

EE3209 - Communications Laboratory

Lab Manual

California State University, Los Angeles
College of Engineering, Computer Science, and Technology
Department of Electrical and Computer Engineering



Instructor: Khashayar Olia



Lab 2: AM Modulation with USRP

Objective:

This laboratory exercise has two objectives. The first is to gain a first experience in actually programming the USRP to act as a transmitter and a receiver. The second is to investigate the concept of **Amplitude Modulation (AM)**, as transmitter and receiver, and the envelope detector.

Modulation and Demodulation in Communication:

In communication systems, modulation is defined as a process in which the amplitude and/or phase and frequency of the carrier wave is modified so as to include the information stored within the message [1]. The aim of modulation is to transfer a baseband signal over an analog bandpass channel at a different frequency range more suitable for transmission.

Common analog modulation techniques are subcategorized into two main groups:

1. Amplitude Modulation in which the amplitude of the carrier signal is encoded the message signal being transmitted. Double-sideband modulation (DSB), Single-sideband modulation (SSB), Vestigial Sideband modulation (VSB) and Quadrature Amplitude Modulation (QAM) are the examples of Amplitude Modulation.
2. Angel Modulation in which the phase and/or frequency of the carrier signal is varied in proportion to the waveform being transmitted. Frequency Modulation (FM), Phase Modulation (PM) and Transpositional Modulation (TM) are the examples of Angel Modulation.

In this lab, the Amplitude modulation (AM) is being covered. Figure 1 shows a simple transmitter architecture in which both **In-phase** and **Quadrature** part of the message is modulated separately while a combiner is added at the end. In this architecture, the mixers are actually simple multipliers and the combiner is an adder.

Every baseband signal consists of two parts, the real part (**In-phase**) and the imaginary part (**Quadrature**):

$$s(t) = s_I(t) + s_Q(t)$$

Eq. 1



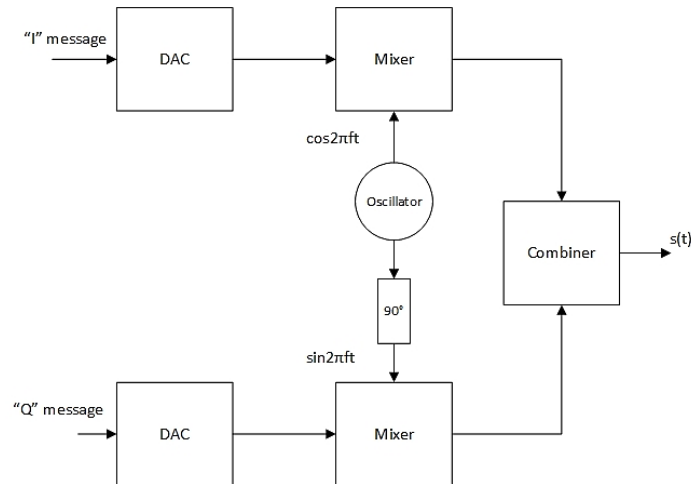


Figure 1: Architecture of Modulator at Transmitter

The in-phase and quadrature component of the signal are passed into the Amplitude Modulator and the signal $s(t)$ is produced as follow:

$$s(t) = s_I(t) \cos(2\pi f_c t) - s_Q(t) \sin(2\pi f_c t)$$

Eq. 2

In this lab, we only modulate the real part (in-phase) of the message, so the signal $s(t)$ will be:

$$s(t) = s_I(t) \cos(2\pi f_c t)$$

Eq. 3

On the receiver side, the demodulator consists of a splitter, which duplicates the input signal at the two outputs, and two mixers, which act as simple multipliers. Figure 2 shows this simple receiver architecture.

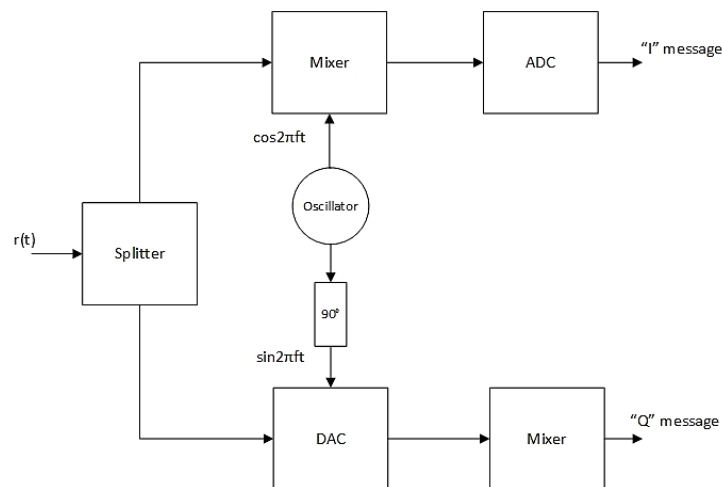


Figure 2: Architecture of Demodulator at Receiver

A more detailed architecture of the receiver is shown in:

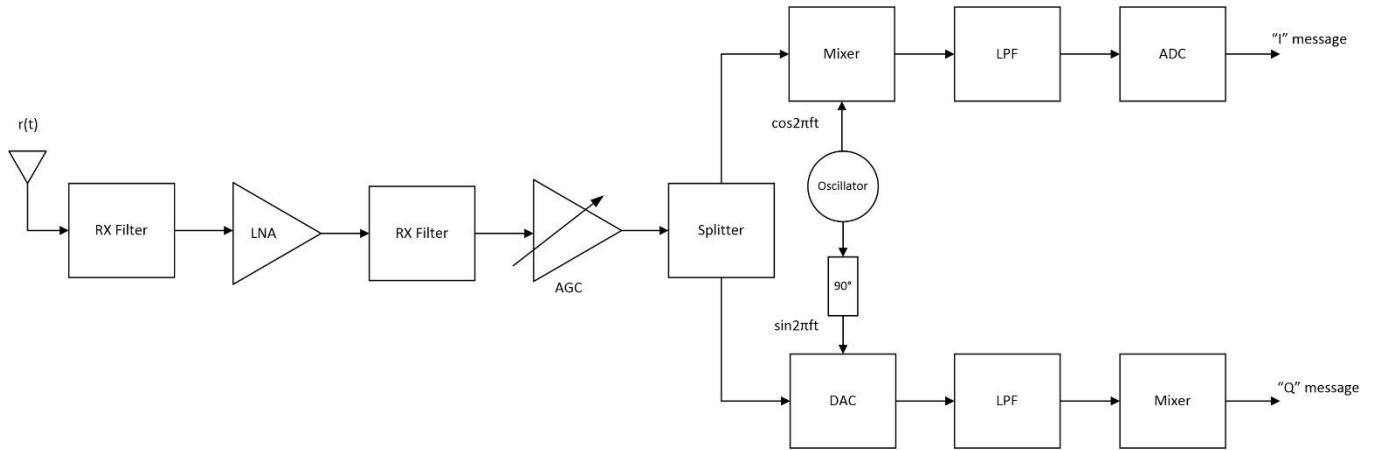


Figure 3: More Detailed Architecture of a Receiver

Where a Low Noise Amplifier (LNA) and Low Pass Filters (LPF) have been added to step up the voltage of received signal $r(t)$, usually in microvolt, to volt and filter out undesired signals around f_c . Automatic Gain Control (AGC) is also used to adjust the gain to compensate for various level of $r(t)$.

Amplitude Modulation (AM):

AM is one of the oldest modulation methods where the overall amplitude of the carrier wave varies linearly about a mean value in accordance with the message signal to carry the message [1]. AM, in analog form, is in use today in AM broadcast radios, and in digital form, it is the most common method for transmitting data over optical fibers.

If $m(t)$ is a baseband “message” signal and $c(t) = A_c \cos(2\pi f_c t + \varphi)$ is a sinusoidal “carrier” wave at carrier frequency f_c and amplitude A_c , then the transmitted AM signal $s(t)$ is of the form:

$$s(t) = A_c[1 + k_a m(t)] \cos(2\pi f_c t + \varphi)$$

Eq. 4

where k_a is a constant called the **amplitude sensitivity** of the modulator which generates the modulated signal. The amplitude of the modulated wave, $A_c[1 + k_a m(t)]$ is called the **envelope** and as it’s obvious that the information-bearing signal $m(t)$ resides only in the envelope.

From Eq. 4, we observe that the envelope of $s(t)$ has the same shape as the message signal $m(t)$ if the following conditions are satisfied:



1. $|k_a m(t)| < 1$

This condition ensures that $1 + k_a m(t)$ is always positive. When this condition is not satisfied ($|k_a m(t)| \geq 1$), the carrier wave becomes overmodulated which results in phase reversal when $1 + k_a m(t)$ crosses zero [1].

2. The carrier frequency f_c is much higher than the highest frequency component of the message $m(t)$:

$$f_c \gg BW_{m(t)}$$

Eq. 5

where $BW_{m(t)}$ is the message bandwidth. This condition ensures that the envelope is visualized and detected, and also ensures that lower sideband and upper sideband do not overlap [1].

For a special case in which $m(t) = m_p \cos(2\pi f_m t)$ and $\varphi = 0$, we can write:

$$s(t) = A_c [1 + \mu \cos(2\pi f_m t)] \cos(2\pi f_c t)$$

Eq. 6

where

$$\mu = k_a m_p$$

Eq. 7

is called the **modulation index** and takes values in the range of $0 \leq \mu \leq 1$ to satisfy the first condition.

From Eq. 6 we can write:

$$s(t) = A_c \cos(2\pi f_c t) + \frac{A_c}{2} \mu \cos[2\pi(f_c - f_m)t] + \frac{A_c}{2} \mu \cos[2\pi(f_c + f_m)t]$$

Eq. 8

In the above expression, the first term is the carrier, and the second and third terms are the lower and upper sidebands, respectively. The calculated average power delivered to a 1 ohm resistor is:

$$P_{Avg} = \underbrace{\frac{1}{2} A_c^2}_{\text{Carrier power}} + \underbrace{\frac{1}{8} \mu^2 A_c^2}_{\text{Lower sideband power}} + \underbrace{\frac{1}{8} \mu^2 A_c^2}_{\text{Upper sideband power}}$$

Eq. 9



As it's clear from Eq. 8 and Eq. 9, the first term doesn't carry any information but requires $\frac{1}{2}A_c^2$ power.

Figure 4 and Figure 5 show a 20 kHz carrier modulated by a 1 kHz sinusoid at 100% ($\mu = 1$) and 50% ($\mu = 0.5$) modulation respectively:

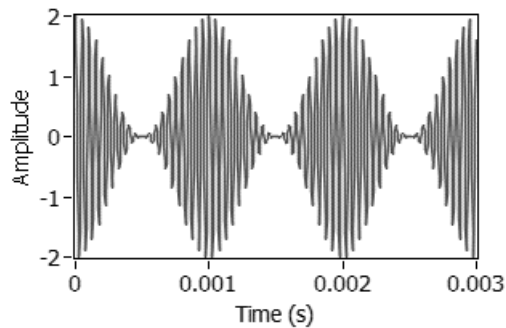


Figure 4: AM with 100% ($\mu=1$) modulation

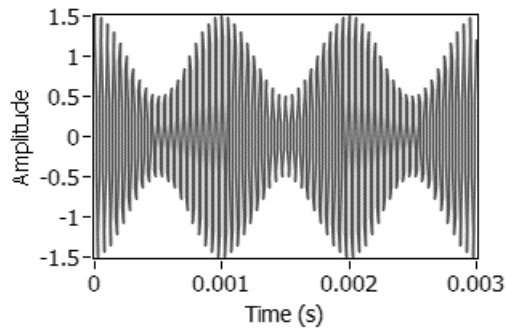


Figure 5: AM with 50% ($\mu=0.5$) modulation

At the receiver, the message $m(t)$ is recovered from the received signal $r(t)$. When the AM signal arrives at the receiver, it has the form of:

$$r(t) = A_r \left[1 + \mu \frac{m(t)}{m_p(t)} \right] \cos(2\pi f_c t + \theta)$$

Eq. 10

where the angle θ represents the difference in phase between the transmitter and receiver carrier oscillators.

We will follow a common practice in receivers and offset the receiver's oscillator frequency f_o from the transmitter's carrier frequency f_c , this method will be discussed in details in Lab 4, Image Rejection. This method provides the signal:

$$r_I(t) = A_r \left[1 + \mu \frac{m(t)}{m_p(t)} \right] \cos(2\pi f_{IF}t + \theta)$$

Eq. 11

where the so-called **intermediate** frequency is given by $f_{IF} = f_c - f_o$. The signal $r_I(t)$ can be passed through a bandpass filter to remove interference from unwanted signals on frequencies near f_c . Usually the signal $r_I(t)$ is amplified because $A_r < A_c$ due to signal attenuation as it passes through the channel.

There are some methods for amplitude demodulation, one of those methods is to detect the envelope and subtract the DC component to obtain the message. An envelope detector can be implemented as a rectifier followed by a Low Pass Filter (LPF). Figure 6 shows an envelope detector circuit which consists of a diode and a low pass RC filter.

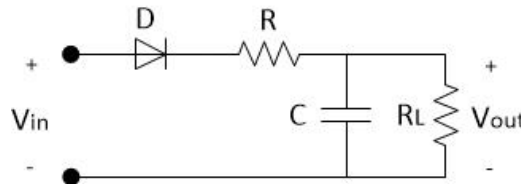


Figure 6: Envelope Detector Circuit

In this course, we implement the envelop detector receiver through the software. The envelope of the $r_I(t)$ is given by $A(t)$:

$$A(t) = A_r \left[1 + \mu \frac{m(t)}{m_p(t)} \right] = A_r + \frac{A_r \mu}{m_p(t)} \times m(t)$$

Eq. 12

which contains the DC component A_r . So, by removing the DC component, the message $m(t)$ will be recovered.

Pre-Lab:

Design Note: So as the result, each team needs to generate AM wave at the transmitter, then receive and demodulate the signal at the receiver. The discrete time signal is generated in LabVIEW and discrete to continuous conversion (DAC) is implemented inside the USRP modules. As it's mentioned earlier, the baseband signal consists of the real part and the imaginary part but in this lab, we only modulate the real part of the message. So, for the transmitter, the AM signal that each team needs to generate is:

$$\tilde{s}_I(nT) = A_c \left[1 + \mu \frac{m(t)}{m_p(t)} \right]$$

Eq. 13

Based on the Eq. 3, we can write:

$$\tilde{s}(nT) = \tilde{s}_I(nT) \cos(2\pi f_c t) = A_c \left[1 + \mu \frac{m(t)}{m_p(t)} \right] \cos(2\pi f_c t)$$

Eq. 14

where f_c and T are carrier frequency and time interval reciprocal of the **IQ rate**.

For the receiver, each team needs to design an envelope detector software to extract the envelope of the signal from the Eq. 11 (received message):

$$r_I(t) = A_r \left[1 + \mu \frac{m(t)}{m_p(t)} \right] \cos(2\pi f_{IF} t + \theta)$$

then rectify and pass it through a LPF.

LabVIEW Note: In order to be able to run all the VI's and use all the LabVIEW icons, run LabVIEW 2015 (32bit) version for this lab.

In order to make an AM transmitter, the first task is to add blocks as needed to the transmitter template provided in the file **Lab2TxTemplate.vi**, and then the pass the AM signal into the **While Loop** to the **Write Tx Data** block. The task to design an AM receiver is to add proper blocks to the receiver template provided in the file **Lab2RxTemplate.vi** to receive and construct the message being transmitted. The following sections describe how to set up the transmitter and receiver step-by-step.



Prior to beginning this lab, every student is expected to know how to connect the USRP modules to their laptops and have the LabVIEW software ready.

Setting up the USRP

Transmitter:

The transmitter template which is shown in Figure 7 contains four interface VI's. LabVIEW interacts with the USRP transmitter by means of these four VI's located in Instrument I/O > Instrument Drivers > NI-USRP > TX. Figure 7 is the starting point for all of the laboratory transmitter exercises in this series.

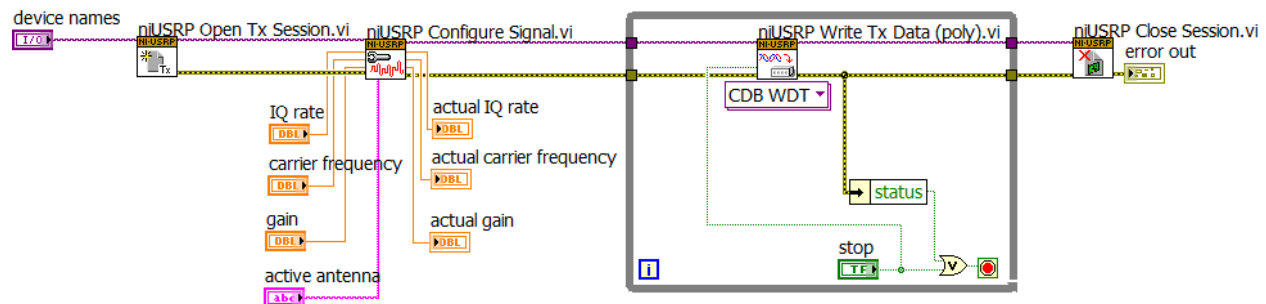


Figure 7: Transmitter Template

Appendix I describes these blocks in details, but we review them here as well:

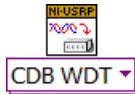


Open TX Session.vi initiates the transmitter session and generates a session handle and an error cluster that are propagated through all VI's. When you use this VI, you must add a Numeric Control icon called “device names” that you will use to inform LabVIEW of the IP address of the USRP (192.168.10.2).



Configure Signal.vi is used to set parameter values in the radio. Attach four Numeric Controls and three Indicators to this VI as shown in Figure 7. To get started, set the IQ rate to 200 kS/sec (the lowest possible rate), the carrier frequency to 915.1 MHz, the gain to 20 dB, and the active antenna to TX1. When the VI runs, the radio will return the actual values of these parameters. These values will be displayed by the indicators you connected. Normally the actual parameter values will match the desired values, but if one or more of the desired values is outside the capability of the radio, the radio will choose the nearest acceptable parameter value, rather than return an error.





Write Tx Data.vi writes the Base Band signal to the USRP for transmission. Placing this VI in a while loop allows a block of Base Band signal samples to be sent over and over until the **STOP** button is pressed. Note that the while loop will also terminate if an error is detected. Base Band signal samples can be provided to the Write Tx Data VI as either an array of complex numbers or as a complex waveform data type. A pull-down tab allows you to choose the data type.

As the transmitted signal is described by Eq. 14, the constant A_c is set by the **gain** parameter and f_c is the carrier frequency.



Close Session.vi terminates transmitter operation once the while loop ends. Note that the vi should be terminated using the STOP button rather than with **Abort Execution** on the toolbar. This is so that the Close Session.vi will be sure to run and will correctly close out the data structures that the VI uses.

Notes for Setting up the Transmitter:

Follow the following steps to create your transmitter:

1. The transmitter template contains some blocks called **Message Generator** to generate the required messages. The Message Generator is set to produce a message signal ($m(t)$) consisting of three tones which are initially set to 1, 2, and 3 kHz, but these frequencies can be changed using front-panel controls. The modulation index (μ) is to be user-settable in the range $0 \leq \mu \leq 1$. The Message Generator blocks create a signal that is the sum of a set of sinusoids of equal amplitude. You can choose the number of sinusoids to include in the set, their frequencies, and their common amplitude. The initial phase angles of the sinusoids are chosen at random, however, and will be different every time you run the VI. This will make the message signal look somewhat different every time you run the VI. Figure 8 shows Message Generator blocks. In order to make it easier to set up the transmitter, the Message Generator is also provided in the transmitter template.



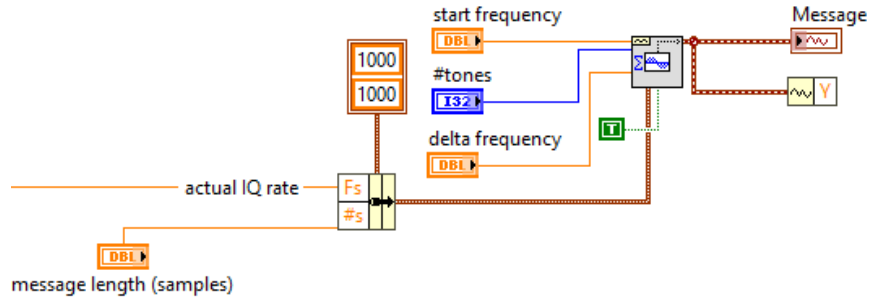


Figure 8: Message Generator

In order to get the data values of the generated signal, we use **Get Waveform Components** icon in the transmitter model (See Programming > Waveform > Get Wfm Comps). The Get Waveform Components icon returns the actual analog waveform and the components of the input signal. You need to use this icon when you want to work on the actual mathematical signals. So, as the result the Message Generator generates the message signal, $m(t)$.

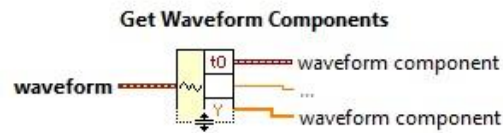


Figure 9: Get Waveform Components Icon

2. As we discussed, the Eq. 13 should be the final message that is sent to the USRP. In order to generate the Eq. 13, you need to calculate the maximum amplitude of the generated signal, $\{m_p\}$. Use **Quick Scale** icon (See Functions > Signal Processing > Sig Operation > Quick Scale) to calculate the maximum absolute value of the generated signals.

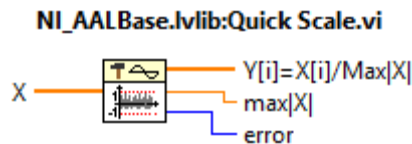


Figure 10: Quick Scale

As Figure 10 shows, the second output of the Quick Scale icon gives us the absolute value of the input signal. So, if the input is $m(t)$, the output will be $m_p(t)$. Make the wire connection from the output of the Get Waveform Component to the input of this block.

3. Now, we have the $m(t)$ and $m_p(t)$, and need to generate the Eq. 13 as follow:

$$\tilde{s}_l(nT) = A_c \left[1 + \mu \frac{m(t)}{m_p(t)} \right]$$

In order to calculate the $\tilde{s}_l(nT)$, set up a **MathScript Node** (See Functions > Mathematics > Script & Formulas > MathScript). MathScript Node evaluates and executes the text-based scripts in the LabVIEW. As we discussed earlier, it's sometimes easier to create scripts than to work with the graphical objects. MathScript Node gives us the opportunity to use text-based scripts in the LabVIEW graphical environment. Set up your MathScript Node to include the data values of the generated signal $\{m\}$, maximum value of the generated signal $\{m_p\}$, and the modulation index $\{mu\}$ or $\{\mu\}$ as inputs. To define inputs, right click on the blue border of the icon and select **Add Input** and name them m_p , m and mu . At this step, you're not able to define the proper output, so just be patient. Figure 11 shows the MathScript Node icon needs to be filled with syntax codes. Please note that you need to use LabVIEW 2015 (32bit) to have the license for this icon.



Figure 11: MathScript Icon

4. Now you need to fill the MathScript with the proper syntaxes as follows:

- I. Generating the part inside the parentheses in Eq. 13 is easy:

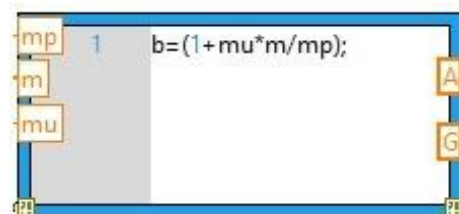


Figure 12: MathScript Icon First Line

Note that $A_c = 1$ for the transmitter.

- II. Before writing any more scripts, note that there is one practical constraint imposed by DAC in the USRP which limits the amplitude of the input signals to 1, so each team is required to scale the generated signals so that the peak value of $|\tilde{s}(nT)|$ does not exceed 1. Set up a text-based script to divide the Base Band signal by the maximum of its absolute value $\{max(abs(b))\}$ to get the scaled Base Band signal. Figure 13 shows an example of such code:

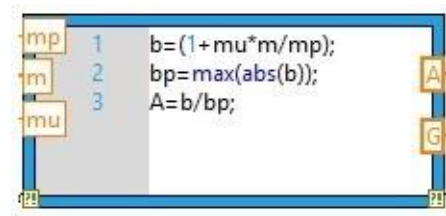


Figure 13: MathScript Icon Third Line

- III. The USRP is designed to transmit using a quadrature modulation approach, so in order to use the radio to transmit an AM signal, it is necessary to represent the signal as a complex sequence, however for quadrature part, we only send zero. The quadrature modulation then transmits the real and complex sequences using two orthogonal waveforms. The real part is sent using a cosine carrier and the complex part is sent using a sine function as the carrier. For the MathScript Node, you need to set up a text-based script to convert the scaled amplitude modulated signal from 1D double to 1D complex double form. The 1D complex double form is attained by multiplying the 1D double form by $\{e^{j*0}\}$.

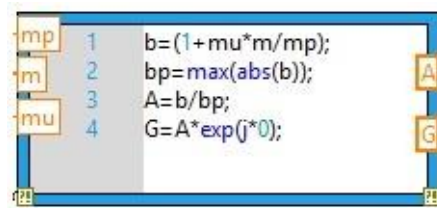


Figure 14: MathScript Node Fourth Line

- IV. Set up both forms of the scaled Base Band signal (A and G as scaled and complex signal respectively) as the outputs of the MathScript Node. For this purpose, right click on the

blue boarder and select **Add Output** and choose A and G among the suggested options.

5. Connect the second output of the Quick Scale to m_p input.
6. For m , use the output of the Get Waveform Block.
7. A Numeric Control input is provided in the template for the **Modulation Index**. Wire the m_u with the Modulation Index icon.
8. Plot the scaled Base Band signal (A) using a Waveform Graph icon, and name it **Base Band Signal**.
9. Input the complex Base Band signal (G) to the data input of the **niUSRP Write Tx** to perform the transmission.
10. Save your transmitter in a file whose name includes the letters “AMTx” and your initial. Be sure to save your transmitter as a “vi” and not a “vit” (template). ***Note: You need to upload your program file with your lab report.***

In the next section, we'll investigate the receiver and run the software.



Receiver:

The receiver template which is shown in Figure 15 contains six interface VI's which are located in Instrument I/O > Instrument Drivers > NI-USRP > Rx. The structure shown in Figure 15 is the starting point for all of the laboratory receiver exercises in this series.

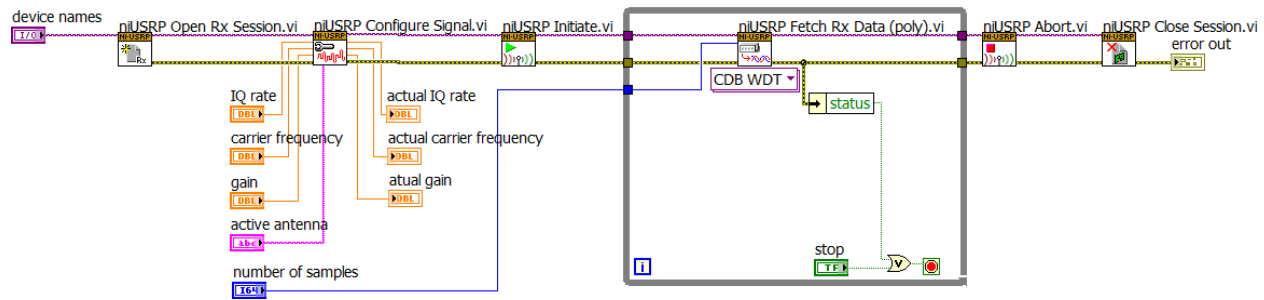


Figure 15: Receiver Template

Appendix I describes these blocks in details, but we review them here as well:



Open Rx Session.vi initiates the receiver session and generates a session handle and an error cluster that are propagated through all six VI's. When you use this VI, you must add a Numeric Control icon called “device names” that you will use to inform LabVIEW of the IP address of the USRP (192.168.10.2).



Configure Signal.vi has the same function as the corresponding VI in the transmitter. Attach four Numeric Controls and three Indicators to this VI in Figure 15. This time, set the IQ rate to 1 MS/s, the carrier frequency to 915.0 MHz, the gain to 0 dB, and the active antenna to RX2. When the VI runs, the radio will return the actual values of these parameters.



Initiate.vi sends the parameter values you selected to the receiver and starts it running.



Fetch Rx Data.vi retrieves the message samples received by the USRP. Placing this VI in a while loop allows the message samples to be retrieved one block at a time until the **STOP** button is pressed. Note that the while loop will also terminate if an error is detected. A **number of samples** control allows you to set the number of samples that will be retrieved with each pass through the while loop. Message

samples can be provided to the user as either an array of complex numbers or as a complex waveform data type. A pull-down tab allows you to choose the data type for the message samples.



Abort.vi stops the acquisition of data once the while loop ends.



Close Session.vi terminates receiver operation. As noted above, use the **STOP** button to terminate execution so that Close Session will be sure to run.

Notes for Setting up the Receiver:

Follow the following steps to create your receiver:

1. First you need to pass the received signal through a Band Pass Filter to remove interference from unwanted signals on frequencies near f_c . So, pass the array returned by the **Fetch Rx Data.vi** through a Band Pass Filter. Note that this array is a complex vector with Real (In-Phase) and imaginary (Quadrature) parts. Use a **Chebyshev** filter (Functions > Signal Processing > Filters) which is shown in Figure 16. Right click on the icon and enable **All Constants** from the **Create** menu and set up the filter as a fifth-order Band Pass Chebyshev filter with a high cutoff frequency of 105 kHz and a low cutoff frequency of 95 kHz. The default passband ripple is 0.1 dB which is acceptable for our case. The sampling frequency input to the filter should be the **actual IQ rate** obtained from the **Configure Signal.vi** (outside of the while loop). Place the wire connection between the output of the **Get Waveform Components, Y** and the input **X**.

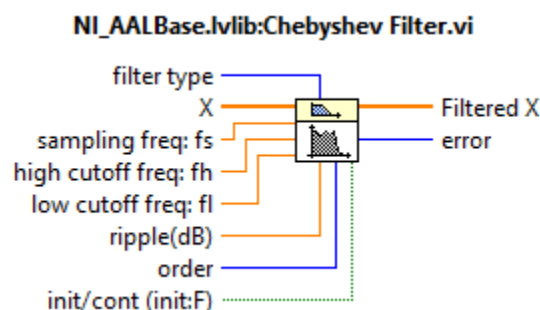


Figure 16: Chebyshev Filter

2. The output of the Chebyshev filter is a complex vector ($x = a + bi$) while you only have real data, so you need to extract the real part from the complex signal. The **Complex to Re/Im** icon



(Functions > Programming > Numeric > Complex > Complex to Re/Im) extracts the real and imaginary parts of a signal. The real part you obtain can be expressed as Eq. 11 where $A_r < A_c$ due to signal attenuation as it passes through the channel. So, wire the Complex to Re/Im icon to read the output of the Chebyshev filter and give you the real value of the signal.

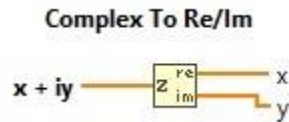


Figure 17: Complex to Re/Im Icon

3. As it's mentioned earlier, one way of Amplitude Demodulation is to detect and extract the envelope. So, in order to extract the envelope, you need to take the absolute value of the signal, which functions as a full-wave rectifier, and pass the result through a Low Pass Filter (LPF). Use the **Absolute Value** (Functions > Programming > Numeric > Absolute Value) to calculate the absolute value of the real part of the signal.



Figure 18: Absolute Value Icon

4. Use a **Butterworth** filter (Functions > Signal Processing > Filters > Butterworth Filter) as the LPF. Right click on the filter icon and enable **All Constants** from the **Create** menu and set up the filter as a second-order Low Pass filter with a low cutoff frequency of 5 kHz. As was the case for the BPF, the sampling frequency input should be the actual IQ rate obtained from the Configure Signal.vi (Outside of the while loop). The output of the Butterworth LPF should be connected to the **Build Waveform** block provided in the template that feed the **Base Band Output** graph. Figure 19 shows the Butterworth filter.
5. Make the proper wire connection between the **dt** component of the Get Waveform block and Build Waveform block. If the Get Waveform block doesn't have the "dt" output, simply put the mouse cursor on the block and expand it to add another output, then click on the new output and select dt.

Figure 20 illustrates how to activate it.

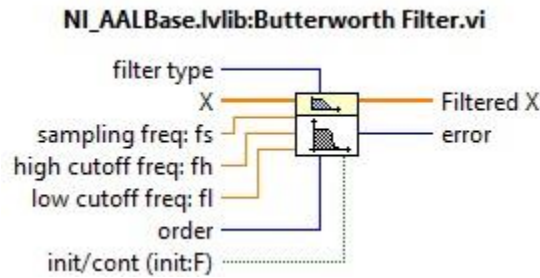


Figure 19: Butterworth Filter Icon

6. Save your receiver in a file whose name includes the letters “AMRx” and your initial. Be sure to save your receiver as a “vi” and not a “vit” (template). **Note: You need to upload your program file with your lab report.**

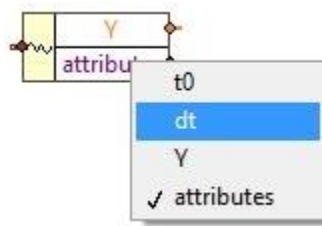


Figure 20: Creating dt output for Get Waveform block

Lab Procedure:

Every team is given a kit containing the required components to do the experiments. At the end, each team is supposed to provide a report based on the experiment covered in this session.

The Kit Contents:

Every team is given the following devices and materials:

1. One NI USRP-2922
2. AC/DC power supply
3. Shielded Ethernet cable
4. SMA (m)-to-SMA (m) cable
5. 30 dB SMA attenuator
6. Lab manual

Run the Test:

1. Connect a loopback cable and the attenuator between the TX 1 and RX 2 antenna connectors.

Connect the USRP to your computer with an Ethernet cable and plug in the power to the radio.

Make sure that the Ethernet port on your computer is set to use IP address 192.168.10.1. At this point LED's **D** (firmware loaded) and **F** (power on) should be illuminated, as should the green light on the left side of the Ethernet connector. Run LabVIEW and open the transmitter and receiver VI's that you created in the prelab.

2. Ensure that the transmitter VI is set up based on the Table 1.

Table 1: TX Settings

Field	Setting	Field	Setting
Device Name	192.168.10.2	Message Length	200,000 samples
IQ Rate	200 kHz	Modulation Index	1.0
Carrier Frequency	915.1 MHz	Start Frequency	1 kHz
Gain	20 dB	Delta Frequency	1 kHz
Active Antenna	TX1	Number of Tones	1

3. Ensure that the receiver VI is set up based on the Table 2.

Table 2: RX Setting

Field	Setting
Device Name	192.168.10.2
IQ Rate	1 MHz
Carrier Frequency	915 MHz
Gain	0 dB
Active Antenna	RX2
Number of Samples	200,000 samples



4. Run the receiver vi, then run the transmitter vi. LED C will illuminate on the USRP if the radio is receiving data. After a few seconds, stop the receiver using the STOP button, then stop the transmitter. Use the horizontal zoom feature on the graph palette to expand the **Message** waveform in the transmitter vi and the **Base Band output** waveform in the receiver. Both waveforms should be identical, except for scaling and the fact that the receiver output has a DC offset. Use this graph as the initial result in your report. (1 point)
5. Do the following tasks and use the results in your final report as well:
 - I. **Multi-Tone message:** Increase the tone to two and then three and repeat the test. Use the results in your report. (1 point)
 - II. **Varying the modulation index:** Set the modulation index to 0.1, run your software and measure the amplitude of the Base Band Output signal at the receiver. Start incrementing the modulation index by 0.1 and fill up Table 3. Describe what you observe in your report. (1 point)

Table 3

Modulation Index	Amplitude (Peak to Peak)
0.1	
0.2	
0.3	
0.4	
0.5	
0.6	
0.7	
0.8	
0.9	
1.0	

- III. **Over modulating:** Choose the modulation index to not satisfy the condition in page 5, and



causes over modulation. Describe why over modulation occurs including screenshot of the result. (1 point)

IV. Elegant receiver: The signal at the output of the BPF has a real part which is given by Eq.

11. The imaginary part is given by:

$$r_2(t) = A_r \left[1 + \mu \frac{m(t)}{m_p(t)} \right] \sin(2\pi f_{IF}t + \theta)$$

Eq. 15

The complex signal at the output of the BPF is therefore:

$$\begin{aligned} \tilde{r}(t) &= r_1(t) + jr_2(t) \Rightarrow \\ &= A_r \left[1 + \mu \frac{m(t)}{m_p(t)} \right] \cos(2\pi f_{IF}t + \theta) + jA_r \left[1 + \mu \frac{m(t)}{m_p(t)} \right] \sin(2\pi f_{IF}t + \theta) \Rightarrow \\ &= A_r \left[1 + \mu \frac{m(t)}{m_p(t)} \right] [\cos(2\pi f_{IF}t + \theta) + j\sin(2\pi f_{IF}t + \theta)] \Rightarrow \\ &= A_r \left[1 + \mu \frac{m(t)}{m_p(t)} \right] e^{j(2\pi f_{IF}t + \theta)} \end{aligned}$$

The magnitude of $\tilde{r}(t)$ is:

$$A_r \left[1 + \mu \frac{m(t)}{m_p(t)} \right]$$

Eq. 16

which is the desired demodulated output. Return to your receiver and connect the BPF output directly to the absolute value block (do not take the real part). Connect the absolute value output directly to the Base Band Output graph. Run the transmitter and receiver, and observe whether the demodulated output is the same as it was in step 6. Include screenshots of your modified software and the result in your report. (2 points)

Report (4 points):

The reports must be prepared in PDF or Microsoft Word format and should follow the standard of laboratory report. The report shall consist of:

1. A title page, including the name and number of the lab



2. The object, purpose and goal of the experiment shall be on the second page
3. A quick description on how the experiment is being done in 1-2 paragraphs
4. Calculations
5. Graphs
6. Pictures
7. Results
8. Conclusion
9. Tx and Rx VI files

All reports are due one week after the receiver experiment is completed and each team must submit only one report along with all the results and necessary files in a zip format file to the Moodle page.

References

[1] S. Haykin and M. Moher, Introduction to Analog & Digital Communications, 2nd ed., John Wiley & Sons, Inc.

