PAM Transmitter and Receiver
Implementing Coherent Detection*

Ed Doering

This work is produced by OpenStax-CNX and licensed under the
Creative Commons Attribution License 2.0

LabVIEW

This module refers to LabVIEW, a software development environment that features a graphical programming language. Please see the LabVIEW QuickStart Guide module for tutorials and documentation that will help you:

- Apply LabVIEW to Audio Signal Processing
- Get started with LabVIEW
- Obtain a fully-functional evaluation edition of LabVIEW

Table 1

Note: Visit LabVIEW Setup to learn how to adjust your own LabVIEW environment to match the settings used by the LabVIEW screencast video(s) in this module. Click the "Fullscreen" button at the lower right corner of the video player if the video does not fit properly within your browser window.

1 Summary

The integrate-and-dump detector is fundamental to coherent detection, the optimal receiver technique that minimizes bit error rate (BER) for a given signal-to-noise ratio $\frac{E_b}{N_0}$. In this project develop a pulse amplitude (PAM) transmitter based on a transmit filter to map a bitstream onto a signaling waveform (rectangular and Manchester pulse shapes), an additive white Gaussian noise (AWGN) channel, and a receiver that implements integrate-and-dump detection. All waveforms throughout the signal processing chain are presented as a stacked chart indicator with a speed control to permit generated waveforms to be studied slowly (i.e., the integrator output ramping up or down) or quickly to process long message bitstreams. Visualizing the critical system signals as waveforms facilitates exploration of the effects of specific values of BER and $\frac{E_b}{N_0}$, and promotes deeper understanding of coherent detection.

*Version 1.2: Nov 29, 2008 2:58 pm -0600
†http://creativecommons.org/licenses/by/2.0/
2 Objectives

1. Implement a binary pulse amplitude modulation (PAM) transmitter
2. Model an additive white Gaussian noise (AWGN) channel impairment with a random number generator
3. Implement a PAM receiver based on the integrate-and-dump form of coherent detection
4. Study the signal processing chain from the source message bitstream to the regenerated bitstream
5. Evaluate system performance using a plot of bit error rate (BER) vs. signal-to-noise ratio (Eb/No)
6. Learn how to use the LabVIEW point-by-point signal processing design pattern

3 Deliverables

1. Summary write-up of your results
2. Hardcopy of all LabVIEW code that you develop (block diagrams and front panels)
3. Any plots or diagrams requested

NOTE: You can easily export LabVIEW front-panel waveform plots directly to your report. Right-click on the waveform indicator and choose "Export Simplified Image."

4 Setup

1. LabVIEW 8.5 or later version

5 Textbook Linkages

Refer to the following textbooks for additional background on the project activities of this module; see the "References" section below for publication details:

- Carlson, Crilly, and Rutledge – Ch 11
- Couch – Ch 6
- Haykin – Ch 5
- Haykin and Moher – Ch 10
- Lathi – Ch 14
- Proakis and Salehi (FCS) – Ch 8
- Proakis and Salehi (CSE) – Ch 7
- Stern and Mahmoud – Ch 4

6 Prerequisite Modules

If you are relatively new to LabVIEW, consider taking the course LabVIEW Techniques for Audio Signal Processing\(^1\) which provides the foundation you need to complete this project activity, including: block diagram editing techniques, essential programming structures, subVIs, arrays, and audio.

\(^1\) [http://cnx.org/content/col10440/latest/](http://cnx.org/content/col10440/latest/)
7 Introduction

Noise represents the most widely-known channel impairment in a communication system. No doubt you have heard "static" while listening to AM radio during a thunderstorm, soft hissing during a telephone conversation, and other types of background noise. Digital communication system noise causes errors in the recovered (regenerated) bit stream at the receiver.

In general, digital receivers rely on one of two detection techniques to regenerate the transmitted bit stream: coherent detection and non-coherent detection. "Coherent" means the receiver maintains synchronism with the transmitter, normally by using special subsystems that extract timing signals directly from the transmitted bit stream. Transmitting timing pulses in a separate channel is usually too expensive for long-haul communication links. The synchronizer establishes the precise beginning and ending of each bit interval. A synchronizer increases the receiver's cost and complexity, but also achieves the lowest bit error rate (BER) of the two techniques for a given signal-to-noise ratio (SNR). Incoherent detection, on the other hand, uses a lower-complexity approach to recover the bit stream, but does not perform as well in terms of BER. In this project the correlation detector scheme is studied in detail.

Figure 1 illustrates a generic communication system (transmitter, channel, and receiver) and a comparator to compare the original source bitstream to the output bitstream and report bit error.

![Generic communication system with comparator](http://cnx.org/content/m18652/1.2/)

This project implements Figure 1 at a moderately realistic level:

1. The source is a bitstream with equiprobable 0s and 1s.
2. The pulse amplitude modulation (PAM) transmitter maps the two source symbols onto rectangular signaling waveforms; these discrete-time waveforms approximate the true analog signaling waveforms that would be applied to a radio transmitter's modulator for wireless communications or a laser diode for fiber optic communications, for example.
3. The channel impairs the transmitted signal with additive white Gaussian noise (AWGN).
4. The receiver is a coherent receiver implemented as a correlation receiver.

8 PAM Transmitter

Figure 2 illustrates the detailed block diagram of the binary pulse amplitude modulation (PAM) transmitter.
The bitstream 1 and 0 values map to the amplitudes $\sqrt{E_b/T_b}$ and $-\sqrt{E_b/T_b}$, where $E_b$ is the energy per bit and $T_b$ is the bit interval. The amplitudes are applied to the prototype pulse shape $p(t)$ with unit amplitude to generate a pair of signaling waveforms $s_1(t) = \sqrt{E_b/T_b}p(t)$ and $s_0(t) = -\sqrt{E_b/T_b}p(t)$. This signaling scheme is called **binary antipodal signaling**.

Many different pulse shapes are used in practice, based on the application. This project considers two specific pulse shapes, namely, **rectangular** and **Manchester**. Both of the pulse shapes are of the polar NRZ (non return to zero) type. The Figure 3 screencast video continues the discussion by describing these two pulse shapes in more detail.

The **signal point mapper** and **pulse generator** of Figure 2 describe the desired amplitudes and pulse shape, while the transmit filter converts the message bitstream into a sequence of signaling waveforms. The transmit filter is an FIR filter driven by an impulse train derived from the signal point mapper amplitudes; the FIR filter coefficients are the pulse shape values. Refer to the screencast video in pam_TransmitFilter.vi for full implementation details.

The **bit sync generator** block sends pulses to the receiver to indicate the beginning and ending a bit.
9 AWGN Channel

**Additive white Gaussian noise (AWGN)** impairs signals as they pass through an electromagnetic medium, including the electronics in the transmitter and receiver. Adding the output of a Gaussian random number generator to the transmitted signal simulates the AWGN impairment of a real channel.

The degree of signal impairment is reported as a ratio of signal strength to noise ratio (SNR). Digital communication systems define signal-to-noise ratio as $E_b/N_0$ (pronounced "ebb know"), where $E_b$ is the energy per bit and $N_0$ is twice the power spectral density of thermal noise at room temperature. The ratio is dimensionless, and is normally reported in decibels. Refer to the screencast video in util_AWGNchannel_PtByPt.vi to learn how to convert a specified $E_b/N_0$ ratio into the standard deviation parameter of a Gaussian random number generator.

10 Coherent Detection Receiver

Figure 4 shows the block diagram of a receiver that implements coherent detection with a correlator, also called an *integrate-and-dump* detector.

![Figure 4: Block diagram of PAM receiver based on coherent detection](http://cnx.org/content/m18652/1.2/)

The **correlator** multiplies the received signal by the same pulse shape used by the transmitter, and then integrates this product signal over one bit interval. The correlator output is sampled at the end of the bit interval by the **sample-and-hold** device, and then compared to the zero threshold. If the sampled correlator output is greater than the threshold, the received bit is declared a 1, otherwise the received bit is declared a 0. The integrator is reset to zero at the beginning of each bit interval.

The receiver requires precise synchronization with the transmitter in two respects: the correlator must multiply the received signal by the pulse shape in the same time location, and the integrator must be reset precisely at the beginning of a new bit interval. These requirements are easy to achieve within a simulation, since the transmitter can send pulses to signal the beginning and ending of the bit interval. In a real system,
synchronization subsystems extract these timing pulses directly from the received signal, adding cost and complexity to the receiver.

Digital communication system performance in the face of AWGN channel impairment is measured in terms of bit error rate (BER) for a given signal quality $E_b/N_0$. Coherent detection with binary antipodal signaling as used in this project has a theoretical BER of

$$ \text{BER} = Q \left( \sqrt{\frac{2E_b}{N_0}} \right) \quad (4) $$

where the Q-function $Q(x)$ describes the area under a zero-mean unit-variance Gaussian probability density function from $x$ to positive infinity, i.e., the area under the positive tail of the Gaussian. Equation (4) serves as the benchmark for the simulated BER of the system constructed in this project.

11 Procedure
11.1 Build the subVIs

Build the subVIs listed below. You may already have some of these available from previous projects.

Demonstrate that each of these subVIs works properly before continuing to the next part.

1. pam_SignalPointMapper.vi
2. pam_RectanglePulse.vi
3. pam_ManchesterPulse.vi
4. pam_TransmitFilter.vi
5. pam_TransmitSync.vi
6. regen_Correlator.vi
7. regen_SampleHold.vi
8. regen_BitstreamBuffer.vi
9. util_BitstreamFromRandom.vi
10. util_AWGNchannel_PtByPt.vi
11. util_MeasureBER.vi
12. util_Qfunction.vi

11.2 Build the transmitter

Assemble the transmitter by translating Figure 2 into a LabVIEW application VI called Transmitter.vi. Create front panel controls with default values as follows:

1. message length - I32 - 5 bits
2. $E_b$, energy per bit interval [J/bit] - DBL - 1.0
3. $T_b$, bit interval [s] - DBL - 1.0
4. pulse shape - enumerated data type - Rectangle
5. $f_s$, sampling frequency [Hz] - DBL - 10.0

Use an enumerated front-panel control to select the pulse shape, and a case structure on the block diagram to select the desired pulse shape. The Figure 5 screencast video explains how to configure the front-panel control and how to use the control as the selector on the case structure.

http://cnx.org/content/m18652/1.2/
Plot the transmitted signal waveform for both the polar NRZ and Manchester pulse shapes, and confirm that the signal waveform amplitude and samples per bit interval respond correctly to various selections for sampling frequency, bit interval, energy per bit, and message length.

11.3 Build the channel and receiver

Visualizing the signal processing chain through the receiver is the main objective of this section. The stacked chart waveform indicator works best because it allows timescale adjustments while maintaining synchronism among all of the displayed signals. The stacked chart emulates a strip chart recorder or oscilloscope display, and is designed to accumulate and display one sample point generated each pass through a repetitive structure such as a for-loop or while-loop. The Figure 6 screencast video introduces the stacked chart waveform indicator, explains how to display multiple signals, and describes how to interact with the indicator to view selected time intervals.

Copy Transmitter.vi to a new file called TransmitterReceiver.vi. Remove the waveform graph indicator. Add the AWGN channel and coherent receiver to this VI by translating the Figure 4 receiver block diagram. Make a front panel control for the channel Eb/No. Embed the entire channel and receiver into a for-loop structure. Include "Programming | Timing | Wait Until Next ms Multiple" inside the for-loop and create a front-panel control called loop delay [ms] to adjust the delay. Place the control inside the for-loop structure so that the processing rate of the receiver can be easily adjusted. Display the following signals on a stacked chart:

1. transmitted signal, s(t)
2. received signal, s(t)+n(t)
3. transmitter bit interval start pulse
4. transmitter bit interval end pulse
5. correlator output
6. sample-and-hold output
7. comparator output

Include a BER measurement (with util_MeasureBER.vi) to compare the transmitted and received message bitstreams.

Include Boolean indicators for the transmitted bitstream, the regenerated (received) bitstream, and the error bitstream.

http://cnx.org/content/m18652/1.2/
Reserve space for the BER vs. Eb/No plot to be added later.

Figure 7 illustrates a suggested front-panel layout for TransmitterReceiver.vi.

![Transmitter and Receiver](http://cnx.org/content/m18652/1.2/)

**Figure 7:** Suggested front-panel layout for TransmitterReceiver.vi

Debug the combined transmitter and receiver with a high value of Eb/No such as 40dB to effectively eliminate channel noise. Ensure that the received message is the same as the transmitted message. The BER should remain zero or nearly so, even for relatively long messages.

To confirm that the AWGN channel works properly, set the front panel controls to these exact values:

1. **message length** = 10,000 bits
2. **Eb** = 1 J/bit
3. **Tb** = 1 s
4. **Eb/No** = 0 dB
5. **pulse shape** = Polar NRZ
6. **fs** = 32 Hz
7. **loop delay** = 0 ms

The BER should be very close to 0.079 each time the VI is run; the theoretical value is 0.07865.

11.4 Experiment with the transmitter, channel, and receiver

Set Eb/No to 40dB to generate a clean transmitter signal at the receiver, and study the correlator output for the polar NRZ pulse shape. Describe the effect of the "integrate-and-dump" operation as applied to the transmitted signal. Use a loop delay of in the range 10 to 50 ms to observe the waveform unfold slowly.

[Note: The diagram and text are aligned and presented as per the requirements.]
Switch to the Manchester pulse shape, and study the correlator output again. The correlator output should look exactly the same as observed for the polar NRZ pulse shape, even though the two pulse shapes are significantly different. Explain why.

Try message lengths from 10 bits to 10,000 bits and higher. Confirm that BER is zero or nearly so for each message.

Set the message length to 10 bits. Gradually decrease Eb/No and observe the effect on the receiver signals. What level of Eb/No causes the received signal to look noisy and yet still be intelligible to the eye? What level of Eb/No causes the received signal to look essentially unusable, and yet the BER remains small (say, 1 percent)? From these observations, explain how coherent detection is able to recover a very useable signal from such a noisy input.

11.5 BER vs. Eb/No performance measure

Add a structure to retain the Eb/No and measured BER in arrays at the end of each simulation run. Plot BER vs. Eb/No as a scatter plot over the domain Eb/No = 0 dB to 10 dB. Include a Boolean control to reset the plot by reinitializing the arrays. See the Figure 8 screencast for implementation details.

![Figure 8:](http://cnx.org/content/m18652/1.2/)[video] Retain values across multiple runs of a VI and visualize values as a scatter plot

Engage the "Run Continuously" mode (the circulating arrows icon next to the "Run" button) to continually add points to the plot. Vary Eb/No from 0 dB to 10 dB for a message length of 100 bits. Make note of the spread of BER values for a particular Eb/No value, as well as the minimum BER. Increase the message length to 1,000 bits and then clear the accumulated plot points. Observe the BER spread and minimum value as Eb/No varies over the same range.

Repeat the previous step for a message length of 10,000 bits. Consider your results for various message lengths, and then explain the relationship between the minimum recorded BER and message length. In addition, describe the relationship between the spread (variance) of BER values as a function of Eb/No. Explain why the spread decreases as the noise level increases, or equivalently, as Eb/No decreases. Add the theoretical BER vs. Eb/No curve for binary antipodal signaling as a solid trace to the scatter plot; refer to the Figure 9 screencast video to learn how to overlay two plots. How well does the simulated scatter plot match theory? What is the critical parameter that causes the measured BER to more closely follow the theoretical value for higher-quality signals, i.e., when Eb/No is closer to 10 dB? What penalty is incurred to achieve a more accurate estimate of BER for higher quality signals?

Include representative plots in your report.

![Figure 9:](http://cnx.org/content/m18652/1.2/)[video] Overlay two plots
12 References