**Migration Process Notes**

Porting guide: [https://wiki.gnuradio.org/index.php/GNU\_Radio\_3.8\_OOT\_Module\_Porting\_Guide#API\_Changes](https://wiki.gnuradio.org/index.php/GNU_Radio_3.8_OOT_Module_Porting_Guide" \l "API_Changes)

\* Most of the problems encountered appear to be related to makefile updates rather than code itself, such as missing library bindings. So this process seems to work well:

1. Prep
   1. Copy your current working 3.7 module somewhere "safe" (aka make a backup)
   2. Copy the current OOT to an OOT37 directory
   3. From the new OOT37 directory, run "gr\_modtool update --complete" to migrate the grc/xml files over to YML
2. Creating the new structure and copyng the files that can be directly copied
   1. gr\_modtool newmod <oot name>
   2. cp <oot37>/include/<module>/\* <oot>/include/<module>/
   3. cp <oot37>/grc/\* <oot>/grc
   4. Search grc/yml files for vlen and make sure it translated the variable name correctly. If there’s a vlen if statement or something in the file, the gr\_modtool isn’t picking up the name right and using it. It’s defaulting to vlen, which may be wrong.
   5. cp <oot37>/lib/\*.c\* <oot>/lib/
   6. Cp <oot37>/lib/\*.h <oot>/lib/
   7. cp <oot37>/swig/<oot>\_swig.i <oot>/swig
   8. cp <oot37>/README.MD to your new <oot>
   9. cp <oot37>/LICENSE to your new <oot>
   10. cp <oot37>/examples to <oot>/examples
   11. If you're tied to a git repo, also copy <oot37>/.git to <oot>. This will preserve your "master" branch tie. Later you'll create a new branch from your existing <oot37> code, so this will keep your tie to master (if that's what you want).
   12. Anything else unique to this module to address?
3. Python Modules
   1. The python directory you have to be a little more careful with. Copy your .py module files but not the others (\_\_init\_\_.py, etc.)
   2. Edit the new \_\_init\_\_.py and add in the custom module imports using a relative . Reference
   3. If you have any python 2.7 print “” statements, convert them over to print() calls. Python tool 2to3 automates 90% of conversion.
   4. If you are using strings in any way with transmission mechanisms such as sockets, Python3 handles these differently. You'll need to strval.encode("UTF-8") and strval.decode("UTF-8") where you just used straight strings before to convert them to/from bytes. This can cause some significant issues if one side of the link is python2 and the other is python3. For instance, using mprpc's RPCClient from python2 on one side to python3 on the other almost won't work due to the byte format change issue. With that said, in most cases adding the .encode() and .decodde() will fix the issue.
4. Lib Makefile
   1. Go through lib/CmakeLists.txt and look for any libraries or find\_package calls that are not standard. Generally this will be around a line that looks like this:
   2. target\_link\_libraries(gnuradio-grnet gnuradio::gnuradio-runtime <ANY MISSING LIBRARIES/MODULES HERE>)
   3. Add all library .cc files back into the "list(APPEND <modulename>\_sources" list
5. At this point you can cmake/build/install
6. Notes
   1. If when testing in GR, it says module-not-found, the most common cause was a missed linked library in lib/Cmake. For instance if you were using a gnuradio component that now isn't in your link list.
   2. Make sure in your yml files that parameters have a dtype defined.
   3. Any hide: attributes will no longer take ''. Change those to none.
7. Once all is working, move the new <oot> directory to <oot38> and restore your old <oot> from the backup you first made (not the one you ran the update --complete in)

**Updating Github**

Prep:

In the 3.7 module copy the .git directory over to the new 3.8 directory (this will have the reference to master so we can make all the master add/update/deletes)

In the 3.7 module:

git tag v3.7

git checkout -b maint-3.7

git push --set-upstream origin maint-3.7

In the new 3.8 module:

Do a git status and resolve any not-added files (like the yml's, etc.)

When ready do a git commit and git push.