Digital Communication Using GNU Radio

Prof Kumar Appaiah

Department of Electrical Engineering

Indian Institute of Technology Bombay

Week-01

Lecture-03

In this lecture, we are going to continue our exploration of GNU radio. Building on top of the knowledge that we have gained in the previous lecture, we are now going to explore several new blocks, in particular those that offer you flexible usage of variables and ranges that allow you to put together dynamic simulations wherein you can change the parameters of the simulation at runtime. We will also see some simple ways to interface hardware. In particular, we will be interfacing the sound card of your PC in order to perform hardware based experiments in GNU radio as well. Going back to our original program, let us now add some interactivity to the signal source. It will be useful for us to be able to change the signal source characteristics.

To do so, we need to change some aspects of the signal source and that too dynamically when the program is executing. Let us change the frequency of the source while we are running the program interactively. To this end, the first step that we will take is to change this 1000 and we will call it FREQ which is a keyword. As you can see, the signal source turns red because GNU radio cannot interpret this FREQ.

Since we want the frequency to be changed interactively, we will add a block to our flow graph called range. So Ctrl F or Cmd F and we will type range and we get the QT-GUI range and add it to our flow graph. We will double click this range and change the ID to the same name that we specified in our frequency source which is FREQ. The type should be float and since our sampling rate is 32K, we will make this go from 100Hz, start at 100Hz and stop at 16000Hz. We will leave the other settings unchanged and say OK.

We can now execute the flow graph and now at the top, you will see a slider with the title FREQ. By changing the slider, you can see that the frequency of the sinusoid ends up changing. You can also enter the frequency by selecting this area and just typing in the frequency that you desire. I have typed in 8000 to verify we will take this peak which is around 16.6 milliseconds.

The next peak is at 16.18 milliseconds, 16.6, 16.72 that is about 0.125 milliseconds that corresponds exactly to 8000Hz.

You can now interactively change the frequencies and view this plot. Let us add the grid and let us also add an auto scale so that you can see this. Now clearly, this does not look like a sinusoid at all. The reason is because even though the sinusoid is being captured correctly, the number of samples that will show that this is a sinusoid is insufficient. So, if you really want to visualize better, you can change the number of points over here.

Let us say to 4096 and now let us make the frequency 5000Hz. You can see that at 5000Hz, it is now a little more evident that it is like a sinusoid although there are still very sharp edges. Let us make this 4000Hz. You can clearly see that there is a triangle like feature although it is not really a triangle, you can see that it is the sinusoid in the background. The reason this happens is because unlike the picture that we saw, GNU radio interpolates samples linearly.

Therefore, if you do not have a sufficient number of samples, you will always end up with plots that do not resemble a sinusoid. That is not of any concern because implicitly the sinusoid lies in the background and as long as you satisfy Nyquist criterion, there is no loss of information. Let us convert this to a stem plot. You can see that these dots are what GNU radio is joining with lines. Another block that is very useful is the QT-GUI frequency sink.

Ctrl F or command F and type freq, you will see instrumentation the QT-GUI frequency sink. Drag and drop it into your flow graph. Change the type from complex to float by double clicking the block and altering the type. One important aspect of GNU radio blocks is that one output can be connected to multiple inputs. Let us take this throttle and connect it to the output of the throttle to the QT-GUI frequency sink.

Let us now execute this flow graph. As you can see, GNU radio neatly lays out the range slider, the amplitude versus time, this is the time sink and the QT-GUI frequency sink as well. Let us set this to a 1000 Hz and let us zoom in over here. You can see that there is a peak exactly at 1 kHz and a peak exactly at minus 1 kHz. There are two peaks for this real signal because cos omega naught t has a Fourier transform that is two sided.

We will see this more closely in the next lecture. Now these peaks are not sharp, they are fat and they have something like a lobe characteristic. Why is that the case? The reason is because by default, GNU radio performs what is called windowing before it

displays to you the frequency characteristics. Let us change this to rectangular. A rectangular window prevents any pre multiplication when the frequency characteristics are shown.

Let us now check what a 1000 Hz looks like. Now you see that a 1000 Hz looks like two really sharp edges. The sharp edges, sharp sticks correspond to exactly plus 1 kHz and minus 1 kHz. However, the peaks are exactly at around minus 6 dB and there are two peaks, one at plus 1 kHz and minus 1 kHz. To understand the properties of the GNU radio frequency sink better, we must first understand that it plots the magnitude response that is it plots mod H of f square.

More importantly, it gives a log plot that is it plots 10 log H of f square that is the same as 20 log H of f in dB. In addition, you got two sticks for 1000 Hz, but you will not get sticks for all frequencies. The reason is because much like any other computational way of finding the frequency, the GNU radio frequency sink uses the discrete Fourier transform for finding the DFT. Therefore, you will get distinct spectral lines only when f upon f s times NFFT is an integer. For example, if you choose 1000 Hz and the sampling rate is 32000, 1000 upon 32000 multiplied by 1024 is an integer.

Now the reason this is minus 6 dB on both of these is because a cosine is half e power j omega naught t plus half e power j minus omega naught t and as you are aware a half corresponds to minus 6 dB. If we change this to any other frequency that has the characteristic that that frequency upon sampling rate multiplied by NFFT is an integer, you will get sharp lines. For example, if we choose 4000 Hz, you will get sharp lines. However, 5000 Hz sorry 5200 Hz, 5300 Hz for example, will not yield sharp lines. The reason is because 5300 upon 32000 multiplied by NFFT is not an integer.

Therefore, you must be careful when interpreting the frequency characteristics. Always remember that GNU radio uses the DFT in the background to interpret the results of the frequency sink correctly. Coming back to our example flow graph, we will now add our first hardware sink. Before we go into this, remember that throttle is required only for those flow graphs that do not have a hardware source or a sink. So the first thing that we will do is to remove the throttle.

You can do so by clicking on the throttle and hitting the delete key or by right clicking on the throttle and saying delete. You can now connect the signal source by clicking on the out to the QT-GUI time sink by clicking on in. Once again, we will connect it to the frequency sink as well. But before we run it, we will add our hardware sink, which is the audio sink. To do so, press Ctrl F or Command F and type audio so that you get a list.

We will choose the audio sink and place it here. We will connect the signal source to the audio sink. In order to hear the audio more clearly, let us also change the range to start at 1000 hertz and make the default value 1000 hertz. This can be done by double clicking on the range and changing the default and start values to 1000 each.

Let us now run this. If you run this, you will be able to hear a sine wave, which is a tone. By changing the frequency, you will hear that the tone also changes frequency and you can clearly hear this. For the remaining part of this lecture, I am going to mute the audio, but you can hear it on your computer when you perform the experiment. So as you can see, by changing the frequency, you will be able to hear different tones and depending on the volume and your speaker characteristics, you can possibly go up to higher frequencies as well. We will now remove the audio sink to perform another simulation, but to do so, we must add back the throttle.

So I am going to click on the audio sink, delete it, click on this arrow, delete it, click on this arrow, delete it, all with the delete key on my keyboard. You could also right click on the arrow and click delete. We will now add back the throttle by typing Ctrl F, Command F and throttle and moving the throttle over here, double clicking on the throttle, changing the time to float, connecting it back to our time sink and frequency sink. The next thing that we are going to do is to filter a signal. We are going to now add a low pass filter, which will allow us to filter the signal that is being output by the signal source.

To add a low pass filter, we hit Ctrl F or Command F and type low pass and we take the low pass filter and bring it over here. As you can see, there are several fields in the low pass filter that are red because you need to specify them. Let us double click the low pass filter. We are now going to say float to float. Decimation or interpolation is when you need to change the rate at which the signals come in and go out.

In this case, we are not performing any decimation or interpolation. So we can choose either of these options interpolating or decimating and keep the interpolation or decimation respectively at 1. The gain is the pass band gain characteristic. The cutoff frequency is where the filter starts cutting the signal out. Let us say that the cutoff frequency is 4500 hertz and the transition width to be a 1000 hertz, which means that signals that are above 4500 hertz will start seeing reduced amplitude at the output of the filter and at around 5.

5 kilohertz onwards, those signals will be very very low in power. We then click OK to return to our flow graph. Next we connect our signal source to the low pass filter. Now to visualize the output of the low pass filter, we can now choose two approaches. One

approach would be to add another QT-GUI time sink and QT-GUI frequency sink.

This is disadvantageous for two reasons. The first is that adding these would complicate the flow graph and more importantly, it would not provide us a direct way to compare the signal source output with the filtered output. Therefore it is better for us to visualize the output of the low pass filter directly within the QT-GUI time sink and frequency sink. To do this, you can double click the QT-GUI time sink. Hold down and you will find the number of inputs as an entry.

Change this to 2 and press OK. Similarly, double click on the QT-GUI frequency sink and change the number of inputs to 2 and say OK. You can now see that both the sinks have two inputs, IN0 and IN1. We can connect the output of the low pass filter to IN1 on both of these and by executing the flow graph, you will now see a frequency plot and a time plot along with our range. Clearly in the time plot, you can see two sinusoids, this is 67 milliseconds, 68 milliseconds, 1 millisecond gap, this is 67.1, this is 68.1. Both of these are 1 kHz sinusoids. The reason for the offset is because the filter adds some amount of delay. In the frequency domain, you see a red peak. However if you click on the data 1 to make the red peak disappear, you will see the blue peak right behind it. This indicates that both of them have roughly the same amount of amplitude.

By increasing the frequency, you will see that till about 2 kHz or so, there is not much of a difference other than the delay the filter introduces. However, as we get close to around 4 kHz, you will see that the red curve starts seeing some diminishing amplitude. This is because the effect of the filter is starting to show up. By taking it to 4.5 kHz, you can clearly see that the amplitude of the red curve is much lower.

Over here in the frequency domain, you can clearly see that the blue peak at about minus 8 and while the red peak is about minus 16, this is about an 8 dB suppression that corresponds to about a one-third in amplitude or so. If you now take it further and take it to around 6.5 kHz, in the time domain, the signal is very diminished. If you right click on this, you will see that in the frequency domain, this is at minus 6 dB, this is around minus 65 dB, there is a 60 dB suppression indicating that the filter performs very well. If you now go back to a low frequency, you get back the old behavior wherein because the filter allows signals in the pass band through without much amplitude change, the two signals look very close to the same other than the delay.

We can repeat this for the case where you have hardware as well. Let us once again remove the throttle and add back the audio sink. We click on the throttle and click delete, control F, audio. We drag the audio sink over here. Connect the output of the signal source, sorry, the low pass filter to the audio sink.

You connect the signal source inputs back because there is no throttle. If we now execute the flow graph, we can hear the 1 kHz tone. We can now hear roughly a 2 kHz tone.

Around 3 kHz also it is audible, even at 3.8 kHz. But the moment you hear to about 4.6 kHz, you will see that the amplitude starts diminishing. And after 5 kHz, it is hardly audible indicating that GNU radio is actually performing real time filtering and cutting off the frequencies above around 4.5 kHz. If you set it to something close to 4.5 kHz, you will be able to hear the signal in a feeble way and the amplitude gets back for lower frequencies. Let us now close this and return to the original configuration with the throttle. So I am going to click on this audio sink and hit delete, control F or command F, throttle. Click back the throttle, double click on it, choose float, remove this connection, connect the signal source to the throttle, throttle to the time sink. Remember that one throttle is needed throughout the flow graph, it does not matter where it is and with this we have got back the old behavior.

You can clearly see that as you increase the frequency, the red signal gets diminished, if you take it back to less than 4 kHz, it is roughly at the same amplitude at which you start. There are several other possibilities that you can explore in GNU radio. These are part of the assignments that are shared. Please go through them and learn GNU radio effectively. In this lecture, we have seen how you can put together the several blocks that GNU radio offers you in order to obtain many practical simulations as well as hardware based experiments.

In the remaining part of the course, throughout the course, we will go through many concepts and implement them on GNU radio to easily visualize as well as see the impact of change of parameters on these. So please try to follow along and use GNU radio with me throughout this lecture series. Thank you. Thank you.