Lab 2: Amplitude Modulation (AM)

Dennis Joosens[1,*] **and Maarten Weyn**[1]

[1]Electronics-ICT, Faculty of Applied Engineering, University of Antwerp, Belgium
[*]corresponding. (dennis.joosens@uantwerpen.be)

ABSTRACT

This laboratory exercise has two objectives: The first is to gain a firsthand experience in actually programming the USRP to act as a transmitter and a receiver. The second is to investigate classical analog amplitude modulation and the envelope detector.

1 Introduction

Amplitude Modulation (AM) is one of the oldest of the modulation methods. It is still in use today in a variety of systems, including, of course, AM broadcast radio. In digital form it is the most common method for transmitting data over optical fiber.
If $\boldsymbol{m(t)}$ is a **baseband message** signal with a **peak value** of $\boldsymbol{m_P}$ and $\boldsymbol{A_c cos(2\pi f_c t)}$ is a **carrier signal** at carried frequency $\boldsymbol{f_c}$, then we can write the **AM signal** $\boldsymbol{g(t)}$ as:

$$g(t) = A_c \left[1 + \mu \frac{m(t)}{m_P}\right] \cos(2\pi f_c t) \tag{1}$$

where the parameter $\boldsymbol{\mu}$ is called the **modulation index** and takes values in the range $\boldsymbol{0 \leq \mu \leq 1}$ in normal operation. If $m(t) = m_P \cos(2\pi f_m t)$ then the AM signal will be expressed as[1]

$$\begin{aligned} g(t) &= A_c \left[1 + \mu \cos(2\pi f_m t)\right] \cos(2\pi f_c t) \\ &= A_c \cos(2\pi f_c t) + \frac{A_c}{2} \mu \cos(2\pi (f_c - f_m) t) + \frac{A_c}{2} \mu \cos(2\pi (f_c + f_m) t) \end{aligned} \tag{2}$$

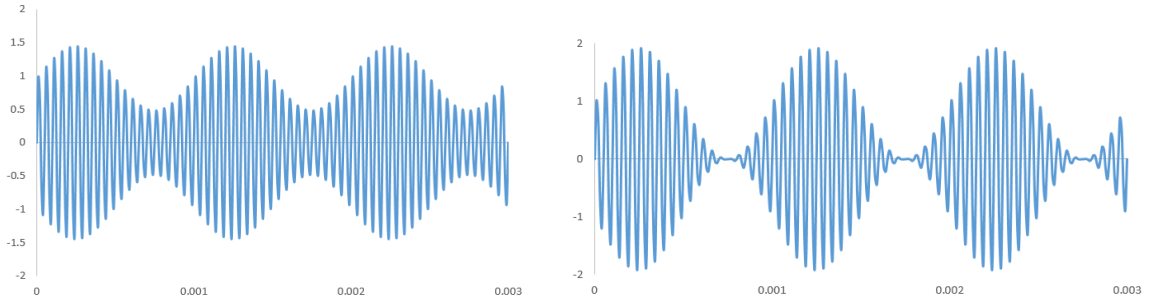
In the above expression the first term is the **carrier**, and the second and third terms are the **lower (LSB) and upper (USB) sidebands**, respectively. Fig.(1) is a plot of a 20 kHz carrier modulated by a 1 kHz sinusoid at 50% and 100% modulation index.

When the AM signal arrives at the receiver, it has the form

$$r(t) = \hat{A} \left[1 + \mu \frac{m(t)}{m_P}\right] \cos(2\pi f_c t + \theta) + n(t) \tag{3}$$

where the carrier amplitude $\hat{A}$ is usually much smaller than the amplitude A of the transmitted carrier, the angle θ represents the difference in phase between the transmitter and receiver carrier oscillators and $n(t)$

[1]using the mathematical product-to-sum identities
$\cos(x) \times \cos(y) = \frac{1}{2}(\cos(x-y) + \cos(x+y))$
$\sin(x) \times \sin(y) = \frac{1}{2}(\cos(x-y) - \cos(x+y))$
$\sin(x) \times \cos(y) = \frac{1}{2}(\sin(x+y) + \sin(x-y))$
$\cos(x) \times \sin(y) = \frac{1}{2}(\sin(x+y) - \sin(x-y))$



(a) $\mu = 0.5$ **(b)** $\mu = 1$

Figure 1. AM Modulated Signal

represents the noise. We will follow a common practice and offset the receiver's oscillator frequency f_o from the transmitter's carrier frequency f_c . This provides the signal

$$\hat{r}(t) = \hat{A}\left[1 + \mu\frac{m(t)}{m_P}\right]\cos(2\pi f_{IF}t + \theta) + n(t) \tag{4}$$

where the so-called **intermediate frequency** is given by $\boldsymbol{f_{IF} = f_c - f_o}$.
At the receiver, the signal $\hat{r}(t)$ can be passed through a band pass filter to remove interference from unwanted signals on frequencies near f_c. Usually the signal $\hat{r}(t)$ is amplified as well. Demodulation of the signal $\hat{r}(t)$ can be done either by using an envelope detector or using synchronous AM detector. An envelope detector can be implemented as a rectifier followed by a low pass filter. The envelope $E(t)$ of $\hat{r}(t)$ is given by

$$E(t) = \hat{A}\left[1 + \mu\frac{m(t)}{m_P}\right] = \hat{A} + \frac{\hat{A}\mu}{m_P}m(t) \tag{5}$$

2 Lab Procedure

1. **AM Simulation**

   This exercise is to simulate an AM signal and study the AM signal parameters.

   AM Transmitter

   - Open a new flowgraph in GNU Radio.
   - Set **Generate Options** to **QT GUI**.
   - Set the **samp_rate** of the ***variable*** block to **200k**.
   - Add another ***variable*** block, call it **freq** and set it to **2k**.
   - Add another ***variable*** block, call it **Tx_ freq** and set it to **20k**.
   - Add a ***Signal Source*** and set the **Output Type** to **float** numbers, **Waveform** to **cosine**, **Frequency** to **freq**, **Amplitude** to **1** and finally **Offset** to **0**.
   - Add a ***Multiply Const***, set **Constant** to **0.5**. Change the **IO Type** to **float**.
   - Add a ***Constant Source*** and set the **Output Type** to **float** numbers, set **Constant** to **1**.
   - Add an ***Add*** block. Set the **IO Type** to **float** numbers. Connect the ***Constant Source*** and ***Multiply Const*** to the ***Add*** block.
   - Add a ***Multiply*** block. Set the **IO Type** to **float** numbers.
   - Add a ***Signal Source*** and set the **Output Type** to **float** numbers, **Waveform** to **Cosine**, **Frequency** to **Tx_ freq**, **Amplitude** to **1** and finally **Offset** to **0**.
   - Add a ***QT GUI Sink*** and set **Type** to **float** and **FFT Size** to **2.048 kHz**.

   The above blocks construct an AM transmitter by applying Eq.(1). Observe the waveform in the time and frequency domain.

   Each block in GNURadio has a **comment section**. To find this, look at the **Advanced** tab of a block. Identify the functionality of each block in the design by placing a small note. This note will be visible in the flowgraph.

AM Receiver

- Use the AM transmitter flowgraph.
- Add another ***variable*** block, call it **Rx_ freq** and set it to **15k**.
- Add a ***Signal Source*** and set the **Output Type** to **float** numbers, **Waveform** to **Cosine**, **Frequency** to **Rx_ freq**, **Amplitude** to **1** and finally **Offset** to **0**.
- Add a ***Multiply*** block. Set the **IO Type** to **float** numbers.
- Add a ***Low Pass Filter*** set **FIR Type** to **float**, **Cutoff Freq** to **10k** and **Transition width** to **1k**.
- Add an ***Abs*** block. Set the **IO Type** to **float** numbers.
- Add a ***Low Pass Filter*** set **FIR Type** to **float**, **Cutoff Freq** to **2k** and **Transition width** to **1k**.
- Add a ***QT GUI Sink*** and set **Type** to **float** and **FFT Size** to **2.048 kHz**.

At the end you should have a flowgraph as shown in figure(2). Run the flowgraph and get the output in the time and frequency domains at the points A, B, C, D, E, and F as shown in figure (3). Change the modulation index to 0.25, 0.75 and 1. What is the difference on the AM signal waveform.

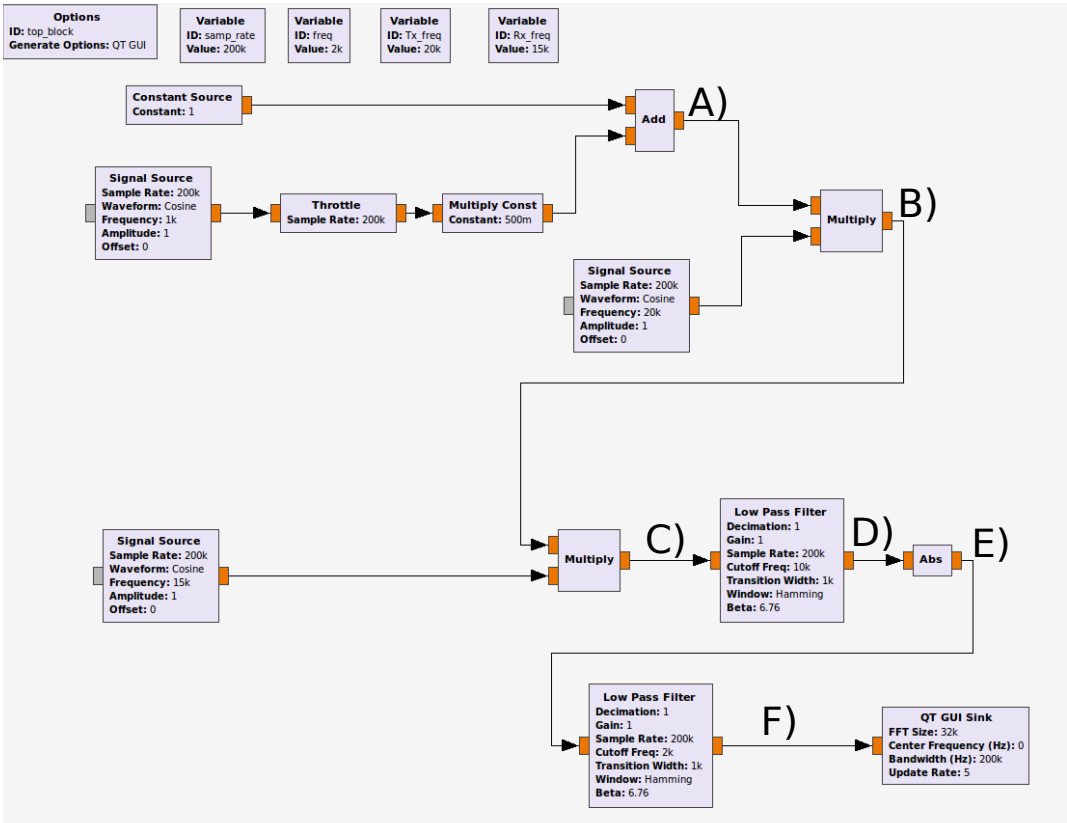


Figure 2. AM modulation in GRC

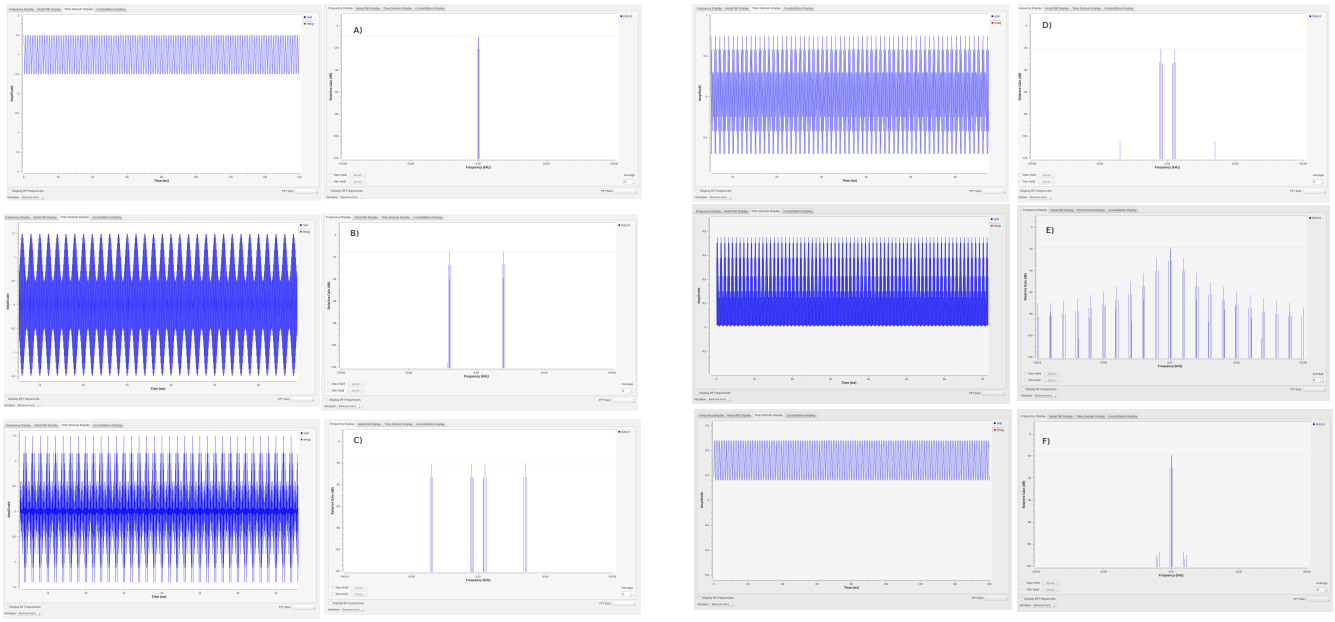


Figure 3. The frequency and the time domains for each point in figure (2)

2. **Transmitting an AM signal via USRP**

   This exercise is to transmit and receive an AM signal via the USRP.

   AM Transmitter

   - Open a new flowgraph in GNU Radio.
   - Set **Generate Options** to **QT GUI**.
   - Set **samp_rate** variable to **500k**.
   - Add another ***variable*** block, call it **freq** and set it to **5k**.
   - Add another ***variable*** block, call it **Tx_freq** and set it to **820.1M**.
   - Add a ***Signal Source*** and set the **Output Type** to **float** numbers, **Waveform** to **Cosine**, **Frequency** to **freq**, **Amplitude** to **1** and finally **Offset** to **0**.
   - Add a ***Constant Source*** and set the **Output Type** to **float** numbers, **Constant** to **1**.
   - Add an ***Add*** block. Change the **IO Type** to **float**.
   - Add ***Multiply Const*** and set the **IO Type** to **float** numbers, **Constant** to **0.5**.
   - Add a ***Float To Complex***.
   - Add ***QT GUI Sink*** and set **Type** to **float** and **FFT Size** to **2.048 kHz**.
   - Add ***UHD: USRP sink*** and set the **Input Type** to **complex float32**, **Center Freq** to **Tx_freq** and **Gain Value** to **20**.

   Connect the above blocks to transmit an AM signal via the USRP.

AM Receiver

- Use the AM transmitter flowgraph.
- Add another **variable** block, call it **Rx_freq** and set it to **820M**.
- Add ***UHD: USRP Source*** and set the **Output Type** to **complex float32**, **Center Freq** to **Rx_freq** and **Gain Value** to **0**.
- Add a ***Band Pass Filter*** set set **Low Cutoff Freq** to **90 kHz**, **High Cutoff Freq** to **110k** and **Transition width** to **10k**.
- Add a ***Complex To Real*** block.
- Add an ***Abs*** block. Change the **IO Type** to **float**.
- Add a ***Low Pass Filter***. Set **FIR Type** to **float**, **Cutoff Freq** to **6 kHz** and **Transition width** to **1 kHz**.
- Add a ***QT GUI Sink*** and set **Type** to **float** and **FFT Size** to **2.048 kHz**.
- Add a ***File Sink*** and set **File** to a destination where you want to save the file on your computer.

At the end you should have a flowgraph as shown in figure(4). Change the GUI sink position between locations A, B, C, D and E, and take screenshots for the output at every location of them. **Change the Modulation index** from 0.5 to 1 and run the flowgraph.

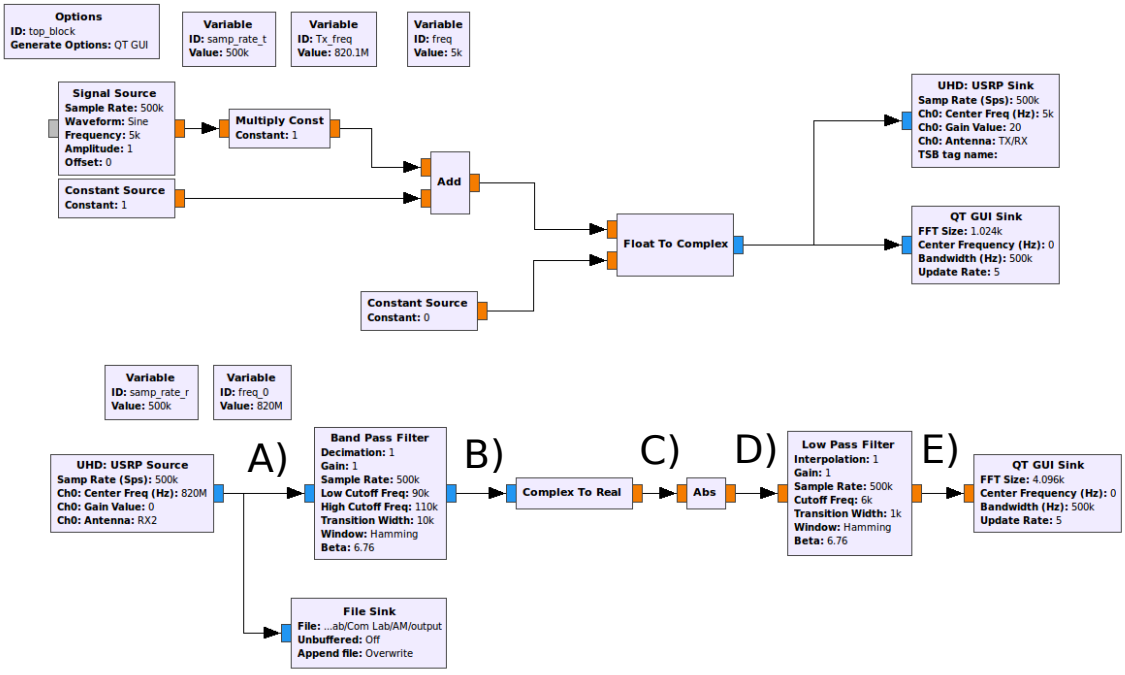


Figure 4. AM modulation in GRC using the USRP blocks

3 Report

Form a report by answering the following questions and submit it via **Blackboard**.

1. Suppose the baseband message $m(t)$ is given by $m(t) = \cos(2k\pi t) + \cos(4k\pi t) + \cos(6k\pi t)$. Assuming that:

   - $\mu = 0.5$
   - f_{IF} = 10 kHz
   - $\theta = 0$
   - $n(t) = 0$

   Build an AM simulator, similar to lab procedure 1.
   Change the values of the blocks so you can retrieve the original signal at the end of the receiver stage.
   Provide screenshots of the flowgraph and the outputs in the frequency domain and time domain at locations A, B, C, D, E and F.

2. Explain what f_{IF} is and why we get it in the received signal?
   What are the values of f_{IF} in the lab procedure section 1 and 2?

3. In the second lab procedure, you received from the USRP (the recorded file) with a modulation index 0.5 the following:

   $$\hat{r}(t) = [1 + 0.5\cos(10k\pi t)]\cos(200k\pi t)$$

   Prove mathematically that you can demodulate this signal **without** the need of the envelope detector (hint: check the mathematical identities on page 1).
   Apply this method in GNU Radio using the recorded file as your input (using ***File Source***, instead of ***UHD:USRP Source***).