Capturing Signals in GNU Radio

Storing Data into a File

The output of any block (including the USRP) can be stored into a data file. Insert the following commands into the Python script:

```
audiodata = gr.file_sink(gr.sizeof_float, "audio.dat")
self.connect(src0, audiodata)
```

This creates a sink named *audiodata* which will store its data (binary) into a file called *audio.dat*. The sink is then connected to the source or block that will be outputting the data to be collected. In this example we have used *src0* to represent a source. In the event that the source/block outputs complex data, use

```
audiodata = gr.file_sink(gr.sizeof_complex, "audio.dat")
```

Converting the Binary File to a MATLAB Readable Form

The file created above contains binary data that MATLAB (or Octave) does not know how to read. There is a set of .m files in gnuradio-core/src/utils that perform the needed conversions. In the case of a binary data file that contains floating-point data use:

```
read_float_binary.m
```

In the case of a binary file that contains complex data use

```
read_complex_binary.m
```

From Octave you can run these with:

```
audio1 = read_float_binary('audio.dat');
```

and the resulting data will appear in a matrix called *audio1*.

Plotting the Data with Octave

You can create a plot of this data in Octave (assuming Gnuplot is also installed) with:

```
plot(audio1)
```

Plotting the Data with MATLAB

If you would like to create a data file that is readable by MATLAB you need to save the *audio1* data in a MATLAB readable file. This can also be done from within Octave using:

```
save ("-v7", "audio2.mat", "audio1")
```

The file *audio2.mat* can now be imported into MATLAB (Mac or PC).