**4G Setup**

# Refer these links

<https://docs.srsran.com/projects/4g/en/latest/>

<https://docs.srsran.com/projects/4g/en/latest/getting_started.html>

<https://docs.srsran.com/projects/4g/en/latest/general/source/1_installation.html#gen-installation>

<https://docs.srsran.com/projects/4g/en/latest/usermanuals/source/1_setup.html#gen-4g>

------------------------------------------------------------------------------------------------------------------------

# Follow the installation from the source section

<https://docs.srsran.com/projects/4g/en/latest/general/source/1_installation.html#gen-installation>

**Step 1**

--You should install all listed libraries under mandatory requirements

**Common**

1. **Cmake**

To install CMake on Ubuntu, you can follow these steps:

1; Open a terminal window.

2; Update the package lists using the following command:

**sudo apt update**

3; Install CMake by running the following command:

**sudo apt install cmake**

Enter your password if prompted and press Enter to confirm the installation.

4; Wait for the installation to complete. Once it's done, you can verify the installation by checking the CMake version:

**cmake –version**

If the installation was successful, you will see the version number displayed in the terminal.

That's it! You have successfully installed CMake on your Ubuntu system

1. **Libfftw**

**Run Both**

To install the specific version of FFTW (3.3.10) on Ubuntu, you will need to build it from source. Here are the steps to follow:

1; Open a terminal window.

2; Update the package lists using the following command:

sudo apt update

3; Install the necessary dependencies for building FFTW by running the following command:

sudo apt install build-essential

4; Download the FFTW source code. You can visit the FFTW website (http://www.fftw.org/download.html) to obtain the latest source code package. Alternatively,

5; you can use the following command to download version 3.3.10 directly:

wget http://www.fftw.org/fftw-3.3.10.tar.gz

6. Extract the downloaded source code package using the following command:

tar -xf fftw-3.3.10.tar.gz

7; Navigate to the extracted directory:

cd fftw-3.3.10

8; Configure the build by running the following command:

./configure

9; Build the FFTW library using the following command:

make

10; Install the library system-wide by running the following command with root privileges:

sudo make install

11; Wait for the installation to complete. Once it's done, you should have FFTW 3.3.10 installed on your system.

That's it! You have successfully installed FFTW 3.3.10 on your Ubuntu system from source

--------------------------------------------------------------------------------------------------------------------

you can use the pkg-config command-line tool to check the presence of the FFTW library and obtain information about its version. Here's the command you can use:

**pkg-config --modversion fftw3**

Running this command will output the version number of the installed FFTW library, if it is present. If the library is not installed, the command will return an error.

Other method

To install FFTW (Fastest Fourier Transform in the West) on Ubuntu, you can follow these steps:

1; Open a terminal window.

2; Update the package lists using the following command:

sudo apt update

3; Install FFTW by running the following command:

sudo apt install libfftw3-dev

Enter your password if prompted and press Enter to confirm the installation.

4; Wait for the installation to complete. Once it's done, you can start using FFTW in your programs.

That's it! You have successfully installed FFTW on your Ubuntu system.

1. **mbedTLS**

To install MbedTLS on Ubuntu, you can follow these steps:

1; Open a terminal on your Ubuntu system.

2; Update the package lists and upgrade the existing packages by running the following commands:

sudo apt update

sudo apt upgrade

3; Install the necessary dependencies for building MbedTLS by executing the following command:

sudo apt install cmake build-essential

4; Download the MbedTLS source code. You can either download it from the official MbedTLS website or clone the GitHub repository. For example, to clone the GitHub repository, use the following command:

git clone https://github.com/ARMmbed/mbedtls.git

5; Change to the MbedTLS source code directory by running the following command:

cd mbedtls

6; Create a build directory and navigate to it:

mkdir build

cd build

7; Generate the build files using CMake. Run the following command:

cmake ..

8; Build the MbedTLS library by executing the following command:

make

9; Optionally, you can run the MbedTLS test suite to verify the library's functionality. Use the command:

make test

10; Once the build process completes successfully, you can install MbedTLS on your system by running:

sudo make install

MbedTLS should now be installed on your Ubuntu system. You can verify the installation by including the necessary headers and linking against the library in your programs.

These steps should guide you through the process of installing MbedTLS on Ubuntu.

---------------------------------------------------------------------------------------------------------------------------

**The error you encountered regarding the "ModuleNotFoundError: No module named 'jinja2'" indicates that the Jinja2 Python library is missing. The generate\_driver\_wrappers.py script in the Mbed TLS repository relies on Jinja2 for generating driver wrappers.**

To resolve this issue, you can install Jinja2 on your system by following these steps:

1; Open a terminal on your Ubuntu system.

Ensure that you have the pip package manager installed. If you don't have it, you can install it by running the following command:

sudo apt install python3-pip

2; Once pip is installed, use it to install Jinja2 by executing the following command:

sudo pip3 install jinja2

Note: You might need to use pip instead of pip3 if you're using a different version of Python.

After installing Jinja2, attempt to build Mbed TLS again by running the make command in the Mbed TLS build directory:

make

This should resolve the "ModuleNotFoundError: No module named 'jinja2'" issue by installing the Jinja2 library. Once the installation is successful, you can proceed with the remaining steps to complete the Mbed TLS installation.

**srsUE**

1. **Boost C++ library**

1; Downlaod the latest version from <https://www.boost.org/>

2; Open a terminal on your Ubuntu system.

3; Navigate to the directory where you have extracted the boost\_1\_82\_0.tar.gz file. You can use the cd command followed by the directory path to move to the desired location.

4; In the terminal, navigate to the Boost directory:

cd boost\_1\_82\_0

5; Run the bootstrap script to configure the installation. Execute the following command:

./bootstrap.sh

6; After the bootstrap process completes, run the b2 command to build and install Boost. Use the following command:

sudo ./b2 install

Note: The sudo command is used here to run the installation with administrative privileges, as it may require writing files to system directories.

7; The Boost libraries will be built and installed on your system. The installation process may take some time.

8; Once the installation is complete, you can verify that Boost is installed correctly by compiling a simple C++ program that uses Boost.

**srsENB**

1. **Boost C++ library**
2. **Lksctp**

To install the Linux Kernel Stream Control Transmission Protocol (SCTP) tools on Ubuntu, you can follow these steps:

1; Open a terminal on your Ubuntu system.

2; Update the package lists by running the following command:

sudo apt update

3; Install the lksctp-tools package, which includes the SCTP tools, by executing the following command:

sudo apt install lksctp-tools

4; During the installation, you may be prompted to enter your password. Provide your password and press Enter to continue.

5; The package manager will fetch the necessary files and install the lksctp-tools package, which includes the SCTP tools.

6; After the installation is finished, you can verify that the SCTP tools are installed by running one of the SCTP-related commands. For example, you can try running the sctp\_status command to display SCTP association information:

sctp\_status

If the SCTP tools are installed correctly, you should see output related to SCTP associations. If you receive an error message stating that the command is not found, double-check the installation steps and ensure that the lksctp-tools package is installed.

These steps should guide you through the process of installing the Linux Kernel SCTP tools on Ubuntu.

1. **Config**

1; Open a terminal on your Ubuntu system.

2; Update the package lists by running the following command:

sudo apt update

3; Install the libconfig development package by executing the following command:

sudo apt install libconfig-dev

This command will install the libconfig library and the necessary header files needed for development.

**srsEPC**

1. **Boost**
2. **lkctp**
3. **config**

**Note**

For example, on Ubuntu, one can install the required libraries with:

**sudo apt-get install build-essential cmake libfftw3-dev libmbedtls-dev libboost-program-options-dev libconfig++-dev libsctp-dev**

**Step 2**

# Install UHD RF front end drive

To install the UHD (USRP Hardware Driver) RF-front end driver on Ubuntu, you can follow these steps:

1; Open a terminal on your Ubuntu system.

2; Update the package lists by running the following command:

sudo apt update

3; Install the necessary dependencies for building UHD by executing the following command:

sudo apt install libboost-all-dev libusb-1.0-0-dev python3-mako doxygen python3-docutils cmake build-essential

This command will install the required development libraries and tools.

4; Clone the UHD repository from the official GitHub repository by running the following command:

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

5; Change to the UHD directory:

cd uhd

6; Create a build directory and navigate to it:

mkdir build

cd build

7; Generate the build files using CMake. Run the following command:

cmake ../host

8; Build UHD by executing the following command:

make

This command will build the UHD library and the RF-front end driver.

9; Install UHD by running the following command with administrative privileges:

sudo make install

This command will install the UHD library and the RF-front end driver system-wide.

10; Optionally, you can run the UHD tests to verify the installation. Use the following command:

make test

UHD and the RF-front end driver should now be installed on your Ubuntu system. You can proceed to use and configure the driver according to your requirements.

These steps should guide you through the process of installing the UHD RF-front end driver on Ubuntu.

**The error message you encountered indicates that there is a mismatch between the source directory specified when generating the CMake cache and the current source directory. This can happen if you have moved or renamed the source directory after generating the cache.**

To resolve this issue, you can follow these steps:

1; Remove the existing build directory to start fresh. In the terminal, navigate to the parent directory of the build directory and delete it using the following command:

rm -rf build

2; Verify that you are in the correct source directory. Double-check that you are in the correct directory where the CMakeLists.txt file is located. You mentioned the path "/home/aruna/Music/New\_4G\_Setup/srsRAN\_4G" in the error message, so make sure you are in that exact directory.

3; Recreate the build directory and navigate to it:

mkdir build

cd build

4; Generate the build files using CMake, ensuring that you provide the correct source directory. Run the following command:

cmake ../

Make sure to specify the correct path to the source directory (the one mentioned in the error message) when running CMake.

5; Continue with the build and installation steps as mentioned in the instructions specific to the software you are trying to install.

**Step 3**

# Download and build srsRAN 4G:

git clone https://github.com/srsRAN/srsRAN\_4G.git

cd srsRAN\_4G

mkdir build

cd build

cmake ../

make

make test

# Install srsRAN 4G:

sudo make install

srsran\_install\_configs.sh user

This installs srsRAN 4G and also copies the default srsRAN 4G config files to ~/.config/srsran\_4g.

**Step 4**

1; After installing SRSRAN 4G copy these config files to “~/.config/srsran”

2; To run the SRS EPC as the core network, run the following commands;

cd ~/.config/srsran

sudo srsepc epc.conf

**When running "sudo srsepc epc.conf" it says as "srsepc: error while loading shared libraries: libboost\_program\_options.so.1.82.0: cannot open shared object file: No such file or directory"**

ChatGPT

The error message you encountered while running "sudo srsepc epc.conf" indicates that the "srsepc" executable is unable to find the shared library file named "libboost\_program\_options.so.1.82.0." This suggests that the required Boost Program Options library is either missing or not located in the expected directory.

To resolve this issue, you can try the following steps:

1; Install the necessary Boost libraries: First, ensure that the Boost library version 1.82.0 or compatible version is installed on your system. You can typically install it using a package manager like apt (for Ubuntu/Debian) or yum (for CentOS/RHEL). The package name might be something like "libboost-program-options-dev" or "boost-devel".

2; Update the library search path: If the library is already installed but not found in the default search path, you can update the library search path by setting the LD\_LIBRARY\_PATH environment variable. For example, you can run the following command before executing "sudo srsepc epc.conf":

export LD\_LIBRARY\_PATH=/path/to/boost/libraries:$LD\_LIBRARY\_PATH

Replace "/path/to/boost/libraries" with the actual directory path where the Boost libraries are installed.

3; Update the dynamic linker cache: If you have installed the Boost libraries recently, you might need to update the dynamic linker cache. Run the following command as root:

sudo ldconfig

This command refreshes the cache and makes the newly installed libraries available for linking.

After performing these steps, try running "sudo srsepc epc.conf" again. If the error persists, ensure that the Boost Program Options library is correctly installed and that the library file "libboost\_program\_options.so.1.82.0" exists in a directory included in the library search path.

How to get the actual directory path where the Boost libraries are installed.

ChatGPT

To determine the actual directory path where the Boost libraries are installed, you can try the following methods:

1; Package Manager Information: If you installed Boost libraries using a package manager, you can use the package manager's information to find the installation path. For example, on Ubuntu/Debian, you can run the following command to list installed packages related to Boost:

dpkg -l | grep libboost

Look for the package name that corresponds to the version you installed, and it may provide information about the installation path.

2; System-wide Library Paths: On Unix-like systems, shared libraries are typically stored in specific system-wide directories, such as /usr/lib or /usr/local/lib. You can search for the Boost library file using the find command. Open a terminal and run:

find /usr/lib /usr/local/lib -name "libboost\_program\_options.so.\*"

This command will search for any file matching the pattern "libboost\_program\_options.so.\*" within the /usr/lib and /usr/local/lib directories. It should display the path to the Boost library file.

3; Custom Installation: If you installed Boost libraries manually from source or through a custom installation process, the installation path may vary. In such cases, you may need to remember or locate the directory you specified during the installation. By default, Boost libraries are often installed in /usr/local/lib or /usr/local/include directories.

Remember that the actual Boost library installation path can depend on various factors, including your system's configuration and the method you used for installation.

"/usr/local/lib/libboost\_program\_options.so.1.82.0" is where the boost library is located

ChatGPT

Thank you for providing the specific location of the Boost library. Based on the information you provided, the Boost Program Options library file "libboost\_program\_options.so.1.82.0" is located at "/usr/local/lib/libboost\_program\_options.so.1.82.0".

Since the error message indicated that the shared object file could not be found, you can try the following steps to resolve the issue:

1; Check the library file: Verify that the Boost library file mentioned above exists at the specified location. You can use the following command to confirm:

ls /usr/local/lib/libboost\_program\_options.so.1.82.0

If the file is found, it should be listed in the output of this command.

2; Update the library cache: If you have recently installed the Boost library or made changes to its location, you might need to update the library cache. Run the following command as root:

sudo ldconfig

This command refreshes the cache and ensures that the system recognizes the newly installed library.

3; Set the library search path: If the Boost library is still not being found, you can set the library search path explicitly using the LD\_LIBRARY\_PATH environment variable. Execute the following command before running the "sudo srsepc epc.conf" command:

export LD\_LIBRARY\_PATH=/usr/local/lib:$LD\_LIBRARY\_PATH

This command adds "/usr/local/lib" to the library search path, allowing the system to locate the Boost library.

After performing these steps, try running "sudo srsepc epc.conf" again. If the error persists, make sure the Boost library is correctly installed and that the library file is accessible at the specified location.

3; To run the ENB as the EnodeB, plug the USRP B210 device to the computer and run the following commands;

cd ~/.config/srsran

sudo srsenb enb.conf

srsenb: error while loading shared libraries: libsrsran\_rf.so.0: cannot open shared object file: No such file or directory

1. 1; Check the library file:

ls /usr/local/lib/libsrsran\_rf.so.0

If the file is found, it should be listed in the output of this command.

2. 2; Update the library cache: If you have recently installed the library or made changes to its location, you might need to update the library cache. Run the following command as root:

sudo ldconfig

This command refreshes the cache and ensures that the system recognizes the newly installed library.

3. 3; Set the library search path: If the library is still not being found, you can set the library search path explicitly using the LD\_LIBRARY\_PATH environment variable. Execute the following command before running the "sudo srsenb enb.conf" command:

export LD\_LIBRARY\_PATH=/usr/local/lib:$LD\_LIBRARY\_PATH

This command adds "/usr/local/lib" to the library search path, allowing the system to locate the Boost library.

--------------------------------------------------------------------------------------------------------------------

4; Copy ip\_route.sh file located in downloads to path "~/.config/srsran" and run it

sudo bash ./ip\_route.sh

---Then what is left is connecting the UE