

An FPGA Based Adaptive Computing Implementation of Chirp Signal Detection

J.C. Zaino¹, R.L. Bassett¹, T.S. Sun¹, J.M. Smith¹, E.K. Pauer¹, S. Kintigh¹, D.W. Tufts²

¹Sanders, A Lockheed Martin Company, Nashua, NH 03061-0868

²University of Rhode Island, Kingston, RI 02881

Abstract

A technique for linear FM chirp signal detection on a COTS Field Programmable Gate Array (FPGA) based reconfigurable computing testbed has been implemented. The scheme used for signal detection was based on a semi-coherent method originated by Lank, et al[1]. The scheme developed addresses, in an adaptive computing environment, the detection of a family of signals that can be hard to detect at reasonable false alarm rates. The techniques we have demonstrated make it possible to enhance the detection and measurement SNR by 10 to 20 dB (depending on signal pulse widths and excursions) and to process incoming data through hardware at real time rates in excess of 30 Million samples per second. The semi-coherent technique for signal detection was one of three signal processing schemes studied using analytical models as well as Monte-Carlo type simulations. We have illustrated the performance potential of Adaptive Computing technology in high bandwidth data streaming applications. The result was a powerful approach to rapid iterative ACS algorithm analysis and visualization using popular commercial tools and confirmation of results in end user preferred formats.

I. INTRODUCTION

Under the AFRL/DARPA Adaptive Computing Systems (ACS) "Reconfigurable Algorithms for Adaptive Computing" (RAAC) program, Sanders, A Lockheed Martin Co. is investigating innovative algorithms that are applicable to the signal processing of digitized RF signals including electronic warfare (EW) operations suitable for field programmable gate array (FPGA) implementation. The latter has been identified as one of the most promising enabling hardware technologies for advanced adaptive computing system realization. Many RF systems and in particular EW systems have two major operational requirements: the detection of electromagnetic emissions and the demodulation or measurement of their characteristics. The criteria for assessing any particular algorithm design are thus based on performance with respect to these operational requirements as well as its efficiency with respect to FPGA implementation. This paper describes the implementation of a technique for linear FM chirp

detection on a COTS FPGA based reconfigurable computing testbed. The scheme used for signal detection is based on a semi-coherent method originated by Lank, et al[1].

The scheme we have demonstrated addresses, in an adaptive computing environment, the detection of a family of signals that can be hard to detect at reasonable false alarm rates. When viewed by a receiver that simply looks for energy in the frequency excursion band of interest the SNR (signal to noise ratio) of many of the signals of interest are very low or negative. Such signals are undetectable without further processing. The techniques we have demonstrated make it possible to enhance the detection and measurement SNR by 10 to 20 dB (depending on signal pulse widths and excursions) and to process incoming data at real time rates in excess of 30 Million samples per second.

The semi-coherent technique for signal detection was one of three signal processing schemes studied using analytical models as well as Monte-Carlo type simulations. The techniques were compared in terms of algorithm complexity and parameter estimation performance. These studies were aimed at defining the optimal operational boundaries of each scheme versus defining which scheme was the best design choice. This will allow a reconfigurable design strategy to be developed to exploit the full potential offered by the adaptive computing architecture. These studies also established new rapid closed form techniques for the calculation of expected processing gain performance for an ACS based detection approach. The algorithms were partitioned into FPGA implementable segments and FFT accelerator implementable segments. Matlab simulations of each technique were generated as part of the trade off process. The semi-coherent detection approach provides the best compromise of performance versus computational complexity, and thus FPGA chip area. The coherent detection approach, which uses multiple matched frequency chirps, provides the best performance overall though at substantial complexity and area cost. Thus, the semi-coherent approach is best suited for real-time and non real-time large file processing. The coherent approach is the best approach for close inspection and measurement of specific signals.

The semi-coherent method consists of two parts: a preprocessing step which involves a sample-by-sample multiplication of the received signal with the complex conjugate of a delayed copy, and an FFT spectral analyzing step. We have implemented the pre-processing step of this scheme in an FPGA using and extending tools developed on the (DARPA ACS) Algorithm Analysis and Mapping Tools program, including coupling of data between Ptolemy (the

"This material is based upon work supported by the AFRL/Information Directorate under Contract No. F30602-98-C-0104. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the AFRL/Information Directorate"

design environment used for the Algorithm Analysis and Mapping Tools program), Matlab and a DoD Signal Processing Analysis package. The result was a powerful approach to rapid iterative ACS algorithm analysis and visualization using popular commercial tools and confirmation of results in end user preferred formats.

We have illustrated the performance potential of Adaptive Computing technology in high bandwidth data streaming applications, using a hybrid FPGA and FFT data streaming architecture to accomplish a function and data rates conventionally achieved only in analog applications. From a modular view, the results illustrate the ability of a single chip ACS implementation to perform the core function of a 9x9 inch conventional analog digital technology module while reducing power requirements from 70W to less than 3 W.

II. PRELIMINARY ASSESSMENT OF FM SIGNAL DETECTION TECHNIQUES

A. Comparison of FM Signal Detection Techniques

The most common paradigm for detection and classification of an unknown signal in a noisy environment consists of three processing stages: signal transformation, peak detection, and parameter estimation. The main purpose of transformation is to increase the signal-to-noise ratio and to represent the signal in a domain where detection and measurement can be made easier and more robust. This is the stage where most approaches differ from each other. We have conducted a preliminary study of five approaches for detection of linear FM signals and compared their processing requirements and performances for some selective cases.

To help clarify the following discussions, we consider the following problem. The objective of each processing approach considered here is to detect and perform waveform measurements of pulse or CW signals described as

$$A \exp[j(\omega t + \beta t^2)] \quad \text{for } t_n \leq t < t_n + T \quad (1)$$

and $t_n = n\tau_{PRI}$

where A is the complex amplitude, ω is the starting frequency, β is the chirp rate, T is the pulse width, and τ_{PRI} is the pulse repetition period. Throughout this and next sections, both frequency and chirp rate contain implicitly the factor of 2π .

For a rigorous evaluation, the performance must be evaluated in terms of detection probability under a specified false alarm rate and the accuracy of estimation for the four most important waveform parameters (ω , β , T , τ_{PRI}). Rather than performing a detailed evaluation by applying such rigorous criteria to all five approaches, the objective of this preliminary study was to quickly identify one or two approaches as the candidate for future design consideration before a rigorous treatment is applied.

Unquestionably, the processing cost is one of the primary considerations for selecting an algorithm for the basis of final architecture design. The cost criterion was not only based on the number of operations as estimated from simple mathematical analyses or computed using MATLAB simulations, but was also based on its efficiency for implementation in FFT engines and FPGA.

1) Least Squares Method

This approach uses short-time FFT to transform the input signal into the spectral domain. Similar to the generation of a spectrogram, the FFT is applied on data collected from a short time frame, which advances in time with some overlap. A threshold detection is performed in each spectral profile to determine whether a signal is present with a certain likelihood. When such detection occurs in a number of consecutive frames, the frequencies of the detected peaks are used to determine the starting frequency and chirp rate of the signal by means of linear regression, i.e., the least squares method. The processing sequence of this method is shown in Fig. 1.

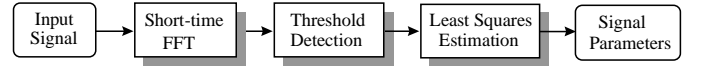


Figure 1. Least squares method for detection and estimation of FM waveforms.

With regards to processing requirements, the major processing load of this method is in the short-time FFT step, which is based on the following discrete Fourier transform:

$$y_{mk} = \sum w_n x(t_m + \tau_n) \exp(-j\omega_k \tau_n) \quad (2)$$

where y_{mk} is the output of the k -th frequency bin for the m -th frame, $x(t)$ is the input signal in complex representation, w_n is the n -th coefficient of a window function, t_m is the starting time of the m -th frame, τ_n denotes the n -th point in the frame, and ω_k represents the (angular) frequency of the k -th bin. In addition, the α -th power of $|y_{mk}|$ is computed for each bin prior to the next processing step where α is generally an even integer.

The number of operations involved in each frame can be approximated by

$$M_{max} N_{max} [N_{CMPLX} \log_2(N_{max}) + 3 + \alpha/2] \quad (3)$$

where $1 \leq k \leq K_{max}$ number of frequency bins \leq FFT size

$1 \leq m \leq M_{max}$ number of frames

$1 \leq n \leq N_{max}$ frame size = FFT size

$N_{CMPLX} = 6$	number of ops per complex multiplication
$\alpha = 2, 4, \dots$	power exponent of FFT output magnitude

The values of these processing parameters need to be optimized with respect to both the performance requirement and processing efficiency. In particular, the value of the FFT size, N_{max} , which controls both frequency and time resolutions, must be chosen in consideration to the signal bandwidth of the frame and at the same time in consideration to the hardware speed performance factor.

The least squares estimation function in Fig. 1 is based on the principle that the best estimate of the linear FM waveform parameters (ω, β) is one that minimizes the following cost function.

$$Q = \sum \rho_{mk} (\omega_{mk} - \omega - 2\beta t_m)^2 \quad (4)$$

where ω_{mk} is the measured frequency of the k -th bin in the m -th frame, t_m is the starting time of the m -th frame, and ρ_{mk} is a weighting factor characteristic of the relative importance of each data point. In practice, the weight ρ_{mk} is made equal to some even power of the amplitude (i.e., $\rho_{mk} = |y_{mk}|^\alpha$ with $\alpha = 2, 4, \dots$). Minimizing Q with respect to ω and β leads to the following system of linear equations from which ω and β can be determined.

$$A_{11} \omega + A_{12} \beta = B_1 \quad (5)$$

$$A_{21} \omega + A_{22} \beta = B_2 \quad (6)$$

where

$$A_{11} = \sum \sum \rho_{mk}, \quad A_{12} = 2A_{21} = 2 \sum \sum \rho_{mk} t_m,$$

$$A_{22} = 2 \sum \sum \rho_{mk} t_m^2$$

$$B_1 = \sum \sum \rho_{mk} \omega_{mk}, \quad B_2 = \sum \sum \rho_{mk} \omega_{mk} t_m$$

The number of operations involved in the computation of these coefficients is approximately equal to $M_{max}N_{max}(\alpha/2 + 9)$.

There are two major advantages of the least squares method. First, its processing is relatively efficient; and secondly, it can estimate chirp rate and starting frequency directly. However, this approach has a major drawback in that it is mainly useful for the detection and estimation of a dominant signal in the field. Its performance can be affected by interference from multiple signals if more than one signals are of equal strength. Besides, its processing gain is the least of the five approaches.

2) Hough Transform Method

Hough transform is a data processing technique widely used to enhance the detection of linear structures in image data. This transform has the unique property in that it maps all the points along a line in the original image domain into a single point in the transform domain. Thus, in principle the detection of the a linear structure becomes the detection of a single point with an enhanced probability. In addition, the location of the point in the transform domain can be used to determine the distance and the slope of the line. Based on the observation that linear FM signals exhibit streaks of slant line segments in a spectrogram, it is logical to exploit this technique for detection of such signals.

This approach also uses short-time FFT in the first stage to transform the time-series signal into a time-frequency 2D representation. The entire processing sequence of this approach is schematically shown in Fig. 2. To reduce processing load, the high-value data selection function in Fig. 2 uses a threshold to select pixels of large magnitudes prior to Hough transform.

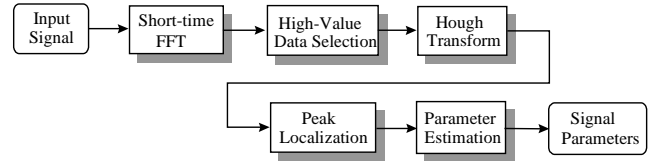


Figure 2. Block diagram of the Hough transform method

The requirement for short-time FFT function is the same as the least squares method and is also the most demanding part of this approach. The second most computationally intensive step is the Hough transform, which involves the following computations.

For each input cell or pixel $z(x_m, y_n)$:

If $z(x_m, y_n) > \gamma_{Th}$

$$\text{Then } q_k = x_m \cos(\phi_k) + y_n \sin(\phi_k) \quad (7)$$

$$i = \text{int}(p_k) \quad (8)$$

$$z_H(i, k) = z_H(i, k) + q(x_m, y_n) \quad (9)$$

where x_m and y_n represent the Cartesian coordinates of a cell in the spectral domain, γ_{Th} is the threshold to control the number of input cells to be computed, and ϕ_k is an independent variable whose range is determined by the range of all chirp rates of interest, and i and k represent the indices of a cell in the parameter (transform) space. The number of operations required for these computations is approximately equal to:

$$5K_{max}N_0 + M_{max}N_{max} \quad (10)$$

where $K_{max} > k$ is the number of resolvable chirp rates, M_{max} is the number of frames in the spectral domain, N_{max} is

the number of frequency bins in the spectral domain, and $N_0 < M_{max} N_{max}$ is the number of input cells with value $> \gamma_{Th}$.

The minimum requirement for the peak localization function is locate the highest peak in the 2D surface generated by the output data. In general, the surface is smoothed first by means of convolution with a 2D kernel to minimize the effect of spurious noisy spikes on the true peak location. The processing routine involves the following computations

$$z_{max} = 0$$

for the row-loop

for the column-loop

$$z_1(x_m, y_n) = \sum \sum v(\Delta x, \Delta y) z(x_m - \Delta x, y_n - \Delta y) \quad (11)$$

If $z_1(x_m, y_n) > z_{max}$

$$z_{max} = z_1(x_m, y_n) \quad (12)$$

$$\text{peak index} = (m, n) \quad (13)$$

where $z_1(x_m, y_n)$ represents the smoothed surface.

The number of operations is approximately equal to $M_{max} N_{max} [(\Delta_M + 1)(\Delta_N + 2)]$, where M_{max} is the number of columns and N_{max} is the number of rows of the 2D data, and Δ_M and Δ_N are the sizes of the smoothing kernel in the x and y dimensions respectively.

With the proper choice of a control threshold, the processing efficiency of the Hough Transform method can be made to be nearly comparable to that of the least squares method. One advantage of this approach compared to the former is the capability to detect multiple signals if they are well separated in the parameter space. However, the waveform estimation performance of this approach in the presence of noise can be inferior to that of the least squares method in terms of accuracy. The performance is also susceptible to interference from multiple sources if they are close in the parameter space.

3) Wigner Distribution Function Method

Beside the short-time FFT, another well-known approach to the time-frequency analysis of signals is based on Wigner distribution function (WDF). The WDF is a bilinear transformation which involves the Fourier transform of the product of a delayed copy and the conjugate of an advanced copy of the signal, i.e., $x(t_m - \tau_n/2) x^*(t_m + \tau_n/2)$. Like the short-time-FFT method, the WDF method also transforms the time-series of a signal into a frequency-time 2D representation. In particular, the WDF of a linear FM signal also exhibits streaks of slant lines in a spectrogram. For this reason, the processing sequence of this approach is identical to that of Fig. 2 except for the first step.

However, unlike the short-time FFT method which cannot

improve both frequency resolution and temporal resolution simultaneously by adjusting the signal frame size, the WDF method can achieve a higher frequency resolution by increasing the frame size without degrading the temporal resolution. Hence it is advantageous to use this method to produce a sharper peak in the parameter space and to improve the detection and estimation performance compared to the second approach.

The following discrete pseudo-WDF expression is widely adopted for computing the WDF of an input signal $x(t)$.

$$y(t_m, \omega_k) = \sum w_n x(t_m - \tau_n/2) x^*(t_m + \tau_n/2) \exp(-j\omega_k \tau_n) \quad (14)$$

where w_n is a sliding window used to smooth the cross spectra arising from the bilinear transformation. The number of operations required for is equal to:

$$M_{max} N_{max} [N_{CMPLX} (1 + \log_2 N_{max}) + 2\delta_w] \quad (15)$$

where M_{max} is the number of frames, N_{max} is the window size, which is usually made equal to the size of FFT for practical purpose, N_{CMPLX} (~ 6) is number of operations per complex multiplication, and δ_w has a value of zero or unity depending on whether a rectangular window is used or not. The WDF process requires an extra amount of $M_{max} N_{max} N_{CMPLX}$ operations in comparison to a short-time FFT process under a similar condition. The number of operations is further increased if a large size of N_{max} is used to improve the performance. This has the benefit of improving the frequency resolution in the frequency-time domain, which translates to a benefit of reducing the peak width in the parameter space; but at the expense of processing.

Compared to the short-time FFT approach, the WDF method has the advantage of providing an improved performance in detection probability and estimation accuracy. However, there are two major drawbacks. First, significantly more computations are required, and second, its performance is affected by cross-spectra in the generation of WDF if two signals are correlated inside the window.

4) Chirp-Matched Filter Method

This approach exploits the classical matched filter concept by using a bank of filters generated from chirp-waveforms [2]. The idea is to design the bank with a sufficient completeness in the expected range of signal chirps so that any input signal can be “de-chirped” by one of the filters to become a monotone signal. The latter can be then analyzed by means of a traditional FFT technique. The output of this method is a 3D representation with Cartesian axes in time, frequency, and

chirp-rate. Fig. 3 shows the structure of the processing flow for each input frame.

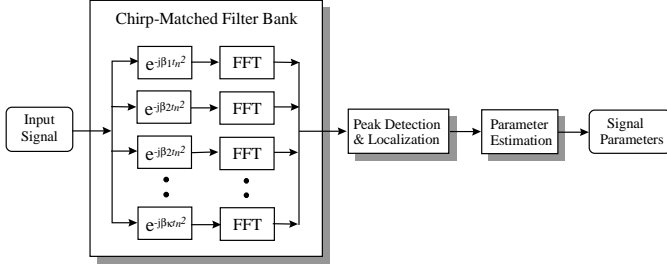


Figure 3. Flow diagram of chirp-matched filter bank method.

The main processing load of this approach is consumed in the first block of Fig. 4 where processing of a single frame with each chirp filter involves the following computation.

$$y(t_m, \beta_i, \omega_k) = [\sum x(t_m - \tau_n) \exp(-j\beta_i \tau_n^2) \exp(-j\omega_k \tau_n)]^2 \quad (16)$$

The number of operations is given by

$$I_{max} M_{max} N_{max} [N_{CMPLX} (1 + \log_2 N_{max}) + 3] \quad (17)$$

where I_{max} represents the size of the chirp filter bank, M_{max} represents the number of input frames to be processed, and N_{max} is the frame size.

Because this approach is based on matched filter principle, one can achieve a nearly optimal coherent processing gain using this approach. Therefore, it offers the best performance in detection and estimation among the five approaches evaluated here. Furthermore, it is also the most robust against multiple-source interference. However, this approach is the most computationally expensive. For practical implementation, it is necessary to control the number of filters and the size of time increment to make the processing problem manageable.

5) Semi-coherent Processing Method

The semi-coherent approach uses a nonlinear transformation involving the product of the input signal with the conjugate of a delayed copy of itself, i.e., $x(t) x^*(t-\tau)$ [3]. The output from this process will contain a monotone signal with a frequency of $(2\beta\tau)$ where τ is the delay time. This is the beat frequency between the signal and its delayed copy. In the absence of noise this is equivalent to the output from a perfectly matched chirp filter described in the above section. Thus, the performance approaches to the coherent result as the SNR increases. The chirp rate β can be determined immediately by performing a FFT to the product sequence.

Once β is determined, we can estimate the value of ω with one of the following two schemes. The first scheme is to

exploit the phase of the output, which contains a frequency-dependent factor. The second scheme is to de-chirp the input signal with a filter constructed from $\exp[-j\beta t^2]$. The processing sequence of the second scheme, which was used in the preliminary study, is shown in Fig. 4.

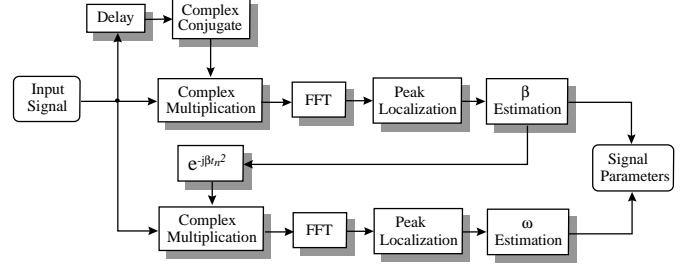


Figure 4. Flow diagram of the semi-coherent method

The main processing step in the semi-coherent approach involves:

$$y(\tau_m, \omega_k) = [\sum x(t_n) x^*(t_n - \tau_m) \exp(-j\omega_k t_n)]^2 \quad (18)$$

Thus, for each delay τ_m , the number of operations required is given by

$$M_{max} N_{max} [N_{CMPLX} (1 + \log_2 N_{max}) + 3] \quad (19)$$

where the processing parameters are defined as before.

This approach is relatively efficient and is capable of producing an accurate estimate of chirp-rate. The frequency estimate is generally less accurate than other approaches as the error in the chirp rate estimation can also propagate to the frequency estimation. This approach is also susceptible to multiple-source interference, which arises from the non-vanishing cross-products in the first processing step. However, this problem can be overcome with algorithmic design based on a careful analysis of their mutual frequency relationship. Care must be taken though, since at low SNR's signal suppression can occur.

As shown in the next section, this approach appears to be the most efficient in the preliminary evaluation and has (a less though,) reasonable performance in comparison to other approaches. Therefore, a more in-depth analysis has been performed on this approach.

6) MATLAB Simulation Results

We have used MATLAB simulations to evaluate the above five approaches. Two cases of input have been simulated in this study. In the first case, the input contains a single signal source with SNR = 0 dB. The main purpose of this part was to study the processing load and the accuracy of waveform parameter estimation of these detection techniques. In the

second case the input contained three strong signal sources of comparable SNR's. The purpose of this part was to get an idea of how these techniques in their most basic form perform under multiple-source situation. The signal sources were of the form:

$$s(t) = A \exp[-j(\omega t + \beta t^2 + \phi_0)] \quad \text{for } t_n \leq t < t_n + T \quad (20)$$

and $t_n = n\tau_{PRI}$

We show in Table 2 the results of MATLAB simulation of the above five detection approaches regarding the processing load (in terms of Mflops), and the results of frequency and chirp rate estimation. Before any conclusion can be drawn, the following points need to be considered to account for this study being a preliminary versus completely rigorous one:

- a) The results shown here are not based on any statistical average. Each of them is regarded as a typical result out of at least three separate runs.
- b) The starting point of the input coincides with the beginning of a pulse of the dominant signal. This assumption reduces the value of M_{max} to unity in the processing requirement of the last two approaches. This has the effect of making the processing cost of these two approaches to appear lower by a factor of two or three.
- c) The processing parameters for each individual method, such as the size of FFT, the size of the peak-smoothing kernel, and the number of de-chirping filters, have not been optimized.
- d) In the peak localization subroutine, the baseline algorithm only searches for the cell with the greatest value. This algorithm does not work well in the case where multiple signals are present.

Summarizing the preliminary results:

- a) In case 1 where only a single signal was present, all five methods performed very well (qualitatively comparable).
- b) Based on the processing cost, these five methods can be divided into two groups: (a) a low-cost group consisting of the least squares method, the Hough transform method and the semi-coherent method; and (2) a high-cost group consisting of the WDF method and the chirp-matched filter method.
- c) In case 2 where three signals were present, the chirp-matched filter method appeared to be the most robust whereas the semi-coherent method appeared to be the least robust. A close examination of the latter revealed that the high value of β (18.18) was due to a combination of two factors: (1) the cross-product of two of the three signals, and (2) the deficiency of single-peak search algorithm. However, it needs to be pointed out that in at least one simulation run, a correct value of β was produced by the semi-coherent method.

d) Based on this study, the last two approaches in Table 2 appear to deserved further evaluation. In particular, the chirp-matched filter method had the best performance whereas the semi-coherent method had the best computational efficiency. An increased focus was then placed on the semi-coherent method. A more in-depth analysis of this method is presented in Section III.

Table 2. Summary of the MATLAB summarizes the results of the two input cases

	Processing Cost (Mflops)	Result of Case 1		Result of Case 2	
		$\omega/2\pi$	$\beta/2\pi$	$\omega/2\pi$	$\beta/2\pi$
Original Waveform (Signal #1)		15.00	20.00	30.00	10.00
Least Squares Method	0.84	15.14	19.75	31.30	9.73
Hough Transform Method	1.37	15.45	20.18	29.96	11.20
WDF Method	9.72	16.20	20.12	30.48	9.22
Chirp-Matched Filter Method	4.47	15.14	19.75	30.08	10.00
Semi-coherent Method	0.69	16.19	19.97	30.27	18.18

III. SEMI-COHERENT METHOD FOR FM SIGNAL DETECTION

A. Probability Distribution Function (PDF) of Semi-coherent Processing Output

If the probability distribution function (PDF) of the processing output is known for all possible values of input SNR's, it is possible to predict the performance of the processing system and hence to design optimal processing parameters. In particular, we can use the PDF for noise-only input to determine a threshold to control the probability of false alarm and use this threshold and the PDF for signal-plus-noise input to predict the probability of detection. As described earlier, the first processing stage of the semi-coherent method involves a sample-by-sample multiplication of the received signal with the complex conjugate of a delayed copy, followed by a Fourier transform. In this case, the PDF of the processing output is a function of three variables: the input SNR, the length of FFT integration, and the size of delay used for the conjugate part. In the following sections, we will show a rigorous derivation of the PDF, developed by Tufts [4], and demonstrate some useful closed-form empirical approximations to PDF-related parametric functions for practical implementation.

1) Theoretical Framework

The output of the first stage in the semi-coherent method

involves

$$y(\tau_m, \omega_k) = \sum x(t_n) x^*(t_n - \tau_m) \exp(-j\omega_k t_n) \quad (21)$$

We assume that the input consists of a deterministic signal and a random noise:

$$x(t) = s(t) + n(t) \quad (22)$$

where $s(t)$ is an analytical signal with amplitude A and $n(t)$ is a complex-valued, zero-mean, Gaussian noise. The real and imaginary parts of $n(t)$ are assumed to be mutually independent, each with variance σ^2 . Therefore the input signal-to-noise ratio ρ_{in} is

$$\rho_{in} = A^2/(2\sigma^2). \quad (23)$$

The probability distribution function (PDF) of $|y(\tau_m, \omega_k)|$ for any given value of ρ_{in} can be derived from the result of Tufts' work[3]. His treatment leads to the following set of equations for computing the probability density of normalized $|y(\tau_m, \omega_k)|$.

$$R(\rho, \gamma) = \frac{\rho}{2\pi \sqrt{(a^2 - b^2)}} e^{-\frac{1}{2} \left\{ \frac{[\rho \cos(\gamma) - 1]^2}{(a+b)} + \frac{[\rho \sin(\gamma)]^2}{(a-b)} \right\}} \quad (24)$$

$$P(\rho) = \int_{-\pi}^{\pi} R(\rho, \gamma) d\gamma \quad (25)$$

where $R(\rho, \gamma)$ is the joint probability density¹ of the magnitude and the angle of the complex-valued random variable $y(\tau_m, \omega_k)$, and $P(\rho)$ is the desired PDF, evaluated with respect to the normalized magnitude ρ . The two parameters a and b are given by the following expressions.

$$a = \frac{1}{N_s \rho_{in}} \left(1 + \frac{1}{2\rho_{in}} \right) \text{ and} \quad (26)$$

$$b = \frac{1}{N_s \rho_{in}} \left(\frac{N_s - M_D}{N_s} \right) \quad (27)$$

where N_s is the number of terms in the summation and M_D is the length of delay.

2) Attributes of PDF and Empirical Models

The function $P(\rho)$ does not have a closed form and must be computed numerically for each value of ρ and for each input SNR. However, for practical purpose it can be approximated by simple analytical functions for certain range of ρ_{in} . In particular, in the limit $\rho_{in} = 0$, $P(\rho)$ can be placed by a Rayleigh function. This represents the probability distribution of the output magnitude when the input is noise only. Furthermore, for a wide range of $\rho_{in} > 0$, $P(\rho)$ can be approximated by a Gaussian function with a unity mean. It is important to point out that the variance of output distribution with both signal and noise differs from the variance of output distribution with noise only. In particular, for a wide range of ρ_{in} , their ratio follows a straight line. Results from MATLAB simulation (marked by asterisks) also confirm this linear relationship. This property is different from Ricean statistics, which is characteristic of random variables generated from linear combinations of a sinusoidal signal and a Gaussian noise. Although the Ricean distribution function can be also replaced by the Rayleigh and Gaussian functions in the noise-only and signal-plus-noise cases, their variances are always equal, regardless of the value of ρ_{in} . For this reason, the empirical formula developed by Albersheim to relate the required SNR to probability of detection and probability of false alarm [5] is not applicable in the semi-coherent case.

The output SNR can be computed with the above set of equations. However, for a wide range of ρ_{in} , it can be computed using the following simple empirical model.

$$\rho_{out} = N_s \rho_{in} / (C + 1/\rho_{in}) \quad \text{with } 2 < C < 4 \quad (28)$$

The value of C in this model depends on N_s and M_D , and must be determined empirically or computed using the equations of $P(\rho)$. Figure 5 shows the result of using this empirical model (solid line) in comparison with the result (marked by circles) obtained from a numerical integration of $P(\rho)$. The results of MATLAB simulation (marked by asterisks), each based on 10^4 runs, are also in excellent agreement with these data.

¹ The phase angle of the semi-coherent processing output can be used to estimate the frequency of the signal. Integration with respect to the normalized magnitude will produce the PDF of the phase angle, which can be then used to predict the accuracy of frequency estimation.

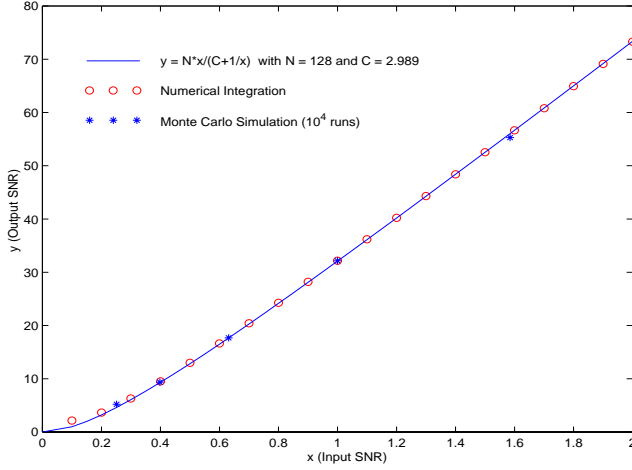


Figure 5: Output SNR as a Function of Input SNR Computed by Means of Empirical Approximation and Numerical Iteration

B. Computations of Input SNR and False Alarm Threshold

The following treatment provides a mathematical foundation for the derivation of a simple algorithmic routine to predict the minimum SNR and to determine the threshold that can satisfy pre-specified probabilities of detection and false alarm (P_D, P_{FA}).

When the signal is absent, we use the Rayleigh distribution.

$$P_{FA} = \exp[-\gamma^2/(2\sigma_1^2)] \quad (29)$$

where γ is the false alarm threshold

$$\gamma = \sigma_1[-2 \ln(P_{FA})]^{1/2} = \sigma_1(2^{1/2}p) \quad (30)$$

$$\text{where } p = [-\ln(P_{FA})]^{1/2}$$

When the signal is present, we use the Gaussian distribution.

$$P_D \simeq 0.5[1 - \text{erf}(u)] \quad (31)$$

$$\text{where } u = \gamma/(2^{1/2}\sigma_2) - \rho_{out}^{1/2} \quad (32)$$

It follows that

$$u = (\sigma_1/\sigma_2)p - \rho_{out}^{1/2} \quad (33)$$

$$\rho_{out} = [(\sigma_1/\sigma_2)p - u]^2 \quad (34)$$

$$= N_S \rho_{in} / (C + 1/\rho_{in}) \quad (35)$$

Finally the required input SNR can be expressed as

$$\rho_{in} = (q^2 - 1)/C \quad (36)$$

where q is the positive root of $a_n q^2 + uq - (a_n + p) = 0$ and $a_n = N_S^{1/2}/C$.

The above equations can be used to compute the required SNR for a given set of (P_D, P_{FA}). To expedite this computation, we have developed a fast routine to determine u for any given P_D , i.e., compute the inverse function of $P_D = 0.5[1 - \text{erf}(u)]$.

C. MATLAB Demonstration of Detection and False Alarm Performance

We conducted two experiments to demonstrate the validity of the above empirical models using MATLAB simulation. In each experiment, a total of 10^4 simulation runs were performed. In the first experiment, we used a constant threshold derived from the above empirical method. The result is presented in Fig.6. The solid curves in Fig. 6 show the empirical PDF's of the processing output magnitude for both noise-only input and signal-plus-noise input. The asterisks represent the MATLAB results. The threshold used for limiting the false alarms is also indicated in this figure. The simulation shows a slightly higher P_D but also a slightly higher P_{FA} . The difference is not significant considering the small number of runs used in the simulation, although it may suggest that the threshold as calculated from the empirical method is slightly lower.

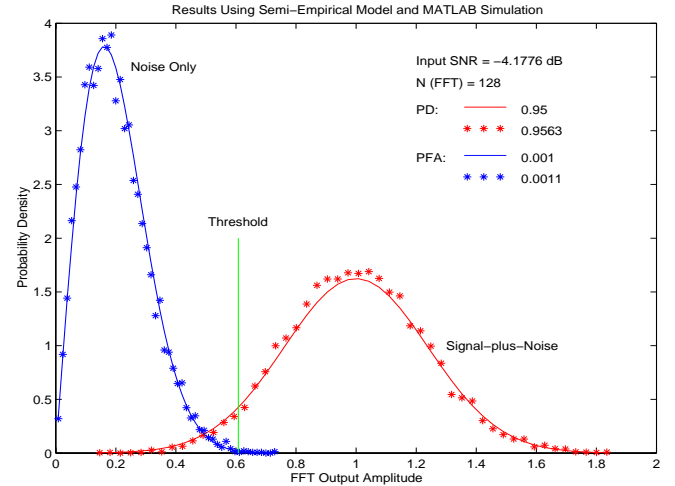


Figure 6: MATLAB Simulation Result and the Empirical Models of PDF for Semi-coherent Processing Output

In the second experiment, we varied the threshold and measured the P_D and P_{FA} respectively. It was shown that the model predictions were in excellent agreement with the results of simulation.

IV. FPGA BASED IMPLEMENTATION OF LINEAR FM CHIRP SIGNAL DETECTION

A. Basic Implementation

Figure 7 illustrates the primary elements of the demonstration. The demonstration was built on a SparcStation 30 with PCI bus. The demonstration utilized Annapolis Microsystems Wildforce module (FPGA) integrated into the workstation and adaptive computing Ptolemy tools [6] developed by the DARPA funded Algorithm Analysis and Mapping Environment (AAME) program for the GUI and implementation aspects of the demonstration. Preparation of this demonstration clearly illustrated the powerful capability and versatility of these tools in allowing a rapid turn around from concept to an adaptive computing hardware/software implementation.

The basic demonstration supported the original design goal of a 25MHz bandwidth, supporting operation at 25 MSamples/second for the complex data (I, Q pairs) of linear FM chirped signals (maximum rate achieved was 40 MS/s as discussed below). The signals were detected by correlating the incoming signal against a delayed copy of itself and looking for resultant constant differential tones. The

Four signal generators were provided and combined. These elements were implemented as Ptolemy stars. Three of the generators produced periodic signals. The fourth generated band limited analytic white noise. The demonstration chirp rates were chosen to provide an interesting range of chirp rate excursions and pulse widths. Chirp Rates ranged from 300KHz/us for the slowest which represented between the first and second bins in the output FFT to 2.3MHz/us which at the widest excursion provided for the near minimum pulse width handled of 10us

The FPGA segment was implemented in an Annapolis Microsystems Wildforce board in two configurations, each developed using the Adaptive Computing System tools [5]. One configuration used a single processing element (PE) FPGA while the other used all five FPGA PEs on the Wildforce board. The single PE implementation was the first hardware implementation of the demonstration. This implementation, which ran at 35 MHz, required five clock cycles per I/Q sample pair due to the serial implementation required by the PE for all input/output operations as well as the algorithm computations. The use of multiple PEs was the second implementation. This required the development of crossbar and systolic interconnect modeling techniques by the Algorithm and Analysis Mapping Tools Program. This

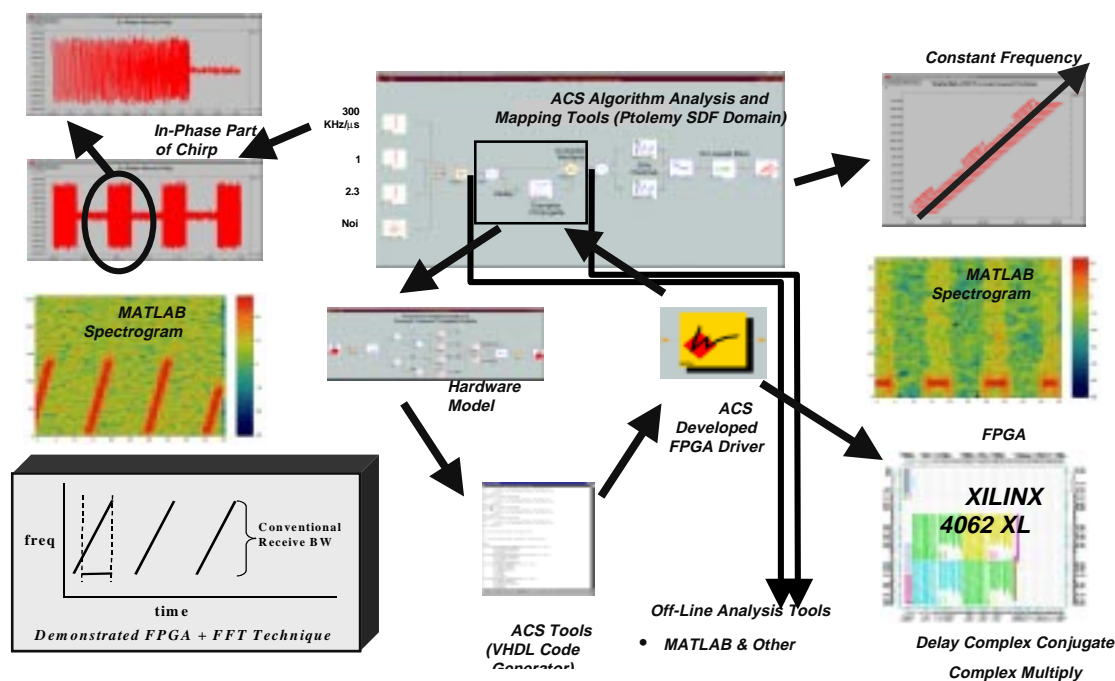


Figure 7: Linear FM Detection

presence of a tone indicated the presence of a chirped signal. The approach provided semi-coherent detection integration across the overlapped delayed and incoming signal. The algorithm was decomposed into FFT and non-FFT components. The non-FFT components were implemented in the Wildforce FPGA while for the demonstration the FFT was implemented on the workstation. The demo used 2 FFTs overlapped 50%. The FFT outputs drove a waterfall display showing the chirp rates.

implementation achieved a 40 MHz processing speed and required one clock cycle per I/Q data pair, resulting in the ability to stream data from memory on the Wildforce board at 40 MS/s. This rate far exceeded the original design goal of 25 MS/s. Actual testing therefore included running the demonstration up to the achieved rates of 35 MHz and 40 MHz, the latter also equaling the I/Q pair sample rate. The FPGA based functions were designed to handle 16 bit data. The development tools used Ptolemy “stars” (function blocks)

and the tool's scheduling capability in a synchronous data flow (SDF) domain to transfer the data to and from the Wildforce module. The Wildforce board used a driver developed by the Algorithm Analysis and Mapping Tools Program to interface with the Ptolemy environment. The size of the data block transferred for the final demonstration was chosen as 4K to demonstrate full rate processing with rapid graphic output display generation. Larger data sets (e.g. 65K) were used during the development stages for timing and validations with MATLAB and a DOD signal processing library package. These data blocks were transferred into the PE associated memories on the Wildforce board, processed at full rate and transferred out to PE memory and then returned to a file on the workstation. The demonstration transferred data in and out of workstation files using the PCI bus connector.

The components of the FPGA implementation included the delay element, a complex conjugator and a complex multiplier. The delay was set at 25 IQ samples. The minimum delay was established by providing just enough delay to make the differential tone appear around the second bin of the output FFT for signals at the slowest chirp rate. The maximum delay was established by ensuring that at the minimum expected pulse width that there was guaranteed to be at least one full set of overlapped pulse data processed by the FFT. The size of the FFT (number of points or number of input samples) and amount of overlapped FFT processing also factored into these considerations. A production system might employ a bank of different delays and replicas of the complete FPGA circuit to process a wide range of chirp rates in parallel. A production system would be expected to implement the FFT using a COTS FFT accelerator module.

The FPGA segment was initially implemented in Ptolemy stars (simulation only) to allow testing of the other Ptolemy elements of the demonstration. The Ptolemy hardware description, which used four multipliers, an adder and a subtractor as shown in Fig. 8 below, was then developed.

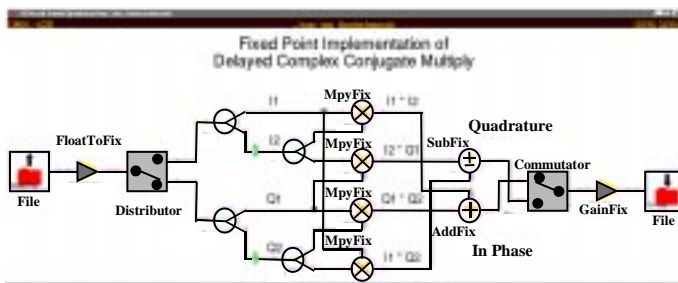


Figure 8: Ptolemy Hardware Description

The Algorithm Analysis and Mapping program ACS tools were then used to generate VHDL code from this hardware description for placing and routing on the FPGA PE(s). The Wildforce driver developed for the ACS tools was then substituted for the Ptolemy algorithm section, allowing for FPGA communication and accelerated processing.

The FFTs processed the output of the delayed and correlated incoming data stream. This output was a tone

directly related to the chirp rate multiplied by the duration of the delay. The FFT applied to the FPGA circuit output coherently integrated the resultant tone over the length of the FFT thus significantly enhancing the detection SNR over that of thresholding single incoming 25Ms/s samples. The chirp rate was assumed to be bounded but unknown. This circuit could handle signals with a range of parameters spanning pulse width, chirp rate and frequency excursion. FFTs were operated with an overlap of 50%. The overlap ensured that as long as the overlapped signal was 1.5 times as long as the collection time for the FFT then one of the FFTs would be able to fully integrate the resultant tone signal. The FFT size was chosen as 128 points for flexibility and speed of implementation and to channelize the incoming band into frequency bins suitable for chirp detection. For example, for a data rate of 25MHz, the width of the frequency bins result in 25MHz/128 or approximately 200KHz.

The waterfall display provided a spectrogram view of the processing results displaying chirp rate space as a function of time. In a production system thresholding at the FFT output would likely replace the waterfall display, but that is a system specific design issue. The ability to zoom was provided.

B. Summary of Results and Discussion

This research and demonstration explored new algorithm structures for the detection and characterization of unknown (though bounded) chirped FM signals that exploit adaptive computing coupled with a structure suited for high rate FFT processing engines. The techniques illustrated provide the capability to enhance the detection and measurement SNR by 10 to 20 dB (depending on signal pulse widths and excursions) while processing input data at real time rates up to 40 MSamples/Second - the below Figures 9 and 10 illustrate a noisy chirp signal and the resultant clear tone detection. The resultant demonstration has illustrated the performance potential of adaptive computing technology in high bandwidth data streaming applications, using an efficiently structured algorithm implemented in a hardware (FPGA) accelerated environment. This demonstration also provided a rich platform for experimentation and critical advancement of Adaptive Computing tools.

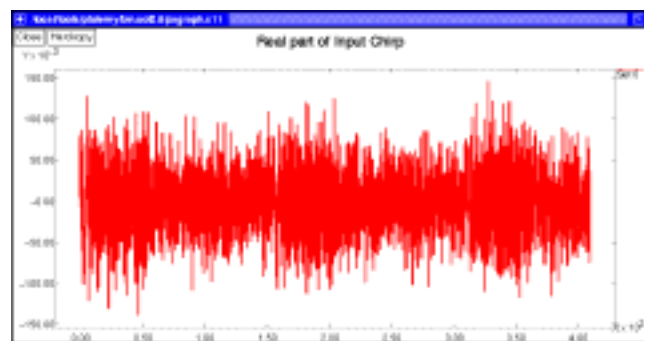


Figure 9: Input Chirp Signal (with noise)

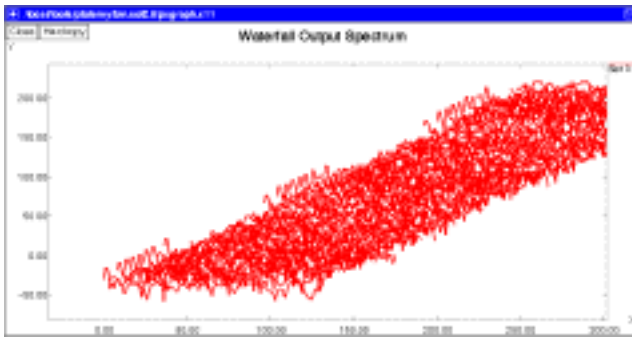


Figure 10: Tone (Chirp Signal) Detection

The single Wildforce PE (FPGA) implementation achieved a board processing speed of 35 MHz and required five clock cycles per I/Q sample pair due to the serial implementation required by the PE for all input/output operations as well as the algorithm computations. The multiple (all five on Wildforce board) PE implementation ran at 40 MHz and required one clock cycle per I/Q data pair, resulting in the ability to stream data from memory on board the Wildforce board at 40 MSamples/Second. The data size handled was 16 bit.

Ptolemy software (hardware simulation) and actual hardware implementations of the algorithm for detection of chirped FM signals were verified using MATLAB. A separate DOD signal processing package was also used in the verification process. The ability to accomplish this multiple platform validation provided high assurance of the techniques developed and validated the demonstration architecture. Using the same input chirp I/Q data set, the output values from a MATLAB simulation of the algorithm were verified to match those produced from the Ptolemy software and hardware implementations. Figures 11 and 12 below show MATLAB spectrograms of an input chirp signal and the corresponding MATLAB output simulation. The MATLAB spectrogram of the FPGA hardware output matched that of the MATLAB simulation output, showing proper hardware algorithm implementation.

Figure 11: MATLAB Spectrogram of Input Chirp Signal

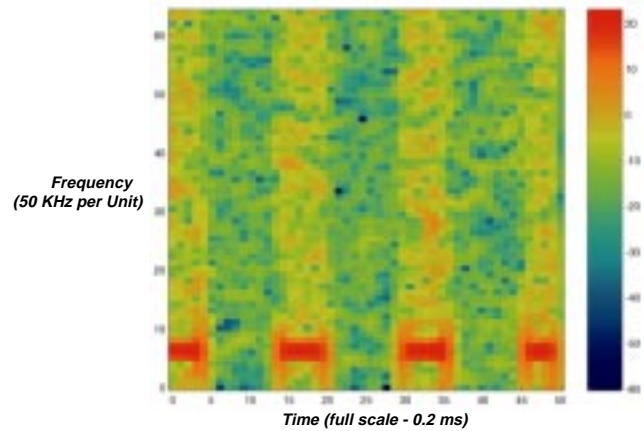
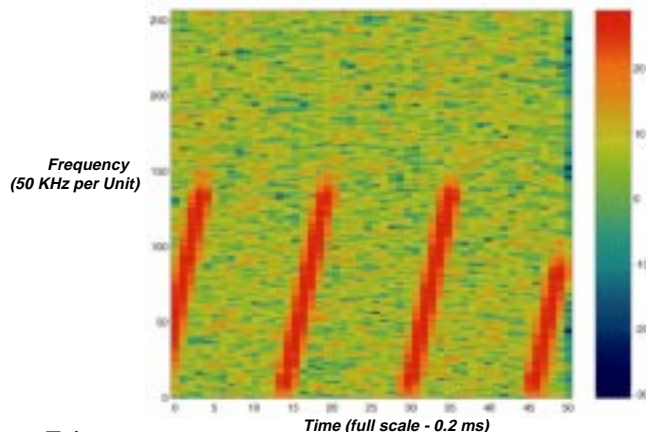


Figure 12: MATLAB Spectrogram of Output from MATLAB Simulation

C. Potential Use in an External Environment

The techniques used in this demonstration for detecting chirped signals would provide a high probability of detection for the chirped signals at relatively inexpensive processing cost. The techniques demonstrated will also provide a level of detection for phase shift keyed signals and can be adapted for some cyclo-stationary signals. This technique also provides a reasonable level of discrimination against interfering signals of other types. It does not provide a thorough means of discrimination against other signals. To achieve higher levels of false alarm discrimination additional processing may be required to remove such signals as CW signals. In general, low level short pulsed signals will not interfere since they will not integrate in the detection process. High level pulsed signals could interfere and must be considered. Inter-modulation products will be generated in a dense signal environment and must also be considered. A production system must provide the additional techniques needed to discriminate against these interfering signals. The specific kinds of discrimination to be used would be a function of the likely interferors in the signal bands of interest for the specific system. The approach demonstrated is very compatible with such techniques and can accommodate them in a production system.



VI. REFERENCES

- [1] G.W. Lank, I.S. Reed, and G.E. Pollon, "A Semicohherent Detection and Doppler Estimation Statistic", IEEE Trans. Aerospace and Electronic Systems, Vol. AES-9, pp. 151-165 (1973).
- [2] D.W. Tufts, "Adaptive Line Enhancement and Spectrum Analysis," *Proceedings of the IEEE (Lett.)*, vol. 65, pp. 169-173 (Jan. 1977).
- [3] R. Kumaresan and S.Verma, "On Estimating the Parameters of Chirp Signals Using Rank Reduction

Techniques,” Proc. of the 21st Asilomar Conference on Signals, Systems, Comp., pp. 555-558 (Nov. 1987).

- [4] D.W. Tufts, “Properties of a Simple, Efficient Frequency Tracker”, Sanders, A Lockheed Martin Co., Internal Memorandum (March 11, 1977).
- [5] W.J. Albersheim, “A Closed-form Approximation to Robertson’s Detection Characteristics”, *Proc. IEEE*, 69 (July 1981).
- [6] E.K. Pauer, P.D. Fiore, and J.M. Smith, “Algorithm Analysis and Mapping Environment for Adaptive Computing Systems: Further Results”, *Proc. IEEE*, Symposium on FPGAs for Custom Computing Machines (FCCM) (Apr. 1999).