THE PERFORMANCE OF A NOISE LEVELING AUTOMATIC GAIN CONTROL SYSTEM

205

by

DIANE MARIE VON THAER

B.S., Kansas State University, 1982

A MASTER'S THESIS

submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Electrical Engineering

KANSAS STATE UNIVERSITY

Manhattan, Kansas

1983

Approved by:

Donald R. Hummels

Major Professor

Table of Contents

List of Figures

I. Introduction

Surveillance receivers frequently consist of a broadband amplifier followed by a square-law detector and a low-pass filter. These receivers are commonly used to detect radar pulses which are then subsequently examined by more sophisticated equipment designed to estimate such parameters as pulse frequency, time-of-arrival, and pulse width. The receiver detection circuit simply compares the low-pass filter output with a preset threshold voltage which has been adjusted to establish a fixed rate of false alarms which occur when the received noise exceeds the threshold. Once set, it is important that the false alarm rate remain constant. The false alarm rate may vary, though, due to changes in receiver gain caused by bandswitching, aging, and temperature fluctuations. In some instances the receiver can be manually adjusted to account for such variations, but for satallite surveillance receivers in orbit maintenance is impossible. In this case, a noise leveling automatic gain control (AGC) system is used to keep the false alarm rate constant. The noise leveling AGC system is the principle subject of this study.

A noise leveling AGC system keeps the false alarm rate constant by sampling the out-of-band noise after the signal detection circuits. The detected noise voltage serves as a feedback control signal to increase or decrease the gain of the system. A block diagram of the noise leveling system is shown in Figure 1. There are two variations of the noise leveling system. One includes a Logarithmic Video Amplifier (LVA) before the video filter and the other variation uses a linear amplifier instead. There are nonlinearities in the system, such as the two square-law detectors, which make analysis of the system very difficult,

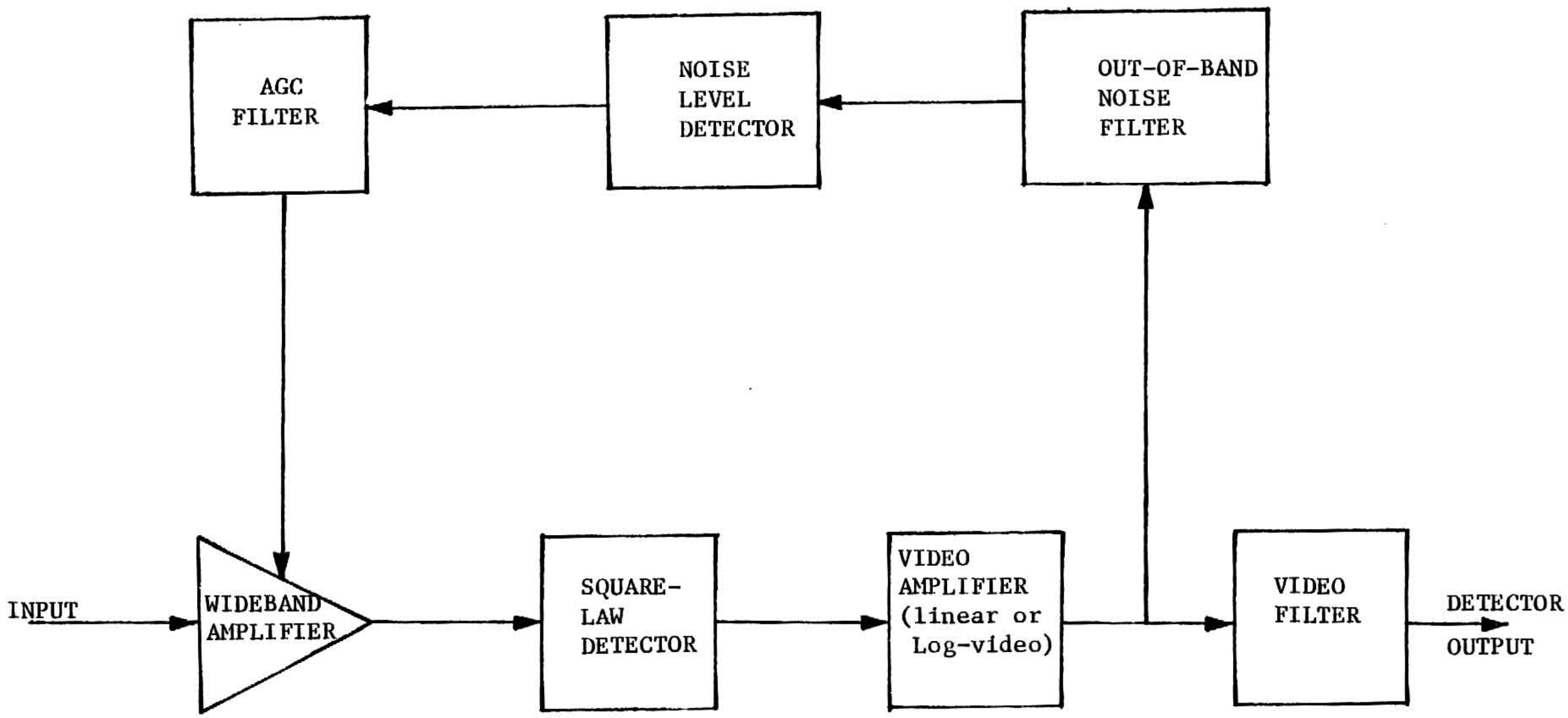


Figure 1. Noise Leveling AGC System

therefore, a Monte Carlo simulation was used to evaluate the system performance. From the simulation it is hoped to learn how to set the gains of the system, the system's response to pulses and other inputs, and factors concerning stability for the two variations of the system.

II. Development of a Mathematical Model

A mathematical model for the Noise Leveling AGC system is needed for simulation purposes. A low-pass equivalent model is derived for ease in simulation because high frequency functions are difficult and time consuming to simulate on a computer. A block diagram of the actual system is shown in Figure 2. Since this is a baseband approach most of the blocks are already low-pass functions. Only the wideband variable gain amplifier, the bandpass filter and the first detector require low-pass modeling.

The low-pass model is derived using the properties of complex envelopes [1]. A bandpass signal, $x(t)$, can be represented by its complex envelope, $\tilde{x}(t)$,

$$x(t) = \mathrm{Re}\left\{x(t)e^{j\omega_c t}\right\} \tag{1}$$

where ω_c is the center frequency in rad/s. This relation can be applied for both signal and noise processes. Equation (1) can be rewritten as

$$x(t) = |\tilde{x}(t)|\cos\left(\omega_c t + \phi(t)\right) \tag{2}$$

where $|\tilde{x}(t)|$ is the envelope and $\phi(t)$ is the phase angle of $\tilde{x}(t)$. The complex envelope of a bandpass signal is independent of the carrier frequency and it is slowly varying and therefore easy to simulate.

Bandpass Filter

The bandpass filter is modeled in a similar manner as the signal using complex envelopes. A bandpass filter with impulse response, $h(t)$, has an equivalent low-pass impulse response, $\tilde{h}(t)$, defined by

$$h(t) = 2\ \mathrm{Re}\left\{\tilde{h}(t)e^{j\omega_c t}\right\} \tag{3}$$

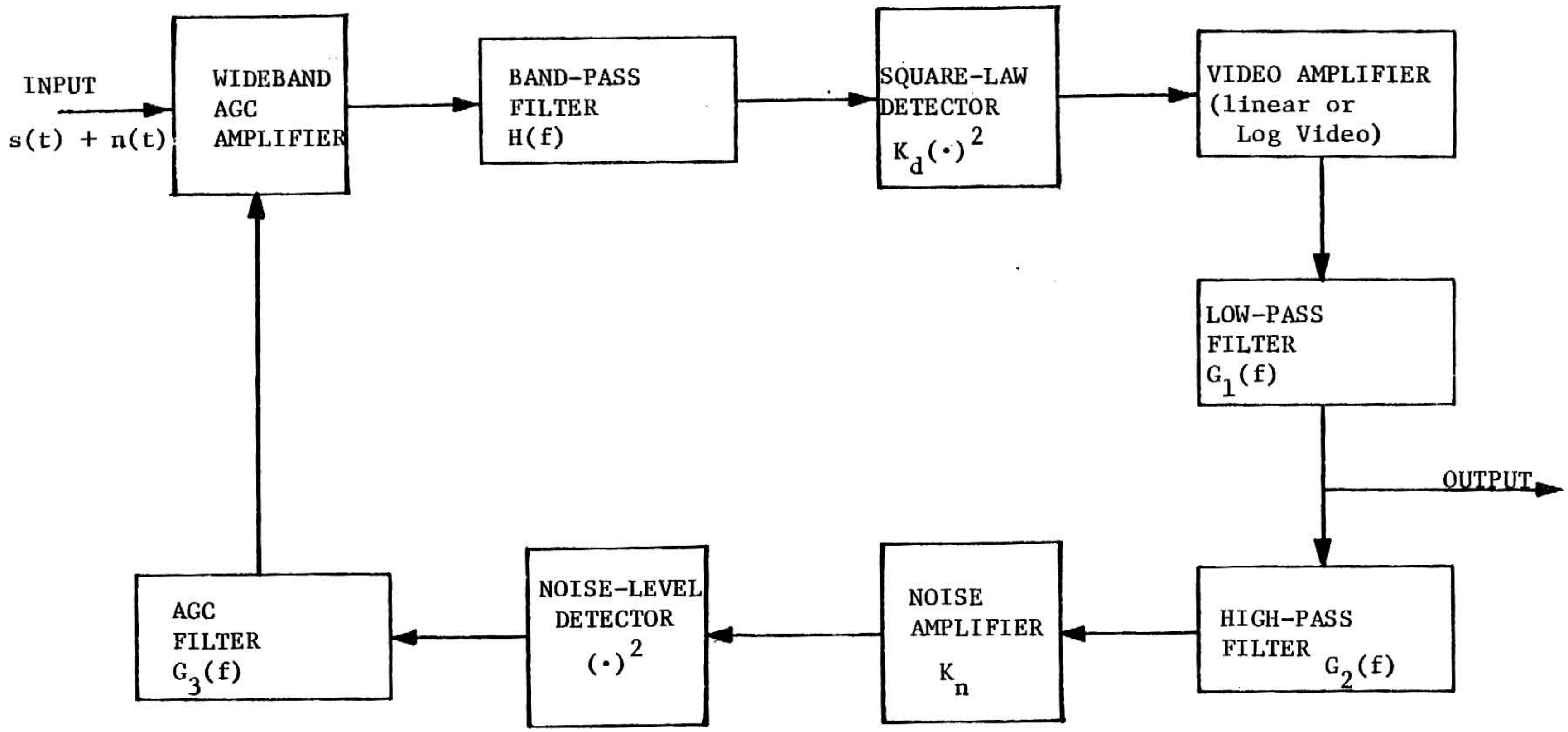


Figure 2. Block Diagram of Noise Leveling System

where ω_c is the center frequency. From this definition, the equivalent low-pass transfer function, $\tilde{H}(f)$, can be derived [1] as

$$H(f) = \tilde{H}(f - f_c) + \tilde{H}^*(-f - f_c) \tag{4}$$

where H(f) and $\tilde{H}(f)$ are the Fourier transforms of h(t) and $\tilde{h}(t)$, respectively. This relationship can be inverted to find the equivalent low-pass transfer function in terms of the actual bandpass transfer function

$$\tilde{H}(f) = [H(f + f_c)]_{\text{Low-pass Term}}. \tag{5}$$

The low-pass equivalent model will have a bandwidth equal to one-half the RF bandwidth. All transform relations for linear systems are also valid for low-pass complex envelope models. As an example, the transform of the output of a filter with transfer function, H(f), given an input with transform, X(f) is

$$Y(f) = H(f)X(f). \tag{6}$$

When using complex envelope equivalent models, the transform of the low-pass equivalent output is

$$\tilde{Y}(f) = \tilde{H}(f)\tilde{X}(f). \tag{7}$$

Detector

The low-pass model for the detector can be found by examining its input-output relationship. The output of a square-law detector is given by

$$y(t) = x^2(t). \tag{8}$$

If the input, x(t), is represented using its complex envelope given as

$$x(t) = |\tilde{x}(t)| \cos\left(\omega_c t + \phi(t)\right), \tag{9}$$

then

$$y(t) = |\tilde{x}(t)|^2 \cos^2\left(\omega_c t + \phi(t)\right). \tag{10}$$

Applying a trigonometric relation,

$$y(t) = \frac{|\tilde{x}(t)|^2}{2} + \frac{|\tilde{x}(t)|^2}{2} \cos\left(2\omega_c t + 2\phi(t)\right). \tag{11}$$

Since the square-law detector is followed by a low-pass filter the double frequency term will be eliminated. Therefore, the equivalent low-pass model for the square-law detector is

$$\tilde{y}(t) = |\tilde{x}(t)|^2. \tag{12}$$

The factor, $\frac{1}{2}$ is included in the gain K_d at the detector output.

Variable Gain Amplifier

The wideband variable gain amplifier has typical gain control characteristics as shown in Figure 3. As can be seen, the curve is very nearly linear, therefore a linear approximation is used to model the device. If the nominal gain of the amplifier under normal operating conditions is A_0, then the appropriate linear approximation is

$$A = A_0 - K_a \left(v_{agc} - V_{nom}\right) \tag{13}$$

where V_{nom} is the AGC voltage required to produce the nominal gain and K_a is the slope of the characteristic curve at V_{nom}. The linear approximation used in Equation 13 is the only approximation in the development of the mathematical model, although it would not be extremely difficuly to model the non-linearity in this case. The gain of the amplifier is clamped with a lower limit of 0.001 to simulate the operating range of an actual device.

Equivalent Low-Pass Model

The mathematical model for the noise leveling AGC system simulation includes the low-pass equivalent models just derived. The model for the LVA was derived in a previous study [2]. The LVA is modeled as

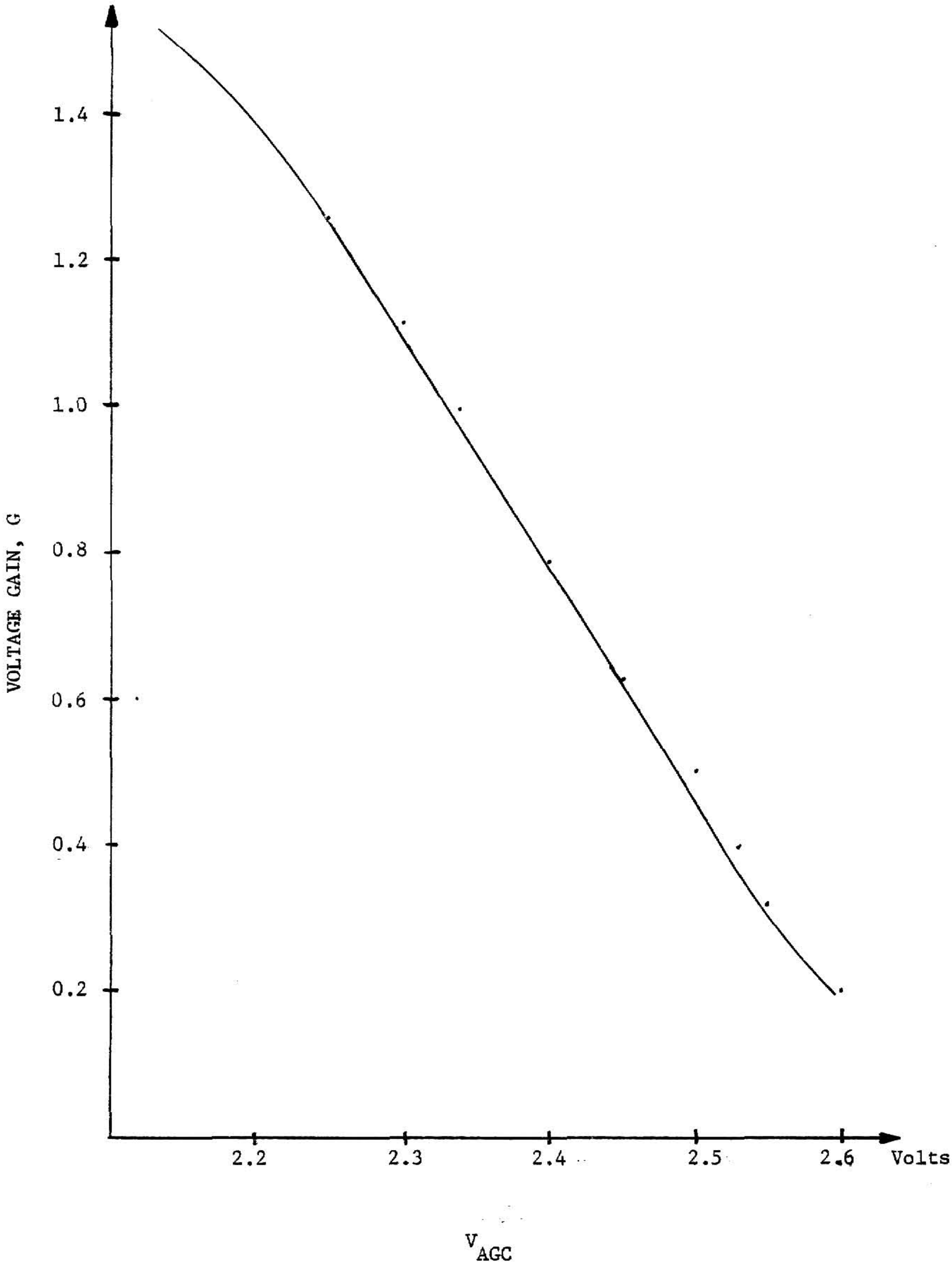


Figure 3. Typical Gain Control Characteristic

$$y(t) = \frac{1}{10\ln 10}\ln\left(x(t) + E_b\right) + 0.6 \qquad (14)$$

where E_b is a bias voltage equal to 10^{-6} volts. The bias is selected to yield an output of zero volts with no input signal. A plot of the LVA function versus RF input signal power in dBm is shown in Figure 4. The entire low-pass model for the simulation of the noise leveling AGC system is shown in Figure 5.

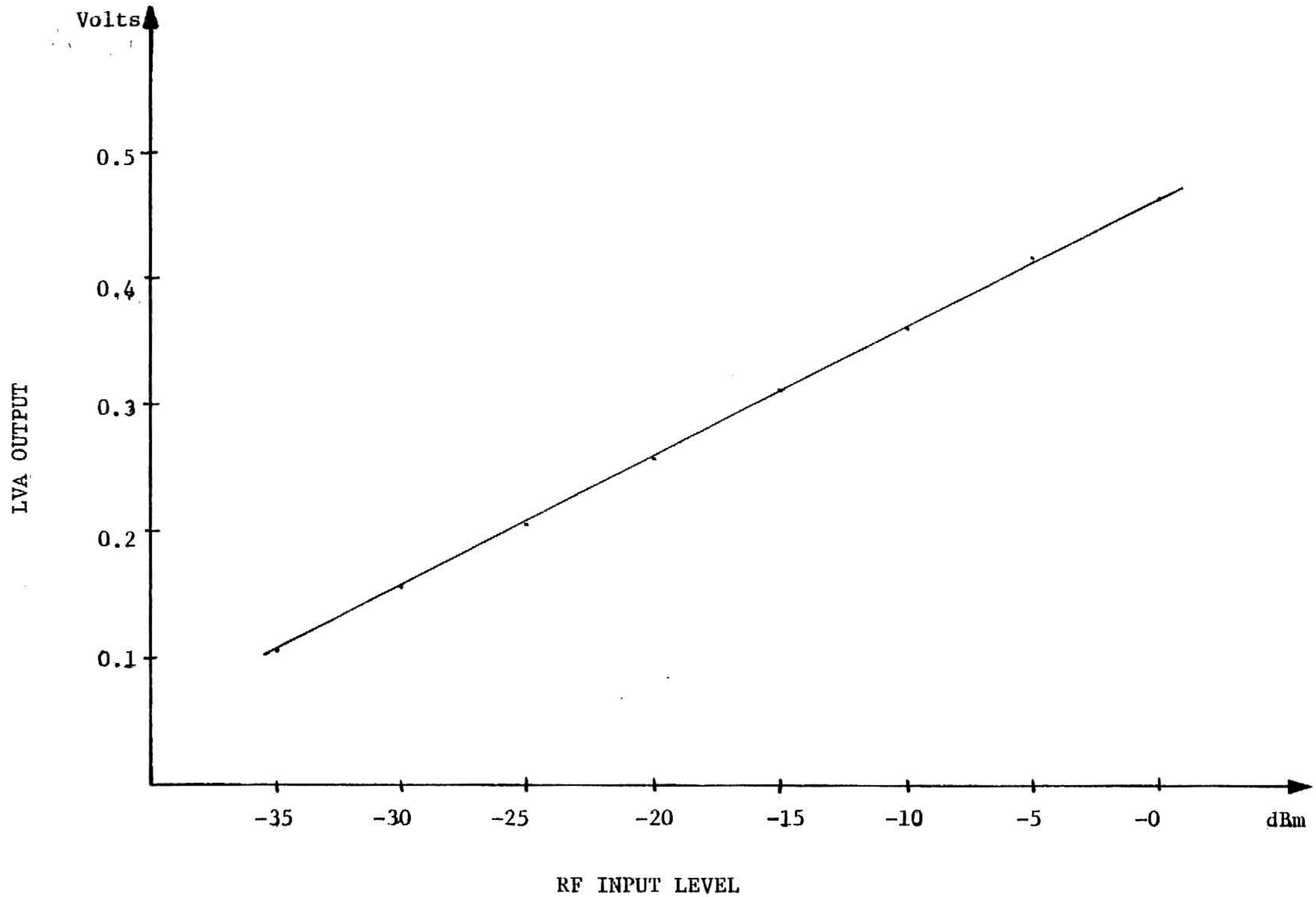


Figure 4. LVA Transfer Characteristics

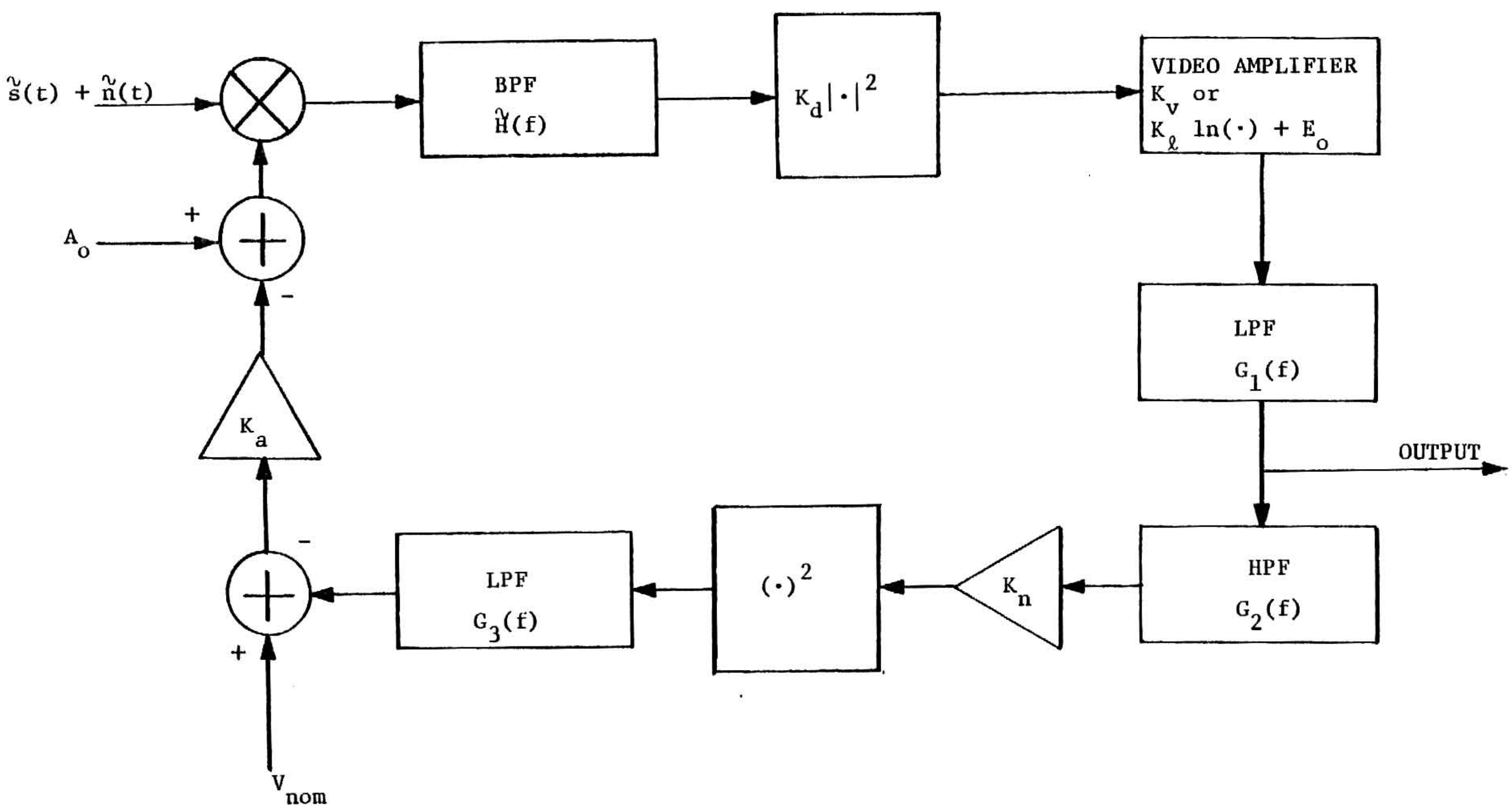


Figure 5. Equivalent Low-Pass Model of the Noise Leveling System

III. Description of the Simulation

The noise leveling AGC system is simulated on a digital computer to evaluate its performance. The mathematical model shown in Figure 5 is implemented in Fortran on a Data General Nova-4X computer. The simulation is done entirely in the time domain due to the dependence of the amplifier gain on the system response to previous inputs. The method is to sample the signal and noise and calculate the system response before considering the next sample. The necessity of a time domain simulation prevents the use of fast computation methods, such as the Fast Fourier Transform. Because of this computer run-times tend to be long. The Fortran program for the simulation is called NLAGC and a copy of it is in Appendix B.

The Input Signal

The input signal for the simulation is generated within the program. The signals of interest in this study are a CW signal and a modulated pulse. Since the simulation uses a low-pass equivalent model, the input signal is the complex envelope of either of these high frequency signals. These complex envelopes are readily determined as follows.

The CW signal can be represented by

$$s(t) = A \cos \omega_c t = \frac{A\, e^{j\omega_c t} + A\, e^{-j\omega_c t}}{2} . \qquad (15)$$

The complex envelope relation given in equation (1) is

$$s(t) = \mathrm{Re}\left\{\tilde{s}(t)\, e^{j\omega_c t}\right\} = \frac{\tilde{s}(t)\, e^{j\omega_c t} + \tilde{s}^*(t)\, e^{-j\omega_c t}}{2} . \qquad (16)$$

By comparison, $\tilde{s}(t) = A$ for the CW case.

The modulated pulse signal can be described by

$$s(t) = p(t)\cos\omega_c t = \frac{p(t)\,e^{j\omega_c t} + p(t)\,e^{-j\omega_c t}}{2} \tag{17}$$

where p(t) is a periodic pulse train. Comparison of this to the complex envelope relation in Equation (16) above gives $\tilde{s}(t) = p(t)$ for the modulated pulse case. For the simulation, p(t) is a periodic train of trapezoidal pulses. The program user chooses either type of signal and supplies the specific parameters; input power, pulse width, pulse rise time, and pulse frequency. The input power is entered in dBm and the signal amplitude is calculated in the program.

Noise

The input noise is also generated within the simulation program. It is assumed to be zero-mean white Gaussian noise. Each noise sample is formed using a random number generator subroutine with a Gaussian distribution. To represent a complex envelope of noise, two independent random values are generated for each sampling instant, a real component and an imaginary component. A property of complex envelopes is that the complex envelope of a sum is the sum of the complex envelopes. Using this property the total complex envelope input to the simulated system is the signal component, $\tilde{s}(k)$, plus the noise component, n(k), where k is the discrete time index.

The variance of the noise is set by determining the variance corresponding to the desired power at the output of the prefilter. The desired power, Pn_i, is specified by the user. The relation between noise power after the prefilter and the variance is

$$\sigma^2 = \frac{Pn_i}{2H_0^2\,T_s\,B_n}\;, \tag{18}$$

where B_n is the noise bandwidth of the prefilter low-pass equivalent and H_o is the voltage gain of the prefilter. The last equation is derived by modifying the discrete power density spectrum of noise by the power response of the prefilter. The noise density spectrum can be found to be a constant, $\frac{\sigma^2}{N}$, where σ^2 is the vairance of each sample and N is the number of samples used in the calculation. If H_k denotes the value of the equivalent low-pass transfer function at the k^{th} frequency, then the k^{th} component will have power $\frac{\sigma^2}{N} |H_k|^2$. The total output power is then

$$Pn_i = \sum_{k=0}^{N-1} \frac{\sigma^2}{N} |H_k|^2 = \frac{\sigma^2}{N} \sum_{k=0}^{N-1} |H_k|^2 . \tag{19}$$

An approximation is

$$2B_n \cong \frac{1}{T H_0^2} \sum_{k=0}^{N-1} |H_k|^2 , \tag{20}$$

where T is the length of time of the noise sequence. The sampling time is thus, $T_s = \frac{T}{N}$. Combining Equations (19) and (20) yields the result in Equation 18. Once the variance is calculated it is used as a parameter in the calling statement for the random Gaussian distribution, n(k) = GAUSS (variance, mean).

Modeling Filter Operations

The filtering operations within the simulation must be modeled in the time domain because of the dependence of the control voltage on the response of the previous input. The analog filter transfer functions were transformed into difference equations which are easily implemented on a digital computer. A generalized function subprogram was developed which could be used as a one, two, three, or four pole low-pass filter. The subprogram incorporates the difference equations for all four cases.

The inputs to the subprogram are the 3 dB bandwidth, the sampling time, the number of poles, and the present signal sample. Also a condition flag is passed to the subprogram which indicates the first call to the filter so that the filter can be initialized the first time only. The calling sequence for the function subprogram is

Y = FILT1 (X, TS, B3, N, FIRST),

where X is the signal sample, TS is the sampling time, B3 is the bandwidth, N is the number of poles and FIRST is the flag. The output of the filter is Y. Since four independent low-pass filters are needed in the system simulation, there are four identical subprograms called FILT1, FILT2, FILT3, and FILT4.

A separate subroutine was developed for the high-pass noise filter. It can also act as a one, two, three, or four pole Butterworth high-pass filter. The inputs to the subprogram are the same as for the low-pass subprogram except the 3 dB cutoff frequency is relevant for this case rather than the bandwidth. Only one high-pass filter is needed for the simulation and its Fortran name is HPF.

A complete derivation of the filter subroutines is given in Appendix A.

IV. Evaluation of System Performance

Test Conditions

Many computer runs of the noise leveling AGC system have been done to evaluate the performance of the overall system. In most of these runs the parameters are the same. The RF bandwidth of the wideband prefilter is 40 MHz and the bandwidth of the low-pass signal filter after the first detector is 10 MHz. The high-pass filter which precedes the noise detector is given a cutoff frequency of 1 MHz, therefore the noise that reaches the noise detector is within the frequency range from 1 to 10 MHz. The bandwidth of the low-pass AGC filter is 100 Hz for cases when the exact system response is critical. The run-times tend to be very long with a 100 Hz bandwidth, so for the initial trial-and-error simulations the bandwidth was widened to 10 KHz to provide a faster response time. The sampling time in all simulations is 2 ns and the period of time simulated varies from 100 μs to 700 μs because the transient time for different signals varies. The computer plots shown in this report display a 100 μs window of the system response after it has reached steady state.

The gain distribution of the system is a variable of each simulation. The amount of gain and its distribution in the system greatly affects the performance of the noise leveling system. This is because the various stages contributing gain are isolated by nonlinear elements. The wideband RF amplifier is used primarily as a gain control element and typically operates with close to unity gain. This operating point is established in the simulation by appropriate selection of the parameter V_{nom}. The value for V_{nom} is determined by observing the open loop response of the system to an input noise level of -40 dBm. The steady state AGC voltage is the value assigned to V_{nom}. An input noise level of -40 dBm is used because it is considered to be typical of levels supplied to noise leveling systems by modern low-noise microwave receivers. This operating point sets the noise at the low end of the dynamic range of available square-law detectors.

Test Cases

The noise leveling AGC system was simulated using several different test cases with different signals and signal-to-noise ratios. One test case involves only noise at several different input levels. The purpose of this test is to examine the leveling error and test the transient response of the system. The expected response of the system is for the gain to adjust to each input noise level so that the output noise appears the same for each case. Another test is noise plus a pulse input at various levels. In this case it is expected that the pulse will come and go before the system responds and compresses the pulse. This would allow the pulse to be detected without the gain of the system changing and thereby modifying the false alarm rate of the system. A third test is a CW signal plus noise for the input. For this case, it is expected that the noise will remain the same and will have an increased mean depending on the CW input level.

Results

The two variations of the system, with or without a Log-Video-Amplifier (LVA), were studied to determine satisfactory gain distributions for acceptable performance. A major difference between the two systems is the dynamic range of the voltage levels throughout the systems. The system with the LVA compresses the voltage variations because of the logarithmic function. This difference means that the operating levels of the two versions are significantly different and require individual analysis to determine a satisfactory gain distribution for each system. The system with the LVA was considered first.

The parameters that need to be determined are the gain constants, K_d, K_n and K_a, and the nominal voltage of the variable gain amplifier, V_{nom}. The latter is dependent on the first two gain constants. To find V_{nom}, the open loop response of the system is simulated by letting K_a=0 and looking at the AGC voltage for a test case of -40 dBm noise. The gain constants K_d and K_n are taken from the previous study which dealt with an actual system. The AGC voltage with the open loop test will level-off after approximately 50 μs when a 10 KHz AGC bandwdith is used. This final value of the AGC voltage in Figure 6 is used for V_{nom}. In Figure 6 the top plot is the output of the noise leveling system and the bottom plot is the AGC voltage.

At this point, the slope at the gain characteristic must be determined for satisfactory performance. This is found by observing the closed loop system response with a 10 dB step reduction in noise at the midpoint of the simulation. Using the 10 kHz bandwidth for the AGC filter allows a fast enough response time that the system output should return to the same level as before the noise reduction. Figure 7a shows an unsatisfactory response to a reduction in noise when K_a = 1/2. After several trials a satisfactory response was obtained when K_a = 2 as shown in Figure 7b.

The gain distribution used in Figure 7b where, K_a = 2, K_d = 19, and K_n = 100, with V_{nom} = 2.0, is used in the remaining tests of the system

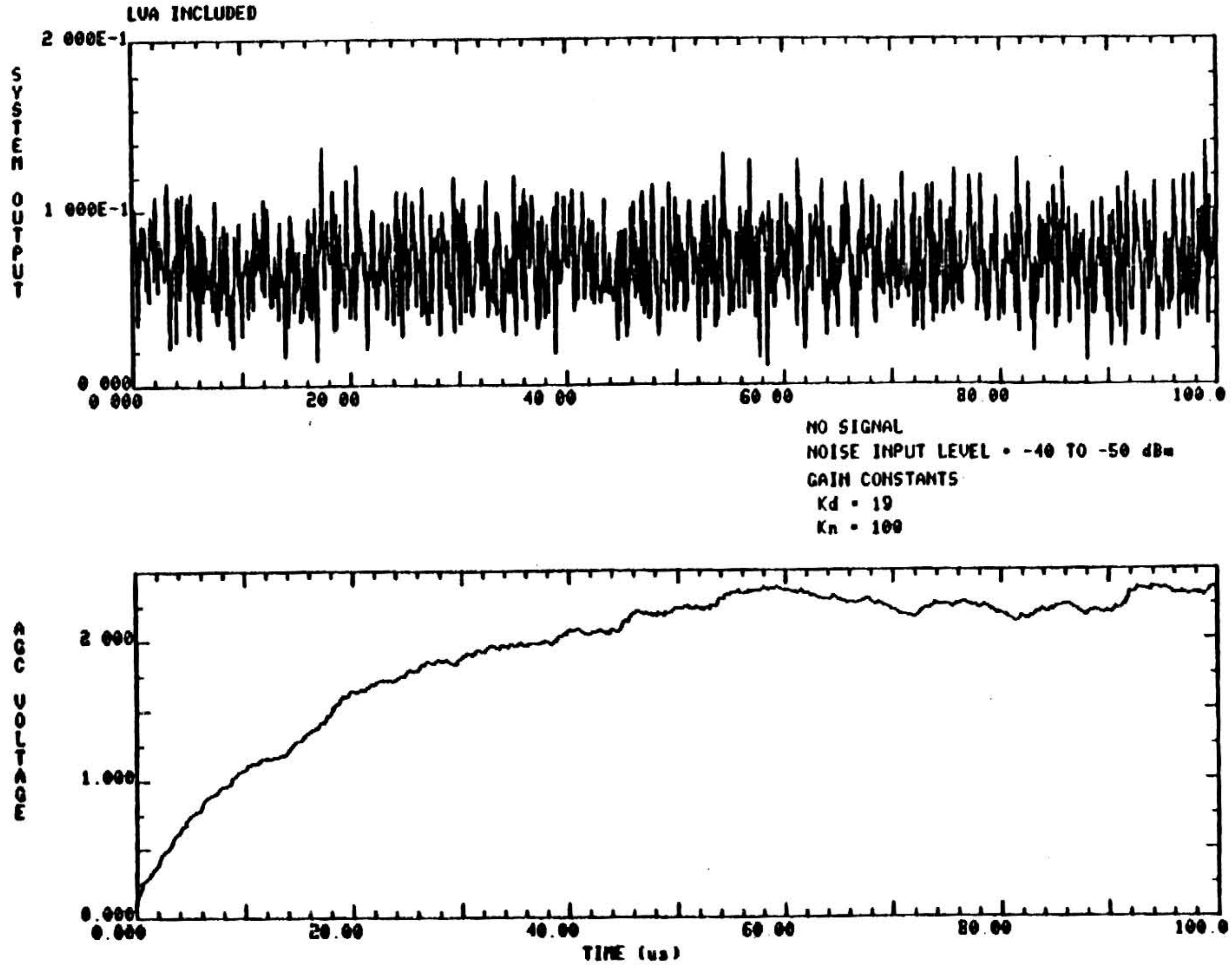


Figure 6. Determining nominal AGC voltage, V_{nom}, for system with a LVA.

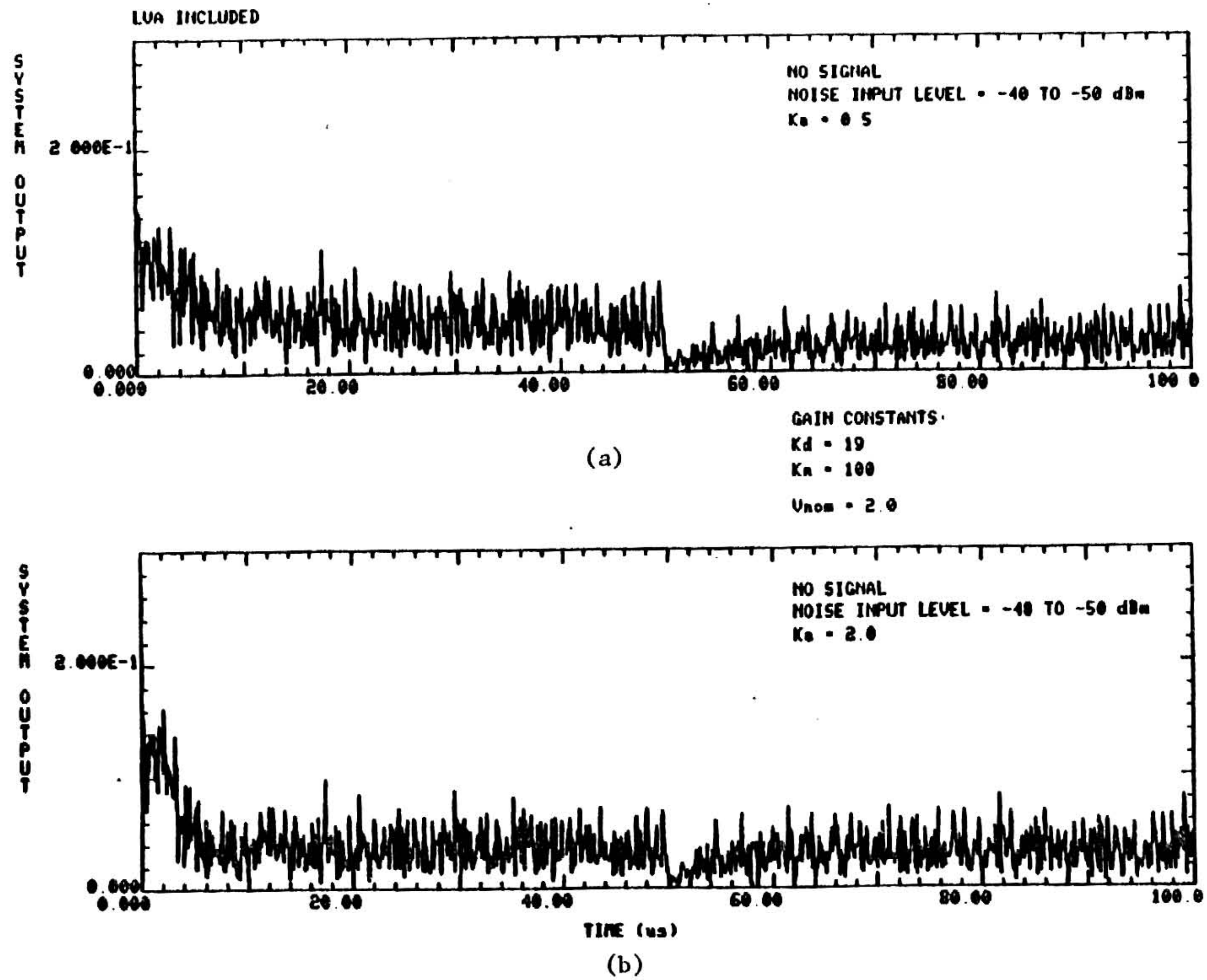


Figure 7. Determining slope of gain characteristic: (a) poor K_a; (b) good K_a.

with the LVA. The AGC filter bandwidth for the remaining tests is 100 Hz because the the exact system response is desired. The slower response time with the narrower bandwidth prevents displaying the adjusting response to changes in the input signal within a 100 μs segment of time. For this reason the system response to each test signal is displayed in a 100 μs window and all of these are compared to evaluate the system. Also, because of the slower response of the AGC filter the length of each simulation is approximately 800 μs, with the last 100 μs displayed, so that the start-up transient has died before the output is viewed.

The noise leveling system responds as expected and desired for inputs of noise only and a pulse plus noise. Figures 8a and 8b show the response of the system to noise only inputs at levels of -40 dBm and -50 dBm. By comparing these two plots it is easy to see that the noise is kept constant at the system output and consequently the false alarm rate remains constant. The system also responds as expected to a pulse input of -20 dBm as shown in Figure 8c. The pulse passes through the system before the system responds to it, therefore allowing the pulse to be detected by the detection circuits.

The AGC System with the LVA is next tested with a CW input signal. A previous study of the system [2] showed that there are problems with stability when the input is a strong CW signal. For a weak CW signal of -40 dBm with -40 dBm noise the system is stable and responses satisfactorily. The plots of -40 dBm noise with and without a -40 dBM CW signal are shown in Figure 9 where it can be seen that the noise level is constant for both cases. When the system is subjected to a strong CW signal of -20 dBm the response is on the verge of instability.

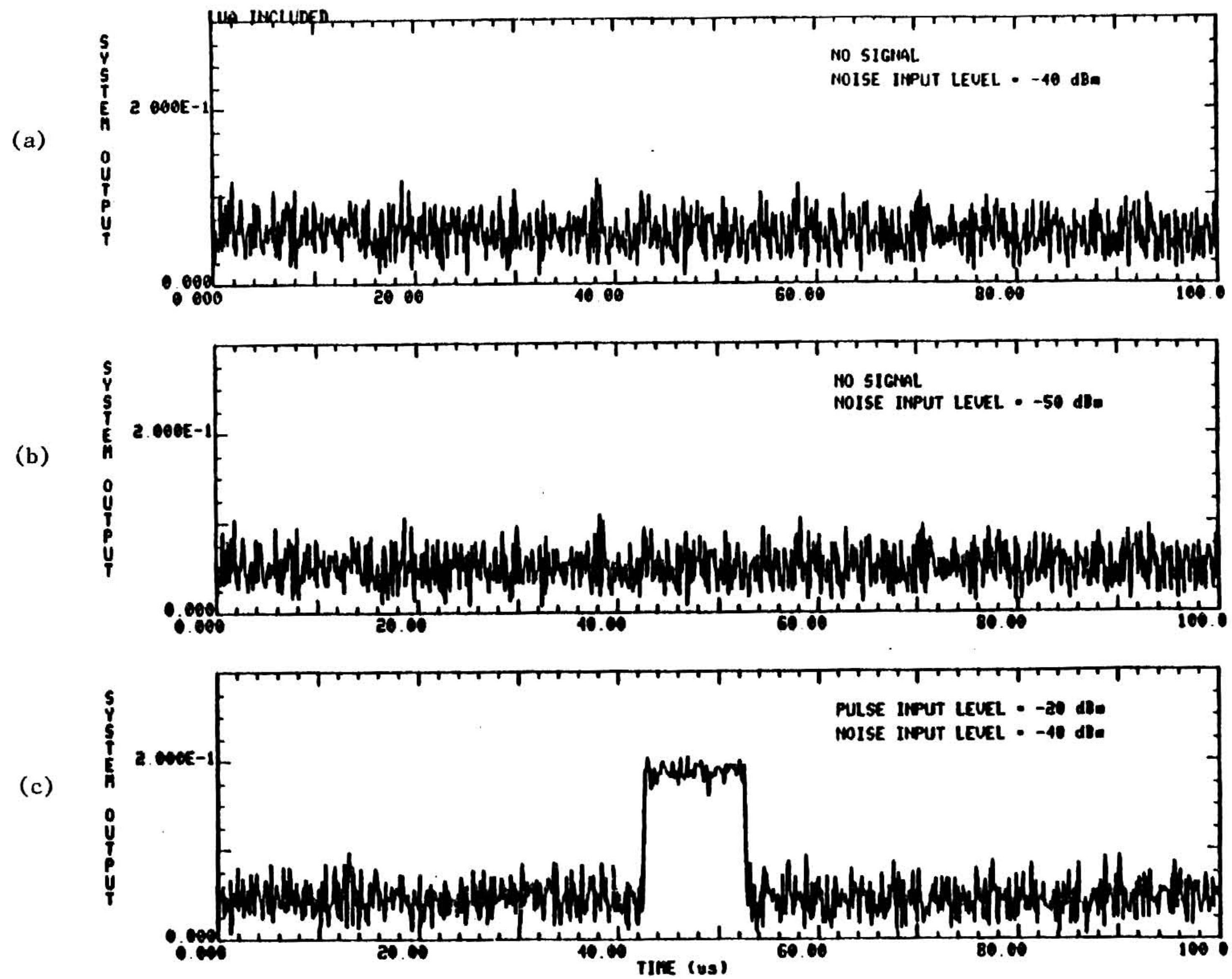
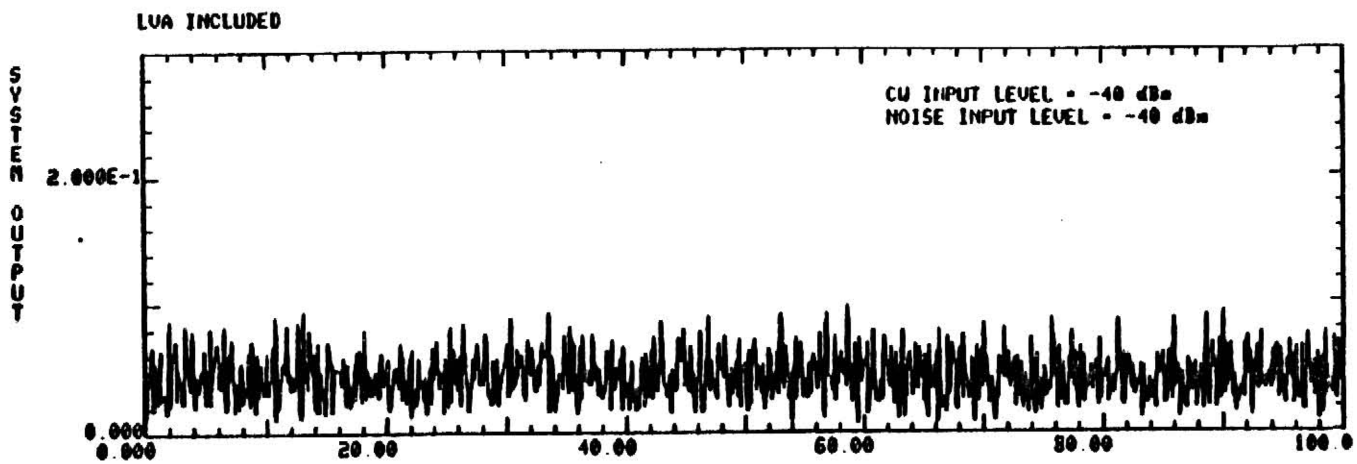
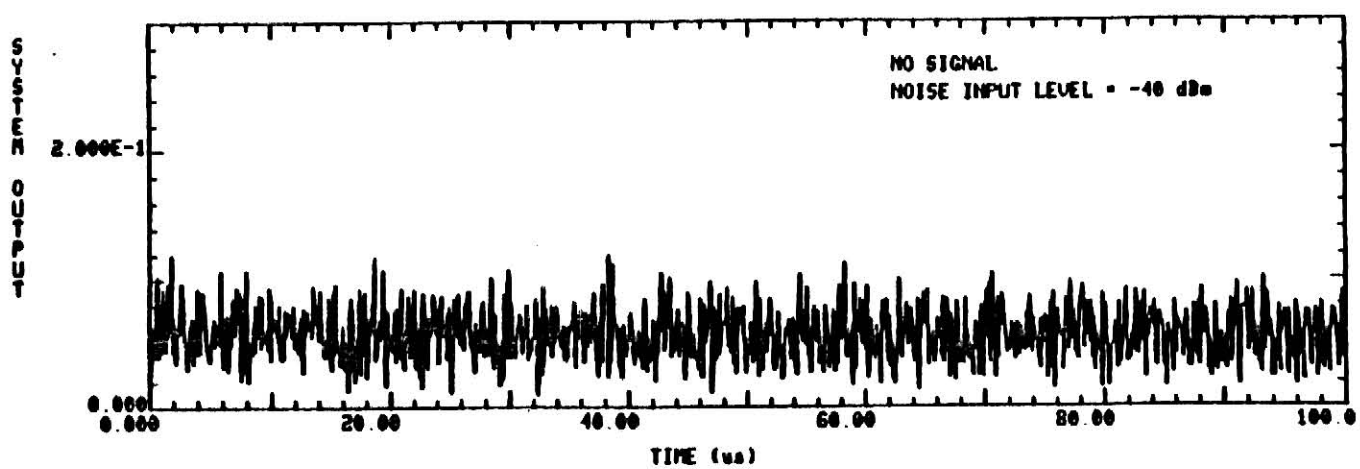


Figure 8. System Response to (a) -40 dBm noise; (b) -50 dBm noise and (c) pulse plus -40 dBm noise.



(a)



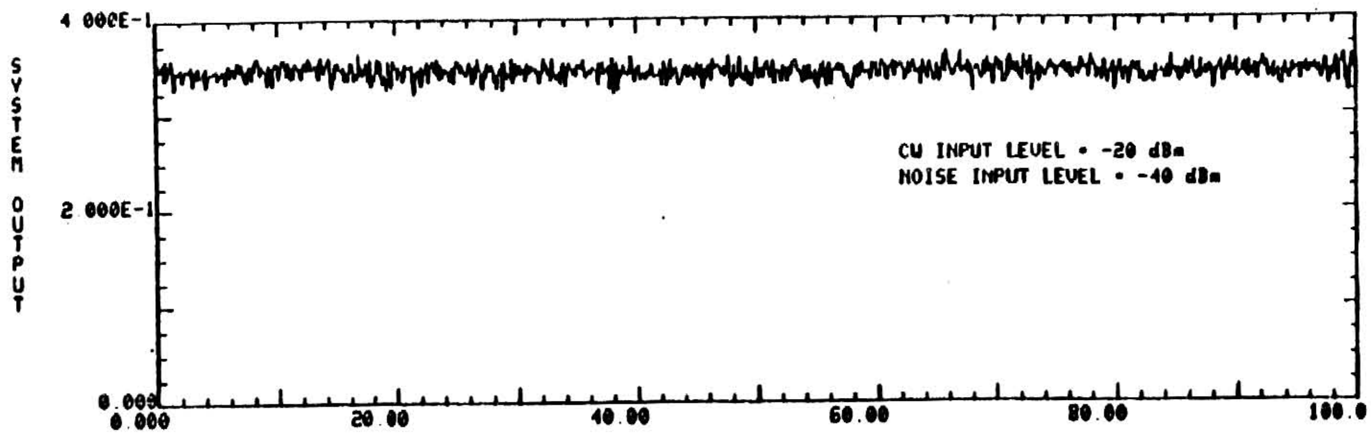
(b)

Figure 9. (a) Good response to weak CW signal plus noise; (b) Response to noise only.

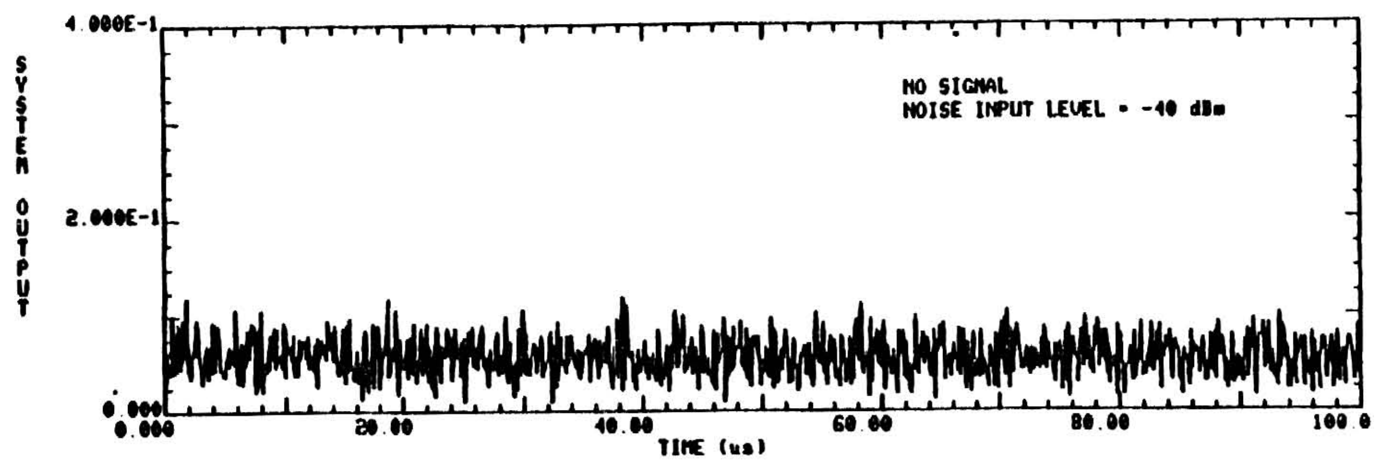
The undesirable responses to a strong CW input with -40 dBm noise is shown in Figure 10a and the response with no signal is shown as a reference in Figure 10b. The noise level is the same for both cases however with the strong CW signal the noise is compressed which could cause instability.

The potential instability of the system can be attributed to the logarithmic function of the LVA. The instability can be explained quantitatively by examining a plot of the log function shown in Figure 11. Recall, that the purpose of the AGC system is to keep to noise-level constant by using the signal at the noise detector to form the AGC voltage. When the noise power at this point increases the AGC voltage causes a decrease in the gain and vice versa. Looking at Figure 11, let ΔX, be the rms noise variation with a weak CW signal. The resulting noise variation at the output of the LVA is ΔY_1. The variation is amplified at this operating point. However, if the same input noise variation is with a strong CW signal the operating point moves to the right on the logarithmic curve. At the higher operating point the output noise variation of the LVA can be described by ΔY_2. This is obviously less than ΔY_1 although the noise at the input was equivalent. In the second case, the AGC voltage will cause an increase in gain of the wideband amplifier. This will increase the noise, but it will also increase the CW level, consequently moving the operating point farther to the right. This results in a positive feedback condition or instability.

A noise leveling AGC system that does not include a Log Video Amplifier was also evaluated. All of the gain constants for this version of the noise leveling system must be determined because there is



(a)



(b)

Figure 10. (a) Marginally stable response to strong CW signal plus noise; (b) Response to noise only.

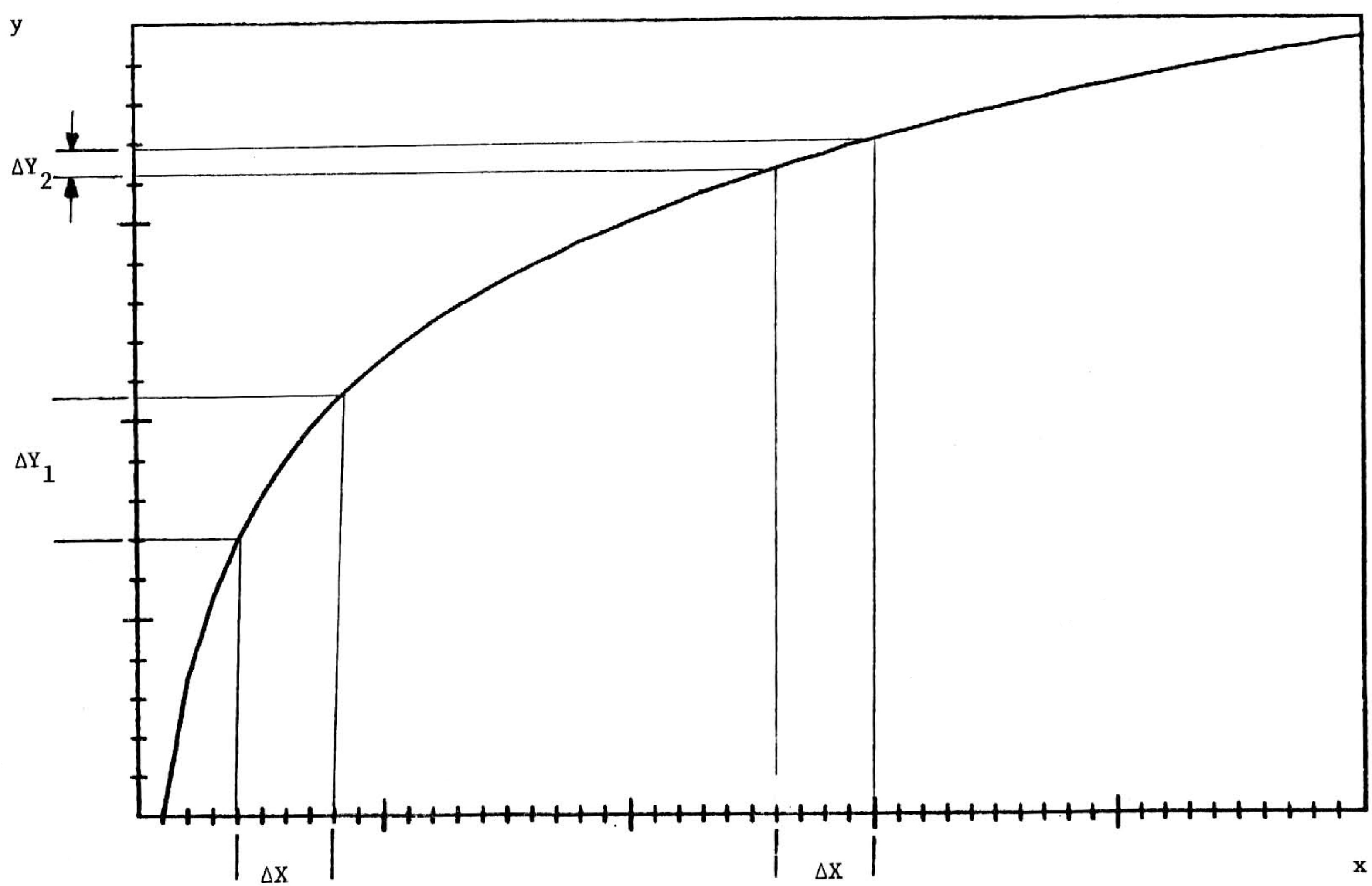


Figure 11. Logarithmic Function: $y = \log(x)$

no background information concerning this system. A desire to have the output voltage and the AGC voltage at the same order of magnitude as the system with the LVA was the criteria used to select the gain constants, K_d and K_n. The output and the AGC voltages for the open loop response of the system were observed for several combinations of K_d and K_n until the voltages were in the desired range. The open-loop system response with the selected gain constants, $K_d = 90{,}000$ and $K_n = 80$, is shown in Figure 12 where the top plot is the output voltage and the bottom plot is the AGC voltage. The nominal voltage, V_{nom} then, is the steady-state value of the AGC voltage in Figure 12. The value for the gain constant, K_a, is found by observing the closed-loop system response to a 10 dB step reduction in noise. A satisfactory response is when the output of the system after the noise reduction returns to the same noise level as before the change. Figure 13a shows a poor response to a step reduction in the noise with $K_a = 1$, and a satisfactory response to the noise reduction with $K_a = 2$ is shown in Figure 13b.

The gain distribution, $K_a = 2$, $K_d = 90{,}000$, and $K_n = 80$ with $V_{nom} = 2.2$, is used in the remaining tests of the noise leveling AGC system. The bandwidth of the AGC filter is set to 100 Hz so the response time is longer and therefore different inputs are displayed individually in a 100 μs window. The system response with the 100 Hz AGC filter to an input of -40 dBm noise only is shown in Figure 14a and the response to -50 dBm noise is shown in Figure 14b. Comparing these two responses verifies that the noise leveling system is working as expected. The system response to a trapezoidal pulse with a width of 10 μs is shown in Figure 14c. It was hoped that the system could not respond fast enough to compress the pulse although it does for this case. This is because

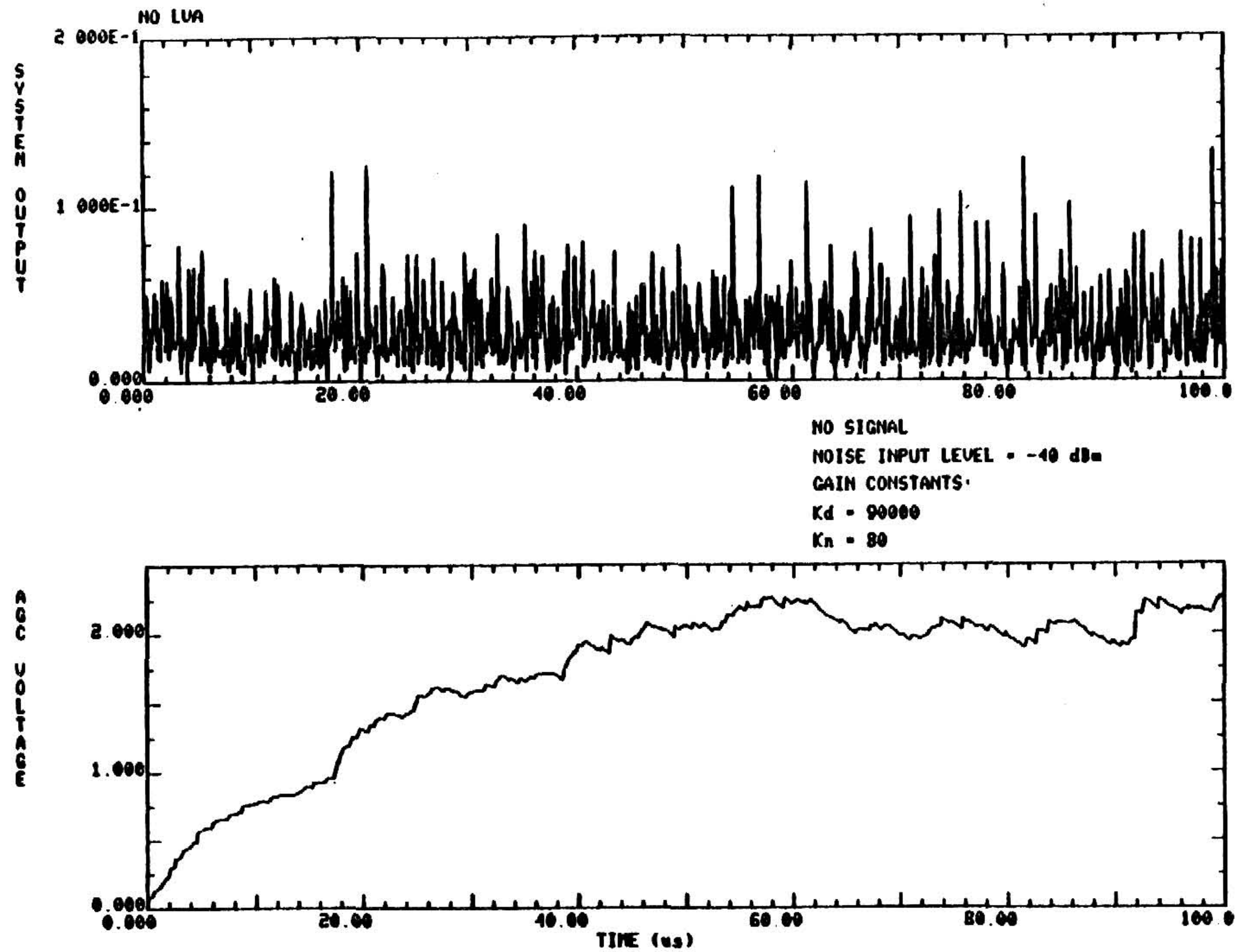


Figure 12. Determining nominal AGC voltage, V_{nom}, for system without an LVA.

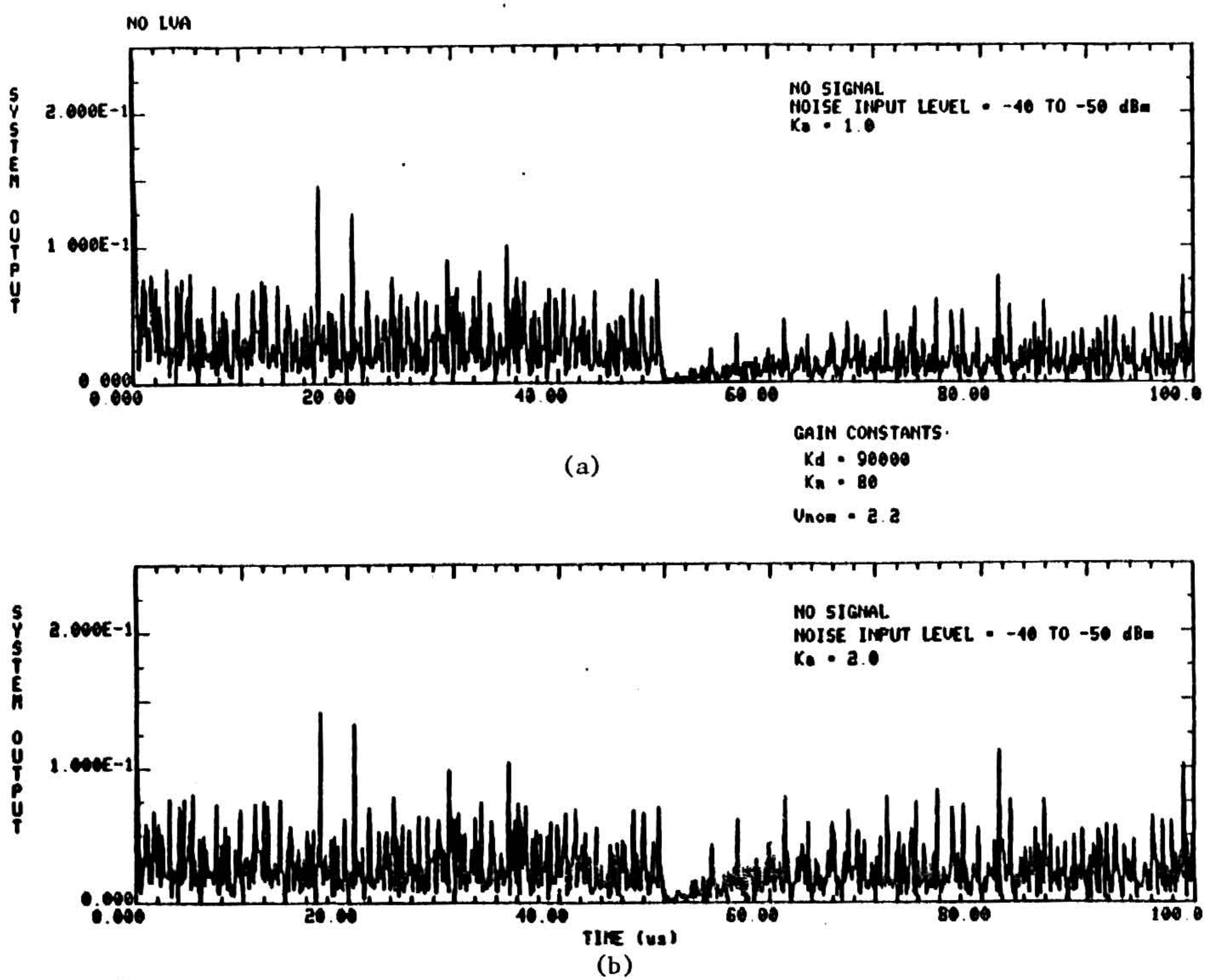


Figure 13. Determining slope of gain characteristic (a) poor K_a; (b) good K_a.

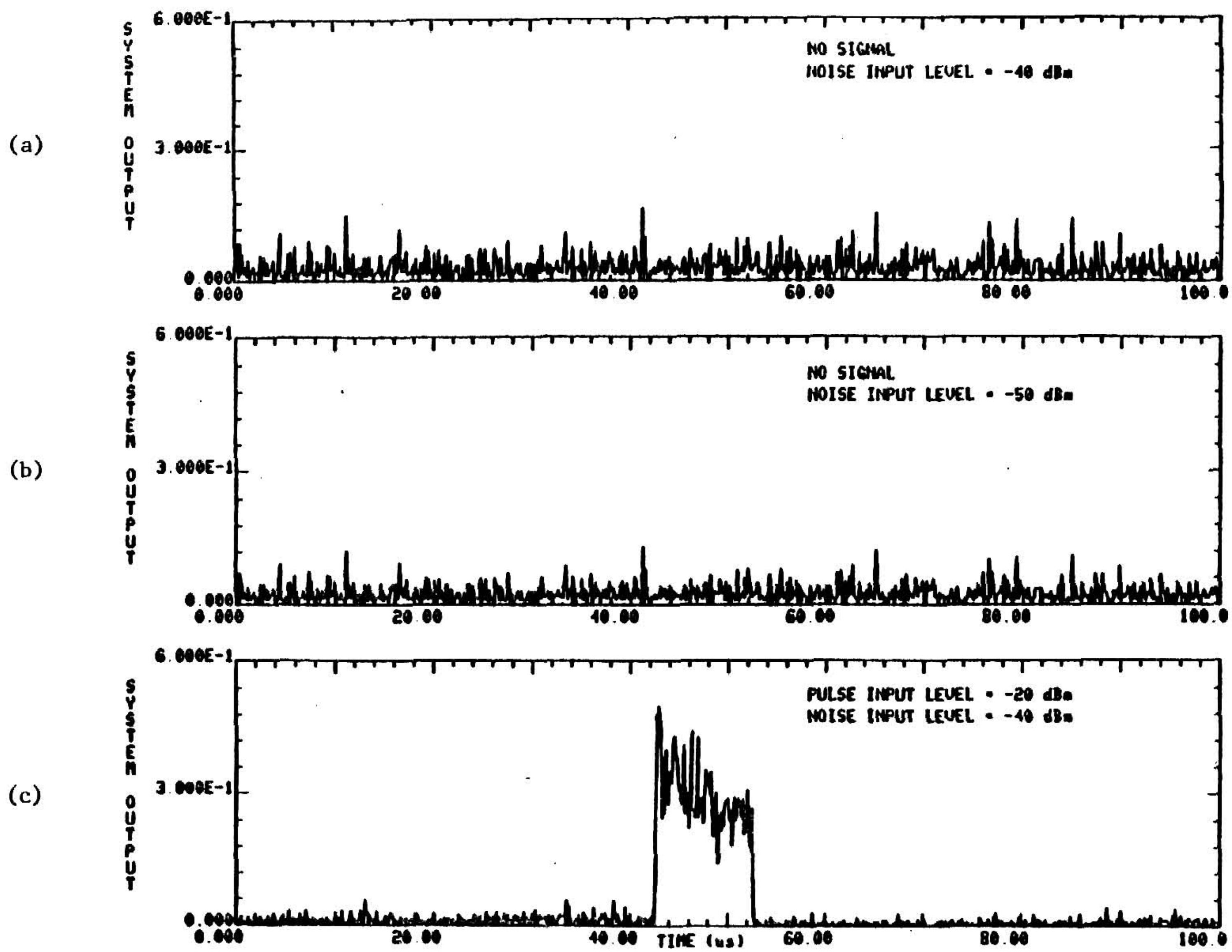


Figure 14. System response to (a) -40 dBm noise; (b) -50 dBm noise; (c) pulse plus noise.

the larger gain constants of this system, as compared to the system with the LVA, increase the bandwidth of the whole system resulting in a quicker response.

The response of the AGC system without the LVA, to a CW input is a very important criteria in comparing the two variations of the noise leveling system. The system without the LVA is stable for a CW input at -20 dBm where the system with the LVA was unstable. The response of the system without the LVA to CW signal levels of -20 dBm and -40 dBm with an input noise level of -40 dBm are shown in Figure 15. Both cases are stable, but most importantly, the response to the -20 dBm CW signal is stable and has the same noise level as the response to the weaker input level.

Summary of Results

Two variations of the noise leveling AGC system were compared based on their responses to several different test cases. Both systems, one with a Log-Video Amplifier (LVA) and the other without, maintain a constant noise level when there is no signal input. The system with the LVA perforams well with a pulse input however it responds undesirably when the input is a CW signal. The system without the LVA responds as expected for a CW input signal, however its response to a pulse input is undesirable.

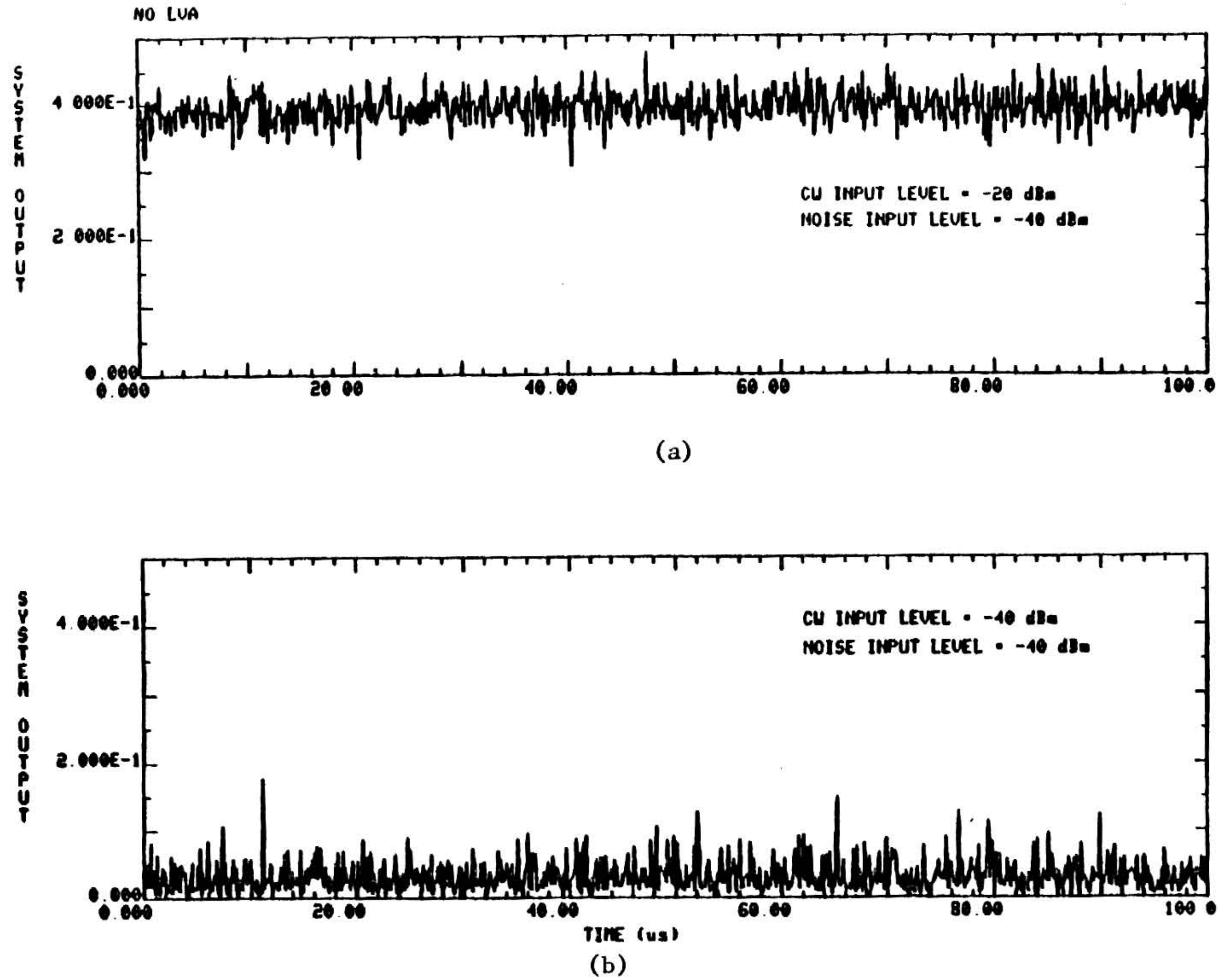


Figure 15. Stable response to (a) strong CW signal plus noise; (b) weak CW signal plus noise.

V. Conclusions

Of the two noise leveling AGC systems that were evaluated in this study, neither performed ideally. The system with the LVA was marginally stable in the presence of a strong CW input signal which agrees with the findings of a previous study [2]. For the system without the LVA the input pulse was compressed by the AGC system because the system bandwidth was wider due to the large gain constants. This could possibly be corrected by deactivating the AGC whenever the signal is greater than the detection threshold and holding the gain constant until the signal goes below the threshold again. Other than the undesired response to the pulse the system with an LVA performs satisfactorily.

APPENDIX A: Development of Filter Subprograms

To perform the computer simulation of the AGC system several digital filters are needed. The filtering must be done in the time domain rather than the frequency domain due to the feedback loop. This is easily done in Fortran by using difference equations to model discrete time filters. This appendix details the development of discrete time filter models from continuous transfer functions. A general Fortran program is needed for the low-pass and the high-pass cases which can be one through four pole filters.

The low-pass case is considered first, followed by the high pass case. The development of the program started with the Laplace-domain transfer functions of Butterworth low-pass filters with bandwidths of 1 rad/s as given in Table A-1.

Table A-1 Butterworth Low-Pass Transfer Functions

n	H(s)
1	$\frac{1}{s + 1}$
2	$\frac{1}{s^2 + \sqrt{2}\, s + 1}$
3	$\frac{1}{s^3 + 2s^2 + 2s + 1}$
4	$\frac{1}{s^4 + 2.6131s^3 + 3.4142s^2 + 2.6131s + 1}$

A discrete time equivalent model for a simple one-pole filter is derived first. The transfer function of a one pole low-pass Butterworth filter with a cutoff freuqncy of ω_3 rad/sec can be obtained from Table

A-1 by replacing s by $\frac{s}{\omega_3}$. The resulting equation is

$$\frac{Y(S)}{X(S)} = \frac{\omega_3}{s + \omega_3} \qquad \text{(A-1)}$$

This can be rearranged to fit the form of a closed loop transfer function from linear control systems analysis,

$$\frac{Y(s)}{X(s)} = \frac{\frac{\omega_3}{s}}{1 + \frac{\omega_3}{s}} \ . \qquad \text{(A-2)}$$

By comparing equation A-2 to the closed-loop form,

$$\frac{Y(s)}{X(s)} = \frac{G(s)}{1 + G(s)\ H(s)} \qquad \text{(A-3)}$$

these relations are observed. The open-loop transfer function $G(s) = \frac{\omega_3}{s}$ and the feedback function H(s) is unity.

The block diagram for this filter can be drawn as in Figure A-1.

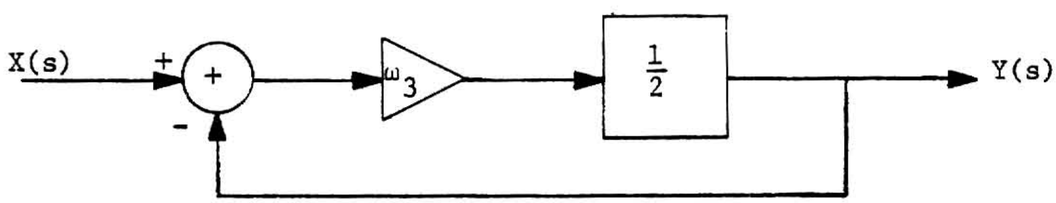


Figure A-1. Block diagram of one-pole filter

The factor, $\frac{1}{s}$, in the Laplace domain represents an integration in the time domain. Therefore, the model shown in Figure A-1 can be redrawn for the time domain by replacing the factor $\frac{1}{s}$ by an integrator. See Figure A-2.

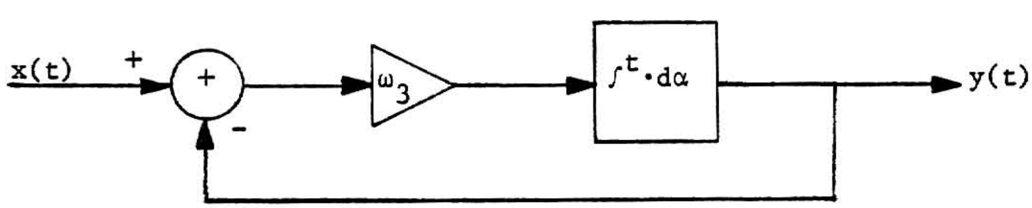


Figure A-2. Time domain model for low-pass filter.

To obtain a discrete time model which is usable for a computer simulation, the input is sampled every T_s seconds. The integrator is approximated using the rectangular rule, which is

$$y(k) = y(k-1) + T_s\, z(k) \tag{A-4}$$

where $z(k)$ is the input at the k^{th} sampling time and $y(k-1)$ is the previous value of the integrator. Using the approximation for the integrator the resulting difference equation is

$$y(k) = (1 - T_s\, \omega_3)\, y(k-1) + T_s\, \omega_3\, x(k). \tag{A-5}$$

Using a method similar to the previous one a discrete time model for a n-pole transfer function can be found. The transfer function for a filter having n-poles is

$$\frac{Y(s)}{X(s)} = \frac{K}{s^n + a_{n-1}\, s^{n-1} + \ldots\, a_1 s + a_0} \tag{A-6}$$

Cross-multiplying and solving for the highest power yields

$$Y(s)s^n = KX(s) - a_{n-1}\, s^{n-1}\, Y(s) - a_{n-2}s^{n-2}Y(s) - \ldots - a_1 sY(s) - a_0 Y(s). \tag{A-7}$$

This equation can be represented by a block diagram having multiple loops as shown in Figure A-3.

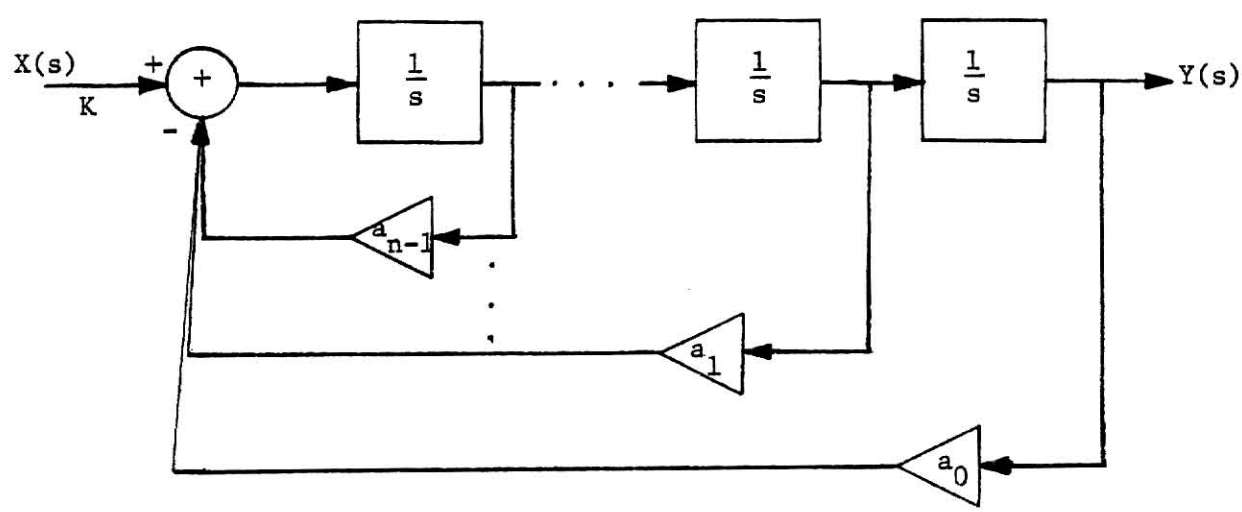


Figure A-3. N-Pole Block Diagram

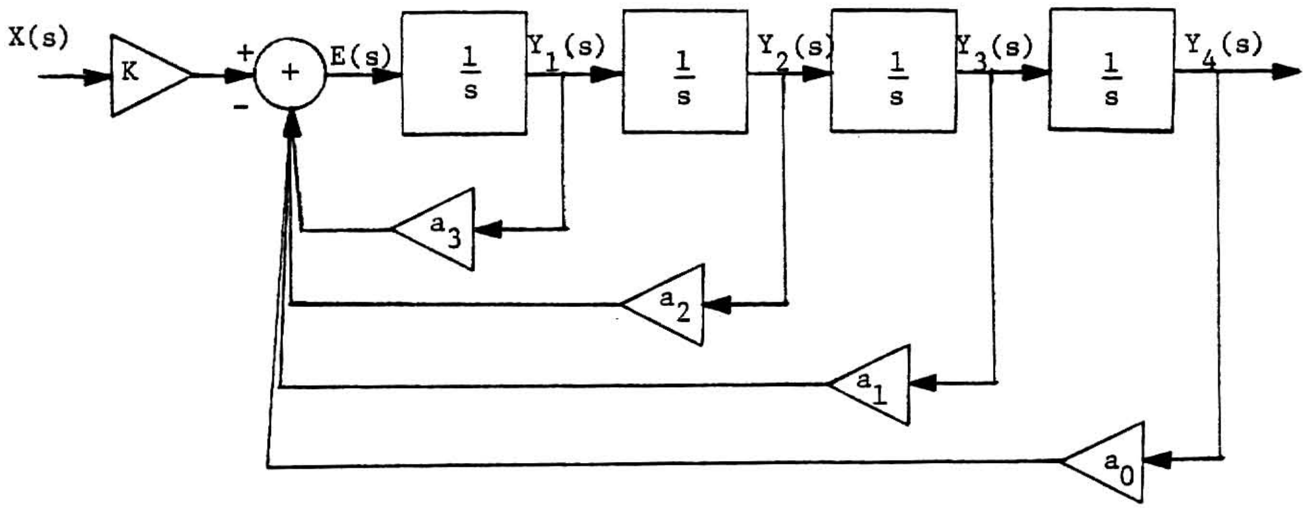


Figure A-4. 4-Pole Block Diagram

Figure A-4 shows the block diagram for a 4-pole transfer function. This block diagram can be used to develop the difference equations for the one through four pole cases that are needed. Referring to Figure A-4, the output of a n-pole filter is $Y_n(s)$ with the appropriate coefficients equal to zero. The transfer function and coefficients can now be determined for a filter with a cutoff frequency of ω_3 rad/sec.

For clarity each case is considered separately, beginning with the four pole case. In this case the output is $Y_4(s)$, so a closed form function is derived for $\frac{Y_4(s)}{X(s)}$. From Figure A-4 it is seen that,

$$E(s) = KX - a_3 s^3\, Y_4(s) - a_2 s^2\, Y_4(s) - a_1\, sY_4(s) - a_0\, Y_4(s) \qquad \text{(A-8)}$$

and $E(s) = s^4\, Y_4(s)$.

Equating the first and the second yields

$$s^4\, Y_4(s) = KX - a_3 s^3\, Y_4(s) - a_2 s^2\, Y_4(s) - a_1 s\, Y_4(s) - a_0\, Y_4(s). \qquad \text{(A-9)}$$

Separating variables and solving for $\frac{Y_4(s)}{X(s)}$ gives

$$\frac{Y_4(s)}{X(s)} = \frac{K}{s^4 + a_3s^3 + a_2s^2 + a_1s + a_0} \quad . \tag{A-10}$$

The Butterworth 4-pole transfer function given in Table A-1 is for a cutoff frequency of 1 rad/sec. It can be frequency scaled to obtain a cutoff frequency of ω_3 rad/sec by replacing s with $\frac{s}{\omega_3}$. The resulting equation is

$$H(s) = \frac{\omega_3^4}{s^4 + 2.6131\omega_3s^3 + 3.4142\omega_3^2s^2 + 2.6131\omega_3^3s + \omega_3^4} \quad . \tag{A-11}$$

Comparing this to Equation A-10 gives the following relations for the four-pole case:

$$\begin{aligned} K &= \omega_3^4 \\ a_3 &= 2.6131\omega_3 \\ a_2 &= 3.4142\omega_3^2 \\ a_1 &= 2.6131\omega_3^3 \\ a_0 &= \omega_3^4 \quad . \end{aligned} \tag{A-12}$$

The same steps are taken to find the coefficients for the three pole filter transfer function with a cutoff frequency of ω_3 rad/sec. For this case $Y_3(s)$ is the output and the coefficient from the outermost loop is set equal to zero.

From Figure A-4 the following relations can be written:

$$E(s) = KX - a_3s^2\ Y_3(s) = a_2\ sY_3(s) - a_1Y_3(s) \tag{A-13}$$

and

$$E(s) = s^3\ Y_3(s). \tag{A-14}$$

From the last two equations the transfer function is

$$\frac{Y_3(s)}{X(s)} = \frac{K}{s^3 + a_3s^2 + a_2s + a_1} \quad . \tag{A-15}$$

The Butterworth transfer function from Table A-1 is frequency scaled to get

$$H(s) = \frac{\omega_3^3}{s^3 + 2\omega_3 s^2 + 2\omega_3^2 s + \omega_3^3} \,. \tag{A-16}$$

Comparing the previous two equations gives the following relations for the three-pole case:

$$\begin{aligned} K &= \omega_3^3 \\ a_3 &= 2\omega_3 \\ a_2 &= 2\omega_3^2 \\ a_1 &= \omega_3^3 \\ a_0 &= 0 \quad . \end{aligned} \tag{A-17}$$

Referring to Figure A-4, the output for a two-pole filter is $Y_2(s)$ with the coefficients of the two outermost feedback loops equal to zero. These two relations are easily written:

$$E(s) = KX - a_3 s\, Y_2(s) - a_2 Y_2(s) \tag{A-18}$$

and

$$E(s) = s^2\, Y_2(s). \tag{A-19}$$

Equating these two and solving for $\frac{Y_2(s)}{X(s)}$ gives

$$\frac{Y_2(s)}{X(s)} = \frac{K}{s^2 + a_3 s + a_2} \,. \tag{A-20}$$

Again the Butterworth filter transfer function from Table A-1 is frequency scaled to get

$$H(S) = \frac{\omega_3^2}{S^2 + \sqrt{2}\,\omega_3 s + \omega_3^2} \,.$$

Comparing these equations gives the coefficients for the two-pole case:

$$K = \omega_3^2$$
$$a_3 = \sqrt{2}\omega_3$$
$$a_2 = \omega_3^2 \qquad \text{(A-21)}$$
$$a_1 = a_0 = 0$$

The one-pole case has only one non-zero coefficient in the feedback loops. With $Y_1(s)$ as the output of concern the following equations are written,

$$E(s) = KX - a_3\, Y_1(s) \qquad \text{(A-22)}$$

$$E(s) = s\, Y_1(s). \qquad \text{(A-23)}$$

The transfer function is easily solved as,

$$\frac{Y_1(s)}{X(s)} = \frac{K}{s + a_3}\,. \qquad \text{(A-24)}$$

This is compared to equation A-1 which is the Butterworth transfer function for a one pole filter with a cut-off frequency of ω_3 rad/sec. to get these relations;

$$K = \omega_3$$
$$a_3 = \omega_3 \qquad \text{(A-25)}$$
$$a_2 = a_1 = a_0 = 0.$$

The next step is to develop the difference equations using the coefficients just derived. A time domain model is acquired from Figure A-4 by replacing the factors $\frac{1}{s}$ by integrations with respect to time. See Figure A-5. To get a discrete model a numerical method for the integrations is needed. A first order approximation using the rectangular rule is easy to implement. The rectangular rule for $y(t) = \int^t x(t)\, dt$ is

$$y(k) = y(k-1) + T_s x(k) \qquad \text{(A-26)}$$

for a sampling time of T_s, using k as the discrete time index.

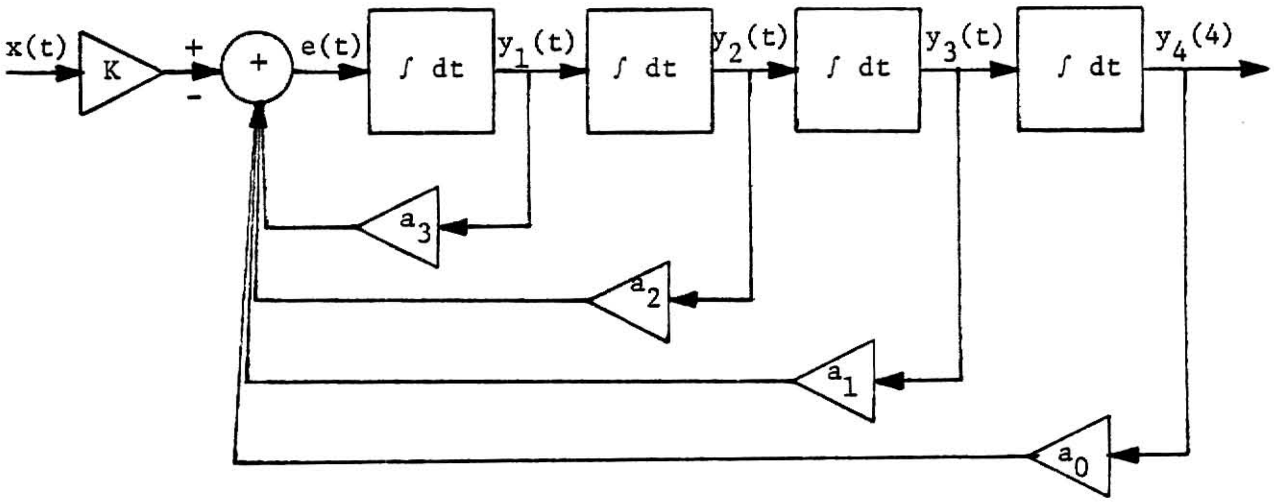


Figure A-5 Time Domain Model for Low Pass Filters

The difference equations to describe a low-pass Butterworth filter can be written by using the rectangular rule for integration and referring to Figure A-5:

$$\begin{aligned} e(k) &= Kx(k) - a_3\, y_1(k-1) - a_2 y_2(k-1) - a_1 y_3(k-1) - a_0 y_4(k-1) \\ y_4(k) &= y_4(k-1) + T_s\, y_3(k) \\ y_3(k) &= y_3(k-1) + T_s\, y_2(k) \\ y_2(k) &= y_2(k-1) + T_s\, y_1(k) \\ y_1(k) &= y_1(k-1) + T_s\, e(k). \end{aligned} \tag{A-27}$$

The coefficients, a_i, as just derived for the four cases, are located in a look-up table at the start of the Fortran subprogram. The bandwidth, ω_3, sampling time, T_s, numbers of poles, n, and the input x(k) are passed by the subroutine call. Also, a logical variable is passed indicating if the subroutine call is the first to initialize the $y_i(k)$ to zero. The output of the subroutine that is returned to the calling program is conditional on the number of poles in the filter. Since four low pass filters are needed in the main program, there are four independent subprograms call FILT1, FILT2, FILT3, and FILT4. Copies of these are in Appendix B.

The first part of this appendix has dealt solely with low-pass Butterworth filters. The derivation for a high-pass Butterworth filter subroutine is very similar. The Laplace-domain transfer functions for Butterworth high-pass filters with a cutoff frequency of ω_3 rad/sec are given in Table A-2.

Table A-2 Butterworth High-Pass Transfer Functions

n	H(s)
1	$\dfrac{1}{s + 1}$
2	$\dfrac{s^2}{s^2 + \sqrt{2}s + 1}$
3	$\dfrac{s^3}{s^3 + 2s^2 + 2s + 1}$
4	$\dfrac{s^4}{s^4 + 2.6131s^3 + 3.4142s^2 + 2.6131s + 1}$

A general form for an n-pole high pass function is

$$\frac{Y(s)}{X(s)} = \frac{s^n}{s^n + a_{n-1}s^{n-1} + \ldots + a_1 s + a_0} \tag{A-28}$$

Rearranging this equation to a usable form gives

$$Y(s) = X(s) - \frac{a_{n-1}\,Y(s)}{s} - \ldots - \frac{a_1\,Y(s)}{s^{n-1}} - \frac{a_0\,Y(s)}{s^n} \tag{A-29}$$

A block diagram which represents Equation A-9 is shown in Figure A-6 for n = 4 where Y(s) is the output.

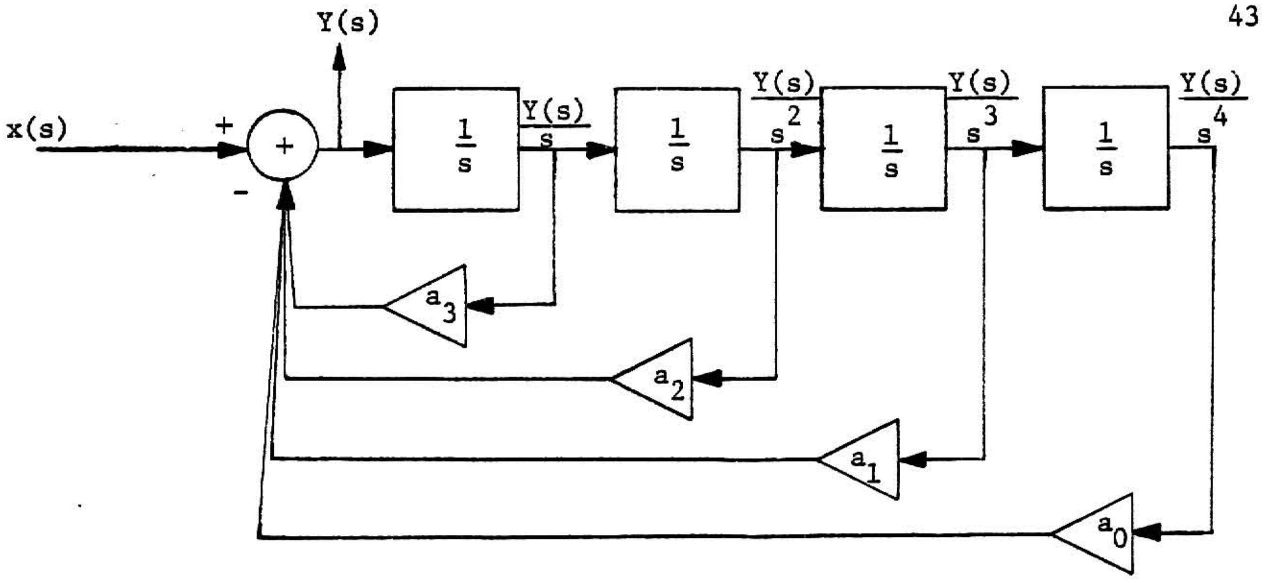


Figure A-6 High-Pass Filter Model

As was done in the low-pass case this block diagram is used to model a one, two, three or four pole filter by setting the appropriate coefficients equal to zero. The output for all high-pass cases is Y(S).

The coefficients to model Butterworth high pass filters with a frequency cutoff of ω_3 rad/sec. are derived next. Each case is considered individually as before, beginning with the four pole case.

For the four pole case all of the coefficients are non-zero. The describing equation is

$$Y(s) = X(s) - a_3 \frac{Y(s)}{s} - a_2 \frac{Y(s)}{s^2} - a_1 \frac{Y(s)}{s^3} - a_0 \frac{Y(s)}{s^4} . \qquad \text{(A-30)}$$

This is easily solved for the transfer function

$$\frac{Y(s)}{X(s)} = \frac{s^4}{s^4 + a_3 s^3 + a_2 s^2 + a_1 s + a_0} . \qquad \text{(A-31)}$$

The Butterworth filter transfer function from Table A-2 is frequency scaled by replacing s with s/ω_3, to get

$$H(s) = \frac{s^4}{s^4 + 2.6131\omega_3 s^3 + 3.4142\omega_3^2 s^2 + 2.6131\omega_3^3 s + \omega_3^4} . \qquad \text{(A-32)}$$

which has a cutoff frequency of ω_3 rad/sec. By comparing the two previous equations the coefficients for a four pole high pass Butterworth filter are:

$$\begin{aligned} a_3 &= 2.6131\ \omega_3 \\ a_2 &= 3.4142\ \omega_3^2 \\ a_1 &= 2.6131\ \omega_3^3 \\ a_0 &= \omega_3^4\ . \end{aligned} \tag{A-33}$$

The same steps are followed to derive the coefficients for the three-pole Butterworth filter. For this case $a_0 = 0$, therefore, the transfer function for Figure A-5 is

$$\frac{Y(s)}{X(s)} = \frac{s^3}{s^3 + a_3 s^2 + a_2 s + a_1}\ . \tag{A-34}$$

The frequency scaled Butterworth transfer function is

$$H(s) = \frac{s^3}{s^3 + 2\omega_3 s^2 + 2\omega_3^2 s + \omega_3^3}\ . \tag{A-35}$$

By comparison the relationships for the three-pole case are:

$$\begin{aligned} a_3 &= 2\omega_3 \\ a_2 &= 2\omega_3^2 \\ a_1 &= \omega_3^3 \\ a_0 &= 0\ . \end{aligned} \tag{A-36}$$

The two pole model in Figure A-5 has both a_1 and a_0 equal to zero. The transfer function is

$$\frac{Y(s)}{X(s)} = \frac{s^2}{s^2 + a_3 s + a_2}\ . \tag{A-37}$$

The Butterworth two-pole transfer function from Table A-2 is frequency scaled to obtain,

$$H(s) = \frac{s^2}{s^2 + \sqrt{2}\,\omega_3 s + \omega_3^2} . \tag{A-38}$$

By comparing the last two equations the coefficients for a two pole Butterworth filter are

$$\begin{aligned} a_3 &= \sqrt{2}\,\omega_3 \\ a_2 &= \omega_3^2 \\ a_1 &= a_0 = 0. \end{aligned} \tag{A-39}$$

The one-pole case has only one non-zero feedback coefficient, a_3. The transfer function is therefore,

$$\frac{Y(s)}{X(s)} = \frac{s}{s + a_3} \tag{A-40}$$

From Table A-2, the two-pole Butterworth filter transfer function can be frequency scaled to get,

$$H(s) = \frac{s}{s + \omega_3} . \tag{A-41}$$

By comparing this equation to the former one the relations for the one-pole Butterworth filter transfer function are

$$\begin{aligned} a_3 &= \omega_3 \\ a_2 &= a_1 = a_0 = 0. \end{aligned} \tag{A-42}$$

With the coefficients for each case known, the next step is to develop the difference equations. Figure A-6 is redrawn in the time domain, replacing the factors $\frac{1}{s}$ with integrations with respect to time. See Figure A-7. A discrete time model is acquired by using the rectangular rule to approximate the integrations.

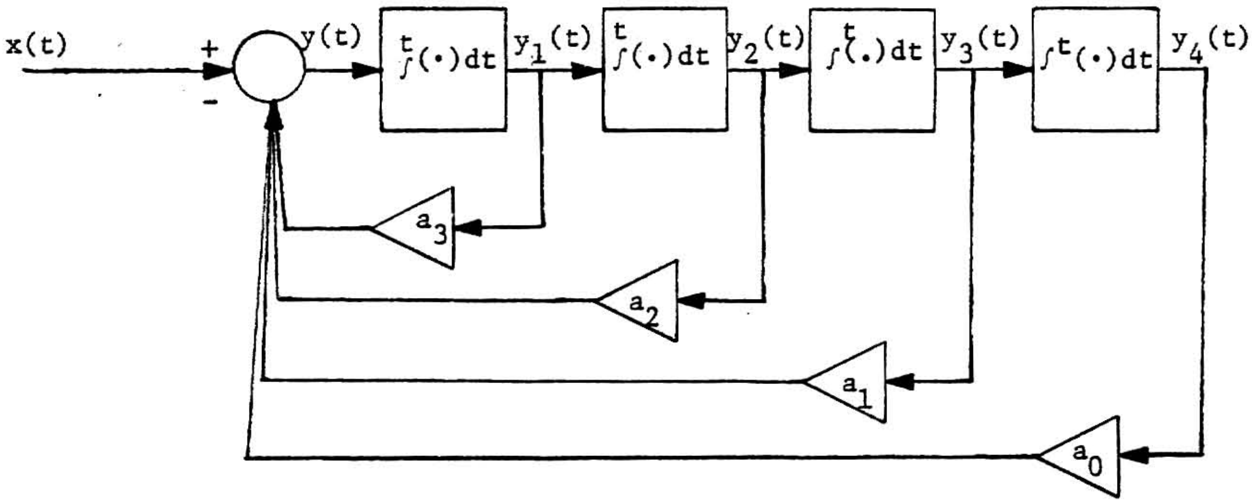


Figure A-7 Time Domain Model for High-Pass Filters

The difference equations are easily written using k as the time index and T_s as the sampling time.

$$
\begin{aligned}
y(k) &= x(k) - a_e y_1(k-1) - a_2 y_2(k-1) - a_1 y_3(k-1) - a_0 y_4(k-1) \\
y_4(k) &= y_4(k-1) + T_s y_3(k) \\
y_3(k) &= y_3(k-1) + T_s y_2(k) \qquad \text{(A-43)} \\
y_2(k) &= y_2(k-1) + T_s y_1(k) \\
y_1(k) &= y_1(k-1) + T_s y(k)
\end{aligned}
$$

The coefficients, a_i, are located in a look-up table at the beginning of the Fortran subprogram. The parameters, ω_3, T_s, n and the input x(k) are passed by the subroutine call. There is also a logical variable indicating the first call to the subprogram is y(k). A copy of the high-pass filter subprogram, HPF is in Appendix B.

Appendix B

COMPUTER PROGRAMS

```
C***********************************************************************
C
C       NOISE-LEVELING AGC
C
C       DG FORTRAN 5 SOURCE FILENAME:            NLAGC.FR
C
C       DEPARTMENT OF ELECTRICAL ENGINEERING     KANSAS STATE UNIV.
C
C       REVISION        DATE                     PROGRAMMER
C       --------        ----                     ----------
C       00.0            OCT 20, 1982             DIANE VON THAER
C       00.1            OCT 26, 1982             DIANE VON THAER
C       00.2            DEC 15, 1982             DIANE VON THAER
C       00.3            JAN 18, 1983             DIANE VON THAER
C       00.4            FEB 10, 1983             DIANE VON THAER
C       01.0            MAR 10, 1983             DIANE VON THAER
C
C***********************************************************************
C
C       PURPOSE
C
C               This program simulates a noise leveling
C               automatic gain control system, which can
C               include an LVA or not.
C
C       ROUTINE(S) CALLED BY THIS ROUTINE
C
C               FILTi
C               HPF
C
C***********************************************************************
C
C
C       INITIALIZATION
C
        COMPILER STATIC
        REAL GNOUT(1024),KFOUT(1024),K,KF,NR,NI,NPOINT,LVA
        REAL F0,K1,K2,G2OUT(1024),GAINT(1024)
        DIMENSION NAME1(13),NAME2(13),NAME3(13),NAME4(13)
        LOGICAL FIRST1,FIRST2,FIRST3,FIRST4,FIRSTH
        FIRST1 = .TRUE.
        FIRST2 = .TRUE.
        FIRST3 = .TRUE.
        FIRST4 = .TRUE.
        FIRSTH = .TRUE.
        ICHAN = 1
        A0 = 1.0
        GAIN = 1.0
        CHGE = 0.0
        IDUMP = 0
        F0 = 1.0
        INDEX = 0
C
C       FORMAT STATEMENTS
C
1       FORMAT ('0BUTTERWORTH PRE-FILTER')
2       FORMAT ('        RF BANDWIDTH                  :  ',G12.6,' Hz')
3       FORMAT ('        NUMBER OF POLES               :  ',I1)
4       FORMAT ('0LOW-PASS POST-FILTER')
```

```
5       FORMAT ('        LOW-PASS BANDWIDTH        :  ',G12.6,' Hz')
6       FORMAT ('        NUMBER OF POLES           :  ',I1)
7       FORMAT ('OHIGH-PASS POST-FILTER')
8       FORMAT ('        HIGH-PASS CUTOFF          :  ',G12.6,' Hz')
9       FORMAT ('        NUMBER OF POLES           :  ',I1)
10      FORMAT ('OAGC FILTER')
11      FORMAT ('        AGC FILTER BANDWIDTH      :  ',G12.6,' Hz')
12      FORMAT ('        NUMBER OF POLES           :  ',I1)
14      FORMAT ('OSIGNAL OUTPUT FILE               :  ',S23)
15      FORMAT (' LOOP VOLTAGE OUTPUT FILE         :  ',S23)
16      FORMAT ('OLENGTH OF SIMULATION  :  ',G12.6,'  seconds')
26      FORMAT (' TIME LENGTH OF OUTPUT FILES:   ',G12.6,'  seconds')
17      FORMAT (' SAMPLING TIME           :  ',G12.6,'  sec')
18      FORMAT (' NOISE VARIANCE                   :  ',G12.6,'  dBm')
19      FORMAT ('OGAIN CONSTANT           :  ',G12.6)
20      FORMAT (' FIRST DETECTOR GAIN              :  ', G12.6)
21      FORMAT (' NOISE DETECTOR GAIN              :  ', G12.6)
22      FORMAT (' SIGNAL LEVEL            :  ',G12.6,'   dBm')
23      FORMAT (' NOMINAL VOLTAGE                  :  ',G12.6)
24      FORMAT (' LVA IS INCLUDED ')
25      FORMAT (' NO LVA ')
C
C*****************************************************************
C
C       INPUT INFORMTION FROM CONSOLE
C
40      ACCEPT  '    RF BANDWIDTH OF PRE-FILTER (Hz) ?   ',B3RF
        ACCEPT  '    NUMBER OF POLES FOR PRE-FILTER (1-4) ?   ', NPH
        TYPE
        ACCEPT  '    BANDWIDTH OF LOW-PASS FILTER (Hz) ?   ', B3G1
        ACCEPT  '    NUMBER OF POLES (1-4) ?  ', NPG1
        TYPE
        ACCEPT  '    3 dB CUTOFF OF HIGH-PASS FILTER (Hz) ?  ', B3HP
        ACCEPT  '    NUMBER OF POLES (1-4) ?   ', NPG2
        TYPE
        ACCEPT  '    BANDWIDTH OF AGC FILTER (Hz) ?   ', B3F
        ACCEPT  '    NUMBER OF POLES FOR AGC FILTER (1-4) ?  ', NPF
        TYPE
        ACCEPT  '    INCLUDE LVA (1) OR NOT (2) ?  ',LOG
        ACCEPT  '    ENTER SYSTEM GAIN CONSTANT:    ', K
        ACCEPT  '    ENTER VALUE FOR K1  :    ', K1
        ACCEPT  '    ENTER VALUE FOR K2  :    ',K2
        ACCEPT  '    ENTER NOMINAL VOLTAGE  :  ', VNOM
        TYPE
C       ACCEPT  '    SAMPLING TIME ? (sec)  ',TS
        ACCEPT  '    ENTER NUMBER OF SAMPLES:   ',NPOINT
        ACCEPT  '    OUTPUT FILES AFTER ____ SAMPLES ?  ',TRPOINT
        ACCEPT  '    ENTER OUTPUT DUMP INTERVAL   ', NDUMP
        ACCEPT  '    NOISE VARIANCE ? (dBm)       ',VARDBM
        ACCEPT  '    TYPE 1 FOR NO VARIANCE CHANGE  :  ',CHGE
        TYPE
        CALL WNAME ('   OUTPUT FILE NAME?                  ',NAME2)
        CALL WNAME ('   LOOP VOLTAGE OUTPUT FILE NAME?    ',NAME3)
        CALL WNAME ('   NOISE VOLTAGE OUTPUT FILE NAME?  ',NAME4)
        CALL WNAME ('   GAIN OUTPUT FILE NAME?            ',NAME5)
C
C       DETERMINE INPUT SIGNAL
C
        TYPE
```

```
      TYPE ' INPUT SIGNAL  '
      ACCEPT  '  PULSE (1) OR CONSTANT (2) ?   ', ISIG
      GO TO (200) ISIG
C
      ACCEPT  '        SIGNAL VOLTAGE  ? (dBm)   ',ADBM
      GO TO 250
200   ACCEPT  '        PULSE RISE TIME (sec) ?  ', TR
      ACCEPT  '        PULSE WIDTH (sec) ?  ', TAU
      ACCEPT  '        PULSE MAGNITUDE ? (dBm)    ', ADBM
      ACCEPT  '        FUNDAMENTAL FREQUENCY  ?  ', FO
      ACCEPT  '        DELAY TIME  ?   ',TD
250   CONTINUE
      GO TO 70
C
C*****************************************************************
C
C       CALCULATE PARAMETERS FROM INPUT
C
70    TS = 2E-9
      NDUMP = IFIX((NPOINT-TRPOINT)/1000)
      B3H = B3RF/2.0
      PY = 10**(VARDBM/10) * 1E-3
      VAR = PY/(2*TS*B3H)
      A = (10**(ADBM/10) * 1E-3)**0.5
      PERIOD = 1/FO
C
C*****************************************************************
C
C       BEGIN SIMULATION OF NOISE LEVELING AGC SYSTEM
C
      T = 0.0
      DO 300 XYZ=0,NPOINT
C
      GO TO (100) ISIG
      SIGNAL = A
      IF (CHGE.EQ.1) GO TO 150
      IF (XYZ.EQ.(NPOINT-TRPOINT)/2.0) VAR = VAR/10.0
      GO TO 150
100   SIGNAL =  0.0
      IF (T.GT.TD.AND.T.LE.TR+TD) SIGNAL = (A*(T-TD))/TR
      IF (T.GE.TR+TD.AND.T.LT.TAU+TD) SIGNAL = A
      IF (T.GE.TAU+TD.AND.T.LT.TAU+TR+TD)
     &SIGNAL = A - (A*(T-TD-TAU))/TR
      T = T + TS
      IF (T.LT.PERIOD+TD) GO TO 150
      T = 0.0
      TD = 0.0
150   CONTINUE
C
      NR = GAUSS(VAR,0)
      NI = GAUSS(VAR,0)
      XR = SIGNAL + NR
      XI = NI
      HNR = GAIN * XR
      HNI = GAIN * XI
      HTR = FILT1(HNR,TS,NPH,B3H,FIRST1)
      HTI = FILT2(HNI,TS,NPH,B3H,FIRST2)
      HSQR = K1 * (HTR**2 + HTI**2)
      GO TO (255,256) LOG
```

```
255     HSQR = 1/(10.0*ALOG(10.0)) * ALOG(HSQR+1E-6) + 0.6
256     GN1 = FILT3(HSQR,TS,NPG1,B3G1,FIRST3)
        GN2 = HPF(GN1,TS,NPG2,B3HP,FIRSTH)
        FIN = (K2 * GN2)**2
        FOUT = FILT4(FIN,TS,NPF,B3F,FIRST4)
        GAIN = A0 - (K * (FOUT - VNOM))
        IF(GAIN.LT.0.001) GAIN = 0.001
260     IF (XYZ.LT.TRPOINT) GO TO 300
270     IDUMP = IDUMP + 1
C
C       OUTPUT DATA FILES AT DUMP INTERVAL
        IF (IDUMP.LT.NDUMP) GO TO 300
        INDEX = INDEX + 1
        GNOUT(INDEX) = GN1
        KFOUT(INDEX) = FOUT
        G2OUT(INDEX) = GN2
        GAINT(INDEX) = GAIN
        IDUMP = 0
300     CONTINUE
        CALL WDATA (ICHAN,'R',INDEX,NAME2,GNOUT)
        CALL WDATA (ICHAN,'R',INDEX,NAME3,KFOUT)
        CALL WDATA (ICHAN,'R',INDEX,NAME4,G2OUT)
        CALL WDATA (ICHAN,'R',INDEX,NAME5,GAINT)
C
C*************************************************************
C
C       DOCUMENT RUN
C
        CALL WAIT
        CALL HEADER ('NOISE-LEVELING AGC')
        IF(LOG.EQ.1) PRINT 24
        IF(LOG.EQ.2) PRINT 25
        PRINT 1
        PRINT 2,  B3RF
        PRINT 3,  NPH
        PRINT 4
        PRINT 5,  B3G1
        PRINT 6,  NPG1
        PRINT 7
        PRINT 8,  B3HP
        PRINT 9,  NPG2
        PRINT 10
        PRINT 11, B3F
        PRINT 12, NPF
        PRINT 16, NPOINT*TS
        PRINT 26, (NPOINT - TRPOINT)*TS
        PRINT 17, TS
        PRINT 22, ADBM
        PRINT 18, VARDBM
        PRINT 19, K
        PRINT 20, K1
        PRINT 21, K2
        PRINT 23, VNOM
        PRINT 14, NAME2(1)
        PRINT 15, NAME3(1)
        END
```

```
C*******************************************************************
C
C       FILTER
C
C       DG FORTRAN 5 SOURCE FILENAME:               FILT1.FR
C
C       DEPARTMENT OF ELECTRICAL ENGINEERING    KANSAS STATE UNIV.
C
C       REVISION        DATE                    PROGRAMMER
C       --------        ----                    ----------
C       01.0            OCT 15, 1982            DIANE VON THAER
C
C*******************************************************************
C
C       CALLING SEQUENCE
C
C               Y = FILT1(X,TS,NPOLE,B3,FIRST)
C
C       PURPOSE
C
C               This function subprogram calculates the value
C               of a transfer function given the sampling time,
C               the 3 dB bandwidth and the number of poles.
C
C       ROUTINE(S) CALLED BY THIS ROUTINE
C
C               NONE
C
C       ARGUMENT(S) REQUIRED FROM THE CALLING ROUTINE
C
C               X       signal value
C               TS      sampling time
C               NPOLE   number of poles
C               B3      3 dB bandwidth
C
C*******************************************************************
C
C       NOTE 1: This subroutine makes no checks on the validity
C               of the data supplied by the calling routine.
C
C       NOTE 2: Argument(s) supplied by the calling routine are
C               not modified by this subroutine.
C
C*******************************************************************
C
        COMPILER STATIC
        FUNCTION FILT1(X,TS,NPOLE,B3,FIRST)
        LOGICAL FIRST
        REAL    Y1,Y2,Z1,Z2,E,K,KK,EE
        W3 = 2.0 * 3.14159 * B3
        IF(.NOT.FIRST) GO TO 500
        FIRST = .FALSE.
        Y1 = 0.0
        Y2 = 0.0
        Z1 = 0.0
        Z2 = 0.0
        GO TO (10,20,30,40) NPOLE
        TYPE "FILTER TYPE NOT AVAILABLE 1"
        RETURN
```

```
C       ONE POLE COEFFICIENTS
10      A0 = 0.0
        A1 = 1.0 * W3
        K = W3
        GO TO 500
C       TWO POLE COEFFICIENTS
20      A0 = 1.0 * W3**2
        A1 = 1.41421 * W3
        K = W3**2
        GO TO 500
C       THREE POLE COEFFICIENTS
30      A0 = 1.0 * W3**2
        A1 = 1.0 * W3
        K = W3**2
        B0 = 0.0
        B1 = 1.0 * W3
        KK = W3
        GO TO 500
C       FOUR POLE COEFFICIENTS
40      A0 = 1.0 * W3**2
        A1 = 0.765 * W3
        K = W3**2
        B0 = 1.0 * W3**2
        B1 = 1.848 * W3
        KK = W3**2
C       DIFFERENCE EQUATIONS FOR LOW-PASS FILTERS
500     E = K*X - A1*Y1 - A0*Y2
        Y2 = Y2 + TS*Y1
        Y1 = Y1 + TS*E
        GO TO (100,100) NPOLE
        XX = Y2
        EE = KK*XX - B1*Z1 - B0*Z2
        Z2 = Z2 + TS*Z1
        Z1 = Z1 + TS*EE
        FILT1 = Z2
        IF(NPOLE.EQ.3) FILT1 = Z1
100     IF(NPOLE.EQ.2) FILT1 = Y2
        IF(NPOLE.EQ.1) FILT1 = Y1
        RETURN
        END
```

```
C*******************************************************************
C
C       FILTER
C
C       DG FORTRAN 5 SOURCE FILENAME:                 FILT2.FR
C
C       DEPARTMENT OF ELECTRICAL ENGINEERING    KANSAS STATE UNIV.
C
C       REVISION        DATE                    PROGRAMMER
C       --------        ----                    ----------
C       01.0            OCT 15, 1982            DIANE VON THAER
C
C*******************************************************************
C
C       CALLING SEQUENCE
C
C               Y = FILT2(X,TS,NPOLE,B3,FIRST)
C
C       PURPOSE
C
C               This function subprogram calculates the value
C               of a transfer function given the sampling time,
C               the 3 dB bandwidth and the number of poles.
C
C       ROUTINE(S) CALLED BY THIS ROUTINE
C
C               NONE
C
C       ARGUMENT(S) REQUIRED FROM THE CALLING ROUTINE
C
C               X       signal value
C               TS      sampling time
C               NPOLE   number of poles
C               B3      3 dB bandwidth
C
C*******************************************************************
C
C       NOTE 1: This subroutine makes no checks on the validity
C               of the data supplied by the calling routine.
C
C       NOTE 2: Argument(s) supplied by the calling routine are
C               not modified by this subroutine.
C
C*******************************************************************
C
        COMPILER STATIC
        FUNCTION FILT2(X,TS,NPOLE,B3,FIRST)
        LOGICAL FIRST
        REAL    Y1,Y2,Z1,Z2,E,K,KK,EE
        W3 = 2.0 * 3.14159 * B3
        IF(.NOT.FIRST) GO TO 500
        FIRST = .FALSE.
        Y1 = 0.0
        Y2 = 0.0
        Z1 = 0.0
        Z2 = 0.0
        GO TO (10,20,30,40) NPOLE
        TYPE 'FILTER TYPE NOT AVAILABLE 2'
        RETURN
```

```
C       ONE POLE COEFFICIENTS
10      A0 = 0.0
        A1 = 1.0 * W3
        K = W3
        GO TO 500
C       TWO POLE COEFFICIENTS
20      A0 = 1.0 * W3**2
        A1 = 1.41421 * W3
        K = W3**2
        GO TO 500
C       THREE POLE COEFFICIENTS
30      A0 = 1.0 * W3**2
        A1 = 1.0 * W3
        K = W3**2
        B0 = 0.0
        B1 = 1.0 * W3
        KK = W3
        GO TO 500
C       FOUR POLE COEFFICIENTS
40      A0 = 1.0 * W3**2
        A1 = 0.765 * W3
        K = W3**2
        B0 = 1.0 * W3**2
        B1 = 1.848 * W3
        KK = W3**2
C       DIFFERENCE EQUATIONS FOR LOW-PASS FILTERS
500     E = K*X - A1*Y1 - A0*Y2
        Y2 = Y2 + TS*Y1
        Y1 = Y1 + TS*E
        GO TO (100,100) NPOLE
        XX = Y2
        EE = KK*XX - B1*Z1 - B0*Z2
        Z2 = Z2 + TS*Z1
        Z1 = Z1 + TS*EE
        FILT2 = Z2
        IF(NPOLE.EQ.3) FILT2 = Z1
100     IF(NPOLE.EQ.2) FILT2 = Y2
        IF(NPOLE.EQ.1) FILT2 = Y1
        RETURN
        END
```

```
C*****************************************************************
C
C       FILTER
C
C       DG FORTRAN 5 SOURCE FILENAME:           FILT3.FR
C
C       DEPARTMENT OF ELECTRICAL ENGINEERING    KANSAS STATE UNIV.
C
C       REVISION        DATE                    PROGRAMMER
C       --------        ----                    ----------
C       01.0            OCT 15, 1982            DIANE VON THAER
C
C*****************************************************************
C
C       CALLING SEQUENCE
C
C               Y = FILT3(X,TS,NPOLE,B3,FIRST)
C
C       PURPOSE
C
C               This function subprogram calculates the value
C               of a transfer function given the sampling time,
C               the dB bandwidth and the number of poles.
C
C       ROUTINE(S) CALLED BY THIS ROUTINE
C
C               NONE
C
C       ARGUMENT(S) REQUIRED FROM THE CALLING ROUTINE
C
C               X       signal value
C               TS      sampling time
C               NPOLE   number of poles
C               B3      3 dB bandwidth

C
C
C*****************************************************************
C
C       NOTE 1: This subroutine makes no checks on the validity
C               of the data supplied by the calling routine.
C
C       NOTE 2: Argument(s) supplied by the calling routine are
C               not modified by this subroutine.
C
C*****************************************************************
C
        COMPILER STATIC
        FUNCTION FILT3(X,TS,NPOLE,B3,FIRST)
        LOGICAL FIRST
        REAL    Y1,Y2,Z1,Z2,E,K,KK,EE
        W3 = 2 * 3.14159 * B3
        IF(.NOT.FIRST) GO TO 500
        FIRST = .FALSE.
        Y1 = 0.0
        Y2 = 0.0
        Z1 = 0.0
        Z2 = 0.0
        GO TO (10,20,30,40) NPOLE
```

```
      TYPE 'FILTER TYPE NOT AVAILABLE 3'
      RETURN
C     ONE POLE COEFFICIENTS
10    A0 = 0.0
      A1 = 1.0 * W3
      K = W3
      GO TO 500
C     TWO POLE COEFFICIENTS
20    A0 = 1.0 * W3**2
      A1 = 1.41421 * W3
      K = W3**2
      GO TO 500
C     THREE POLE COEFFICIENTS
30    A0 = 1.0 * W3**2
      A1 = 1.0 * W3
      K = W3**2
      B0 = 0.0
      B1 = 1.0 * W3
      KK = W3
      GO TO 500
C     FOUR POLE COEFFICIENTS
40    A0 = 1.0 * W3**2
      A1 = 0.765 * W3
      K = W3**2
      B0 = 1.0 * W3**2
      B1 = 1.848 * W3
      KK = W3**2
C     DIFFERENCE EQUATIONS FOR LOW-PASS FILTERS
500   E = K*X - A1*Y1 - A0*Y2
      Y2 = Y2 + TS*Y1
      Y1 = Y1 + TS*E
      GO TO (100,100) NPOLE
      XX = Y2
      EE = KK*XX - B1*Z1 - B0*Z2
      Z2 = Z2 + TS*Z1
      Z1 = Z1 + TS*EE
      FILT3 = Z2
      IF(NPOLE.EQ.3) FILT3 = Z1
100   IF(NPOLE.EQ.2) FILT3 = Y2
      IF(NPOLE.EQ.1) FILT3 = Y1
      RETURN
      END
```

```
C*********************************************************************
C
C       FILTER
C
C       DG FORTRAN 5 SOURCE FILENAME:             FILT4.FR
C
C       DEPARTMENT OF ELECTRICAL ENGINEERING      KANSAS STATE UNIV.
C
C       REVISION        DATE                      PROGRAMMER
C       --------        ----                      ----------
C       01.0            OCT 15, 1982              DIANE VON THAER
C
C*********************************************************************
C
C       CALLING SEQUENCE
C
C               Y = FILT4(X,TS,NPOLE,B3,FIRST)
C
C       PURPOSE
C
C               This function subprogram calculates the value
C               of a transfer function given the sampling time,
C               the 3 dB bandwidth and the number of poles.
C
C       ROUTINE(S) CALLED BY THIS ROUTINE
C
C               NONE
C
C       ARGUMENT(S) REQUIRED FROM THE CALLING ROUTINE
C
C               X       signal value
C               TS      sampling time
C               NPOLE   number of poles
C               B3      3 dB bandwidth

C
C
C*********************************************************************
C
C       NOTE 1: This subroutine makes no checks on the validity
C               of the data supplied by the calling routine.
C
C       NOTE 2: Argument(s) supplied by the calling routine are
C               not modified by this subroutine.
C
C*********************************************************************
C
        COMPILER STATIC
        FUNCTION FILT4(X,TS,NPOLE,B3,FIRST)
        LOGICAL FIRST
        REAL    Y1,Y2,Z1,Z2,E,K,KK,EE
        W3 = 2 * 3.14159 * B3
        IF(.NOT.FIRST) GO TO 500
        FIRST = .FALSE.
        Y1 = 0.0
        Y2 = 0.0
        Z1 = 0.0
        Z2 = 0.0
        GO TO (10,20,30,40) NPOLE
```

```
      TYPE "FILTER TYPE NOT AVAILABLE 4"
      RETURN
C     ONE POLE COEFFICIENTS
10    A0 = 0.0
      A1 = 1.0 * W3
      K = W3
      GO TO 500
C     TWO POLE COEFFICIENTS
20    A0 = 1.0 * W3**2
      A1 = 1.41421 * W3
      K = W3**2
      GO TO 500
C     THREE POLE COEFFICIENTS
30    A0 = 1.0 * W3**2
      A1 = 1.0 * W3
      K = W3**2
      B0 = 0.0
      B1 = 1.0 * W3
      KK = W3
      GO TO 500
C     FOUR POLE COEFFICIENTS
40    A0 = 1.0 * W3**2
      A1 = 0.765 * W3
      K = W3**2
      B0 = 1.0 * W3**2
      B1 = 1.848 * W3
      KK = W3**2
C     DIFFERENCE EQUATIONS FOR LOW-PASS FILTERS
500   E = K*X - A1*Y1 - A0*Y2
      Y2 = Y2 + TS*Y1
      Y1 = Y1 + TS*E
      GO TO (100,100) NPOLE
      XX = Y2
      EE = KK*XX - B1*Z1 - B0*Z2
      Z2 = Z2 + TS*Z1
      Z1 = Z1 + TS*EE
      FILT4 = Z2
      IF(NPOLE.EQ.3) FILT4 = Z1
100   IF(NPOLE.EQ.2) FILT4 = Y2
      IF(NPOLE.EQ.1) FILT4 = Y1
      RETURN
      END
```

```
C*******************************************************************
C
C       HIGH PASS FILTER
C
C       DG FORTRAN 5 SOURCE FILENAME:              HPF.FR
C
C       DEPARTMENT OF ELECTRICAL ENGINEERING    KANSAS STATE UNIV.
C
C       REVISION         DATE                      PROGRAMMER
C       --------         ----                      ----------
C       00.0             OCT 15, 1982              DIANE VON THAER
C
C*******************************************************************
C
C       CALLING SEQUENCE
C
C               Y = HPF(X,TS,NPOLE,B3,FIRST)
C
C       PURPOSE
C
C               This function subprogram calculates the value
C               of a transfer function given the sampling time,
C               the 3 dB cutoff frequency and the number of poles.
C
C       ROUTINE(S) CALLED BY THIS ROUTINE
C
C               NONE
C
C       ARGUMENT(S) REQUIRED FROM THE CALLING ROUTINE
C
C               X        signal value
C               TS       sampling time
C               NPOLE    number of poles
C               B3       3 dB cutoff
C
C
C*******************************************************************
C
C       NOTE 1: This subroutine makes no checks on the validity
C               of the data supplied by the calling routine.
C
C       NOTE 2: Argument(s) supplied by the calling routine are
C               not modified by this subroutine.
C
C*******************************************************************
C
        COMPILER STATIC
        FUNCTION HPF(X,TS,NPOLE,B3,FIRST)
        LOGICAL FIRST
        REAL    Y1,Y2,Z1,Z2,E,EE
        W3 = 2 * 3.14159 * B3
        IF(.NOT.FIRST) GO TO 500
        FIRST = .FALSE.
        Y1 = 0.0
        Y2 = 0.0
        Z1 = 0.0
        Z2 = 0.0
        GO TO (10,20,30,40) NPOLE
        TYPE 'FILTER TYPE NOT AVAILABLE HP'
```

```
        RETURN
C       ONE POLE COEFFICIENTS
10      A0 = 0.0
        A1 = 1.0 * W3
        GO TO 500
C       TWO POLE COEFFICIENTS
20      A0 = 1.0 * W3**2
        A1 = 1.41421 * W3
        GO TO 500
30      A0 = 1.0 * W3**2
        A1 = 1.0 * W3
        B0 = 0.0
        B1 = 1.0 * W3
        GO TO 500
40      A0 = 1.0 * W3**2
        A1 = 0.765 * W3
        B0 = 1.0 * W3**2
        B1 = 1.848 * W3
        KK = W3**2
500     E = X - A1*Y1 - A0*Y2
        Y2 = Y2 + TS*Y1
        Y1 = Y1 + TS*E
        GO TO (100,100) NPOLE
        XX = E
        EE = XX - B1*Z1 - B0*Z2
        Z2 = Z2 + TS*Z1
        Z1 = Z1 + TS*EE
        HPF = EE
100     IF(NPOLE.LT.3) HPF = E
        RETURN
        END
```

References

[1] M. Schwartz, W. Bennett, and S. Stein, Communication Systems and Techniques, McGraw-Hill, Inc., New York, 1966.

[2] D. Hummels, "An Analysis of Factors Influencing the Stability of the Omni Pulse Receiver AGC Loop," Motorola Technical Memo, April 8, 1980.

[3] N. Ahmed and T. Natarajan, Discrete-Time Signals and Systems, Reston Publishing Company, Inc., Reston, Virginia, 1983.

Acknowledgements

I would like to express my thanks to the members of my committee, Dr. Nasir Ahmed, Dr. Frank Tillman, and especially Dr. Donald R. Hummels, my major professor. I'd also like to thank Bob Schneider for all of his programming assistance. I also appreciate the support of my husband, Lewis, and my parents.

THE PERFORMANCE OF A NOISE LEVELING AUTOMATIC GAIN CONTROL SYSTEM

by

DIANE MARIE VON THAER

B.S., Kansas State University, 1982

AN ABSTRACT OF A MASTER'S THESIS

submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Electrical Engineering

KANSAS STATE UNIVERSITY

Manhattan, Kansas

1983

Abstract

A comparison of two variations of a noise leveling automatic gain control was made in this study. Noise leveling AGC systems are used to keep the false alarm rate constant in surveillance receivers where manual adjustments are impossible. The difference between the systems is that one system used a Logarithmic Video Amplifier (LVA) and the other used only linear amplifiers. A digital computer simulation was used to compare the systems' response to various inputs. The performance of the system without the LVA was better than that of the system with the LVA because the latter is marginally a stable in some cases.