# Controlling PlutoSDR

The necessary prerequisites have been installed on the local lab machine. However, if you are on your own machine the same tools necessary for this lab is available here.

To make sure we can see and connect to PlutoSDR open a terminal or command prompt. Type "iio_info -n 192.168.2.1" to scan for devices. You should see an output like Figure 1.
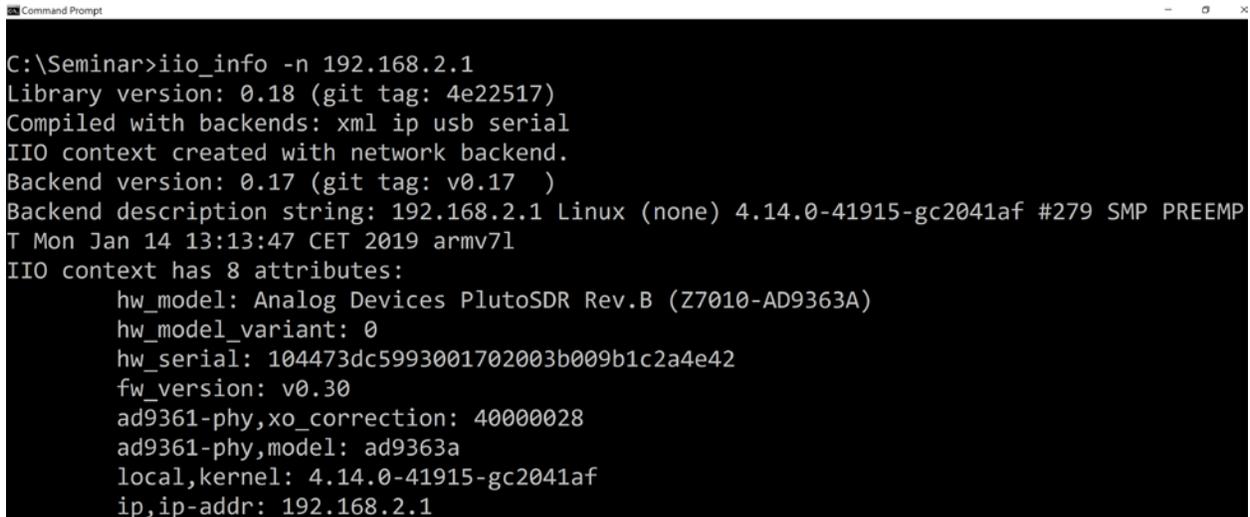

Figure 1

Your device should have a unique serial number and URI. The URI is specific to the interface, here it is displayed as [ip,ip-addr: 192.168.2.1], and can be used to interface with a specific device when required. When you have serial access to a device, the URI will be associated with an USB address such as: [usb:X.X.X]. If you do not observe a device, make sure PlutoSDR is connected to the correct USB port and has a one solid LED and one flashing LED. "iio_info" is a useful tool for identifying devices locally connected and on the network as well.

Another useful libIIO command line tool is "iio_attr", which can be used to inspect, as well as read and write attributes. The help for this tool is shown in Figure 2, which outlines that "iio_attr" can be used to access different types of attributes including: device, channel, debug, buffer, and context.

Figure 2

The first step when using "iio_attr" is to select a device to inspect. We will next to PlutoSDR again using the network context feature and list the devices using the command "iio_attr -u ip:192.168.2.1 -d". Use this command to list your devices as shown in Figure 3.

Figure 3

Next, we will examine some of the channel attributes for a given device. Run the command "iio_attr -u ip:192.168.2.1 -c ad9361-phy", as done in Figure 4, which lists the channels associated with the PHY driver. These channels are related to certain aspects of the transceiver like TX, RX, temp sensors, and even some custom filters in the FPGA data path.

Figure 4

By selecting one of these channels, in this case voltage0 which is the real channel names for both TX and RX, we can view some of the attributes related to those channels. For reference, input channels are with respect to RX and output channels are with respect to TX. To list these attributes run the command "iio_attr -u ip:192.168.2.1 -c ad9361-phy voltage0".

Figure 5

To filter out one of these attributes simply add it to the end of the last used command. For example, inspecting the hardwaregain would be called as "iio_attr -u ip:192.168.2.1 -c ad9361-phy voltage0 hardwaregain", as shown in Figure 6.
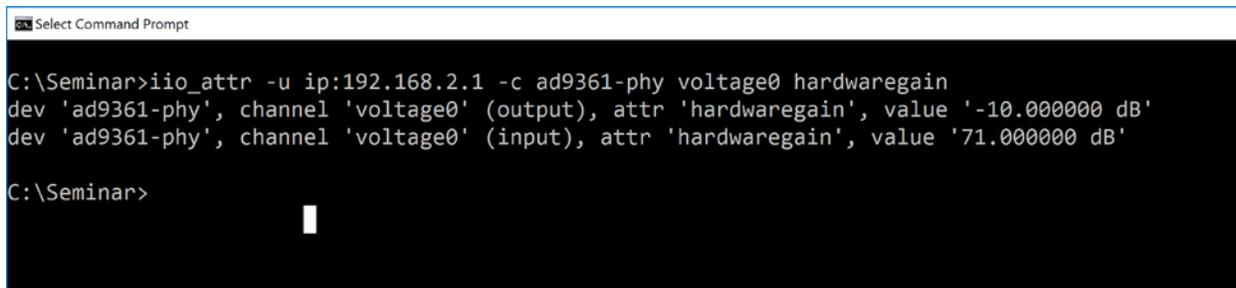
Figure 6

Next, we want to only view output signals (those related to the transmitter), which we can do by adding the output filter tag "iio_attr -u ip:192.168.2.1 -o -c ad9361-phy voltage0 hardwaregain".
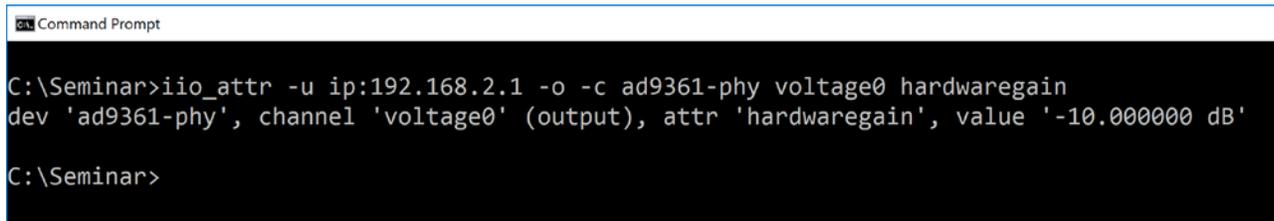
Figure 7

Finally, lets write to the attribute by adding a value to the end of the last command "iio_attr -u ip:192.168.2.1 -o -c ad9361-phy voltage0 hardwaregain -30". When values are written, they are first always read from beforehand, and then read from after the update is applied. If the value supplied is invalid, you will get an error.

Figure 8

Going through the context, to the devices, to their channels, and finally to channel attributes, you can see how the driver controlling PlutoSDR is structed. We will build off these ideas in following labs, but first we will do something more interesting.

Using the command line alone is slow, non-intuitive, and repetitive. You can write your own applications, like "iio_attr" in C, or in other languages like Python or MATLAB. However, there exists a

specific GUI application for interacting with IIO devices called IIO-Oscilloscope (OSC), which is may more civilized than poking around at attributes through "iio_attr" all day.

IIO-Oscilloscope (OSC)

As a prerequisite for these labs you should have already installed OSC, if not please install it from the [Release Page here](#).

Launch OSC, and you should be greeted by a connection window like Figure 9.
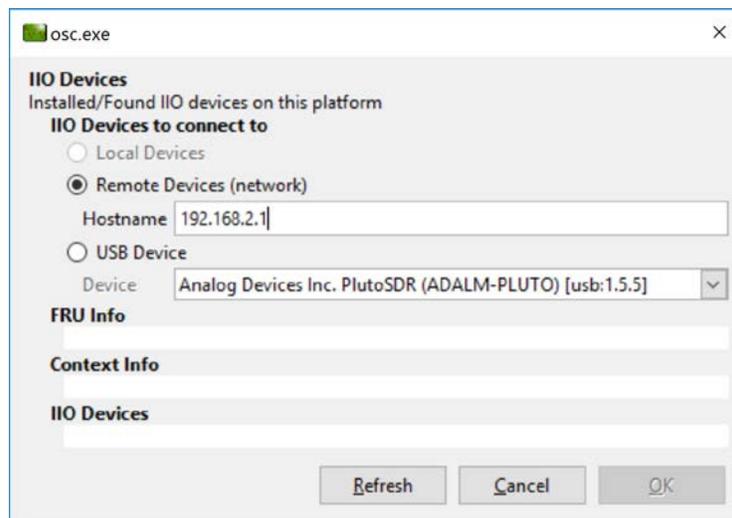
Figure 9

Next, select the "Remote Devices (network)", and type in the IP address of PlutoSDR. Click the refresh button, and then OK. Next the main window should appear with the necessary control tabs.

Explore IIO oscilloscope, and the control tabs. These tabs let you control the high to the low levels of the device and visualize the signals that are going in/out of the device. Some specific parts to look at are shown in Figure 10.
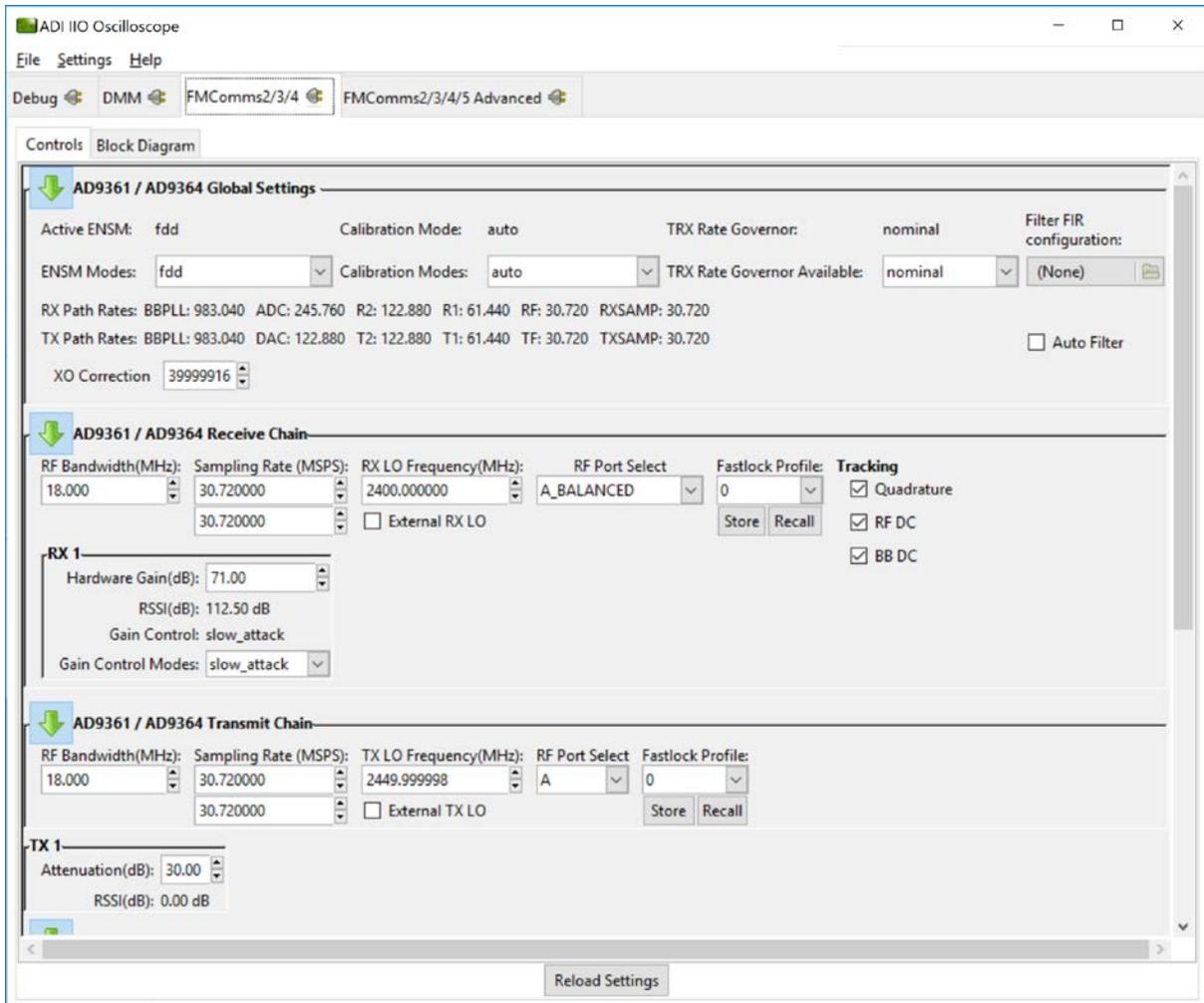
Figure 10

From the "FMComms2/3/4" tab shown in Figure 10, you can examine and change the RF bandwidth, the DAC sample rate, the Tx LO Frequency, and the signals that are being sent from the FPGA to the AD9363 inside the ADALM-PLUTO SDR. This tab, as the name define is shared among a few SDR that proceeded PlutoSDR, specifically FMComms2/3/4.

Using the controls in the "FMComms2/3/4" tab, set the DDS in the "FPGA Settings" to a single tone at -3dB scale.
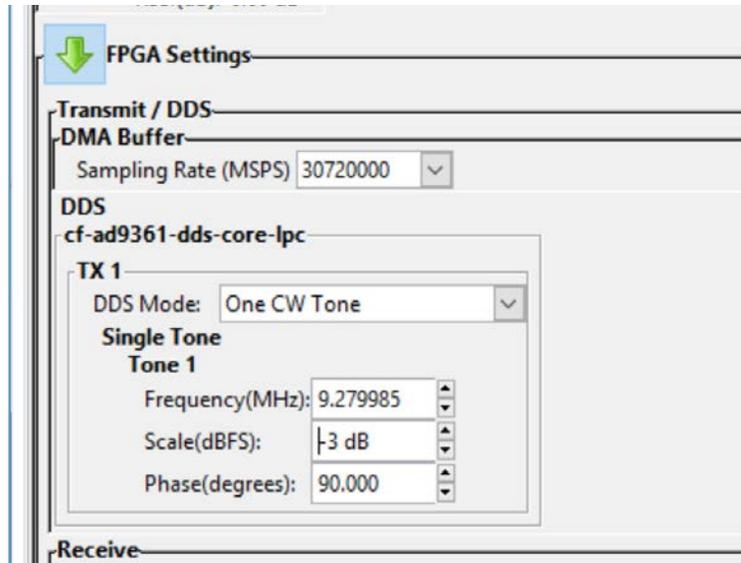
In the Receive and Transmit Chain panels, set the Sample Rate to 30.72 MSPS, with a 18 MHz RF Bandwidth, at a TX and RX LO of 2450 MHz.
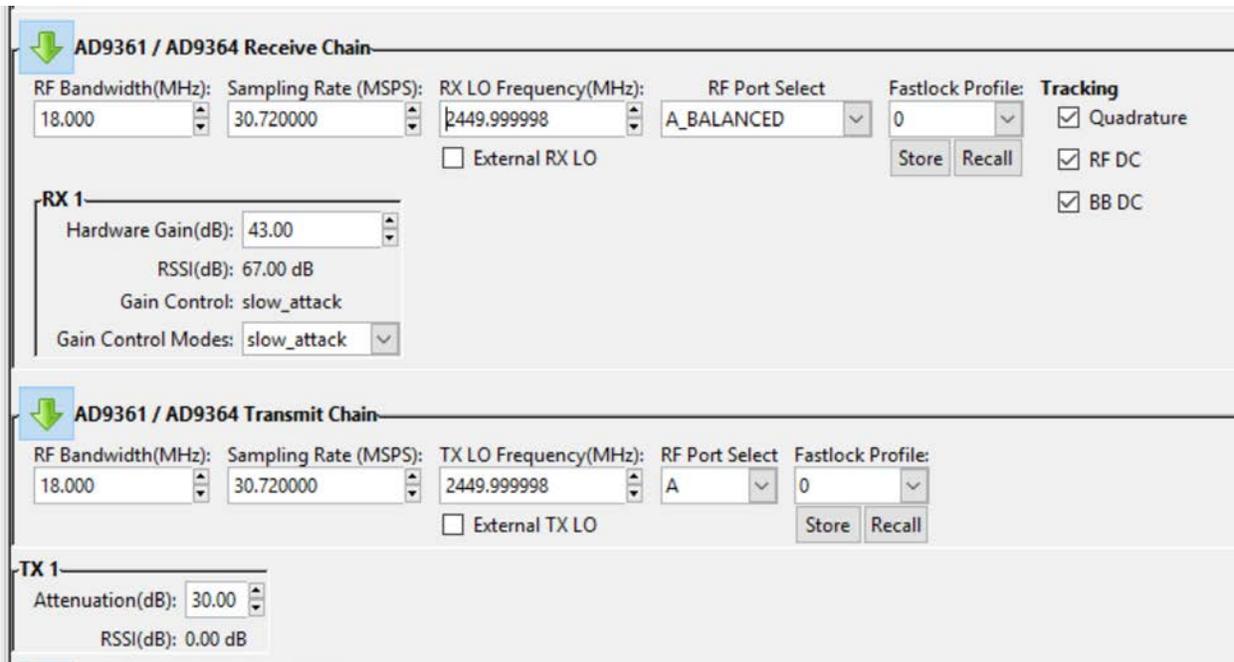
OSC also visualization capabilities, which we will explore next.

Visualizations

Create a new plot window from the file menu.
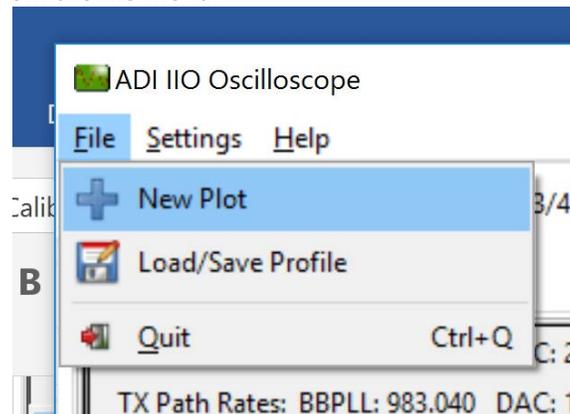
These plot, or capture, windows provide real-time visualization with time domain plots, frequency domain plots, cross-correlation, and constellation diagrams. Enable some of the voltage channels and play with the scopes a bit to get a feeling for them. Once you are done, move on to the next section.
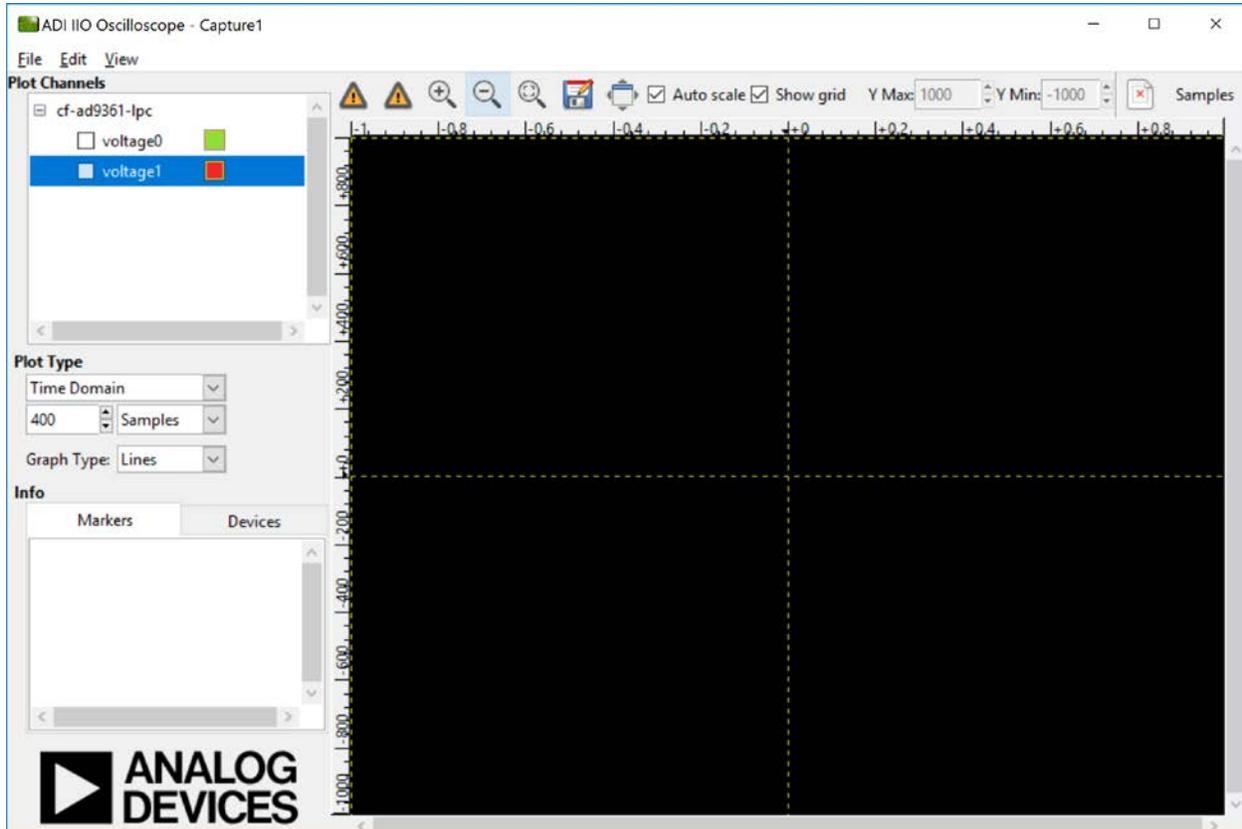
For now, enable both voltage channels, "voltage0" and voltage1, and select the constellation diagram plot. Then press the play button to start capturing data.

Next, we will setup the transmitter to send some useful data. Back in the "FMComms2/3/4" tab, in the FPGA Settings panel. As show in Figure 11, select "DAC Buffer Output" as the DDS Mode and load the file "qpsknofilt_30M.txt" for voltage0 and voltage1. Make sure to hit load. This will set the transmitter to continuously repeat a buffer of data, which are just some QPSK symbols.
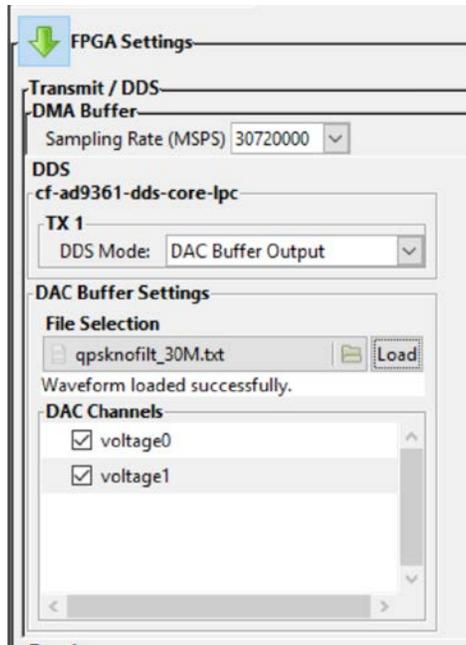

Figure 11

Back on the constellation diagram, you should observe something like Figure 12. Since there is a fractional delay between the antennas we are observing some inter-symbol interference. This will be random for each user and each radio initialization.
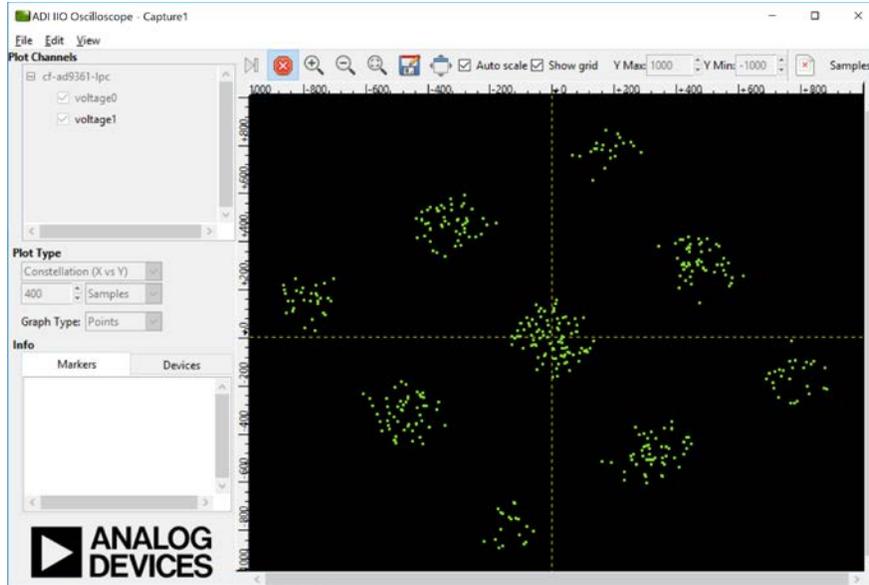
Figure 12

Now for debugging it can be useful to view digitally looped back data, which has not gone out to the RF. To enable this loopback go to the "FMComms2/3/4/5 Advanced" tab, the BIST subtab, and select "Digital TX -> Digital RX" as the Loopback mode as in Figure 13.
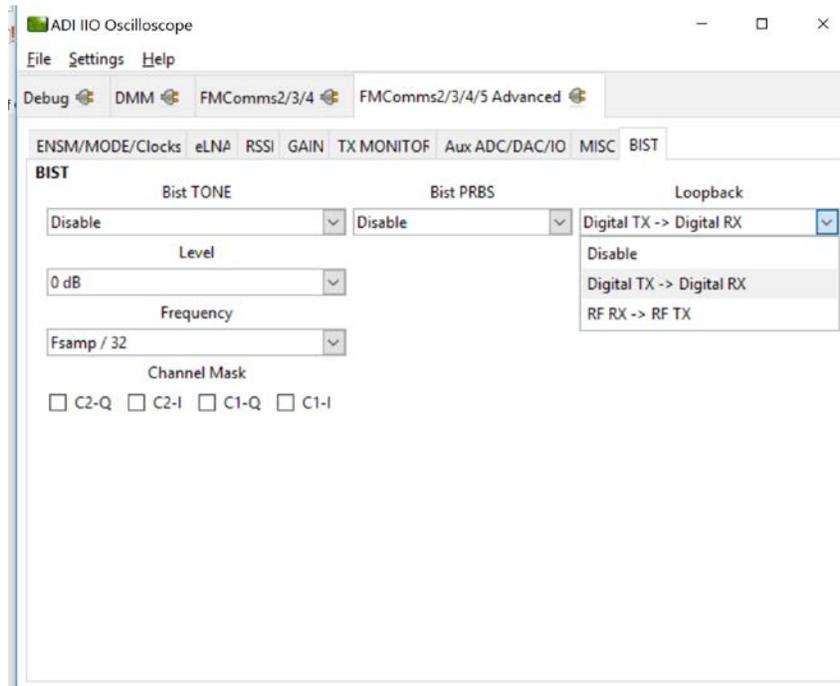

Figure 13

Going back to the constellation diagram, enabling lines instead of points, we should observe a perfect constellation with perfect crossing in the middle as in Figure 14.
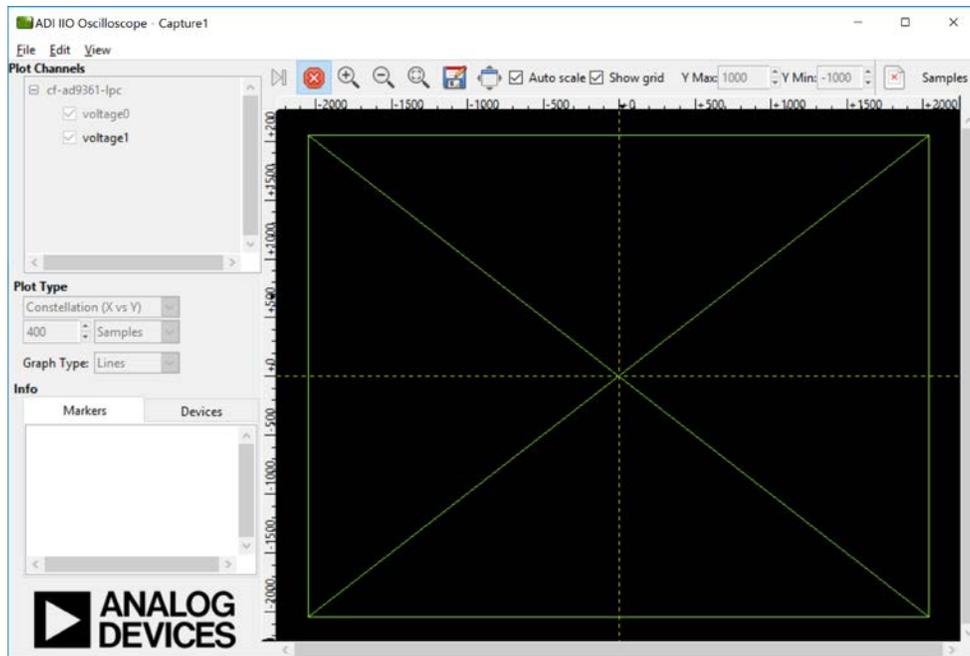
Figure 14

With free time remaining before the next lecture section explore OSC on your own and play with all the knobs if you wish.