Wideband Active Antenna Cancellation

By

Hana L. Adaniya

SUBMITTED TO THE DEPARTMENT OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF ENGINEERING
IN ELECTRICAL ENGINEERING AND COMPUTER SCIENCE
AT THE
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

JUNE 2008

Signature of Author:_____
Department of Electrical Engineering and Computer Science
May 23, 2008

Certified by:_____
John Stueve
Lincoln Laboratory Technical Staff
VI-A Thesis Supervisor

Certified by:_____
James K. Roberge
Professor of Electrical Engineering
MIT Thesis Supervisor

Accepted by:_____
Arthur C. Smith
Professor of Electrical Engineering and Computer Science
Chairman, Department Committee on Graduate Theses

Wideband Active Antenna Cancellation

by

Hana L. Adaniya

Submitted to the Department of Electrical Engineering and Computer Science
on May 23, 2008 in partial fulfillment of the
requirements for the Degree of Master of Engineering in
Electrical Engineering and Computer Science

ABSTRACT

There exists a simultaneous transmit and receive antenna system where the transmitted signal is creating wideband interference of the receiver. To resolve this interference problem, the isolation between the transmit antenna and the receive antenna must be increased. This thesis analyzes and discusses various strategies for antenna isolation and demonstrates the feasibility of an adaptive filtering approach on active signal cancellation. The final system design demonstrates that, with a broadband interference source in close proximity to a receiver, it is possible to provide 30 dB of isolation by using active cancellation.

VI-A Company Thesis Supervisor: John Stueve
Title: LL Technical Staff

MIT Thesis Supervisor: James K. Roberge, Sc.D.
Title: Professor of Electrical Engineering

Acknowledgements

First, I would like to thank Bob Atkins and Group 45 for presenting me with a challenging and interesting task for my thesis project. Most of all, I would like to thank my thesis supervisor John Stueve who guided me and motivated me throughout this year; and I'd like to thank my past supervisor Joe Pacheco who first taught me what research really meant.

I consider myself blessed to have been a part of the best group at Lincoln Lab; the potential to do great work with this group kept me motivated. This project could not have been completed without the help of everyone in Group 45. I would especially like to acknowledge the support I received from Phil Davis who always made himself available to answer questions and help me with any issue. I want to show my appreciation for the VI-A Program which presented me with this opportunity to work with Lincoln Laboratory and extend my learning to outside the classroom.

Above all, I would like to thank my family who encouraged and supported me in every step of my academic career.

Contents

Table of Figures

1 INTRODUCTION

1.1 VISION

Currently there exists a need in the field for a mobile transmit and receive antenna system to operate simultaneously in order to locate target devices. For the purposes of the application, a signal is broadcasted by the transmit antenna. The receive antenna mounted approximately 6 feet away listens for a very small signal emitted by the target device. For the purposes of this application, the two actions of transmitting and receiving must happen simultaneously. The difficulty with trying to receive and detect simultaneously lies in the close coupling between the transmit and the receive antenna. Coupling, defined as the transfer of energy between the two antenna, results in the interference of the receive antenna by the transmitted signal (1). The coupling interference is the unintentional side effect of having the two antenna mounted in such close proximity clamped on the driver and passenger side of an army vehicle. Since the transmitted signal is so loud, it saturates the front end of the receiver, effectively jamming the receive antenna from being able to detect the weaker target signal. This thesis addresses the question: How can we achieve both transmission and detection from the two antennas described when they are so closely coupled together?

In order for both the transmit antenna and the receive antenna to function without interference, in other words without saturation of the receiver, the desired isolation (the opposite of coupling) between the two must reach a level of 60 dB (i.e. a coupling level of -60dB). Current native isolation between two co-polarized Vivaldi antennas measures around 30 dB, leaving an unmet need for an additional 30 dB of active cancellation. This thesis examines the feasibility of achieving this additional isolation through a canceller system that subtracts out the transmit signal from the receive antenna utilizing the concepts of feedback. The value of the

thesis lies in the real-world applications and so the final goal for the project is to develop a prototype solution for the jamming/target detection interference problem.

1.2 THESIS OVERVIEW

Chapter 1 introduces and motivates the project. It explains the purpose of the work and describes the different stages of the research as well as the organization and development of the thesis itself.

Chapter 2 provides background for this thesis project. It sets up the transmit and receive antenna system and describes why receiver saturation is such a problem. This chapter also reviews previous approaches to increasing isolation between antennas. This discussion includes the current system solution in the field as well as the work of other researchers involved in the field of antenna isolation. The concept of signal cancellation has been well studied in other related fields, especially that of Acoustic Noise Cancellers. Exploring the work done in these related fields gives us insight into factors which affect the performance of the feedback cancellation implemented for this thesis project.

Chapter 3 begins the discussion of the thesis work. In the original set-up of the system, the transmit antenna broadcasted in the narrowband. Consequently, the first stage of the project initially developed an adaptive and fast-converging approach towards narrowband cancellation. The emphasis of the narrowband cancellation was placed on the fast-convergence algorithm used to achieve the desired levels of cancellation.

The second stage of the thesis work is presented in Chapter 4. In this stage, threats in the theater had evolved such that broadband jammers were now on board the vehicles and so cancellation on the receive antenna was required across a wide range of frequencies. Dealing with broadband signals is a much more complicated and consequently more interesting case of

signal cancellation. Because the purpose of the thesis project is to model a solution for signal cancellation that can be implemented in the field, some time was taken to design a compact hardware approach for implementing the canceller system design. A description of this hardware approach can be found in the same section.

Chapter 5 addresses the final stage of the thesis project. The final concept for wideband cancellation and the final demonstration in the lab is discussed in this section. This chapter also discusses some of the issues that needed to be resolved for the final implementation of the wideband canceller. The results are discussed in Chapter 6.

Chapter 7 summarizes the contributions of the thesis and discusses the potential for future work.

2 BACKGROUND

2.1 RECEIVE ANTENNA PROBLEM DEFINITION

Antenna can be considered as a spatial sampler of electromagnetic radiation as it propagates. It takes in airborne signals and converts them into electrical signals for processing. In order to have detection of weak target signals, receiver sensitivity cannot be compromised. However, the strong interference signal imposes impractical dynamic range requirements on later receiver components such as the Analog-to-Digital converter. Dynamic range is defined as the ratio of the highest to lowest input values that each component can process. Both values can be adjusted simultaneously by tuning the system with attenuators or amplifiers, but a constant ratio of input extremes must be preserved. Therefore, if the difference between the power levels of the highest received signal and the smallest detectable signal exceeds the dynamic range, then the receiver will saturate.

For the set-up described in the Introduction, the difference between the jamming signal and the weaker target signal exceeds the dynamic range of the receiver. With additional antenna isolation in place, the incoming power levels of the interference signal entering into the receive antenna will still need additional isolation such that it is within an acceptable range for concurrent operation of both transmitter and receiver. The consequence of receiver saturation is signal distortion (2). Therefore, input signal power causing saturation is not acceptable. The receiver saturation point and the receiver sensitivity often pull in opposite directions due to dynamic range constraints, as is made clear from study of a simple receiver block diagram in Figure 1 (3).

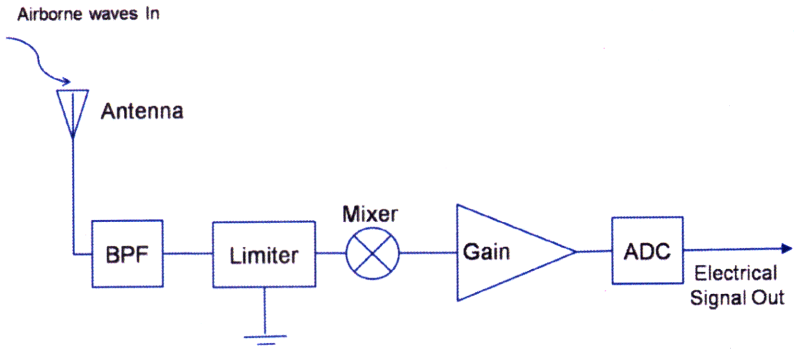

Figure 1. *Receiver Block Diagram*

First received by an antenna, the incoming signal is sent through a series of blocks. In the example shown, the signal first goes through a band pass filter which separates the desired signal from all the others picked up by the antenna (1). It then travels through a limiter stage which can shunt excess power to ground, as well as through a mixer stage which mixes down the RF input signal to an intermediate frequency (IF). In the final steps, Figure 1 shows the signal going through a gain stage and an Analog to Digital Converter (ADC). The ADC takes in an analog signal and has a fixed range of input power levels that it can digitize. If the input signal is too small, the level is less than the least-significant-bit of the ADC and will not be captured. A systems designer needs to design the system such that the expected input signal falls within the ADC's dynamic range. For a system such as ours looking for a weak signal, the amplifier in the gain stage is used to increase the signal level to within the range of the ADC thus increasing sensitivity. For a system with larger input signals, the amplifier can be reduced, eliminated, or possibly attenuated to reduce the input signal prior to the ADC. Since the receive antenna in the setup is receiving both weak and large signals, a careful balance must be struck to avoid overloading, i.e. saturating the components, in any later stage of the receiver (2). Cancellation

provides the mechanism to reduce the larger input signal prior to the ADC such that smaller input signals can be detected.

The problem of receiver saturation has been well studied since the advent of electronic warfare back in World War II (4). When done intentionally by the enemy, Electromagnetic Interference can prevent radar systems from working by saturating the receiver with a high energy signal. However, inadvertent jamming is just as much a problem. It comes from friendly sources that simply operate in the range of the affected receiver, as is the case with the Transmit and Receive Antenna System described in the introduction. Strategies, known as electronic countermeasures, deal with both types of jamming (4) (5). A discussion of some of the strategies developed for dealing with friendly interference can be found in the following section.

2.2 PASSIVE ISOLATION STRATEGIES

Friendly interference can be just as big a problem as intentional jamming since it accomplishes the same effect of preventing proper receiver function. To prevent friendly interference, one may take either a passive or an active approach. Passive strategies have many advantages especially in ease of implementation, but are limited in how they may be applied. While this thesis ultimately determines that for the particular application under examination an active approach is necessary, it is useful to examine passive approaches. With radar, one type of commonly experienced interference from friendly sources is that of aircraft whose radar can pick up signals from its own transponders. That kind of interference is dealt with by having the radar turn off its receive capabilities when it needs to transmit and the opposite when it needs to receive (6). This strategy, while effective, does not apply to the situation at hand because as the setup describes, simultaneous transmit and receive is required. Likewise, the setup does not allow for reduction in receiver sensitivity or a reduction in the transmitted signal power, which are two different simple yet effective strategies for avoiding interference (7).

Inadvertent electronic jamming is most easily prevented by carefully planning the location of radar. By moving the receiver out of the range of the transmit antenna, coupling (and consequently the interference) between the two antennas will decrease (1). This decrease in coupling is due to attenuation or path loss, a reduction in electromagnetic power with distance as a signal propagates through space (8).

$$Attenuation\ [dB] = \alpha \left[\frac{dB}{Mhz*cm}\right] \times l[cm] \times f[Mhz] \quad (2\text{-}1)$$

Attenuation, as is seen by the equation above, depends on the distance between the two radar, the frequency of the signal, and other environmental variables such as the attenuation coefficient α which varies by propagation media (8). Here again, though, the Transmit and Receive Antenna System does not allow for the application of spatial isolation to increase the path loss effects between the two antennas because of the physical geometry of the setup. The two antennas are mounted on either side of an army vehicle and therefore length l is fixed. To solve the need for isolation between the two antennas under spatial constraints, designers were forced to turn to other isolation strategies (9).

2.3 CURRENT IMPLMENTATION

In developing the jamming and detection antenna setup described in the introduction, developers were primarily concerned with rapid deployment of the system. The first challenge therefore was to quickly implement a strategy to achieve the isolation levels needed to keep the receiver from saturating. Designers adopted a passive approach to isolation in order expedite sending the system into the field. While no solution yet exists for achieving 60 dB of antenna isolation with co-polarized antennas, the current implementation achieves the isolation levels required by changing the polarity of the antennas to make them cross-polar. There are four basic polarities for antenna: horizontal, vertical, right-hand circular, and left-hand circular (10). A

transmitter and a receiver of different polarities results in signal loss. This signal loss is maximized if the antennas are cross-polarized with directly opposite polarities (10) (5).

In the case studied for two Vivaldi antennas, by giving the two antennas differing polarities, an increased isolation of around 30 dB can be achieved. Figure 2 shows the coupling levels between the antennas when they are either co-polarized or cross-polarized. At -60 dB, the dotted black line represents the desired coupling level needed between the two antennas to avoid receiver saturation. The red dashed line in the figure describes the native isolation of the co-polarized vertical antennas. As the figure indicates, the native isolation is around 30 dB. By changing the polarization of the receive antenna into a horizontal orientation the cross-polarization solution is achieved. The coupling in this cross-polarized situation is presented by the solid blue line in the figure and results in a native isolation that nears the desired level of 60dB. While this approach achieves the desired isolation levels, the phenomenology of the application would be better suited if the both antennas were vertically polarized. Therefore, the goal of this project is to lower the dashed red line to the levels of the dotted black line representing the desired level of isolation needed to avoid receiver saturation. In order for the two co-polarized antennas to achieve similar isolation levels as in the current setup, an active cancellation system must be implemented.

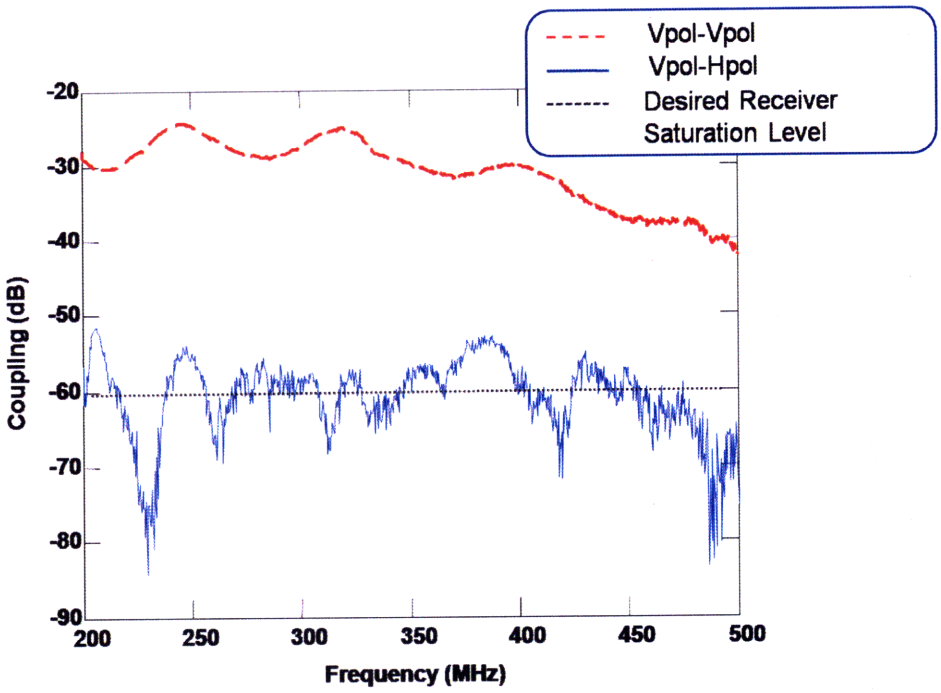

Figure 2. *Antenna Coupling Data for Co-polarized and Cross-polarized Vivaldi Antenna*

2.4 PREVIOUS WORK ON INTERFERENCE

Research has been done to deal with jamming involving phased arrays and adaptive beamformers. Bernard Widrow and other researchers worked on the problem of adaptive noise cancellation at Lincoln Laboratory in the 1980s. Their approach was to examine different adaptive beamformers to create a notch in the frequency response of the receive antenna in the jammer direction (11). Phased arrays use multiple antennas with phase difference such that the combination of all the signals results in a more powerful gain in the receive direction while suppressing the signals from other directions (12). However, these methods apply to the smart antenna themselves, while the approach this thesis project chose was to perform filtering on the signal of interest and cancel it through digital signal processing rather than through the properties of the antenna itself. Because of this signal processing approach, more can be learned from study of research in related fields such as acoustics. Acoustical noise cancellation and the

application of adaptive signal processing on the problem of noise control have gained momentum in recent years. Insight can be gained by studying the feedforward and feedback methods employed in the function of noise cancellation.

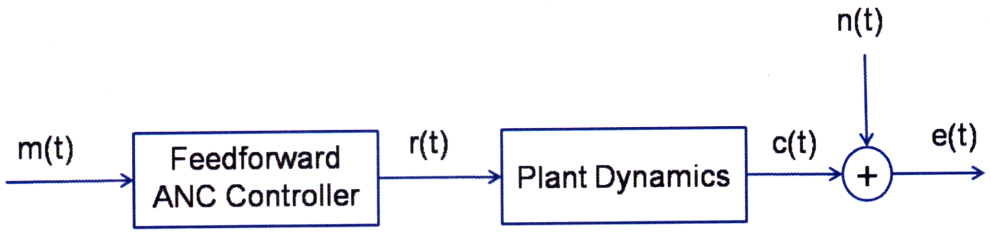

Figure 3. Feedforward Approach to Active Noise Control

In the feedforward system shown in Figure 3, the reference signal m(t) is measured by a microphone somewhere upstream from where the desired noise cancellation should occur (13). The goal of the system is to use that reference signal m(t) and generate an r(t) to cancel out the noise signal n(t) and therefore minimize the residual error e(t) (14) (13). Big problems with feedforward systems include the difficulty in finding an appropriate reference signal to be captured by a microphone as well as the possibility of feedback from the cancelling speaker into the input microphones. However, some researchers have had success implementing feedforward techniques in conjunction with patch antennas for simultaneous transmit/receive applications (15). The research presented by Karode and Fusco in 1999 is very similar to the narrowband canceller approach undertaken for this project and described in Chapter 3. In their research, Karode and Fusco achieve signal cancellation by designing a feedforward circuit to sum the leakage signal with an equal amplitude sample from the input but with destructive phase introduced.

The introduction of feedback resolves some of the limitations of feedforward systems. A feedback system such as the one seen in Figure 4 bases its canceller signal r(t) on measurements

taken of both the reference signal m(t) as well as the error e(t) that is being minimized (13) (16). Feedback is especially useful in adaptive control of dynamic systems such as with noise cancellation and thus inspired the application of adaptive digital signal processing techniques on the Transmit and Receive Antenna system that is the focus of this thesis.

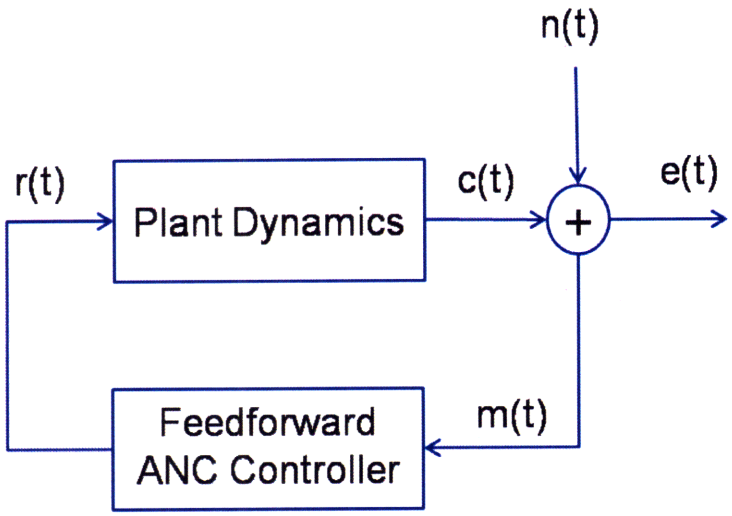

Figure 4. *Feedback Approach to Active Noise Control*

3 NARROWBAND CANCELLATION APPROACH

3.1 CANCELLATION SYSTEM MODEL

In the narrowband set-up of the problem, the transmit and receive antennas are tuned to radiate at a single frequency. The curves in Figure 5 represent the return loss or coupling between the two antennas and illustrate the fact that they are tuned at 5 MHz. Each antenna can be tuned at any frequency but for illustration purposes one frequency is enough to represent the fact that there is a need for a decrease in the antenna coupling levels (represented by the solid black curve). Moreover, with a native isolation of only 10dB at this frequency there is a need for 50 additional decibels of cancellation.

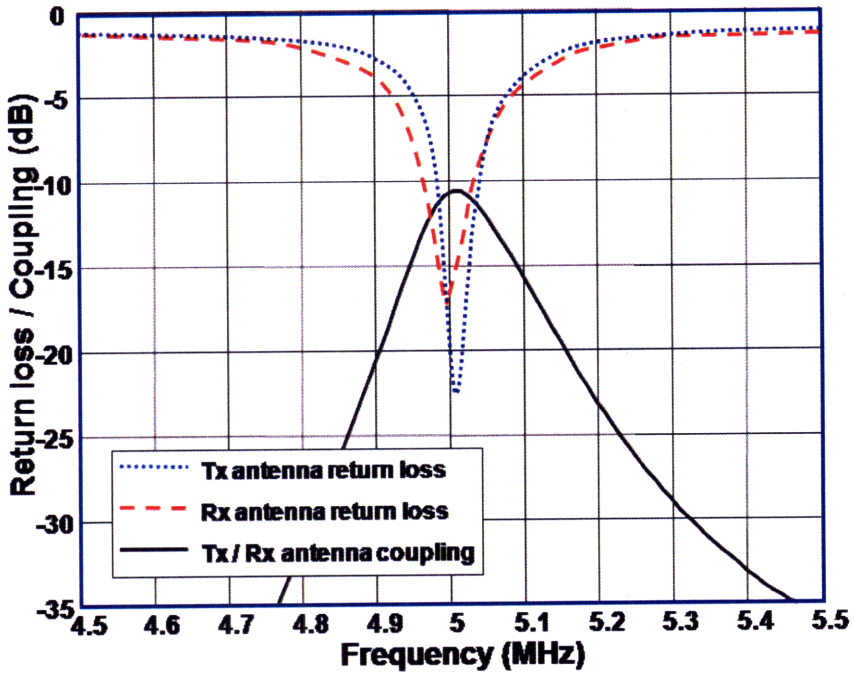

Figure 5. Antenna Tuning and Narrowband Coupling Plot

The problem, again, with the coupling levels demonstrated by the figure come from the overloading or saturation of the receiver. What is happening can be demonstrated by imagining a narrowband signal, or a single tone at 5 MHz whose amplitude exceeds the saturation window of the receiver. In order to prevent overloading of the receiver, the received signal must be added to a canceller signal with equal amplitude but destructive phase. When the received signal and the canceller signal are added then the residual error will be well within the receiver saturation window. This simple concept of narrowband signal cancellation through a matching of attenuation and phase inspired the model shown in Figure 6.

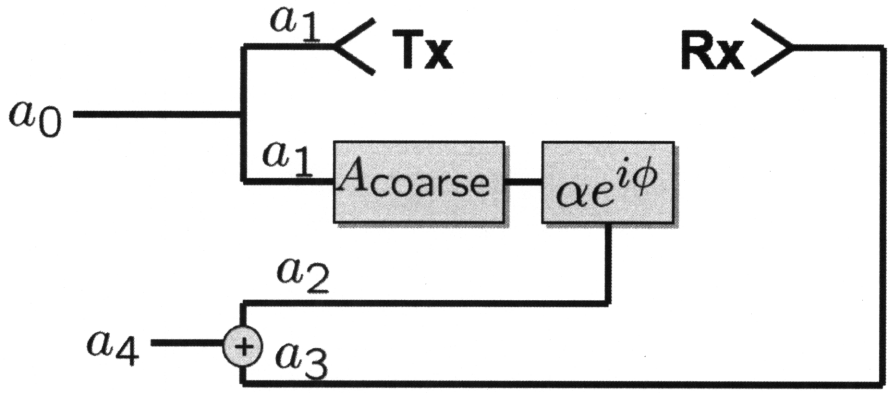

Figure 6. Narrowband Adaptive Nulling Model

In this model a_0 represents the original narrowband transmitted signal. In the model the original signal is split so that part of it is sent as a_1 through the Transmit-Receive Antenna channel where it is subjected to an unknown amount of attenuation and phase shift. Received by the Receive Antenna as the signal a_3, the receive signal is a modified version of the original signal. The other part of the original transmitted signal is sent through a filter that sets its attenuation and phase in an effort to match the effects of the antenna channel on the original signal. The goal is for the signal sent through the attenuator and phase shifter to become the canceller signal a_2, a destructive phase copy of a_3 such that when the signals are combined they

cancel each other out and minimize the residual error a_4. The goal is therefore to minimize the function:

$$P = |a_4|^2 = |a_3 + a_1 A_{coarse} \alpha e^{i\varphi}|^2 \quad (3\text{-}1)$$

In the equation shown, a_4 is the final signal from a_0 that the receiving antenna receives. The model also makes clear that the attenuation is split into two pieces the attenuation is split up into two pieces A_{coarse} and α.

3.2 FAST COVERGENCE ADAPTIVE NULLING ALGORITHM

To set the attenuation A_{coarse} and α as well as the phase shift φ, a fast-converging adaptive nulling algorithm was implemented operating on the method of steepest descent or gradient search. The method of steepest descent is recursive in the sense that its formulation or computation of a filter happens in a step-by-step manner utilizing the most recent gradient information (17). Starting with an initial guess for the attenuation and phase settings, the method of steepest descent calculates a gradient at that point and moves in the direction of the negative of the gradient. In other words, this method starts with a cost function J(w) that is continually differentiable. In the case of the model in Figure 6, the cost function is the error at a_4 and the weights w are the variables of attenuation and phase. In this cost function, the attenuation and phase settings have discreet steps. Therefore, since the power function is not differentiable, a gradient is approximated by dithering the amplitude and dithering the phase and calculating power differences.

The idea behind gradient search is that with the cost function $J(w)$, it is possible to find an optimum solution $J(w_0)$ such that:

$$J(w_0) \leq J(w) for\ all\ w. \quad (3\text{-}2)$$

Local iterative descent means that starting with an initial guess w(0), we can update w(n) to a w(n+1) such that the cost function maintains the condition:

$$J(w(n+1)) \leq J(w(n)). \tag{3-3}$$

The adjustments to the weights (in this case the attenuation and the phase settings) are in the direction opposite the gradient of J(w). The gradient of J(w) is the derivative of the cost function with respect to the weights w.

$$\nabla J(w) = \frac{\partial J(w)}{\partial w}. \tag{3-4}$$

Formally, the steepest descent algorithm can be written as:

$$w(n+1) = w(n) - \frac{1}{2}\mu \nabla J(w) \tag{3-5}$$

In the weight update equation, n is the iteration and μ is the step-size parameter. The step-size can be modified so that as the algorithm approaches a minimum it takes smaller and smaller steps. Adjustments were made in step-size on the basic algorithm to accelerate the convergence. The gradient approximation (calculated by dithering the amplitude and phase) refined the step-size as it approached the null taking smaller and smaller steps with each sub-iteration k as described by equations (3-6) and (3-7).

$$A_{j+1} = A_j + \Delta A r_{aj} 2^{(k-1)} \tag{3-6}$$

$$\emptyset_{j+1} = \emptyset_j + \Delta \emptyset r_{\emptyset j} 2^{(k-1)} \tag{3-7}$$

The gradient search algorithm can be visualized with the color map representation in the figure below.

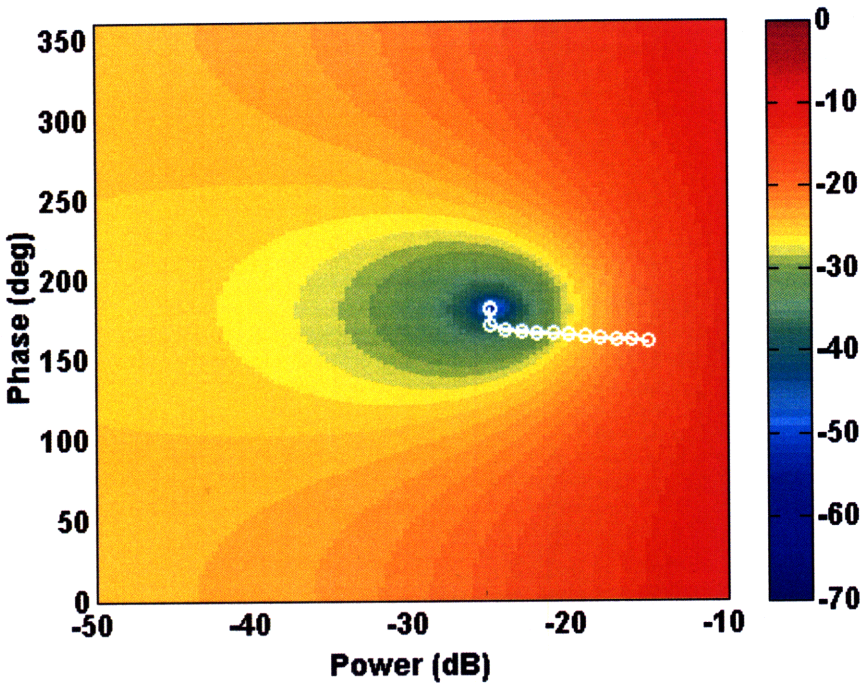

Figure 7. *Narrowband Gradient Search*

In Figure 7, the search space is represented by the two variables attenuation and phase. Colors represent the cost function of power where the null is shown in the center in blue. The algorithm starts at an initial guess of Attenuation around -15 dB and a phase of around 180°. From that initial guess, the algorithm adjusts the weights (in this narrowband case the attenuation and phase settings) in the direction opposite the gradient so that each evaluation of the algorithm brought the decreased the power levels until it reaches the desired isolation levels in blue of around -60 dB.

The drawbacks to this approach come from the search space being limited to two degrees of freedom, namely the amplitude and the phase. With this limitation in weights, the cancellation approach described in this chapter is limited to narrowband.

4 WIDEBAND CANCELLATION SYSTEM DESIGN

4.1 CANCELLATION CONCEPT

Threats in the theater of operations evolved as problems were solved and consequently the scope of the cancellation efforts had to adapt to these changes. Currently, broadband jammers can be found on board army vehicles so that the transmitted signal causing interference encompasses a wide range of frequencies. Therefore a need exists for a wideband cancellation design to replace the narrowband implementation discussed in the previous section.

Unlike narrowband, a wideband signal can be made up of multiple frequencies and therefore is more difficult to cancel out. However, for each frequency, the same concept of amplitude matching and destructive phase matching can still apply for broadband signals as it did with the narrowband approach. The difficulty lies in the fact the amplitude and phase matching can no longer be accomplished with a simple attenuator and phase shifter as was possible with narrowband signals; rather, it must encompass a more complicated adaptive filtering process.

Determining the weight settings on the filter that adapts the original signal is key to the new filtering process. In general, all adaptive filtering models use the input signal x(t) to track or approximate the desired signal d(t) through a linear filter $H(\omega)$ with impulse response h(t). Three different methods of determining the linear filter h(t) were examined: Wiener Filters, Least-Squares Method, and Linear-Least-Mean Squares Estimation.

4.2 WIENER FILTERS

Figure 8 shows the representative block diagram for a Wiener filter. The goal of Wiener filters, first discovered in the 1940s by Norman Wiener, was to find the linear filter h(t) that would filter out noise that had corrupted a signal (12). The end result of the Wiener Filtering process is that with an optimum filter h(t), then the error signal e(t) is minimized such that:

$$[x(t)e^*(t)] = 0 \tag{4-1}$$

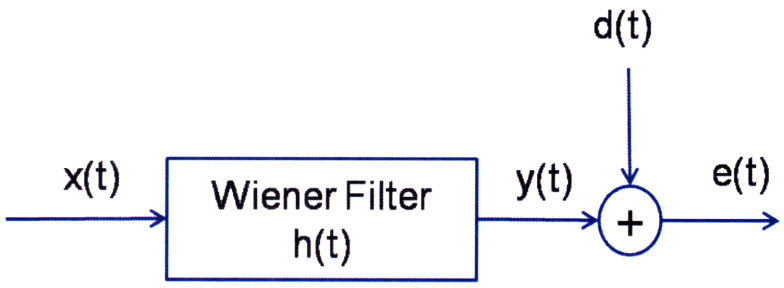

Figure 8. Wiener Filter Block Diagram

Since this thesis performs cancellation using digital signal processing, let us use discrete time to define FIR Wiener Filters. The idea starts by defining an error signal in equation (4-2).

$$e[n] = d[n] - y[n] \tag{4-2}$$

In the error signal, y[n] is the signal generated from the original signal to cancel out the desired signal d[n] as in equation (4-2). The Wiener filter h[n] is of order N with coefficients or weights w_i.

$$y[n] = x[n] * h[n] = \sum_{i=0}^{N} w_i x[n-i] \tag{4-3}$$

Plugging (4-2) into (4-1) the error is then:

$$e[n] = d[n] - \sum_{i=0}^{N} w_i x[n-i] \tag{4-4}$$

Then the value of the expected squared error is taken as $|e[n]|^2$, so that the optimal weight values w_i minimize this mean-squared-error.

The mean-squared error is written as:

$$E\{e^2[n]\} = E\{(d[n] - y[n])^2 \tag{4-5}$$

$$= E\{d^2[n]) - 2E\{d[n]y[n]\} + E\{y^2[n]\} \quad . \tag{4-6}$$

Plugging equation (4-3) in for y[n] yields:

$$E\{e^2[n]\} = E\{d^2[n]) - 2E\{d[n]\sum_{i=0}^{N} w_i x[n-i]\ \} + E\{(\sum_{i=0}^{N} w_i x[n-i]\ )^2\}. \quad (4\text{-}7)$$

To find the minimum expected mean-squared-error (MMSE), one must take the derivative with respect to the weights w_i and determine the optimum w_i from i = 1....N:

$$\frac{\partial}{\partial w_i} E\{e^2[n]\} = -2E\{d[n]x[n-i]\} + 2E\{(\sum_{j=0}^{N} w_j x[n-j])x[n-i]\} \quad (4\text{-}8)$$

$$= -2E\{x[n]d[n-i]\} + 2\sum_{j=0}^{N} E\{w_j x[n-j]x[n-i]\}. \quad (4\text{-}9)$$

By introducing auto-correlation and correlation factors $R_{xx}[m]$ and $R_{dx}[m]$,

$$R_{xx}[m] = E\{x[n]x[n-m]\} \quad (4\text{-}10)$$

$$R_{xd}[m] = E\{x[n]d[n-m]\} \quad (4\text{-}10)$$

then the derivative in equation (4-9) can be expressed as:

$$\frac{\partial}{\partial w_i} E\{e^2[n]\} = -2R_{xd}[i] + 2\sum_{j=0}^{N} w_j R_{xx}[j-i] \ for\ i = 0, \dots, N. \quad (4\text{-}11)$$

Setting the derivative in equation (4-11) equal to zero, results in:

$$R_{xd}[i] = \sum_{j=0}^{N} w_j R_{xx}[j-i] for\ i = 0, \dots, N. \quad (4\text{-}12)$$

Equation (4-12) can then be rewritten in matrix form as:

$$\begin{bmatrix} R_{xx}[0] & R_{xx}[1] & \cdots & R_{xx}[N] \\ R_{xx}[1] & R_{xx}[0] & \dots & R_{xx}[N-1] \\ \vdots & \vdots & \ddots & \vdots \\ R_{xx}[N] & R_{xx}[N-1] & \dots & R_{xx}[0] \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_N \end{bmatrix} = \begin{bmatrix} R_{xd}[0] \\ R_{xd}[1] \\ \vdots \\ R_{xd}[N] \end{bmatrix}. \quad (4\text{-}13)$$

Notably, the Wiener Filter method has two limitations. First, it requires calculations of matrices for matrix multiplication. The implication is that all the entries in the matrix must be known in order to solve for the filter coefficients w_i. Because of this, matrix multiplication, this method is not directly adaptive. Secondly, the Wiener filter method relies on knowing the auto-correlation $R_{xx}[m]$ and the cross-correlation $R_{xd}[m]$.

4.3 LINEAR LEAST-SQUARES METHOD

The linear least-squares process uses the principles of mathematical regression to solve for the filter weights w_i. This method chooses the tap weights on the filter to approximate the solution of a set of linear equations relating the original signal x[n] with the desired signal d[n] that is received by the receive antenna. This solution minimizes the least-squared cost-error function from equation (4-5). The system of equations is solved in matrix form as:

$$\hat{h} = (A^H A)^{-1} A^H \tilde{d} \tag{4-14}$$

where A is the data matrix and $\tilde{d}$ is the sample of the desired signal d[n]. The least-squares filter was applied to user-generated data sent through a Gaussian random noise filter to simulate the channel distortion. Figure 9 shows the desired signal d[n] in solid blue as the True Received Signal. The plot demonstrates how the linear-least-squares filter was able to generate a very close estimate of that signal for cancellation as is shown with the dashed green line representing y[n]. The 30-tap filter that generated the signal y[n] is shown in Figure 10.

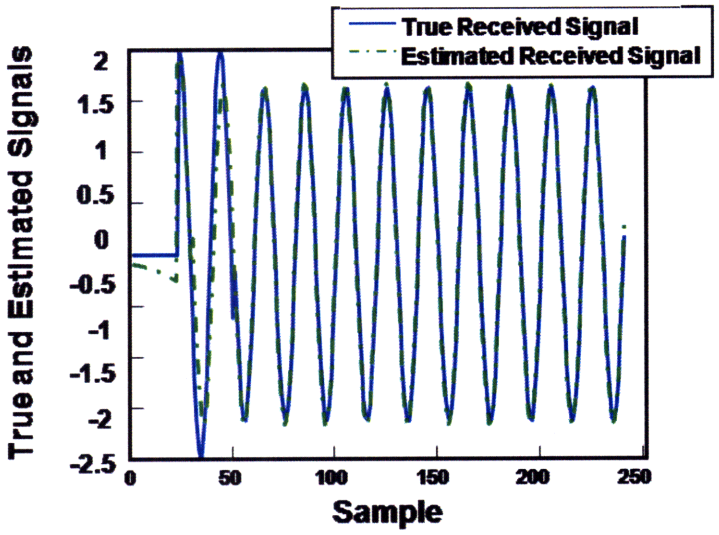

Figure 9. *Cancellation with Least-Squares Filtering*

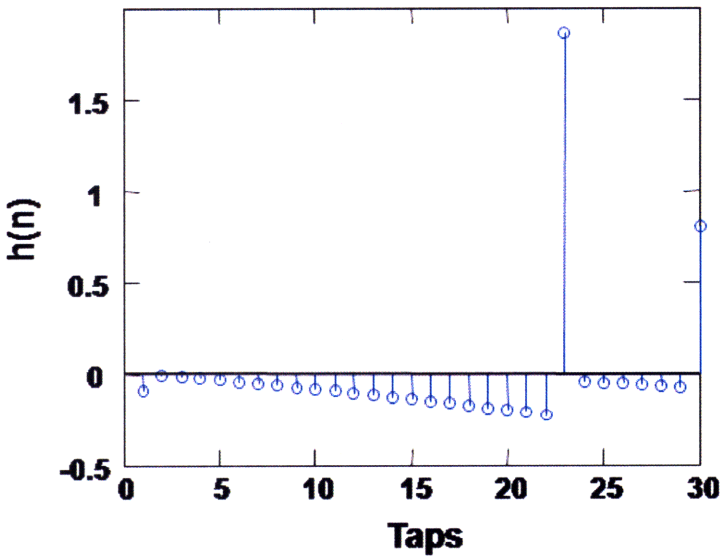

Figure 10. Least-Squares Filter Weights

The Least-Squares Filter method notably uses $\tilde{d}$, to set the filter weights where $\tilde{d}$ is a sampling of the desired signal d[n] meaning the method designs the filter based only on a set of training data. This is considered batch-processing (18) (19). As can be seen from equation (4-14), all the measurement information must be taken before the method can be used and like the Wiener Filter, the Least-Squares method also calculates filter coefficients by relying on complicated matrix operations, in this case a matrix inverse. For these reasons, this method was discarded in favor of the linear-least-mean-squares method discussed in the following section.

4.4 LEAST-MEAN-SQUARES METHOD

The least-mean-squares method differs from the other wideband cancellation methods analyzed because it bases its filter weights on the most recent values of the input signal, the output signal, and the error signal. The least-mean-squares method is thus the most adaptive to changing data and the most numerically robust. If the method were used to minimize current error rather than the mean-squared error the LMS method would converge to the same filter

solution as that of Wiener filters. The model used for the least-mean-squares filter is shown in Figure 11 below.

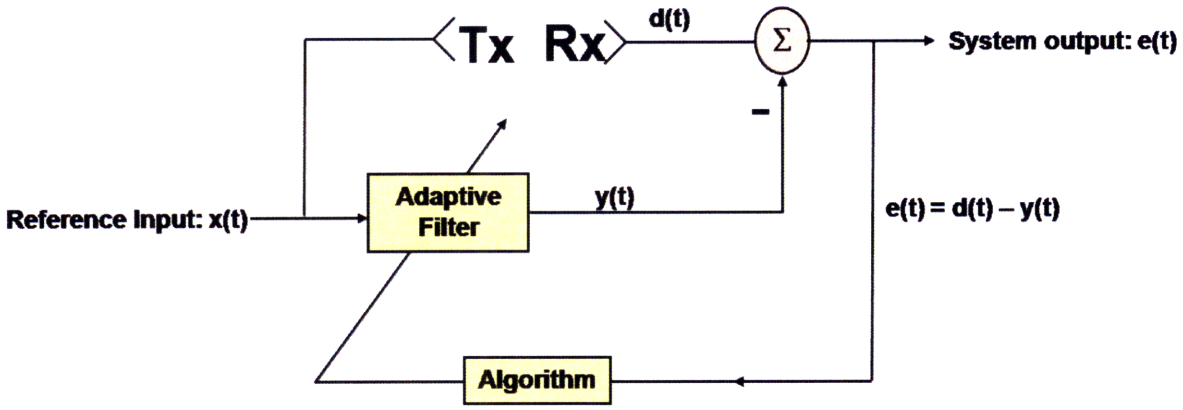

Figure 11. Adaptive Nulling Model

The LMS Algorithm takes the input, output, and error data and uses it to control the weights of the Adaptive Filter. The Adaptive Filter attempt to match the unknown system effects of the channel distorting the transmitted signal x(t) into the version received by the receive antenna y(t). A closer look at the Adaptive Filter shows how the Least-Mean-Squares algorithm calculates the weights (17).

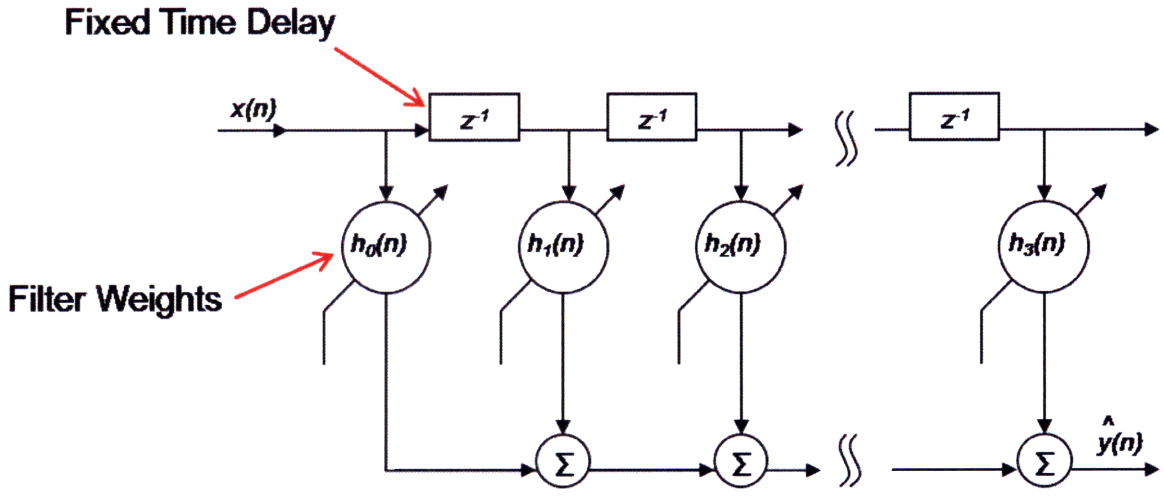

Figure 12. Least-Mean-Square Algorithm

The least-mean-square adaptive filter operates utilizing the same concept of gradient search described in section 3.2. The gradient search used to calculate the phase and attenuation for a narrowband cancellation process constitutes one leg of the filter described in Figure 12. The LMS process calculates an instantaneous estimate of the gradient vector found in equation 4-15 (20).

$$\hat{\nabla}J[n] = -2\boldsymbol{u}[n]d^*[n] + 2\boldsymbol{u}[n]\boldsymbol{u}^H[n]\hat{\boldsymbol{w}}[n] \tag{4-15}$$

$\hat{\nabla}J[n]$ in equation (4-15) is the gradient of the instantaneous squared error $|e(n)|^2$. $\boldsymbol{u}[n]$ is the tap-input vector of the same order as estimated weight filter $\hat{\boldsymbol{w}}[\boldsymbol{n}]$. More specifically, if $\hat{\boldsymbol{w}}[\boldsymbol{n}]$ is order M, then $\boldsymbol{u}[n]$ is the M-by-1 tap-input vector taken from the original signal x[n] at time n.

$$\boldsymbol{u}(\boldsymbol{n}) = [\boldsymbol{u}(\boldsymbol{n}), \boldsymbol{u}(\boldsymbol{n}-\boldsymbol{1}), \ldots, \boldsymbol{u}(\boldsymbol{n}-\boldsymbol{M}+\boldsymbol{1})]^T \tag{4-16}$$

This method is recursive in that the filter is computed iteratively in a step-by-step fashion. First, with the current estimate of the tap-weight vector $\hat{\boldsymbol{w}}[\boldsymbol{n}]$, the filter output is calculated by using $\boldsymbol{u}(n)$, the most recent M samples of the original signal x[n], and multiplying it through the filter as in equation (4-17). $\hat{\boldsymbol{w}}[\boldsymbol{n}]$ can be initialized with prior knowledge or it can be set to $\hat{\boldsymbol{w}}[\boldsymbol{0}] = \boldsymbol{0}$.

$$y(n) = \hat{\boldsymbol{w}}^H(n)\boldsymbol{u}(n) \tag{4-17}$$

Using the same estimation error as before where d(n) is the desired response at time n:

$$e(n) = d(n) - y(n). \tag{4-18}$$

Plugging (4-17) into this estimation error yields

$$e(n) = d(n) - \hat{\boldsymbol{w}}^H(n)\boldsymbol{u}(n). \tag{4-19}$$

The tap-weight adaptation is calculated with step-size μ to be:

$$\hat{\boldsymbol{w}}(n+1) = \hat{\boldsymbol{w}}(n) + \mu \boldsymbol{u}(n) e^{*}(n). \tag{4-20}$$

Substituting (4-19) into the tap-weight adaptation yields:

$$\hat{\boldsymbol{w}}(n+1) = \hat{\boldsymbol{w}}(n) + \mu \boldsymbol{u}(n)[d^{*}(n) - \boldsymbol{u}^{H}(n)\hat{\boldsymbol{w}}(n)]. \tag{4-21}$$

The computational complexity is on the order of O(M) where M in the number of tap-weights used. So the more precise in tap-weights, the longer the filter takes to calculate. The user must balance the trade-off between accuracy and fast-convergence time. The advantages of the LMS method arise from the intrinsic properties of the filter. As the LMS filter is calculated, a very strong coupling or correlation develops between the tap-weight vector and the interfering signal such that the mean-square-error using this method may yield much better results than that of a Wiener filter (17).

Applying a 20-tap LMS filter on user-generated transmit data and coupling channel data, resulted in the development of the example filter in Figure 13, which successfully generated an estimate of the desired signal as is seen in Figure 14.

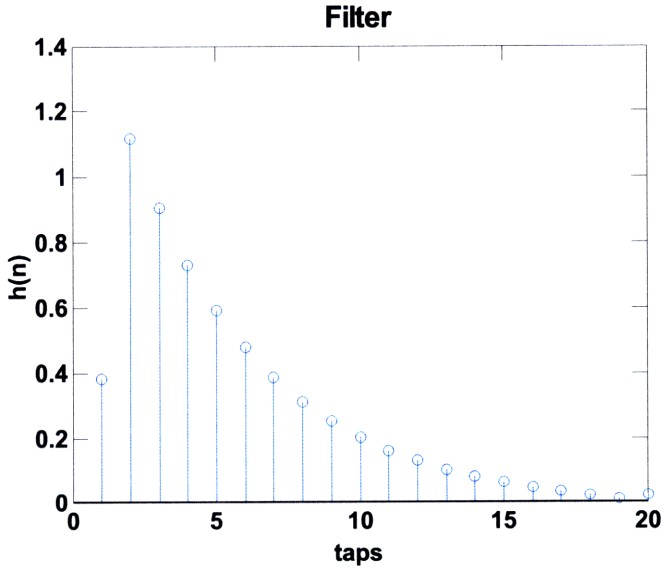

Figure 13. LMS Filter Development

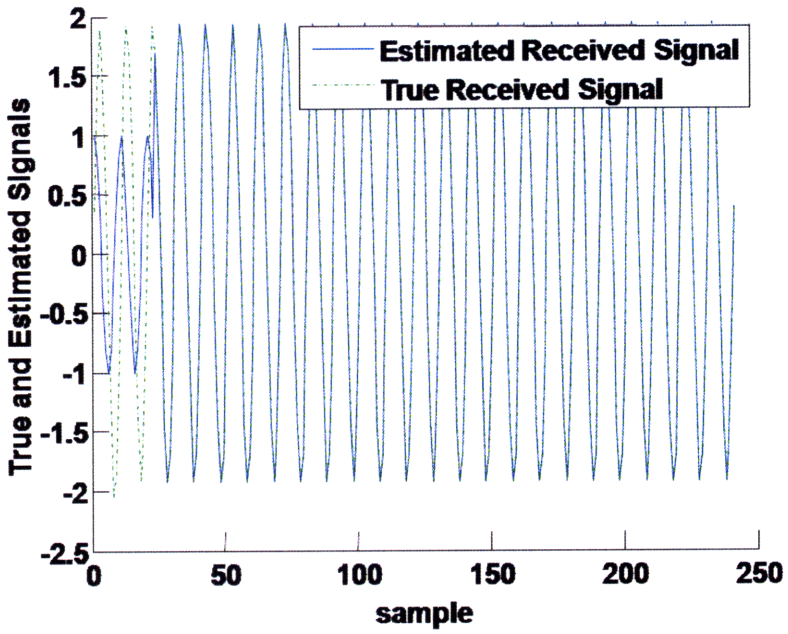

Figure 14. LMS Filter Results

As can be seen from Figure 14 the filter initialized with w(0) = 0 requires a few iterations related to the length of the filter before its output closely matches the true received signal in the dashed line. From the figure it is clear that a strong coupling develops between the filter and the received signal, and, with the adaptive process based on only the most recent data, as the received signal changes, then the filter adapts with it.

4.5 OFFLINE WIDEBAND CANCELLATION EXAMPLE

Once the algorithm was chosen and developed satisfactorily, the next step was to apply it to actual coupling data between the two Vivaldi antenna described in the set-up. Coupling data was taken from August 18, 2004 measurements of two vertically polarized antennas in an anechoic chamber. From this data, a channel was characterized for the effects of attenuation and phase shift on the transmitted signal. A user-generated 50ns Gaussian pulse (i.e. 20 MHz bandwidth

signal) was sent through the channel characterization as seen in Figure 15. The native isolation due to path-loss between the two antennas resulted in an attenuated pulse with a coupling of 30 dB as seen in Figure 16.

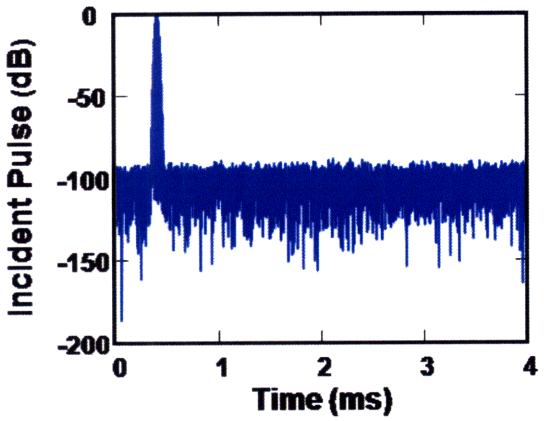

Figure 15. *Incident Gaussian Pulse*

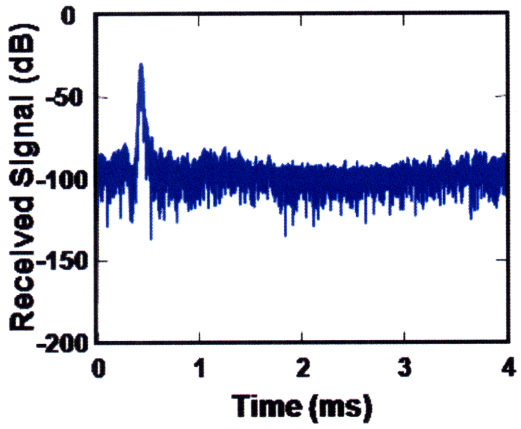

Figure 16. *Received Signal from Gaussian Pulse*

Once the original transmitted signal from Figure 15 and the received signal in Figure 16 were applied to the least-mean-square algorithm, the adaptive filter was able to cancel out the received signal and knock the pulse down to the levels of the noise floor, achieving the desired

cancellation of 60dB that would solve the problem of excess antenna interference. The results can be seen in Figure 17, where the received signal from Figure 16 is shown with the post-cancellation version on the right.

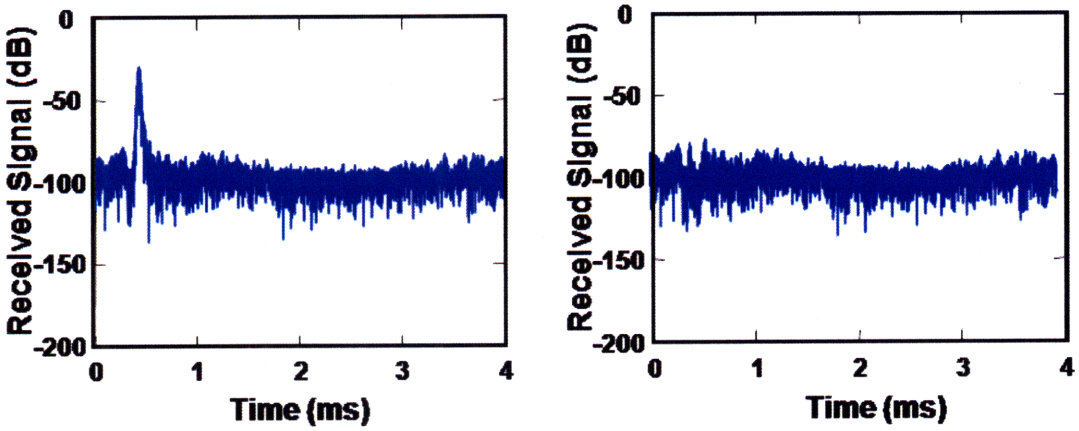

Figure 17. Received Signal Pre-Cancellation (left) and Post-Cancellation (right)

4.6 HARDWARE CONCEPT

4.6.1 DSP BOARD AND COMPONENTS

With the wideband cancellation approach shown to work in software post-processing, the next step was to implement the system in hardware. The end goal of the project was to make the canceller system effective, fast, and compact so that it could be easily mounted onto the army vehicle set-up described in the Introduction. To this end, it was decided to embed the algorithm onto a TI Digital Signal Processor Board. For experimentation, the TMS320F2812 evaluation board was used (Figure 18); it was chosen for its compact packaging, speed, and accuracy. A 3 inch by 5.35 inch board, the F2812 has an embedded Digital Signal Processor chip. Also found on the stand-alone board is a 12-bit Analog-to-Digital converter with a voltage range of 0-3V. These accuracy levels mean that the board can measure input voltages with an accuracy of:

$$\Delta V = {}^{3}/_{2^{12}} = 7.324 \times 10^{-4}\ \text{V} \tag{4-22}$$

Because of the limited voltage range of the board, a log-amplifier was necessary to compress the analog input data from antenna input power levels in the 0 dBm -100 dBm range into the 0-3V range readable by the board (see Figure 19 for log-amp specifications). This amplification makes lower power levels more visible and permits analysis of very small power levels with a coarse voltage.

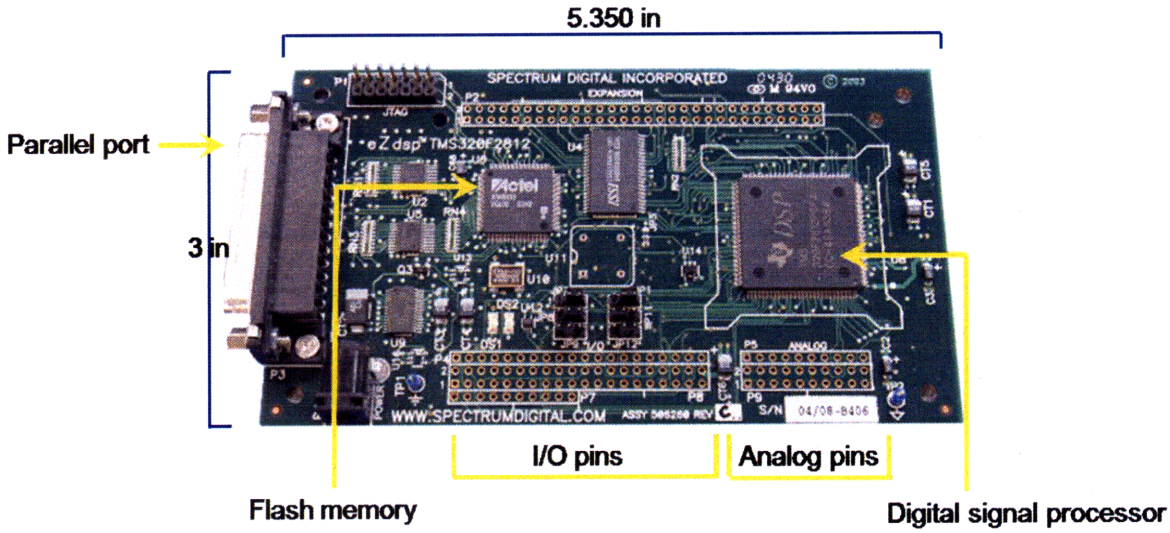

Figure 18. Texas Instruments TMS320F2812 Digital Signal Processing Board

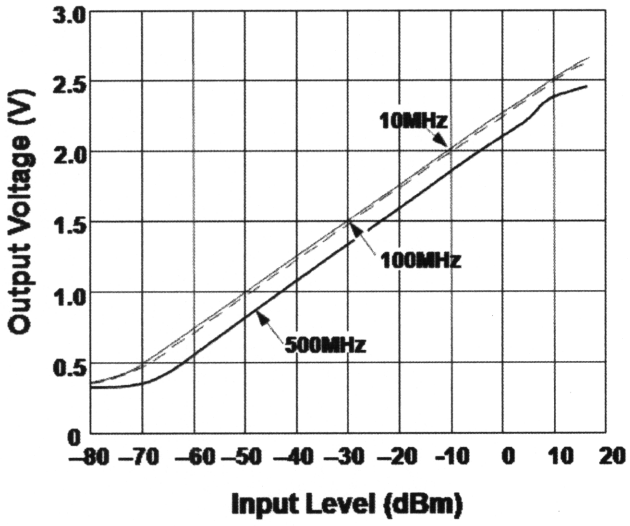

Figure 19. Log-Amp Response

The TI TMS320F2812 DSP chips have a memory of 18K x 16, but an additional 128K x 16 block of flash memory is available off-chip. The memory must be allocated according to a memory map, and the on-board components can be programmed through a parallel port interface to a computer running Code Composer Studio.

4.6.2 SYSTEM ARCHITECTURE

The thesis project followed the same approach to deal with the hardware concept as with the system model. Namely, the transmitted signal is split so that part of the signal is sent through the channel between the antennas and the other part is sent through an adaptive filter. Together the two signals recombine to minimize the sum. Since jammers may use Direct Digital Synthesis chips (DDS) to generate their broadband signals, the hardware concept uses a DDS to create the original transmitted signal. The adaptive filter in Figure 20 is made up of analog delays in place of the fixed time delays seen in Figure 12. Variable attenuators set the filter weights. The canceller algorithm is run on the Digital Signal Processing Board, which takes input from the signals and uses a control line to set the weights on the variable attenuators at the correct filter tap values. Because the DSP board cannot handle our input frequency range, the receiver mixes

down the RF input before it is input into the DSP board. Because of the dynamic range issues, the transmitted signal is initially kept within the dynamic range of the analog-to-digital converter in the receiver, and the digital processing line serves to set the initial tap weights using the digital error signal. Once the power of the transmitted signal is raised beyond the receiver threshold, the analog processing line is used to fine-tune the filter by perturbing the original signal and observing the power spectrum data on the error signal.

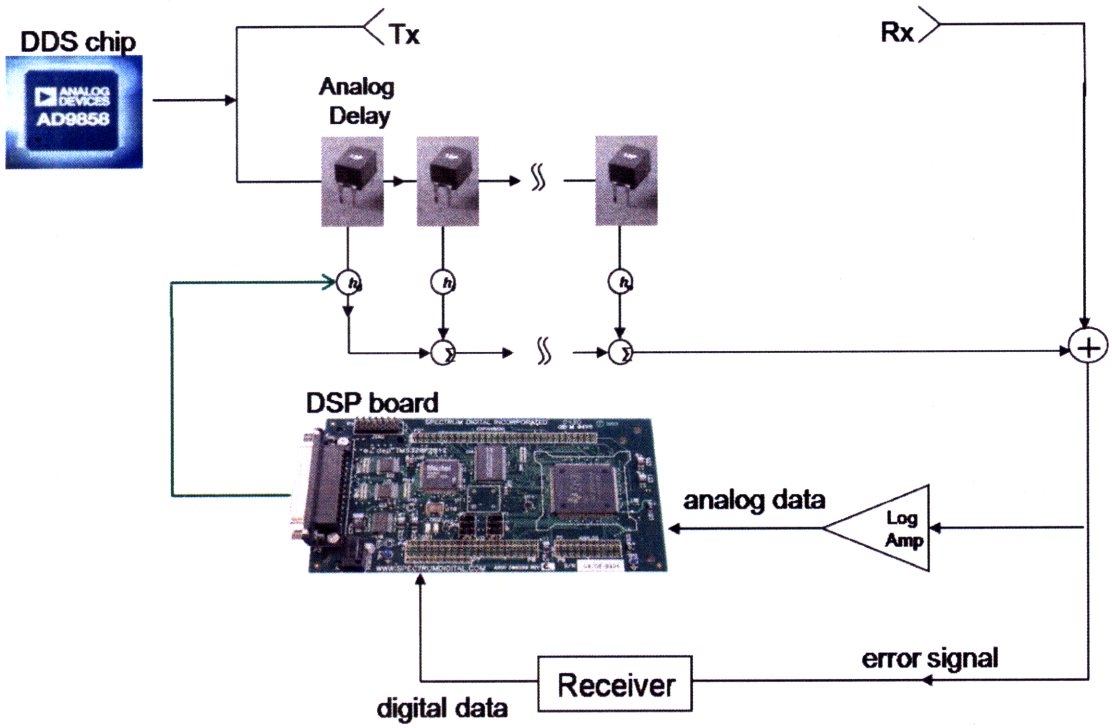

Figure 20. Hardware Concept

4.6.3 IMPLEMENTATION ISSUES

The hardware concept for the system was modeled in Code Composer studio and Simulink. Characterizations of the approach using the tap filter showed that it would not be practical to implement. While the approach was technically feasible, it was found to be impractical with

currently available components. The biggest obstacle was the analog tap delay line. While analog tap delays are available and inexpensive, they are not as easily found with the precision required to make this approach feasible. While phase shifters using simple lengths of coaxial cable could have been used, the number of analog tap delays (i.e. the order of the filter) that were needed to reach the desired levels of cancellation also made this approach prohibitive. Around filter of order 50-60 taps was needed to reach the desired isolation levels. The design approach changed to accommodate the practical limitations of the design project.

5 ARBITRARY WAVEFORM APPROACH

As mentioned in Section 4, jammers may use Direct Digital Synthesizer (DDS) chips to generate the broadband waveform signals they emit. By study of work on a previous system involving using two channels of a DDS chip, it was determined to discard the approach with variable attenuators and analog delays for an approach using a second DDS to cancel out the first. To get some initial results, this approach was demoed using test-equipment: 2 arbitrary signal generators to create the signals and a spectrum analyzer to receive.

5.1 DIRECT DIGITAL SYNTHESIZERS

Direct Digital Synthesizers chips can digitally create any arbitrary waveform from a single source frequency. A Numerically Controlled Oscillator (NCO) generates the source frequency. It is used to establish a sampling time from which digital samples of sinusoids can be computed; each increment in time drives a phase accumulator. Phase for a sinusoid is linear from 0 to 2π and so, given a reference clock period from the NCO, the phase can be precisely determined by

$$\Delta Phase = \omega\delta t \tag{5-1}$$

where δt is the reference clock period provided by the NCO. This can be rewritten as,

$$\omega = \frac{\Delta Phase}{\delta t} = 2\pi f. \tag{5-2}$$

Solving (5-2) for the frequency and substituting the source frequency from the NCO ($f_{source} = 1/\delta t$) , results in:

$$f = \frac{\Delta Phase \times f_{source}}{2\pi}. \tag{5-3}$$

Using the basic equation in (5-3), phase accumulators can then provide a phase to serve as a reference or an address for the DDS chip to map each phase to an amplitude stored in Random Access Memory (RAM). The RAM serves as a look-up table matching phase references to waveform amplitudes. The user can thus digitally program the look-up table to build any

arbitrary waveform. For a periodic waveform, one pass through the look-up table creates one period of the waveform. In the final steps, the digital samples are fed into a digital-to-analog converter (DAC) and to a low-pass interpolating filter which produces a smooth analog RF signal (21).

For the system, rather than stand-alone DDS chips, two arbitrary signal generators were used—the Agilent E8267D and the Agilent E4438C. These arbitrary signal generators have DDS circuits inside that allow them to load and extract user-defined arbitrary waveforms and play them as an RF signal. One of the advantages of using the arbitrary signal generators was the convenient interface with the equipment to a computer through LAN. These units could be controlled remotely by running MATLAB and sending SCPI commands to the equipment. One of the key features of the signal generators was the 16 bit Digital-to-Analog converters that they used to take the waveform sample user-data and transform it into an RF signal. Additionally, the arbitrary signal generators could load and play waveforms of user-generated data of up to 320Mbs.

The spectrum analyzer, the Agilent E4440A, read back both frequency spectrum and time domain data received on its RF input port. This spectrum analyzer had an "analysis bandwidth" of up to 10 MHz on which it could digitize data for further analysis and processing in time, frequency, or modulation domain. A digital oscilloscope, the Agilent 54832, was used in testing to display multiple signals: the trigger, the arb waveforms, and the cancellation error.

5.2 NARROWBAND DEMONSTRATION

5.2.1 HARWARE SET-UP

To begin testing the new approach, the hardware set-up was used to reproduce the results from the narrowband cancellation experiments. In these experiments, isolation from a

narrowband signal was achieved by setting attenuation and phase through a fast-converging gradient search approach. In this hardware set-up, one Arb served to transmit a single tone, while the second Arb was added to it by a power combiner to subtract it. The power combiner added an unavoidable ~3 dB to the signals. The sum port from the power combiner outputted the cancellation error and was sent to the spectrum analyzer and read by the computer. All three pieces of equipment were controlled remotely by a computer running the fast-converging nulling algorithm on MATLAB and sending SCPI commands to the equipment. The set-up can be seen in Figure 21 below.

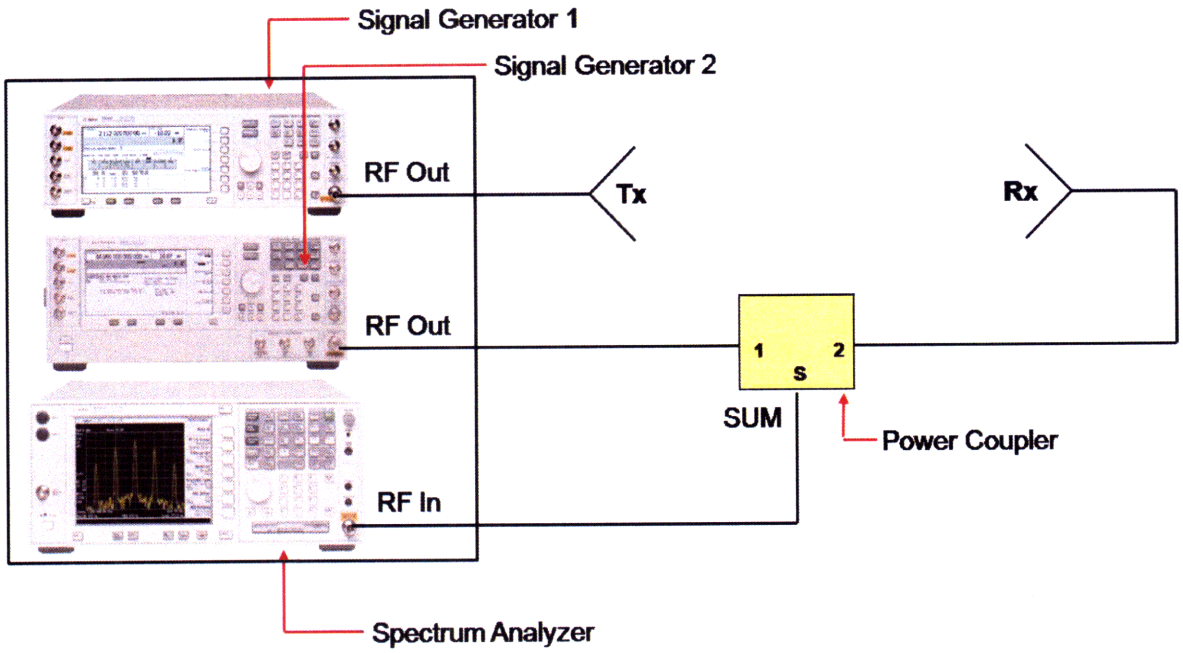

Figure 21. Narrowband Experiment Set-up

5.2.2 INITIAL TESTING

Initial testing took place with a closed-loop set-up where Signal Generator 1 was connected directly to the power combiner. Without the antenna channel, the only attenuation in the closed-loop model came from small wire-loss as is seen in the Baseline reading from the spectrum

analyzer on the left of Figure 22. In the demonstration, Signal Generator 1 transmitted a single tone at 200 MHz received by the spectrum analyzer at a power level around 0 dBm. After the cancellation signal was added with the algorithmically determined attenuation and phase settings, the isolation reached -52.6 dBm. With a sampling of tones across a 5 – 500 MHz bandwidth, active cancellation of 50-55dB was achieved consistently in about 50-60 evaluations of the LMS algorithm.

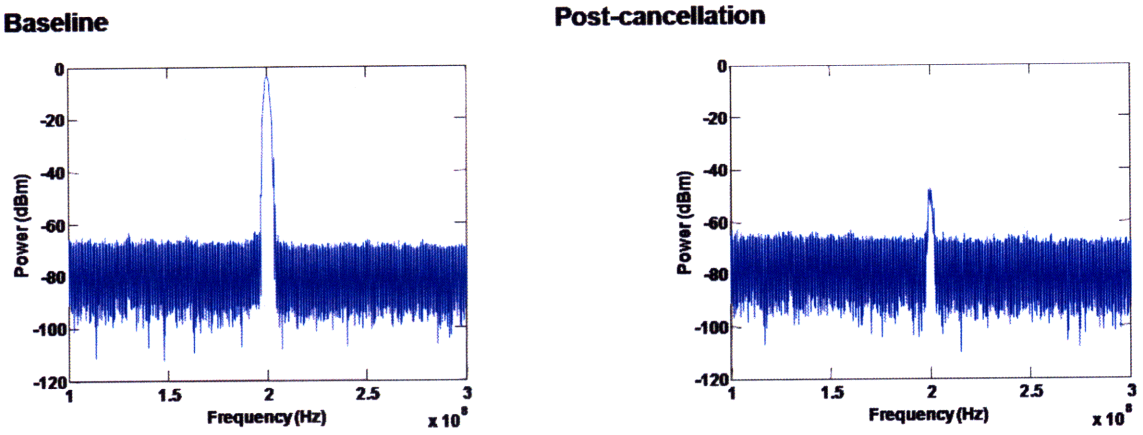

Figure 22. Narrowband Cancellation of 200 MHz Signal

Progress of the algorithm could be observed in the time-domain adjustments of attenuation and phase as the power of the received signal at the transmitted frequency was decreased until a null was reached. The progress of the attenuation and phase settings is shown below in Figure 23. The progress demonstrates how an initial guess was made for the phase setting by roughly calculating power gradients using different phases. As the algorithm progresses towards the null, the adjustments on attenuation and phase became finer and finer as seen by the zoomed in portion of the phase setting in the plot to the right of Figure 23. Cancellation progress with each iteration of the algorithm can be seen in Figure 24. A null is reached of 60 dB in 60-70 evaluations.

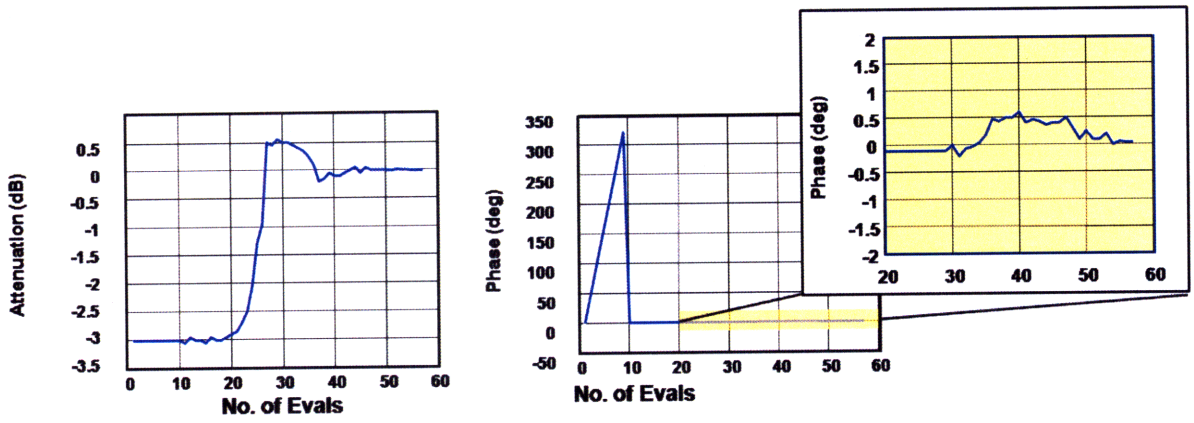

Figure 23. Narrowband Cancellation: Attenuation and Phase Settings

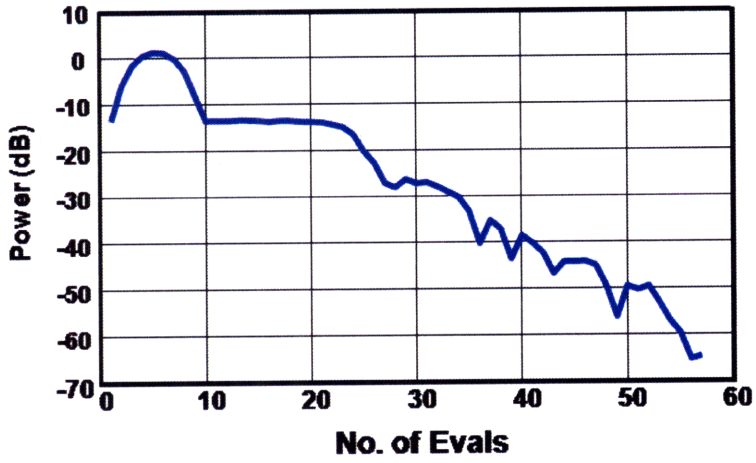

Figure 24. Narrowband Cancellation Progress

5.2.3 LIMITATIONS

The main limitations of this method came from noise on the spectrum analyzer and signal drift from a deteriorating phase lock between the two signal generators. There exists a trade-off between longer sweep time on the spectrum analyzer, which would increase the accuracy of the algorithm, and the fact that the algorithm should converge quickly to a solution and that the signal was drifting with time. A solution to the signal drift problem was remedied by phase locking the signal generators together as was done in Section 5.5.

5.3 ANTENNA DEMONSTRATIONS

For the next step of the project, it was important to see how an unpredictable antenna channel would affect the effectiveness of the algorithm. To test this, two omni-directional antenna with RF adapters were used in an office environment. The antennas were similar to an external antenna designed to radiate wireless signals for cell phones. Native isolation between the two antennas of around 30 dB helped begin the cancellation process. Additional processing was performed by adding in the cancellation signal; isolation levels were achieved similar to the closed-loop model of reaching 50-55 dB (including the 30 dB of native isolation) across a 500 MHz spectrum.

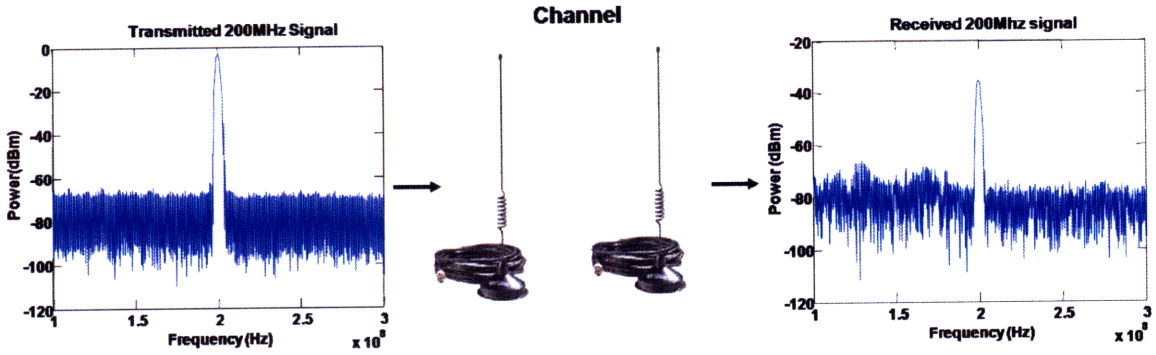

Figure 25. Antenna Narrowband Demonstration Results

In the initial set-up the two signal generators were not phase-locked and so again, the results were affected by the drifting phase on the signals coming from the antenna.

5.4 LOADING AND EXTRACTING USER-DEFINED WAVEFORMS

Once the narrowband cancellation method was shown to work with the lab equipment, the wideband cancellation method required implementation of the arbitrary waveform capability of

the signal generators. The Agilent E8267D and E4438C arbitrary signal generators play externally created waveform data using IQ Modulation.

5.4.1 IQ MODULATION

IQ Modulation is a modulation technique that efficiently transfers information and works very well with digital data. The "I" is the in-phase part of the waveform and the "Q" is the quadrature component. The concept behind IQ Modulation is that by modulating a carrier frequency with a waveform, it changes the carrier frequency in a way that allows you to treat the modulating signal like a phasor (22).

To provide background for signal modulation, a sine wave can be represented as:

$$A_c \cos(2\pi f_c t + \emptyset). \tag{5-1}$$

In equation (5-1), A_c is the amplitude of the carrier frequency, f_c is the frequency, and φ is the phase offset. Frequency is the rate of change of the sine wave, and so together f_c and φ make up the phase angle. In a polar representation, the instantaneous state of a signal captured by its magnitude and phase angle can be represented as a phasor as in Figure 26 below. By trigonometry the same phasor can be translated to a Cartesian plane with (x, y) coordinates as seen in Figure 26. On a Cartesian coordinate system, the phasor's projection on Real axis represents the "I" or in-phase part, while the Imaginary axis projection represents the "Q" or quadrature component.

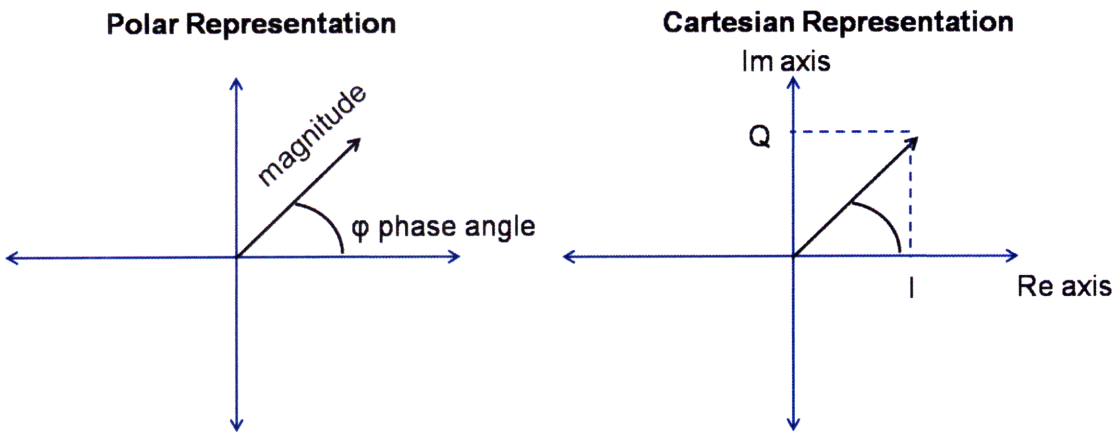

Figure 26. Polar coordinate and Cartesian coordinate representation of a sine wave

In the Cartesian Representation of the sine wave in Figure 26, the in-phase and quadrature parts of the signal with time are:

$$I(t) = M(t)\cos\varphi(t) \tag{5-2}$$

$$Q(t) = M(t)\sin\varphi(t). \tag{5-3}$$

The purpose to modulation is to change the amplitude, frequency, and/or phase of a carrier sine wave with time to transmit data. Because circuitry that changes the phase of a high-frequency carrier signal is difficult to implement in hardware, I and Q modulation is used to manipulate the phase of an RF carrier (23) using the trig identity in equation 5-4.

$$A\cos(2\pi f_c t + \varphi) = A\cos(2\pi f_c t)\cos(\varphi) - A\sin(2\pi f_c t)\sin\varphi \tag{5-4}$$

By taking equations (5-2) and (5-3) and manipulating them as:

$$I = A\cos\varphi \tag{5-5}$$

$$Q = A\sin\varphi \tag{5-6}$$

then by plugging (5-5) and (5-6) into equation (5-4) we get:

$$A\cos(2\pi f_c t + \varphi) = I\cos(2\pi f_c t) - Q\sin(2\pi f_c t). \tag{5-7}$$

In (5-7), I is the amplitude of the "in-phase carrier" $\cos(2\pi f_c t)$ and Q is the amplitude of the "quadrature-phase carrier" a 90° out of phase shifted copy of $\cos(2\pi f_c t)$. The hardware solution for an IQ modulator is shown in Figure 27 below.

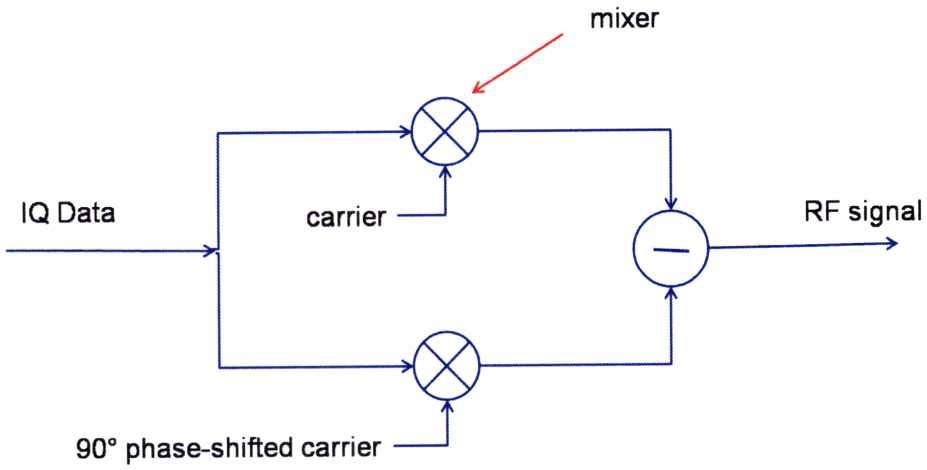

Figure 27. Hardware Diagram of an IQ modulator

In the hardware diagram, the mixers perform frequency multiplication mixing the I waveform with the carrier frequency and the Q waveform with a 90° offset of the same carrier frequency. The two signals are subtracted to create a hardware representation of the trigonometric equation (5-7) that allowed for phase modulation of a signal. This IQ modulation system allows for the I and the Q data to represent the changing magnitude and phase of any signal. The Arbitrary waveform generators can apply IQ modulation to any carrier frequency and output the modulated signal.

Figure 28 describes how the Agilent signal generators take an input waveform on the left and converts it into an RF output (22). It begins by sending the original signal through an analog-to-digital converter, and then processes and encodes the digital signal to get I and Q data. It runs the I and Q data through a filter and onto a mixer where it gets multiplied to an intermediate frequency and then to RF. The final RF signal is sent through a band-pass filter and a gain stage before being transmitted through the RF output port on the signal generators.

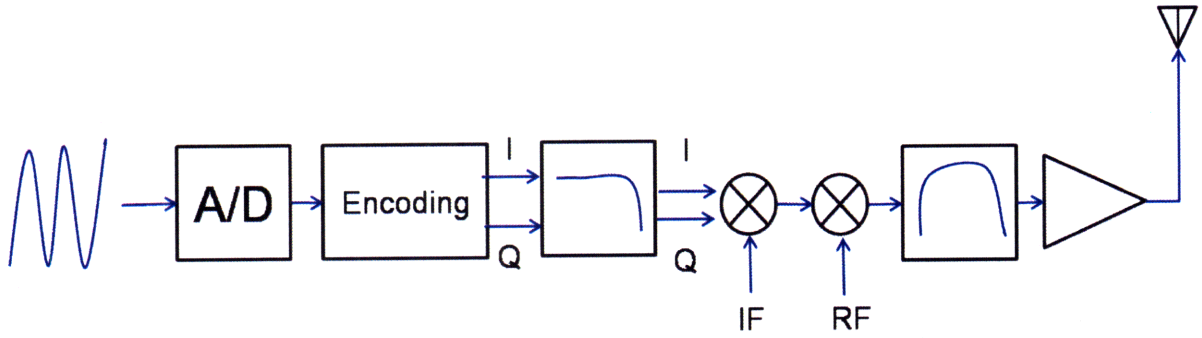

Figure 28. Agilent IQ Transmitter

The spectrum analyzer locks onto that carrier frequency and can, by treating the original signal as a phasor, read back the original I and Q data.

5.4.2 WAVEFORM REQUIREMENTS

For the thesis project, it was necessary to implement the arbitrary waveform functionality on the signal generators. The arbitrary waveform generators required three different pieces of information: the IQ data points, a file header, and a marker file. The IQ data points generated by the user needed to be formatted as 2-byte integer values in signed 2s complement falling in the range of −32768 to 32767 so that they fell in the range of the arbitrary waveform generator's 16-bit DACs. The data file required interleaving of the I and Q data and could be up to 320 MB long. The arbitrary waveform generator took the sample points and used an interpolation

algorithm to produce an RF waveform signal. An issue that arose from the DAC was the over-range problem where the interpolation algorithm sometimes causes an overshoot of the DACs dynamic range that result in an error. The DAC over-range problem becomes an issue because the interpolation algorithm works to create an analog waveform by running at 4 times the sample clock rate to smooth out the waveform given by the digital data. If the data were to exceed the DAC's dynamic range, then an overshoot was introduced into the generated analog waveform introducing a ripple or ringing effect at the peaks that exceed the range. To prevent over-range from becoming an issue, it was important to suitably scale the IQ data points. Scaling reduces the amplitude of the desired signal without otherwise distorting the shape of the desired waveform.

The file header informed the waveform generator of how to play the waveform. It gave information such as the sample rate of the IQ data. Marker files became very important in resolving some of the hardware issues that were presented by the project. A marker file used one byte of information for every waveform point in the I/Q Data file. The marker file set the four different available markers to either on (1) or off (0). The state of the markers sent out signals to the different rear panel connectors of the signal generators. Marker 1, for example, when in an on state sent an output trigger signal to the EVENT 1 connector. This functionality was used to trigger the start of the waveform playback on the two arbitrary waveform generators.

5.5 HARDWARE ISSUES

Before beginning the implementation, several hardware issues had to be resolved. One of these was the issue of phase discontinuities. With user-generated waveform data, errors in the data can arise because of the manner in which the waveforms are repeated. Upon reaching the end of the waveform sample, the signal generator repeats the waveform again from the start. If

the last sample in the waveform data does not match the beginning of the next cycle, then it leads to periodic spectral re-growth and distortion. The way to avoid phase discontinuities was to ensure that each periodic waveform segment uploaded to the signal generator simulates an integer number of cycles.

The second big issue to solve before the hardware solution could be implemented had to do with repeatable triggering of both arbitrary signal generators to ensure that waveform playback was synchronized. First, a common clock source is necessary. To achieve this with the equipment available, one arbitrary signal generator was designated as the master and the second arbitrary signal generator was designated as the slave. The master signal generator ran on its own internal 10 MHz clock, while the slave signal generator based its clock off the input it received in its Baseband GEN CLK IN connector in the rear panel. The 10 MHz output from the master signal generator was tied to this Baseband GEN CLK IN as in Figure 29.

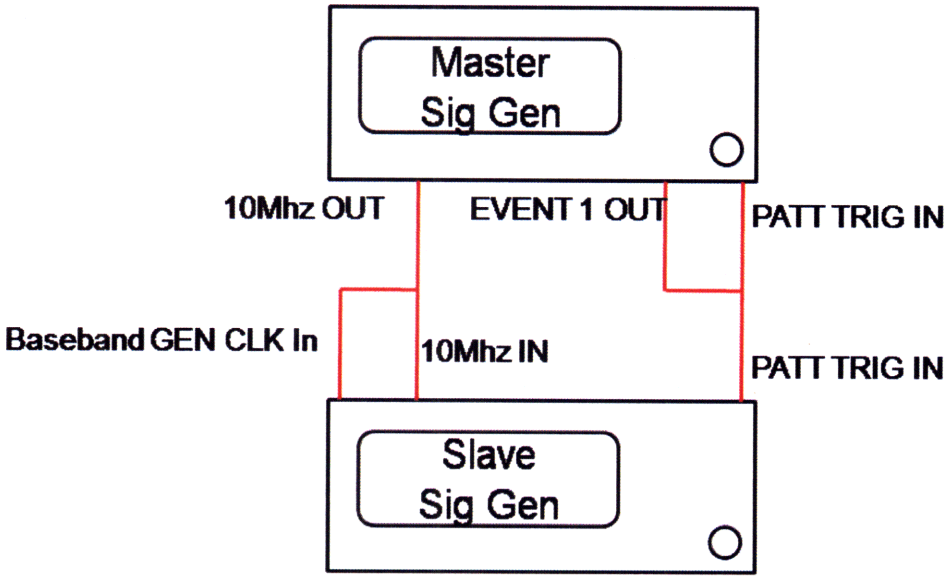

Figure 29. Master-Slave Connection Set-up

Once the two arbitrary signal generators shared a common clock source they are able to play the waveforms with the same sample rate. In addition to the clock, the two arbitrary signal generators needed to share a common trigger to start their waveform playback at the same time. To achieve this, the Marker File, described in Section 5.4.2, was used. The first byte in the Marker 1 sequence served as a flag by turning it on to correspond with the first sample point of the waveform. This flag triggered each signal generator through a connection from their Pattern Trigger Input to the EVENT 1 output on the Master Signal Generator. This signal synchronized the start of the waveform playback on each arbitrary signal generator.

With the phase lock in place and a repeatable trigger source, the signals were sent and displayed on the Agilent 54832 Oscilloscope. The initial tests were done using a user-generated narrowband waveform signal consisting of a single frequency. Each waveform was made by user-defined sample points of IQ Data with an interpolation algorithm generating the RF output. The important thing to note in Figure 30 below is the fact that the signals were phase locked together and that both arbitrary signal generators were triggering off the pulse from the EVENT 1 Connector. The results show that the cancellation was successful with an error signal in the plot of only a few millivolts.

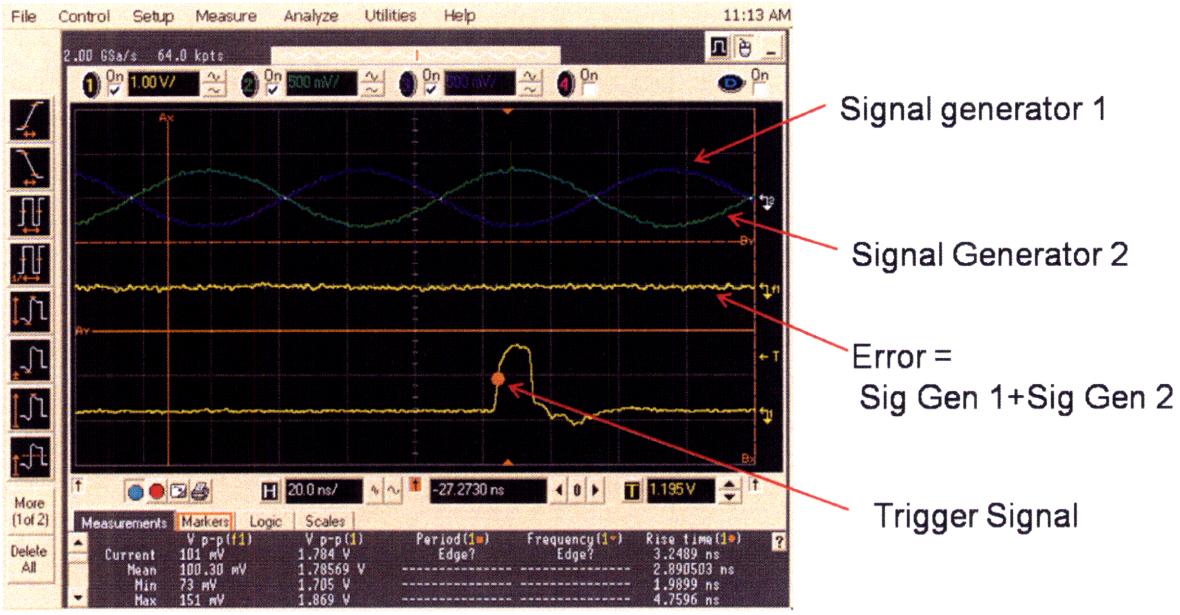

Figure 30. Narrowband Oscilloscope Results

5.6 FINAL HARDWARE SET-UP

In the final hardware set-up, the first signal generator was connected to the power coupler where it was summed to the second canceller signal generator. The sum output was received by the spectrum analyzer and the results of the cancellation can be seen in the following section. A diagram of the hardware set-up appears in Figure 31 below.

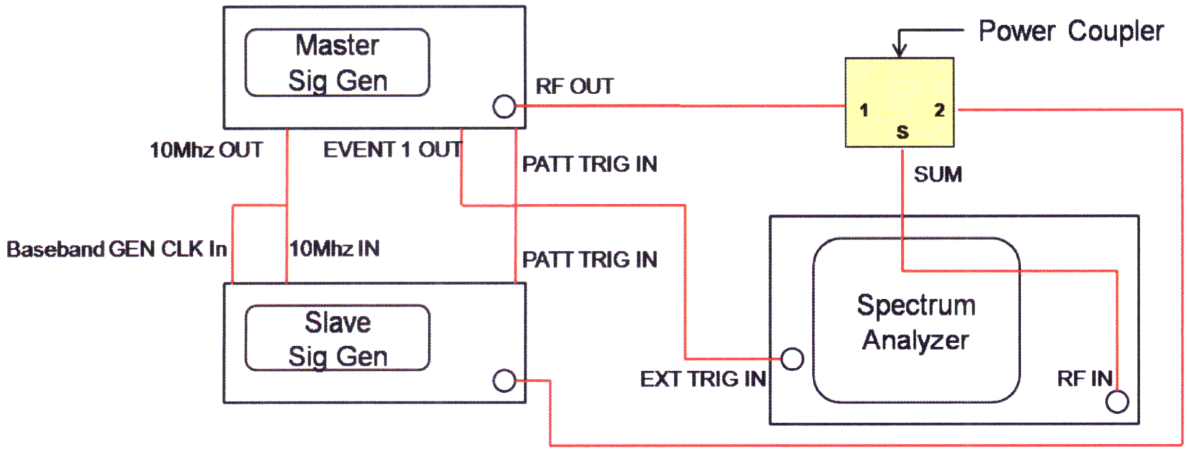

Figure 31. Final Cancellation Set-up

The spectrum analyzer had its external trigger input tied to the EVENT 1 connector so that collection of data on the analysis bandwidth of the spectrum analyzer was triggered with the start of the waveform playback. This allowed for time-domain analysis of the received signal to be synchronized with the user-generated waveforms.

One of the limitations revealed in the hardware set-up described in Figure 31 came from an intrinsic time delay due to the synchronization of the two signal generators. Since the setup used the master-slave configuration rather than an external timing source for the baseband generator clocks, a time skew was introduced due to the time it took the triggers to travel the length of the cables from the master to the slave. So even after the two signal generators were phase locked and triggered together, a gradient search had to be run on the final output to the spectrum analyzer in order to determine and adjust for the time skew in the signals. With a gradient search, then it was possible to correctly delay the user-defined waveforms on the signal generators by simply taking different samples of the IQ data to get maximum cancellation of the summed RF signal being input into the spectrum analyzer.

6 RESULTS

6.1 NARROWBAND CANCELLATION RESULTS

The ultimate goal of the project was to achieve receiver isolation from a broadband jammer by summing the received signal with a generated cancelling signal. The first step to achieving this cancellation was to address a narrowband interference signal. To demonstrate the feasibility of the approach, a transmitted signal and a canceller signal were generated with two arbitrary signal generators as described in Section 5. The transmitted signal for the narrowband demonstration consisted of a single tone (6-1).

$$IQData = .5 \times e^{i2\pi ft} \tag{6-1}$$

The emitted IQ Data signal had a frequency of 5 MHz and was modulated with a carrier frequency of 10 MHz. This lower frequency was used in conjunction with an increased sampling rate of 100 MHz in order to allow for fine grain control during testing for aligning the signals in order to compensate for the time skew, described in Section 5.6, introduced by the master-slave set-up. As the time domain results shown in Figure 30 demonstrate, the emitted signal was a simple sine wave and could be cancelled with a 180° phase shifted copy of itself with matching amplitude. The two signals when summed together resulted in an error signal that was as close to zero as possible. The frequency domain results (Figure 32) for the final cancellation demonstrated a drop in the power from the transmitted signal received by the receiver of 35 dB. The 35 dB of active cancellation when added to the approximate native isolation of 30 dB was sufficient to meet the desired total isolation of 60 dB which would effectively prevent receiver saturation and allow for successful simultaneous transmit and receive capabilities.

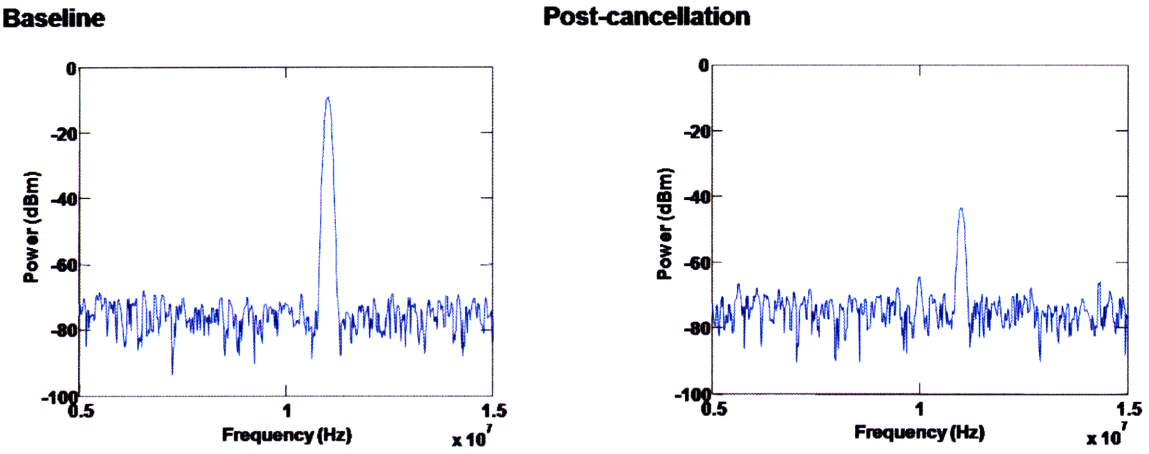

Figure 32. Arbitrary Waveform Generator Narrowband Results

One of the limitations of the final approach was the fact that the master-slave set-up introduced the time skew in the triggering of the slave arbitrary waveform playback. Time skew could also be introduced by the different signal paths traveled by the I and the Q signals inside the arbitrary waveform generators. The introduction of phase skew or I/Q path delay resulted in an increase in the error vector magnitude. This issue was resolved with adjustment of the samples taken of the IQ Data to compensate for the introduction of delays and to minimize the cancellation error.

Another limitation to the set-up was the unavoidable hardware issue involved with generating I/Q Data. The Digital-to-Analog converters on the signal generators often became misaligned along with the offset, gain balance, and quadrature error of the I/Q signals. The Agilent signal generators provided an Align DAC routine and an IQ Calibration routine that would remedy the problem and optimize IQ performance that had degraded over time.

6.2 WIDEBAND CANCELLATION RESULTS

Broadband jamming signifies that a signal consisting of multiple frequencies must be cancelled. The transmitted signal used for demonstration purposes involved multiple tones as seen in equation (6-2).

$$IQData = .5 \times [e^{i2\pi f_1 t} + e^{i2\pi f_2 t} + e^{i2\pi f_3 t}] \quad (6\text{-}2)$$

The frequencies used were 1 MHz, 3 MHz, and 5 MHz modulated with a 10 MHz carrier. This transmitted signal was produced by signal generator 1 and appears in time domain as seen in Figure 33.

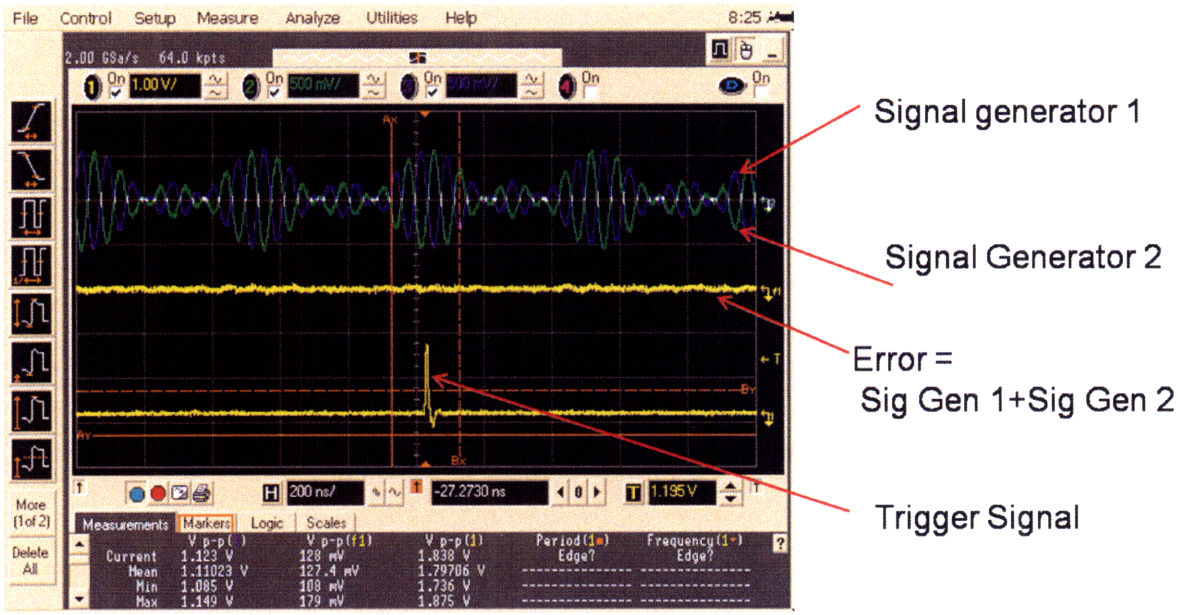

Figure 33. Multitone Cancellation: Time Domain

The cancellation signal was created by taking the transmitted signal, reading it and producing an inverse of it to create the cancellation effect when the two signals are added by the power coupler. The inverted signal was produced by the second canceller signal generator and can also be seen in Figure 33. The oscilloscope output shows how the two user-generated

waveforms both trigger of the EVENT 1 trigger signal produced by the Master signal generator's Marker file. The oscilloscope output also reveals the cancellation error to be close to the noise floor as evidenced by Figure 34, a zoomed in look at the sum of the two RF signals generated by the Arbitrary Waveform generator.

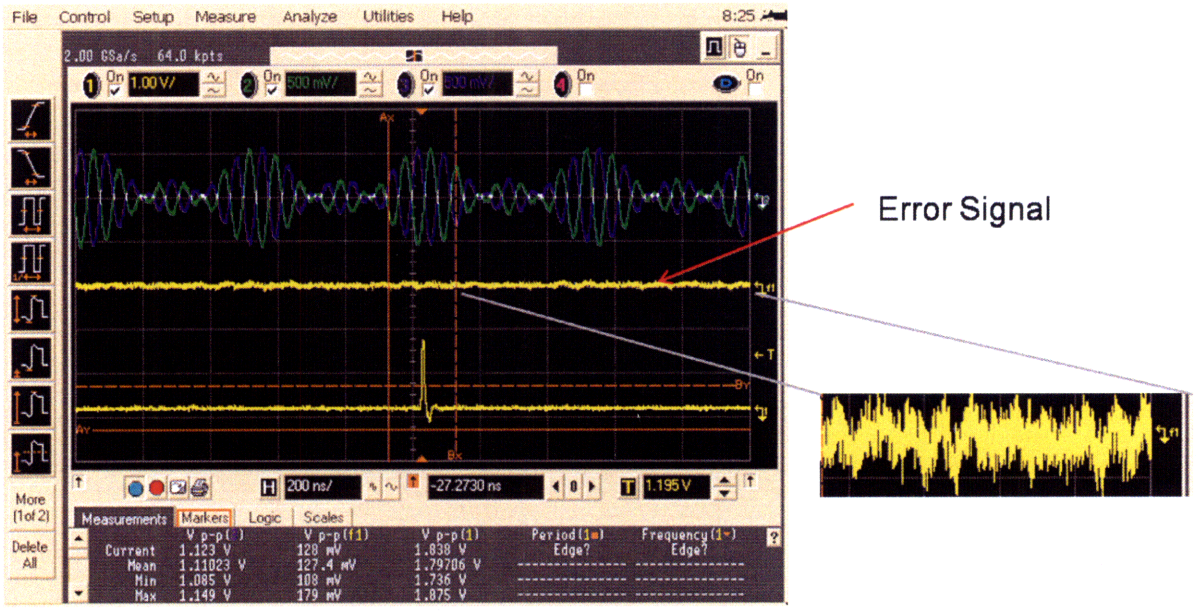

Figure 34. Error Signal magnification on Multitone Cancellation

The frequency domain power levels were the results that truly demonstrated the effectiveness of the isolation approach. As the spectrum analyzer showed, the original received power coming from the transmitted signal in the closed-loop model showed the three frequency peaks at around -15 dB. Post-cancellation, the effective transmit power received was decreased by 30-35 dB across the multiple frequencies. Final isolation levels reached between the transmit and the receive side of the system were at 45 to 50 dBm. Considering that the system tested was a closed-loop demonstration without antenna, introduction of native isolation between two antenna will only help the total reduction in coupling between the transmit and receive sides of the system.

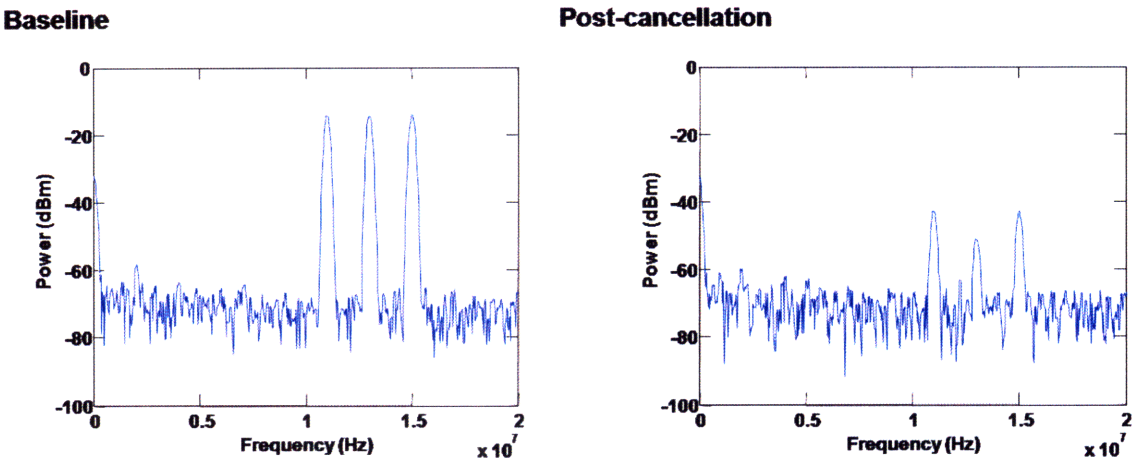

Figure 35. Arbitrary Waveform Generator: Multitone Results

7 SUMMARY AND CONCLUSION

The project started out with a vision of building a dual antenna system with a broadband jammer while preserving the capabilities of the receiver in that system across that broadband. The set-up of the application required simultaneous transmit and receive from the two closely-coupled antennas. To achieve this goal without receiver saturation, active cancellation of 30 dB was necessary.

The steps that were undertaken in this project began with the development of a cancellation system for a narrowband interference signal. The narrowband approach worked off a fast-converging nulling algorithm and achieved isolation levels of 60 dBm. The next step in the project was to achieve the same kind of cancellation across a wide range of frequencies. For a broadband signal, the design approach involved a linear least-mean-square algorithm able to create a filter to generate a canceller signal. The feasibility of this approach was proven with antenna-coupling data. The hardware implementation revealed limitations in the original design, and so the design approach was modified to a two Direct Digital Synthesis (DDS) model. In the resulting model, one DDS generated the jamming signal and the other generated an inverse of that signal designed to cancel out the first signal at the receiver side of the system. A proof-of-concept demonstration of the final design approach was achieved in hardware at the laboratory with two spectrum analyzers to generate the transmit signal and the cancel signals as well as a spectrum analyzer to serve as the receiver. With the new hardware set-up, the results for the original narrowband cancellation approach were repeated and the new approach involving arbitrary user-defined waveforms was implemented and applied to both singletone and multitone signals.

With wideband signals, the system achieved 30-35 dB of active cancellation. The next steps are to make improvements on the cancellation algorithm that adjusts for the time-delay in the waveform playback. Open-loop antenna tests are the next big step as they will demonstrate how native isolation adds to the decrease in antenna coupling while distorting the channel between the transmit and receive antennas. The channel effects can still be compensated for with the system set-up by capturing the received signal's I and Q data from the spectrum analyzer's analysis bandwidth. The final steps for this work are to get the cancellation system out of the lab and off test equipment and onto a single chip by implementing it with two DDS chips and a DSP board to perform the processing on the signals to maximize cancellation.

The main contribution and the final take-away of the thesis project was to show that the transmit and receive antenna system with a broadband jammer can have co-polarized and closely-coupled antenna and still achieve at least 30 dB of active cancellation across a wide range of frequencies. This cancellation will have the effect of preventing receiver saturation and preserving the functionality of the system without compromising the jamming power levels or the receive capability for target signal detection.

A. APPENDIX: SOURCE CODE

A.1 Narrowband Cancellation

```
% Adaptive Nulling with accelerated convergence
function [store_power,test_alphaDB_out,test_phi_out,alphaLIST,phiLIST] = ...
    adapt_nullDB(native_isolation,phaseANG,noise_sigma,varargin)

zim = sqrt(-1);

% model
%
%                      |--a1---< Tx           Rx >-----
%                      |                               |
% [signal (a0)] ======>|                               |
%                      |                               |
%                      |--a1-[phase (phi)]-----|       |
%                                              |       |
%                           |=a2===[alpha]=====|       |
%                      a4 ==|                           |
%                           |=a3=======================|
%
% ai are amplitudes, alpha is attenuation factor, phi is phase shift
% a2 = a1*alpha*exp(i*phi)

% initial guesses
if nargin==5
  alphaS0 = varargin{1};
  phiS0 = varargin{2};
  flagS0 = 1;
  DISPon = 0;
elseif nargin==4
  DISPon = varargin{1};
  flagS0 = 0;
else
  flagS0 = 0;
  DISPon = 0;
end


% input amplitudes
a0 = 1;
a1 = 1;
a3 = a1*undb(-native_isolation,'voltage')*exp(zim*phaseANG*(pi/180));

% quantization parameters
quantizer = inline('round(x/q)*q','x','q');
phi_quanta = 360/(2^10-1);
alphaDB_quanta = 32/(2^10-1);
```

```
% step sizes (linear amplitude and degrees)
dA0 = alphaDB_quanta;
dphi0 = phi_quanta;

% Parameter settings
Jmax = 20;        % Max iterations
Kmax = 7;         % Max sub-iterations

% coarse attentuation factor
coarse_alpha = undb(-15,'voltage');


% Threshold to achieve (includes native isolation)
Pthreshold = -65;

fun1 = inline('x');

if (DISPon)
  % range of phase and amplitude to display
  alpha = geomspace(undb(-50,'voltage'),undb(-10,'voltage'));
  phi = 0:0.7:360;
  [alphaMAT,phiMAT] = meshgrid(alpha,phi);
  nullMAT = nullFun(noise_sigma,a1,a3,1,alphaMAT,phiMAT,'cont');

  figure(1);clf
  pcolor(db(alphaMAT,'voltage'),phiMAT,nullMAT);
  xlabel('Power (dB)');
  ylabel('Phase (deg)');
  caxis([-70 0]);colorbar;
  colorbar;
  shading flat
end

% gradient search algorithm prototype:

% Initialize power eval counter
nevals = 0;
store_power = [];
alphaLIST = [];
phiLIST = [];

if ~flagS0
  % Set initial guess for attenuation (default none)
  start_alphaDB = 0-alphaDB_quanta;

  % Determine initial guess for phase angle
  phi0 = quantizer(0:40:320,phi_quanta);
  Ptest0 =
fun1(nullFun(noise_sigma,a1,a3,coarse_alpha,undb(start_alphaDB,'voltage'),phi
0));

  store_power = [store_power Ptest0];
  alphaLIST = [alphaLIST repmat(start_alphaDB,size(Ptest0))];
```

```
  phiLIST = [phiLIST phi0];
  nevals = nevals + length(phi0);     % count evaluations

  % sort powers to determine best phi to start with
  [Ptest0min,ll] = min(Ptest0);
  start_phi = phi0(ll);

else

  start_alphaDB = alphaS0;
  start_phi = phiS0;
end


% First round
jj = 1;
test_alphaDB(jj) = start_alphaDB;
test_phi(jj) = start_phi;
Ptest(jj) =
fun1(nullFun(noise_sigma,a1,a3,coarse_alpha,undb(test_alphaDB(jj),'voltage'),
test_phi(jj)));
store_power(nevals+1) = Ptest(1);
alphaLIST(nevals+1) = test_alphaDB(jj);
phiLIST(nevals+1) = test_phi(jj);
nevals = nevals + 1;  % count evaluations

while ((jj<=Jmax)*(Ptest(jj)>Pthreshold))
  dA = dA0;
  dphi = dphi0;

  % calculate change in P with respect to dA
  alpha2 = undb(min(0,test_alphaDB(jj)+dA),'voltage');
  phi2 = test_phi(jj);
  alpha1 = undb(min(0,test_alphaDB(jj)-dA),'voltage');
  phi1 = test_phi(jj);
  power1 = fun1(nullFun(noise_sigma,a1,a3,coarse_alpha,alpha1,phi1));
  power2 = fun1(nullFun(noise_sigma,a1,a3,coarse_alpha,alpha2,phi2));
  dPAj = power2-power1;

  store_power(nevals+1) = power1;
  alphaLIST(nevals+1) = test_alphaDB(jj)-dA;
  phiLIST(nevals+1) = phi1;

  store_power(nevals+2) = power2;
  alphaLIST(nevals+2) = test_alphaDB(jj)+dA;;
  phiLIST(nevals+2) = phi2;

  nevals = nevals + 2;  %count evaluations

  % calculate change in P with respect to phi
  alpha2 = undb(test_alphaDB(jj),'voltage');
  phi2 = test_phi(jj)+dphi;
  alpha1 = undb(test_alphaDB(jj),'voltage');
  phi1 = test_phi(jj)-dphi;
  power1 = fun1(nullFun(noise_sigma,a1,a3,coarse_alpha,alpha1,phi1));
```

```
  power2 = fun1(nullFun(noise_sigma,a1,a3,coarse_alpha,alpha2,phi2));
  dPpj = power2-power1;
  %phitemp = linspace(150,170,1000);
  %for ii = 1:length(phitemp),
  %    powertemp(ii) =
fun1(nullFun(noise_sigma,a1,a3,coarse_alpha,alpha1,phitemp(ii)));
  %end

  store_power(nevals+1) = power1;
  alphaLIST(nevals+1) = test_alphaDB(jj);
  phiLIST(nevals+1) = phi1;

  store_power(nevals+2) = power2;
  alphaLIST(nevals+2) = test_alphaDB(jj);
  phiLIST(nevals+2) = phi2;
  nevals = nevals + 2;  % count evaluations

  % update weights
  rAj = -(dPAj/2/dA) / sqrt((dPAj/2/dA)^2 + (dPpj/2/dphi)^2);
  rpj = -(dPpj/2/dphi) / sqrt((dPAj/2/dA)^2 + (dPpj/2/dphi)^2);

  if isnan(rpj)
      keyboard
  end

  % update weights
  %rAj = -(dPAj/2/dA);
  %rpj = -(dPpj/2/dphi);

  % being sub-iterations
  k = 1;flag = 1;
  Pold = Ptest(jj);
  while (k<Kmax)*flag
    if (Pold>-40)
      phiPOWER = 2.0;
      alphaPOWER = 2.0;
    elseif (Pold>-50)
      phiPOWER = 2.0;
      alphaPOWER = 1.7;
    elseif (Pold>-55)
      phiPOWER = 1.7;
      alphaPOWER = 1.5;
    elseif (Pold>-60)
      phiPOWER = 1.5;
      alphaPOWER = 1.3;
    elseif (Pold>-65)
      phiPOWER = 1.3;
      alphaPOWER = 1.1;
    else
      phiPOWER = 1.0;
      alphaPOWER = 1.0;
    end
    %phiPOWER = 2;
    %alphaPOWER = 1.6;
```

```
    phiK = quantizer(test_phi(jj)+dphi*rpj*(phiPOWER^(k-1)),phi_quanta);
    %fprintf('test_phi[%d] is %g for k=%g and phiK is %g \n', jj,
test_phi(jj),k, phiK);
    alphaDBK = min(0,quantizer(test_alphaDB(jj)+dA*rAj*(alphaPOWER^(k-
1)),alphaDB_quanta));
    % debug
    debug= quantizer(test_alphaDB(jj) + dA*rAj*(alphaPOWER^(k-1)),
alphaDB_quanta);
    %
    PtestK =
fun1(nullFun(noise_sigma,a1,a3,coarse_alpha,undb(alphaDBK,'voltage'),phiK));
    store_power(nevals+1) = PtestK;
    alphaLIST(nevals+1) = alphaDBK;
    phiLIST(nevals+1) = phiK;
    nevals = nevals + 1;    % count evaluations
    if (PtestK>Pold)
      flag = 0;
      phiK = quantizer(test_phi(jj)+dphi*rpj*(phiPOWER^(k-2)),phi_quanta);
      alphaDBK = min(0,quantizer(test_alphaDB(jj)+dA*rAj*(alphaPOWER^(k-
2)),alphaDB_quanta));
      PtestK =
fun1(nullFun(noise_sigma,a1,a3,coarse_alpha,undb(alphaDBK,'voltage'),phiK));

      store_power(nevals+1) = PtestK;
      alphaLIST(nevals+1) = alphaDBK;
      phiLIST(nevals+1) = phiK;
      nevals = nevals + 1;    % count evaluations

    else
      Pold = PtestK;
      k = k + 1;
    end
  end

  jj = jj + 1;
  test_alphaDB(jj) = alphaDBK;
  test_phi(jj) = phiK;
  Ptest(jj) = PtestK;
  nevalsT(jj) = nevals;

end

test_phi = mod(test_phi,360);
phiLIST = mod(phiLIST,360);

if (DISPon)
  hold on;
  plot(db(coarse_alpha,'voltage')+test_alphaDB,mod(test_phi,360),'wo-');
  %plot(db(coarse_alpha,'voltage')+alphaLIST,phiLIST,'wo-');

  [test_alphaDB' test_phi' Ptest']
  nevals

  fprintf('Native isolation = %g dB\n',native_isolation);
  fprintf('Native phase = %g deg\n',phaseANG);
```

```
  fprintf('Coarse attenuation (dB) = %g\n',db(coarse_alpha,'voltage'));
  fprintf('Attenuator setting (dB) = %d x %g = %g\n', ...
      test_alphaDB(end)/alphaDB_quanta,alphaDB_quanta,test_alphaDB(end));
  fprintf('Phase setting = %g x %g = %g\n',test_phi(end)/phi_quanta, ...
      phi_quanta,test_phi(end));
  fprintf('Achieved isolation = %g, %g\n', Ptest(end), ...

nullFun(noise_sigma,a1,a3,coarse_alpha,undb(test_alphaDB(end),'voltage'),test
_phi(end)));
  fprintf('Number power evaluations = %d\n',nevals);

  orient landscape

  joe = nullFun(noise_sigma, a1, a3, coarse_alpha, undb(-10.0411, 'voltage'),
179.824);
  fprintf('output is %f\n', joe);
end



test_alphaDB_out = test_alphaDB(end);
test_phi_out = test_phi(end);
```

A.2 Null Function

```
function out = nullFun(noise_sigma,a1,a3,coarse_alpha,alpha,phi,varargin);

% outputs power in dBm;

if nargin==7
  outtype = varargin{1};
else
  outtype = 'discrete_logamp';
end

if ~strcmp(outtype,'cont')
  % max/min of alpha
  alpha(find(alpha>1)) = 1;
  alpha(find(alpha<undb(-32,'voltage'))) = undb(-32,'voltage');
  alpha = coarse_alpha*alpha;
else
  alpha = coarse_alpha*alpha;
end

out = abs(a3+a1.*alpha.*exp(sqrt(-1).*phi*pi/180));

% noise in linear scale
out = out + noise_sigma*randn(size(out));

if strcmp(outtype,'cont');
  out = db(out,'voltage');
```

```
elseif strcmp(outtype,'discrete_logamp')
  voltage_quanta = 3.0/(2^11-1);
  out = logamp(db(out,'voltage'));
  out(find(out>3.0)) = 3.0;
  out = voltage_quanta*round(out/voltage_quanta);
  % reverse back to power scale
  out = logampINV(out);
else
  error('unknown outtype');
end
```

A.3 Test Adaptation

```
clear all

isolation = max(21,21 + rand(1)*10);
phaseANG = randn(1)*360;

isolation = 25;
phaseANG = 0;

noise_sigma = 0*undb(-75,'voltage');
[store_power,alpha0,phi0,alphaLIST,phiLIST] =
adapt_nullDB(isolation,phaseANG,noise_sigma,1);
[store_powerx,alpha0x,phi0x,alphaLISTx,phiLISTx] =
adapt_nullDB_unquantize(isolation, phaseANG, noise_sigma, 1);

dt = 10e-6; % clock for adaptive nuller
times = (1:length(store_power))*dt/1e-3;

maxT = 0.1*ceil(max(times)/0.1);

figure(2);clf;
subplot(311);
plot(times,alphaLIST,'.-')
ylabel('Attenuation (dB)');
grid on;
axis([0 maxT -32 0]);

subplot(312);
plot(times,phiLIST,'.-')
ylabel('Phase Angle (deg)');
grid on;
axis([0 maxT 0 360]);

subplot(313);
plot(times,store_power,'.-')
xlabel('Time (ms)');
ylabel('Achieved Null (dB)');
grid on;
axis([0 maxT -80 0]);
```

```
%% Energy Integral Calculations %%

A = alphaLISTx(end);  %attenuation without quantization
A_hat = alphaLISTx(end);    %take off the 'x' for attenuation with rounding
error
w = 5*10^6;
delta_w = 0.1*10^6;
T = 2*pi/w;
PHI = phiLISTx(end);    % phase without quantization
PHI_hat = phiLISTx(end); %phase in degrees *take off the 'x' for rounding
error

%For delay line phase shifter
c = 299792458; % meters/second speed of light
l_default = 100; % line length in meters for delay line phase shifter phi =
k*l, k = w/c
PHIk_hat = inline('w*l/299792458', 'w', 'l');

E_0=(1/2)*(abs(A*exp(sqrt(-1)*PHI) - A_hat*exp(sqrt(-1)*PHI_hat)))^2;    %w =
w_hat
fprintf('E at w = w_hat is %g  = %g dB\n', E_0, db(E_0, 'power'));
Elist = [w E_0];

syms t;
for w_count = -1:-1:-200
    w_hat = w + w_count*delta_w;
%    PHI_hat = PHIk_hat(w_hat, l_default);      % w/ delay line phase shifter
    SS = @(t)(A*cos(w*t + PHI) - A_hat*cos(w_hat*t + PHI_hat)).^2;
    E = (1/T)*quad(SS, 0, T);
  %  E = double(E);
    Elist = [w_hat E; Elist];
end

for w_count = 1:200
    w_hat = w + w_count*delta_w;
%    PHI_hat = PHIk_hat(w_hat, l_default);       % w/ delay line phase
shifter
    SS = @(t)(A*cos(w*t + PHI) - A_hat*cos(w_hat*t + PHI_hat)).^2;
    E = (1/T)*quad(SS, 0, T);
 %   E = double(E);
    Elist = [Elist;w_hat E];
end

EDBlist = [Elist(:,1), db(Elist(:,2), 'power')];
figure(3);
%plot(Elist(:,1) ,Elist(:,2));
plot(EDBlist(:,1) ,EDBlist(:,2));
xlabel('Fequency centered at 5Mhz');
ylabel('Energy of Summed Signal');
title('Energy of Summed Signal versus Frequency Deviations');
grid on;
```

A.4 Get Instrument Object

```
function [obj1] = getInstrObj(instrIP, instrPort)

% global DEF

% Find a tcpip object.
obj1 = instrfind('Type', 'tcpip', 'RemoteHost', instrIP, 'RemotePort',
instrPort, 'Tag', '');

if ~isempty(obj1)
    fclose(obj1);
    delete(obj1);
    clear 'obj1';
end

if isempty(instrIP)
    obj1 = [];
else
    instrIP = RmvLeadingZeros(instrIP);
    if ispc
        CmdStr = sprintf('ping -w 500 -n 2 %s', instrIP);
    else
        CmdStr = sprintf('ping -o -t 1 %s', instrIP);
    end;
    % LogEntry('%s\n', CmdStr);
    [Status Result] = system(CmdStr);
    % LogEntry('%s\n', Result);
    if Status ~= 0
        obj1 = [];
    else
        obj1 = tcpip(instrIP, instrPort);
        set(obj1,'InputBufferSize',500000);
        set(obj1,'OutputBufferSize',16000);

        % Connect to instrument object, obj1.
        fopen(obj1);
        %    end;
        flushinput(obj1);
        flushoutput(obj1);
    end;
end;

function IPaddr = RmvLeadingZeros(IPaddr)

sexpr = '\.0{1,2}([0-9]{1,2})';
rexpr = '\.$1';
IPaddr = regexprep(IPaddr, sexpr, rexpr);
```

A.5 Spectrum Analyzer Screen Capture

```
function Waveform = HanaSA_Capture(obj, SweepTime,SampleRate,CenterFrequency)
% To set object Spectrum Analyzer>>  SA = getInstrObj('155.34.109.225', 5025)
% *Note: set input buffersize to 8000000 or else errror: buffer overflow
% >> Sweep Time = 2ms, Sample Rate = 100e6 , Center Frequency = 10Mhz
```

```
%Waveform = HanaSA_Capture(SA, .002, 100e6, 10e6);
%I = Waveform.Amplitude(1:2:end);
%Q = Waveform.Amplitude(2:2:end);
%t = Waveform.Time(1:2:end);
%w = complex(I,Q);
%pwelch(w,[],[],[],40e6)
%To get 10,000 samples at 40e6 Mhz then take
%I = I(1:10000);
%Q = Q(1:10000);
%t = t(1:10000);
%combine signals with
%Test = complex(I, Q);

SweepTimeStr = sprintf('%e', SweepTime);
SampleRateStr = sprintf('%e', SampleRate);
CenterFrequencyStr = sprintf('%e', CenterFrequency);
PrecisionStrSA = 'REAL,32';
PrecisionStrML = 'float32';
ATT = '20';
    Waveform.Amplitude = NaN;
    Waveform.Time = NaN;
    Waveform.SweepTime = SweepTime;
    Waveform.SampleRate = SampleRate;
    Waveform.CenterFrequency = CenterFrequency;
    fprintf('>>> check valid\n');
        if isInstrumentValid(obj)
            flushinput(obj);
            flushoutput(obj);
            fprintf('>>> is valid\n');
            StartStr1 = [ ...
                '*CLS;' ...
                ':inst:sel BASIC;' ...
                '*RST;' ...
                ':sense:power:rf:att ' ATT ';' ...
                ':disp:enab 0;' ...
                ':wav:ncpt 1;' ...
                ':init:wav;' ...
                ':init:cont off;' ...
                ':status:operation:enable 25;' ...
                ':wav:ifp wide;' ...
                ':format:bord norm;' ...
                ':format ' PrecisionStrSA ';' ...
                ];

            fprintf(obj,'%s\n', StartStr1);

                fprintf(obj,':freq:cent %s\n', CenterFrequencyStr);
                [DataIn,count,msg] = query(obj, ':freq:cent?');
                fprintf('RESP> %d, %s, %s\n',count,msg,DataIn);
                fprintf(obj,':wav:swe:time %s\n', SweepTimeStr);
                [DataIn,count,msg] = query(obj, ':wav:swe:time?');
fprintf('RESP> %d, %s, %s\n',count,msg,DataIn);
                fprintf(obj,':wav:srat %s\n', SampleRateStr);
                [DataIn,count,msg] = query(obj, ':wav:srat?');
fprintf('RESP> %d, %s, %s\n',count,msg,DataIn);
```

```
                %fprintf(obj,':wav:trig:sour imm');  %fix this to trigger off
External Trigger Input?
                fprintf(obj,':wav:trig:sour ext');  %trigger off external
                fprintf(obj,':init:immediate');

        end;

        if isInstrumentValid(obj)
            fprintf('SA trace\n');
            set(obj,'Timeout', 30);
            DataIn = query(obj, '*OPC?');
            fprintf('SA done trace\n');
            fprintf(obj, ':calc:data0?');

           fprintf('read\n');
            [BufferW, count, msg] = binblockread(obj, PrecisionStrML);
             fprintf('Count = %d <%s>\n', count, msg);
           fprintf('SA done read\n');

            Waveform.Amplitude = BufferW;
            Waveform.Time = (0:length(BufferW)-1)./SampleRate;
            set(obj,'Timeout', 10);
        end;




%fprintf('DoneCapture\n');




%             FreqN = length(Buffer);
%             FreqInc = (FreqStop - FreqStart)/(FreqN-1);
%             Trace{k}.Frequency = (FreqStart:FreqInc:FreqStop)';
%             toc;
%         end; % if isInstrumentValid
%     end; % if k
% end; % for k
```

A.6 SPECTRUM ANALYZER SCREEN TRACE

```
function [Screen Trace] = HanaSA_ScreenTrace(obj)

%Modified Screen Shot of Spectrum Analyzer "obj";

Precision = 'float64';
PrecisionStr = 'REAL,64';

% SendStrClrAvg = [ ...
%     '*CLS;' ...
```

```
%      ':sense:average:clear;' ...
%      ':format ' PrecisionStr ';' ...
%      ':format:border NORM;' ...
%      ':init:cont OFF;' ...
%      ':init:cont ON;'];
%      ':init:immediate;' ...
%      '*WAI;'];
SendStrRequestData = [
    ':format ' PrecisionStr ';' ...
    ':mmemory:store:screen ''C:\TEMPSCR.GIF'';' ...
    ':trace:data? TRACE1;' ...
    ':mmemory:data? ''C:\TEMPSCR.GIF'';' ...
    ':sense:frequency:start?;' ...
    ':sense:frequency:stop?;' ...
    ];     % '*WAI;' ...

SendStr2 = [...
    '*CLS;' ...
    ':mmemory:data? ''C:\TEMPSCR.GIF'';' ...
    ];

SendStrEnd = [...
    ':mmemory:delete ''C:\TEMPSCR.GIF'';' ...
    '*CLS;' ...
    ];
%     ':init:cont ON;' ...

Trace.Amplitude = NaN;
Trace.Frequency = NaN;
Screen.X = NaN;
Screen.map = NaN;
if isInstrumentValid(obj);
    flushinput(obj);
    % fprintf(SA{k},'%s\n', SendStrClrAvg);
end;

pause(.1);

if isInstrumentValid(obj);
    fprintf(obj,'%s\n', SendStrRequestData);
end;

pause(.1);
if isInstrumentValid(obj);
    [BufferT, count, msg] = binblockread(obj, Precision);
%   VLogEntry('Count = %d <%s>\n', count, msg)
    Trace.Amplitude = BufferT;

    [BufferS, count, msg] = binblockread(obj, 'char');
    %VLogEntry('Count = %d <%s>\n', count, msg);

    [DataIn, count, msg] = fgetl(obj);
    %VLogEntry('Count = %d <%s> <%s>\n', count, DataIn, msg);

    FreqLim = sscanf(DataIn, ';%e',2);
```

```
    FreqStart = FreqLim(1);
    FreqStop = FreqLim(2);

%             [DataIn, count, msg] = fgets(SA{k});
%             VLogEntry('Count = %d <%s> <%s>\n', count, DataIn, msg);
%             FreqStop = sscanf(DataIn, '%e');

    FreqN = length(BufferT);
    FreqInc = (FreqStop - FreqStart)/(FreqN-1);
    Trace.Frequency = (FreqStart:FreqInc:FreqStop)';

    fprintf(obj,'%s\n', SendStrEnd);

    % Screen{k}.Raw = BufferS;
    pause(.1);
    GIFFilename = sprintf('Screen.gif');

    sf = fopen((GIFFilename), 'w');
    fwrite(sf, BufferS, 'char');
    fclose(sf);

    [X,map] = imread((GIFFilename), 'gif');
    Screen.X = X;
    Screen.map = map;
end; % if isInstrumentValid
```

A.7 NULLING SINGLETONE

```
% Nulling Sine Waves
clearInstrObj
close all
clear all
%% Set up connection with signal generator
SA = getInstrObj('155.34.109.225', 5025);
SG1 = getInstrObj('155.34.109.226', 5025);
SG2 = getInstrObj('155.34.109.227', 5025);
validity = isInstrumentValid(SA);
if(validity <1) return; end
validity = isInstrumentValid(SG1);
if(validity <1) return; end
validity = isInstrumentValid(SG2);
if(validity <1) return; end

%% Initialize Guess
freq = 350e6;
fprintf(SG1, 'OUTP OFF'); % turns SG on/off

%Set frequency and power of signal generator
fprintf(SG2, 'OUTP ON');
strcmd = sprintf('SOURce:FREQuency %d', freq);
fprintf(SG2, strcmd);                % sets frequency of original signal on
signal generator #2
fprintf(SG2, 'POWer 0');
```

```
%Read back peak from Spectrum Analyzer
strcmd = sprintf('FREQuency:CENT %d', freq);
fprintf(SA, strcmd);
fprintf(SA, 'CALC:MARK1:MAX');
fprintf(SA, 'CALC:MARK1:Y?');
peak_original = fscanf(SA);
peak_original = str2double(peak_original);

%Screen Shot
[Screen_original Trace_original] = HanaSA_ScreenTrace(SA);

%Generate initial guess for canceller amplitude (adjust for 3dB gain from
%splitter/coupler)
peakadj = peak_original + 3;
%fprintf(fsiggen2, 'OUTP OFF'); % turns SG on/off

%Set frequency and power of canceller signal generator to initial guess
strcmd = sprintf('SOURce:FREQuency %d', freq);
fprintf(SG1, strcmd);               % sets frequency of canceller signal on
signal generator #1
strcmd = sprintf('POWer %d', peakadj);
fprintf(SG1, strcmd);

fprintf(SG1, 'OUTP ON')
%% Adaptive Nulling

%step sizes (linear amplitude and degrees)
dalpha = .1;     %dB
dphi = 2;        %deg

%Parameter settings
Jmax = 35;       %max iterations (20)
Kmax = 10;        %max sub-iterations (7)

%Threshold to achieve (includes native isolation)
Pthreshold = -65;

% fun1 = inline('x');
DISPon = 0;
if (DISPon)
  % range of phase and amplitude to display
  alpha = geomspace(undb(-50,'voltage'),undb(-1,'voltage'));
  phi = 0:0.7:360;
  [alphaMAT,phiMAT] = meshgrid(alpha,phi);
%Fill in nullMAT
%     for i = 1:length(alphaMAT)
%         for j = 1:length(phiMAT)
%             strcmd = sprintf('SOURce:PHASe %d deg', phiMAT(j)); %adjust the
phase
%             fprintf(SG1, strcmd);
%
%             strcmd = sprintf('POWer %d', alphaMAT(i)); %adjust the
amplitude
%             fprintf(SG1, strcmd);
```

```
%
%             fprintf(SA, 'CALC:MARK1:MAX');      %check peak level
%             fprintf(SA, 'CALC:MARK1:Y?');
%             peakMAT(i, j) = str2double(fscanf(SA));
%         end
%     end
  noise_sigma = 0; native_isolation = 25; a1 = 1; a3 = a1*undb(-
native_isolation,'voltage')*exp(sqrt(-1)*0*(pi/180));
  nullMAT = nullFun(noise_sigma,a1,a3,1,alphaMAT,phiMAT,'cont');

  figure(1);clf
  pcolor(db(alphaMAT,'voltage'),phiMAT,nullMAT);
  xlabel('Power (dB)');
  ylabel('Phase (deg)');
  caxis([-70 0]);colorbar;
  colorbar;
  shading flat
end

%GRADIENT SEARCH ALGORITHM

%Initialize power eval counter
nevals = 0;
peakLIST = [];
alphaLIST = [];
phiLIST = [];

%Determine initial guess for phase angle
for phi0 = 0:40:360
    strcmd = sprintf('SOURce:PHASe %d deg', phi0);
    fprintf(SG1, strcmd);               % adjusts phase of canceller

    fprintf(SA, 'CALC:MARK1:MAX');      %check peak level
    fprintf(SA, 'CALC:MARK1:Y?');
    peak0 = str2double(fscanf(SA));
    peakLIST = [peakLIST peak0];     %store peak level

    fprintf(SG1, 'SOURce:POW?');        %check amplitude level
    alpha0 = str2double(fscanf(SG1));
    alphaLIST = [alphaLIST alpha0];     %store amplitude level

    phiLIST = [phiLIST, phi0];           %store phase
    nevals = nevals + 1;
end

% sort peak powers to determine best phi to start with
[peakmin, ll] = min(peakLIST);
start_phi = phiLIST(ll);
start_alpha = alphaLIST(ll);

% First round
jj = 1;
test_alpha(jj) = start_alpha;
test_phi(jj) = start_phi;
```

```
strcmd = sprintf('SOURce:PHASe %d deg', test_phi(jj));
fprintf(SG1, strcmd);                % adjusts phase of canceller

fprintf(SA, 'CALC:MARK1:MAX');       %check peak level
fprintf(SA, 'CALC:MARK1:Y?');
Ptest(jj) = str2double(fscanf(SA));
peakLIST(nevals + 1) = Ptest(jj);    %store peak level
alphaLIST(nevals + 1) = test_alpha(jj);
phiLIST(nevals + 1) = test_phi(jj);

while ((jj<= Jmax)*(peakLIST(jj) > Pthreshold))

    %calculate change in peak power with respect to dA
    alpha2 = test_alpha(jj) + dalpha;
    alpha1 = test_alpha(jj) - dalpha;
    phi2 = test_phi(jj);
    phi1 = test_phi(jj);

    %For alpha1
        strcmd = sprintf('POWer %d', alpha1); %adjust the amplitude
        fprintf(SG1, strcmd);

        fprintf(SA, 'CALC:MARK1:MAX');       %check peak level
        fprintf(SA, 'CALC:MARK1:Y?');
        peak1 = str2double(fscanf(SA));
        peakLIST(nevals + 1) = peak1;    %store peak level
        alphaLIST(nevals + 1) = alpha1;
        phiLIST(nevals + 1) = phi1;

    %For alpha2
        strcmd = sprintf('POWer %d', alpha2); %adjust the amplitude
        fprintf(SG1, strcmd);

        fprintf(SA, 'CALC:MARK1:MAX');       %check peak level
        fprintf(SA, 'CALC:MARK1:Y?');
        peak2 = str2double(fscanf(SA));
        peakLIST(nevals + 1) = peak2;    %store peak level
        alphaLIST(nevals + 1) = alpha2;
        phiLIST(nevals + 1) = phi2;

    nevals = nevals + 2;    %count evaluations
    dPAj = peak2 - peak1;

    %Calculate change in P with respect to phi;
    alpha2 = test_alpha(jj);
    alpha1 = test_alpha(jj);
    phi2 = test_phi(jj) + dphi;
    phi1 = test_phi(jj) - dphi;

    %For phi1
        strcmd = sprintf('SOURce:PHASe %d deg', phi1);  %adjust the phase
        fprintf(SG1, strcmd);

        fprintf(SA, 'CALC:MARK1:MAX');       %check peak level
```

```
        fprintf(SA, 'CALC:MARK1:CPE:STATe ON'); %continuous peak search ---
marker tracks peak
        fprintf(SA, 'CALC:MARK1:Y?');
        peak1 = str2double(fscanf(SA));
        peakLIST(nevals + 1) = peak1;    %store peak level
        alphaLIST(nevals + 1) = alpha1;
        phiLIST(nevals + 1) = phi1;

    %For phi2
        strcmd = sprintf('SOURce:PHASe %d deg', phi2);
        fprintf(SG1, strcmd);

        fprintf(SA, 'CALC:MARK1:MAX');      %check peak level
        fprintf(SA, 'CALC:MARK1:Y?');
        peak2 = str2double(fscanf(SA));
        peakLIST(nevals + 1) = peak2;    %store peak level
        alphaLIST(nevals + 1) = alpha2;
        phiLIST(nevals + 1) = phi2;

    nevals = nevals + 2;    %count evaluations
    dPpj = peak2 - peak1;

    %Update weights
    rAj = -(dPAj/2/dalpha) / sqrt((dPAj/2/dalpha)^2 + (dPpj/2/dphi)^2);
    rpj = -(dPpj/2/dphi) / sqrt((dPAj/2/dalpha)^2 + (dPpj/2/dphi)^2);

    % Being the sub-iterations
    k = 1; flag = 1;
    Pold = Ptest(jj);
    while(k<Kmax)*flag
        if (Pold >-20)
            phiPOWER = 2.0;
            alphaPOWER = 2.0;
        elseif (Pold>-50)
            phiPOWER = 2.0;
            alphaPOWER = 1.7;
        elseif (Pold>-55);
            phiPOWER = 1.7;
            alphaPOWER = 1.5;
        elseif (Pold>-60)
            phiPOWER = 1.5;
            alphaPOWER = 1.3;
        elseif (Pold>-65)
            phiPOWER = 1.3;
            alphaPOWER = 1.1;
        else
            phiPOWER = 1.0;
            alphaPOWEr = 1.0;
        end

        phiK = test_phi(jj) + dphi*rpj*(phiPOWER^(k-1));
        alphaK = test_alpha(jj) + dalpha*rAj*(alphaPOWER^(k-1));


        strcmd = sprintf('SOURce:PHASe %d deg', phiK); %adjust the phase
```

```
        fprintf(SG1, strcmd);

        strcmd = sprintf('POWer %d', alphaK); %adjust the amplitude
        fprintf(SG1, strcmd);

        fprintf(SA, 'CALC:MARK1:MAX');      %check peak level
        fprintf(SA, 'CALC:MARK1:Y?');
        peakK = str2double(fscanf(SA));
        peakLIST(nevals + 1) = peakK;       %store peak level
        alphaLIST(nevals + 1) = alphaK;
        phiLIST(nevals + 1) = phiK;
        nevals = nevals + 1;                %count evaluations

        if (peakK > Pold)
            flag = 0;
            phiK = test_phi(jj) + dphi*rpj*(phiPOWER^(k-2));
            alphaK = test_alpha(jj) + dalpha*rAj*(alphaPOWER^(k-2));

            strcmd = sprintf('SOURce:PHASe %d deg', phiK); %adjust the phase
            fprintf(SG1, strcmd);

            strcmd = sprintf('POWer %d', alphaK); %adjust the amplitude
            fprintf(SG1, strcmd);

            fprintf(SA, 'CALC:MARK1:MAX');      %check peak level
            fprintf(SA, 'CALC:MARK1:Y?');
            peakK = str2double(fscanf(SA));
            peakLIST(nevals + 1) = peakK;       %store peak level
            alphaLIST(nevals + 1) = alphaK;
            phiLIST(nevals + 1) = phiK;
            nevals = nevals + 1;                %count evaluations

        else
            Pold = peakK;
            k = k+1;
        end
    end         %end sub-iteration while loop

    jj = jj + 1;
    test_alpha(jj) = alphaK;
    test_phi(jj) = phiK;
    Ptest(jj) = peakK;
    nevalsT(jj) = nevals;

end

test_phi = mod(test_phi, 360);
phiLIST = mod(phiLIST, 360);

%Setting canceller to achieve max isolation
[minpeak minindex] = min(peakLIST);
strcmd = sprintf('SOURce:PHASe %d deg', phiLIST(minindex)); %adjust the phase
fprintf(SG1, strcmd);
```

```
strcmd = sprintf('POWer %d', alphaLIST(minindex)); %adjust the amplitude
fprintf(SG1, strcmd);

fprintf(SA, 'CALC:MARK1:MAX');      %check peak level
fprintf(SA, 'CALC:MARK1:Y?');
peak_final = str2double(fscanf(SA));

cancellation = abs(peak_original - peak_final);

fprintf('Attenuator setting (dB) = %g\n', alphaLIST(minindex));
fprintf('Phase setting (deg) = %g\n', phiLIST(minindex));
fprintf('Achieved Isolation = %g\n', peak_final);
fprintf('Achieved Cancellation = %g\n', cancellation);
fprintf('Number power evaluations = %d\n', nevals);

[Screen_Final Trace_Final] = HanaSA_ScreenTrace(SA);

beep;   %beeping sound signals end

%% Plotting
if (DISPon)
    hold on;
    plot(test_alpha, test_phi, 'wo-');
end


[Screen_final Trace_final] = HanaSA_ScreenTrace(SA);
figure;
subplot(121); plot(Trace_original.Frequency, Trace_original.Amplitude);
v = axis;
subplot(122); plot(Trace_final.Frequency, Trace_final.Amplitude)
axis(v);
xlabel('Frequency (Hz)');
ylabel('Amplitude (dBm)');

%% Creating Graphical Output
figure;
plot(Trace_original.Frequency, Trace_original.Amplitude);
xlabel('Frequency (Hz)');
ylabel('Amplitude (dBm)');
titlestr = sprintf('Original %dMHz Signal', freq/1e6);
title(titlestr);
boldify;
v = axis;


figure;
plot(Trace_final.Frequency, Trace_final.Amplitude);
axis(v);
xlabel('Frequency (Hz)');
ylabel('Amplitude (dBm)');
titlestr = sprintf('Cancelled %dMHz Signal', freq/1e6);
title(titlestr);
boldify;
```

A.8 CREATE WAVEFORM FILE (AGILENT)

```
function main
% Creating and downloading waveform files.

%% PART I -- Create Simple IQ Signal
% This signal is a single tone on the upper
% side of the carrier
%

points = 1000;  % Number of points in the waveform
cycles = 101;   % Determines the frequency offset from the carrier

phaseInc = 2*pi*cycles/points;
phase = phaseInc * [0:points-1];

Iwave = cos(phase);
Qwave = sin(phase);

%% PART II -- Save waveform in internal format
% Convert the I and Q data into the internal arb format
% The internal arb format is a single waveform containing interleaved IQ
% data.  The I/Q data is signed short integers (16 bits).
% The data has values scaled between +-23767 where
%   DAC value   Description
%   32767       Maximum positive value of the DAC
%       0       Zero out of the DAC
%  -32767       Maximum negative value of the DAC
% The internal arb expects the data bytes to be in Big Endian format.
% This is opposite of how short integers are saved on a PC (Little Endian).
% For this reason the data bytes are swapped before being saved.

% Interleave the IQ data
waveform(1:2:2*points) = Iwave;
waveform(2:2:2*points) = Qwave;
%[Iwave;Qwave];
%waveform = waveform(:)';

% Normalize the data between +-1
waveform = waveform / max(abs(waveform));   % Watch out for divide by zero.

% Scale to use full range of the DAC
waveform = round(waveform *32767);          % Data is now effectively signed
short integer values

% PRESERVE THE BIT ATTERn but convert the waveform to
% unsigned short integers so the bytes can be swapped.
% Note: Can't swap the bytes of signed short integers in MatLab.
waveform = uint16(mod(65536 + waveform, 65536));

%If on a PC swap the bytes to Big Endian
if ispc
    waveform = bitor(bitshift(waveform, -8), bitshift(waveform, 8));
end
```

```
% Save the data to a file
% Note: The waveform is saved as unsigned short integers.  However,
%       the actual bit pattern is that of signed short integers and
%       that is how the Agilent MXG/ESG/PSG interprets them.
filename = 'C:\Temp\PSGTestFile';
[FID, message] = fopen(filename, 'w');  % Open a file to write data
if FID == -1 error('Cannot Open File'); end
fwrite(FID, waveform, 'unsigned short');     % write to the file
fclose(FID);                                 % close the file
```

A.8 LOAD AND PLAY WAVEFORM FILES

```
% Single Tone Cancel

% fs = 100e6; %sampling freq
% T = 1/fs;
% t = [1:20000]*T;
% f1 = 1e6;
% CancelData = .5*[exp(i*2*pi*f1*t)];
%
% shift = 16;
% SingleCancel = CancelData(shift:9999+ shift);
%
% [status, status_description] = agt_waveformload(siggen2, SingleCancel,
'singletone-cancel', 100000000,'play','no_normscale', Markers);


%% Three-Tone Cancel
fs = 100e6; %sampling freq
T = 1/fs;
t = [1:20000]*T;
f1 = 1e6;
f2 = 3e6;
f3 = 5e6;

IQData = .5*[exp(i*2*pi*f1*t)+ exp(i*2*pi*f2*t) + exp(i*2*pi*f3*t)];
% Normalize to range of +/-0.7.  Do not include the following section if
% passing the flat to normalize and scale the data within the function
% agt_waveformload
maximum = max(abs([real(IQData) imag(IQData)]));
IQData = 0.7*IQData/maximum;

I = real(IQData);
Q = imag(IQData);

ThreeTone = complex(I,Q);

CancelData = complex(-I,-Q);

shift = 1; %103
```

```
ThreeCancel = CancelData(shift:9999 + shift);

[status, status_description] = agt_waveformload(siggen2, ThreeCancel,
'threetone-cancel', 100000000,'play','no_normscale', Markers);
%%
shift_original = 3; %103
ThreeOriginal = ThreeTone(shift_original:9999 + shift_original);

[status, status_description] = agt_waveformload(siggen, ThreeOriginal,
'threetone', 100000000,'play','no_normscale', Markers);

%% Plotting
figure;
subplot(3,1,1); plot(real(CancelData(shift:999+shift)), 'g');
subplot(3,1,2); plot(-real(ThreeTone(shift_original:999+shift_original)),
'm');
subplot(3,1,3); plot((real(CancelData(shift:999+shift))) +
(real(ThreeTone(shift_original:999+shift_original))) , 'y');

%multitone
%% Set up connection with signal generator
siggen2 = agt_newconnection('tcpip', '155.34.109.226');
siggen = agt_newconnection('tcpip', '155.34.109.227');
specan = agt_newconnection('tcpip', '155.34.109.225');

[status, status_description, query_result] = agt_query(siggen, '*idn?');
if(status<0) return; end

%% Generate IQ Data
fs = 100e6; %sampling freq
T = 1/fs;
t = [1:10000]*T;
f1 = 1e6;
f2 = 3e6;
f3 = 5e6;

IQData = .5*[exp(i*2*pi*f1*t)+ exp(i*2*pi*f2*t) + exp(i*2*pi*f3*t)];
%IQData = .5*[exp(i*2*pi*f1*t)];        % single tone

% Generate frequency content of IQData
Y = fft(IQData);       % convert to freq domain, take 512-point FFT of signal
IQData
Pyy = Y.*conj(Y) / length(Y);     % power spectrum (measures power at various
frequencies)
p = (length(Y)/2);
f = fs*(0:p) / length(Y);       % Graph half the points (the other half are
redundant) on a meaningful frequency axis
figure;
plot(f, Pyy(1:(p+1)));
title('Frequency Content of IQData');
xlabel('frequency (Hz)');

% Normalize to range of +/-0.7.  Do not include the following section if
% passing the flat to normalize and scale the data within the function
% agt_waveformload
```

```
maximum = max(abs([real(IQData) imag(IQData)]));
IQData = 0.7*IQData/maximum;

% Generate markers
Markers = zeros(2, length(IQData));
Markers(1,1) = 1;


%% Send SCPI strings to signal generator

%Set frequency and power of signal generator
[status, status_description] = agt_sendcommand(siggen, 'SOURce:FREQuency
10e6');
[status, status_description] = agt_sendcommand(siggen, 'POWer 0');

%download signal 'agtsample'
%[status, status_description] = agt_waveformload(siggen, IQData,
'singletone', 100000000,'play','no_normscale', Markers);
[status, status_description] = agt_waveformload(siggen, IQData, 'threetone',
100000000,'play','no_normscale', Markers);

[status, status_description ] = agt_sendcommand(siggen, 'OUTPut:STATe ON' );

[status, status_description ] = agt_sendcommand(siggen, 'OUTPut:MOD ON' );

%Set center frequency and span of spectrum analyzer
[status, status_description] = agt_sendcommand(specan, 'FREQuency:CENT
10e6');
[status, status_description] = agt_sendcommand(specan, 'FREQuency:SPAN
10e6');

% %% Generate Cancel Signal
% %CancelData = -sin(2*pi*f1*t) - sin(2*pi*f2*t);
% CancelData = .5*[exp(i*2*pi*f1*t + pi) + exp(i*2*pi*f2*t + pi)];
% X = fft(CancelData);
% Pxx = X.*conj(X) / length(X);
% figure;
% plot(f, Pxx(1:(p+1)))
% title('Cancel Data')
%
% Z = fft(CancelData + IQData);
% Pzz = Z.*conj(Z)/length(Z);
% figure;
% plot(f, Pzz(1:(p+1)))
% title('Cancelled Signal')
%
% maximum = max(abs([real(CancelData) imag(CancelData)]));
% CancelData = 0.7*CancelData/maximum;
%
% % Generate markers
% Markers = zeros(2, length(IQData));
% Markers(1,1) = 1;
%
% % %% Send Canceller SCPI strings to signal generator 2
%
```

```
% %Set frequency and power of signal generator
% [status, status_description] = agt_sendcommand(siggen2, 'SOURce:FREQuency
10e6');
% [status, status_description] = agt_sendcommand(siggen2, 'POWer 0');
%
% %download signal 'agtsample'
% [status, status_description] = agt_sendcommand(siggen2, 'SOURce:RAD:ARB
ON');
%
% [status, status_description] = agt_waveformload(siggen2, CancelData,
'multitone-cancel', 100000000,'play','no_normscale', Markers);
%
% [status, status_description ] = agt_sendcommand(siggen2, 'OUTPut:STATe ON'
);
%
% [status, status_description ] = agt_sendcommand(siggen2, 'OUTPut:MOD ON' );
```

8 BIBLIOGRAPHY

1. **Balanis, C.A.** *Antenna Theory: Analysis and Design.* New York : Harper and Row, 1982.
2. **Cripps, Steve C.** *RF Power Amplifiers for Wireless Communications.* Boston : Artech House, 1999.
3. **Haykin, Simon.** *Adaptive Radar and Signal Processing.* Hoboken, NJ : Wiley Interscience, 2007.
4. **Nitzberg, Ramon.** *Radar Signal Processing and Adaptive Systems.* Boston : Artech House, 1999.
5. **Skolnik, Merrill.** *Radar Handbook, 2nd Edition.* Boston : McGraw Hill, 1990.
6. **Hovanessian, Shahan.** *Introduction to Sensor Systems.* Norwood, MA : Artech House, 1988.
7. *FPGA Based Real Time Solution for Sensitivity Time Control.* **Prakasam, L.G.M. and Meena, D.** 2008, Electronic Design, Test and Applications, 2008. DELTA 2008. 4th IEEE International Symposium on, pp. 244-248.
8. **Weisman, Carl J.** *The Essential Guide to RF and Wireless.* Upper Saddle River, NJ : Prentice Hall, 2000.
9. *On the MIMO Channel Capacity Saturation for Spatially Constrained Multielement Antenna Systems.* **Wu, Yujiang and Nie, Zaiping.** 2006, Wireless Communications, Networking and Mobile Computing, 2006. WiCOM 2006. International Conference on, pp. 1-4.
10. **Smith, Glenn S.** *Classical Electromagnetic Radiation.* Cambridge, United Kingdom : Cambridge University Press, 1997.
11. *Signal Cancellation Phenomena in Adaptive Antennas: Causes and Cures.* **Widrow, Bernard et. al.** 1982, IEEE Transactions on Antenna and Propogation, pp. 469-478.
12. **Sarkar, Tapan K. and Wicks, Michael C.** *Smart Antennas.* Hoboken, NJ : Wiley-Interscience, 2003.
13. **Brown, Stanley.** A Feasibility Analysis of Single-Sensor Active Noise Cancellation in the Interior of an Automobile. *Master of Science Thesis.* Cambridge, MA : MIT, 1995.
14. **Tokhi, M and Leitch, R.** *Active Noise Control.* Oxford, England : Clarendon Press, 1992.
15. *Dual polarised microstrip patch antenna using feedforward isolation enhancement for simultaneous transmit/receive applications.* **Karode, S.L. and Fusco, V.F.** 1999, Antennas and Propagation, 1999. IEE National Conference on., pp. 49-52.
16. *Active Noise Control using Adaptive Digital Signal Processing.* **Eriksson, L. J., Allie, M. C. and Bermigan, C. D.** 1988, 1988 IEEE International Conference on Acoustics, Speech, and Signal Processing, pp. 2594-2597.
17. **Haykin, Simon.** *Adaptive Filter Theory.* Upper Saddle River, NJ : Prentice Hall, 2002.
18. **Zarchan, Paul and Musoff, Howard.** *Fundamentals of Kalman Filtering, 2nd Edition.* Virginia : American Institute of Aeronautics and Astronautics, 2005.
19. **Minkler, G. and Minkler, J.** *Theory and Application of Kalman Filtering.* Palm Bay, FL : Magellan Book Company, 1993.
20. **Haykin, Simon and Widrow, Bernard.** *Least-Mean-Square Adaptive Filters.* Hoboken, NJ : Wiley Interscience, 2003.
21. *A Direct Frequency Synthesizer.* **Tierney, Joseph, Rader, Charles and Gold, Bernard.** 1991, Audio and Electroacoustic, IEEE Transactions on, pp. 48-57.
22. **Agilent Technologies.** IQ Modulation. *Agilent Education Resources.* [Online] [Cited: April 1, 2008.] http://education.tm.agilent.com/index.cgi?CONTENT_ID=4.
23. **National Instruments.** NI Developer Zone. *National Instruments.* [Online] [Cited: May 4, 2008.] http://zone.ni.com/devzone/cda/tut/p/id/4805.