CpE 5430 / EE 5430 / SysEng 5323 Wireless Networks
Lab assignment A.3

IMPORTANT: the installation from source (GIT -> compile/make) is REQUIRED to continue Advanced GNU Radio Module (Lab Module C)! If you plan to work on Advanced Ns3 Lab Module only, then you can install GNU Radio from repositories (e.g. “apt install gnuradio” on Ubuntu)

Prerequisite: Setup the GNU Radio (preferred version 3.8 from the main branch, but 3.9 should also be acceptable)

Setup Ubuntu on Win10 using WSL/WSL2 (https://docs.microsoft.com/en-us/windows/wsl/install-win10 , https://wiki.ubuntu.com/WSL). Then install dependencies for the GNU Radio (https://wiki.gnuradio.org/index.php/UbuntuInstall). Finally, download and install GNU Radio source code (https://wiki.gnuradio.org/index.php/InstallingGR#From_Source). Also install Xorg based server for Win10 (e.g. https://sourceforge.net/projects/vcxsrv/)

Note that there is a potential errors/issues occurring:

A) due to Qt5 library issues – to solve it execute following command:

```
strip --remove-section=.note.ABI-tag /usr/lib/x86_64-linux-gnu/libQt5Core.s
```

B) PYTHON library path -> check https://wiki.gnuradio.org/index.php/ModuleNotFoundError#B._Finding_the_Python_library

C) Set Python and library environmental settings (https://wiki.gnuradio.org/index.php/ModuleNotFoundError#B._Finding_the_Python_library) e.g.:

```
export PYTHONPATH=/usr/local/lib/python3/dist-packages:$PYTHONPATH
export LD_LIBRARY_PATH=/usr/local/lib:$LD_LIBRARY_PATH
```

D) The Linux files (home folder) can be found in C:\Users**YOUR_USERNAME**\AppData\Local\Packages\CanonicalGroupLimited.Ubuntu20.04onWindows_79rhkp1fndgsc\LocalState\rootfs\home**YOUR_USERNAME**

The files in Linux have to be properly setup (access control). If downloading/changing through Win10 – you may need to change permissions – useful commands:

```
ls –al
chmod a+rw filename.ext
chmod a+rw *
```

--or--

Create an Ubuntu virtual machine. Here is a link to a tutorial: http://www.wikihow.com/Install-Ubuntu-on-VirtualBox Contact your TA for a special iso image and substitute this image for the one mentioned in the tutorial. **We do not recommend this method because GNU Radio has been known to have graphical bugs inside virtual machines.**

--or--

Use the Linux servers on campus network (check it.mst.edu for guidance to use these) Note: the GNU Radio installation have not been verified for most recent versions.

CpE 5430 / EE 5430 / SysEng 5323 Wireless Networks
Lab assignment A.3

---

1. Open "CpE_5430_Lab1-2.grc"
2. Notice this flow graph has four file sinks and will create 4 files when it is run. Also, notice on the file source, that Repeat is set to Yes. This means the file will be read and re-read continuously in a loop. There is no content in TOP_BLOCK window (it is visible but empty).
3. The file type from the file source is "byte" or "char" (indicated by the color on the out port) meaning that 1 byte of data will leave the file source at the sampling rate (i.e. 32k bytes/second). Similarly, the first file sink saves the data 1 byte at time. This file sink will record a perfect duplicate of everything transmitted.
4. The second file sink has 2 function blocks before it: Unpack K Bits, and Pack K Bits. Unpack K bits with K = 8 will separate each bit form every byte into 8 separate bytes with one bit each. For example:

   A single byte with bits b_n = b_7 b_6 b_5 b_4 b_3 b_2 b_1 b_0;
   b_7 b_6 b_5 b_4 b_3 b_2 b_1 b_0 -> Unpack K Bits -> 000000b_0, 000000b_1, 000000b_2, 000000b_3, 000000b_4, 000000b_5, 000000b_6, 000000b_7

   Similarly Pack K Bits with K= 2 will take the LSB of 2 subsequent bytes it's given into new bytes. Given two bytes where the only the LSB is significant: 000000b_0, 000000b_1
   000000b_0, 000000b_1 -> Pack K Bits -> 000000b_1b_0

   The result with K = 8 for Unpack K Bits and K = bits/symbol for Pack K Bits is that the second file sink will record a representation of every symbol sent.
5. The PSK Demod block outputs decoded symbols as bytes, but in a binary representation. This means its output looks exactly like that of Unpack K Bits in the previous example. Notice the PSK Demod block has two file sinks. The first features a Pack K bits with K = 8. This sink will reconstruct the sent file. Its output should be identical to sent_bytes.txt file. As you may have guessed the second file sink reconstructs data received in each symbol.
6. You will be running a series of simulations to determine the effect of channel attenuation and channel noise on symbol error rate and indirectly on bit error rate.
   a. Pick 3 noise voltage levels in the range [0,0.5]
   b. Pick 3 attenuation levels in the range [1,0]
   c. Use M=2 with samples_per_sym =2 for BPSK and M=4 with samples_per_sym=4 for QPSK

   There should lead to 3x3x2 = 18 combinations. Make sure to include a control with a perfect channel noise=0 and attenuation=1 (you might be surprised at the results).

Procedure:

1. Open a terminal to the same directory with your working files.
2. Edit noise and attenuation levels by changing their variable values.
3. Execute the flow graph and let run 2-3 seconds. Stop it by clicking the red "X" next to the execute button.

4. Calculate the symbol error rate using the following commands entered in a terminal in following order:
    a. hexdump sent_symbols.bin -v -e '1/1 "%02X\n"' > in.txt
    b. hexdump recieved_symbols.bin -v -e '1/1 "%02X\n"' > out.txt
    c. sdiff in.txt out.txt > diff.txt
    d. grep -o '<\|>\||' diff.txt | wc -l
5. The last command will return the symbol rate error.
6. Repeat starting at step 2 for all 18 combinations.

For your lab report

- Observations from execution and results.
- Plot symbol error rate vs noise (i.e. those simulations with attenuation=1 and noise level as a variable).
- Plot symbol error rate vs channel attenuation (i.e. those simulations with noise=0 and attenuation level as a variable).
- Create a 3D plot of channel attenuation & noise level vs symbol error rate.
- Briefly explain the error rate results.

Additional Notes

- Those experienced with shell scripting or python scripting are encouraged to automate this process
- Highly recommended GNU Radio video tutorial series: http://www.youtube.com/watch?v=N9SLAnGlGQs&list=PL618122BD66C8B3C4
- Tutorial from gnuradio.org: https://wiki.gnuradio.org/index.php/HowToUse , https://wiki.gnuradio.org/index.php/TutorialsWritePythonApplications
- Installation Instructions for the daring: https://wiki.gnuradio.org/index.php/InstallingGR