



# NoximGUI User & Development Guide

---



*January 2017*

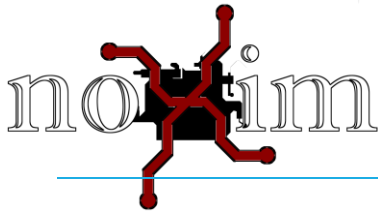
*Version 1.0*

**Michael V Thanh**

*Department of Electrical & Computer Engineering*

*Colorado State University*

*Fort Collins, CO 80031*



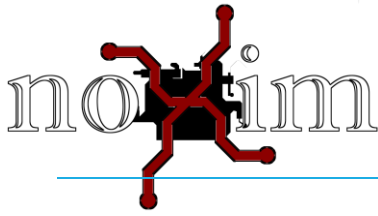
© 2017 Michael V Thanh

GNU GENERAL PUBLIC LICENSE  
Version 3, 29 June 2007

## Citations/Credits:

V. Catania, A. M. Ineo, S. Montebone, M. Palesi, D. Patti. Noxim : An Open, Extensible and Cycle-accurate Network on Chip Simulator. IEEE International Conference on Application-specific Systems, Architectures and Processors 2015. July 27-29, 2015, Toronto, Canada.



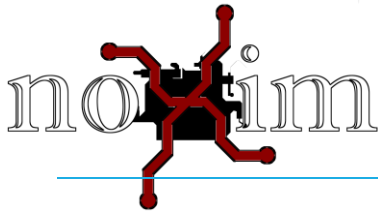


---

## Document Revisions

Date	Version Number	Document Changes
12/05/2016	0.1	Created guide and made changes from template, added introduction and scope of project
12/07/2016	0.2	Wrote "Installation" section, General Usage
12/12/2016	0.3	Added GNU Public License information
12/16/2016	0.4	Wrote "Installation" section, For Development
12/20/2016	0.5	Added Usage section
12/21/2016	0.6	Modified/editied Usage section, added Error Reporting
12/30/2016	0.7	Added in links between document sections, added screenshots
01/04/2017	0.8	Modified Run Configurations section, general document editing
01/12/2017	0.9	Additions to the "Development" section... technical issues encountered
01/24/2017	0.10	Edits, modifications
01/26/2017	1.0	Fixed Development section, final edits, sent to Prof. Sudeep





## Table of Contents

<b>1</b>	<b>Introduction .....</b>	<b>5</b>
1.1	.... <i>Scope and Purpose</i> .....	5
1.2	.... <i>Process Overview</i> .....	5
<b>2</b>	<b>Installation .....</b>	<b>6</b>
2.1	.... <i>[For General Usage]</i> .....	6
2.2	.... <i>[For Development]</i> .....	7
2.2.1	Setting Up the Environment .....	7
2.2.2	Contributions .....	9
<b>3</b>	<b>Usage.....</b>	<b>9</b>
3.1	.... <i>[Quick Start]</i> .....	9
3.2	.... <i>[Saving, Loading, and Resetting]</i> .....	11
3.2.1	Saving a Configuration .....	11
3.2.2	Loading a Configuration .....	11
3.2.3	Reset a Configuration .....	12
3.3	.... <i>[Simulation Configuration Parameters]</i> .....	12
3.4	.... <i>[Run Configuration Parameters]</i> .....	14
3.5	.... <i>[Power Configuration]</i> .....	15
3.5.1	Adding a Power Configuration.....	15
3.5.2	Editing a Power Configuration.....	15
3.5.3	Deleting a Power Configuration .....	16
<b>4</b>	<b>Errors, Bug Reporting, &amp; Feature Suggestions .....</b>	<b>16</b>





## **1 Introduction**

### **1.1 Scope and Purpose**

NoximGUI provides a graphical user interface to interact with [Noxim](#), a Network-On-Chip simulation program written in SystemC and developed at the University of Catania, Italy. Some of the advantages of using this GUI include easy organization and viewing of all simulation parameters and extended convenience capabilities such as printing the results for presentation in reports. These features eliminate the need for the user to work with Noxim purely through the command line interface, as it was originally created.

This guide is intended to be used by both users of NoximGUI as well as those who plan to contribute any additional development on the product. Users are assumed to be able to read instructions, have access to an Internet connection, and have minimal knowledge of Terminal. Developers are assumed to have proficient fundamental knowledge of C++, Linux and terminal commands, and Git version control. This software guide also assumes you have a working installation of Noxim. If not, you can follow instructions to install that on the [Github Page](#). As of right now, this software has only been tested on Ubuntu Linux 16.04 (Xenial Xerus). Installing either this software or the QT Creator Studio on a different Ubuntu version or Linux flavor will be done at the user's own risk and discretion.

### **1.2 Process Overview**

For general usage, all required dependencies will come packaged with the software. However, in order to continue development, extra software will be required in order to meet the dependencies of the both the software itself as well as the libraries required by the corresponding GUI studio, QT.

Following download, a user can simply follow the instructions to properly configure the software and then build the packages for their system. A developer can continue on from there to install all required packages and libraries for development, including the QT Community edition for open-source software. All instructions are outlined in this guide.

---

## 2 Installation

The *Installation* section is divided into two sections for two distinct purposes: the first is for general usage, such as research purposes as well as for school and/or university classes. The second is for development, such as contributions to the project and/or modifications for a specific purpose.

- **NOTE: You will need user write permissions for the directory you install NoximGUI in.** If you install this in a write-protected directory, you will need to start the program with elevated privileges, and this will likely introduce complications on public systems. Best bet is to install in your home directory
- **NOTE: The terms “command line” and “terminal” will be used interchangeably.**
- **NOTE: As of version 1.0.0, this software is released as statically compiled for Ubuntu 16.04 Xenial Xerus, 64-bit ONLY. Any attempts to run this software on other Linux flavors, operating systems, or the 32-bit version are done at your own risk.**

### 2.1 [For General Usage]

1. First, install *git* in your Ubuntu system. Fetching and pulling the repositories on the command line is much easier than downloading and unzipping the files.

```
sudo apt-get install git
```

2. Then, install Noxim. Go to the [Noxim Github Page](#) and follow the instructions in the README. There should be a snippet displaying a bash command, and this will automatically install Noxim and all of its dependencies. Copy that command and execute it in terminal. NoximGUI does not require any additional dependencies on top of these.

```
bash <(wget -qO- --no-check-certificate  
https://raw.githubusercontent.com/davidepatti/noxim/master/other/setup/ubuntu.sh)
```

3. Then, navigate to [Michael's home page](#) and scroll down to the 'Portfolio' section. One of the modules should be named *NoximGUI*. Click on it, scroll down to the bottom, and download the latest version available. As of this document's current revision, the latest is 1.0.0.

- a. Note: These are statically-compiled packages. All other required libraries not installed by Noxim will be built into the program.

4. Open the newly downloaded file with your favorite archive manager, or un-tar the file on the command line (replace X.X.X with the respective version):

```
mkdir NoximGUI  
tar -xzf noximgui-X.X.X-linux.ubuntu.16.04.tgz -C NoximGUI
```

5. The executable is located in the newly-extracted directory. You may immediately execute it with the following command, and proceed to the Usage section of this guide:

```
./NoximGUI
```

[WARNING: These steps have been tested to be working on a clean VirtualBox VM running Ubuntu 16.04, 64-bit. No guarantees can be made for other Linux flavors or operating systems at this time.]

## 2.2 [For Development]

 **BEFORE YOU BEGIN: Follow steps #1 and #2 for General Usage installation above to install Git and Noxim, then proceed to the steps below.**

### 2.2.1 Setting Up the Environment

1. Install Qt libraries and IDE.
  - a) Go to [Qt's Home Page](#) and click on the person icon at the top right corner. This will redirect to a sign-in page. Scroll down and click on 'Create Qt Account'. Fill in the information, accept the terms, then verify your account.
  - b) Go back to the main Qt home page, and then click on the green 'Download Qt' button. Answer the short survey asking about usage purposes until you get to the 'Download Now' button. Then download it.
    - a. If confused on the survey questions, choose the following answers:
      - i. *In-house deployment, private use, or student use*
      - ii. *Yes*
      - iii. *No, it's not a concern -or- Not sure*
      - iv. *Dynamically -or- Not sure*
      - v. *Yes*
  - c) Then, you'll have to modify the permissions of this downloaded file and make it executable before you can execute it. Open a terminal and do the following:

```
cd ~/path/to/download/location/  
chmod +x qt-unified-linux-x64-2.0.4-online.run  
./qt-unified-linux-x64-2.0.4-online.run # OR double-click in Files
```
  - d) It will ask you for login information. Fill in your sign-in information, or if you didn't sign up before, create the account.
  - e) Select a location for the Qt installation. Two common locations are in `/home/YOUR_USERNAME` and `/opt/`.
  - f) On the next screen, select 'Qt 5.4'. If you're feeling adventurous, you may attempt to install later versions of Qt and test to see if this software works. But this is untested on newer versions.
  - g) Click 'Agree', and then begin the installation. There should be approximately 2 GB of files to be installed, and depending on the speed of your system, this may take a little while.
2. After installing Qt, you will need to install *yaml-cpp*, a C++ YAML parser written by [Jesse Beder](#).
  - a. First, clone the repository. You will not need to install any additional dependencies because Cmake 3.5.1 is distributed with Ubuntu 16.04 by default.

```
git clone https://github.com/jbeder/yaml-cpp
```



- b. Then, change into the new directory, create a directory called 'build', change into it, then run Cmake with the parent directory as an argument. Finally, make the packages\*\*:

```
cd yaml-cpp
mkdir build
cd build
cmake ../
make
```

\*\*Note that there is now a file called *libyaml-cpp.a* in the *build* directory after *make*.

- c. **NOTE: Follow the next set of instructions very carefully.** You will need to adjust the 'include' headers since there seems to be an issue with their paths. Change directories into the 'include/yaml-cpp' folder to find all of the \*.h files.

```
cd ../include/yaml-cpp # Assuming you are still in build directory
```

- d. Execute the following commands EXACTLY AS WRITTEN from the 'include/yaml-cpp' folder (you may choose to copy + paste these commands):

```
sed -i 's/yaml-cpp\\\\\\\\g' *.h
cd contrib
sed -i 's/yaml-cpp\\\\contrib\\\\\\\\g' *.h
sed -i 's/yaml-cpp\\\\\\\\.\\\\\\\\g'
cd ../node
sed -i 's/yaml-cpp\\\\node\\\\\\\\g' *.h
sed -i 's/yaml-cpp\\\\\\\\.\\\\\\\\g'
cd detail
sed -i 's/yaml-cpp\\\\node\\\\detail\\\\\\\\g' *.h
sed -i 's/yaml-cpp\\\\node\\\\\\\\.\\\\\\\\g' *.h
sed -i 's/yaml-cpp\\\\\\\\.\\\\\\\\.\\\\\\\\g'
```

3. Install the proper OpenGL libraries (**NOTE: The following command assumes you are developing on a Ubuntu 16.04 Virtual Machine.** You will need to download different GL libraries based on your graphics processor or card. More information can be found here in [chaos's Github answer](#):

```
sudo apt-get install libgl1-mesa-dev mesa-common-dev
```

4. Now, it's time to clone the source code of the NoximGUI project. Go back to your home directory or wherever you want to install this, and run:

```
git clone https://github.com/mvxt/noximgui
```

5. Open Qt Creator and then select to open a project. Navigate to the cloned source code folder, and Qt Creator should recognize the project file.
6. Now, inside Qt Creator, open the NoximGUI.pro project file, and delete lines 30 – 54. You will need to link to the location of your own *yaml-cpp* library.

- a. To link to your *yaml-cpp*, right-click on the project in the folder view (labeled 'NoximGUI [master]') and select 'Add Library'. On the first menu, select 'External Library' and then click Next.
- b. In the windows, you will be able to select a 'Library' file and an 'Include' directory. Link to the following inside your *yaml-cpp* installation from earlier:

- i. *Library: /path/to/yaml/cpp/build/libyaml-cpp.a*





- 
- ii. *Include: /path/to/yaml/cpp/include/yaml-cpp*
    - c. Then click OK. This should automatically fill in the appropriate lines in your *NoximGUI.pro* file, and now your project should compile.
  7. Now, go to the 'Build' menu at the top and then click 'Run qmake.' This should build the necessary recipes and configurations for Qt. Then, click 'Run'. Qt will detect that you have made some changes to files, and it will automatically build the application before running it. By default, the application will deploy the build in a new folder in your home directory, usually labeled '*build-NoximGUI-....*'. Do not make any changes to that folder, any subsequent builds will overwrite changes in that folder. **Always develop and make changes in the original-cloned noximgui folder.**
  8. Ensure the software runs as built and executed by the Qt Creator.

## 2.2.2 Contributions

When making changes and contributions to the repository, always follow this procedure:

1. First, create a new branch with the following naming convention:
  - a. *Feature/NAME* : If you are introducing new capability that is not currently present, begin the new branch name with tag 'Feature' followed by a forward slash and any name you give your feature. For example, if you are introducing an automation feature, you may create a branch called '*Feature/Automation*'.
  - b. *Refactor/NAME* : If you are rewriting certain portions of the code without necessarily changing its function or purpose, name the new branch with tag 'Refactor' followed by a forward slash and any identifier for the current item you are refactoring. For example, if you are refactoring the way configurations are managed, you may name your branch '*Refactor/Configurations*'.
  - c. *Bug/Issue\_NUMBER* : If you are fixing a bug and/or addressing any kind of issue reported on Github, please name the branch with tag 'Bug' followed by 'Issue', an underscore, and the Issue # (as assigned by Github). For example, if you are fixing the first reported bug, the branch name will be '*Bug/Issue\_1*'.
2. Make any changes you want to make to the code. Ensure that the code is well-documented for documentation generation for other developers. **NOTE: If you are making contributions to the project, you *\*\*must\*\** have documentation-style comments above your methods and other comments explaining your code in order for the pull request to be accepted. Undocumented code will be rejected always.**
3. Along with any changes you make, create corresponding
4. Finally, when you have finished, make a pull request on the Github page for the NoximGUI project. Administrator(s) will review the changes you've made and

## 3 Usage

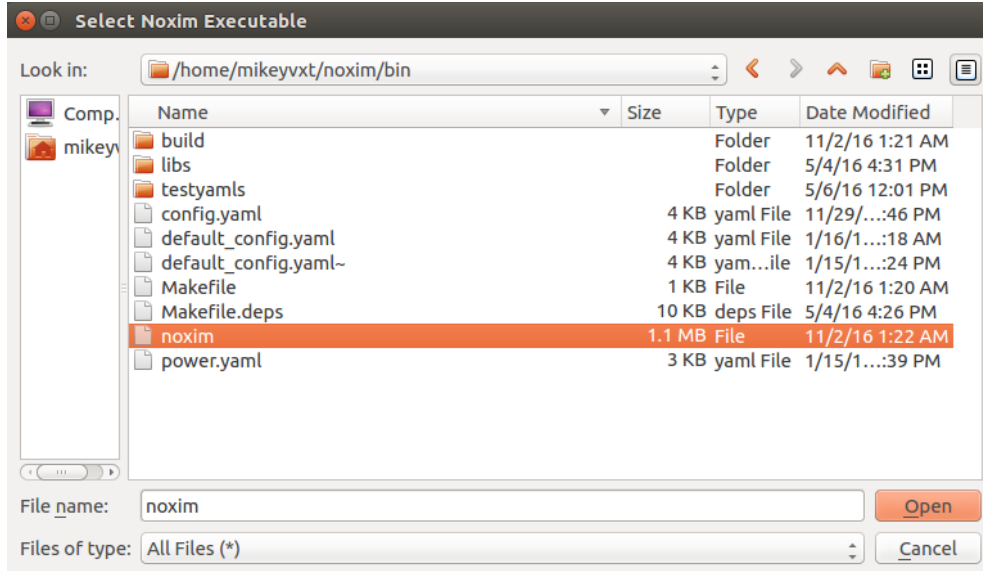
### 3.1 [Quick Start]


1. To start NoximGUI, run the executable via a command line:

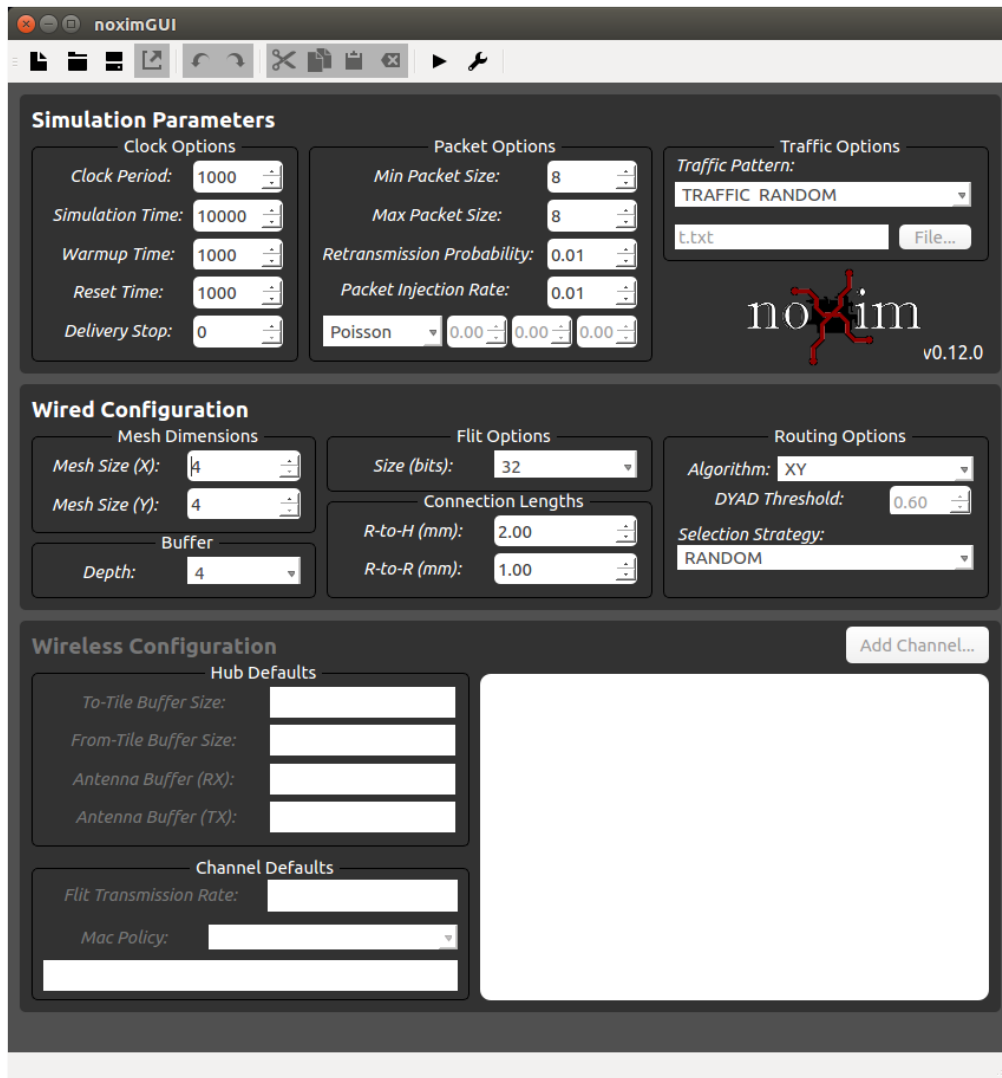


./NoximGUI

2. Upon first starting the program, it will ask you to locate the executable file of Noxim. Navigate to it in the file explorer and then select it. As a note, the file should be called *noxim*, and it should be inside of the *bin* directory of the noxim folder. See the below picture for an example.




3. Once you've selected the Noxim executable, it should show the splash screen, and then the program should promptly load.
  - a. NOTE: NoximGUI will always open with the default simulation parameters and default power configuration. If you want to run the simulation with custom parameters and custom power, you must subsequently load those files. Instructions are below.
4. Following this, you may immediately begin changing parameters and running simulations. Please note that as of this time, Wireless Capability has not yet been implemented, and subsequently that module is disabled. To run a simulation, click the  button at the top toolbar, or use the keyboard shortcut Ctrl+R.
  - a. **NOTE: If you have not saved or loaded a custom configuration, then the current configuration will not be persisted. If you wish to access this particular configuration again, you must save it to a file. This is covered in [Section 3.2.1](#).**
5. Once the simulation runs, it will print the output of the simulation to a pop-up window. If you would like to save these results to a file, you may click the 'Print' button, which will then open another dialog for saving.




## 3.2 [Saving, Loading, and Resetting]

By default, NoximGUI will always start with the default configuration parameters (i.e., what was shipped with the original Noxim software's "default\_config" file.

### 3.2.1 Saving a Configuration

After changes are made, you are able to save your changed/custom configuration by clicking on the square icon of a floppy disk , 3<sup>rd</sup> from the left at the top toolbar. Set a file name, and then click 'Save'. You will then be able to load this custom file at any time, during any session. This feature is a good way to store and keep track of different configurations, for different purposes, named according to your preferences and organization.


### 3.2.2 Loading a Configuration

To load a custom configuration, click the icon of a folder  in the top menu bar. Select the file you wish to load, and then click 'OK.' It should be noted that as of the latest version (1.0.0), loading a custom configuration does not currently validate the file you attempt to

---

load. If there are somehow syntax errors, or if the file is sabotaged and/or corrupted, the program will crash. Features to introduce graceful warnings and exit will be introduced later.

### 3.2.3 Reset a Configuration

To reset the entire configuration to the default values at any given time, click the icon of a blank sheet . This will automatically change the values shown on the screen, and the next time a user runs a simulation, the new, default values will be used for the simulation.

## 3.3 [Simulation Configuration Parameters]

In the main window, there are a number of configuration parameters you may adjust and choose from. They are explained in order (up to down, left to right) below:

- *Clock Period (ps)*
  - The length of time of a single clock cycle in picoseconds
- *Simulation Time (cycles)*
  - The total number of cycles the simulation should run for
  - **Note: This value must be larger than the Warmup Time**
  - **Note: This value overrides *Delivery Stop* when less than *Delivery Stop***
  - **Note: This value is overridden by *Delivery Stop* when greater than *Delivery Stop***
- *Warmup Time (cycles)*
  - The number of cycles to run before beginning to collect statistics
  - **Note: This value must be less than the total Simulation Time**
- *Reset Time (cycles)*
  - The number of cycles to send a reset signal before beginning simulation
- *Delivery Stop (flits)*
  - The number of flits to be collected before simulation stops
  - **Note: This value is overridden by *Simulation Time* - if the Simulation Time has been reached, the simulation will stop regardless of whether the flit quota has been met**
  - **Note: This value overrides *Simulation Time* when it is reached before the Simulation Time has elapsed**
- *Min. Packet Size (flits)*
  - Packet sizes are randomly generated with a random number generator – the minimum packet size dictates the minimum size any given packet will be
- *Max Packet Size (flits)*
  - The maximum packet size any given packet will be
  - **Note: To set a single packet size, set *Min Packet Size* = *Max Packet Size***
- *Retransmission Probability [0..1]*
  - A real number between 0 and 1 inclusive dictating likelihood a packet will retransmit if dropped




- 
- *Packet Injection Ratio (0..1)*
    - How many packets are inserted into the network stream per clock cycle, greater than 0 and less than or equal to 1
    - The corresponding dropdown box below designates the types of traffic distribution:
      - **Poisson**
        - Poisson distribution, memoryless (default)
      - **Burst**
        - Burst distribution with given real burstness  $R$
      - **Pareto**
        - Self-similar Pareto distribution with given parameters  $Alfa-On$ ,  $Alfa-Off$ , and  $R$
      - **Custom**
        - Custom distribution with given real probability of retransmission
  - *Traffic Pattern*
    - Spatial distribution of traffic
    - The corresponding *File* and *Textbox* below are for selecting a text file of table routing information, enabled only for *TRAFFIC\_TABLE\_BASED*
  - *Mesh Size (X)*
    - Number of nodes in the X dimension (currently minimum value of 4)
  - *Mesh Size (Y)*
    - Number of nodes in the Y dimension (currently minimum value of 4)
  - *Depth (flits)*
    - Buffer size of each channel/node in flits
  - *Size (bits)*
    - Size of each flit in bits
  - *R-to-H (mm)*
    - Router-to-Hub connection size in millimeters
  - *R-to-R (mm)*
    - Router-to-Router connection size in millimeters
  - *Algorithm*
    - Routing algorithm used by the channels and nodes
    - *DYAD Threshold (0..1)*
      - Real threshold of DYAD-specific routing
  - *Selection Strategy*
    - The selection strategy for packet forwarding

---

It should be noted some of these parameters such as Depth, Size, R-to-H, and R-to-R are limited to exactly what power values are configured in the Power Configuration, covered in [Section 3.5](#). Subsequently, if there are sparse options or if there are certain values unavailable, you will need to specify their corresponding power values in the Power Configuration.

### 3.4 [Run Configuration Parameters]

Run configuration parameters are options not directly related to the topology or other parameters of a network, but rather details of the simulation itself. They can be found by clicking on the icon of a wrench in the top toolbar . Upon clicking the button, the Run Configurations dialog window will pop up, giving you the following options:

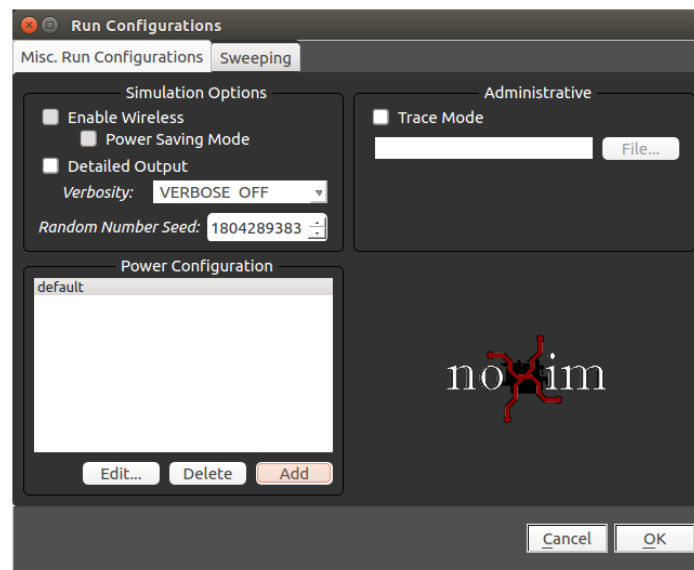
- *Enable Wireless (currently unavailable)*
  - This enables the wireless widget shown on the main window to edit hub configurations and other wireless parameters. This option is currently unavailable and unimplemented (as of version 1.0.0)
- *Power Saving Mode (currently unavailable)*
  - This option is to enable an experimental power-saving mode for wireless configuration. This option is current unavailable (as of version 1.0.0)
- *Detailed Output*
  - Checking this option prints out verbose details of trace signals and other items on the output window
  - **Note: This option is essentially the 'Verbose' option for people familiar with command-line software**
- *Verbose*
  - It is currently unknown what this option actually does (see [issue #16](#) on Github)
- *Random Number Seed*
  - A large, real, and integer number for seeding the random number generator. The upper limit will be the maximum size of the *INTEGER* variable of a given operating system
- *Trace Mode*
  - Turns on tracing of signals to a *.vcd* file. Enabling it subsequently enables the *File* button and the *TextBox* below for setting the file to trace the signals to
  - **Note: VCD visualization is not included with this program - you will need to download and install your own separate VCD visualizer to view those files**



### 3.5 [Power Configuration]

The power configuration is a list of different power configuration files. These power configuration files are key to Noxim's simulations, as they specify static values of energy expenditure for certain buffer sizes, nodes, connection lengths, and a number of other parameters. There are some parameters (mentioned at the end of [Section 3.3](#)) that are dependent upon the currently selected power configuration and its values. When a specific power configuration is selected and the user clicks 'OK', the parser reads the configuration and corresponding adds or removes available configuration parameters from the main screen.

When first starting the program, only a single power configuration is available: the default. You cannot edit or delete the default configuration. To make modifications, you should create a new, custom power configuration.



#### 3.5.1 Adding a Power Configuration

To add a new power configuration, click the 'Add' button below the ListBox. This will create a new item in the list, and you may click on the list item to adjust the new power configuration's name accordingly. By default, when creating a new power configuration, it will simply copy the *default*. To change the values, you must edit the new configuration.

#### 3.5.2 Editing a Power Configuration

To edit a power configuration, select the configuration you wish to edit and click the 'Edit' button. This will open your configuration in your system's default text editor. Your system's default text editor can be changed in system settings, which falls outside of the jurisdiction of this program.

When the power configuration is opened, you will see that it contains the same default values as the *default* power configuration to serve as a template. You may make modifications to this file and save it. **Note: as of version 1.0.0, there is no syntax parsing available for the power configurations. If you introduce syntax errors or other issues to a power configuration, the program will encounter a bug when you click 'OK' and**

---

pop a warning dialog. Simulations may not work at this point, so you will need to go back in and fix the error or select a different configuration.

### 3.5.3 Deleting a Power Configuration

To delete a power configuration, selection the configuration's name in the list box and then click 'Delete'. **Note: this will permanently delete the power configuration from the system, so there is no way to recover it. Think twice before deleting one!**

## 4 Errors, Bug Reporting, & Feature Suggestions

To submit any errors, bugs, suggestions, and questions for the program, go to the [NoximGUI Github page](#) and then click on 'Issues'. When filing a bug or error, please make sure to adhere to the following procedure:

1. Always detail the exact steps or what exact actions you were doing before the bug or error occurred. The more detailed and precise, the easier it is to recreate and subsequently fix the problem.
2. If possible, run the GNU Project Debugger (gdb) with the program, and then make it crash again. When that happens, run the *bt* and *where* commands to print the backtrace logs. Copy and paste that information into the issue.

```
gdb ./NoximGUI
run # To start the program
bt # After it crashes
where # After it crashes
```

