A Method for Characterizing a Medium for the Purpose of Definining a RadioLAN Standard and for Assuring Conformance to that Standard
Part I-Introduction

by
Lourens Van Der Jagt
Knowledge Implementations, Inc.
32 Conklin Road, Warwick, NY 10990
Voice (914) 986-3492
Fax (914) 986-6441

Abstract

A conceptual framework has been developed for defining a RadioLAN PHY layer. This framework detailed in IEEE P802.11-92/4 requires the developement of a medium specification that is a companion to a PHY specification. A conformant PHY-Layer entity will meet the requirements of the IEEE 802.11 PAR when operated over a medium that meets the requirements of the companion medium specification. The geographic area over which a real world environment exhibits the characteristics defined in an IEEE P802.11 medium specification is defined as the coverage area for a specific instance of a Premises Local Area Network (PLAN). The purpose of this document is to attempt to provide a medium specification methodology with suffficent specificity to allow it to be used in the standardization process. The medium specification must be detailed enough to insure interoperability among conformant PHY-Layer entities and to provide a basis for conformance testing of those entities.

A secondary goal of this document is to provide a very specific methodology for describing media so that those in a position to obtain measured data of real world environments will perform those measurements in a way that is useful for the standardization process and uniform enough to allow for apples-to-apples comparisons between data obtained in different environments and by different investigators.

This document is an executable *Mathematica* Notebook. It has been generated without any outside funding support. We have attempted to make this notebook both accurate and complete. We welcome any input regarding the content of this notebook and would be happy for any constructive criticism of the approaches used or the theoretical underpinnings of those approaches. The cells developed for this paper have been designed for readability and hopefully, clarity. No serious attempt has been made at this time to optimize this code for speed of execution or memory efficency. This is especially evident in the sections on RF modulation.

Our primary business at KII is contract engineering and we would welcome the opportunity to obtain funding for the continuation of our efforts in this area.

Introduction

It is typical in communication systems analysis to use the concept of the complex impulse response of a medium as the basis for generating a set of parameters that characterize that medium. Examples of statistics that can be calculated are mean excess delay, rms delay spread and various attenuation oriented statistics. Using this method to describe a system allows the signal arriving at a receiver to be the convolution of the complex impulse response of the medium and the transmit signal plus any noise or interference encountered during the transmission process.

The measurement of complex impulse response can be performed by repeatedly transmitting a carrier modulated by a psuedo-noise sequence with specific periodic autocorrelation characteristics and receiving this signal after propagation on the channel with a quadrature receiver. The receiver used provides information about the projection of the arriving signal onto a locally generated Sin and Cos carrier and hence provides the ability to determine how the transmitted signal has changed in amplitude and phase as a result of propagating through the medium. The received in-phase and quadrature waveforms are processed by a correlator to provide the real and imaginary component waveforms that are commonly referred to as the complex impulse response of the medium.

The above provides a general description of a measurement process. In order for such a process to be useful for the purposes of developing a standard it is necessary to provide a detailed description of the transmit signal to be used, of the characteristics of the circuits through which that signal passes to gain access to the medium and of the characterisitcs of the circuits that exist in the receiver. The bandwidth of the measurement process is a very important parameter. In the paragraphs and cells that follow the mathematical foundation for this approach will be detailed and specific choices associated with a specific measurment scenarios will be established in future contributions.

This document is intended to provide a detailed description of the mathematics associated with a measurement of this type. This is done both to introduce those who may not be familiar to various statistics that are popular in the trade publications and to provide a common basis for documenting media within the IEEE 802.11 committee. The document describes a method for describing the transmit signal. In the first section of this document this signal is a baseband, non-bandlimited signal. This ideal signal is used to illustrate the fact that the periodic crosscorrelation between an m-sequence and an m-sequence distorted by reflections is indeed a good representation of the impulse response of a multipath channel. In the next section a method for determining the mean excess delay, rms delay spread and average power of a specific instance of an impulse response is developed. Next a description of a common data format that might be used to track conditions under which data is taken is described. Finally, a demonstration of the development of complex impulse response from a measurement of a signal that has been transmitted at RF is given. Details of how to obtain statistics from individual complex impulse response samples are also provided.

This analysis is intended to be introductory in nature. It deals strictly with obtaining statistics for one instance of transmit to receive path. It is planned that the next installment of this work will involve data reduction and characterization of statistics from a set of measurements taken in this manner. Next an installment dealing with time dependent aspects of these measurements is planned. An installment of this work dealing with the impact of adding multiple simultaneous transmitters to the analysis detailed in the previous installments would seem to be required. Finally, an installment of this work dealing with the characterization of the impact of interferers , gaussian noise, and impulsive noise sources on the channel model is required.

Preliminary Definition of Transmit Signal

To begin this process we will assume that we have a transmit signal of the form:

```
s[t_]:=Re[u[t] Exp[I 6.283185 carrierfrequency t]];
```

Where **u[t]** represents the lowpass waveform that is modulated onto a carrier of radian frequency equal to **2 Pi carrierfrequency**. The function **Re[]** returns the real portion of the complex quantity enclosed in brackets and the function **Exp[]** raises Euler's Number e to the power enclosed in brackets. **I** is the pure imaginary number complex number (0,1).

For the purposes of measuring the complex impulse response of the channel the lowpass waveform utilized will be an m-sequence of length 255. This waveform is chosen because the off peak periodic autocorrelation values are all equal to -1 making the numerical adjustments needed to get impulse response from the output of the correlator straightforward. As has already been pointed out, the bandwidth and spectral occupancy of the signal used to probe the channel are important in determining the output of the measurement process. For this reason, the shape of the transmitted signal must be tightly specified. For the purposes of clarity this issue of defining the actual shape of the transmit waveform will not be addressed in this section. Rather an ideal, non-bandlimited waveform is used during this introductory section.

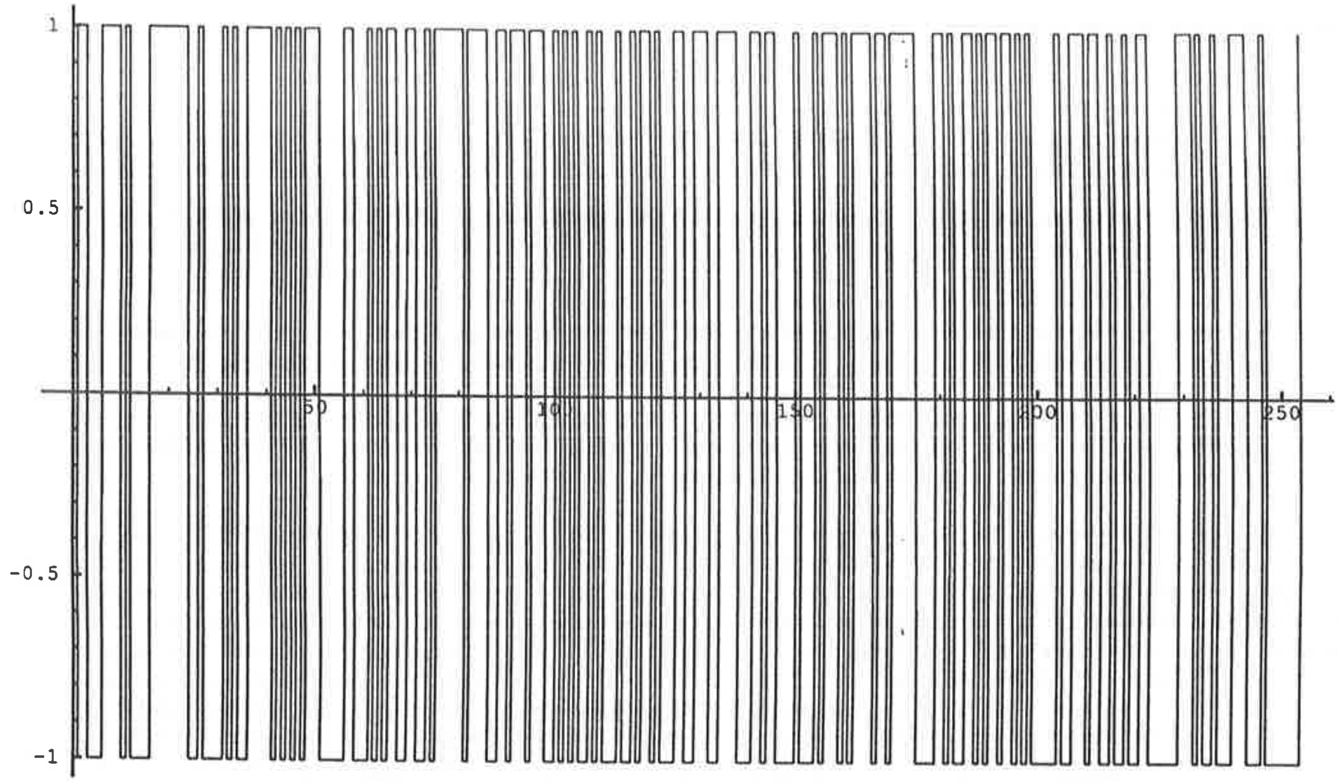
With this in mind the basic M-sequence to be transmitted is generated in the following cell. It should be noted that the function **gencode** used to generate this m-sequence is defined and documented in an Appendix. The function **u[t]** is defined in order to change the list **txsequence** into a function of time that can be sampled in future analysis. In this section all analysis will be done in the baseband and all impulse responses obtained will be real. Later in the document an RF analysis will be detailed and impulse reponses obtained will be complex. In the cell which follows some constants are itialized and the m-sequence to be transmitted is obtained. The final line of this cell sums all of the entries in the **txsequence** list as a reasonableness check. If **txsequence** is indeed an m-sequence this calulation should yield 1, which it does.

```
Clear[txsequence];
sampleperiod=1;
gencode[8,1,1,txsequence];
u[t_]:=txsequence[[window[t]]];
Apply[Plus,txsequence]
```

```
1
```

A plot of this sequence appears in the following cell.

```
Plot[u[t],{t,0,254},PlotStyle->Thickness[.001],PlotPoints->255]
```



-Graphics-

This sequence has a flat power density function and a periodic autocorrelation function that is a unit weight impulse at zero with sidelobes that are absolutely flat with a value of -1 divided by the length of the sequence. The following calculations provide verification of this. In each case the function is flat with the exception of the first point. The first 10 points of each function are plotted. Two plots of the periodic autocorrelation function are shown. The first is produced by the brute force method of using the function **crosscorr** defined in an Appendix. The second is using the fact that the inverse Fourier Series of the power density function is the autocorrelation function. This is computationally much quicker and will be the approach typically used throughout the remainder of the document. In mathematical notation this method is based on the following relationship.

$$\mathcal{FT}[f_1(t) \diamond f_2(t)] = \mathcal{F}_1[u] \bullet \mathcal{F}_2^{*}[u]$$

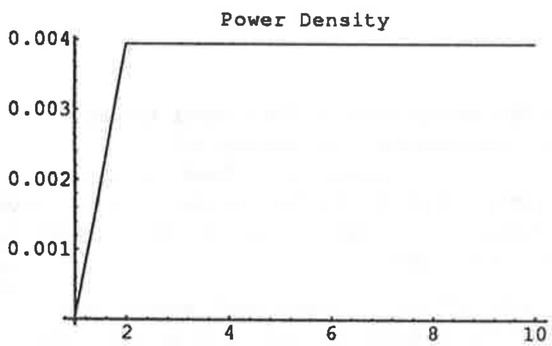
❖ Correlation ●Multiplication

★ Conjugation $\mathcal{FT}$ Fourier Transformation

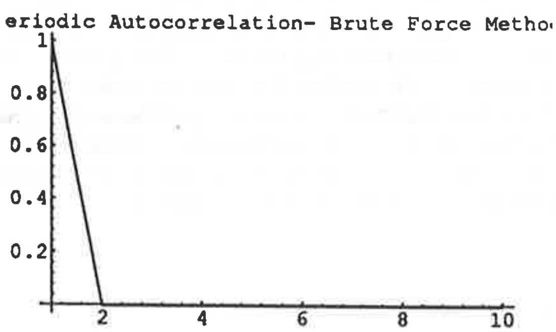
$\mathcal{F}_1[u]$ is the Fourier Transform of $f_1(t)$

The first equation creates a list that contains the sequence. The Fourier Series expansion of this list times its own the complex conjugate yields a list that represents the power spectral density function for the signal.

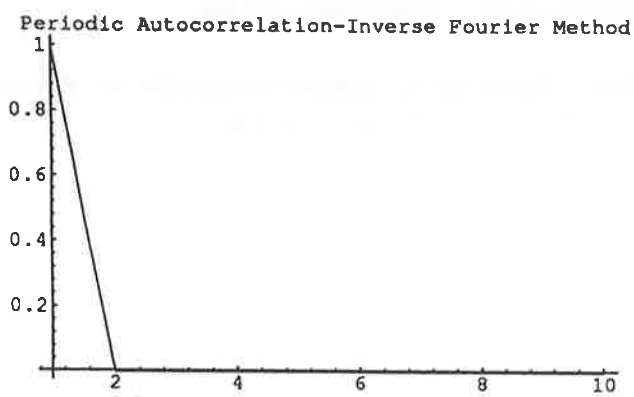
```
Table[u[t],{t,0,254}];
EEFourier[%];
powerdensity=% Conjugate[%];
ListPlot[Take[powerdensity,10],PlotRange->All,PlotLabel->"Power Density",
PlotJoined->True, AxesOrigin->{1,0}, PlotStyle->Thickness[.001]]
correlation=crosscorr[255,txsequence,txsequence];
ListPlot[Take[correlation,10],PlotRange->All,PlotLabel->"Periodic Autocorrelation- Brute
Force Method", PlotJoined->True, AxesOrigin->{1,0}, PlotStyle->Thickness[.001]]
correlation=EEInverseFourier[powerdensity];
ListPlot[Take[correlation,10],PlotRange->All,PlotLabel->"Periodic Autocorrelation-Inverse
Fourier Method", PlotJoined->True, AxesOrigin->{1,0}, PlotStyle->Thickness[.001]]
```



-Graphics-



-Graphics-



-Graphics-

Other characteristics of the **txsequence** generated above are an average power of 1. This can be verified by taking the sum of all of the elements of the **powerdensity** list. The same value is also the zero delay (first) element of the **correlation** function list. The next cell provides verification of these relationships.

```
averagepower=N[Re[Apply[Plus,powerdensity]]]
correlation[[1]]
```

1.

1.

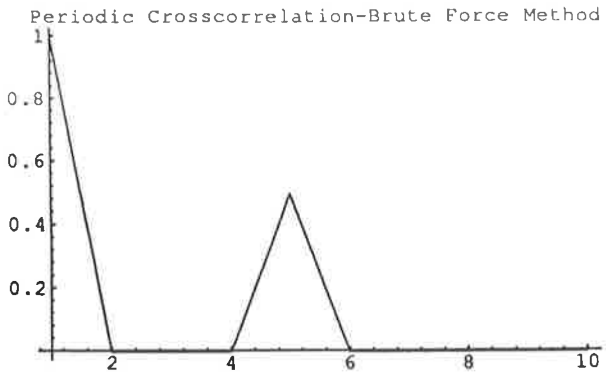
As mentioned above, the baseband signal **u[t]** is not a real world signal and this analysis needs to address the impact of bandlimiting on the signal that will ultimately be transmitted. Before moving on to that analysis, however, a brief review of the impact of multipath on an ideal signal will be presented and examples of this impact will be provided. This preliminary presentation of the impact of multipath is made in the interest of clarity.

■ Multipath DistortionEffects

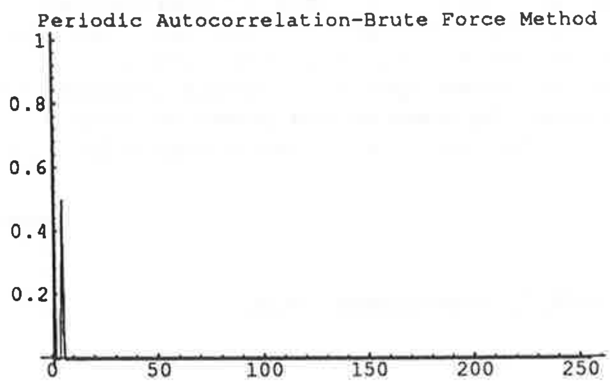
The multipath distortion model of a typical indoor radio environment is known to be complicated and probably best described in terms of a representative group of experimentally obtained complex impulse response samples. In order to illustrate the impact of multipath on the measurement process being described in this document a set of simple examples is presented here. This approach is taken in order to give the reader an intuitive feel for the method and the results that are obtained.

To start off, a signal will be generated that is the sum of **u[t]** generated above with another copy of itself at 1/2 the amplitude and shifted by 4 time units. This will be crosscorrelated with the original **txsequence**. The result will show that the original single impulse is now two impulses separated by 4 samples. The second impulse will be 1/2 the amplitude of the first impulse. The following cells implement this analysis. First the multipath affected signal is generated and stored in **multitxsequence**. Then the brute force cross correlation is generated for this signal. In order to illustrate the validity of the computational approach, the Inverse Fourier Method is also performed. The results show that the two approaches are indeed identical as theory dictates. From here on the brute force method will be abandoned in favor of the computationally more efficent inverse Fourier approach for calculating impulse responses from received signals.

```
multitxsequence=Table[u[t]+.5  u[t-4],{t,0,254}];

multicorrelation=crosscorr[255,multitxsequence,txsequence];
ListPlot[Take[multicorrelation,10],PlotRange->All,PlotLabel->"Periodic
Crosscorrelation-Brute Force Method", PlotJoined->True, AxesOrigin->{1,0},
PlotStyle->Thickness[.001]]
ListPlot[multicorrelation,PlotRange->All,PlotLabel->"Periodic Crosscorrelation-Brute Force
Method", PlotJoined->True, AxesOrigin->{1,0}, PlotStyle->Thickness[.001]]
```

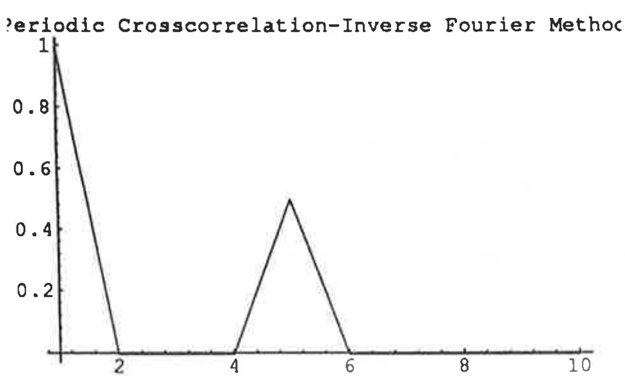


-Graphics-

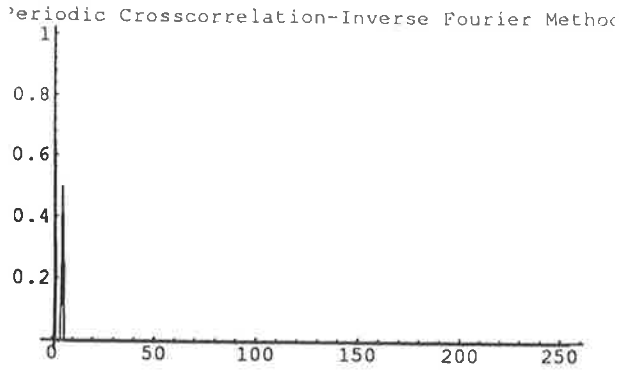


-Graphics-

```
EEFourier[txsequence];
EEFourier[multitxsequence];
fouriermulti=EEInverseFourier[% Conjugate[%%]];
ListPlot[Take[fouriermulti,10],PlotRange->All,PlotLabel->"Periodic Crosscorrelation-Inverse Fourier Method", PlotJoined->True, AxesOrigin->{1,0}, PlotStyle->Thickness[.001]]
ListPlot[fouriermulti,PlotRange->All,PlotLabel->"Periodic Crosscorrelation-Inverse Fourier Method", PlotJoined->True, AxesOrigin->{1,0}, PlotStyle->Thickness[.001]]
```



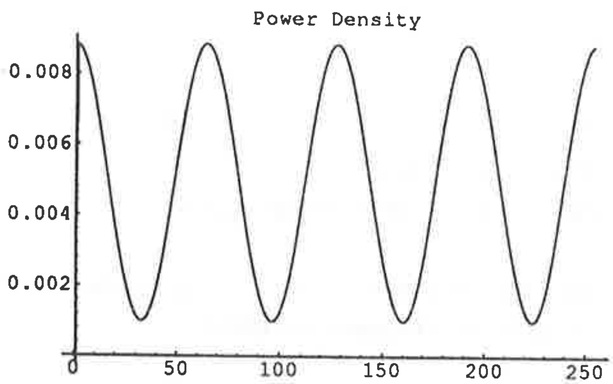
-Graphics-



-Graphics-

Once reflections are introduced into the situation the impulse response is represented by the periodic cross correlation of the received signal with the pure m-sequence. The power density of the signal is represented by the autocorrelation. The following cells illustrate the impact of this single reflection on the power spectral density function of the signal. The cell also calculates the average power by summing the power density components. As anticipated this value is the sum of the power in the main signal(1) plus the power in the reflection (.5 x.5 =.25).

```
EEFourier[multitxsequence];
powerdensity=% Conjugate[%];
ListPlot[%,PlotRange->All,PlotLabel->"Power  Density", PlotJoined->True,
AxesOrigin->{1,0}, PlotStyle->Thickness[.001]]
N[Apply[Plus,powerdensity]]
```

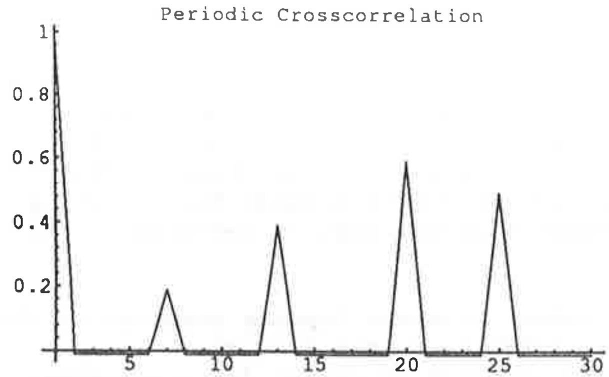


-Graphics-

1.24608 + 0. I

The single reflection situation examined in the previous few cells can be generalized to an n-ray model situation. The following cell generates a 5-ray signal and illustrates the impulse response that would apply to this situation.

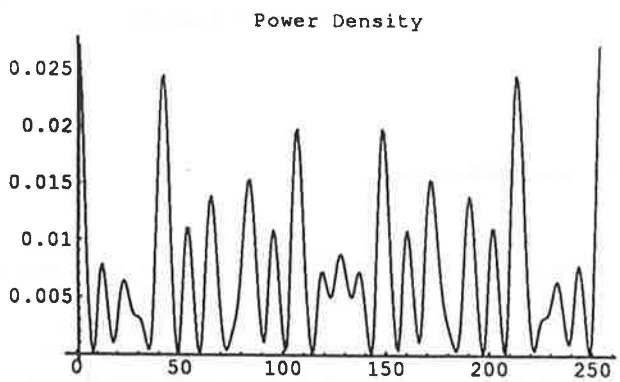
```
multitx5=Table[u[t]+.2  u[t-6]+.4  u[t-12]+.6  u[t-19]+.5  u[t-24],{t,0,254}];
EEFourier[multitx5];
EEFourier[txsequence];
fourraydensity=%% Conjugate[%];
autocorr=EEInverseFourier[fourraydensity];
ListPlot[Take[autocorr,30],PlotRange->All,PlotLabel->"Periodic  Crosscorrelation",
PlotJoined->True, AxesOrigin->{1,0}, PlotStyle->Thickness[.001]]
```



-Graphics-

The power density function of the resulting signal can also be calculated and is illustrated in the following cell. Again the average power can be found by Parsevals Theorem and is as expected 1*1+.2*.2+.4*.4+.6*.6+.5*.5=1.81. The variation between 1.81 and 1.78851 that is calculated is caused by the non-zero off peak value of the autocorrelation function associated with an m-sequence.

```
EEFourier[multitx5];
powerdensity=% Conjugate[%];
ListPlot[%,PlotRange->All,PlotLabel->"Power Density", PlotJoined->True,
AxesOrigin->{1,0}, PlotStyle->Thickness[.001]]
N[Re[Apply[Plus,powerdensity]]]
```



-Graphics-

1.78851

This concludes the section of the document that illustrates the generation of impulse responses from ideal transmit and receive signals where the received signals are only distorted by multipath. In the next section, the final impulse response calculated in this section will be used to demonstrate the generation of mean excess delay and rms delay spread statistics from an individual impulse response sample.

■ Statistics Derived From Impulse Responses

The analysis of autocorrelation, power spectral density, and impulse response detailed above provides a means of obtaining raw measurements of a channel that can be used to derive statistics that are considered useful. In particular, the mean delay spread, the rms delay spread, and attenuation characteristics of the channel can be obtained from these measurements. This section of this document will deal with methods for obtaining these statistics given a measurement of the equivalent baseband impulse response of the channel.

To begin with we have a list representing the equivalent baseband impulse response of the channel. For the purposes of this analysis this is taken to be the periodic crosscorrelation between the received signal and the transmitted signal (an m-sequence). This can be calculated by the function **crosscorr** or through the Inverse Fourier Transform Method described earlier. In order to calculate the mean delay spread of the channel we take the list that represents the impulse response of the channel, square it and sum it to obtain the total arriving average power. When dealing with actual measured results we must establish a threshold under which all signals will be squelched. This value will be called the **cutoffvoltage**. The setting of this cutoff voltage has a significant impact on the results of data reduction. A **cutoffvoltage** that is set low will result in longer mean excess delay and rms delay spread. The question of whether this should be a fixed receive level or a function of the level below the highest peak also has an impact. For the purposes of this analysis a combination of these two approaches for setting the cutoff level has been taken. The **cutoffvoltage** will be either **peaktoreflectionlimit** dB down from the largest voltage sample in a receive sequence or **receiversquelchlimit** in dBmv whichever is greater. The setting of this threshold is accomplished by the following cell. Note the impulse response that is being examined is the impulse response named **autocorr** that is left over from the previous section of this document.

```
peaktoreflectionlimit=-30;
receiversquelchlimit=-80;
cutoffvoltage=Max[N[10^(peaktoreflectionlimit/20) Max[autocorr]],
N[10^(receiversquelchlimit/20) .001]];
```

The calculation of mean excess delay and rms delay spread is accomplished by the following cell. To begin with, a list of all of the positions in the impulse response list that exceed the parameter **cutoffvoltage** is generated. A detailed list of the sample numbers for which this is true is stored in the variable **spread**. Given this list a table named **samples** is generated that is a list of sample numbers starting from 1 and incrementing to the total number of samples between when the data first exceeds the **cutoffvoltage** until it falls below the **cutoffvoltage** for the final time. The value of the impulse response function is then squared and summed over the range between the lowest position in **spread** to the highest position. This represents the total average power in the arriving signal for this particular instance of transmission. This range of samples will be referred to as the impulse response duration in subsequent descriptions. The **meanexcessdelay** is calculated by taking the square of all of the values in impulse response duration and weighting them with their corresponding sample number from the **samples** table and dividing by the **totalavepower** calculated earlier. Finally, the **rmsdelayspread** is calculated by weighting the square of the values in the impulse response duration by the square of the **samples** table number and summing those values. This weighted sum is then divided by the total energy and the second moment calculation is performed. In mathematical notation this calculation looks as follows. The value of **range** is the number of samples in the impulse response duration. The variable τ_k is the sample number within the impulse response duration. The variable α_k is the value of the impulse response at that location within the impulse response range.

Mean Excess Delay ($\bar{\tau}$) RMS Delay Spread (σ)

$$\bar{\tau} = \frac{\sum_{k=1}^{range} \tau_k \alpha_k^2}{\sum_{k=1}^{range} \alpha_k^2} \qquad \sigma = \sqrt{\overline{\tau^2} - (\bar{\tau})^2} \qquad \overline{\tau^2} = \frac{\sum_{k=1}^{range} \tau_k^2 \alpha_k^2}{\sum_{k=1}^{range} \alpha_k^2}$$

Calculating these statistics for the impulse response stored in **autocorr** yields the following results.

```
spread=Position[autocorr,x_/;x>cutoffvoltage];
samples=Table[i,{i,Min[spread],Max[spread]}];
totalavepower=Apply[Plus,Take[autocorr^2,{Min[spread],Max[spread]}]];
Apply[Plus,samples  Take[autocorr^2,{Min[spread],Max[spread]}]];
meanexcessdelay=N[%/%%]
Apply[Plus,samples^2  Take[autocorr^2,{Min[spread],Max[spread]}]];
rmsdelayspread=N[Sqrt[%/totalavepower-meanexcessdelay^2]]
```

9.20879

9.80086

Having established a calculation method for mean excess delay, rms delay spread, and total energy, attention will now be placed on the definition of how to calculate attenuation distance associated with a particular measurement set.

In the previous sections of this document the average power in an arriving signal was found by summing the elements of the power spectral density function. This number, calcuated to be 1.78851 over the full length of a transmit sequence period should be very close to the number obtained in the previous calculation of **totalavepower**. There will be minor discrepancies in these numbers as a result of the fact that the calculation method used in this section squelches all power that falls outside the region considered to be the part of the impulse response duration. The number calculated here is:

```
totalavepower
```

1.76963

This is in good agreement with earlier calculations. In order to determine attenuation distance between transmitter and receiver it is necessary to develop a reference against which the power arriving at the receiver is compared to determine attenuation distance. In the literature this reference is typically an individual measurement taken in the proximity of the transmit antennae at a location just outside of the near field. A distance that has been used by some investigators is 10 wavelengths at the carrier frequency. In general the choice of this location and the impact of antennae directivity will result in a lack of consistancy between the measurements of a particular environment if the reference values are not tightly specified. If the antennae is not of the directed antennae type, this is probably not a problem as there will be sufficent variation between different geographies to overshadow the impact of this inconsistancy. In the directed antennae case, however, the situation is probably not so straightforward. As a means of making progress on this issue, it is suggested that directed antennae be ignored for the present time, and an average of five measurements (north,south,east,west and above) taken 10 wavelengths from the transmitting attennae be used as a reference. Once this decision has been reached the average power of a sample can be calculated by any of the methods listed in previous sections.

Because of the impact of multipath it is important that data be gathered on both a macroscopic scale and a microscopic scale. Both theory and research shows that there are significant variations in impulse response and hence statistics derived from impulse response when the position of the receiver and transmitter are changed by frations of wavelengths. Data that provides only averages over a number of samples taken at microshifted locations in the vicinity of a macroscopic test site tend to obscure these variations. The RadioLAN hardware may or may not be able to take advantage of averaging a number of geographically diverse measurements in operation. For this reason it is critical that data be collected in a form that will allow both macroscopic and microscopic analysis. A future installment of this document that deals with combining the results of individual measurements statistics to form environmental statistics will require this type of data. At this time, however, it seems appropriate to specify a common data format for measurements and with this format a technique for imbedding location and other critical information into the data format.

■ Common Data Format

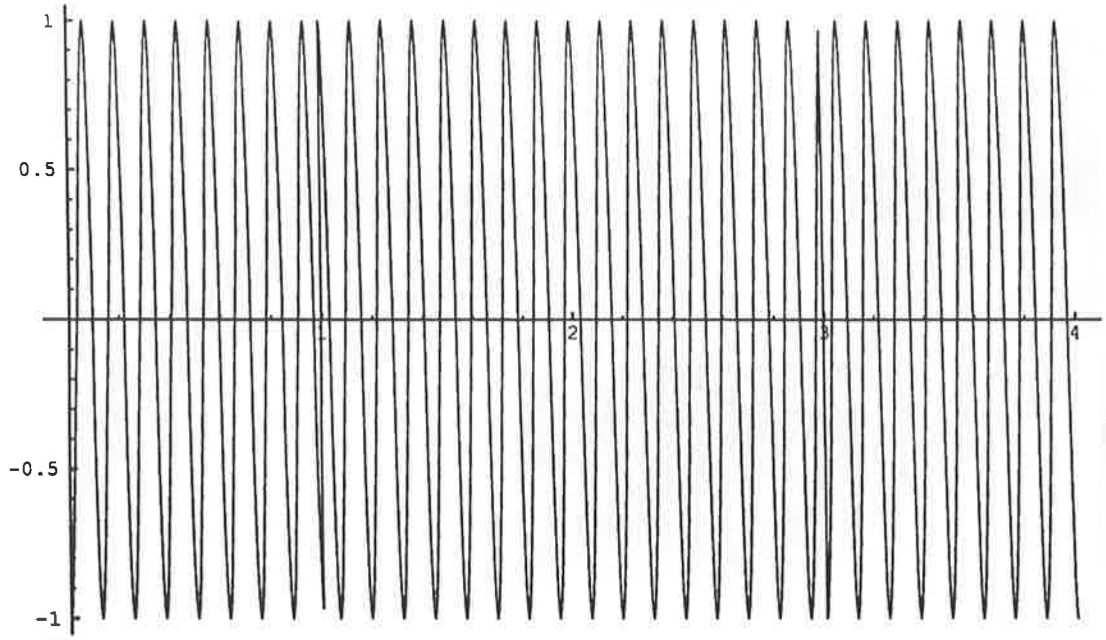
It is suggested that a data sample convention be adopted for both naming data files and for imbedding critical measurement parameters into those data files. This convention should include cartesian coordinates on two scales, the macroscopic scale and the microscopic scale. While in principle a single cartesian triplet of sufficent accuracy should be able to suffice for both cases, in practice a measurement can more easily be characterized by a gross measurement that is accurate to within a few meters, and a fine measurement that is accurate to within a fraction of a wavelength measured relative to a point chosen for the gross measurement. The following information should be appended to the raw data to make it useful. The current proposal is to append seven lists as detailed below to the list of data values. The lists to be appended are:

1) units of data - {bits or volts} if bits provided resolution and full scale range
2) time per sample - {T} time between samples in nanoseconds
3) transmit sequence - {list gencode parameters to get sequence}
4) macroscopic distance - {x,y,z} in approximate meters
5) microscopic distance - {x,y,z} relative to macroscopic distance in cm
6) time - {t} time sample was started in milliseconds if possible
7) geography description (optional) - {"ascii"} a description site being measured

■ RF Analysis and Introduction of Complex Impulse Response

The process of moving from baseband to passband signals introduces an additional dimension to the analysis of channel characteristics. The transmit signal is used to modulate a sinusoidal carrier. Without multipath impairments the signal propagates through space with the phase of the signal being strictly a function of time and the radian frequency of the carrier. A signal that has propagated without impairment will arrive at the receiver with some phase shift relative to the local oscillator at the receiver. This phase shift may result in a degree of cancellation when the signal is down converted and filtered to reproduce the baseband signal. The signal lost to this cancellation can be recovered by down converting with two local oscillators shifted from each other by 90 degrees. These are called the in-phase and quadrature local oscillators. The amount of signal arriving in the detectors driven by each oscillator is a function of the relative phase of the local oscillator with respect to the transmit oscillator and the amount of phase precession that has occurred as a result of the transit delay of the medium. For the purposes of illustration a carrier frequency of 8 times the sample rate will be used in the following analysis. The modulation scheme used in this analysis will be Binary PSK without filtering or waveshaping. This is chosen strictly for ease of understanding. The actual waveshape chosen for measurements may differ from this. A small sample of the transmitted waveshape is illustrated by the following cell.

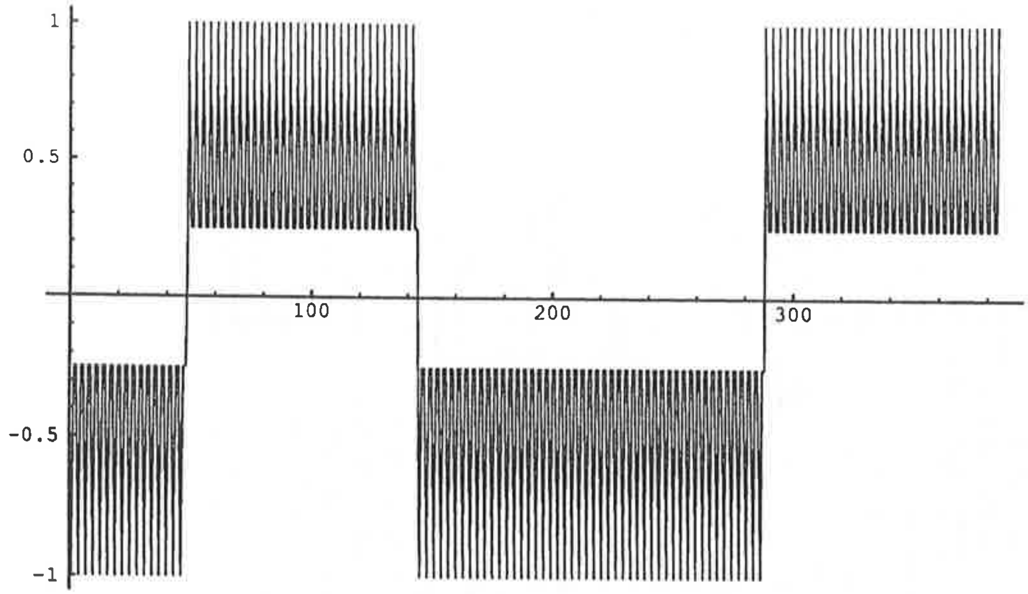
```
carrierfrequency=8;
Plot[s[t],{t,0,4},PlotStyle->Thickness[.001]]
```



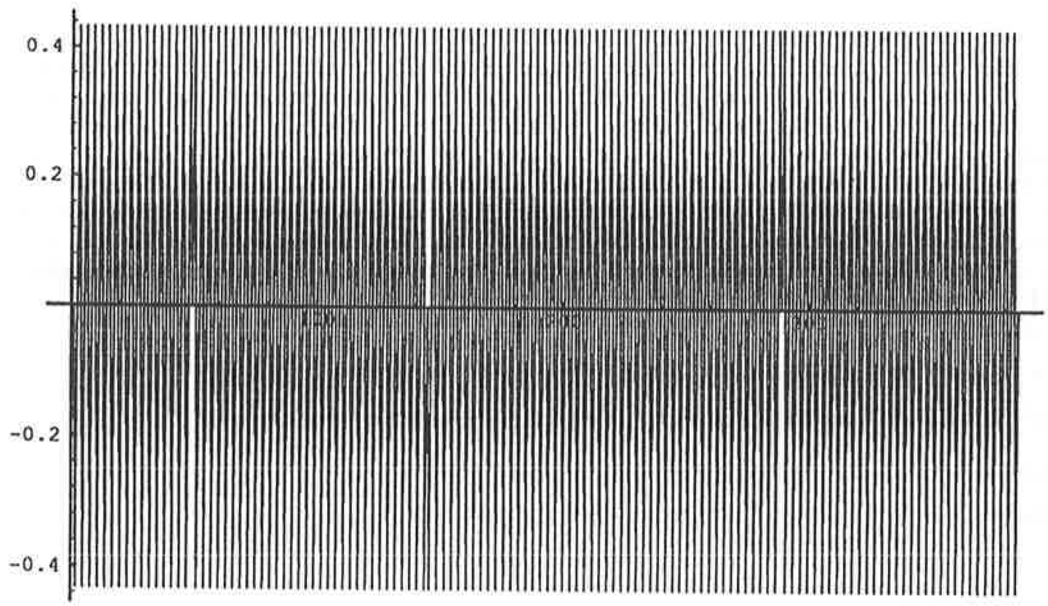
-Graphics-

One method to demodulate this signal is to multiply it by a local oscillator of frequency equal to the carrier frequency and to low pass filter the twice frequency components out of the resulting waveform. The cells that follow generated two tables that represent the sampling of the output of the demodulators that multiply by the Sin and Cos local oscillators. The signal **s[t]** chosen as the received waveform in this case is one that would exist if the receiver was hardwired to the transmitter through a distortionless link of zero length (or length of an integral multiple of the carrier frequency). For this reason all of the arriving signal is demodulated by the Cos demodulator. An abbreviated example of the results for a propagation path of non-integral wavelength length is shown after this first example.

```
signalI=Table[s[t] Cos[2 Pi carrierfrequency t],{t,0,8,1/(6 carrierfrequency)}];
signalQ=Table[s[t] Sin[2 Pi carrierfrequency t],{t,0,8,1/(6 carrierfrequency)}];

ListPlot[signalI,PlotJoined->True,PlotStyle->Thickness[.001]]
ListPlot[signalQ,PlotJoined->True,PlotStyle->Thickness[.001]]
```



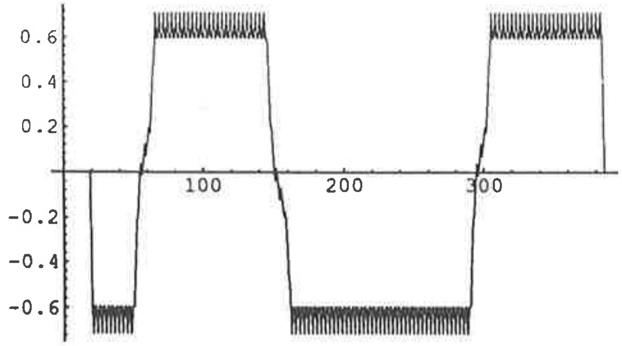
-Graphics-



-Graphics-

The final stage of the demodulation process is to low pass filter the results generated above to reproduce the equivalent baseband signal **u[t]** that was originally modulated onto the carrier. The following cells illustrate the result of this process.

```
filter[signalI]
ListPlot[output,PlotJoined->True,PlotStyle->Thickness[.001],PlotRange->All]
```



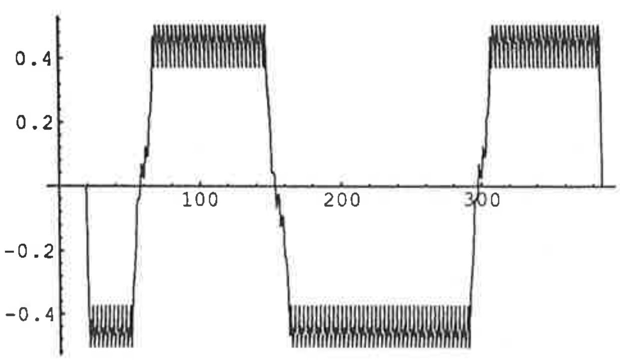
-Graphics-

```
filter[signalQ]
ListPlot[output,PlotJoined->True,PlotStyle->Thickness[.001],PlotRange->{-.707,.707}]
```

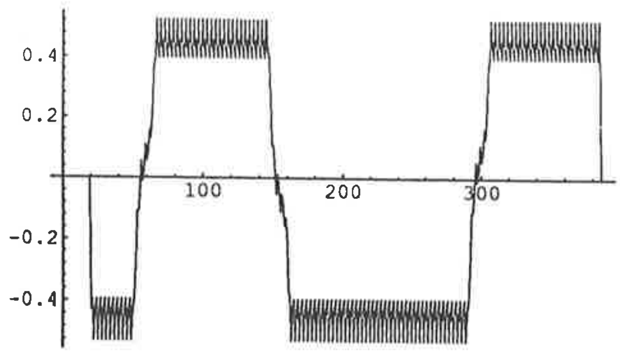
-Graphics-

The next cell performs the same example detailed above with a propagation time that causes 1/2 of the arriving signal to arrive in the Cos demodulator and 1/2 to arrive in the Sin demodulator.

```
signalI=Table[s[t-.015625] Cos[2 Pi carrierfrequency t],{t,0,8,1/(6 carrierfrequency)}];
signalQ=Table[s[t-.015625] Sin[2 Pi carrierfrequency t],{t,0,8,1/(6 carrierfrequency)}];
filter[signalI]
ListPlot[output,PlotJoined->True,PlotStyle->Thickness[.001],PlotRange->All]
filter[signalQ]
ListPlot[output,PlotJoined->True,PlotStyle->Thickness[.001],PlotRange->All]
```



-Graphics-



-Graphics-

Given the fact that when the **u[t]** is modulated onto a carrier and demodulated back to an equivalent baseband signal it arrives with components in both the In-phase (Cos) and Quadrature (Sin) channels, it is useful to view the arriving signal as a complex one in which the the In-phase component is the real portion and the quadrature component is the imaginary portion. In this representation then the demodulated signal is defined as **signalI + I signalQ**. With this representation, the analysis detailed above with respect to impulse response remains basically intact with the only change being that quantities that were real before now become complex.

By way of illustration, a signal that is the RF equivalent of the baseband signal **multitxsequence** analyzed above can be generated. This is a signal with one reflection equal in magnitude to the main signal, occurring with an offset of 4 chips of the m-sequence. This signal is generated at **carrierfrequency** by the following cell. Also generated in this cell is a table that contains the m-sequence with 48 samples per chip so that it has an equivalent sample rate to the sampled RF signal. The first cell generated the output of the in-Phase and quadrature demodulators.

```
multitxsequenceI=Table[(s[t]+s[t-4])Cos[2 Pi carrierfrequency t],{t,0,255-(1/(6
carrierfrequency),1/(6 carrierfrequency)}];
multitxsequenceQ=Table[(s[t]+s[t-4])Sin[2 Pi carrierfrequency t],{t,0,255-(1/(6
carrierfrequency),1/(6 carrierfrequency)}];
```

These signal are saved in files named **Icomponent** and **Qcomponent** in the interest of saving time and memory.

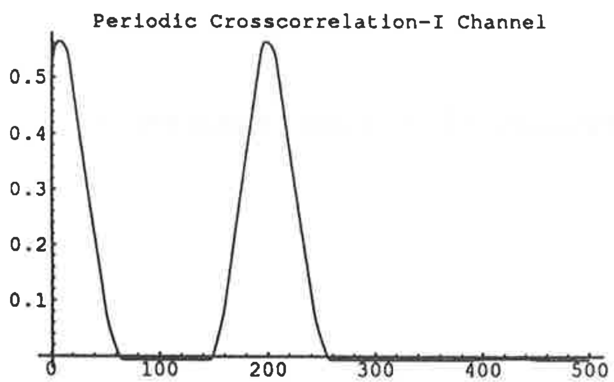
```
multitxsequenceI>>Icomponent;

multitxsequenceQ>>Qcomponent;

<<Icomponent;
multisequenceI=%//N;

<<Qcomponent;
multisequenceQ=%//N;
```

The 48 sample per chip version of the transmit m-sequence is generated by the next cell.

```
txsequencelong=Flatten[txsequence/.{1->Table[1,{a,0,47}],-1->Table[-1,{b,0,47}]}];
```

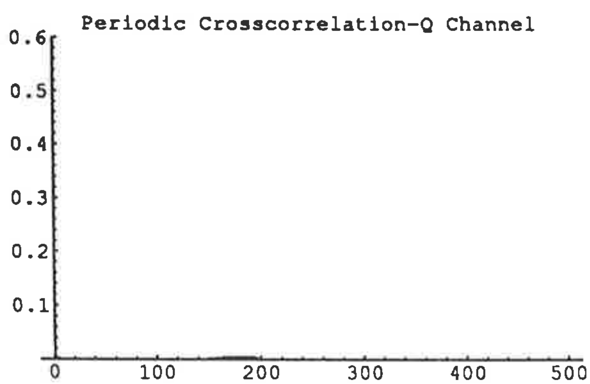
The main calculation can now be performed by filtering the output of the I channel and performing the inverse fourier method to generate the impulse response. The same process is repeated for the Q channel. It can be noted from the results that when the reflection arrives exactly at time 4 it is in-phase with the receive Cos oscillator and hence no component is noted in the Q dimension. The following cells and plots illustrate these calculations and results.

```
filter[multisequenceI];

ttt=EEFourier[txsequencelong];

EEFourier[output];
fouriermultI=EEInverseFourier[% Conjugate[ttt]];
ListPlot[Take[fouriermultI,500],PlotRange->All,PlotLabel->"Periodic Crosscorrelation-I Channel", PlotJoined->True, AxesOrigin->{1,0}, PlotStyle->Thickness[.001]]
```



-Graphics-

```
filter[multisequenceQ];

EEFourier[output];

output>>out;

fouriermultQ=EEInverseFourier[% Conjugate[ttt]];

ListPlot[Take[fouriermultQ,500],PlotRange->{0,.6},PlotLabel->"Periodic Crosscorrelation-Q Channel", PlotJoined->True, AxesOrigin->{1,0}, PlotStyle->Thickness[.001]]
```

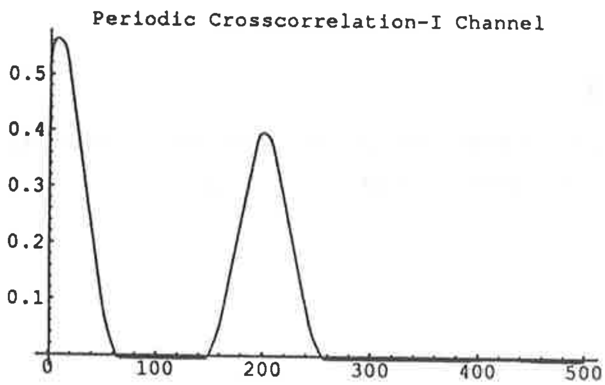


-Graphics-

Finally, in order to see the impact of delay times that are not in phase with the Cos oscillator (the condition that causes the impulse resposne to become complex) another set of signals is generated with the reflection arriving -45 degrees in phase to the Cos oscillator. The impact is illustrated in the following cells and plots.

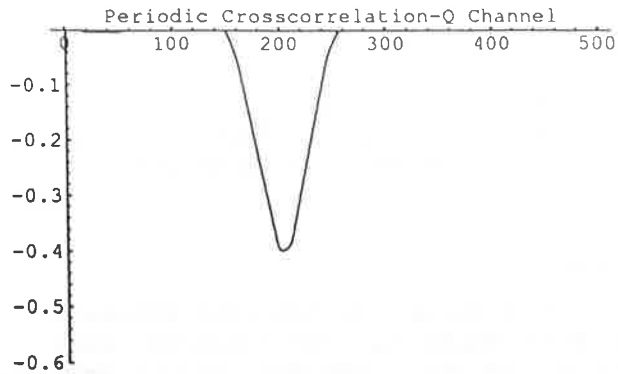
```
carrierfrequency=8;
multitxsI=Table[(s[t]+s[t-3.984375]) Cos[2 Pi carrierfrequency t],{t,0,255-(1/(6 carrierfrequency)),1/(6 carrierfrequency)}];
```

```
multitxsQ=Table[(s[t]+s[t-3.984375]) Sin[2 Pi carrierfrequency t],{t,0,255-(1/(6 carrierfrequency)),1/(6 carrierfrequency)}];

multitxsI>>Icomp1;

multitxsQ>>Qcomp1;

<<Icomp1;
multisequenceI=%//N;

<<Qcomp1;
multisequenceQ=%//N;

txsequencelong=Flatten[txsequence/.{1->Table[1,{a,0,47}],-1->Table[-1,{b,0,47}]}];

filter[multisequenceI];

ttt=EEFourier[txsequencelong];
Conjugate[ttt]>>ttconj;

EEFourier[output];
fouriermultI=EEInverseFourier[% Conjugate[ttt]];

Take[fouriermultI,500]>>Iplotout;

ListPlot[Take[fouriermultI,500],PlotRange->All,PlotLabel->"Periodic Crosscorrelation-I Channel", PlotJoined->True, AxesOrigin->{1,0}, PlotStyle->Thickness[.001]]
```



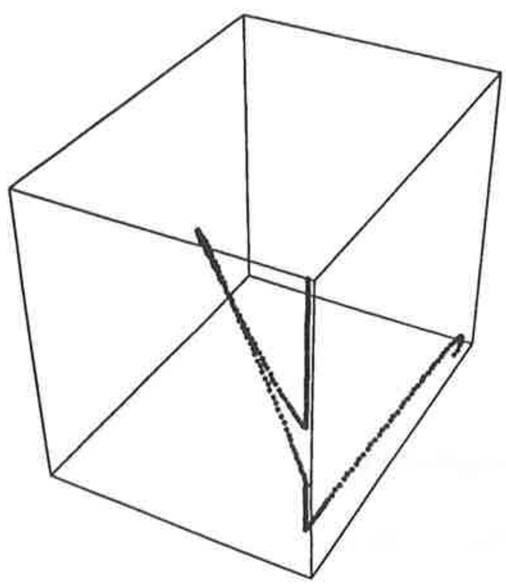
-Graphics-

```
filter[multisequenceQ];

EEFourier[output];

fouriermultQ=EEInverseFourier[% Conjugate[ttt]];
Take[fouriermultQ,500]>>Qplotout;

<<Qplotout;

ListPlot[%%%,PlotRange->{-.6,0},PlotLabel->"Periodic Crosscorrelation-Q Channel", PlotJoined->True, AxesOrigin->{1,0}, PlotStyle->Thickness[.001]]
```



-Graphics-

```
<<Iplotout;
t1=%;

<<Qplotout;
t2=%;

<<Graphics`Graphics3D`

t3=Table[N[{t2[[k]]  10,t1[[k]]  10,k/100}],{k,1,500}];

ScatterPlot3D[t3]
```



-Graphics3D-

■ Conclusion

In the analysis detailed above a basic introduction to the process of generating and analyzing complex impulse response measurements has been given. As noted in the introduction this is only the start of the required analysis, but represents the basics upon which all later work must rest. It is my hope to be able to provide additional contributions that elaborate on this analysis, should the committe find this work useful.

Appendix

This appendix contains a number of function definitions used in the body of the text. Since these definitions are unrelated to the topic under discussion they have been relegated to an appendix. Items in this appendix are flagged as initialization cells in order to assure that they have been defined prior to their use in the body of the document.

Definition of Gencode Function for Generating M-sequences

The function **gencode** is designed to generate m-sequences using a method that simulates the modular shift register approach to generating these sequences. The function takes four parameters. The first parameter is the length of the shift register used to generate the code. The second parameter is an integer number that will be used to select the specific code of a given length which will be generated. There are a limited number of codes for each length. These numbers are 1 for length 2, 1 for length 3, 1 for length 4, 3 for length 5, 3 for length 6, 9 for length 7, and 8 for length 8. The third parameter is an integer number representing the phase of the code that is desired. There are 2^n-1 phases possible for each code where n is the number of shift registers stages. The phase number given in this parameter is converted into a binary number and used as the preload into the registers. The fourth parameter is a string that will be the name of the new code. This parameter should not have a value before the function is used to assign a value to it. When the function is evaluated the code desired is assigned to the name parameter and returned in the Out vector. The actual code for this function is as follows.

```
gencode[length_,invocation_,phase_,name_]:=
(taps={{1,0},{0,1,0},{0,0,1,0},
  {{0,0,1,0,0},{1,1,1,0,0},{1,0,1,1,0}},
  {{0,0,0,0,1,0},{1,0,0,1,1,0},{1,0,1,1,0,0}},
  {{0,0,0,0,0,1,0},{0,0,0,1,0,0,0},{0,0,0,1,1,1,0},
  {0,0,1,1,1,0,0},{1,1,0,0,1,0,0},{1,0,1,0,0,1,0},
  {1,1,0,0,1,0,0},{1,1,1,0,1,1,0},{0,1,1,1,1,1,0}},
  {{0,0,0,1,1,1,0,0},{0,1,1,0,1,0,0,0},{0,1,1,0,0,1,0,0},
  {0,0,1,0,1,0,1,0},{0,1,1,0,0,0,1,0},{1,1,0,0,0,0,1,0},
  {1,1,1,0,0,1,1,0},{0,1,0,1,1,1,1,0}}};
name=Table[t,{t,1,2^length-1}];
uphase=phase;
state=Table[p=Floor[uphase/(2^(length+1-k))];
  uphase=uphase-Floor[uphase/(2^(length+1-k))]*(2^(length+1-k));
  p,{k,2,length+1}];
Do[If[state[[-1]]==1,state=Mod[state+taps[[length-1]][[invocation]],2];
   state=RotateRight[state,1],state=RotateRight[state,1]];
   name[[i]]=(2  state[[-1]]-1),{i,1,2^length-1}];
name)
```

■ Definition of Cross Correlation Function

The function "crosscorr" takes the cross correlation of two codes and produces the cross correlation of those codes. The cross correlation is code2 against code1 presented twice. In other words, the output vector, corrvector represents the correlation of code1 against code2 and all shifts of code1 against code2. The parameter samples which is the first parameter is the number of shifts that you want to see. If the sample number is less than the full sequence length the output vector is padded with zeros. If code1 and code2 are not the same length the longer code should be code1 in order to see all of the possible cross correlation characteristics. Note, samples must be equal to or less than the length of code1 or else error will be reported. This function has not been optimized and takes some time to execute. In general, in the above document, the fact that the inverse Fourier transform of the power density function yields the autocorrelation function is used to calculate autocorrelations.

```
crosscorr[samples_,code1_,code2_]:=
(tlength=Length[code1];
clength=Length[code2];
tcode1=code1;
tcode2=code2;
corrvector=Table[0,{t1,1,tlength}];
temp=Table[0,{t1,1,tlength}];
Do[corrvector[[t1]]=Sum[tcode1[[k1]]  tcode2[[k1]],{k1,1,clength}];
 tcode1=RotateLeft[tcode1,1],{t1,1,samples}];
corrvector/clength)
```

■ Definition of EEFourier Functions

The function "EEFourier" adapts the Fourier Transform Algorithm that comes standard with the ***Mathematica*** package to be the discrete Fourier Transform common to electrical engineering. In particular, the list that is generated by the function Fourier is divided by a factor of Sqrt[n]. This provides the proper scaling for Parseval's Power Theorem to work and for the first item in the Fourier Transform list to be the average value of the signal. All of this scaling is based on the assumption that the signal is a voltage across a resistor of value 1 ohm. The fact that the *Mathematica* Fourier transform uses the opposite sign in the coefficent of the exponential is thought to be irrelevant because of the modulo 2 Pi symmetry of the list.

```
EEFourier[timeseries_]:=Fourier[timeseries]/N[Sqrt[Length[timeseries]]]
EEInverseFourier[timeseries_]:=Chop[InverseFourier[timeseries]
N[Sqrt[Length[timeseries]]]]
```

■ Miscellaneous Functions

The function window is used to translate ad sampled data sequence into a simulated continuous signal. It does so by returning an index that the nearest integer less than **t/sampleperiod** where sampleperiod is the sample time period of the sequence being created.

```
window[t_]:=Mod[Quotient[t/sampleperiod,1],Length[txsequence]]+1;
```

■ A Quick Review of Relationships between Correlations, Spectral Density Functions and Fourier Transforms.

The method detailed in this document deals with the use of periodic signals for characterizing a channel. A periodic signal has infite energy and finite average power. The Fourier Transform of a periodic signal does not exist but the exponential Fourier series expansion does exist. In the case where the periodic signal is represented by a finite number of discrete time samples there are only N (where N is the number of time samples in a period) series coefficents needed to describe the signal. If you take the reciprocal of the time period of an individual sample, you have the "analog" frequency associated with 2 Pi radians of "digital" frequency. Each individual series coefficent then represents the signal component at "analog" frequency/N Hertz.

The zero frequency term of the exponential Fourier series expansion of a periodic signal is the average value of that signal. The product of the Fourier Series with the complex conjugate of that series is the Power Spectral Density function of the signal and the sum of all of the terms of this series is the average power of the signal.

The periodic autocorrelation function of a signal is the inverse Fourier Series expansion of the Power Spectral Density Function. The zero time shift term of the periodic autocorrelation function is the average power of the signal.

A very useful relationship when thinking about the impact of interference on correlators is that FT[F1(t)**F2(2)]=FT[F1(t)]x Conjugate[FT[F2(t)]] where ** is correlation and FT is Fourier Transform and x is multiplication. This relationship is used throughout the text in the context that if F1(t) is the arriving baseband signal and F2(2) is an ideal m-sequence, the inverse Fourier transform of the Fourier transform of F1(t) times the complex conjugate of the Fourier transform of F2(t) is a good representation of the impulse response of a channel.

■ Low Pass Filter

This describes the implementation of an FIR digital filter with **nt** taps. The vector **hwiggle[k]** is the desired frequency response specified at n points around the unit circle. The vector **h** is the impulse response of the filter which is calculated by using an inverse z transform on the **hwiggle** vector. The first cell performs initializations and calculates coefficents. The cutoff frequency of the filter is set by the parameter cutoff that is set to 2 for a Pi cutoff, 4 for a Pi/2 cutoff, 8 for a Pi/4 cutoff, etc.

```
nt=16;
cutofffreq=8;
h=Table[0,{k,0,nt-1}];
hwiggle=Table[0,{k,0,nt-1}];
Do[hwiggle[[k]]=1,{k,1,nt/cutofffreq}]
hwiggle[[(nt/cutofffreq)+1]]=.5;
coefficents=Table[0,{k,0,nt-1}];
Do[h[[n]]=1/nt Sum[hwiggle[[k]] Exp[I (2 Pi/nt) (k-1) (n-1)],{k,1,nt}],{n,1,nt}];
coefficents=Abs[h//N];
```

This next cell defines a function **filter** with an input sequence x and an output sequence y that performs the desired filtering operation. The result of the filtering operation is stored in a vector called **output**.

```
filter[x_]:=(y=Table[0,{k,0,Length[x]-1}];
Do[y[[n]]=Sum[coefficents[[k]]  x[[n-k-1]],{k,1,nt}],{n,(nt+4),Length[x]-1}];
output=y;)

Options[ScatterPlot3D]  =  {PlotJoined->False}

ScatterPlot3D[l3:{{_,  _,  _}..},  opts___]  :=
 If[(PlotJoined  /.  {opts}  /.  Options[ScatterPlot3D]),
  Show[Graphics3D[Line[l3]]  ],
  Show[Graphics3D[Map[Point,l3],PlotRange->All,Boxed->False]   ]
 ]
```

{PlotJoined -> False}