Building and Installing the USRP Open-Source Toolchain (UHD and GNU Radio) on Linux

Contents

AN-445

This Application Note provides a comprehensive guide for building, installing, and maintaining the open-source toolchain for the USRP (UHD and GNU Radio) from source code on the Linux platform. The Ubuntu and Fedora distributions are specifically discussed. Several other alternate installation methods are also discussed.

http://files.ettus.com/manual/page_build_guide.html#build_instructions_unix

UHD is fully supported on Linux, using the GCC compiler, and should work on most major Linux distributions.

This document applies only to the USRP X300, X310, B200, B210, B200mini, N200, N210 devices. The E310 and E312 devices are embedded devices, and are fundamentally different from the other non-embedded USRP devices, and are not addressed by this document.

If you already have a recent version of Linux installed, then you may be able to skip this section. If you are starting from scratch, or simply want to start with a fresh new installation of Linux, then please follow the instructions and recommendations in this section.

We suggest that you use either Ubuntu 16.04.5, Ubuntu 18.04, Ubuntu 18.10, Fedora 27, 28, 29, and that you use a 64-bit architecture, not a 32-bit architecture. There are several re-spins of Ubuntu, such as Xubuntu, Lubuntu, Kubuntu, Linux Mint, all of which should also work. For the purposes of this document, these re-spins can be considered equivalent. Both Ubuntu and Fedora are known to work well with UHD and GNU Radio.

Download and install Ubuntu, Xubuntu, Linux Mint, or Fedora from the links below. Download the appropriate ISO image, and write it to a USB flash drive. Be sure to verify that the ISO file was not corrupted during the download process by checking the MD5 and/or SHA1 hash.

- Ubuntu download page
- Xubuntu download page
- Linux Mint download page
- Fedora download page

You can learn more about Ubuntu, Xubuntu, Linux Mint, and Fedora at the links below.

- Wikipedia article on Ubuntu
- Wikipedia article on Xubuntu
- Wikipedia article on Linux Mint
- Wikipedia article on Fedora

There are many tools for writing an ISO image to a USB flash drive. In Linux, you can use the "dd" utility, or the UNetbootin utility. On Ubuntu systems, there is also the Startup Disk Creator utility as well.

- UNetbootin homepage
- Wikipedia article on UNetbootin

- Startup Disk Creator homepage
- Wikipedia article on Startup Disk Creator
- Article about Startup Disk Creator

Be sure to use a USB flash drive with at least 8 GB capacity, and use a USB 3.0 flash drive, not a USB 2.0 flash drive. If you use a slower USB 2.0 flash drive, then the install process will take significantly longer.

Before building UHD and GNU Radio, you need to make sure that all the dependencies are first installed.

However, before installing any dependencies, you should first make sure that all the packages that are already installed on your system are up-to-date. You can do this from a GUI, or from the command-line, as shown below.

On Ubuntu systems, run:

```
sudo apt-get update
```

On Fedora 21 systems, run:

```
sudo yum update
```

On Fedora 22, 23, 24 and 25 systems, run:

```
sudo dnf update
```

Once the system has been updated, then install the required dependencies for UHD and GNU Radio.

On Ubuntu 22.04 systems, run:

```
sudo apt-get -y install autoconf automake build-essential ccache cmake cpufrequtils doxygen ethtool fort77 g++ gir1.2-gtk-3.0 git gobject-
```

On Ubuntu 20.04 systems, run:

```
sudo apt-get -y install autoconf automake build-essential ccache cmake cpufrequtils doxygen ethtool fort77 g++ gir1.2-gtk-3.0 git gobject-
```

On Ubuntu 18.04 systems, run:

```
sudo apt-get -y install git swig cmake doxygen build-essential libboost-all-dev libtool libusb-1.0-0 libusb-1.0-0-dev libudev-dev libncurs
```

On Ubuntu 16.04 systems, run:

```
sudo apt-get -y install git swig cmake doxygen build-essential libboost-all-dev libtool libusb-1.0-0 libusb-1.0-0-dev libudev-dev libncurs
```

On Fedora 21 systems, run:

```
sudo yum -y groupinstall "Engineering and Scientific" "Development Tools" "Software Development Tools" "C Development Tools and Libraries"
```

```
sudo yum -y install fftw-devel cppunit-devel wxPython-devel boost-devel alsa-lib-devel numpy gsl-devel python-devel pygsl python-cheetah p
```

On Fedora 22, 23, 24 and 25 systems, run:

```
sudo dnf -y groupinstall "Engineering and Scientific" "Development Tools" "C Development Tools and Libraries"
```

```
sudo dnf -y install fftw-devel cppunit-devel wxPython-devel boost-devel alsa-lib-devel numpy gsl-devel python-devel pygsl python-cheetah p
```

After installing the dependencies, you should reboot the system.

If the installation of the dependencies completes without any errors, then you can proceed to build and install UHD and GNU Radio.

UHD is open-source, and is hosted on GitHub. You can browse the code online at the link below, which points to version 3.14.0.0, which is the the latest release at the time of this writing.

- UHD repository on GitHub

There are several good reasons to build GNU Radio from source code, especially for doing development and prototyping. It it enables an easy way to customize the location of the installation, and to install multiple UHD versions in parallel, and switch between them. It also provides much more flexibility in upgrading and downgrading versions, and allows the user to modify the code and create customized versions, which could possibly include a patch or other bug-fix.

To build UHD from source code, clone the GitHub repository, check out a branch or tagged release of the repository, and build and install. Please follow the steps below. Make sure that no USRP device is connected to the system at this point.

First, make a folder to hold the repository.

```
cd $HOME
mkdir workarea
cd workarea
```

Next, clone the repository and change into the cloned directory.

```
git clone https://github.com/EttusResearch/uhd
cd uhd
```

Next, checkout the desired UHD version. You can get a full listing of tagged releases by running the command:

```
git tag -l
```

Example truncated output of `git tag -l`*:*

```
$ git tag -l
...
release_003_009_004
release_003_009_005
release_003_010_000_000
...
```

Note: As of UHD Version 3.10.0.0, the versioning scheme has changed to be a quadruplet format. Each element and version will follow the format of: **Major.API.ABI.Patch**. Additional details on this versioning change can be found here.

After identifying the version and corresponding release tag you need, check it out:

```
# Example: For UHD 3.9.5:
git checkout release_003_009_005

# Example: For UHD 3.14.0.0
git checkout v3.14.0.0
```

Next, create a build folder within the repository.

```
cd host
mkdir build
cd build
```

Next, invoke CMake.

```
cmake ..
```

Note: By default, UHD will be installed into the prefix `/usr/local`. This can be changed by adding `-DCMAKE_INSTALL_PREFIX=XXX` to install into the prefix `XXX`.

Note: if the shell `PATH` is set such that `/bin` comes before `/usr/bin`, then this step is likely to fail because cmake will set the `FIND_ROOT_PATH` to `/` and this setting will fail as a prefix for Boost headers or libraries. The cmake output will include messages such as

```
--   Boost include directories: /include
--   Boost library directories: /lib/x86_64-linux-gnu
```

and

```
CMake Error in lib/CMakeLists.txt:
  Imported target "Boost::chrono" includes non-existent path
    "/include"
  in its INTERFACE_INCLUDE_DIRECTORIES.  Possible reasons include:
  * The path was deleted, renamed, or moved to another location.
  * An install or uninstall procedure did not complete successfully.
  * The installation package was faulty and references files it does not provide.
```

One of the following 3 options should fix this situation:

1. `/usr/bin/cmake ..`
2. `PATH=/usr/bin:$PATH cmake ..`
3. `cmake -DCMAKE_FIND_ROOT_PATH=/usr ..`

Once the cmake command succeeds without errors, build UHD.

```
make
```

Note: Use the `-j` parameter to use more than the default one CPU core, e.g. `make -j8` for 8 CPU cores.

Next, you can optionally run some basic tests to verify that the build process completed properly.

```
make test
```

Next, install UHD, using the default install prefix, which will install UHD under the /usr/local/lib folder. You need to run this as root due to the permissions on that folder.

```
sudo make install
```

Next, update the system's shared library cache.

```
sudo ldconfig
```

Finally, make sure that the `LD_LIBRARY_PATH` environment variable is defined and includes the folder under which UHD was installed. Most commonly, you can add the line below to the end of your `$HOME/.bashrc` file:

```
export LD_LIBRARY_PATH=/usr/local/lib
```

On Fedora 22/23/24/25 you will need to set the `LD_LIBRARY_PATH` to `/usr/local/lib64`.

```
export LD_LIBRARY_PATH=/usr/local/lib64
```

If the `LD_LIBRARY_PATH` environment variable is already defined with other folders in your `$HOME/.bashrc` file, then add the line below to the end of your `$HOME/.bashrc` file to preserve the current settings.

```
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/lib
```

For Fedora 21/22/23/24/25

```
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/lib64
```

For this change to take effect, you will need to close the current terminal window, and open a new terminal.

At this point, UHD should be installed and ready to use. You can quickly test this, with no USRP device attached, by running `uhd_find_devices`. You should see something similar to the following.

```
linux; GNU C++ version 4.8.4; Boost_105400; UHD_003.010.000.HEAD-0-g6e1ac3fc

No UHD Devices Found
```

You can now download the UHD FPGA Images for this installation. This can be done by running the command `uhd_images_downloader`.

```
$ sudo uhd_images_downloader
```

Note: Since this installation is being installed to a system level directory (e.g. `/usr/local`), the `uhd_images_downloader` command requires `sudo` privileges.

Example ouput for UHD 3.13.3.0:

```
$ sudo uhd_images_downloader
Images destination:      /usr/local/share/uhd/images
Downloading images from: http://files.ettus.com/binaries/images/uhd-images_003.010.003.000-release.zip
Downloading images to:   /tmp/tmpm46JDg/uhd-images_003.010.003.000-release.zip
57009 kB / 57009 kB (100%)

Images successfully installed to: /usr/local/share/uhd/images
```

Example output for UHD 3.13:

```
$ sudo uhd_images_downloader
[INFO] Images destination: /usr/local/share/uhd/images
[INFO] No inventory file found at /usr/local/share/uhd/images/inventory.json. Creating an empty one.
00006 kB / 00006 kB (100%) usrp1_b100_fw_default-g6bea23d.zip
19484 kB / 19484 kB (100%) x3xx_x310_fpga_default-g494ae8bb.zip
02757 kB / 02757 kB (100%) usrp2_n210_fpga_default-g6bea23d.zip
02109 kB / 02109 kB (100%) n230_n230_fpga_default-g494ae8bb.zip
00522 kB / 00522 kB (100%) usrp1_b100_fpga_default-g6bea23d.zip
00474 kB / 00474 kB (100%) b2xx_b200_fpga_default-g494ae8bb.zip
02415 kB / 02415 kB (100%) usrp2_n200_fpga_default-g6bea23d.zip
```

```
05920 kB / 05920 kB (100%) e3xx_e320_fpga_default-g494ae8bb.zip
15883 kB / 15883 kB (100%) n3xx_n310_fpga_default-g494ae8bb.zip
00506 kB / 00506 kB (100%) b2xx_b205mini_fpga_default-g494ae8bb.zip
18676 kB / 18676 kB (100%) x3xx_x300_fpga_default-g494ae8bb.zip
00017 kB / 00017 kB (100%) octoclock_octoclock_fw_default-g14000041.zip
04839 kB / 04839 kB (100%) usb_common_windrv_default-g14000041.zip
00007 kB / 00007 kB (100%) usrp2_usrp2_fw_default-g6bea23d.zip
00009 kB / 00009 kB (100%) usrp2_n200_fw_default-g6bea23d.zip
00450 kB / 00450 kB (100%) usrp2_usrp2_fpga_default-g6bea23d.zip
00142 kB / 00142 kB (100%) b2xx_common_fw_default-g3ff4186b.zip
00460 kB / 00460 kB (100%) b2xx_b200mini_fpga_default-g494ae8bb.zip
00319 kB / 00319 kB (100%) usrp1_usrp1_fpga_default-g6bea23d.zip
00009 kB / 00009 kB (100%) usrp2_n210_fw_default-g6bea23d.zip
11537 kB / 11537 kB (100%) n3xx_n300_fpga_default-g494ae8bb.zip
05349 kB / 05349 kB (100%) e3xx_e310_fpga_default-g494ae8bb.zip
00866 kB / 00866 kB (100%) b2xx_b210_fpga_default-g494ae8bb.zip
[INFO] Images download complete.
```

As with UHD, GNU Radio is open-source and is hosted on GitHub. You can browse the code online at the link below, which points to version `v3.7.13.4`, which is the the latest release at the time of this writing.

- GNU Radio repository on GitHub

Note: GNU Radio is currently transitioning from major branches of 3.7.x.x to 3.8.x.x. It is generally recommend at this time to use either the `v3.7.13.4` or `maint-3.7` branch of GNU Radio. The `master` branch includes many major changes such as converting to use Python 3 and may be unstable.

As with UHD, there are several good reasons to build GNU Radio from source code, especially for doing development and prototyping. It it enables an easy way to customize the location of the installation, and to install multiple GNU Radio versions in parallel, and switch between them. It also provides much more flexibility in upgrading and downgrading versions, and allows the user to modify the code and create customized versions, which could possibly include a patch or other bug-fix.

Similar to the process for UHD, to build GNU Radio from source code, clone the GitHub repository, check out a branch or tagged release of the repository, and build and install. Please follow the steps below. Make sure that no USRP device is connected to the system at this point.

First, make a folder to hold the repository.

```
cd $HOME
cd workarea
```

Next, clone the repository.

```
git clone --recursive https://github.com/gnuradio/gnuradio
```

Next, go into the repository and check out the desired GNU Radio version.

```
cd gnuradio
```

To checkout the `v3.7.13.4` branch:

```
git checkout v3.7.13.4
```

Or to checkout the `maint-3.7` branch:

```
git checkout maint-3.7
```

Next, update the submodules:

```
git submodule update --init --recursive
```

Next, create a build folder within the repository, invoke CMake, and build GNU Radio:

```
mkdir build
cd build
cmake ../
make
```

Next, you can optionally run some basic tests to verify that the build process completed properly.

```
make test
```

Next, install GNU Radio, using the default install prefix, which will install GNU Radio under the /usr/local/lib folder. You need to run this as root due to the permissions on that folder.

```
sudo make install
```

Finally, update the system's shared library cache.

```
sudo ldconfig
```

At this point, GNU Radio should be installed and ready to use. You can quickly test this, with no USRP device attached, by running the following quick tests.

```
gnuradio-config-info --version
gnuradio-config-info --prefix
gnuradio-config-info --enabled-components
```

There is a simple flowgraph that you can run that does not require any USRP hardware. It's called the dialtone test, and it produces a PSTN dial tone on the computer's speakers. Running it verifies that all the libraries can be found, and that the GNU Radio run-time is working.

```
python $HOME/workarea/gnuradio/gr-audio/examples/python/dial_tone.py
```

You can try launching the GNU Radio Companion (GRC) tool, a visual tool for building and running GNU Radio flowgraphs.

```
gnuradio-companion
```

If "gnuradio-companion" does not start and complains about the `PYTHONPATH` environment variable, then you may have to set this in your `$HOME/.bashrc` file, as shown below.

```
    export PYTHONPATH=/usr/local/lib/python2.7/dist-packages
```

On Fedora 21/22/23/24, the `PYTHONPATH` environment variable will need to be set to:

```
    export PYTHONPATH=/usr/lib/python2.7/site-packages:/usr/local/lib64/python2.7/site-packages/
```

On Linux, udev handles USB plug and unplug events. The following commands install a udev rule so that non-root users may access the device. This step is only necessary for devices that use USB to connect to the host computer, such as the B200, B210, and B200mini. This setting should take effect immediately and does not require a reboot or logout/login. Be sure that no USRP device is connected via USB when running these commands.

```
    cd $HOME/workarea/uhd/host/utils
    sudo cp uhd-usrp.rules /etc/udev/rules.d/
    sudo udevadm control --reload-rules
    sudo udevadm trigger
```

For USRP devices that use Ethernet to connect to the host computer, such as the N200, N210, X300, X310, set a static IP address for your system of 192.168.10.1, with a netmask of 255.255.255.0. The default IP address of the USRP is 192.168.10.2, with a netmask of 255.255.255.0. You should probably set the IP address using the graphical Network Manager. If you set the IP address from the command line with `ifconfig`, Network Manager will probably overwrite these settings.

The installation of UHD and GNU Radio should now be complete. At this point, connect the USRP to the host computer.

If the interface is Ethernet, then open a terminal window, and try to ping the USRP with "ping 192.168.10.2". The USRP should respond to the ping requests.

If the interface is USB, then open a terminal window, and run "`lsusb`". You should see the USRP listed on the USB bus with a VID of `2500` and PID of `0020`, `0021`, `0022`, for B200, B210, B200mini, respectively.

Also try running "`uhd_find_devices`" and "`uhd_usrp_probe`".

When UHD spawns a new thread, it may try to boost the thread's scheduling priority. If setting the new priority fails, the UHD software prints a warning to the console, as shown below. This warning is harmless; it simply means that the thread will retain a normal or default scheduling priority.

```
UHD Warning:
    Unable to set the thread priority. Performance may be negatively affected.
    Please see the general application notes in the manual for instructions.
    EnvironmentError: OSError: error in pthread_setschedparam
```

To address this issue, non-privileged (non-root) users need to be given special permission to change the scheduling priority. This can be enabled by creating a group `usrp`, adding your user to it, and then appending the line `@usrp - rtprio 99` to the file `/etc/security/limits.conf`.

```
    sudo groupadd usrp
    sudo usermod -aG usrp $USER
```

Then add the line below to end of the file `/etc/security/limits.conf`:

```
    @usrp - rtprio  99
```

You must log out and log back into the account for the settings to take effect. In most Linux distributions, a list of groups and group members can be found in the `/etc/group` file.

There is further documentation about this in the User Manual at the link below.

- Threading Notes section of the User Manual