# NetSpeed Release Flow

## Goal:

Automated construction of customized Orion releases including NocStudio, RTL, Testbench, C++ Model, example scripts, and associated documentation.

## Method:

The directory trunk/sw/noc\_stable/src has a Makefile with targets that create a trunk/sw/noc\_release directory containing the release. It also creates a tarball of this directory for customers.

Directory contents are as follows:

* NocStudio Executable and GUI.jar – needed to run NocStudio
* nocinit.txt – startup script for NocStudio
* custom\_header.txt – header that will be placed on all NocStudio-produced files
* run\_bench.sh – script to run a testbench generated by NocStudio
* docs/ - Documentation files go in this dir
* examples/ - Example NocStudio scripts
* model/ - C++ model
* rtl\_common/ - All RTL files needed to build NoC
* synth\_common/ - Example Synthesis scripts
* templates/ - Template files to customize bridge creation
* verif\_common/ - All verification files including templates used by NocStudio

## Build options:

### RTL

RTL generation can be in any of the following three modes:

* Enabled: Full RTL is generated for customer; this RTL is in two files: $PROJ\_noc.v and $PROJ\_noc\_modules.v. ($PROJ is name of project) The noc\_modules.v file creates modules for each router and bridge in the noc by instantiating the generic router/bridge designs with parameters and tying off any pins that are not needed at the top level. The noc.v file connects these custom routers and bridges to each other, adding pipeline modules and exposing the bridge external interfaces.
* Disabled: No RTL generation available; user will get an error message notifying that RTL generation is not available. If RTL generation is disabled, the rtl\_common/ and synth\_common/ directories are not included in the release.
* Bridgeless: This mode generates a noc without any bridges, exposing the router->bridge connections at the top of the noc.v file. The structure is otherwise the same as rtl generation enabled. *Note: It is currently untested and may not be working at the moment.*

### Testbench

Testbench generation can be enabled or disabled. If the testbench generation is enabled, full RTL generation is automatically enabled. If the testbench generation is disabled, the verif\_common/ directory will not be included in the release.

### C++ Model

The C++ model can be included or excluded from the release. If included in the release, the model includes a library libnetspeed.a and a C++ header file NocStudio.h to be used by customers. An example program using this interface is also included as source, called APITest.

### Internal

This build option for NocStudio includes all RTL and Testbench generation as well as a few internal-only commands such as self-test (to run self-tests), run\_model (to test the model from API), and verify\_adj (to check that port connections are valid). As well, it enables the ignore\_unmappable\_hop\_dep property, allows wider range of data\_widths for interfaces, and allows cell size as small as one.

## Obfuscation

RTL obfuscation is done through the vrename tool. It renames the parameters internal to the Verilog source, generating random tokens encoding the parameter names. It also produces a name mapping file that is converted indicating the encrypted names of each parameter. This is transformed into a GenRTL\_obf.h file for nocstudio compilation, allowing genrtl to produce Verilog with working obfuscated parameters.

*Current status:* incomplete; streaming bridge generation isn’t written to use genrtl\_obf, possibly some newer parameters of routers are also not able to be obfuscated.

# Technical details

## Makefile helper targets

The makefile has the following targets for adding things to the release:

* Copy\_release\_base:

Makes release folder, nocstudio executable, copies nocstudio + gui + nocinit + custom\_header and templates/ into release

* Copy\_release\_rtl:

Makes and populates rtl\_common/ and synth\_common/. The makefile includes the list of files to be included. This includes creating noc\_common.vc that lists all rtl files copied into rtl\_common

* Copy\_release\_verif

Makes and populates verif\_common/. Makfile contains the list of files to be included. A verif\_common.vc file is created that lists all files copied. Verif\_common/templates

* Copy\_release\_docs

Copies documentation and example scripts into release. All test\_scripts/release/ scripts are copied, the documentation copied is listed in Makefile.

* Make\_release\_tarball

Creates a tarball from the release folder

## Makefile full-release targets

The following targets are shortcuts for building a complete release of various forms

* Nogen-release

NocStudio built without GEN\_RTL, copy\_release\_{base, docs, tarball}

* Notb-release

NocStudio built with GEN\_RTL (no GEN\_TB), copy\_release\_{base,rtl,docs,tarball}

* Testbench-release

NocStudio built with GEN\_RTL and GEN\_TB, copy\_release\_{base,rtl,docs,verif,tarball}

* Edna-release

NocStudio built with GEN\_RTL and GEN\_TB, copy\_release\_{base,rtl,docs,verif,model,tarball}

* Amazon-release: shortcut for notb-release