1985

An Approach to Noise Estimation and Elimination in Communication Systems

Jerome J. Viviano
University of Central Florida

AN APPROACH TO NOISE ESTIMATION AND ELIMINATION IN COMMUNICATION SYSTEMS

BY

JEROME JOSEPH VIVIANO
B.S.E.E., Southern Illinois University at Edwardsville, 1980

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Engineering
the Graduate Studies Program of the College of Engineering
University of Central Florida
Orlando, Florida

Summer Term
1985

ABSTRACT

This paper is a description of the analysis and testing of a method to estimate noise mixed in with a desired signal and to then subtract out the estimated noise. The paper documents the situation in which noise becomes a problem, the method of noise elimination, the mathematics used to analyze the noise eliminator, a digital/analog hybrid simulation of a system utilizing the techniques described, and comparative results between expected performance and actual system performance.

Also described is the practicality of the system, the limitations of the system and its unique capabilities.

Since the system is a totally new concept, it was being developed for the writing of this thesis. Therefore many changes were implemented with respect to the original design. These modifications are documented along with the reasons for the deviations from the original idea.

Conclusions and a summary are provided. Several suggestions as to where the system would be beneficial are also supplied.

ACKNOWLEDGEMENTS

The author would like to express his appreciation toward those who were patient enough to tolerate the years of his graduate studies and the months spent in preparing this paper. Those individuals include his wife, My Dung Thi, and his two sons, Jesse and Joseph. Only they and the author understand the strains which resulted from the long hours spent away from each other. The author also feels obliged to the personnel of the Technical Computing Center of Martin Marietta Aerospace in Orlando, Florida, for their generosity in providing lab facilities and equipment to test the concept. Indeed, the lab provided the tools to study an actual system and to uncover many erroneous ideas which were then replaced with the proper ones.

TABLE OF CONTENTS

SECTION I. INTRODUCTION 1

SECTION II. THE NOISE ESTIMATOR CONCEPT 5
System Practicality 15

SECTION III. MATHEMATICAL ANALYSIS 19
Signal Descriptions 23

SECTION IV. THE HYBRID SIMULATION 46

SECTION V. RESULTS AND CONCLUSIONS 55

Appendix A. A MATHEMATICAL PROOF 61

Appendix B. FORTRAN SIMULATION PROGRAM LISTING 65

LIST OF REFERENCES 134

LIST OF FIGURES

1. System Block Diagram 7
2. Sample Message Signal To Be Sent 8
3. Gated Message And Gate Signals 10
4. Total Noise And Noise Estimate 11
5. Actual Noise And Noise Estimate 12
6. Signal + In-Band Noise And Message Signal 14
7. Cleaned Signal And Message Signal 16
8. Variable Placement In System 22
9. Gating Function 23
10. Fourier Representations Of Transmitter Signals 27
11. Receiver Low Pass Filters 33
12. Theoretical System Performance 45
13. Simulation Equipment Layout 47
14. Transmitter Schematic 50
15. Filter Schematic 51
16. Receiver Schematic 52
17. Schematic Of Power Spectral Density Circuit 54

18. Theoretical vs. Actual System Performance I 57

19. Theoretical vs. Actual System Performance II 58

20. Transmit And Receive, No Noise Estimation 59

21. Transmit And Receive, With Noise Estimation 60

SECTION I
INTRODUCTION

In all communication systems, a desired signal is sent through a medium of some sort from a point of origin to a point of destination. Quite often, the message is transduced into an electrical or electromagnetic form of energy and is transmitted through a wire or the atmosphere to the destination, where reciprocal transduction returns the signal back to its original form. Another approach is to transduce the signal into electrical format and then to record the information onto a magnetic medium. The magnetic medium is then physically brought to the destination point and is either stored for later playback or played back immediately. All of these methods of transferring energy suffer from the same deficiency. They are all susceptible to electromagnetic noise interference.

Much effort has gone into developing techniques to keep out-of-band noise from affecting the signal content of interest. These include bandlimiting filters and adjacent channel interference notch filters. These usually require high order filters which are expensive and have no effect on in-band noise. Preemphasis/deemphasis has been one of the more widely used methods of combatting this in-band noise.

This method is useful in attenuating in-band noise when the noise spectrum conforms to an expected format. When in-band noise resides in unexpected areas, it is gained up rather than attenuated. Also, if the deemphasis network is not the exact reciprocal of the preemphasis network, undesired spectral shaping occurs.

In digital communications, a commonly used technique is called matched filtering (Ziemer and Trantner 1976). The concept here is to produce a voltage at the receiver whose value is affected most by the desired input symbol over its entire period and is affected least by the undesired input noise. There is no analogous counterpart in continuous wave (CW) systems.

Note that in the above-mentioned methods, none of them contain any way of extracting the noise from the desired signal. At best, the noise is attenuated to some more optimal level. That is the difference in the system described in this paper -- the noise is actually estimated and *subtracted* out of the system, thus eliminating it totally. Let it be stated from the onset that there are many occasions when this can only be done partially, producing an inaccurate noise estimate. This results in marginal improvement. There are even occasions where a completely clean incoming received signal can be corrupted to a much lower signal-to-noise ratio by the noise estimator/subtracter circuitry. The pu✓pose of this study was to demonstrate the fact that the designing and building of a system to estimate and separate in-band noise from desired signal is feasible at all. The utility of the theory developed is left to another study.

The progression of events in the study started with a simple time domain concept. A phase-shift-keyed digital modulator and demodulator utilizing the concept was built on an Electronic Associates Incorporated model 781 analog computer. Although the results were not as good as hoped for, the concept showed marginal ability in the system. There was no mathematical analysis of the theory for the preliminary simulator and the imprecise results could not be explained.

At this point, a complete mathematical analysis of the system was performed in the frequency domain. The analysis showed limitations of the system which could be possible explanations for the non-ideal performance. A subsystem to combat the shortcomings was designed and analyzed in the frequency domain. Now a large scale simulation of a general purpose transmitter/receiver was built utilizing two Electronic Associates Incorporated model 781 analog computers. The added subsystem was also incorporated into the model. Results from the simulation indicated that the added subsystem was not performing as expected. The mathematical model was then inspected for errors and one was found. After correction, the math model showed that the subsystem could not theoretically eliminate the shortcomings of the system. Comparison of the subsystem output with math model predictions correlated well. At this point it was decided that the subsystem would be eliminated from the simulation because it was accepted that it could not do that for which it was originally designed.

The reason for the inclusion of the story of the ineffective subsystem is to substantiate for the reader the extent of the analysis undertaken. The example also illustrates the importance of actually building a working model of any new concept prior to having it released to the public.

At this time the reader might be wondering if the overall concept does or does not work. The concept as originally conceived predicted total noise elimination in all cases. The simulation showed that total elimination was only possible under certain circumstances and partial elimination under others. Throughout the study, efforts were constantly shifting from math analysis to real world testing, back to math analysis and again to testing of the revised theory. It is doubtful that a completely error-free system could have been developed on paper alone, or on the simulator alone, one without the other. Moreover, the simulator provided a high level of confidence in the math model since all math model steps were able to be verified prior to their inclusion in this paper.

SECTION II
THE NOISE ESTIMATOR CONCEPT

The noise estimator is based on the concept that if the transmitter is not always sending a message signal, the receiver can be used to estimate the noise coming into its front end during the signal-off time. If the signal transmission is toggled on and off at a much higher frequency than the highest spectral component of the message signal, low-pass filters in the receiver can be used to damp out the "gated" characteristic of the received signal. The signal at this point will contain both signal plus channel noise.

During the time that the signal is gated off, another path is turned on in the receiver that passes the incoming signal through another low pass filter. Since the transmitter signal is turned off at this time, the only signal left is the noise signal. The gated noise signal is then passed through the noise estimation low-pass filter and the "gated" characteristic of the noise is then removed. This yields an estimation of the incoming "in-band" noise. The term "in-band" is used to emphasize the fact that the outputs of both the "signal+noise" and noise filters are confined to the desired message bandwidth by these filters. The in-band noise estimation is then subtracted from the in-band signal+noise to yield a cleaned recovered signal. Figure 1 is an illustration of the essential components of the system.

The above is a very brief description of the basic elements of the overall system. The actual workings of the scheme require a much more thorough explanation. Section 3 contains a frequency domain mathematical description covering the basis for each of these topics. The remainder of this section provides pictorial illustrations of the pertinent signals in both the transmitter and signal recovery* unit and a brief explanation of each. The graphics are taken from the hybrid simulation of the system to be described in Section 4. Referencing Figure 1 periodically thoughout the following explanations will assist in understanding the concepts.

Figure 2 is an illustration of a typical message signal to be sent through the system. The message here is derived from a band limited white Gaussian noise (WGN) source (Cooper and McGillem 1971) and is typical of any type of continuous wave (CW) transmission signal. This corresponds to the output of the message source block.

This is a good spot to mention the fact that the noise estimation scheme described in this paper is not confined only to CW baseband

---

*Note that the term *'signal recovery unit'* is used rather than *'receiver'* here. This is mentioned since the receiving unit could be, for example, a tape recording play-back device. The two terms will be used interchangeably.

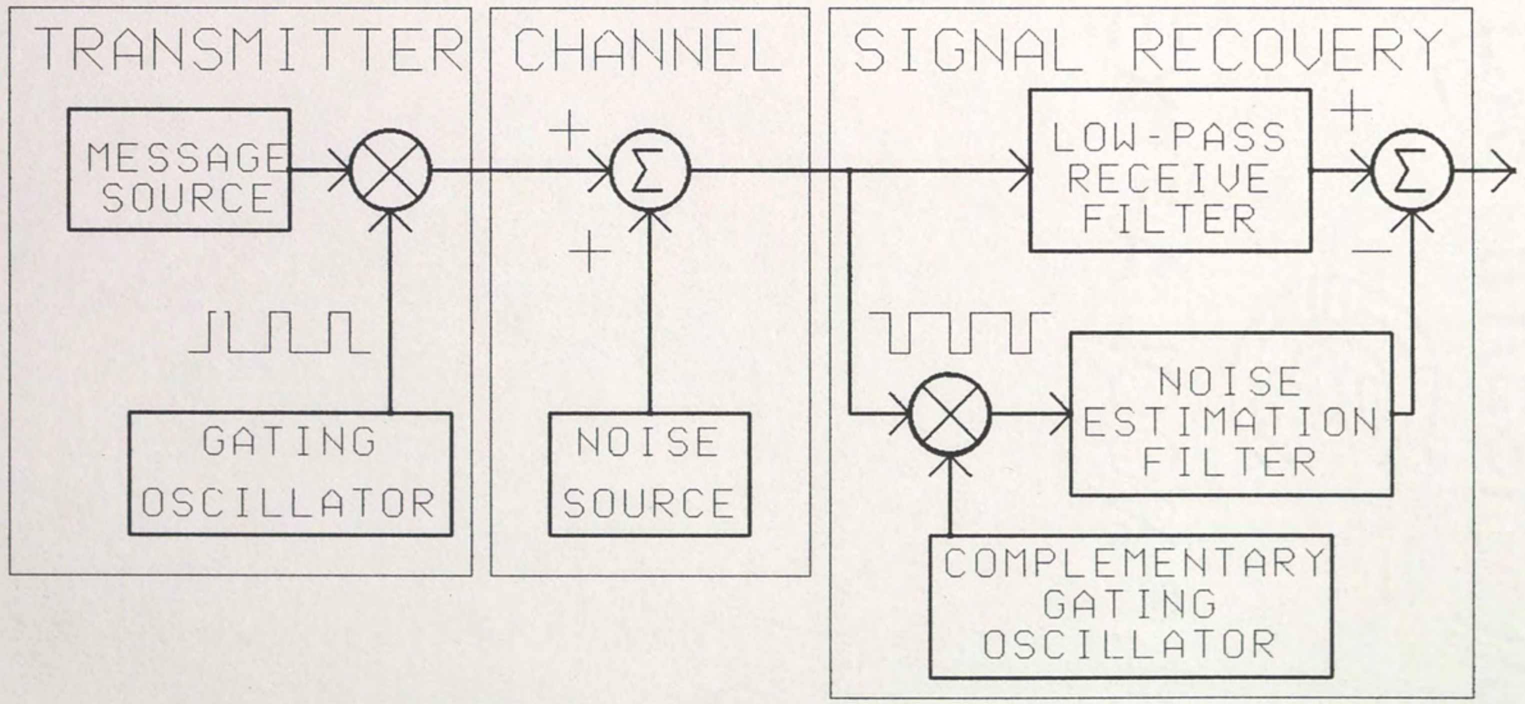

Figure 1. System Block Diagram

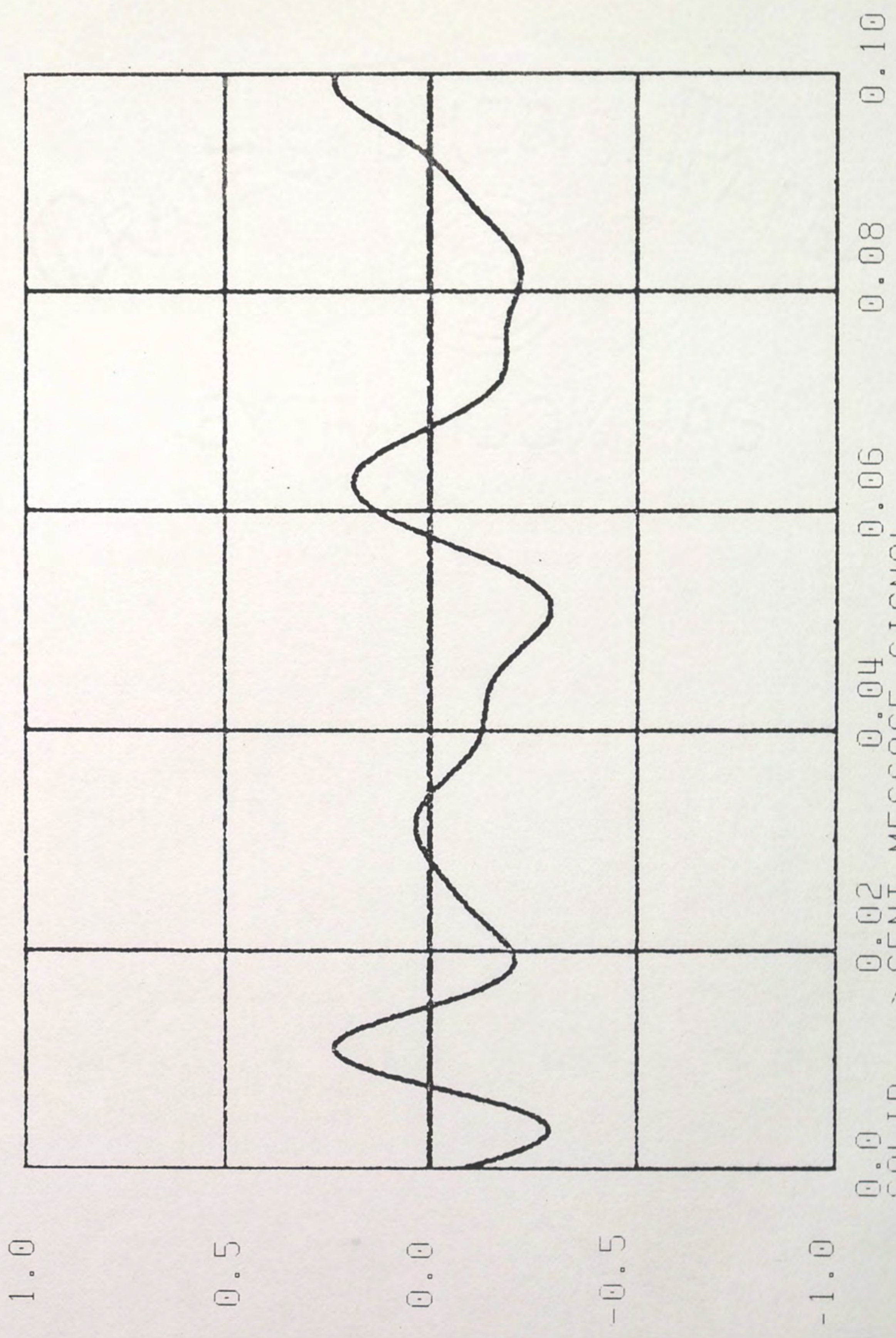

Figure 2. Sample Message Signal To Be Sent

signals. The system is adaptable to digital and/or modulated intelligence as well. Baseband CW signal transmission was chosen to keep the subject matter uncluttered with extraneous communication topics which do not shed light on the target topic.

Figure 3 shows both the gating signal and the gated message signal within the transmitter. The gating signal is the output of the gating oscillator. The gated message signal is the output of the subsequent multiplier and is transmitted through the channel to the receiver. Note that the gated signal is gated off when the gate signal is in its low state and gated on when the gate signal is in its high state.

Figure 4 is the total noise input to the receiver. This noise is band limited white Gaussian noise independent of the message source. The bandwidth of the noise is set by the user of the simulation.

After the received signal is gated, it is then processed by a low pass filter. This is the "noise estimation filter" shown in Figure 1. This filter removes the high frequency noise and the gated characteristic of the noise signal. It also has a gain associated with it which is a function of the gating signal duty cycle "alpha". This gain is explained in Section 3 where the mathematical aspects of the system are analyzed. The output of the filter is then the noise estimation signal to be subtracted from the signal+noise output of the receive low pass filter. This filter output will be explained shortly. Figure 5 shows a comparison of the actual and estimated in-band noise.

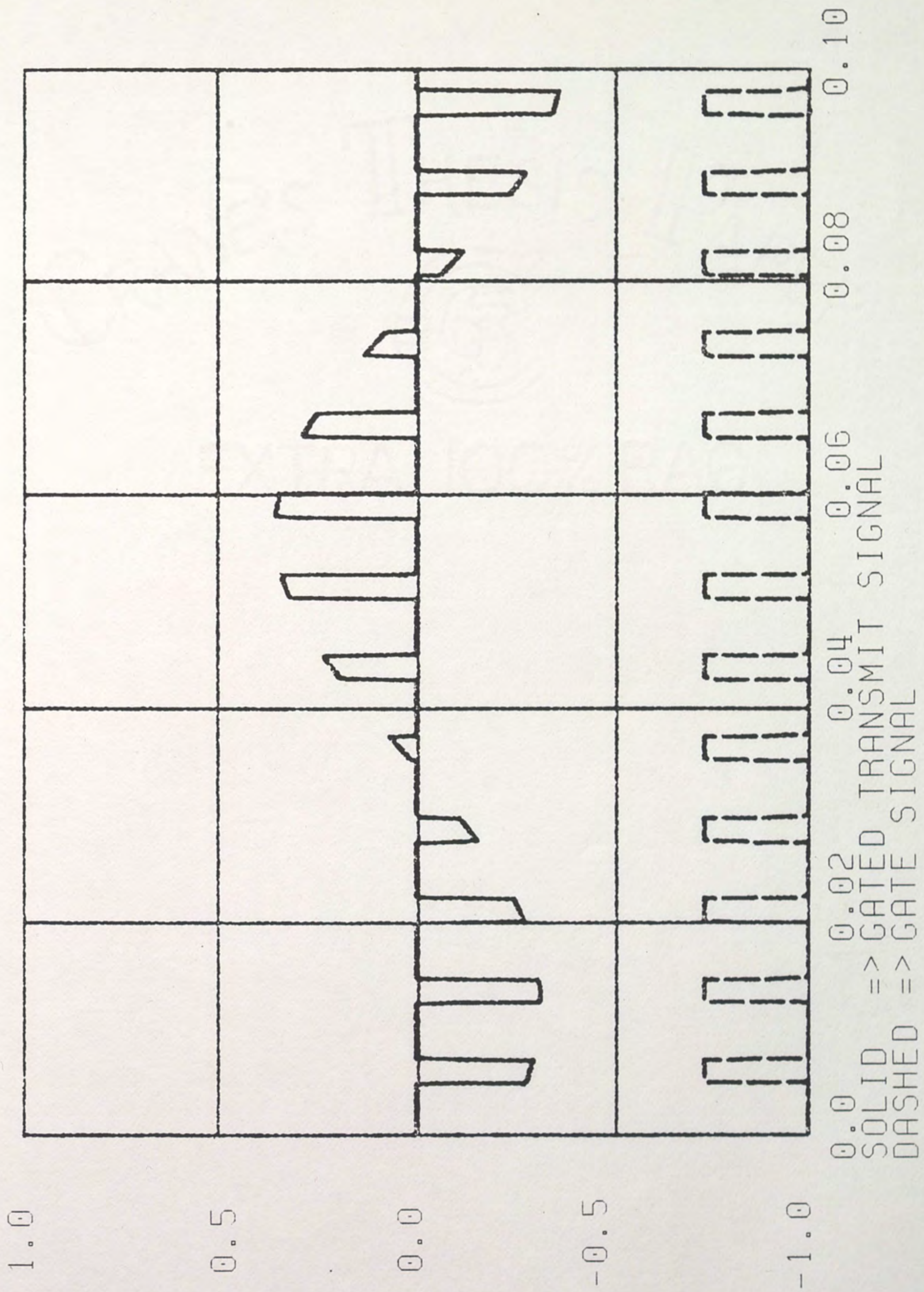

Figure 3. Gated Message And Gate Signals

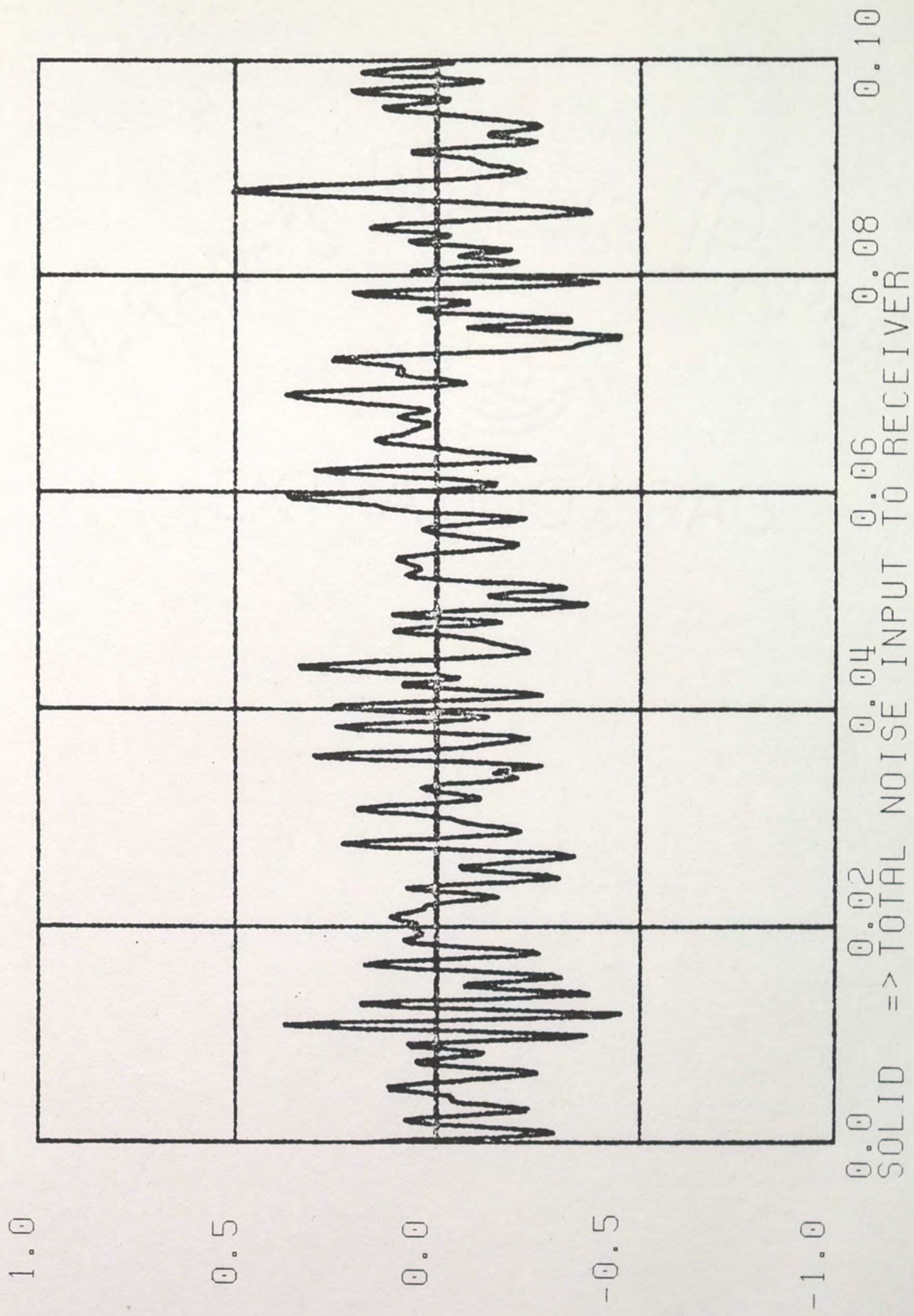

Figure 4. Total Noise Input To Receiver

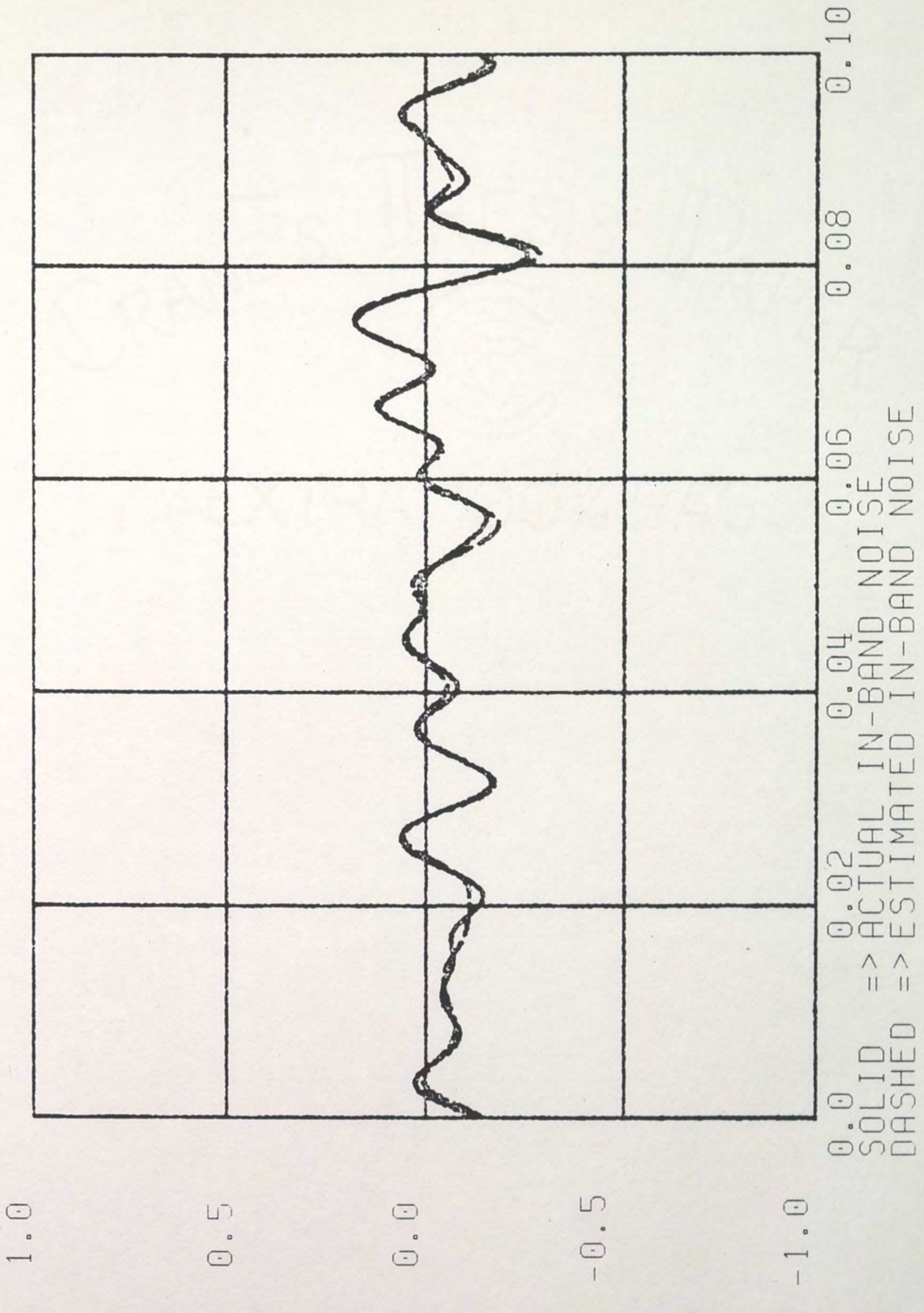

Figure 5. Actual Noise And Noise Estimate

Figure 6 shows the message signal and the output of the low pass receive filter. The input to this filter is the received gated signal+wideband noise. The output is what would be provided by a conventional receiver without a noise estimation circuit in a system where the transmitter power is equivalent to that used in this system.

The receive low pass filter performs two functions. The first is to eliminate any out-of-band noise. This is a common feature in any communications receiver. The second is to eliminate the gated characteristic of the sent gated signal. The gated characteristic of the transmitted signal resides in a series of spectral components of much higher frequency than the desired message signal, hence they are easily eliminated. There is still a spectrum of message signal components identical to that of the original message signal except for a constant gain factor. There is also a spectrum of noise introduced in the channel residing in the same spectral region as the message information. This filter cannot eliminate that noise.

Due to the transfer function of the receive filter, frequency dependent group delay and gain characteristics process the signal and noise passing through it. These characteristics should not affect the message signal significantly in any way other than the addition of a linear phase delay, and hence a constant group delay. Since the filter has these traits, the noise will likewise be affected. Therefore, in order

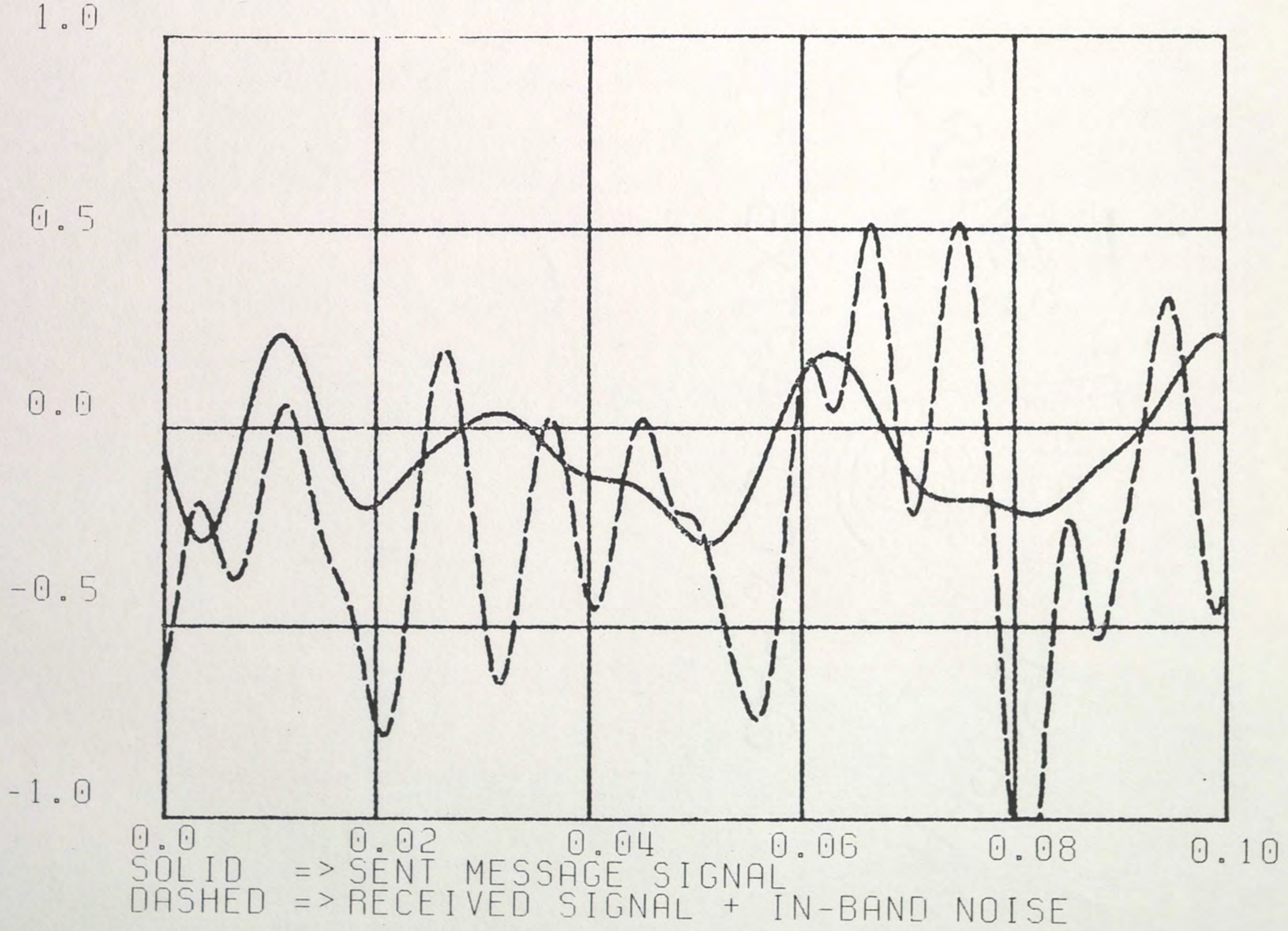

Figure 6. Signal + In-Band Noise And Message Signal

to subtract the noise estimation from the receive filter processed noise, the noise estimation should be subject to identical processing. This creates a situation where a noise component is delayed and attenuated through the receive low pass filter and its estimate is likewise delayed and attenuated by the same factors. Therefore, the system does not yield a time or gain skew between the actual and estimated noise signals. This is accomplished by utilizing identical filters for noise estimation and low pass receiving. This is a welcome simplification. Furthermore, this can also extend to the level of using only one filter with differencing done on its inputs rather than two identical filters differencing their outputs.

Figure 7 shows the message signal and the output of the entire signal recovery unit. At this point, the noise estimate has been subtracted from the signal+noise value. Note how much more correlated the recovered signal is to the message signal than the signal+noise value is of Figure 6. This is the intent of the noise estimation system.

System Practicality

The system described so far may seem to give something for nothing: a clean signal through a noisy channel. Shannon's information theory predicts a finite limit to the rate of intelligence that can be recovered through a given bandlimited channel as a function of noise

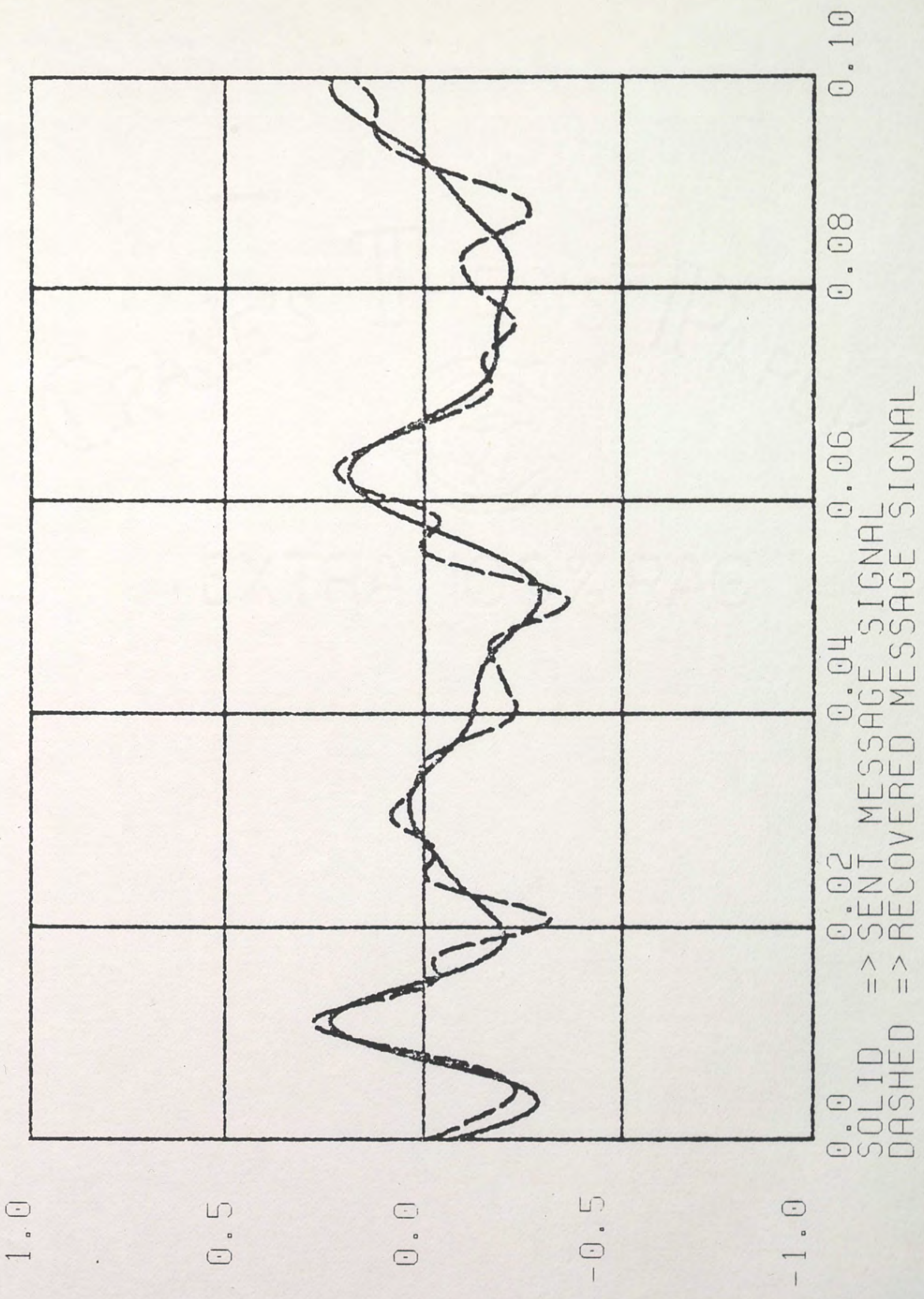

Figure 7. Cleaned Signal And Message Signal

power (Shanmugam 1979). Yet the noise estimator seems to eliminate the effects of noise due to the channel. The key to what seems to be a contradiction to that theory is that so far the channel has not been band limited. Therefore, in agreement with Shannon's theory, the information rate is likewise unlimited. Further, the gating circuitry of the receiver causes noise from bands outside the message spectrum to be modulated down into the message spectrum. Therefore, what could be transmitted cleanly through a channel with no noise in the message spectral occupancy may be severely corrupted by out-of-band noise.

The first realization that an infinite bandwidth channel be made available constrains the system to use in dedicated channel systems such as a wire, magnetic recording tape, or laser beam. The second consideration that out-of-band noise may contaminate the recovered signal suggests that some circumstances could preclude the use of the system even in dedicated channels where there is a preponderance of out-of-band noise.

Along with these considerations it must be kept in mind that the purpose of the study was to show that the system could be used to separate noise from signal, even though it may be confined to a limited number of practical situations.

As far as hardware complexity goes, the system requires no new circuitry technologies or advancements. The gating process only requires a high speed gate or mixer, as does the complementary gating at the receiver. Any of a multitude of existing filtering techniques may be used in the receiver. The only complexity in the entire system is the phase locked loop required at the receiver for gate timing recovery. Taking the absolute value of the received signal provides a strong spectral component at the gating frequency. It should be noted that the simulation of the system did not include a timing recovery module. The gate timing of the receiver was taken directly from the transmitter in order to allocate more time to the study of the newer concepts.

SECTION III
MATHEMATICAL ANALYSIS

This section contains a mathematical description of the noise estimation concept and the signals present in the various sections of the system. Also presented is a closed form expression for the improvement in signal to noise ratio at the receiver output over a system transmitting the same average power without a noise estimator included. The description will proceed in a logical order from the transmitter input to the receiver output. Frequency domain techniques are used throughout. Figure 8 on page 22 provides a visual guide to system variable locations. Pertinent variables are shown in both time and frequency domain nomenclature. Lower case indicates time domain and upper case is used for frequency domain. Table 1 provides a tabular description of the signals associated with both the time and frequency domains. Also provided is a list of other system variables and their respective functions.

Two notes of clarification are in order. First, the message signal is assumed to be a general, nonperiodic signal to show no dependence on message periodicity. A band limited white Gaussian noise source is used to provide a message signal in the simulation. Secondly, all random processes such as message signal to be sent and channel noise are assumed to be stationary zero-mean band limited white Gaussian. Several places in this section use the frequency domain representaion of these signals. This requires these signals to be Fourier transformable utilizing the standard Fourier transform equation 3-1.

$$\mathfrak{F}[f(t)] = \int_{-\infty}^{+\infty} f(t')[\cos(\omega t')-j\sin(\omega t')]dt' \qquad (3\text{-}1)$$

It could be argued that these signals are power signals rather than energy signals and their respective Fourier transforms do not exist since the integral of equation 3-1 might not be finite. The use of these frequency domain quantities is justified by the following consideration. The system can be utilized for any finite period of time, say from t_1 to t_2. Therefore, each of the signals which are present throughout the system would exist only during that period of time and are zero at any time before t_1 or after t_2. This allows any of them to be considered energy signals and their Fourier transforms can exist. This follows since equation 3-1 would be time limited as in equation 3-2 below.

$$\mathfrak{F}[f(t)] = \int_{t_1}^{t_2} f(t')[\cos(\omega t')-j\sin(\omega t')]dt' \qquad (3\text{-}2)$$

TABLE 1.

VARIABLE NAMES

TIME DOMAIN	DESCRIPTION	FREQUENCY DOMAIN
$m(t)$	message signal to be transmitted	$M(\omega)$
$g(t)$	gating signal	$G(\omega)$
$g'(t)$	complement of gating signal	$G'(\omega)$
$s(t)$	transmitted signal	$S(\omega)$
$n(t)$	noise signal which is summed with transmitted signal	$N(\omega)$
$i(t)$	receiver input signal	$I(\omega)$
$v_1(t)$	gated received signal	$V_1(\omega)$
$v_2(t)$	low passed receive signal	$V_2(\omega)$
$n'(t)$	estimated noise	$N'(\omega)$
$r(t)$	recovered signal/receiver	$R(\omega)$

ω_m = single sided bandwidth of message signal in rad/sec

ω_r = single sided bandwidth of receiver low pass filters in rad/sec

ω_n = single sided bandwidth of noise signal in rad/sec

ω_g = fundamental gating signal frequency in rad/sec

G_n = n^{th} Fourier series coefficient of gating signal

α = 'alpha'; duty cycle of gating signal; equivalent to G_0

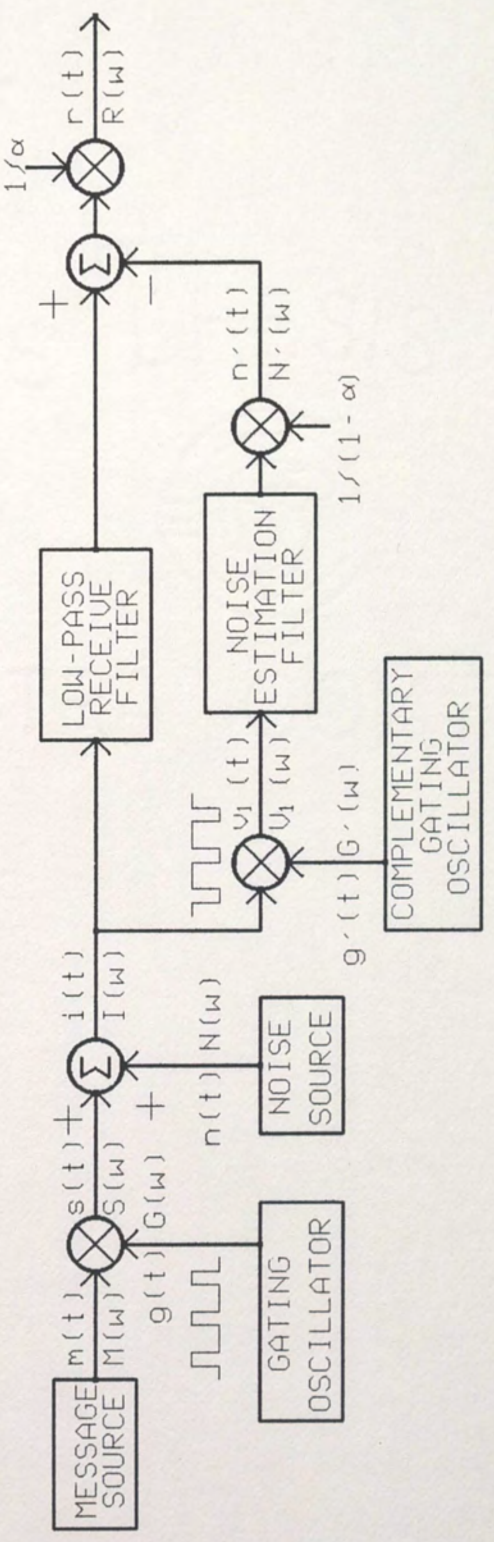

Figure 8. Variable Placement In System

Signal Descriptions

The message signal is modeled band limited white with a Gaussian probability density function. The frequency domain representation of this signal could be written as in equation 3-3 and is depicted in Figure 10 a.

$$M(\omega) = \begin{cases} 1 & |\omega| \leq \omega_m \\ 0 & |\omega| > \omega_m \end{cases} \tag{3-3}$$

The message signal is then multiplied by g(t), the gating function. The function g(t), shown in Figure 9, is a rectangular wave whose duty cycle may be selected by the user. Since this function is periodic, it may be rerpresented in the time domain by its Fourier series expansion as in

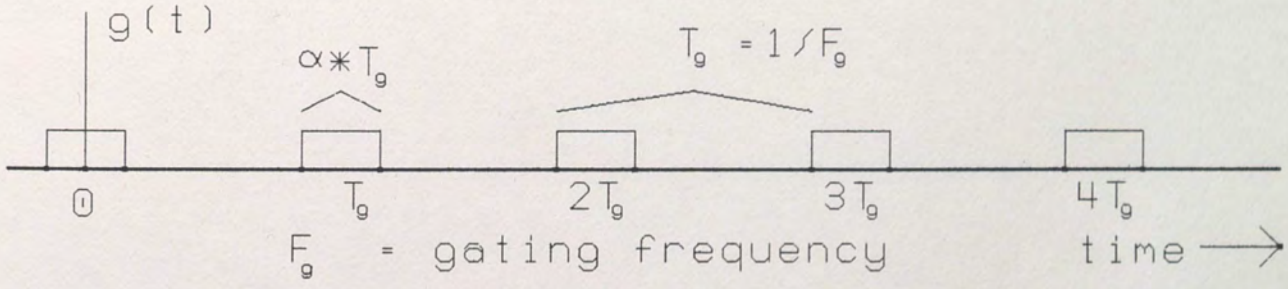

Figure 9. Gating Function

equations (3-4) a and b. The Fourier transform of the gating function

$$g(t)=\sum_{n=-\infty}^{+\infty} G_n \cdot [\cos(n\omega_g t)+j\sin(n\omega_g t)] \qquad (3\text{-}4)\text{b}$$

$$G_n=[\sin(n\pi\alpha)]/[n\pi] \qquad (3\text{-}4)\text{b}$$

is derived from the Fourier series of the gating function by the standard method for transforming a periodic function, yielding equation 3-5 (Lathi 1974, Beyer 1978). The Fourier spectrum of g(t) is depicted in Figure 10 b.

$$G(\omega)=2\pi\sum_{n=-\infty}^{+\infty} G_n\delta(\omega-n\omega_g) \qquad (3\text{-}5)$$

The transmitted, or sent signal, s(t), is developed by multiplying the message signal, m(t), by the gating signal, g(t) as in equation 3-6. The Fourier spectrum of s(t), $S(\omega)$, is given in equation 3-7. Note that the '$*$' sign indicates convolution. This convolution is carried out in the following development.

$$s(t)=m(t)\cdot g(t) \qquad (3\text{-}6)$$

$$S(\omega)=M(\omega)*G(\omega) \qquad (3\text{-}7)$$

$$S(\omega)=[1/(2\pi)]\int_{-\infty}^{+\infty} G(\omega-\omega')\cdot M(\omega')d\omega'$$

Using 3-5:

$$G(\omega)=2\pi\sum_{n=-\infty}^{+\infty} G_n\delta(\omega-n\omega_g)$$

$$G(\omega-\omega')=2\pi\sum_{n=-\infty}^{+\infty} G_n\delta(\omega-\omega'-n\omega_g)$$

$$S(\omega)=[1/(2\pi)]\int_{-\infty}^{+\infty}[2\pi\sum_{n=-\infty}^{+\infty} G_n\delta(\omega-\omega'-n\omega_g)]\cdot M(\omega')d\omega'$$

$$S(\omega)=\sum_{n=-\infty}^{+\infty}\int_{-\infty}^{+\infty}[G_n\delta(\omega-\omega'-n\omega_g)\cdot M(\omega')]d\omega'$$

Use of the sifting property yields the final expression for S(ω).

$$S(\omega)=\sum_{n=-\infty}^{+\infty} G_n M(\omega-n\omega_g) \qquad (3\text{-}8)$$

Although the above expression is a summation, only one of the summands contribute to the result for any one particular value of ω. This is because ω_g must be at least twice as great as ω_m to reconstruct the sent gated signal at the receiver without aliasing. The result is illustrated in Figure 10 c. Note how the sidelobes of the resulting spectrum correspond in both placement and amplitude with the gating signal spectrum of 11 b. This is expected since the gating signal is comprised of a series of sinusoids and each of these sinusoids modulates the message signal. Each sinusoidal component translates a respective message bandwidth to a point centralized on its own frequency.

The signal input to the recovery unit (receiver) is denoted as i(t). It is the summation of the sent signal, s(t), and any channel noise, n(t). The time and frequency domain equations are provided in equations 3-9 a, b, and c.

$$i(t)=s(t)+n(t) \tag{3-9)a}$$

$$I(\omega)=S(\omega)+N(\omega) \tag{3-9)b}$$

$$I(\omega)=\sum_{n=-\infty}^{+\infty} G_n M(\omega-n\omega_g)+N(\omega) \tag{3-9)c}$$

The signal input to the receiver takes two paths as can be seen in figures 1 and 8. One path is through a recieve low pass filter, and another is through the noise estimation filter. The noise estimation path

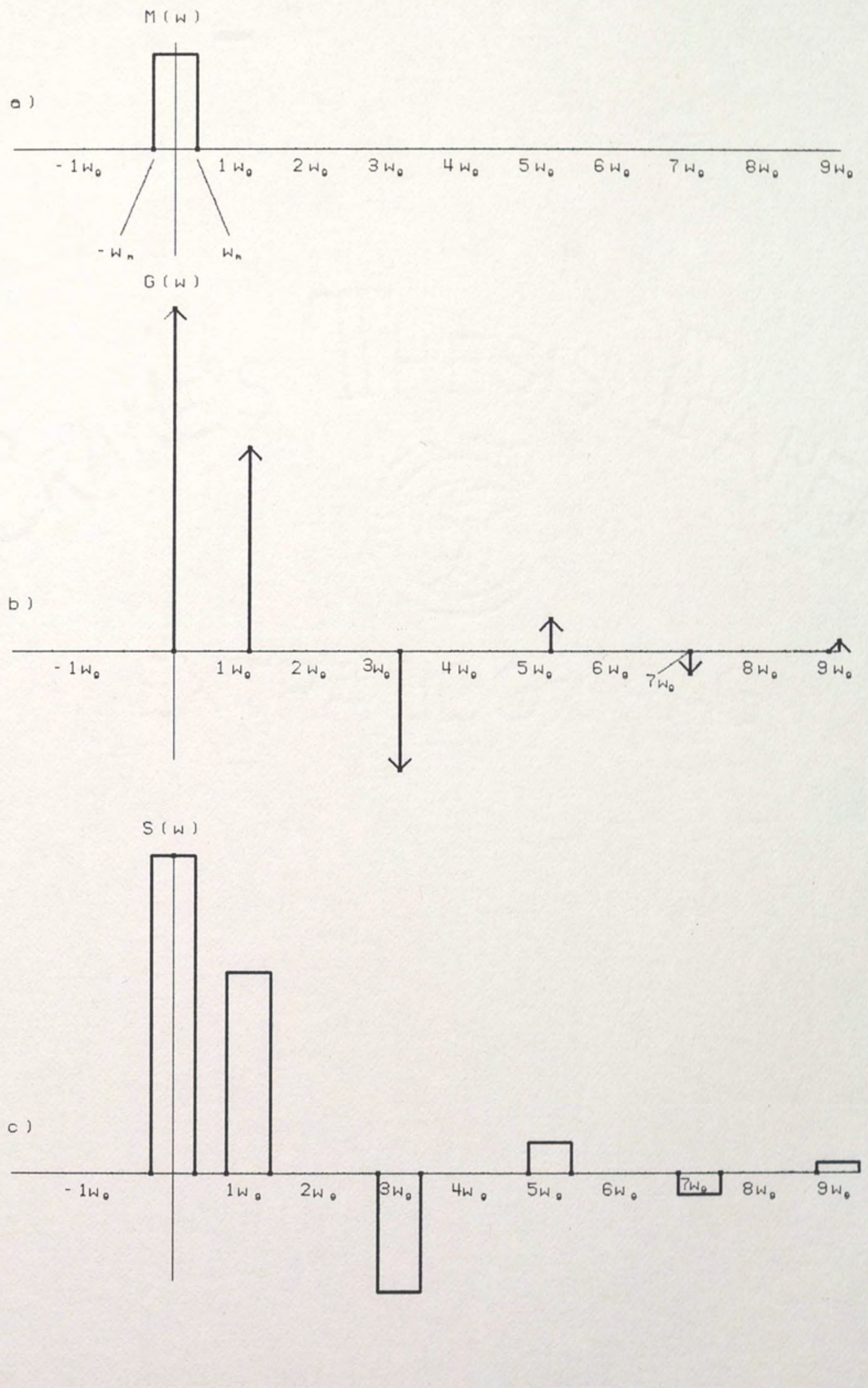

Figure 10. Fourier Representations Of Transmitter Signals

will be analyzed first. The received signal, i(t), is multiplied by a rectangular wave which is the complement of the gating function. This complementary function is called g'(t). Equations 3–10 a, b, and c are the time and frequency domain representations of g'(t).

$$g'(t)=1-g(t) \qquad (3\text{–}10)a$$

$$G'(\omega)=2\pi\delta(\omega)-G(\omega) \qquad (3\text{–}10)b$$

$$G'(\omega)=2\pi\delta(\omega)-2\pi\sum_{n=-\infty}^{+\infty}G_n\delta(\omega-n\omega_g) \qquad (3\text{–}10)c$$

The output of the multiplier is given the name $v_1(t)$.

$$v_1(t)=i(t)\cdot g'(t) \qquad (3\text{–}11)a$$

$$V_1(\omega)=I(\omega)*G'(\omega) \qquad (3\text{–}11)b$$

This convolution is carried out below.

$$V_1(\omega)=[1/(2\pi)]\int_{-\infty}^{+\infty}I(\omega')\cdot G'(\omega-\omega')d\omega' \qquad (3\text{–}12)$$

$$=[1/(2\pi)]\int_{-\infty}^{+\infty}[S(\omega')+N(\omega')]\cdot[2\pi\delta(\omega-\omega')-2\pi\sum_{n=-\infty}^{+\infty}G_n\delta(\omega-\omega'-n\omega_g)]d\omega'$$

$$=\int_{-\infty}^{+\infty}[\sum_{n=-\infty}^{+\infty}G_nM(\omega'-n\omega_g)+N(\omega')]\cdot[\delta(\omega-\omega')-\sum_{n=-\infty}^{+\infty}G_n\delta(\omega-\omega'-n\omega_g)]d\omega'$$

$$V_1(\omega)=\sum_{n=-\infty}^{+\infty}G_nM(\omega'-n\omega_g)]\cdot\delta(\omega-\omega')d\omega'$$

$$+\int_{-\infty}^{+\infty}N(\omega')\delta(\omega-\omega')d\omega'$$

$$-\int_{-\infty}^{+\infty}[\sum_{n=-\infty}^{+\infty}G_nM(\omega'-n\omega_g)]\cdot[\sum_{k=-\infty}^{+\infty}G_k\delta(\omega-k\omega_g-\omega')]d\omega'$$

$$-\int_{-\infty}^{+\infty}N(\omega')\sum_{n=-\infty}^{+\infty}G_n\delta(\omega-n\omega_g-\omega')d\omega' \qquad (3\text{-}13)$$

Equation 3-13 is comprised of four integrals. These will be simplified individually below.

First integral:

$$\int_{-\infty}^{+\infty}\Big[\sum_{n=-\infty}^{+\infty} G_n M(\omega'-n\omega_g)\Big]\cdot\delta(\omega-\omega')d\omega'$$

$$=\sum_{n=-\infty}^{+\infty}\Big\{G_n\int_{-\infty}^{+\infty}[M(\omega'-n\omega_g)\cdot\delta(\omega-\omega')]d\omega'\Big\} \qquad = \sum_{n=-\infty}^{+\infty} G_n M(\omega-n\omega_g)$$

Second integral:

$$\int_{-\infty}^{+\infty} N(\omega')\delta(\omega-\omega')d\omega' \qquad = N(\omega)$$

Third integral:

$$-\int_{-\infty}^{+\infty}[\sum_{n=-\infty}^{+\infty} G_n M(\omega'-n\omega_g)]\cdot[\sum_{k=-\infty}^{+\infty} G_k \delta(\omega-k\omega_g-\omega')]d\omega'$$

$$=-\int_{-\infty}^{+\infty}[\sum_{n=-\infty}^{+\infty}\sum_{k=-\infty}^{+\infty} G_n G_k M(\omega'-n\omega_g)\cdot\delta(\omega-k\omega_g-\omega')d\omega']$$

$$=-\sum_{n=-\infty}^{+\infty}\sum_{k=-\infty}^{+\infty} G_n G_k \int_{-\infty}^{+\infty} M(\omega'-n\omega_g)\cdot\delta(\omega-k\omega_g-\omega')d\omega'$$

$$= -\sum_{n=-\infty}^{+\infty}\sum_{k=-\infty}^{+\infty} G_n G_k M(\omega-k\omega_g-n\omega_g)$$

Fourth integral:

$$-\int_{-\infty}^{+\infty} N(\omega')\sum_{n=-\infty}^{+\infty} G_n \delta(\omega-n\omega_g-\omega')d\omega'$$

$$-\sum_{n=-\infty}^{+\infty} G_n \int_{-\infty}^{+\infty} N(\omega')\delta(\omega-n\omega_g-\omega')d\omega' \qquad = -\sum_{n=-\infty}^{+\infty} G_n N(\omega-n\omega_g)$$

Combining the four expressions in their simplified form yields the general expression fo $V_1(\omega)$ in equation 3-14.

$$V_1(\omega)=N(\omega)-\sum_{n=-\infty}^{+\infty} G_n N(\omega-n\omega_g)+\sum_{n=-\infty}^{+\infty} G_n M(\omega-n\omega_g)$$

$$-\sum_{n=-\infty}^{+\infty}\sum_{k=-\infty}^{+\infty} G_n G_k M(\omega-k\omega_g-n\omega_g) \qquad (3\text{-}14)$$

Equation 3-14 contains two summations which are functions of the sent message signal. Since there are two switches in series between the message signal generator and V_1, and there is no time in which they are both simultaneously on, it would seem that no message signal could reach V_1. In fact, it doesn't. The two terms cancel out exactly, but the proof that there is no message signal in the noise estimate, which is a function of V_1, is much simpler than trying to show that V_1 itself is not a function of the message signal. This proof is carried out next.

The noise estimate, $N'(\omega)$, is developed by processing $V_1(\omega)$ through a unity gain low pass filter, $H_R(\omega)$, and multiplying by constant. $H_R(\omega)$ is shown in Figure 11. This is the noise estimation filter of Figure 8.

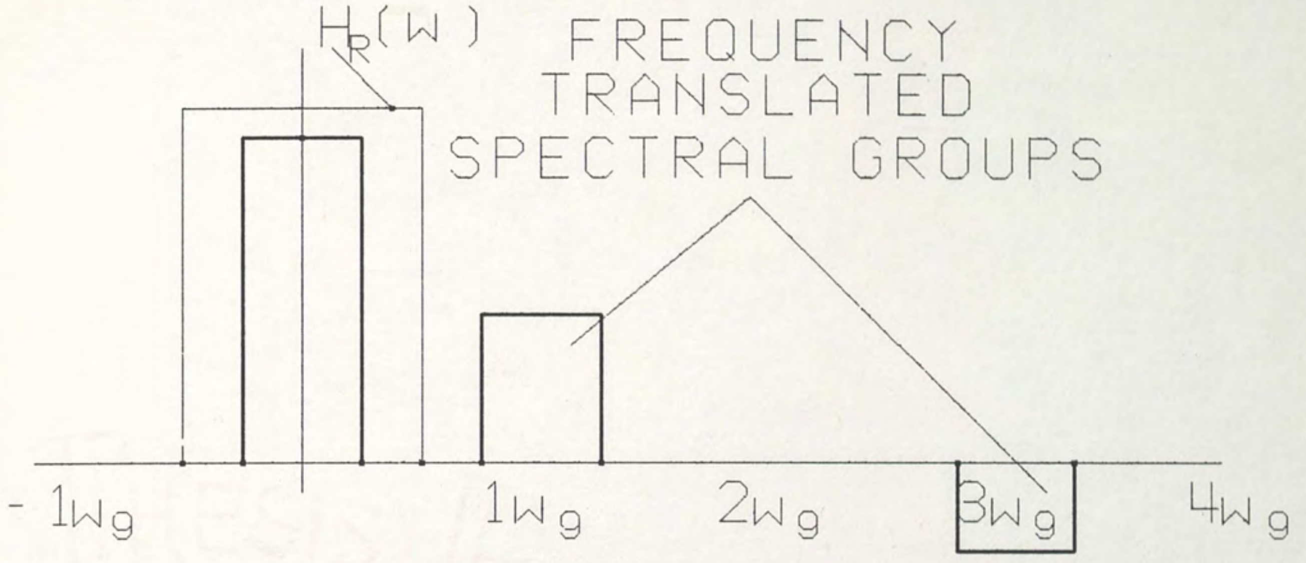

Figure 11. Receiver Low Pass Filters

The composition of N'(ω) is stated in equation form in equation 3–15.

$$N'(\omega) = H_R(\omega)\cdot V_1(\omega)/(1-\alpha) \qquad (3\text{–}15)$$

Equation 3–14 contains terms such as $M(\omega - n\omega_g)$. These terms can be viewed as message signal spectral component groups translated in frequency to locations centered around multiples of ω_g. It can be seen that $H_R(\omega)$ eliminates any spectral groups centered around non–zero multiples of ω_g. Therefore, the following approximation may be used.

$$N'(\omega) \approx \begin{cases} H_R(\omega)\cdot V_1(\omega)/(1-\alpha) & |\omega| \le \omega_r \\ 0 & |\omega| > \omega_r \end{cases} \tag{3-16}$$

For $|\omega| \le \omega_r$:

$$M(\omega-n\omega_g) \quad = \begin{cases} M(\omega) & n=0 \\ 0 & n<>0 \end{cases} \tag{3-17)a}$$

$$M(\omega-n\omega_g-k\omega_g) \quad =. \begin{cases} M(\omega) & n=0 \\ 0 & n<>0 \end{cases} \tag{3-17)b}$$

Using equations 3-14 through 3-17b, the following expression for the noise estimation is developed.

$$N'(\omega)=H_R(\omega)\cdot [G_0M(\omega)+N(\omega)-\sum_{n=-\infty}^{+\infty} G_nG_{-n}M(\omega)$$

$$-\sum_{n=-\infty}^{+\infty} G_nN(\omega-n\omega_g)]/(1-\alpha) \tag{3-18}$$

To proceed further it is necessary to prove that G_n and G_{-n} are equal.

$$G_n=\sin(n\pi\alpha)/(n\pi)$$

$$G_{-n}=\sin(-n\pi\alpha)/(-n\pi)=-\sin(n\pi\alpha)/(-n\pi)$$

$$=\sin(n\pi\alpha)/(n\pi) \qquad = G_n \tag{3-19}$$

Using 3–19, 3–18 can be simplified.

$$N'(\omega)=H_R(\omega)\cdot[M(\omega)(G_0-\sum_{n=-\infty}^{+\infty}G_n{}^2)+N(\omega)$$

$$-\sum_{n=-\infty}^{+\infty}G_nN(\omega-n\omega_g)]/(1-\alpha) \qquad (3\text{–}20)$$

The following identity is proven in Appendix A.

$$G_0=\sum_{n=-\infty}^{+\infty}G_n{}^2$$

Using this identity, 3–20 can be further reduced.

$$N'(\omega)=H_R(\omega)\cdot[N(\omega)-\sum_{n=-\infty}^{+\infty}G_nN(\omega-n\omega_g)]/(1-\alpha)$$

(3–21) Inspection of equation 3–19 shows that G_0 is equivalent to α, the gating signal duty cycle. Using this, 3–21 is rewritten.

$$N'(\omega)=H_R(\omega)N(\omega)-H_R(\omega)\sum_{\substack{n=-\infty \\ n\neq 0}}^{+\infty} G_n N(\omega-n\omega_g)/(1-\alpha) \qquad (3\text{-}22)$$

Equation 3-22 is the final general form for the noise estimate. It contains no message signal components as expected. Further, if there is no noise outside of the bandwidth of the message signal, equation 3-22 reduces to the receive low pass transfer function times the noise exactly. The next step is to show how the noise estimate is used to eliminate noise in the receive signal. $V_2(\omega)$ is the output of the receive low pass filter as seen in Figure 8.

$$V_2(\omega)=H_R(\omega)\cdot I(\omega)$$

$$V_2(\omega)=H_R(\omega)\cdot[\ S(\omega)\ +\ N(\omega)\ ]$$

$$V_2(\omega)=H_R(\omega)\cdot[N(\omega)+\sum_{n=-\infty}^{+\infty} G_n M(\omega-n\omega_g)] \qquad (3\text{-}23)$$

Inspection of Figure 11 shows that H_R can only pass the message information centered around DC. All of the signal translated to locations centered around multiples of ω_g is eliminated. Thus, the following approximation will be made.

$$V_2(\omega)=H_R(\omega)\cdot[N(\omega)+G_0M(\omega)] \qquad (3\text{-}23)$$

The recovery unit output, $R(\omega)$, is produced by subtracting the noise estimate, $N'(\omega)$, from $V_2(\omega)$ and dividing the result by α.

$$R(\omega)=[V_2(\omega)-N'(\omega)]/\alpha$$

$$R(\omega)=\{H_R(\omega)N(\omega)+H_R(\omega)G_0M(\omega)-H_R(\omega)N(\omega)$$

$$+[1/(1-\alpha)]H_R(\omega)\sum_{\substack{n=-\infty\\n\neq 0}}^{+\infty}G_nN(\omega-n\omega_g)\}/\alpha$$

$$R(\omega)=\{H_R(\omega)G_0M(\omega)+[1/(1-\alpha)]H_R(\omega)\sum_{\substack{n=-\infty\\n\neq 0}}^{+\infty}G_nN(\omega-n\omega_g)\}/\alpha$$

Assuming that H_R has sufficient bandwidth to pass the message signal without affecting it significantly*, the expression may be reduced further.

*The filter may add an approximately constant group delay without being considered as significantly affecting the signal. This is true in most communications systems.

$$R(\omega)=M(\omega)+\{1/[\alpha(1-\alpha)\}H_R(\omega)\sum_{\substack{n=-\infty \\ n\neq 0}}^{+\infty} G_n N(\omega-n\omega_g))\} \qquad (3\text{-}24)$$

The expression shows that the message signal is recovered and that noise from other spectral regions has been brought in. If the effect of the noise in the other bands is less than the effect of noise in the in-band spectrum, there is an overall improvement. Usually this is the case if the duty cycle, α, and the gating frequency are chosen properly. Note again that if the noise is confined to the spectral region of the message signal, it is completely eliminated. This is because all of the $N(\omega-n\omega_g)$ terms in the summation are zero for $n<>0$.

Next, an expression for the improvement in signal-to-noise ratio, SNR, will be developed to compare a system with a noise estimator to one without a noise estimator on the basis that both system transmitters emit the same average output power.

First, an expression for the average signal power (not noise power) at the output of the receiver will be developed. This will be done by integrating over the squared amplitude of the Fourier spectrum of the signal portion of the signal+noise output developed in equation 3-24. This is justified on the basis that the signal is a stationary process. The result will be denoted as $\hat{P}_{RS}$, for average signal power at the recovery unit output.

$$\hat{P}_{RS} = \int_{-\infty}^{+\infty} \left| M(\omega) \right|^2 d\omega \tag{3-25}$$

The average transmitted power, $\hat{P}_t$, is calculated by showing that the message signal, m(t), is turned on $\alpha \cdot 100\%$ of the time. Again, α is the gating signal duty cycle. Therefore, the transmitted power is α^2 times what the transmitted power would be had there been no gating taking place.

$$\hat{P}_T = \alpha^2 \cdot \int_{-\infty}^{+\infty} \left| M(\omega) \right|^2 d\omega \tag{3-26}$$

The value $\hat{P}_T$ is the average amount of power sent from the transmitter of the described system. This value will be used for both the noise estimator and non-noise estimator systems for transmitted power.

The noise power at the receiver output is calculated by taking the noise expression of equation 3-24 and integrating the square of its absolute value over ω. This will be denoted as $\hat{P}_{RN}$, for recovery unit noise output.

$$\hat{P}_{RN}=\int_{-\infty}^{+\infty} 1/[\alpha^2(1-\alpha)^2 \left|H_R(\omega)\right|^2 \cdot \left|\sum_{\substack{n=-\infty \\ n\neq 0}}^{+\infty} G_n N(\omega-n\omega_g)\right|^2 d\omega$$

$$\hat{P}_{RN}=1/[\alpha^2(1-\alpha)^2 \int_{-\infty}^{+\infty} \left|H_R(\omega)\right|^2 \left|\sum_{\substack{n=-\infty \\ n\neq 0}}^{+\infty} G_n N(\omega-n\omega_g)\right|^2 d\omega] \qquad (3\text{-}27)$$

The signal-to-noise ratio at the recovery unit output is denoted as SNR_R and is taken from equations 3-25 and 3-27.

$$SNR_R = \hat{P}_{RS}/\hat{P}_{RN}$$

$$SNR_R = \frac{\alpha^2(1-\alpha)^2 \int_{-\infty}^{+\infty} \left|M(\omega)\right|^2 d\omega}{\int_{-\infty}^{+\infty} \left|H_R(\omega)\right|^2 \left|\sum_{\substack{n=-\infty \\ n\neq 0}}^{+\infty} G_n N(\omega-n\omega_g)\right|^2 d\omega} \qquad (3\text{-}28)$$

For the case with no noise estimation circuitry, the signal power at the output of the receive low pass filter would be the transmitted power as stated in equation 3-26. This quantity will be denoted as $\hat{P}_{NS}$ for the signal power with no noise estimation circuitry. The expression for $\hat{P}_{NS}$ is listed in equation 3–29.

$$\hat{P}_{NS} = \alpha^2 \cdot \int_{-\infty}^{+\infty} \left| M(\omega) \right|^2 d\omega \qquad (3\text{-}29)$$

The expression for the noise power at the output of the system without the noise estimation system is provided in equation 3–30 and is denoted as $\hat{P}_{NN}$.

$$\hat{P}_{NN} = \int_{-\infty}^{+\infty} \left| H_R(\omega) \right|^2 \left| N(\omega) \right|^2 d\omega \qquad (3\text{-}30)$$

The signal–to–noise ratio at the receiver low pass filter for the system with no noise estimation is given in equation 3–31. Equations 3–29 and 3–30 are used.

$$SNR_N = \hat{P}_{NS} / \hat{P}_{NN}$$

$$\mathrm{SNR_N} = \frac{\alpha^2 \int_{-\infty}^{+\infty} \left| M(\omega) \right|^2 d\omega}{\int_{-\infty}^{+\infty} \left| H_R(\omega) \right|^2 \left| N(\omega) \right|^2 d\omega} \qquad (3\text{-}31)$$

Finally, the impovement in signal-to-noise ratio is denoted as SNRIMP and is the ratio of the "signal-to-noise ratio" with noise estimation to the "signal-to-noise ratio" without noise estimation. This is expressed in equation 3-32, developed from equations 3-28 and 3-31.

$$\mathrm{SNRIMP} = \mathrm{SNR_R}/\mathrm{SNR_N}$$

$$\mathrm{SNRIMP} = \frac{(1-\alpha)^2 \cdot \int_{-\infty}^{+\infty} \left| H_R(\omega) \right|^2 \left| N(\omega) \right|^2 d\omega}{\int_{-\infty}^{+\infty} \left| H_R(\omega) \right|^2 \left| \sum_{\substack{n=-\infty \\ n\neq 0}}^{+\infty} G_n N(\omega - n\omega_g) \right|^2 d\omega} \qquad (3\text{-}31)$$

The approximation that the receive low pass filter is an ideal square low pass filter will be made here. This reduces 3-31 to the following form. The square characteristic allows the integral boundaries to become finite at $\pm\omega_R$, the positive and negative cutoff frequencies of the low pass filter.

$$\mathrm{SNRIMP} = \frac{(1-\alpha)^2 \cdot \int_{-\omega_R}^{+\omega_R} \left| N(\omega) \right|^2 d\omega}{\int_{-\omega_R}^{+\omega_R} \left| \sum_{\substack{n=-\infty \\ n\neq 0}}^{+\infty} G_n N(\omega - n\omega_g) \right|^2 d\omega} \qquad (3\text{-}32)$$

Figure 12 shows how the improvement performs as a function of α, the gating signal duty cycle. As α is decreased, the improvement increases at a cost in channel bandwidth. The numerator of equation 3-32 increases as α decreases in accordance with the behavior of the graph. The denominator terms are affected also (the G_n coefficients are functions of α). G_0 is equivalent to α, the gating signal duty cycle, and is the largest term in the denominator. This again agrees with the behavior of the equation.

There are two disadvantages in continuing to decrease α indefinitely. First an inspection of the formula for the $S(\omega)$ reveals that the smaller the α, the wider the bandwidth of the transmitted signal. The equation is repeated here for clarity. For large values of α, the G_n

$$S(\omega) = \sum_{n=-\infty}^{+\infty} G_n M(\omega - n\omega_g) \qquad (3\text{-}33)$$

values decrease rapidly with n. Secondly, a smaller value of α requires a tighter clock recovery system at the receiver.

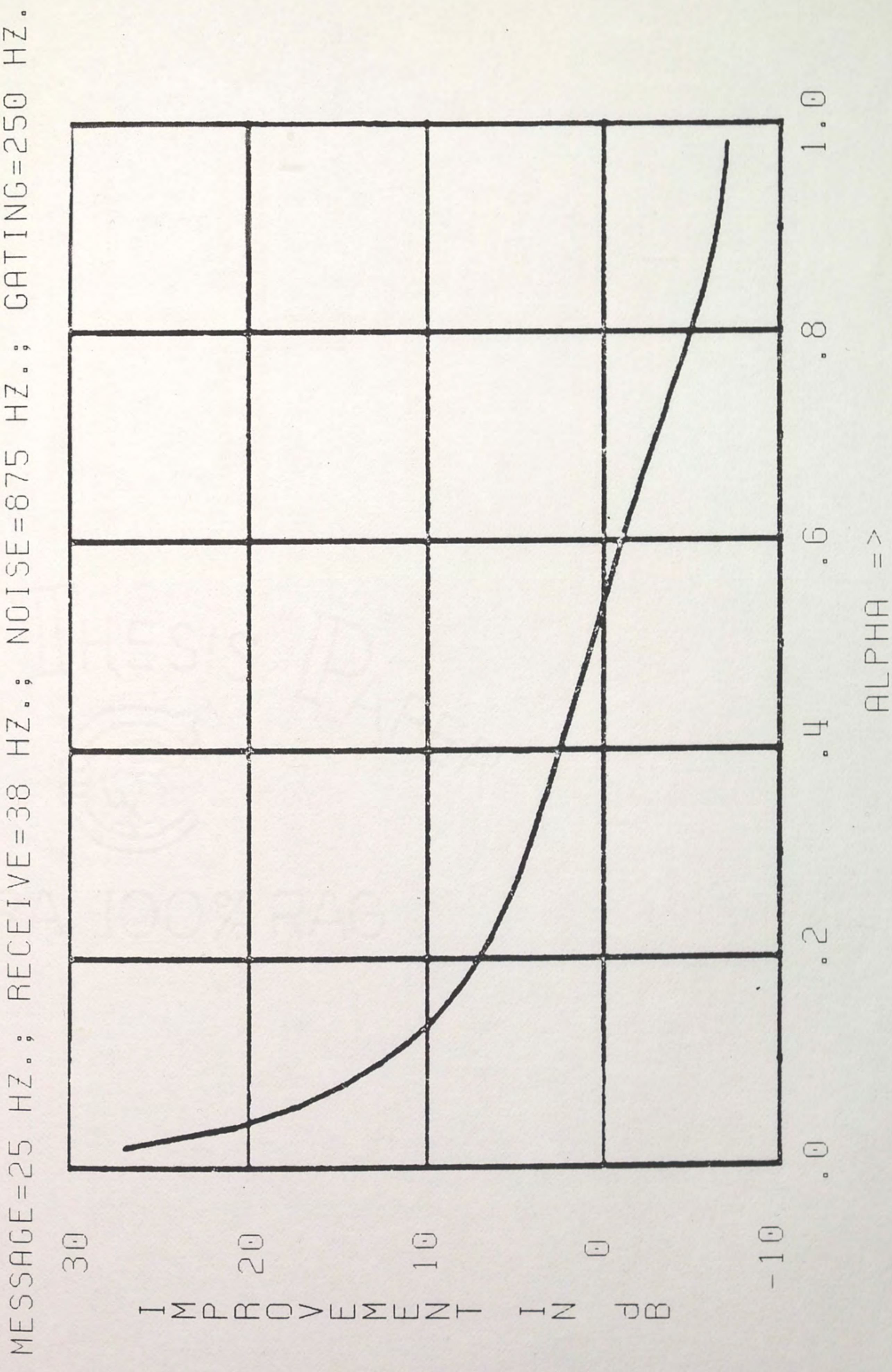

Figure 12. Theoretical System Performance

SECTION IV
THE HYBRID SIMULATION

In order to prove the theory behind any new concept, it is imperative to create a working model. This provides a means to verify mathematical results. Indeed, several mathematical errors were uncovered by use of the simulator. In addition, the simulator often displayed beneficial aspects of the noise estimator concept that were overlooked in the math analysis.

The simulator utilizes two Electronic Associates Incorporated model 781 analog computers and a Perkin Elmer model 8/32 digital computer. Figure 13 depicts the layout of the system (Gardner 1985). Figure 14 shows the schematic of the transmitter circuitry. There are several locations throughout the system where eighth order low pass filters were used. One generic type was designed and used in several locations. For each location, the user is able to individually select filter type and bandwidth. Filter types include:

1. Butterworth
2. Bessel
3. Tchebyshev

When selecting Tchebyshev, the user has control over passband ripple.

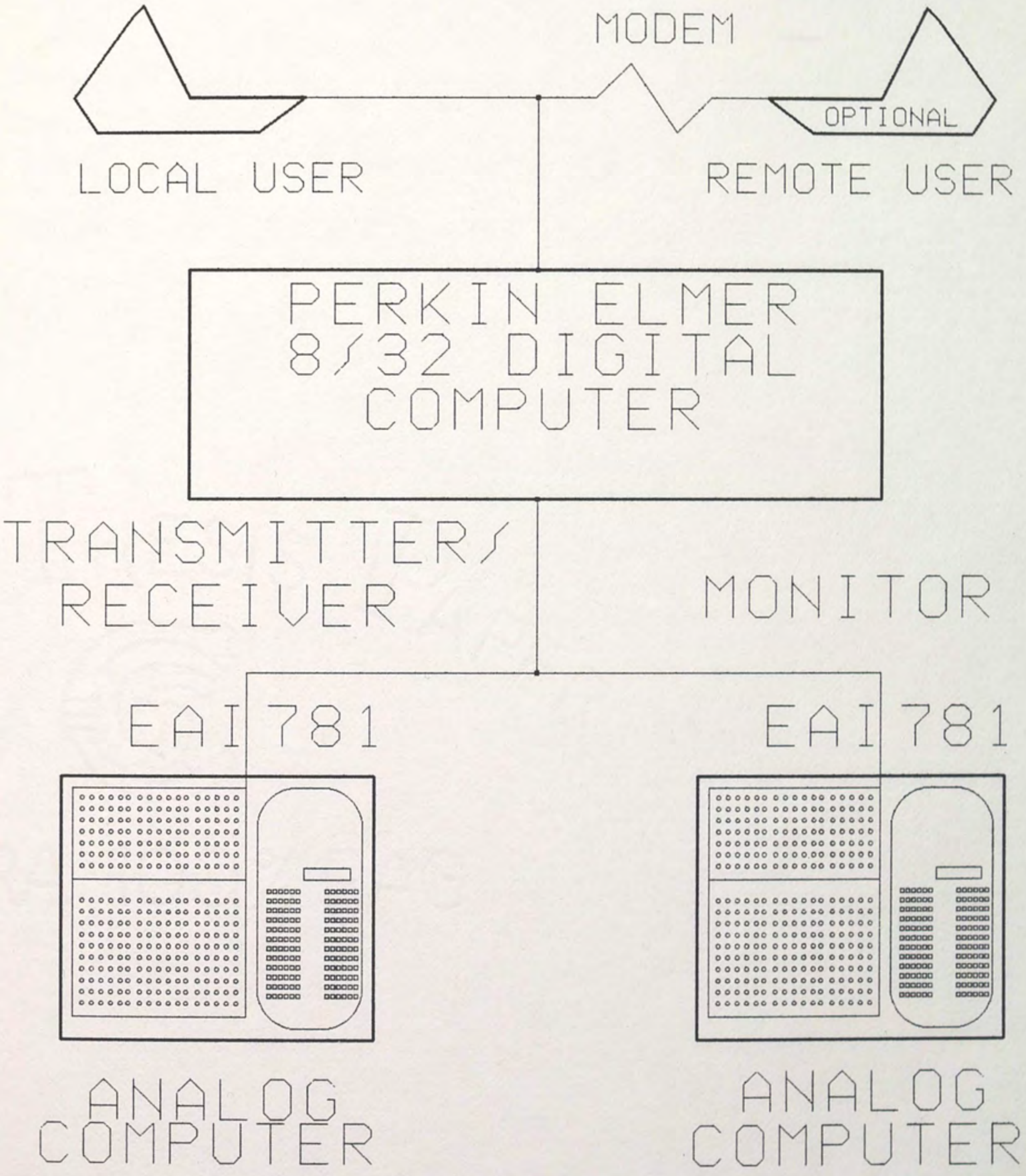

Figure 13. Simulation Equipment Layout

A schematic of the generic filter is supplied in Figure 15. Figure 16 is a schematic of the receiver circuitry. The analog portion of the simulation utilized:

* 20 operational amplifiers
* 45 time-scaled integrators
* 3 dynamic multipliers
* 109 digitally controlled attenuators
* 1 polar-to-recangular angle resolver
* 6 1-bit d/a switches
* 2 white Gaussian noise generators
* 8 14-bit a/d convertors
* 8 function relays
* 4 comparators
* 2 S-R flip flops

The analog consoles are controlled by the digital computer. The program written for the simulation is approximately 1800 lines of mixed FORTRAN and assembly level source. The assembly subroutines were necessary in time critical applications such as high speed analog to digital conversion processes. Appendix B provides a listing of the program. Section 2 provides some typical graphic outputs of time sampled signals from within the system. The plots were generated by sampling selected signals with high speed a/d convertors, storing the samples in an array, processing them, and driving plot software on a Tektronix graphic terminal. Besides sampling data, the program allows the user to:

1. control message low pass filter type and bandwidth
    a. Butterworth
    b. Bessel
    c. Tchebyshev/passband ripple
2. control noise low pass filter type and bandwidth
    a. Butterworth
    b. Bessel
    c. Tchebyshev/passband ripple
3. adjust gating frequency
4. adjust gating duty cycle, α
5. adjust signal-to-noise ratio
6. adjust sampling frequency
7. adjust receive low pass filter type and bandwidth
    a. Butterworth
    b. Bessel
    c. Tchebyshev/passband ripple
8. view power spectral densities of various signals in the system
9. predict signal-to-noise ratio improvements
10. measure signal-to-noise ratio improvements

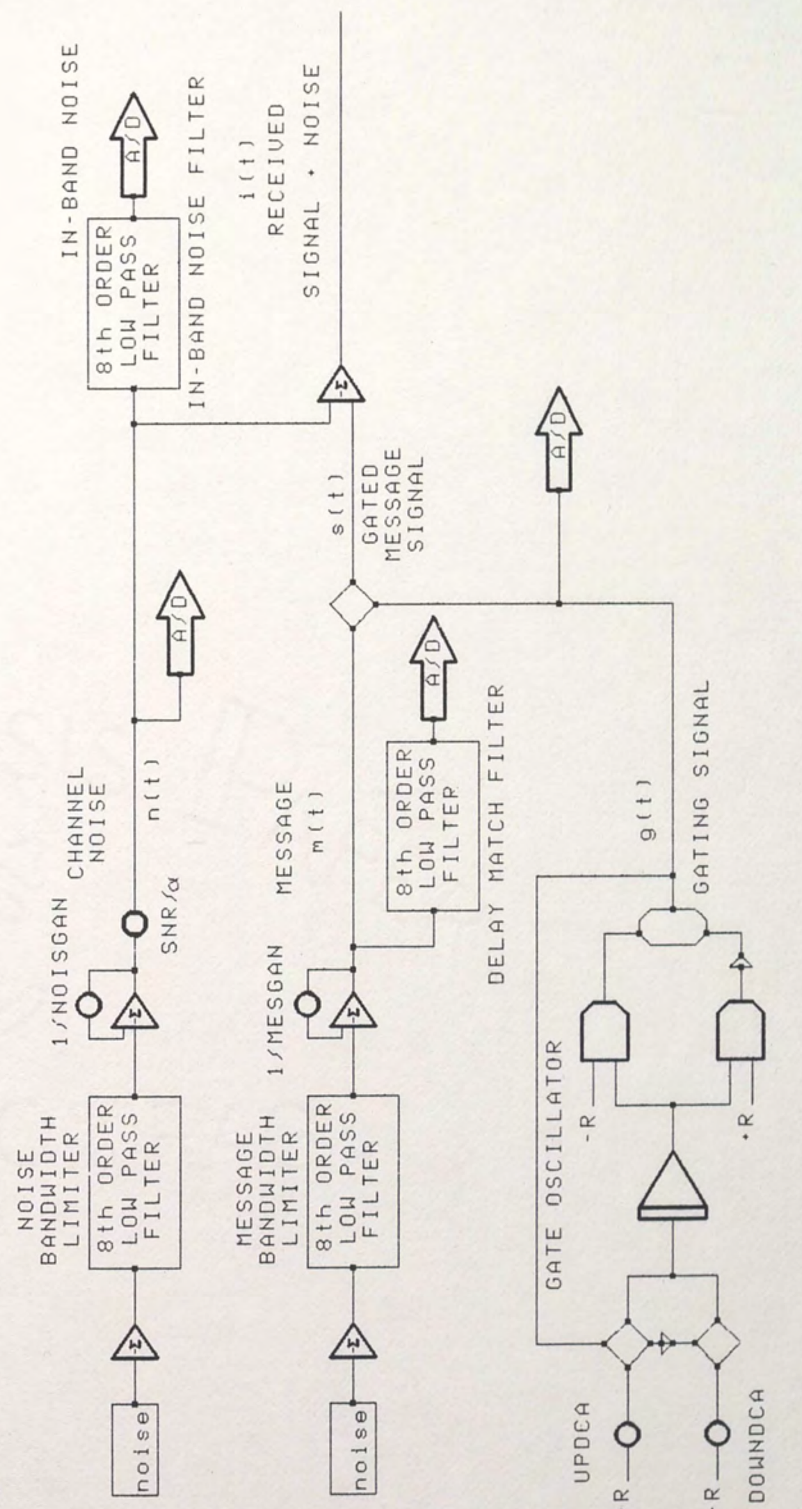

Figure 14. Transmitter Schematic

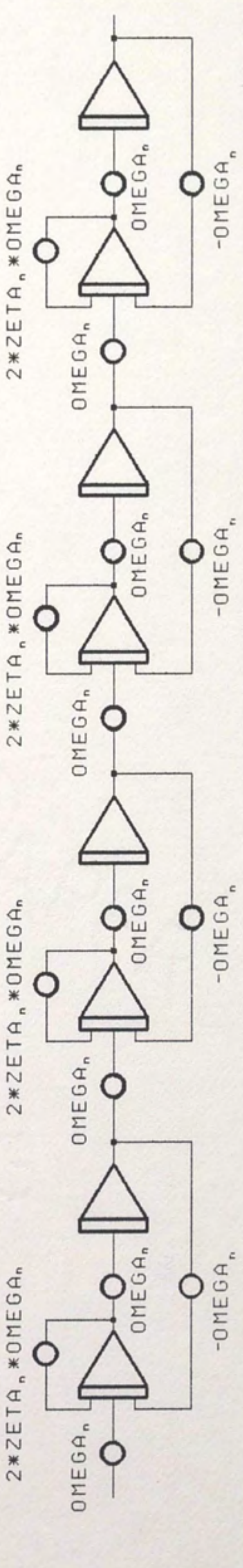

Figure 15. Filter Schematic

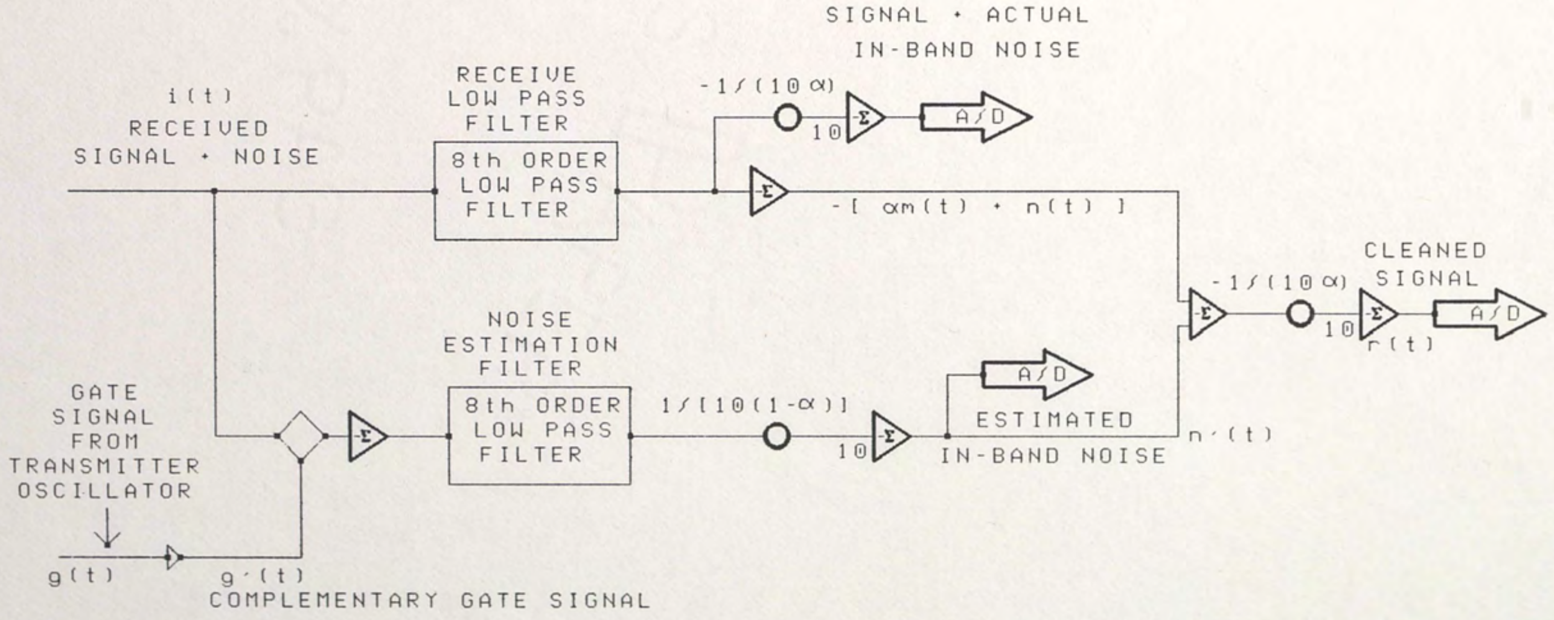

Figure 16. Receiver Schematic

In addition to the transmitter/receiver simulation, a power spectral density circuit was developed to monitor and verify signal activity at various locations throughout the system. A switch matrix was developed for selecting signal pick-up points. The algorithm used employed a direct implementation of a real time realization of the Fourier transform equation listed below. This equation utilizes the finite time modification

$$\mathfrak{F}[f(t)] = \int_{t_1}^{t_2} f(t')[\cos(\omega t') - j\sin(\omega t')]dt' \tag{4-1}$$

described in the beginning of Section 3. Figure 17 provides a schematic of the circuit.

In using the psd software, the user has control of the time spent evaluating each point, the lowest and highest frequencies evaluated, the number of steps evaluated between upper and lower frequency boundaries, and whether the frequency increments are arithmetically or geometrically progressive.

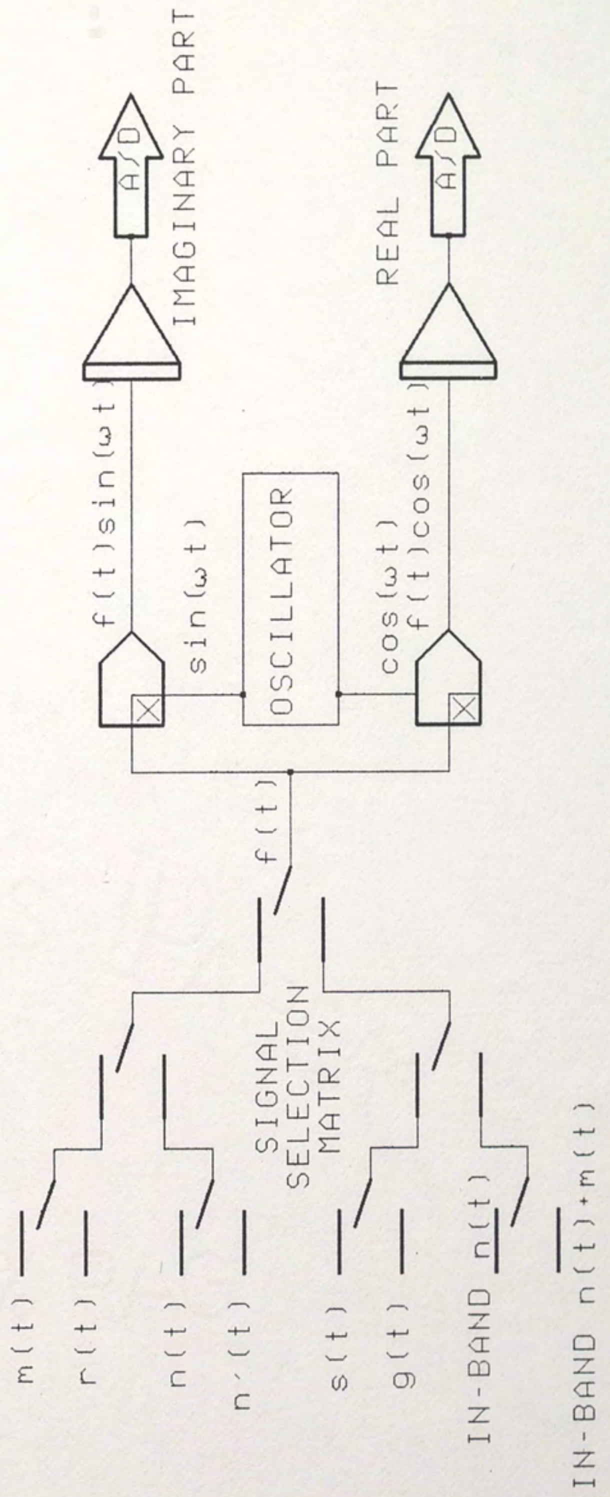

Figure 17. Schematic Of Power Spectral Density Circuit

SECTION V
RESULTS AND CONCLUSIONS

This section contains a summary of the results derived from the simulator and conclusions pertaining to the system in general. A graph of the theoretical expectation of the signal-to-noise ratio improvement is provided in Figure 18. Also plotted on the chart are actual data points taken from the simulator. The theoretical data charted is derived using equation 3-32, the signal-to-noise ratio improvement expression. Figure 19 is a similar type of comparison for a different system and noise configuration. Figure 20 depicts a comparison of message signal and receiver output without noise estimation. Figure 21 shows the same message signal but with receiver output using noise estimation. This configuration was set up for optimal system performance.

It is felt that the system performs as predicted by the mathematical relations developed in Section 3. This is evidenced by figures 18 and 19. Figures 20 and 21 indicate that at least in certain instances, the realization of the theory, and hence the actual system, provides an improvement in signal-to-noise ratio at the receiver output.

The math developed also predicts that noise can be translated down into the message bandwidth from areas where it normally would not affect the message. This has also been verified on the simulator. All of this indicates that careful consideration should be given to the noise characteristics of the channel medium prior to determining whether the system is suitable for a particular application or not. Another consideration is available channel bandwidth.

Again, it should be mentioned that the purpose of the study was not to prove the practicality of the system in general, but to show that noise and signal can be separated from the same spectral occupancy. It is felt by the author that this has been done successfully.

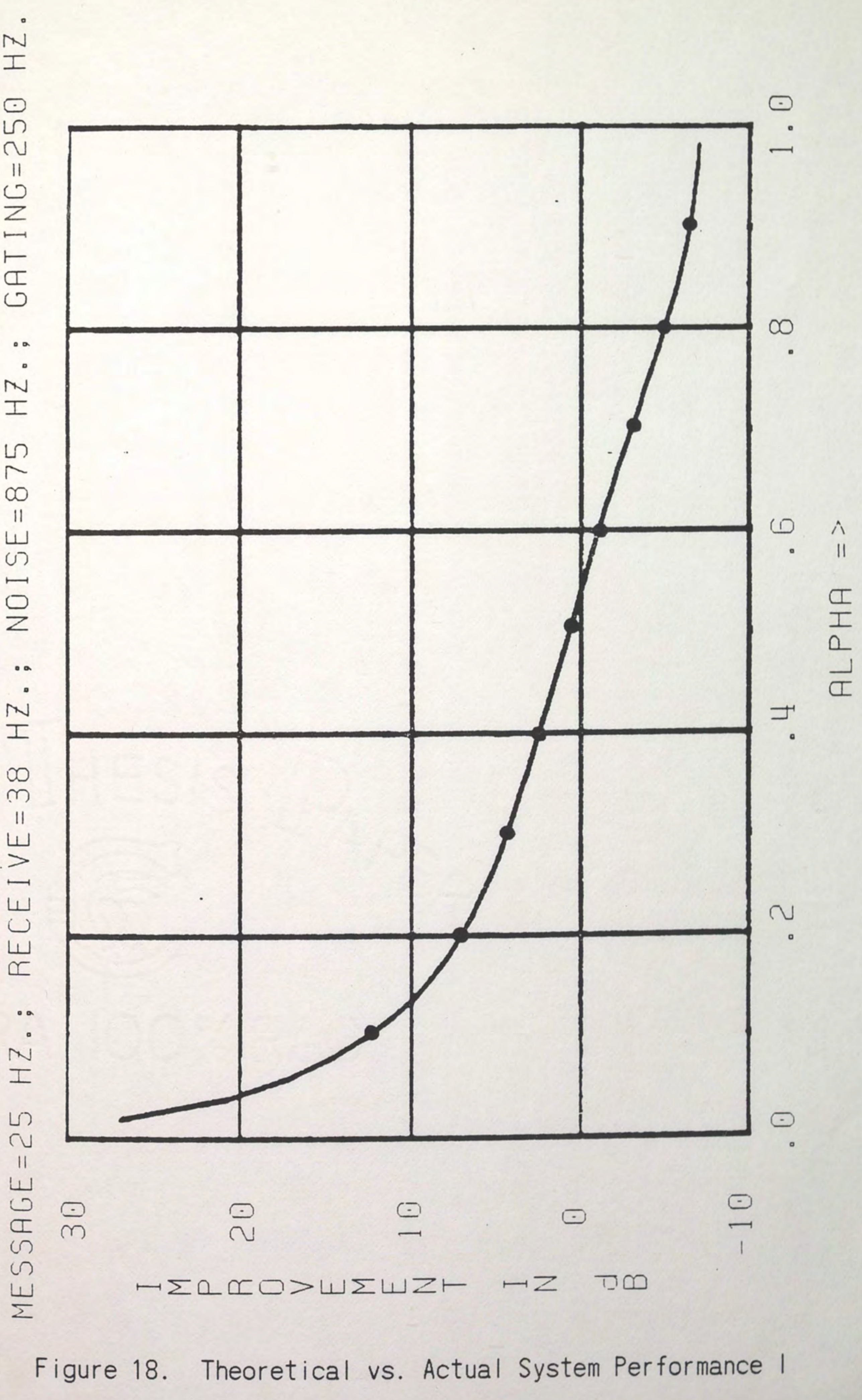

Figure 18. Theoretical vs. Actual System Performance I

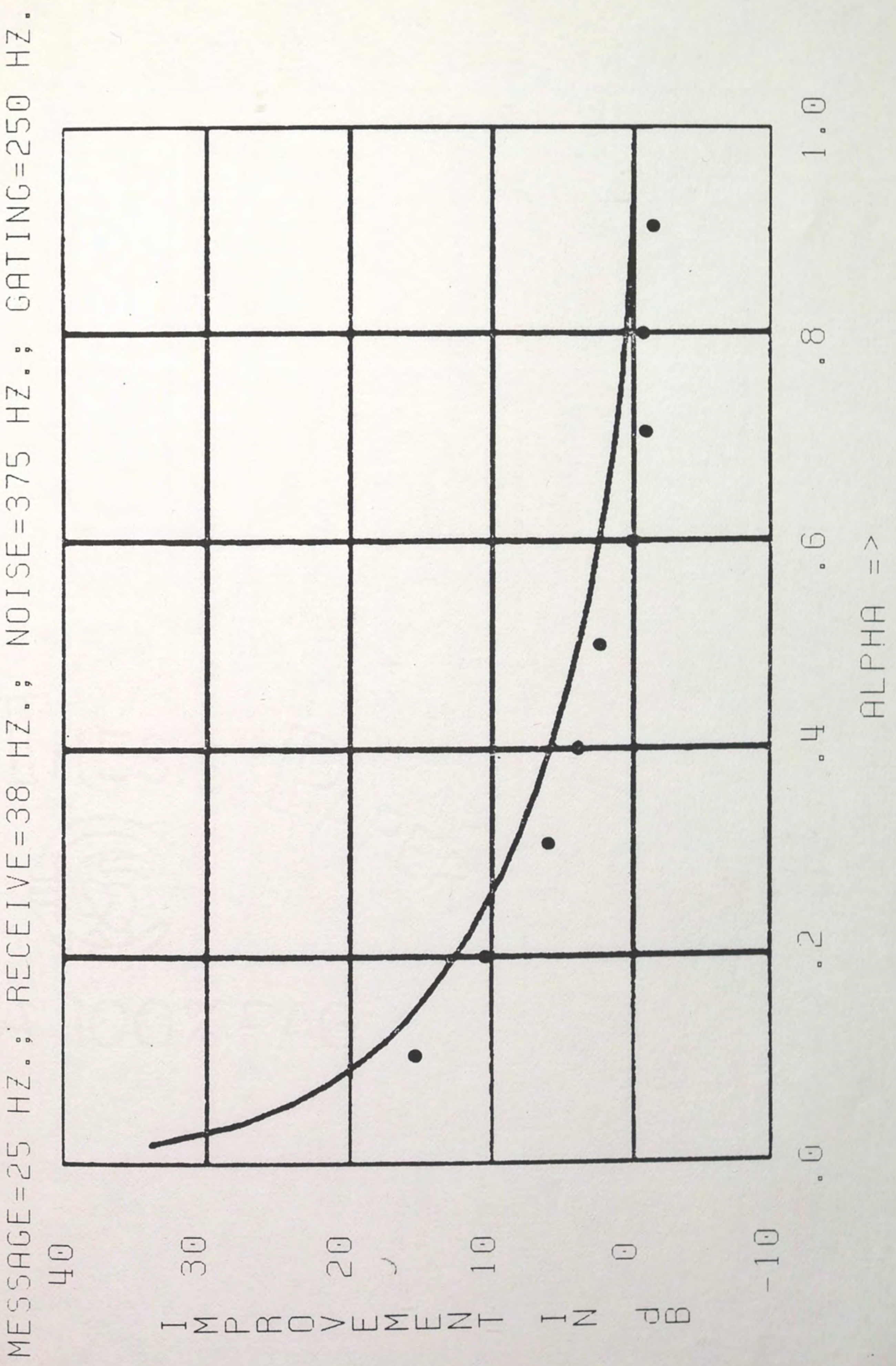

Figure 19. Theoretical vs. Actual System Performance II

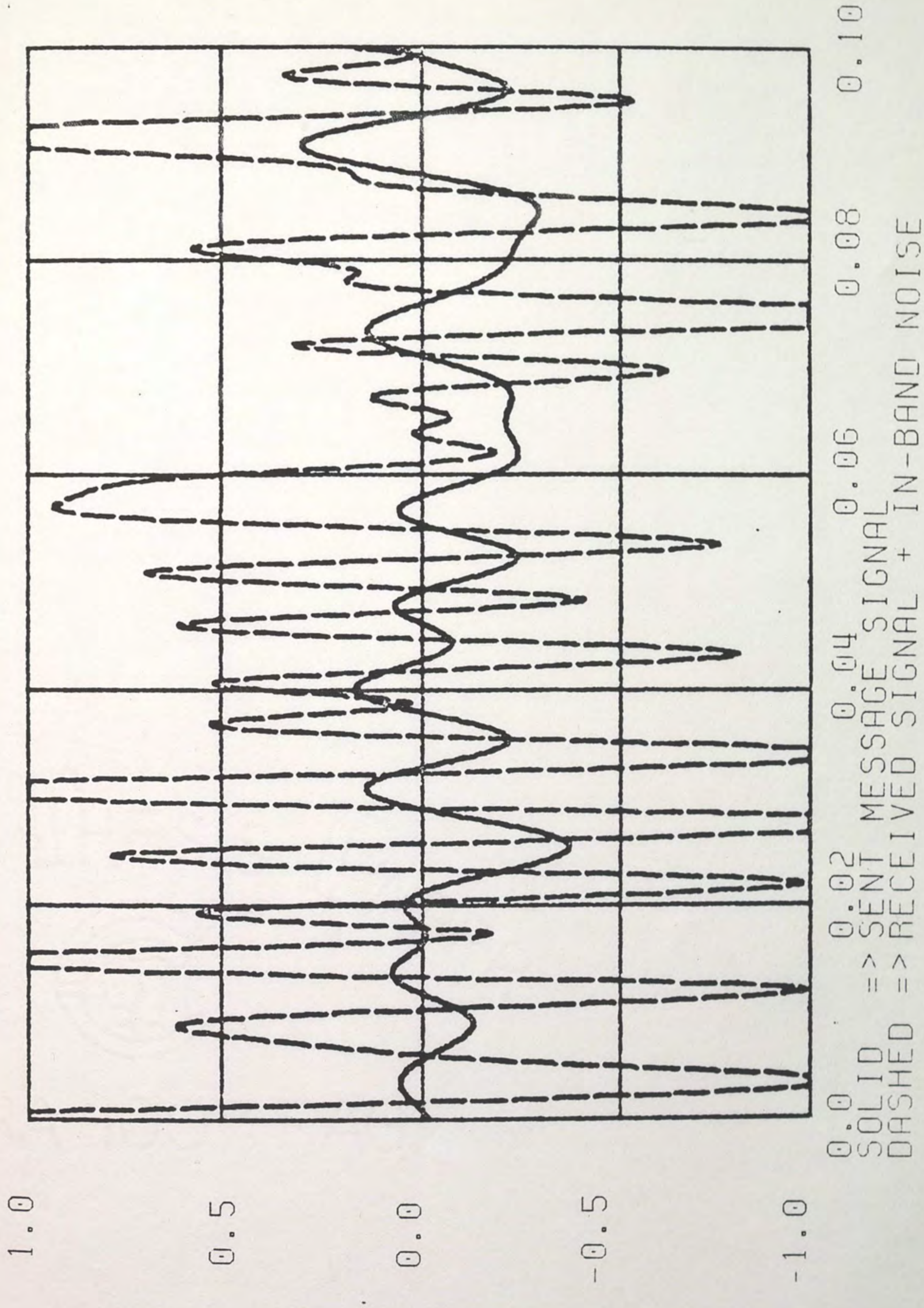

Figure 20. Transmit and Receive, No Noise Estimation

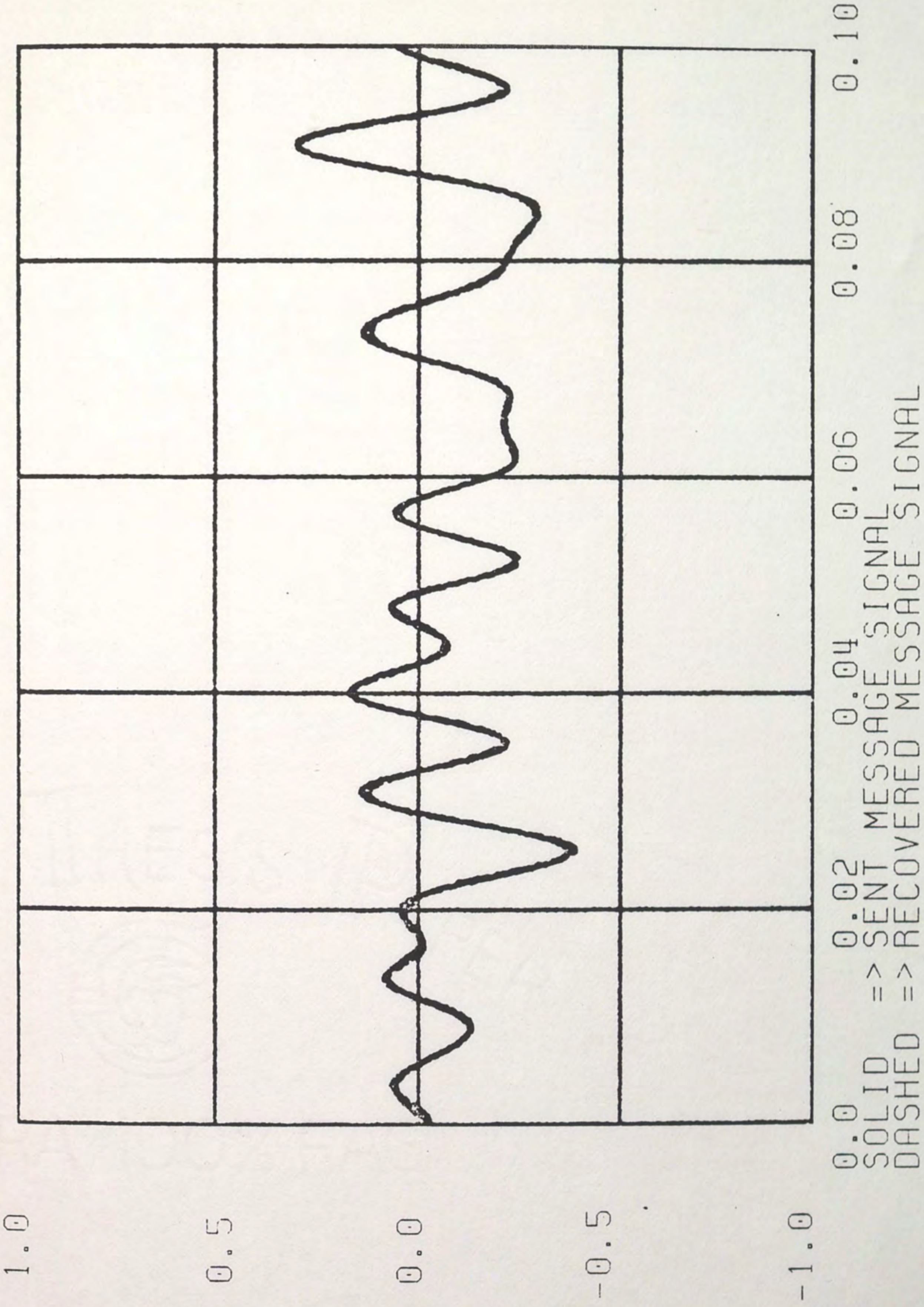

Figure 21. Transmit and Receive, With Noise Estimation

APPENDIX A

A MATHEMATICAL PROOF

Section 3 required the use of the following identity.

$$G_0 = \sum_{n=-\infty}^{+\infty} G_n{}^2 \qquad \text{(A-1)}$$

This relationship will be proven in this appendix.

$$G_n = \sin(n\pi\alpha)/(n\pi) \qquad \text{(A-2)}$$

$$\sum_{n=-\infty}^{+\infty} G_n{}^2 = \sum_{n=-\infty}^{+\infty} [\sin^2(n\pi\alpha)/(n\pi)^2] \qquad \text{(A-3)}$$

$$\sum_{n=1}^{+\infty} [\cos(n\theta)/(n^2)] = (\pi^2/6)-(\pi\theta)/2+(\theta^2/4) \qquad 0 \le \theta \le 2\pi \qquad \text{(A-4)}$$

$$\sum_{n=-\infty}^{+\infty}[\sin^2(n\pi\alpha)/(n\pi)^2]=\lim_{\varepsilon\to 0}\sin^2(\varepsilon\pi\alpha)/(\varepsilon\pi)^2+\sum_{\substack{n=-\infty\\ n\neq 0}}^{+\infty}[\sin^2(n\pi\alpha)/(n\pi)^2]$$

Using L'Hopital's Rule twice on the limit in the above equation, it reduces to the following expression.

$$\sum_{n=-\infty}^{+\infty}[\sin^2(n\pi\alpha)/(n\pi)^2] \qquad = \alpha^2 + \sum_{\substack{n=-\infty\\ n\neq 0}}^{+\infty}[\sin^2(n\pi\alpha)/(n\pi)^2] \qquad \text{(A-5)}$$

Using equations A-3 and A-5, and noting that the summands are even functions of n, the following formulation results.

$$\sum_{n=-\infty}^{+\infty}G_n{}^2 \qquad = \alpha^2 + 2\cdot\sum_{n=1}^{+\infty}[\sin^2(n\pi\alpha)/(n\pi)^2] \qquad \text{(A-6)}$$

$$\sum_{n=-\infty}^{+\infty}G_n{}^2 \qquad = \alpha^2 + [1/\pi^2]\cdot\sum_{n=1}^{+\infty}[1-\cos(2n\pi\alpha)]/n^2]$$

$$\sum_{n=-\infty}^{+\infty} G_n{}^2 = \alpha^2 + [1/\pi^2]\cdot\sum_{n=1}^{+\infty} 1/n^2 - [1/\pi^2]\cdot\sum_{n=1}^{+\infty}\cos(2n\pi\alpha)]/n^2$$

$$\sum_{n=1}^{+\infty} 1/n^2 = \pi^2/6$$

$$\sum_{n=-\infty}^{+\infty} G_n{}^2 = \alpha^2 + 1/6 - [1/\pi^2]\cdot\sum_{n=1}^{+\infty}\cos(2n\pi\alpha)]/n^2$$

$$\sum_{n=-\infty}^{+\infty} G_n{}^2 = \alpha^2 + 1/6 - [1/\pi^2]\cdot\sum_{n=1}^{+\infty}\cos(n\Theta)]/n^2 \qquad \Theta=2\pi\alpha$$

Using the identity of A–4, the following substitutions are made.

$$\sum_{n=-\infty}^{+\infty} G_n{}^2 = \alpha^2 + 1/6 - [1/\pi^2]\cdot[\pi^2/6 - \pi(2\pi\alpha)/2 + (2\pi\alpha)^2/4]$$

$$\Theta=2\pi\alpha$$

$$\sum_{n=-\infty}^{+\infty} G_n{}^2 \quad = \quad \alpha^2 + 1/6 - 1/6 + \alpha - \alpha^2 \qquad \text{(A-7)}$$

Using A–2 and A–7 completes the proof.

$$\sum_{n=-\infty}^{+\infty} G_n{}^2 \quad = \quad \alpha \ = G_0 \qquad \text{(A-8)}$$

APPENDIX B
FORTRAN SIMULATION PROGRAM LISTING

```
$PROG MAIN


$INCLUDE THESIS.COM
$END

      CALL DCASET(RADCON,'C512',.5,1)


      CALL RSINDX(LIU,LOU,LTWU,LPRU)

      STOP
      END
```

```
PROG ASINH
      FUNCTION ASINH(ARG)
INCLUDE THESIS.COM
      ASINH = ALOG( ARG + SQRT( ARG*ARG  +  1 ) )

      RETURN
      END
```

```
PROG BLKDAT
     BLOCK DATA
INCLUDE THESIS.COM
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

>>>>> ALL ENTRIES TO BE ALPHBETICALLY ORDERED IF POSSIBLE <<<<<

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>


     DATA BESPOL /
     +(-1.59656, 0.24761),(-1.48705, 0.74766),(-1.24811,
     1.26121),
    +(-0.81113, 1.81541)/




     DATA (BQDDCA( 1,I) ,I=1,4) /'C001','C000','C002','C003'/
     DATA (BQDDCA( 2,I) ,I=1,4) /'C011','C010','C012','C013'/
     DATA (BQDDCA( 3,I) ,I=1,4) /'C021','C020','C022','C023'/
     DATA (BQDDCA( 4,I) ,I=1,4) /'C031','C030','C032','C033'/
     DATA (BQDDCA( 5,I) ,I=1,4) /'C041','C040','C042','C043'/
     DATA (BQDDCA( 6,I) ,I=1,4) /'C051','C050','C052','C053'/
     DATA (BQDDCA( 7,I) ,I=1,4) /'C241','C240','C242','C243'/
     DATA (BQDDCA( 8,I) ,I=1,4) /'C251','C250','C252','C253'/
     DATA (BQDDCA( 9,I) ,I=1,4) /'C201','C200','C202','C203'/
     DATA (BQDDCA(10,I) ,I=1,4) /'C211','C210','C212','C213'/
     DATA (BQDDCA(11,I) ,I=1,4) /'C221','C220','C222','C223'/
     DATA (BQDDCA(12,I) ,I=1,4) /'C231','C230','C232','C233'/
     DATA (BQDDCA(13,I) ,I=1,4) /'C402','C400','C401','C403'/
     DATA (BQDDCA(14,I) ,I=1,4) /'C500','C502','C503','C501'/
     DATA (BQDDCA(15,I) ,I=1,4) /'C412','C410','C411','C413'/
     DATA (BQDDCA(16,I) ,I=1,4) /'C422','C420','C421','C423'/
     DATA (BQDDCA(17,I) ,I=1,3) /'C521','C523','C522'/
     DATA (BQDDCA(18,I) ,I=1,3) /'C430','C432','C431'/
     DATA (BQDDCA(19,I) ,I=1,3) /'C433','C441','C440'/
     DATA (BQDDCA(20,I) ,I=1,3) /'C443','C450','C442'/
```

```
      SVC 1 OUTPUT ASCII FUNCTION CODE

       DATA FC /Y'29'/

C      DEFINE DEFAULT RADIO CONSOLE AND MONITORING CONSOLE.

      DATA RADCON,MONCON /2,1/

      2 BYTE PAGE AND COPY TEKTRONIX COMMANDS

      DATA ICOPY,IPAGE /X'1B17',X'1B0C'/

      SETUP TEKTRONIX DEFAULT GRAPHICS DELAY
      DATA IDELAY/480/

      INITIALIZE LU ASSIGNMENTS

      DATA LIU,LOU,LTWU,LPRU /3,3,3,-1/

      DATA MESAMP /.5/
      DATA MESCOMP/
      0,6.3,12.5,25.1,50.5,100.,201.,402.,804.,1608.,
     +              3217.,6434.,
     +
      0,.02,.03,.04,.06,.10,.17,.28,.35,.45,.50,.50 /

      DATA NCYCLE /500/


      DATA ONE,TWO,THREE,FOUR,FIVE,SIX /1,2,3,4,5,6/


      DATA PI/3.1415927/

      DATA DLAFRQ /4.6,125,505,3920,6260,9660,14880/
      DATA DLACNT /100000,4000,1000,100,50,20,1/


      DATA SIGNAM /
     ;'NONE',
     ;'SENT MESSAGE SIGNAL',
     ;'RECOVERED MESSAGE SIGNAL',
     ;'ACTUAL IN-BAND NOISE',
     ;'ESTIMATED IN BAND NOISE',
;'GATED TRANSMIT SIGNAL',
;'GATE  SIGNAL',
;'TOTAL NOISE INPUT TO RECEIVER',
;'RECEIVED SIGNAL + IN-BAND NOISE',
;'DIFFERENCE BETWEEN SENT AND RECOVERED SIGNAL',
;'DIFFERENCE BETWEEN ACTUAL AND ESTIMATED IN-BAND NOISE'
  /

 END
```

```
PROG CLSR
     SUBROUTINE CLSR(ICONSL,CLNUMBER,SETTING,ERRFLG)


INCLUDE THESIS.COM

     SETS THE CONTROL LINE 'CLNUMBER' ON 'ICONSL' TO 0 IF
     'SETTING' IS 0 OR .FALSE. , AND SETS IT TO 1 IF 'SETTING'
     IS NON-ZERO OR .TRUE. . NOTE THAT 'SETTING' MAY BE INTEGER
     OR LOGICAL. DISPLAYS ERROR MESSAGES IF ERRFLG IS NON-ZERO
     OR .TRUE. . RETURNS 'IERR' TO CALLING ROUTINE THROUGH
     COMMON.

     INTEGER ICONSL,CLNUMBER,SETTING,ERRFLG


     IF(SETTING .EQ.0) THEN
        CALL QREFF7(ICONSL,CLNUMBER,IERR)
      ELSE
        CALL QSEFF7(ICONSL,CLNUMBER,IERR)
     ENDIF

     IF ( ERRFLG .EQ. 1 & IERR .NE. 1)THEN

        WRITE(LOU,100)'CONTROL LINE ERROR'

         IF( IERR .EQ. 2 ) WRITE(LOU,200) ICONSL,CLNUMBER
        IF( IERR .EQ. 3 ) WRITE(LOU,300) ICONSL,CLNUMBER
        IF( IERR .EQ. 4 ) WRITE(LOU,400) ICONSL,CLNUMBER

 100    FORMAT(' ',A,I2)
  200    FORMAT(' ',I1,'CL',I2,' , CONSOLE DISCONNECTED')
 300    FORMAT(' ',I1,'CL',I2,' , NON-EXISTING COMPONENT
        REQUESTED')
 400    FORMAT(' ',I1,'CL',I2,' , NON-EXISTING CONSOLE
        REQUESTED')

     ENDIF

     RETURN
     END
```

```
PROG COPY
     SUBROUTINE COPY

     SENDS COMMAND TO LOU FOR HARD COPY

INCLUDE THESIS.COM

     FC=Y'29'        SVC 1 FUNCTION CODE
     ICOPY=X'1B17'   2 BYTE COMMAND SEQUENCE FOR HARD COPY


     CALL SYSIO(IDUM,FC,LOU,ICOPY,2,0) ;SENDS 2 BYTE COMMAND TO
     LOU

     CALL WAIT(7,2,ISTAT) ; WAIT 7 SECONDS FOR COMPLETION.


     RETURN
     END
```

```
PROG CRA
     SUBROUTINE CRA(*,*)

     RINGS BELL THREE TIMES AT LOU AND REQUESTS DIRECTIVE
     IN NEXT PROCESS; CONTINUE, REPEAT, OR ABORT.

                CALLING SEQUENCE:

                                 CALL CRA(*LN1,*LN2)

                WHERE           :

                                LN1=LINE# OF CONTINUE PROCESS
                                LN2=LINE# OF REPEAT   PROCESS


     '/*' INPUT WILL BE PROCESSED AS ABORT AND WILL CALL INDEX2

INCLUDE THESIS.COM


     INTEGER CHOICE

  50 WRITE(LOU,100)
 100 FORMAT(' (1)  CONTINUE',/,
     +       ' (2)  REPEAT  ',/,
    +       ' (/*) RETURN TO INDEX2  ',//,
     +       ' ENTER DIRECTIVE: 1,2,OR /*.')

     CALL RING(3,LOU)

     READ(LIU,*,ERR=50,END=1000)CHOICE
     IF( CHOICE < 1 .OR. CHOICE > 2 ) GO TO 50

     RETURN CHOICE

1000 CALL STOPIT
     RETURN

     END
```

```
PROG DBTODEC

     FUNCTION DBTODEC(DB)

     CONVERTS DB VALUE TO DECIMAL RATIO.


     DBTODEC=10.**(DB/20.)

     RETURN
     END
```

```
PROG DCARED
     SUBROUTINE DCARED (ICONSL,ADDR,VALUE,ICTRL)

INCLUDE THESIS.COM

     INVOKES QRDCA7 TO READ A SINGLE DCA.

     IF AN ERROR CONDITION OCCURS A COMPREHENSIVE MESSAGE IS
     SENT TO LOU.  ICTRL=0 WILL SUPRESS THE ERROR MESSAGE.
C     ON ERROR CONDITION, CRA IS CALLED FOR USER DECISION-
     CONTINUE, REPEAT, INDEX2

      INTEGER ADDR


 60  CALL QRDCA7 (ICONSL,ADDR,VALUE,IERR)
     ERROR=IERR

     GOTO (100,10,20,30,10,50) IERR*ICTRL
     GOTO 100

  10 WRITE (LOU,11) ICONSL,ADDR
  11 FORMAT (1X,I2,A4,' ** INVALID ADDRESS ON DCA READ ** ')
      GOTO 100

  20 WRITE (LOU,21) ICONSL,ADDR
   21 FORMAT (1X,I2,A4,' ** ULI BUSY (READ) **')
     GOTO 100

  30 WRITE (LOU,31) ICONSL,ADDR
  31 FORMAT (1X,I2,A4,' ** CONSOLE DISCONNECT (READ) **')
      GOTO 100

  50 WRITE (LOU,51) ICONSL,ADDR
  51 FORMAT (1X,I2,A4,' ** INVALID CONSOLE (READ) **')

 100 IF(ERROR .GT. 1)CALL CRA(*110,*60)
  110 RETURN

     END
```

```
PROG DCASET
      SUBROUTINE DCASET (ICONSL,ADDR,VALUE,ICTRL)

INCLUDE THESIS.COM

      INVOKES QSDCA7 TO SET A SINGLE DCA.

      IF AN ERROR CONDITION OCCURS A COMPREHENSIVE MESSAGE IS
      SENT TO LOU.  ICTRL=0 WILL SUPRESS THE ERROR MESSAGE.
      CRA IS CALLED UPON ERROR, FOR USER DECISION--CONTINUE
      REPEAT, OR RETURN TO INDEX2.
      THE DCA IS READ TO VERIFY SETTING AND AN INCORRECT
      SETTING WILL BE INDICATED ON LOU.

      INTEGER ADDR


      IF(VALUE .EQ. 1) VALUE=.9999

   70 CALL QSDCA7 (ICONSL,ADDR,VALUE,IERR)
      ERROR=IERR

      GOTO (100,10,20,30,10,50,60) IERR*ICTRL
      GOTO 100
   10 WRITE (LOU,11) ICONSL,ADDR
   11 FORMAT (1X,I2,A4,' ** INVALID ADDRESS (SET) **')
      GOTO 100
    20 WRITE (LOU,21) ICONSL,ADDR,VALUE
   21 FORMAT (1X,I2,A4,' ** DCA OVERFLOW, VALUE=',F7.4,'
      (SET)**')
       GOTO 100
   30 WRITE (LOU,31) ICONSL,ADDR
   31 FORMAT (1X,I2,A4,' ** CONSOLE DISCONNECT (SET)**')
      GOTO 100
   50 WRITE (LOU,51) ICONSL,ADDR
   51 FORMAT (1X,I2,A4,' ** INVALID CONSOLE (SET)**')
       GOTO 100
   60 WRITE (LOU,61) ICONSL,ADDR
   61 FORMAT (1X,I2,A4,' ** ULI BUSY (SET)**')
  100 IF(ERROR .GT. 1)CALL CRA(*110,*70)
  110 CALL DCARED(ICONSL,ADDR,VALC,ICTRL)

      IF (ABS(VALUE-VALC).LE.0.0003) RETURN
      WRITE (LOU,105) ICONSL,ADDR,VALUE,VALC
```

```
105 FORMAT (1X,I2,A4,'=',F7.4,'   SET TO ',F7.4,' ,OUT OF
   TOL.')
   CALL CRA(*150,*110)
150 RETURN
    END
```

```
PROG DECTODB

      FUNCTION DECTODB(DEC)

      CONVERTS DECIMAL VALUE TO DB EQUIVALENT

      DECTODB=20.*ALOG10(DEC)

      RETURN
      END
```

```
PROG FLT
     SUBROUTINE FLT

     CALL FLT0
     CALL FLT1
     CALL SNR1

      RETURN
     END
```

```
PROG FLTO
    SUBROUTINE FLTO

INCLUDE THESIS.COM



     IF( WCO(1)==0 .OR. WCO(2)==0 .OR. WCO(3)==0 ) THEN

       IF( WCO(TRANSMIT)==0 ) THEN

  10        WRITE(LOU,*)'FOR TRANSMIT MESSAGE FILTER, ENTER
            CUTOFF'
            WRITE(LOU,*)'FREQUENCY IN HERTZ'
            READ(LIU,*,ERR=10,END=1000) FCO
            IF( FCO < 0 ) GO TO 10

            WCO(TRANSMIT)=2*PI*FCO

  40        WRITE(LOU,45)
  45        FORMAT(' ENTER FILTER TYPE',/,' 1 BUTTERWORTH',/,
   +          ' 2 BESSEL',/,' 3 CHEBYSHEV')
            READ(LIU,*,ERR=40,END=1000) FILTYP(TRANSMIT)
            IF( FILTYP(TRANSMIT) < 1  .OR.  FILTYP(TRANSMIT) >
            3)GOTO 40

            IF( FILTYP(TRANSMIT) == CHEBY ) THEN

  47           WRITE(LOU,*) 'ENTER PASSBAND RIPPLE IN DB > 0'
               READ(LIU,*,ERR=47,END=1000)RIPPLE(TRANSMIT)
               IF( RIPPLE(TRANSMIT) <= 0 ) GO TO 47

            ENDIF

       ENDIF

       IF( WCO(LOPASS)==0 ) THEN

  50        WRITE(LOU,*)'FOR LOW  PASS RECEIVE FILTER, ENTER
            CUTOFF'
            WRITE(LOU,*)'FREQUENCY RELATIVE TO TRANSMIT MESSAGE
            FILTER'
```

```
        READ(LIU,*,ERR=50,END=1000) RELFRQ(LOPASS)
         IF( RELFRQ(LOPASS) < 0 ) GO TO 50

        WCO(LOPASS)=WCO(TRANSMIT)*RELFRQ(LOPASS)


60      WRITE(LOU,65)
65      FORMAT(' ENTER FILTER TYPE',/,' 1 BUTTERWORTH',/,
  +      ' 2 BESSEL',/,' 3 CHEBYSHEV')
        READ(LIU,*,ERR=60,END=1000) FILTYP(LOPASS)
        IF( FILTYP(LOPASS) < 1  .OR.  FILTYP(LOPASS) > 3 )

        GO TO 60

        IF( FILTYP(LOPASS) == CHEBY ) THEN

 67         WRITE(LOU,*) 'ENTER PASSBAND RIPPLE IN DB > 0'
           READ(LIU,*,ERR=67,END=1000)RIPPLE(LOPASS)
           IF( RIPPLE(LOPASS) <= 0 ) GO TO 67


70      WRITE(LOU,*)'FOR NOISE FILTER, ENTER CUTOFF'
        WRITE(LOU,*)'FREQUENCY RELATIVE TO TRANSMIT MESSAGE
        FILTER'
         READ(LIU,*,ERR=70,END=1000) RELFRQ(NOISE)
        IF( RELFRQ(NOISE) < 0 ) GO TO 70

        WCO(NOISE)=WCO(TRANSMIT)*RELFRQ(NOISE)


72      WRITE(LOU,75)
75      FORMAT(' ENTER FILTER TYPE',/,' 1 BUTTERWORTH',/,
  +      ' 2 BESSEL',/,' 3 CHEBYSHEV')
         READ(LIU,*,ERR=72,END=1000) FILTYP(NOISE)
        IF( FILTYP(NOISE) < 1  .OR.  FILTYP(NOISE) > 3 ) GO
         TO 70

        IF( FILTYP(NOISE) == CHEBY ) THEN
```

```
77          WRITE(LOU,*) 'ENTER PASSBAND RIPPLE IN DB > 0'
           READ(LIU,*,ERR=77,END=1000)RIPPLE(NOISE)
           IF( RIPPLE(NOISE) <= 0 ) GO TO 77

        ENDIF

      END IF


    ELSE

 80    WRITE(LOU,*)'ENTER FILTER ID'
      WRITE(LOU,*)'1 TRANSMIT'
      WRITE(LOU,*)'2 LOPASS'
      WRITE(LOU,*)'3 NOISE'
       READ(LIU,*,ERR=80,END=1000) FILTID

      IF( FILTID < 1  .OR.  FILTID > 3 ) GO TO 80

      IF ( FILTID == TRANSMIT ) THEN

 95      WRITE(LOU,*)'ENTER THE MESSAGE CUTOFF FREQUENCY IN
         HERTZ'
         READ(LIU,*,ERR=95,END=1000)FCO
         IF( FCO < 0 ) GO TO 95
          WCO(TRANSMIT)=2*PI*FCO
         WCO(LOPASS)=RELFRQ(LOPASS)*WCO(TRANSMIT)
         WCO(NOISE)=RELFRQ(NOISE)*WCO(TRANSMIT)

      ELSE


100      WRITE(LOU,*)'ENTER BANDWIDTH RELATIVE TO MESSAGE
         BANDWIDTH'
         READ(LIU,*,ERR=100,END=1000)RELFRQ(FILTID)
         IF( RELFRQ(FILTID) < 0 ) GO TO 100

         WCO(FILTID)=RELFRQ(FILTID)*WCO(TRANSMIT)

      ENDIF

110   WRITE(LOU,115)
115   FORMAT(' ENTER FILTER TYPE',/,' 1 BUTTERWORTH',/,
     +        ' 2 BESSEL',/,' 3 CHEBYSHEV')
         READ(LIU,*,ERR=110,END=1000) FILTYP(FILTID)
         IF( FILTYP(FILTID) < 1  .OR.  FILTYP(FILTID) > 3 ) GO
         TO 110

         IF( FILTYP(FILTID) == CHEBY ) THEN

 117        WRITE(LOU,*) 'ENTER PASSBAND RIPPLE IN DB > 0'
            READ(LIU,*,ERR=117,END=1000)RIPPLE(FILTID)
            IF( RIPPLE(FILTID) <= 0 ) GO TO 117

         ENDIF


      ENDIF


      RETURN

1000  CALL STOPIT

       RETURN
      END
```

```
PROG FLT1
     SUBROUTINE FLT1
INCLUDE THESIS.COM



     MAXGAIN=0
     MAXNGAIN=0

     DO FILTID=TRANSMIT,NOISE
        ID=FILTID

        DO POLNUM=1,4

           POLE=NRMPOL( FILTYP(FILTID) , POLNUM ,
           RIPPLE(FILTID)  )

C           IF( FILTID==HIPASS ) POLE=1/POLE
           POLE=WCO(FILTID)*POLE

            POLES(FILTID,POLNUM)=POLE

           WC=CABS(POLE)
            ZETA=-REAL(POLE)/WC

           IF( FILTID==NOISE ) THEN

              IF( WC > MAXNGAIN ) MAXNGAIN=WC
              IF( 2*ZETA*WC > MAXNGAIN ) MAXNGAIN=2*ZETA*WC

           ELSE

              IF( WC>MAXGAIN ) MAXGAIN=WC
               IF( 2*ZETA*WC>MAXGAIN ) MAXGAIN=2*ZETA*WC

           END IF

           END DO

        END DO

     LOGGAIN=LOG10(MAXGAIN)
     LOGNGAIN=LOG10(MAXNGAIN)
     ILOG=INT(LOGGAIN)
```

```
    INLOG=INT(LOGNGAIN)

     CALL INTCL(INLOG,RADCON,10,11,12)

    INTGAN=10**ILOG
     NINTGAN=10**INLOG

     GAINLINE=ILOG+2
    GOTO(101,102,103,104,105,106) GAINLINE
    GO TO 106

101 CALL QSSLOO(RADCON,IERR)
    CALL QSSLOO(MONCON,IERR)
    GOTO 110
102 CALL QSMEDO(RADCON,IERR)
    CALL QSMEDO(MONCON,IERR)
     GOTO 110
103 CALL QSFSTO(RADCON,IERR)
    CALL QSFSTO(MONCON,IERR)
    GOTO 110
104 CALL QMSLOO(RADCON,IERR)
     CALL QMSLOO(MONCON,IERR)
    GOTO 110
105 CALL QMMEDO(RADCON,IERR)
    CALL QMMEDO(MONCON,IERR)
    GOTO 110
106 WRITE(LOU,107)
 107 FORMAT(' SPECIFIED FILTER PARAMAETERS REQUIRE TOO HIGH
             AN',/,
   +        ' INTEGRATOR GAIN, PROBABLY DUE TO TOO HIGH A ',/,
   +        ' REQUESTED CUT-OFF FREQUENCY. PLEASE MODIFY ',/,
   +        ' FILTER CHARACTERISTICS.')

    CALL STOPIT

 110 CONTINUE

     DO FILTID=TRANSMIT,NOISE
       CALL SETFLT(FILTID)
       END DO

    CALL MESLVL1        ;SET MESSAGE LEVEL TO COMPENSATE FOR LPF

    RETURN
```

```
END
```

```
$PROG GATE
      SUBROUTINE GATE

      CALL GATE0
      CALL GATE1

      RETURN
      END
```

```
PROG GATEO
     SUBROUTINE GATEO


INCLUDE THESIS.COM


  10 WRITE(LOU,*)'ENTER GATING FREQUENCY RELATIVE TO'
     WRITE(LOU,*)'TRANSMIT MESSAGE CUTOFF FREQUENCY'
     READ(LIU,*,ERR=10,END=1000) RELGATE

  20 WRITE(LOU,*)'ENTER GATING DUTY CYCLE, ALPHA'
     WRITE(LOU,*)'0.0 <= ALPHA <= 1.0'

     READ(LIU,*,ERR=20,END=1000) ALPHA
      IF( ALPHA < 0 .OR. ALPHA > 1 ) GO TO 20

      RETURN

1000 CALL STOPIT
     RETURN
     END
```

```
PROG GATE1
     SUBROUTINE GATE1

INCLUDE THESIS.COM

      GATEFREQ=RELGATE*FCO
     GATETIME=1/GATEFREQ
     TIMEUP=ALPHA*GATETIME
      TIMEDOWN=(1-ALPHA)*GATETIME

     UPGAIN=2/TIMEUP
     DOWNGAIN=2/TIMEDOWN

      IF( UPGAIN > DOWNGAIN ) THEN
        MAXGAIN=UPGAIN
     ELSE
        MAXGAIN=DOWNGAIN
     END IF

     LOGGAIN=LOG10(MAXGAIN)
     ILOG=LOGGAIN

     GAINCNST=10.**(ILOG+1)

      UPDCA=UPGAIN/GAINCNST
     DOWNDCA=DOWNGAIN/GAINCNST

     IF( ILOG <= -1 ) THEN
        GATECL(1)=1
         GATECL(2)=0
        GATECL(3)=0
     ELSE IF ( ILOG==0 ) THEN
        GATECL(1)=1
        GATECL(2)=1
        GATECL(3)=0
     ELSE IF ( ILOG==1 ) THEN
        GATECL(1)=0
        GATECL(2)=1
        GATECL(3)=0
     ELSE IF ( ILOG==2 ) THEN
        GATECL(1)=1
        GATECL(2)=1
         GATECL(3)=1
     ELSE IF ( ILOG==3 ) THEN
```

```
         GATECL(1)=0
         GATECL(2)=1
         GATECL(3)=1
      ELSE IF ( ILOG==4 ) THEN
         GATECL(1)=0
         GATECL(2)=0
         GATECL(3)=1
      ELSE

         WRITE(LOU,*)'GATING FREQUENCY TOO HIGH!'

      ENDIF

      DO 100 I=1,3

         CALL CLSR(RADCON,I,GATECL(I),1)
100      CONTINUE

      CALL DCASET(RADCON,'C550',UPDCA,1)
      CALL DCASET(RADCON,'C551',DOWNDCA,1)

      DCAVAL= 1/( 10 * ( 1 - ALPHA ) )
      CALL DCASET(RADCON,'C332',+DCAVAL,1)

      DCAVAL=-1/( 10 * ALPHA )
      CALL DCASET(RADCON,'C121',DCAVAL,1)
      CALL DCASET(RADCON,'C321',DCAVAL,1)


      RETURN
      END
```

```
PROG INTCL
     SUBROUTINE INTCL( IGAIN , ICON , CL10 , CLP1 , CLP1MS )
INCLUDE THESIS.COM

     INTEGER CL10,CLP1,CLP1MS
     SETS THE THREE CONTROL LINES SPECIFIED IN CL10, CLP1, AND,
     CLP1MS ON CONSOLE ICON TO THE PROPER VALUES TO CAUSE AN
     INTEGRATOR
     WITH THOSE THREE CONTROL LINES INPUT TO THE 10, .1, AND
     .1MS
     LOGIC INPUTS TO HAVE A GAIN OF 10**IGAIN.


     IF( IGAIN == -1 ) THEN

        ICL10=1
        ICLP1=0
        ICLP1MS=0

     ELSE IF( IGAIN == 0 ) THEN

        ICL10=1
        ICLP1=1
        ICLP1MS=0

     ELSE IF( IGAIN == 1 ) THEN

        ICL10=0
        ICLP1=1
        ICLP1MS=0

     ELSE IF( IGAIN == 2 ) THEN

        ICL10=1
        ICLP1=1
         ICLP1MS=1

     ELSE IF( IGAIN == 3 ) THEN

        ICL10=0
        ICLP1=1
        ICLP1MS=1

     ELSE IF( IGAIN == 4 ) THEN
```

```
      ICL10=0
       ICLP1=0
      ICLP1MS=1

   END IF

   CALL CLSR(ICON,CL10,ICL10,1)
   CALL CLSR(ICON,CLP1,ICLP1,1)
    CALL CLSR(ICON,CLP1MS,ICLP1MS,1)

    RETURN
   END
```

```
PROG MESLVL
     SUBROUTINE MESLVL
INCLUDE THESIS.COM


     CALL MESLVL0
     CALL MESLVL1

     RETURN
     END
```

```
$PROG MESLVLO
      SUBROUTINE MESLVLO
INCLUDE THESIS.COM

   10 WRITE(LOU,*)'ENTER DESIRED MESSAGE AMPLITUDE'
      WRITE(LOU,*)' 0 <= MESAMP <= 1.0'

       READ(LIU,*,ERR=10,END=1000) MESAMP
      IF( MESAMP < 0 .OR. MESAMP > 1.0 ) GO TO 10




      RETURN

      END
```

```
PROG MESLVL1
     SUBROUTINE MESLVL1
INCLUDE THESIS.COM



     DO 100 I=1,12

        IF( WCO(TRANSMIT) < MESCOMP(I+1,1) ) GO TO 200
  100    CONTINUE

 200 CALL NTRPLT( WCO(TRANSMIT) , PEAK , MESCOMP(I,1) ,
    +             MESCOMP(I+1,1) , MESCOMP(I,2) ,
     MESCOMP(I+1,2)  )

     MESGAIN=MESAMP/PEAK

     DCAVAL=1./MESGAIN
     CALL DCASET(RADCON,'C132',DCAVAL,1)

     RETURN
      END
```

```
PROG NRMPOL
     FUNCTION NRMPOL( TYPE , NPOLE , PASRIP )

         THIS SUBROUTINE CALCULATES THE "NTH" NORMALIZED
      LOW-PASS
      POLE FOR ANY OF THREE TYPES OF FILTER:

          1) BUTTERWORTH
          2) BESSEL
          3) CHEBYSCHEV

          CALLING ARGUMENTS:

          TYPE   => FLTER TYPE  1) BUTTERWORTH
                                   :INTEGER*4
                                2) BESSEL
                                3) CHEBYSCHEV


          NPOLE  => "NTH" POLE NUMBER  1<=NPOLE<=FILORD
                     :INTEGER*4

*          PASRIP => PASS BAND RIPPLE IN DB FOR CHEBY     :REAL


        THIS FUNCTION SUPPLIES ONLY THE POLE WITH THE POSITIVE
     IMAGINARY PART. THE LOWER THE NPOLE NUMBER, THE CLOSER THE
     CALCULATED POLE WILL BE TO THE REAL AXIS.

        THE EQUATIONS USED TO CALCULATE THE POLE LOCATIONS ARE
     DESCIBED IN "PRINCIPLES OF ACTIVE NETWORK SYNTHESIS AND
     DESIGN" BY GOBIND DARYANANI, PAGES 100-114 FOR THE BUTTER-
     WORTH AND CHEBYSCHEV FILTERS. THE BESSEL FILTER EQUATIONS
     WERE
     TAKEN FROM "DIGITAL FILTERS: ANALYSIS AND DESIGN" BY
     ANTONIOU, PAGE 129. THE RESULTS WERE STORED IN AN ARRAY
     TO BE LOOKED UP SINCE CALCULATING THEM ON LINE IS A TIME
     CONSUMING PROCESS.


INCLUDE THESIS.COM

     INTEGER TYPE,NPOLE
```

```
REAL IMPART


IF( TYPE == BUTTER ) THEN

   ANGINC=PI/8
   ANGLE=PI-(NPOLE-0.5)*ANGINC

   REPART=COS(ANGLE)
   IMPART=SIN(ANGLE)

    NRMPOL=CMPLX(REPART,IMPART)

ELSE IF( TYPE == CHEBY  ) THEN

   EPSILN=SQRT( 10.**(PASRIP/10.) - 1.0 )
    SINHK=SINH(  ASINH(1./EPSILN) / 8  )
   COSHK=COSH(  ASINH(1./EPSILN) / 8  )
   K    =(3*8)/2 - 1   +   NPOLE
   ARG= PI/2 * (1. + 2.*K)  /  8
   REPART=SINHK*SIN(ARG)
   IMPART=COSHK*COS(ARG)

   NRMPOL=CMPLX(REPART,IMPART)

 ELSE          ; TYPE == BESSEL

    NRMPOL=BESPOL(NPOLE)

END IF

RETURN
END
```

```
PROG NTRPLT
     SUBROUTINE NTRPLT(XI,YD,XLOW,XHIGH,YLOW,YHIGH)


INCLUDE THESIS.COM

C     THIS IS AN INTERPOLATION ROUTINE. 'XI' IS THE INDEPENDENT
     VARIABLE. 'YD' IS THE DEPENDENT VARAIBLE. 'XLOW' AND
     'XHIGH'
     ARE THE BREAKPOINTS WHICH STRADLE THE VARIABLE 'XI', AND
     LIKEWISE FOR 'YD'. IF 'XI' IS NOT BETWEEN 'XLOW' AND
     'XHIGH',
     AN EXTRAPOLATED VALUE WILL BE RETURNED IN 'YD'.

      YD = YLOW + (XI-XLOW ) * (  ( YHIGH-YLOW ) / (XHIGH-XLOW
      )  )


     RETURN
     END
```

```
PROG PAGE
     SUBROUTINE PAGE

     SENDS COMMAND TO LOU FOR PAGE

$INCLUDE THESIS.COM

     FC=Y'29'        SVC 1 FUNCTION CODE
     IPAGE=X'1B0C'   2 BYTE COMMAND SEQUENCE FOR PAGE



     CALL SYSIO(IDUM,FC,LOU,IPAGE,2,0) ;SENDS 2 BYTE COMMAND TO
                LOU

      CALL WAIT(1,2,ISTAT) ;WAIT FOR PAGE COMPLETION.


     RETURN
     END
```

```
PROG POTRED
     SUBROUTINE POTRED(OUTFILE)
************************************************************

     USE:  CALL POTRED(OUTFILE)

                OUTFILE= NAME OF FILE TO WRITE CONSOLE, POT
                ADDRESS,
                         AND READ VALUE OF POTS FOR CONSOLES
                         71,72,73
                         (FILE SHOULD PRESENTLY EXIST AS A COPY
                OF
                          INITIAL SETUP FILE. IE. 'SETUP.NN'

     PURPOSE:   POTRED WILL READ A RECORD OF FILE INDICATED AS
                OUTFILE,
                CALL DCARED PASSING CONSOLE NUMBER(CONSOLE),
                POT ADDRESS
                (ADDRESS), RETURN VARIABLE FOR POT
                COEFFICIENT(POTVAL),
                AND ERROR CONTROL(ERCNTL)[ALLOWS/DISALLOWS
                ERROR OUTPUT
                MESSAGES]. THE RETURNED POT COEFFICIENT IS THEN
                WRITTEN
                TO THE OUTFILE RECORD.

***************************************************************
INCLUDE THESIS.COM
     CHARACTER*8 OUTFILE
  5  OPEN(UNIT=8,FILE=OUTFILE,STATUS='OLD',ERR=200)
 10   READ(8,20,END=100,ERR=80)CONSOLE,ADDRESS




      WRITE(8,30,ERR=90)CONSOLE,ADDRESS,POTVAL
      GO TO 10
 80   WRITE(LOU,*)'SETUP FILE READ ERROR DURING POTRED ROUTINE'
      CLOSE(UNIT=8)
      CALL STOPIT
 90   WRITE(LOU,*)'SETUP FILE WRITE ERROR DURING POTRED
      ROUTINE'
     CLOSE(UNIT=8)
100  CLOSE(UNIT=8)
210   RETURN
```

```
200   WRITE(LOU,*)'ERROR OPENING SETUP FILE',OUTFILE
      CALL CRA(*210,*5)
 20   FORMAT(I1,2X,A4)
  30   FORMAT(I1,2X,A4,2X,F7.4)
     RETURN
     END
```

```
PROG POTSET
     SUBROUTINE POTSET(INFILE)
************************************************************

     USE:  CALL POTSET(INFILE)

                 OUTFILE= NAME OF FILE TO READ CONSOLE, POT
                 ADDRESS,
                          AND COEFFICIENTS OF POTS FOR CONSOLES
                 71,72,73
                          (FILE SHOULD PRESENTLY EXIST)

     PURPOSE:    POTSET WILL READ A RECORD OF FILE INDICATED AS
                 INFILE AND
                 CALL DCASET PASSING CONSOLE NUMBER(CONSOLE),
                 POT ADDRESS
                 (ADDRESS), POT COEFFICIENT(POTVAL),AND ERROR
                 CONTROL
                 (ERCNTL)[ALLOWS/DISALLOWS ERROR OUTPUT
MESSAGES].

**************************************************************
INCLUDE THESIS.COM
      CHARACTER*8 INFILE
     OPEN(UNIT=8,FILE=INFILE,STATUS='OLD',ERR=200)
 10   READ(8,20,END=100,ERR=90)CONSOLE,ADDRESS,POTVAL
      CALL DCASET(CONSOLE,ADDRESS,POTVAL,ERCNTL)
      GO TO 10
 90   WRITE(LOU,*)'SETUP FILE READ ERROR DURING POTSET ROUTINE'
     CLOSE(UNIT=8)
      CALL STOPIT
100  CLOSE(UNIT=8)
     RETURN
200  WRITE(LOU,*)'ERROR OPENING SETUP FILE--',INFILE
     CALL STOPIT
     RETURN
 20  FORMAT(I1,2X,A4,2X,F7.4)
     END
```

```
$PROG PREDICT
     SUBROUTINE PREDICT
$INCLUDE THESIS.COM


     WN=WCO(NOISE)
     WG=2*PI*GATEFREQ
     WR=WCO(LOPASS)

     SNROSNR4(0)=49
     ALPHARA(0)=49

      DO 500 IALPH=1,49

        TALPHA=IALPH/50.


        DO 100 N=-20,20
            IF( N==0 ) THEN
              G(N)=TALPHA
            ELSE
              G(N)=SIN(N*3.1415926*TALPHA)   /   (N*3.1415926)
           ENDIF

 100       CONTINUE
        DELTAW=2*WR/200
        SUMINT=0

        DO 300 I=-100,100
           W=I*DELTAW
           WTEMP=(W-WN)/WG
           IF( WTEMP == INT(WTEMP) ) THEN
              LOWLIM=WTEMP
           ELSE
              LOWLIM=WTEMP+1
           ENDIF

           HILIM=(W+WN)/WG

           SUMSUM=0
           DO 200 N = LOWLIM,-1
 200          SUMSUM=SUMSUM+G(N)*G(N)
           DO 210 N = 1,HILIM
```

```
210            SUMSUM=SUMSUM+G(N)*G(N)


            SUMINT=SUMINT + SUMSUM*DELTAW

300         CONTINUE

         DEN=SUMINT/(1-TALPHA)**2
         NUM=2*WR

         ALPHARA(IALPH)=TALPHA
500      SNROSNR4(IALPH)=10*LOG10(NUM/DEN)



        CALL INITT(IDELAY)
       CALL BINITT

       CALL CHECK( ALPHARA(0),SNROSNR4(0) )
        CALL DSPLAY( ALPHARA(0),SNROSNR4(0) )

       ENCODE(COMMENT,600)
 600 FORMAT('ALPHA')
       CALL WRITEK(457,5,5,0)
        ENCODE(COMMENT,610)
 610 FORMAT('IMPROVEMENT IN DB')
       CALL WRITEK(10,575,17,1)
        ENCODE(COMMENT,620)
 620 FORMAT('THEORETICAL RESULTS FOR PRESENT SETUP & VARIOUS
               ALPHAS')
       CALL WRITEK(107,745,54,0)
       ENCODE(COMMENT,630)(WCO(I)/(2*PI),I=1,3),GATEFREQ
 630 FORMAT('MESSAGE=',F5.0,' HZ. RECEIVE=',F5.0,' HZ.
NOISE=',F5.0,
    +       ' HZ. GATING=',F5.0' HZ.')
        CALL WRITEK(0,720,68,0)

        CALL FINITT


       RETURN
       END
```

```
PROG PSD
      SUBROUTINE PSD
INCLUDE THESIS.COM


      CALL PSD0
      CALL PSD1
      CALL PSD2

      RETURN
      END
```

```
PROG PSDO
     SUBROUTINE PSDO
INCLUDE THESIS.COM

   5 WRITE(LOU,*)'ENTER SIGNAL TO BE TESTED'
     WRITE(LOU,20)( I,SIGNAM(I),I=1,8 )
  20 FORMAT(' ',I1,') ',A)

     READ(LIU,*,ERR=5,END=1000)PSDNAM

     IF( PSDNAM < 1  .OR.  PSDNAM > 8 ) GO TO 5

     WRITE(LOU,30) FCO
  30 FORMAT(' MESSAGE CUTOFF FREQUENCY IS ',F4.0,' HZ.' )
   40 WRITE(LOU,*) 'ENTER LOWEST FREQUENCY TO BE TESTED'
     READ(LIU,*,ERR=40,END=1000) PSDLOW
  50 WRITE(LOU,*) 'ENTER HIGHEST FREQUENCY TO BE TESTED  < 5500
     HZ'
      READ(LIU,*,ERR=50,END=1000) PSDHIGH
     IF( PSDHIGH > 5500 ) GO TO 40

   60 WRITE(LOU,*) 'HOW MANY POINTS TO BE TESTED  <1501 '
     READ(LIU,*,ERR=60,END=1000) PSDNPTS
     IF( PSDNPTS > 1500 ) GO TO 60

  65 WRITE(LOU,70)
  70 FORMAT(' ENTER 1 FOR LINEAR',/,' ENTER 2 FOR LOGRITHMIC' )
     READ(LIU,*,ERR=65,END=1000)PLOTYP
     IF( PLOTYP < 1  .OR. PLOTYP > 2 ) GO TO 65

     RETURN
1000 CALL STOPIT
     END
```

```
PROG PSD1
     SUBROUTINE PSD1
INCLUDE THESIS.COM
NWARN

     CLWORD=PSDNAM-1
     CALL CLSR(RADCON,4,BTEST(CLWORD,29),1)
     CALL CLSR(RADCON,5,BTEST(CLWORD,30),1)
     CALL CLSR(RADCON,6,BTEST(CLWORD,31),1)

     IF( PLOTYP == LINEAR ) THEN
         PSDINC=( PSDHIGH-PSDLOW )/( PSDNPTS-1 )
     ELSE
        PSDINC=( PSDHIGH/PSDLOW )**(1./( PSDNPTS-1 ))
      ENDIF

     PSDFRQ=PSDHIGH
     GAINLOG=4

     XAXIS(1)=PSDNPTS
     PSDARA(1)=PSDNPTS

      DO I=1,PSDNPTS

        WRITE(LOU,90) PSDFRQ
 90     FORMAT(' INTEGRATING ',F6.1, ' HERTZ.')



        DO IEXP=-1,4
           IF( PSDFRQ < .555*10.**IEXP ) GO TO 100
          END DO

 100    CALL INTCL(IEXP,RADCON,7,8,9)

        PNRMFRQ=PSDFRQ/(.555*10.**IEXP)

         CALL DCASET(RADCON,'C510',PNRMFRQ,1)

        PSDTIME=NCYCLE/PSDFRQ


 200    NMSEC=1000*PSDTIME
        FOURGAIN=1./PSDTIME
```

```
CALL INTCL(GAINLOG,MONCON,0,1,2)
CALL CLSR(MONCON,3,1,1)    ;SET IC LINE
CALL CLSR(MONCON,4,0,1)    ;RESET HOLD LINE

ICWAIT= 1 + 40*0.4**GAINLOG
CALL WAIT(ICWAIT,1,ISTAT) ;WAIT FOR INTEGRATE CAP. TO
DISCHARGE



CALL CLSR(MONCON,3,0,1)   ;RESET IC LINE
CALL WAIT(NMSEC,1,ISTAT)  ;ALLOW INTEGRATION OVER
NCYCLE CYCLES
CALL CLSR(MONCON,4,1,1)   ;SET HOLD LINE

CALL QRDALO(MONCON,'A000',SINAMP,IERR)
CALL QRDALO(MONCON,'A002',COSAMP,IERR)

CALL QTEFF7(MONCON,0,OVERLOAD,IERR)
 CALL QTEFF7(MONCON,0,OVERLOAD,IERR)
AMP=SQRT(SINAMP*SINAMP  +  COSAMP*COSAMP)

IF( AMP < .05  & GAINLOG  < 4 ) THEN

    GAINLOG=GAINLOG+1
   PSDTIME=1.1*PSDTIME
   GO TO 200

ELSE IF( OVERLOAD & GAINLOG > -1 )  THEN

   GAINLOG=GAINLOG-1
   GO TO 200

END IF

AMP=AMP*FOURGAIN/10**GAINLOG
PSDARA(I+1)=DECTODB(AMP)
XAXIS(I+1) =PSDFRQ


CALL DCASET(MONCON,'C030',AMP,1)

IF( PLOTYP==LINEAR ) THEN
   PSDFRQ=PSDFRQ - PSDINC
   ELSE
      PSDFRQ=PSDFRQ / PSDINC
   END IF


END DO


RETURN
END
```

```
PROG PSD2
      SUBROUTINE PSD2
INCLUDE THESIS.COM


     CALL INITT(IDELAY)
     CALL BINITT
     CALL XTYPE(PLOTYP)      ;LINEAR OR LOG X-AXIS SELECTION
     CALL CHECK(XAXIS(1),PSDARA)
      CALL DSPLAY(XAXIS(1),PSDARA)

     ENCODE(COMMENT,300)
 300 FORMAT('POWER IN DB')
     CALL WRITEK(0,508,11,1)
     ENCODE(COMMENT,400)
 400 FORMAT('FREQUENCY IN HERTZ')
     CALL WRITEK(377,10,18,0)
     ENCODE(COMMENT,500)SIGNAM(PSDNAM)
 500 FORMAT('PSD OF ',A60)
     CALL WRITEK(100,730,67,0)

     CALL TINPUT(II)

     RETURN
      END
```

```
PROG RING
NOPTIMIZE
     SUBROUTINE RING(NTIMES,LU)

     SENDS COMMAND TO LU FOR BELL N TIMES

INCLUDE THESIS.COM

     IBELL=Y'07070707'
     FC=Y'29'

     DO 100,I=1,NTIMES

      CALL SYSIO(IDUM,FC,LU,IBELL,1,0)

     CALL WAIT(100,1,ISTAT)

 100 CONTINUE

     RETURN
     END
```

```
PROG SAMPLE
      SUBROUTINE SAMPLE
INCLUDE THESIS.COM

      CALL SAMPLE0
 100 CALL SAMPLE1
     CALL SAMPLE2
      CALL YESRNO('SAMPLE AGAIN ?')
     IF( YES )GO TO 100

      RETURN
     END
```

```
PROG SAMPLEO
     SUBROUTINE SAMPLEO
INCLUDE THESIS.COM



 10  WRITE(LOU,20) FCO
 20  FORMAT(' MESSAGE CUTOFF FREQUENCY IS 'F6.1,' HERTZ.')
     WRITE(LOU,*)'ENTER SAMPLING FREQUENCY RELATIVE TO     '
      WRITE(LOU,*)'MESSAGE CUTOFF FREQUENCY'

     READ(LIU,*,ERR=10,END=1000)RELSAMPL
     SAMPLFRQ=FCO*RELSAMPL

  30 WRITE(LOU,*)'HOW MANY POINTS  (LESS THAN 1025)'
     READ(LIU,*,ERR=30,END=1000) NSAMPLE

     IF( NSAMPLE > 1024 ) GO TO 30

     RETURN

1000 CALL STOPIT
     RETURN

     END
```

```
PROG SAMPLE1
      SUBROUTINE SAMPLE1
INCLUDE THESIS.COM

     INTEGER REGSAV(16)


     DO 100 I=2,7
        IF( DLAFRQ(I) > SAMPLFRQ  ) GO TO 110
 100    CONTINUE

     IF( I > 7 ) I=7
 110 CALL
NTRPLT(SAMPLFRQ,RSAMDLA,DLAFRQ(I-1),DLAFRQ(I),DLACNT(I-1),
     :             DLACNT(I)  )

     SAMPLDLA=RSAMDLA
     IF( SAMPLDLA < 1 ) SAMPLDLA=1

     MAXINDEX=20*(NSAMPLE-1)

ASSM
SETS REGSAV
USES REGSAV

USY     EQU      8
REATER  EQU      2
DCADR   EQU     15
MNDREG  EQU     14
ELAYR   EQU     13
NDEX    EQU     12
SRREG   EQU     11
OLDREG  EQU     10
LOWREG   EQU     10

        ENTRY SAMPLBEG,SAMPLEND


SAMPLBEG STM       LOWREG,REGSAV

        LHI      ADCADR,X'6A'
        SSR      ADCADR,SSRREG
        BTBS     BUSY,1
```

```
         LHI      CMNDREG,0
         WHR      ADCADR,CMNDREG           SEND BLOCK LIMITS

         LHI      INDEX,0

AMPLOOP  L        HOLDREG,SAMPLDLA
LALOOP   SHI      HOLDREG,1
         BTC      GREATER,DLALOOP

         LHI      CMNDREG,X'E1'
         OCR      ADCADR,CMNDREG

         LHI      CMNDREG,X'E0'
          OCR      ADCADR,CMNDREG

         LHI      CMNDREG,X'08'
         OCR      ADCADR,CMNDREG

          SSR      ADCADR,SSRREG
         BTBS     BUSY,1

         RHR      ADCADR,HOLDREG
         STH      HOLDREG,SAMPLARA+00(INDEX)

         RHR      ADCADR,HOLDREG
          STH      HOLDREG,SAMPLARA+02(INDEX)

         RHR      ADCADR,HOLDREG
          STH      HOLDREG,SAMPLARA+04(INDEX)

         RHR      ADCADR,HOLDREG
         STH      HOLDREG,SAMPLARA+06(INDEX)

         RHR      ADCADR,HOLDREG
         STH      HOLDREG,SAMPLARA+08(INDEX)

         RHR      ADCADR,HOLDREG
         STH      HOLDREG,SAMPLARA+10(INDEX)

         RHR      ADCADR,HOLDREG
         STH      HOLDREG,SAMPLARA+12(INDEX)

         RHR      ADCADR,HOLDREG
         STH      HOLDREG,SAMPLARA+14(INDEX)
```

```
        AHI       INDEX,20
        C         INDEX,MAXINDEX
        BFC       GREATER,SAMPLOOP

AMPLEND LM        LOWREG,REGSAV

FORT


     DO I=1,NSAMPLE
         SAMPLARA( 9,I)=SAMPLARA(1,I)-SAMPLARA(2,I) ;NOISE
         EST. ERROR
        SAMPLARA(10,I)=SAMPLARA(3,I)-SAMPLARA(4,I) ;SIGNAL EST.
        ERROR
        END DO



      RETURN

     END
```

```
PROG SAMPLE2
     SUBROUTINE SAMPLE2
INCLUDE THESIS.COM


    5 WRITE(LOU,10)(I,SIGNAM(I),I=0,10)
   10 FORMAT(' ENTER 2 DESIRED SIGNALS TO BE DISPLAYED',/,
    ; 11( ' ',I2,') ',A,/ )   )

     READ(LIU,*,ERR=5,END=1000)SIGNAL1,SIGNAL2
     IF(SIGNAL1<0 .OR. SIGNAL1>10 .OR. SIGNAL2<0 .OR.
     SIGNAL2>10)
    ;   GO TO 5

     IF( SIGNAL1 == 0   &  SIGNAL2 == 0 ) RETURN


     DO I=1,NSAMPLE


        IF( SIGNAL1 == 0 ) THEN
           SIG1ARA(I)=0
        ELSE
           SIG1ARA(I)=SAMPLARA(SIGNAL1,I)/32768.
        ENDIF


         IF( SIGNAL2 == 0 ) THEN
           SIG2ARA(I)=0
        ELSE
           SIG2ARA(I)=SAMPLARA(SIGNAL2,I)/32768.
        ENDIF


        END DO

     SAMPLTIM=1/SAMPLFRQ

     CALL INITT(IDELAY)
      CALL BINITT
     CALL NPTS(NSAMPLE)

      DO I=1,NSAMPLE
        N=I
```

```
         XAXIS(I)=(N-1)*SAMPLTIM
          END DO

      CALL DLIMY(-1.,+1.)

      CALL CHECK(XAXIS,SIG1ARA)
      CALL DSPLAY(XAXIS,SIG1ARA)

      CALL LINE(4)
      CALL CHECK(XAXIS,SIG2ARA)
      CALL CPLOT(XAXIS,SIG2ARA)

      ENCODE(COMMENT,500)
  500 FORMAT('SOLID -> ')
      CALL WRITEK(0,25,9,0)
      ENCODE(COMMENT,501) SIGNAM(SIGNAL1)
  501 FORMAT(A60)
      CALL WRITEK(135,25,55,0)
C

      ENCODE(COMMENT,600)
  600 FORMAT('DASHED-> ')
      CALL WRITEK(0,0,9,0)
      ENCODE(COMMENT,601)SIGNAM(SIGNAL2)
  601 FORMAT(A60)
      CALL WRITEK(135,0,55,0)

      CALL TINPUT(I)
      CALL FINITT(0,760)

      GO TO 5

      RETURN

 1000 CALL STOPIT
      RETURN
      END
```

```
$PROG SETFLT
     SUBROUTINE SETFLT(FLTID)
INCLUDE THESIS.COM

     INTEGER FLTID


     IF( FLTID==TRANSMIT ) THEN

        BQDREF=0
        DO 100 POLNUM=1,4

            WC=CABS( POLES(FLTID,POLNUM) )
           ZETA=-REAL( POLES(FLTID,POLNUM) )/WC

           BQDNUM=BQDREF+POLNUM

           DCAVAL=WC/(10.*INTGAN)
           CALL DCASET(RADCON,BQDDCA(BQDNUM,1), DCAVAL,1)
           CALL DCASET(RADCON,BQDDCA(BQDNUM,2), DCAVAL,1)
           CALL DCASET(RADCON,BQDDCA(BQDNUM,3),-DCAVAL,1)


           DCAVAL=2*ZETA*WC/(10.*INTGAN)
           CALL DCASET(RADCON,BQDDCA(BQDNUM,4), DCAVAL,1)
 100       CONTINUE

     ELSE IF( FLTID==LOPASS ) THEN

        BQDREF=4

        DO 200 POLNUM=1,4

           WC=CABS( POLES(FLTID,POLNUM) )
           ZETA=-REAL( POLES(FLTID,POLNUM) )/WC

           DO 200 OFFSET=0,4,4

              BQDNUM=BQDREF+POLNUM+OFFSET
              DCAVAL=WC/(10.*INTGAN)


              CALL DCASET(RADCON,BQDDCA(BQDNUM,1), DCAVAL,1)
              CALL DCASET(RADCON,BQDDCA(BQDNUM,2), DCAVAL,1)
```

```
                CALL DCASET(RADCON,BQDDCA(BQDNUM,3),-DCAVAL,1)
               CALL DCASET(MONCON,BQDDCA(BQDNUM,1), DCAVAL,1)
               CALL DCASET(MONCON,BQDDCA(BQDNUM,2), DCAVAL,1)
               CALL DCASET(MONCON,BQDDCA(BQDNUM,3),-DCAVAL,1)

               DCAVAL=2*ZETA*WC/(10.*INTGAN)

                CALL DCASET(RADCON,BQDDCA(BQDNUM,4), DCAVAL,1)
               CALL DCASET(MONCON,BQDDCA(BQDNUM,4), DCAVAL,1)

200            CONTINUE

      ELSE

         BQDREF=12
         DO 300 POLNUM=1,4

            WC=CABS( POLES(FLTID,POLNUM) )
             ZETA=-REAL( POLES(FLTID,POLNUM) )/WC

            BQDNUM=BQDREF+POLNUM

            DCAVAL=WC/(10.*NINTGAN)
            CALL DCASET(RADCON,BQDDCA(BQDNUM,1), DCAVAL,1)
            CALL DCASET(RADCON,BQDDCA(BQDNUM,2), DCAVAL,1)
            CALL DCASET(RADCON,BQDDCA(BQDNUM,3),-DCAVAL,1)


            DCAVAL=2*ZETA*WC/(10.*NINTGAN)
            CALL DCASET(RADCON,BQDDCA(BQDNUM,4), DCAVAL,1)
 300         CONTINUE

         BQDREF=16

         DO 300 POLNUM=1,4

            WC=CABS( POLES(FLTID,POLNUM) )
            ZETA=-REAL( POLES(FLTID,POLNUM) )/WC

C            BQDNUM=BQDREF+POLNUM

            DCAVAL=WC/(10*HINTGAN)
            CALL DCASET(RADCON,BQDDCA(BQDNUM,1), DCAVAL,1)
            CALL DCASET(RADCON,BQDDCA(BQDNUM,2), DCAVAL,1)
            DCAVAL=2*ZETA*WC/(10*HINTGAN)
            CALL DCASET(RADCON,BQDDCA(BQDNUM,3), DCAVAL,1)

300         CONTINUE

      ENDIF

      RETURN
      END
```

```
$PROG SNR
      SUBROUTINE SNR

      CALL SNR0
      CALL SNR1

      RETURN

      END
```

```
PROG SNRO
     SUBROUTINE SNRO
INCLUDE THESIS.COM

     LOGICAL ASKBANWTH

     ASKBANWTH=.TRUE.


     IF( FCO == 0 ) THEN

        WRITE(LOU,*)'FILTER PARAMETERS MUST BE SET FIRST'
        CALL FLTO
        ASKBANWTH=.FALSE.

     END IF


     IF( ALPHA == 0 ) THEN

        WRITE(LOU,*)'ALPHA OF GATING PARAMETERS MUST BE SET
        FIRST'
        CALL GATEO
        CALL GATE1

     ENDIF


     IF( ASKBANWTH ) THEN

  40    WRITE(LOU,50)FCO

  50    FORMAT(' MESSAGE CUTOFF FREQUENCY IS ',F6.1,' . ',/,
   :          ' ENTER DESIRED RELATIVE NOISE BANDWIDTH.')
        READ(LIU,*,ERR=40,END=1000)RELFRQ(NOISE)
        WCO(NOISE)=RELFRQ(NOISE)*WCO(TRANSMIT)

      END IF

      SNRLIM=DECTODB( ALPHA )
 100 WRITE(LOU,*)'ENTER DESIRED TOTAL SIGNAL TO NOISE RATIO IN
     DB'
     WRITE(LOU,200) SNRLIM
 200 FORMAT(' SNR >= ',F5.1,' DB')
```

```
      READ(LIU,*,END=1000,ERR=100)SNRDB
      IF( SNRDB < SNRLIM ) GO TO 100
       RETURN
 1000 CALL STOPIT
      RETURN
      END
```

```
PROG SNR1
     SUBROUTINE SNR1
INCLUDE THESIS.COM

      CALL FLT1


     DO 100 I=1,12

        IF( WCO(NOISE) < MESCOMP(I+1,1) ) GO TO 200
 100    CONTINUE

 200 CALL NTRPLT( WCO(NOISE) , PEAK , MESCOMP(I,1) ,
     +              MESCOMP(I+1,1) , MESCOMP(I,2) ,
      MESCOMP(I+1,2)  )

      RNSAMP=MESAMP
      NOISEGN=RNSAMP/PEAK

      DCAVAL=1./NOISEGN
      CALL DCASET(RADCON,'C313',DCAVAL,1)

      SNRDEC=DBTODEC(SNRDB)

     DCAVAL= ALPHA * (1/SNRDEC)  ;LARGER ALPHA => MORE SIGNAL
     POWER
     CALL DCASET(RADCON,'C133',DCAVAL,1)

     RETURN

     END
```

```
PROG SNR2
     SUBROUTINE SNR2
INCLUDE THESIS.COM

     WRITE(LOU,100)SNRDB
 100 FORMAT(' PRESENT SET SIGNAL TO NOISE VALUE IS ',F6.2,'
     DB')
      RETURN

     END
```

```
$PROG SNRTST
      SUBROUTINE SNRTST
INCLUDE THESIS.COM

      CALL SNRTST0
       CALL SNRTST1

      RETURN
      END
```

```
PROG SNRTSTO
      SUBROUTINE SNRTSTO
INCLUDE THESIS.COM



   10  WRITE(LOU,20) FCO
  20  FORMAT(' MESSAGE CUTOFF FREQUENCY IS 'F6.1,' HERTZ.')
      WRITE(LOU,*)'ENTER SAMPLING FREQUENCY RELATIVE TO      '
      WRITE(LOU,*)'MESSAGE CUTOFF FREQUENCY'

      READ(LIU,*,ERR=10,END=1000)RELSAMPL
      SAMPLFRQ=FCO*RELSAMPL

   30 WRITE(LOU,*)'HOW MANY POINTS  (LESS THAN 10001)'
      READ(LIU,*,ERR=30,END=1000) NSAMPLE

      IF( NSAMPLE > 10000 ) GO TO 30

      RETURN

1000 CALL STOPIT
      RETURN

      END
```

```
PROG SNRTST1
     SUBROUTINE SNRTST1
INCLUDE THESIS.COM

      INTEGER REGSAV(16)


      DO 100 I=2,7
         IF( DLAFRQ(I) > SAMPLFRQ  ) GO TO 110
 100     CONTINUE

      IF( I > 7 ) I=7
  110 CALL
     NTRPLT(SAMPLFRQ,RSAMDLA,DLAFRQ(I-1),DLAFRQ(I),DLACNT(I-1),
    :            DLACNT(I)  )

     SAMPLDLA=RSAMDLA
      IF( SAMPLDLA < 1 ) SAMPLDLA=1

     MAXINDEX=2*(NSAMPLE-1)

ASSM
SETS REGSAV
USES REGSAV

USY     EQU        8
REATER  EQU        2
DCADR   EQU       15
MNDREG  EQU       14
ELAYR   EQU       13
NDEX    EQU       12
SRREG   EQU       11
IGNAL   EQU       10
EFORE   EQU        9
FTER    EQU        8
OWREG   EQU        8


AMPLBEG STM       LOWREG,REGSAV

        LHI       ADCADR,X'6A'
        SSR       ADCADR,SSRREG
        BTBS      BUSY,1
        LHI     CMNDREG,0
        WHR     ADCADR,CMNDREG          SEND BLOCK LIMITS

        LHI     INDEX,0

AMPLOOP L       DELAYR,SAMPLDLA
LALOOP  SHI     DELAYR,1
        BTC     GREATER,DLALOOP

        LHI     CMNDREG,X'E1'
        OCR     ADCADR,CMNDREG

        LHI     CMNDREG,X'E0'
        OCR     ADCADR,CMNDREG

        LHI     CMNDREG,X'08'
         OCR     ADCADR,CMNDREG

        SSR     ADCADR,SSRREG
         BTBS    BUSY,1

        RHR     ADCADR,SIGNAL

        RHR     ADCADR,AFTER
        SR      AFTER,SIGNAL
         STH     AFTER,AFTERARA(INDEX)

        RHR     ADCADR,BEFORE
        RHR     ADCADR,BEFORE
         RHR     ADCADR,BEFORE
        RHR     ADCADR,BEFORE
        RHR     ADCADR,BEFORE
        RHR     ADCADR,BEFORE

        SR      BEFORE,SIGNAL
        STH     BEFORE,BEFORARA(INDEX)

        AHI     INDEX,2
        C       INDEX,MAXINDEX
        BFC     GREATER,SAMPLOOP

AMPLEND LM      LOWREG,REGSAV

FORT
```

```
       BEFORPWR=0
      AFTERPWR=0
      DO I=1,NSAMPLE
         BEFORPWR=BEFORPWR + BEFORARA(I)**2
         AFTERPWR=AFTERPWR + AFTERARA(I)**2
         END DO

      SNRIMPRV=10*LOG10(BEFORPWR/AFTERPWR)

      WRITE(LOU,200)SNRIMPRV
  200 FORMAT(' SIGNAL TO NOISE RATIO IMPROVEMENT IS ',F5.1,'
             DB')


      RETURN

      END
```

```
PROG STOPIT
      SUBROUTINE STOPIT

      STOPIT ALLOWS ABORT OF ANY ROUTINE AND ENTRY TO INDEX2

       CALL INDEX2
      STOP
   10 RETURN
      END
```

```
PROG WRITEK

     SUBROUTINE WRITEK(XCOORD,YCOORD,NCHARS,VERHRZ)

     THIS ROUTINE MOVES THE CURSOR ON THE TEK: TO THE X,Y
     COORDINATES (XCOORD,YCOORD) , THEN DISPLAYS THE
     FIRST NCHARS CHARACTERS FROM THE ARRAY 'COMMENT'.
     IF VERHRZ=1, THE CURSOR WILL DISPLAY A VERTICAL COLUMN
     OF CHARACTERS , OTHERWISE A HORIZONTAL ROW OF CHARACTERS
C     WILL BE PRODUCED.

INCLUDE THESIS.COM

     INTEGER XCOORD,YCOORD,NCHARS,VERHRZ,CHRCTR,XC,YC,NC

     NC=NCHARS
     XC=XCOORD
     YC=YCOORD

     IF(NC>80)NC=80

     DO 100 I=1,NC
     CHRCTR=0

     CALL ILBYTE(CHRCTR,COMMENT,I-1)

      CALL NOTATE(XC,YC,1,CHRCTR)

     IF(VERHRZ .EQ. 1) THEN
        YC=YC-23
     ELSE
        XC=XC+15
     END IF

 100 CONTINUE

      RETURN
     END
```

```
PROG YESRNO
      SUBROUTINE YESRNO(QSTRNG)

$INCLUDE THESIS.COM


      CHARACTER QSTRNG(80),LENGTH*2,DUMARA*20,REPLY
      CHARACTER*2 FRMARA(10)
      DATA DUMARA / '( 1H ,  A1,3X,3HY/N)'/
      EQUIVALENCE(LENGTH,FRMARA(4)),(DUMARA,FRMARA)

      DO 50 I=1,80
      IF(QSTRNG(I) .EQ. '?')THEN
         NCHRS=I
         ENCODE(LENGTH,20) NCHRS
   20    FORMAT(I2)
         GO TO 70
      END IF
   50 CONTINUE

   70 WRITE(LOU,FMT=FRMARA)(QSTRNG(I),I=1,NCHRS)
      READ(LIU,100,ERR=70,END=300)REPLY
  100 FORMAT(A)

      IF( REPLY<>'Y' & REPLY<>'N' ) GO TO 70

      IF(REPLY .EQ. 'Y') THEN

          YES=.TRUE.

      ELSE

         YES=.FALSE.

      ENDIF

      RETURN
  300 CALL STOPIT
      RETURN
      END
```

LIST OF REFERENCES

Beyer, W. H. Standard Mathematical Tables. West Palm Beach: CRC Press, Inc., 1978.

Cooper, G. D., and McGillem, C. D. Probabalistic Methods of Signal and System Analysis. Chicago: Holt, Rinehart, and Winston, 1971.

Gardner, C. R. A Hybrid Simulation Incorporating Multiple Modulation Techniques. The Proceedings of the 1985 Summer Computer Simulation Conference. Chicago: North-Holland, 1985.

Lathi, B. P. Signals, Systems, and Controls. New York: Harper and Row, 1974.

Shanmugam, K. S. Digital and Analog Communication Systems. New York: John Wiley & Sons, 1979.

Ziemer, R. E., and Trantner, W. H. Principles of Communications. Boston: Houghton Mifflin Co., 1976.