



L2X Configuration Guide

Version 20.7.1, 15 July 2020

Registered Address	Support	Sales
26, Kingston Terrace, Princeton, New Jersey 08540, United States		
		+91 80 4850 5445
http://www.rtbrick.com	support@rtbrick.com	sales@rtbrick.com

©Copyright 2020 RtBrick, Inc. All rights reserved. The information contained herein is subject to change without notice. The trademarks, logos and service marks ("Marks") displayed in this documentation are the property of RtBrick in the United States and other countries. Use of the Marks are subject to RtBrick's Term of Use Policy, available at <https://www.rtbrick.com/privacy>. Use of marks belonging to other parties is for informational purposes only.

Table of Contents

1. Introduction	3
1.1. Port and VLAN Cross-connects	3
1.2. L2X 802.1ad Ethertype Support	3
2. L2X Configuration Examples	4
2.1. Ingress VLAN Operation	4
2.2. Supported Match Type Validations	4
2.3. L2X Ingress Configuration Examples	6
2.3.1. Port and Any VLAN (ifp, any_vlan)	6
2.3.2. Port and Untagged VLAN (ifp, untagged)	7
2.3.3. Port and Outer VLAN (ifp, outer_vlan))	8
2.3.4. Port and Dual VLAN Tags (ifp, outer_vlan, inner_vlan)	9
2.3.5. Port + Outer VLAN + Inner VLAN Any (ifp, outer_vlan, any inner_vlan) ..	10
2.4. Egress Interface Action	11
2.5. Support for Bidirectional Local Cross Connect	11
2.6. Deleting L2x Configuration	12
3. Viewing the L2X Configuration	14
3.1. Viewing the L2X Statistics	14

1. Introduction

Layer 2 Cross-Connect (L2X) is a data plane feature that connects two physical ports (IFPs)) using Layer 2 switching. At the simplest, L2X can switch all the traffic between two IFPs to provide the trunk service of an Ethernet switch.

1.1. Port and VLAN Cross-connects

Both types of L2X switches Layer 2 traffic from input interface to output interface. The difference is that a port cross-connect switches all Layer 2 traffic arriving at an input interface, but a VLAN cross-connect only switches the Layer 2 traffic associated with a specific VLAN. A port-based L2X means a port-only configuration, so there are no VLANs involved.

Both single-tag and double-tagged (inner and outer VLAN tags) are supported. The port and VLAN L2X support both local and remote L2X configurations. In remote L2X connections, the VLAN cross-connects are typically configured on the MPLS tunnel ingress router.

Untagged traffic on L2X interfaces is also supported. However, there is no way to select only untagged traffic for cross-connecting. Therefore, only port crossconnects are supported for untagged traffic.

1.2. L2X 802.1ad Ethertype Support

RBFS supports VLAN-operations like VLAN add, VLAN swap, and VLAN delete on egress interface. RBFS supports similar functionality at the ingress side as well. That is, RBFS supports the below VLAN operations:

- Single-VLAN-Add with an option to configure encapsulation (that is, 802.1q or 802.1ad)
- Single-VLAN-Delete
- Swap-Outer-VLAN

By default the encapsulation method is 802.1q. If an encapsulation method is not specified, the 802.1q will be the default mode.

In addition to setting the Ethertype for a VLAN operation, the 802.1ad support includes that ingress traffic for all tagged match options will match on both Ethertype 0x8100 (802.1q) and 0x8A88 (802.1ad) by default.

2. L2X Configuration Examples

These examples cover the various possibilities for VLAN value editing in local L2X.

2.1. Ingress VLAN Operation

RBFS supports VLAN-operations like VLAN add, VLAN swap and VLAN delete on ingress interface.

The current functionality has been extended all existing CLIs to accept ingress vlan-operation and ingress VLAN encapsulation value.

Both 802.1q and 802.1ad encapsulations are supported. The default encapsulation is 802.1q.

Traffic will be matched at ingress direction based on the match criterion. RtBrick Full Stack (RBFS) supports below match parameters.

There can be five different match types on a physical interface, that is, traffic can be matched based on the following:

1. (ifp)
2. (ifp, outer_vlan)
3. (ifp, outer_vlan, inner_vlan)
4. (ifp, outer_vlan, any inner_vlan)
5. (ifp, any vlan)

A few modes in above list are mutually exclusive (for example (ifp, outer_vlan, inner_vlan) and (ifp, outer_vlan, any inner_vlan) configuration on the same interface is conflicting configuration).

For more information, see the [L2X Ingress Configuration Examples](#) section below.

2.2. Supported Match Type Validations

Case1 : IFP A, *



The asterisk (*) indicates *any* or *no vlan* tags.

Configuration A	Configuration B	Support
IFP A, *	IFP A, ov 10	Not supported
IFP A, *	IFP A, ov 10, iv 20	Not supported
IFP A, *	IFP A, ov 10, *	Not supported

IFP A, *	IFP A, untagged	Not Supported
----------	-----------------	---------------

Case 2: IFP A, untagged:

Configuration A	Configuration B	Support
IFP A, untagged	IFP A, *	Not supported
IFP A, untagged	IFP A, ov 10	supported
IFP A, untagged	IFP A, ov 30, iv 20	Supported
IFP A, untagged	IFP A, ov 20, *	Supported

Case 3: IFP A, outer_vlan:

Configuration A	Configuration B	Support
IFP A, ov 10	IFP A, *	Not supported
IFP A, ov 10	IFP A, ov 10, *	Not supported
IFP A, ov 10	IFP A, ov 20	Supported
IFP A, ov 10	IFP A, ov 10 , iv 20	Not supported
IFP A, ov 10	IFP A, ov 40 , iv 7	supported
IFP A, ov 10	IFP A, ov 30, *	supported
IFP A, ov 10	IFP A, untagged	supported

Case 4: IFP A, outer_vlan, inner_vlan:

Configuration A	Configuration B	Support
IFP A,ov 10, iv 20	IFP A, *	Not supported
IFP A,ov 10, iv 20	IFP A, ov 10, *	Not supported
IFP A,ov 10, iv 20	IFP A, ov 10	Not supported
IFP A,ov 10, iv 20	IFP A, ov 30	Supported
IFP A,ov 10, iv 20	IFP A, ov 20 , *	supported
IFP A,ov 10, iv 20	IFP A, untagged	supported
IFP A,ov 10, iv 20	IFP A, ov 10, iv 30	supported

Case 5: IFP A, outer_vlan, *

The asterisk (*) indicates *any* or *no vlan* tags.

Configuration A	Configuration B	Support
-----------------	-----------------	---------

IFP A,ov 10, *	IFP A, *	Not supported
IFP A,ov 10, *	IFP A, ov 10	Not supported
IFP A,ov 10, *	IFP A, ov 10, iv 20	Not supported
IFP A,ov 10, *	IFP A, ov 20, iv 7	Supported
IFP A,ov 10, *	IFP A, ov 30	supported
IFP A,ov 10, *	IFP A, untagged	Supported
IFP A,ov 10, *	IFP A, ov 40, *	supported

2.3. L2X Ingress Configuration Examples

The following command shows how to configure ingress VLAN operation (Local Cross-Connect)

```
rtb confd set static l2x <l2x-name> direction <l2x-direction> incoming-interface <incoming-interface> vlan-operation <ingress-vlan-action> <vlan-id> encapsulation <encapsulation> outgoing-interface <outgoing-interface> vlan-operation <vlan-action> <vlan_id> encapsulation <encapsulation1>
```

The following command shows how to configure ingress VLAN operation (remote Cross-Connect)

```
rtb confd set static l2x <l2x-name> direction <l2x-direction> incoming-interface <incoming-interface> vlan-operation <ingress-vlan-action> <vlan-id> encapsulation <encapsulation> next-hop <nexthop> lookup-instance <lookup_instance> <lookup_af> <lookup_safi> service-label <service_label>
```

2.3.1. Port and Any VLAN (ifp, any vlan)

You can run the following commands in this mode:

```
set incoming-interface <incoming-interface> outgoing-interface <outgoing-interface>
```

```
set incoming-interface <incoming-interface> outgoing-interface <outgoing-interface> vlan-operation <vlan-action> <vlan_id>
```

```
set incoming-interface <incoming-interface> next-hop <nexthop> lookup-instance <lookup_instance> <lookup_af> <lookup_safi> service-label <service_label>
```

set incoming-interface <incoming-interface> **vlan-operation** <vlan-action> <vlan_id> **next-hop** <nexthop> **lookupinstance** <lookup_instance> <lookup_afi> <lookup_safi> **service-label** <service_label>

Command Arguments

Command Argument	Description
<incoming-interface>	Incoming interface is where the traffic originates
<outgoing-interface>	Outgoing interface traffic is going to
<vlan-action>	Indicates the VLAN action such as Single-Vlan-Add, Single-Vlan-Delete, or Swap-Outer-Vlan
<vlan_id>	VLAN ID value (2 to 4095)
<nexthop>	Next-Hop address
<lookup_instance>	Instance name
<lookup_afi>	AFI value: ipv4 or ipv6
<lookup_safi>	SAFI value: safi values are unicast, labeled-unicast
<service_label>	Service label value

Configuration Example

```
rtb confd set static l2x name test1 direction ingress incoming-interface ifp-0/0/4 outgoing-interface ifp-0/0/8
rtb confd set static l2x name test2 direction ingress incoming-interface ifp0/0/4 vlan-operation Single-Vlan-Add 100 outgoing-interface ifp-0/0/8
rtb confd set static l2x name test3 direction ingress incoming-interface ifp-0/0/4 next-hop 10.1.1.2 lookup-instance default ipv4 labeled-unicast service-label 1000
rtb confd set static l2x name test4 direction ingress incoming-interface ifp0/0/4 vlan-operation Single-Vlan-Delete next-hop 10.1.1.2 lookup-instance default ipv4 labeled-unicast servicelabel 2000
```

2.3.2. Port and Untagged VLAN (ifp, untagged)

You can run the following commands in this mode:

set incoming-interface <incoming-interface> **match-untagged** **outgoing-interface** <outgoing-interface>

set incoming-interface <incoming-interface> **match-untagged** **outgoing-interface** <outgoing-interface> **vlan-operation** <vlan-action> <vlan_id>

set incoming-interface <incoming-interface> **match-untagged** **vlan-operation** <vlan-action> <vlan_id> **next-hop** <nexthop> **lookup-instance** <lookup_instance> <lookup_afi> <lookup_safi> **service-label** <service_label>



For (IFP, untagged) match criteria, the **match_type** attribute is mandatory when you perform L2X configuration using curl.

Command Arguments

See the [Command Arguments](#) section.

Configuration Example

```
rtb confd set static l2x name test1 direction ingress incoming-interface ifp-0/0/4 match-untagged vlan-operation Single-Vlan-Add 100 outgoing-interface ifp-0/0/8
rtb confd set static l2x name test1 direction ingress incoming-interface ifp-0/0/4 match-untagged outgoing-interface ifp-0/0/8 vlan-operation Single-Vlan-Add 100
rtb confd set static l2x name test3 direction ingress incoming-interface ifp-0/0/4 match-untagged vlan-operation Single-Vlan-Add 200 next-hop 10.1.1.2 lookup-instance default ipv4 labeled-unicast service-label 1000
```

2.3.3. Port and Outer VLAN (ifp, outer_vlan))

You can run the following commands in this mode:

set incoming-interface <incoming-interface> **outer-vlan** <vlan-id> **outgoing-interface** <outgoing-interface>

set incoming-interface <incoming-interface> **outer-vlan** <vlan-id> **outgoing-interface** <outgoing-interface> **vlan-operation** <vlan-action> <vlan_id>

set incoming-interface <incoming-interface> **outer-vlan** <vlan-id> **next-hop** <nexthop> **lookup-instance** <lookup_instance> <lookup_afi> <lookup_safi> **service-label** <service_label>

set incoming-interface <incoming-interface> **outer-vlan** <vlan-id> **vlan-operation** <vlan-action> <vlan_id> **next-hop** <nexthop> **lookup-instance** <lookup_instance> <lookup_afi> <lookup_safi> **service-label** <service_label>

Command Arguments

See the [Command Arguments](#) section.

Configuration Example

```

rtb confd set static l2x name test1 direction ingress incoming-interface ifp-
0/0/4 outer-vlan 100 outgoing-interface ifp-0/0/8
rtb confd set static l2x name test2 direction ingress incoming-interface ifp-
0/0/5 outer-vlan 200 vlan-operation Single-Vlan-Add 100 outgoing-interface
ifp-0/0/8
rtb confd set static l2x name test3 direction ingress incoming-interface ifp-
0/0/4 outer-vlan 300 next-hop 10.1.1.2 lookup-instance default ipv4 labeled-
unicast service-label 1000
rtb confd set static l2x name test4 direction ingress incoming-interface
ifp0/0/4 outer-vlan 400 vlan-operation Single-Vlan-Add 100 next-hop 10.1.1.2
lookup-instance default ipv4 labeledunicast service-label 2000

```

2.3.4. Port and Dual VLAN Tags (ifp, outer_vlan, inner_vlan)

You can run the following commands in this mode:

set incoming-interface <incoming-interface> **outer-vlan** <vlan-id> **inner-vlan** <inner-vlan-id> **outgoing-interface** <outgoing-interface>

set incoming-interface <incoming-interface> **outer-vlan** <vlan-id> **inner-vlan** <inner-vlan-id> **outgoing-interface** <outgoing-interface> **vlan-operation** <vlan-action> <vlan_id>

set incoming-interface <incoming-interface> **outer-vlan** <vlan-id> **inner-vlan** <inner-vlan-id> **next-hop** <nexthop> **lookup-instance** <lookup_instance> <lookup_afi> <lookup_safi> **service-label** <service_label>

s*et incoming-interface <incoming-interface> **outer-vlan** <vlan-id> **inner-vlan** <inner-vlan-id> **vlan-operation** <vlanaction> <vlan_id> **next-hop** <nexthop> **lookup-instance** <lookup_instance> <lookup_afi> <lookup_safi> **service-label** <service_label>

Command Arguments

See the [Command Arguments](#) section.

Configuration Example

```
rtb confd set static l2x name test1 direction ingress incoming-interface ifp-0/0/4 outer-vlan 100 inner-vlan 500 outgoing-interface ifp-0/0/8
rtb confd set static l2x name test2 direction ingress incoming-interface ifp0/0/4 outer-vlan 200 inner-vlan 600 vlan-operation Single-Vlan-Add 100 outgoing-interface ifp-0/0/8
rtb confd set static l2x name test3 direction ingress incoming-interface ifp0/0/4 outer-vlan 300 inner-vlan 700 vlan-operation Single-Vlan-Add 100 next-hop 10.1.1.2 lookup-instance default ipv4 labeled-unicast service-label 1000
rtb confd set static l2x name test4 direction ingress incoming-interface ifp-0/0/4 outer-vlan 400 inner-vlan 800 next-hop 10.1.1.2 lookup-instance default ipv4 labeled-unicast service-label 2000
```

2.3.5. Port + Outer VLAN + Inner VLAN Any (ifp, outer_vlan, any inner_vlan)

You can run the following commands in this mode:

set incoming-interface <incoming-interface> **outer-vlan** <vlan-id> **match-inner-any outgoing-interface** <outgoing-interface>

set incoming-interface <incoming-interface> **outer-vlan** <vlan-id> **match-inner-any outgoing-interface** <outgoing-interface> **vlan-operation** <vlan-action> <vlan_id>

set incoming-interface <incoming-interface> **outer-vlan** <vlan-id> **match-inner-any next-hop** <nexthop> **lookup-instance** <lookup_instance> <lookup_afi> <lookup_safi> **service-label** <service_label>

set incoming-interface <incoming-interface> **outer-vlan** <vlan-id> **match-innerany vlan-operation** <vlan-action> <vlan_id> **next-hop** <nexthop> **lookup-instance** <lookup_instance> <lookup_afi> <lookup_safi> **service-label** <service_label>

Command Arguments

See the [Command Arguments](#) section.

Configuration Example

```

rtb confd set static l2x name test1 direction ingress incoming-interface ifp-
0/0/4 outer-vlan 100 match-inner-any outgoing-interface ifp-0/0/8
rtb confd set static l2x name test2 direction ingress incoming-interface
ifp0/0/4 outer-vlan 200 match-inner-any vlan-operation Single-Vlan-Add 100
outgoing-interface ifp-0/0/8
rtb confd set static l2x name test3 direction ingress incoming-interface
ifp0/0/4 outer-vlan 300 match-inner-any vlan-operation Single-Vlan-Add 100
next-hop 10.1.1.2 lookup-instance default ipv4 labeled-unicast service-label
1000
rtb confd set static l2x name test4 direction ingress incoming-interface ifp-
0/0/4 outer-vlan 400 match-inner-any next-hop 10.1.1.2 lookup-instance
default ipv4 labeled-unicast service-label 2000

```

2.4. Egress Interface Action

In the Egress direction, incoming-interface is not required, and RBFS supports the below commands.

set service-label <service_label> **outgoing-interface** <outgoing-interface>

set service-label <service_label> **outgoing-interface** <outgoing-interface> **vlan-operation** <vlan-action> <vlan_id>

Command Arguments

See the [Command Arguments](#) section.

Configuration Example

```

rtb confd set static l2x name test1 direction egress service-label 1000
outgoing-interface ifp-0/0/8
rtb confd set static l2x name test1 direction egress service-label 1000
outgoing-interface ifp-0/0/8 vlan-operation Single-Vlan-Add 100

```

2.5. Support for Bidirectional Local Cross Connect

In the earlier releases of RBFS, to establish cross connection between two ports, there was a need to install two l2x configurations, that is, one connection for the ifp1 to ifp2 and another connection for ifp2 to to ifp1.

With bidirectional cross connect, you can establish cross connection between two local ports say ifp1 ↔ ifp2 with single l2x configuration.

To support this feature, the existing L2X commands are extended to accept “bi-directional” direction attribute, and this bi-directional attribute is applicable to only for local cross connect. If bi-directional direction is used for remote cross connect (that is, if nexthop4, nexthop6 or service-label is present) configuration will fail with

an error.



The VLAN operations are not supported for bi-directional local cross connect.

Syntax

```
rtb confd set static l2x name <l2x name> direction bi-directional  
incoming-interface <incoming-interface> outgoing-interface <outgoing-  
interface>
```

Example

```
rtb confd set static l2x name test4 direction bi-directional incoming-  
interface memif-4/4/4 outgoing-interface memif-8/8/8
```

The following example shows the show running-configuration in JSON format.

```
"static": [  
  {  
    "l2x:test4 bi-directional": {  
      "incoming-interface": "memif-4/4/4",  
      "outer-vlan": 65535,  
      "inner-vlan": 65535,  
      "outgoing-interface": "memif-8/8/8"  
    }  
  }  
],
```

2.6. Deleting L2x Configuration

To delete the L2X configuration, enter the following command:

Syntax

```
delete l2x <l2x-name> direction <direction>
```

Command Arguments

Command Argument	Description
<l2x-name>	Name of the Layer 2 Cross-Connect
<direction>	Indicates the direction of the traffic, that is egress or ingress

Example

```
rtb confd delete static test1 direction ingress
```

3. Viewing the L2X Configuration

3.1. Viewing the L2X Statistics

To view the L2X statistics, enter the following command:

```
rtb fibd show bcm l2x statistics
```

Example

```
ubuntu@rtbrick:~$ rtb fibd show bcm l2x statistics
-----
L2X name                               l2x-
l2x_circuit_match_port_no_manipulation_1/0
-----
Physical Interface Name                 ifp-0/0/1
IFL type                               L2x ingress vlan interface
Port Mapping Core                       0
Vlan Port Id                           1149251592
In Forward Packets                      50
In Forward Bytes                        4361
In Drop Packets                         0
In Drop Bytes                          0
Out forward Packets                     0
Out forward Bytes                       0
Out Drop Packets                        0
Out Drop Bytes                         0
-----
L2X name                               l2x-
l2x_circuit_match_port_no_manipulation_2/0
-----
Physical Interface Name                 ifp-0/0/2
IFL type                               L2x ingress vlan interface
Port Mapping Core                       0
Vlan Port Id                           1149251594
In Forward Packets                      43
In Forward Bytes                        3923
In Drop Packets                         0
In Drop Bytes                          0
Out forward Packets                     0
Out forward Bytes                       0
Out Drop Packets                        0
Out Drop Bytes                         0
-----
L2X name                               l2x-
2132b0941383167b4b9d5994ab3fd429421a90aa7201d6f0
```

```
-----  
Physical Interface Name          ifp-0/0/1  
IFL type                        L2x egress vlan interface  
Port Mapping Core               0  
Vlan Port Id                    1149251593  
In Forward Packets              0  
In Forward Bytes                0  
In Drop Packets                 0  
In Drop Bytes                   0  
Out forward Packets             38  
Out forward Bytes               3412  
Out Drop Packets                0  
Out Drop Bytes                  0  
-----
```

```
-----  
L2X name                        l2x-  
c87f4533715b1f520d0475a39a56e261ccecbl3443ff173e  
-----
```

```
-----  
Physical Interface Name          ifp-0/0/2  
IFL type                        L2x egress vlan interface  
Port Mapping Core               0  
Vlan Port Id                    1149251591  
In Forward Packets              0  
In Forward Bytes                0  
In Drop Packets                 0  
In Drop Bytes                   0  
Out forward Packets             45  
Out forward Bytes               3836  
Out Drop Packets                0  
Out Drop Bytes                  0  
ubuntu@rtbrick:~$
```