

Application Note  
Examples Provided with the USRP Hardware Driver  
Ettus Research

## **Introduction**

This document will provide information on code examples provided with the USRP Hardware Driver (UHD™). UHD is a driver developed by Ettus Research and is compatible with all USRP™ (Universal Software Radio Peripheral) software defined radios. UHD allows development on multiple operating systems including Linux, Windows and MacOS. The abstraction provided by the driver allows substantial portability between USRP products. UHD provides an application programming interface (API), which provides access to various functions of the USRP including synchronization, sample streaming and configuration. UHD is currently used in several development environments including GNU Radio, LabVIEW, Matlab, Simulink and other custom frameworks.

Example programs are provided with UHD to allow basic functional testing of one or more USRP devices. The source code for the example programs is also provided for developers that want to gain a better understanding of various UHD API features.

## **Locating the UHD Examples**

In typical Linux installations, the UHD path is /usr/local/share/uhd/. Windows installations typically place UHD in C:/Program Files(x86)/UHD, and the examples are found in:

(base UHD directory)/share/uhd/examples

## **Modifying a Sample Program in Linux**

Any sample program can be modified and recompiled. This provides an easy way to learn how to interact with the UHD API. After changing the source code of an example UHD application, recompile with:

```
make
```

In order to copy the modified application to the UHD install location, type:

```
sudo make install
```

If you are in Windows additional build instructions can be found:

[http://files.ettus.com/uhd\\_docs/manual/html/build.html#build-instructions-windows](http://files.ettus.com/uhd_docs/manual/html/build.html#build-instructions-windows)

## **Command Line Arguments**

All of the example applications provided with UHD use one or more command line arguments to configure the application for a particular test. One can determine what arguments can be used with an application by typing “--help” as an argument. For example, typing:

```
./tx_waveforms --help
```

will produce the following output:

UHD TX Waveforms Allowed options:

--help	help message
--args arg	single uhd device address args
--spb arg (=0)	samples per buffer, 0 for default
--rate arg	rate of outgoing samples
--freq arg	RF center frequency in Hz
--ampl arg (=0.300000012)	amplitude of the waveform [0 to 0.7]
--gain arg	gain for the RF chain
--ant arg	daughterboard antenna selection
--subdev arg	daughterboard subdevice specification
--bw arg	daughterboard IF filter bandwidth in Hz
--wave-type arg (=CONST)	waveform type (CONST, SQUARE, RAMP, SINE)
--wave-freq arg (=0)	waveform frequency in Hz
--ref arg (=internal)	waveform type (internal, external, mimo)

Typically, a large portion of the arguments provided are optional, and have default values assigned. In some cases, values must be provided. In these cases, the program will perform error checking to ensure all required fields have an associated command line argument.

In one example, a user may wish to transmit a 100 kHz Sine wave tone, with an RF center frequency of 400 MHz, baseband amplitude of 0.5, out of antenna TX/RX. The command line would look like:

```
./tx_waveforms --wave-freq 100e3 --wave-type SINE --freq 400e6 --ampl 0.5 --ant  
TX/RX
```

	Common Description	Type	Example	Interpretation
rate	Used to set the host sample rate for a transmit or receiver streamer.	double	4e6	Stream at 4 MSPS(complex)
nsamps	Numbers of samples to process/transmit/receive	double/int	100e3	Stream 100,000 samples
secs	Duration to run the test, or delay start of streaming.	double	5	Stream for 5 seconds
args	Address arguments(esp. "-args addr=192.168.10.2")	string	addr=192.168.10.2	
freq	Center frequency	double	400e6	Set center freq to 400 MHz
ant	Antenna selection(i.e. "TX/RX", "RX2")	string	TX/RX	Use TX/RX port
ampl	Baseband amplitude(min = 0.0, max 1.0)	double	0.5	Baseband amplitude is 0.5
bw	Bandwidth setting of daughterboard in Hz	double	60e6	Set daughterboard BW to 60 MHz
ref	Reference clock selection	string	external	Use external clock
gain	Daughterboard rx or tx gain - range depends on dB	int	25	Set gain to 25dB
subdev	Subdevice specification	string	A:AB	Daughterboard slot A, channel AB I/q pair

**Table 1 - Common Command Line Arguments**

## **Listing of Example Applications Supplied with UHD**

To help new users navigate through the example applications provided with UHD, a table describing the function of each application, its transmit/receive direction, and whether or not it can be used with multiple USRP devices is shown below.

Example Filename	Multiple USRP Support	Tx/Rx	Description
benchmark_rate	No	Tx/Rx	Useful for testing throughput capability of PC + USRP. Will report number of sent frames, overflows, dropped frames, etc.
rx_timed_samples	Yes	Rx	Illustration of how to use UHD time_spec to begin sampling at a specific point in time.
tx_waveforms	Yes	Tx	Transmit various signal types( Sine wave, square, ramp, constant).
latency_test	No	Tx/Rx	Attempts to receive a sample, then transmit same sample rtt later. Reports number of underflows, errors, and acknowledgements. Rtt argument is a good indicator of interface latency when majority of packets are ack'd.
test_pps_input	No	n/a	Tests USRP(s) for 1 PPS signal. Reports whether test passed or failed.
test_dboard_coercion	No	n/a	Sweeps frequency and gain ranges of daughterboard to ensure it tunes to all of the advertised frequencies and gains
test_timed_commands	Yes	n/a	Tests timed commands feature for PLL alignment between multiple USRPs
tx_bursts	Yes	Tx	Illustration of how to use metadata to specify the time to transmit samples.
transport_hammer	No	n/a	Calls for random numbers of rx/tx samples to test transport layer for various errors.
rx_multi_samples	Yes	Rx	This is a demonstration of how to receive aligned data from multiple channels.
rx_samples_to_file	Yes	Rx	Creates receive streamer for specified USRP(s) and writes all samples to file at the end of data collection.
tx_samples_from_file	Yes	Tx	Create transmit streamer(s) to USRP(s), and transmit all samples from specified file in continuous mode.
tx_timed_samples	No	Tx	Creates a transmit streaper to a USRP, sends specified numbers of samples, than transmit and End-of-Stream command w/ ack verification.
rx_samples_to_udp	Yes	Rx	Receives samples from a USRP streamer, forwards to UDP socket.

**Table 2 - UHD Example Application Summary**

### **Other Helpful Examples - GNU Radio**

GNU Radio, a common framework used with UHD and the USRP family of products, also provides two useful example programs for functional testing. When GNU Radio is installed, `uhd_fft.py` and `uhd_siggen_gui.py` are installed as executable commands. `uhd_fft.py` provides an FFT, Waterfall, or Scope display of a signal received with a USRP device and daughterboard. There are many parameters that can be used to configure the receiver sample rate, frequency, etc. `uhd_siggen_gui.py` is an application that allows the user to generate various output types at specific frequencies, sample rates, and gain settings. The programs can be executed simply by typing:

```
uhd_fft.py  
-OR-  
uhd_siggen_gui.py
```

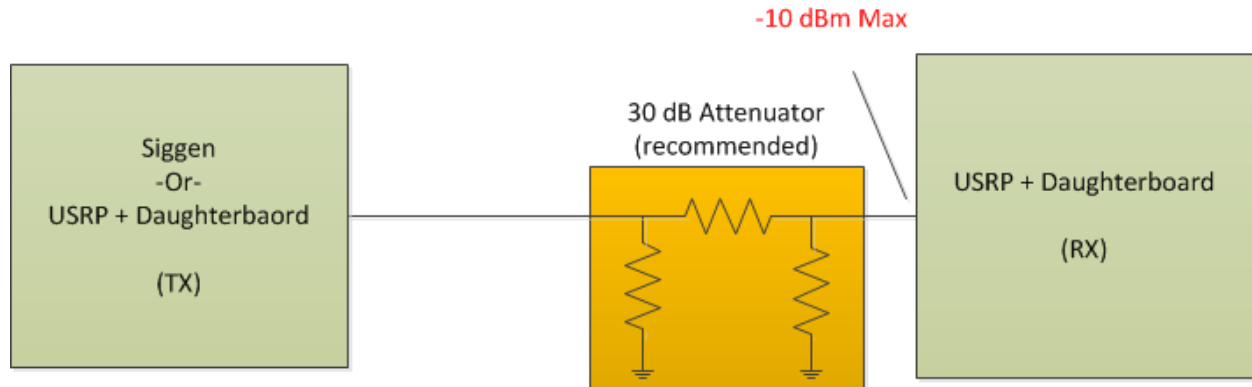
into a command line interface. More information on how this can be used to perform a functional test of the USRP device and daughterboard will be provided later in this document.

### **Functional Testing of the USRP**

The UHD and GNU Radio sample applications can be used to perform functional testing of one or more USRP radios. This is useful for new USRP users wanting a better understanding of how the USRP functions.

#### **Receive Testing - uhd\_fft.py**

`uhd_fft.py`, which was mentioned earlier in this document, is a helpful tool for testing the receive functionality of a USRP radio. A block diagram showing a possible test method is shown in Figure 1. In this test, a signal source operating within the frequency range of the daughterboard and USRP device being tested is used to generate a test signal. This signal source can be an RF signal generator or another USRP radio. Care must be exercised to guarantee that the maximum input specification of the particular daughterboard is not exceeded. This level is -10 dBm for all daughterboards other than BasicRX and LFRX. The input for the BasicRX and LFRX must be less than 2 Vpk-pk (10dBm). LFRX provides DC coupling with unity gain, so the absolute voltage level present at the LFRX input(s) should not exceed 3.3V.



**Figure 1 - Configuration for USRP Receive Testing**

After connecting the signal source with an appropriate level, `uhd_fft.py` can be started via command line. The real-time controls can be used to set the frequency, gain, and sample rate of the application.

When the FFT (default) view is used, the x-scale is the frequency, and the y-scale is amplitude. The y-scale shows the amplitude with “counts,” and the values do not typically correlate to a specific, absolute power input. The amplitude read on the display is useful for approximate comparisons. The level for a given input amplitude will vary a few dB across frequency and from unit to unit. Also, receiver daughterboards provide various levels of amplification in their analog chains, which will affect the amplitude result in the FFT.

### **Receive Testing - `uhd_fft.py` - Common Errors**

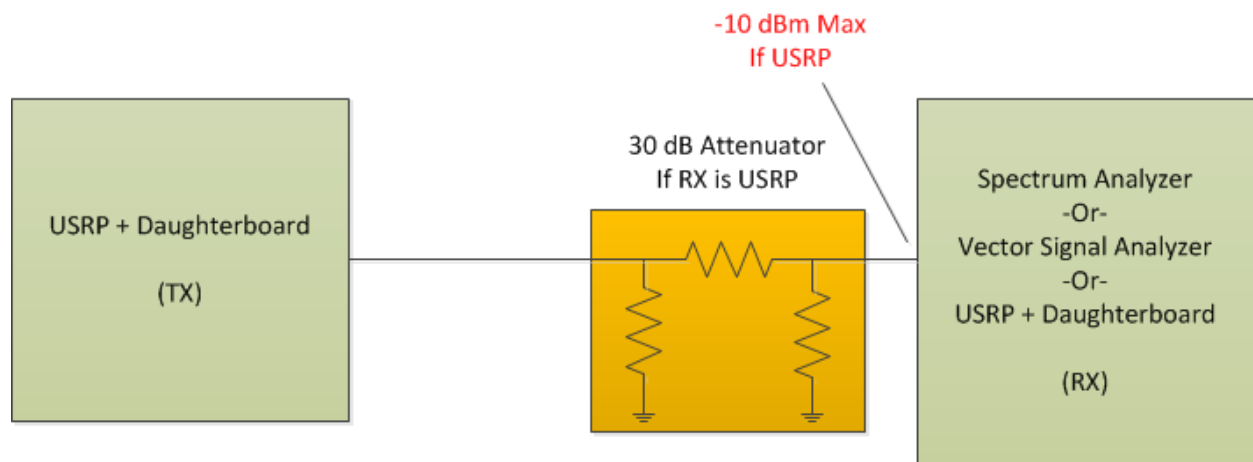
There are several errors new users often make when using this program. A few points to verify:

- a. Ensure the input level has not exceeded the maximum input specifications for the daughterboard.
- b. Verify the proper subdev specification and antenna selection are provided at the command line.
- c. Ensure the frequency selected is within the range specified for the daughterboard and the signal generator is properly set.

### **Transmit Testing - `tx_waveforms`**

The example application “`tx_waveforms`” provides an easy way to test the operation of the transmit function of the USRP device and daughterboard. A block diagram of a test apparatus is shown in Figure 2. Simple testing can be performed with a spectrum analyzer. It is also possible to use another USRP device that is accessed with `uhd_fft.py`. Once again, if another USRP is used, it is important to emphasize the maximum input specification of the receiving USRP device. An attenuated connection, such as that provided by the Ettus Research LPBK-

KIT, should be used in this setup. If the signal is not attenuated, it is possible to damage the receiving USRP hardware.



**Figure 2 - Configuration for Testing USRP Transmit Capability**

Once the signal receiver/spectrum analyzer is connected, tx\_waveforms can be executed. An example of tx\_waveforms is:

```
./tx_waveforms -freq 1200e6 --ampl 0.5 --wave-type SINE --wave-freq 250e3 --rate 5e6 --ant TX/RX
```

This command will start tx\_waveforms, with an RF center frequency of 1.2 GHz, and a baseband signal of 250 kHz and amplitude of 0.5. The baseband amplitude is normalized to the peak swing of the DAC. 0.5 is used to ensure the quadrature modulator is not driven into compression.

This allows several useful functions to be tested. Parameters such as single sideband suppression and image rejection and output power can be measured. This also provides a qualitative method to determine that the USRP is functioning.

GNU Radio also provides uhd\_siggen\_gui.py, which provides similar functionality with a graphical interface.

### **tx\_waveforms - Common Errors**

Some common errors, listed below, often come up when new users experiment with tx\_waveforms.

1. Proper care is not exercised when receiving the waveform with another USRP device, and the daughterboard receiver is damaged.
2. Amplitude and or gain are set too high, resulting in high intermodulation distortion, harmonic distortion, and other artifacts

3. The “ampI” argument is set above 1.0, which is invalid and will cause clipping/rounding in the digital domain.

## **Conclusion**

This document provides a basic introduction to the example applications provided with GNU Radio and UHD. These applications can be used to experiment and learn the basic signal processing elements at work within the host PC, USRP device, and daughterboard. They can also provide a mechanism for functional testing to assure the USRP device is operating correctly. Since the error is so common, it is worth taking the time for one last warning: DO NOT EXCEED THE MAXIMUM INPUT ALLOWED FOR THE DAUGHTERBOARD.

If you have any questions or would like to provide feedback for this application note, please send an e-mail to [support@ettus.com](mailto:support@ettus.com).