

# GNU Radio on the ThinkRF WSA1000U

## Combining the awesome versatility of GNU Radio with the high-performance cost-effective ThinkRF WSA1000U RF Digitizer.

GNU Radio is a software development toolkit that provides the signal processing capability to implement software radios and is distributed under the terms of the GNU General Public License.

### WSA1000U Key Features

- Integrated RF/digital acquisition system
- Up to 85 MHz instantaneous RF bandwidth
- Scan rate greater than 10 GHz/sec
- 400 MHz to 4 GHz frequency coverage
- Large dynamic range
- High speed connectivity via USB 2.0
- Streaming modes supporting 32 MB/s
- Signal recording
- Low power consumption
- Very cost-effective

The WSA1000U RF Digitizer provides the ability to run GNU Radio RF receive applications. The WSA1000U combines high-performance RF with leading edge ADC technology to provide an efficient combination of cost, size and performance that is commercially deployable.

This application note overviews the requirements and capability of the combined system.

### Configuring the WSA1000U for GNU Radio

The WSA1000U hardware can support different personalities for different applications. The user can change the hardware's personality and application at any time using ThinkRF's hardware configuration tool. Whereas one personality supports GNU Radio, ThinkRF also provides a hardware personality that supports sophisticated triggering, spectrum analysis and standard APIs.

The WSA1000's GNU Radio hardware personality contains the digital down converters (DDC) and cascaded integrator-comb (CIC) filters so as to be functionally compatible with the USRP receive path. The USRP is a radio peripheral well-known in the GNU Radio community. High-level GNU Radio documentation regarding the use of the USRP receive path also applies to the WSA1000U.

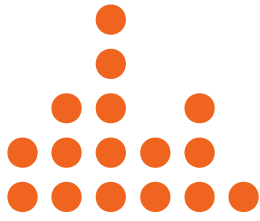
### Installing GNU Radio on the WSA1000U

ThinkRF supports GNU Radio on CentOS Linux 5 and Red Hat Enterprise Linux (RHEL) 5. GNU Radio has dependences on software packages not included in the default CentOS/RHEL repositories. Some of these packages can be found in a third-party repository (EPEL). The remainder of the packages, including the GNU Radio package, are provided in a repository maintained by ThinkRF.

ThinkRF's GNU Radio package is a special ThinkRF build where USRP signal source components refer to the WSA1000U device.

For those wishing to use WSA1000U with GNU Radio on other Linux distributions, or to use a newer version of GNU Radio, ThinkRF maintains and makes available the source-level changes to GNU Radio in the form of a Git repository.





# GNU Radio on the ThinkRF WSA1000U

## Application Example #1 - Displaying a Spectrum Graph using 'usrp\_fft.py'

GNU Radio applications are primarily written using the Python programming language and the performance-critical signal processing path is implemented in C++ using processor floating point extensions where available. The WSA1000U will run standard GNU Radio applications although some enhanced features of the WSA1000U are not available via the controls provided by the standard.

The GNU Radio software application 'usrp\_fft.py' provides an easy but visual example using real-time FFT to display a spectrum graph. The illustration shows a modified version of the application that provides additional controls. Its usage is as follows:

### WSA1000U Specifications

#### Instantaneous Bandwidth

- Up to 85 MHz (limited to ~8 MHz streaming)

#### Frequency Range

- 400-4000 MHz

#### Spur-free Dynamic Range

- > 63 dB

#### Total Operating Amplitude Range

- 120 dB

#### Max. Input Power

- +20 dBm

#### Input damage level

- +25 dBm

#### Phase Noise @ 2.45 GHz with internal VCO

- -89 dBc/Hz typ. @ 10 kHz offset
- -114 dBc/Hz typ. @ 100 kHz offset
- -134 dBc/Hz typ. @ 1 MHz offset
- -148 dBc/Hz typ. @ 5 MHz offset

#### Displayed Average Noise Level

- < -100 dBm

#### Input Supply Voltage

- 6V +/- 5%

#### Input Current

- 1.5A max.

#### Operating temperature

- 0 to 50C

#### Dimensions (L x W x H)

- 9" x 6.5" x 2.1"
- 22.9 cm x 16.5 cm x 5.2 cm

#### Weight

- 4 lbs / 2 kg

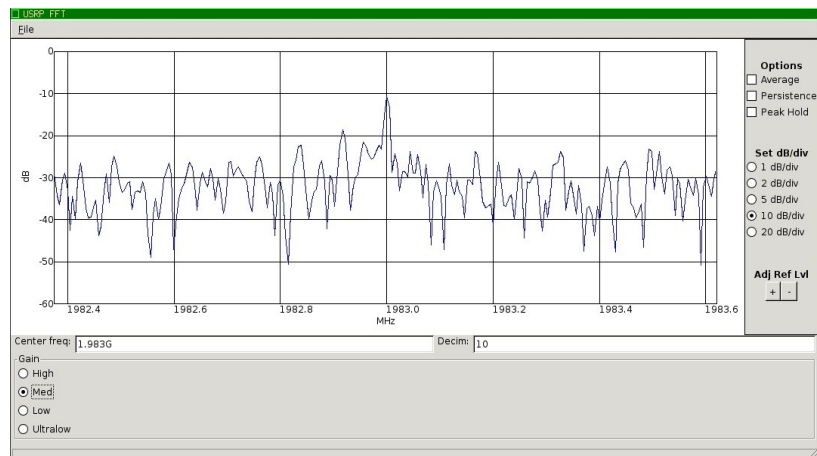
```
Usage: usrp_fft.py [options]
```

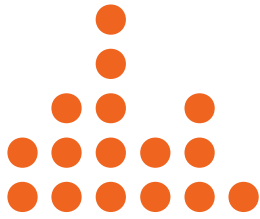
#### Options:

```
-d DECIM, --decim=DECIM      Set fgpa decimation rate to
                              DECIM [range 8 to 128, default=16]
-f FREQ, --freq=FREQ         Set frequency to FREQ
                              [specified without the Hz, but with the
                              SI prefix.. ie: 900M, 2.4G, etc]
-g GAIN, --gain=GAIN         Set gain in dB [allowable
                              values "Ultralow", "Low", "Med", or
                              "High", default is "Ultralow"]
-W, --waterfall              Enable waterfall display
--fft-size=FFT_SIZE          Set FFT frame size,
                              [allowable values are power of 2,
                              default=1024]
```

The following is an example command that configures the WSA1000U to sample at a center frequency of 1.983 GHz with a decimation rate of 10. The samples are fast Fourier transformed to the frequency domain and displayed as illustrate.

```
usrp_fft.py -d 10 -f 1.983G
```





# GNU Radio on the ThinkRF WSA1000U

## Application Example #2 - Time-Domain Display

'usrp\_oscope.py' is used to display real time results of time-domain data. Its usage is as follows:

```
Usage: usrp_oscope.py [options]
```

Options:

```
-d DECIM, --decim=DECIM  
-f FREQ, --freq=FREQ  
-g GAIN, --gain=GAIN
```

Options -d, -f and -g all have the same behaviour as the 'usrp\_fft.py' program.

## Application Example #3 - Capturing Data to a File

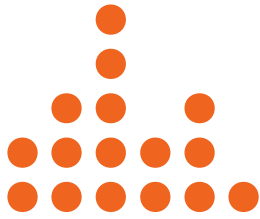
'usrp\_rx\_cfile.py' is used to capture complex floating point data to a file. An example of its usage is as follows:

```
usrp_rx_cfile.py -f 2450M -N 15000 -d 8 -g Med out.dat
```

Options -d, -f and -g all have the same behaviour as the 'usrp\_fft.py' program. The option -N is the number of samples to capture, after which the program will terminate. The last argument is the name of the file to write to.

The output file is formatted in such a way that the following example companion script for MATLAB or Octave will read the data from it.

```
% read I-Q data from GNU Radio and plot  
  
printf("Loading data...");  
fid=fopen('float.dat');  
  
readsize=2000;  
% e.g. if you want to read 2000 samples  
  
ir=1:2:2*readsize-1;  
ii=2:2:2*readsize;  
[val, count] = fread (fid, readsize*2, 'float');  
% times 2 for reading I/Q data  
  
s=complex(val(ir),-val(ii));  
  
% put the data into a complex vector  
fclose(fid);
```



# GNU Radio on the ThinkRF WSA1000U

## Installing GNU Radio on Ubuntu from source

### Build requirements

```
apt-get install libtool sdcc gsl-bin libgsl0-dev libusb-dev \  
guile-1.8 libboost1.40 libboost1.40-dev libcppunit-dev \  
libfftw3-dev swig automake autoconf libsdl1.2-dev \  
libxrender-dev libfontconfig-dev python-dev \  
python-wxgtk2.8 python-qt4 python-qt3d-qt4 \  
libqwt5-qt4-dev python-cheetah python-lxml
```

### Get Source

Obtain the source from

<http://www.thinkrf.com/downloads/gnuradio-v1.1-7d964ef.zip>

### Configuring

```
./bootstrap  
./configure --prefix=/opt/gnuradio-thinkrf --enable-ursp \  
--enable-gr-usrp --disable-usrp2 --disable-gr-usrp2
```

### Building

```
sudo make all install
```

### Configuring Udev

When you plug in a usb device, by default it is owned by root and you won't have access to it as a normal user. To remedy this, run the following commands.

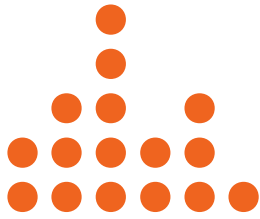
```
sudo cp contrib/wsa1000u.rules /etc/udev/rules.d/  
sudo invoke-rc.d udev restart
```

### Setting up Environment

Because you have installed into a non-standard location (/opt/gnuradio-thinkrf/), you need to update your path so that the gnuradio components can find what it needs. You can do this by running the following commands.

```
source contrib/env
```





# GNU Radio on the ThinkRF WSA1000U

## References and Resources

The official GNU Radio project home page (<http://gnuradio.org>) redirects to the project wiki where one will find links to "Getting Started", "Download" and "Build Guide", communication, documentation and a tutorial on how to write applications for GNU Radio in Python (<http://gnuradio.org/redmine/wiki/gnuradio/TutorialsWritePythonApplications>).

The Comprehensive GNU Radio Archive Network (CGRAN) is a free open source repository for 3rd party GNU Radio applications (<https://cgran.org/>). The RadioWare Project (<https://radioware.nd.edu/>) provides a good alternative source of documentation. GRC is a graphical tool for building gnuradio flowgraphs (<http://gnuradio.org/redmine/wiki/gnuradio/GNURadioCompanion>).

Some of the text within this document is derived from these resources.



## Contact Us

For more information on ThinkRF's products, applications or services, please contact us our sales team by phone +1.613.369.5104 ext 1 or by email [sales@thinkrf.com](mailto:sales@thinkrf.com) or visit our website at [www.thinkRF.com](http://www.thinkRF.com)

© ThinkRF Corporation 2010