Rogers

BPI UI Naming Config Guide



Author: Joshua Michael

Creation Date: November 1, 2022

Last Updated: December 15, 2022

Document Ref: Rogers

Version: V1.0

# Document Control

## Change Record

5

| Date | Author | Version | Change Reference |
| --- | --- | --- | --- |
| 1-Nov-2022 | Joshua Michael | V0.5 | No Previous Document |
| 16-Dec-2022 | Joshua Michael | V1.0 | Document Updated with information from setup |
|  |  |  |  |
|  |  |  |  |

## Reviewers

| Name |
| --- |
| Joshua Michael |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

## Approvers

| Name | Date |
| --- | --- |
|  |  |
|  |  |

Contents

1 Document Control ii

1.1 Change Record ii

1.2 Reviewers ii

1.3 Approvers ii

2 Overview 1

2.1 Naming Configuration General Template 1

2.2 Shared Configuration Fields 2

3 Device Naming 4

3.1 RHP Naming 5

3.2 IPRAN Dynamic Role Naming 5

4 Logical Connection Naming 6

4.1 Example Output 6

5 Slot Position Naming 7

5.1 Example Output 7

5.2 Slot Naming in Metadata Modeler 8

6 Physical Port Naming 9

6.1 RHP Port Naming 10

7 Open and Closed Issues 11

7.1 Open Issues 11

7.2 Closed Issues 11

# Overview

This Customer Configuration Guide documents and explains the fields, methods, and operations for the Naming Configuration tab within the BluePlanet UI. The Naming Configuration tab functions to assist with the creation and modification of naming fields throughout a specific device, card, connection, or other product type that Rogers may be looking to provide. This document may be updated in the future to comply with edits or revisions made to the UI, whether that be general graphic interface adjustments or redefined function calls.

## Naming Configuration General Template

To set up custom naming for the various Roger devices, the user must go into the BluePlanet UI under the ‘Naming Configuration’ tab. Once there, the user will be presented with a series of tables for Device, Logical Connection, Physical Port, and Slot Position naming rules. Steps as to including and editing this dynamic naming will be listed throughout the various sections.

Graphical user interface, text, application, Word, email

Description automatically generated

Once here, it is possible to Add/Modify/Delete the rules in each section. To start off you can click the ‘Add Rule’ label over the respective table that you wish to make a naming rule for. From here, a row will be generated that allows the user to fill out columns to define the path and method which will decide what respective product will have its calls adjusted. What this row looks like is below.

Graphical user interface, text, application

Description automatically generated

Depending on the column, there is the option of either using defined dropdown selections, or inputting user values. Certain columns relate explicitly to one another, so if you choose an ID within the ARCHETYPE\_ID, the respective ARCHETYPE\_NAME for that device will appear in its adjacent column. After filling out all the columns, by pressing ‘Apply’ you can then save that rule into the lookup table. After this rule is saved, you can then either click the trashcan symbol to delete this rule or, you can click the pen symbol to edit the rule if it requires modification.

* Currently editing a naming rule to include new strings will run an error, you must recreate the rule and delete the one prior.

## Common Configuration Fields

Throughout the interface, there are fields which are repeated through the four different applicable tables. The common columns for all Naming Configuration tables are defined below.

|  |  |
| --- | --- |
| FIELD | DESCRIPTION |
| METAMODELID | A pre-populated and cannot be changed. It reflects a type of object which will be affected by the rule. |
| PRIORITY | An integer and intended to control the power of the rule. A higher number denotes a higher priority, so rules with a higher priority will be executed by the Naming engine if two or more rules have satisfied. |
| NAMING\_STRING | Columns need to be populated according to the design. It can contain static and dynamic elements and symbols + between the elements.  ● Static elements must be wrapped in double quotes and can contain any characters.  ● Dynamic elements must be wrapped in angled brackets and can contain only tokens from the design document. |
| Groovy Function | Used should the user wish to make a specific unique naming call rather than having to stay defined to the limits of tokens and tags that are pre-existing within the backend. The user can write out a string of Groovy Code within the column to invoke the naming on the device. Otherwise, the field should be filled with ‘No’ to follow regular tag naming. |

### Naming String Tokens

An example of these naming rules in action can be seen below.

* Dynamic Tokens are underlined in red,
* Static strings are underlined in blue.

There are multiple different types of tokens, which are defined in the backend of your personal environment.

Graphical user interface, application, table

Description automatically generated

Throughout the various sections, a table will be listed with all the available tokens that you can call for each Naming Table. These are good substitutes that you can work with in the case that the user is not comfortable or able to use the Groovy Naming customization.

# Device Naming

Device Naming Rules work by taking the information in the fields below and generating a dynamic naming system when creating equipment. The common configuration fields and the table below define the Device Rule table.

|  |  |
| --- | --- |
| FIELD | DESCRIPTION |
| DOMAIN | A selected keyword based on how the communication system works for a specific device. The currently offered domains are split between IPRAN, RHP, TRANSPORT and MICROWAVE domains. |
| ENTITY\_TYPE | An alphanumeric 3-character code issued by Telcordia indicating the role of a Device. By putting its value within this column, you can then call it within the naming string using the dynamic token - <ENTITYTYPE> |
| ARCHETYPE\_ID  ARCHETYPE\_NAME | Column ARCHETYPE\_ID is a dropdown which holds the archetype IDs for all the deployed devices. You can either select through a search to find a device based on its ID, or with Column ARCHETYPE\_NAME you can find a Device by its name. Selecting an archetype will fill in the correlating archetype for the other. |
| LOCATION\_ARCHETYPE\_ID  LOCATION\_ARCHETYPE\_NAME | Column LOCATION\_ARCHETYPE\_ID is a dropdown which holds the archetype IDs for all the existing location types. You can either select through a search to find a location type based on its ID, or with Column LOCATION\_ARCHETYPE\_NAME you can look up either building, country, room, floor, etc. Selecting an archetype will fill in the correlating archetype for the other. |
| DEVICE\_NAME\_NAMING\_STRING | Used to auto-populate the ‘Device Name’ field when creating an equipment of the specified archetype from prior columns. |
| DEVICE\_TID\_NAMING\_STRING | Used to auto-populate the ‘Device TID’ field when creating an equipment of the specified archetype from prior columns. |
| START\_SEQUENCE | The field which determines where the iterative naming would start at. Whether at 0, 1, or any other respective integer. |

Altogether the exact table from left to right generally follows the order defined below.

|  |
| --- |
| DEVICE RULE TABLE |
| [ METAMODELID / DOMAIN / ENTITY\_TYPE / ARCHETYPE\_ID / ARCHETYPE\_NAME / LOCATION\_ARCHETYPE\_ID / LOCATION\_ARCHETYPE\_NAME / DEVICE\_NAME\_NAMING\_STRING / DEVICE\_TID\_NAMING\_STRING / GROOVY\_FUNCTION / START\_SEQUENCE / PRIORITY ] |

## RHP Naming

For the RHP Devices, the device naming strings can be defined down below with their respective tokens

|  |  |
| --- | --- |
| FIELD | DESCRIPTION |
| DEVICE\_NAME\_NAMING\_STRING | <SITELOCATIONCLLI>+<ENTITYTYPE>+"."+<DEVICENAMECOUNT>   * Here <SITELOCATIONCLLI> is the unique 8-digit CLLI identifier of a building while <DEVICENAMECOUNT> is a 3-digit sequence number guaranteeing the uniqueness of the Device Name in the archetype location |
| DEVICE\_TID\_NAMING\_STRING | "(x)-"+<LOCATIONCODE>+"-"+<DEVICETIDCOUNT>   * Here <LOCATIONCODE> is the unique 5-digit identifier of a location archetype such as a building/room, where the (x) portion is a placeholder. Inside that placeholder you would put the device specific acronym. |

Graphical user interface, application

Description automatically generated

Alternatively, one can use Groovy code to display the naming rules. You’re able to make more specific naming rules in exchange for having a rudimentary understanding of the source code for the BPI naming. This allows a bit more customization in orders. The Groovy function to configure for the RHP devices can be seen below for the G9.

|  |  |
| --- | --- |
| FIELD | DESCRIPTION |
| GROOVY\_FUNCTION | class RHPDevice { void deviceName(deviceVO, namingDao, deviceRole, sessionContext) { if (deviceVO.getSITELOCATIONCLLI() != null && deviceVO.getENTITYTYPE() != null && deviceVO.getLOCATIONCODE() != null) { String deviceName = deviceVO.getSITELOCATIONCLLI() + deviceVO.getENTITYTYPE(); Long deviceCount = namingDao.fetchIPRANDeviceCount(deviceName, deviceVO.getLOCATIONCODE()) + deviceVO.getSequence(); if (deviceCount != null) { deviceVO.setDeviceName(deviceName + '.' + (deviceCount + deviceVO.getSequence())); deviceVO.setDeviceTID( "G9-" + deviceVO.getLOCATIONCODE() + '-' + (deviceCount )); } } } }   * For the Device TID snippet at the end, one can easily edit the string to label the proper device TID header. Whether G6, or C20, etc. |

This snippet of code will produce a Device Properties tab which looks akin to the image on the following page.

Graphical user interface, text, application, email

Description automatically generated

## IPRAN Dynamic Role Naming

For the IPRAN device names, naming works slightly different. Rather than defining a static name at the start of the Device creation process, the name should dynamically change dependent on which ‘Role’ is selected. Depending on the role, the entity code, which is displayed within the Device Name, can change. Therefore, the naming requires a dynamic lookup table which can be called from, alongside a custom groovy script. The information for such and a respective output can be seen below.

|  |  |
| --- | --- |
| FIELD | DESCRIPTION |
| GROOVY\_FUNCTION | class IPRANDevice { void deviceName(deviceVO, namingDao, deviceRole, sessionContext) { if(deviceVO.getSITELOCATIONCLLI() != null && deviceRole != null && deviceVO.getLOCATIONCODE() != null) { String[] deviceRoles = deviceRole.split(":"); deviceVO = namingDao.fetchEntityCodeForIPRANDevice(deviceRoles[0], deviceVO, sessionContext); String deviceName = deviceVO.getSITELOCATIONCLLI() + deviceVO.getEntityCode() + "-" + deviceVO.getLOCATIONCODE(); Long deviceCount = namingDao.fetchIPRANDeviceCount(deviceName, deviceVO.getLOCATIONCODE()) + deviceVO.getSequence(); if(deviceCount != null) { deviceVO.setDeviceName(deviceName + "-" + String.format("%02d", deviceCount)); } String deviceTID = deviceRole + "-" + deviceVO.getLOCATIONCODE() + "-"; Long deviceTIDCount = namingDao.fetchIPRANDeviceTIDCount(deviceTID, deviceVO.getLOCATIONCODE(), deviceVO.getLocationEmgCode(), sessionContext); deviceVO.setDeviceTID(deviceTID + String.format("%02d", deviceTIDCount));}}} |

Graphical user interface, text, application, email, website

Description automatically generated

In this example, the role ‘IPRAN’ needs to be defined first, because this is a check that is defined within the Groovy function. From here, depending on the role chosen next (A2, CSD1, S1, etc), the appropriate entity code will display itself and autofill the information for Host Name, Device Name, CLLI Name, and Device TID. To change the entity code after having set up these steps, you must cancel out of both the IPRAN and A2 role and reselect them in the respective order.

## Device Tokens

If a user does not want to use Groovy or any of the specified naming up above, the below tokens can be put within the Naming String field in order a bit more customization.

|  |
| --- |
| DEVICE RULE TABLE |
| [ "DEVICEARCHETYPENAME", "DEVICEARCHETYPEABBREV", "DEVICELOCATIONNAME", "DEVICELOCATIONSITEID", "LOCATIONCODE", "SITELOCATIONCLLI", "ENTITYTYPE", "DEVICENAMECOUNT", "DEVICETIDCOUNT"] |

# Logical Connection Naming

Logical Connection Naming Rules work by taking the information in the fields below and generating a dynamic naming system when creating equipment. The common configuration fields and the table below define the Logical Connection Rule table.

|  |  |
| --- | --- |
| FIELD | DESCRIPTION |
| ARCHETYPE\_ID  ARCHETYPE\_NAME | Column ARCHETYPE\_ID is a dropdown which holds the archetype IDs for all the deployed logical connections. You can either select through a search to find a connection based on its ID, or with Column ARCHETYPE\_NAME you can find it by its name. Selecting a specific archetype will fill in the correlating archetype for the other. |
| SIGNAL\_TYPE\_ID  SIGNAL\_TYPE\_NAME | Column SIGNAL\_TYPE\_ID is a dropdown which holds the archetype IDs for all the existing deployed logical signals. You can either select through a search to find a connection based on its ID, or with Column SIGNAL\_TYPE\_NAME |
| FACILITY\_TYPE | An alphanumeric character code relating to the facility a signal exists in. By putting its value within this column, you can then call it with the naming string by using the token - <FACILITYTYPE> |
| NAMING\_STRING | Used to auto fill the Logical Connection fields when creating equipment. |

Altogether the exact table from left to right generally follows the order defined below.

|  |
| --- |
| DEVICE TOKEN TABLE |
| [ METAMODELID / ARCHETYPE\_ID / ARCHETYPE\_NAME / SIGNAL\_TYPE\_ID / SIGNAL\_TYPE\_NAME / FACILITY\_TYPE / GROOVY\_FUNCTION / NAMING\_STRING / PRIORITY ] |

## Connection Naming Configuration

For the RHP and IPRAN Connections, the device naming strings can be defined down below with their respective tokens after having chosen the Signal Type and Channelization that you wish to make a rule for.

|  |  |
| --- | --- |
| FIELD | DESCRIPTION |
| NAMING\_STRING | <LOGICALCONNECTIONSEQUENCENUMBER> + "/" + <FACILITYTPE> + ”/” + <LOGICALCONNECTIONDEVICEANAME> + "/" + <LOGICALCONNECTIONDEVICEZNAME>   * In this example, <LOGICALCONNECTIONSEQUENCENUMBER> is what number in the inventory the Connection is. <FACILITYTPE> relates to the FACILITY\_TYPE field which is defined in the Naming Rule itself, and the remaining two tokens refer to the 4-digit identification notes in the Source and Target locations for the Connection. |

Graphical user interface, text, application, email

Description automatically generatedGraphical user interface

Description automatically generated

## Logical Connection Tokens

If a user does not want to use Groovy or any of the specified naming up above, the below tokens can be put within the Naming String field in order a bit more customization.

|  |
| --- |
| LOGICAL CONNECTION TOKEN TABLE |
| ["LOGICALCONNECTIONDEVICEANAME", "LOGICALCONNECTIONDEVICEZNAME", "LOGICALCONNECTIONUSEDESIGNATION", “LOGICALCONNECTIONCOUNT","LOGICALCONNECTIONSEQUENCENUMBER", "LOGICALCONNECTIONBANDWIDTHCODE", "LOGICALCONNECTIONARCHETYPENAME", "LOGICALCONNECTIONCOUNTPADDED", "CUSTOMER","CUSTOMERSEQUENCE","FACILITYTYPE" ] |

# Slot Position Naming

Slot Position Naming Rules work by taking the information in the fields below and generating a dynamic naming system when creating equipment. The common configuration fields and the table below define the Slot Position Rule table.

|  |  |
| --- | --- |
| FIELD | DESCRIPTION |
| DEVICE\_ARCHETYPE\_ID  DEVICE\_ARCHETYPE\_NAME | Column DEVICE\_ARCHETYPE\_ID is a dropdown which holds the archetype IDs for all the deployed devices. You can either select through a search to find a device based on its ID, or with Column DEVICE\_ARCHETYPE\_NAME you can find a specified Device by its name. Selecting a specific archetype will fill in the correlating archetype for the other. |
| PARENT\_CARD\_ARCHETYPE\_ID  PARENT\_CARD\_ARCHETYPE\_NAME | Column PARENT\_CARD\_ARCHETYPE\_ID is a dropdown which holds the archetype IDs for all the deployed cards. You can either select through a search to find a device based on its ID, or with Column PARENT\_CARD\_ARCHETYPE\_NAME you can find a specified card by its name. Selecting a specific archetype will fill in the correlating archetype for the other. |
| NAMING\_STRING | Used to auto fill the Slot fields in a device when creating equipment. |

* For a slot position rule, you can only have either the parent card archetypes selected, or the device archetypes. You cannot have both, otherwise the rules will not work as intended.

Altogether the exact table from left to right generally follows the order defined below.

|  |
| --- |
| DEVICE RULE TABLE |
| [ METAMODELID / DEVICE\_ARCHETYPE\_ID / DEVICE\_ARCHETYPE\_NAME / PARENT\_CARD\_ARCHETYPE\_ID / PARENT\_CARD\_ARCHETYPE\_NAME / NAMING\_STRING / Groovy Function / PRIORITY ] |

## Slot Naming in Metadata Modeler

Within RHP and IPRAN Devices, since the metadata modeler has specified instance archetypes for slots, it allows for ease in naming static elements rather than having to go through making specific rules. Given that slots are not dynamic throughout RHP devices, naming could be done either within the Naming Configuration or by the modeler.

To adjust a slot position’s name, you can go to your respective equipment and find the slot positions. By default, the Archetype Instance Name will go ‘Slot 1, Slot 2, … etc.’ for however many slot types you have in an iterative process down the metadata model.

Table

Description automatically generated

From here, if you wish to adjust an Archetype Instance Name, you click on the respective metadata node’s field for the Instance Name, and you’ll notice a value come up at the top of the Metadata Modeler. Like the tokens for the Naming Configuration tab, you can adjust this value by either having a static string surrounded by single quotes to create a non-dynamic instance string, or you can use operations and the ‘n’ variable that is created to have any respective integers change alongside the creation of new slots. An example which would output ‘Slot 1’ within the Ribbon G9 device is below.

A screenshot of a computer

Description automatically generated

## Slot Position Configuration

If the user decides not to use the Metadata modeler, or the slot positions require more dynamic name changes outside of the Archetype instances, Groovy naming is an option. As used in an IPRAN device, an example of slot naming can be seen in the code below. This code would override the archetype instances within the backend and instead take the function defined in the Slot Naming rule.

|  |  |
| --- | --- |
| FIELD | DESCRIPTION |
| GROOVY\_FUNCTION | class SlotNaming {  String slotName(slotVO) {  return (slotVO.getShelfSlotNumber() + 1) + "/" +(slotVO.getSlotNumber() + 1);  } } |

## Slot Position Tokens

If a user does not want to use Groovy or any of the specified naming up above, the below tokens can be put within the Naming String field in order a bit more customization.

|  |
| --- |
| SLOT POSITION TOKEN TABLE |
| [ "CISCOGENERIC", "CISCOGENERIC10GB", "NODEDEFSLOTLIST", "ASR-920-12SZ-IM\_SLOTNAME", "SLOTNAME", "CARDSLOTNAME", "SHELFSLOTNUMBER", "ASRSLOTNAME", "ASRNEWSLOTNAME", "SHELFNUMBER", "SLOTNUMBER", "SHELFSLOTNAME", "3800XSLOTNAME", "SLOTNUMBER\_PLUS1", "SLOTNUMBER\_PLUS2", "SLOTNUMBER+3", "ASR1006SLOT", "SLOTNUMBER-1", "TRANSPORTSLOTFNAME", "OM5200SLOT", "FLX4300SLOT", "CISCO-DCM-9900\_SLOTNAME", "CISCO-DCM-9902\_SLOTNAME", "Evertz-3080IPX-16-10G\_SLOTNAME", "Evertz-3080IPX-32-10G\_SLOTNAME", "Evertz-3080IPX-64-10G\_SLOTNAME", "Harmonics-CP6000-Slot\_SLOTNAME", "Evertz-7880IPG-NAT-Engine\_SLOTNAME" , "FLX4100SLOT", "FLX7120SLOT", "FLX9500SLOT", "FLX600", "FLX150SLOT", "OM3500SLOT", "OC192DXSLOT", "OC48SDMSSLOT", "1648SMSLOT", "FLX2400SLOT", "FLX4500SLOT", "CH2016BSLOT", "CERAGONIPSLOT", "CERAGONIP10GSLOT", "CERAGONIP20ASLOT", "ERICIP6CSLOT", "ERICIP20CSLOT", "ERICI2PBSLOT", "OMX", "ALU1830SLOT", "OME6500SLOT","CPLSLOT", "ACCESSGENERICSLOTNAME" ] |

# Physical Port Naming

Physical Port Naming Rules work by taking the information in the fields below and generating a dynamic naming system when creating equipment. The common configuration fields and the table below define the Physical Port Rule table.

|  |  |
| --- | --- |
| FIELD | DESCRIPTION |
| DEVICE\_ARCHETYPE\_ID  DEVICE\_ARCHETYPE\_NAME | Column DEVICE\_ARCHETYPE\_ID is a dropdown which holds the archetype IDs for all the deployed devices. You can either select through a search to find a device based on its ID, or with Column DEVICE\_ARCHETYPE\_NAME you can find a specified Device by its name. Selecting a specific archetype will fill in the correlating archetype for the other. |
| PARENT\_CARD\_ARCHETYPE\_ID  PARENT\_CARD\_ARCHETYPE\_NAME | Column PARENT\_CARD\_ARCHETYPE\_ID is a dropdown which holds the archetype IDs for all the deployed cards. You can either select through a search to find a device based on its ID, or with Column PARENT\_CARD\_ARCHETYPE\_NAME you can find a specified card by its name. Selecting a specific archetype will fill in the correlating archetype for the other. |
| PLUGGABLE\_ARCHETYPE\_ID  PLUGGABLE\_ARCHETYPE\_NAME  PORT\_ARCHETYPE\_ID  PORT\_ARCHETYPE\_NAME | The logic defined in the two rows above, follows for the Columns PLUGGABLE\_ARCHETYPE\_ID and PLUGGABLE\_ARCHETYPE\_NAME for defining and choosing a specific SFP pluggable. It also follows for the Column PORT\_ARCHETYPE\_ID and Column PORT\_ARCHETYPE\_NAME for physical ports themselves. |
| NAMING\_STRING | Used to auto fill the Physical Port fields in a device when creating equipment. |

* Unlike slot positions, within physical port naming you can have multiple of the archetype duos picked out. This is done to limit down and specify the relationships between device, cards, and possibly SFPs when prompting the values which will go into the Physical Port fields.

Altogether the exact table from left to right generally follows the order defined below.

|  |
| --- |
| DEVICE RULE TABLE |
| [ METAMODELID / DEVICE\_ARCHETYPE\_ID / DEVICE\_ARCHETYPE\_NAME / PARENT\_CARD\_ARCHETYPE\_ID / PARENT\_CARD\_ARCHETYPE\_NAME / PLUGGABLE\_ARCHETYPE\_ID / PLUGGABLE\_ARCHETYPE\_NAME / PORT\_ARCHETYPE\_NAME / PORT\_ARCHETYPE\_ID / NAMING\_STRING / Groovy Function / PRIORITY ] |

## Port Naming Configuration

For Port Naming, the Groovy Functions need to be used due to the lack of token support in the backend. However, with the customization involved it allows for a lot more ease in customization for the respective devices. Given each device for the IPRAN and RHP tends to have a different way of formatting their Physical Ports, whether it’s extra dividers for that of shelf number, slot number, port number, etc.

An example below can be seen for a Gigabit Ethernet port on the ASR-903 device in IPRAN. Upon opening specific pluggables you can see under the ‘Port’ section of the device there is the newly entered ‘Gi1/(x)’ section that would go throughout the many slots in that device.

|  |  |
| --- | --- |
| FIELD | DESCRIPTION |
| GROOVY\_FUNCTION | class PortNaming {  String portName(portVO) {  return "Gi" + (portVO.getShelfSlotNumber() + 1) + "/" + (portVO.getCardSlotNumber() + 1);  } } |

Graphical user interface, application

Description automatically generated

Whilst doing the Port Naming specifically, it’s important to realize which fields to choose for your Naming Rule. If you just define a rule with the only constraints being ‘Device Name’ and ‘Port Name’, it’ll go and make the naming rule of ‘Gi1/(x)’ for all ports of a specific type, no matter the card. This can help cut down on the number of Port Naming rules if a large portion of them will end up being formatted in the same general way.

## Physical Port Tokens

If a user does not want to use Groovy or any of the specified naming up above, the below tokens can be put within the Naming String field in order a bit more customization.

|  |
| --- |
| PHYSICAL PORT TOKEN TABLE |
| [ "CISCO\_DEVICEPORTNAME", "CISCOGENERIC10GB", "SHELFSLOTNAME", "PORTNUMBER", "PARENTCARDSLOTNAME", "SLOTPOSITION", "CARDTYPEPORTLIST", "NODEDEFPORTLIST", "1648\_TRIB", "2400PORTNAME", "ALU1830PORTNAME", "CARDSHORTNAME", "TRANSSLOTNUMBER", "PORTNAMETRNS", "CLOCKPORTNAME", "CPLPORT", "Ceragon10", "Ceragon10G", "Harris5000", "EricssonMini", "Ceragon20A", "FL4100PORTNAME", "FLX4020PORT", "FLX4500PORT", "FLX7120PORTNAME", "LindsayPORT", "MW\_LEGACY\_PORTNAMES", "OC192NE", "OC48Lite", "OME6500PORT", "OMXPORTNAME", "PORTSHORTNAME", "SlOTSHORNAME", "TRANSPORTNAME", "GROOVY" ] |

# Open and Closed Issues

## Open Issues

| ID | Issue | Responsibility | Need by Date |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

## Closed Issues

| ID | Issue | Resolution | Resolution Date |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |