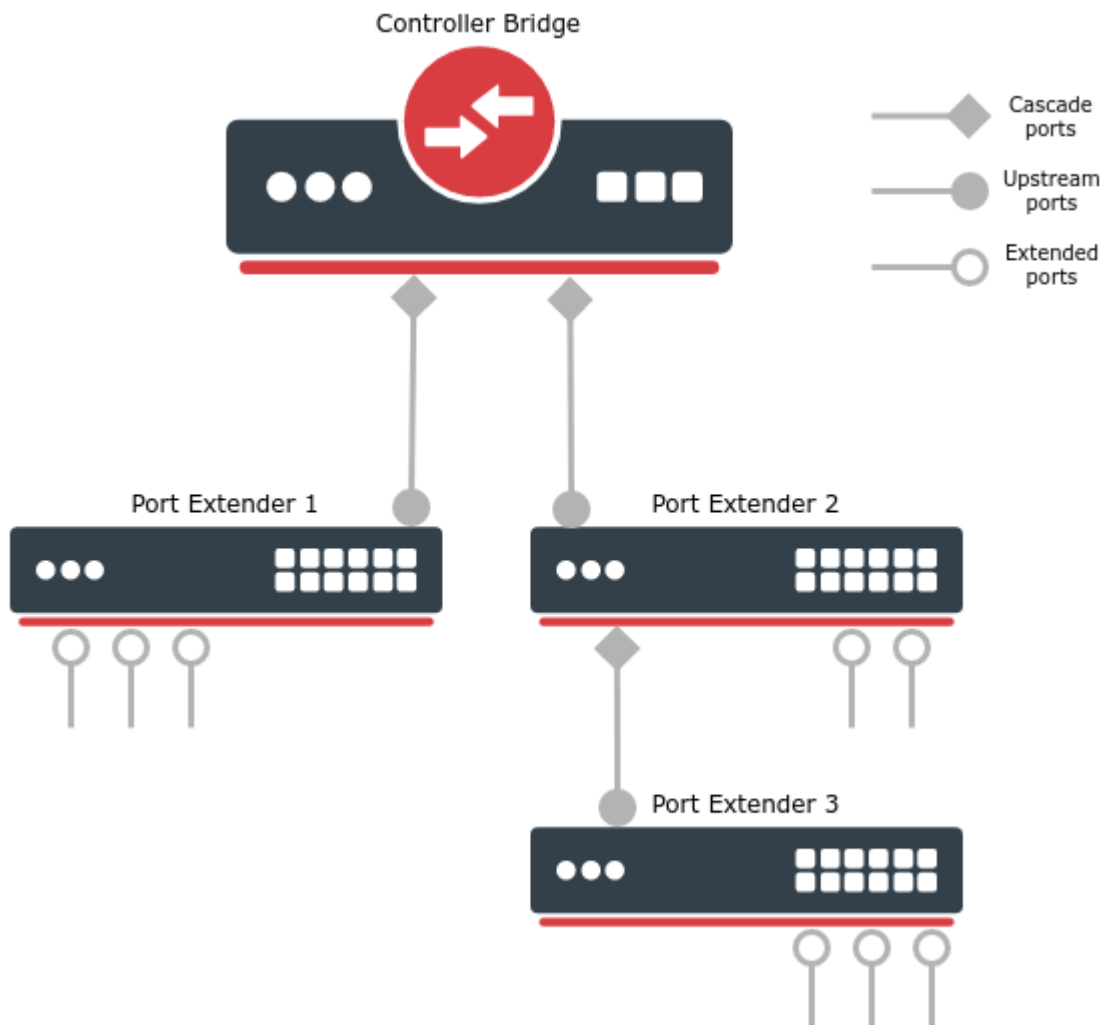


Controller Bridge and Port Extender

- Summary
 - Limitations
- Quick setup
- Discovery and control protocols
- Packet flow
- Controller Bridge settings and monitoring
- Port Extender settings
- Configuration examples
 - Basic CB and PE configuration
 - Trunk and Access ports
 - Cascading multiple Port Extenders and using bonding interface
 - Configuration modification and removal

Summary

Controller Bridge (CB) and Port Extender (PE) is an IEEE 802.1BR standard implementation in RouterOS for CRS3xx series switches. It allows virtually extending the CB ports with a PE device and manage these extended interfaces from a single controlling device. Such configuration provides a simplified network topology, flexibility, increased port density and ease of manageability. An example of Controller Bridge and Port Extender topology can be seen below.



The Controller Bridge establishes communication with the Port Extender through a **cascade port**. Similarly, the Port Extender will communicate with the Controller Bridge only through an **upstream port**. On a PE device, control ports must be configured and only one port (closest to the CB) will act as an upstream port, other control ports can act as a backup for upstream port or even cascade port for switches connected in series (e.g. Port Extender 2 and 3 in the image above). Cascade and upstream ports are used to transmit and receive control and network traffic. **Extended ports** are interfaces that are controlled by the CB device and they are typically connected to the end hosts. Extended ports only transmit and receive network traffic.

See supported features for each switch model below.

Model	Controller Bridge	Port Extender
netPower 15FR (CRS318-1Fi-15Fr-2S)	-	+
netPower 16P (CRS318-16P-2S+)	-	+
CRS310-1G-5S-4S+ (netFiber 9/IN)	-	+
CRS326-24G-2S+ (RM/IN)	-	+
CRS328-24P-4S+	-	+
CRS328-4C-20S-4S+	-	+
CRS305-1G-4S+	-	+
CRS309-1G-8S+	+	+
CRS317-1G-16S+	+	+
CRS312-4C+8XG	+	+
CRS326-24S+2Q+	+	+
CRS354-48G-4S+2Q+	+	+
CRS354-48P-4S+2Q+	+	+

Limitations

Although controller allows to configure port extender interfaces, some bridging and switching features cannot be used or will not work properly. Below are the most common controller and extender limitations. The list might change along upcoming RouterOS releases.

Feature	Support
Bonding for cascade and upstream ports	+
Bridge VLAN filtering	+
Bonding for extended ports	-
Dot1x authenticator (server)	-
Ingress and egress rate	-
Mirroring	-
Port ingress VLAN filtering	-
Port isolation	-
Storm control	-
Switch rules (ACL)	-
L3HW offloading	-
MLAG	-

Quick setup

In this example, we will create a Controlling Bridge (e.g. a CRS317-1G-16S+ switch) that will connect to a single Port Extender (e.g. a CRS326-24G-2S+ switch) through an SFP+1 interface.

First, configure a bridge with enabled VLAN filtering on a CB device:

```
/interface bridge
add name=bridge1 vlan-filtering=yes
```

On the same device, configure a port that is connected to the PE device and will act as cascade port:

```
/interface bridge port-controller
set bridge=bridge1 cascade-ports=sfp-sfpplus1 switch=switch1
```

Last, on a PE device, simply configure a control port, which will be selected as an upstream port:

```
/interface bridge port-extender
set control-ports=sfp-sfpplus1 switch=switch1
```

Once PE and CB devices are connected, all interfaces that are on the same switch group (except for control ports) will be extended and can be further configured on a CB device. An automatic bridge port configuration will be applied on the CB device which adds all extended ports in a single bridge, this configuration can be modified afterward.



In order to exclude some port from being extended (e.g. for out-of-band management purposes), additionally, configure `excluded-ports` property.



Make sure not to include the cascade-ports and control-ports in any routing or bridging configurations. These ports are recommended only for a CB and PE usage.

Discovery and control protocols

Before frame forwarding on extended ports is possible, Controlling Bridge and Port Extender must discover each other and exchange with essential information.

CB and PE enabled devices are using a neighbor discovery protocol LLDP with specific Port Extension TLV. This allows CB and PE devices to advertise their support on cascade and control ports.



CB and PE configuration can override the neighbor discovery settings, for example, if a cascade port is not included in a neighbor discovery interface list, the LLDP messages will be still sent.

Once LLDP messages are exchanged between CB and PE, a Control and Status Protocol (CSP) over an Edge Control Protocol (ECP) will initiate. The CSP is used between CB and PE to assert control and receive status information from the associated PE - it assigns unique IDs for extended ports, controls data-path settings (e.g. port VLAN membership) and sends port status information (e.g. interface stats, PoE-out monitoring). The ECP provides a reliable and sequenced frame delivery (encoded with EtherType 0x8940).

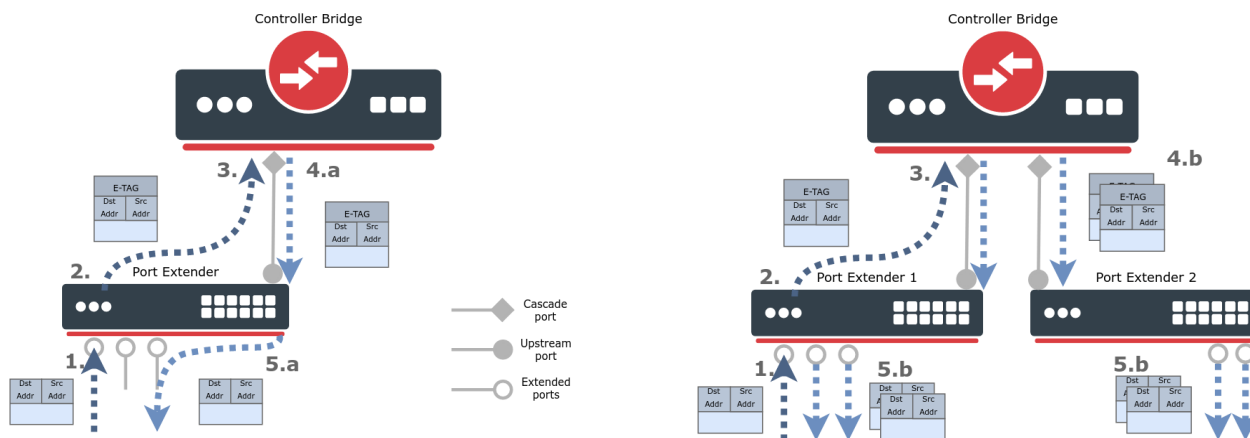


The current CB implementation does not support any failover techniques. Once the CB device becomes unavailable, the PE devices will lose all the control and data forwarding rules.

Packet flow

To better understand the underlying principles of Controlling Bridge and Port Extender, a packet walkthrough is provided below:

1. An L2 packet is received on the extended port;
2. The Port Extender encapsulates the packet with an E-TAG header (EtherType 0x893F) and forwards it through an upstream port, towards the Controller Bridge. An E-TAG packet contains information regarding the PE source port ID. The PE device does not make any local switching decisions;
3. The Controller Bridge receives the E-TAG packet and knows exactly which extended interface received it. The CB then internally decapsulates the packet and proceed it through a regular switching decision (host learning, destination address lookup, VLAN filtering, etc.);
4. Once a switching decision is made, the CB will again encapsulate the original packet with an E-TAG and send it through a cascade port, towards Port Extender;
 - a. For a single destination packet (unicast), the CB will include the PE destination port ID in the E-TAG and send it through a correct cascade port;
 - b. For a multi-destination packet (broadcast, multicast or unknown-unicast), the CB will include a target group mark and source port ID in the E-TAG and send a single packet replica per every cascade port;
5. Once a PE device receives an E-TAG packet on the upstream port, PE decapsulates it and sends the original L2 packet through the extended port;
 - a. For a single destination packet (unicast), the PE will send the packet only to the correct extended port;
 - b. For a multi-destination packet (broadcast, multicast or unknown-unicast), the PE will send a single packet replica per every extended port (except for the source port where the packet was received).



Controller Bridge settings and monitoring

This section describes the Controller Bridge settings and monitoring options.

Sub-menu: `/interface bridge port-controller`

Property	Description
bridge (<i>name</i> ; Default: none)	The bridge interface where ports will be extended. The CB will only enable when <code>bridge</code> and <code>switch</code> properties are specified, otherwise, it will be in a disabled state.
cascade-ports (<i>interfaces</i> ; Default: none)	Interfaces that will act as cascade ports. A bonding interface with 802.3ad or balance-xor mode is also supported.
switch (<i>name</i> ; Default: none)	The switch that will act as the CB and ensure the control and network traffic. The CB will only enable when <code>bridge</code> and <code>switch</code> properties are specified, otherwise, it will be in a disabled state.

After CB and PE devices are configured and connected, each PE device will be automatically visible on the device menu, use `print` and `monitor` commands to see more details.

```
[admin@Controller] > interface bridge port-controller device print
Flags: I - inactive
 0 name="pe1" pe-mac=64:D1:54:EB:AE:BC descr="MikroTik RouterOS 6.48beta35 (testing) CRS328-24P-4S+"
  control-ports=pe1-sfpplus1,pe1-sfpplus2

 1 name="pe2" pe-mac=64:D1:54:C7:3A:58 descr="MikroTik RouterOS 6.48beta35 (testing) CRS326-24G-2S+"
  control-ports=pe2-sfpplus1
[admin@Controller] > interface bridge port-controller device monitor pe2
      name: pe2
      status: active
connected-via-ports: sfp-sfpplus1==pe1-sfpplus1,pe1-sfpplus2==pe2-sfpplus1
connected-via-devs: controller,pe1
```

Sub-menu: /interface bridge port-controller device

Property	Description
connected-via-devs (<i>name</i>)	Shows the connected devices in the path from PE to CB.
connected-via-ports (<i>name</i>)	Shows the connection path from PE to CB.
control-ports (<i>interfaces</i>)	PE device control ports.
descr (<i>name</i>)	Short PE device description.
name (<i>name</i>)	Automatically assigned PE device name.
pe-mac (<i>MAC address</i>)	PE device MAC address.
status (<i>active inactive</i>)	PE device status.

Additionally, each PE device interface can be monitored on the port menu, use **print** and **monitor** commands to see more details.

```
[admin@Controller] > interface bridge port-controller port print where !disabled
Flags: I - inactive, X - disabled, R - running, U - upstream-port, C - cascade-port
# NAME DEVICE
0 I pe1-ether1 pe1
1 R pe1-ether2 pe1
2 R pe1-ether3 pe1
3 R pe1-ether4 pe1
4 U pe1-sfpplus1 pe1
5 RC pe1-sfpplus2 pe1
6 I pe2-ether1 pe2
7 R pe2-ether2 pe2
8 R pe2-ether3 pe2
9 R pe2-ether4 pe2
10 U pe2-sfpplus1 pe2
[admin@Controller] > interface bridge port-controller port monitor [find where !disabled]
      name: pe1-ether1 pe1-ether2 pe1-ether3 pe1-ether4 pe1-sfpplus1 pe1-sfpplus2 pe2-ether1 pe2-ether2
pe2-ether3 pe2-ether4 pe2-sfpplus1
      status: unknown link-ok link-ok link-ok no-link link-ok unknown link-ok
link-ok link-ok no-link
      rate: 1Gbps 1Gbps 1Gbps 10Gbps 10Gbps 1Gbps
1Gbps 1Gbps 10Gbps
      port-status: not-added ok ok ok ok not-added ok
ok ok
      pcid: 457 458 459 480 481 509
510 511 532
```

Sub-menu: /interface bridge port-controller port

Property	Description
device (<i>name</i>)	Automatically assigned PE device name.

name (<i>name</i>)	Automatically assigned PE port name.
pcid (<i>integer</i>)	Automatically assigned port identifier.
port-status (<i>dev-inactive not-added ok</i>)	PE port status.
rate (<i>bps</i>)	Data rate of the connection.
status (<i>link-ok no-link unknown</i>)	PE port link status.

The Controller Bridge can monitor the PoE-out related information from Port Extenders on the port poe menu, use `print` and `monitor` commands to see more details. For more information regarding PoE-out, please visit the [PoE-out manual](#).

```
[admin@Controller] > interface bridge port-controller port poe print
# NAME                               DEVICE
0 pe1-ether1                          pe1
1 pe1-ether2                          pe1
2 pe1-ether3                          pe1
3 pe1-ether4                          pe1
4 pe1-ether5                          pe1
5 pe1-ether6                          pe1
6 pe1-ether7                          pe1
...
[admin@Controller] > interface bridge port-controller port poe monitor pe1-ether2,pe1-ether3
      name: pe1-ether2 pe1-ether3
  poe-out-status: powered-on powered-on
  poe-out-voltage: 52.8V    52.9V
  poe-out-current: 123mA   95mA
  poe-out-power: 6.4W     5W
```

Port Extender settings

This section describes the Port Extender settings.

Sub-menu: `/interface bridge port-extender`

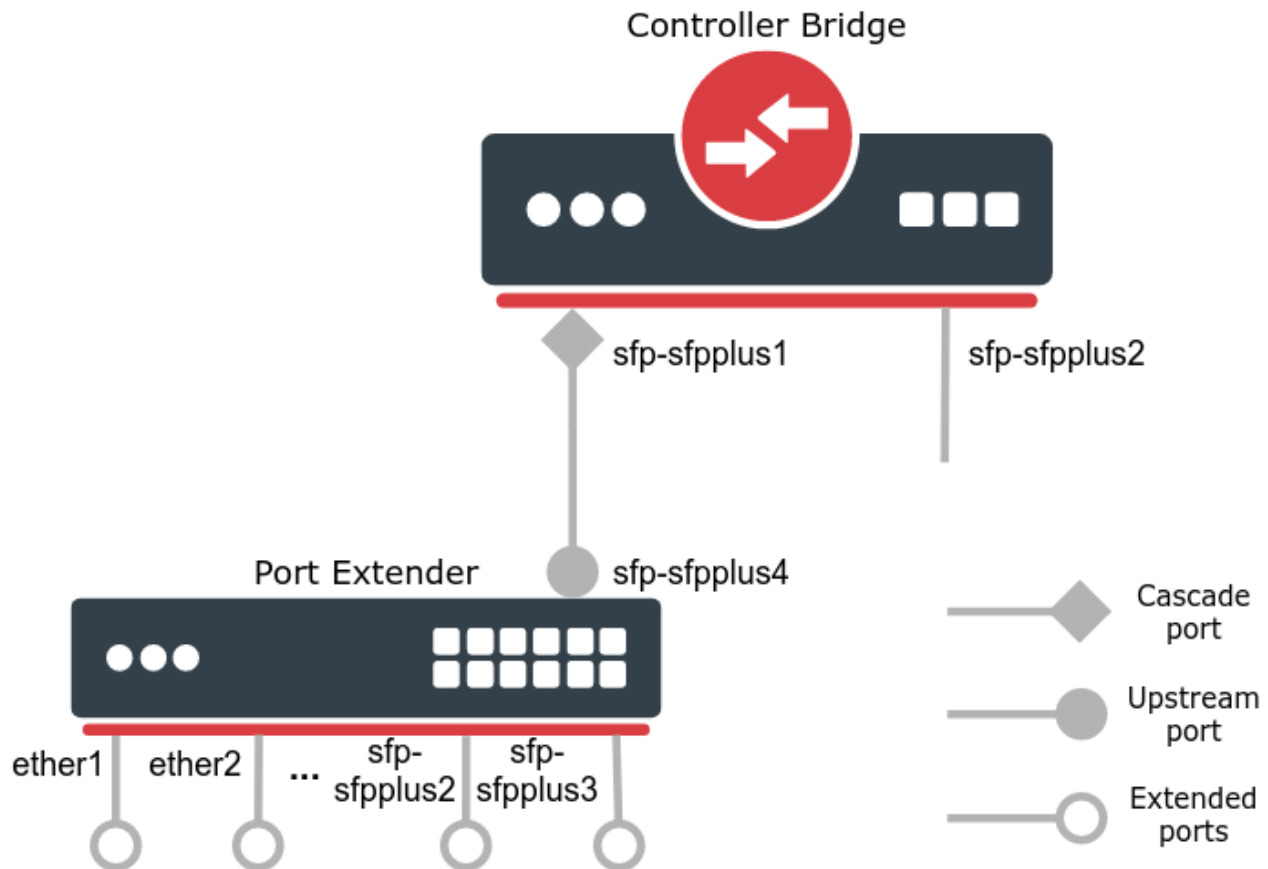
Property	Description
control-ports (<i>interfaces</i> ; Default: none)	Interfaces that will either connect to the CB (upstream port) or connect other PE devices in series (cascade port). A bonding interface with 802.3ad or balance-xor <code>mode</code> is also supported.
excluded-ports (<i>interfaces</i> ; ; Default: none)	Interfaces that will not be extended.
switch (<i>name</i> ; Default: none)	The switch that will act as the extender and ensure the control and network traffic. The PE will only enable when this property is specified, otherwise, it will be in a disabled state.

Configuration examples

Below are described the most common configuration examples. For CB and PE configuration to work properly, a bridge VLAN filtering needs to be enabled, so make sure to understand the filtering principles first - [bridge VLAN filtering](#), [bridge VLAN table](#).

Basic CB and PE configuration

In this example, a CRS317-1G-16S+ device is used as a Controller Bridge and CRS328-24P-4S+ as a Port Extender, see the connection scheme below.



First, configure the CB device. This can be done by adding a bridge interface with enabled VLAN filtering. Additionally, add any local interfaces to the same bridge, it allows to forward traffic between any local interfaces and extended interfaces. In this example, an sfp-sfpplus2 interface is added.

```
/interface bridge
add name=bridge1 vlan-filtering=yes
/interface bridge port
add bridge=bridge1 interface=sfp-sfpplus2
```

To enable CB, specify the bridge, switch and at least one cascade port. Make sure that cascade ports are not included in the bridge or routing configurations. These ports are recommended only for a CB and PE usage.

```
/interface bridge port-controller
set bridge=bridge1 cascade-ports=sfp-sfpplus1 switch=switch1
```

To enable PE, configure control ports and switch. Additionally, configure one or multiple interfaces that should not be extended with `excluded-ports` property (e.g. for out-of-band management purposes). In this example, all switch ports will be extended.

```
/interface bridge port-extender
set control-ports=sfp-sfpplus4 switch=switch1
```

Once PE and CB devices finish the discovery and start the Control and Status Protocol (CSP), the RouterOS will permanently create new interfaces and add them into bridge on the CB device. Interfaces are named by the automatically assigned PE device name, plus the default interface name, these interface names can be modified afterwards. Note that control and excluded ports will be also displayed into the interface list, but they are not included into the bridge.

```
[admin@Controller_Bridge] > /interface print where name~"pe"
Flags: D - dynamic, X - disabled, R - running, S - slave
```

#	NAME	TYPE	ACTUAL-MTU	L2MTU	MAX-L2MTU
0	RS pe1-ether1	extport	1500	1584	
1	RS pe1-ether2	extport	1500	1584	
2	RS pe1-ether3	extport	1500	1584	
3	S pe1-ether4	extport	1500	1584	
4	S pe1-ether5	extport	1500	1584	
5	S pe1-ether6	extport	1500	1584	
6	S pe1-ether7	extport	1500	1584	
7	S pe1-ether8	extport	1500	1584	
8	S pe1-ether9	extport	1500	1584	
9	S pe1-ether10	extport	1500	1584	
10	S pe1-ether11	extport	1500	1584	
11	S pe1-ether12	extport	1500	1584	
12	S pe1-ether13	extport	1500	1584	
13	S pe1-ether14	extport	1500	1584	
14	S pe1-ether15	extport	1500	1584	
15	S pe1-ether16	extport	1500	1584	
16	S pe1-ether17	extport	1500	1584	
17	S pe1-ether18	extport	1500	1584	
18	S pe1-ether19	extport	1500	1584	
19	S pe1-ether20	extport	1500	1584	
20	S pe1-ether21	extport	1500	1584	
21	S pe1-ether22	extport	1500	1584	
22	S pe1-ether23	extport	1500	1584	
23	S pe1-ether24	extport	1500	1584	
24	RS pe1-sfpplus1	extport	1500	1584	
25	RS pe1-sfpplus2	extport	1500	1584	
26	RS pe1-sfpplus3	extport	1500	1584	
27	pe1-sfpplus4	extport	1500	1584	

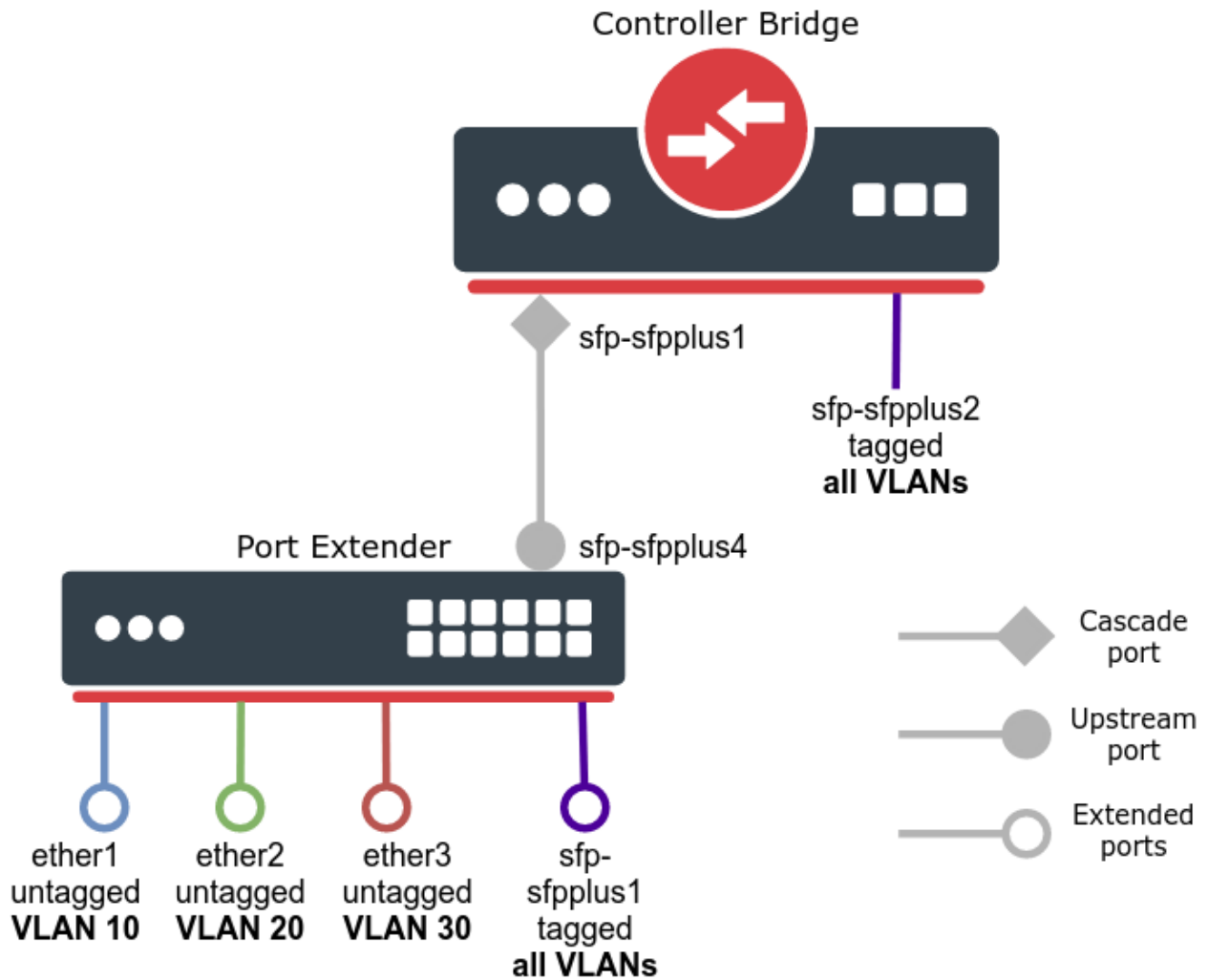
```
[admin@Controller_Bridge] > interface bridge port print
Flags: X - disabled, I - inactive, D - dynamic, H - hw-offload
```

#	INTERFACE	BRIDGE	HW	PVID	PRIORITY	PATH-COST	INTERNAL-PATH-COST	HORIZON
0	H sfp-sfpplus2	bridgel	yes	1	0x80	10	10	none
1	H pe1-ether1	bridgel	yes	1	0x80	10	10	none
2	H pe1-ether2	bridgel	yes	1	0x80	10	10	none
3	H pe1-ether3	bridgel	yes	1	0x80	10	10	none
4	I H pe1-ether4	bridgel	yes	1	0x80	10	10	none
5	I H pe1-ether5	bridgel	yes	1	0x80	10	10	none
6	I H pe1-ether6	bridgel	yes	1	0x80	10	10	none
7	I H pe1-ether7	bridgel	yes	1	0x80	10	10	none
8	I H pe1-ether8	bridgel	yes	1	0x80	10	10	none
9	I H pe1-ether9	bridgel	yes	1	0x80	10	10	none
10	I H pe1-ether10	bridgel	yes	1	0x80	10	10	none
11	I H pe1-ether11	bridgel	yes	1	0x80	10	10	none
12	I H pe1-ether12	bridgel	yes	1	0x80	10	10	none
13	I H pe1-ether13	bridgel	yes	1	0x80	10	10	none
14	I H pe1-ether14	bridgel	yes	1	0x80	10	10	none
15	I H pe1-ether15	bridgel	yes	1	0x80	10	10	none
16	I H pe1-ether16	bridgel	yes	1	0x80	10	10	none
17	I H pe1-ether17	bridgel	yes	1	0x80	10	10	none
18	I H pe1-ether18	bridgel	yes	1	0x80	10	10	none
19	I H pe1-ether19	bridgel	yes	1	0x80	10	10	none
20	I H pe1-ether20	bridgel	yes	1	0x80	10	10	none
21	I H pe1-ether21	bridgel	yes	1	0x80	10	10	none
22	I H pe1-ether22	bridgel	yes	1	0x80	10	10	none
23	I H pe1-ether23	bridgel	yes	1	0x80	10	10	none
24	I H pe1-ether24	bridgel	yes	1	0x80	10	10	none
25	H pe1-sfpplus1	bridgel	yes	1	0x80	10	10	none
26	H pe1-sfpplus2	bridgel	yes	1	0x80	10	10	none
27	H pe1-sfpplus3	bridgel	yes	1	0x80	10	10	none

Now the CRS317-1G-16S+ device has extended its ports using the CRS328-24P-4S+ device and packet forwarding can be done between all bridged ports.

Trunk and Access ports

In this example, untagged (access) and tagged (trunk) port configuration will be created on the Controller Bridge device, see the network diagram below.



First, configure the CB and PE devices, the configuration is identical to the previous example. Use this configuration for CB device.

```
/interface bridge
add name=bridge1 vlan-filtering=yes
/interface bridge port
add bridge=bridge1 interface=sfp-sfpplus2
/interface bridge port-controller
set bridge=bridge1 cascade-ports=sfp-sfpplus1 switch=switch1
```

Use this configuration for PE device.

```
/interface bridge port-extender
set control-ports=sfp-sfpplus4 switch=switch1
```

After extended ports are successfully created and added to the bridge on the CB device, we can start configuring VLAN related properties. First, configure access ports to their respective VLAN ID using a `pvid` property. Use a `print` command in `/interface bridge port` menu to find out the exact interface name.

```
/interface bridge port
set [find interface=pe1-ether1] pvid=10
set [find interface=pe1-ether2] pvid=20
set [find interface=pe1-ether3] pvid=30
```

Then add bridge VLAN entries and specify tagged, untagged ports. Note that there are two tagged ports - local port named sfp-sfpplus2 and extended port named pe1-sfpplus1.

```
/interface bridge vlan
add bridge=bridge1 tagged=pe1-sfpplus1,sfp-sfpplus2 untagged=pe1-ether1 vlan-ids=10
add bridge=bridge1 tagged=pe1-sfpplus1,sfp-sfpplus2 untagged=pe1-ether2 vlan-ids=20
add bridge=bridge1 tagged=pe1-sfpplus1,sfp-sfpplus2 untagged=pe1-ether3 vlan-ids=30
```

At this point VLANs are configured and devices should be able to communicate through the ports. However, it is recommended to go even a step further and apply some additional filtering options. Enable port [ingress-filtering](#) on local bridge ports and use frame filtering based on the packet type with [frame-types](#) setting.

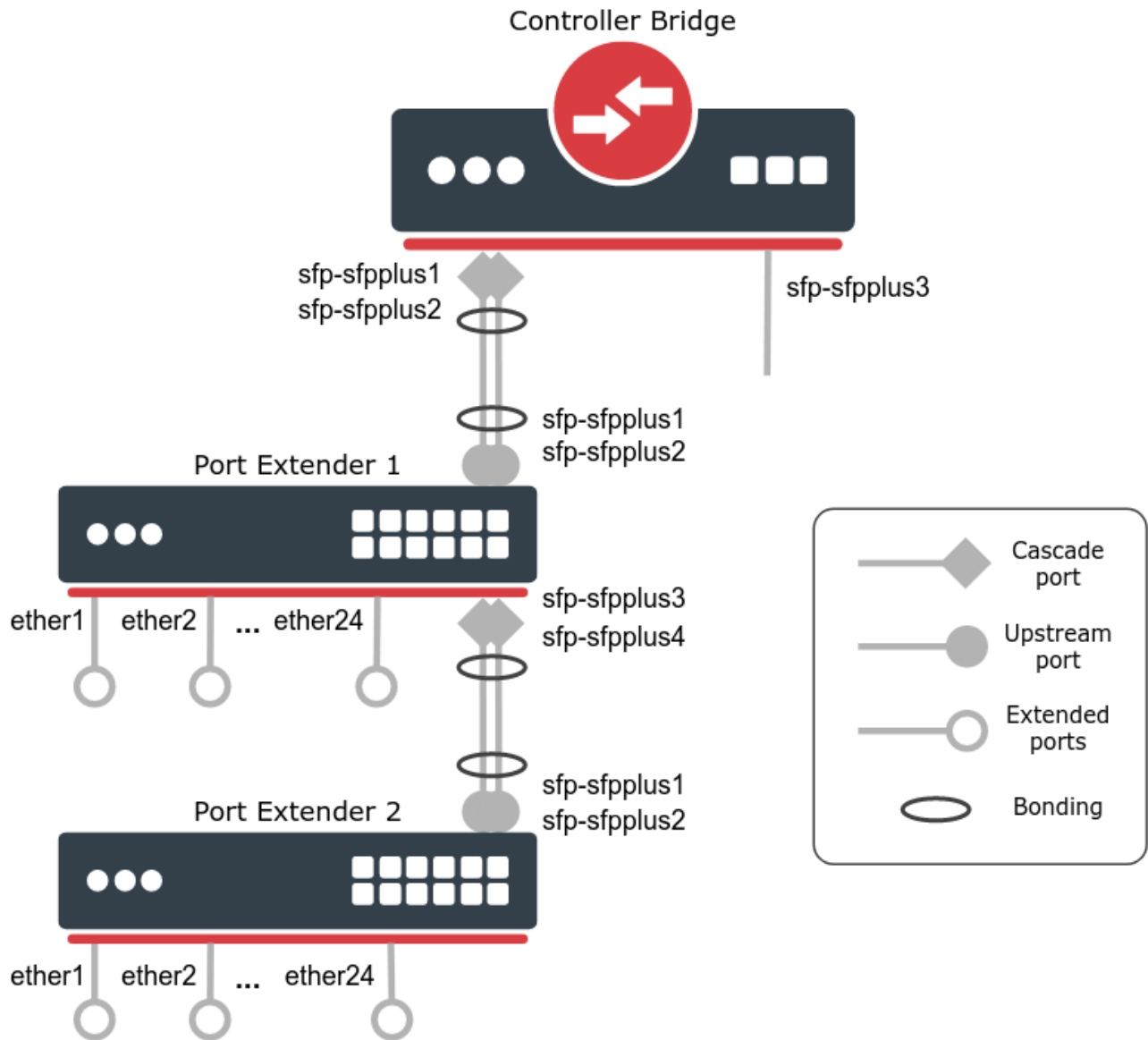
```
/interface bridge port
set [find interface=pe1-ether1] frame-types=admit-only-untagged-and-priority-tagged
set [find interface=pe1-ether2] frame-types=admit-only-untagged-and-priority-tagged
set [find interface=pe1-ether3] frame-types=admit-only-untagged-and-priority-tagged
set [find interface=pe1-sfpplus1] frame-types=admit-only-vlan-tagged
set [find interface=sfp-sfpplus2] frame-types=admit-only-vlan-tagged ingress-filtering=yes
```



Port ingress VLAN filtering is not supported on extended ports.

Cascading multiple Port Extenders and using bonding interface

In this example, two PE devices (CRS328-24P-4S+ and CRS326-24G-2S+) will be added to the CB (CRS317-1G-16S+). To increase throughput for upstream and cascade ports, [bonding interfaces](#) will be created. See the network diagram below.



The CB and PE configuration is similar to the first example, the main difference is the bonding interface usage. First, configure the CB device - create a bonding interface for cascade port, create a bridge and add any needed local bridge ports, last enable the CB. Use the following commands:

```

/interface bonding
add mode=802.3ad name=bond1 slaves=sfp-sfpplus1,sfp-sfpplus2
/interface bridge
add name=bridge1 vlan-filtering=yes
/interface bridge port
add bridge=bridge1 interface=sfp-sfpplus3
/interface bridge port-controller
set bridge=bridge1 cascade-ports=bond1 switch=switch1

```

Then configure the Port Extender 1 device. This device needs two bonding interfaces - the first one will be used as an upstream port and the second one will be a cascade port for the Port Extender 2 device. Additionally, configure one or multiple interfaces that should not be extended with `excluded-ports` property (e.g. for out-of-band management purposes). In this example, all switch ports will be extended.

```

/interface bonding
add mode=802.3ad name=bond1 slaves=sfp-sfpplus1,sfp-sfpplus2
add mode=802.3ad name=bond2 slaves=sfp-sfpplus3,sfp-sfpplus4
/interface bridge port-extender
set control-ports=bond1,bond2 switch=switch1

```

Last, configure the Port Extender 2 device - create a bonding interface and enable PE. Additionally, configure one or multiple `excluded-ports` if necessary. In this example, all switch ports will be extended.

```

/interface bonding
add mode=802.3ad name=bond1 slaves=sfp-sfpplus1,sfp-sfpplus2
/interface bridge port-extender
set control-ports=bond1 switch=switch1

```

Now the CRS317-1G-16S+ device has extended its ports with additional 48 Gigabit Ethernet ports and packet forwarding can be achieved between all bridged ports.

Use the `monitor` command in the device menu to see the PE device connection path. Also, use `print` command in the port menu to see which PE interfaces are used as upstream and cascade ports.

```

[admin@Controller_Bridge] > interface bridge port-controller device monitor [find]
      name: pe1                pe2
      status: active           active
connected-via-ports: bond1==pe1-cntrl-bond1 bond1==pe1-cntrl-bond1
                        pe1-cntrl-bond2==pe2-cntrl-bond1
connected-via-devs: controller controller
                        pe1

[admin@Controller_Bridge] > interface bridge port-controller port print where running or upstream-port
Flags: I - inactive, X - disabled, R - running, U - upstream-port, C - cascade-port
#  NAME                                DEVICE

0 R  pe1-ether2                        pe1
1 R  pe1-ether3                        pe1
2 R  pe1-ether4                        pe1
3 U  pe1-sfpplus1                      pe1
4 U  pe1-sfpplus2                      pe1
5 RC pe1-sfpplus3                     pe1
6 RC pe1-sfpplus4                     pe1
7 R  pe2-ether1                        pe2
8 R  pe2-ether2                        pe2
9 R  pe2-ether3                        pe2
10 R pe2-ether4                        pe2
11 U pe2-sfpplus1                      pe2
12 U pe2-sfpplus2                      pe2

```

Configuration modification and removal

In certain situations, CB and PE device configuration needs to be adjusted (e.g. PE device needs new control ports) or removed completely. To modify the PE device configuration, all related PE device configuration should be removed from the CB device first. Only then the new configuration can be applied.

First, to remove PE configuration from CB, disable the PE using the following command:

```
/interface bridge port-extender set switch=none control-ports="" excluded-ports=""
```

Then, on the CB device, remove the related bridge and other RouterOS configuration where PE interfaces were used (e.g. see the export from "/interface bridge port" and "/interface bridge vlan" menus). For example, to remove all bridge ports from a specific PE device, use the command below:

```
/interface bridge port remove [find interface~"pe1"]
```

Once the configuration is removed, PE can be removed from the CB device list. This command will also automatically remove all the PE device interfaces from the CB interface list. In case some PE interface configuration is still applied on the CB, it will not be valid anymore. Use `print` command to find out the PE device name.

```
/interface bridge port-controller device remove [find name=pe1]
```