Laboratory 2: Amplitude Modulation

Cory J. Prust, Ph.D.
Electrical Engineering and Computer Science Department
Milwaukee School of Engineering
Last Update: 28 July 2018

Contents

0 Laboratory Objectives and Student Outcomes

0.1 Laboratory Objectives

This laboratory will assist students in understanding the theoretical foundations of amplitude modulation. Students utilize SDR hardware to explore amplitude modulation using both analog and digital messages signals. Additional topics include complex-envelope models, coherent and non-coherent detection, and quadrature-amplitude modulation.

0.2 Student Outcomes

Upon successful completion of this laboratory, the student will:

- Observe transmission of information using amplitude modulation over a wireless channel.
- Describe and characterize DSB-LC and DSB-SC waveforms based on time-domain and frequency-domain measurements.
- Explain the concept of frequency-division multiplexing.
- Utilize complex-envelope notation to describe amplitude modulated waveforms.
- Simulate transmission of two independent messages using quadrature-amplitude modulation.
- Construct a real-time DSB-LC receiver using their personal SDR device.
- Construct a real-time DSB-LC transmitter and broadcast a waveform that meets a radio-frequency channel specification.

1 Background

Appendices A.1 and A.2 provide an overview and key results for double-sideband suppressed-carrier (DSB-SC) and double-sideband large-carrier (DSB-LC) communication systems, respectively.

Appendix A.3 provides an overview and key results for quadrature-amplitude modulated (QAM) communication systems.

Appendix B describes the equivalence between the complex-valued signal models used in describing SDR transceivers and I/Q transceivers.

2 Observing AM Waveforms

In this part of the laboratory you will use your personal SDR to observe amplitude modulated communication signals. Your instructor will broadcast radio-frequency signals in the laboratory at a frequency in the 902MHz to 928MHz frequency band.

The broadcast waveform consists of both a DSB-LC and a DSB-SC communication signal, separated from one another by 30kHz. That is, the carrier frequencies of the two passband signals differ by 30kHz. Both passband signals are the result of modulating a sinusoidal carrier with a sinusoidal message signal.

2.1 Frequency-domain

Using the Spectrum Analyzer model you created in Lab 1, observe the communication signals in the frequency-domain. You should see a spectrum similar to Figure 1. You may need to adjust the sampling rate of your receiver to more clearly see the two waveforms. A sampling rate of 240000 should work reasonably well.

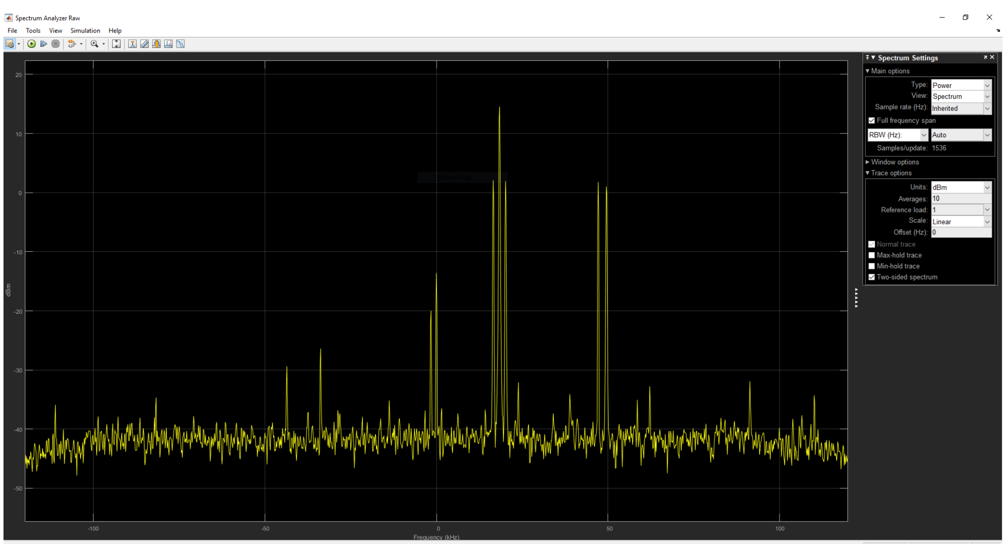


Figure 1: Example spectrum analyzer window during broadcast of DSB-LC and DSB-SC waveforms.

Question 2.1: Take a screen capture of the Spectrum Analyzer window. Annotate this image by identifying the following:

- the DSB-LC communication signal
- the DSB-SC communication signal
- the upper and lower sidebands of the DSB-LC and DSB-SC signals

Question 2.2: What is the actual carrier frequency of the DSB-LC passband waveform that is being transmitted in the laboratory? What is the carrier frequency of the DSB-SC passband waveform?

Question 2.3: What is the message frequency of the DSB-LC communication signal? That is, what is the frequency of the sinusoidal message that is modulating the carrier?

Question 2.4: What is the message frequency of the DSB-SC communication signal? That is, what is the frequency of the sinusoidal message that is modulating the carrier?

Question 2.5: Determine the modulation index of the DSB-LC communication signal using measurements from its spectrum.

2.2 Time-domain

We will now add time-domain visualization to the Simulink model. The new model is shown in Figure 2.

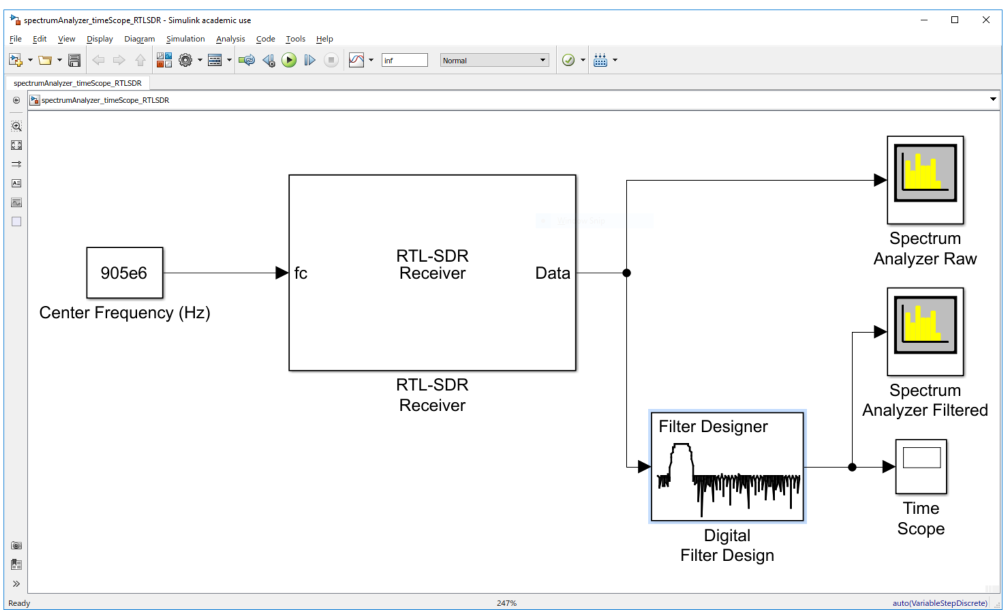


Figure 2: Simulink model for viewing signals in the time-domain and frequency-domain.

- The **Time Scope** (found in `DSP System Toolbox` → `Sinks`) block displays the signals in the time domain, similar to an oscilloscope.
- The **Digital Filter Design** (found in `DSP System Toolbox` → `Filtering` → `Filter Implementations`) is used to isolate either the DSB-LC or DSB-SC waveforms. Configure this block to implement a band-pass filter surrounding one, but not both, of the AM signals. Without this isolation, the time domain

plot would contain the superposition of the two waveforms making it challenging to interpret. It is suggested to use an 60th order Bandpass FIR Equiripple filter. You will need to set the stopband and passband frequencies according to your spectrum observations. Consult with your instructor if you have questions.

- A second **Spectrum Analzyer** has been added to help in confirming operation of the bandpass filter.

You should see time domain waveforms similar to that shown Figure 3.

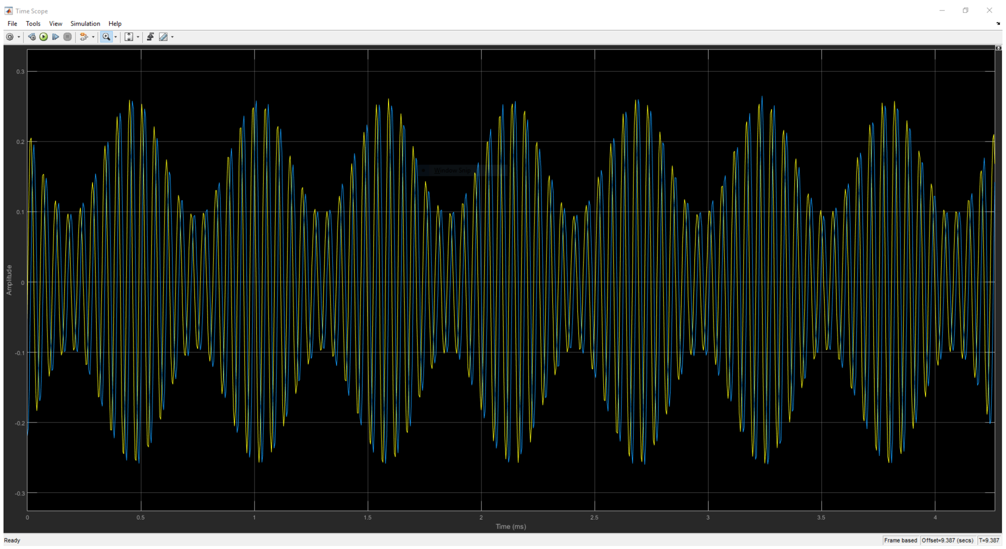


Figure 3: Example time scope window during broadcast of DSB-LC and DSB-SC waveforms.

Question 2.6: Take a screen capture of the Time Scope window showing the DSB-SC signal. Annotate this image by identifying the location/presence of the message signal. Determine the message frequency.

Question 2.7: Take a screen capture of the Time Scope window showing the DSB-LC signal. Annotate this image by identifying the location/presence of the message signal. Determine the message frequency.

Question 2.8: Determine the modulation index of the DSB-LC communication signal using measurements from the time domain plot.

3 Simulation of a QAM Transceiver

During lecture we examined Simulink models that simulated a DSB-SC transceiver system. You will now extend this example to the case of a quadrature-amplitude modulated (QAM) transceiver system capable of transmitting two analog messages using the same carrier frequency. The key feature of such a system is the orthogonality of the cosine and sine carrier waves.

Figure 4 gives a block diagram for a QAM transceiver.

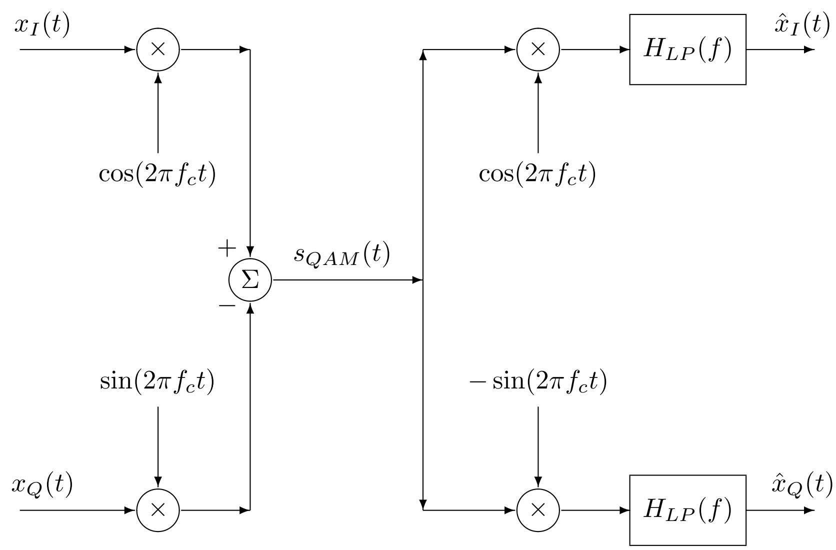


Figure 4: Block diagram representation of QAM transceiver.

The key result, developed in Appendix A.3, is that the messages $x_I(t)$ and $x_Q(t)$ can be recovered from the passband signal $s_{QAM}(t)$ when the transmit and receive oscillators are phase-locked.

$$\begin{aligned}\hat{x}_I(t) &= \frac{1}{2}x_I(t)\\ \hat{x}_Q(t) &= \frac{1}{2}x_Q(t)\end{aligned}$$

Construct a model of the QAM transceiver and simulate its behavior. The following sequence of steps and parameter settings is suggested:

1. Configure the Simulink model to utilize a sampling frequency of $f_s = 1e5$. It is suggested to declare the variable

   ```
   fs = 1e5;
   ```

   in the `InitFcn` callback in the `Model Properties` window and then use it throughout the model wherever the sampling rate is needed.

2. Set the Simulation stop time to 1 second.

3. Configure the transmit carrier oscillators. It is suggested to use the **Sine Wave** block from `DSP System Toolbox/Sources`. Configure the block to give a 10kHz `Complex` output. This yields a sampled version of $\exp(j2\pi f_c t) = \cos(2\pi f_c t) + j\sin(2\pi f_c t)$. Then, use a **Complex to Real-Imag** block to separate

the cosine and sine carrier waveforms. Verify the carrier signals by connecting them to a Time Scope block and running the simulation.

4. Use additional **Sine Wave** blocks to create real-valued message signals for the in-phase and quadrature-phase message signals. Use a 800Hz sinusoid for the in-phase message. Use a 1500Hz sinusoid for the quadrature-phase message.

5. Add multiplier and adder blocks, bringing together the transmitter. Verify the transmitter by connecting the output to Time Scope and Spectrum Analyzer blocks.

6. Construct the receiver. For our purposes, you may ignore the negative sign associated with the quadrature-phase oscillator (i.e., the sin term).

   Initially, be sure the receiver carrier oscillators use the same `Phase offset (rad)` setting (a parameter of the **Sine Wave** block) as the transmit carriers. In this configuration, the receiver and transmitter will be coherent.

   Use the **Digital Filter Design** block from `DSP System Toolbox/Filtering/Filter Designs`. It is suggested to use a 60th order FIR filter, with passband edge frequency of 2000Hz.

7. Connect the transmitter output to the receiver and run the simulation, verifying that the in-phase and quadrature outputs of the receiver match the original message waveforms. Use Time Scope and Spectrum Analyzer blocks as needed.

8. Simulate non-coherent behavior by adjusting the `Phase offset (rad)` setting of the receiver (or, equivalently, the transmitter).

Question 3.1: Submit a screen capture of your Simulink model. Appropriately label each of the key components, including naming Time Scope and Spectrum Analyzer blocks according to the attached signals.

Question 3.2: For the case of **coherent** operation, submit screen captures of key simulation results (time-domain and/or frequency domain waveforms) and explain the results. Your response will be scored based on the accuracy and completeness of your explanations.

Question 3.3: For the case of **non-coherent** operation, submit screen captures of key simulation results (time-domain and/or frequency domain waveforms) and explain the results. Your response will be scored based on the accuracy and completeness of your explanations.

4 DSB-LC Receiver

In this portion of the laboratory you will contruct a DSB-LC receiver using your personal SDR device. You will use this receiver to demodulate a voice signal that your instructor will broadcast in the laboratory. This form of AM voice communication has been used for many decades, and is still in use today. Air traffic control communications, such as those between an airport control tower and approaching aircraft, utilize AM voice transmissions in the VHF band.[1]

4.1 DSB-LC Demodulation using the Complex Envelope

Consider a DSB-LC passband signal of the form

$$s_{DSBLC}(t) = [A + m(t)]\cos(2\pi f_c t) \tag{1}$$

where f_c is the carrier frequency, $m(t)$ is the message signal, and A is chosen such that $A + m(t) > 0$ for all values of t. This choice of A means that the information signal $m(t)$ is contained in the envelope of the passband signal.

Our demodulation strategy can be summarized as follows:

- Tune the receiver's center frequency near, but not exactly to, the center frequency f_c. In reality, it would be extremely challenging to tune to exactly f_c. Furthermore, even if we could, the oscillators in both the transmitter and receiver are likely to drift slightly over time (e.g., due to temperature changes). We will tune the receiver to $f_c - f_i$ where $f_i << f_c$. The offset f_i must be small enough that the DSB-LC signal remains within the instantaneous bandwidth of the SDR receiver.
- Filter the received signal to limit noise.
- Recover the message information by extracting the envelope of the received DSB-LC waveform. This is a form of non-coherent demodulation.

In reference to the complex signal model for an SDR receiver (see Appendix B), the SDR receiver generates samples of $\hat{z}(t)$. Therefore, using the DSB-LC signal above we have

$$z(t) = \text{LPF}\{s_{DSBLC}(t)e^{-j(2\pi(f_c - f_i)t+\phi)}\} \tag{2}$$

where f_i is the frequency offset and ϕ is a constant phase offset. Simplify this result:

$$\begin{aligned}
z(t) &= \text{LPF}\{[A + m(t)]\cos(2\pi f_c t)e^{-j(2\pi(f_c - f_i)t+\phi)}\} && (3)\\
&= \text{LPF}\{[A + m(t)]\frac{1}{2}(e^{j2\pi f_c t} + e^{-j2\pi f_c t})e^{-j(2\pi(f_c - f_i)t+\phi)}\} && (4)\\
&= \frac{1}{2}\text{LPF}\{[A + m(t)](e^{j(2\pi f_i t-\phi)} + e^{-j(2\pi(2f_c + f_i)t-\phi)})\} && (5)\\
&= \frac{1}{2}[A + m(t)]e^{j(2\pi f_i t-\phi)} && (6)
\end{aligned}$$

where LPF$\{\cdot\}$ denotes the low pass filter operation that removes the double-frequency term near $2f_c$. Note that f_i is sufficiently small such that the signal of interest lies within the instantaneous bandwidth of the receiver.

To recover the message signal, denoted $\hat{m}(t)$, we compute the magnitude of the complex signal $z(t)$

$$\begin{aligned}
\hat{m}(t) &= |\hat{z}(t)| && (7)\\
&= \left|\frac{1}{2}[A + m(t)]e^{j(2\pi f_i t-\phi)}\right| && (8)
\end{aligned}$$

[1] Students interested in learning more might start with https://en.wikipedia.org/wiki/Airband.

$$= \frac{1}{2}|A+m(t)|\left|e^{j(2\pi f_i t-\phi)}\right| \quad (9)$$

$$= \frac{1}{2}[A+m(t)] \quad (10)$$

since, per our initial assumption, $A + m(t) > 0$. Note that the message (plus the constant scalar A, which can simply be subtracted) is recovered regardless of the frequency offset f_i and the phase offset ϕ. The key fact is that the complex envelope lies in both the in-phase and quadrature-phase components embedded in $v(t)$. Figure 5 illustrates the demodulation procedure using frequency-domain plots.

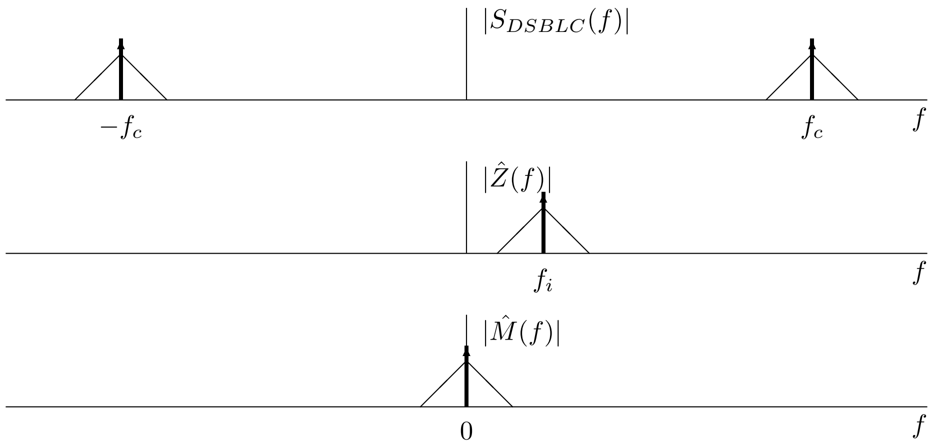


Figure 5: Frequency domain plots of DSB-LC Demodulation using the complex envelope.

4.2 Creating the Simulink model

1. Open a new Simulink model and create the model shown in Figure 6.

   - **RTL-SDR Receiver** configuration:
     - Sampling rate: 240e3
     - Output data type: `single`
     - Samples per frame: 1000
     - "Lost samples output port" enabled. This signal will be non-zero when samples are lost, indicating that the model is not running in real-time.
     - Tuner gain controlled via `Input port`. Use a **Constant** block and **Slider Gain**, allowing you to adjust the RTL-SDR receiver gain while the model is running. Most RTL-SDR devices allow gain values between 0 and 60 dB. [2]
   - Multiple **Spectrum Analyzer** and **Time Scope** blocks are included. You may need to comment these out to achieve good performance. Use them as needed to debug your model and/or view signals of interest.

[2] To find valid gain settings for your particular RTL-SDR device, see `https://www.mathworks.com/help/supportpkg/rtlsdrradio/ug/comm.sdrrtlreceiver.info.html`

- Use the **Digital Filter Design** (found in `DSP System Toolbox` → `Filtering` → `Filter Implementations`) block to create a bandpass filter that will pass the DSB-LC waveform. It is suggested to begin with a 60th order Bandpass FIR Equiripple filter. Consult with your instructor if you have questions regarding proper use of this block.
- The **Complex to Magnitude-Angle** (found in `Simulink` → `Math Operations`) block will demodulate the DSB-LC signal at f_i to 0Hz.
- The **FIR Decimation** (found in `DSP System Toolbox` → `Filtering` → `Multirate Filters`) block will change the sample rate of the signal. Specify a decimation factor of 5, which means the sample rate will be reduced by a factor of 5 at the output of the block. Therefore, the sample rate for the remainder of the model will be $240e3/5 = 48e3$ which is a standard sample rate for audio signals.
- The **DC Blocker** (found in `DSP System Toolbox` → `Signal Operations`) block does exactly as its name implies, which in our case removes the offset A.
- The **Audio Device Writer** (found in `DSP System Toolbox` → `Sinks`) block provides an interface to your PC sound card. The default settings should work for most PCs.

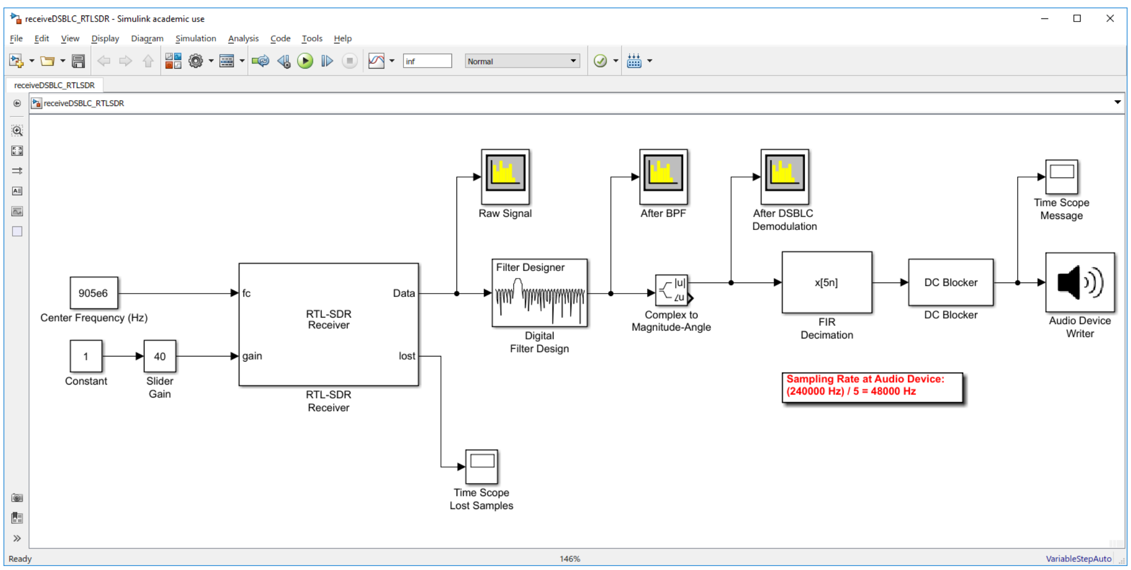


Figure 6: Simulink model for the DSB-LC receiver.

2. Set the Simulation stop time to `inf`.
3. Set the center frequency according to the instructions provided by your instructor. You may need to adjust the center frequency according to the design of your bandpass filter (or vice versa).
4. Carefully examine the Spectrum Analyzer windows. Compare what you are seeing to the theory presented in Section 4.1.
5. Listen to the resulting audio. You may need to experiment with the receiver gain setting in order to improve the audio quality.

Question 4.1: Submit screen captures of Spectrum Analyzer windows showing the various signals throughout the model. Provide brief explanations of what is seen in each capture.

Question 4.2: What happens to the recovered audio when you adjust the gain of the SDR receiver block? Summarize your findings.

Question 4.3: What happens when you remove the Bandpass Filter from the model? *Hint:* You can do this easily by right-clicking the block and selecting "Comment Through". Summarize your findings.

5 Laboratory SDR Hardware and Installation

5.1 Ettus Research B200 USRP

The Ettus Research [1] B200 Universal Software Radio Peripheral (USRP) is a high-performance software-radio transceiver providing high bandwidth, simultaneous transmit and receive operations, and numerous other advanced capabilities. The B200 communicates with a host PC through a USB 3.0 interface. It utilizes the USRP Hardware Driver, or UHD [2], which is supported on Windows, Linux, and Mac OS X operating systems.

- **Frequency Coverage**: 70MHz to 6GHz
- **Instantaneous Bandwidth**: up to 56MHz

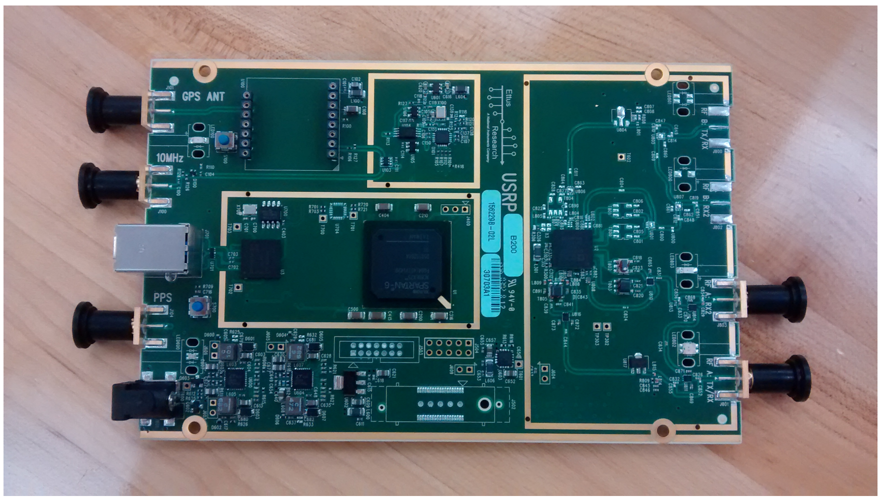


Figure 7: Ettus Research B200 USRP.

More detailed information on the B200 can be found in references [3] and [4]. MATLAB and Simulink documentation is available at https://www.mathworks.com/hardware-support/usrp.html.

5.1.1 Installation in MATLAB/Simulink

Support for the B200 USRP is provided in MATLAB 2014b or later. To install:

1.
2. Navigate to the `Home` tab on the MATLAB Toolstrip, then to `Add-Ons -> Get Hardware Support Packages`.
3. Search for “USRP” and choose the “Communications System Toolbox Support Package for USRP Radio” option.
4. Click the option to “Install”
5. You will need to log in to a Mathworks account in order to proceed. You can create one, if necessary.
6. You will be prompted to agree to various licensing agreements. Continue with the installation.

When the installation is complete, you should receive notification that “Your Hardware Support Package requires configuration.” If you have access to a USRP radio, choose “Setup Now” and continue with the installation. Otherwise you can resume configuration at a later time by typing `targetupdater` at the MATLAB command prompt.

5.1.2 Verifying the Ettus B200 USRP Installation

Connect a B200 SDR to your laptop. Be sure to use a USB 3.0 connection. It may take Windows a few minutes to recognize the hardware and configure the necessary drivers.

To verify the MATLAB configuration and communication with the USRP, type `findsdru` at the MATLAB prompt. After a few moments, you should see the message

```
>> findsdru
Checking radio connections...

ans =

  struct with fields:

     Platform: 'B200'
    IPAddress: ''
    SerialNum: '30703A1'
       Status: 'Success'
```

where the `SerialNum` entry will vary depending on your specific B200.

6 DSB-LC Transmitter

In this portion of the laboratory you will construct a DSB-LC transmitter using the USRP hardware. Anyone in the laboratory will be able to receive and listen to your transmission!

Just like radio-frequency transmissions in the real-world, care must be taken to ensure that users do not interfere with each other. Therefore, each team of students will be assigned a carrier frequency and a channel bandwidth for their transmission. Make sure your transmission does not exceed your allotted portion of the radio-frequency spectrum!

The following sections will lead you through the development of your transmitter.

6.1 Using MATLAB to generate the baseband DSB-LC signal

We will use MATLAB to generate the baseband waveform, $[A + m(t)]$, for the DSB-LC transmission. The baseband waveform will be sent to the USRP hardware, which will in turn translate it to the assigned carrier frequency. A mathematical model for the SDR transmitter is presented in Appendix B.

Create a MATLAB script that generates the baseband DSB-LC waveform:

1. Select a voice or audio file that you wish to transmit. Please do not use anything with objectionable content, such as a song containing profanity. If you have difficulties finding something to broadcast, consult your instructor.

2. Use the `audioread` function to load the audio file into MATLAB. The function will also return the sample rate. Be sure to make note of it. If the audio file contains stereo audio, use just the left or right audio signal (or sum left and right channels together). Limit the length of the audio clip to no more than 20 seconds.

3. Your DSB-LC transmission should not exceed the channel bandwidth assigned by your instructor. Filter your message signal in accordance with your bandwidth allocation.

4. Generate the DSB-LC waveform by adding the DC offset A. You may use a modulation index of your choosing.

5. Normalize the amplitudes of baseband signal. For best compatibility with the USRP, all samples in the data vector should be in the range $[0, 0.9]$. Plot your vector in MATLAB to ensure you have a proper DSB-LC baseband message and that the amplitude scaling is correct.

6. Resample the data vector to a sampling rate of 240000 samples per second using the `resample` function in MATLAB. This is the sample rate at which data will stream across the USB interface into the USRP hardware.

Use the MATLAB Help information and the example script in Figure 8.

```
%% example_generateComplexEnvelope_DSBLC.m
%   Example script showing how to generate a complex envelope
%   for broadcasting with Ettus B200 USRP.
%   This version generates a DSB-LC waveform with a voice message signal.
%
%   Cory J. Prust, Ph.D.
%   Last Modified:  7/2/2018

%%  USRP Configuration
%   The script generates the complex envelope sampled at 240kHz
%   So, the B200 must be configured for MasterClockRate/Interpolation = 240000.
%    e.g., MasterClockRate = 9600000 and Interpolation = 40
clear all
close all

%% load .wav file
[m, fs_audio] = audioread('myAudioFile.wav');

%% trim length and extract left/right audio as needed
% ADD CODE HERE

%% design and apply an FIR lowpass filter to limit the message bandwidth
f_cutoff = ?                             % cutoff frequency in Hz
order = 50;                              % filter order (higher order gives sharper
    cutoff)
h = fir1(order,f_cutoff/(fs_audio/2));   % see "help fir1"
freqz(h,1,1e5,fs_audio);                 % view your filter design
m = filter(h,1,m);                       % apply filter to the message

% resample to USRP sample rate
fs = 240000;                     %USRP baseband sample rate
m = resample(m',fs,fs_audio);    %note transpose here to convert to row vector

% generate DSB-LC
mu = 1.2;    % modulation index
a = max(m);
A = mu*a;

% generate complex envelope
z = (A + m);

% normalize complex envelope
z = z/max(abs(z)) * 0.8;  % ensure magnitude < 1 to prevent saturation in USRP

% time-domain plots
N = length(m);
t = 0:(1/fs):(N-1)*(1/fs);
figure
plot(t,real(z))

% frequency-domain plot
f = -fs/2:(fs/N):(fs/2 - fs/N);
figure
plot(f,20*log10(abs(fftshift(fft(z)))))
```

Figure 8: MATLAB example for creating the complex envelope of a DSB-LC waveform.

6.2 Creating the Simulink model

1. Open a new Simulink model and create the model shown in Figure 9.
   - **SDRu Transmitter** (found in `Communications System Toolbox Support Package for USRP`). Configure exactly as shown in Figure 10. This block will act as a signal sink operating at 240000 complex samples per second.
   - **Signal From Workspace** (found in `DSP System Toolbox` → `Sources`)
     - Specify signal `z` (or whatever name you used for your baseband message)
     - Sample time: 1/240e3
     - Samples per frame: 1000
     - Form output after final data value by: Cyclic repetition

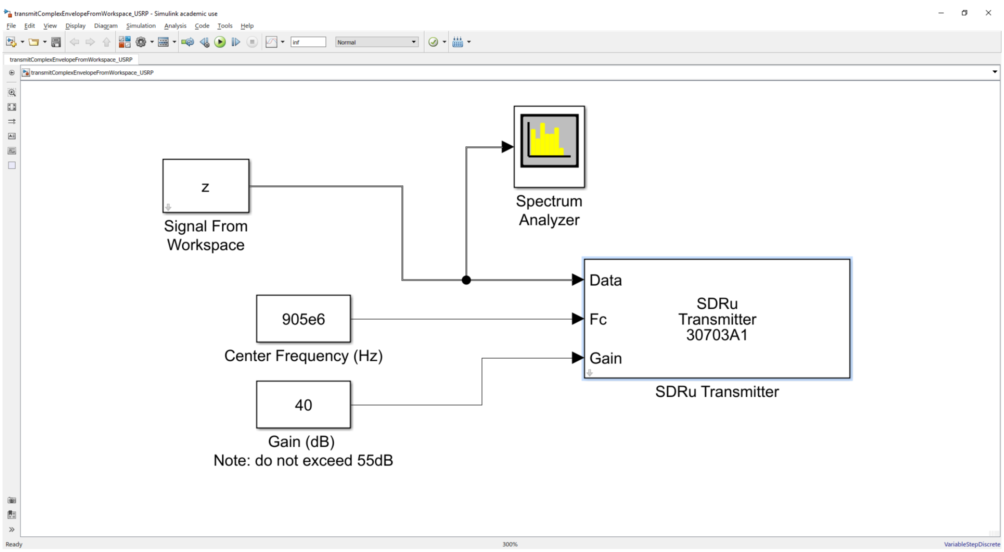


Figure 9: Simulink model for the DSB-LC transmitter.

2. Set the Simulation stop time to `inf`.
3. Temporarily comment out the **SDRu Transmitter** block.
4. Begin execution by clicking the play button (`Run`). Carefully examine the spectrum of your DSB-LC waveform in the Spectrum Analyzer window and verify that it is as expected. In particular, confirm that it matches your bandwidth allocation.
5. Stop your simulation. Uncomment the **SDRu Transmitter** block. Click `Run`. In a few moments, you will be broadcasting your waveform in the laboratory!
6. Ask a lab-mate or your instructor to tune their DSB-LC receiver to your broadcast. Can you hear the resulting audio?
7. If the received signal is weak or noisy, you may need to increase the transmit gain. **Under no circumstances should you increase this setting beyond 55 dB.**

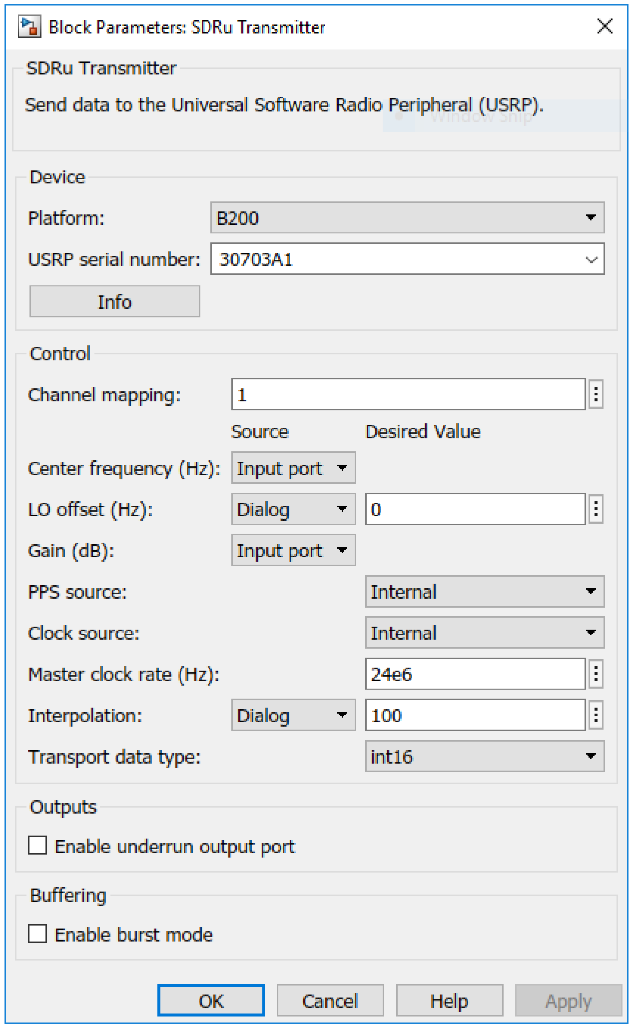


Figure 10: SDRu block configuration for DSB-LC transmitter.

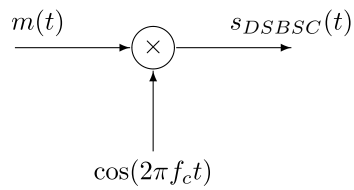
Question 6.1: Submit a time-domain plot of your baseband message signal generated in MATLAB.

Question 6.2: Submit a screen capture of the spectrum analyzer window showing your transmitted DSB-LC communication signal as it is being received on a lab-mate's personal SDR receiver.

A Amplitude Modulation

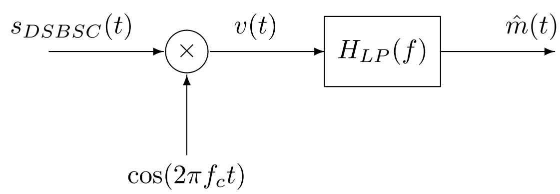
A.1 Double-Sideband Suppressed Carrier (DSB-SC)

1. The standard modulator is



   where $m(t)$ is the message signal and f_c is the carrier frequency.

2. The standard demodulator is



   where $\hat{m}(t)$ is the recovered message and low-pass filter $H_{LP}(f)$ is a linear time-invariant system.

3. Summary of time-domain results:

$$
\begin{aligned}
s_{DSBSC}(t) &= m(t)\cos(2\pi f_c t) && (11)\\
\hat{m}(t) &= \mathrm{LPF}\{s_{DSBSC}(t)\cos(2\pi f_c t)\} && (12)\\
&= \mathrm{LPF}\{m(t)\cos(2\pi f_c t)\cos(2\pi f_c t)\} && (13)\\
&= \frac{1}{2}\mathrm{LPF}\{m(t)[1+\cos(2\pi(2f_c)t]\} && (14)\\
&= \frac{1}{2}m(t) && (15)
\end{aligned}
$$

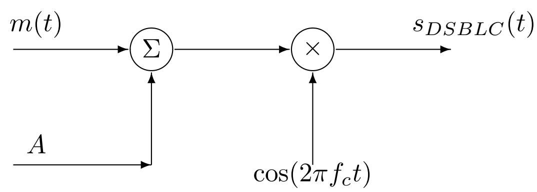
   where LPF means low pass filter.

4. Summary of frequency-domain results:

$$
\begin{aligned}
S_{DSBSC}(f) &= \mathcal{F}\{m(t)\cos(2\pi f_c t)\} && (16)\\
&= \mathcal{F}\{m(t)\} * \mathcal{F}\{\cos(2\pi f_c t)\} && (17)\\
&= M(f) * \frac{1}{2}(\delta(f-f_c)+\delta(f+f_c)) && (18)\\
&= \frac{1}{2}M(f-f_c)+\frac{1}{2}M(f+f_c) && (19)\\
V(f) &= S_{DSBSC}(f) * \mathcal{F}\{\cos(2\pi f_c t) && (20)\\
&= \left(\frac{1}{2}M(f-f_c)+\frac{1}{2}M(f+f_c)\right) * \left(\frac{1}{2}\delta(f-f_c)+\frac{1}{2}\delta(f+f_c))\right) && (21)\\
&= \frac{1}{2}M(f)+\frac{1}{4}M(f+2f_c)+\frac{1}{4}M(f-2f_c) && (22)\\
\hat{M}(f) &= H_{LP}(f)V(f) && (23)\\
&= \frac{1}{2}M(f) && (24)
\end{aligned}
$$

A.2 Double-Sideband Large Carrier (DSB-LC)

1. The standard modulator is



   where $m(t)$ is the message signal and f_c is the carrier frequency.

2. The DSB-SC demodulator can be used for coherent reception. Non-coherent receivers such as envelope detectors are commonly used.

3. Summary of time-domain results:

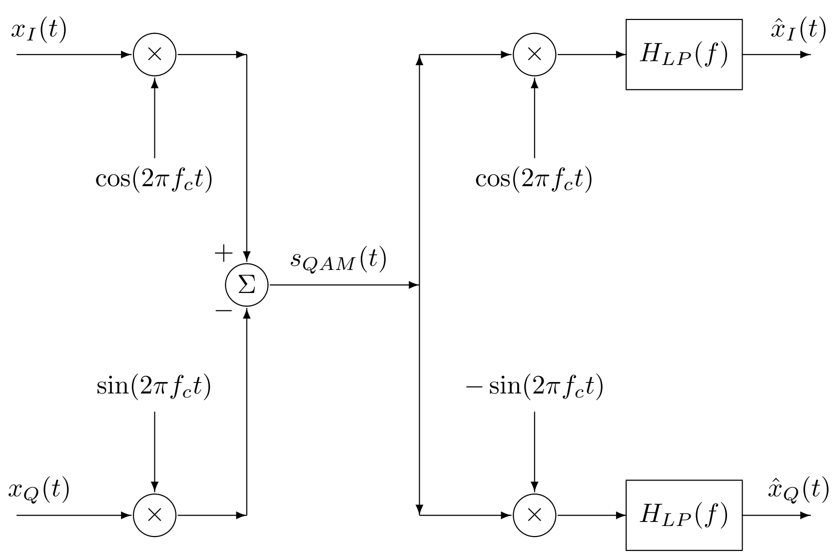
$$s_{DSBLC}(t) = [m(t) + A]\cos(2\pi f_c t) \tag{25}$$

4. Summary of frequency-domain results:

$$\begin{aligned} S_{DSBLC}(f) &= \mathcal{F}\{[m(t) + A]\cos(2\pi f_c t)\} && (26) \\ &= [M(f) + A\delta(f)] * \frac{1}{2}(\delta(f - f_c) + \delta(f + f_c)) && (27) \\ &= \frac{A}{2}\delta(f - f_c) + \frac{1}{2}M(f - f_c) + \frac{A}{2}\delta(f + f_c) + \frac{1}{2}M(f + f_c) && (28) \end{aligned}$$

A.3 Quadrature Amplitude Modulation (QAM)

1. The standard QAM transceiver is



2. Summary of time-domain results:

$$
\begin{aligned}
s_{QAM}(t) &= x_I(t)\cos(2\pi f_c t) - x_Q(t)\sin(2\pi f_c t) && (29)
\end{aligned}
$$

$$
\begin{aligned}
s_{QAM}(t)\cos(2\pi f_c t) &= x_I(t)\cos^2(2\pi f_c t) - x_Q(t)\sin(2\pi f_c t)\cos(2\pi f_c t) && (30)\\
&= x_I(t)\frac{1}{2}[1+\cos(2\pi(2f_c)t)] - x_Q(t)\frac{1}{2}[\sin(2\pi(2f_c)t)+\sin(0)] && (31)\\
&= \frac{1}{2}x_I(t)[1+\cos(2\pi(2f_c)t)] - \frac{1}{2}x_Q(t)\sin(2\pi(2f_c)t) && (32)
\end{aligned}
$$

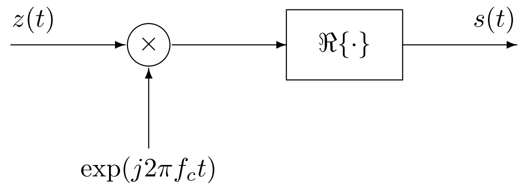
$$
\begin{aligned}
-s_{QAM}(t)\sin(2\pi f_c t) &= -x_I(t)\cos(2\pi f_c t)\sin(2\pi f_c t) + x_Q(t)\sin^2(2\pi f_c t) && (33)\\
&= -x_I(t)\frac{1}{2}[\sin(2\pi(2f_c)t)+\sin(0)] + x_Q(t)\frac{1}{2}[1-\cos(2\pi(2f_c)t)] && (34)\\
&= -\frac{1}{2}x_I(t)\sin(2\pi(2f_c)t) + \frac{1}{2}x_Q(t)[1-\cos(2\pi(2f_c)t)] && (35)\\
& && (36)
\end{aligned}
$$

Therefore

$$
\begin{aligned}
\hat{x}_I(t) &= \text{LPF}\{s_{QAM}(t)\cos(2\pi f_c t)\} && (37)\\
&= \frac{1}{2}x_I(t) && (38)\\
\hat{x}_Q(t) &= \text{LPF}\{-s_{QAM}(t)\sin(2\pi f_c t)\} && (39)\\
&= \frac{1}{2}x_Q(t) && (40)\\
& && (41)
\end{aligned}
$$

B Equivalence of Complex and I/Q Models

(a) The complex signal model for an SDR transmitter is



where $z(t)$ is defined as the complex-valued baseband signal

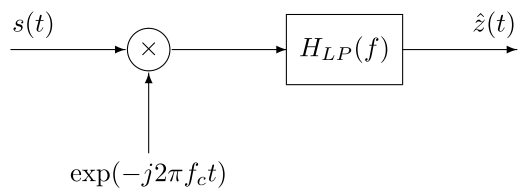
$$z(t) = x_I(t) + jx_Q(t) \tag{42}$$

Note that $x_I(t)$ and $x_Q(t)$ are both real-valued. Then

$$\begin{aligned} s(t) &= \Re\{z(t)\exp(j2\pi f_c t)\} & (43)\\ &= \Re\{[x_I(t) + jx_Q(t)][\cos(2\pi f_c t) + j\sin(2\pi f_c t)]\} & (44)\\ &= x_I(t)\cos(2\pi f_c t) - x_Q(t)\sin(2\pi f_c t) & (45) \end{aligned}$$

Therefore, $s(t)$ is exactly $s_{QAM}(t)$ in Appendix A.3. Note that $s(t)$ is a real-valued passband signal.

(b) The complex signal model for an SDR receiver is



Begin by computing the output of the multiplier:

$$\begin{aligned} s(t)\exp(-j2\pi f_c t) &= [x_I(t)\cos(2\pi f_c t) - x_Q(t)\sin(2\pi f_c t)][\cos(2\pi f_c t) - j\sin(2\pi f_c t)] & (46)\\ &= x_I(t)\cos^2(2\pi f_c t) - jx_I(t)\cos(2\pi f_c t)\sin(2\pi f_c t) & \\ &\quad -x_Q(t)\sin(2\pi f_c t)\cos(2\pi f_c t) + jx_Q(t)\sin^2(2\pi f_c t) & (47)\\ &= \frac{1}{2}x_I(t)[1+\cos(2\pi(2f_c)t)] + j\frac{1}{2}x_I(t)[\sin(2\pi(2f_c)t) + \sin(0)] & \\ &\quad -\frac{1}{2}x_Q(t)[\sin(2\pi(2f_c)t) + \sin(0)] + j\frac{1}{2}x_Q(t)[1-\cos(2\pi(2f_c)t)] & (48) \end{aligned}$$

Then, apply the LPF to get

$$\begin{aligned} \hat{z}(t) &= \mathrm{LPF}\{s(t)\exp(-j2\pi f_c t)\} & (49)\\ &= \frac{1}{2}x_I(t) + j\frac{1}{2}x_Q(t) & (50) \end{aligned}$$

Therefore, the recovered signal $\hat{z}(t)$ contains the real-valued messages $x_I(t)$ and $x_Q(t)$.

References

[1] Ettus Research LLC. Ettus research - home. http://www.ettus.com.

[2] Ettus Research LLC. UHD start. http://code.ettus.com/redmine/ettus/projects/uhd/wiki.

[3] Ettus Research LLC. Ettus research - usrp b200 software defined radio. http://www.ettus.com/product/details/UB200-KIT.

[4] Ettus Research LLC. Usrp hardware driver and usrp manual. http://files.ettus.com/manual/.

[5] Mathworks. Communications Toolbox. Online Resource. https://www.mathworks.com/products/communications.html.

[6] Mathworks. USRP Support Package from Communications Toolbox. Online Resource. http://www.mathworks.com/discovery/sdr/usrp.html.

[7] Mathworks. RTL-SDR Support Package from Communications Toolbox. Online Resource. https://www.mathworks.com/hardware-support/rtl-sdr.html.

[8] Leon W. Couch III. *Modern Communication Systems*. 1995.

[9] Simon Haykin. *Communication Systems*. 4th edition, 2001.