMPL	000C	TMPH	0000	KNT	0008	MARK	000A
LSO3	0006	LSO2	0007	LSO1	0008	LSO	0009	SUBHI	0001	SUBLO	0000	MINHI	0003	MINLO	0002	NEG	0004
CARRY	0005	TEMP	0014	TYPE	0016	TVALL	0017	TVALH	0018	FFLAG	0019	LOOKCO	E06F	CKOM	E078	CKLO	E084
CKREL	E080	CKVCO	E096	CAOC	E09F	OOAT	E100	XOOAT	E10A	NSREAO	E10B	LOOKC2	E11C	CHECK	E128	NEXT	E12F
KNTPOS	E141	XIO	E142	XNSRO	E16A	STOMP	E16B	LOOKC3	E17F	XSTOMP	E18E	GETONE	E18F	XGETON	E198	AOC	E198
ANEXT	E1AB	AAGAIN	E1C2	XAOC	E1E5	RONSNO	E200	LOOKC1	E209	XRONSN	E215	IOAT	E216	WSENO	E224	INC	E228
POS1	E239	NEGR1	E247	POS2	E25B	NEGR2	E269	POS3	E27C	NEGR3	E28A	ASCII	E292	NEXT1	E2A3	LOCKC4	E2A9
XIDAT	E2B9	ZERO	E2BA	NOT	E2C4	PATCH	E2CA	SUB16	E300	L1	E315	L2	E328	L3	E33A	XSUB16	E340
TIMER	E400	ARNO	E410	AGAIN	E443	XTIMER	E465	READ	E466	XREAO	E475	SENO	E476	XSENO	E470	TIMOUT	E47E
BACK	E480	XTIMOU	E486	RELAY	E500	RDREL	E52T	XREL	E53A	VCO	E600	GETFRQ	E610	AROUN	E61A	GETAMP	E61E
NCCING	E644	XVCO	E665	CAOO	E6F0												

MODIFICATIONS MADE IN THE EXISTING EPROM IN MC 6802 INTERFACE

The following modifications were made in the existing assembly program in the MC 6802 interface EPROM. The modifications were made in the ADC assembly subroutine so that the ADC could be triggered by an external signal and samples be taken. The memory address at which the modifications were made are clearly indicated. The original listing of the assembly is also shown. The new EPROM contents are also shown.

```
MEMORY LOCATION E1 A8
JMP     MODPRGMST     ! JUMP TO THE MODIFIED PROGRAM ADDRESS
MEMORY LOCATION E7 10
ORG     MODPRGMST     ! START OF THE MODIFIED PROGRAM AT E7 10
        LDX           #$6000
        SEI                        !SET INTERRUPT MASK
        LDA A         CRB2
        ORA A         #$01         ! SET BIT0 TO ENABLE INTERRUPT
        STA A         CRB2
        LDA A         DDRB2        ! CLEAR BIT7 OF CONTROL REG.
        CLI                        ! CLEAR INTERRUPT MASK
        WAI                        ! WAIT FOR INTERRUPT
        SEI                        ! SET INTERRUPT MASK
        JMP           RTNADDR      ! GO BACK TO THE ORIGINAL PRGM.
RTNADDR IS E1 AB                   ' CONTINUE ORIGINAL PRGM. AT
                                   ! RETURN ADDRESS E1 AB
```

As soon as an interrupt occure the microprocessor fetches the interrupt service vector and this is initialized at location E7 F8. The contents of thie location are E5 50 and the interrupt

service routine starts at this location ie. E5 50.

```
INTERRUPT SERVICE ROUTINE STARTS AT E5 50
ORG     INTROUTIN           ! START OF  INTERRUPT ROUTINE(E5 50)
        NOP                 ! NO OPERATION
        LDA A       DDRB2   ! CLEAR BIT7 OF CONTROL REG.
        LDA A       CRB2    ! LOAD ACC. WITH [CONTROL REG.B.]
        AND A       #$FE    ! CLEAR BIT0 TO DISABLE INTERRUPTS
        STA A       CRB2    !
        RTI                 ! RETURN FROM THE INTERRUPT ROUTINE
```

Thus these modifications made in the existing ADC subroutine enabled the use of a trigger reference signal to initiate sampling. The modified EPROM contents are shown on the next page.

CONTENTS OF THE MODIFIED EPROM

```
000  00 5F 00 17 00 05 00 02   BE 00 50 B6 00 B7 20 01
010  B7 20 03 86 FF B7 20 00   B7 20 02 B6 04 B7 20 03
020  B6 3C B7 20 01 B6 00 B7   40 01 B7 40 03 B7 40 00
030  B7 40 02 86 04 B7 40 03   86 3C B7 40 01 86 03 B7
040  80 00 B6 01 B7 B0 00 26   40 02 64 30 44 44 44 8B
050  00 97 13 B6 E0 B9 00 97   11 DE 11 EE 00 B6 93 B7
060  A0 01 B6 00 B7 A0 00 FF   A0 04 86 83 B7 A0 01 BD
070  E1 8F B1 26 26 05 BD E1   00 20 F4 81 3F 35 05 BD
080  E2 00 20 E8 B1 23 26 05   BD E4 00 20 E2 B1 25 26
090  05 BD E5 00 20 D9 81 21   26 05 BD E6 00 20 D0 81
0A0  41 36 CC BD E1 9B 20 C7   FF FF FF FF FF FF FF FF
0B0  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
0C0  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
0D0  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
0E0  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
0F0  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
100  BD E1 09 96 10 27 03 BD   E1 6B 39 B6 00 97 10 97
110  0E 97 0F 97 0C 97 0D 97   0B 4C 97 0A BD E1 BF B1
120  E4 27 08 B0 30 36 7C 00   10 2D F1 96 10 27 3B 3B
130  D6 0B 26 0D 97 0E 7C 00   0B 96 10 91 0B 27 2B 20
140  EE 5F 0C 4B 59 97 0C D7   0D 48 59 48 59 0C 9B 0C
150  D9 0D 7A 00 0B 26 EB 9B   0E D9 0F 97 0E D7 0F 7C
160  00 0A 96 0A 97 0B 91 10   26 C5 39 96 0E B7 20 00
170  96 0F B7 20 02 02 02 B6   40 01 B4 F7 B7 40 01 B6
180  40 02 84 40 27 F9 B6 40   01 BA 0B B7 40 01 39 B6
190  80 00 47 24 FA 86 80 01   39 FF FF CE B0 F0 DF 0E
1A0  BD E1 6B CE 00 50 DF 0E   7E E7 10 BD E1 6B B6 40
1B0  00 A7 00 0B B6 40 02 A7   00 08 BC 54 00 26 EC CE
1C0  60 00 A6 00 43 97 00 0B   A6 00 43 84 0F 97 01 0B
1D0  DF 15 CE 00 3F BD E4 7E   BD E2 24 BD E2 09 DE 15
1E0  8C 64 00 26 DD 39 FF CE   60 00 B6 40 03 2A FB B6
1F0  40 02 7E E1 AB FF FF FF   FF FF FF FF FF FF FF FF
200  CE 00 34 BD E4 7E BD E2   16 B6 B0 00 46 46 24 F9
210  B6 0D B7 80 01 39 B6 40   00 43 97 00 B6 40 02 43
220  B4 0F 97 01 CE 00 06 B6   03 C6 00 E7 00 0B 4A 2A
230  FA B6 E8 97 02 86 03 97   03 BD E3 00 96 04 B1 00
240  26 05 7C 00 06 20 F2 96   00 BB E8 97 00 96 01 B9
250  03 97 01 86 64 97 02 B6   00 97 03 BD E3 00 96 04
260  B1 00 26 05 7C 00 07 20   F2 86 64 33 00 97 00 96
270  01 B9 00 97 01 86 0A 97   02 7F 00 03 BD E3 00 9B
280  04 B1 00 25 05 7C 00 0B   20 F2 96 00 8B 0A 97 09
290  C6 04 09 86 30 AB 00 A7   00 3A 26 F6 CE 00 05 C6
2A0  04 D7 0B A6 00 81 30 27   11 F6 80 00 56 56 24 F9
2B0  B7 B0 01 0B 7A 00 0B 26   11 39 D6 0B C1 01 26 04
2C0  B6 30 20 E5 7A 00 0B 08   20 D9 A6 00 30 DB FF FF
2D0  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
2E0  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
2F0  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
300  96 03 36 96 02 36 7F 00   05 7F 00 04 43 0C 3B 01
310  24 03 7C 00 05 D6 03 53   DB 05 97 02 D7 03 7F 00
320  05 0C 96 00 9B 02 24 03   7C 00 05 D6 01 DB 03 DB
330  05 97 00 D7 01 2A 03 73   00 04 32 97 02 32 97 03
340  39 FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
350  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
360  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
370  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
380  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
390  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
3A0  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
3B0  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
```

```
3C0  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
3D0  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
3E0  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
3F0  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
400  7F 00 16 BD E1 8F 83 30   48 CE E6 EE DF 14 9B 15
410  97 15 BD E1 8F 81 30 27   04 8S 20 97 16 CE 60 F0
420  DF 0E BD E1 6B 86 D0 97   0F 86 05 BD E4 76 86 01
430  BD E4 76 86 09 BD E4 76   DE 14 EE 00 BD E4 7E C6
440  05 D7 14 86 0D BD E4 76   86 0F BD E4 76 CE A0 F0
450  DF 0E BD E1 6B CE 00 D0   DF 0E BD E1 6B BD E4 66
460  7A 00 14 26 DE 39 BD E2   00 CE 60 F0 DF 0E BD E1
470  6B 86 D0 97 0F 39 9A 16   97 0E BD E1 6B 39 86 5D
480  4A 26 FD 09 26 F8 39 FF   FF FF FF FF FF FF FF FF
490  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
4A0  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
4B0  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
4C0  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
4D0  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
4E0  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
4F0  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
500  BD E1 8F 81 57 26 20 BD   E1 0B DE 0E DF 17 CE 70
510  F0 DF 0E BD E1 6B DE 17   DF 0E 96 0F 84 0F 8A 10
520  97 0F BD E1 6B 20 13 CE   A0 F0 DF 0E BD E1 6B CE
530  00 10 DF 0E BD E1 6B BD   E2 00 39 FF FF FF FF FF
540  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
550  01 B6 40 02 B6 40 03 84   FE B7 40 03 3B FF FF FF
560  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
570  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
580  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
590  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
5A0  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
5B0  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
5C0  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
5D0  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
5E0  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
5F0  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
600  BD E1 0B 96 0E 84 03 48   48 8A B0 97 16 7F 00 19
610  BD E1 0B 96 10 26 03 7C   00 19 96 0E 97 17 9D E1
620  0B 96 10 27 1F D6 0E 37   CE 60 F0 DF 0E BD E1 6B
630  33 02 D7 0E 96 16 8A 01   97 0F BD E1 6B 96 16 97
640  0F BD E1 6B 96 19 26 1D   CE 60 F0 DF 0E BD E1 6B
650  96 17 43 97 0E 96 16 8A   02 97 0F 9D E1 6B 96 16
660  97 0F BD E1 6B 39 FF FF   FF FF FF FF FF FF FF FF
670  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
680  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
690  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
6A0  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
6B0  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
6C0  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
6D0  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
6E0  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
6F0  02 88 19 58 FD 7D FF FF   FF FF FF FF FF FF FF FF
700  02 88 19 58 FD 7D FF FF   FF FF FF FF FF FF FF FF
710  CE 60 00 0F B6 40 03 8A   01 B7 40 03 B6 40 02 0E
720  3E 0F 7E E1 AB FF FF FF   FF FF FF FF FF FF FF FF
730  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
740  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
750  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
760  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
770  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
780  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
790  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
7A0  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
7B0  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
7C0  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
7D0  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
7E0  FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
7F0  FF FF FF FF FF FF FF FF   E3 30 FF FF FF FF E0 00
```

APPENDIX 4

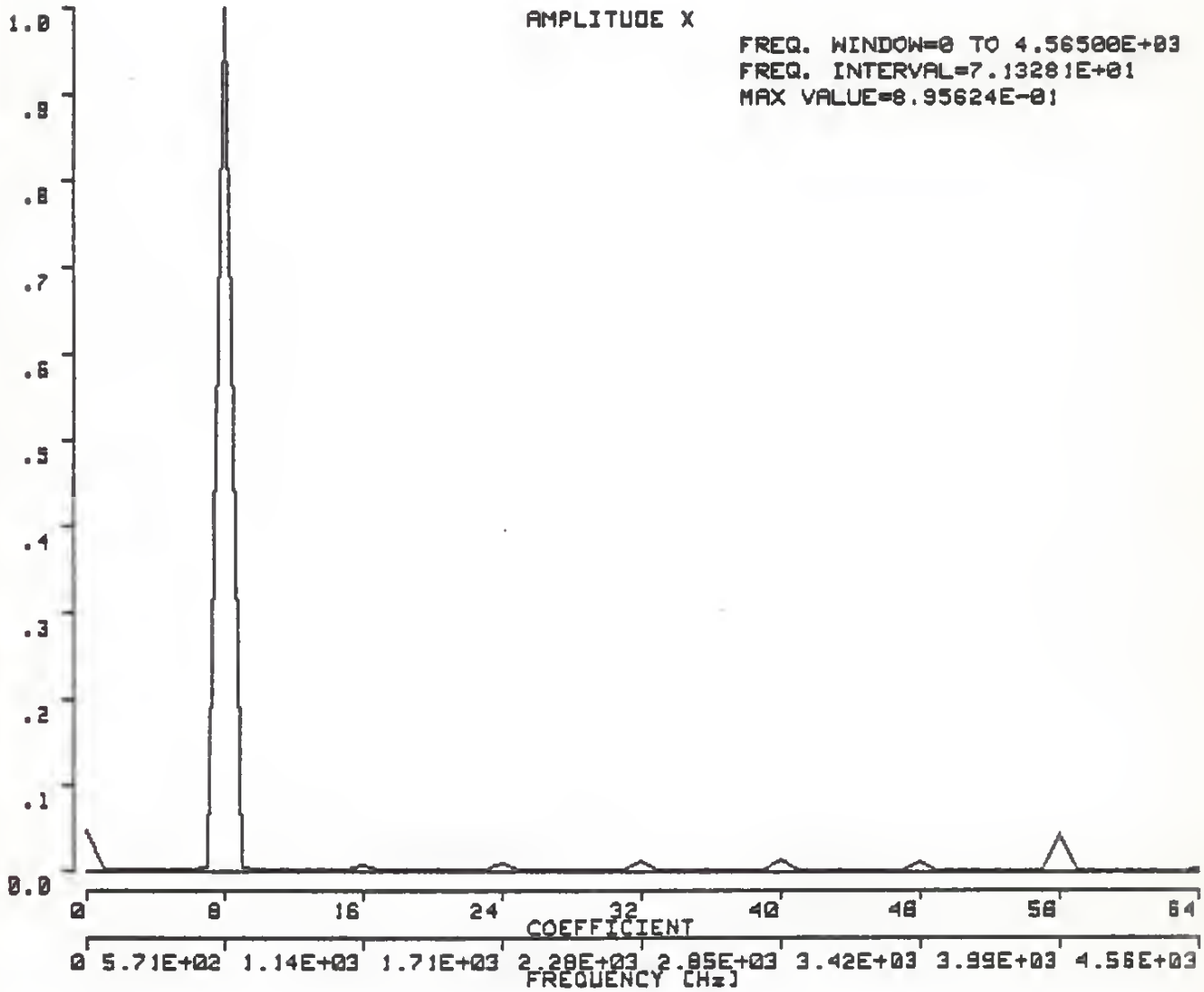
TABLE SHOWING THE READINGS OBTAINED FOR THE ADAPTIVE METHOD

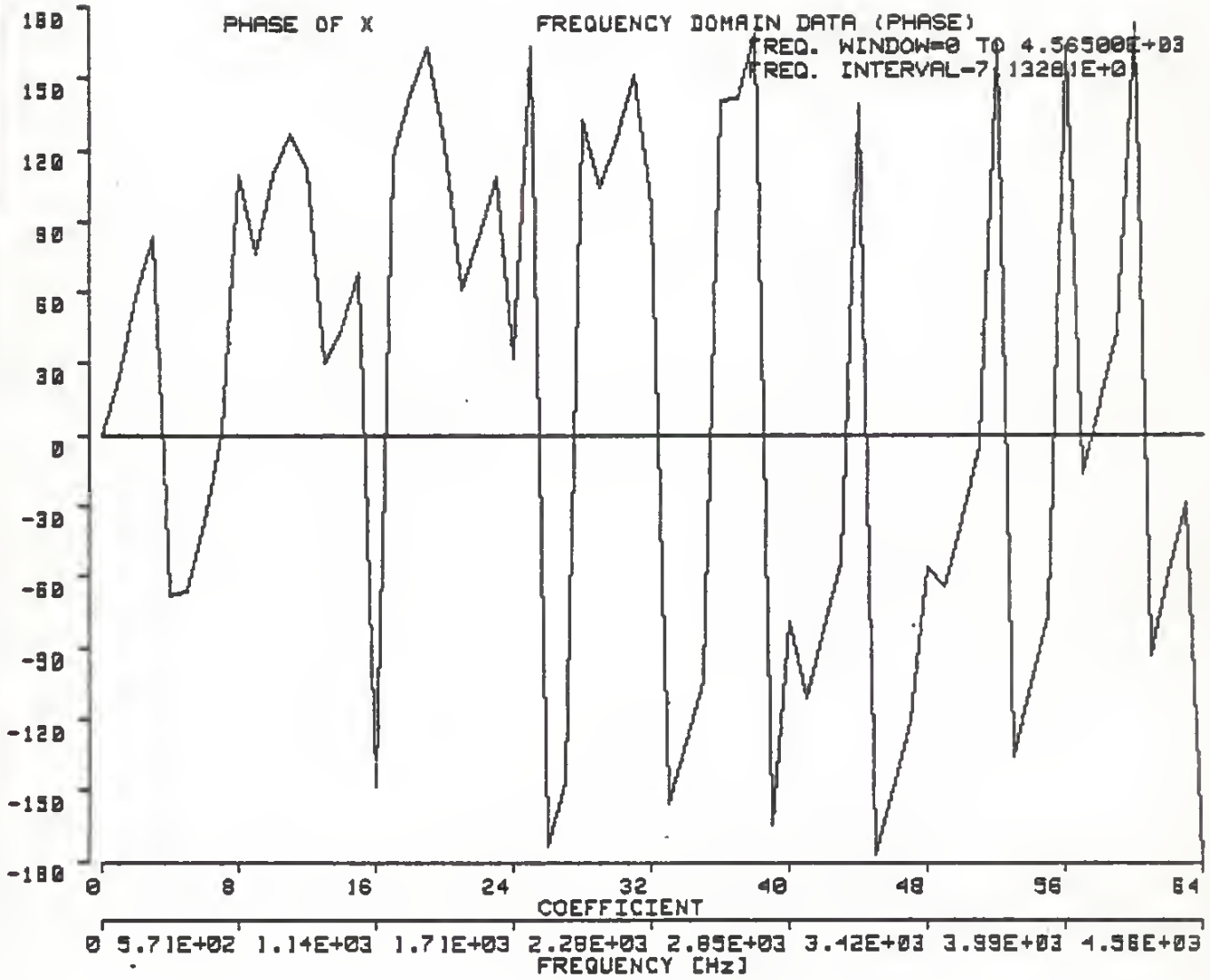
FREQUENCY F(Hz.)	CALCULATED PHASE P(Deg)	ACTUAL PHASE (Deg)	ERROR (Deg)
200	60.042437	60.78165	0.739213
300	50.935272	50.013108	0.922164
400	41.280897	41.807545	0.526648
500	34.611122	35.58437	0.973248
600	30.582739	30.807028	0.224289
700	26.862562	27.072203	0.209641
800	23.599043	24.095689	0.496646
900	21.049831	21.679956	0.630125
1000	19.271924	19.686781	0.414857
2000	10.005898	10.142929	0.137031

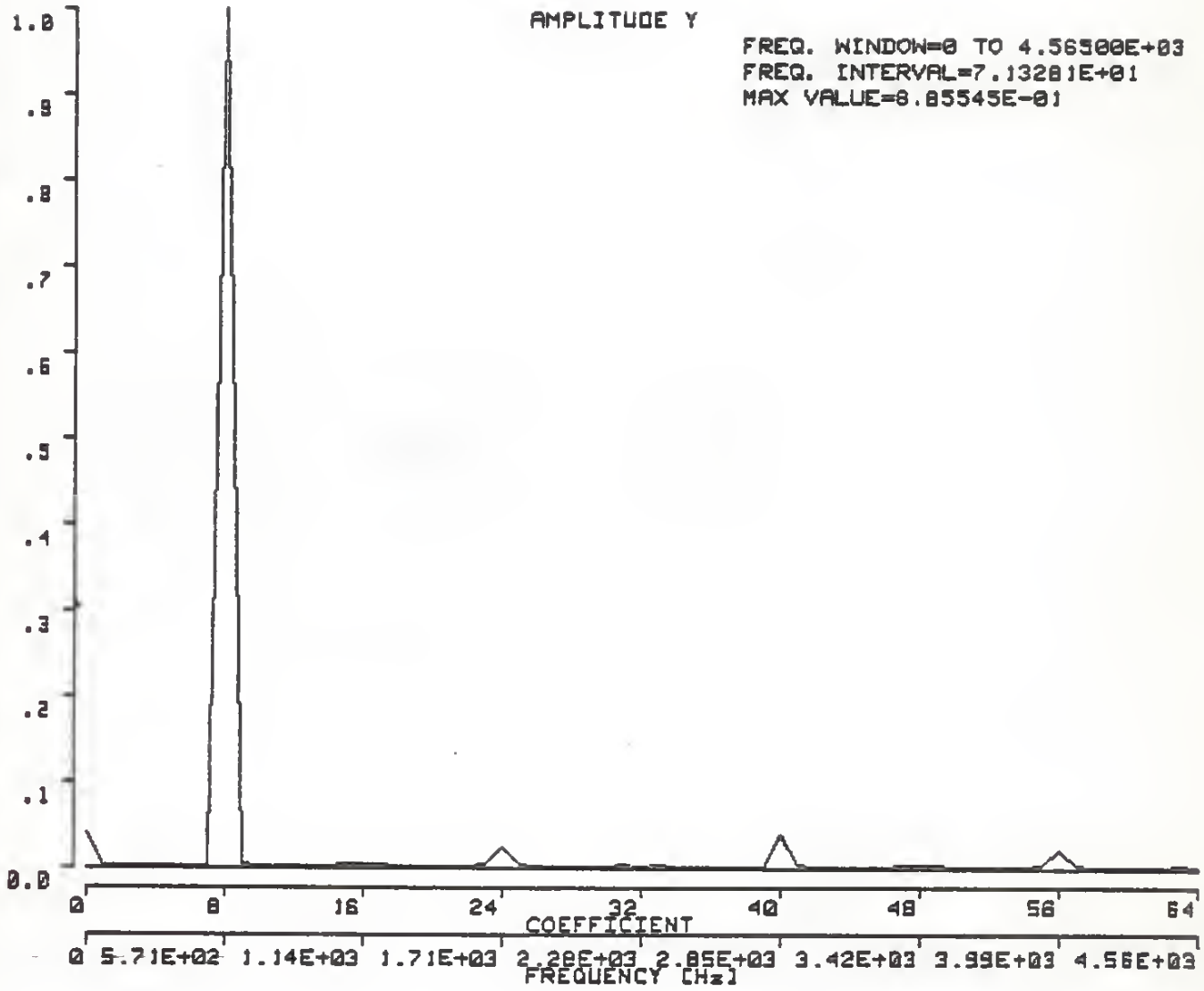
The actual phase was obtained from the RC circuit used. The values of R and C were found using a digital bridge.(Genrad RLC Digibridge). R=101.15K and C=4.394 nF.

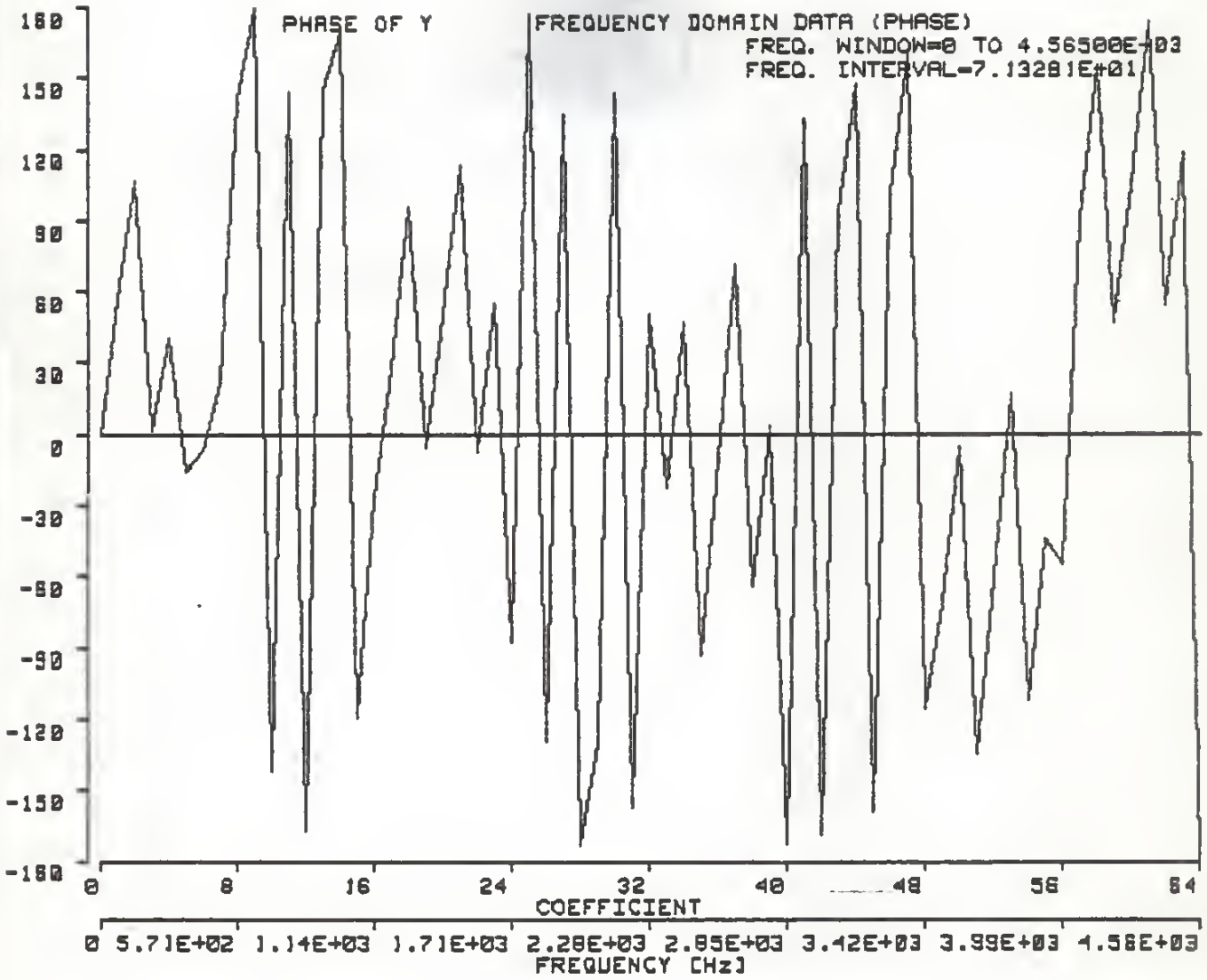
APPENDIX 5

Plots obtained for the FFT method with actual data samples.




PHASE OF X
FREQUENCY DOMAIN DATA (PHASE)
FREQ. WINDOW=0 TO 4.56500E+03
FREQ. INTERVAL=7.1328[1E+0]
180
150
120
90
60
30
0
-30
-60
-90
-120
-150
-180
0
8
16
24
32
40
48
56
64
COEFFICIENT
0 5.71E+02 1.14E+03 1.71E+03 2.28E+03 2.85E+03 3.42E+03 3.99E+03 4.56E+03
FREQUENCY [Hz]


AMPLITUDE Y
FREQ. WINDOW=0 TO 4.56500E+03
FREQ. INTERVAL=7.13281E+01
MAX VALUE=8.85545E-01
1.0
.9
.8
.7
.6
.5
.4
.3
.2
.1
0.0
0
8
16
24
32
40
48
56
64
COEFFICIENT
0
5.71E+02
1.14E+03
1.71E+03
2.28E+03
2.85E+03
3.42E+03
3.99E+03
4.56E+03
FREQUENCY [Hz]


PHASE OF Y
FREQUENCY DOMAIN DATA (PHASE)
FREQ. WINDOW=0 TO 4.56500E+03
FREQ. INTERVAL=7.13281E+01
180
150
120
90
60
30
0
-30
-60
-90
-120
-150
-180
0
8
16
24
32
40
48
56
64
COEFFICIENT
0 5.71E+02 1.14E+03 1.71E+03 2.28E+03 2.85E+03 3.42E+03 3.99E+03 4.56E+03
FREQUENCY [Hz]

Printout of the results obtained for X

FREQUENCY DOMAIN DATA

FREQUENCY WINDOW=0 TO 4.56500E+03[Hz]
FREQUENCY INTERVAL= 7.13281E+01[Hz]

COEFF.	FREQUENCY[Hz]	REAL	IMAG	MAGNITUDE	PHASE[DEG]
DC TERM	0.00000E+00	3.40156E-02			
MAX FREQ.	4.56500E+03	-1.76563E-03			
1	7.13281E+01	1.22235E-03	1.87971E-03	2.24219E-03	56.96
2	1.42656E+02	-1.99268E-04	6.49395E-04	6.79280E-04	107.06
3	2.13984E+02	8.24246E-04	2.80697E-05	8.24724E-04	1.95
4	2.85312E+02	8.35506E-04	7.07762E-04	1.09499E-03	40.27
5	3.56641E+02	3.80316E-04	-1.06828E-04	3.95034E-04	-15.69
6	4.27969E+02	1.57764E-03	-1.70660E-04	1.58684E-03	-6.17
7	4.99297E+02	1.86504E-03	7.39181E-04	2.00618E-03	21.62
8	5.70625E+02	-6.86691E-01	5.59146E-01	8.85545E-01	140.85
9	6.41953E+02	-3.81833E-03	7.48294E-05	3.81906E-03	178.88
10	7.13281E+02	-4.75131E-04	-3.71229E-04	6.02960E-04	-142.00
11	7.84609E+02	-9.93946E-04	7.23897E-04	1.22962E-03	143.93
12	8.55937E+02	-1.00074E-03	-2.31644E-04	1.02720E-03	-166.97
13	9.27266E+02	-6.69685E-05	4.64290E-05	8.14889E-05	145.27
14	9.98594E+02	-1.28368E-03	2.54455E-04	1.30865E-03	168.79
15	1.06992E+03	-1.11979E-03	-1.99692E-03	2.28946E-03	-119.28
16	1.14125E+03	2.72667E-03	-1.35389E-03	3.04430E-03	-26.41
17	1.21258E+03	2.05029E-03	1.06645E-03	2.31106E-03	27.48
18	1.28391E+03	-1.17571E-04	1.20100E-03	1.20674E-03	95.59
19	1.35523E+03	1.38866E-04	-1.42109E-05	1.39591E-04	-5.84
20	1.42656E+03	6.49400E-04	7.67064E-04	1.00504E-03	49.75
21	1.49789E+03	-2.58167E-04	5.99868E-04	6.53063E-04	113.29
22	1.56922E+03	7.86887E-04	-1.03435E-04	7.93656E-04	-7.49
23	1.64055E+03	1.45348E-03	2.02627E-03	2.49367E-03	54.35
24	1.71187E+03	9.20364E-04	-2.20526E-02	2.20718E-02	-87.61
25	1.78320E+03	-2.54125E-03	1.66115E-04	2.54667E-03	176.26
26	1.85453E+03	-6.29228E-04	-7.85747E-04	1.00664E-03	-128.69
27	1.92586E+03	-3.97233E-04	4.06247E-04	5.68182E-04	134.36
28	1.99719E+03	-1.10891E-03	-1.37416E-04	1.11739E-03	-172.94
29	2.06852E+03	-2.53258E-04	-3.10299E-04	4.00530E-04	-129.22
30	2.13984E+03	-9.11379E-04	6.77362E-04	1.13553E-03	143.38
31	2.21117E+03	-2.33126E-03	-9.72983E-04	2.52616E-03	-157.35
32	2.28250E+03	4.21875E-04	5.00000E-04	6.54201E-04	49.84
33	2.35383E+03	2.09620E-03	-8.86866E-04	2.27609E-03	-22.93
34	2.42516E+03	6.83043E-04	7.26162E-04	9.96925E-04	46.75
35	2.49648E+03	-1.73746E-05	-3.32170E-04	3.32624E-04	-92.99
36	2.56781E+03	9.91854E-04	-1.53995E-04	9.05051E-04	-9.80
37	2.63914E+03	1.23064E-04	3.61867E-04	3.82221E-04	71.22
38	2.71047E+03	3.98217E-04	-8.35660E-04	9.25691E-04	-64.52
39	2.78180E+03	2.24207E-03	1.20244E-04	2.24529E-03	3.07
40	2.85312E+03	-3.34958E-02	-4.32263E-03	3.37736E-02	-172.65
41	2.92445E+03	-1.74962E-03	1.90004E-03	2.58299E-03	132.64
42	2.99578E+03	-1.04267E-03	-2.08757E-04	1.06337E-03	-168.68
43	3.06711E+03	-3.46964E-05	5.16044E-04	5.17209E-04	93.85
44	3.13844E+03	-9.46894E-04	6.26218E-04	1.13523E-03	146.52
45	3.20977E+03	-4.39817E-04	-1.70756E-04	4.71801E-04	-158.78
46	3.28109E+03	-2.06128E-04	1.07738E-03	1.09692E-03	100.83
47	3.35242E+03	-2.41989E-03	8.77726E-04	2.57415E-03	160.06
48	3.42375E+03	-1.64855E-03	-3.46327E-03	3.83562E-03	-115.45
49	3.49508E+03	7.50390E-04	-2.26680E-03	2.38778E-03	-71.68
50	3.56641E+03	8.92867E-04	-8.35841E-05	8.96771E-04	-5.35
51	3.63773E+03	-3.08087E-04	-3.16747E-04	4.41867E-04	-134.21
52	3.70906E+03	4.51612E-04	-8.13781E-04	9.30695E-04	-60.97
53	3.78039E+03	2.90333E-04	8.68241E-05	3.03037E-04	16.65
54	3.85172E+03	-3.70103E-04	-9.14404E-04	9.86464E-04	-112.04
55	3.92305E+03	1.64585E-03	-1.59044E-03	2.28873E-03	-44.02
56	3.99437E+03	1.08919E-02	-1.57174E-02	1.91225E-02	-55.28
57	4.06570E+03	2.01785E-05	2.39354E-03	2.39362E-03	89.52
58	4.13703E+03	-1.02150E-03	4.86180E-04	1.13129E-03	154.55
59	4.20836E+03	2.72912E-04	2.89780E-04	3.98062E-04	46.72
60	4.27969E+03	-2.71833E-04	9.99892E-04	1.03619E-03	105.21
61	4.35102E+03	-4.95025E-04	6.26499E-05	4.98974E-04	172.79
62	4.42234E+03	5.43003E-04	7.53391E-04	9.28681E-04	54.22
63	4.49367E+03	-1.13086E-03	2.14277E-03	2.42287E-03	117.82

Printout of the results obtained for Y

FREQUENCY DOMAIN DATA

FREQUENCY WINDOW=0 TO 4.56500E+03[Hz]
FREQUENCY INTERVAL= 7.13281E+01[Hz]

COEFF.	FREQUENCY[Hz]	REAL	IMAG	MAGNITUDE	PHASE[DEG]
DC TERM	0.00000E+00	4.15313E-02			
MAX FREQ.	4.56500E+03	-3.23438E-03			
1	7.13281E+01	1.66399E-03	7.40356E-04	1.82126E-03	23.99
2	1.42656E+02	6.58785E-04	1.05810E-03	1.24643E-03	58.09
3	2.13984E+02	7.18324E-05	6.28126E-04	6.32220E-04	83.48
4	2.85312E+02	1.49137E-04	-3.64706E-04	3.94021E-04	-67.76
5	3.56641E+02	5.77635E-04	-1.27403E-03	1.39886E-03	-65.61
6	4.27969E+02	1.25536E-03	-9.23592E-04	1.55851E-03	-36.34
7	4.99297E+02	2.56984E-03	3.51685E-05	2.57008E-03	.78
8	5.70625E+02	-3.02829E-01	8.42874E-01	8.95624E-01	109.76
9	6.41953E+02	6.80751E-04	2.73284E-03	2.81635E-03	76.01
10	7.13281E+02	-7.42573E-04	2.03421E-03	2.16550E-03	110.05
11	7.84609E+02	-9.44134E-04	1.27048E-03	1.58288E-03	126.62
12	8.55937E+02	-1.88722E-04	4.74429E-04	5.10587E-04	111.69
13	9.27266E+02	5.90451E-04	3.41098E-04	6.81894E-04	30.01
14	9.98594E+02	9.96977E-04	9.87556E-04	1.40329E-03	44.73
15	1.06992E+03	6.92318E-04	1.77616E-03	1.90632E-03	68.70
16	1.14125E+03	-5.00028E-03	-3.12894E-03	5.89856E-03	-147.96
17	1.21258E+03	-9.46978E-04	1.86353E-03	2.09033E-03	116.94
18	1.28391E+03	-1.27492E-03	9.60528E-04	1.59625E-03	143.01
19	1.35523E+03	-8.76743E-04	2.72215E-04	9.18031E-04	162.75
20	1.42656E+03	-1.11337E-04	1.76192E-04	2.08422E-04	122.29
21	1.49789E+03	3.89231E-04	7.05350E-04	8.05617E-04	61.11
22	1.56922E+03	1.64240E-04	1.54459E-03	1.55330E-03	83.93
23	1.64055E+03	-6.60228E-04	1.93674E-03	2.04618E-03	108.82
24	1.71187E+03	5.96011E-03	3.72849E-03	7.03026E-03	32.03
25	1.78320E+03	-1.88106E-03	5.92221E-04	1.97208E-03	162.52
26	1.85453E+03	-1.47615E-03	-1.79063E-04	1.48697E-03	-173.08
27	1.92586E+03	-6.41646E-04	-4.35621E-04	7.75549E-04	-145.83
28	1.99719E+03	-8.13507E-05	9.00143E-05	1.21328E-04	132.11
29	2.06852E+03	-2.07432E-04	8.15659E-04	8.41622E-04	104.27
30	2.13984E+03	-8.77118E-04	1.23319E-03	1.51331E-03	125.42
31	2.21117E+03	-1.62918E-03	8.88379E-04	1.85565E-03	151.40
32	2.28250E+03	-1.17188E-03	1.10625E-02	1.11244E-02	96.05
33	2.35383E+03	-1.69537E-03	-7.92281E-04	1.87136E-03	-154.95
34	2.42516E+03	-8.86456E-04	-1.10080E-03	1.41335E-03	-129.84
35	2.49648E+03	-1.71617E-04	-6.82582E-04	7.03826E-04	-104.11
36	2.56781E+03	-5.73362E-05	4.80902E-05	7.48338E-05	140.01
37	2.63914E+03	-6.94632E-04	5.52688E-04	8.87681E-04	141.49
38	2.71047E+03	-1.49144E-03	3.19633E-04	1.52530E-03	167.90
39	2.78180E+03	-1.83952E-03	-5.16091E-04	1.91055E-03	-164.33
40	2.85312E+03	2.61538E-03	-1.28253E-02	1.30893E-02	-78.47
41	2.92445E+03	-6.33980E-04	-1.64877E-03	1.76645E-03	-111.03
42	2.99578E+03	2.10252E-04	-1.32674E-03	1.34330E-03	-81.00
43	3.06711E+03	3.81765E-04	-5.38089E-04	6.59760E-04	-54.64
44	3.13844E+03	-5.97700E-05	5.22814E-05	7.94090E-05	138.82
45	3.20977E+03	-8.56893E-04	-4.13355E-05	8.57890E-04	-177.24
46	3.28109E+03	-1.24583E-03	-7.44279E-04	1.45122E-03	-149.15
47	3.35242E+03	-9.75723E-04	-1.59961E-03	1.87371E-03	-121.38
48	3.42375E+03	5.93778E-03	-8.84769E-03	1.06555E-02	-56.13
49	3.49508E+03	7.81589E-04	-1.63352E-03	1.81088E-03	-64.43
50	3.56641E+03	1.11659E-03	-7.82550E-04	1.36351E-03	-35.02
51	3.63773E+03	6.74851E-04	-5.20057E-05	6.76852E-04	-4.41
52	3.70906E+03	-6.69471E-05	1.97127E-05	6.97889E-05	163.59
53	3.78039E+03	-5.78859E-04	-5.75020E-04	8.15920E-04	-135.19
54	3.85172E+03	-3.69291E-04	-1.37335E-03	1.42214E-03	-105.05
55	3.92305E+03	4.36710E-04	-1.73543E-03	1.78956E-03	-75.88
56	3.99437E+03	-3.69340E-02	1.16951E-02	3.87414E-02	162.43
57	4.06570E+03	1.66736E-03	-5.01385E-04	1.74112E-03	-16.74
58	4.13703E+03	1.33533E-03	3.08470E-04	1.37050E-03	13.01
59	4.20836E+03	5.25430E-04	4.93000E-04	7.20504E-04	43.18
60	4.27969E+03	-8.36732E-05	1.25637E-05	8.46112E-05	171.46
61	4.35102E+03	-3.47331E-05	-7.69708E-04	7.70491E-04	-92.58
62	4.42234E+03	7.51246E-04	-1.19657E-03	1.41286E-03	-57.88
63	4.49367E+03	1.56498E-03	-8.83121E-04	1.79696E-03	-29.44

APPENDIX 6

Correlation (CXY) results.

FREQUENCY DOMAIN DATA

FREQUENCY WINDOW=0 TO 6.40000E+01[Hz]
FREQUENCY INTERVAL= 1.00000E+00[Hz]

COEFF.	FREQUENCY[Hz]	REAL	IMAG	MAGNITUDE	PHASE[DEG]
DC TERM	0.00000E+00	3.21394E-01			
MAX FREQ.	6.40000E+01	-2.18750E-11			
1	1.00000E+00	3.45436E-11	5.65294E-11	6.62483E-11	58.57
2	2.00000E+00	-8.86105E-12	3.50774E-11	3.61794E-11	104.18
3	3.00000E+00	-1.80304E-11	1.88755E-11	2.61032E-11	133.69
4	4.00000E+00	-1.35038E-11	1.23185E-11	1.82783E-11	137.63
5	5.00000E+00	-6.52535E-12	9.84500E-12	1.18112E-11	123.54
6	6.00000E+00	-1.22730E-11	-1.97721E-12	1.24313E-11	-170.85
7	7.00000E+00	9.89935E-12	5.48738E-12	1.13185E-11	29.00
8	8.00000E+00	-1.96406E-11	9.52094E-12	2.18267E-11	154.14
9	9.00000E+00	-9.60620E-12	-1.23714E-11	1.56630E-11	-127.83
10	1.00000E+01	-1.85432E-11	-1.05196E-11	2.13192E-11	-150.43
11	1.10000E+01	-1.90448E-12	-1.66749E-11	1.67833E-11	-96.52
12	1.20000E+01	1.26353E-13	2.80601E-11	2.80604E-11	89.74
13	1.30000E+01	-4.61009E-11	-3.86646E-11	6.01684E-11	-140.01
14	1.40000E+01	3.69434E-11	-5.55326E-11	6.66984E-11	-56.37
15	1.50000E+01	3.13103E-11	8.78624E-13	3.13227E-11	1.61
16	1.60000E+01	6.79688E-11	-1.87300E-11	7.05075E-11	-15.42
17	1.70000E+01	3.10930E-11	2.30258E-11	3.86906E-11	36.52
18	1.80000E+01	-3.03866E-13	2.80092E-11	2.80109E-11	90.62
19	1.90000E+01	-2.25839E-12	1.45991E-11	1.47728E-11	98.79
20	2.00000E+01	-1.05219E-11	1.56381E-11	1.88483E-11	123.93
21	2.10000E+01	-1.77134E-11	-1.00770E-11	2.03792E-11	-150.36
22	2.20000E+01	1.68419E-11	-1.73246E-12	1.69308E-11	-5.87
23	2.30000E+01	-1.42060E-11	3.02854E-11	3.34517E-11	115.13
24	2.40000E+01	-1.57220E-11	-1.58373E-11	2.23159E-11	-134.79
25	2.50000E+01	-6.24638E-14	5.21805E-12	5.21842E-12	90.69
26	2.60000E+01	-6.34926E-12	2.21869E-14	6.34930E-12	179.80
27	2.70000E+01	-9.78391E-12	7.90984E-12	1.25814E-11	141.05
28	2.80000E+01	-9.40631E-12	-1.08281E-11	1.43431E-11	-130.98
29	2.90000E+01	2.91247E-11	-1.04532E-11	3.09438E-11	-19.74
30	3.00000E+01	2.59814E-12	1.83971E-11	1.85797E-11	81.96
31	3.10000E+01	-1.08310E-11	1.29769E-11	1.69030E-11	129.85
32	3.20000E+01	-2.65625E-11	1.56250E-12	2.66084E-11	176.63
33	3.30000E+01	-1.59152E-11	-1.51612E-11	2.19808E-11	-136.39
34	3.40000E+01	-1.60990E-12	-1.62354E-11	1.63150E-11	-95.66
35	3.50000E+01	5.56480E-12	-1.20275E-11	1.32525E-11	-65.17
36	3.60000E+01	1.80151E-11	2.85356E-12	1.82397E-11	9.00
37	3.70000E+01	-4.03243E-12	1.73009E-11	1.77646E-11	103.12
38	3.80000E+01	-1.32042E-11	-5.62071E-13	1.32162E-11	-177.56
39	3.90000E+01	-1.35261E-11	-1.04938E-11	1.71194E-11	-142.20
40	4.00000E+01	1.02134E-11	-2.20374E-11	2.42892E-11	-65.13
41	4.10000E+01	6.21642E-12	2.37291E-11	2.45299E-11	75.32
42	4.20000E+01	-3.59422E-11	-6.11353E-12	3.64584E-11	-170.35
43	4.30000E+01	9.38661E-12	-6.68287E-11	6.74847E-11	-82.00
44	4.40000E+01	5.64577E-11	-4.88726E-11	7.46726E-11	-40.88
45	4.50000E+01	8.15542E-11	4.49011E-11	9.30978E-11	28.84
46	4.60000E+01	-9.05427E-12	6.54026E-11	6.60264E-11	97.88
47	4.70000E+01	8.73989E-12	-3.36321E-12	9.36466E-12	-21.05
48	4.80000E+01	4.98097E-01	4.35779E-02	5.00000E-01	5.00
49	4.90000E+01	-6.54945E-11	-2.27697E-11	6.93397E-11	-160.83
50	5.00000E+01	-2.12258E-11	-1.42064E-11	2.55413E-11	-146.21
51	5.10000E+01	-3.55531E-11	-2.98265E-12	3.56780E-11	-175.20
52	5.20000E+01	-4.43890E-11	-4.31932E-11	6.19358E-11	-135.78
53	5.30000E+01	1.94628E-11	-6.90197E-11	7.17113E-11	-74.25
54	5.40000E+01	3.88924E-11	-1.04234E-12	3.89063E-11	-1.54
55	5.50000E+01	4.55578E-12	-1.65883E-11	1.72025E-11	-74.64
56	5.60000E+01	2.82742E-11	-1.16793E-11	3.05914E-11	-22.44
57	5.70000E+01	2.02272E-11	-7.99883E-13	2.02430E-11	-2.26
58	5.80000E+01	2.88099E-11	1.18146E-11	3.11383E-11	22.30
59	5.90000E+01	1.12513E-11	2.39436E-11	2.64554E-11	64.83
60	6.00000E+01	1.97191E-12	2.42575E-11	2.43375E-11	85.35
61	6.10000E+01	-1.25759E-11	2.33479E-11	2.65194E-11	118.31
62	6.20000E+01	3.28107E-12	7.39544E-12	8.09060E-12	66.07
63	6.30000E+01	-1.33102E-11	3.60833E-11	3.84600E-11	110.25

APPENDIX 7

Program listing and results obtained for the Numerical Method.

```
10 ! THIS PROGRAM DETERMINES THE PHASE BY SIMULATION OF TWO SIGNALS
20    ! X(K) AND Y(K) BY FITTING A THIRD DEGREE POLYNOMIAL AT THE ZERO CROSSINGS
30    ! OF THE SIGNALS AND USES THE NEWTON'S METHOD TO FIND THE ROOT AND THEN
40    ! CALCULATES THE PHASE... SHOWS ALSO THE  ERROR BETWEEN THE ACTUAL PHASE
50    ! AND THE CALCULATED PHASE...SHOWS ALSO THE EFFECT OF VARYING THE RATIO
60    ! OF SAMPLING FREQUENCY TO THE FREQUENCY OF THE SIGNALS....SHOWS THAT
70    ! THE ERROR IS MINIMUM AT A RATIO OF 6.
80    ! GENERATE Y1(K) AND Y2(K) THE TWO SIGNALS OF INTEREST
90    ! Y2 LEADS Y1
100 DIM Y1(200),Y2(200),X(200),I(50),J(50)
110  ! SAMPLING FREQ. IS F1 AND THE FREQUENCY OF THE SIGNALS IS F
120  ! BY VARYING F1 WE CAN STUDY THE EFFECTS OF THE RATIO F1/F.
130 F=100
140 INPUT F1
150  PRINT "SAMP.FREQ.IS ",F1
160   PRINT "FREQ. OF SIGNAL=",F
170 FOR P=0 TO 90 STEP 10
180   ! HERE P IS THE PHASE IN DEGREES
190 F0=2*(22/7)*F
200 T=1/F1
210 P0=P*(22/7)/180
220 FOR K=1 TO 199
230 Y1(K)=SIN(F0*K*T)
240 Y2(K)=SIN(F0*K*T+P0)
250 NEXT K
260 L1=1
270 L2=1
280 P3=0
290  ! DETERMINATION OF ZERO CROSSINGS
300 FOR K=1 TO 197
310 IF SGN(Y1(K))-SGN(Y1(K+1))=-2 THEN GOTO 330
320 GOTO 350
330 I(L1)=K
340 L1=L1+1
350 NEXT K
360 FOR K=1 TO 197
370 IF SGN(Y2(K))-SGN(Y2(K+1))=-2 THEN GOTO 390
380 GOTO 410
390 J(L2)=K
400 L2=L2+1
410 NEXT K
420   ! FIND THE PHASE AT THE FIRST 10 ZERO CROSSINGS AND TAKE THE AVERAGE
430   ! FIRST SET UP THE REQD. VALUES FOR THE DIFFERENCE TABLE
440 FOR N=1 TO 10
450 K=I(N)
460 X(K)=Y1(K)
470 X(K-1)=Y1(K-1)
480 X(K+1)=Y1(K+1)
490 X(K+2)=Y1(K+2)
500 ! USE SUBROUTINE FOR FITTING THE POLYNOMIAL
510 GOSUB 650
520   ! HERE T1 DENOTES THE ZERO CROSSING OF FIRST SIGNAL
530 T1=X0
540   ! REPEAT FOR SECOND SIGNAL
550 K=J(N)
560 X(K)=Y2(K)
570 X(K-1)=Y2(K-1)
580 X(K+1)=Y2(K+1)
590 X(K+2)=Y2(K+2)
600 GOSUB 650
```

```
610   ! T2 REPRESENTS ZERO CROSSING OF SECOND SIGNAL
620 T2=X0
630 GOTO 990
640   ! SUBROUTINE FOR FITTING THE POLYNOMIAL USING THE DIFFERENCE TABLE
650 REM FORM DIFFERENCE TABLE
660 IF ABS(X(K))<.001 THEN X(K)=0
670 IF ABS(X(K+1))<.001 THEN X(K+1)=0
680 A0=X(K-1)
690 A1=X(K)
700 A2=X(K+1)
710 A3=X(K+2)
720 B0=A1-A0
730 B1=A2-A1
740 B2=A3-A2
750 C0=(B1-B0)/2
760 C1=(B2-B1)/2
770 D0=(C1-C0)/3
780 W1=F0*T
790 W2=W1*W1
800 W3=W2*W1
810 E0=A0-B0*(K-1)*W1+C0*K*(K-1)*W2-D0*K*(K-1)*(K+1)*W3
820 E1=B0-C0*(2*K-1)*W1+D0*(3*K*K-1)*W2
830 E2=C0-D0*(3*K)*W1
840 E3=D0
850 REM THE ABOVE ARE THE COEFFS. OF POLYNOMIAL.
860 REM NEWTONS METHOD
870 DEF FNP(V)=E0+E1*V+E2*V*V+E3*V*V*V
880 DEF FNQ(V)=E1+E2*V*2+E3*V*V*3
890 X0=W1*K+W1/2
900 X0=X0-FNP(X0)/FNQ(X0)
910 IF ABS(FNP(X0))-1E-4<0 THEN 950
920 M1=M1+1
930 IF M1>10 THEN 770
940 GOTO 900
950 M1=0
960 RETURN
970   ! OBTAIN THE DIFFERENCE BETWEEN THE ZERO CROSSINGS OF THE TWO SIGNALS
980   ! AND CALCULATE THE PHASE
990 P1=T1-T2
1000 P2=P1/W1*360*F/F1
1010 P3=P3+P2
1020 NEXT N
1030 PRINT "AVERAGE OF THE PHASE VALUES IS",P3/(N-1)
1040 PRINT "ACTUAL PHASE IS",P,"ERROR IS ",P-P3/(N-1)
1050  PRINT " "
1060  P3=0
1070  ! REPEAT FOR NEXT VALUE OF PHASE
1080 NEXT P
1090 END
```

```
SAMP.FREQ.IS              600
FREQ. OF SIGNAL=          100
AVERAGE OF THE PHASE VALUES IS      0
ACTUAL PHASE IS           0         ERROR IS                0

AVERAGE OF THE PHASE VALUES IS      9.77897919153
ACTUAL PHASE IS           10        ERROR IS                .2210208085

AVERAGE OF THE PHASE VALUES IS      19.3094805375
ACTUAL PHASE IS           20        ERROR IS                .6905194625

AVERAGE OF THE PHASE VALUES IS      28.7222999216
ACTUAL PHASE IS           30        ERROR IS                1.2777000784

AVERAGE OF THE PHASE VALUES IS      38.1637038994
ACTUAL PHASE IS           40        ERROR IS                1.8362961006

AVERAGE OF THE PHASE VALUES IS      47.7834279363
ACTUAL PHASE IS           50        ERROR IS                2.2165720637

AVERAGE OF THE PHASE VALUES IS      59.9999991558
ACTUAL PHASE IS           60        ERROR IS                .0000008442

AVERAGE OF THE PHASE VALUES IS      69.7789819638
ACTUAL PHASE IS           70        ERROR IS                .2210180362

AVERAGE OF THE PHASE VALUES IS      79.3094892334
ACTUAL PHASE IS           80        ERROR IS                .6905107666

AVERAGE OF THE PHASE VALUES IS      88.7222917675
ACTUAL PHASE IS           90        ERROR IS                1.2777082325
```

APPENDIX B

Listing of the program used for the Adaptive Predictor.

```
10   REM*************************************************************
20   REM THIS PROGRAM IS AN ADAPTIVE PREDICTOR WHICH ACTS AS A
30   REM SIGNAL ENHANCER.
40   REM THE INPUT SIGNAL IS STORED IN THE ARRAY X) WHICH
50   REM CONTAINS 500 SAMPLES OF THE INPUT SIGNAL X
60   REM THE ARRAY X1(*) IS USED TO GENERATE THE N PAST VALUES
70   REM OF THE INPUT AND THIS ACTS AS THE INPUT TO THE FILTER
80   REM THE ARRAY G(*) IS USED TO STORE THE OUTPUT OF THE
90   REM FILTER WHICH IS AN ENHANCED VERSION OF THE INPUT
100  REM SIGNAL - GETS RID OF THE NOISE IN INPUT SIGNAL
110  REM SINCE THE ADAPTIVE FILTER TAKES A FEW ITERATIONS TO
120  REM ADAPT ITSELF . THE OUTPUT G(*) MUST BE TAKEN AFTER A
130  REM FEW ITERATIONS FOR SUBSEQUENT PROCESSING.
140  REM*************************************************************
150  REM FORM THE DIFFERENT ARRAYS
160  REM*************************************************************
170  DIM X(500),X1(16),G(500),W(16)
180  REM*********************************************************
190  REM INITIALIZE FILTER COEFFICIENTS
200  REM*********************************************************
210  FOR I=1 TO 16
220  W(I)=0
230  NEXT I
240  REM*********************************************************
250  REM DEFINE THE NUMBER OF WEIGHTS OF THE FILTER TO BE
260  REM USED (N)
270  REM*********************************************************
280  N=10
290  REM*********************************************************
300  REM DEFINE THE CONVERGENCE PARAMETER V1
310  REM*********************************************************
320 V1=.002
330  REM*********************************************************
340  REM START PROCESSING
350  REM*********************************************************
360 FOR K=N TO 499
370  REM*********************************************************
380  REM COMPUTE INPUT SIGNAL ENERGY AND USE IT TO UPDATE
390  REM THE CONVERGENCE PARAMETER
400  REM*********************************************************
410 V2=V2*(1-V1)+V1*X (K)*X (K)
420 IF V2<.001 THEN 450
430 V=V1/V2
440 GOTO 470
450 V=0
```

```
460  REM*********************************************************
470  REM GENERATE X1 THE INPUT TO THE FILTER
480  REM*********************************************************
490  FOR J=1 TO N
500  M=J-1
510 X1(J)=X (K-M)
520  NEXT J
530  G0=0
540  REM*********************************************************
550  REM FIND THE OUTPUT OF THE FILTER G0
560  REM*********************************************************
570  FOR L= 1 TO N
580  G1=W(L)*X1(L)
590  G0=G0+G1
600  NEXT L
610  REM*********************************************************
620 REM COMP. ERROR
630  REM*********************************************************
640 E=X(K+1)-G0
650  REM*********************************************************
660  REM STORE THE OUTPUT OF THE FILTER IN G(*)
670  REM*********************************************************
680G(K+1)=G0
690  FOR I=1  TO N
700  REM*********************************************************
710 REM UPDATE COEFFS. USING THE ERROR E
720  REM*********************************************************
730  W(I)=W(I)+V*E*X1(I)
740  NEXT I
750  NEXT K
760  REM*********************************************************
770 REM AFTER THE REQUIRED NUMBER OF ITERATIONS G(*)
780 REM CONTAINS THE ENHANCED OUTPUT... ALLOW ABOUT 100
790 REM ITERATIONS FOR THE ADAPTATION OF THE FILTER.
800  REM*********************************************************
810 RETURN
```

APPENDIX 9

Program listing for the PLOT package.

```
NAME  PRD TYPE R10    REM THIS PRDGRAM SHDWS HDW THE Plot SUSRDUTIHE CAN SE USED
20    REM TO PLDT THE REQUIRED DATA
30    REM HERE   RANDOH IS TO BE PLDTTED
40    DPTIDH BASE 1
50    DIM He(100),Ve(100)
60    RAHDDHIZE
70    IHPUT Hpointe
80    FOR I=1 TO 100
90    He(I)=I
100   Ve(I)=SIH(2*PI*10*I)
110   HEXT I
120   GOSUB Plot
130   EHD
140 Plot:  REM   THIS IS A SUSRDUTIHE TO PLDT DATA IH THE ARRAYS
150   REM He AHD Ve . THE PRDCEDURE IS TO FILL THE ARRAYS AHD THEH
160   REH CALL THE SUSRDUTIHE Plot
170   REH THE DATA HAY SE DH FLDPPY DISK OR OTHER STORAGE DEVICE
180   REM THIS DATA HAS TO BE ACCESSED AHD USED TO FILL THE ABDVE
190   REM HENTIDHED ARRAYS
200   REM Find max and min values of the data in the arrays
210   Hemin=Vemin=1E99
220   REM Hpoints IS THE NUHSER OF POINTS TO SE PLOTTED
230   FDR I=1 TO Hpointe
240   IF He(I)<Hemin THEH Hemin=He(I)
250   IF Ve(I)<Vemin THEH Vemin=Ve(I)
260   HEXT I
270   Hemax=Vemax=-1E99
280   FDR I=1 TO Hpoints
290   IF He(I)>Hemax THEH Hemax=He(I)
300   IF Ve(I)>Vemax THEH Vemax=Ve(I)
310   HEXT I
320   PRIHT Hemin,Hemax,Vemin,Vemax
330   REM TO find limits Her,Hel,Vet,Veb
340   GOTO 620
350   ! ACTUALLY A SUSRDUTIHE
360   REH THIS SUSRDUTIHE USES THE VALUE IN Value AHD RETURHS A
370   REH HEW VALUE DEPEHDIHG WHICH DF THE LIHITS Her,Hel,Vet,Veb
380   REH IS TO BE DETERHIHED
390   IF ABS(INT(Value))-ABS(Value)=0 THEH Retern
400   IF ABS(Value)>1 THEN There
410   IF Value=1 THEN Retern
420   GOTO 430
430   Decimal=1
440   Dplace=10*Decimal
450   Dplace=1/Dplace
460   IF ASS(Value)>Dplace THEH Here
470   Decimal=Decimal*10
480   GOTO 440
490 There:   Dplace=1
500 Here:  Hark=Dplace
510   !
520   IF Lim=1 THEN Getmax  *
530   IF Lim=0 THEH Getmin
540 Getmax:   Haxad=Value DIV Dplace+1
550           IF Haxad=0 THEH Haxad=-2
560           Value=Haxad*Dplace
570   GOTD Retern
580 Getmin:   Hinad=Value DIV Dplace-1
590   IF Hinad=0 THEN Hinad=.9
600           Value=Hinad*Dplace
```

```
610 Return:   RETURN
620   ! Find the limits
630   ! Hor
640   Value=Hemax
650   Lim=1
660   GOSUB 350
670   Hor=Value
680   ! Hel
690   Value=Hemin
700   Lim=0
710   GOSUB 350
720   Hel=Value
730   ! Vet
740   Value=Vemax
750   Lim=1
760   GOSUB 350
770   Vet=Value
780   ! Veb
790   Value=Vemin
800   Lim=0
810   GOSUB 350
820   Veb=Value
830   REM THE TIC SPACINGS FOR DRAWING THE AXES ARE THEN FOUND
840   ! FIND Xtic,Ytic ETC
850   Htic=(Hor-Hel)/20
860   Hmaj=5
870   Vtic=(Vet-Veb)/20
880   Vmaj=5
890   PRINT Hel,Hor,Veb,Vet,Htic,Vtic
900   REM THESE SUBROUTINES DETERMINE THE DIFFERENT PARAMETERS TO BE
910   REM USED IN FINDING INTERSECTIONS AND THE SPACINGS NEEDED FOR
920   REM LABELING THE AXES
930   IF (Hel<0) AND (Hor<0) AND ((Veb<0) AND (Vet<0)) THEN GOSUB One
940   IF (Hel<0) AND (Hor)0) AND ((Veb<0) AND (Vet<0)) THEN GOSUB Two
950   IF (Hel)0) AND (Hor)0) AND ((Veb<0) AND (Vet<0)) THEN GOSUB Three
960   IF (Hel<0) AND (Hor<0) AND ((Veb<0) AND (Vet)0)) THEN GOSUB Four
970   IF (Hel<0) AND (Hor)0) AND ((Veb<0) AND (Vet)0)) THEN GOSUB Five
980   IF (Hel)0) AND (Hor)0) AND ((Veb<0) AND (Vet)0)) THEN GOSUB Six
990   IF (Hel<0) AND (Hor<0) AND ((Veb)0) AND (Vet)0)) THEN GOSUB Seven
1000  IF (Hel<0) AND (Hor)0) AND ((Veb)0) AND (Vet)0)) THEN GOSUB Eight
1010  IF (Hel)0) AND (Hor)0) AND ((Veb)0) AND (Vet)0)) THEN GOSUB Nine
1020  PRINT Xint,Yint
1030  FOR I=5 TO 20 STEP 5
1040  PRINT Hel+Htic*I
1050  PRINT Veb+Vtic*I
1060  NEXT I
1070  REM START PLOTTING OPERATIONS
1080  PLOTTER IS 7,5,"9572A"
1090  GRAPHICS
1100  LIMIT 50,300,50,200
1110  FRAME
1120  LOCATE 25,145,25,90
1130  FRAME
1140  SCALE Hel,Hor,Veb,Vet
1150  AXES Htic,Vtic,Xint,Yint,Hmaj,Vmaj,5
1160  FRAME
1170  REM ACTUAL PLOTTING OF DATA
1180  FOR I=1 TO Npoints
1190  PLOT He(I),Ve(I)
1200  NEXT I
```

```
1210  GOTO 1500
1220 One:Xint=Ner
1230  Yint=Vst
1240  RETURN
1250 Two: Xint=0
1260  Yint=Vet
1270  RETURN
1280 Three: Xint=Hel
1290  Yint=Vet
1300  RETURN
1310 Four: Xint=Ner
1320  Yint=0
1330  RETURN
1340 Five: Xint=Yint=0
1350  RETURN
1360 Six: Xint=Hel
1370  Yint=0
1380 RETURN
1390 Seven: Xint=Ner
1400  Yint=Veb
1410  RETURN
1420 Eight: Xint=0
1430  Yint=Veb
1440  RETURN
1450 Nine: Xint=Nel
1460  Yint=Veb
1470  RETURN
1480  REM  THE VALUES OF Sgnx AND Sgny DETERMINE THE RELATIVE MOVEMENTS
1490  REM OF THE PEN FOR LABELING THE AXES
1500  IF Nel>=0 THEN Sgnx=-1
1510  IF Nel<=0 THEN Sgnx=1
1520  IF Veb<=0 THEN Sgny=1
1530  IF Veb>=0 THEN Sgny=-1
1540  IF (Xint=0) AND (Yint=0) THEN Sgnx=Sgny=-1
1550  REM LABEL THE X AXIS
1560  MOVE Nel,Veb
1570  FOR I=0 TO 20 STEP 5
1580  Xd=Nel+I*Netic
1590  Yd=Yint+1.5*Vetic*Sgny
1600  MOVE Xd,Yd
1610  LABEL USING "MD.DDE";Xd
1620  NEXT I
1630  FOR I=0 TO 20 STEP 5
1640 Xd=Xint+3*Netic*Sgnx
1650  Yd=Veb+I*Vetic
1660  MOVE Xd,Yd
1670  LABEL USING "MD.DDE";Yd
1680  NEXT I
1690  MOVE Nel+5*Netic,Yint+4*Vetic*Sgny
1700  PRINT "XLABEL"
1710  INPUT Bs
1720  LABEL Bs
1730  REM LABEL THE Y AXIS; ROTATE MOTION OF PEN BY 90 DEGREES
1740  DEG
1750  LDIR 90
1760  MOVE Xint+3.50*Netic*Sgnx,Veb+5*Vetic
1770  PRINT "Y LABEL"
1780  INPUT As
1790  LABEL As
1800  LDIR 0
```

```
1810  RAD
1820  REM INITIALIZE PEN DIRECTION
1830  REM END OF PLOTTING OPERATION
1840  PEN 0
1850  RETURN
```

BIBLIOGRAPHY

1. B. Widrow et al., "Adaptive Noise Cancelling: Principles and Applications", Proc. IEEE, Vol.63, pp.1692-1721, December 1975.

2. "Discrete time signals and systems" by N.Ahmed and T.Natarajan., Reston Publishing Co., Inc. Reston, Virginia.

3. North Star System Software Manual #25013B, Revision 2.1,1980. North Star Computers, Inc. 1440 Fourth St., Berkeley, CA 94710.

4. Multiprogrammer Model 6940B Operating and Service Manual. June 1979. Hewlett Packard, New Jersey Division, Green Pond Road, Rockaway, NJ 07866.

5. "An automated laboratory test system", by N.L.Fernandes, M.S. Report, 1983, Dept. of Electrical Engineering, Kansas State University, Manhattan, KS 66506.

6. Numerical Analysis by L.W.Johnson and R.D.Riess, Second edition, Addison - Wesley Publishing Co., Reading, Massachusetts.

7. TTL Data Book, National Semiconductor Corporation.

8. Techiques for Computer-based instrumentation by M.F.Wagdy, Doctor's Dissertation, 1983, Kansas State University, Manhattan, Kansas.

9. Hewlett-Packard 9845B Desktop Computer, Operating and Programming manual, HP Desktop Computer Division, 3404 East Harmony Road, Fort Collins, Colorado 80525.

10. High Speed A/D Converter card 69422A, Hewlett-Packard, New Jersey Division, Green Pond Road, Rockaway, New Jersey 07866.

ACKNOWLEDGEMENTS

I take this opportunity to express my sincere thanks to my major advisor Dr.M.S.P. Lucas. His great encouragement, views and stimulating discussions have contributed abundantly towards this entire project. His timely help and above all his remarkable sense of humor are deeply appreciated.

I also wish to express my sincere thanks to Dr.G.L. Johnson for his valuable support and suggestions and for extending his help in initiating me on the spinwriter which made the typing of this report a pleasure. Special thanks to Dr.P.M. Young for his support and suggestions.

I would like to express my gratitude to my parents who are a perpetual source of inspiration and encouragement.

APPLICATIONS OF SIGNAL PROCESSING TECHNIQUES IN MICROPROCESSOR BASED INSTRUMENTATION

by

MADHUKAR DUGGIRALA

B.S., College of Engineering, Guindy, Madras, India, 1982

AN ABSTRACT OF A MASTER'S THESIS

Submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Electrical Engineering

KANSAS STATE UNIVERSITY

Manhattan, Kansas

1984

ABSTRACT

The applications of signal processing techniques in microprocessor or computer based instrumentation were studied and a specific application in the measurement of phase was considered. The analyses were carried out in the discrete time (DT) domain using "samples" of the original signals. Algorithms based on Adaptive filtering, FFT, Correlation and Numerical methods have been developed. These methods were verified by using computer simulation and implemented on an automated test system.

The algorithm based on Adaptive filtering was developed using the Widrow's LMS (least mean square) method in the cancellation mode. Samples of the original signals form the primary and reference inputs to the adaptive filter and after a sufficient number of processing steps the phase difference is obtained from the coefficients of the filter. The FFT method uses the information from the phase spectra of the two signals in the frequency domain and determines the phase difference. The Correlation approach uses the amplitude spectra of each signal and also that of the cross correlated sequence obtained by multiplying the two signals in the DT domain and then calculates the phase difference. The Numerical method involves curve fitting and Newton's method to obtain an estimate of the zero crossings of the signals from which the phase difference is obtained.

The software development and system implementation was done on a North Star Horizon II, Z80 based microcomputer (NS) in conjunction with an MC6802 interface and a Hewlett-Packard Multiprogrammer (HPM); the whole forming an automated test

system. The MC 6802 interface serves as a communication link between the NS and the HPM. Data samples of the signals are obtained by using the High Speed Analog to Digital Converter card available on the HPM and these samples are then transmitted to the NS for subsequent processing by the algorithms discussed earlier. Computer simulation was done on a Hewlett-Packard 9845B Desktop minicomputer.

The design aspects of a suitable system for implementing these methods is also discussed. The role of the microprocessor as the control/communicating device is emphasized. Other applications in instrumentation are considered. A general purpose plot package was developed for the HP 9845B.