327a

DIGITAL CORRELATOR FOR SONAR SIGNALS

DESIGN AND IMPLEMENTATION OF A REAL-TIME DIGITAL REPLICA CORRELATOR USING BIT SLICE MICROPROCESSOR FOR PROCESSING SONAR SIGNALS

by

JOHN KING-ON MAN, B.Eng

A Thesis
Submitted to the Faculty of Graduate Studies
in Partial Fulfilment of the Requirements
for the Degree
Master of Engineering

McMaster University
September 1978

MASTER OF ENGINEERING (1978)
(Electrical Engineering)

McMASTER UNIVERSITY
Hamilton, Ontario

TITLE: Design and Implementation of a Real-Time Digital Replica Correlator Using Bit Slice Microprocessor for Processing Sonar Signals

AUTHOR: John King-on Man, B.Eng. (McMaster University)

SUPERVISOR: Dr. C.R. Carter

NUMBER OF PAGES: xi, 187

ABSTRACT

In the past, analog circuits, discrete digital logic circuits or minicomputers have been used to implement the signal processing section of a sonar systems. More recently, microprocessor based logic circuit designs have produced a new breed of system design approach which gives designers the flexibility that has never been available through the use of analog or discrete logic circuits; however, due to the inherent slow speed of the metal-oxide semiconductor (MOS) logic circuits, incorporating microprocessors in the implementation of a sonar signal processor is not feasible. With the advent of bipolar Schottky large scale integrated circuit technology, the speed performance of the microprocessors have been improved considerably, and signal processor designs employing microprocessors are now feasible.

The main objective of this work is to design, implement, and test a real-time digital sonar signal processor for processing pulsed CW signals. With design based on the use of the bit slice microprocessor, a signal processor has been constructed that has an 8 bit input, a 16 bit output. The processor is capable of detecting 16 different Doppler shifts. Laboratory generated signals are used in the testing and the experimental results show good agreement with the theory. A possible means of expanding the existing single channel signal processor into a multichannel processor has also been outlined.

ACKNOWLEDGEMENTS

The author gratefully acknowledges the help and guidance given by his supervisor, Dr. C.R. Carter, throughout the course of this work.

He would also like to thank the research personnel of the Communications Research Laboratory at McMaster, and in particular, Dr. H.C. Chan and Mr. W.J. Caldwell for many helpful discussions and suggestions.

Finally, the author wishes to thank his parents and wife for their constant encouragement and support throughout the whole work.

TABLE OF CONTENTS

TABLE OF CONTENTS (continued)

LIST OF ILLUSTRATIONS

LIST OF ILLUSTRATIONS (continued)

LIST OF ILLUSTRATIONS (continued)

LIST OF ILLUSTRATIONS (continued)

LIST OF TABLES

CHAPTER 1

INTRODUCTION

The word sonar is derived from sound navigation and ranging and is used synonymously with underwater sound and underwater acoustics. Today, the applications and developments of sonar systems are no longer confined as its name implies to the fields of navigation and ranging; they are now being used for a wide variety of purposes both in the military and in industry. In military applications, underwater sound is used for depth sounding; navigation; ship and submarine detection, ranging, and tracking; underwater communications; mine detection; and for guidance and control of torpedoes and other weapons. A modern echo-ranging sonar system for surface ships is illustrated in Fig. 1.1. Some of the commercial applications of underwater acoustic are listed in Table 1.1.

As the result of the demands for high precision underwater navigation equipment and new applications for underwater acoustics, sonar systems have become increasingly complex and sophisticated. In general, an active sonar system is a system capable of discovering the presence of an underwater target by transmitting an acoustic pulse and detecting the echo with some form of filtering technique. Most of the existing sonar systems employ analog filtering devices and automatic gain control to accomplish the desired goal. The analog system has performed satisfactorly in past years, however, it is costly and frequently bulky in size which is a disadvantage for systems on-board ships having limited space.

With the development of integrated digital circuits and digital signal processing techniques, many sonar signal processing systems designs incorporating analog and digital techniques were proposed and implemented [20]. Some of these systems have performed well; however, due to the technology available at the time, the benefits of the digital system were not fully exploited. The advanced technology in the large scale integrated (LSI) circuits in recent years has brought a new dimension to the system designers. The advantages of having a digital system are both technical and commercial. The main beneficial results are; smaller size, low power consumption, improved reliability, easier maintenance, and reduced cost of ownership. Just the reduction of the system size alone is important. For example, a reduction ratio of 11:1 was found on the signal processing section of a passive active detection and location sonar system (PADLOC) [1].

A digital sonar signal processor designed based on LSI digital circuits has been developed [2] by the Communication Research Laboratory at McMaster University. While the evaluation of the processor has clearly demonstrated the technical and commercial advantages of the digital system over the analog, it was felt that the implementation of the digital processor using a microcomputer would be more flexible and perhaps more economical. Before the development of Schottky bipolar LSI, microprocessors based on metal-oxide-semiconductor (MOS) technology have become popular as programmable LSI digital circuits replacing discrete logic and customized integrated circuits in the low speed applications. However, medium and high speed applications are not in the reach of the MOS devices. With the advent of Schottky bipolar LSI digital circuits, a new family of computing elements has been developed. The bit slice microprocessor, the new bipolar device, not only is faster and much more flexible than its predecessors, but also has a microprogrammable central processing element which can be programmed to suit the processing functions that a system designer needs.

A significant portion of this thesis is devoted in the design, implementation, and testing of a digital sonar processor based on the bit slice microprocessor. Due to the different types of energy and propagating medium, the propagation of underwater soundwave is not quite the same as the propagation of electromagnetic wave in air; therefore, a brief discussion on the highlights of the characteristics of the sea as a medium and derivation of the sonar equation are included in Chapter 2. In Chapter 3 various digital signal processing techniques are studied and compared for suitability of hardware implementation. The actual implementation of a single channel signal processor and a proposal for extending this to a multichannel processor are presented and discussed in Chapter 4. Test results are given in Chapter 5 and, finally, some recommended topics for futher studies are described in Chapter 6.

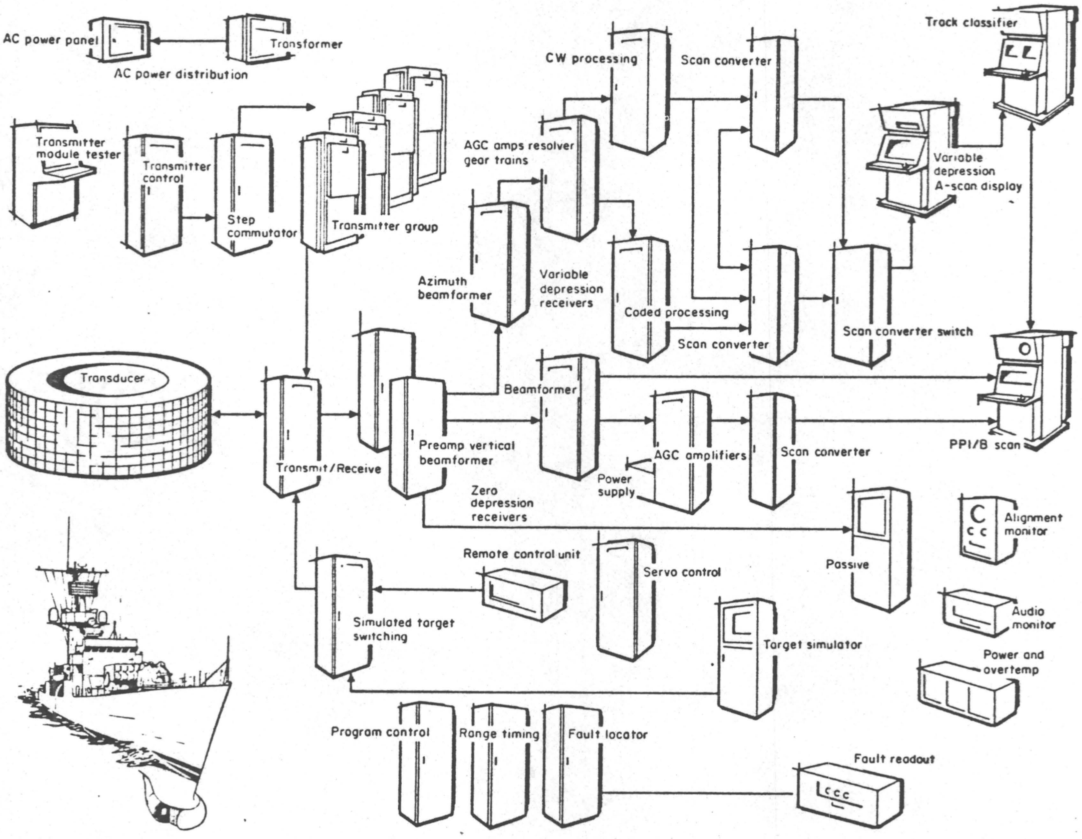
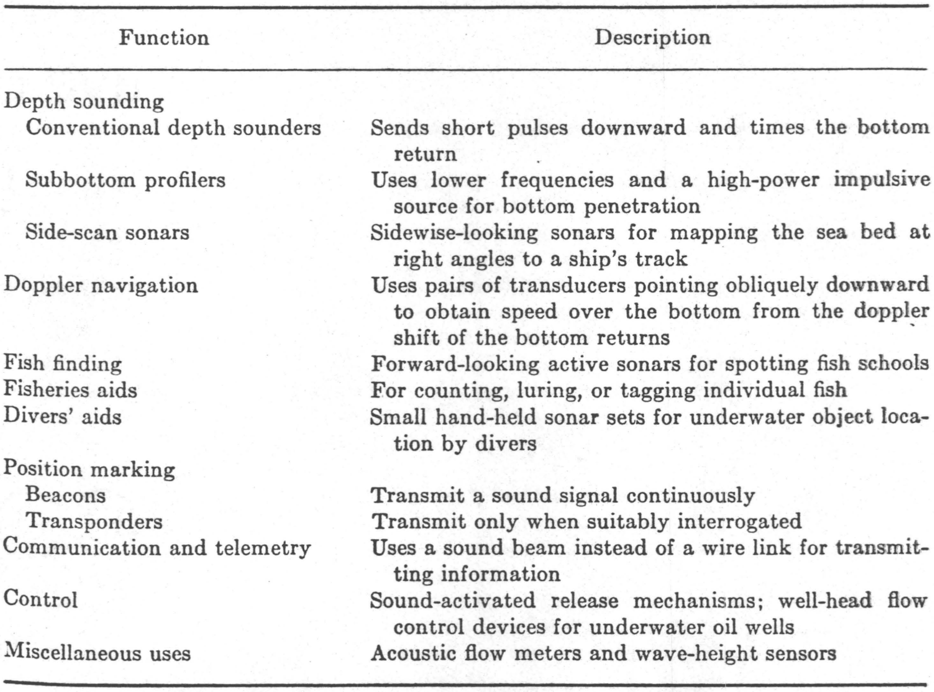


Fig. 1.1 The functional block diagram of the A/N-SQS-26 sonar for surface ships.



Function	Description
Depth sounding	
Conventional depth sounders	Sends short pulses downward and times the bottom return
Subbottom profilers	Uses lower frequencies and a high-power impulsive source for bottom penetration
Side-scan sonars	Sidewise-looking sonars for mapping the sea bed at right angles to a ship's track
Doppler navigation	Uses pairs of transducers pointing obliquely downward to obtain speed over the bottom from the doppler shift of the bottom returns
Fish finding	Forward-looking active sonars for spotting fish schools
Fisheries aids	For counting, luring, or tagging individual fish
Divers' aids	Small hand-held sonar sets for underwater object location by divers
Position marking	
Beacons	Transmit a sound signal continuously
Transponders	Transmit only when suitably interrogated
Communication and telemetry	Uses a sound beam instead of a wire link for transmitting information
Control	Sound-activated release mechanisms; well-head flow control devices for underwater oil wells
Miscellaneous uses	Acoustic flow meters and wave-height sensors

Table 1.1 Nonmilitary uses of sonar [8].

CHAPTER 2

THE SEA AS SOUND PROPAGATION MEDIUM

It has long been known that water is quite superior to air as a medium for sound transmission, and recurrent investigations of possible methods have proven sound to be superior to other means of transmitting information through water. However various physical and chemical effects in the sea make the transmission and reception of sound a complex problem. There are many ways in which equipment can be designed and used; however, an intelligent choice from the different alternatives depends on accurate knowledge of the factors affecting the performance.

For example, an adequate estimation of a target range depends on the knowledge of speed of sound travelling in the sea and the paths at which the sound travels to the target and returns to the point of transmission. The radiated power required to achieve the maximum effective operation range of a sonar system is, naturally, depended upon the power lost due to spatial spreadings and other various attenuating factors in the medium; the knowledge of sound ducts existing in the sea proffer the possibility of increasing the operation range without comparable increasing in the system output power. Also, in designing a sonar system, the knowledge of the characteristics of reverberation and noise played an important role. For instance, the frequency distribution of reverberation and sea ambient noise are important for the design of filter to suppress reverberation and noise, and amplitude distribution of reverberation plays an important role in determining the effective range of a sonar system.

In this chapter the highlights of some of the complexities of the medium and effects of these complexities on sound propagating in such medium are described, and, finally, a simple sonar equation that ties all these factors together is formulated to form a basic outline for estimating the performance of a sonar system.

2.1 SPEED OF SOUND IN THE SEA

Amongst all the factors governing the peculiar behaviour of sound travelling in the sea, the variation of the speed of sound in the medium is of most importance. In certain aspects the propagation of sound energy is similar to its counterpart in the air, the propagation of electromagnetic energy. The speed of the electromagnetic energy propagating in air changes little with respect to the variation of the properties of the air such as temperature, density, chemical components, etc.. This is in contrast to the speed of sound in the sea since the speed varies with depth, the seasons, geographic location, salinity, and time at fixed locations. The most prominent causes of variation of the speed of sound in the sea are due to the pressure, temperature, and salinity.

Various empirical formula exist relating the three major factors to the speed of sound. The accuracy of these formula lies within 0.2 per cent [3]; however, they are complicated expressions containing many high order terms. A simpler expression, just as accurate as the others over a limited range of conditions at sea, has been proposed by Leory [4]. The expression for the speed of sound at depth Z, is:

$$C = 1492.9 + 3.0(T-10) - 6 \times 10^{-3}(T-10)^2 - 4 \times 10^{-2}(T-18)^2$$
$$+ 1.2(S-35) - 10^{-2}(T-18)(S-35) + Z/61 \qquad (2.1)$$

where C is the speed of sound in meters per second, T is the temperature in degrees Celsius, and S is the salinity in parts per thousand (PPT). This formula is accurate to 0.1m/sec for T less than 20°C and for Z less than 800 meters.

As can be seen from Eqn. (2.1), the speed of sound in the sea increases with depth, temperature, and salinity. Table 2.1 gives an approximate variation of coefficients for the rate of change with these three quantities. Figure 2.1 illustrates the variation of sound speed in distilled water and seawater at zero depth for various salinities and temperature. The variation of the speed of sound as function of depth and temperature for a fixed salinity is shown in Fig. 2.2.

2.2 VELOCITY PROFILE

The speed of sound in the sea can be obtained by direct measurement with velocimeter or, alternatively, by hydrographic observations of the temperature, depth, and salinity and calculations using these parameters. The effect of the depth and temperature on the variation of the sound speed can best be demonstrated by the velocity profile of the sound. The velocity profile would naturally refer to the variation of sound speed with depth; however, the temperature of sea is a function of depth, thus, as a result, the velocity profile is a graphic display of the variation of the speed of sound as functions of depth and temperature. A typical velocity profile is depicted in Fig. 2.3, and it can be classified into the following layers:

A. Surface Layer

This is the region in which the speed is subject to daily and local changes of heating, cooling, and wind action. Because of the mixing of the water due to the wind action, an isothermal layer may exist. Under this condition sound ducts may be formed; sound trapped in these ducts can propagate a long distance. After a prolonged calm and sunny condition the sound duct disappears and is replaced by waters in which the velocity decreases with depth.

B. Seasonal Thermocline

The velocity profile in this region has a negative gradian which varies with the seasons. This is mainly caused by the change of temperature as depth increases. Seasonal thermocline is usually well defined in the summer and fall when the near surface water is warm. During the winter and spring tends to be indistinguishable from the surface layer.

C. Main Thermocline

This is situated just below the seasonal thermocline; the temperature change of the water due to the seasonal variation is minimum. The major increase of temperature over deep cold sea occurs in the region, and the velocity of sound in this region decreases with depth.

D. Deep Isothermal Layer

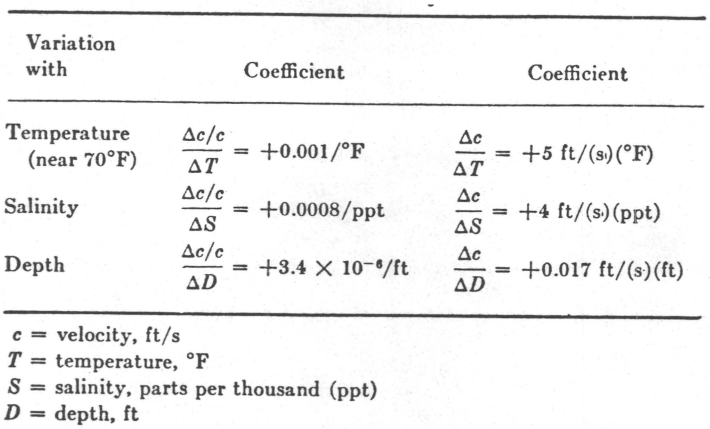
In this region the seawater temperature stays nearly constant at $4^{o}C$. The increase of speed of sound in this layer is almost linear with increase of depth. This deep isothermal layer extends to the bottom of the sea.

The velocity profile varies as the seasons and geographic locations change. Figure 2.4 shows the velocity profile for the four seasons at a fixed geographic location and the velocity profiles of different ocean areas of the world are shown in Fig. 2.5. With all these velocity profiles, there exists a minimum velocity point for each of them, and soundwave tends to be focused along at this depth. The minimum velocity point could vary with latitude as shown in Fig. 2.6.

In general, the velocity profile in shallow water tends to be extremely unpredictable due to the variables such as the surface heating and cooling, water currents, and changes of salinity caused by nearby fresh water sources. The variable temperature profiles in shallow water at various locations of the Eastern seaboard of the United States and Gulf Coast are shown in Fig. 2.7.

2.3 PROPAGATION PATH OF SOUND IN SEA WATER

Due to the variations of the speed of sound in the sea, the propagation path of a soundwave can best be described by the ray theory. Like its counterpart in optics, ray theory presents a picture of the soundwave propagation in the form of a ray diagram. The ray theory approach works well in predicting the behaviour of the propagation path of the soundwave in open sea, however, it does have limitations under certain conditions. The ray theory does not yield good solution for the conditions when the radius of curvature of the ray or the pressure amplitude changes appreciably over the distance of one wavelength. Therefore, the ray theory description of the propagation of sound in the sea is restricted to the sound of high frequency or short wavelengths. The propagation path described by the ray theory, employing Snell's law, can be plotted using a digital computer to gain more insight. A typical computer produced ray path diagram generated by a source in a mixed layer is displayed in Fig. 2.8.



Variation with	Coefficient	Coefficient
Temperature (near 70°F)	$\frac{\Delta c/c}{\Delta T} = +0.001/°\text{F}$	$\frac{\Delta c}{\Delta T} = +5\ \text{ft}/(\text{s})(°\text{F})$
Salinity	$\frac{\Delta c/c}{\Delta S} = +0.0008/\text{ppt}$	$\frac{\Delta c}{\Delta S} = +4\ \text{ft}/(\text{s})(\text{ppt})$
Depth	$\frac{\Delta c/c}{\Delta D} = +3.4 \times 10^{-6}/\text{ft}$	$\frac{\Delta c}{\Delta D} = +0.017\ \text{ft}/(\text{s})(\text{ft})$

c = velocity, ft/s
T = temperature, °F
S = salinity, parts per thousand (ppt)
D = depth, ft

Table 2.1 Approximate coefficient of sound velocity [8].

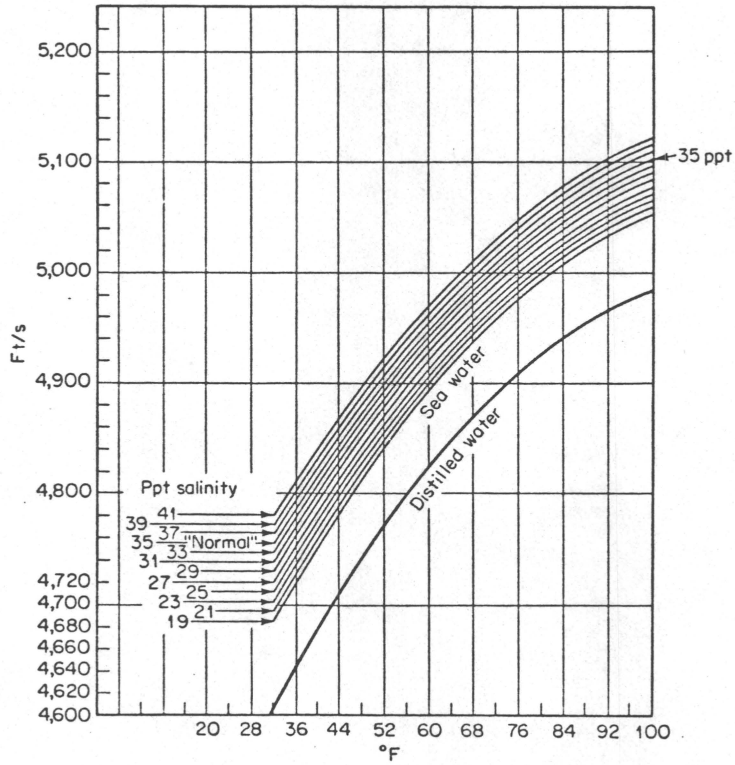


Fig. 2.1 Sound velocity in distilled water and in seawater at zero depth for various salinities and temperatures[14].

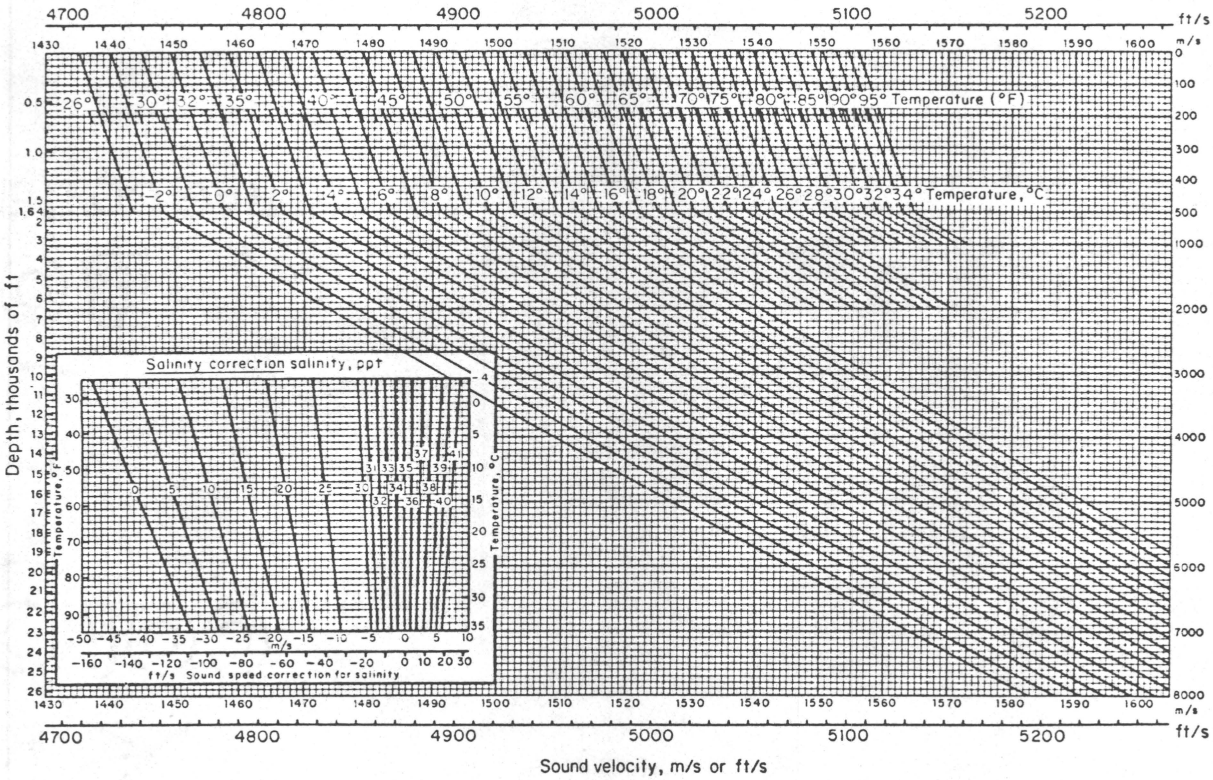


Fig. 2.2 Sound velocity as a function of depth and temperature. The inset gives the correction salinities different from 35ppt [15].

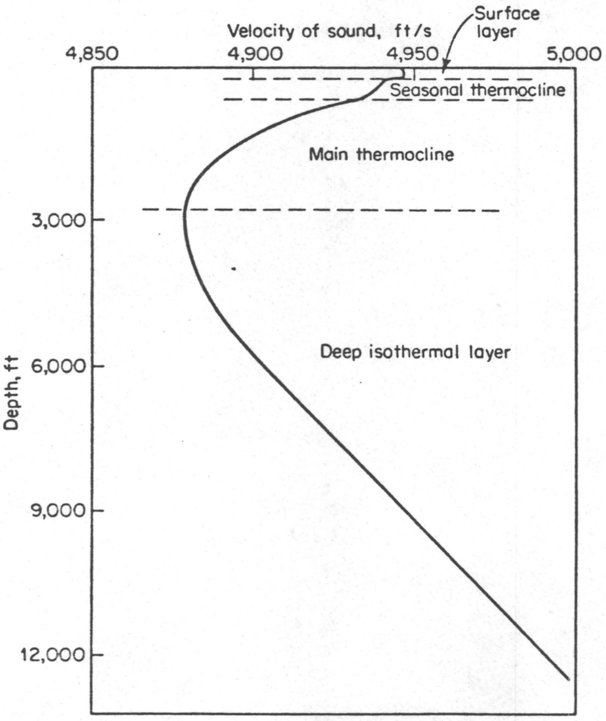


Fig. 2.3 Typical deep sea velocity profile divided into layers [8].

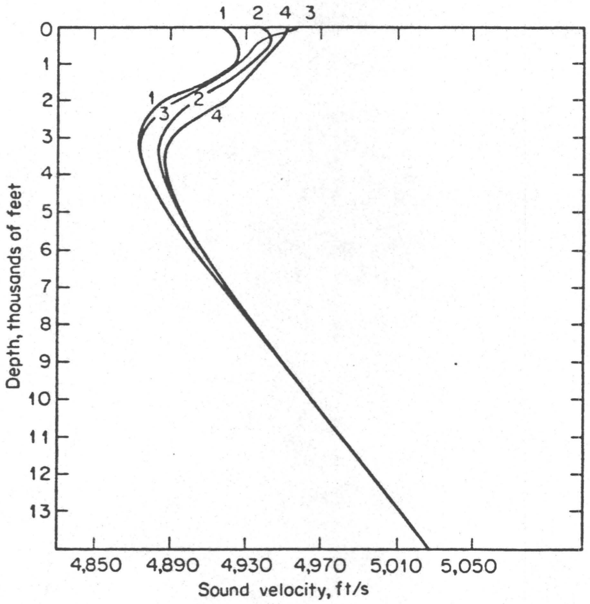


Fig. 2.4 Average velocity profiles in different seasons in an area halfway between Newfoundland and Great Britain. Latitudes 43° to 55°N, logitudes 20° to 40°W. [16].
(1) Winter, (2) Spring, (3) Summer, (4) Autumn.

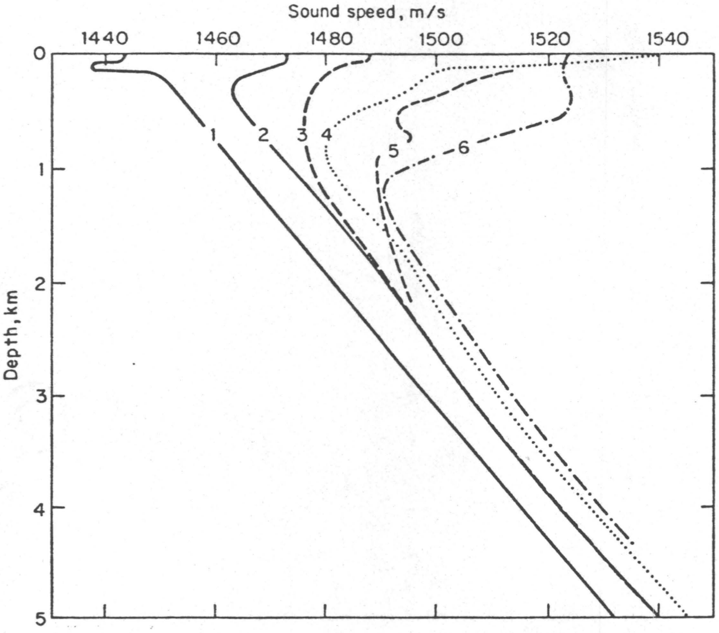


Fig. 2.5 Velocity profiles for the deep-ocean areas of the world [17].
1. Antarctic ocean (60°S).
2. North Pacific, high latitudes (45° to 55°).
3. Southern oceans, high latitude (45° to 55°).
4. Pacific and South Atlantic, low latitudes (4o°).
5. Indian Ocean under influence of Red sea outflow.
6. North Atlantic under influence of Mediterranean sea outflow.

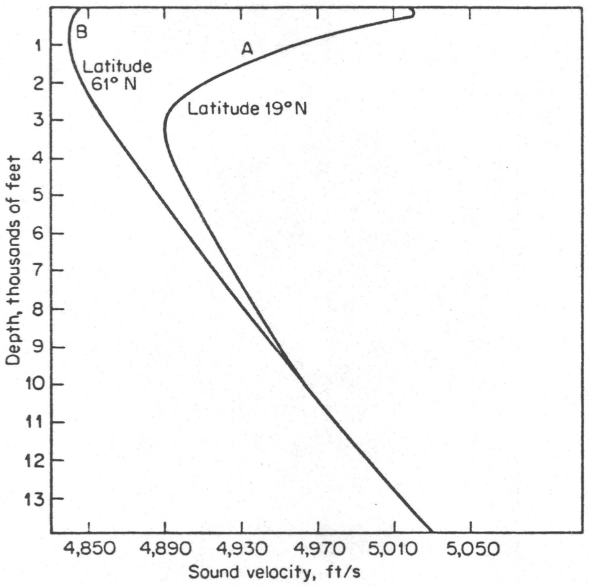
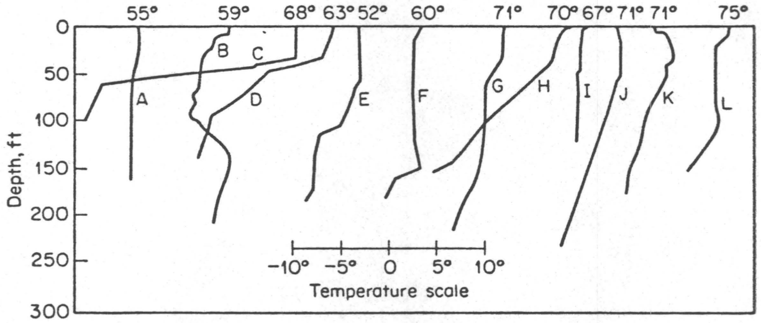


Fig. 2.6 Velocity profiles in different latitudes [16].
A. 18°50'N, 30°01'W, spring. B. 61°02'N, 34°01'N, spring.



Profile	Latitude	Longitude	Month
A	41°N	67°W	Aug.
B	40	71	July
C	42	70	Aug.
D	42	69	Aug.
E	42	70	Oct.
F	40	71	Nov.
G	26	80	Feb.
H	26	80	Feb.
I	29	84	Mar.
J	28	90	Apr.
K	40	71	July
L	28	90	Apr.

Fig. 2.7 Temperature profiles in shallow water of various locations of the Eastern seaboard of the United Steates and Gulf Costs. Top figures are surface temperatures [8].

2.4 TRANSMISSION LOSSES

Intensity of the sound, in general, tends to fall off as the distance between a source and a receiver increases. To characterize the efficiency of transmission, all the losses between a source and a receiver are lumped together and called transmission losses. These losses may be divided into two sections; (A) the spreading losses and (B) the attenuation losses.

A. Spreading Loss

This is due to the geometrical effects which cause the regular weakening of a soundwave as it propagates outward from the source. These include:

i) Spherical Spreading: This type of spreading usually occurs when a nondirectional source is used in transmitting a sound signal. As a result, the extent of the loss of sound intensity is proportional to the square of the distance between the transmitting and receiving devices as shown in Fig. 2.9(a).

ii) Cylindrical Spreading: This kind of spreading typically occurs in a sound channel where the power of the sound signal is distributed uniformly over the surface of a cylinder of a radius equal to the range and height equal to the distance between two boundaries of the sound duct as shown in Fig. 2.9(b). The loss incurred by this spreading is linearly proportional to the range.

iii) Time Stretching: A special type of spreading called time stetching occurs when the transmission of the sound signal is in pulse form. The pulsed signal is usually stretched in time, as a result, the power spreads. This kind of spreading is caused by multipath propagation effects and is particularly acute in long range transmission. If the medium is unbound, this time stretching loss is found to be proportional to the cube of the range.

B. Attenuation Losses

This includes a variety of losses that do not fall into one of the catagories described before, such as absorption losses, scattering losses, and losses due to interaction effects. Among all these losses, the absorption loss is probably the most important factor in attenuating the sound signal. In general, the absorption occurring in seawater can be accounted for by three effects:

i) shear viscosity

ii) volume viscosity

iii) ionic relaxation

At frequency above 1,000 KHz, the shear viscosity and volume viscosity account for the major causes of the absorption. Below 1,000 KHz, the dominant cause of absorption in seawater is the ionic relaxation due to the chemical $MgSO_4$ [5]. At frequency below 5 KHz additional ionic relaxation process occur due to the boron-borate in the seawater [6]. Furthermore, it has been found that the absorptions due to the ionic relaxations and viscosities are both frequency as well as temperature dependent [7]. Figure

2.10 shows the curves of measured absorption coefficient in seawater and distilled water and the theoretical absorption due to shear viscosity.

2.5 SOUND CHANNELS

It has been observed that the maximum effective range of a sonar system is increased substantially under certain conditions. This phenomenon is usually induced by the ducts or sound channels, where sound is prevented from scattering in all directions. A number of such sound channels can be formed under certain conditions, and, in general, they can be classified into three catagories:

A. The Mixed Sound Channel

The creation of this sound channel is caused by a layer of isothermal water, just beneath the surface of the ocean, maintained by turbulant wind action which mixes the near-surface water of the sea. The upward refraction of the soundwave caused by the differential sound speed at various depths traps a portion of the sound energy radiated from a source located in the layer. By successive reflections from the surface of the sea, the sound propagates along the ray path in form of long arces of circle as shown in Fig. 2.8. Sound radiated from a source at a small angle will remain in the sound channel while the sound radiates from a source at a larger angles travel down to the depths of the sea. Just below the channel a shadow zone, with minimum sound intensity due to the source, is produced; consequently, a target hiding in this area will not be detected. There exists a cutoff frequency below which the channel ceases to exists. The cutoff frequency depends on the thickness of the channel as well as the index of refraction of the seawater.

B. Deep Sound Channel

This channel is formed at minimum veleocity depth as mentioned in the discussion of velocity profile. The depth at which the channel occurs varies from about 4,000 ft. at mid-latitudes to near the surface in the polar regions. Two peculiar transmission characteristics of the channel are that the path with the greatest excursion from this depth has the shortest travelling time and the shortest path has the longest travelling time, since the shortest path is at the minimum velocity depth. Due to the time variation taken for various paths to arrive the receiver, the sound received in such a channel tends to increase and cutoff sharply. A typical ray path diagram of the deep sound channel is illustrated in Fig. 2.11.

C. Shallow Water Sound Channel

In this channel, sound propagates to a distance by successive reflections from both the surface and bottom of the sea and is trapped between these two boundaries. The characteristic of the acoustic field in this channel is, therefore, dependent on the sea surface water and the sea bottom.

Within the sound channels described above, occasionally, high sound intensity exist along the ray paths. These high sound intensity areas are formed as the result of the intersection of adjacent sound rays. Although the quality of sound reception improves in these areas, it is difficult to pinpoint their locations, since the locations of these areas are a function of the positions of the source. Figure 2.12 illustrates the migration of this convergent zone as the depth of the source vaires from 3000 to 12,000 ft.

There are other similar sound channels that exist under the conditions described, since they are seasonal in nature, thus the significance of the effects on a sonar system is not as prominent as the ones mentioned, therefore, these secondary sound channels will not be discussed here.

2.6 REVERBERATION

As sound propagates in the sea, scattering occurs due to the inhomogeneities in the physical properties of the medium. Scattering occurs mainly due to the following irregularities and objects in the sea:

A. Irregularities of the ocean surface.

B. Temperature irregularity of the sea.

C. Irregularities that are caused by abrupt changes or fluctuation of the sound speed.

D. Irregularities of the sea bottom.

E. Irregularities of the density of seawater.

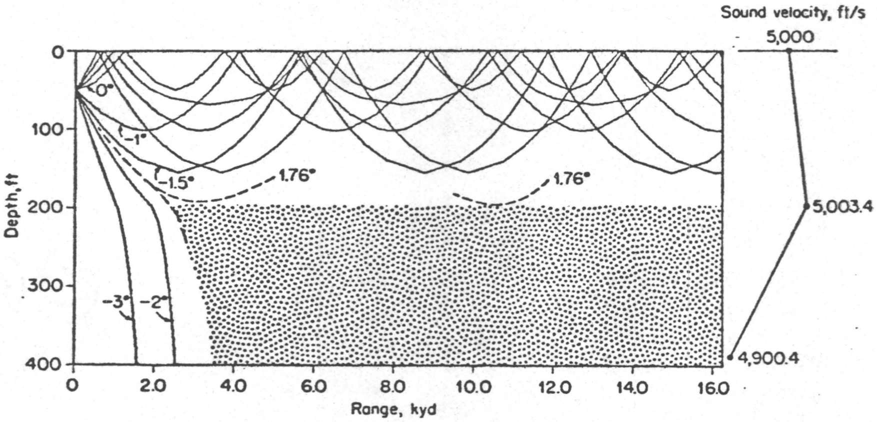


Fig. 2.8 Computer generated ray diagram for sound transmission from a 50-ft source in a 200-ft mixed layer. Rays are drawn at 1° intervals with 1.5° and 1.76° added, for the profile at the right [8].

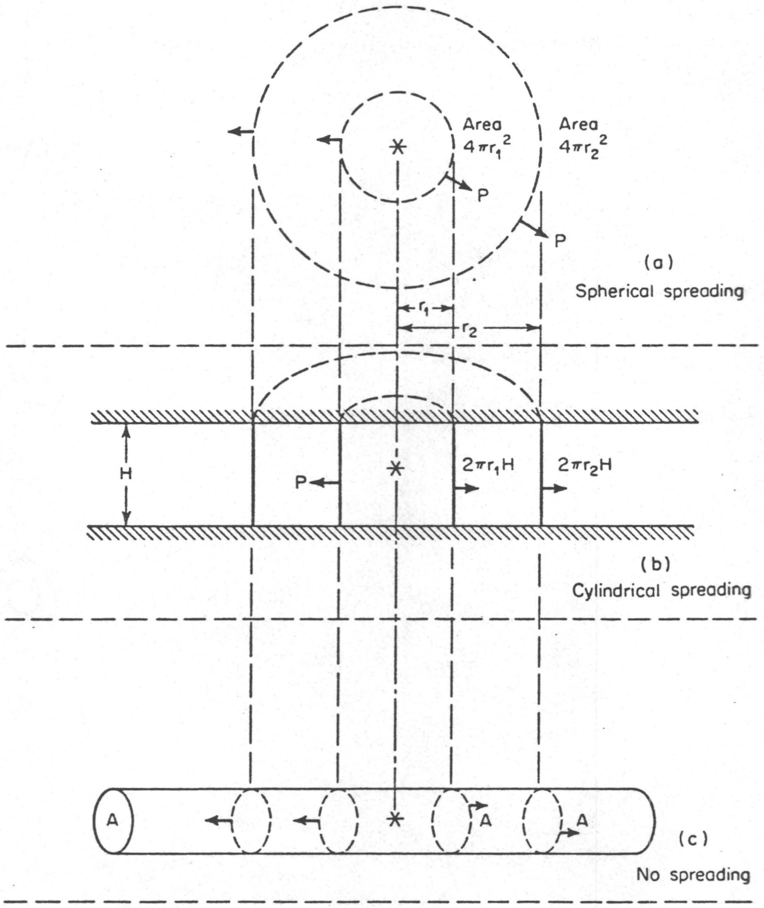


Fig. 2.9 Spreading in (a) an unbounded medium, (b) a medium between two parallel planes, (c) a tube. [8]

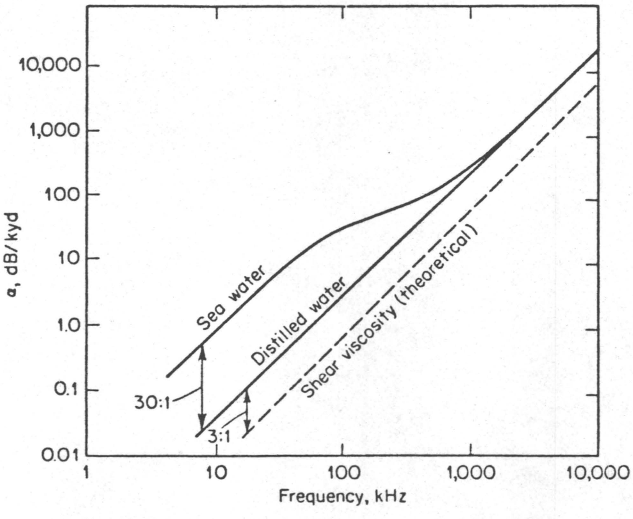


Fig. 2.10 Absorption coeficients in seawater and in distilled water and theoretical absorption due to shear viscosity alone. [8]

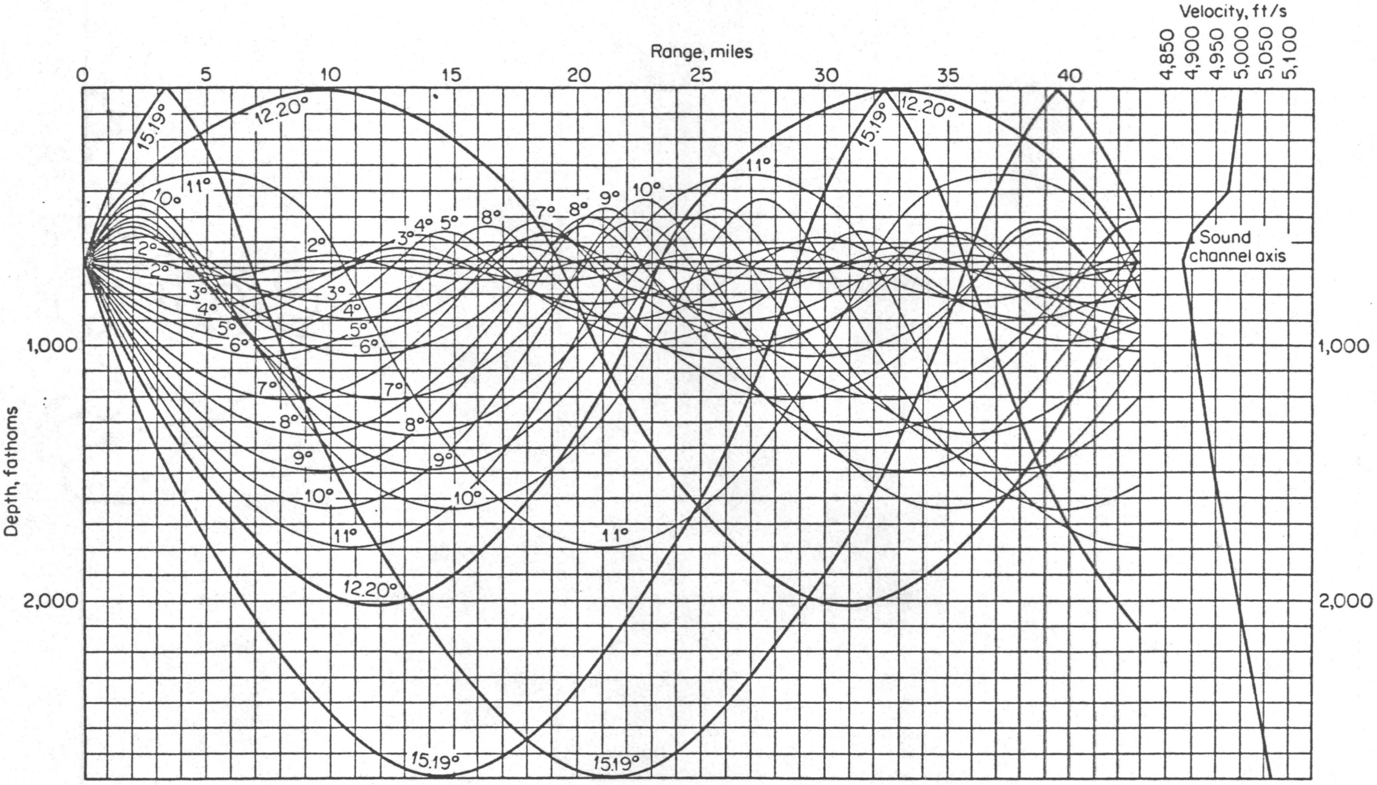


Fig. 2.11 Ray path diagram of transmission in the deep sound channel. [18]

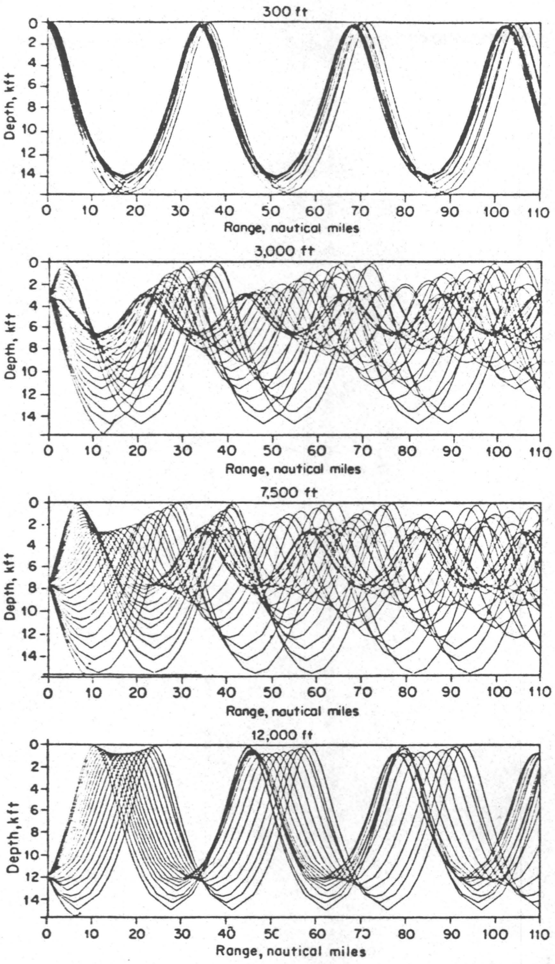


Fig. 2.12 Migration of convergent zone as depth of the source varies.[8]

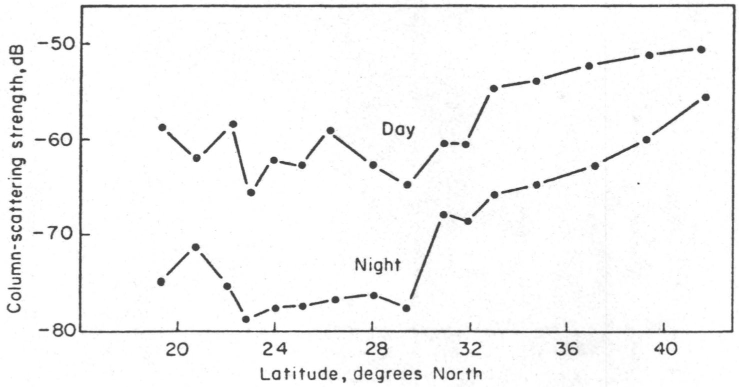


Fig. 2.13 Variation of scattering strength of DSL with latitude by day and by night between Halifax, Nova Scotia, Canada, and Puerto Rico. Frequency band 1.6-3.2 KHz. [8]

F. Irregularities of the sea bottom soil compositions.

G. Solid particles.

H. Fish, marine creature, micro organisms.

I. Air bubbles.

The reception of the sound intensity due to this scattering appears as long duration fluctuation having gradual decay with time. This process, in general, is referred to as reverberation, and can be defined as a time variation of all the scattered sound field observed at the point of reception after the transmission of an acoustic signal. Therefore, the onset of reverberation can be attributed to the transmission of signals in a statistically inhomogenous medium. Since reverberation is caused by the statistical inhomogeneities of the medium and the combined scattered sound field from them, it is very difficult to isolate the cause of sound scattering. However, reverberation can be classified according to the nature of the scatters in the sea. There are three types of reverberation, namely, volume reverberation, surface reverberation, and boundary reverberation. A convenient way of classifying each type of reverberation is found to be as follows:

A. Volume Reverberations: This is due to the scattering by the inhomogeneities found in unbounded space.

B. Layer Reverberation: This is due to the scattering by the inhomogeneities concentrated in a layer.

C. Boundary Reverberation: This is due to the scattering by the interface separating two media.

Since the nature of reverberation is entirely dependent on the physical properties of the medium and the objects in it, a detailed analysis of the reverberation process is difficult due to the statistical, geographical, and time varying features. There are two ways of studying reverberation; first, is the statistical aspect of reverberation and second is the energy aspect of reverberation. Both of these studies involve complicated mathematic formulation which very often fails to give an intuitive feeling toward the subject, thus the detailed formulation of the theories behind these two studies will not be discussed. However, since the specific knowledge of the frequency spectrum and the strength of reverberation are required for the proper design of filters and estimation of the range performance of a sonar system, a particular type of reverberation will be examined below such that the complexities and factors involved in the study of reverberations can be revealed.

The volume scatterers which cause the volume reverberation are not uniformly distributed in depth, but tend to be concentrated in a diffused layer called deep scattering layer (DSL). It has been observed that the cause of volume reverberation is mainly biological in nature, and the organisms responsible are of many different kinds, such as eccphauside, squid, copepods, and fish. It is understandable that a layer as complex as this may be expected to process a scattering strength which varies with frequency, location, seasons, and time of day. Since the deep scattering layer is usually the dominant source of volume reverberation, knowlege of the acoustic characteristics of the layer is important in understanding the nature of the reverberation. These characteristics of the deep scattering layer may be summarized as follows [8]:

A. The scattering strength of the DSL varies over a broad region of ocean due to the changes in biological content in different areas. It has been found by Canadian observers that the scattering strength has a maximum near latitude 45° North and 45° South, and a minimum near 20° North and 20° South, with a secondary maximum near the Equator. However, in a series of other observation opposite result was found that the decreasing the scattering strength occurs bettween latitudes 20° and 45° North. A plot of the scattering strength as a function of latitude is shown in Fig. 2.13.

B. The DSL tends to rise at sunset and desecend at sunrise, and remains at constant depth during the daytime and nighttime. This movement of the DSL may be due to the feeding habits of marine life.

C. The depth of the DSL lies between 180 and 900 meters by day in mid-latitude, and becomes more shallow by night. The depth of DSL in the Arctic does not vary with time of day, it always lies beneath the ice cover.

D. At low frequencies the scattering strength of the DSL is variable and unpredictable. However, it is found that at frequencies near 24 KHz the volume scattering strength within the layer varies from place to place between -70 db to -80 db.

E. It is also found that the scattering strength of the DSL is frequency selective between 1.6 and 12 KHz, with different frequencies occurring at different depths.

F. At some geographic locations the DSL occurs in shallow waters as well. This is mostly due to schools of fish feeding at a particular depth.

In Fig. 2.14 the diurnal migration of DSL is illustrated by plotting the scattering strength of the volume reverberation against the depth. A typical variation of the volume reverberation between daytime and nighttime can be seen in Fig. 2.15; the onset of the reverberation was caused by a 2.0 ms long pulse at a frequency of 41 KHz. During the day there are almost no scatterers at shallow depths; at night they produce intense scattering within the first 200 ft or so below the surface.

The characteristics of reverberation other than its intensity is summarizied below:

A. Amplitude Distribution

The instantaneous reverberation amplitude is found to be Rayleigh distributed [9].

B. Coherence

The coherence of the reverberation received at two different depths at the same location and same time of day decreases as the vertical distance of two receivers increases. It is also found that bottom reverberation near the onset of the bottom return is highly coherent and less so for the deep scattering layer. In both cases the crosscorrelation coefficient decreases for both sources of reverberation with increasing receiver separation and increasing frequency.

C. Frequency Distribution

In general, observations show that the reverberations following a sinusodial pulse do not lie entirely at the frequency of the pulse. The returned frequency spectrum spreads about the centre frequency. The bandwidth of the spectrum is dependent on various moving objects such as the movement of marine organisms, motion of the receiver or transmitter, and water current.

Each one of the characteristics of the reverberation described above plays an important role in the design of a sonar system. For example, the frequency distribution of reverberation is important in the designing of filters for suppressing reverberation, the coherence of reverberation is important in the designing of hydrophone arrays to minimize the reverberation received by receivers, and the amplitude distribution of reverberation is important in determining the effective range of the sonar system.

2.7 AMBIENT NOISE IN THE SEA

Ambient noise is referred to here as the noise of the sea itself. It is the residual noise background in the absence of identifiable sources. To the ear, it often sounds like a low rumble at low frequencies and a cracking hiss at high frequencies. Ambient noise is an important parameter when using a passive sonar system.

Ambient noise, in general, is made up of

A. Physical Effects: tides, waves, rain, seismic disturbance, water currents, and oceanic turbulance.

B. Biological Noise: fish and marine life.

C. Man-Made Noise: shipping traffic and industry.

A. Noise Due to Physical Effects

Tides and waves cause pressure changes of large amplitude at the low frequency end of the spectrum. Fortunately this part of the spectrum lies far below the frequencies of interest. However, the current produced by the tide changes the temperature of water thus changing the output voltage of piezoelectric transducers. Furthermore, the tidal current may cause flow induced noise at higher frequencies.

One of the important causes of low frequency noise in the sea is the constant state of seismic activity of the earth. In addition to the major and minor earthquakes and volcanic activities, a continuous source of low frequency noise is due to the microseisms.

Oceanic turbulance such as caused by seismic activities or other factors can cause additional noise in the sea. One important addition is the turbulent pressure change which may be picked up by pressure sensitive hydrophones located in the turbulent regions.

Surface waves are another source of high frequency noise. These noises are governed by the roughness of the sea surface.

Cavitation, the collapse of air bubbles formed by turbulent wave action also contributes to the noise in the frequency range between 0.1 to 1 KHz.

B. Biological Noise

Biological noise as the name implies, is the noise made by biological organisms in the sea. There are only three groups of marine animals which are known to make sound:

i) Certain kinds of shell fish, the most important noise maker in this catagory is the snapping shrimp. It makes noise by snapping its claws together and produces noise in frequency range of 500 Hz to 20 KHz.

ii) Certain kinds of fish like the croakers and other similar species are also noise makers. They make noise by means of the contraction of drumming muscles attached to air bladder just as the beating of drum.

iii) The marine mammals such as whales and porpoises makes noise by blowing air through the larynx. Propoises also produce frequency modulated whistles.

C. Man-Made Noise

The noise generated by shipping traffic and other industrial activities can be classified as man made noise. It has been estimated that there are, on the average, 1,100 ships under way in the North Atlantic Ocean [10] at any given time. This shipping traffic noise can be heard or detected underwater as far as 1,000 miles away. The shipping traffic is responsible for most of the noise that falls in the frequency range between 50 and 500 Hz.

Industrial activities in harbours, ports, and bays and offshore drilling contribute considerable ambient noise to the waters near the shores.

The frequency spectrum of the ambient noise in the sea due to various types of sources is summarized by Wenz [10] and is displayed in graphic form in Fig. 2.16.

2.8 THE SONAR EQUATION

In this section the parameters concerning the nature of the target, the medium, and the equipment are grouped to formulate an expression upon which the design and performance of a sonar system can be based.

A simple and intuitive sonar equation, similar to that of a radar equation, can be derived as follows: If the total output power P of a sonar is transmitted by an omnidirectional transducer, which radiates power uniformly in all directions,

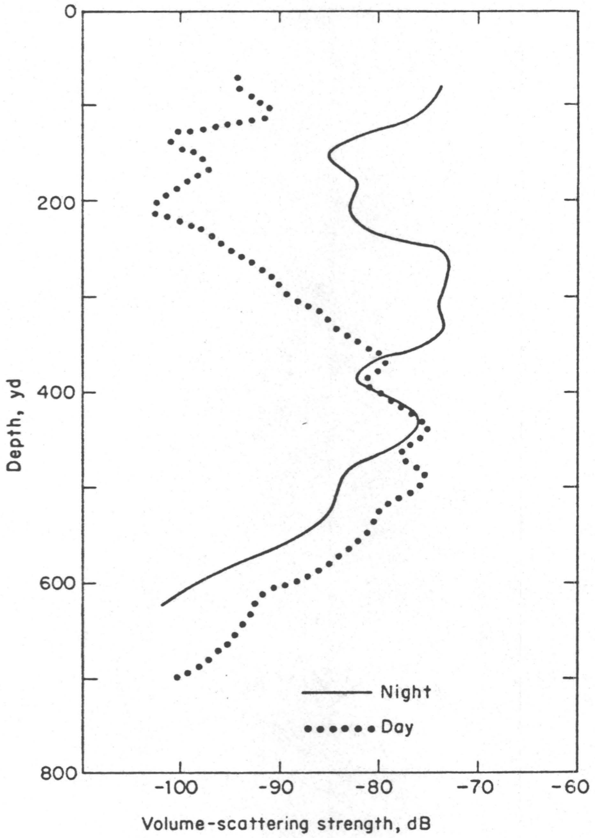


Fig. 2.14 Diurnal migration of volume reverberation with depth. Frequency 5 KHz. [19]

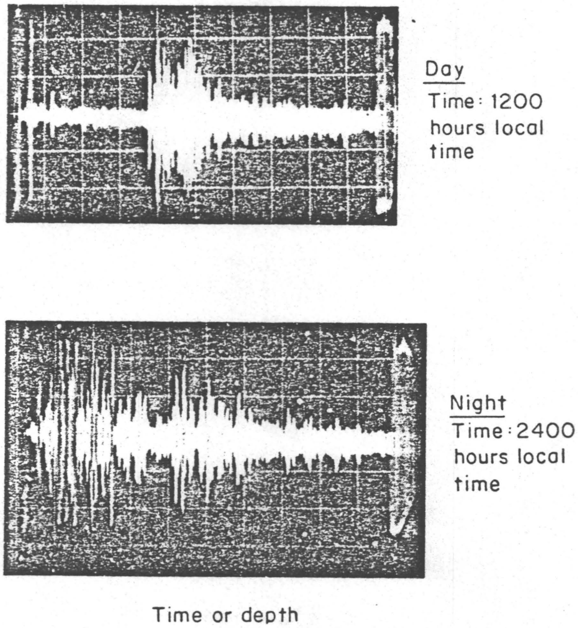


Fig. 2.15 Volume reverberation received on a depth sounder. The bottom echo is seen at the right. The water depth was 600 ft. in Dabob Bay, Puget Sound. The depth sounder had a pulse length of 2ms at a frequency of 41 KHz. The horizontal scale is 23.4 ms per division. [8]

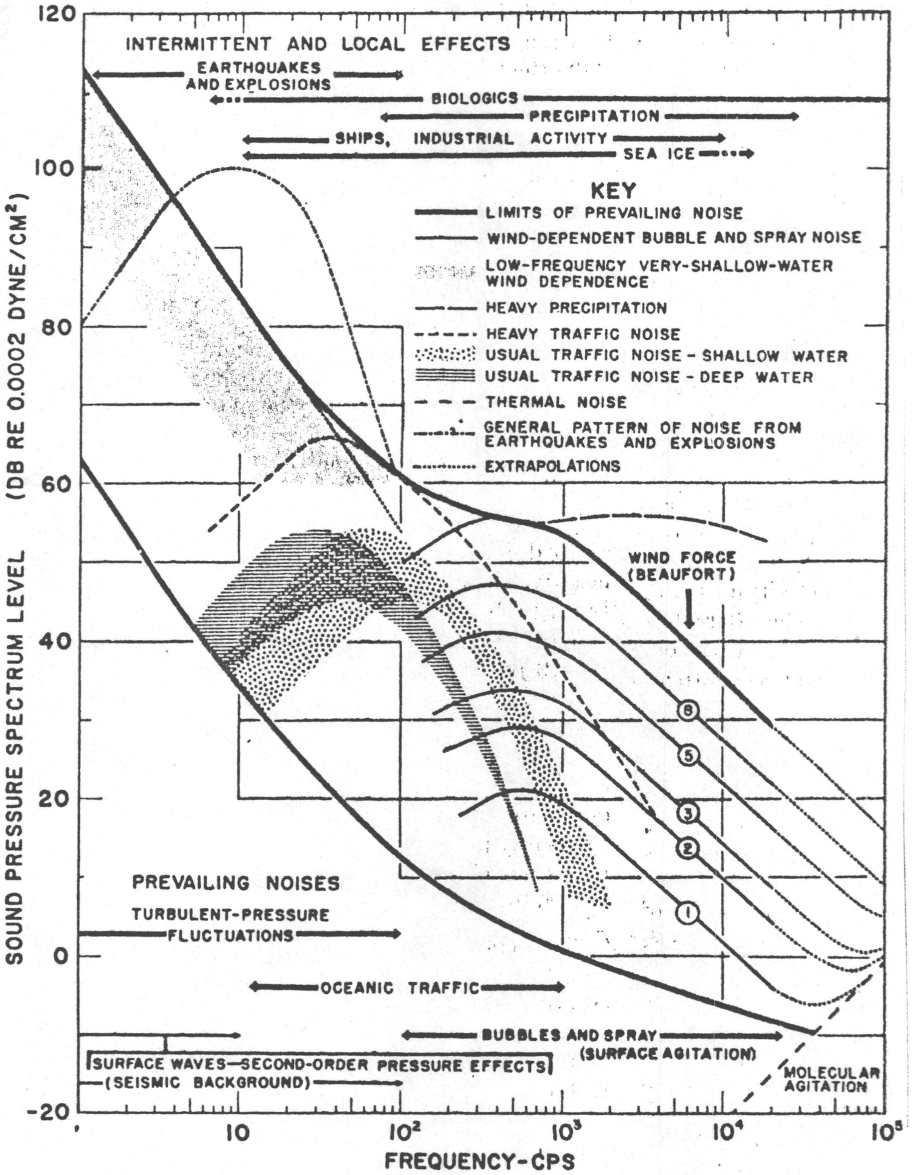


Fig. 2.16 A graphic summary of sea ambient noise spectrum. [10]

assuming spherical spreading, then the intensity of the sound signal at a distance R from the sonar is equal to the transmitted power divided by the surface area $4\pi R^2$ of an imaginary sphere of radius R. Sonars usually employ directive tranducer arrays instead of omnidirectional transducer to focus most of the radiated power P into a particular direction. The power gain in the direction where the directive transducer array is pointing to an omnidirectional transducer is G. The radiated sound signal intensity is attenuated by a factor T as it propagates toward a target; the sound intensity at the target at range R from a transducer array is

$$P_{dt} = \frac{PGT}{4\pi R^2} \tag{2.2}$$

The target intercepts a portion of the radiated power P_t and reradiates it in the direction of the sonar as shown in Fig. 2.17; the reflected sound signal travels back toward the sonar through the same path as it came, thus the echo intensity at the transducer array is then

$$P_{ds} = \frac{PGT}{4\pi R^2} \frac{P_t GT}{4\pi R^2} \tag{2.3}$$

Here, the same transducer array is assumed for both transmission and reception of sound signal.

The maximum sonar range R_{max} is the distance beyond which the target can no longer be detected; this occurs when the received echo signal just equals to the minimum detectable signal

S_{min}. Thus

$$S_{min} = \frac{P\,P_t\,G^2\,T^2}{(4\pi\,R_{max}^2)^2} \tag{2.4}$$

$$R_{max} = \frac{P\,P_t\,G^2\,T^2}{(4\pi)^2\,S_{min}}^{1/4} \tag{2.5}$$

Equation (2.5) is the sonar equation which describes range performance under the assumption that the radiated power is spherically spreaded.

Other forms of sonar equations exist, however, in general, all the sonar equations do not adequately describe the performance of practical sonars. The predicted values of sonar range are usually optimistic. In many cases the actual range might be less than half that predicted. This discrepancy is mainly due to the unpredictability of the transmission losses incurred between the time of transmission and reception. Moreover, the echo power returned by a target and minimum detectable signal S_{min} are both statistical in nature; this further deteriorates the accuracy of the sonar equations.

The ability of a sonar signal processor to detect a weak echo signal is limited by the noise and reverberation that occupy the same portion of the frequency spectrum as does the signal. The weakest signal the processor can detect is called the minimum detectable signal. Detection of a target is based on establishing a threshold level at the output of the process. If the processor output exceeds the threshold, a signal is said to be present. If the output of a typical sonar signal processor is as shown in Fig. 2.18, the envelope has a fluctuating appearance caused by the sea noise. If the threshold level is set sufficiently high, the envelope does not generally exceed the threshold if sea noise alone is present, but does exceed it if a strong signal is present. If the signal is small, however, it is more difficult to detect its presence. The threshold level must be low if weak signals are to be detected, but it cannot be so low that noise peaks often cross the threshold and give a false indication of the presence of targets. The selection of the proper threshold level is a compromise that depends upon how important it is if a mistake is made either by failing to recognise a signal that is present or by falsely indicating the presence of a signal when none exists.

In the case of active sonar system, the threshold detection is further complicated by the presence of reverberations due to the transmission of a CW pulse. In many cases the reverberation level received by a hydrophone array is in the same order of magnitude or even higher than the actual echo signal. Undoubtedly, the presence of reverberation would increase the false alarm rate of the threshold detection if reverberation is not suppressed. Reverberation is the consequence of transmitting pulsed signal; therefore, it is difficult if not impossible to suppress it. Fortunately, reverberation is narrow-band in nature, centered at the transmitted frequency. A bank of narrow-band filters may be employed to isolate reverberation from the echo signal. This is possible because a transmitted CW pulse would undergo a Doppler shift upon reflecting off a moving target, thus the echo signal frequency is displaced from the transmitted frequency. If the frequency displacement is large enough such that the echo signal and reverberation fall into two different passbands of the narrow-band filters, then the echo signal would be separated from reverberation. Figure 2.19 illustrates the frequency response of a bank of narrow-band filters and the separation of echo signal from reverberation. When this is achieved, a higher threshold level could be used at the outputs of nth and (n+1)th filters to minimize the false alarm rate. since both the echo and reverberation have been found to fall off with range in the manner as shown in Fig. 2.20, at the point where the echo signal is equal to reverberation level, the outputs of nth and (n+1)th filters can no longer be used for detection, and the operation range for these two filters is said to be reverberation limited. As for the rest of the filters, the reverberations has no effect on them, and their operating range is, then, limited by the sea noise.

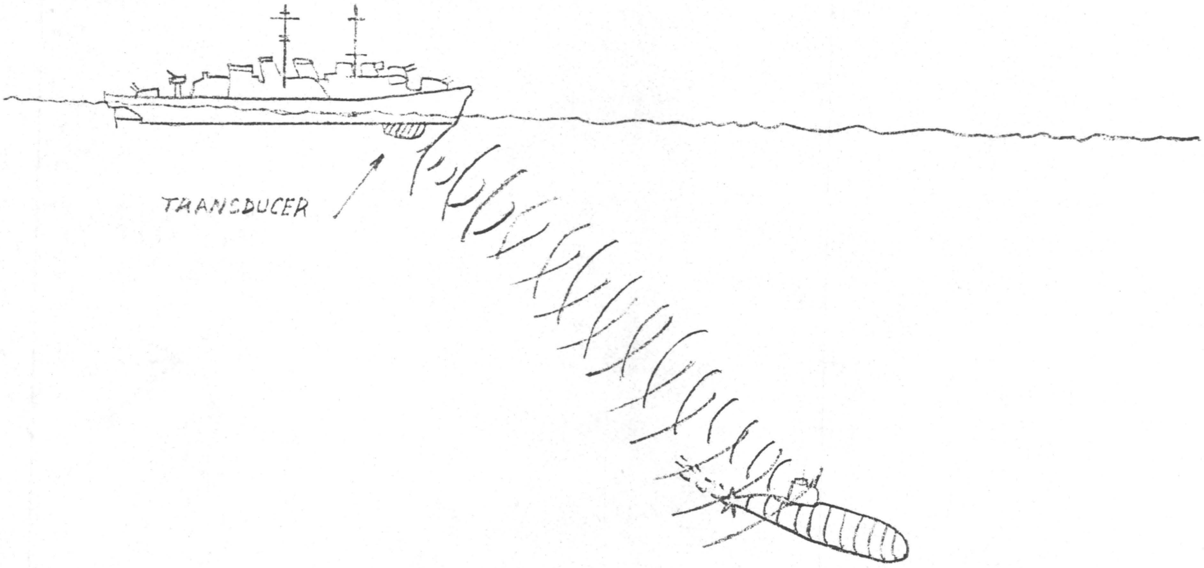


FIG. 2-17 TARGET DETECTION USING SONAR SYSTEM

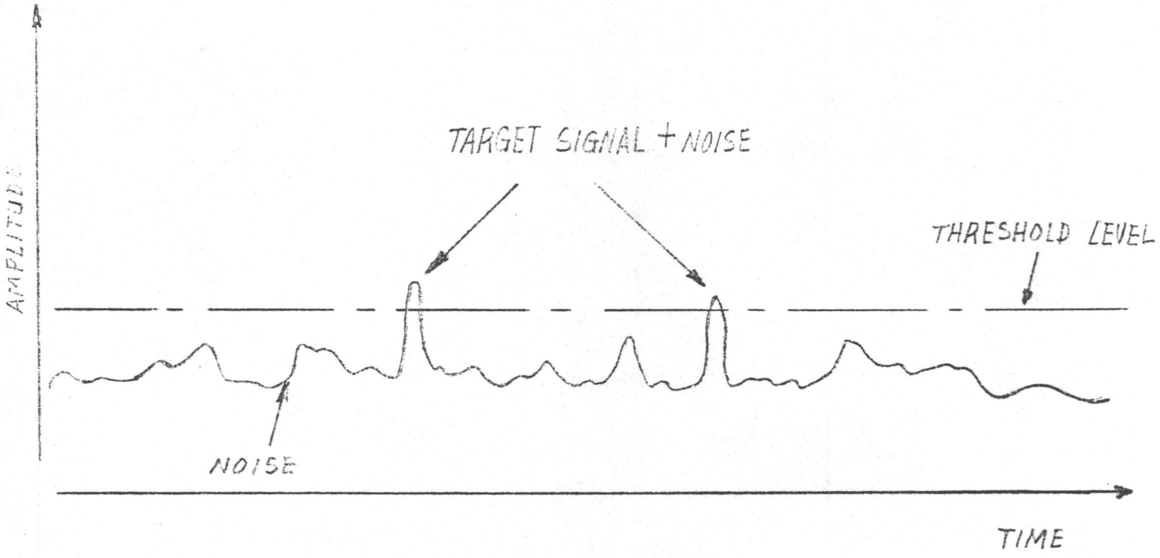


FIG. 2-18 A TYPICAL SIGNAL PROCESSOR OUTPUT

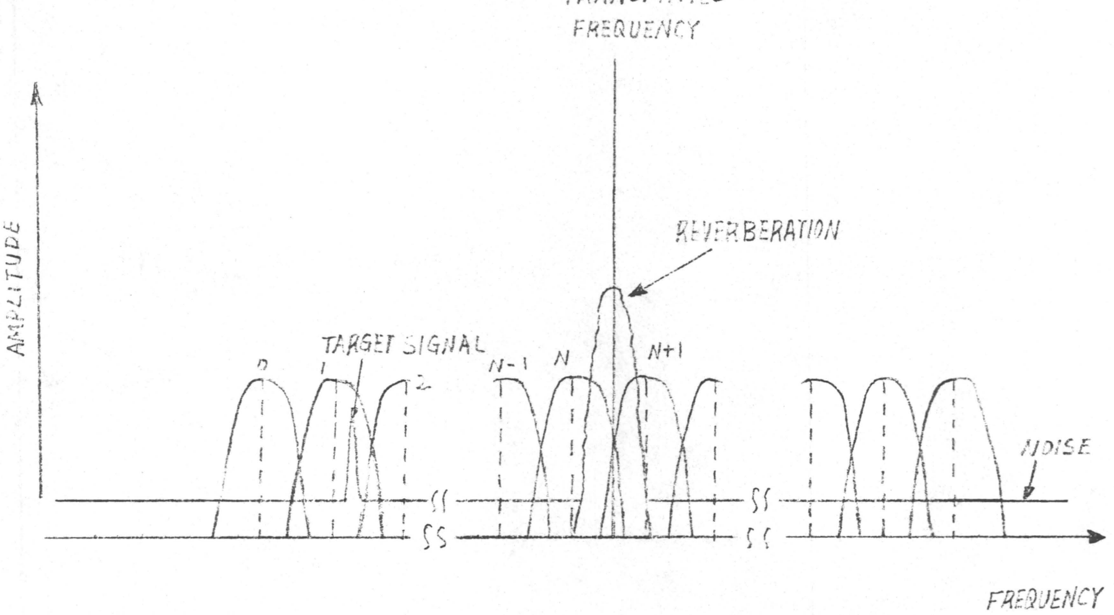


FIG. 2-19 SEPARATION OF REVERBERATION AND A TARGET SIGNAL USING A BANK OF NARROWBAND FILTERS

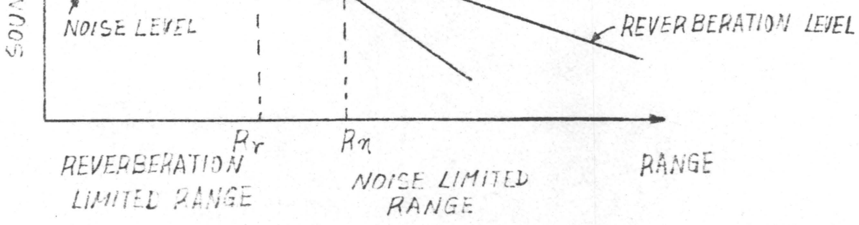


FIG. 2-20 ECHO, NOISE, AND REVERBERATION AS FUNCTIONS OF RANGE

CHAPTER 3

DESIGN OF SONAR SIGNAL PROCESSOR

3.1 SONAR SIGNAL PROCESSOR IN GENERAL

In this chapter, the means by which the information concealed in sonar is extracted, discussed and examined for implementation. Numerous signal processing systems have been developed for active sonar systems for the purpose of detection and location. Depending on the use of the sonar system, the signal processing technique adopted for a sonar system may vary, but the physical configuration of sonar systems are alike.

The concept of resolution cells is the most commonly implemented configuration of a sonar system for the purposes mentioned above. The resolution cells divide all the immediate space surrounding the sonar system into cells, or elements of volume, each having the largest volume within which the sonar is not capable of distinguishing the presence of more than one target. That is, if two targets are within the same resolution cell, they will appeaer as a single target. For the sonar system to distinguish two targets as separate objects, the two targets must occupy two different resolution cells. A diagram of an azimuthal resolution segment is depicted in Fig. 3.1. This is one of the simplest forms of resolution cell, capable of distinguishing multiple targets providing the targets are separated in azimuth by an amount greater than the angular resolution of the system, ψ. From Fig. 3.1, it is clear that the number of azimuthal resolution segments, N, associated with the system is

$$N = \frac{360}{\psi} \tag{3.1}$$

Twenty to 50 azimuthal segments are usually needed for sonar systems that search the entire 360 degrees. Each segment has its own transducer (or transducer array) responsible for the generation and the reception of sound signals in the segment.

A simplified block diagram for a sonar system with multiple resolution cells is shown in Fig. 3.2. It consists of hydrophone arrays connected to a display or a detector unit through a signal processor. For serial processing, the signals received from all resolution cells are multiplexed in time into the same signal processor, and the outputs of the processor are multiplexed to preassigned locations for each resolution cell on a display unit. For parallel processing, the received signals from all resolution cells are simultaneously processed by a number of signal processors, and the outputs of the processors may be multiplexed to a display unit as in the case of serial processing.

Either analog or digital circuits may be employed in the implementation of signal processors. However, until recently, the design of underwater acoustic signal processors were centered around the analog technology. Because of the complexity, difficulties in maintenance, and size of early digital circuits, digital signal processors were mostly confined to laboratories for simulating the analog circuit designs. Now, with the advent of large scale integration (LSI) circuit technology, use of digital circuits in a signal processor has become increasingly attractive. Digital system design employing large scale integrated digital circuits have several advantages over the analog circuits; these include reduction in the system complexity, reduced size, smaller power consumption, lower cost of manufacturing and increased system reliability.

In general, an active sonar system operates as follows (see Fig. 3.3). The transmitter generates a pulsed CW signal at frequency f_o, and sound energy radiates out into the sea through the transducers. Some of this energy hits targets and is scattered in the direction of the sonar system where it is received by the hydrophone array. The received signal may be shifted in frequency from the transmitted frequency by an amount $\pm f_d$, provided that the target is in motion relative to the sonar. The sign of the shifted frequency f_d is determined by the direction in which the target is travelling, approaching or receding with respect to the sonar, a positive sign implies an approaching target and a negative sign implies a receding target.

Once the received echo at frequency $f_o \pm f_d$ enters the sonar, it is heterodyned in the mixer with a signal of frequency f_o to produce a Doppler frequency f_d. This Doppler frequency is then amplified to a level where it can operate an indicator or display device. In practical cases, the expected range of Doppler frequency shifts is usually much wider than the frequency spectrum occupied by the signal energy. Therefore a wideband amplifier is needed to accommodate the expected range of frequencies. Normally, this would result in an increase in threshold due to noise and reverberation. As mentioned before, this problem can be solved by using a bank of narrow band filters spaced throughout the expected range of frequencies to improve the signal-to-noise ratio. The frequency response of the filter bank is shown in Fig. 2.19. Because of the variations of the characteristics of the components used in analog circuits, it is desirable to process the echo signals in each resolution cell serially by the same processor such that all the received signals pass through the same mixer, amplifier, filters, and are thus treated equally.

Due to the amount of computation and the limited speed of the LSI digital circuit, parallel processing is commonly used in digital signal processing system for sonar. The system configuration of a digital signal processor with N identical channels is depicted in Fig. 3.4. Each processor is responsible for processing the echo signals received in its own resolution cell. The signals are first sampled and converted into digital data and then stored in a scratchpad memory for later use. The processors, controlled by hardwired digital logic circuits or software algorithms retrieve the samples sequentially from the

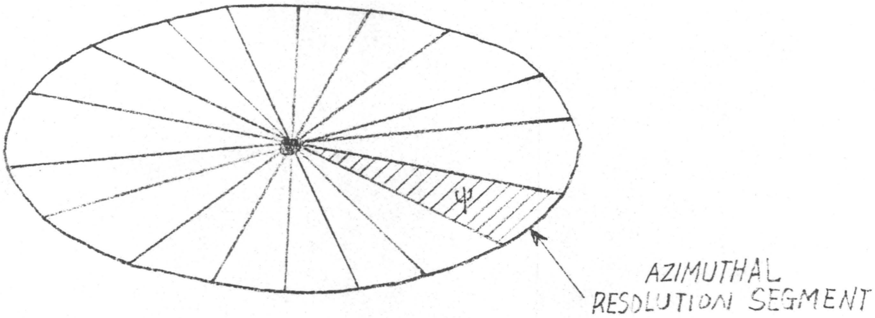


FIG. 3-1 AZIMUTHAL RESOLUTION SEGEMENT

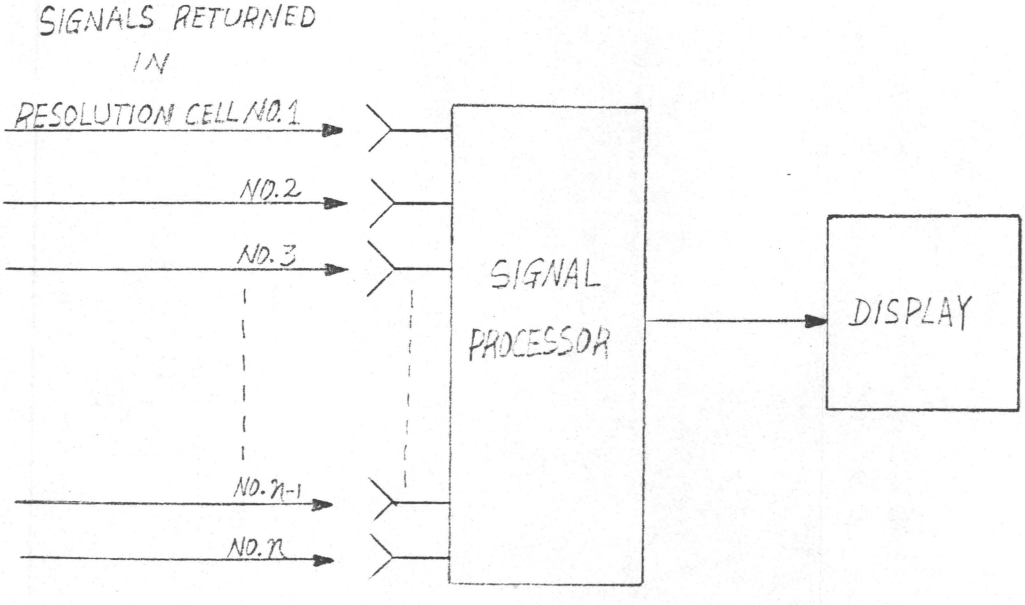


FIG. 3-2 SIMPLE SONAR SIGNAL PROCESSOR

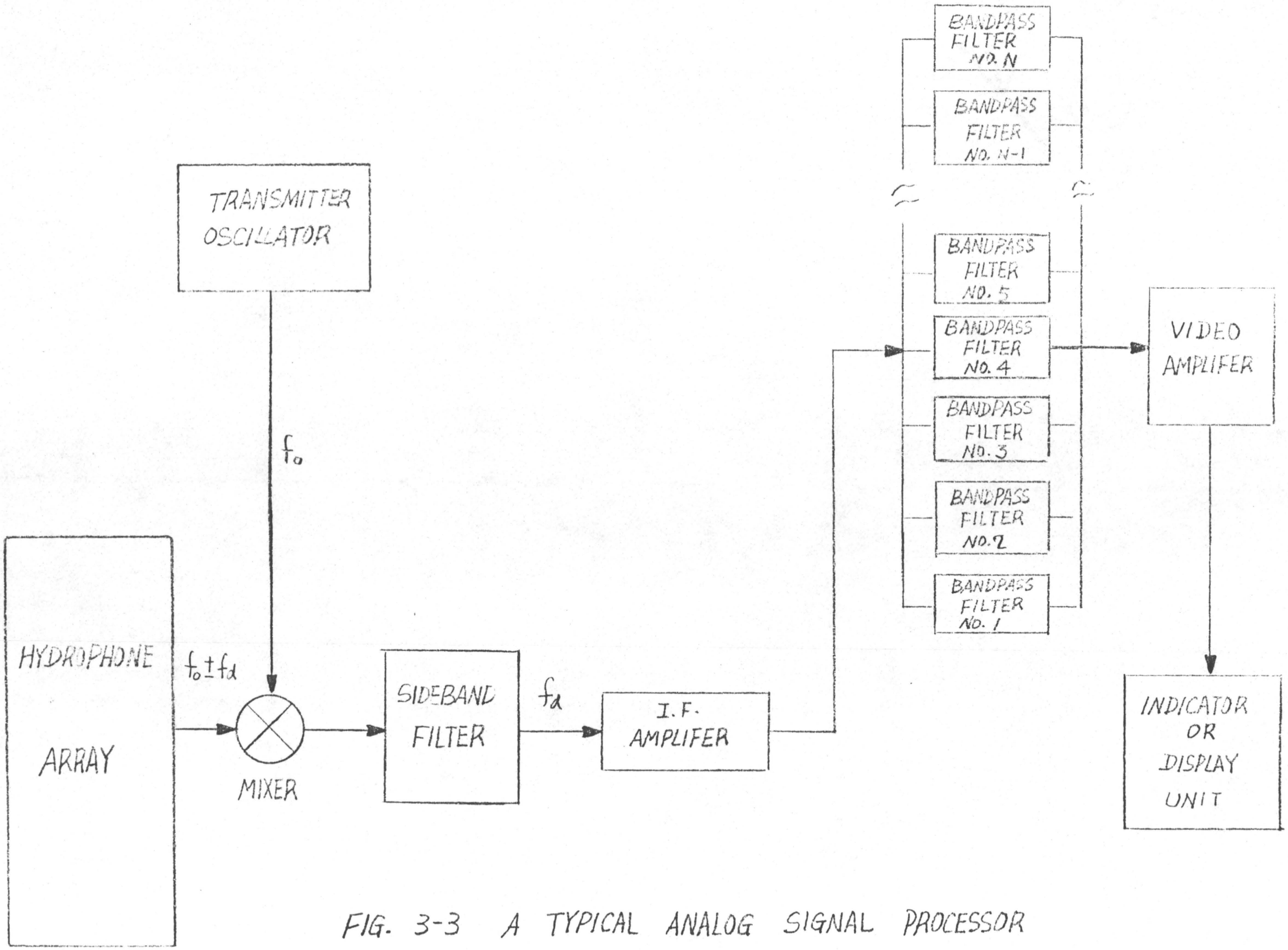


FIG. 3-3 A TYPICAL ANALOG SIGNAL PROCESSOR

* f_d = DOPPLER FREQUENCY

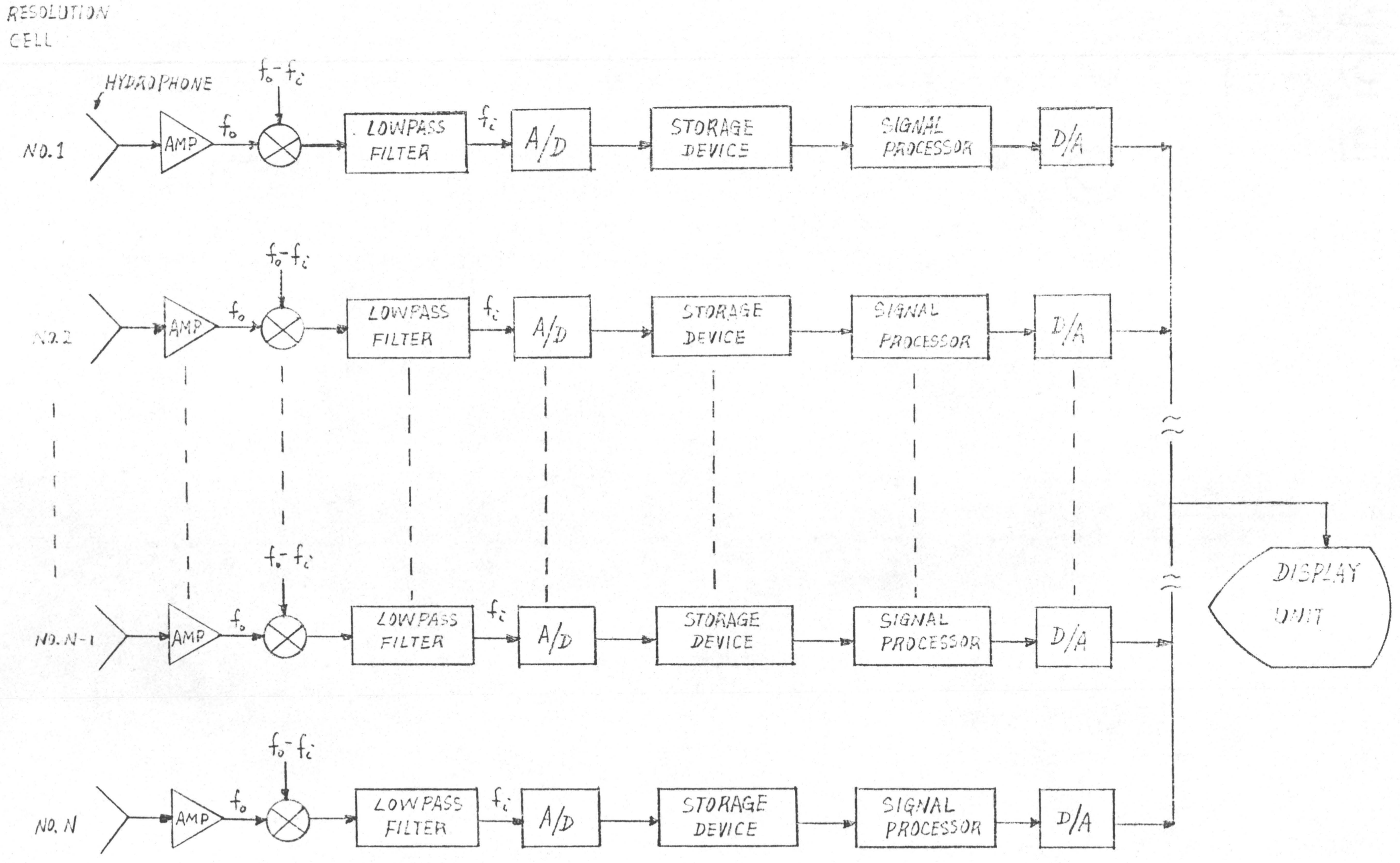


FIG. 3-4 BLOCK DIAGRAM OF A DIGITAL SONAR SIGNAL PROCESSING SYSTEM WITH N RESOLUTION CELLS.

scrathpad memory for processing, and the results are output to a detector or display unit.

3.2 PROCESSING TECHNIQUE

Depending on the functions and requirements of a signal processor, numerous different techniques can be employed to process the received signals. An optimum technique known as the matched filter [11] is briefly described here.

For the purposes of discussion, the echo signals are assumed to be the transmitted signal s(t) plus the white noise w(t). Then, the Fourier transform of the output signal $s_o(t)$ of the filter is equal to H(f) S(f), and $s_o(t)$ is given by the inverse Fourier transform

$$s_o(t) = \int_{-\infty}^{\infty} H(f)\ S(f)\ \exp(j2\pi ft)df \qquad (3.2)$$

where H(f) is transfer function of the filter and S(f) is the Fourier transform of the transmitted signal s(t). Furthermore, the spectral density $S_{no}(f)$ of the filter noise $N_o(t)$ is equal to the spectral density of the input noise w(t) times the squared magnitude of the transfer function H(f). Since w(t) is white with constant spectral density $N_o/2$, therefore, the mean square value of the output noise $N_o(t)$ is

$$E[N_o^2, t] = \int_{-\infty}^{\infty} S_{no}(f)\, df$$

$$= \frac{N_o}{2} \int_{-\infty}^{\infty} |H(f)|^2\, df \tag{3.3}$$

where $N_{o,t}$ is a random variable of $N_o(t)$ sampled at time t. To make the processor more effective, it requires that at time t = T the filter makes the instantaneous power in the ouptut signal $s_o(t)$ as large as possible compared to the mean power in the output noise $n_o(t)$. In essence, this is equivalent to maximizing the output signal-to-noise ratio defined as

$$(SNR)_o = \frac{\left| \int_{-\infty}^{\infty} H(f)\, S(f)\, \exp(j2\pi fT) df \right|^2}{\frac{N_o}{2} \int_{-\infty}^{\infty} |H(f)|^2\, df} \tag{3.4}$$

Now, the problem is to find the transfer function H(f) which will make $(SNR)_o$ a maximum while holding the Fourier transform of the input signal S(f) fixed. The Schwarz inequality states

$$\left| \int_{-\infty}^{\infty} H(f)\, S(f)\, \exp(j2\pi fT) df \right|^2 \leq \int_{-\infty}^{\infty} |H(f)|^2\, df \int_{-\infty}^{\infty} |S(f)|^2 df \tag{3.5}$$

By applying the inequality and using Eqn. (3.4), it follows that

$$(SNR)_o \leq \frac{2}{N_o} \int_{-\infty}^{\infty} |S(f)|^2\, df \tag{3.6}$$

The signal and noise energies in Eqn. (3.6) are assumed to be known, therefore, the output signal-to-noise ratio will be maximum, when H(f) is chosen so that the quality in Eqn. (3.5) holds, then

$$(SNR)_o = \frac{2}{N_o} \int_{-\infty}^{\infty} |S(f)|^2 \, df \qquad (3.7)$$

Since $H_o(f)$ denotes the optimum value of H(f), it follows from Eqns. (3.3) and (3.6) that

$$H_o(f) = k \, S^*(f) \exp(-j2\pi fT) \qquad (3.8)$$

where k is a constant and $S^*(f)$ is the complex conjugate of the Fourier transform of the transmitted signal s(t). Eqn. (3.8) states that except for the time delay factor exp(-j2πfT), the transfer function of the optimum filter is the same as the complex conjugate of the spectrum of the input signal. The impulse response, $h_o(t)$, of the optimum filter can be obtained by taking the inverse Fourier transform of $H_o(f)$ in Eqn. (3.8), that is

$$h_o(t) = \int_{-\infty}^{\infty} S^*(f) \exp[-j2\pi f(T-t)]df \qquad (3.9)$$

since $S^*(f) = S(-f)$

$$h_o(t) = \int_{-\infty}^{\infty} S(-f) \exp[-j2\pi f(T-t)] \, df \qquad (3.10)$$

Eqn. (3.10) shows that the impulse response of the optimum filter is a time-reversed and delayed version of the transmitted signal s(t), that is, "matched" to the transmitted signal, therefore, the name matched filter.

3.3 DIGITAL SIGNAL PROCESSING

It has been mentioned in proceeding sections that certain advantages can be gained by using digital signal processing techniques. In implementing a digital processor by digital computer or special purpose hardware, the input-output relation must be converted to a computational algorithm. The algorithm is essentially specified in terms of a set of basic computation or elements. Since the amount of accessing and storing of data in numerical computation algorithm is generally proportional to the number of arithmetic operations, it is generally accepted that a meaningful measure of complexity, or, of the time required to implement a computational algorithm is the number of multiplications and additions required. In this section the most commonly employed digital signal processing techniques, namely, the fast Fourier transform (FFT), digital filters, and digital correlator are compared for their hardware complexity and computational time required for their realizations. Based on this comparison, one of the three digital processing techniques will then be chosen for implementation.

3.3.1 FAST FOURIER TRANSFORM (FFT)

An optimum matched filter for a pulsed CW signal with white Gaussian noise is the FFT processor. A FFT processor can be employed for determining the Doppler frequency shift contained in returned signals by measuring the frequency separation between the main frequency component of the frequency spectrum of the transmitted signal and the main frequency component of the frequency spectrum of the received signal. For example, if the main frequency component of the transmitted signal is at frequency equal to zero, that is at DC, and the main frequency component of the received signal is 25 Hz, then the Doppler shift of the received signal is 25 Hz. For digital signal processing, the discrete frequency spectrum of the received signal can be obtained by taking the discrete Fourier transform of the signal. Essentially, a FFT processor is a processor which uses the "fast Fourier transform" (FFT) algorithm for computing the discrete Fourier transform of its input signal.

Except for a scaling constant, the discrete Fourier transform of a finite-duration sequence $\Sigma_{n=0}^{N-1}$ x(n) is defined as

$$X(k) = \sum_{n=0}^{N-1} x(n)\, W_N^{kn} \qquad k = 0, 1, \ldots, N-1 \quad (3.11)$$

where X(k) is the kth component of the discrete frequency spectrum of the finite-duration sequence and W_N^{kn} is equal to exp(-j (2πkn/N)). Since both x(n) and W_N^{kn} may be complex, it is clear for each of the N possible values of k, a direct computation of

X(k) requires (n-1) complex additions and N complex multiplications. This means that the direct computation of discrete Fourier transform of N samples requires a total of N(N-1) complex additions and N^2 complex multiplications. However, by using the FFT decimation in time [12], the number of arithmetic operations is reduced to N $\log_2 N$ complex additions and N $\log_2$ N complex multiplications. This represents a considerable saving in the computation effort for large N.

In the case where 16 Doppler frequency shifts resolution over a bandwidth of 375 Hz is required, a 16 complex point baseband FFT processor can be used. For a 40.37 ms CW pulse signal, 16 complex samples spaced 2.52 ms apart are processed at a time, the processor requires 64 complex multiplications and 64 complex additions for each set of transform operations. After the FFT operation, the discrete frequency spectrum contains 16 discrete frequencies ranging from -187.5 Hz to +187.5 Hz in 25 Hz steps. A sampled rectangular pulse and its discrete frequency spectrum are shown in Fig. 3.5(a) and 3.5(b) respectively. The harware realization of a baseband FFT processor is also depicted in Fig. 3.6.

3.3.2 DIGITAL FILTER

A digital filter is functionally no different from an analog filter, since both attenuate unwanted signal at their inputs and pass the desired signal to the output. To realize this

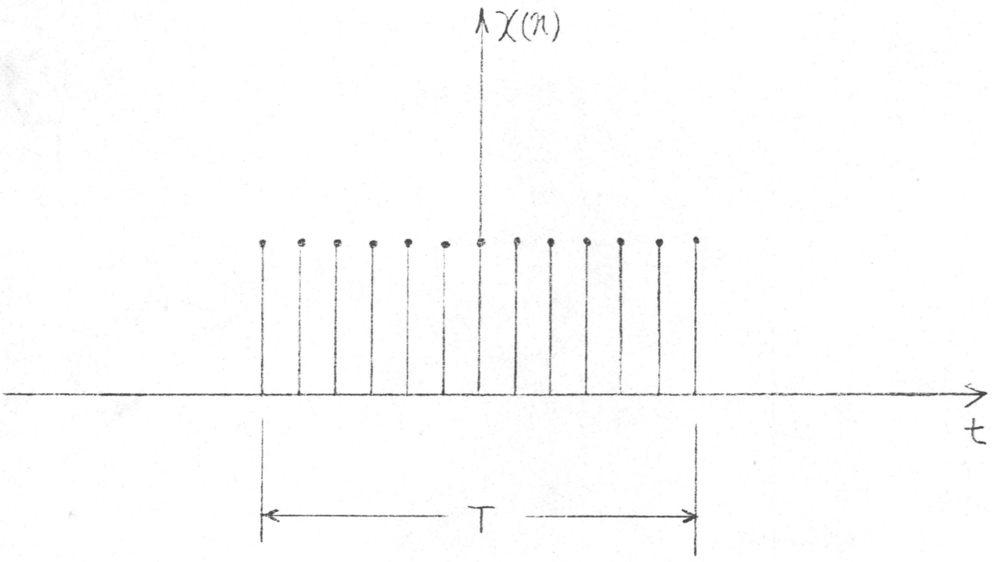


FIG. 3-5(a) A SAMPLED RECTANGULAR PULSE

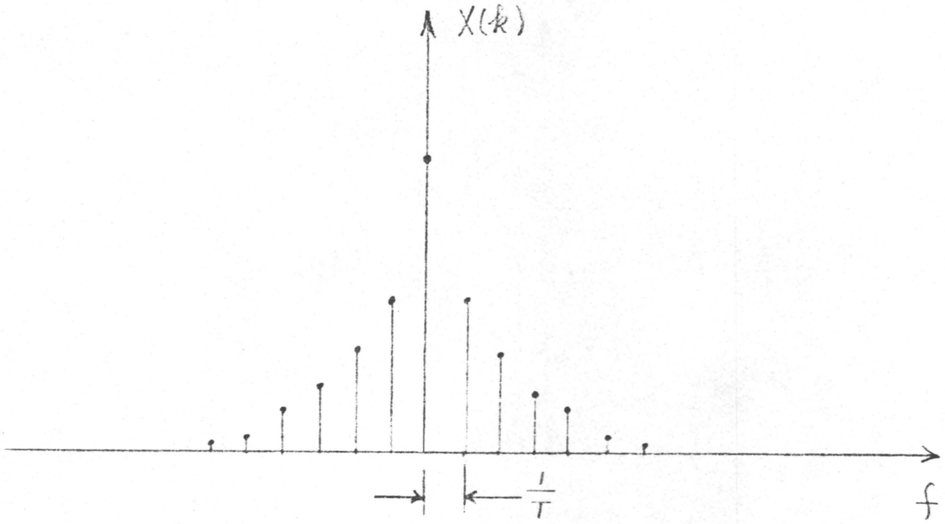


FIG. 3-5(b) FREQUENCY COMPONENTS OF THE ABOVE SAMPLED WAVEFORM

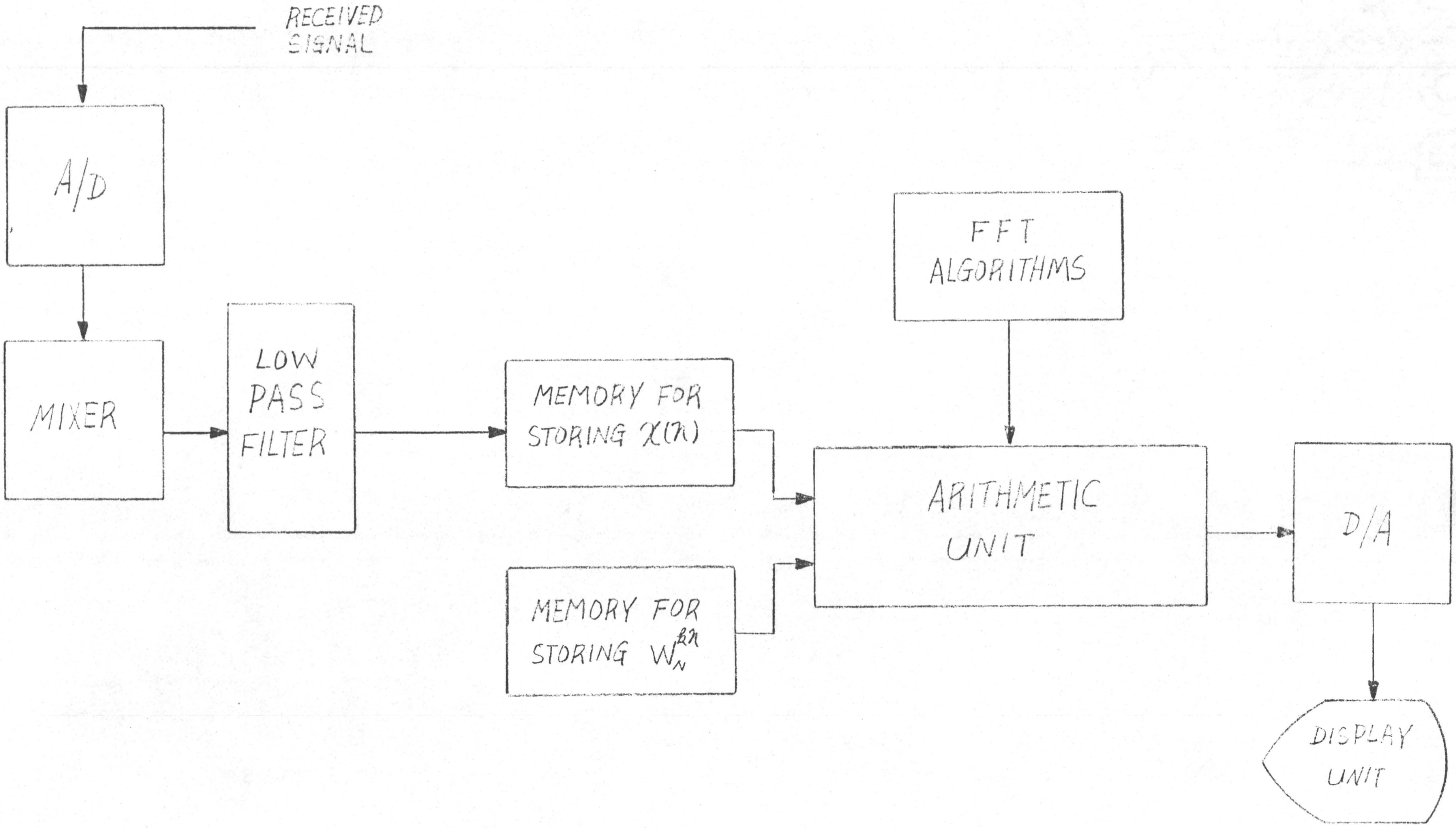


FIG. 3-6 IMPLEMENTATION OF FFT AT BASEBAND

function, an analog filter uses analog components such as capacitors and inductors whereas a digital filter employs storage elements and finite precision arithmetic. The design of a digital filter involves specifying desired properties of the system, approximating the specification using a discrete time system, and realizing the system using finite precision arithmetic.

Usually the digital filter is used to filter a digital signal that is derived from an analog signal by means of periodic sampling. Given a set of specifications, the next step is to find a discrete time linear system having a frequency response falling within the prescribed tolerance. Various design techniques exist that may be used to approximate the desired frequency response of a digital filter. Depending on the choice of the discrete time linear systems and design technique used for the realization, digital filters may be categorized into two types called the infinite impulse response (IIR) system and finite impulse response (FIR) system. A infinite impulse response system approximates the desired frequency response by a rational function. A finite impulse response system approximates the desired frequency response by a polynomial.

Usually, IIR filters have the advantage that a variety of frequency selective filters such as lowpass, bandpass, and highpass can be designed using the traditionally well known analog filter design techniques. That is, once the appropriate kind of filter such as Butterworth, Chebyshev, or elliptic has been specified, the coefficients of the approximating function of the desired digital filter can be obtained by straightforward substitution into a set of design equations. This type of filter does not in general have a linear phase characteristic, however. Thus, the impulse response rings.

In contrast, most of FIR filter design methods utilize optimization techniques which require rather powerful computational facilities. The advantage gained from this more complicated design procedure can be measured in terms of flexibility in the attainable filter response. In addition, FIR filters can have precisely a linear phase response which is generally not obtainable with IIR filters. However, it is usually true that disregarding the phase response, a given amplitude response specification will be met most efficiently, as far as the hardware complexity is concerned, with an IIR filter.

Given the choice of type of filter IIR or FIR, a digital network structure must be chosen for implementing the filter as a computer program or in hardware. Choosing a structure for a specific filter is complicated procedure; it involves the tradeoffs in the hardware or software complexity and difficulty in obtaining the parameters for the filter. Although it is desirable to minimize the length of the register that must be provided to store the filter parameters, this may not be possible for every case. Since almost all the design techniques developed for the specification of the transfer function in terms of parameter assume unlimited parameter wordlength, the accuracy with which the parameters can be specified is limited by the wordlength of the registers. In general, the coefficients used in implementing a given filter will not be exact, therefore, the poles and zeros of the transfer function will be different from the desired poles and zeros. This movement of the poles and zeros away from the desired location may result in a frequency response that is different from the desired frequency response. In some cases when the coefficient quantization errors are large, that is, if minimum register wordlength is used, the filter may fail to meet the designed specifications. In the case of IIR filters, one or more of the poles may move outside the unit circle in Z-plane resulting in an unstable system. Usually, the effect of coefficient quantization is highly dependent on the structure used to implement the system.

In general, the transfer function of an IIR filter can be expressed in the form

$$H(Z) = \frac{\sum_{k=0}^{M} b_k Z^{-k}}{1 - \sum_{k=1}^{N} a_k Z^{-k}} \tag{3.12}$$

where the coefficients a_k and b_k are real constants and Z^{-k} is the delay operator. The input sequence x(n) and output sequence y(n) of a system, whose transfer function is defined in Eqn. (3.12), are related by the constant coefficient difference equation

$$y(n) = \sum_{k=1}^{M} a_k\ y(n-k) + \sum_{k=0}^{M} b_k\ x(n-k) \qquad (3.13)$$

The computational algorithm used to calculate the output sequence y(n) is said to be recursive, since the present sample value of the system output depends on the present and past sample values of the input as well as the past sample values of the output. The realization of the difference equation Eqn. (3.13) may be represented by the signal flow graph shown in Fig. 3.7. Due to the recursive nature of the computational algorithm employed to obtain the system output, IIR filters, in general, are not optimum filters for pulsed CW signals.

On the other hand, the transfer function of an FIR filter is commonly expressed in terms of the impulse respones h(k) as shown

$$H(z) = \sum_{k=0}^{N-1} h(k)\ Z^{-k} \qquad (3.14)$$

The ouptut sequence y(n) of a FIR filter, whose transfer function is defined in Eqn. (3.14), can be obtained by the convolution sum of the impulse response h(k) and the input sequence x(n) as shown by

$$y(n) = \sum_{k=0}^{N-1} h(k)\ x(n-k) \qquad (3.15)$$

The computational algorithm used to realize Eqn. (3.15) is said to be nonrecursive, since the present sample value of the filter output depends only on the past and present sample values of the

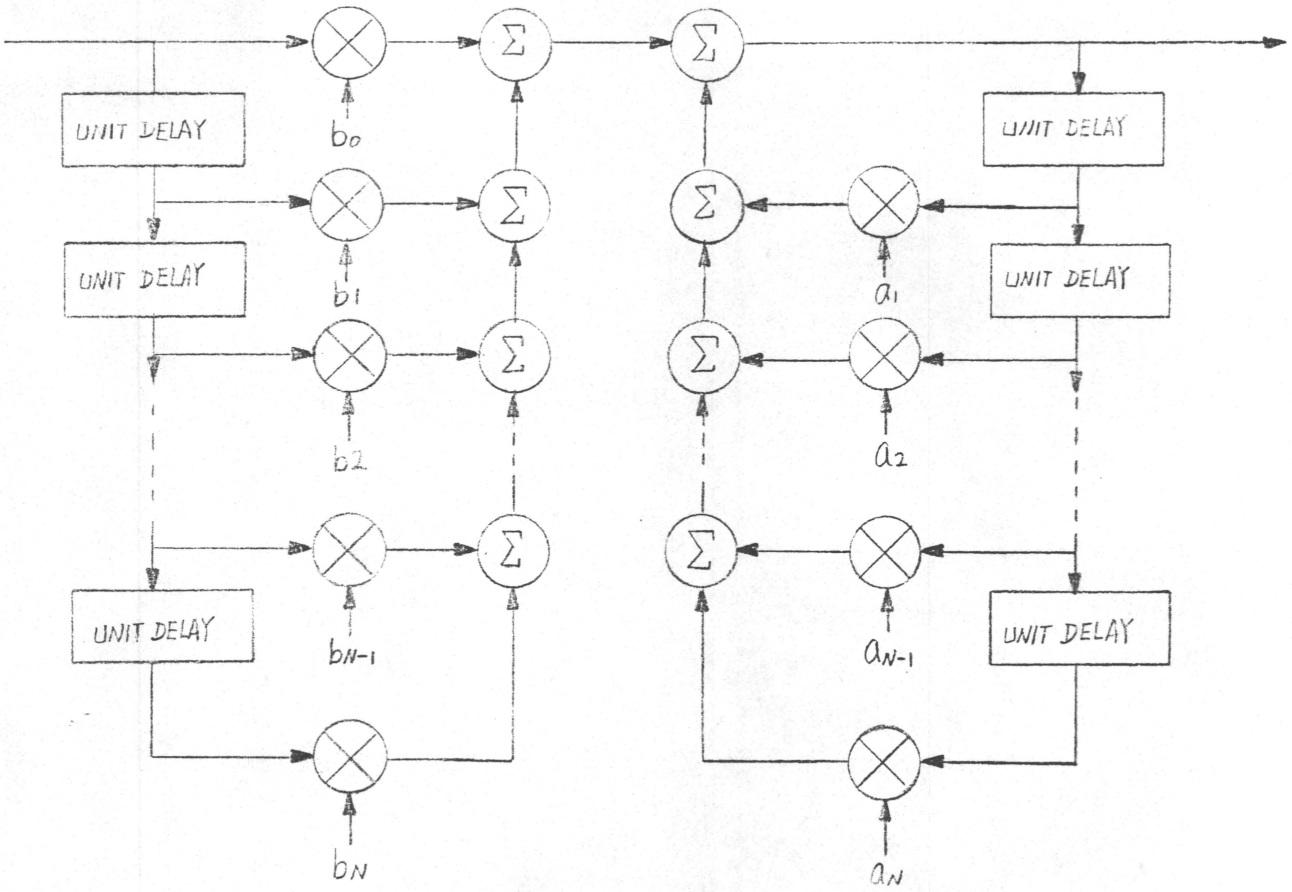


FIG. 3-7 DIRECT FORM OF REALIZATION OF NTH ORDER DIFFERENCE EQUATION

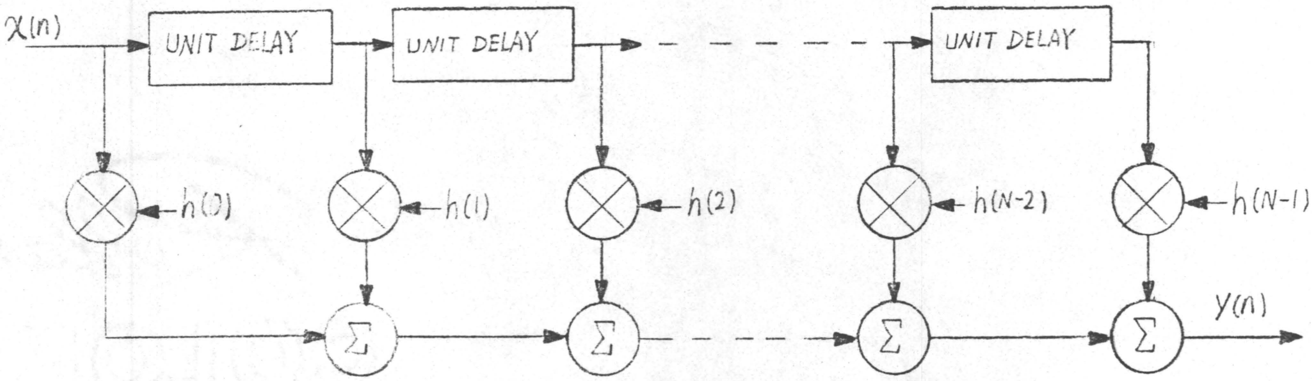


FIG. 3-8 DIRECT FORM OF REALIZATION OF AN FIR SYSTEM

input. The signal flow graph of Eqn. (3.15) is shown in Fig. 3.8. From Eqn. (3.15), it is clear that the impulse response sequence of the system can be chosen to "match" the input sequence. When this occurs, the FIR filter is a matched filter for the expected input sequence.

For complex signal processing, the computations required for the direct calculation of y(n) for each value of n are 2(n-1) real additions and 2N real multiplications. In the case when 16 Doppler frequency filters are required to process a 40.32 ms CW pulse having 16 complex samples spaced 2.52 ms apart, the processor requires 240 real additions and 256 real multiplications for each set of input sequence. The hardware implementation of a baseband FIR digital filter is similar to the one shown in Fig. 3.6.

3.3.3 DIGITAL CORRELATOR

A correlator is optimum filter for a pulsed CW signal immersed in Gaussian noise. The correlator measures the similarities between the reference (usually the transmitted signal) and the received signal. The closer the two signal resemble each other, the higher is the correlator output. To detect one of the various Doppler frequency shifts, that may be concealed in a returned signal, a bank of reference signals can be used to cover the expected range of the Doppler shift. In this case, the reference signals are essentially the Doppler shifted versions of transmitted signal.

The correlation of two signals is defined as

$$R(k) = \sum_{n=0}^{N-1} u(n)\ x(n-k) \qquad (3.16)$$

where u(n) is the reference signal sequence and the x(n) is the input sequence. The process of the correlation of two signals u(n) and x(n-k) is illustrated in Fig. 3.9. Each sample of u(n) in Fig. 3.9 is multiplied by the corresponding input sample below, the sum of these products is the output of the correlator. A correlator which uses the transmitted signal as its reference signal is called the replica correlator. Such a correlator can be extremely versatile, since the replica signal, hence the signal to which the processor is matched to, can be easily changed. Therefore, the correlator is capable of being a matched filter for any paticular waveform which can be used as its reference.

For complex signal processing, the amount of computation required for the realization of Eqn. (3.16) for each value of k consists of 2(N-1) real additions and 2N real multiplications. The computational effort needed for the realization of a digital correlator is the same as the one for the realization of a FIR digital filter. This is expected, since, except the difference in notation, Eqn. (3.15) and Eqn. (3.16) are identical. However, when the term digital filter is used, it usually implies that the signal processing is performed at baseband. Although baseband operations require less computational time, it requires more

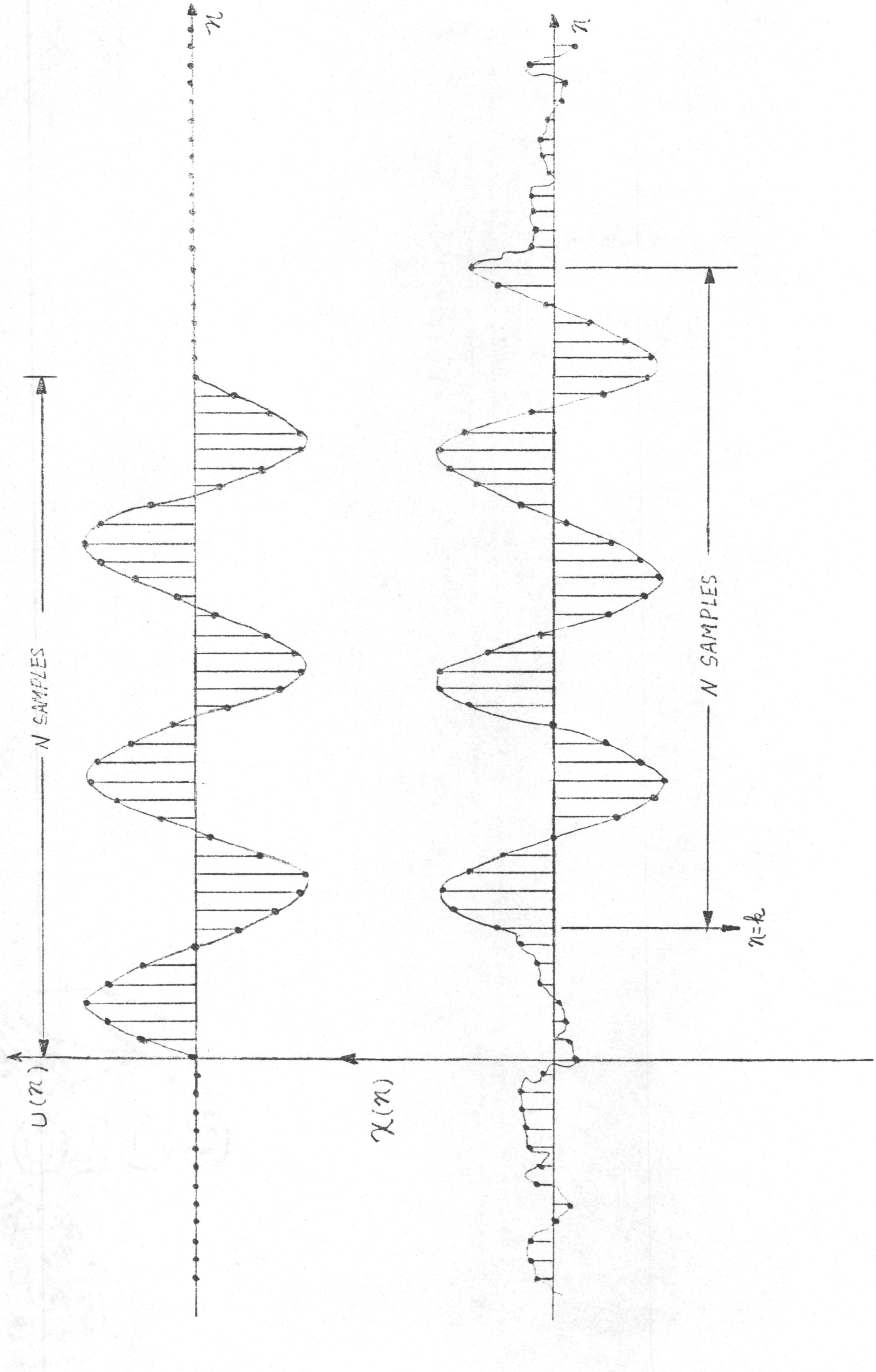
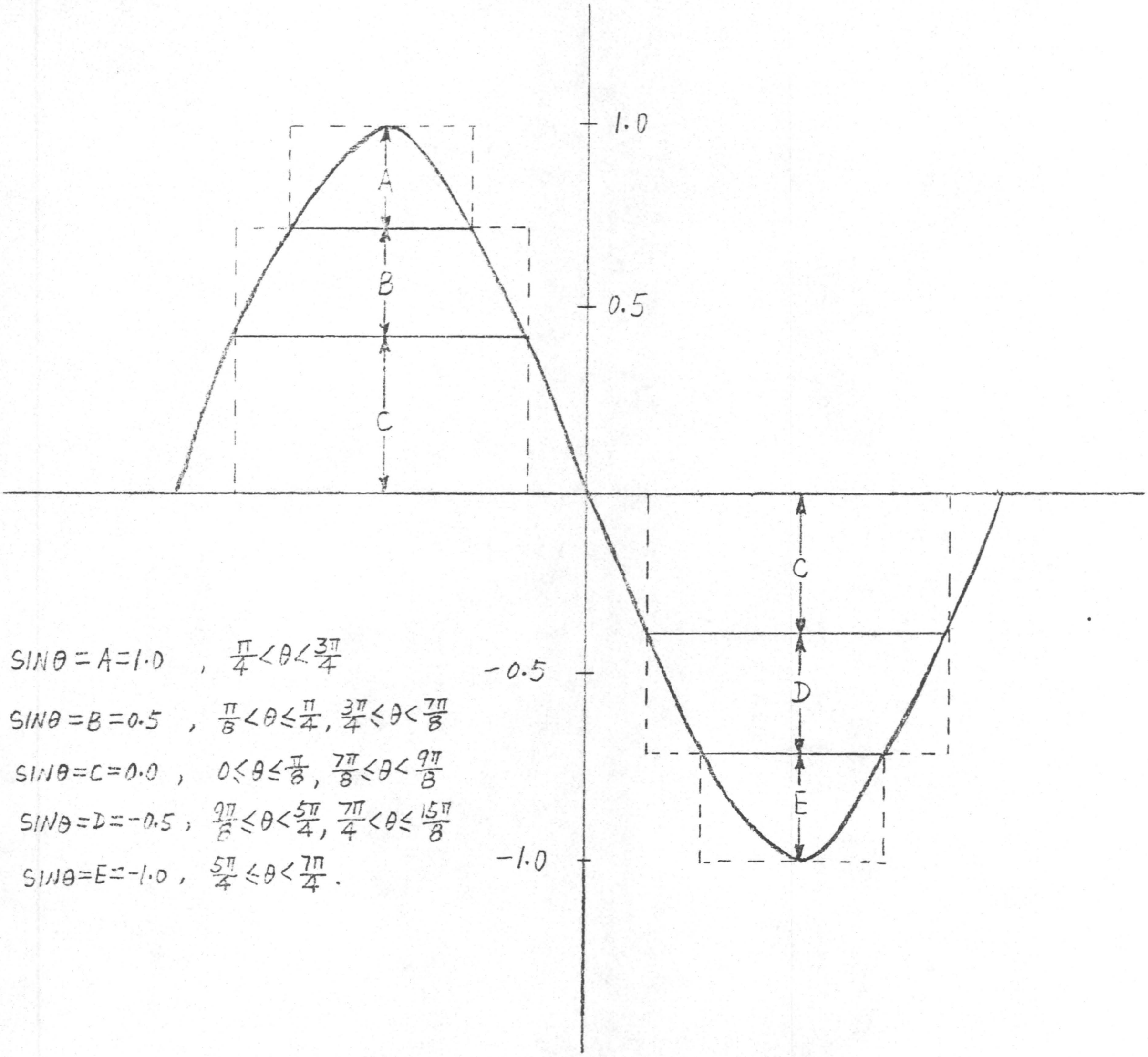


FIG. 3-9 WAVEFORMS IN THE REPLICA CORRELATION PROCESSOR



$\sin\theta = A = 1.0 \quad , \quad \frac{\pi}{4} < \theta < \frac{3\pi}{4}$

$\sin\theta = B = 0.5 \quad , \quad \frac{\pi}{8} < \theta \leq \frac{\pi}{4}, \frac{3\pi}{4} \leq \theta < \frac{7\pi}{8}$

$\sin\theta = C = 0.0 \quad , \quad 0 \leq \theta \leq \frac{\pi}{8}, \frac{7\pi}{8} \leq \theta < \frac{9\pi}{8}$

$\sin\theta = D = -0.5 \; ; \; \frac{9\pi}{8} \leq \theta < \frac{5\pi}{4}, \frac{7\pi}{4} < \theta \leq \frac{15\pi}{8}$

$\sin\theta = E = -1.0 \; , \; \frac{5\pi}{4} \leq \theta < \frac{7\pi}{4}.$

FIG. 3-10 FIVE LEVEL QUANTIZATION OF A SINE WAVE

hardware than I.F. operations.

Normally real-time signal processing is not done at I.F. because of the prohibitively large amount of real time computations required to ensure a good result. However, for low bandwidth signals, such as acoustic signals, I.F. frequency is low. For a 40.32 ms CW pulse sampled at every 315.0 μs, there are 128 samples in the replica of each CW pulse. Therefore, each complex correlation output requires 256 real additions and 256 real multiplications. Because multiplication is a time consuming operation, it is highly desirable to replace multiplication with some other simpler arithmetic operations. To accomplish this, the CW pulse replica of a correlator may be approximated by a 5 level quantized sine wave as shown in Fig. 3.10 [2], any values inside region A, B, C, D, and E are replaced by a vnalue of 1, 1/2, 0, -1/2, and -1, respectively. With these approximations, the use of a complicated hardware multiplier or time consuming software routine is eliminated.

3.3.4 COMPARISON OF THREE METHODS

Three digital signal processing techniques, FFT, digital filtering, and replica correlation have been examined, mainly, for the computational requirement for the realization of each technique. As mentioned previuosly, although baseband processing requires less computational efforts, it increases the complexity of the system, since mixing to baseband and low pass filtering is required. Since the frequency of the signals to be processed are low, the extra computational time demanded by the I.F. processing is far less of a complication than the system hardware requirements for a baseband processor.

An I.F. correlator is an optimum filter for a CW pulse signal in the presence of noise, whose statistical characteristics vary with time and locations, and also has the advantage over a FFT processor in that; the multiplication required during a correlation process can be completely replaced with two simpler arithmetic operations, add and shift and add, by the use of 5 level quantized sine wave. Moreover, the replica signals of a correlator can be easily altered to achieve a better Doppler frequency shift resolution without increasing the sampling rate at the input. Thus, an I.F. replica correlator appears to be best suited for detecting the Doppler shift that may be concealed in an sonar echo returned by a target.

3.4 THE DESIGNS OF REPLICA CORRELATOR

Several types of practical correlators have been constructed, depending upon the manner in which the references and input signal storage is accomplished. All are time compression systems. All can be classed as one of three general types, the sum correlator, the difference correlator, and dc correlator. All of the correlators have in common, the sum of all the products of two band-limited signals, the reference sequence and the input

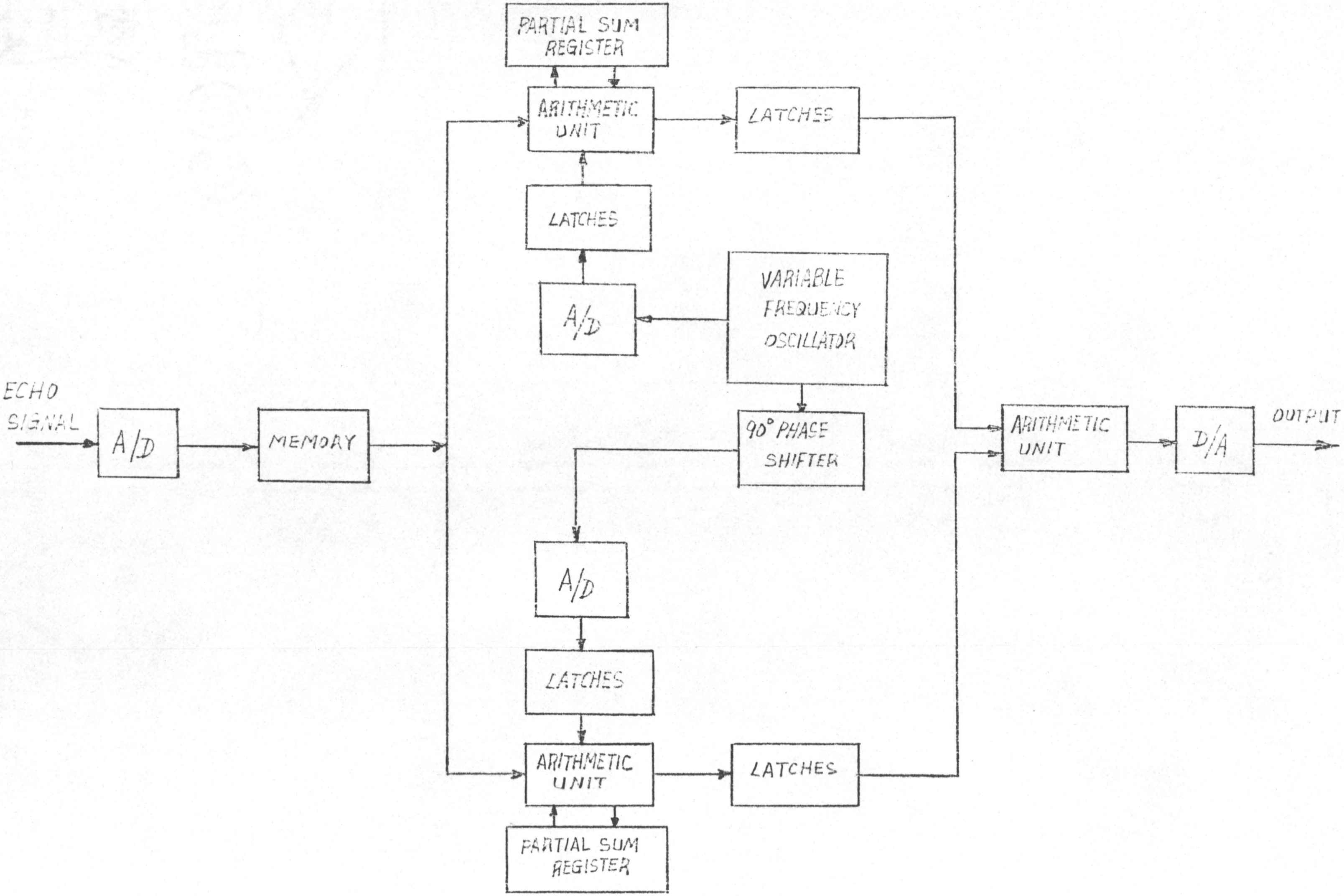


FIG. 3-11 BLOCK DIAGRAM OF A DIGITAL REPLICA CORRELATOR EMPLOYING PARALLEL PROCESSING PATH FOR COMPLEX SIGNAL PROCESSING.

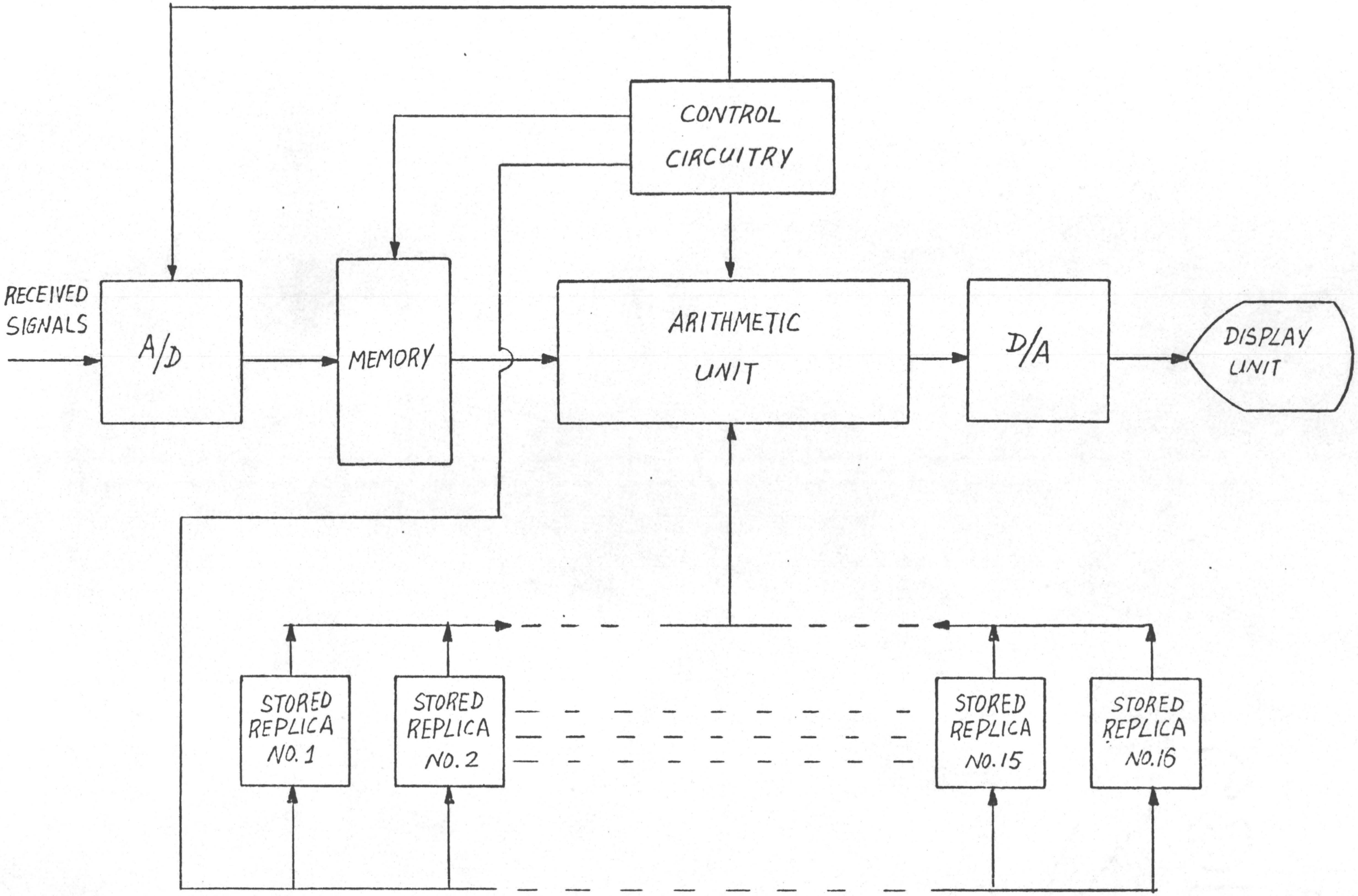


FIG. 3-12 BLOCK DIAGRAM FOR THE PROPOSED DIGITAL REPLICA CORRELATOR

sequence. The sum correlator requires that the reference signal be a spectrum inverted version of the expected signal. The difference correlator requires the reference signal to be identical to the expected signal. The dc correlator is basically a difference correlator at baseband.

Here the designs of two different correlators will be considered. Figure 3.11 shows one version of the correlator. The incoming analog signal is sampled and read into the memory. The memory is sufficiently large to hold simultaneously all of the samples required for a correlation. In operation, all the input samples equivalent to the time duration of the expected signal are read out and correlated with reference signal in a duration equal to the time between samples of the input signal. Such a technique constitutes a time-compression scheme. In order to update the contents of the memory, the oldest sample in the memory is replaced by a new sample. The reference signal is obtained from a variable frequency oscillator whose output covers a range of frequency from f_o-f_d to f_o+f_d, where f_o is the transmitted frequency of the CW pulse and f_d is the expected maximum Doppler frequency shift at input. Doppler detection can be made by "scanning" the input signal with the output of the oscillator. The Doppler shift of the input signal is equal to the oscillator frequency output, which gives the highest envelope output, minus the transmitted frequency f_o. The quadrature component of the reference signal, obtained by passing the oscillator output through a 90° phase shifter, and the inphase component are sampled continuously for correlation with the input samples. The inphase and quadrature processing of the input signal are performed simultaneously on two identical processing paths as shown in Fig. 3.10 and the correlation results are used by a third arithmetic unit for envelope calculation.

A second version of the design, shown in Fig. 3.12, differs from the first in that the variable oscillator for the reference signal in Fig. 3.11 has been replaced by a bank of read-only-memory (ROM) units which stores the quadrature and inphase components of the Doppler shifted versions of transmitted signal at discrete frequencies. This enables the complex signal processing to be performed by one processing unit, thus reduces the system complexity. The principle of operation of this design is identical to that of the first design; the input signal is scanned by the stored replica signals to detect any Doppler shifts.

It is clear that the first design is a much faster processor and it is capable of giving an accurate indication of the Doppler shift of an input signal. However, this advantage is achieved at the expense of increasing systems hardware complexity. Depending upon the end use of the processor and the frequency of the input signal, the second design may be a more attractive option. Since the contents of the ROM units can be readily changed, fine Doppler shift resolution can be obtained by employing two banks of ROM units, the first bank of ROM units provides a course value of the Doppler shift of an input signal and the second bank of ROM units can supply a more accurate indication of the Doppler shift. Note, for human operator, there exists a lower limit between the distance of two different frequencies that can be individually identified. Beyond this limit, the resolution of the correlator can not be improved without resorting to some kind of automatic decision device. Finally, although the second correlator is a slower processor by design, it is less complicated and adequate computaional speed can be provided by the use of a fast arithmetic unit. Thus, due to simplicity in design, adequate Doppler shift resolution, and moderate computational speed, the systems in Fig. 3.12 has been implemented in this work.

CHAPTER 4

HARDWARE IMPLEMENTATION OF A DIGITAL REPLICA CORRELATOR

4.1 GENERAL

In this chapter, the hardware implementation for a real-time digital replica correlator based on Fig. 3.12 is described. The correlator is preferred over the FFT processor due to its hardware simplicity and adquate Doppler frequency resolution over the expected Doppler shift.

Nominally, a target return from a 40.32 ms pulsed duration CW signal, sampled every 315.0 μs, contains 128 samples. Since sampling of the analog signal at the processor input is continuous and the correlator processor can process only 128 samples at a time, the target return may be split into two successive frames. When this occurs, a degradation of the output signal magnitude may result. This is most serious in the case where the returned target samples are split equally between two successive frames, since a 3 dB degradation of the output magnitude occurs. The correlator outputs for the sample frames containing 0, 1/4, 1/2, 3/4, and all the target samples are shown in Fig. 4.1, the lowest and highest outputs in the figure corresponds to no target sample and all the target samples in the frame processed, respectively. If a complete correlation of the replica and returned waveform is

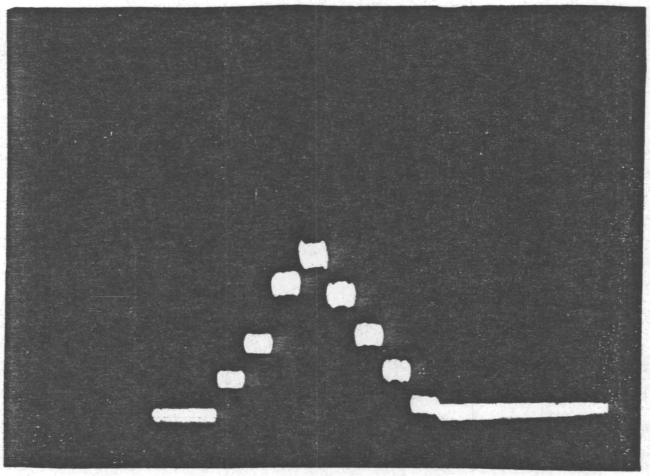


Fig. 4.1 Correlator outputs with 0, 1/4,1/2,3/4, and all the target samples in the sample frame. The lowest output corresponds to 0 target sample in the farme and the highest output corresponds to all the target samples in the frame.

performed once every 10.08 ms, i.e. 4 correlations in 40.32 ms, with each frame displaced by 1/4 of the target samples, then one of the frames would contain at least 112 of the 128 target samples, and degradation of the correlator output would be at most 112/128 or 0.58 dB.

In the implementation of correlator, the analog signal at the input of the processor is sampled and converted into digital form and stored in RAM units at a rate of 3174 samples per second. Sixteen five-level quantized replicas are stored in 16 different ROM units. With the help of the timing signals from the control circuitry, the input sample and replica sample are read out from the RAM and ROM, respectively, to the arithmetic unit for processing. Each sample frame containing 128 input samples is serially correlated with 16 replicas before the frame is updated. Finally, the digital outputs from 16 replica channels are displayed on a storage oscilloscope through a D/A converter.

4.2 BIT SLICE MICROPROCESSOR

The most important and challenging portion in this implementation of the digital correlator is to make the entire system as simple and yet as flexible as possible. If arithmetic units like SN 74181 multiplier units are used, then a complicated control circuitry is required to monitor the operations of these units. The new fast multiplier chips capable of performing 8 bit by 8 bit multiplications in 100 ns or less tend to be expensive and require fast RAM units; consequently, they are not well suited to this application. Conventional MOS technology microprocessors offer simplicity and flexibility in system design, however, they do not have adequate speed to perform all the processing operations required in a real time environment. In recent years, a new type of microprocessor employing Schottky bipolar technology has been introduced to the market. Manufactured with Schottky bipolar LSI circuit technology, these bit slice microprocessors differ from the MOS microprocessors not only in the speed performance (about 10 times faster than MOS) but also in the architecture of the microprocessors.

Many different types of bit slice microprocessors are now available on the market. They consist mainly of software programmable central processing elements (CPE) representing 1, 2, or 4 bits of the data processing section of a computer and a microprogram address control unit (MCU) which performs the function of a microprogram address counter. Several CPEs may be arrayed in parallel to form a processor of any desired word length.

Before the hardware realization of the correlator is explained, perhaps it is appropriate to briefly describe the organization and function of the CPEs and MCU to provide a better understanding of the hardware and software designs. The bit slice microprocessor described herein is the Intel 3000 series bit slice microprocessor.

4.2.1 CENTRAL PROCESSING ELEMENT (CPE)

Each CPE [13] in Fig. 4.2 carries two bits of 3 input buses and two output buses. Typically, the I- and M-input buses carry data from external memory or input/output devices, and the K-input bus is used either for microprogram mask or to provide a constant value to the arithmetic unit of the CPE. A-bus outputs are connected to the memory address output buffer, and D-bus outputs are connected to the accumulator output buffer. When CPEs are wired together, all the data paths registers and buses expand accordingly. A sixth bus, the microinstruction bus, carries the microinstructions controlling the internal operation of the CPE by selecting the operands and the operations to be performed. The CPUs of the MOS microprocessors are hardwired to perform a fixed set of system functions among which many of them are rarely used in dedicated systems. Conversely, the CPEs of a bit slice microprocessor perform over 40 Boolean and binary functions including 2's complement arithmetic and logical AND, OR, NOT, and exclusive-NOR. Therefore, by microprogramming, the system functions can be tailored to the system needs. A list of the representative microfunctions performed by the CPE arithmetic logic sections is provided in Appendix 2.

The versatility of the CPE is increased further by the incorporation of three other novel techniques. The first of these is the use of carry lines and logic during non-arithmetic operations for bit testing and zero detection. During these two operations the carry circuits perform a word-wide logical OR, i.e., ORing adjacent bits, of a selected result from the arithmetic section. The result of the OR is passed along the carry lines to be ORed with the result of an identical operation taking place simultaneously in the adjacent higher order CPE. Clearly, if at least one bit is in the logical 1 state, this will result in a positive carry output from the highest order CPE. This output can, then, be used by the MCU to determine which microprogram sequence to follow. With the ability to mask any desired bit or set of bits, via the K-bus inputs included in the carry OR, a powerful bit-testing and zero-detection facility is realized.

The second innovation is the use of tri-state outputs on the shift right output (RO) and carry output (CO) lines. Since RO and CO outputs will not be utilized at same time, this tri-state capability enable the RO and CO lines to be tied together and sent as a single input to the MCU for testing and branching. During a right shift operation, the CO output is placed in the high impedance state, and the shifted data is active on the RO line. For all other CPE operations, the RO line is placed in the high impedance state, and carry data is active on the CO line. Left shift operation utilizes the carry lines, instead of shift lines, to propagate data.

The third innovation is the capability of the CPE to perform nondestructive testing on the register data through the

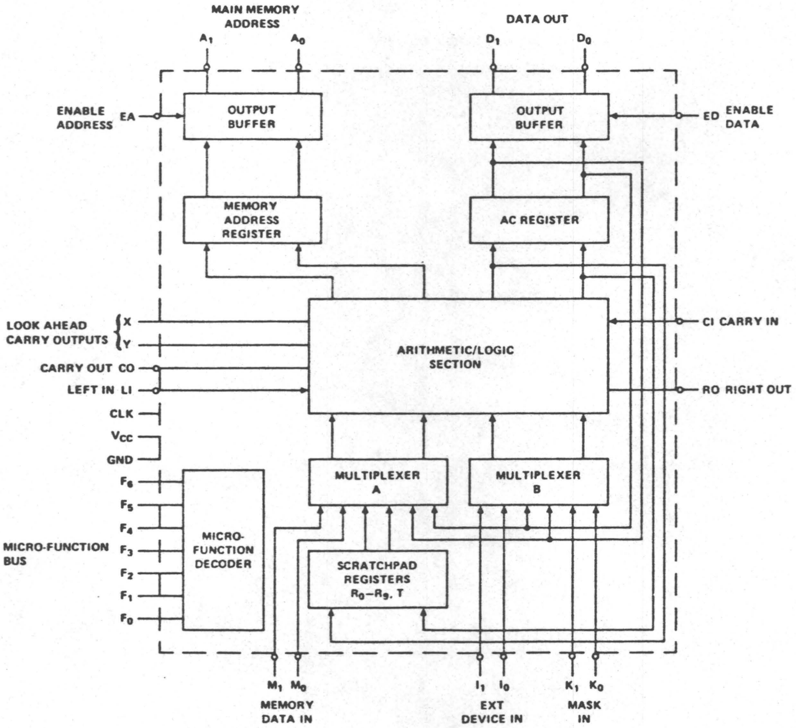


Fig. 4.2 Functional block diagram of an INTEL 3002 CPE.

use of conditional clocking which momentarily freezes the CPE clock and permits the CPE micro-function to be performed, but stops the results from being clocked into the specified registers. Since the arithmetic section is combinatorial, rather than sequential, the test result is available to be used as a jump instruction by the MCU. To realize conditional clocking, one extra bit is required in the microinstruction to selectively control the gating of clock pulse to the CPE array.

4.2.2 MICROPROGRAM CONTROL UNIT (MCU)

A system function in the main program memory of a bit slice microprocessor is realized by supplying the operation code (OP code) of the function to the primary and secondary instruction buses of the MCU [13], as shown in Fig. 4.3. Upon receiving the OP code, the MCU decodes it to determine the first location of a sequence of microinstructions which direct the arithmetic section of the CPE to perform the system function. In order to have an efficient use the micro-codes and thus the microprogram memory, an alternative addressing scheme other than the use of a conventinal sequential program counter has to be adopted.

One way to overcome this problem is to incorporate an address control field that contains the necessary information to determine the address of the next microinstruction in each microinstruction. The next address logic and address control functions of the MCU are based on a unique scheme of memory addressing.

Unlike ordinary program counter, which has linearly sequenced address, the microprogram addresses are organized as a two dimensional array or matrix. Each microinstruction is pinpointed by its row and column addresses in the matrix. For a given location in the matrix, it is possible to jump unconditionally to any other location in that row or column or conditionally to a fixed subset of the microprogram addresses. Incorporating a pump operation in every address control field of microinstructions improves performance by allowing processing functions to be executed in parallel with program branch. Also reductions in micro-code are achieved because common microprogram sequences can be shared without the time space penalty.

The next microinstruction addresses are completely determined on the basis of: the MCU's current microprogram address; the address control function; the data on the primary and secondary instruction buses; and also on the flag input value from the CPE. An Intel 3001 MCU is capable of addressing 512 microinstruction locations through a 9-bit microprogram address bus. The 32 rows by 16 columns matrix is addressed by the 5 most significant bits and 4 lest significant bit, respectively.

4.3 IMPLEMENTATION

The schematic diagram for the implementation of the system in Fig. 3.12, utilizing an Intel 3000 series bit slice microprocessor, is depicted in Fig. 4.4 [21-22]. The system can be

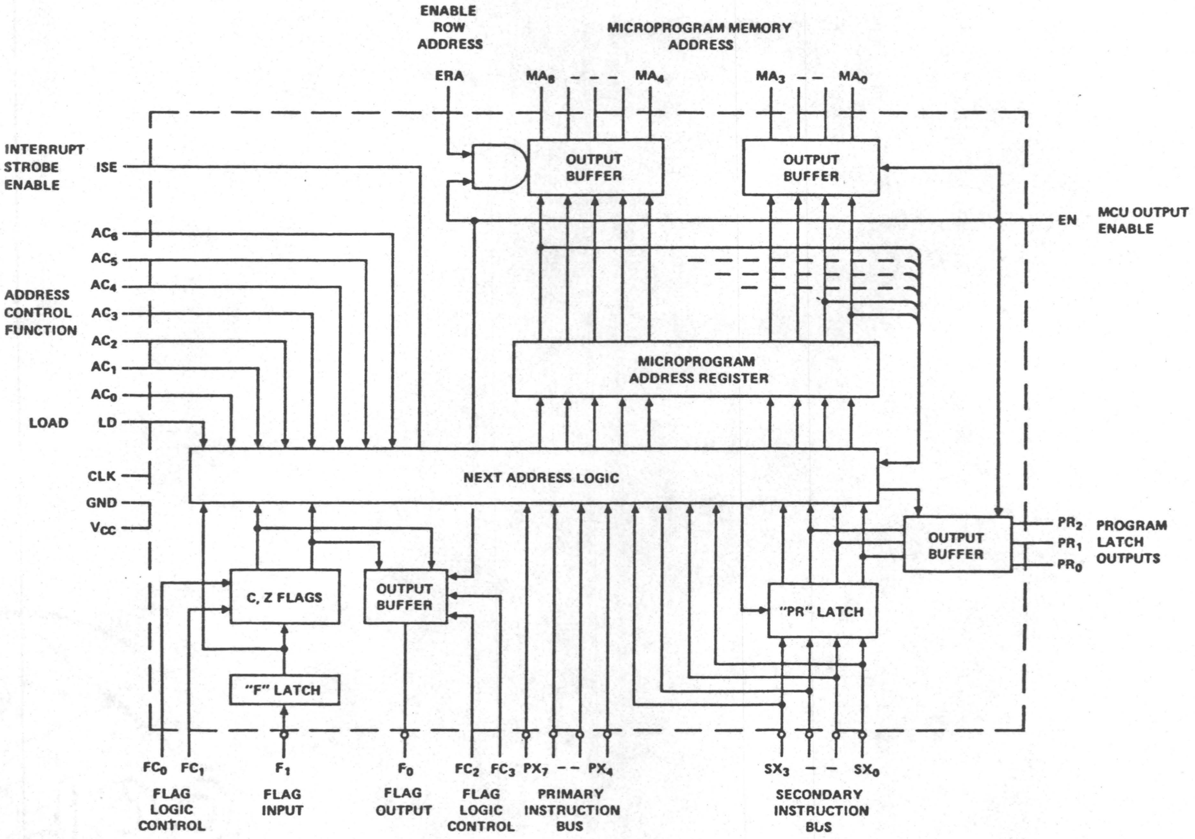


Fig. 4.3 Functional block diagram of an INTEL 3001 MCU

divided into three sections. These are: the input and output circuitry, the processor, and the system control circuitry.

4.3.1 INPUT AND OUTPUT CIRCUITRY

The analog input, containing target echoes and noise or just noise, is sampled by a Sample and Hold (S & H) device at the rate of 3174 samples per second. Having converted from analog form into digital form, the samples are sequentially stored in RAMs. As described later, the samples are retrieved, in manner of first in first out, by the processor to be correlated with the replica samples.

Every 315.0 μs a Sample and Hold signal, issued by the system control circuitry, causes the Sample and Hold device to change from tracking mode to holding mode. Triggered by the capacitively delayed S & H signal in Fig. 4.5(a), the A/D converter changes the analog sample into 8-bit digital data whose logic levels are subsequently inverted by logic level inverters (74504). This sample remains at the input of tri-state D type flip flops (74S373). Approximately 310 μs later, as indicated in Fig. 4.5(b), with the address supplied by the processor, the digital data is transfered into RAMs through the D type flip flops by the Read Command signal from system control circuitry and TS and R/W signals from microinstructions. This particular arrangement of delaying the transfering of digital data into RAMs has the advantage of facilitating the use of a slow A/D converter.

When a stored digital input test signal is used to replace the analog input signal, the RAMs are disabled by switching the toggle switch in Fig. 4.5(a) from logic level LOW position to HIGH. When this is accomplished, the RAMs are de-selected, and the ROMs where the data are stored are enabled. The use of stored digital data here provides a repetitive correlation outputs that aids in debugging the system.

At the end of each correlation, the envelope value is calculated and output to the output latches (74S174) through inverters by an output command signal ED from the memory control and I/O field of microinstructions. Since there exists a delay between the arrival of the ED signal and a set of steady data on the D-bus of the central processing array (CPA), the output latches are clocked by the output of a monostable multivibrator output initialized by ED. The circuit diagram and timing diagram are depicted in Fig. 4.6(a) and 4.6(b) respectively. The latched output is converted back into analog output and displayed on a storage oscilloscope. 315.0 μs is needed for processing each of the inphase and quadrature components for a replica channel. Since there are 16 Dopper channels to be correlated with each set of input data, 10.08 ms is required to calculate all the correlation results.

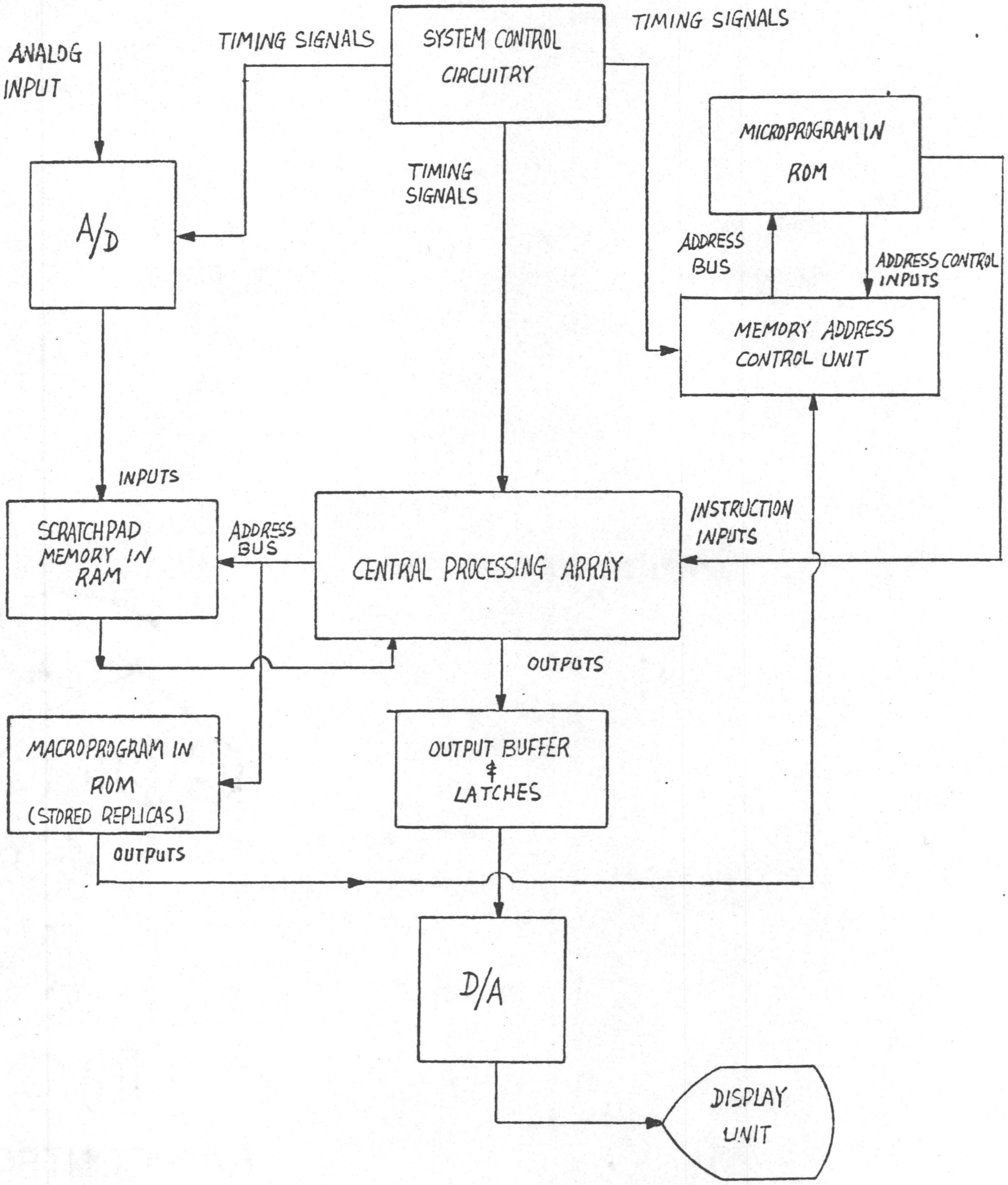


FIG. 4-4 THE BLOCK DIAGRAM OF A DIGITAL REPLICA CORRELATOR USING BIT SLICE MICROPROCESSOR

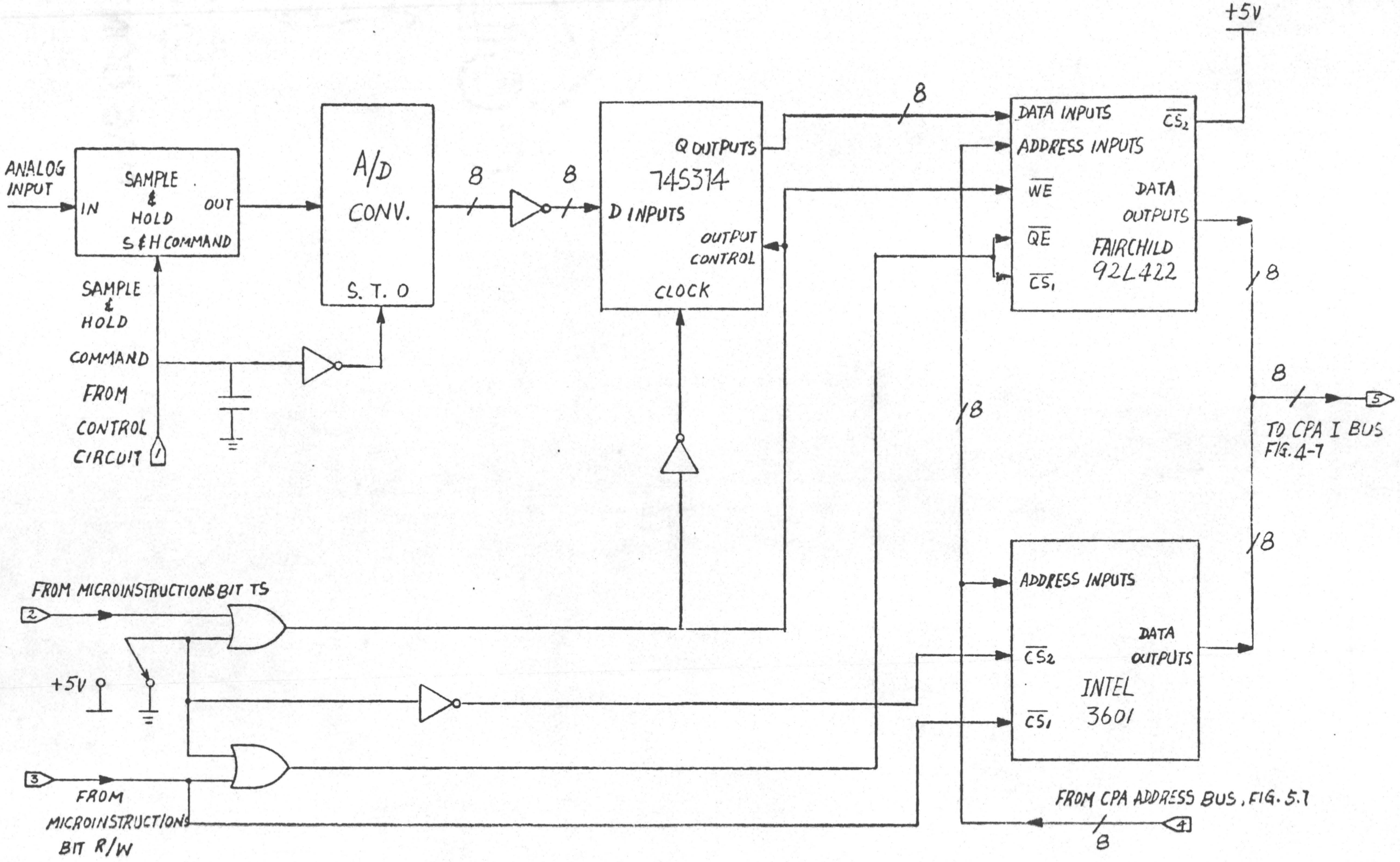


FIG. 4-5(a) SYSTEM INPUT CIRCUIT DIAGRAM

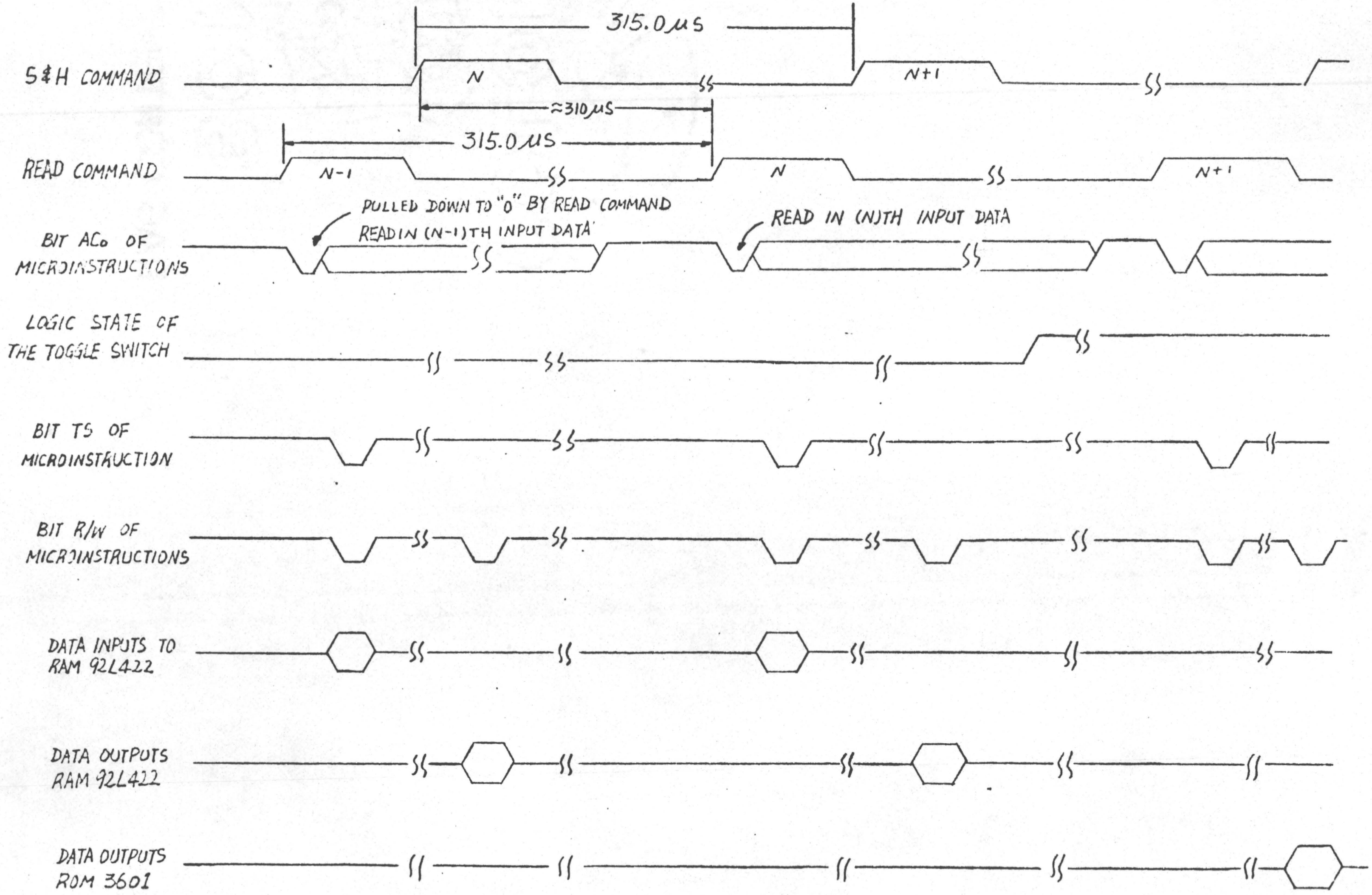

315.0 μS
S&H COMMAND
N
N+1
≈310 μS
315.0 μS
READ COMMAND
N-1
N
N+1
PULLED DOWN TO "0" BY READ COMMAND
READIN (N-1)TH INPUT DATA
READ IN (N)TH INPUT DATA
BIT AC₀ OF MICROINSTRUCTIONS
LOGIC STATE OF THE TOGGLE SWITCH
BIT TS OF MICROINSTRUCTION
BIT R/W OF MICROINSTRUCTIONS
DATA INPUTS TO RAM 92L422
DATA OUTPUTS RAM 92L422
DATA OUTPUTS ROM 3601

FIG. 4-5(b) SYSTEM INPUT CIRCUIT TIMING DIAGRAM

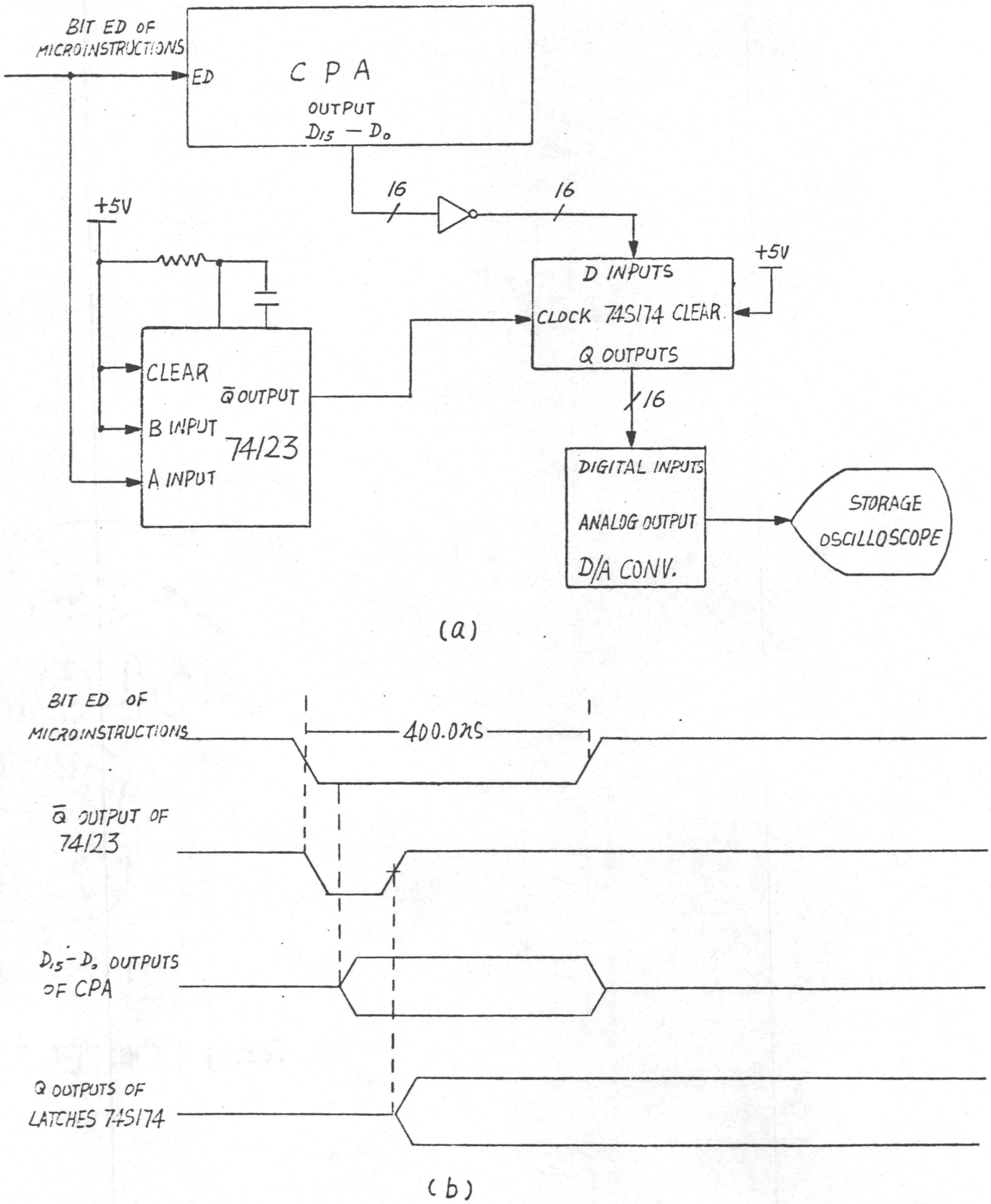


FIG. 4-6. (a) OUTPUT CIRCUIT DIAGRAM. (b) TIMING DIAGRAM OF (a).

4.3.2 THE HARDWARE OF THE PROCESSOR

The processor considered here consists mainly of a functional section and a memory section. The functional section contains an Intel 3000 series bit slice microprocessor which includes a 16-bit CPA assembled from eight chips of 2-bits per chip CPEs (Intel 3002), a look ahead carry generator (Intel 3003) for improving the performance of the CPA and a MCU (intel 3001) for generating the next microprogram address. The memory section is made up of two ROM banks, one for the microprogram and the other for the stored replica signals. The former is addressed by the MCU, and the latter is addressed through CPA's A-bus. Fig. 4.7 shows the connection diagram for the major components of the processor.

As the correlation process is initialized, the CPA under microprogram control sends out input addresses for transfering the input data into the RAMs. Having acquired 128 input samples, the CPA loads the memory address register (MAR) with the first input address of the RAMs, and reads the data in that location into the accumulator. The corresponding address of the first stored replica channel is also read out from the ROMs and presented at the input of the primary instruction bus on the MCU during the subsequent CPA cycle. The replica data stored in the ROMs are actually a sequence of 5 different code words generated from the 16 Doppler shifted version of the transmitted signal by means of 5-level quantization. The Fortran program used to generate and assemble these code words is listed in Appendix 2. Each code word serves as an index for the MCU's next address logic. Upon receiving the code word, the MCU causes the microprogram to jump to one of the five designated addresses where one of the five operations is performed on the data in the accumulator. These are: add content of the accumulator to the partial sum; shift the content of the accumulator to the right and add to the patial sum; discard the content of the accumulator and do nothing to the partial sum; shift the content of the accumulator to the right and subtract from the partial sum; and subtract the content of accumulator from the partial sum. The above described operation sequence is repeated for 128 input samples.

At the end of 128th execution of the fore-mentioned sequence, the microprogram checks whether the processor has performed the correlation operations for both the inphase and quadrature components of the replica channel being processed. If the test result is negative, the microprogram stores the correlation result obtained from the inphase component in one of the scratchpad registers and, then, goes into a loop which idles the processor. After 315.0 μs, the system control circuit issues a Read command signal which forces the least significant bit of the address control function to logic level LOW (see Fig. 4.5(b)) pulling the microprogram out of the idling loop; subsequently, the CPA sends out an input address to facilitate the transfer of the input sample into one of the RAMs. Having done this, the microprogram proceeds to perform the correlation between the quadrature component and the same 128 input samples using the same operation sequences described in the last paragraph.

If the test result is positive, the microprogram proceeds to calculate the envelope value of the correlation results using an approximation to the quantity $(x^2+y^2)^{1/2}$ (see Appendix 1) where x is the correlation result from the inphase component and y is the correlation result from the quadrature component. After outputing the envelope value to the output latches, the processor goes into the idle state and is reactivated under the same procedures and conditions as described before. The processor either starts a new correlation process with the same frame of 128 input samples and a different replica channel, or, if the last correlation performed was the 16th replica channel, the samples contained in the frame are updated by replacing the 32 oldest samples with 32 most recent samples. This is equivalent to incrementing the frame starting address by 32, e.g., from 0 to 31. This updated frame containing 96 old samples and 32 new samples is subsequently employed to calculate the successive correlation output for each replica channel. A microprogram flow chart illustrating all the sequences described above is shown in Fig. 4.8, and the microprogram listing with its address map are shown in Appendix 2.

Since the emulator for the Intel 3000 series bit slice microprocessor was not available at the time, the microprogram was

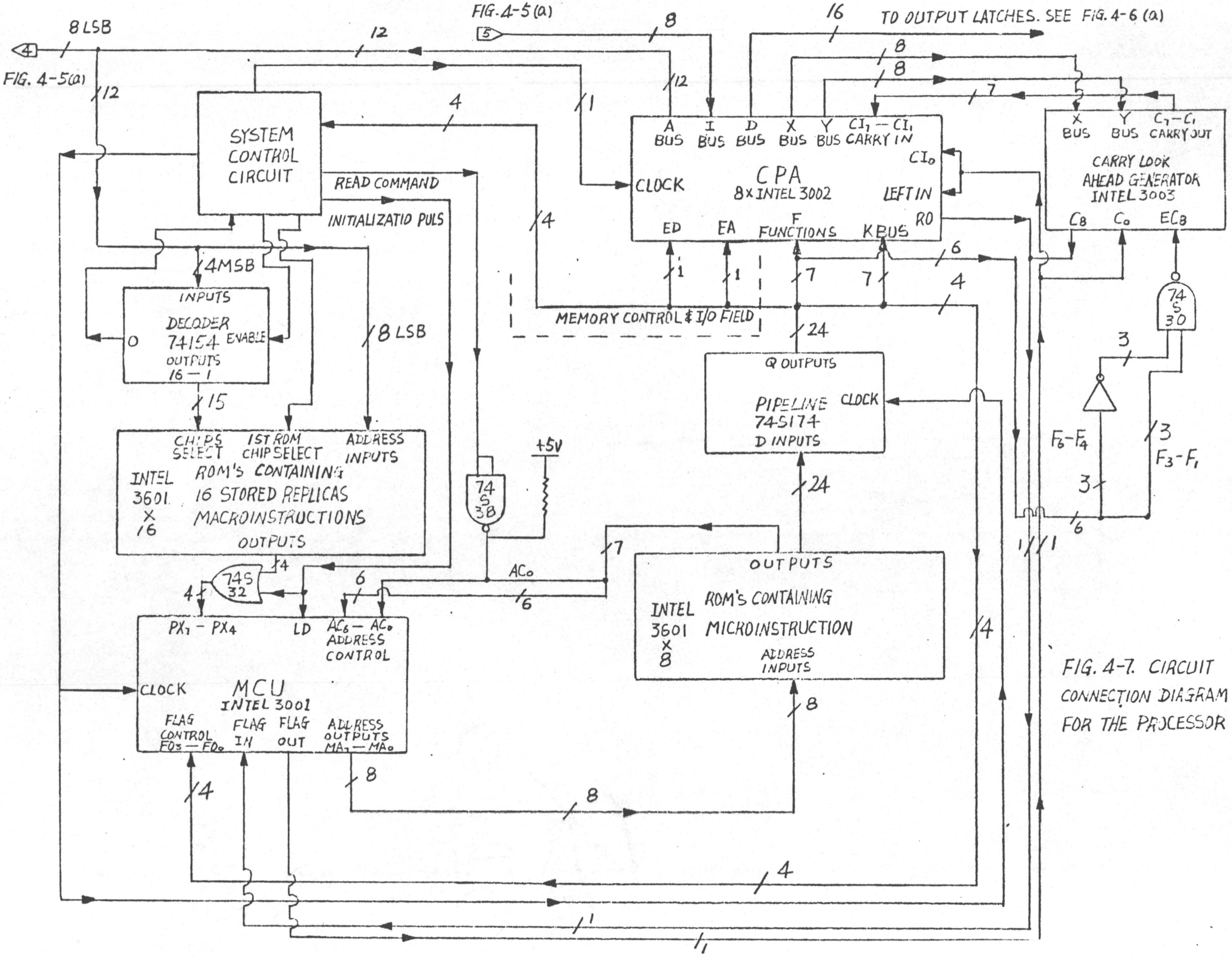


FIG. 4-7. CIRCUIT CONNECTION DIAGRAM FOR THE PROCESSOR

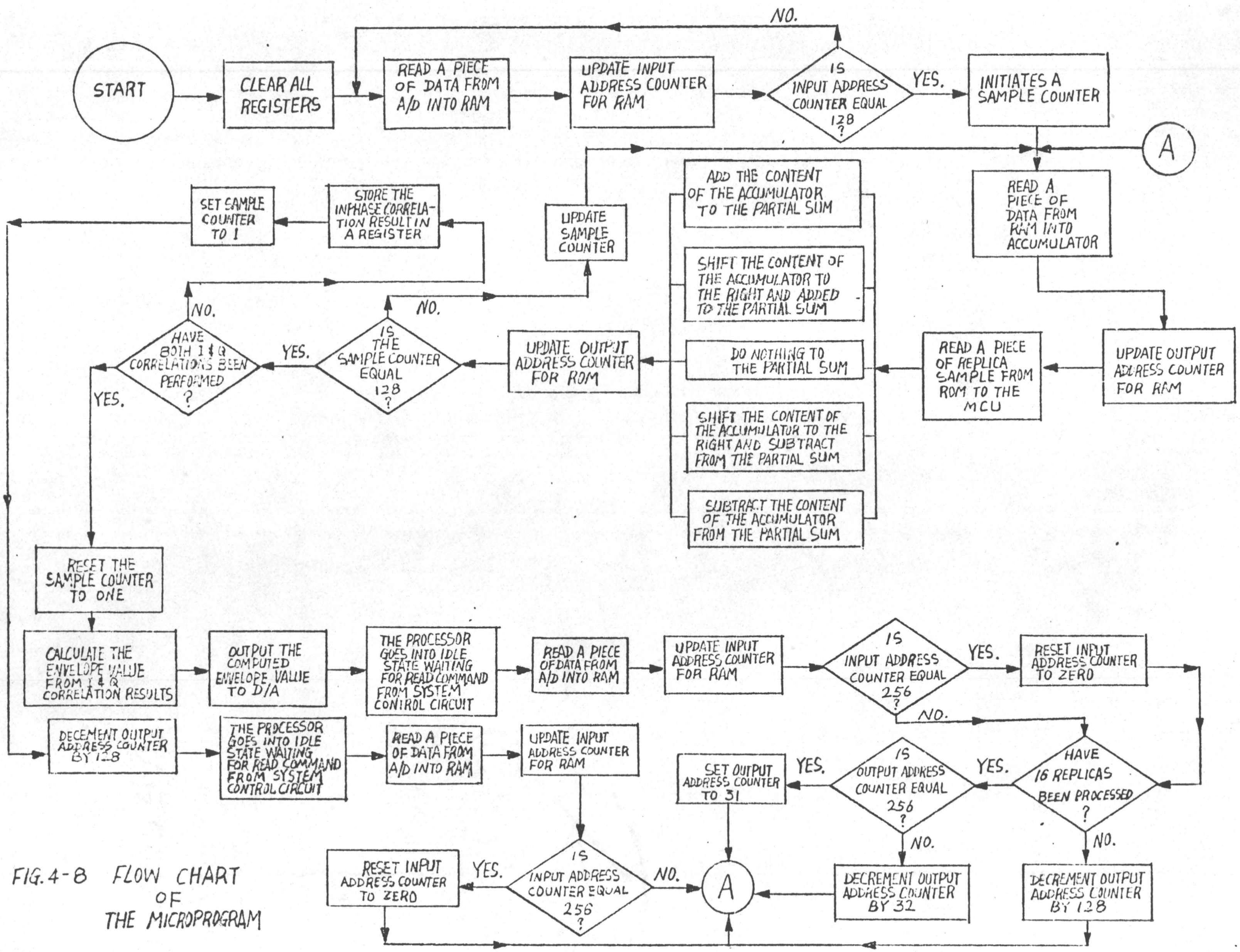

START
CLEAR ALL REGISTERS
READ A PIECE OF DATA FROM A/D INTO RAM
UPDATE INPUT ADDRESS COUNTER FOR RAM
IS INPUT ADDRESS COUNTER EQUAL 128 ?
NO.
YES.
INITIATES A SAMPLE COUNTER
A
READ A PIECE OF DATA FROM RAM INTO ACCUMULATOR
UPDATE OUTPUT ADDRESS COUNTER FOR RAM
READ A PIECE OF REPLICA SAMPLE FROM ROM TO THE MCU
ADD THE CONTENT OF THE ACCUMULATOR TO THE PARTIAL SUM
SHIFT THE CONTENT OF THE ACCUMULATOR TO THE RIGHT AND ADDED TO THE PARTIAL SUM
DO NOTHING TO THE PARTIAL SUM
SHIFT THE CONTENT OF THE ACCUMULATOR TO THE RIGHT AND SUBTRACT FROM THE PARTIAL SUM
SUBTRACT THE CONTENT OF THE ACCUMULATOR FROM THE PARTIAL SUM
UPDATE OUTPUT ADDRESS COUNTER FOR ROM
IS THE SAMPLE COUNTER EQUAL 128 ?
NO.
UPDATE SAMPLE COUNTER
YES.
HAVE BOTH I & Q CORRELATIONS BEEN PERFORMED ?
NO.
STORE THE INPHASE CORRELATION RESULT IN A REGISTER
SET SAMPLE COUNTER TO 1
YES.
RESET THE SAMPLE COUNTER TO ONE
CALCULATE THE ENVELOPE VALUE FROM I & Q CORRELATION RESULTS
OUTPUT THE COMPUTED ENVELOPE VALUE TO D/A
THE PROCESSOR GOES INTO IDLE STATE WAITING FOR READ COMMAND FROM SYSTEM CONTROL CIRCUIT
READ A PIECE OF DATA FROM A/D INTO RAM
UPDATE INPUT ADDRESS COUNTER FOR RAM
IS INPUT ADDRESS COUNTER EQUAL 256 ?
YES.
RESET INPUT ADDRESS COUNTER TO ZERO
NO.
HAVE 16 REPLICAS BEEN PROCESSED ?
YES.
IS OUTPUT ADDRESS COUNTER EQUAL 256 ?
YES.
SET OUTPUT ADDRESS COUNTER TO 31
NO.
DECREMENT OUTPUT ADDRESS COUNTER BY 32
NO.
DECREMENT OUTPUT ADDRESS COUNTER BY 128
DECEMENT OUTPUT ADDRESS COUNTER BY 128
THE PROCESSOR GOES INTO IDLE STATE WAITING FOR READ COMMAND FROM SYSTEM CONTROL CIRCUIT
READ A PIECE OF DATA FROM A/D INTO RAM
UPDATE INPUT ADDRESS COUNTER FOR RAM
IS INPUT ADDRESS COUNTER EQUAL 256 ?
YES.
RESET INPUT ADDRESS COUNTER TO ZERO
NO.
A

FIG. 4-8 FLOW CHART OF THE MICROPROGRAM

simulated on a PDP/8A minicomputer before being implemented in micro-code. The simulation program is also listed in Appendix 2.

4.3.3 THE MICRO-CODES

In order to simplify the system control circuitry and thus make the processor more flexible, most system control functions have been incorporated into the microinstructions. A microinstruction for this system is a 31 bit long code word consisting of 5 functional fields; the micro-funtion field which controls the arithmetic section of the CPA; the flag control function field which sets or resets the flag flip flops inside the MCU; the momory control and I/O fields which controls the read and write function of the RAMs and ROMs, conditional clocking for performing non-destructive tests on the data in the registers, and the elongation of the system clock for CPA's memory cycles; the K-bus field which provides a flexible masking and bit testing capabilities for the CPA and, finally, the address control field, together with the flag control function field and next address logic in the MCU which determines the next address of the microprogram. Each microinstruction consists of groupings of 7, 4, 6, 7, and 7 bits of the fields described above, respectively. A copy of the micro-codes for the microprogram is listed in Appendix 2.

Amongst the five functional fields, the memory control and I/O field is the only one having a set of code words that are specified by the designer, and each of the rest of 4 functional fields has a set of pre-designated code words controlling the specific functional blocks of the processor. Table 4.1 contains a list of the functions for each bit in the memory control and I/O field and the collective action performed by each word of the field is explained in Table 4.2.

The 31-bit microinstructions are stored in 8 ROM units organized in a 256 word by 4 bit format to give a 32-bit word-length with one bit unused. When microinstructions are read out of the ROMs, all the fields but the address control field pass through a pipeline structure made of D type flip flops (74S174). This pipeline structure enables the MCU to obtain the next micropgoram address while CPA is carrying out the its operation. The consequence of this is higher system speed performance.

4.3.4 SYSTEM CONTROL CIRCUITRY

Basically, the system control circuitry provides the following functions:

(a) initializes and synchronizes the systems.

(b) provides clocks for the MCU and the CPA.

(c) supplies timing signals (e.g. Sample and Hold, Read Command).

(d) demultiplexes the data from the replica channels.

(A) An HP86608 frequency synthesizer, capable of providing a precise frequency output, is used as the master oscillator. Operating at 5.0 MHz and amplitude 2 volts, the output of the

MNEMONIC	LOGIC STATES	FUNCTIONS
EA	1	disables the memory address output buffer of the CPA.
	0	enables the memory address output buffer of the CPA.
ED	1	disables the output buffer of the CPA.
	0	enables the output buffer of the CPA and clocks the output latches.
R/W	1	memory are not accessed.
	0	memory are accessed; it is used as read/write control for the RAM's and chip selects for the ROM's.
INH	1	conditional clocking is required.
	0	conditional clocking is not required.
RF	1	memory reference instructions; elongated clock cycles are called for.
	0	non-memory reference instructions.
TS	1	disables the output control of the input latches at system input.
	0	enables the output contorl of the input latches at system input.

Table 4.1 Functions performed by each bit in the memory contorl and I/O field of microinstructions.

MNEMONIC	MICRO CODES						FUNCTIONS
	TS	RF	INH	R/W	ED	EA	
ADS	0	1	0	0	1	0	enables the output control of the input latches, transfer the address from the memory address buffer to the memory being accessed, and provides an elongated clock cycle.
RAM	1	1	0	0	1	0	send the address from the memory address buffer to the input RAM's, provides an elongated clock cycle, and read the data into the CPA.
INH	1	0	1	1	1	1	conditional clocking, the time elapsed between two clock pulses are forced to logic state HIGH.
ROT	1	1	0	1	0	0	send the output device number to the output device, output the correlation result, and clocks the output latches.

Table 4.2 Functions performed by each word in the memory control and I/O field of microinstructions.

synthesizer is biased by the resistive network, shown in Fig. 4.9(a), to produce a voltage swing from zero to +4 volts. The system clock is derived by feeding the biased sine wave signal to a monostable multivibrator (74123) producing a train of 160 ns duration pulses. This re-shaped 5.0 MHz system clock is then divided by 1575 and 1550 to obtain timing pulses A and B that provide the Read Command and Sample and Hold, respectively. The timing diagram for the circuit of Fig. 4.9(a) is illustrated in Fig. 4.9(b).

To initialize the system (after power on), toggle switch No. 1 in Fig. 4.10 is switched from ground level position to the +5 V position and switch No. 2 is closed momentarily. Switch No. 1 causes the Q output of the J-K flip flop (7473) to be reset to logic level LOW. This action is taken primarily to block the system clock and timing signal so that the system can be synchronously initialized by the subsequent pushing of the momentary switch No. 2. Upon pressing switch No. 2, a portion of the timing pulse B is allowed to pass the NAND gate (74S00), and causing the Q output of the J-K flip flop (7473) to change from logic level LOW to HIGH. The transition from logic level HIGH to LOW of the $\bar{Q}$ output subsequently enables a monostable multivibrator (74123) which initializes the MCU by forcing the logic level on the primary bus to go HIGH. The logic HIGH level on the primary instruction bus is loaded into the microprogram address register in the MCU by simultaneously clocking the LD input with the Q output of the monostable multivibrator. In the meantime, the system clock and timing pulses are allowed to proceed to their destinations because of the logic level HIGH of the Q output of the flip flop. Any further clocking of the flip flop is prevented by ANDing its $\bar{Q}$ output which is at logic level LOW with the timing signal B; therefore, the clock input is forced to logic level HIGH, and no change of logic state at the flip flop output is possible.

(B) The clock pulse of the MCU and CPA are extended whenever a memory reference cycle is required by the processor. This elongated clock pulse is needed, because the maximum access time of the memory together with the propagation delay of the MCU and CPA is longer than a standard 160 ns clock pulse. An elongated clock pulse can be obtained from ORing the system clock with the output of a monostable multivibrator in Fig. 4.11(a). Having been triggered by the RF signal from the memory control and I/O field of microinstruction, the monostable multivibrator produces a pulse which is approximately 300 ns wide. Since the system clock cycle is only 200 ns, it is obvious that an elongated system clock pulse is obtained at the output of the OR gate whenever the monostable multivibrator is triggered. The output of the OR gate is used to drive the MCU and the pipeline.

When nondestructive testing on the data in the CPA's accumualtor or registers are called for, the test results must be prevented from clocking into specific registers. This means that the transition between logic level HIGH to LOW of CPA clock must be avoided. In other words, whenever nondestructive testing is required, the CPA clock pulse is extended as in the case of the memory reference cycle. This can be easily accomplished with a second OR gate having MCU clock and INH signal from the memory control and I/O field of microinstructions as its inputs. INH signal will go from logic level LOW to HIGH whenever a nondestructive test is needed; therefore this extends the CPA clock pulse for the particular cycle. The circuit and timing diagrams for generating extended clock pulses for the memory reference cycle and nondestructive test cycle are shown in Fig. 4.11.

(C) At the end of every correlation cycle, the system control circuitry sends a Read Command pulse to the processor, forcing the microprogram to jump out of the idle state. To achieve this, the address control field of microinstructions have to be altered to realize the transition. The circuit diesigned to initialize the jump is depicted in Fig. 4.12(a). Before the arrival of the Read Command signal, Q_1 and Q_2 are both at logic level LOW. Q_1 will change logic level when the flip flop is clocked by the Read Command signal. This change is propagated to Q_2 on the MCU cycle immediately following the Read Command signal. Now in the logic level HIGH state, Q_2 output is inverted through on open collector output NAND gate having its output tied to the least significant bit (AC_o) of the address control functions. Bit AC_o will be pulled LOW whenever Q_2 is HIGH, thus changing the next microprogram address. Leaving the idle loop, the processor, then, carries out the data transfer mentioned earlier. Q_1 is reset to LOW when Q_2 goes HIGH; therefore, when the next MCU clock pulse arrives, Q_2 will also go LOW, and bit AC_o will be allowed to follow its original bit pattern in the microprogram. Figure 4.12(b) illustrates the timing of the events described above.

(D) Normally, 16 replica channels are time multiplexed onto the primary instruction bus of the MCU to be correlated with the input samples as described in Section 4.3.2. In an interval of 80.64 ms, time needed for the correlation results to raise from minimum to maximum and fall back to minimum value, 128 different correlation results may be displayed on the storage oscilloscope. If this occurs, it may cause difficulties in trying to visually identify the outputs from the same replica channel across the scope screen as shown in Fig. 4.13. This difficulty can be alleviated by demultiplexing the replica channels such that only the output from one particular replica channel is displayed at one time.

Each Doppler shifted replica signal is stored in a ROM organized in a 256 word by 4 bit format of which the first 128 samples are the inphase components and the last 128 samples are the quadrature components. Twelve address bits are required to address a bank of 16 ROMs; the 4 most significant bits are decoded through a 4-line to 16-line decoder for the chip selects on the

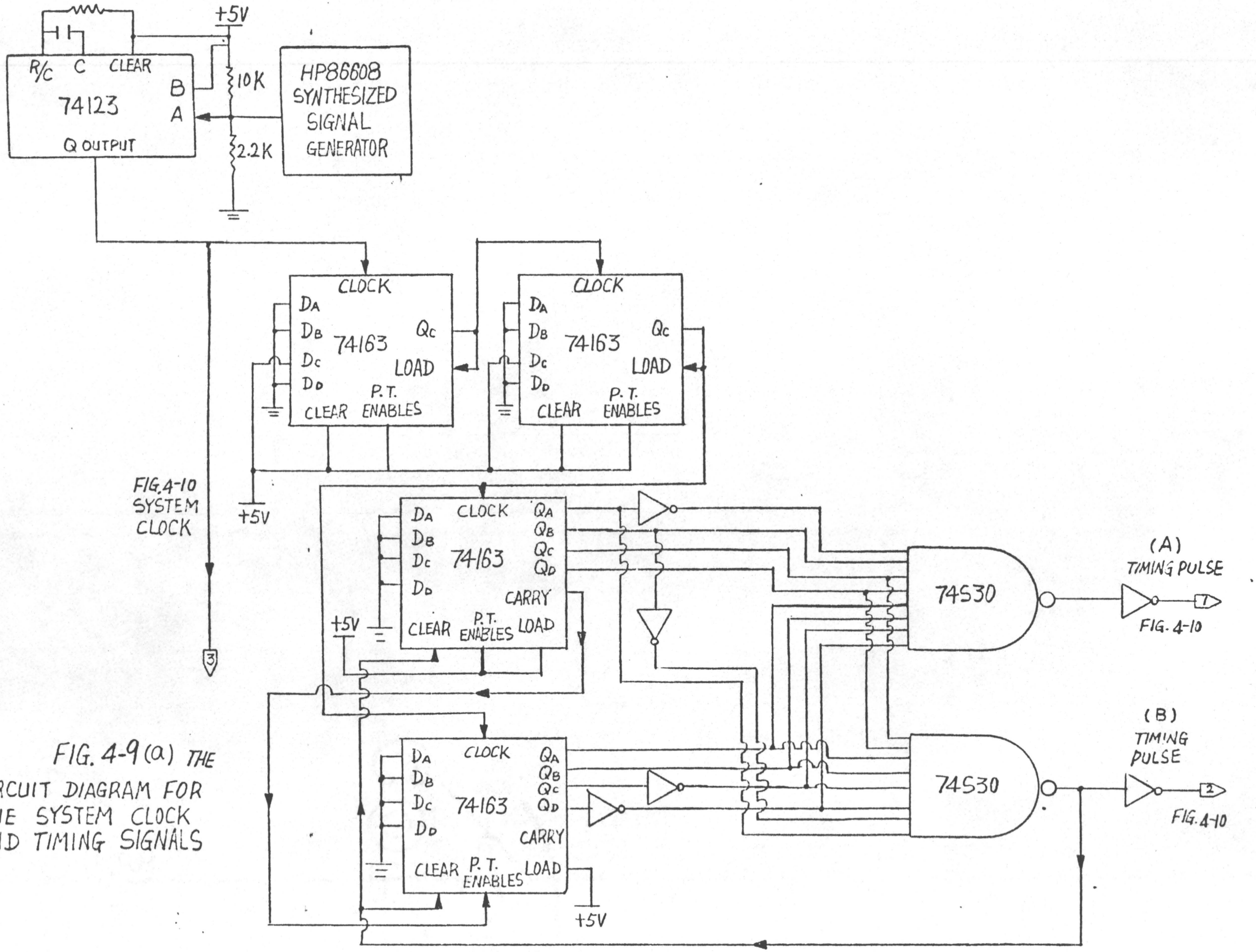


FIG. 4-9 (a) THE CIRCUIT DIAGRAM FOR THE SYSTEM CLOCK AND TIMING SIGNALS

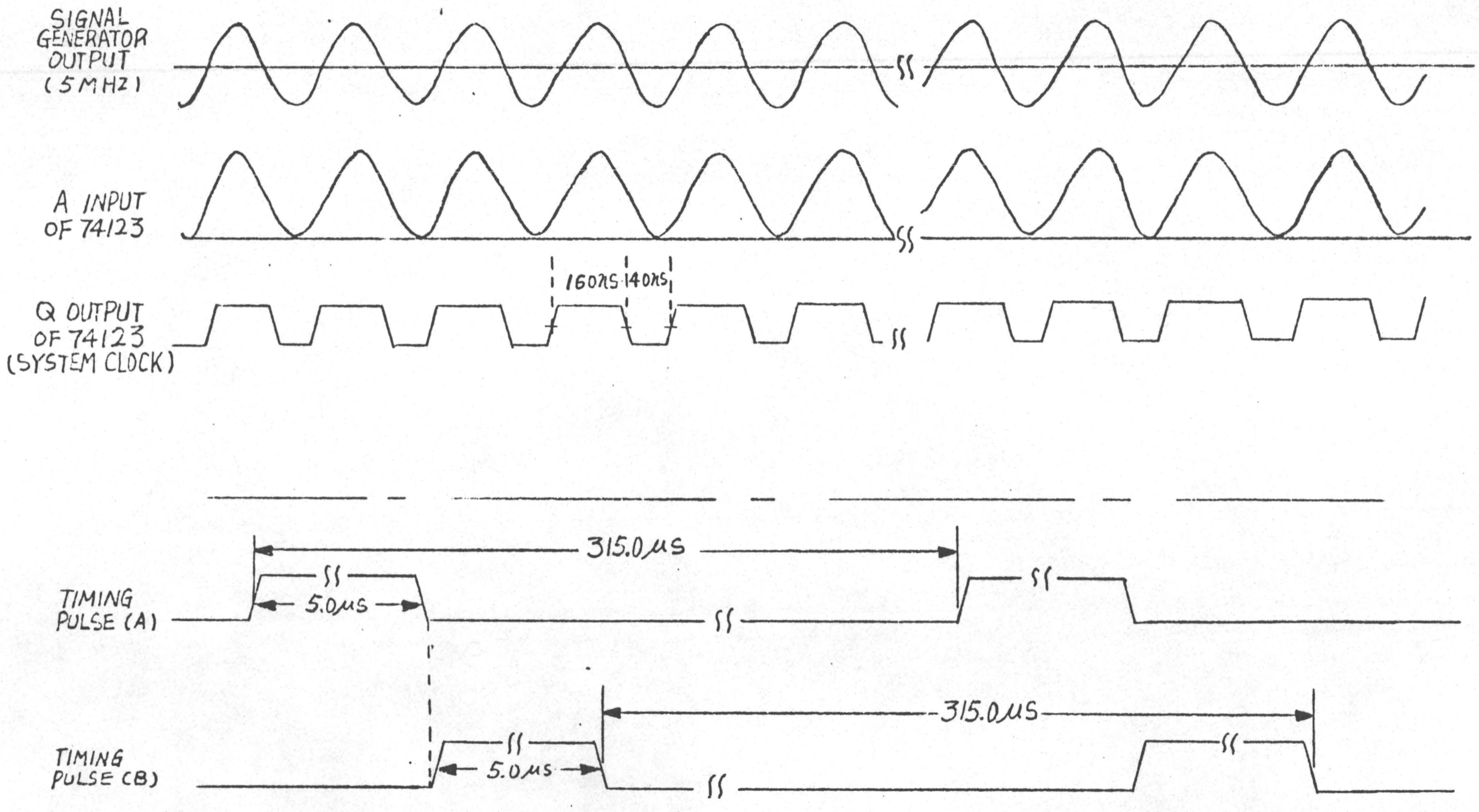


FIG. 4-9(b) TIMING DIAGRAM FOR THE CIRCUIT IN (a)

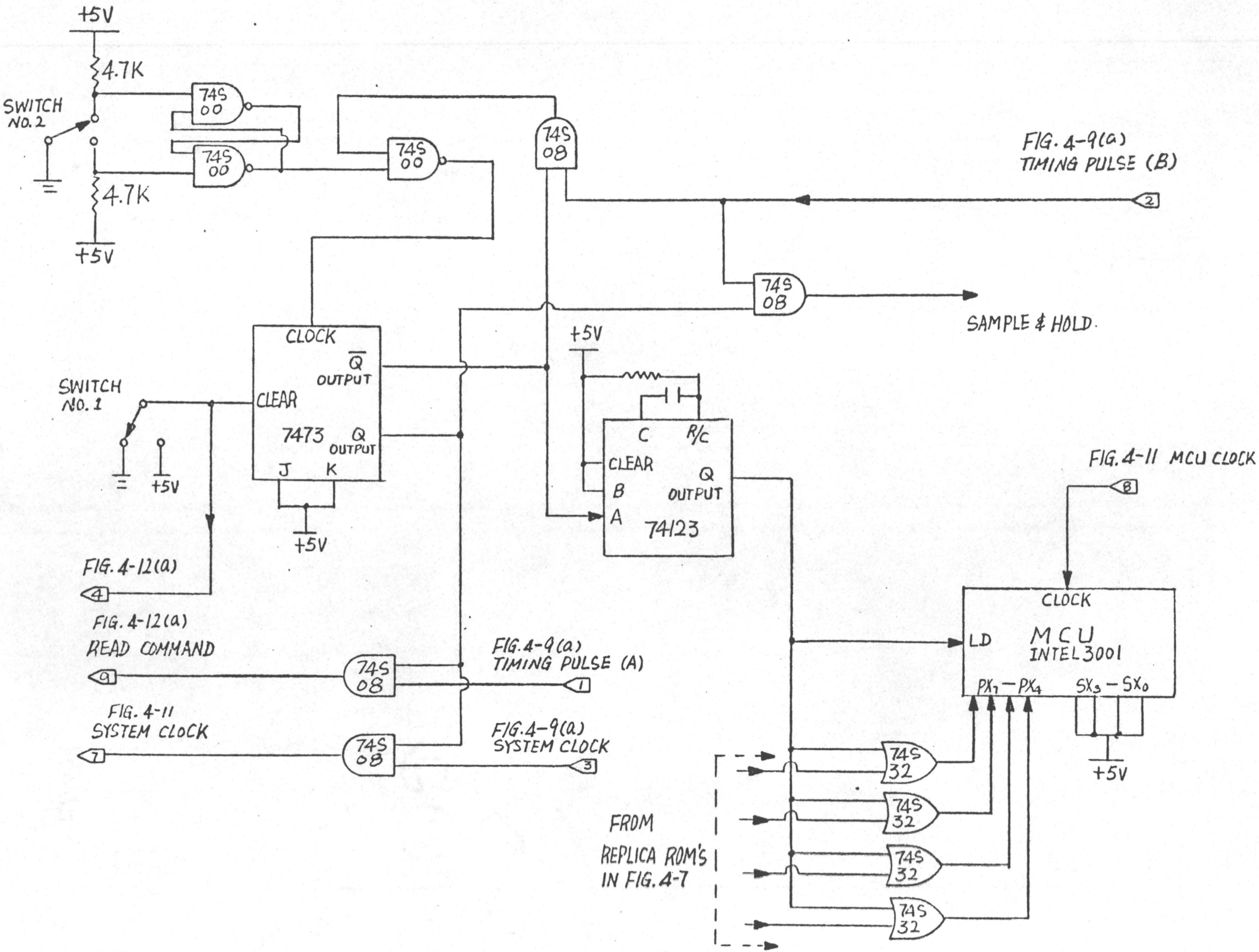
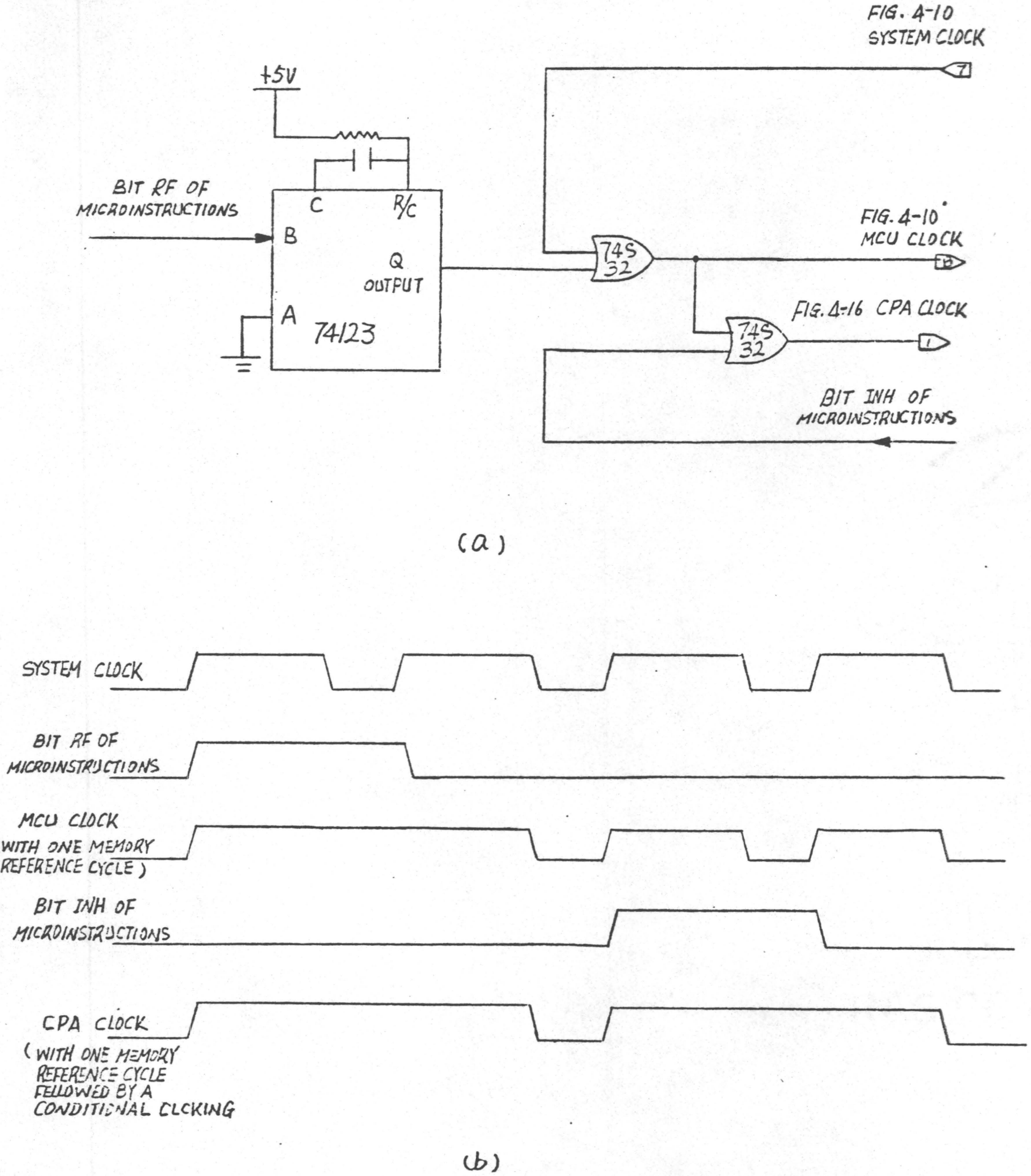


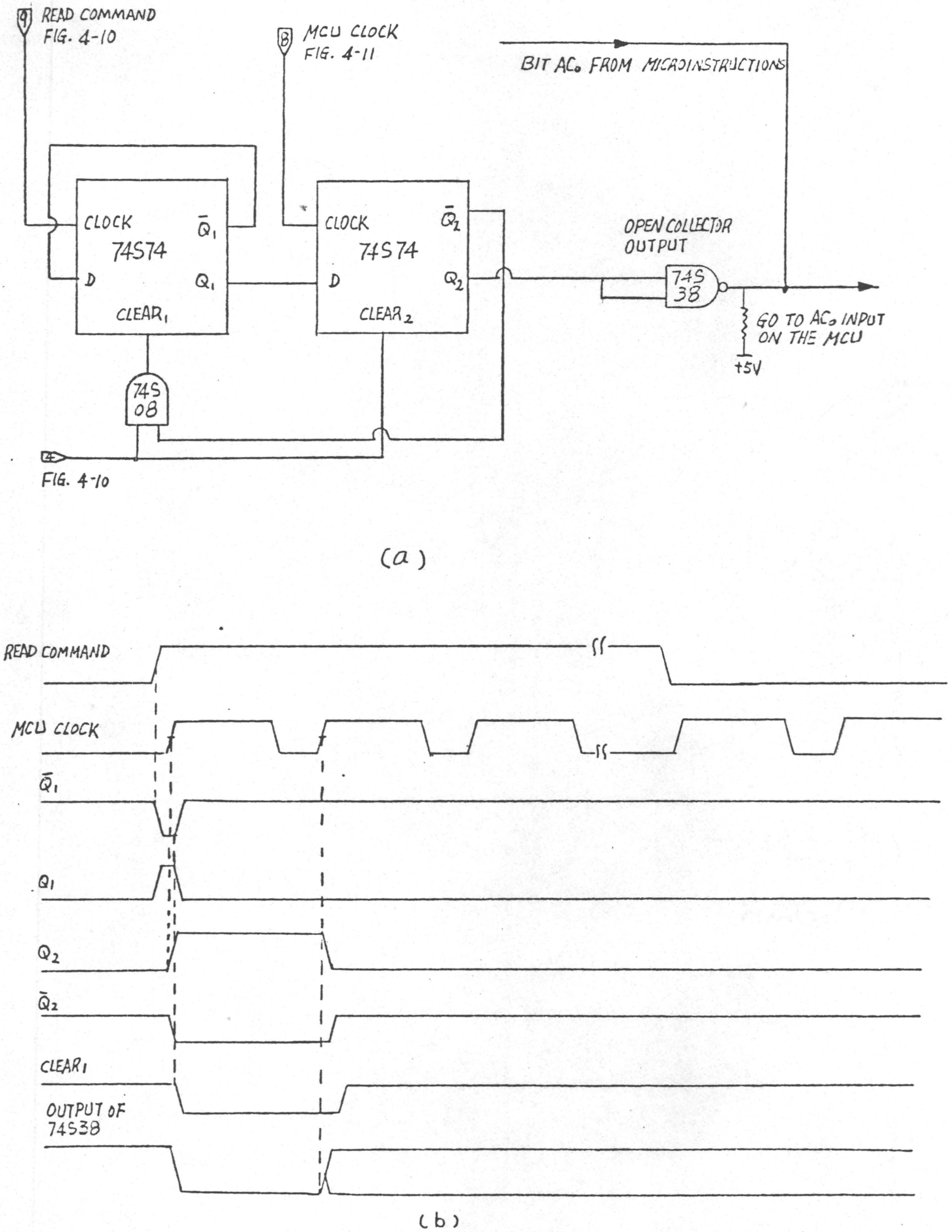
FIG. 4-10 THE CIRCUIT DIAGRAM FOR INITIALIZING THE SYSTEM



(a)

(b)

FIG. 4-11 (a) THE CIRCUIT DIAGRAM FOR GENERATING ELONGATED CLOCK PULSES FOR MEMORY REFERENCE CYCLE AND CONDITIONAL CLOCKING
(b) TIMING DIAGRAM FOR THE CIRCUIT IN (a)



(a)

(b)

FIG. 4-12 (a) THE CIRCUIT DIAGRAM FOR INITIALIZING A DATA TRANSFER FROM THE SYSTEM INPUT TO THE RAM'S. (b) TIMING DIAGRAM OF THE CIRCUIT IN (a)

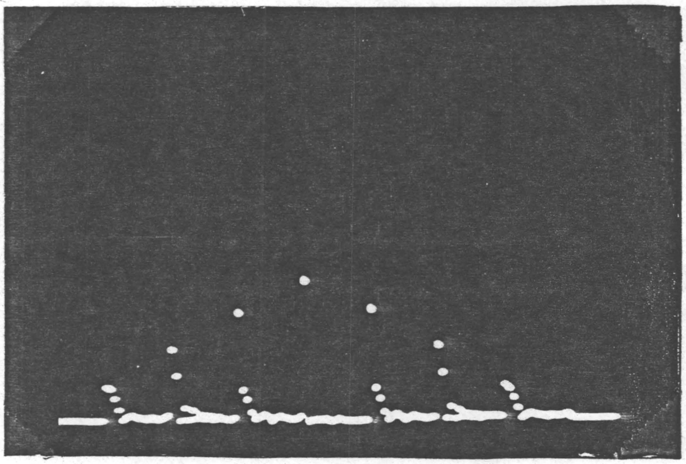


Fig. 4.13 A picture of the correlator output before demultiplexed.

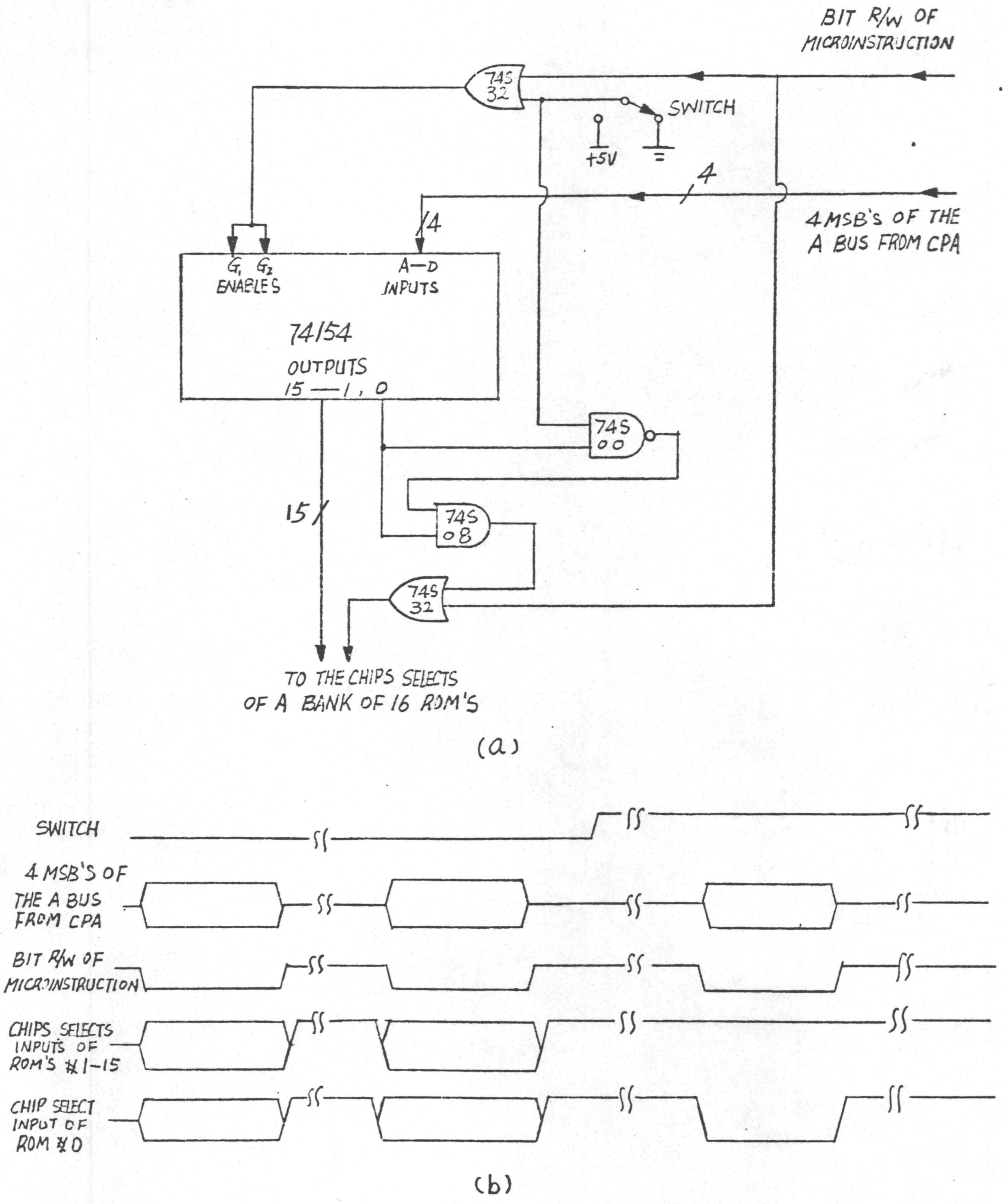


FIG. 4-14 (a) THE CIRCUIT DIAGRAM FOR MULTIPLEXING AND DEMULTIPLEXING OF A BANK OF 16 ROM'S. (b) THE TIMING DIAGRAM FOR THE CIRCUIT IN (a).

ROMs and 8 least significant bits are used to select a word in each ROM. When all the replica channels are time multiplexed, each ROM is selected for a duration of 630.0 μs in a cycle of 10.08 ms.

With the toggle switch in Fig. 4.14(a) at ground level, the output of the OR gate follows the logic state of the R/W signal from the memory control and I/O field of microinstructions. The decoder (74154) is enabled whenever R/W is LOW. It is clear from Fig. 4.14(a) that under the present condition anyone of the 16 ROMs can be selected. However, if the toggle switch is placed at +5 V, the decoder is disabled, and all 16 outputs stay at logic level HIGH. When this happens, the output of the AND gate (74S08) stays at LOW as long as the switch remains at +5 V position. Thus, as shown in Fig. 4.14(b), only ROM #0 can be selected when R/W goes LOW. As the result of this, only the replica signal at the position of ROM #0 can be correlated with the input signal; consequently, only the correlation results from this channel are displayed.

4.4 MULTICHANNEL PROCESSOR

As mentioned before, to obtain a 360° coverage in azimuth of the immediate surroundings of a sonar system, one of two techniques may be employed. Either a single processor can be used to process the signals in each resolution cell sequentially, or several signal processors paralleled together can be used such that signals in all the resolution cells are processed simultaneously. In the previous sections the hardware realization of a single channel real-time digital correlator based on the Intel 3000 series bit slice microprocessor has been described. Thus, a logical extension of this work is to examine the means by which this single channel processor may be expanded into a multi-channel processor.

The most direct and intuitive method of achieving this goal is to duplicate every component of the existing processor as many times as the number of the channel in a system. Although this may be the easiest and most obvious approach, it is not a economical way as far as the hardware requirements are concerned. If the hardware realization of the single channel processor is examined closely, it is not difficult to find that some of the components need not be duplicated. In fact, with a little more additional system control circuitry, the only components needing duplication are the CPA and carry look-ahead generator. Since all the channels would be synchronized and using the same replica correlation technique, it is apparent that only one set of 16 stored replica signals is required. Furthermore, by rearranging the microinstruction sequence of the microprogram, the duplication of the MCU can also be avoided.

The requirement for one MCU and the microprogram it addresses to control all the processors, clearly, will have to be that all the processors require identical microinstruction sequences for their processing needs even after a conditional test and branching. To have all the CPAs follow the identical microinstruction sequences is feasible here only if, at any instance, the input sample to each individual channel are the same. Unfortunately, this is not the case. The input sample for each individual channel differs from other channels, therefore, during the processing, after each conditional test, the microprogram branches into two different microinstruction sequences, depending on the test results, each CPA selecting one of the sequences for execution.

During each correlation operation, each piece of input data read into the accumulator of a CPA is tested for its polarity, i.e., negative or positive. The test result is stored for restoring the polarity of processed data. Moreover, before calculating the envelope value of the correlation results from inphase and quadrature processing, any negative correlation results of these two components must be inverted into positive number. Also, because of the algorithm used to approximate the envelope value, the knowledge of the relative magnitude of the two components is required. For all these tests each channel has its own possible outcome, and depending on each outcome, the CPA selects and executes one of the two possible microinstruction sequences that merge at later stage of the microprogram. If somehow the test result of the input data can be stored and retrieved from a separate register and branching of the microprogram after each conditional test is realized serially instead of parallely for all the channels, then a microprogram containing all the microinstruction sequences can be used to control an array of CPAs.

The Intel 3000 series microprocessor is static in nature in the sense that it does not require a minimum clock frequency to keep the device in operation. Furthermore, when the clock of the CPA goes LOW, the CPA becomes inactive and it will stay inactive as long as the clock is LOW. By using these two characteristics of the device, depending on the needs of each individual channel, the microinstructions can be selectively inhibited. To realize this, the existing microprogram must be rearranged such that all jump instructions (except the system jump instruction which is defined here as the jump instruction after a system conditional test resulting a uniform outcome for all the channels, e.g., test to see whether both inphase and quadrature correlations have been performed for a replica signal) after a conditional test are omitted and two different microinstruction sequences, one for each outcome, are placed consecutively immediately after each test as shown in Fig. 4.15. Regardless of each test outcome, all the channels will receive the same microinstruction sequence. Hence, it is up to the hardware and the test result of a particular channel to select which portion of the microprogram is to be executed by the CPA. The functions of the additional control

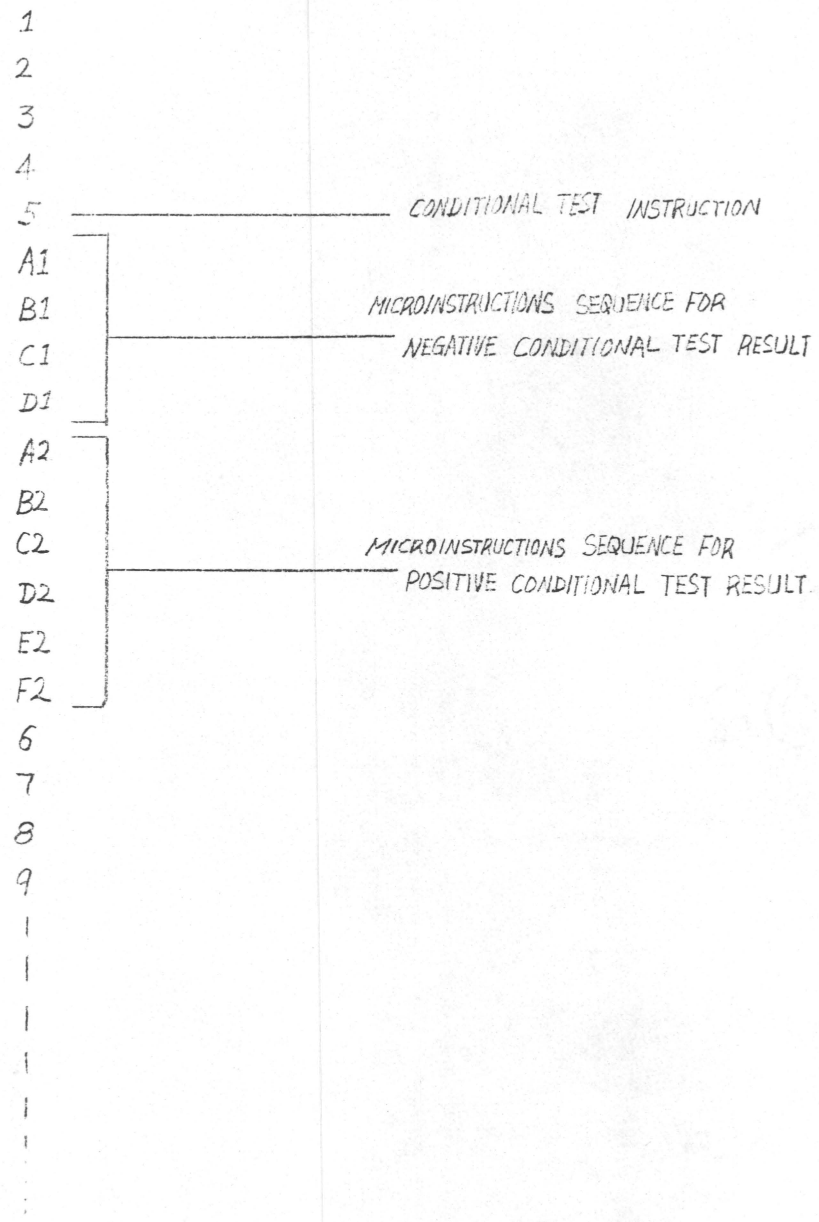

1
2
3
4
5
A1
B1
C1
D1
A2
B2
C2
D2
E2
F2
6
7
8
9
CONDITIONAL TEST INSTRUCTION
MICROINSTRUCTIONS SEQUENCE FOR NEGATIVE CONDITIONAL TEST RESULT
MICROINSTRUCTIONS SEQUENCE FOR POSITIVE CONDITIONAL TEST RESULT

FIG. 4-15 THE ORGNIZATION OF THE MICROPROGRAM FOR MULTICHANNEL CPA PROCESSING.

circuit needed and the outcome of each conditional test for each channel is utilized as follows: After a conditional test, the CPA clocks to all the channels whose outcome is positive are held at logic level LOW and are not activated until the microprogram reaches address A2 in Fig. 4.15; therefore, the microinstructions for the negative outcome, from A1 to D1, are not executed by the CPAs of the channels having a positive conditional test outcome. On the other hand, the clocks to the CPAs of the channels having a negative conditional test outcome are allowed to go through, thus, allowing the executions of the microinstructions from A1 to D1. At the end of the execution of D1 and the beginning of A2 the clocks to the CPAs of the channels having a negative conditional outcome are held at logic level LOW and are not activated until the microprogram reaches address 6; therefore, microinstruction for the positive outcome, from A2 to F2, are not executed by the CPAs of the channels having a negative conditional test outcome. Note, at address 6, clocks to all the CPAs have been restored. Since the microinstruction in address 6 and thereafter are common to all the channels. These operations described above is repeated whenever a conditional test on the channel level is called for.

4.5 HARDWARE REQUIREMENTS FOR A MULTICHANNEL PROCESSOR

In the realization of the expansion of the single channel processor to a multichannel system, not only a CPA and a carry look ahead generator are requried for each channel. Also required are the hardware for storing and retrieving the carry bit from the most significant bit of each channel and the hardware for stopping the CPA's clock at appropriate instance. A new micro field, the conditional test field, could be incorporated in microinstructions to facilitate the hardware implementaton.

The simple circuit designed for these two task is depicted in Fig. 4.16(a). When a system conditional test is required, the outcome of the test for the first channel (or any other channel for that matter) is allowed to go to the flag input on the MCU. This is so because MF signal from the conditional test field of microinstruction is at logic level HIGH. The test result is subsequently clocked into the F latch (see Fig. 4.3) in the MCU by the negative transition of the MCU dock, and is used by next address logic to derive the address for the next microinstruction. During all other conditional tests, the carry bit is prevented from reaching the flag due to logic level LOW of the MF signal. Clearly, this portion of the circuit needs only be incorporated into one of the system channels, since, for system conditional test, the outcome for all the channels should be the same.

The most significant bits (MSB) of all the input data are shifted out of accumulators to determine polarities. Once this is accomplished, the polarities of the data must be restored. Since the MCU can only store one bit from one channel, another storage and retrieval scheme must be found. The circuit designed to store and retrieve the most significant bits of accumulators can be realized with a D type flip flop (74S74) and an AND gate, as shown in section A of Fig. 4.16(a). As the MSB rotates out of the accumulator, it is latched by the D type flip flop by clocking the clock input of the flip flop with the ST signal from the conditional test field of the microinstructions. When the MSB is required, the logic level of RR signal from the conditional test field of the microinstruction goes from LOW to HIGH and thus enables the stored value to pass the AND gate (74S09) to be presented at the left in (LI) input (see Fig. 4.2) on the CPA.

When the execution of the microprogram reaches a conditional test (depending on the test outcome), the undesired portion of the subsequent microinstruction sequence is prevented from being executed by blocking the clock to the CPA. The clock to the CPA is not restored until at the end of the undesired sequence as described in the last section. A circuit designed to realize the oepration described above is shown in section B of Fig. 4.16(a). At the beginning of the microprogram the J-K flip flop (7476) is set by the PR signal from the conditional test field of microinstructions; thus, the clock to the CPA is not blocked by the AND gate (74S08). After a conditional test, as usual, the result is output on the carry line (C_8) by the carry look ahead generator. Since the MF signal is normally at logic level HGIH (see Fig. 4.16(b)), depending on the test results, the J-K flip flop may be cleared. In the case when the test result is negative, Q_1 output stays at logic level HIGH, and microinstruction sequence, from address A1 to D1 in Fig. 4.15, for the negative result, is executed. At the end of the execution of the microinstruction in address D1, Q_1 is toggled from HIGH to LOW by the CB signal from the conditional test field of the microinstructions. Therefore, the clock to the CPA is blocked; execution of the subsequent microinstructions are prohibited. The CPA's clock is not restored by the PR signal until the beginning of the execution of the microinstruction in address 6 in Fig. 4.15. Similarly, if the conditional test result is positive, then J-K flip flop (7476) is cleared; clock passage to the CPA is blocked, and the microinstruction sequence for the negative test result is not executed. Q_1 is set to logic level HIGH at the end of negative microinstruction sequence, this is at address D1, thus enabling the clock to reach the CPA. Consequently, the execution of the microprogram by the CPA is resumed.

The expansion technique described here is clearly more economical and labour saving than the intuitive approach. However, since all the microinstructions in the microprogram are needed during each correlation operation, the time required for performing a correlation is increased slightly. In the present case, this should have no effect on the system performance, because the longest correlation calculation has been found to be approximately 290 μs, and the single channel processor has up to 315.0 μs for each correlation. This longer execution time

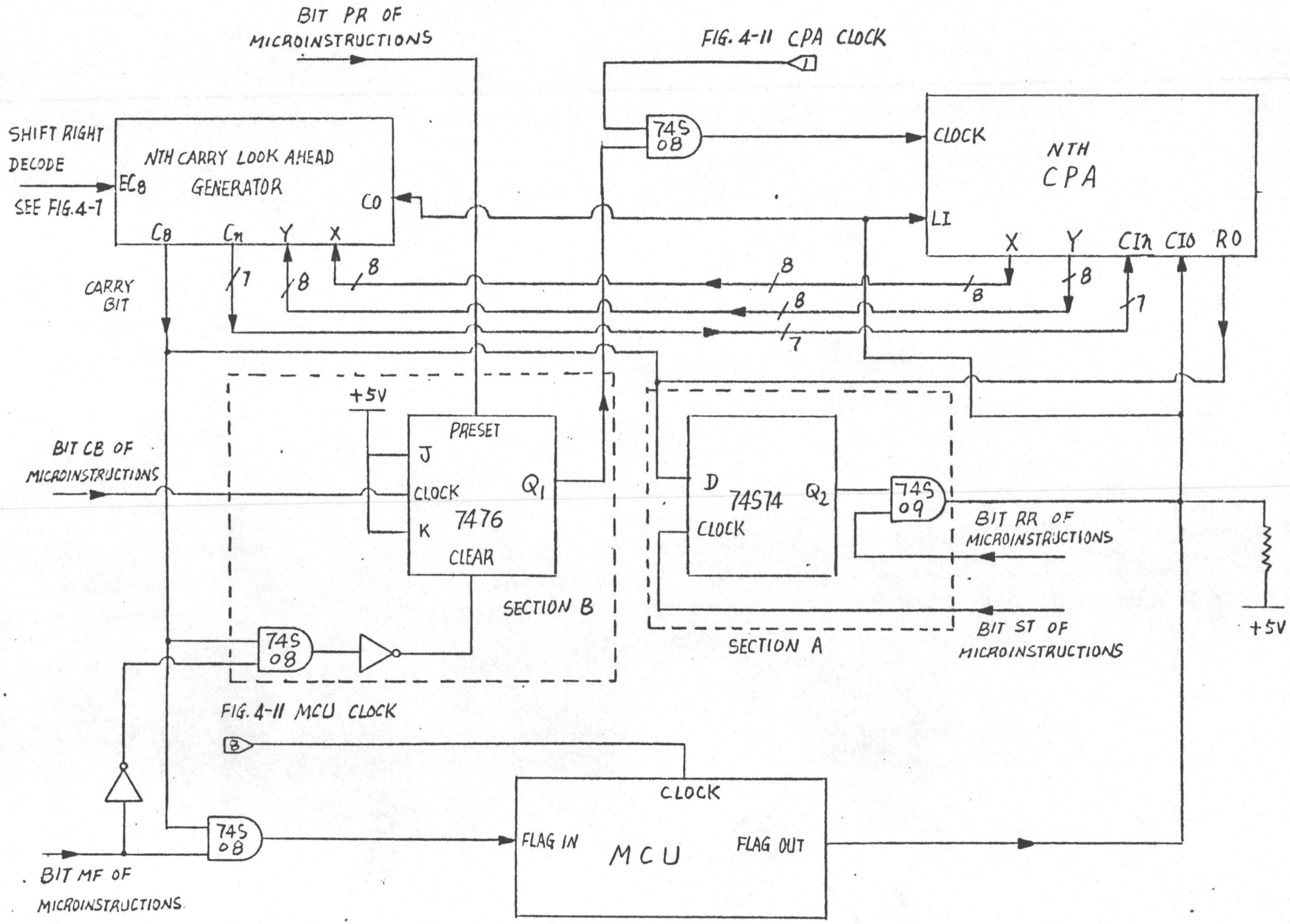
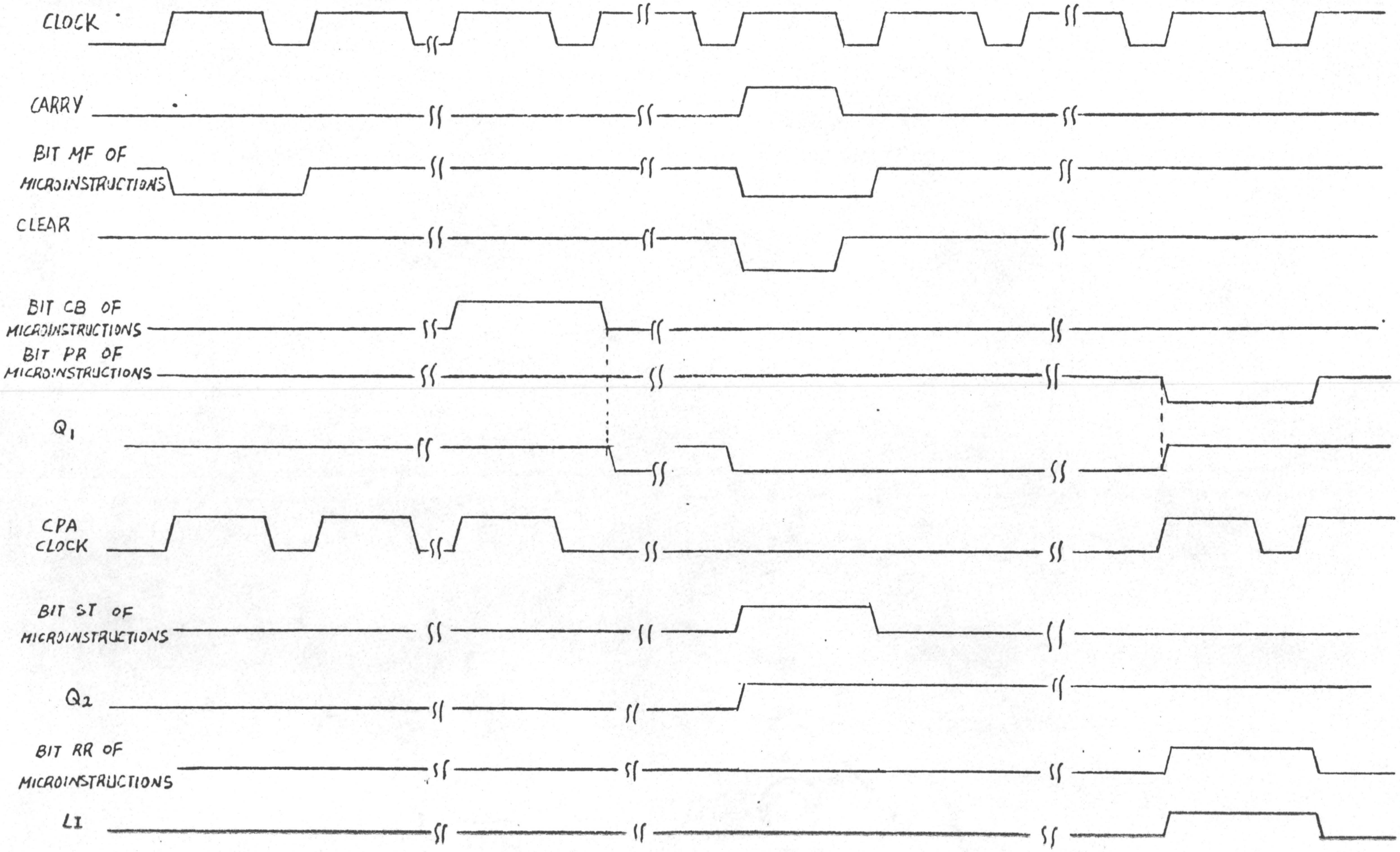


FIG. 4-16(a) THE CIRCUIT DIAGRAM OF THE NTH CPA CHANNEL OF A MULTICHANNEL PROCESSING SYSTEM.


CLOCK
CARRY
BIT MF OF MICROINSTRUCTIONS
CLEAR
BIT CB OF MICROINSTRUCTIONS
BIT PR OF MICROINSTRUCTIONS
Q1
CPA CLOCK
BIT ST OF MICROINSTRUCTIONS
Q2
BIT RR OF MICROINSTRUCTIONS
L1

(b)

FIG. 4-16(b) THE TIMING DIAGRAM OF THE CIRCUIT IN (a).

resulted from the expansion simply means a reduction of the idle time at the end of each correlation.

4.6 SUMMARY

A detailed hardware implementation of a single channel real-time digital replica correlator capable of resolving 16 Doppler frequency shifts has been described. The use of bit slice microprocessor in the design not only provides adequate speed performance but also enhances the systems flexibility through the incorporations of control signals into the microprogram. Based on the characteristics of the bit slice microprocessor, an expansion of the single channel processor into a multichannel system also has been discussed. It has been shown that by utilizing the static nature of the microprocessor the expansion of a single channel correlator into a N-channel system can be achieved with relatively simple hardware.

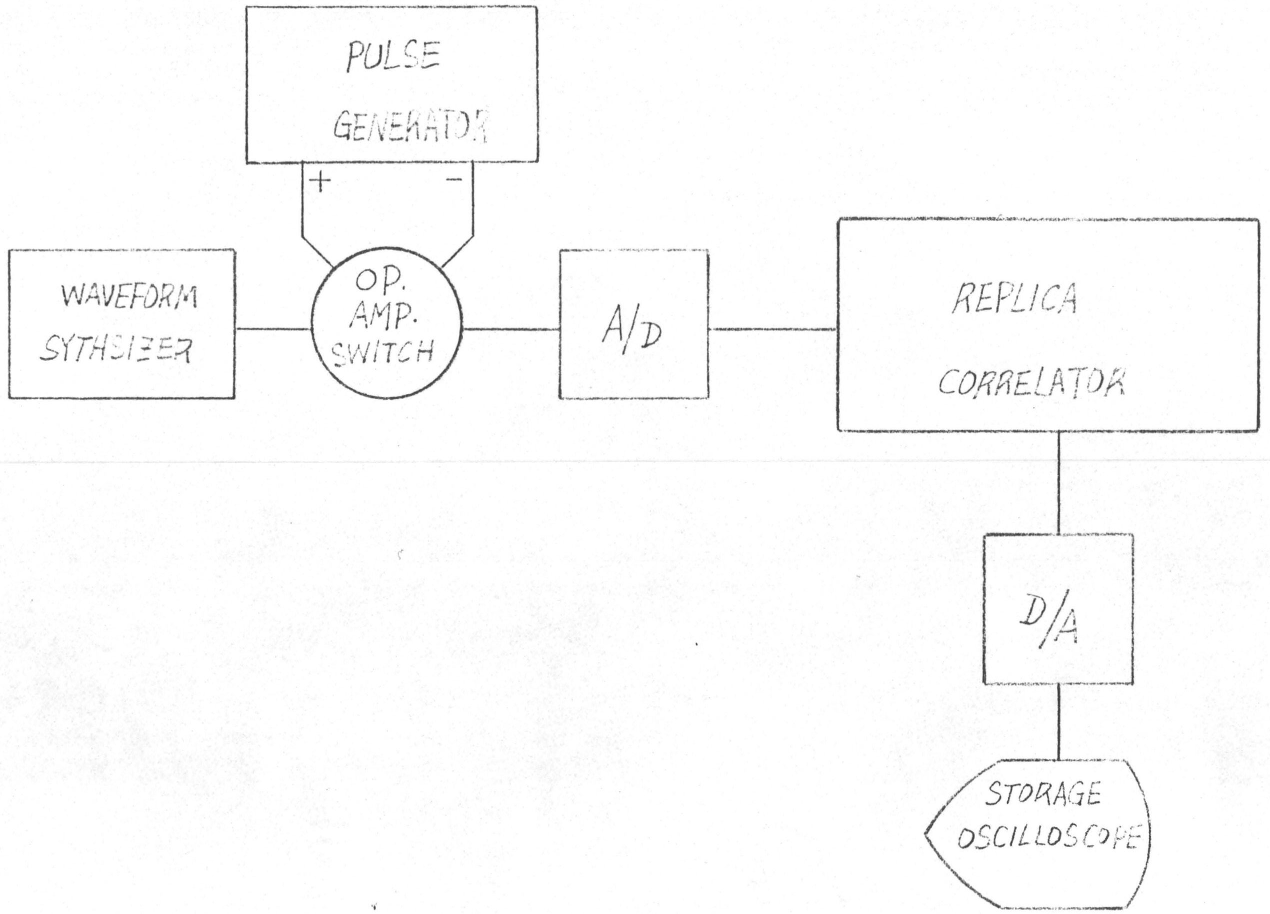


FIG. 5-1 TEST SETUP FOR CROSSCOUPLING MEASUREMENTS.

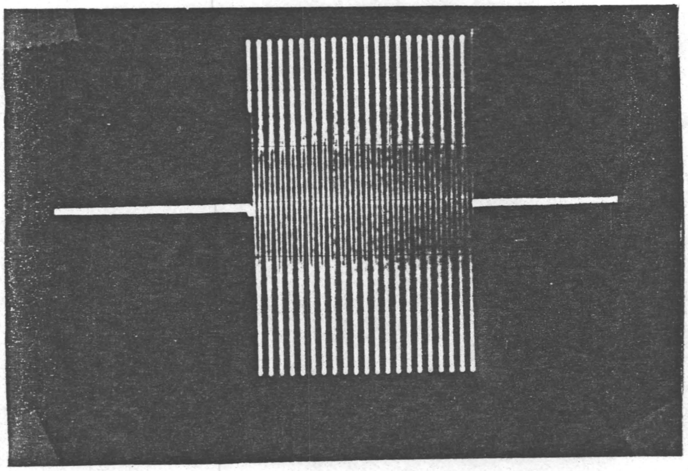


Fig. 5.2 A pulsed CW signal

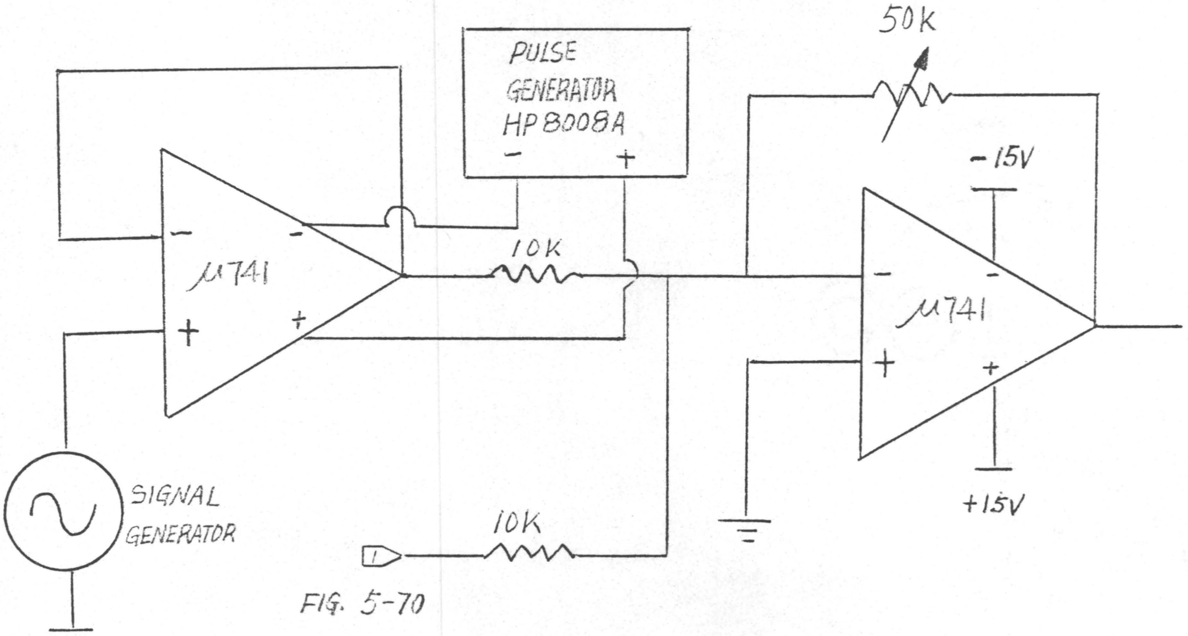


FIG. 5-3 OPERATIONAL AMPLIFIER SWITCH USED IN FIG. 5-1.

CHAPTER 5

EXPERIMENTAL RESULTS

A single channel real-time digital correlator described in the last chapter has been constructed and tested. In this chapter, three different tests have been conducted. They are:

(A) Crosscoupling Test: The output of a replica correlator measures the resemblance between the input signal and the replica signal. Naturally, the closer the two signals resemble each other the higher the correlation output. Moreover, due to the use of the five-level quantized sine wave as the replica signal, harmonic signals are generated which may fall into the expected Doppler frequency bandwidth from 300 to 700 Hz, thus causing an unexpected high correlation output at the harmonic frequency.

(B) Signal-to-Noise Ratio Test: Signals returned from a target are inevitably immersed in noies and reverberation; the signal-to-noise ratio test is a measure of the capability of the correlator in detecting an echo signal in a corrupted environment, in this case Gaussian noise.

(C) Extended Target Return Test: Due to the nature of the target and properties of the medium such as time stretching and multipath effect, a transmitted pulse signal returned by a target is usually stretched in time. The effect of this extended target return on the correlator output is investigated in this test.

5.1 CROSSCOUPLING MEASUREMENTS

The block diagram of the test setup for the cross-coupling measurements is shown in Fig. 5.1. Whenever the switch is pulsed, a pulsed CW signal shown in Fig. 5.2 emerges at its output. The switch comprises two cascaded operational amplifiers (μ7541), as shown in Fig. 5.3, the first of which is a voltage follower with power supplies pulsed by a HP8008A pulse generator. Variable amplitude gain output of the pulsed signal is achieved by the second operational amplifier which, in essence, is a variable gain analog adder. The second input of the adder is used in the signal-to-noise ratio measurement and in the investigation of the effects on the system output due to non-coherent extended target returns.

Pulsed CW at frequencies from 312.5 Hz to 687.5 Hz in the steps of 25 Hz are used for the crosscoupling measurement for the stored replicas of 312.5 Hz and 512.5 Hz. Figure 5.4 through Fig. 5.19 show the correlator output for stored replica of 312.5 Hz with digitally stored input CW signal at various frequencies. Note that the maximum output occurred only when the input frequency is at 312.5 Hz. Figure 5.20 through Fig. 5.35 show the correlator output for stored replica of 512.5 Hz with various input CW signal frequencies. Again, the output peaks only at when the input frequency is the same as the stored replica. The correlator output for fine frequencies shifts around 512.5 Hz are measured and the results are shown in Fig. 5.36 through 5.46.

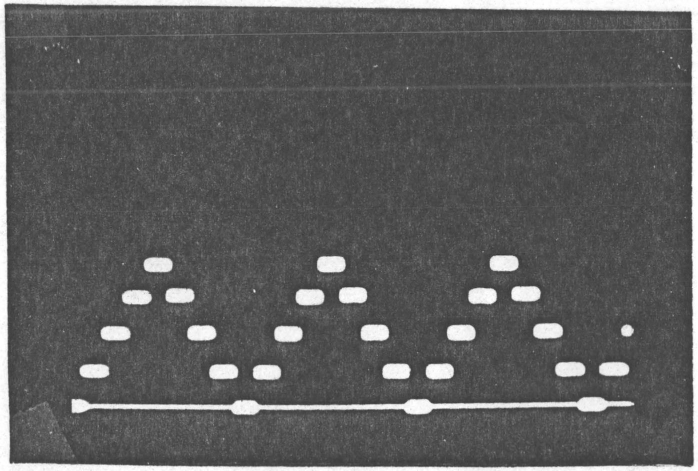


Fig. 5.4 f_{in} = 312.5 Hz
Scale: 0.5v/50ms

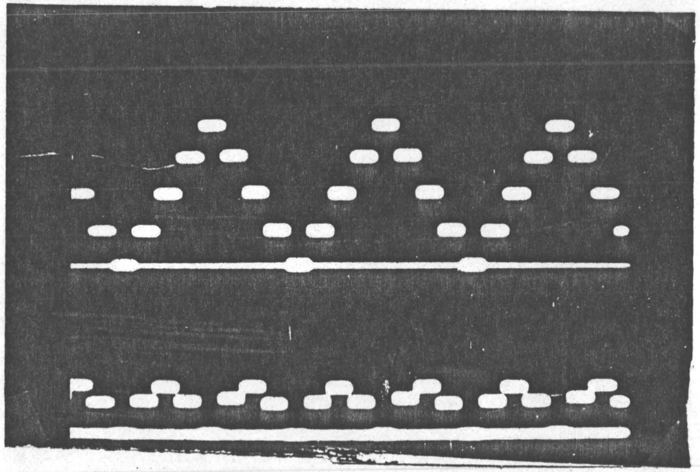


Fig. 5.5 f_{in} = 337.5 Hz (lower)
Scale: 0.5v/50ms
f_{in} = 312.5 Hz (upper)

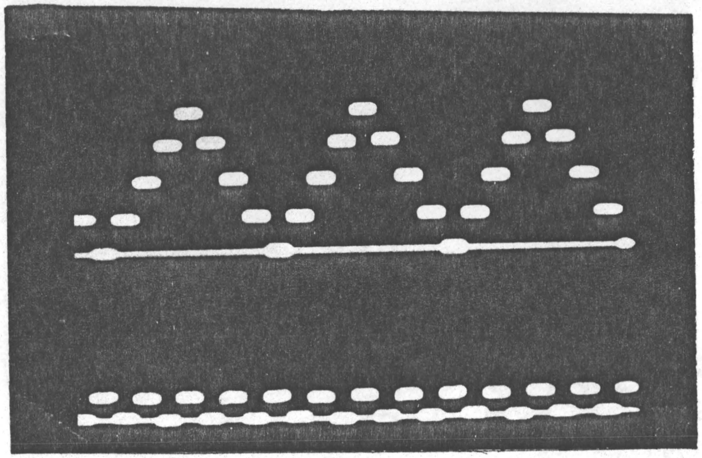


Fig. 5.6 f_{in} = 362.5 Hz (lower)
Scale: 0.5v/50ms

$f_{in=}$ = 312.5 Hz (upper)

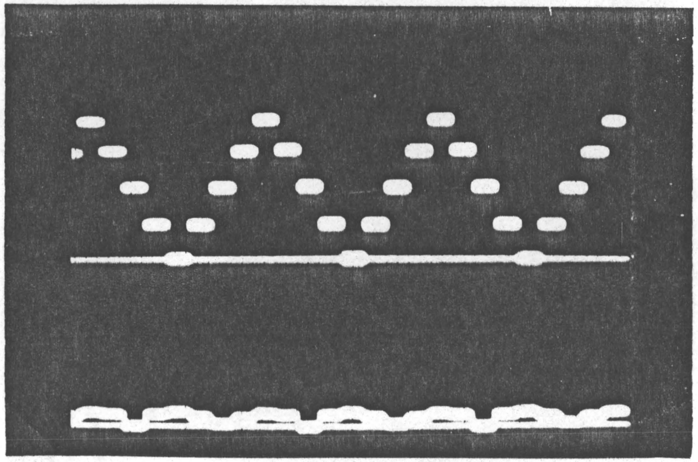


Fig. 5.7 f_{in} = 387.5 Hz (lower)
Scale: 0.5v/50ms

f_{in} = 312.5 Hz (upper)

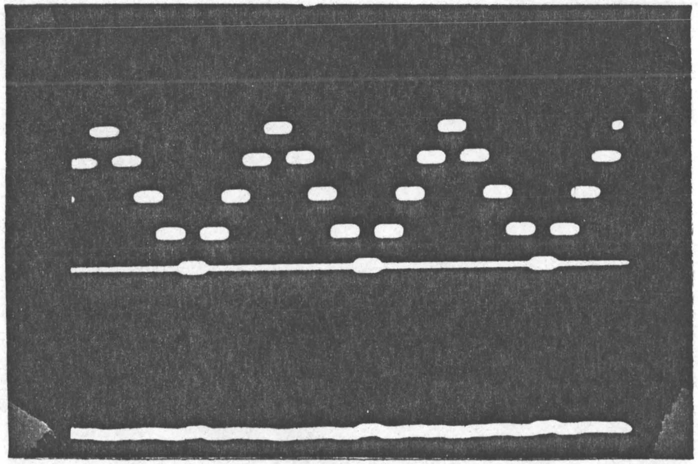


Fig. 5.8 f_{in} = 412.5 Hz (lower)
Scale: 0.5v/50ms

f_{in} = 312.5 Hz (upper)

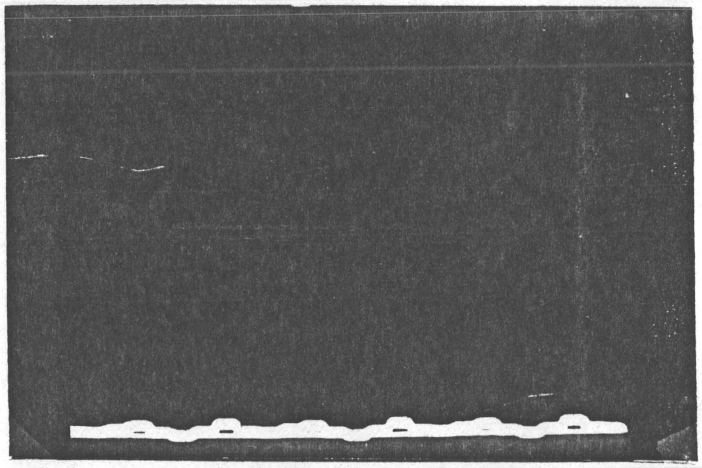


Fig. 5.9 f_{in} = 437.5 Hz
Scale: 0.5v/50ms

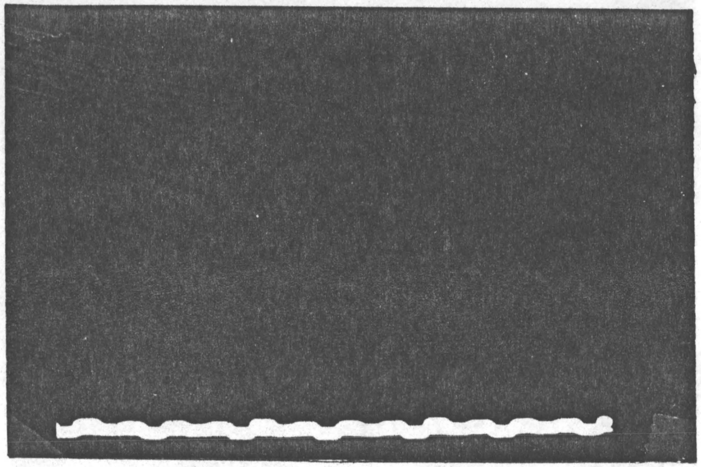


Fig. 5.10 f_{in} = 462.5 Hz
Scale: 0.5v/50ms

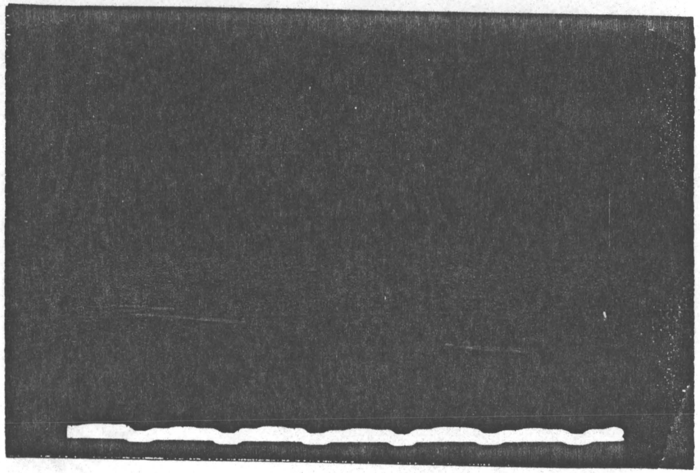


Fig. 5.11 f_{in} = 487.5 Hz
Scale: 0.5v/50ms

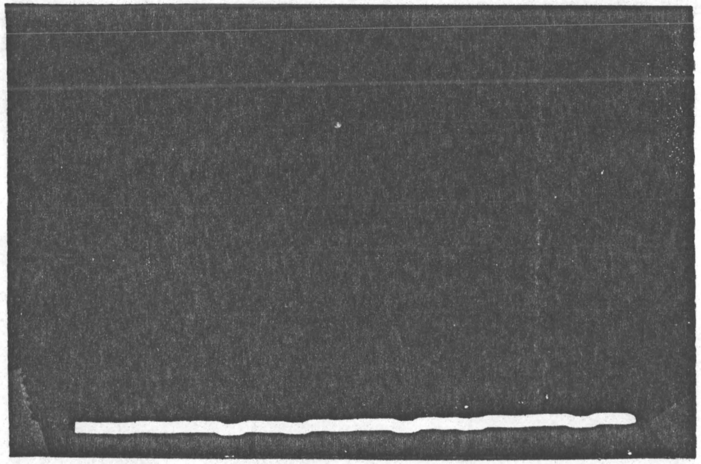


Fig. 5.12 f_{in} = 512.5 Hz
Scale: 0.5v/50ms

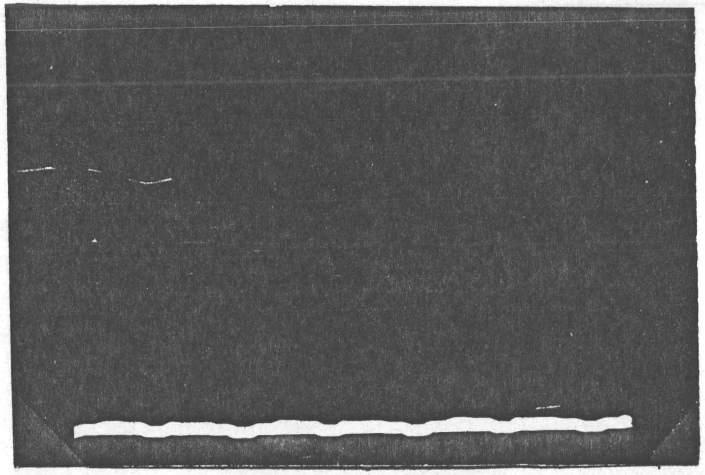


Fig. 5.13 f_{in} = 537.5 Hz
Scale: 0.5v/50ms

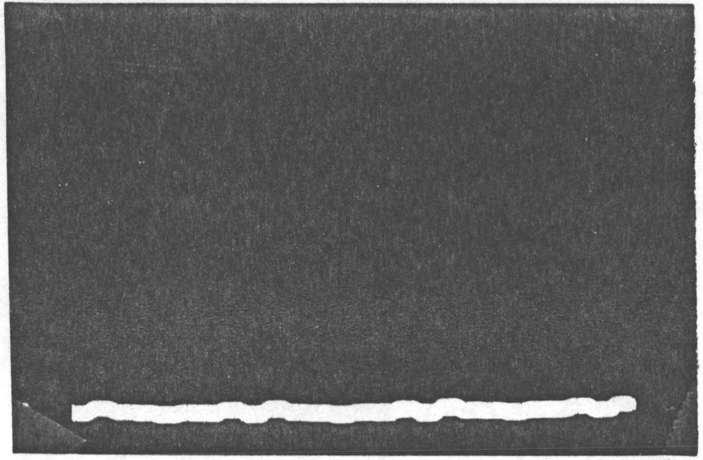


Fig. 5.14 f_{in} = 562.5 Hz
Scale: 0.5v/50ms

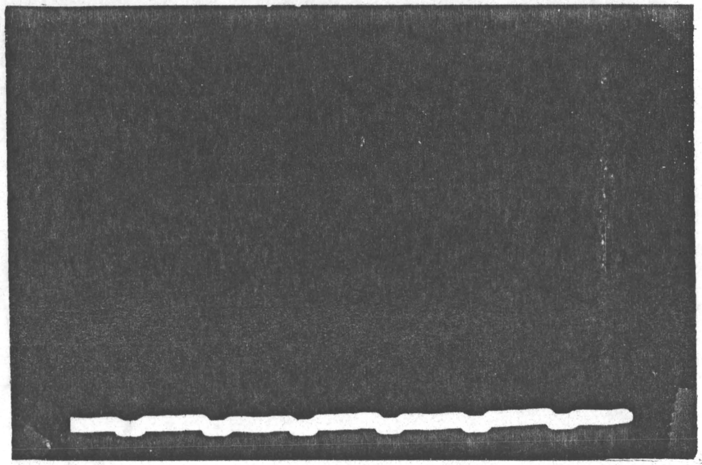


Fig. 5.15 f_{in} = 587.5 Hz
Scale: 0.5v/50ms

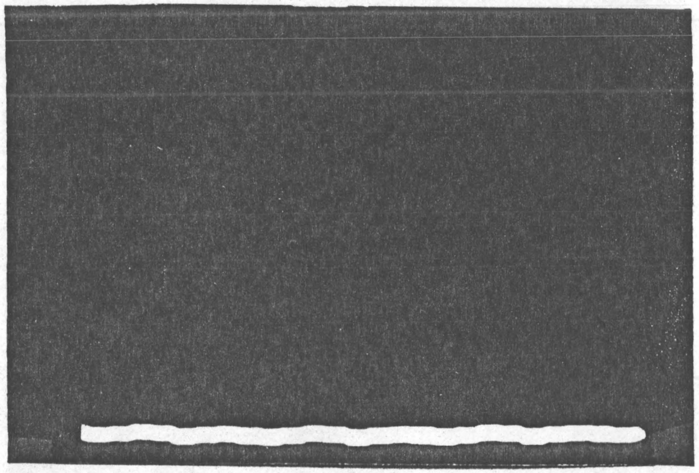


Fig. 5.16 $f_{in} = 612.5$ Hz
Scale: 0.5v/50ms

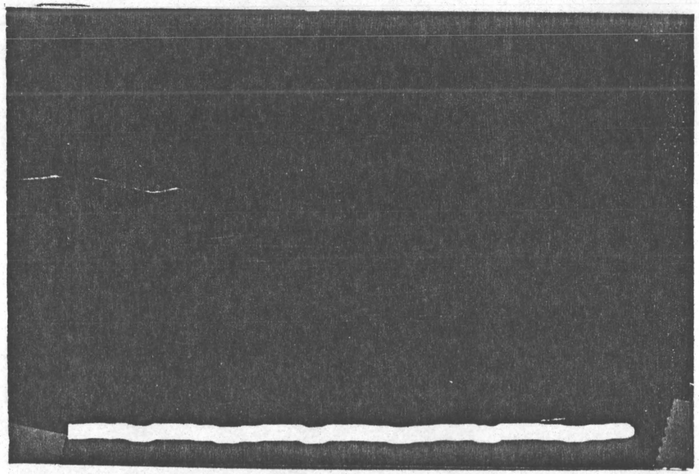


Fig. 5.17 $f_{in} = 637.5$ Hz
Scale: 0.5v/50ms

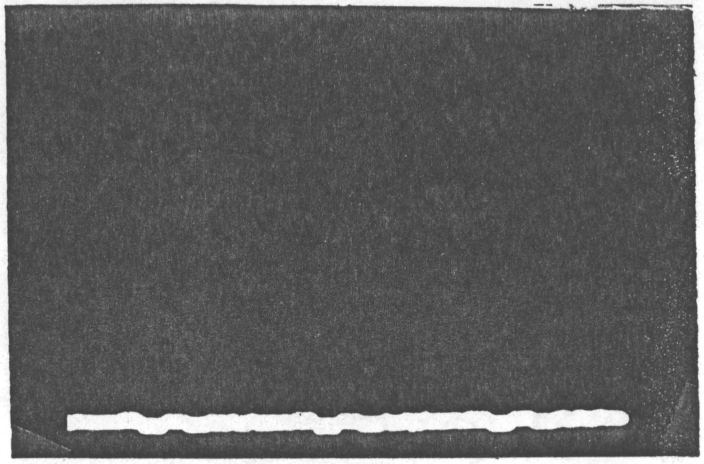


Fig. 5.18 $f_{in} = 662.5$ Hz
Scale: 0.5v/50ms

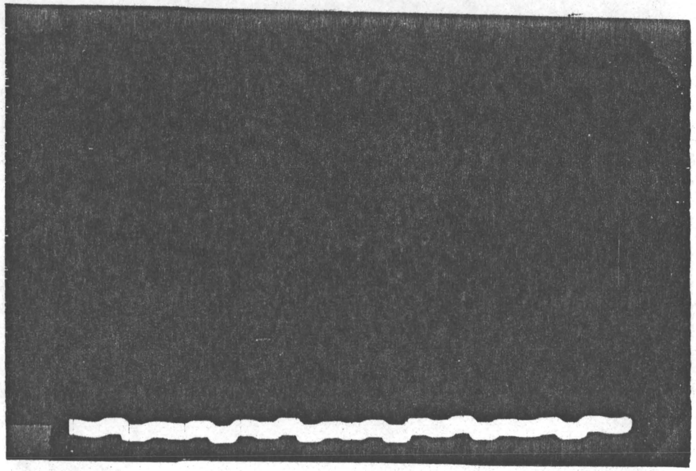


Fig. 5.19 $f_{in} = 687.5$ Hz
Scale:0.5v/50ms

It should be noted here that the theoretical response of the correlator is of the form

$$y(f) = \frac{1}{\pi(f_1-f_2)T} \sin \pi(f_1-f_2)T \qquad (5.1)$$

where f_1 is the frequency of the incoming signal, f_2 is the frequency of the replica and T is the pulse duration. The amplitude of the correlator output of detected pulse drops as $1/[\pi(f_1-f_2)T]$. In the case where f_1-f_2 = 25 Hz and T = 40.32 ms, this represents a 10 dB reduction in the response from the adjacent replicas when the input signal is centered at any particular replica frequency and a 3.8 dB reduction in response when the input signal frequency is at the center of two adjacent replica frequencies. The results obtained from the measurements are 9.8 dB and 3.7 dB reduction for the first and second cases, respectively, which is consistant with the theoretical predictions.

The crosscoupling between a raised cosine pulsed CW signal and stored replicas has also been measured; the results agree with the prediction that the crosscoupling would be more prominent in the adjacent replica channels when the input signal is centered at any particular replica and smaller in the other replica channels than the case when rectangular pulse are used. Figure 5.47 through Fig. 5.62 shows the correlator output for a digitally stored raised cosine pulsed signal with various replica channels.

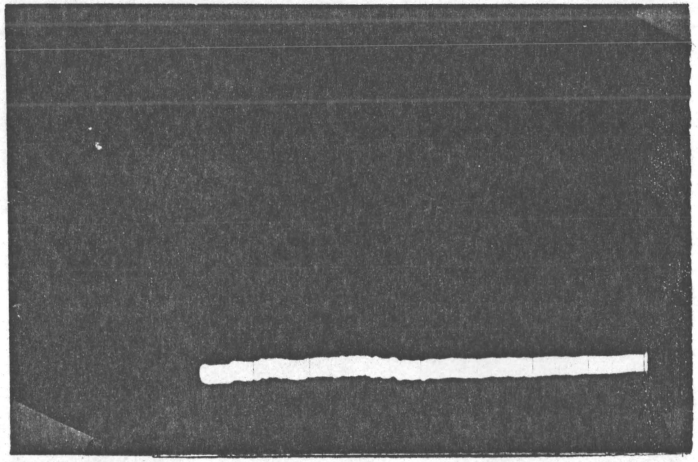


Fig. 5.20 f_{in} = 312.5 Hz
Scale: 0.5v/20ms

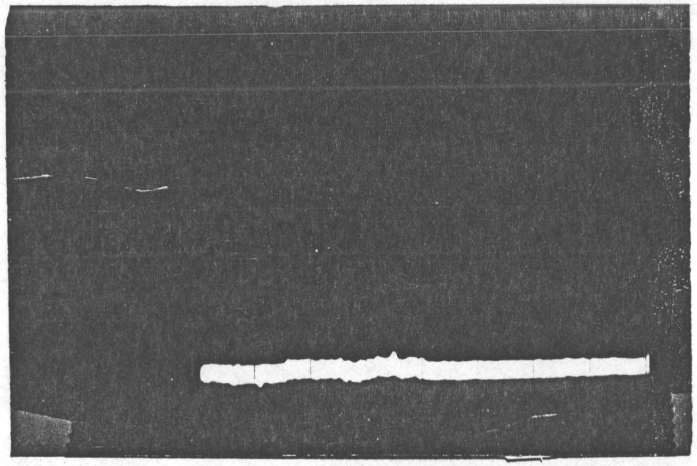


Fig. 5.21 f_{in} = 337.5 Hz
Scale: 0.5v/20ms

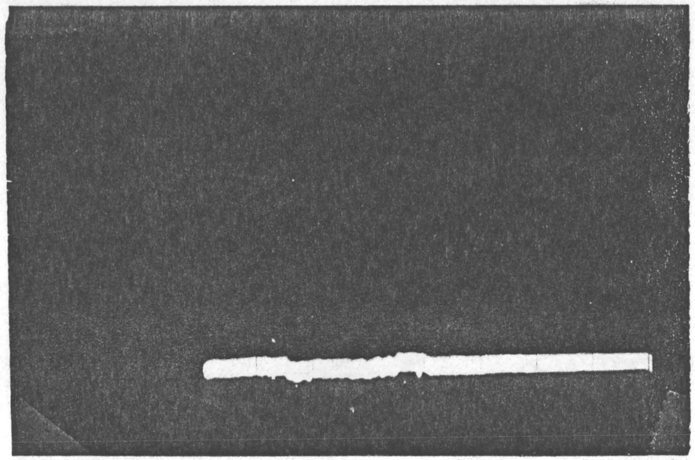


Fig. 5.22 f_{in} = 362.5 Hz
Scale: 0.5v/20ms

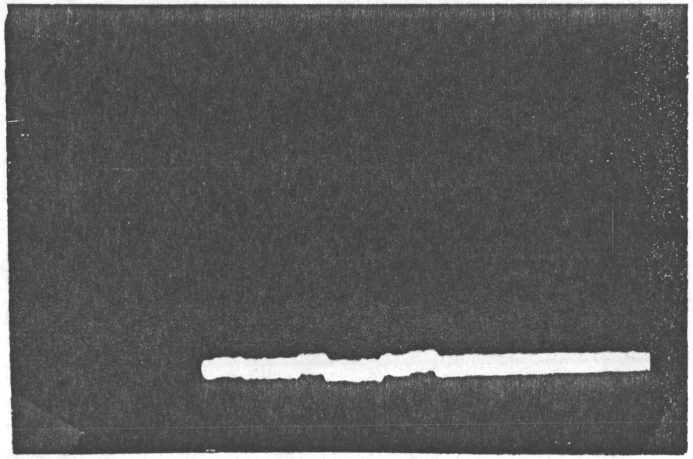


Fig. 5.23 f_{in} = 387.5 Hz
Scale: 0.5v/50ms

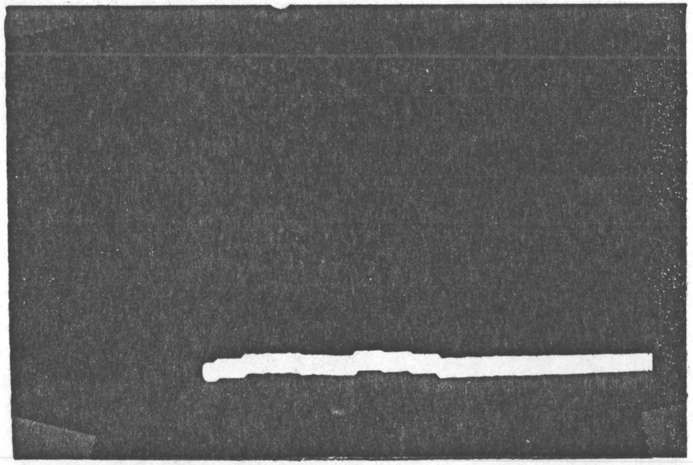


Fig. 5.24 f_{in} = 412.5 Hz
Scale: 0.5v/20ms

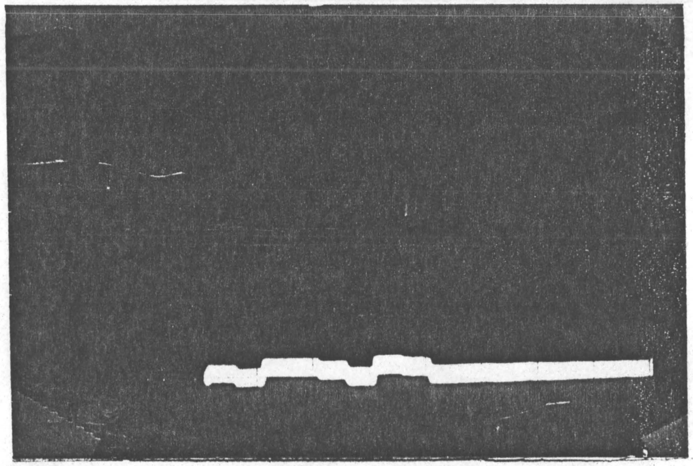


Fig. 5.25 f_{in} = 437.5 Hz
Scale: 0.5v/20ms

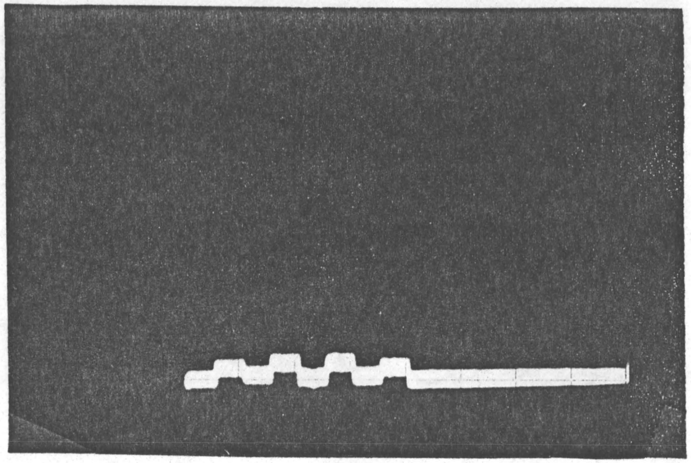


Fig. 5.26 f_{in} = 462.5 Hz
Scale: 0.5v/20ms

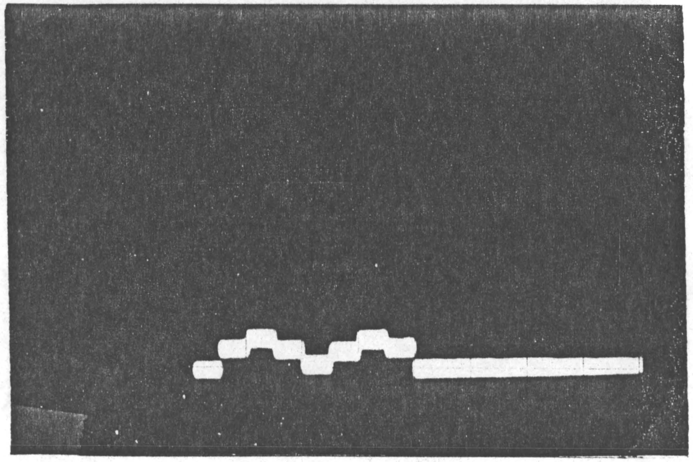


Fig. 5.27 f_{in} = 487.5 Hz
Scale: 0.5v/20ms

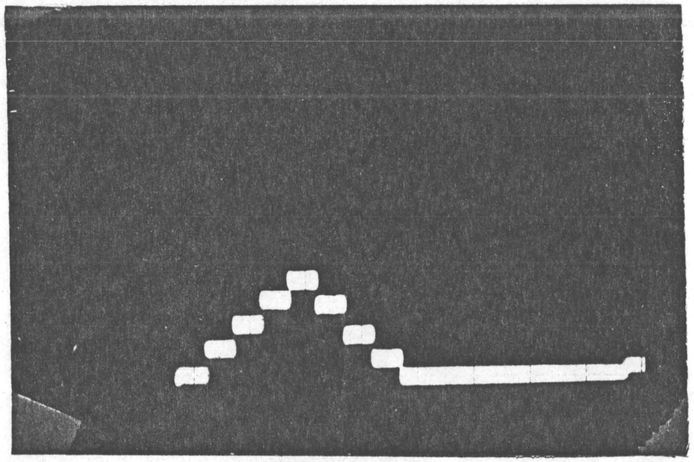


Fig. 5.28 f_{in} = 512.5 Hz
Scale: 0.5v/20ms

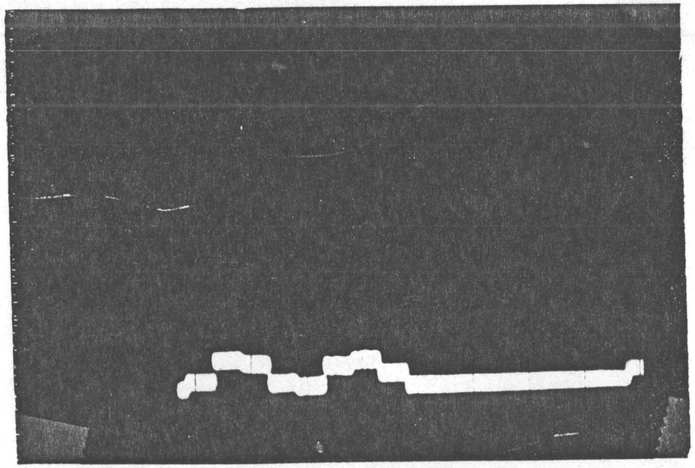


Fig. 5.29 f_{in} = 537.5 Hz
Scale: 0.5v/20ms

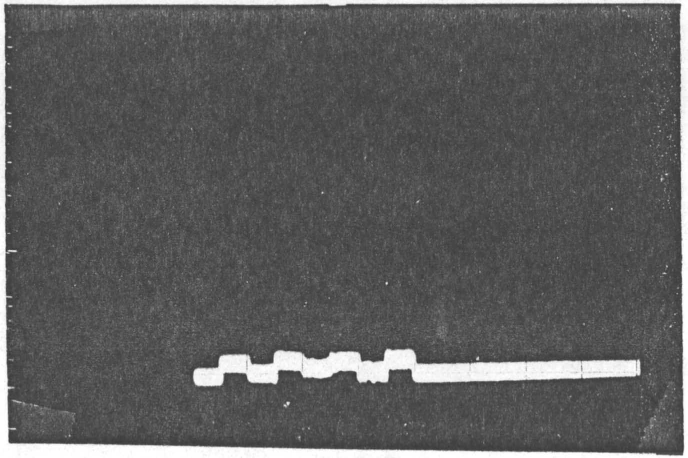


Fig. 5.30 f_{in} = 562.5 Hz
Scale: 0.5v/20ms

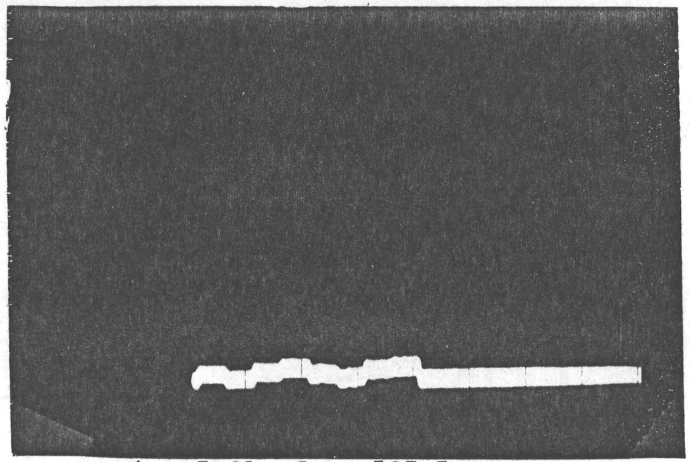


Fig. 5.31 f_{in} = 587.5 Hz
Scale: 0.5v/20ms

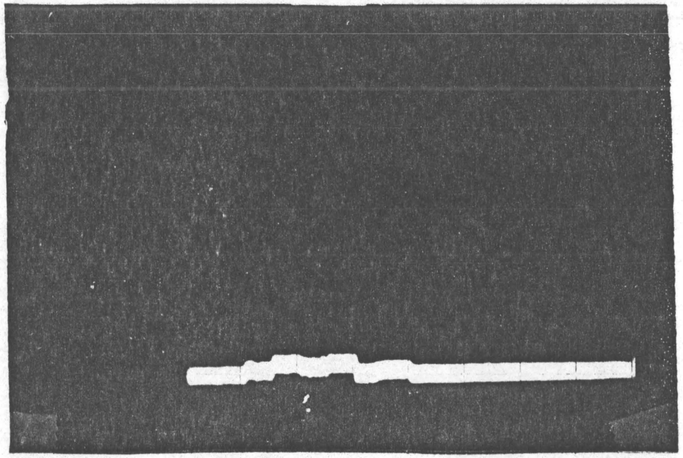


Fig. 5.32 f_{in}= 612.5 Hz
Scale: 0.5v/20ms

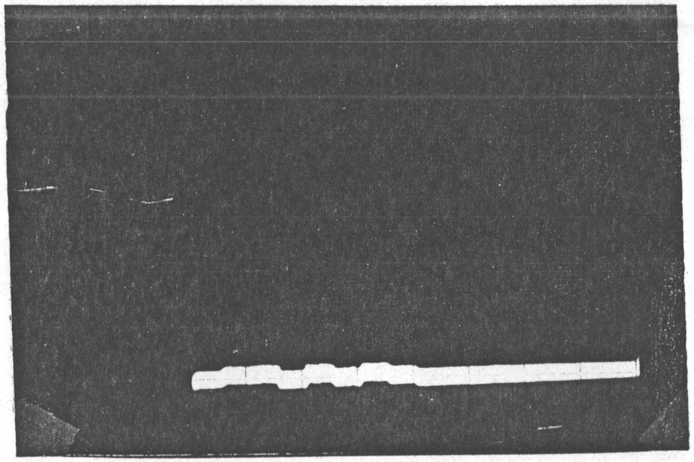


Fig. 5.33 f_{in}= 637.5 Hz
Scale: 0.5v/20ms

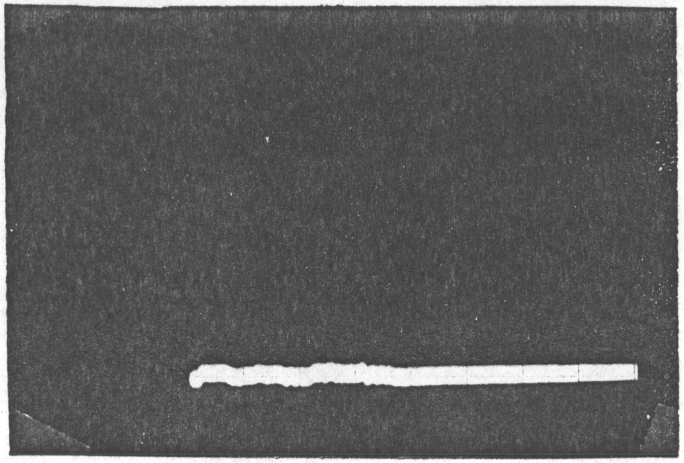


Fig. 5.34 f_{in}= 662.5 Hz
Scale: 0.5v/20ms

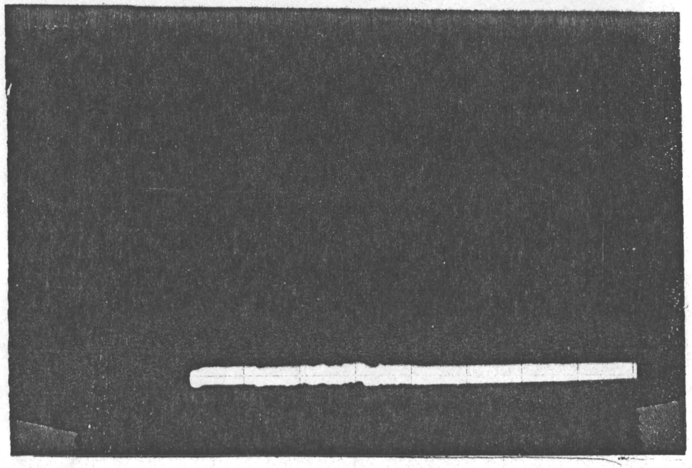


Fig. 5.35 f_{in}= 687.5 Hz
Scale: 0.5v/20ms

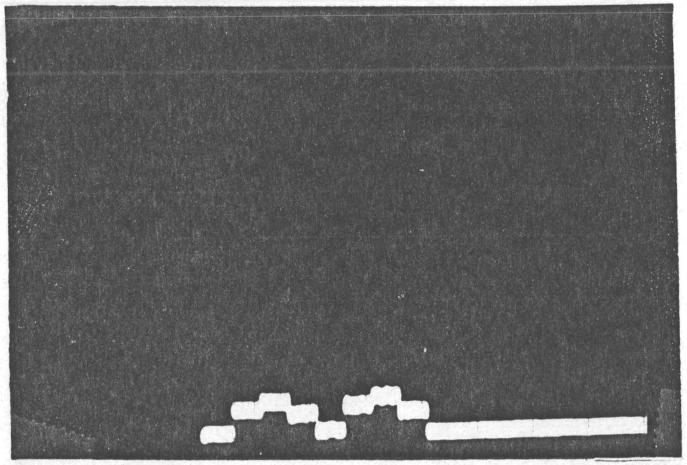


Fig. 5.36 f_{in} = 487.5 Hz
Scale: 0.5v/20ms

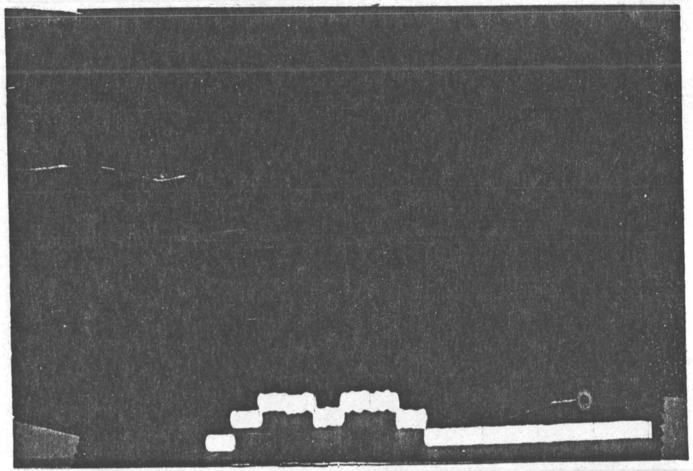


Fig. 5.37 f_{in} = 492.5 Hz
Scale: 0.5v/20ms

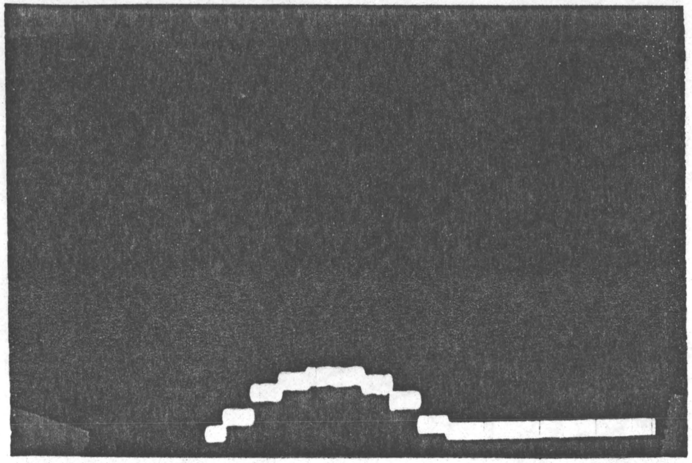


Fig. 5.38 f_{in} = 497.5 Hz
Scale: 0.5v/20ms

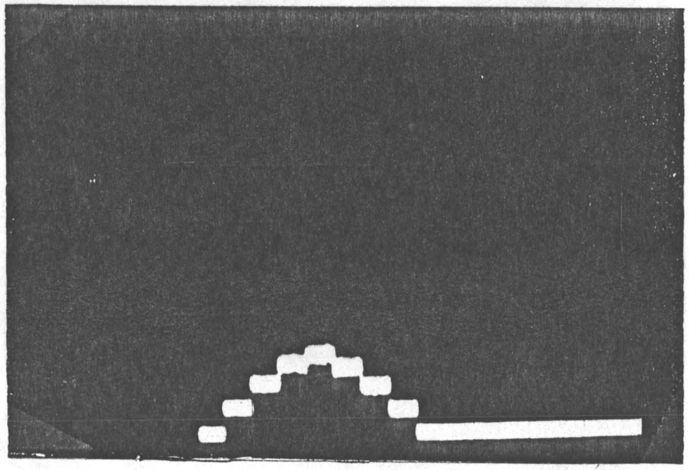


Fig. 5.39 f_{in} = 502.5 Hz
Scale: 0.5v/20ms

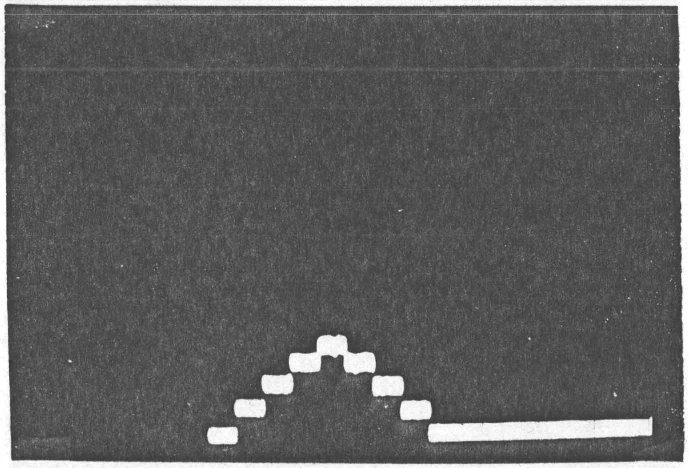


Fig. 5.40 f_{in} = 507.5 Hz
Scale: 0.5v/20ms

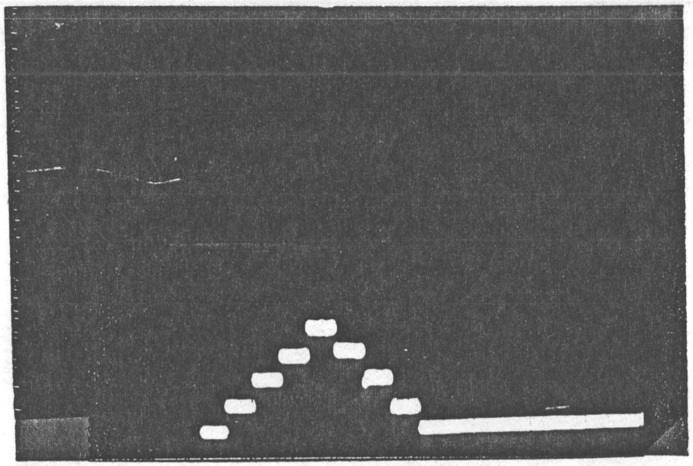


Fig. 5.41 f_{in} = 512.5 Hz
Scale: 0.5/20ms

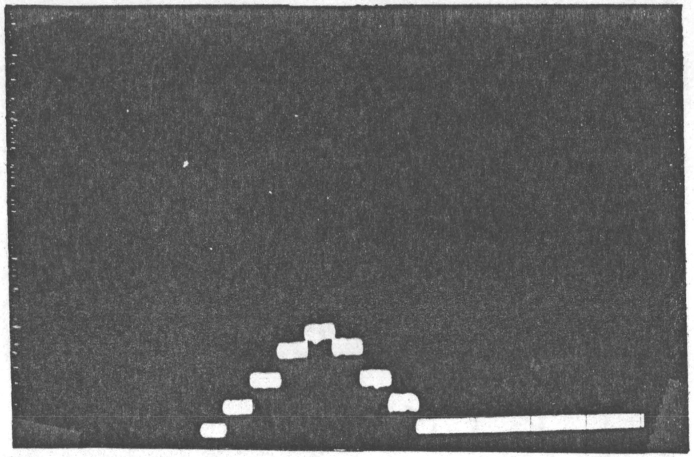


Fig. 5.42 f_{in} = 517.5 Hz
Scale: 0.5v/20ms

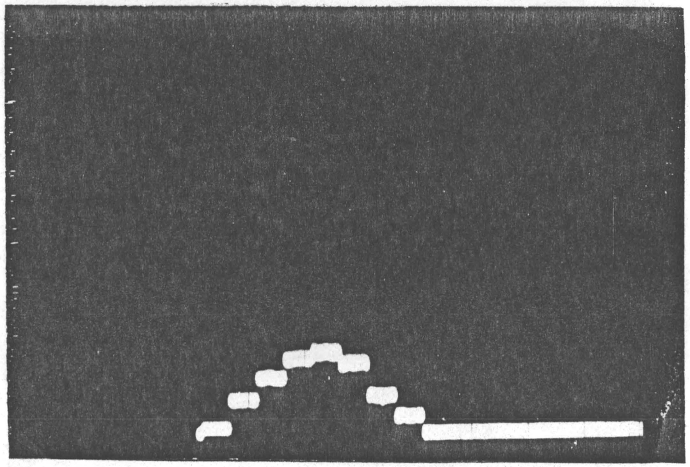


Fig. 5.43 f_{in} = 522.5 Hz
Scale: 0.5v/20ms

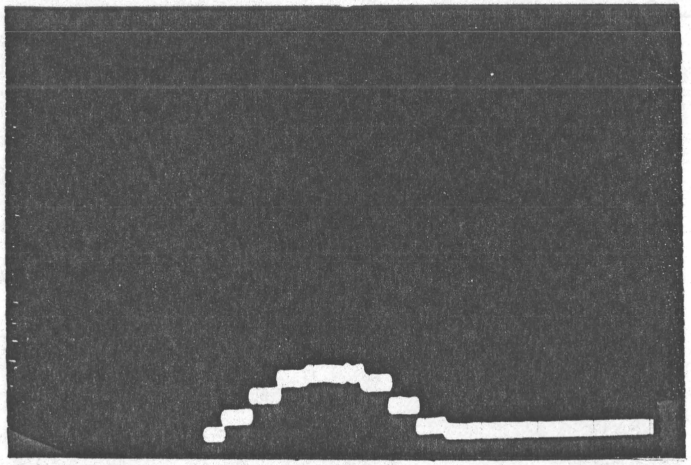


Fig. 5.44 $f_{in} = 527.5$ Hz
Scale: 0.5v/20ms

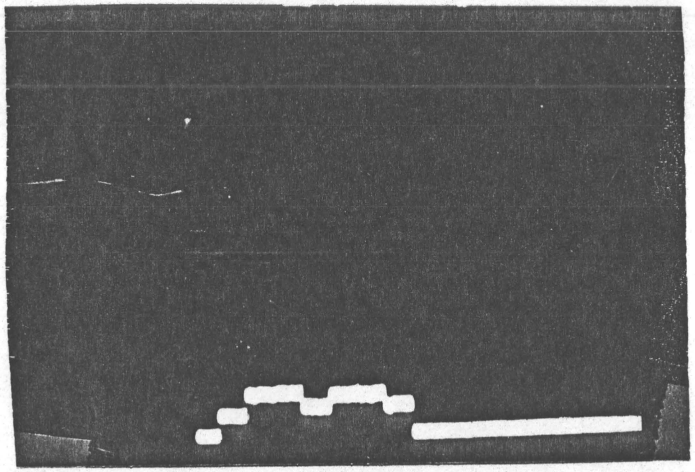


Fig. 5.45 $f_{in} = 532.5$ Hz
Scale: 0.5v/20ms

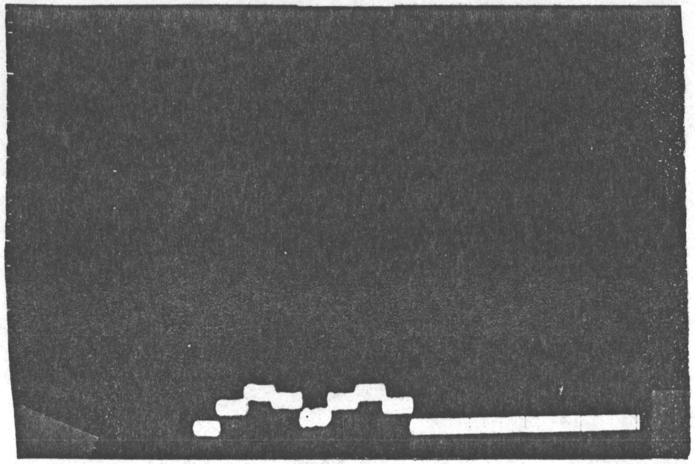


Fig. 5.46 $f_{in} = 537.5$ Hz
Scale: 0.5v/20ms

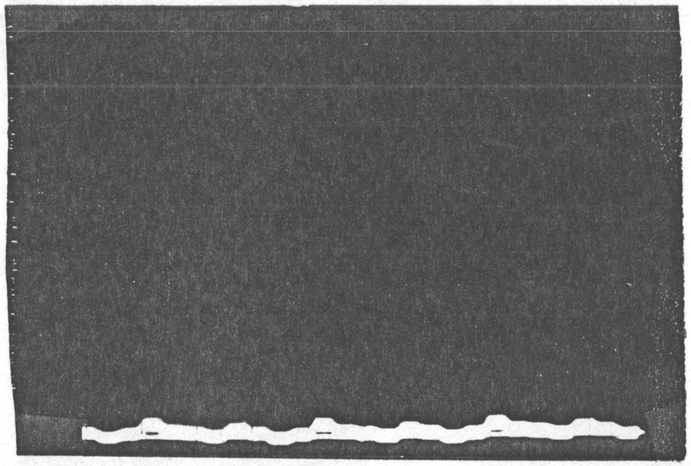


Fig. 5.47 f_{in} = 312.5 Hz
Scale: 0.5v/20ms

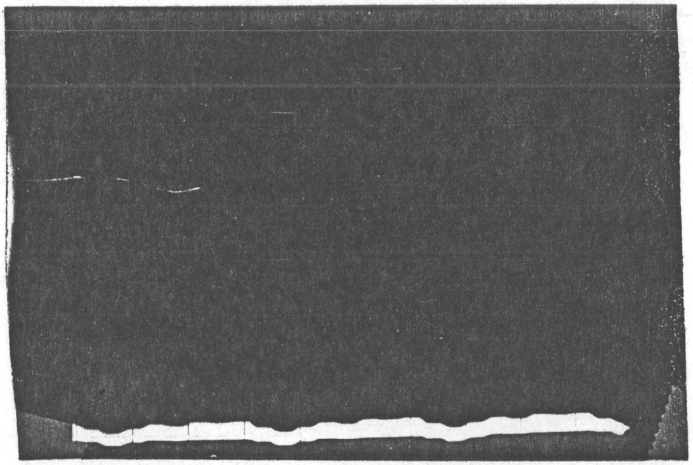


Fig. 5.48 f_{in} = 337.5 Hz
Scale: 0.5v/20ms

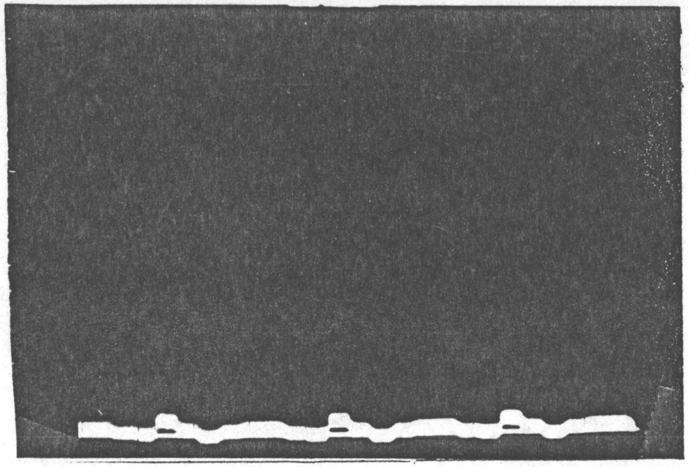


Fig. 5.49 f_{in} = 362.5 Hz
Scale: 0.5v/20ms

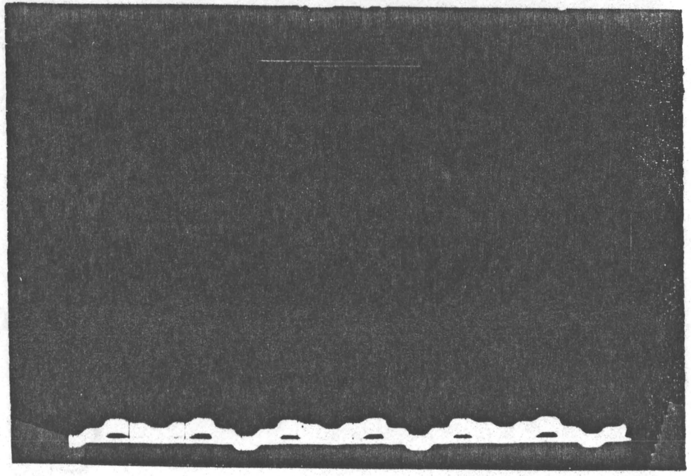


Fig. 5.50 f_{in} = 387.5 Hz
Scale: 0.5v/20ms

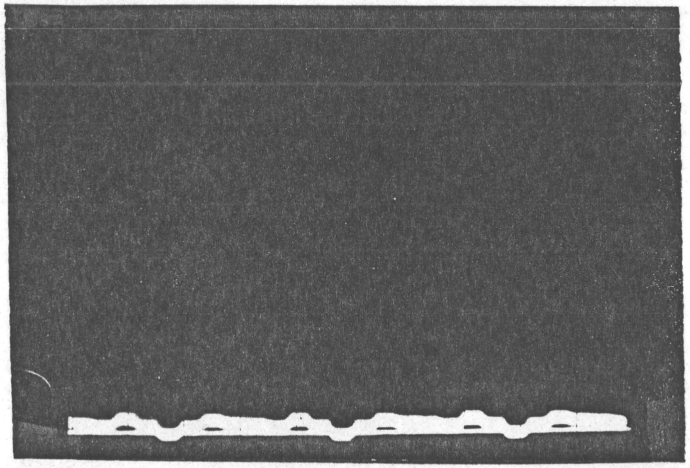


Fig. 5.51 f_{in} = 412.5 Hz
Scale: 0.5v/20ms

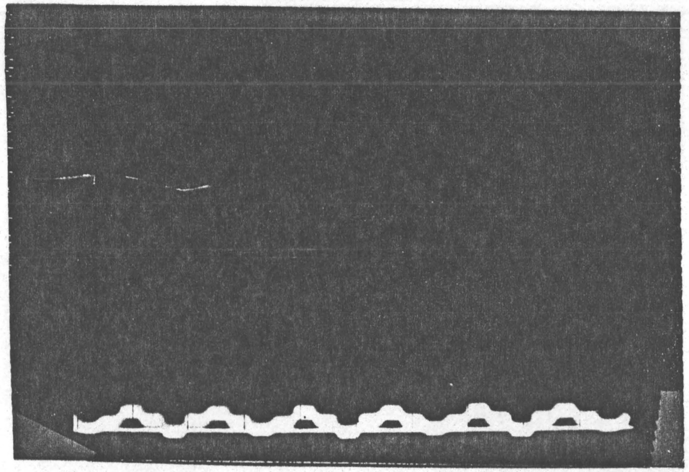


Fig. 5.52 f_{in} = 437.5 Hz
Scale: 0.5v/20ms

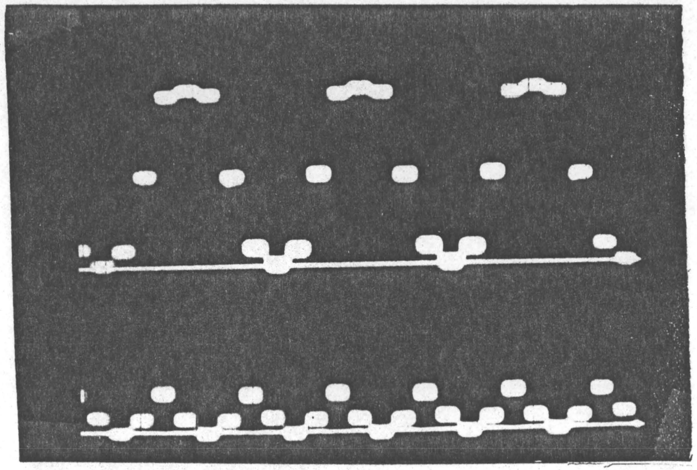


Fig. 5.53 Upper f_{in} = 512.5 Hz
Lower f_{in} = 462.5 Hz
Scale: 0.5v/20ms

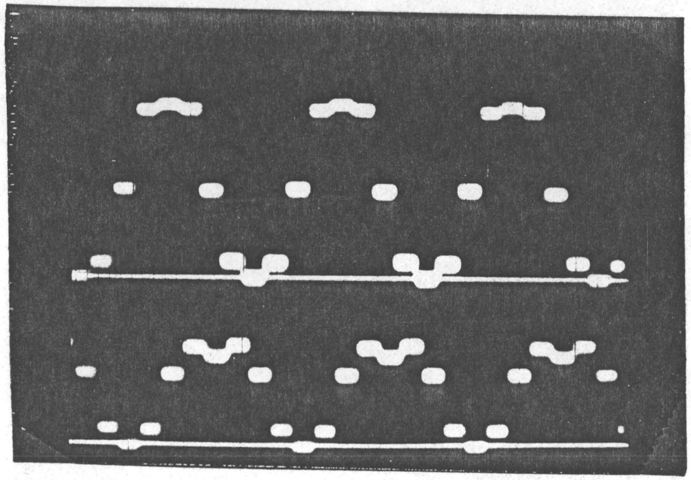


Fig. 5.54 Upper f_{in} = 512.5 Hz
Lower f_{in} = 487.5 Hz
Scale: 0.5v/20ms

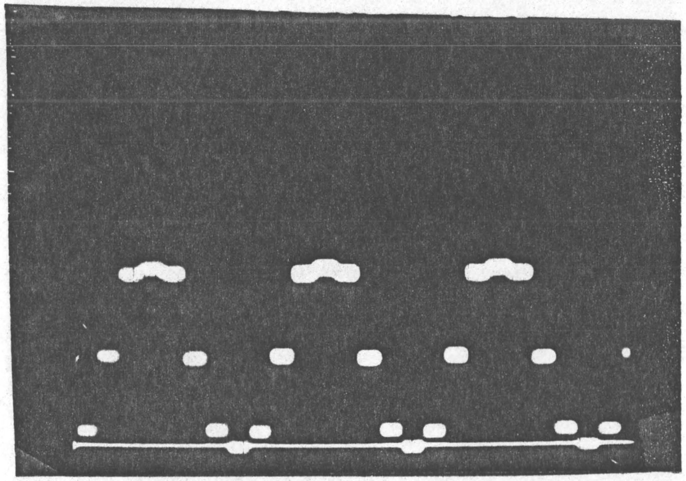


Fig. 5.55 f_{in} = 512.5 Hz
Scale: 0.5v/20ms

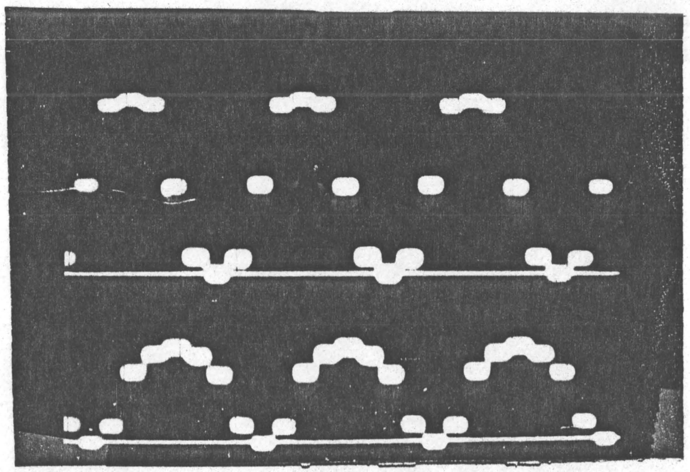


Fig. 5.56 Upper f_{in} = 512.5 Hz
Lower f_{in} = 537.5 Hz
Scale: 0.5/20ms

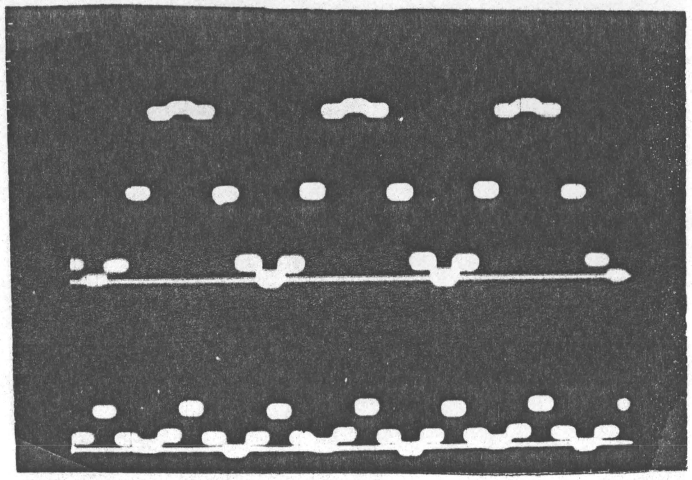


Fig. 5.57 Upper f_{in} = 512.5 Hz
Lower f_{in} = 562.5 Hz

Scale: 0.5v/20ms

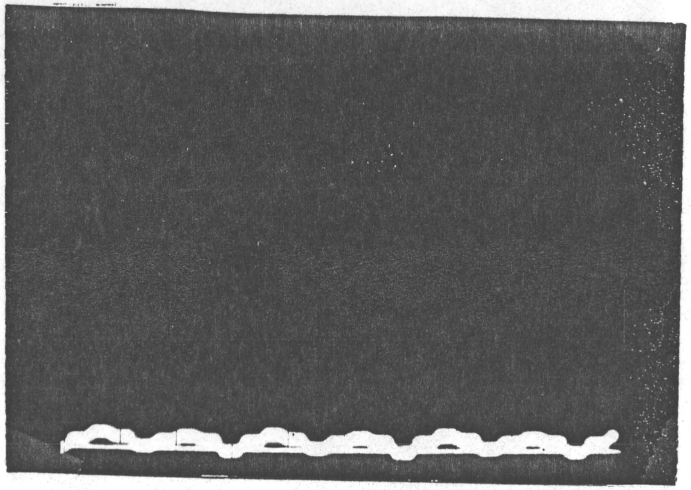


Fig. 5.58 f_{in} = 587.5 Hz

Scale: 0.5/20ms

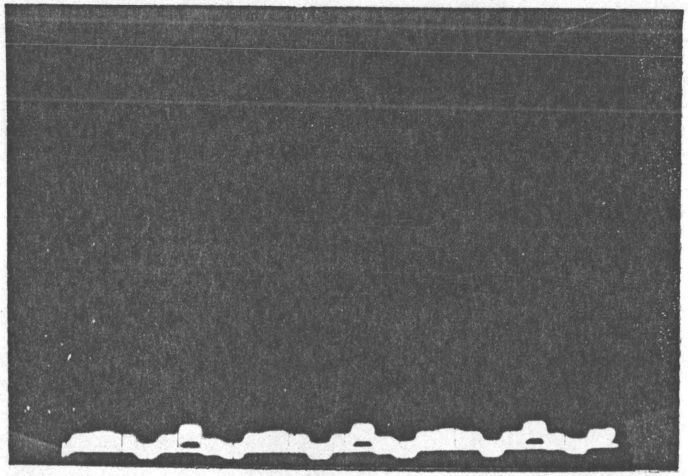


Fig. 5.59 f_{in} = 612.5 Hz
Scale: 0.5v/20ms

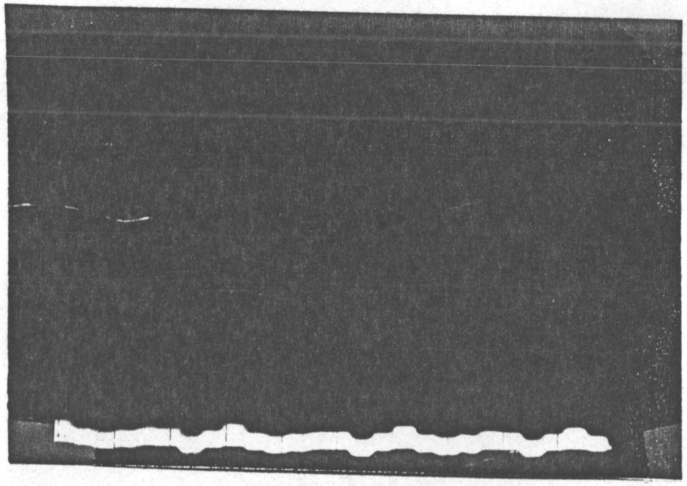


Fig. 5.60 f_{in} = 637.5 Hz
Scale: 0.5v/20ms

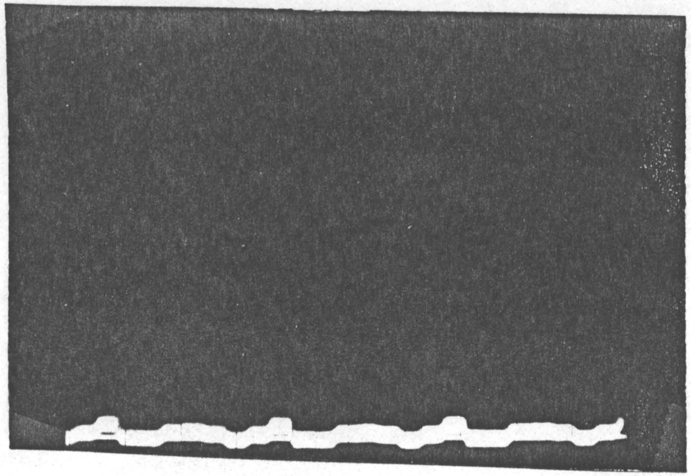


Fig. 5.61 f_{in} = 662.5 Hz
Scale: 0.5v/20ms

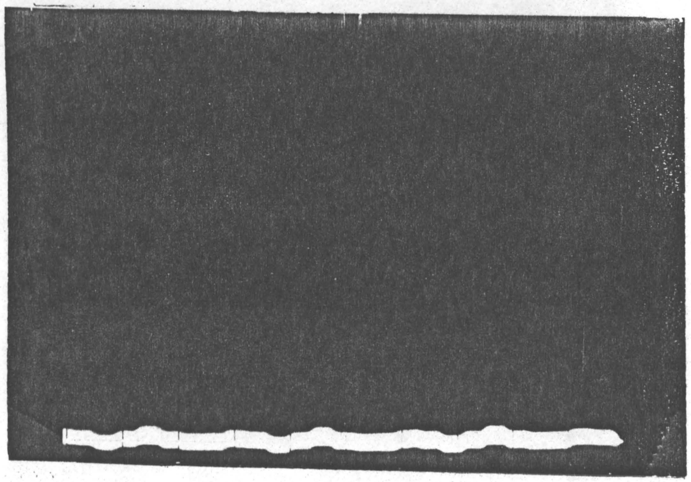


Fig. 5.62 f_{in} = 687.5 Hz
Scale: 0.5v/20ms

5.2 OUTPUT SIGNAL-TO-NOISE RATIO MEASUREMENT

A block diagram of the setup for the signal-to-noise ratio measurement is shown in Fig. 5.63. For this measurement the input pulse is adjusted to be 40 ms in duration and frequency is adjusted to the frequency of replica. The noise is filtered through a bandpass filter with lower and upper cutoff at 300 Hz and 700 Hz, respectively.

During the measurement, the SNR of the input signal is kept at 0 dB. Figure 5.64 shows a 40 ms duration CW pulse at 512.5 Hz embedded in noise and Fig. 5.65 is the correlator output when the signal is correlated with 512.5 Hz replica frequency. In this case, it appears that the ratio between the average voltage output due to the noise and the peak voltage output caused by the pulsed signal is approximately 1/4. This corresponds to a 12 dB signal-to-noise ratio gain. Since all the 16 replica channels have the identical output response characteristics, one signal-to-noise ratio measurement is suffice to conclude that the output signal-to-noise ratio gain of the correlator is approximately 12 dB. Finally, this results agrees well with the predicted signal-to-noise ratio improvement of 16:1 (12 dB) due to bandwidth reduction.

5.3 EFFECTS OF COHERENT EXTENDED TARGET RETURNS

As mentioned at the beginning of this chapter, due to the time stretching and multipath effect of propagating sound signals

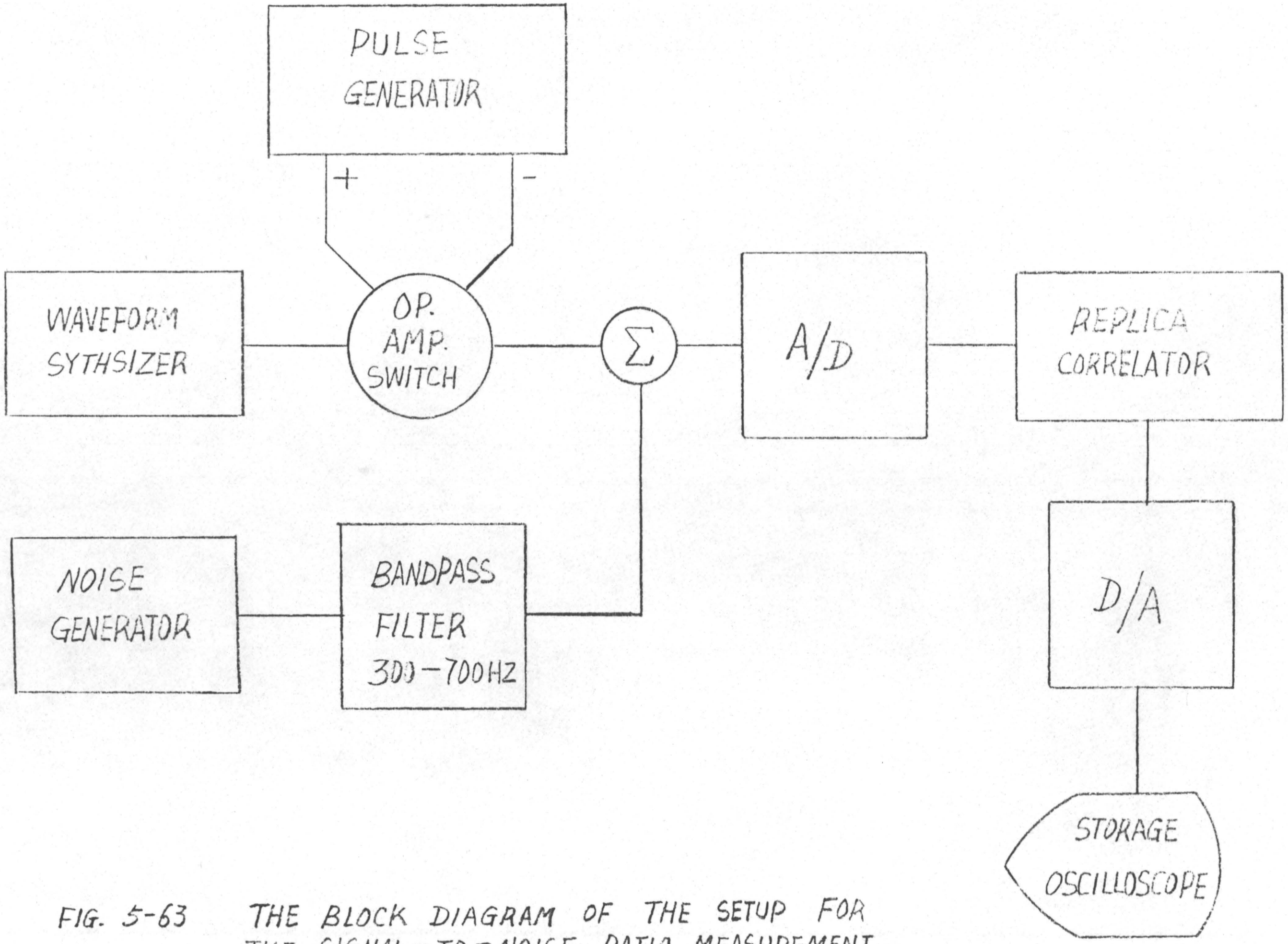


FIG. 5-63 THE BLOCK DIAGRAM OF THE SETUP FOR THE SIGNAL-TO-NOISE RATIO MEASUREMENT.

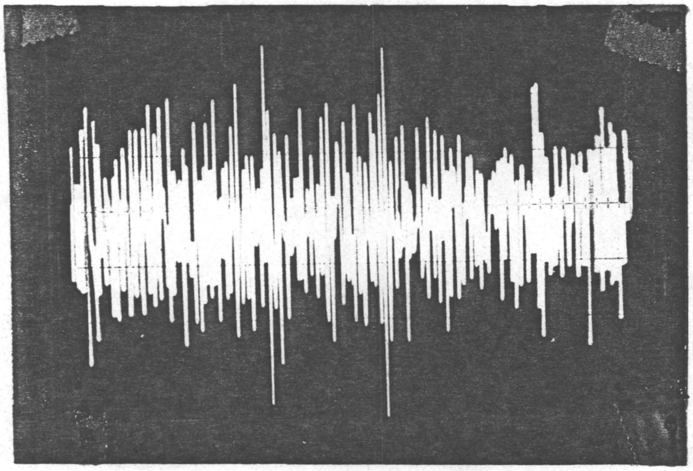


Fig. 5.64 A picture of 40ms pulsed CW signal at 512.5 Hz embedded in noise

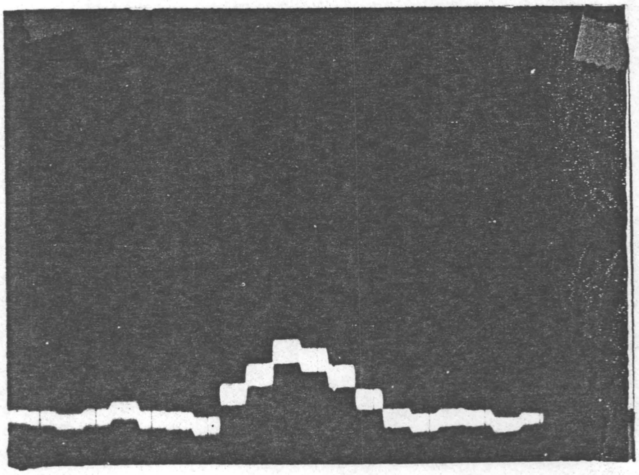


Fig. 5.65 Correlator output with the signal in Fig. 5.64 as input.

in the sea, and echoes from an extended target, target echoes are usually smeared and extended in time. The effects of extended coherent and noncoherent target returns on the output of the correlator have been investigated. The setup for the extended coherent target echoes tests is the same as the one shown in Fig. 5.1, except the duration of pulsed CW signals have been adjusted to values of 40, 50, 60, and 70 ms to simulate the extended signals. These extended signals with frequency 512.5 Hz have been fed to the correlator with the corresponding 512.5 Hz replica being used. The resulting outputs are shown from Fig. 5.66 to Fig. 5.69. It has been found that the output increases to the maximum, and then stays constant for an interval of 10, 20, 30 or even 40 ms before decreasing to minimum. The outcome of these observations is as expected, because, as the extended signal at the input approaches the replica pulse in Fig. 3.9, the input signal starts to overlap with the replica signal. As the overlapping increases, the correlation value also increases. A maximum correlator output results when the replica signal is completely overlapped by the elongated input pulse. The output value will not start to decrease until the overlapping decreases as the input pulse passes the replica pulse. Hence, the duration at which the output dwells on the maximum level is dependent on the length of the pusle at the input of the correlator.

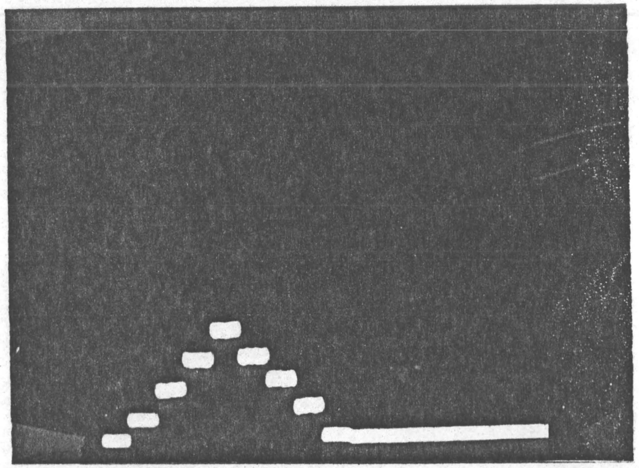


Fig. 5.66 f_{in}= 512.5 Hz

Duration : 40ms

Scale: 0.5v/20ms

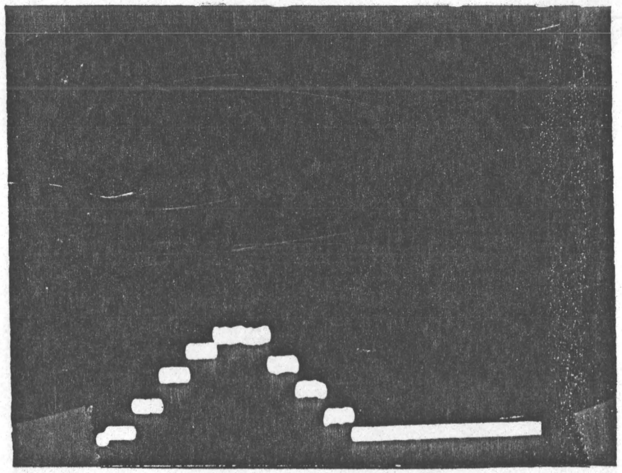


Fig. 5.67 f_{in}= 512.5 Hz

Duration : 50ms

Scale: 0.5v/20ms

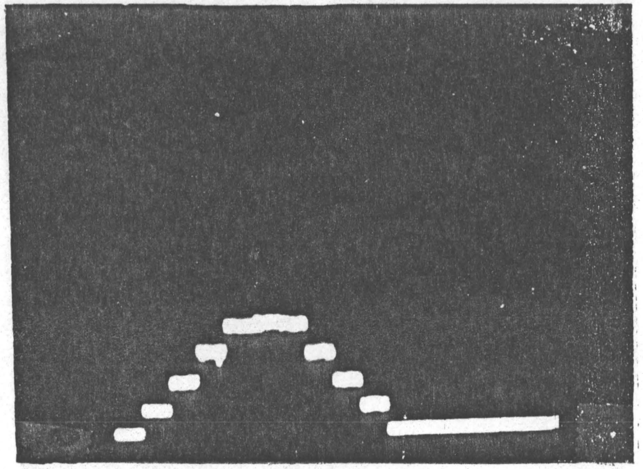


Fig. 5. 68 f_{in}= 512.5 Hz

Duration : 60ms

Scale: 0.5v/20m

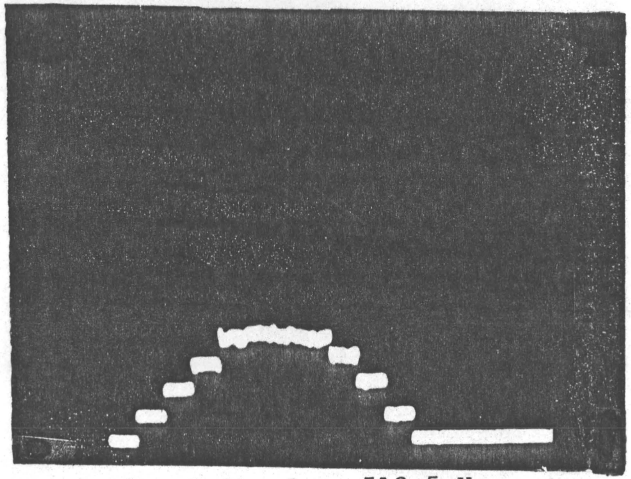


Fig. 5.69 f_{in}= 512.5 Hz

Duration : 70ms

Scale: 0.5v/20ms

5.4 EFFECTS OF NON-COHERENT EXTENDED TARGET RETURNS

Although extended coherent returns can occur in the real operating environment, non-coherent returns are more likely. For the case of non-coherent returns, the signal at the input of a sonar system is the result of the superpositioning of various echoes at random phases induced by the multipath effects or extended target due to a single transmitted pulse. To simulate extended non-coherent returns, a second switch in Fig. 5.70 has been incorporated into the switch in Fig. 5.1.

The second switch has variable delay with respect to first switch shown in Fig. 5.3. Two retriggerable monostable multivibrator (74123) have been used to achieve this. Having been triggered by the rising edge of a positive pulse from an HP 8008A pulse generator, the falling edge of the output Q_1 of the first monostable multivibrator is used to trigger the second monostable multivibrator whose output Q_2 is a 40 ms pulse. The duration for which the Q_1 is at logic level HIGH may be varied through a variable resister which connects power supply +5 V to the R/C input of the monostable multivibrator. By varying the resistor value, the Q_1 pulse can be made to extend or receed, therefore, the time at which the Q_2 pulse appears with respect to the pulse triggering the first monostable multivibrator can be varied. Using the Q_2 pulse and a second signal generator with two voltage followers and two inverting amplifiers connected in the manner shown in Fig. 5.70, a second pulsed CW signal with variable time delay with respect to the first pulsed CW signal is obtained.

When two signals of same frequency are superposed depending on the phases of the signals, the resultant signal power may be strengthened or weakened. The envelopes of the resultant signal from the superpostion of two equal amplitude, 40 ms, pulsed CW signals at 512.5 Hz with the second signal delayed by 10 ms with respect to the first are shown in Fig. 5.71. The envelopes shown on the left and right hand side in Fig. 5.71 have not been expected. These deviations are due to the instability of one of the two signal generators used to produce the pulsed signals. However, these unexpected signals may be interpreted in a hypothetical situation as the echoes returned from two targets in the same resolution cell both travelling at a constant but slightly different speeds. This can be easily proven as follows: The echo signals returned from two targets travelling at constant speeds V_1 and V_2 contain Doppler shifted carrier frequencies at f_1 and f_2, respecitvely. If Δf and f_o are defined as $(f_1-f_2)/2$ and $(f_1+f_2)/2$, respectively. Then the superpostion of these two echo signals can be expressed as

$$\begin{aligned} a(t) &= A \cos 2\pi f_1 t + A \cos 2\pi f_2 t \\ &= A \cos 2\pi(f_o+\Delta f)t + A \cos 2\pi(f_o-\Delta f)t \\ &= A \cos(2\pi f_o t)\cos(2\pi\Delta ft)-A \sin(2\pi f_o t)/\sin(2\pi\Delta ft) + \\ &\quad A \cos(2\pi f_o t)\cos(2\pi\Delta ft) + A \sin(2\pi f_o t)/\sin(2\pi\Delta ft) \\ &= 2A \cos(2\pi f_o t) \cos(2\pi\Delta ft) \qquad (5.2) \end{aligned}$$

The result of this superpostion, according to Eqn. (5.2), is a modulating effect, and is shown in Fig. 5.72. The resultant envelope shapes in Fig. 5.71 can be explained by Fig. 5.73. When two equal ampitude and pulse duration signals are used, and the second signal is delayed by T seconds with respect to the first signal, then, the positions of the two signals with respect to each other is as shown in Fig. 5.73. As can be seen there are portions of the two signals which overlaps with each other, and the modulation of the kind shown in Fig. 5.72 occurs. For the portions that are not overlapped, thus not modulated, stay as they were before the superpostion. Therefore, the resultant signal is a extended signal with the middle portion modulated by the expression in Eqn. (5.2). When Δf is small, a small segment of the modulating signal $\cos(2\pi\Delta ft)$ would appear as a straight line; this is the reason why the modulated portions of the envelopes shown in Fig. 5.71 do not appear as a part of a sine wave. Fig. 5.74 to Fig. 5.77 illustrate some of the correlation outputs as results of various extended non-coherent echo inputs.

Although some of these outputs seem to be odd-shaped, they do, in general, conform with predictions. For example, the output shown in the middle of Fig. 5.76 is caused by an input signal having an envelope of the shape shown on the left side of Fig. 5.71.

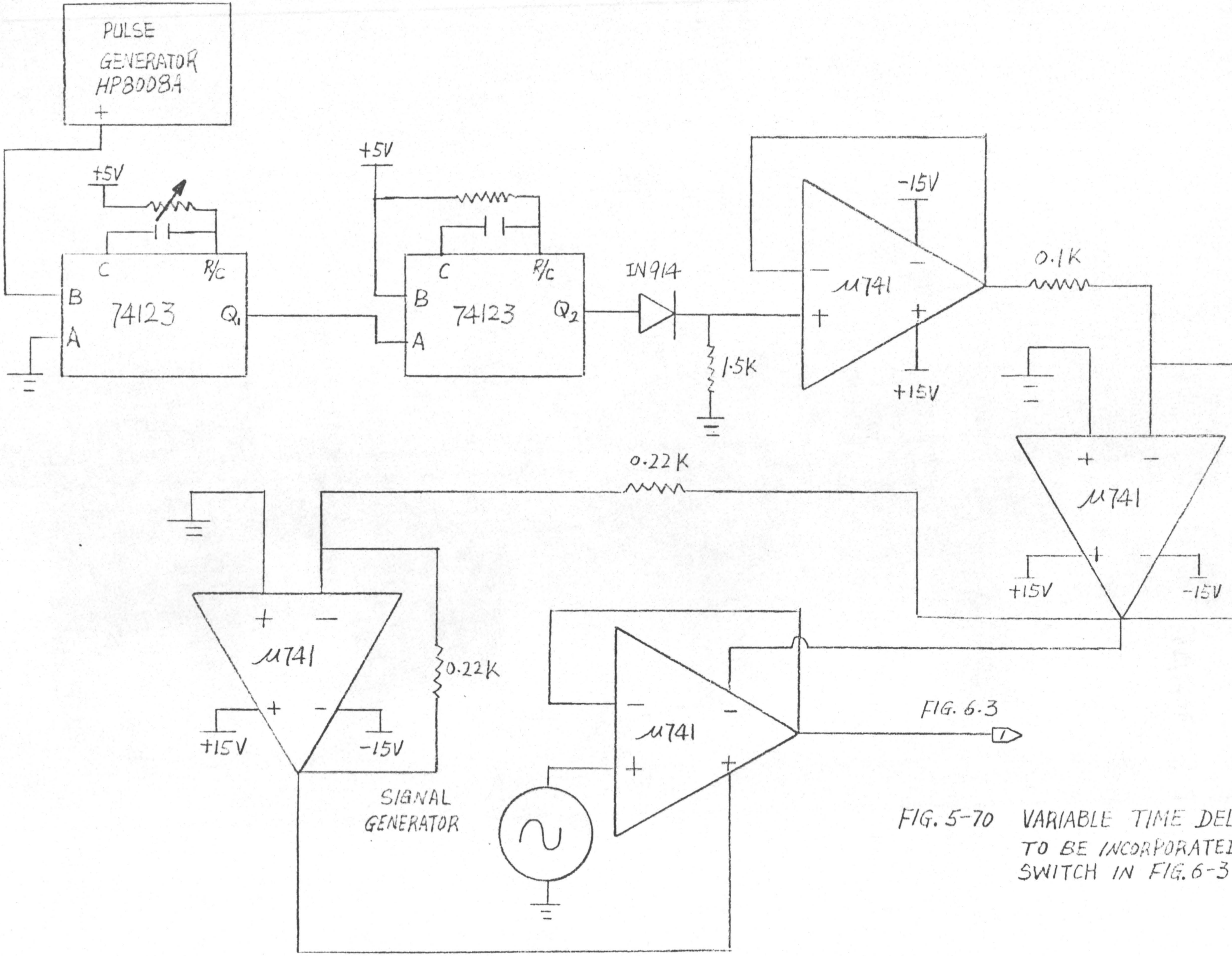


FIG. 5-70 VARIABLE TIME DELAY SWITCH TO BE INCORPORATED INTO THE SWITCH IN FIG. 6-3.

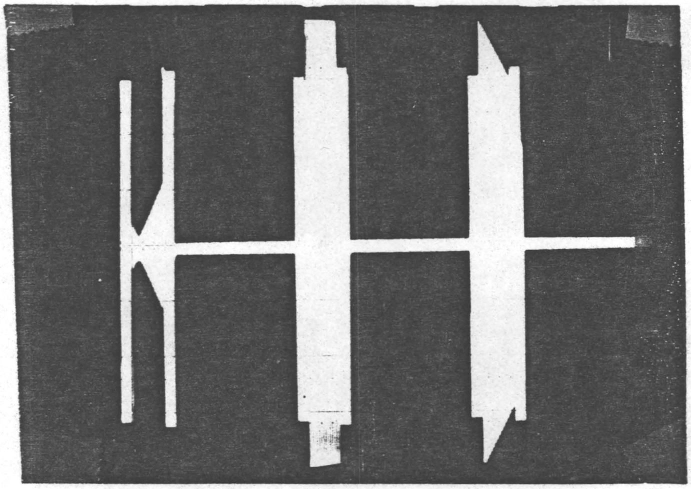


Fig. 5.71 Envelopes of some of the non-coherent extended target returns.

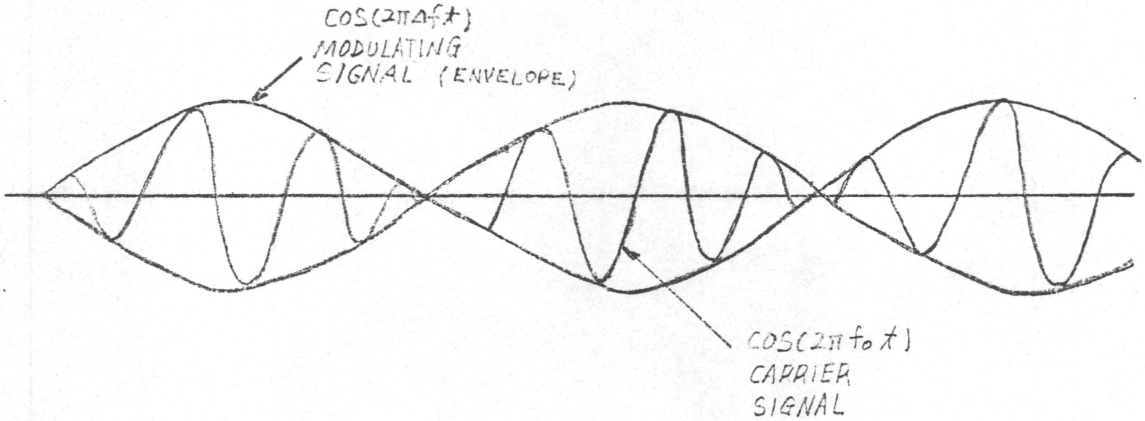


FIG. 5-72 A SIGNAL RESULT FROM THE SUPERPOSITION OF TWO SIGNALS WITH SLIGHTLY DIFFERENT FREQUENCIES.

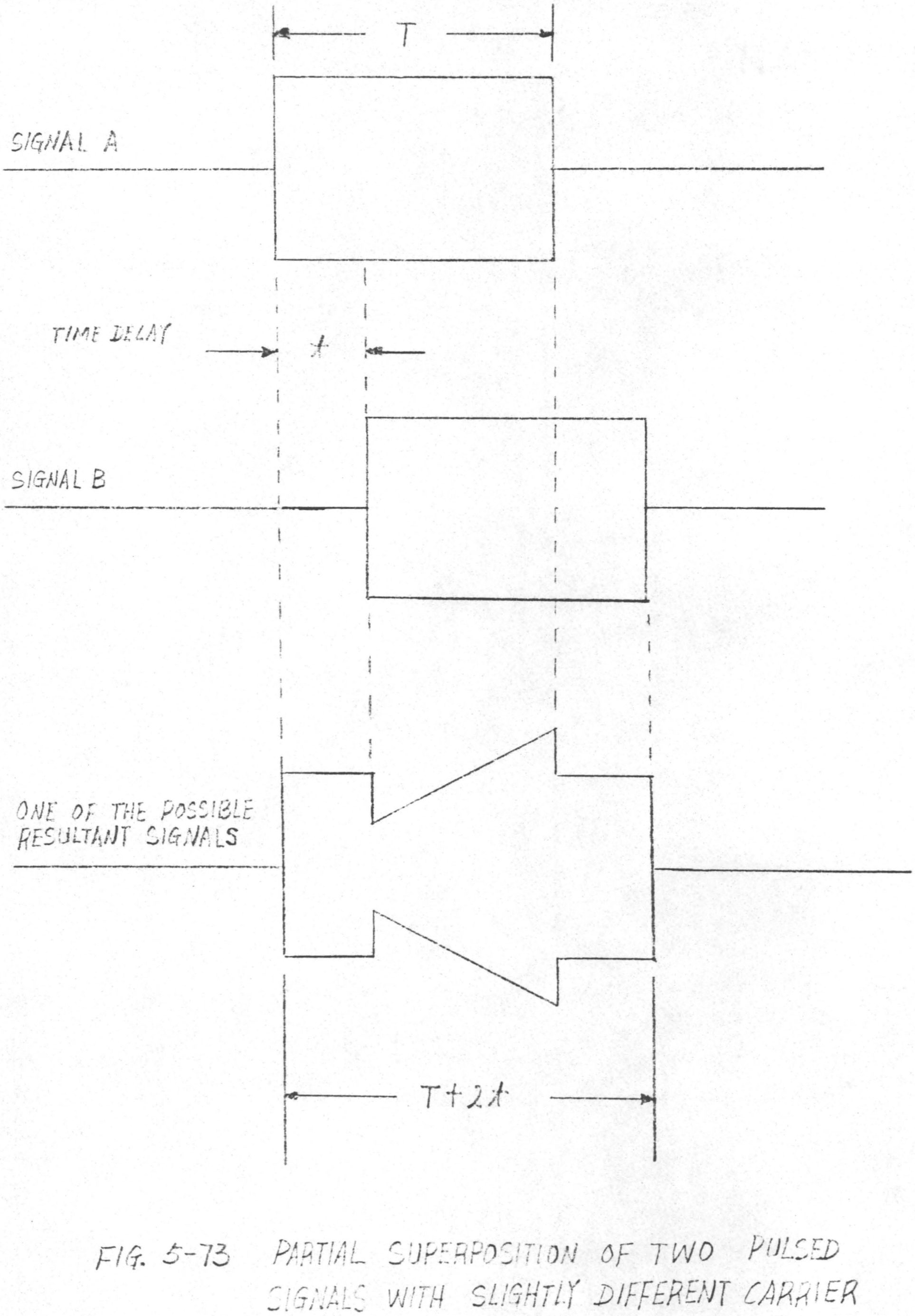


FIG. 5-73 PARTIAL SUPERPOSITION OF TWO PULSED SIGNALS WITH SLIGHTLY DIFFERENT CARRIER FREQUENCIES

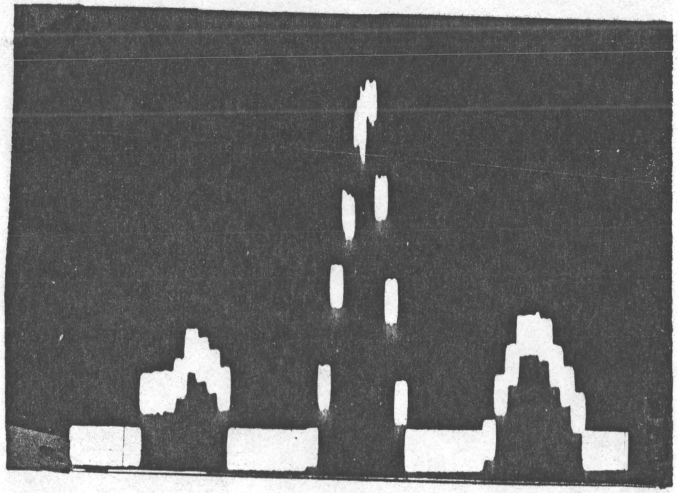


Fig. 5.74 Input signal duration ; 50ms

Scale: 0.5v/0.1sec

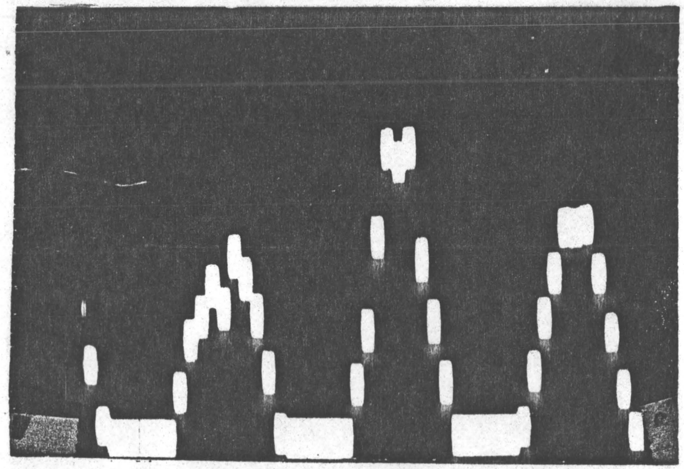


Fig. 5.75 Input signal duration : 50ms

Scale: 0.5v/0.1sec

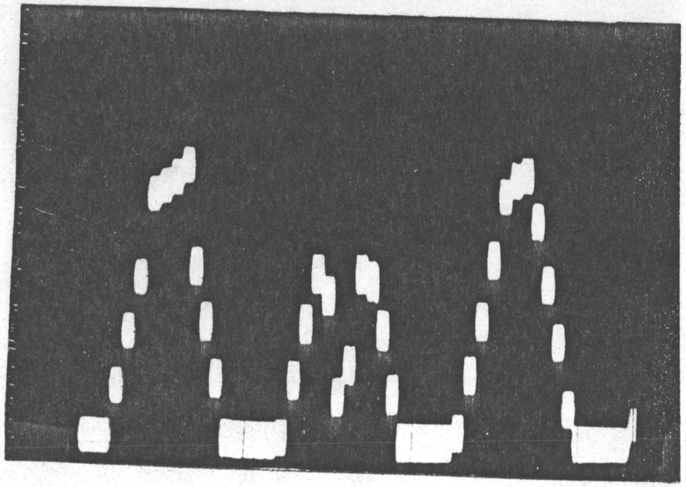


Fig. 5.76 Input signal duration : 60ms

Scale: 0.5v/0.1sec

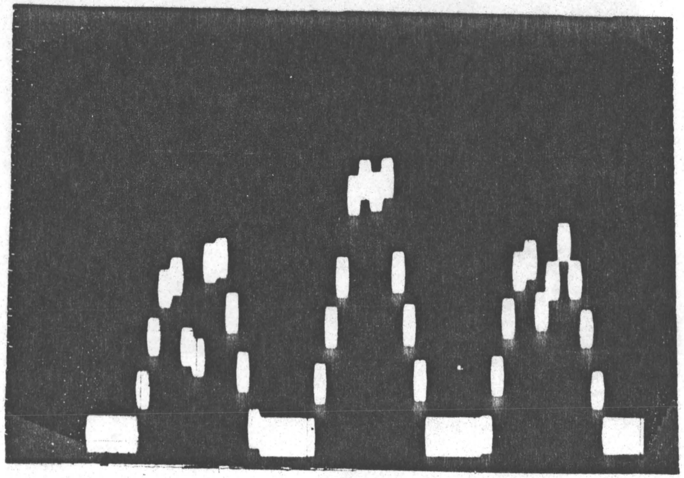


Fig. 5.77 Input signal duration : 60ms

Scale: 0.5/0.1sec

CHAPTER 6

CONCLUSIONS AND RECOMMENDATIONS

6.1 CONCLUSIONS

A. The processing of sonar signals using digital techniques has been examined and three different methods of filtering have been identified. It was concluded that for the case when the input samples are of a bandpass nature, the replica correlator offers advantages over the FFT and the digital filter.

B. The bit slice microprocessor is well suited for performing replica correlation calculations for real-time processing of sonar signals due to its high speed of operation. Furthermore, it has been shown that the system design can be considerably simplified by incorporating most of the control signals into the microprogram.

C. By storing the replica signals in ROM units, the processor can detect signals in different frequency bands by simply changing the ROM units. Thus, interfrequency spacing can be increased or decreased as desired.

D. The crosscoupling, signal-to-noise ratio, and the effects of coherent and non-coherent extended target returns have been measured and investigated. The results show good agreement between theory and measurement for laboratory simulated signals.

E. Finally, by taking advantage of the static operation property of the bit slice microprocessor, a simple scheme for extending the present single channel processor into a multichannel processor has been proposed for possible future expansion.

6.2 RECOMMENDATIONS

A. Investigate the performance of the signal processor in the presence of reverberation resulting from a CW pulse.

B. Investigate the use of mis-matched replicas to improve the performance of the signal processor in the presence of reverberation.

C. Investigate the use of other practical transmitter signals for improving the detection capability in the presence of reverberation.

REFERENCES

[1] Sarine, C.O.; Microelectronics in the PADLOC Systems. Sperry Engineering Review, (U.S.A.) 19:10-16 (1966).

[2] Haykin, S.S., Carter, C.R.; Multichannel Digital Signal Processor for Sonar Signals. Communications Research Laboratory internal report series No. CRL-44, McMaster University, Hamilton, Ontario, Canada.

[3] Mackenzie, K.V.; Formulas for the Computation of Sound Speed in Sea Water. J. Acoust. Soc. Am., 32:100 (1960).

[4] Leroy, C.C.; Development of Simple Equations for Accurate and Realistic Calculation of the Speed of Sound in Sea Water. J. Acoust. Soc. Am. 46:216 (1969).

[5] Leonard, R.W., Combs, P.C., and Skidmore, L.R.; Attenuation of Sound in Synthetic Sea Water, J. Acoust. Soc. Am., 21:63 (1949).

[6] Yeagor, E., Fisher, F.H., Micell, J., and Bressel, R.; Origin of the Low-Frequency Sound Absorption in Sea Water, J. Acoust. Soc. Am., 53:1750 (1973).

[7] Liebermann, L.N.; Origin of Sound Absorption in Water and in Sea Water, J. Acoust. Soc. Am., 20:868 (1948).

[8] Urick, R.J.; "Principles of Underwater Sound"; Chap. 5.9, McGraw-Hill Inc., 1975.

[9] Rayleigh, Lord; "The Theory of Sound." Vol. I, pp. 35-42, Dover Publications Inc., New York, 1945.

[10] Wenz, G.M.; Acoustic Ambient Noise in the Ocean: Spectra and Sources, J. Acoust. Soc. Am., 34:1936 (1962).

[11] Haykin, S.S.; "Signal Processing and Modulation System." Vol. I, Chap. 12, McMaster University, Hamilton, Ontario, Canada.

[12] Oppenheim, A.V.; "Digital Signal Processing." Chap. 4, Prentice-Hall Inc., Englewood Cliffs, New Jersey, 1975.

[13] "Intel Series 3000 Reference Manual." Intel Corporation 1975.

[14] Del Grosso, V.A.; Velocity of Sound in Sea Water at Zero Depth, U.S. Nov. Res. Lab. Rep. 4002, 1952.

[15] Handbook of Oceangraphic Tables, U.S. Nov. Oceanogr. Office Spec. Publ. 68 (1966).

[16] An Interim Report on the Sound Velocity Distribution in the North Atlantic Ocean, U.S. Navy Oceanogr. Office Tech. Rep. 171, 1965.

[17] "Ocean Science Program of the U.S. Navy, "Office of the Oceanographer of the Navy, Alexandria, Va., 1970.

[18] Ewing, M., and J.L. Worzel; Long-Range Sound Transmission, Geol. Soc. Am. Memo 27, 1948.

[19] Vent, R.J.; Acoustic Volume Scattering Measurements at 3.5, 5.0 and 12.0 KHz in the Eastern Pacific Ocean, J. Acoust. Soc. Am., 52:373 (1972).

[20] Allen, W.B., and Westerfield, E.C.; Digital Compressed-time correlators and matched filters for Active Sonar. J. Acoust. Soc. Am., 36:121-139 (January 1964).

[21] Man, J., and Carter, C.R.; Processing Sonar Signals by Chip Slice Microcomputer, Proc. 21st Midwest Symposium on Circuits and Systems, Ames, IA, Aug. 1978, pp. 531-535.

[22] Man, J., and Carter, C.R.; A Microcomputer for Processing Sonar Signals, submitted to CEC'78.

APPENDIX 1

APPROXIMATION OF $(X^2+Y^2)^{1/2}$

Calculation of exact value of $(x^2+y^2)^{1/2}$ using 2's complement arithmetic is both difficult and time consuming. Therefore, an approximation of the expression is employed rather than the exact solution.

The approximation used for the expression is (31x/32)+(3y/8) with x always bigger than y. This means that a test must be made to determine the relative magnitude of the two values, x and y, and assign the bigger one as x in the approximation algorithm. The maximum error incurred as the result of this approximation is at the instance when x is equal to y in the approximation algorithm. This translates to a maximum error equal to

$$\frac{\sqrt{2} - (\frac{31}{32} + \frac{3}{8})}{\sqrt{2}} \times 100\% = 4.98\%$$

This is equivalent to an error of 0.250 for a 5 V full scale output.

APPENDIX 2

DOCUMENTED LISTINGS OF COMPUTER PROGRAMS

This appendix contains documented listings of the computer programs used in simulating the microprogram, used by the digital correlator on a PDP8/A minicomputer, a Fortran program for generating the macroprogram, and microprogram used to run the digital correlator.

Simulated microprogram (p. 164) is the program written in PDP8/A assembly language for simulating the microprogram to be used in the bit slice microprocessor.

The Fortran program (p. 174) is used to generate 16 five-level quantized Doppler shifted versions of a pulsed CW signal. The ouptut of this program is a sequence of 5 different octal coded words which is subsequently used as the macroprogram of the bit slice microprocessor.

Microprogram (p. 178) is the program which controls the operation of the CPA and supplies the address control information to the MCU. The locations of each microinstruction in a 16x8 matrix are shown in p. 181 and the micro-codes for the microinstruction are listed at p. 182. Finally a list of representative micro-function performed by the CPE arithmetic logic section is provided on p. 186.

```
/MICROCOMPUTER PROGRAM SIMULATIO   PAL8-V9B   03/29/78   PAGE 1

                           /MICROCOMPUTER PROGRAM SIMULATION/

                    /THE MAIN PROGRAM SIMULATES THE MICROPROGRAM USED IN THE INTEL 3000
                    /MICROCOMPUTER.  THIS PROGRAM PERFORMS THE QUADRATURE AND INPHASE
                    /CROSS CORRELATION OF THE STORED REPLICAS AND SAMPLED SIGNALS ON
                    /SIXTEEN DIRRERENT DOPPLER FREQUENCY BANDS.

              0200  *200
      00200   7300  MICRO,   CLA CLL
      00201   4777' READ,    JMS WAIT           /WAITE COMMAND TO READ IN DATA
      00202   4776'          JMS DATAIN         /READ IN DATA
      00203   3521           DCA I R8           /STORE THE DATA IN MEMORY
      00204   2121           ISZ R8             /IS THIS THE 128TH INPUT DATA?
      00205   1121           TAD R8
      00206   1100           TAD K7000
      00207   1113           TAD K156
      00210   7510           SPA
      00211   5201           JMP READ           /NO, GO READ MORE DATA.
      00212   7200           CLA
      00213   2125  YES1,    ISZ R5             /YES, READ IN THE REPLICA COEFFICIENT
      00214   1526           TAD I R6
      00215   2126           ISZ R6
      00216   3120           DCA RA1
      00217   4776'          JMS DATAIN
      00220   2127           ISZ R4
      00221   3103           DCA K133
      00222   7200           CLA
      00223   1112           TAD K147
      00224   1103           TAD K133
      00225   7440           SZA
      00226   5231           JMP ARRS
      00227   7000  NOPTON,  NOP                /IF COEFFICIENT IS "2" PERFORM NOP
      00230   5274           JMP ADRE
      00231   7500  ARRS,    SMA
      00232   5253           JMP RS
      00233   7200           CLA
      00234   1111           TAD K146
      00235   1103           TAD K133
      00236   7440           SZA
      00237   5246           JMP ADD
```

```
00240  7300  RAD,    CLA CLL          /IF COEFFICIENT IS "1" SHIFT THE INPUT DATA RIGHT
00241  1120          TAD RA1          /AND ADD TO THE AC.
00242  7010          RAR
00243  1122          TAD RA
00244  3122          DCA RA
00245  5274          JMP ADRE
00246  7300  ADD,    CLA CLL          /IF COEFFICIENT IS "0" ADD THE INPUT DATA TO THE
00247  1120          TAD RA1          /AC.
00250  1122          TAD RA
00251  3122          DCA RA
00252  5274          JMP ADRE
00253  7200  RS,     CLA
00254  1110          TAD K145
00255  1103          TAD K133
00256  7440          SZA
00257  5267          JMP SUB
00260  7300  RSB,    CLA CLL          /IF COEFFICIENT IS "3" SHIFT THE INPUT DATA RIGHT
00261  1120          TAD RA1          /AND SUB. FROM THE AC.
00262  7010          RAR
00263  7041          CIA
00264  1122          TAD RA
00265  3122          DCA RA
00266  5274          JMP ADRE
00267  7300  SUB,    CLA CLL          /IF COEFFICIENT IS "4" SUB. THE INPUT DATA FROM
00270  1120          TAD RA1          /AC.
00271  7041          CIA
00272  1122          TAD RA
00273  3122          DCA RA
00274  7300  ADRE,   CLA CLL          /IS THIS THE 128TH OPERATIONS?
00275  1125          TAD R5
00276  1113          TAD K156
00277  7440          SZA
00300  5213          JMP YES1         /NO, CONTINUE THE OPERATIONS.
00301  7200          CLA              /YES,HAVE I&Q CORRELATION BEEN PERFORMED?
00302  3125          DCA R5
00303  1124          TAD R7
00304  1111          TAD K146
00305  7450          SNA
00306  5336          JMP YES2         /YES, GO TO YES2.
00307  7200  NO2,    CLA              /NO, STORE THE I CORRELATION RESULT IN R9.
```

```
00310  1122            TAD RA
00311  3123            DCA R9
00312  2124            ISZ R7
00313  4777' INPUT,    JMS WAIT          /WAITE FOR COMMAND TO READ IN NEXT INPUT DATA.
00314  4776'           JMS DATAIN
00315  3521            DCA I R8
00316  2121            ISZ R8
00317  7200            CLA
00320  1121            TAD R8
00321  1100            TAD K7000
00322  1107            TAD K143
00323  7450            SNA               /IS THIS THE 256TH INPUT DATA?
00324  5326            JMP I1
00325  5330            JMP I2
00326  1114  I1,       TAD K161          /YES, SET INPUT DATA ADDRESS COUNTER TO 0, AND GO TO
00327  3121            DCA R8            /YES1
00330  7200  I2,       CLA               /NO, UPDATE THE OUTPUT DATA ADDRESS COUNTER, AND GO TO
00331  1113            TAD K156          /YES1.
00332  1126            TAD R6
00333  3126            DCA R6
00334  3122            DCA RA
00335  5213            JMP YES1
00336  1123  YES2,     TAD R9            / IF I & Q CORRELATIONS HAVE BEEN PERFORMED
00337  0106            AND K141          /PROCEED TO TEST THE RESULTS OF THE CORRELATIONS,
00340  7440            SZA               /CONVERT ANY -VE VALUE TO +VE.
00341  5775'           JMP CONVE1
00342  1122  YES3,     TAD RA
00343  0106            AND K141
00344  7440            SZA
00345  5774'           JMP CONVE2
00346  1122  MAGN,     TAD RA            /AFTER THE MAGNITUDE CONVERSION
00347  7041            CIA               /PROCEED TO CALCULATE THE ENVOLPE VALUE OF THE
00350  1123            TAD R9            /RESULTS.
00351  7510            SPA
00352  5773'           JMP EXCHAN
00353  5772'           JMP DIVIS
00372  0400
00373  0437
00374  0432
00375  0425
```

```
00376  0600
00377  0727
       0400' PAGE
00400  7300  DIVIS,   CLA CLL
00401  1123           TAD R9
00402  7110           CLL RAR
00403  7110           CLL RAR
00404  7110           CLL RAR
00405  7110           CLL RAR
00406  7110           CLL RAR
00407  7041           CIA
00410  1123           TAD R9
00411  3123           DCA R9
00412  7300  DONE,    CLA CLL
00413  1122           TAD RA
00414  7110           CLL RAR
00415  7110           CLL RAR
00416  7110           CLL RAR
00417  3122           DCA RA
00420  1122           TAD RA
00421  1122           TAD RA
00422  1122           TAD RA
00423  1123           TAD R9
00424  5247           JMP OUT
00425  7300  CONVE1,  CLA CLL
00426  1123           TAD R9
00427  7041           CIA
00430  3123           DCA R9
00431  5777'          JMP YES3
00432  7300  CONVE2,  CLA CLL
00433  1122           TAD RA
00434  7041           CIA
00435  3122           DCA RA
00436  5776'          JMP MAGN
00437  7300  EXCHAN,  CLA CLL
00440  1123           TAD R9
00441  3102           DCA K122
00442  1122           TAD RA
00443  3123           DCA R9
00444  1102           TAD K122
```

```
00445  3122           DCA RA
00446  5200           JMP DIVIS
00447  4775'  OUT,    JMS DATAOT                /OUTPUT THE CALCULATED ENVOLPE RESULT
00450  7300   TEST,   CLA CLL
00451  3122           DCA RA
00452  3124           DCA R7
00453  1127           TAD R4
00454  1101           TAD K121
00455  7440           SZA
00456  5261           JMP I5
00457  3102           DCA K122
00460  5262           JMP I6
00461  3102   I5,     DCA K122
00462  4774'  I6,     JMS WAIT          /WAITE FOR COMMAND TO READ IN THE NEXT INPUT DATA.
00463  4773'          JMS DATAIN
00464  2121           ISZ R8
00465  7300           CLA CLL
00466  1121           TAD R8            /IS THIS THE 256TH INPUT DATA?
00467  1100           TAD K7000
00470  1107           TAD K143
00471  7450           SNA
00472  5274           JMP I3
00473  5276           JMP I4
00474  1114   I3,     TAD K161          /YES, RESET THE INPUT DATA ADDRESS COUNTER,CONTINUE.
00475  3121           DCA R8
00476  7300   I4,     CLA CLL           /NO, CONTINUE.
00477  1102           TAD K122          /IS THIS THE 16TH DOPPLER FREQUENCY BAND?
00500  7450           SNA
00501  5307           JMP.CONT
00502  7300           CLA CLL           /NO, DECREMENT THE OUTPUT DATA ADDRESS COUNTER BY 128
00503  1113           TAD K156          /AND GO TO YES1.
00504  1126           TAD R6
00505  3126           DCA R6
00506  5772'          JMP YES1
00507  3127   CONT,   DCA R4            /YES, THIS IS THE 16TH DOPPLER FREQUENCY BAND
00510  7300           CLA CLL
00511  1126           TAD R6            /IS OUTPUT DATA ADDRESS COUNTER EQUAL 256?
00512  1100           TAD K7000
00513  1107           TAD K143
00514  7440           SZA
```

```
00515  5322          JMP R6AD32
00516  1114          TAD K161         /YES, SET THE OUTPUT DATA ADDRESS COUNTER TO 31, THEN,
00517  1105          TAD K137         /GO TO YES1.
00520  3126          DCA R6
00521  5772'         JMP YES1
00522  7300  R6AD32, CLA CLL          /NO, INCREMENT THE OUTPUT DATA ADDRESS COUNTER BY 32
00523  1104          TAD K136         /,THEN, GO TO YES1.
00524  1126          TAD R6
00525  3126          DCA R6
00526  5772'         JMP YES1
             /THE FOLLOWING SUBROUTINES ARE USED TO PERFORM THE INPUT AND OUTPUT OF THE
             /MAIN PROGRAM.
00572  0213
00573  0600
00574  0727
00575  0676
00576  0346
00577  0342
       0600  *600
00600  0000  DATAIN, 0
00601  7300  START,  CLA CLL
00602  6046          TLS
00603  4242          JMS CRLF
00604  1266  NXTNUM, TAD M4
00605  3267          DCA COUNTR
00606  1270          TAD K770
00607  3271          DCA TEMP
00610  4251  NXTDIG, JMS LISN
00611  3671          DCA I TEMP
00612  2271          ISZ TEMP
00613  2267          ISZ COUNTR
00614  5210          JMP NXTDIG
00615  4242          JMS CRLF
00616  4220          JMS PACK
00617  5600          JMP I DATAIN
00620  0000  PACK,   0
00621  3272          DCA STORE
00622  1266          TAD M4
00623  3267          DCA COUNTR
00624  1270          TAD K770
```

```
00625  3271            DCA TEMP
00626  1272  PAKDIG,   TAD STORE
00627  7104            CLL RAL
00630  7006            RTL
00631  1671            TAD I TEMP
00632  1273            TAD M260
00633  3272            DCA STORE
00634  2271            ISZ TEMP
00635  2267            ISZ COUNTR
00636  5226            JMP PAKDIG
00637  7300            CLA CLL
00640  1272            TAD STORE
00641  5620            JMP I PACK
00642  0000  CRLF,     0
00643  7300            CLA CLL
00644  1274            TAD K215
00645  4260            JMS TYPE
00646  1275            TAD K212
00647  4260            JMS TYPE
00650  5642            JMP I CRLF
00651  0000  LISN,     0
00652  6032            KCC
00653  6031            KSF
00654  5253            JMP .-1
00655  6036            KRB
00656  6046            TLS
00657  5651            JMP I LISN
00660  0000  TYPE,     0
00661  6041            TSF
00662  5261            JMP .-1
00663  6046            TLS
00664  7200            CLA
00665  5660            JMP I TYPE
00666  7774  M4,       -4
00667  0000  COUNTR,   0
00670  0772  K770,     772
00671  0000  TEMP,     0
00672  0000  STORE,    0
00673  7520  M260,     7520
00674  0215  K215,     215
```

```
00675  0212  K212,   212
00676  0000  DATAOT, 0
00677  3324          DCA NUMBER
00700  7300          CLA CLL
00701  3272          DCA STORE
00702  6046          TLS
00703  7742          SMA CRLF
00704  1266          TAD M4
00705  3267          DCA COUNTR
00706  1324          TAD NUMBER
00707  7004          RAL
00710  1272  UNPACK, TAD STORE
00711  7004          RAL
00712  7006          RTL
00713  3272          DCA STORE
00714  1272          TAD STORE
00715  0325          AND MASK7
00716  1326          TAD K260
00717  4260          JMS TYPE
00720  2267          ISZ COUNTR
00721  5310          JMP UNPACK
00722  4242          JMS CRLF
00723  5676          JMP I DATAOT
00724  0000  NUMBER, 0
00725  0007  MASK7,  7
00726  0260  K260,   260
00727  0000  WAIT,   0
00730  6046          TLS
00731  6032          KCC
00732  4242          JMS CRLF
00733  6031  ACCEPT, KSF
00734  5333          JMP .-1
00735  6036          KRB
00736  6046          TLS
00737  1362          TAD K327
00740  7440          SZA
00741  5333          JMP ACCEPT
00742  7200          CLA
00743  1370          TAD K240
00744  4260          JMS TYPE
```

```
00745  1363            TAD K311
00746  4260            JMS TYPE
00747  1364            TAD K316
00750  4260            JMS TYPE
00751  1365            TAD K320
00752  4260            JMS TYPE
00753  1366            TAD K325
00754  4260            JMS TYPE
00755  1367            TAD K324
00756  4260            JMS TYPE
00757  1371            TAD K277
00760  4260            JMS TYPE
00761  5727            JMP I WAIT
00762  7451  K327,     7451
00763  0311  K311,     311
00764  0316  K316,     316
00765  0320  K320,     320
00766  0325  K325,     325
00767  0324  K324,     324
00770  0240  K240,     240
00771  0277  K277,     277
       0100  *100
00100  1000  K7000,    1000
00101  0000  K121,     0
00102  0000  K122,     0
00103  0000  K133,     0
00104  0040  K136,     40
00105  0037  K137,     37
00106  4000  K141,     4000
00107  7400  K143,     7400
00110  7775  K145,     7775
00111  7777  K146,     7777
00112  7776  K147,     7776
00113  7600  K156,     7600
00114  7000  K161,     7000
00115  0000  K162,     0
00116  0000  K163,     0
00117  7000  K164,     7000
00120  0000  RA1,      0
00121  7177  R8,       7177
```

```
00122  1000  RA,    1000
00123  1500  R9,    1500
00124  0001  R7,    1
00125  0177  R5,    177
00126  7177  R6,    7177
00127  4000  R4,    4000
             $
```

```
      PROGRAM TST (INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT)
       R1=3.0*3.141592654/4.0
       R2=7.0*3.141592654/8.0
       BASE=0.20
       SAMPLE=315.0E-06
       SEVL1=SIN(R1)
       SEVL2=SIN(R2)
       DO 1 K = 1,16
       K1=K-1
       HZ=312.5+(25.0*FLOAT(K1))
       W=2.0*3.141592654*HZ
       DISPL=W*SAMPLE
       WRITE(6,10)
 10    FORMAT(1H1,5X,*MACRO*,4X,*VALUES*,5X,*NO. OF OP..*,7X,*NO.*,5X,*A
     CCCUMULATED OP.*,5X,*TIME NEEDED (US)*,5X,*MACRO CODE*,5X,*HEX. COD
     CE*,//)
       WRITE (6,11) HZ
 11    FORMAT (1H0,10X,*FREQUENCY=*,2X,F7.2,//)
       L=0
       DO 30 I=1,256
       N=I-1
       IF(N.GT.127) GO TO 40
       ANG=DISPL*FLOAT(N)
       VALU1=SIN(ANG)
       IF ( VALU1 .LE. 1.0 .AND. VALU1 .GT. SEVL1) GO TO 50
       IF ( VALU1 .LE. SEVL1 .AND. VALU1 .GT. SEVL2 ) GO TO 60
       IF ( VALU1 .LE.SEVL2 .AND. VALU1 .GE. (-SEVL2)) GO TO 70
       IF ( VALU1 .LT. (-SEVL2) .AND. VALU1 .GE. (-SEVL1)) GO TO 80
       IF ( VALU1 .LT. (-SEVL1) .AND. VALU1 .GE. (-1.0)) GO TO 90
 50    IF ( N .EQ. 127) GO TO 110
       M=10
       WRITE (6,51) VALU1,M,N
 51    FORMAT (5X,*ADD*,3X,F7.4,10X,I2,10X,I4,46X,*0000*,12X,*0*)
       L=L+M
       GO TO 30
 60    IF ( N .EQ. 127) GO TO 120
       M=11
       WRITE (6,61) VALU1,M,N
 61    FORMAT (5X,*RAD*,3X,F7.4,10X,I2,10X,I4,46X,*0001*,12X,*1*)
       L=L+M
       GO TO 30
 70    IF ( N .EQ. 127) GO TO 130
       M=10
       WRITE (6,71) VALU1,M,N
 71    FORMAT (5X,*NOP*,3X,F7.4,10X,I2,10X,I4,46X,*0010*,12X,*2*)
       L=L+M
       GO TO 30
 80    IF ( N .EQ. 127) GO TO 140
       M=12
       WRITE (6,81) VALU1,M,N
 81    FORMAT (5X,*RSB*,3X,F7.4,10X,I2,10X,I4,46X,*0011*,12X,*3*)
       L=L+M
       GO TO 30
 90    IF ( N .EQ. 127) GO TO 150
       M=11
       WRITE (6,91) VALU1,M,N
 91    FORMAT (5X,*SUB*,3X,F7.4,10X,I2,10X,I4,46X,*0100*,12X,*4*)
```

```
      L=L+M
      GO TO 30
110   CONTINUE
      M=16
      L=L+M
      T=BASE*FLOAT(L)
      WRITE(6,111)  VALU1,M,N,L,T
111   FORMAT (5X,*ADD*,3X,F7.4,10X,I2,10X,I4,10X,I6,13X,F7.3,10X,*0000*
     ),12X,*0*)
      L=0
      GO TO 30
120   CONTINUE
      M=17
      L=L+M
      T=BASE*FLOAT(L)
      WRITE(6,121)  VALU1,M,N,L,T
121   FORMAT (5X,*RAD*,3X,F7.4,10X,I2,10X,I4,10X,I6,13X,F7.3,10X,*0001*
     ),12X,*1*)
      L=0
      GO TO 30
130   CONTINUE
      M=16
      L=L+M
      T=BASE*FLOAT(L)
      WRITE(6,131)  VALU1,M,N,L,T
131   FORMAT (5X,*NOP*,3X,F7.4,10X,I2,10X,I4,10X,I6,13X,F7.3,10X,*0010*
     ),12X,*2*)
      L=0
      GO TO 30
140   CONTINUE
      M=18
      L=L+M
      T=BASE*FLOAT(L)
      WRITE(6,141)  VALU1,M,N,L,T
141   FORMAT (5X,*RSB*,3X,F7.4,10X,I2,10X,I4,10X,I6,13X,F7.3,10X,*0011*
     ),12X,*3*)
      L=0
      GO TO 30
150   CONTINUE
      M=17
      L=L+M
      T=BASE*FLOAT(L)
      WRITE(6,151)  VALU1,M,N,L,T
151   FORMAT (5X,*SLB*,3X,F7.4,10X,I2,10X,I4,10X,I6,13X,F7.3,10X,*0100*
     ),12X,*4*)
      L=0
      GO TO 30
40    N1=N-128
      ANG=DISPL*FLOAT(N1)
      VALU1=COS(ANG)
      IF ( VALU1 .LE. 1.0 .AND. VALU1 .GT. SEVL1 ) GO TO 250
      IF ( VALU1  .LE. SEVL1 .AND. VALU1 .GT. SEVL2 ) GO TO 260
      IF ( VALU1 .LE. SEVL2 .AND. VALU1 .GE. (-SEVL2)) GO TO 270
      IF ( VALU1 .LT. (-SEVL2) .AND. VALU1 .GE. (-SEVL1)) GO TO 280
      IF ( VALU1 .LT. (-SEVL1) .AND. VALU1 .GE. (-1.0)) GO TO 290
250   IF (N .EQ. 255) GO TO 300
      M=10
      WRITE (6,251) VALU1,M,N
  251 FORMAT (5X,*ADD*,3X,F7.4,10X,I2,10X,I4,46X,*0000*,12X,*0*)
      L=L+M
      GO TO 30
  260 IF (N .EQ. 255) GO TO 310
      M=11
      WRITE (6,261) VALU1,M,N
  261 FORMAT (5X,*RAD*,3X,F7.4,10X,I2,10X,I4,46X,*0001*,12X,*1*)
      L=L+M
      GO TO 30
  270 IF (N .EQ. 255) GO TO 320
      M=10
      WRITE (6,271) VALU1,M,N
  271 FORMAT (5X,*NOP*,3X,F7.4,10X,I2,10X,I4,46X,*0010*,12X,*2*)
      L=L+M
      GO TO 30
  280 IF (N .EQ. 255) GO TO 330
      M=12
      WRITE (6,281) VALU1,M,N
  281 FORMAT (5X,*RSB*,3X,F7.4,10X,I2,10X,I4,46X,*0011*,12X,*3*)
      L=L+M
      GO TO 30
  290 IF (N .EQ. 255) GO TO 340
      M=11
      WRITE (6,291) VALU1,M,N
  291 FORMAT (5X,*SUB*,3X,F7.4,10X,I2,10X,I4,46X,*0100*,12X,*4*)
      L=L+M
      GO TO 30
  300 CONTINUE
      M=58
      L=L+M
      T=BASE*FLOAT(L)
      WRITE(6,301) VALU1,M,N,L,T
  301 FORMAT (5X,*ADD*,3X,F7.4,10X,I2,10X,I4,10X,I6,13X,F7.3,10X,*0000*
     ),12X,*0*)
      GO TO 30
  310 CONTINUE
      M=59
      L=L+M
      T=BASE*FLOAT(L)
      WRITE(6,311) VALU1,M,N,L,T
  311 FORMAT (5X,*RAD*,3X,F7.4,10X,I2,10X,I4,10X,I6,13X,F7.3,10X,*0001*
     ),12X,*1*)
      GO TO 30
  320 CONTINUE
      M=58
      L=L+M
      T=BASE*FLOAT(L)
      WRITE(6,321) VALU1,M,N,L,T
  321 FORMAT (5X,*NOP*,3X,F7.4,10X,I2,10X,I4,10X,I6,13X,F7.3,10X,*0010*
     ),12X,*2*)
      GO TO 30
  330 CONTINUE
      M=60
      L=L+M
      T=BASE*FLOAT(L)
      WRITE(6,331) VALU1,M,N,L,T
```

```
331     FCRMAT (5X,*RSB*,3X,F7.4,10X,I2,10X,I4,10X,I6,13X,F7.3,10X,*0011*
       ),12X,*3*)
        GO TO 30
340     CCNTINUE
        M=59
        L=L+M
        T=BASE*FLOAT(L)
        WRITE(6,341)  VALU1,M,N,L,T
341     FCRMAT (5X,*SUB*,3X,F7.4,10X,I2,10X,I4,10X,I6,13X,F7.3,10X,*0100*
       ),12X,*4*)
30      CCNTINUE
1       CCNTINUE
        END
```

```
 /MICROPROGRAM/

/THE FOLLOWING IS A LISTING OF THE MICROPROGRAM THAT IS USED IN A
/INTEL 3000 MICROCOMPUTER FOR THE PURPOSE OF PROCESSING SONAR SIGNALS.
/THIS PROGRAM CALCULATES THE CROSS CORRELATION FUNCTION OF SAMPLED DATAS
/AND 16 SETS OF STORED REPLICAS.

INT0,    CLR(AC)                 JCC(INT1)       /CLEAR ALL THE REGISTERS.
INT1,    CLR(T)                  JCC(INT2)
INT2,    CLR(R0)                 JCC(INT3)
INT3,    CLR(R1)                 JZR(INT4)
INT4,    CLR(R2)                 JCC(INT5)
INT5,    CLR(R3)                 JCC(INT6)
INT6,    CLR(R4)                 JCC(INT7)
INT7,    CLR(R5)                 JCC(INT8)
INT8,    CLR(R6)                 JCR(INT9)
INT9,    CLR(R7)                 JCC(INT10)
INT10,   CLR(R8)                 JCC(INT11)
INT11,   CLR(R9)                 JCR(READ)
READ,    NOP                     JCR(READ)       /START READING IN DATA.
READ+1,  LMI(R8),FF1             JCR(READ+2)     /UPDATE THE INPUT ADDRESS
READ+2,  NOP,ADS                 JCR(READ+3)     /COUNTER.
READ+3,  TZR(R8),K0000100,INH    JCR(READ+4)     /IS THIS THE 128TH DATA?
READ+4,  NOP                     JFL(NO0,YES0)
NO0,     NOP                     JCR(READ)       /NO, GO AND READ MORE.
YES0,    INR(R5),FF1             JZR(YES0+1)     /YES, IT IS 128TH DATA.
YES0+1,  LMI(R6),FF1             JCC(YES0+2)     /UPDATE THE OUTPUT DATA
YES0+2,  LDI(AC),FF1,RAM,STC     JZR(FETCH)      /ADDRESS COUNTER AND READ
FETCH,   LMI(R4),FF1             JCR(FETCH+1)    /OUT THE DATA.
FETCH+1,ALR(AC),STZ,RAM          JCC(FETCH2)     /UPDATE THE MACRO PC..
FETCH2,  SRA(AC),FFZ             JPX(ADD,RAD,NOP,RSB,SUB)
ADD,     ALR(A)                  JZR(ADRE)       /ADD DATA TO AC AND GO TO ADRE
RAD,     SRA(AC),FFZ             JCC(AD1)        /SHIFT DATA RIGHT AND ADD TO
AD1,     ALR(A)                  JZR(ADRE)       /AC, GO TO ADRE.
NOP,     NOP                     JZR(ADRE)       /NOP , GO TO ADRE
SUB,     CIA(AC)                 JCC(SUB1)       /SUB. DATA FROM AC, GO TO ADRE
SUB1,    ALR(A)                  JZR(ADRE)
RSB,     SRA(AC),FFZ             JCC(RSB1)       /SHIFT DATA RIGHT AND SUB.FROM
RSB1,    CIA(AC),FF1             JCR(RSB2)       /AC, GO TO ADRE.
RSB2,    ALR(A)                  JZR(ADRE)
ADRE,    TZR(R5),K0000100,INH    JCC(TEST1)      /IS THIS 128TH OPERATIONS?
TEST1,   NOP                     JFL(NO1,YES1)
NO1,     INR(R5),FF1             JZR(YES0+1)     /NO, GO TO YES0+1.
```

```
YES1,   TZR(R7)                  JCR(YES1+1)      /YES, HAVE I & Q BEEN PERFORMED?
YES1+1, DSM(R5),K1111100,FF1     JFL(NO2,YES2)
NO2,    ILR(A)                   JCR(NO2+1)       /NO, STORE CURRENT RESULT
NO2+1,  SDR(R9),FF1              JCR(NO2+2)       /IN R9.
NO2+2,  INR(R7),FF1              JCR(INPUT)
INPUT,  NOP                      JCR(IN)          /UPDATE THE INPUT DATA ADDRESS
IN,     NOP                      JCR(IN)          /COUNTER.
IN+1,   LMI(R8),FF1              JCR(IN+2)        /READ IN THE NEXT DATA.
IN+2,   NOP,ADS                  JCC(IN+3)        /IS INPUT DATA ADDRESS COUNTER
IN+3,   TZR(R8),K0001000,INH     JCR(INT+4)       /EQUAL TO 256?
IN+4,   CLR(A)                   JFL(NO4,YES4)
YES4,   CLR(R8)                  JCR(NO2+3)       /YES, SET THE COUNTER TO 0.
NO4,    CLR(AC)                  JCR(NO2+3)       /NO, CLEAR THE AC.
NO2+3,  DSM(R6),K1111011,FF1     JZR(YES0+1)      /DECREMENT THE OUTPUT DATA
YES2,   ILR(A)                   JZR(YES2+1)      /ADDRESS COUNTER BY 128.
YES2+1, ALR(AC),STZ              JCR(YES2+2)      /YES, I&Q ARE DONE.  GO TO
YES2+2, SRA(AC),FFZ              JZF(NO7,YES7)    /DO THE MAGNITUDE CONVERSION.
NO7,    SDR(A),FF1               JCR(TSTR9)
YES7,   CIA(AC),FF1              JCR(YES7+1)
YES7+1, SDR(A),FF1               JCR(TSTR9)
TSTR9,  ILR(R9)                  JCR(TSTR91)
TSTR91, ALR(AC),STZ              JCC(TSTR92)
TSTR92, SRA(AC),FFZ              JZF(NO8,YES8)
NO8,    SDR(R9),FF1              JCR(MAGN)
YES8,   CIA(AC),FF1              JCR(YES8+1)
YES8+1, SDR(R9),FF1              JCR(MAGN)
MAGN,   ILR(A)                   JCC(COMP)        /AFTER THE MAGNITUDE CONVERSION
COMP,   CIA(AC),FF1              JCC(COMP+1)      /GO TO DO THE ENVOLPE DETECTION.
COMP+1, SDR(T),FF1               JCR(COMP+2)
COMP+2, ILR(R9)                  JCR(COMP+3)
COMP+3, ALR(T)                   JCC(TEST3)
TEST3,  NOP                      JFL(NO3,YES3)
YES3,   ILR(R9)                  JCR(YES31)
YES31,  SRA(AC)                  JCR(YES32)
YES32,  SRA(AC)                  JCR(YES33)
YES33,  SRA(AC)                  JCC(YES34)
YES34,  SRA(AC)                  JCR(YES35)
YES35,  SRA(AC)                  JCC(YES36)
YES36,  CIA(AC),FF1              JCC(YES37)
YES37,  ALR(R9)                  JCR(YES38)
YES38,  ILR(A)                   JCC(YES39)
YES39,  SRA(AC)                  JCR(YES310)
YES310, SRA(AC)                  JCR(YES311)
```

```
YES311, SRA(AC)                 JCC(YES312)
YES312, SDR(A),FF1              JCR(YES313)
YES313, ADR(A)                  JCR(YES314)
YES314, ADR(A)                  JCC(YES315)
YES315, ILR(A)                  JCC(YES316)
YES316, ALR(R9)                 JCR(YES317)
YES317, CLR(T)                  JCC(YES318)    /OUTPUT THE CALCULATED
YES318, LMI(T)                  JCC(YES319)    /ENVOLPE RESULT
YES319, NOP,ROT                 JZR(CLEAR)
NO3,    ILR(A)                  JCR(NO3+1)
NO3+1,  SDR(T),FF1              JCC(NO3+2)
NO3+2,  ILR(R9)                 JCR(NO3+3)
NO3+3,  SDR(A),FF1              JCC(NO3+4)
NO3+4,  ILR(T)                  JCC(NO3+5)
NO3+5,  SDR(R9),FF1             JCR(YES3)
CLEAR,  CLR(R7)                 JCC(TEST4)     /IS THIS THE 16TH DOPPLER
TEST4,  TZR(R4),K0100000,INH,STC JCC(TEST41)   /FREQUENCY?
TEST41, CLR(AC)                 JCR(GET)
GET,    NOP                     JCR(GET)       /UPDATE THE INPUT DATA
GET+1,  LMI(R8),FF1             JCR(GET+2)     /ADDRESS COUNTER AND READ
GET+2,  NOP,ADS                 JCR(GET+3)     /IN NEXT DATA.  IS
GET+3,  TZR(R8),K0001000,INH    JCR(GET+4)     /THE INPUT DATA ADDRESS
GET+4,  CLR(A)                  JFL(NO,YES)    /COUNTER EQUAL TO 256?
YES,    CLR(R8)                 JCF(NO6,YES6)  /YES, RESET THE COUNTER TO 0.
NO,     NOP                     JCF(NO6,YES6)  /NO,CONTINUE.
NO6,    DSM(R6),K1111011,FF1    JZR(YES0+1)    /NO, THIS IS NOT THE 16TH
                                               /DOPPLER FREQUENCY. DECREMENT
                                               /THE OUTPUT DATA ADDRESS COUNTER
                                               /BY 128 AND CONTINUE THE
                                               /OPERATIONS.
YES6,   CLR(R4)                 JCR(NEXT)      /YES, THIS IS THE 16TH DOPPLER
NEXT,   DSM(R2),K0000001,FF1    JZR(YES0+A)    /FREQUENCY. RESET THE MACRO.
YES0+A, ILR(R20                 JCR(YES0+B)    /INSTRUCTIONS COUNTER TO 0 AND
YES0+B, SDR(R9),FF1             JCR(YES0+1)    /INCREMENT THE OUTPUT DATA
                                               /ADDRESS COUNTER BY 32.
                                               /CONTINUE.

$
```

			JFL,JCF,JZF		JLL COLUMN RESTRICT						JFZ,JCF,JZF		JRL COLUMN RESTRICT			
			F,C,Z = 0	F,C,Z = 1							F,C,Z = 0	F,C,Z = 1				
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	INT0	INT4	*	YES2 + 1	YES0 + 1	ADRE	YES0 + A	YES0 + B	CLEAR	YES2 + 2	NO	YES	*	*	FETCH	FETCH 1
1	INT1	INT5	NO1	YES1	YES0 + 2	TEST1	YES1 + 1	COMP + 3	*	*	NO6	YES6	NEXT	YES3 19	COMP + 2	COMP + 1
2	INT2	INT6	NO2	YES2	NO2 + 1	NO2 + 2	IN + 1	IN	INPUT	IN + 2	*	*	*	YES3 17	YES3 16	COMP
3	INT3	INT7	NO4	YES4	NO3 + 2	NO2 + 3	NO3 + 3	IN + 4	*	IN + 3	*	*	*	YES3 18	YES3 15	*
4	ADD	RAD	NOP	RSB	SUB	*	NO3 + 4	TEST3	TEST4	*	*	*	YES3 12	YES3 13	YES3 14	FETCH 2
5	INT10	AD1	RSB + 2	RSB + 1	SUB + 1	GET + 2	GET + 1	GET	TEST4 + 1	YES3 10	GET + 3	GET + 4	YES3 11	YES3 4	YES3 5	YES3 9
6	INT9	INT8	YES3	NO3	NO3 + 1	YES3 1	NO3 + 5	YES3 2	TESTR9 2	*	NO8	YES8	YES8 + 1	YES3 3	YES3 6	MAGN
7	INT11	READ + 2	NO0	YES0	READ + 1	READ	READ + 3	READ + 4	TESTR9 1	*	NO7	YES7	YES7 + 1	TESTR9	YES3 7	YES3 9

* means not used

MNEMONIC	MICRO FUNCTIONS	FLAG. CONDITION	MC AND I/O CONTROL	K BUS	ADDRESS CONTROL	HEX. CODE
INT0	00110010	1100	011000	0000000	1111110	3 2 C 6 0 0 7 E
INT4	00111101	1100	011000	0000000	1111110	3 D C 6 0 0 7 E
*	00010010	1100	011000	0000000	1011111	1 2 C 6 0 0 5 F
YES2+1	01110010	1110	011000	1111111	1000110	7 2 E 6 3 F C 6
YES0+1	01101001	0000	011000	0000000	1111110	6 9 0 6 0 0 7 E
ADRE	00101010	1100	010000	0000100	1111110	2 A C 4 0 2 7 E
YES0+A	01111101	1100	011000	0000000	1001000	7 D C 6 0 0 4 8
YES0+B	01011001	0000	011000	1111111	1001011	5 9 0 6 3 F C B
CLEAR	00111000	1100	011000	0000000	1111011	3 8 C 6 0 0 7 B
YES2+2	01110000	0100	011000	0000000	0100000	7 0 4 6 0 0 2 0
NO	00010010	1100	011000	0000000	0101110	1 2 C 6 0 0 2 E
YES	00110111	1100	011000	0000000	0101110	3 7 C 6 0 0 2 E
*	00010010	1100	011000	0000000	1011111	1 2 C 6 0 0 5 F
*	00010010	1100	011000	0000000	1011111	1 2 C 6 0 0 5 F
FETCH	01101011	0000	011000	0000000	1000000	6 B 0 6 0 0 4 0
FETCH1	01110010	1110	001101	1111111	1111011	7 2 E 3 7 F F B
INT1	00110011	1100	011000	0000000	1111101	3 3 C 6 0 0 7 D
INT5	00111100	1100	011000	0000000	1111101	3 C C 6 0 0 7 D
NO1	01001010	0000	011000	0000000	1011011	4 A 0 6 0 0 5 B
YES1	00101000	1100	011000	1111111	1001001	2 8 C 6 3 F C 9
YES0+2	01010000	0001	001101	1111111	1010001	5 0 1 3 7 F D 1
TEST1	00010010	1100	011000	0000000	0111110	1 2 C 6 0 0 3 E
YES1+1	01101010	0000	011000	1111100	0111101	6 A 0 6 3 E 3 D
COMP+3	01110011	1100	011000	1111111	1111011	7 3 C 6 3 F F B
*	00010010	1100	011000	0000000	1011111	1 2 C 6 0 0 5 F
*	00010010	1100	011000	0000000	1011111	1 2 C 6 0 0 5 F
NO6	01101001	0000	011000	1111011	1011011	6 9 0 6 3 D D B
YES6	00111011	1100	011000	0000000	1000011	3 B C 6 0 0 4 3
NEXT	01101101	0000	011000	0000001	1011001	6 D 0 6 0 0 D 9
YES319	00010010	1100	001011	0000000	1010111	1 2 C 2 C 0 5 7
COMP+2	01110110	1100	011000	0000000	1001000	7 6 C 6 0 0 4 8
COMP+1	01010011	0000	011000	1111111	1000001	5 3 0 6 3 F C 1
INT2	00111111	1100	011000	0000000	1111100	3 F C 6 0 0 7 C
INT6	00111011	1100	011000	0000000	1111100	3 B C 6 0 0 7 C

MNEMONIC	MICRO FUNCTIONS	FLAG CONDITION	MC AND I/O CONTROL	K BUS	ADDRESS CONTROL	HEX. CODE
NO2	01111111	1100	011000	0000000	1001011	7 F C 6 0 0 4 B
YES2	01111111	1100	011000	0000000	1011100	7 F C 6 0 0 5 C
NO2+1	01010110	0000	011000	1111111	1001010	5 6 0 6 3 F C A
NO2+2	01001000	0000	011000	0000000	1000111	4 8 0 6 0 0 4 7
IN+1	01100111	0000	011000	0000000	1000110	6 7 0 6 0 0 4 6
IN	00010010	1100	011000	0000000	1001000	1 2 C 6 0 0 4 8
INPUT	00010010	1100	011000	0000000	1001000	1 2 C 6 0 0 4 8
IN+2	00010010	1100	101101	0000000	1111100	1 2 C B 4 0 7 C
*	00010010	1100	011000	0000000	1011111	1 2 C 3 0 0 5 F
*	00010010	1100	011000	0000000	1011111	1 2 C 3 0 0 5 F
*	00010010	1100	011000	0000000	1011111	1 2 C 3 0 0 5 F
YES317	00110011	1100	011000	0000000	1111100	3 3 C 6 0 0 7 C
YES316	01110110	1100	011000	1111111	1000010	7 6 C 6 3 F C 2
COMP	01100000	0000	011000	0000000	1111110	6 0 0 6 0 0 7 E
INT3	00111110	1100	011000	0000000	1011110	3 E C 6 0 0 5 E
INT7	00111010	1100	011000	0000000	1111001	3 A C 6 0 0 7 9
NO4	00110010	1100	011000	0000000	1001010	3 2 C 6 0 0 4 A
YES4	00110111	1100	011000	0000000	1001010	3 7 C 6 0 0 4 A
NO3+2	01110110	1100	011000	0000000	1001001	7 6 C 6 0 0 4 9
NO2+3	01101001	0000	011000	1111011	1011011	6 9 0 6 3 D D B
NO3+3	01011111	0000	011000	1111111	1111011	5 F 0 6 3 F F B
IN+4	00111111	1100	011000	0000000	0111100	3 F C 6 0 0 3 C
*	00010010	1100	011000	0000000	1011111	1 2 C 6 0 0 5 F
IN+3	00100111	1100	010000	0001000	1001000	2 7 C 4 0 4 4 8
*	00010010	1100	011000	0000000	1011111	1 2 C 3 0 0 5 F
*	00010010	1100	011000	0000000	1011111	1 2 C 3 0 0 5 F
*	00010010	1100	011000	0000000	1011111	1 2 C 3 0 0 5 F
YES318	01100011	1100	011000	0000000	1111110	6 3 C 6 0 0 7 E
YES315	01111111	1100	011000	0000000	1111101	7 F C 6 0 0 7 D
*	00010010	1100	011000	0000000	1011111	1 2 C 6 0 0 5 F
ADD	01111111	1100	011000	1111111	1011010	7 F C 6 3 F D A
RAD	01110000	0100	011000	0000000	1111010	7 0 4 6 0 0 7 A
NOP	00010010	1100	011000	0000000	1011010	1 2 C 6 0 0 5 A
RSB	01110000	0100	011000	0000000	1111010	7 0 4 6 0 0 7 A

MNEMONIC	MICRO FUNCTIONS	FLAG CONDITION	MC AND I/O CONTROL	K BUS	ADDRESS CONTROL	HEX. CODE
SUB	01100000	0000	011000	0000000	1111010	6 0 0 6 0 0 7 A
*	00010010	1100	011000	0000000	1011111	1 2 C 6 0 0 5 F
NO3+4	01110011	1100	011000	0000000	1111001	7 3 C 6 0 0 7 9
TEST3	00010010	1100	011000	0000000	0111001	1 2 C 6 0 0 3 9
TEST4	00101011	1101	010000	0100000	1111010	2 B D 4 1 0 7 A
*	00010010	1100	011000	0000000	1011111	1 2 C 6 0 0 5 F
*	00010010	1100	011000	0000000	1011111	1 2 C 6 0 0 5 F
*	00010010	1100	011000	0000000	1011111	1 2 C 6 0 0 5 F
YES312	01011111	0000	011000	1111111	1000010	5 F 0 6 3 F C 2
YES313	01001111	1100	011000	1111111	1000001	4 F C 6 3 F C 1
YES314	01001111	1100	011000	1111111	1111100	4 F C 6 3 F F C
FETCH2	01110000	0100	011000	0000000	0000111	7 0 4 6 0 0 0 7
INT10	00110111	1100	011000	0000000	1111000	3 7 C 6 0 0 7 8
AD1	01111111	1100	011000	1111111	1011010	7 F C 6 3 F D A
RSB+2	01111111	1100	011000	1111111	1011010	7 F C 6 3 F D A
RSB+1	01100000	0000	011000	0000000	1001101	6 0 0 6 0 0 4 D
SUB+1	01111111	1100	011000	1111111	1011010	7 F C 6 3 F D A
GET+2	00010010	1100	101101	0000000	1000101	1 2 C B 4 0 4 5
GET+1	01100111	0000	011000	0000000	1001010	6 7 0 6 0 0 4 A
GET	00010010	1100	011000	0000000	1001000	1 2 C 6 0 0 4 8
TEST4+1	00110010	1100	011000	0000000	1001000	3 2 C 6 0 0 4 8
YES310	01110000	1100	011000	0000000	1000011	7 0 C 6 0 0 4 3
GET+3	00100111	1100	010000	0001000	1000100	2 7 C 4 0 4 4 4
GET+4	00111111	1100	011000	0000000	0111111	3 F C 6 0 0 3 F
YES311	01110000	1100	011000	0000000	1111011	7 0 C 6 0 0 7 B
YES34	01110000	1100	011000	0000000	1000001	7 0 C 6 0 0 4 1
YES35	01110000	1100	011000	0000000	1111001	7 0 C 6 0 0 7 9
YES39	01110000	1100	011000	0000000	1000110	7 0 C 6 0 0 4 6
INT9	00111000	1100	011000	0000000	1111010	3 8 C 6 0 0 7 A
INT8	00111001	1100	011000	0000000	1001111	3 9 C 6 0 0 4 F
YES3	01111111	1100	011000	0000000	1001011	7 F C 6 0 0 4 B
NO3	01110110	1100	011000	0000000	1001010	7 6 C 6 0 0 4 A
NO3+1	01010011	0000	011000	1111111	1111100	5 3 0 6 3 F F C

MNEMONIC	MICRO FUNCTIONS	FLAG CONDITION	MC AND I/O CONTROL	K BUS	ADDRESS CONTROL	HEX. CODE
YES31	01110000	1100	011000	0000000	1001000	70C60048
NO3+5	01010110	0000	011000	1111111	1001100	56063FCC
YES32	01110000	1100	011000	0000000	1000010	70C60042
TESTR92	01110000	0100	011000	0000000	0100001	70460021
*	00010010	1100	011000	0000000	1011111	12C6005F
NO8	01010110	0000	011000	1111111	1000000	56063FC0
YES8	01100000	0000	011000	0000000	1000011	60060043
YES8+1	01010110	0000	011000	1111111	1000000	56063FC0
YES33	01110000	1100	011000	0000000	1111010	70C6007A
YES36	01100000	0000	011000	0000000	1111000	60060078
MAGN	01111111	1100	011000	0000000	1111101	7FC6007D
INT11	00110110	1100	011000	0000000	1001010	36C6004A
READ+2	00010010	1100	101101	0000000	1001001	12CB4049
NO0	00010010	1100	011000	0000000	1001010	12C6004A
YES0	01001010	0000	011000	0000000	1011011	4A06005B
READ+1	01100111	0000	011000	0000000	1001110	6706004E
READ	00010010	1100	011000	0000000	1001010	12C6004A
READ+3	00100111	1100	010000	0000100	1001000	27C40248
READ+4	00010010	1100	011000	0000000	0111000	12C60038
TESTR91	01110010	1110	011000	1111111	1111001	72E63FF9
*	00010010	1100	011000	0000000	1011111	12C6005F
NO7	01011111	0000	011000	1111111	1000010	5F063FC2
YES7	01100000	0000	011000	0000000	1000011	60060043
YES7+1	01011111	0000	011000	1111111	1000010	5F063FC2
TESTR9	01110110	1100	011000	0000000	1000111	76C60047
YES37	01110110	1100	011000	1111111	1000000	76C63FC0
YES38	01111111	1100	011000	0000000	1111010	7FC6007A

F-GROUP	R-GROUP	MICRO-FUNCTION	
	I	$R_n + (AC \wedge K) + CI \rightarrow R_n, AC$	
0	II	$M + (AC \wedge K) + CI \rightarrow AT$	
	III	$AT_L \wedge (\overline{I_L \wedge K_L}) \rightarrow RO$ $[AT_L \wedge (I_L \wedge K_L)] \vee [AT_H \vee (I_H \wedge K_H)] \rightarrow AT_L$	$LI \vee [(I_H \wedge K_H) \wedge AT_H] \rightarrow AT_H$
	I	$K \vee R_n \rightarrow MAR$	$R_n + K + CI \rightarrow R_n$
1	II	$K \vee M \rightarrow MAR$	$M + K + CI \rightarrow AT$
	III	$(\overline{AT} \vee K) + (AT \wedge K) + CI \rightarrow AT$	
	I	$(AC \wedge K) - 1 + CI \rightarrow R_n$	
2	II	$(AC \wedge K) - 1 + CI \rightarrow AT$	(see Note 1)
	III	$(I \wedge K) - 1 + CI \rightarrow AT$	
	I	$R_n + (AC \wedge K) + CI \rightarrow R_n$	
3	II	$M + (AC \wedge K) + CI \rightarrow AT$	
	III	$AT + (I \wedge K) + CI \rightarrow AT$	
	I	$CI \vee (R_n \wedge AC \wedge K) \rightarrow CO$	$R_n \wedge (AC \wedge K) \rightarrow R_n$
4	II	$CI \vee (M \wedge AC \wedge K) \rightarrow CO$	$M \wedge (AC \wedge K) \rightarrow AT$
	III	$CI \vee (AT \wedge I \wedge K) \rightarrow CO$	$AT \wedge (I \wedge K) \rightarrow AT$
	I	$CI \vee (R_n \wedge K) \rightarrow CO$	$K \wedge R_n \rightarrow R_n$
5	II	$CI \vee (M \wedge K) \rightarrow CO$	$K \wedge M \rightarrow AT$
	III	$CI \vee (AT \wedge K) \rightarrow CO$	$K \wedge AT \rightarrow AT$
	I	$CI \vee (AC \wedge K) \rightarrow CO$	$R_n \vee (AC \wedge K) \rightarrow R_n$
6	II	$CI \vee (AC \wedge K) \rightarrow CO$	$M \vee (AC \wedge K) \rightarrow AT$
	III	$CI \vee (I \wedge K) \rightarrow CO$	$AT \vee (I \wedge K) \rightarrow AT$
	I	$CI \vee (R_n \wedge AC \wedge K) \rightarrow CO$	$R_n \overline{\oplus} (AC \wedge K) \rightarrow R_n$
7	II	$CI \vee (M \wedge AC \wedge K) \rightarrow CO$	$M \overline{\oplus} (AC \wedge K) \rightarrow AT$
	III	$CI \vee (AT \wedge I \wedge K) \rightarrow CO$	$AT \overline{\oplus} (I \wedge K) \rightarrow AT$

NOTES:

1. 2's complement arithmetic adds 111 . . . 11 to perform subtraction of 000 . . . 01.
2. R_n includes T and AC as source and destination registers in R-group 1 micro-functions.
3. Standard arithmetic carry output values are generated in F-group 0, 1, 2 and 3 instructions.

SYMBOL	MEANING
I, K, M	Data on the I, K, and M busses, respectively
CI, LI	Data on the carry input and left input, respectively
CO, RO	Data on the carry output and right output, respectively
R_n	Contents of register n including T and AC (R-Group I)
AC	Contents of the accumulator
AT	Contents of AC or T, as specified
MAR	Contents of the memory address register
L, H	As subscripts, designate low and high order bit, respectively
+	2's complement addition
–	2's complement subtraction
$\wedge$	Logical AND
$\vee$	Logical OR
$\overline{\oplus}$	Exclusive-NOR
→	Deposit into

K-BUS = 00 MICRO-FUNCTION	MNEMONIC	K-BUS = 11 MICRO-FUNCTION	MNEMONIC
$R_n + CI \rightarrow R_n, AC$	ILR	$AC + R_n + CI \rightarrow R_n, AC$	ALR
$M + CI \rightarrow AT$	ACM	$M + AC + CI \rightarrow AT$	AMA
$AT_L \rightarrow RO \quad AT_H \rightarrow AT_L \quad LI \rightarrow AT_H$	SRA	(See Appendix A)	–
$R_n \rightarrow MAR \quad R_n + CI \rightarrow R_n$	LMI	$11 \rightarrow MAR \quad R_n - 1 + CI \rightarrow R_n$	DSM
$M \rightarrow MAR \quad M + CI \rightarrow AT$	LMM	$11 \rightarrow MAR \quad M - 1 + CI \rightarrow AT$	LDM
$\overline{AT} + CI \rightarrow AT$	CIA	$AT - 1 + CI \rightarrow AT$	DCA
$CI - 1 \rightarrow R_n$ See Note 1	CSR	$AC - 1 + CI \rightarrow R_n$ See Note 1	SDR
$CI - 1 \rightarrow AT$ See Notes 1,4	CSA	$AC - 1 + CI \rightarrow AT$ See Notes 1,4	SDA
(See CSA above)	–	$I - 1 + CI \rightarrow AT$	LDI
$R_n + CI \rightarrow R_n$	INR	$AC + R_n + CI \rightarrow R_n$	ADR
(See ACM above)	–	(See AMA above)	–
$AT + CI \rightarrow AT$	INA	$I + AT + CI \rightarrow AT$	AIA
$CI \rightarrow CO \quad 0 \rightarrow R_n$	CLR	$CI \vee (R_n \wedge AC) \rightarrow CO \quad R_n \wedge AC \rightarrow R_n$	ANR
$CI \rightarrow CO \quad 0 \rightarrow AT$	CLA	$CI \vee (M \wedge AC) \rightarrow CO \quad M \wedge AC \rightarrow AT$	ANM
(See CLA above)	–	$CI \vee (AT \wedge I) \rightarrow CO \quad AT \wedge I \rightarrow AT$	ANI
(See CLR above)	–	$CI \vee R_n \rightarrow CO \quad R_n \rightarrow R_n$	TZR
(See CLA above)	–	$CI \vee M \rightarrow CO \quad M \rightarrow AT$	LTM
(See CLA above)	–	$CI \vee AT \rightarrow CO \quad AT \rightarrow AT$	TZA
$CI \rightarrow CO \quad R_n \rightarrow R_n$	NOP	$CI \vee AC \rightarrow CO \quad R_n \vee AC \rightarrow R_n$	ORR
$CI \rightarrow CO \quad M \rightarrow AT$	LMF	$CI \vee AC \rightarrow CO \quad M \vee AC \rightarrow AT$	ORM
(See NOP above)	–	$CI \vee I \rightarrow CO \quad I \vee AT \rightarrow AT$	ORI
$CI \rightarrow CO \quad \overline{R_n} \rightarrow R_n$	CMR	$CI \vee (R_n \wedge AC) \rightarrow CO \quad R_n \overline{\oplus} AC \rightarrow R_n$	XNR
$CI \rightarrow CO \quad \overline{M} \rightarrow AT$	LCM	$CI \vee (M \wedge AC) \rightarrow CO \quad M \overline{\oplus} AC \rightarrow AT$	XNM
$CI \rightarrow CO \quad \overline{AT} \rightarrow AT$	CMA	$CI \vee (AT \wedge I) \rightarrow CO \quad I \overline{\oplus} AT \rightarrow AT$	XNI

[4] The more general operations, CSR and SDR, should be used in place of the CSA and SDA operations, respectively.