TSN Configuration Guide

Application Note

CONFIDENTIAL

AN1185
Rev. APPL-2024.06
2024-06-28

Table of Contents

1. Introduction

This document explains how to setup Time-Sensitive Networking (TSN) features.

Time-Sensitive Networking is a number of IEEE 802 standards that are defined by the IEEE TSN task group. These standards define mechanisms for deterministic real-time communication over Ethernet networks.

The following links give an overview of the purpose and current status of the various TSN standards:

- https://en.wikipedia.org/wiki/Time-Sensitive_Networking
- http://www.ieee802.org/1/pages/tsn.html

The examples used in this document are valid for `IStaX` builds only.

2. Using TSN Functions with PTP Time Synchronisation

When using TAS and PSFP between network elements, it is required to have a common global time reference provided by PTP. When booting the device, it will take some time for a configured PTP application to get locked to the common time reference. It may cause malfunctioning of TAS and PSFP if config-change is issued before PTP time is in a Locked or Locking state. A function which can delay the issue of config-change until PTP is Locked/Locking or a configurable time has passed, can be configured with the CLI command: `tsn ptp-check`.

The configuration of PTP is out of scope for this configuration guide.

The syntax for TSN delayed start function is:

```
tsn ptp-check procedure {none | ptp | wait}
tsn ptp-check ptp-port <0-3>
tsn ptp-check timeout <10-200>
no tsn ptp-check procedure
no tsn ptp-check ptp-port
no tsn ptp-check timeout
```

Where:

```
none        Procedure: Start TSN functions immediately without any delay (default)
ptp         Procedure: Monitor the status of PTP time. Start if it is Locking or
Locked.
            If Locking or Locked is not achieved within wait time, then start anyway
wait        Procedure: Wait timeout number of seconds before starting TSN functions
ptp-port    The PTP port to use for sensing PTP status
timeout     Set ptp-check-procedure timeout in seconds
```

An example is shown below:

```
(config)# tsn ptp-check procedure ptp
(config)# tsn ptp-check ptp-port 2
(config)# tsn ptp-check timeout 30
```

It is intended that after configuring tsn ptp-check, it should be saved to startup-config, and the delay function will only be executed once after a power cycle or `reload cold`.

The current time can be displayed with `show tsn current-time` in CLI EXEC mode.

3. Credit Based Shaper

Credit Based Shaper is defined in the IEEE 802.1Q-2014 standard and is the ability to control the traffic access bandwidth based on priorities. The highest priority queue can be assigned a higher access bandwidth relative to the available bandwidth, which in turn gives higher chance for packets to be transmitted in a busy network.

The mechanism is realized through increasing/decreasing credit value of the specific queues, i.e., the credit for high priority queues increases faster and therefore reaches the transmitting threshold more frequent than the low priority queues. The algorithm includes two parts, namely assigning priorities to traffic classes/queues and assigning relative access bandwidth (reflected in credits) to the queues.

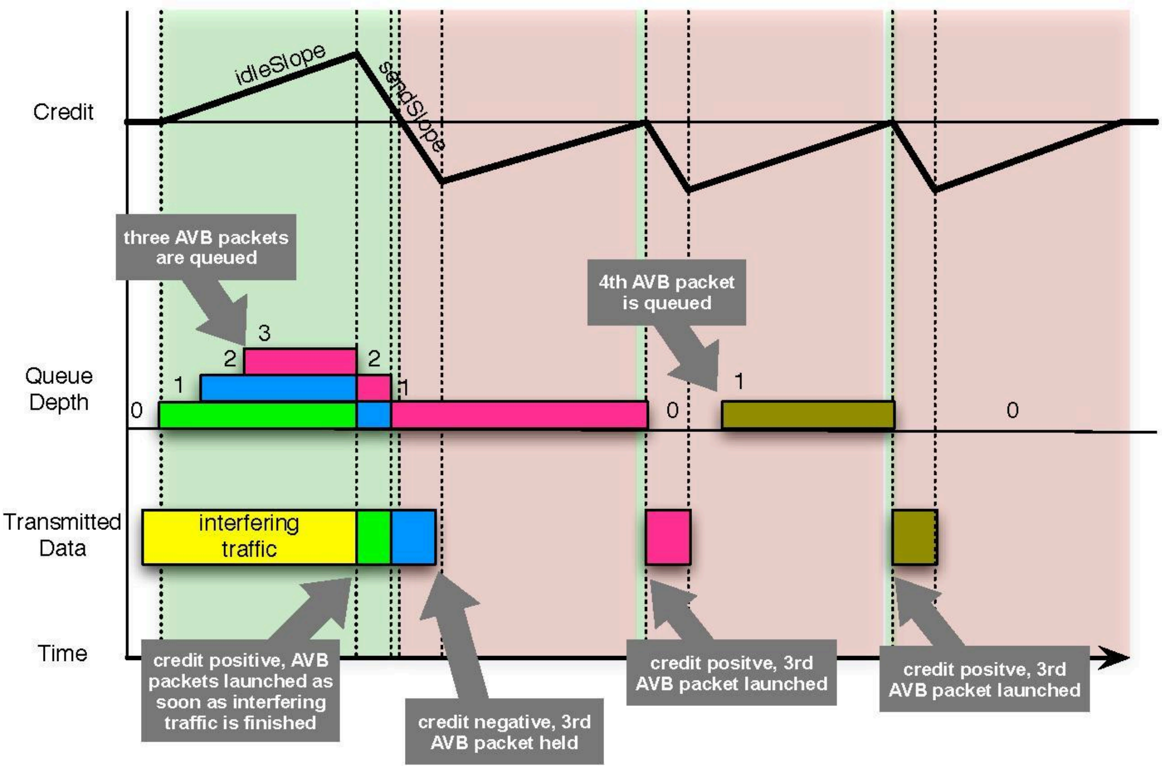


Figure 1. Credit Based Shaper

3.1. Configuration

To confure a credit based shaper on port 1, navigate to Configuration → QoS → Port Shaping and click on port 1.

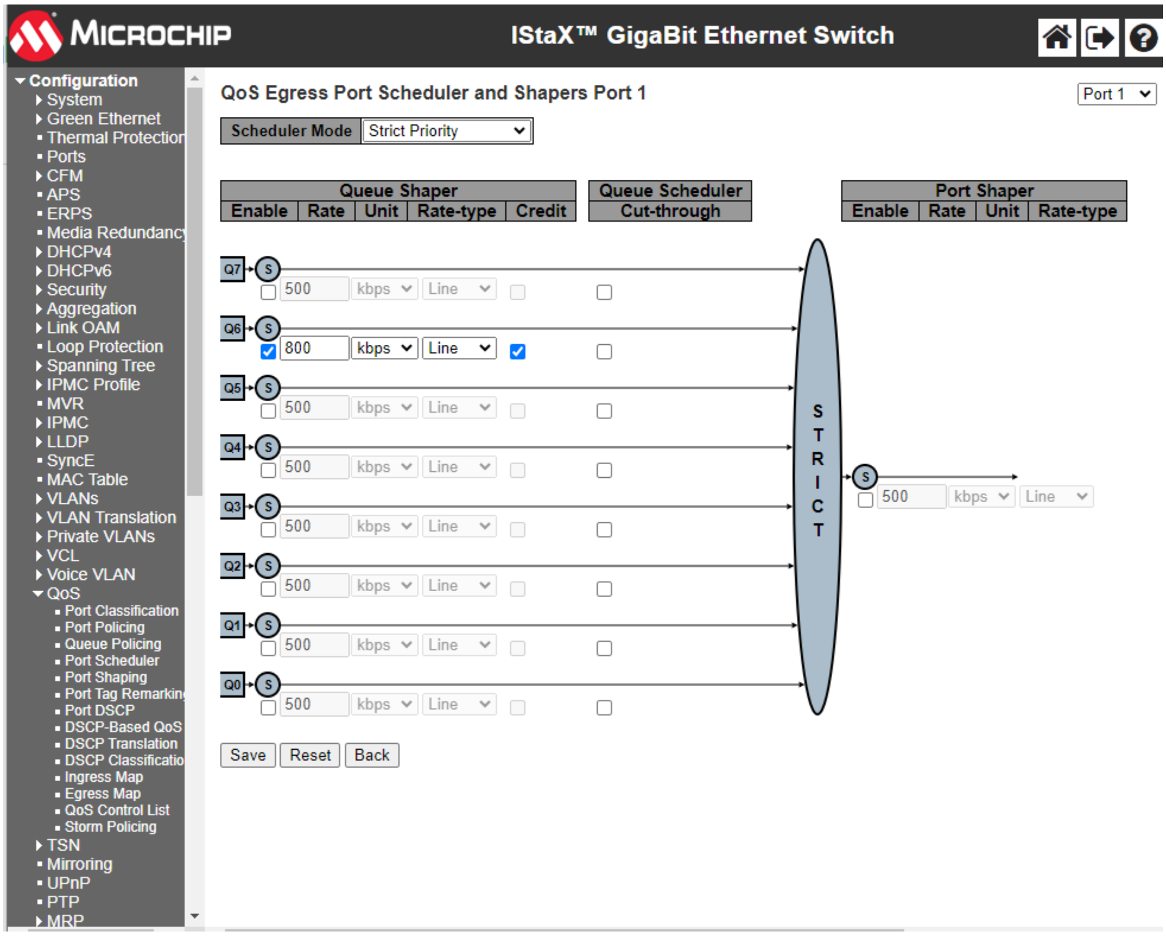


Figure 2. Qos Egress Port Scheduler

In the example, a credit based shaper is created for queue 6 with a line rate of 800kbps. Please note the the checkbox "Credit" has been clicked.

The equivalent CLI commands for creating and deleting the shaper are:

```
! Enable a 800 kbps credit based shaper on port 1, queue 6.
# configure terminal
(config)# interface 10GigabitEthernet 1/1
(config-if)# qos queue-shaper queue 6 800 kbps credit
```

```
! Disable the shaper on port 1, queue 6.
# configure terminal
(config)# interface 10GigabitEthernet 1/1
(config-if)# no qos queue-shaper queue 6
```

```
! Show configuration using 'show running-config'.
# show running-config interface 10GigabitEthernet 1/1
Building configuration...
interface 10GigabitEthernet 1/1
 qos queue-shaper queue 6 800 kbps credit
!
end
```

The status of the quality of service related configuration can also be seen with the command:

```
! Show status using 'show qos'.
# show qos interface 10GigabitEthernet 1/1
[...]
qos queue-shaper queue 0: disabled, rate 500 kbps, mode: line-rate, excess:
disabled, credit: disabled
qos queue-shaper queue 1: disabled, rate 500 kbps, mode: line-rate, excess:
disabled, credit: disabled
qos queue-shaper queue 2: disabled, rate 500 kbps, mode: line-rate, excess:
disabled, credit: disabled
qos queue-shaper queue 3: disabled, rate 500 kbps, mode: line-rate, excess:
disabled, credit: disabled
qos queue-shaper queue 4: disabled, rate 500 kbps, mode: line-rate, excess:
disabled, credit: disabled
qos queue-shaper queue 5: disabled, rate 500 kbps, mode: line-rate, excess:
disabled, credit: disabled
qos queue-shaper queue 6: enabled, rate 800 kbps, mode: line-rate, excess: disabled,
credit: enabled
qos queue-shaper queue 7: disabled, rate 500 kbps, mode: line-rate, excess:
disabled, credit: disabled
```

4. Time Aware Shaper

Time Aware Shaper is defined in the IEEE 802.1Qbv standard and is the ability to allow transmission from each queue to be scheduled relative to a known global timescale.

The global time is maintained by using a specific version of Precision Time Protocol (PTP) as defined in IEEE 802.1AS-Rev.

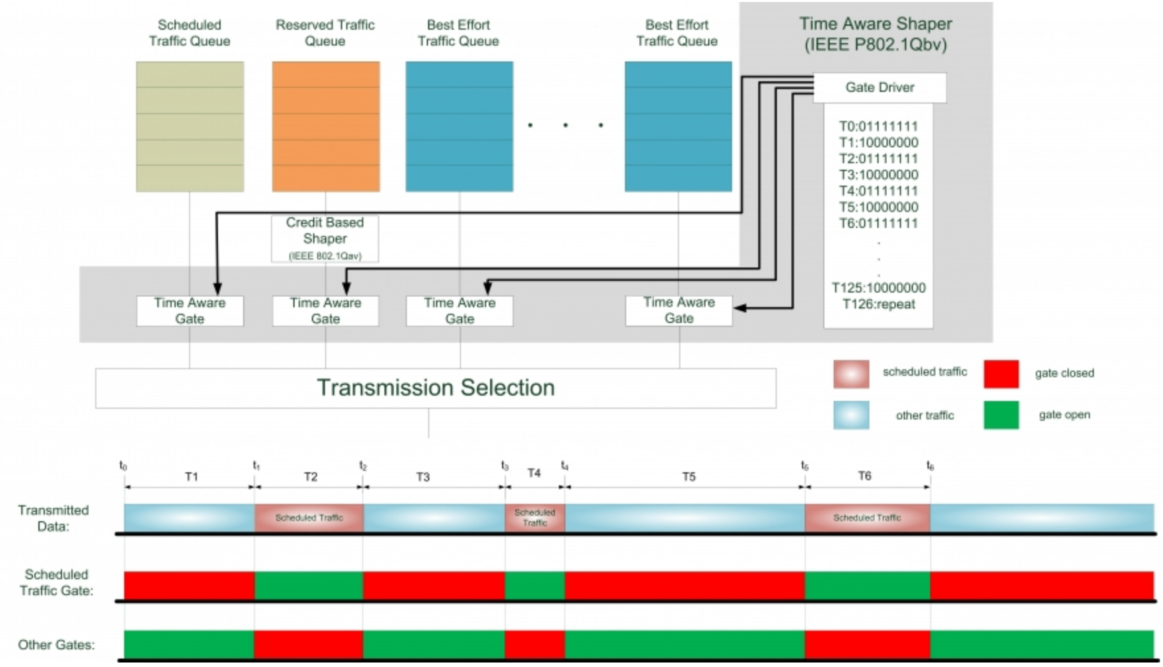


Figure 3. Time Aware Shaper

4.1. Configuration of Parameters

The syntax for TSN TAS global level CLI configuration command is:

```
tsn tas always-guard-band
no tsn tas always-guard-band
```

Where:

```
always-guard-band:    Guard band is implemented for any queue to scheduled queues
transition.
no always-guard-band: Guard band is implemented for non-scheduled queues to scheduled
                      queues transition.
```

The syntax for TSN TAS interface level CLI configuration command is:

```
tsn tas base-time seconds <seconds> nanoseconds <nanoseconds>
tsn tas config-change
tsn tas control-list index <index> gate-state queue <queue> {open | closed}
time-interval <interval> [operation {set | set-hold | set-release}]
tsn tas control-list-length <length>
tsn tas cycle-time <time> {ms | us | ns}
tsn tas cycle-time-extension <extension>
tsn tas gate-enabled
tsn tas gate-states queue <queue> {open | closed}
tsn tas max-sdu queue <queue> <sdu>
```

Where:

```
The following parameters are defined in IEEE802.1Q: ieee8021STMib

base-time                Admin Base Time.
config-change            Start a configuration change
control-list             Admin Control List
control-list-length      Admin Control List Length
cycle-time               Admin Cycle Time
cycle-time-extension     Admin Cycle Time Extension
gate-enabled             Enabled state of Time Aware Shaping
gate-states              Initial gate state for each queue
max-sdu                  Queue Max SDU configuration
queue                    Traffic class. 0-7.
index                    Index of Gate Control Entry
gate-state               Admin Gate State
time-interval            Time Interval in Nanoseconds
operation                set | set-hold | set-release
```

4.2. Configuration Example

To create and start a Time Aware Shaper schedule on port 1 where the schedule contains three gate control entries:

- 0: Open queue 7 and close all other queues for 20 milliseconds
- 1: Open queue 5-6 and close all other queues for 30 milliseconds
- 2: Open queue 0-4 and close all other queues for 50 milliseconds

The schedule is repeated every 110 milliseconds and the TAS is scheduled to start at seconds 4300 nanoseconds 500.

In the web, navigate to TSN → TAS → Ports and configure the following:

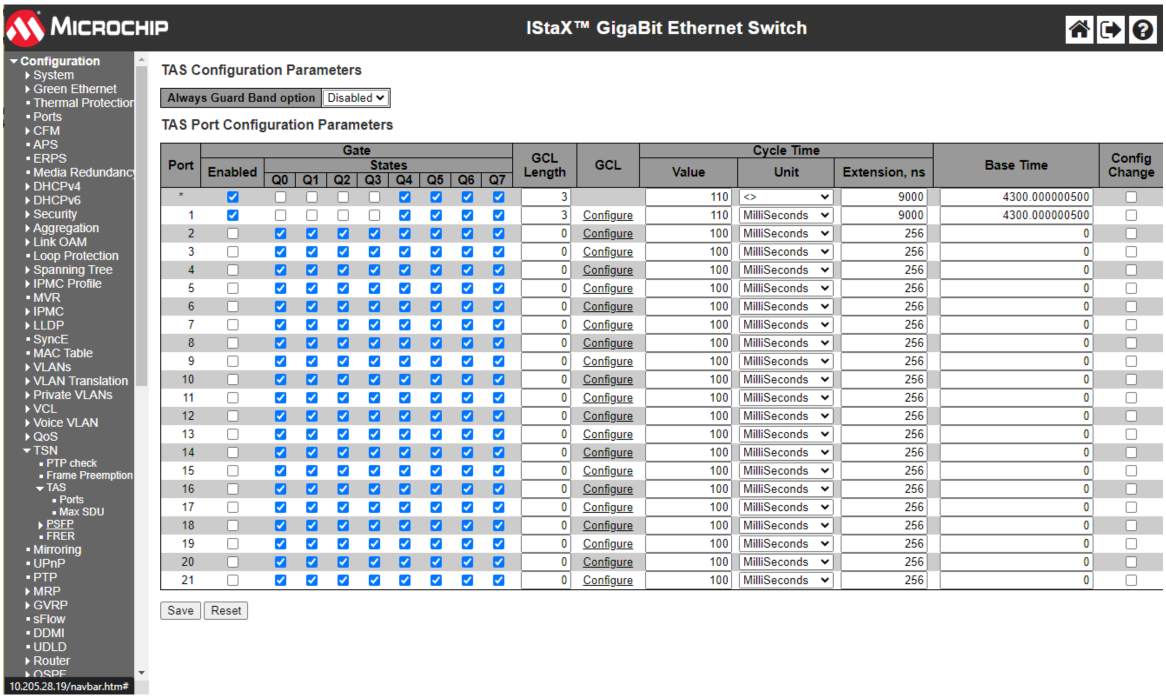


Figure 4. Configuration of TSN TAS ports

When the port configuration is saved, click on "configure" in the GCL column and configure the gate control list as follows:

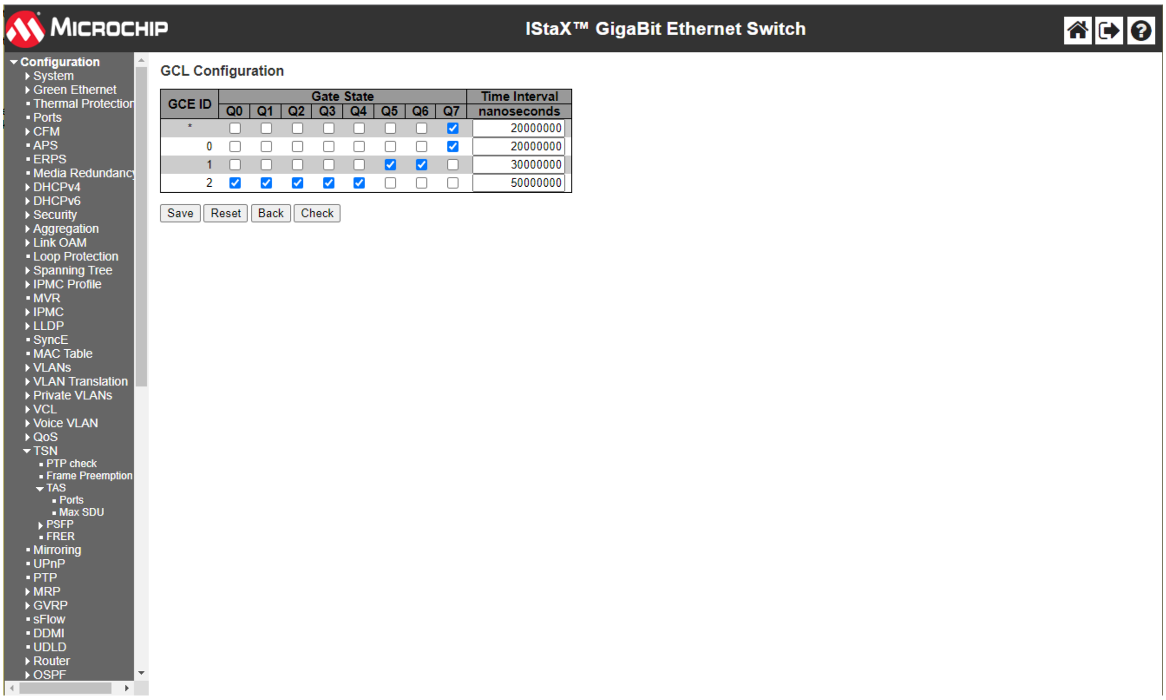


Figure 5. Configure the Gate Control List

Finally, when port 1 is configured, activate the configuration using the "Config Change" checkbox:

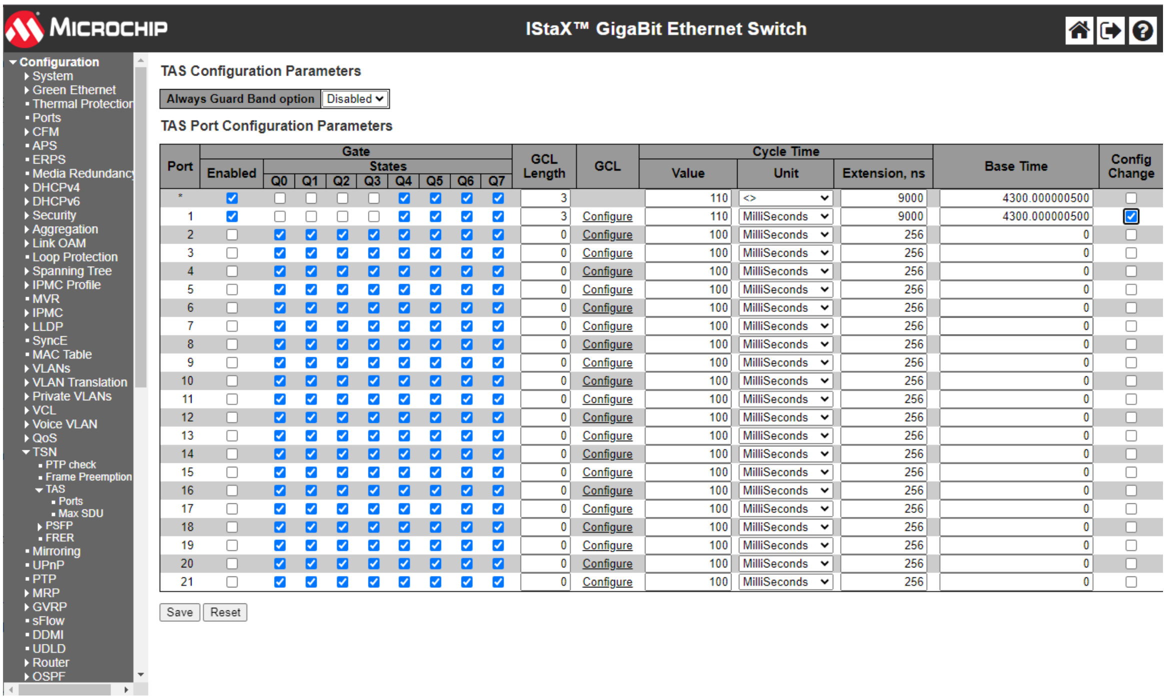


Figure 6. Apply the configuration for Port 1

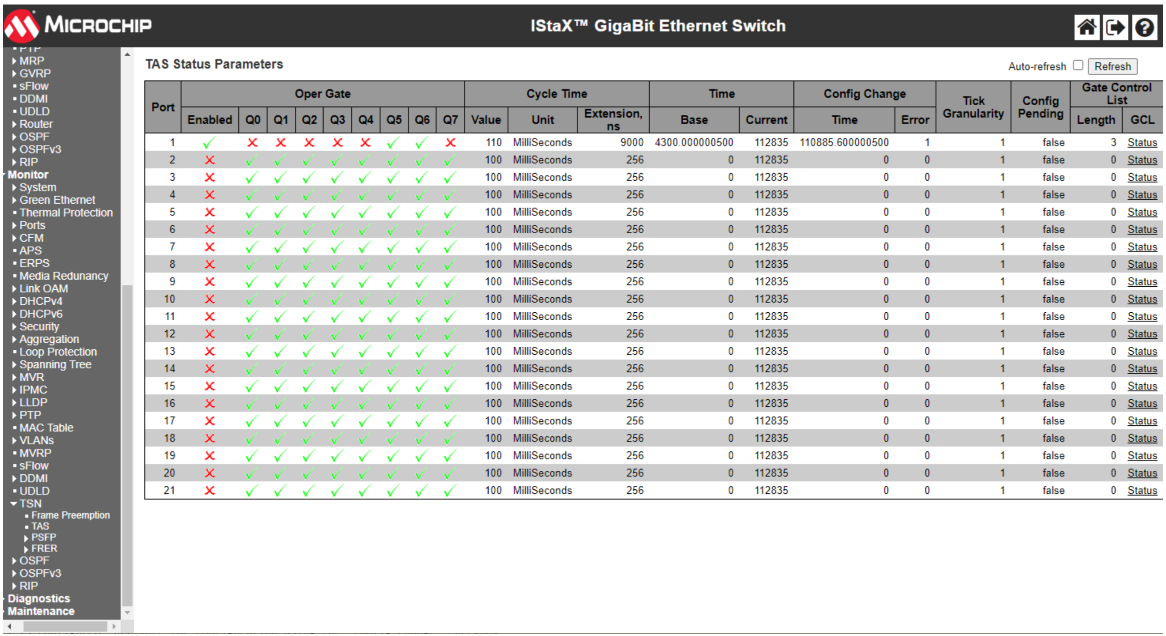
The equivalent CLI commands are:

```
! Create and start a Time Aware Shaper schedule on port 1.
! The schedule contains three gate control entries:
! 0: Open queue 7 and close all other queues for 20 milliseconds
! 1: Open queue 5-6 and close all other queues for 30 milliseconds
! 2: Open queue 0-4 and close all other queues for 50 milliseconds
! The schedule is repeated every 110 milliseconds
! TAS is scheduled to start at seconds 4300 nanoseconds 500

# configure terminal

! Disable always-guard-band
(config)# no tsn tas always-guard-band
(config)# interface 10GigabitEthernet 1/1
!
! Set max sdu size for queue 5 to 512 bytes
(config-if)# tsn tas max-sdu queue 5 512
!
! Enable Time Aware Shaper
(config-if)# tsn tas gate-enabled
!
! Set cycle-time to 110 milliseconds
(config-if)# tsn tas cycle-time 110 ms
!
! Set tsn tas cycle-time-extension 9000
(config-if)# tsn tas cycle-time-extension 9000
!
! Set gate state for queues 0-3 to closed
(config-if)# tsn tas gate-states queue 0-3 closed
!
! Set start time of schedule
(config-if)# tsn tas base-time seconds 4300 nanoseconds 500
(config-if)# !
!
! Configure gate control list
(config-if)# tsn tas control-list-length 3
(config-if)# tsn tas control-list index 0 gate-state queue 7 open time-interval
20000000 operation set-hold
(config-if)# tsn tas control-list index 1 gate-state queue 5-6 open time-interval
30000000 operation set-release
(config-if)# tsn tas control-list index 2 gate-state queue 0-4 open time-interval
50000000 operation set

! Start schedule
(config-if)# tsn tas config-change
```

```
! Show configuration using 'show running-config'.
# show running-config
no tsn tas always-guard-band
[...]
interface 10GigabitEthernet 1/1
 tsn tas max-sdu queue 5 512
 tsn tas gate-enabled
 tsn tas gate-states queue 4-7 open
 tsn tas cycle-time 110 ms
 tsn tas cycle-time-extension 9000
 tsn tas base-time seconds 4300 nanoseconds 500
 tsn tas control-list-length 3
 tsn tas control-list index 0 gate-state queue 7 open time-interval 20000000
operation set-hold
 tsn tas control-list index 1 gate-state queue 5,6 open time-interval 30000000
operation set-release
 tsn tas control-list index 2 gate-state queue 0-4 open time-interval 50000000
operation set
 tsn tas config-change
```

4.3. Status

The status of TAS can be found by navigating to Monitor → TSN → TAS:



Port	Oper Gate									Cycle Time			Time		Config Change		Tick Granularity	Config Pending	Gate Control List	
	Enabled	Q0	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Value	Unit	Extension, ns	Base	Current	Time	Error			Length	GCL
1	✓	✗	✗	✗	✗	✗	✓	✓	✗	110	MilliSeconds	9000	4300.000000500	112835	110885.600000500	1	1	false	3	Status
2	✗	✓	✓	✓	✓	✓	✓	✓	✓	100	MilliSeconds	256	0	112835	0	0	1	false	0	Status
3	✗	✓	✓	✓	✓	✓	✓	✓	✓	100	MilliSeconds	256	0	112835	0	0	1	false	0	Status
4	✗	✓	✓	✓	✓	✓	✓	✓	✓	100	MilliSeconds	256	0	112835	0	0	1	false	0	Status
5	✗	✓	✓	✓	✓	✓	✓	✓	✓	100	MilliSeconds	256	0	112835	0	0	1	false	0	Status
6	✗	✓	✓	✓	✓	✓	✓	✓	✓	100	MilliSeconds	256	0	112835	0	0	1	false	0	Status
7	✗	✓	✓	✓	✓	✓	✓	✓	✓	100	MilliSeconds	256	0	112835	0	0	1	false	0	Status
8	✗	✓	✓	✓	✓	✓	✓	✓	✓	100	MilliSeconds	256	0	112835	0	0	1	false	0	Status
9	✗	✓	✓	✓	✓	✓	✓	✓	✓	100	MilliSeconds	256	0	112835	0	0	1	false	0	Status
10	✗	✓	✓	✓	✓	✓	✓	✓	✓	100	MilliSeconds	256	0	112835	0	0	1	false	0	Status
11	✗	✓	✓	✓	✓	✓	✓	✓	✓	100	MilliSeconds	256	0	112835	0	0	1	false	0	Status
12	✗	✓	✓	✓	✓	✓	✓	✓	✓	100	MilliSeconds	256	0	112835	0	0	1	false	0	Status
13	✗	✓	✓	✓	✓	✓	✓	✓	✓	100	MilliSeconds	256	0	112835	0	0	1	false	0	Status
14	✗	✓	✓	✓	✓	✓	✓	✓	✓	100	MilliSeconds	256	0	112835	0	0	1	false	0	Status
15	✗	✓	✓	✓	✓	✓	✓	✓	✓	100	MilliSeconds	256	0	112835	0	0	1	false	0	Status
16	✗	✓	✓	✓	✓	✓	✓	✓	✓	100	MilliSeconds	256	0	112835	0	0	1	false	0	Status
17	✗	✓	✓	✓	✓	✓	✓	✓	✓	100	MilliSeconds	256	0	112835	0	0	1	false	0	Status
18	✗	✓	✓	✓	✓	✓	✓	✓	✓	100	MilliSeconds	256	0	112835	0	0	1	false	0	Status
19	✗	✓	✓	✓	✓	✓	✓	✓	✓	100	MilliSeconds	256	0	112835	0	0	1	false	0	Status
20	✗	✓	✓	✓	✓	✓	✓	✓	✓	100	MilliSeconds	256	0	112835	0	0	1	false	0	Status
21	✗	✓	✓	✓	✓	✓	✓	✓	✓	100	MilliSeconds	256	0	112835	0	0	1	false	0	Status

Figure 7. Status of TSN Tas

The equivalent CLI commands are:

```
! Show status using 'show tsn tas status'.
# show tsn tas status interface 10GigabitEthernet 1/1
interface
 GateEnabled           : TRUE
 OperGateStates        : 0x1f
 OperCycleTime         : 110 ms
 OperCycleTimeExtension: 9000 nanoseconds
 OperBaseTime          : 4300 seconds, 500 nanoseconds
 ConfigChangeTime      : 4300 seconds, 500 nanoseconds
 TickGranularity       : 0 tenths of nanoseconds
 CurrentTime           : 4311 seconds, 827669856 nanoseconds
 ConfigPending         : FALSE
 ConfigChangeError     : 0
 SupportedListMax      : 256
 OperControlListLength : 3
 GateControlEntry 0    : GateStates 0x80, TimeInterval 20000000 nanoseconds,
GateOperation set-hold
 GateControlEntry 1    : GateStates 0x60, TimeInterval 30000000 nanoseconds,
GateOperation set-release
 GateControlEntry 2    : GateStates 0x1f, TimeInterval 50000000 nanoseconds,
GateOperation set
```

```
! Disable Time Aware Shaper on port 1.
# configure terminal
(config)# interface 10GigabitEthernet 1/1
(config-if)# no tsn tas gate-enabled
```

4.4. Configuration Considerations

The `max-sdu` parameter is used only to calculate the guard band time:

`gbt = max-sdu * 8 / LINK_SPEED`.

Please note, that on SparX-5i, frames larger than max-sdu are not rejected.

The max-sdu is defined for each interface, queue and as a result, gbt’s can be configured for each traffic class on an interface.

The required guard band time can be reduced if preemption is used. If the traffic class being closed consist of preemptible frames, and the class being opened consists of express frames, then a set-hold operation can be included as part of the gate operation. This causes any currently transmitting preemptible frame to be preempted, reducing the latency before the port is ready to transmit express frames.

When a gate operation closes an input in a scheduler element, that input is permanently blocked until another gate operation opens it again. Similarly, a set-hold on a port remains in effect until another gate operation does a set-release.

This needs to be kept in mind when stopping a TAS list. If the last gate operation in the TAS list leaves any scheduler element input closed, or leaves a set-hold in effect, they can be left indefinitely, possibly causing frames to be blocked in the switch.

If one of the gate operations in a TAS list opens everything, then the TAS list can be arranged so that this is the last operation in the TAS list. A TAS list always completes its cycle before stopping, thus this leaves everything open after the TAS list is stopped.

Alternatively, after stopping a TAS list that leaves inputs closed or set-hold in effect, it is necessary to configure a dummy TAS list with an "open all" gate operation and run it for one cycle.

4.4.1. Configuration of Gate Control Entry

A Gate Control Entry (GCE) consists of 3 elements:

- gate-state: Specify for each queue wether the gate shall be open or closed in this interval.
- time-interval: The time in nanoseconds where the gate have the open/close state as defined by the gate-state parameter.
- operation: The value may be one of set, set-hold, set-release.
  These options are as defined by IEEE 802.1Q-2018, table 8-7.
  - set: The gates are immediately set to the states indicated in the gate-state parameter. After time-interval have elapsed control passes to the next gate operation.
  - set-hold: Performs all of the actions defined for the set operation. In addition, the start of this operation marks the point in the sequence of gate operations at which the MAC associated with the port is to have stopped transmitting preemptible frames. If frame preemption is not supported or not enabled, this operation behaves the same as set operation.
  - set-release: Performs all of the actions defined for the set operation. In addition, the start of this operation marks the point in the sequence of gate operations at which the MAC associated with the port is permitted to resume transmitting preemptible frames; if an express frame is currently being transmitted by the MAC, the release takes effect at the end of that transmission. If frame preemption is not supported or not enabled this operation behaves the same as set operation.

The value of a time-interval should always be larger than the guard band time (as specified thru the values of max-sdu and LINK_SPEED).

An open queue will always be opened for a small amount of time, even if the guard band time is larger than the configured time-interval.

There are some restrictions:

For a GCE with set-hold, all queues opened must be Express queues. For a GCE with set-release all queues opened must be Preemptable queues.

The same queue cannot be open in both a set-hold and a set-release operation.

4.4.2. Configuration of Gate Control List

A Gate Control List (GCL) is a list of gate control entries (GCE). A GCL is configured by the control-list parameter. The number of GCEs in a control-list is defined by control-list-length parameter.

When defining a control-list, start with setting control-list-length. Then configure each GCE. The sum of all time-interval in a control-list must be equal to or less than the cycle-time. Each queue must be open in at least one QCE.

4.4.3. Configuration of always-guard-band Option.

The always-guard-band defines how the guard band values are calculated and has the following effect:

If a GCL do not contain set-hold and/or set-release operations the always-guard-band has no effect. If a GCL do contain set-hold and set-release operations then:

- When always-guard-band=true a guard band is implemented on all queues, both Express and Preemptible queues.
- When always-guard-band=false a guard band is only implemented on Preemptible queues.

4.4.4. Calculation of Guard Band Times.

The Maximum SDU size parameter is used to calculate the guard band time:

```
gbt = max_sdu[] * 8 / LINK_SPEED
```

If frame preemption is enabled and a gate operaton is set-hold, the guard band time in preemptable queues is automatically selected as the frame preemption min fragment size plus 64 bytes.

NOTE	A queue is said to be preemptible, if frame preemption is enabled, and if this queue is not opened in a set-hold gate operation.

4.4.5. Using config-change and base-time

The command "tsn tas config-change" signals the start of a configuration change. If the value of parameter base-time is in the future, the configuration change will be executed at base-time. If base-time is in the past, the configuration change will be executed as soon as possible. In practice it will be within approx 2 seconds, at a time which is an integral number of cycle-times ahead of the configured value of base-time. This way, the synchronisation between schedules in elements across a sceduled network can be maintained.

4.4.6. Uncertainty Related to Last Frame in TAS Gate Open Interval

In the SparX-5i implementation of Time Aware Shaper function (IEEE 802.1Q-2018, Enhancements for Scheduled Traffic), frames that are buffered in the disassembler FIFO on the egress port when the TAS gate closes are transmitted after the gate close time Ts.

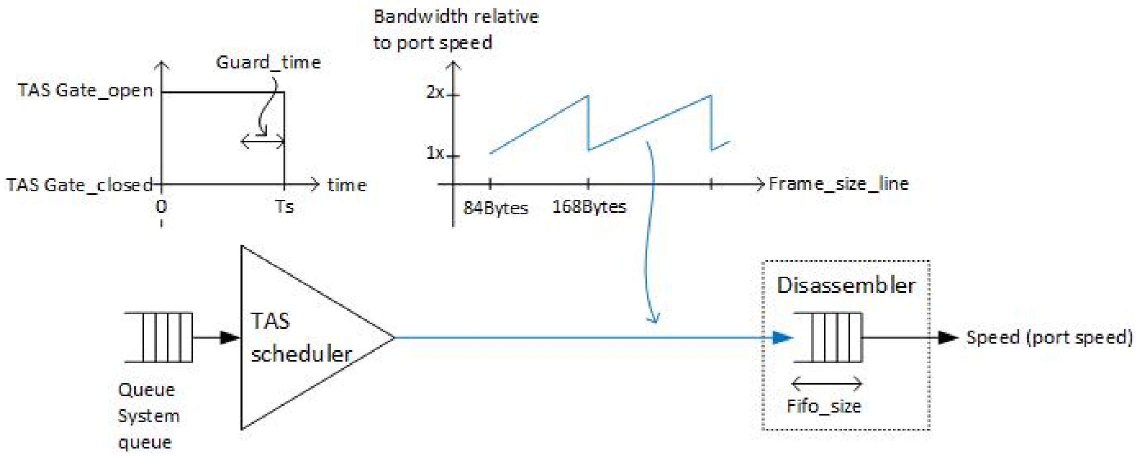


Figure 8. Time Aware Shaper

In a test where same size frames are transmitted using TAS, the maximum number of frames transmitted, N, is calculated as follows:

```
Ts: gate open time interval
max_sdu_line: configured value (including 8 bytes preamble + 12 bytes IFG)
speed: link speed
guard_time: max_sdu_line * 8 / speed
frame_size: size of frames sent
frame_size_line: size of frames sent including 8 bytes preamble + 12 bytes IFG
fifo_size: size of FIFO on the egress port. Depends on port speed
N = (Ts - guard_time) * speed/frame_size_line + Max(fifo_size/frame_size, 1)
```

Example:

```
Ts = 100 us
speed = 1000,000,000 bit/sec
frame_size = 148 bytes
frame_size_line = 168 bytes
max_sdu_line = 276 bytes
guard_time =  276 * 8 / 1000,000,000 = 2.208 us
fifo_size: 1024 bytes (1G interface)
N = (100 us - 2.208 us) * 1000,000,000 / (168*8) + Max(1024/148, 1)
N = 72.76 + 6.9 = 79.68 frames
```

Frames can be prevented from transmission after Ts by increasing the guard time to accommodate for the amount of traffic queued up in the disassembler FIFO as follows:

```
guard_time_safe * speed/frame_size_line =  Max(fifo_size/frame_size, 1) <=>
guard_time_safe =  Max(fifo_size/frame_size, 1) * frame_size_line/speed
```

Using the parameters from Example 1:

```
guard_time_safe = 6.9 * 168 bytes * 8bits/byte  / 1000,000,000 bit/sec = 9.3us
```

And thereby:

```
max_sdu_safe = guard_time_safe * speed / 8 bits/byte = 1162 bytes
max_sdu_line = 276 bytes + 1162 bytes = 1438 bytes.
```

5. Cut-Through

Cut-through is defined in the IEEE 802.1Qcc standard and is the ability to start forwarding a frame before the complete frame has been received.

This is in contrast to the normal store-and-forward mode where the complete frame is received and checked for errors before the forward decision is taken.

Cut-through is configured per egress port and per queue and is disabled by default which means that all frames are forwarded in store-and-forward mode.

Both unicast and multicast traffic is supported in cut-through mode.

Cut-through forwarding is only possible if a number of conditions are met:

- In case of multicast traffic, the cut-through decision is taken based on whether all the egress ports for a particular frame are enabled for cut-through.
- The speed of the egress port must be equal to or less than the speed of the ingress port(s).
- The egress port(s) must be idling. I.e. not currently transmitting anything.

If one or more of these conditions are not met, the frame is forwarded in store-and forward mode.

Cut-through can forward frames with errors, since the Frame Check Sequence (FCS) cannot be calculated before the entire frame has been received.

IMPORTANT Due to a hardware limitation, do not enable cut-through on ports that carry management IP traffic to the switch itself.

5.1. Configuration

To enable cut-through for port 1, queue 6 and 7, navigate to Configuration → QoS → Port Shaping, Port 1:

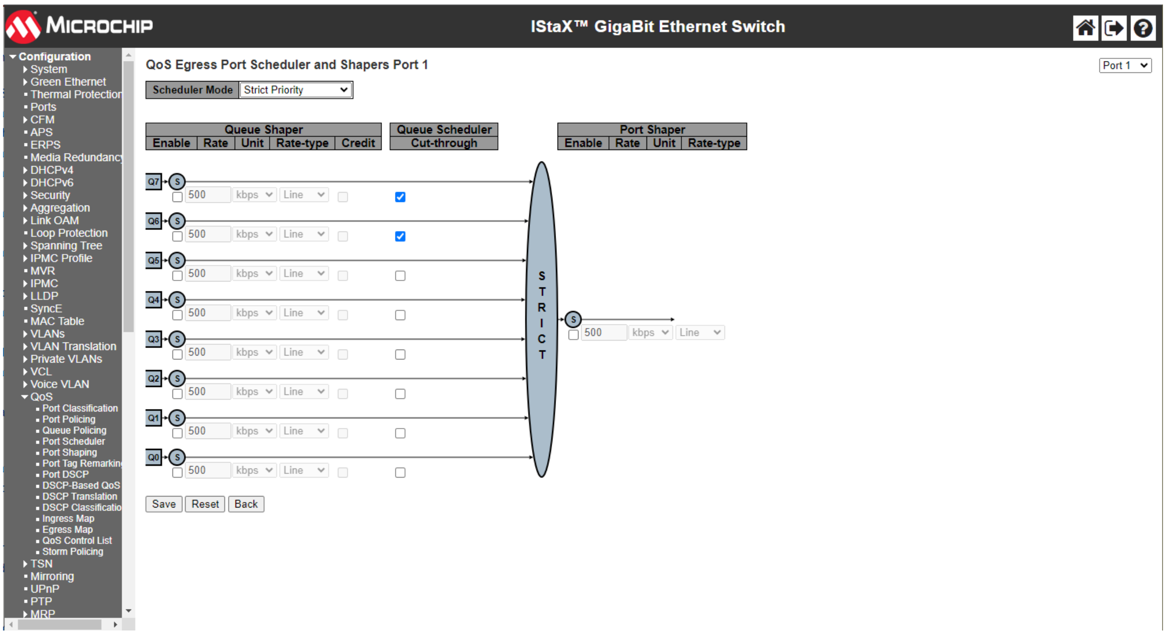


Figure 9. Enable Cut-through

The equivalent CLI commands are:

```
! Enable cut-through on port 1, queue 6 and 7.
# configure terminal
(config)# interface 10GigabitEthernet 1/1
(config-if)# qos cut-through queue 6-7
```

```
! Disable cut-through on port 1, queue 6 and 7.
# configure terminal
(config)# interface 10GigabitEthernet 1/1
(config-if)# no qos cut-through queue 6-7
```

```
! Show configuration using 'show running-config'.
# show running-config
[...]
qos cut-through queue 6
qos cut-through queue 7
```

```
! Show status using 'show qos'.
# show qos interface 10GigabitEthernet 1/1 | include cut-through
qos cut-through queue 0: disabled
qos cut-through queue 1: disabled
qos cut-through queue 2: disabled
qos cut-through queue 3: disabled
qos cut-through queue 4: disabled
qos cut-through queue 5: disabled
qos cut-through queue 6: enabled
qos cut-through queue 7: enabled
```

6. Frame Preemption

Frame Preemption is defined in the IEEE 802.1Qbu and IEEE 802.3br standards.

Frame Preemption is the ability to suspend the transmission of a non time-critical frame and allow for one or more time-critical frames to be transmitted. When the time-critical frames have been transmitted, the transmission of the non time-critical frame is resumed. A non time-critical frame could be preempted multiple times.

The use of Frame Preemption with Time Aware Shaping reduces the guard band needed from the size of the largest possible interfering frame to the size of the largest possible interfering fragment as shown below.

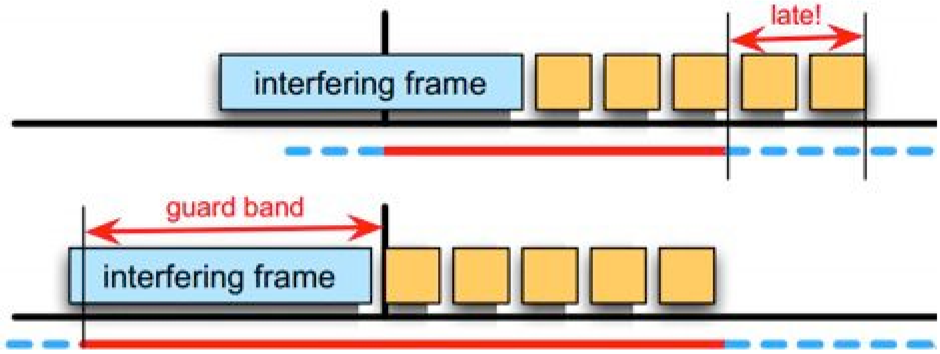


Figure 10. Time Aware Shaping without Frame Preemption

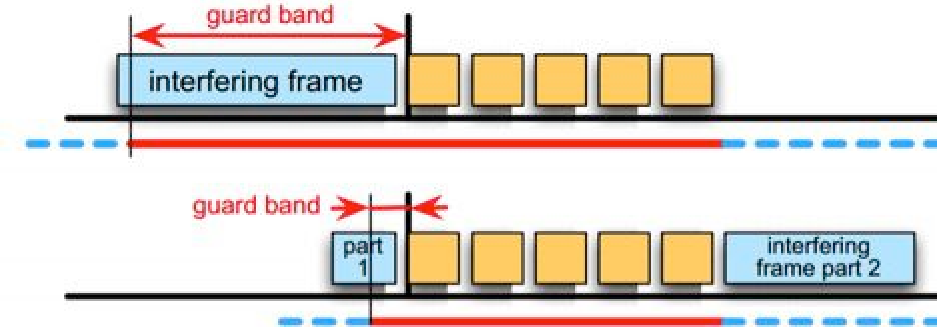


Figure 11. Time Aware Shaping with Frame Preemption

Frame Preemption must be enabled on both egress port level and on egress port/ queue level.

Frame Preemption is disabled by default on port level and disabled by default on port/ queue level.

6.1. Configuration

To configure frame preemption for port 1 queue 0-1, navigate to Configuration → TSN → Frame Preemption.

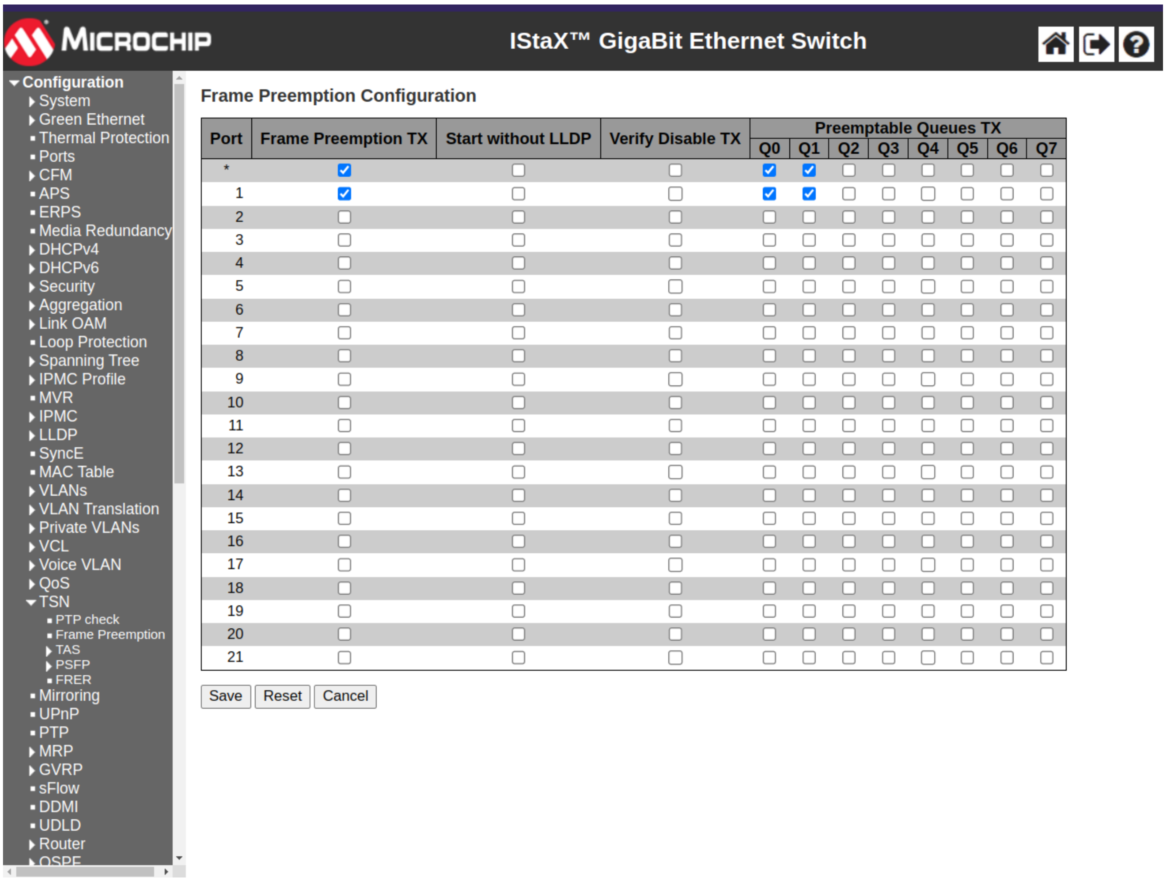


Figure 12. Configuration of Frame Preemption

Frame preemption capabilities are usually negotiated using LLDP. There may, however, be equipment that support frame premption without having implemented the LLDP negotiation. To accomodate for that case, it is possible to configure frame premention to start even when not negotiated through LLDP and it is possible to disable to LLDP negotiation if that is causing disturbance at the link partner.

Starting frame preemption without LLDP information from link partner is configured in the "Start without LLDP" field. Disabling of LLDP negotiation is configured in the field "Verify Disable TX".

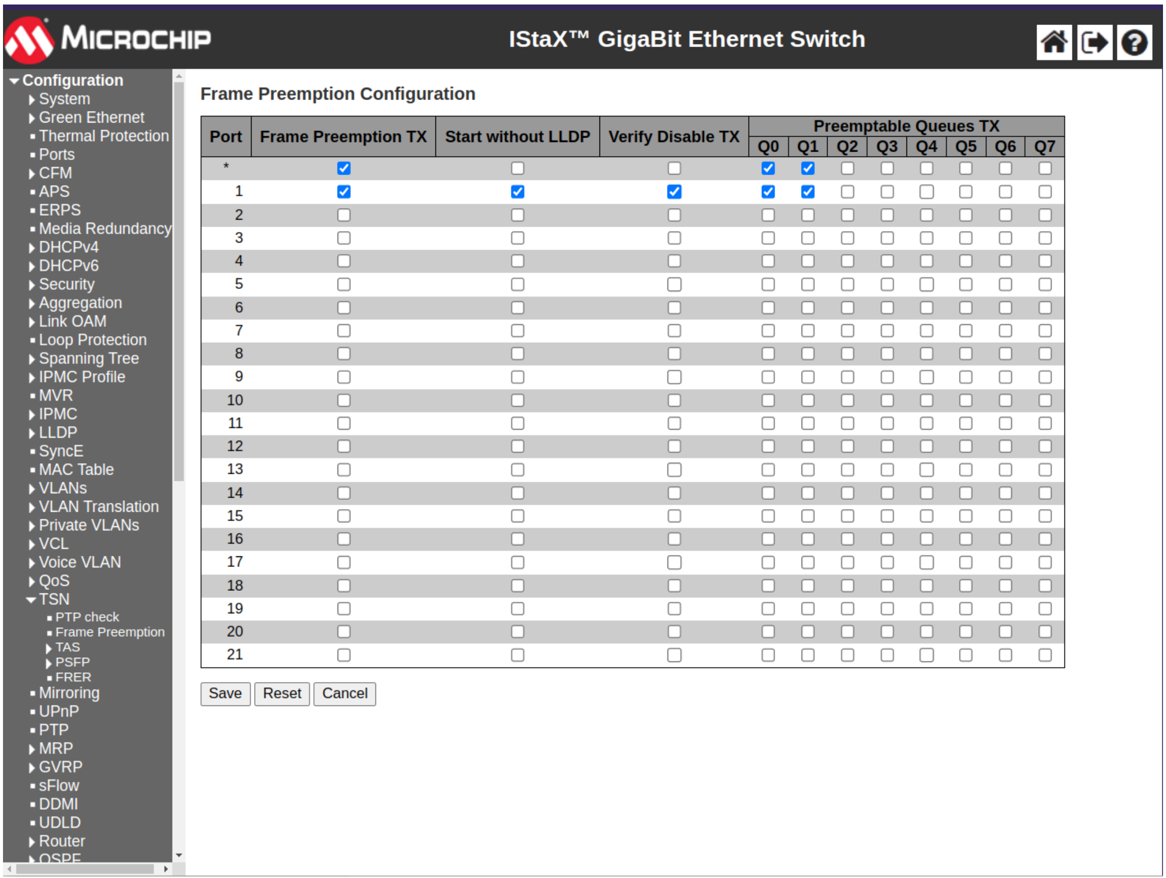


Figure 13. Configuration of Frame Preemption without LLDP

The equivalent CLI commands are:

```
! Enable frame-preemption on port level for port 1.
# configure terminal
(config)# interface 10GigabitEthernet 1/1
! Enable frame-preemption on port (disabled by default)
(config-if)# tsn frame-preemption
```

```
! Enable frame-preemption on queue level for port 1, queue 0 and 1.
# configure terminal
(config)# interface 10GigabitEthernet 1/1
! Enable frame-preemption on queue 0 and 1
(config-if)# tsn frame-preemption queue 0-1
```

```
! Disable verification of preemption capability of link partner.
# configure terminal
(config)# interface 10GigabitEthernet 1/1
! Disable verification of preemption capability of link partner.
(config-if)# tsn frame-preemption verify-disable
```

```
! Do not wait to receive lldp message before enabling frame-preemption in transmit
direction.
# configure terminal
(config)# interface 10GigabitEthernet 1/1
! Do not wait to receive lldp message before enabling frame-preemption in transmit
direction.
(config-if)# tsn frame-preemption ignore-lldp
```

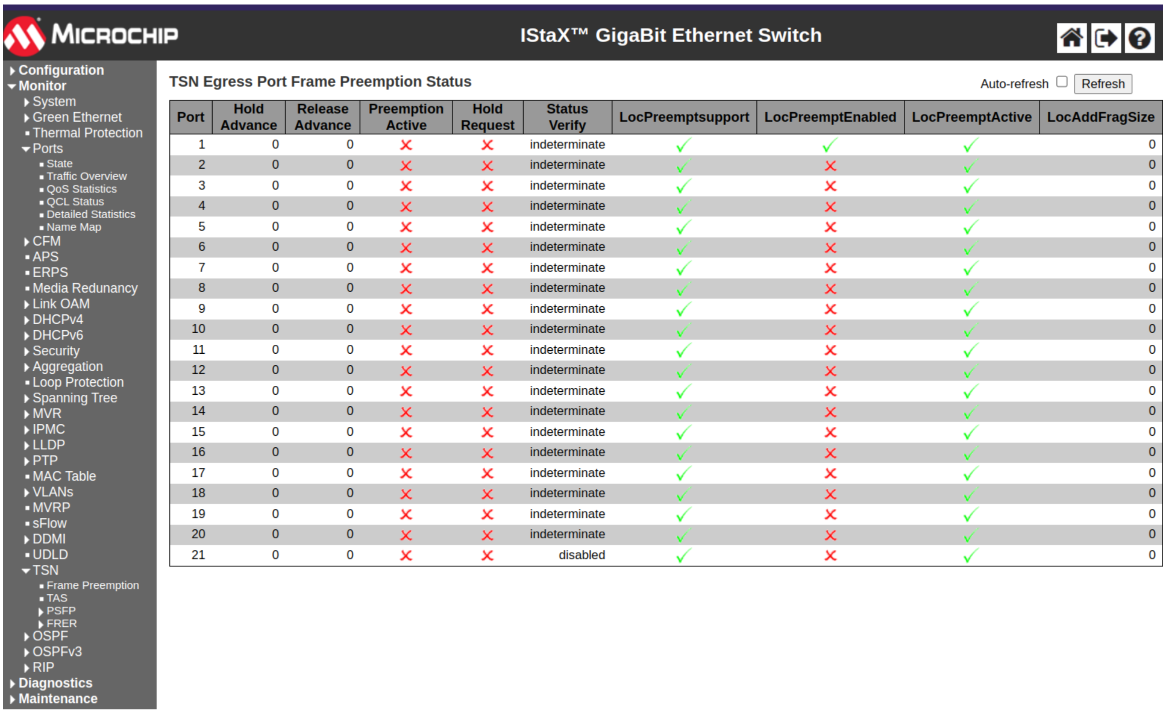
```
! Disable frame-preemption on port level for port 1
# configure terminal
(config)# interface 10GigabitEthernet 1/1
! Disable frame-preemption on port
(config-if)# no tsn frame-preemption
```

```
! Disable frame-preemption on on queue level for port 1, queue 0 and 1.
# configure terminal
(config)# interface 10GigabitEthernet 1/1
! Disable frame-preemption on queue 0 and 1
(config-if)# no tsn frame-preemption queue 0-1
```

```
! Show configuration using 'show running-config'.
! Note that frame-preemption on port level is disabled by default
! and not shown unless the 'all-defaults' option is used.
# show running-config
[...]
tsn frame-preemption queue 0
tsn frame-preemption queue 1
```

6.2. Status

To see status of frame preemption, navigate to Monitor → TSN → Frame Preemption



Port	Hold Advance	Release Advance	Preemption Active	Hold Request	Status Verify	LocPreemptsupport	LocPreemptEnabled	LocPreemptActive	LocAddFragSize
1	0	0	✗	✗	indeterminate	✓	✓	✓	0
2	0	0	✗	✗	indeterminate	✓	✗	✓	0
3	0	0	✗	✗	indeterminate	✓	✗	✓	0
4	0	0	✗	✗	indeterminate	✓	✗	✓	0
5	0	0	✗	✗	indeterminate	✓	✗	✓	0
6	0	0	✗	✗	indeterminate	✓	✗	✓	0
7	0	0	✗	✗	indeterminate	✓	✗	✓	0
8	0	0	✗	✗	indeterminate	✓	✗	✓	0
9	0	0	✗	✗	indeterminate	✓	✗	✓	0
10	0	0	✗	✗	indeterminate	✓	✗	✓	0
11	0	0	✗	✗	indeterminate	✓	✗	✓	0
12	0	0	✗	✗	indeterminate	✓	✗	✓	0
13	0	0	✗	✗	indeterminate	✓	✗	✓	0
14	0	0	✗	✗	indeterminate	✓	✗	✓	0
15	0	0	✗	✗	indeterminate	✓	✗	✓	0
16	0	0	✗	✗	indeterminate	✓	✗	✓	0
17	0	0	✗	✗	indeterminate	✓	✗	✓	0
18	0	0	✗	✗	indeterminate	✓	✗	✓	0
19	0	0	✗	✗	indeterminate	✓	✗	✓	0
20	0	0	✗	✗	indeterminate	✓	✗	✓	0
21	0	0	✗	✗	disabled	✓	✗	✓	0

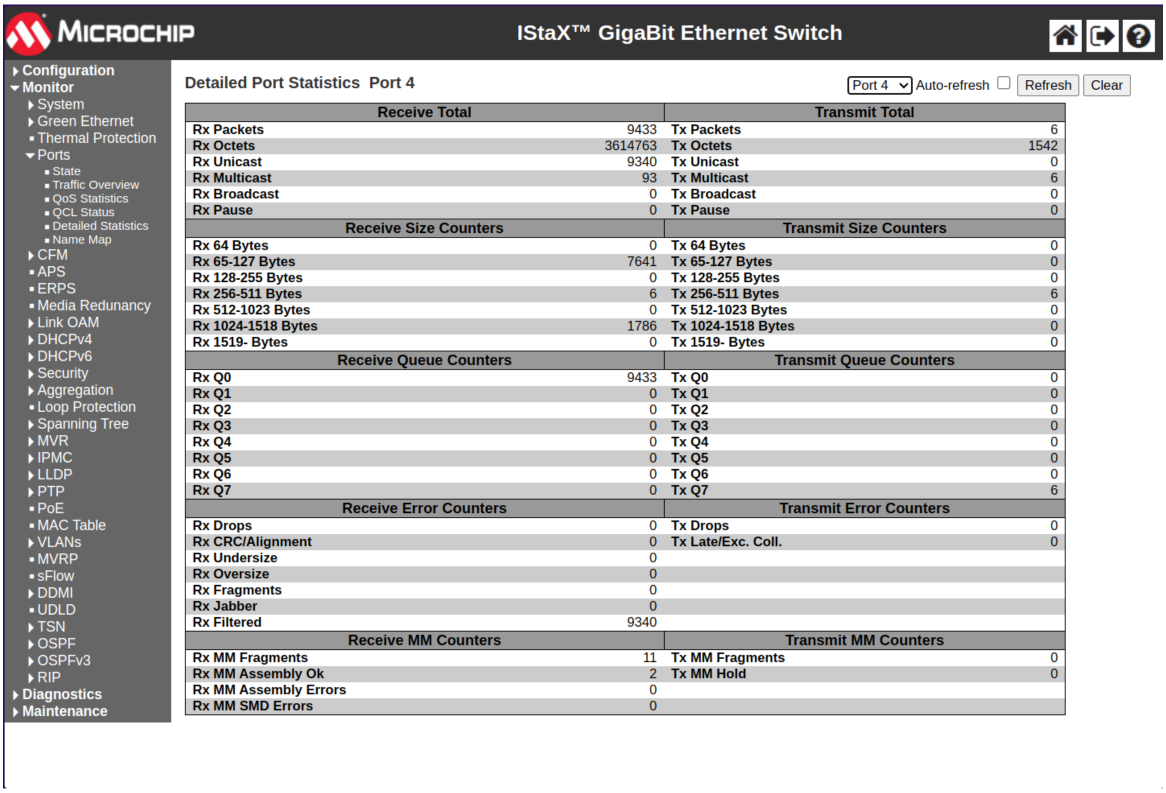
Figure 14. Status of Frame Preemption

The equivalent CLI commands are:

```
! Show frame preemption port status using 'show tsn frame-preemption status'.
# show tsn frame-preemption status interface 10GigabitEthernet 1/1
interface 10GigabitEthernet 1/1
 HoldAdvance          : 1016 nanoseconds
 ReleaseAdvance       : 1016 nanoseconds
 PreemptionActive     : FALSE
 HoldRequest          : FALSE
 StatusVerify         : disabled
 LocPreemptSupported : TRUE
 LocPreemptEnabled   : TRUE
 LocPreemptActive     : FALSE
 LocAddFragSize       : 0 (64 octets)
```

6.3. Statistics

To see statistics for frame preemption, navigate to Monitor → Ports → Detailed Statistics and select the port for which statistics shall be shown.



Receive Total		Transmit Total	
Rx Packets	9433	Tx Packets	6
Rx Octets	3614763	Tx Octets	1542
Rx Unicast	9340	Tx Unicast	0
Rx Multicast	93	Tx Multicast	6
Rx Broadcast	0	Tx Broadcast	0
Rx Pause	0	Tx Pause	0
Receive Size Counters		**Transmit Size Counters**	
Rx 64 Bytes	0	Tx 64 Bytes	0
Rx 65-127 Bytes	7641	Tx 65-127 Bytes	0
Rx 128-255 Bytes	0	Tx 128-255 Bytes	0
Rx 256-511 Bytes	6	Tx 256-511 Bytes	6
Rx 512-1023 Bytes	0	Tx 512-1023 Bytes	0
Rx 1024-1518 Bytes	1786	Tx 1024-1518 Bytes	0
Rx 1519- Bytes	0	Tx 1519- Bytes	0
Receive Queue Counters		**Transmit Queue Counters**	
Rx Q0	9433	Tx Q0	0
Rx Q1	0	Tx Q1	0
Rx Q2	0	Tx Q2	0
Rx Q3	0	Tx Q3	0
Rx Q4	0	Tx Q4	0
Rx Q5	0	Tx Q5	0
Rx Q6	0	Tx Q6	0
Rx Q7	0	Tx Q7	6
Receive Error Counters		**Transmit Error Counters**	
Rx Drops	0	Tx Drops	0
Rx CRC/Alignment	0	Tx Late/Exc. Coll.	0
Rx Undersize	0		
Rx Oversize	0		
Rx Fragments	0		
Rx Jabber	0		
Rx Filtered	9340		
Receive MM Counters		**Transmit MM Counters**	
Rx MM Fragments	11	Tx MM Fragments	0
Rx MM Assembly Ok	2	Tx MM Hold	0
Rx MM Assembly Errors	0		
Rx MM SMD Errors	0		

Figure 15. Statistics for Frame Preemption

During normal operation where traffic is being preempted, the counters "Rx MM Fragments" and "Rx MM Assembly OK" will be incrementing. The counter "Rx MM Fragments" counts the number of times a frame has been preempted (a large frame can be preempted more than once) while "Rx MM Assembly OK" counts the number of times fragments have been assembled. As large frames can be preempted more than once, the assembling of a frame may eat several fragments so in general "Rx MM Fragments" is expected to be large than "Rx MM Assembly OK".

The equivalent CLI command is:

```
# show interface GigabitEthernet 1/4 statistics
GigabitEthernet 1/4 Statistics:
Rx Packets:                         9603   Tx Packets:                          17
Rx Octets:                       3637147   Tx Octets:                         4369
Rx Unicast:                         9340   Tx Unicast:                           0
Rx Multicast:                        263   Tx Multicast:                        17
Rx Broadcast:                          0   Tx Broadcast:                         0
Rx Pause:                              0   Tx Pause:                             0

Rx 64:                                 0   Tx 64:                                0
Rx 65-127:                          7800   Tx 65-127:                            0
Rx 128-255:                            0   Tx 128-255:                           0
Rx 256-511:                           17   Tx 256-511:                          17
Rx 512-1023:                           0   Tx 512-1023:                          0
Rx 1024-1518:                       1786   Tx 1024-1518:                         0
Rx 1519-    :                          0   Tx 1519-    :                         0

Rx Priority 0:                      9603   Tx Priority 0:                        0
Rx Priority 1:                         0   Tx Priority 1:                        0
Rx Priority 2:                         0   Tx Priority 2:                        0
Rx Priority 3:                         0   Tx Priority 3:                        0
Rx Priority 4:                         0   Tx Priority 4:                        0
Rx Priority 5:                         0   Tx Priority 5:                        0
Rx Priority 6:                         0   Tx Priority 6:                        0
Rx Priority 7:                         0   Tx Priority 7:                       17

Rx Drops:                              0   Tx Drops:                             0
Rx CRC/Alignment:                      0   Tx Late/Exc. Coll.:                   0
Rx Undersize:                          0
Rx Oversize:                           0
Rx Fragments:                          0
Rx Jabbers:                            0
Rx Filtered:                        9340

Rx MM Fragments:                      11   Tx MM Fragments:                      0
Rx MM Assembly Ok:                     2   Tx MM Hold:                           0
Rx MM Assembly Errors:                 0
Rx MM SMD Errors:                      0
```

7. Streams and Stream Collections

A stream is an ingress property, where a subset of traffic gets identified by certain frame properties, such as DMAC, SMAC, VLAN tags, and layer 3 properties.

Streams are used by two other TSN protocols described later. One is `Per-Stream Filtering and Policing` and the other is `Frame Replication and Elimination for Reliability`.

Multiple streams can be bundled into a stream collection, which may be used in both PSFP and FRER, as we shall see later.

7.1. Stream Configuration

A stream is identified by an ID, which ranges from 1 to the maximum supported for the platform - typically 127.

A stream is instantiated at CLI’s global configuration level:

```
stream <inst>
```

Where:

```
inst    Stream instance number
```

When configuring a stream, the following list of stream configuration level CLI commands is available:

```
[no] dmac {<mac_addr> [/ <mac_addr>] | multicast | broadcast | unicast | not-broadcast | not-unicast |
any}
[no] smac {<mac_addr> [/ <mac_addr>] | any}
[no] outer-tag {none | vid {{<0-4095> [/ <uint16>] | any} [pcp <0-7> [/ <0-7>]] [dei <0-1>] [{c-tag |
s-tag}]}}
[no] inner-tag {none | vid {{<0-4095> [/ <uint16>] | any} [pcp <0-7> [/ <0-7>]] [dei <0-1>] [{c-tag |
s-tag}]}}
[no] etype <0x600-0xffff>
[no] llc <0x0-0xff> <0x0-0xff>
[no] snap {<0x0-0xffffff> | rfc-1042 | snap-8021h} <0x0-0xffff>
[no] ipv4 [sip {<ipv4_subnet> | any}] [dip {<ipv4_subnet> | any}] [dscp {<vcap_vr> | <dscp> | any}]
[fragment {yes | no | any}] [proto {<0-255> | tcp | udp | any}] [dport {<vcap_vr> | any}]
[no] ipv6 [sip {<ipv6_subnet> | any}] [dip {<ipv6_subnet> | any}] [dscp {<vcap_vr> | <dscp> |
any}]                        [proto {<0-255> | tcp | udp | any}] [dport {<vcap_vr> | any}]
```

In the following, the CLI help texts are used to describe the individual options for DMAC, SMAC, and VLAN tags.

```
# configure terminal
(config)# stream 1

! DMAC Options:
(config-stream)# dmac?
    dmac          Specify a destination MAC to match against incoming frames

(config-stream)# dmac ?
    <mac_addr>    A destination MAC address to match against incoming frames
    any           Match any destination MAC address
    broadcast     Match the broadcast destination MAC address
    multicast     Match any multicast destination MAC address (excluding broadcast)
    not-broadcast Match any MAC address, except the broadcast MAC address
    not-unicast   Match any multicast or the broadcast destination MAC address
    unicast       Match any unicast MAC address

(config-stream)# dmac 00:00:00:00:00:01 ?
    /             Specify a mask. If no mask is specified, all bits of the destination MAC address
shall
                  match the incoming frame
    <cr>

(config-stream)# dmac 00:00:00:00:00:01 / ?
    <mac_addr>    A mask in the form xx:xx:xx:xx:xx:xx, that specifies which bits of the destination
MAC
                  address that shall match the incoming frames. Default is to match all 48 bits

(config-stream)# no dmac?
    dmac          Don't match on incoming destination MAC address

! SMAC Options:
(config-stream)# smac?
    smac          Specify a source MAC mask to match against incoming frames

(config-stream)# smac ?
    <mac_addr>    A source MAC address to match against incoming frames
    any           Match any source mac address

(config-stream)# smac 00:00:00:00:00:01 ?
    /             Specify a mask. If no mask is specified, all bits of the source MAC address shall
                  match the incoming frame

(config-stream)# smac 00:00:00:00:00:01 / ?
    <mac_addr>    A mask in the form xx:xx:xx:xx:xx:xx, that specifies which bits of the source MAC
                  address that shall match the incoming frames. Default is to match all 48 bits

(config-stream)# no smac?
    smac          Don't match on incoming source MAC address

! Outer VLAN Tag Options:
(config-stream)# outer-tag?
    outer-tag     Configuration of an outer tag to match against incoming frames

(config-stream)# outer-tag ?
    none          The frame must be untagged
    vid           The frame must be tagged. The next keyword tells whether all VLANs are matched or
                  only a specific with an optional mask

(config-stream)# outer-tag vid ?
    0-4095        VLAN ID to match incoming frames against
    any           Match any incoming VLAN ID

(config-stream)# outer-tag vid 17 ?
    /             Specify a mask. If no mask is specified, all bits of the VLAN ID shall match the
                  incoming frame
    c-tag         If specified, only match C-tagged frames (EtherType = 0x8100)
    dei           Specify a DEI value to match the incoming frame against
```

```
    pcp           Configuration of a PCP value to match against incoming frames
    s-tag         If specified, only match S-tagged frames (EtherType = 0x88a8)
    <cr>

(config-stream)# outer-tag vid 17 / ?
    <uint16>      A mask specified as an integer, that specifies which bits of the VLAN ID that
                  shall match the incoming frames

(config-stream)# outer-tag vid any pcp ?
    <0-7>         The PCP value to match the incoming frames against

(config-stream)# outer-tag vid any pcp 6 ?
    /             Configuration of a mask. If no mask is specified, all bits of the PCP value shall
                  match the incoming frame

(config-stream)# outer-tag vid any pcp 6 / ?
    <0-7>         A mask that specifies the bits of the PCP value that shall match the incoming frame

(config-stream)# outer-tag vid any dei ?
    <0-1>         The DEI value that shall match the incoming frame

(config-stream)# no outer-tag?
    outer-tag     Don't use outer tag for matching. It may be both tagged and untagged

! Inner VLAN Tag Options are identical to Outer VLAN Tag Options.
! However, it is not possible to specify an inner-tag if outer-tag is set to
! none.
```

Layer 3 properties are mutually exclusive in the sense that it is not possible, for example, to match both IPv4 and IPv6 frame properties in the same stream.

This also means that if you start by specifying e.g. an `ipv4` line and then specify an `ipv6` line, the `ipv4` line will disappear silently.

```
# configure terminal
(config)# stream 1

! EtherType options:
(config-stream)# etype?
    etype          Match EtherType frames

(config-stream)# etype ?
    <0x600-0xffff> Matched EtherType

(config-stream)# no etype?
    etype          Don't match incoming frames' EtherType

! Logical Link Control frames:
(config-stream)# llc?
    llc            Match Logical Link Control (LLC) frames, i.e. frames with
                   EtherType/TypeLength field less than 0x600

(config-stream)# llc ?
    <0x0-0xff>     Matched LLC Destination Service Access Point (DSAP)

(config-stream)# llc 0xab ?
    <0x0-0xff>     Matched LLC Source Service Access Point (SSAP)

(config-stream)# no llc?
    llc            Don't match LLC frames

! SubNetwork Access Protocol frames:
(config-stream)# snap?
    snap           Match Subnetwork Access Protocol (SNAP) frames, i.e. frames with
                   EtherType/TypeLength field less than 0x600 and DSAP = 0xaa and SSAP = 0xAA
                   and Control field = 0x03

(config-stream)# snap ?
    <0x0-0xffffff> SNAP OUI (Range 0x000000 - 0XFFFFFF)
    rfc-1042       SNAP OUI is speficied in RFC1042, that is, 00:00:00
    snap-8021h     SNAP OUI is specified in 802.1H, that is, 00:00:F8

(config-stream)# snap rfc-1042 ?
    <0x0-0xffff>   Protocol ID (Range: 0x0 - 0xFFFF). If OUI is all-zeros (rfc-1042),
                   then this must be a valid EtherType (>= 0x600)

(config-stream)# no snap?
    snap           Don't match SNAP frames

! IPv4 frames:
(config-stream)# ipv4?
    ipv4           Match IPv4 frames

(config-stream)# ipv4 ?
    dip            Match on destination IPv4 address
    dport          Setup matching on UDP/TCP destination port
    dscp           Match on DSCP
    fragment       Setup matching on IPv4 fragments
    proto          Match on IP protocol
    sip            Match on source IPv4 address
    <cr>

(config-stream)# ipv4 sip ?
    <ipv4_subnet>  Match on source IPv4 address/mask, e.g. 1.2.3.4/32 or 1.2.0.0/16
    any            Match on any source IPv4 address

(config-stream)# ipv4 dip ?
    <ipv4_subnet>  Match on destination IPv4 address/mask, e.g. 1.2.3.4/32 or 1.2.0.0/16
    any            Match on any destination IPv4 address

(config-stream)# ipv4 dscp ?
```

```
    <vcap_vr>      Matched DSCP value/range (e.g. 17-33 or 17)
    af11           Assured Forwarding PHB AF11 (DSCP 10)
    af12           Assured Forwarding PHB AF12 (DSCP 12)
    af13           Assured Forwarding PHB AF13 (DSCP 14)
    af21           Assured Forwarding PHB AF21 (DSCP 18)
    af22           Assured Forwarding PHB AF22 (DSCP 20)
    af23           Assured Forwarding PHB AF23 (DSCP 22)
    af31           Assured Forwarding PHB AF31 (DSCP 26)
    af32           Assured Forwarding PHB AF32 (DSCP 28)
    af33           Assured Forwarding PHB AF33 (DSCP 30)
    af41           Assured Forwarding PHB AF41 (DSCP 34)
    af42           Assured Forwarding PHB AF42 (DSCP 36)
    af43           Assured Forwarding PHB AF43 (DSCP 38)
    any            Match any DSCP
    be             Default PHB (DSCP 0) for best effort traffic
    cs1            Class Selector PHB CS1 precedence 1 (DSCP 8)
    cs2            Class Selector PHB CS2 precedence 2 (DSCP 16)
    cs3            Class Selector PHB CS3 precedence 3 (DSCP 24)
    cs4            Class Selector PHB CS4 precedence 4 (DSCP 32)
    cs5            Class Selector PHB CS5 precedence 5 (DSCP 40)
    cs6            Class Selector PHB CS6 precedence 6 (DSCP 48)
    cs7            Class Selector PHB CS7 precedence 7 (DSCP 56)
    ef             Expedited Forwarding PHB (DSCP 46)
    va             Voice Admit PHB (DSCP 44)

(config-stream)# ipv4 fragment ?
    any            Match any values of IPv4 header's MF bit and fragment offset value
    no             Match IPv4 headers with MF bit cleared and fragment offset 0
    yes            Match IPv4 headers with MF bit set or a fragment offset > 0

(config-stream)# ipv4 proto ?
    <0-255>        Match a custom IP protocol number
    any            Match any IP protocol
    tcp            Match TCP frames (protocol number 6)
    udp            Match UDP frames (protocol number 17)

(config-stream)# ipv4 dport ?
    <vcap_vr>      Match UDP/TCP destination port value/range (e.g. 123-345 or 123)
    any            Match any UDP/TCP destination port

! IPv6 frames are just like IPv4 frames, except that it's not possible to match
fragments (and IP addresses are 128 bits rather than 32 :-)).
```

If a stream is configured to match multiple properties, e.g. DMAC and IPv4, all must match in the incoming stream for the stream to be hit.

Streams are added to hardware in ID order. This means that a lower numbered stream has higher priority in the matching process than higher numbered streams.

Example:

```
# configure terminal
(config)# stream 1
(config-stream)# dmac 00:00:00:00:00:00
(config-stream)# stream 2
(config-stream)# dmac 00:00:00:00:00:00 / FF:FF:FF:FF:FF:00
```

Stream 1 matches DMAC address `00:00:00:00:00:00` only.
Stream 2 matches DMAC addresses `00:00:00:00:00:00` through `00:00:00:00:00:FF`.

Since stream 1 comes before stream 2, stream 2 can never be hit by a frame with DMAC `00:00:00:00:00:00`, so essentially, stream 2 only matches DMAC addresses `00:00:00:00:00:01` through `00:00:00:00:00:FF`, which is not possible to specify in one single stream.

Whether or not PSFP and/or FRER uses a stream, the stream will be added to hardware. In the above example, it could make sense to have stream 1 just added to hardware and let PSFP and/or FRER attach to stream 2.

If stream 1 and stream 2 from the example were swapped, stream 2 could never be hit.

Streams must be added to one or more port interfaces before they really take effect.

TIP It is always a good idea to make room for additional streams in between other streams. So start the first stream you create with e.g. ID 10 and space subsequent streams accordingly.

7.2. Stream Collection Configuration

Stream collections may be used by PSFP and FRER to aggregate multiple streams into the same PSFP filter or FRER instance.

Configuration is straight forward:

A stream collection is identified by an ID, which ranges from 1 to half the number of supported streams (see previous section).

A stream collection is instantiated at CLI’s global configuration level:

```
stream-collection <inst>
```

Where:

```
inst    Stream collection instance number
```

Configuration of a stream collection goes like this:

```
# configure terminal
(config)# stream-collection 1
(config-stream-collection)# stream-id-list ?
    <1~127>   List of stream IDs. This indirectly gives the ingress ports. Example: "1-3,17"

(config-stream-collection)# exit
(config)# no stream-collection ?
    <1~63>    Delete one or more stream collections
    all       Delete all stream collections
```

Example: Below, we configure three streams (1, 2 and 3) on Gi 1/1 and assign them to stream collection 1.

```
# configure terminal
(config)# stream 1
(config-stream)# ipv4 dip 1.2.3.4/32
(config-stream)# stream 2
(config-stream)# ipv4 sip 4.3.0.0/16
(config-stream)# stream 3
(config-stream)# ipv6 dip 2001:db8::/32
(config-stream)# stream-collection 1
(config-stream-collection)# stream-id-list 1-3
(config-stream-collection)# interface GigabitEthernet 1/1
(config-if)# stream-id 1-3
(config-if)# end
#
```

It is not possible to specify a `stream-id-list` containing streams that are not yet created.

Likewise, if a stream is part of a stream collection and the stream gets deleted, it silently gets removed from the stream collection as well.

A stream that is part of a stream-collection cannot be used directly in PSFP and FRER. If attempting to, a configurational warning inside those modules will be issued.

7.3. Stream and Stream Collection Status

Stream status is shown in CLI EXEC mode with the command `show stream <1~127> status [details]`.

Example:

```
# show stream status
Stream ID Warnings Attached Clients (ID)
--------- -------- --------------------------
        1 No       Part of a stream collection
        2 No       Part of a stream collection
        3 No       Part of a stream collection
        4 No       PSFP (2), FRER (1)
        5 YES!
```

This shows that the first three streams are part of one more more stream collections, and can’t be attached to directly.

Stream 4 is connected to by both PSFP (instance 2) and FRER (instance 1).

Stream 1-4 do not have any configurational warnings, whereas stream 5 does. To catch the reader’s eye, warnings are written in capitals followed by an exclamation mark.

To see the warnings, add the `details` keyword to the command:

```
# show stream 5 status details
Stream ID:                5
Configurational Warnings: The stream does not have any member ports
PSFP:                     Not attached
FRER:                     Not attached
```

This shows the only configurational warning a stream (at the time of writing) can have: That it is not instantiated on any interfaces. To mend this, add it to at least one port interface, e.g.:

```
# configure terminal
(config)# interface GigabitEthernet 1/1,7
(config-if)# stream-id 5
(config-if)# end
# show stream 5 status details
Stream ID:                5
Configurational Warnings: None
PSFP:                     Not attached
FRER:                     Not attached
```

Stream collections also have status. This can be shown in CLI EXEC mode with `show stream-collection <1~63> status [details]`.

Example:

```
# show stream-collection status
Stream Coll. ID Warnings Attached Clients (ID)
--------------- -------- ---------------------
              1 No       PSFP (1)
              2 YES!
```

Here, PSFP stream filter instance 1 is attached to stream collection 1, whereas stream collection 2 doesn't have any attached clients but configurational warnings. To see those warnings, add the `details` keyword to the command:

```
# show stream-collection 2 status details
Stream Collection ID:     2
Configurational Warnings: No streams attached
                          No clients attached
PSFP:                     Not attached
FRER:                     Not attached
```

The configurational warnings are displayed on separate lines. Here, they indicate that no streams are aggregated into the stream collection and that neither PSFP nor FRER is connected to the stream collection, implying that the stream collection is not useful.

Possible configurational warnings for stream collections are:

- `No streams attached`: The stream collection is empty.

- `No clients attached`: Neither PSFP nor FRER is using this stream collection.
- `At least one of the attached streams has configurational warnings`: Use `show stream status details` to see those warnings.

8. Per-Stream Filtering and Policing

Per-Stream Filtering and Policing (PSFP), as defined in the IEEE 802.1Qci standard, provides filtering, policing and service class selection for a stream.

A PSFP stream filter references sub-components to make up the entire stream filter. Sub-components are:

- a mandatory stream or stream collection,
- an optional flow meter that defines the policing behaviour, and
- an optional stream gate that defines when the gate towards the egress queues is open and closed.

A stream or stream collection may only be referenced by one stream filter.
Both flow meters and stream gates may be referenced by more than one stream filter.

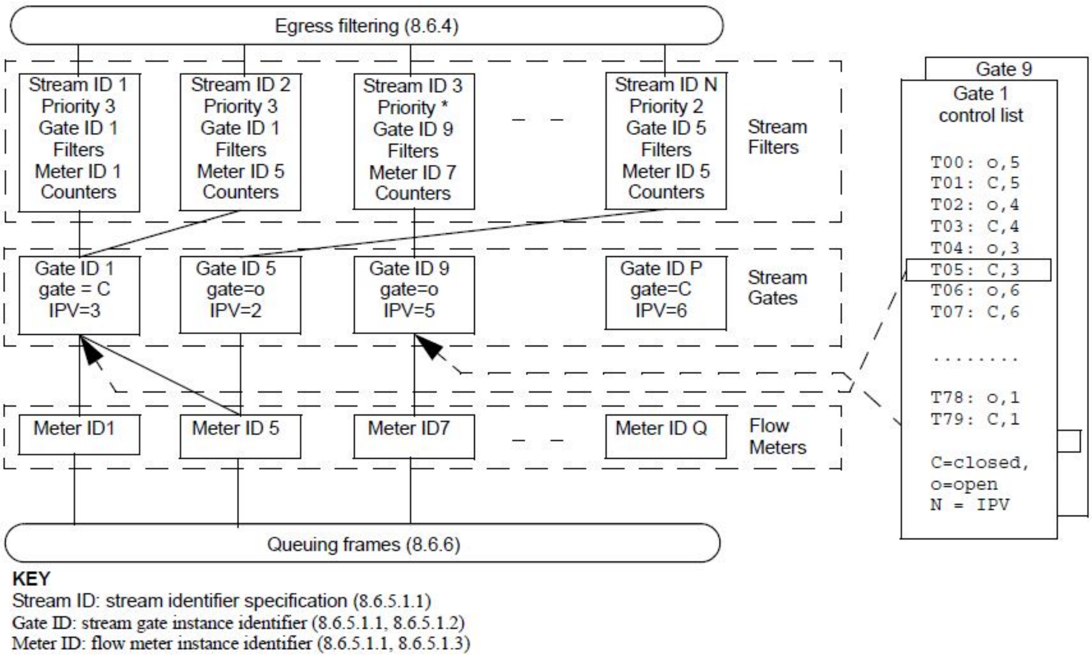


Figure 16. Per-Stream Filtering and Policing

The following sections take you through configuration of the PSFP-related sub-components.

8.1. Flow Meter Configuration

Flow meters are identified by an ID ranging from 0 to a platform specific value.

A flow meter is instantiated at CLI’s global configuration level:

```
! The range is platform specific. This is for LAN9668:
tsn flow meter <0-255>
```

Where:

```
<0-255>  Flow meter instance number
```

Flow meters can be removed with the `no tsn flow meter <0-255>` command.

When configuring a flow meter, the following flow meter configuration level CLI commands are available:

```
[no] cir <uint>
[no] cbs <uint>
[no] eir <uint>
[no] ebs <uint>
[no] coupling-flag
[no] color-mode
[no] drop-on-yellow
[no] mark-red-enable
```

In the following, the CLI help texts are used to describe the individual options.

```
# configure terminal
(config)# tsn flow meter 1
(config-flow-meter)# cir ?
    <uint>     Committed Information Rate measured in kbps. Gets rounded up to the nearest value
               supported by the policer and will be reflected in running-config
(config-flow-meter)# cbs ?
    <uint>     Committed Burst Size measured in bytes. Gets rounded up to the nearest value
               supported by the policer and will be reflected in running-config
(config-flow-meter)# eir ?
    <uint>     Excess Information Rate measured in kbps. Gets rounded up to the nearest value
               supported by the policer and will be reflected in running-config
(config-flow-meter)# ebs ?
    <uint>     Excess Burst Size measured in bytes. Gets rounded up to the nearest value
               supported by the policer and will be reflected in running-config
(config-flow-meter)# coupling-flag?
    <boolean>  Coupling flag. When set, frames that would overflow the committed bucket
               will be added to the excess bucket unless it's full
(config-flow-meter)# color-mode?
   <boolean>   Color mode. When cleared (no-form), the frame starts green, when set, the frame
               starts at the classified color based on its DEI value
(config-flow-meter)# drop-on-yellow?
   <boolean>   If cleared (no-form), frames will have their DEI value set to 1,
               otherwise frames marked yellow are discarded
(config-flow-meter)# mark-red-enable?
   <boolean>   If set, all subsequent frames are discarded if a red frame is seen
```

In the following example, we configure flow meter 20 with a committed information rate of 400 kbps and a committed burst size of 8192 bytes. If frames are exceeding the configured rate, we block the stream permanently (`mark-red-enable`):

```
# configure terminal
(config)# tsn flow meter 20
(config-flow-meter)# cir 400
(config-flow-meter)# cbs 8192
(config-flow-meter)# mark-red-enable
(config-flow-meter)# end
#
```

Flow meters don't have status or statistics along with them.

8.2. Stream Gate Configuration

Stream gates are identified by an ID ranging from 0 to a platform specific value.

A stream gate is instantiated at CLI's global configuration level:

```
! The range is platform specific. This is for LAN9668:
tsn stream gate <0-255>
```

Where:

```
<0-255>  Stream Gate instance number
```

Stream gates can be removed with the `no stream gate <0-255>` command.

When configuring a stream gate, the following stream gate configuration level CLI commands are available:

```
[no] state {open | closed}
[no] ipv <0-7>
[no] close-due-to-invalid-rx-enable
[no] close-due-to-octets-exceeded-enable
[no] cycle-time <1-1000000000> {ms | us | ns}
[no] time-extension <1-1000000000> {ms | us | ns}
[no] base-time seconds <0-4294967295> [nanoseconds <0-999999999>]
     control-list-length <uint>
[no] control-list index <uint> gate-state {open | closed} time-interval <1-999999999>
                  {ms | us | ns} [ipv <uint>] [octet-max <uint>]
[no] enable
     config-change
```

In the following, the CLI help texts are used to describe the individual options.

```
# configure terminal
(config)# tsn stream gate 1
(config-stream-gate)# state ?
    closed         Set initial gate state to closed
    open           Set initial gate state to open
(config-stream-gate)# ipv ?
    <0-7>          Set frame's initial priority value (egress queue). May be overridden by
                   a control list entry later
(config-stream-gate)# no ipv?
    <0-7>          Let frame retain its original internal priority value (egress queue).
                   May be overridden by a control list entry later
(config-stream-gate)# close-due-to-invalid-rx-enable?
    <boolean>      If set, a stream gate gets permanently closed if receiving a frame
                   during a closed gate state
(config-stream-gate)# close-due-to-octets-exceeded-enable?
    <boolean>      If set, a stream gate gets permanently closed if receiving a frame
                   that exceeds the configured 'octet-max'.
(config-stream-gate)# cycle-time ?
    <1-1000000000> Set the gate's cycle time. A cycle time of up to 1 second can be
                   specified
(config-stream-gate)# cycle-time 1 ?
    ms             Set cycle time value in units of milliseconds. With this unit,
                   the cycle time cannot exceed 1000 ms
    us             Set cycle time value in units of microseconds. With this unit,
                   the cycle time cannot exceed 1,000,000 us
    ns             Set cycle time value in units of nanoseconds. With this unit,
                   the cycle time cannot exceed 1,000,000,000 ns
(config-stream-gate)# no cycle-time?
                   Set the gate's cycle time to 0.
(config-stream-gate)# time-extension ?
    <1-1000000000> Set the gate's cycle time extension. An extension of up to 1 second
                   can be specified
(config-stream-gate)# time-extension 1 ?
    ms             Set admin cycle time extension value in units of milliseconds.
                   With this unit, the cycle time extension cannot exceed 1000 ms
    us             Set admin cycle time extension value in units of microseconds.
                   With this unit, the cycle time extension cannot exceed 1,000,000 us
    ns             Set admin cycle time extension value in units of nanoseconds.
                   With this unit, the cycle time extension cannot exceed 1,000,000,000 ns
(config-stream-gate)# no time-extension?
                   Set the gate's cycle time extension to 0.
(config-stream-gate)# base-time?
                   Set the time for the next config-change to take place
(config-stream-gate)# base-time seconds ?
                   Specify seconds
    <0-4294967295> Seconds
(config-stream-gate)# base-time seconds 0 nanoseconds ?
    <0-999999999>  Nanoseconds. Default is 0

! The following range (0-4) is platform specific. Here, it is for LAN9668.
(config-stream-gate)# control-list-length ?
    <0-4>          Length of gate control list

(config-stream-gate)# control-list index ?
    <0-3>          Select an index into the Gate Control List. Only indices smaller
                   than the configured control-list-length are allowed.
(config-stream-gate)# no control-list index ?
    <0-3>          Index into the Gate Control List that will be defaulted
(config-stream-gate)# control-list index 0 gate-state ?
    closed         Close stream gate
    open           Open stream gate
(config-stream-gate)# control-list index 0 gate-state closed time-interval ?
    <1-999999999>  Set gate control entry's time interval (default is 1 nanosecond)
(config-stream-gate)# control-list index 0 gate-state open time-interval 1 ?
    ms             Unit of time interval is milliseconds.
                   With this unit, the interval cannot exceed 999 ms
    us             Unit of time interval is microseconds.
```

```
                   With this unit, the interval cannot exceed 999,999 us
    ns             Unit of time interval is nanoseconds.
                   With this unit, the interval cannot exceed 999,999,999 ns
(config-stream-gate)# control-list index 0 gate-state open time-interval 1 ms ipv?
                   Configure frame's internal priority value. If left out, the IPV it
                   has received so far will be kept
(config-stream-gate)# control-list index 0 gate-state open time-interval 1 ms ipv ?
    <0-7>          Select frame's internal priority value (egress queue).
(config-stream-gate)# control-list index 0 gate-state open time-interval 1 ms octet-max?
                   Configure the size of the largest frame that can slip through this gate.
                   If left out, any-sized frame is allowed
(config-stream-gate)# control-list index 0 gate-state open time-interval 1 ms octet-max ?
    <uint>         Size of the largest frame to let through gate. Use 0 to disable check
(config-stream-gate)# enable?
    <boolean>      Enable the gate. Use the no-form to disable the gate. When disabled,
                   the gate is not programmed to hardware
(config-stream-gate)# config-change?
    <boolean>      One-shot parameter. Apply configuration to hardware. Requires gate to be enabled
```

The configuration can be thought of as divided into two: Global parameters that are applied immediately without using `config-change`, and control-list related parameters that require `config-change` to be issued before they are applied to hardware.

The global parameters are `state` and `ipv`. The remaining are control-list related parameters.

The control-list related parameters have three sets of configuration:

1. Configuration currently in effect in hardware (operational),
2. configuration that is pending and will be applied when the current time equals the configured `base-time`,
3. configuration that the user is updating with the intent of making it pending later by setting `config-change`.

NOTE

During boot, only global parameters from `startup-config` are applied to hardware. Control-list related configuration will be held back until PTP time has been acquired or a timeout has occurred. See Using TSN Functions with PTP Time Synchronisation for details.

When setting `config-change`, the configured `base-time` is compared to the current time. If `base-time` is in the past or less than two seconds ahead of current time, configuration will be applied to hardware two seconds later.

Otherwise, configuration will become pending and only applied to hardware when `base-time` reaches current time.

The current time can be shown with both `show tsn current-time` and `show tsn stream gate status`.

Example:

Create stream gate instance 30 with the following properties:

Cycle time is 100 ms. The gate is closed for 99 ms and open for 1 ms, where the internal priority is modified to 6. Block the stream permanently if frames are received during the closed time interval.

```
# configure terminal
(config)# tsn stream gate 30
(config-stream-gate)# cycle-time 100 ms
(config-stream-gate)# close-due-to-invalid-rx-enable

! Apply the configuration two seconds after we issue a config-change.
! 0 seconds is default, but included here for clarity.
(config-stream-gate)# base-time seconds 0

(config-stream-gate)# control-list-length 2
(config-stream-gate)# control-list index 0 gate-state closed time-interval 99 ms
(config-stream-gate)# control-list index 1 gate-state open time-interval 1 ms ipv 6
(config-stream-gate)# enable
(config-stream-gate)# end
#
```

8.3. Stream Gate Status

Until now, we haven't made a `config-change`. Let's have a look at the stream gate status.

The format of the command is: `show tsn stream gate [<0~255>] status [details]`

```
# show tsn stream gate 30 status details
Stream Gate #30:
Enabled:                       Yes
Config Pending:                No
Gate State:                    Closed
IPV:                           Disabled
Closed Due To Invalid Rx:      No
Closed Due To Octets Exceeded: No
Config Change Errors:          0
Current Time (seconds):        59299.225175939
Current Time (ISO 8601):       1970-01-01T16:28:19.225Z

Configuration                    Operational              Pending                  Configured
-------------------------------- ------------------------ ------------------------
------------------------
Config Change Time (seconds)                              -                        -
0.000000000
Config Change Time (ISO 8601)                             -                        -
1970-01-01T00:00:00.000Z
Cycle Time                                                -
-                    100 ms
Cycle Time Extension                                      -
-                      0 ns
Control List Length                                       0
-                         2
Control List Index 0 Gate State                           -
-                    Closed
Control List Index 1 Gate State                           -
-                      Open
Control List Index 0 IPV                                  -
-                  Disabled
Control List Index 1 IPV                                  -
-                         6
Control List Index 0 Time Interval                        -
-                     99 ms
Control List Index 1 Time Interval                        -
-                      1 ms
Control List Index 0 Octet Max                            -
-                  Disabled
Control List Index 1 Octet Max                            -
-                  Disabled
```

The beginning of the status contains global state parameters, hereunder whether the stream gate is currently (permanently) closed.

The current time is shown in two different formats: Number of seconds since 1st of January 1970 and in ISO 8601 format (see e.g. https://en.wikipedia.org/wiki/ISO_8601).

The second part of the status shows the three aforementioned configuration sets. Since the gate has just been created and enabled, there is no operational configuration; the `Control List Length` is explicitly mentioned as 0 for clarity. The remaining fields are shown with dashes.

Likewise, since `config-change` has not yet been issued, the `Pending` column also only contains dashes.

The `Configured` column holds what we have configured until now.

Now, set `config-change` and look at the status within two seconds after:

```
# configure terminal
(config)# tsn stream gate 30
(config-stream-gate)# config-change
(config-stream-gate)# do show tsn stream gate 30 status details
Stream Gate #30:
Enabled:                       Yes
Config Pending:                Yes
Gate State:                    Closed
IPV:                           Disabled
Closed Due To Invalid Rx:      No
Closed Due To Octets Exceeded: No
Config Change Errors:          1
Current Time (seconds):        60028.612291984
Current Time (ISO 8601):       1970-01-01T16:40:28.612Z

Configuration                     Operational              Pending                  Configured
--------------------------------- ------------------------ ------------------------
------------------------
Config Change Time (seconds)                             -              0.000000000
0.000000000
Config Change Time (ISO 8601)                            -  1970-01-01T00:00:00.000Z
1970-01-01T00:00:00.000Z
Cycle Time                                               -                   100
ms                  100 ms
Cycle Time Extension                                     -                     0
ns                    0 ns
Control List Length                                     0
2                         2
Control List Index 0 Gate State                          -
Closed                    Closed
Control List Index 1 Gate State                          -
Open                      Open
Control List Index 0 IPV                                 -
Disabled                  Disabled
Control List Index 1 IPV                                 -
6                         6
Control List Index 0 Time Interval                       -                    99
ms                     99 ms
Control List Index 1 Time Interval                       -                     1
ms                      1 ms
Control List Index 0 Octet Max                           -
Disabled                  Disabled
Control List Index 1 Octet Max                           -
Disabled                  Disabled
```

Since we managed to show the status within two seconds after the `config-change` we can see that the configured configuration now became pending and the `Config Pending` row changed from "No" to "Yes".

Also notice that the `Config Change Errors` incremented from 0 to 1. This is because the configured `base-time` is in the past compared to the current time.

Let’s look at the status again:

```
(config-stream-gate)# do show tsn stream gate 30 status details
Stream Gate #30:
Enabled:                       Yes
Config Pending:                No
Gate State:                    Closed
IPV:                           Disabled
Closed Due To Invalid Rx:      No
Closed Due To Octets Exceeded: No
Config Change Errors:          1
Current Time (seconds):        60212.762751879
Current Time (ISO 8601):       1970-01-01T16:43:32.762Z

Configuration                      Operational              Pending                 Configured
---------------------------------- ------------------------ ------------------------
------------------------
Config Change Time (seconds)                60030.100000000                        -
0.000000000
Config Change Time (ISO 8601)      1970-01-01T16:40:30.100Z                        -
1970-01-01T00:00:00.000Z
Cycle Time                                           100 ms
-                    100 ms
Cycle Time Extension                                   0 ns
-                      0 ns
Control List Length                                       2
-                         2
Control List Index 0 Gate State                      Closed
-                    Closed
Control List Index 1 Gate State                        Open
-                      Open
Control List Index 0 IPV                           Disabled
-                  Disabled
Control List Index 1 IPV                                  6
-                         6
Control List Index 0 Time Interval                    99 ms
-                     99 ms
Control List Index 1 Time Interval                     1 ms
-                      1 ms
Control List Index 0 Octet Max                     Disabled
-                  Disabled
Control List Index 1 Octet Max                     Disabled
-                  Disabled
```

Now the configuration became operational, and there is no pending configuration.

8.4. Stream Gate Control

If a stream gate is closed due to invalid Rx or octets exceeded, it can be re-opened with the following CLI EXEC command:

`clear tsn stream gate [<0~255>] [gate-closed-due-to-octets-exceeded | gate-closed-due-to-invalid-rx]`

If omitting `gate-closed-due-to-XXX` both of them will be cleared and the gate will re-open.

8.5. Stream Filter Configuration

Stream filters are used to tie the individual sub-components together.

Stream filters are identified by an ID ranging from 0 to a platform specific value.

A stream filter is instantiated at CLI's global configuration level:

```
! The range is platform specific. This is for LAN9668:
tsn stream filter <0-255>
```

Where:

```
<0-255>  Stream Filter instance number
```

Stream filters can be removed with the `no tsn stream filter <0-255>` command.

When configuring a stream filter, the following stream filter configuration level CLI commands are available:

```
[no] stream-id <uint>
[no] stream-collection-id <uint>
[no] flow-meter id <uint>
[no] gate id <uint>
[no] max-sdu <uint>
[no] block-due-to-oversize-enable
```

In the following, the CLI help texts are used to describe the individual options. The ranges for IDs are platform specific. Here, they are shown for LAN9668.

```
# configure terminal
(config)# tsn stream filter 1
(config-stream-filter)# stream-id ?
    <1-127>   ID of the stream to attach this filter to
(config-stream-filter)# stream-collection-id ?
    <1-63>    ID of the stream collection to attach this filter to
(config-stream-filter)# flow-meter id ?
    <1-63>    ID of a flow meter to use with this filter
(config-stream-filter)# gate id ?
    <0-255>   ID of a stream gate to use with this filter
(config-stream-filter)# max-sdu ?
    <uint>    Set maximum allowed frame size for the filter. Any frame exceeding this value will be
discarded. A value of 0 disables the feature
(config-stream-filter)# block-due-to-oversize-enable?
    <boolean> If enabled and a frame exceeds the max-sdu size, all subsequent frames will be discarded
as well
```

It is not possible to specify both a `stream-id` and a `stream-collection-id` simultaneously. If you happen to have specified a `stream-id` and want to use a stream collection instead, you must first execute `no stream-id` before you can do a `stream-collection-id <ID>`.

The next section shows an example of configuring a stream filter along with showing of status and statistics.

8.6. Stream Filter Example

First, we create a stream. Streams or stream collections are required when configuring a stream filter. Without them, the stream filter is useless.

The following creates stream number 10. The stream matches VLAN-tagged traffic with VLAN ID 1 and SMACs in the range `00-01-c1-00-00-00` through `00-01-c1-ff-ff-ff`. The stream is instantiated on Gi 1/7 and Gi 1/8.

```
# configure terminal
(config)# stream 10
(config-stream)# dmac any
(config-stream)# smac 00-01-c1-00-00-00 / ff-ff-ff-00-00-00
(config-stream)# outer-tag vid 1
(config-stream)# exit

! Associate stream 10 with Gi 1/7,8:
(config)# interface GigabitEthernet 1/7-8
(config-if)# stream-id 10
(config-if)# end
#
```

Next, we create a flow meter. Use of flow meters is optional in stream filters.

The following example creates flow meter 20 and configures it with a committed information rate of 400 kbps and a committed burst size of 8192 bytes. If frames are exceeding the configured rate, we block the stream permanently:

```
# configure terminal
(config)# tsn flow meter 20
(config-flow-meter)# cir 400
(config-flow-meter)# cbs 8192
(config-flow-meter)# mark-red-enable
(config-flow-meter)# end
#
```

Next, we create a stream gate. Use of stream gates is optional in stream filters.

The following example creates stream gate 30 and configures it with a cycle-time of 100 ms and two control-list entries. The first closes the gate for 99 ms and the seconds opens it for 1 ms and modifies the internal priority to 6. The stream is configured to block the stream permanently if frames are received during the closed time interval.

```
# configure terminal
(config)# tsn stream gate 30
(config-stream-gate)# cycle-time 100 ms
(config-stream-gate)# close-due-to-invalid-rx-enable
(config-stream-gate)# base-time seconds 0
(config-stream-gate)# control-list-length 2
(config-stream-gate)# control-list index 0 gate-state closed time-interval 99 ms
(config-stream-gate)# control-list index 1 gate-state open time-interval 1 ms ipv 6
(config-stream-gate)# enable
(config-stream-gate)# config-change
(config-stream-gate)# end
#
```

Finally, it's time to tie the components together to form the filter. In this example we create filter instance 1 and use the sub-components defined above. We also block the stream permanently if frames larger than 1518 bytes are received.

```
# configure terminal
(config)# tsn stream filter 1
(config-stream-filter)# stream-id 10
(config-stream-filter)# flow-meter id 20
(config-stream-filter)# gate id 30
(config-stream-filter)# max-sdu 1518
(config-stream-filter)# block-due-to-oversize-enable
(config-stream-filter)# end
#
```

The final `running-config` looks like this (we add `feature tsn` to the command to omit irrelevant configuration):

```
Building configuration...
stream 10
 smac 00-01-c1-00-00-00 / ff-ff-ff-00-00-00
 outer-tag vid 1
!
!
interface GigabitEthernet 1/1
!
interface GigabitEthernet 1/2
!
interface GigabitEthernet 1/3
!
interface GigabitEthernet 1/4
!
interface GigabitEthernet 1/5
!
interface GigabitEthernet 1/6
!
interface GigabitEthernet 1/7
 stream-id 10
!
interface GigabitEthernet 1/8
 stream-id 10
!
tsn flow meter 20
 cir 400
 cbs 8192
 mark-red-enable
!
tsn stream gate 30
 close-due-to-invalid-rx-enable
 cycle-time 100 ms
 control-list-length 2
 control-list index 0 gate-state closed time-interval 99 ms
 control-list index 1 gate-state open time-interval 1 ms ipv 6
 enable
 config-change
!
tsn stream filter 1
 stream-id 10
 flow-meter id 20
 gate id 30
 max-sdu 1518
 block-due-to-oversize-enable
!
end
#
```

NOTE

The `config-change` in the stream gate configuration is always added to `running-config` when the gate is enabled. The reason is that if `running-config` is saved to `startup-config`, a subsequent boot and appliance of `startup-config` needs to apply the stream gate configuration to hardware.

8.7. Stream Filter Status

Now it's time to check the stream filter for configurational warnings. We do so by showing its status:

```
# show tsn stream filter status
Filter ID Blocked due to Configurational
          Oversize Frame Warnings
--------- -------------- ---------------
        1 No             No
```

Luckily, there were no configurational warnings. If there were, the output would look like this:

```
# show tsn stream filter status
Filter ID Blocked due to Configurational
          Oversize Frame Warnings
--------- -------------- ---------------
        1 No             YES!
```

And they could be examined by looking at the detailed status:

```
# show tsn stream filter status details
Stream Filter ID:              1
Blocked due to Oversize Frame: No
Configurational warnings:      The specified stream gate is not enabled
```

In fact, there are a number of possible configurational warnings, as outlined below:

- `Neither a stream or a stream collection is specified`: For a stream filter to be useful, it needs to attach to a stream or a stream collection.
- `The specified stream ID does not exist`: You have specified a stream ID that is non-existent. Create the stream or change the stream-id of the filter.
- `The specified stream collection ID does not exist`: You have specified a stream collection ID that is non-existent. Create the stream collection or change the stream-collection-id of the filter.
- `Unable to attach to the specified stream, possibly because it is part of a stream collection`: The stream ID exists, but it is probably part of a stream collection, in which case it cannot be attached to directly. Either change the filter to attach to the stream collection or change the stream collection to not include the stream ID.
- `Unable to attach to the specified stream collection`: There is no good reason for this one.
- `The specified stream has configurational warnings`: The stream this filter is attached to has configurational warnings. Use `show stream status <stream-id> details` to see them.

- `The specified stream collection has configurational warnings`: The stream collection this filter is attached to has configurational warnings. Use `show stream-collection status <stream-collection-id> details` to see them.
- `The specified flow meter ID does not exists`: The filter points to a non-existent flow meter. Create the flow meter or let the filter point to the correct flow meter or none.
- `The specified stream gate ID does not exist`: The filter points to a non-existent stream gate. Create the stream gate or let the filter point to the correct stream gate or none.
- `The specified stream gate is not enabled`: The filter points to a disabled stream gate. Enable the stream gate.

Suppose the stream filter status shows that the stream is blocked due to oversize frames, i.e.:

```
# show tsn stream filter status
Filter ID Blocked due to Configurational
          Oversize Frame Warnings
--------- -------------- ---------------
        1 Yes            No
```

This condition can be cleared like this:

```
# clear tsn stream filter stream-blocked-due-to-oversize-frame
# show tsn stream filter status
Filter ID Blocked due to Configurational
          Oversize Frame Warnings
--------- -------------- ---------------
        1 No             No
```

8.8. Stream Filter Statistics

A stream filter carries along with it a set of statistics that can be shown with `show tsn stream filter [instances] statistics`

```
# show tsn stream filter statistics
Filter ID Matching       Passing        Not Passing    Passing SDU    Not Passing SDU Red
--------- -------------- -------------- -------------- -------------- --------------
--------------
        1            200            200              0            200
0              0
```

All the counters are frame counters.

Statistics can be cleared with: `clear tsn stream filter [instance(s)] statistics`

```
# clear tsn stream filter 1 statistics
# show tsn stream filter statistics
Filter ID Matching        Passing        Not Passing     Passing SDU     Not Passing SDU Red
--------- -------------- -------------- -------------- -------------- --------------
--------------
        1              0              0              0              0
0              0
```

NOTE Statistics is shared with FRER in generator mode, so if both PSFP and FRER attach to the same stream or stream collection, clearing PSFP statistics also clears part of FRER statistics, and clearing FRER statistics clears all PSFP statistics.

9. Frame Replication and Elimination for Reliability

Frame Replication and Elimination for Reliability (FRER), as specified in the IEEE 802.1CB-2017 standard, provides increased reliability (reduced packet loss rates) for a stream.

This is achieved by

1. Sequence numbering and replicating packets in the talker (source) end system and/or in relay systems in the network. In the following, this is called generation.
2. Eliminating those replicates in the listener (destination) end system and/or in other relay systems. In the following, this is called recovery.

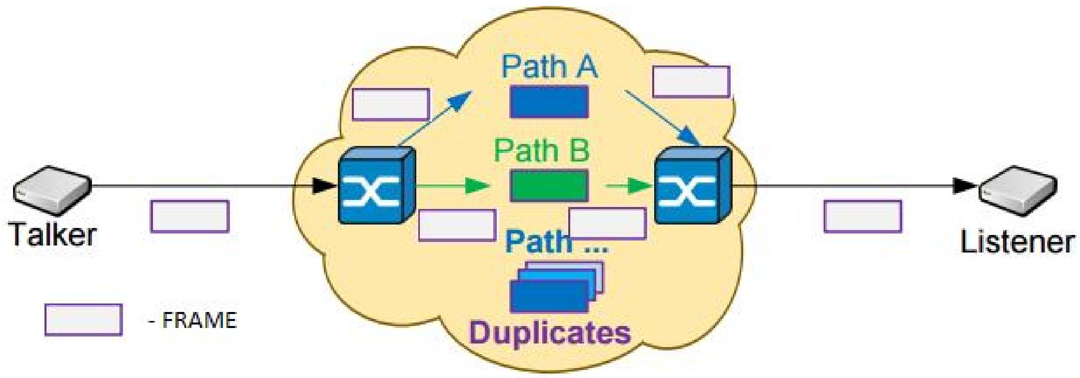


Figure 17. Frame Replication and Elimination for Reliability

FRER is supported by Microchip's SparX-5i, LAN9668, and LAN969x switch chips running IStaX software. There are differences between these, which will be explained as we go along.

9.1. Overview

The basic idea is to send frames from the talker end through two or more paths in the network to increase the likelihood of the frames reaching their destination (listener end).

In order not to and send multiple copies of a given frame to the listener, the FRER generator augments every frame with a six byte so-called R-tag, as specified in clause 7.8 of 802.1CB-2017, before transmitting the frame on two or more ports.

The R-tag contains a two byte sequence number, which is incremented for every frame received by the generator. This sequence number is used by subsequent recovery functions in the FRER network to eliminate copies of a frame before presenting it to the listener or to the next switch in the FRER network.

Several system components come into play when configuring FRER:

1. Streams or stream collections are used for identifying and classifying frames ingressing a switch to a given FRER instance,
2. VLANs and/or MAC addresses are used to guide the frames the correct way in the FRER network,
3. MSTP is used to prevent loops, and
4. FRER configuration is used to connect the dots and configure a FRER instance as either a sequence generator or a sequence recovery instance.

It is often desirable to perform central management of the switches making up the FRER network, which in turn may result in layer 2 loops unless guarded by a loop prevention protocol such as spanning tree. In this guide, we provide examples that not only show how to configure FRER, but also how to configure the FRER switches for central, loop-free management. In the entire guide, we assume that the management VLAN is running untagged in VLAN 1.

In a FRER network, besides the R-tag, it is common practice to augment every frame egressing the generator with a VLAN tag containing a VLAN ID - the so-called FRER VLAN - which is used further down the FRER network to classify the frames as belonging to a given FRER instance. This FRER VLAN tag is supposed to be removed on the last switch in the FRER network before presenting the original frames to the listener end system.

Although not impossible to operate untagged, the presence of a VLAN tag containing a FRER VLAN makes network administration easier.

In the following, we assume that all FRER-protected frames carry an outer FRER VLAN tag. With this in place, there are two methods to guide the frames in the right direction:

1. VLAN flooding and
2. MAC address table forwarding

Let’s examine these two methods separately.

9.1.1. Method 1: VLAN Flooding

The idea is that frames egress the FRER generator with this outer FRER VLAN tag and get flooded to all ports on subsequent switches that are members of this VLAN.

All switches in the FRER network must disable learning in the FRER VLAN and make sure that frames only egress ports towards the final recovery function.

To avoid loops, the ingress ports along the way in the FRER network may **not** be members of the FRER VLAN.

With this method, the network administrator must assign a unique FRER VLAN to every FRER instance.

9.1.2. Method 2: MAC Address Table Forwarding

With this method, the listener’s MAC address is pre-provisioned in every switch along the way, so that the <FRER VLAN, Listener MAC> tuple points out the egress ports.

With this method, one single FRER VLAN can accommodate all FRER instances, but it requires the listener’s MAC address to be known beforehand.

Later on, we shall see examples of configuring switches with both methods. For now, let’s look at FRER configuration parameters.

9.2. Configuration

A FRER instance represents a unidirectional function which can be either in generation or recovery mode. When a FRER instance is disabled, it has no impact on the frames passing through the switch.

Upon enabling a FRER instance, a sanity check on the provided parameters will be performed and if the combination of parameters lies within the acceptable range, FRER will start to operate.

When you configure a system which implements a full FRER network, you also have to consider configuration of one or more of the following components: VLAN, Streams or stream collections, MAC address table.

The syntax for FRER global level CLI configuration command is:

```
tsn frer <inst>
```

Where:

```
inst                    FRER instance number
```

The syntax of FRER configuration level CLI command is:

```
admin-state {enable | disable}
egress interface (<port_type> [<port_list>])
frer-vlan <vid>
ingress stream-id-list <stream_list>
ingress stream-collection-id <stream_collection_id>
ingress outer-tag pop
mode {generation | recovery}
recovery algorithm {match | vector [history-length <history_len>]}
recovery individual
recovery latent-error-detection [difference <diff>] [period <period>] [paths
<paths>] [reset-period <reset_period>]
recovery reset-timeout <reset_timeout>
recovery take-no-sequence
recovery terminate
```

Where:

```
admin-state             Enable or disable a FRER instance
egress                  Select egress ports that this FRER instance will hit
frer-vlan               Select the VLAN ID that ingress flows get classified to
mode                    Choose this FRER instance's mode of operation
                        (generation or recovery)
outer-tag pop           In generation mode, remove a possible outer VLAN tag from
                        ingressing frames before egressing with an R-tag.
recovery                Set a recovery mode parameter
stream-id-list          Select the ingress streams that should map to this FRER
                        instance. Only one stream ID can be specified in
                        generator mode
stream-collection-id    If more than one stream is to be matched in generator
                        mode, use stream collections
generation              FRER instance generates R-tags
recovery                FRER instance operates in recovery mode
algorithm               Choose which recovery algorithm to run
match                   Run match recovery algorithm (802.1CB, clause 7.4.3.5)
vector                  Run vector recovery algorithm (802.1CB, clause 7.4.3.4)
history-length          Select the vector algorithm's history length
individual              When individual recovery is enabled, each member stream
                        runs the recovery function before presenting it to the
                        compound recovery function
latent-error-detection Enable recovery's latent error detection function
difference              Set the maximum allowed difference between discarded
                        packets and passed packets before triggering the
                        detection of a latent error
period                  Set the period with which the latent error test function
                        runs
reset-period            Set the period between running the latent error reset
                        function
reset-timeout           Configure recovery function's reset timeout
take-no-sequence        Accept all frames whether they are R-tagged or not
terminate               This option allows to strip an R-tag from a frame before
                        presenting it on egress

no egress interface                Unset the egress interfaces of this FRER
                                   instance
no frer-vlan                       Default the VLAN ID that ingress frames get
                                   classified to
no mode                            Default this FRER instance's mode of
                                   operation
no ingress stream-id-list          Clear the list of ingress stream IDs
no ingress stream-collection-id    Clear the ingress stream collection ID
no ingress outer-tag pop           Preserve a possible outer VLAN tag beneath
                                   the R-tag on egress (generation mode, only)
no recovery algorithm              Default the recovery algorithm (vector)
no recovery individual             When individual recovery is disabled, all
                                   member streams are sent directly to the
                                   compound recovery function without performing
                                   individual recovery first
no recovery latent-error-detection Disable recovery's latent error detection
                                   function
no recovery reset-timeout          Default the recovery function's reset timeout
no recovery take-no-sequence       The recovery function discards frames that
                                   are not R-tagged (default)
```

`no recovery terminate`	`Do not strip an R-tag from a frame before presenting it on egress`

9.3. Simple Configuration Example

This example shows a Talker and a Listener connected by two switches, A and B, with two paths in-between. These two paths are protected by FRER, which implements splitting and recovery such that if one of the paths between the two switches fails, traffic will continue to flow uninterrupted between Talker and Listener.

The System Switch allows for managing both FRER switches. It must run the Multiple Spanning Tree Protocol (MSTP) at least in the management VLAN. Configuration of this switch is not otherwise part of this guide.

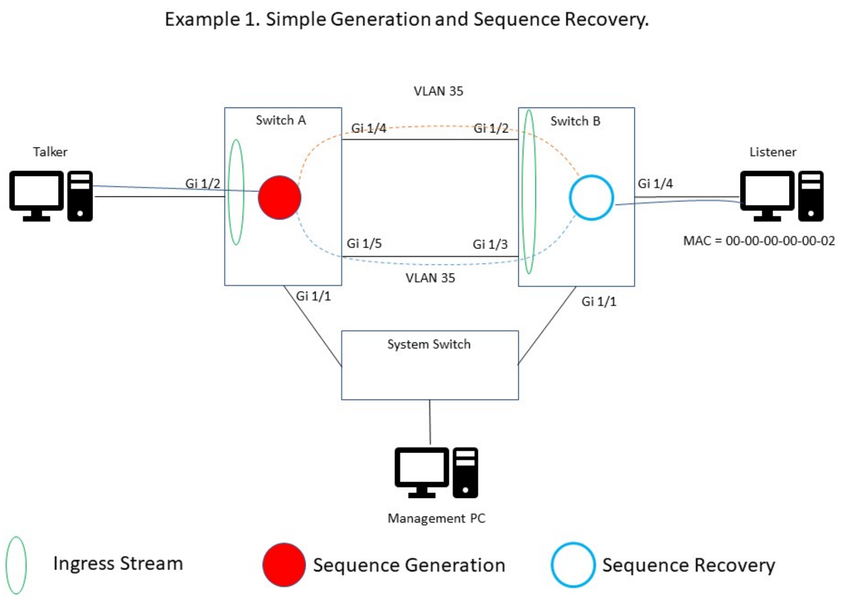


Figure 18. FRER Generation and Sequence Recovery.

Switch A takes care of classifiying the frames from the Talker to a FRER instance and splitting the stream into two paths. Switch B takes care of merging the two paths and presenting the original frames to the Listener.

Switch A's frame classification happens through the use of so-called streams. See Streams and Stream Collections for a description of configuring streams and stream collections.

In this example, we have chosen to map all frames ingressing Gi 1/2 with a destination MAC address (DMAC) matching that of the Listener to the FRER instance. The Listener's DMAC is 00-00-00-00-00-02.

Switch A's FRER instance is configured to be in generation mode, and generates two identical copies of the frame on its configured egress ports.

When the frames exit Gi 1/4 and Gi 1/5 of Switch A, they include a 6-byte R-tag containing a sequence number.

On top of that, it is common practice to also push an outer VLAN tag containing a VLAN ID used to identify the stream (or possibly multiple streams) on subsequent switches in the FRER network. The VLAN ID of this VLAN tag holds the so-called FRER VLAN.

It is typically the last switch in the FRER network that takes care of popping both the R-tag and the FRER VLAN tag from the frame before presenting the original frame to the Listener.

If the frame itself contained a VLAN tag on ingress to Switch A, the default is that that VLAN tag will become the inner tag upon egress of Switch A. A frame can therefore have between one and three tags on egress of Switch A:

1. Possibly 4 bytes outer VLAN tag with VLAN ID set to the FRER VLAN. This tag is only pushed if the egress port is configured to tag the FRER VLAN. The TPID is determined by the egress port’s tag type (C- or S-),
2. 6 bytes R-tag containing the generated sequence number,
3. If not configured to pop an outer tag in the ingressing frames (default), possibly 4 bytes inner VLAN tag containing the original ingress frame’s VLAN tag. Only available if the original frame contained a VLAN tag. This is not available when the generator is configured to pop a possible outer tag.

In both the VLAN flooding example and the MAC Address Table Forwarding examples, switch B will be configured to look for frames arriving on Gi 1/2 and Gi 1/3 with an outer VLAN tag containing a VLAN ID corresponding to the FRER VLAN.

In the MAC Address Table Forwarding example, this matching will also match on the frames' DMAC.

The goal is to let the Listener receive the same frames as the Talker transmitted while providing reliability, so that either of the two links between Switch A and Switch B can go down without the two parties noticing.

9.3.1. Example 1-1: VLAN Flooding

This first example will make use of VLAN flooding.

Switch A: Generation using VLAN Flooding

Command	Purpose
`#` **`configure terminal`**	Enter configuration mode.
`(config)#` **`prompt Switch-A-Flood`**	Give this switch a name. Use 'Flood' to distinguish this setup from the MAC address table based configuration.
`Switch-A-Flood(config)#` **`vlan 1`** `Switch-A-Flood(config-vlan)#` **`interface GigabitEthernet 1/1`** `Switch-A-Flood(config-if)#` **`switchport access vlan 1`** `Switch-A-Flood(config-if)#` **`switchport mode access`**	Make sure the management VLAN is defined and that the management port is a member of this VLAN. These are all default settings and will not be shown in the running-config.

`Switch-A-Flood(config)#` **`interface *`** `Switch-A-Flood(config-if)#` **`spanning-tree`** `Switch-A-Flood(config-if)#` **`exit`**	Make sure all ports (could be just the ones we show in the figure) run MSTP. This is the default and will not be shown in the running-config, but is included for clarity.
`Switch-A-Flood(config)#` **`spanning-tree mst te vlan 35`**	By default, all 4K VLANs are included in the CIST MSTI, so if MSTP finds a loop, it will block all 4K VLANs. We know that there is a loop on the FRER VLAN between Switch A and Switch B, so in order to have frames in the FRER VLAN egress both ports on Switch A, we will have to exclude this VLAN from blocking. We do that by putting it into a special MSTI called `te`, which stands for Traffic Engineering. VLANs in the TE MSTI are always forwarding. As we shall see later, on Switch-B we make sure that the ingress ports are NOT members of the FRER VLAN, because if they were, we would indeed have a loop.
`Switch-A-Flood(config)#` **`stream 17`** `Switch-A-Flood(config-stream)#` **`dmac 00-00-00-00-00-02`** `Switch-A-Flood(config-stream)#` **`exit`**	Create a stream for classifying frames from the Talker to a FRER instance. The stream ID (17) is arbitrarily chosen and must be unique within this switch. The stream may be used in conjunction with PSFP. Let the stream match frames with DMAC equal to that of the listener and exit the stream configuration mode.
`Switch-A-Flood(config)#` **`interface GigabitEthernet 1/2`** `Switch-A-Flood(config-if)#` **`stream-id 17`**	Assign Stream ID 17 to Gi 1/2, the ingress port receiving frames from the talker.
`Switch-A-Flood(config-if)#` **`no switchport hybrid ingress-filtering`** `Switch-A-Flood(config-if)#` **`switchport hybrid allowed vlan none`** `Switch-A-Flood(config-if)#` **`switchport mode hybrid`**	Later on in the configuration, we will make sure that all frames matching the newly configured stream get classified to the FRER VLAN. We don’t want Gi 1/2 to be member of the FRER VLAN, because we don’t want frames received on other ports that happen to get classified to the FRER VLAN to egress our ingress port. If Gi 1/2 is not member of the FRER VLAN, all frames would get discarded with the default VLAN configuration mode (`switchport mode access`). Therefore, we put the ingress port into hybrid mode, which is the only VLAN mode that allows us to disable ingress filtering. At a first glance, it looks dangerous to disable ingress filtering, because any VLAN is then accepted on the ingress port. This, however, is not the case when MSTP is enabled on the port. Behind the scenes, MSTP enables ingress filtering on all ports except for VLANs in the Traffic Engineering MSTI. Since the FRER VLAN is the only VLAN in this MSTI, it will be the only VLAN that has ingress filtering disabled. Notice that ingress filtering is disabled by default in hybrid mode, so `show running-config` will not print this line. We leave the port to not being member of any VLAN, but it could be configured to being member of any VLAN excluding the FRER VLAN.

`Switch-A-Flood(config-if)#` **`interface GigabitEthernet 1/4,5`** `Switch-A-Flood(config-if)#` **`switchport hybrid allowed vlan 1,35`** `Switch-A-Flood(config-if)#` **`switchport hybrid native vlan 1`** `Switch-A-Flood(config-if)#` **`switchport hybrid port-type c-port`** `Switch-A-Flood(config-if)#` **`switchport mode hybrid`** `Switch-A-Flood(config-if)#` **`exit`**	Configure our two egress ports, Gi 1/4 and Gi 1/5. We can do this in one tempo, because we - in this example - configure them identically. This might not be the case in other scenarios. We make the ports members of VLAN 1 - the management VLAN - and VLAN 35, the FRER VLAN. The ports could be members of more VLANs, but they must be members of at least the FRER VLAN. In order to force a C-tag on R-tagged frames, we keep the native VLAN (PVID) at 1 and make the port a C-port before switching to the newly configured hybrid mode. Notice that neither the `native vlan` nor the `port-type` lines will appear in `show running-config`, because these are the defaults in hybrid mode. They are included here for clarity. Finally exit interface configuration mode.
`Switch-A-Flood(config)#` **`no mac address-table learning vlan 35`**	As said in the beginning of this example, we make use of VLAN flooding to make sure that frames classified to the FRER VLAN get transmitted on all ports that are members of the FRER VLAN. In order to make sure that this happens, we must disable learning and enable flooding in the FRER VLAN. If learning was not disabled, a frame ingressing Gi 1/2 to a particular destination MAC address (DMAC), which is present in the MAC address table on a particular port in the FRER VLAN, would only be sent on that port and not on all ports in the FRER VLAN. Flooding is enabled in all VLANs by default, so nothing required there.
`Switch-A-Flood(config)#` **`tsn frer 10`** `Switch-A-Flood(config-frer)#` **`mode generation`** `Switch-A-Flood(config-frer)#` **`ingress stream-id-list 17`** `Switch-A-Flood(config-frer)#` **`frer-vlan 35`** `Switch-A-Flood(config-frer)#` **`egress interface GigabitEthernet 1/4,5`** `Switch-A-Flood(config-frer)#` **`admin-state enable`**	Create a FRER instance with ID 10 (arbitrarily chosen, but must be unique within this switch) and put it in generation mode. Let it work on the stream we just created and have frames classified to our FRER VLAN (35). When using the VLAN flooding model, it is important that the FRER VLAN is not being used for any other purposes than FRER on this particular FRER instance; different FRER instances must use different FRER VLANs. If you want to have only a single FRER VLAN for multiple FRER instances, you must use the MAC table forwarding model, as explained later. Despite its name, the `stream-id-list` can only have one single member in generation mode. If you wish to match several streams in generator mode, use stream collections. Also specify the egress ports, Gi 1/4 and Gi 1/5. Replicated frames egressing these ports will have the same R-tag pushed. Finally enable this instance.
`Switch-A-Flood(config-frer)#` **`end`** `Switch-A-Flood#`	We are now done with configuration of Switch A.

NOTE	If other ports are members of the FRER VLAN, frames will also egress those. On LAN9668, the frames will egress *with* an R-tag and on SparX-5i and LAN969x, they will egress *without* an R-tag.

To summarize, here's a list of commands, where default and irrelevant commands are omitted.

```
prompt Switch-A-Flood

vlan 1

no mac address-table learning vlan 35
spanning-tree mst te vlan 35

stream 17
 dmac 00-00-00-00-00-02

interface GigabitEthernet 1/2
 switchport hybrid allowed vlan none
 switchport mode hybrid
 stream-id 17

interface GigabitEthernet 1/4,5
 switchport hybrid allowed vlan 1,35
 switchport mode hybrid

tsn frer 10
 mode generation
 ingress stream-id-list 17
 frer-vlan 35
 egress interface GigabitEthernet 1/4,5
 admin-state enable
```

Switch B: Recovery using VLAN Flooding

In this example, Switch B terminates the FRER flows and will transmit it on the designated egress port(s) without R-tags.
We define Switch B's FRER VLAN to be 100. This is in order to emphasize that the terminating switch's FRER VLAN doesn't need to be the same as on previous switches inside the FRER network, as long as it's unique on the terminating switch. It could, however, just as well be the same as on the previous switches in the FRER network.

Regarding management, we will skip describing the steps taken to making Gi 1/1 an access port in VLAN 1 and how to enable MSTP on all ports, because this is default and the same thing as on `Switch-A-Flood`. We will, however, emphasize how to prevent MSTP from blocking the FRER VLANs.

Command	Purpose
`# configure terminal`	Enter configuration mode.
`(config)# prompt Switch-B-Flood`	Give this switch a name.

`Switch-B-Flood(config)# spanning-tree mst te vlan 100`	As on Switch-A, we need to add VLANs that are not supposed to be blocked to the TE MSTI. In this case, we must have the new FRER VLAN ID (VLAN 100) in the TE MSTI, because we need this VLAN ID to always be forwarding.

`Switch-B-Flood(config-stream)# stream 20`
`Switch-B-Flood(config-stream)# outer-tag vid 35`
`Switch-B-Flood(config-stream)# exit`
	Create a stream (arbitrarily chosen stream ID) that matches the R-tagged frames coming from Switch A.
	Since - in the VLAN flooding model - it is "guaranteed" that only one stream uses this VLAN ID, we don’t need to match any other properties of the frames.

`Switch-B-Flood(config)# interface GigabitEthernet 1/2,3`
`Switch-B-Flood(config-if)# stream-id 20`
`Switch-B-Flood(config-if)# switchport hybrid allowed vlan 1`
`Switch-B-Flood(config-if)# switchport mode hybrid`
	Instantiate the stream on both ingress ports.
	Since the R-tagged frames get classified to the FRER VLAN (100), the FRER VLAN must be accepted on the ingress ports, but the ports cannot be members of it, because then frames ingressing Gi 1/2 would also egress Gi 1/3 and vice versa.
	To overcome this, we put the ingress ports in hybrid mode, which disables ingress filtering. Ingress filtering, however, gets enabled again behind the scenes by MSTP in all VLANs except for those in the TE MSTI.
	In this particular example, we have chosen the ports only to be members of the Management VLAN (1), so that management frames can flow freely between Switch-A and Switch-B. It is up to the network administrator to make the ingress ports members of other required VLANs (except for the FRER VLAN).

`Switch-B-Flood(config-if)# interface GigabitEthernet 1/4`
`Switch-B-Flood(config-if)# switchport access vlan 100`
`Switch-B-Flood(config-if)# exit`
`Switch-B-Flood(config)# vlan 100`
`Switch-B-Flood(config-vlan)# exit`
	In this example, we configure the egress port as a simple access port in Switch B’s FRER VLAN (100). Access ports are only members of their native VLAN (PVID), and the frames get untagged on transmission.
	Access ports require their PVID to be explicitly created (or they won’t be members of any VLANs), so we also create VLAN 100.

`Switch-B-Flood(config)# no mac address-table learning vlan 100`	This step is not so important when we only have one egress port. However, if we were not recovering and terminating a FRER flow with one egress port, but recovering and re-splitting the flows into several egress ports, this is an important step (see explanation in Switch A’s configuration).

`Switch-B-Flood(config)# tsn frer 20`
`Switch-B-Flood(config-frer)# mode recovery`
`Switch-B-Flood(config-frer)# ingress stream-id-list 20`
`Switch-B-Flood(config-frer)# frer-vlan 100`
`Switch-B-Flood(config-frer)# egress interface GigabitEthernet 1/4`
`Switch-B-Flood(config-frer)# recovery terminate`
`Switch-B-Flood(config-frer)# admin-state enable`
	Create a FRER instance (ID is arbitrarily chosen, but must be unique within this switch) and put it in recovery mode.
	Let it work on the stream we just created and have frames classified to VLAN 100.
	Specify the egress interface where the recovered frames are to be transmitted.
	The `recovery terminate` line configures the switch to remove the R-tag from frames egressing the specified egress interfaces.

`Switch-B-Flood(config-frer)# end` `Switch-B-Flood#`	We are now done with configuration of Switch B.

NOTE

If another port is member of VLAN 100 and that port is not specified in the FRER instance’s egress interface list, then frames transmitted on that port will not be recovered, that is, multiple copies will be transmitted - one per ingress port. Furthermore, on SparX-5i and LAN969x, these frames will still contain the R-tag, whereas on LAN9668, the R-tag will be stripped.

To summarize, here’s a list of commands, where default and irrelevant commands are omitted.

```
prompt Switch-B-Flood

vlan 1,100

no mac address-table learning vlan 100
spanning-tree mst te vlan 100

stream 20
 outer-tag vid 35

interface GigabitEthernet 1/2,3
 switchport hybrid allowed vlan 1
 switchport mode hybrid
 stream-id 20

interface GigabitEthernet 1/4
 switchport access vlan 100

tsn frer 20
 mode recovery
 ingress stream-id-list 20
 frer-vlan 100
 egress interface GigabitEthernet 1/4
 recovery terminate
 admin-state enable
```

9.3.2. Example 1-2. MAC Table Forwarding

The previous example uses VLAN flooding to get frames from ingress to egress on a particular switch. An alternative is to pre-provision the Listener’s MAC address in Switch A and Switch B’s MAC address tables. This requires somewhat more configuration than the VLAN flooding method, but is at the same time safer, because frames only egress the ports pre-provisioned with the Listener’s MAC address, and not all ports that are members of the FRER VLAN.
Another advantage of using the MAC table forwarding approach is that all FRER instances may use the same FRER VLAN.

A disadvantage of using the same FRER VLAN for all FRER instances is that broadcast and multicast frames can't be controlled per FRER instance. If this is required, each FRER instance must use a separate FRER VLAN. With this in place, either VLAN flooding or MAC table forwarding can be used.

The following configuration utilizes the same IDs and setup as the previous example. Focus will be on the differences to the previous example.

In the example, we assume that the Listener's MAC address is 00-00-00-00-00-02.

Switch A: Generation using MAC Table Forwarding

Command	Purpose
`# configure terminal`	Enter configuration mode.

`(config)# prompt Switch-A-MAC`
`Switch-A-MAC(config)# spanning-tree mst te vlan 35`
`Switch-A-MAC(config)# stream 17`
`Switch-A-MAC(config-stream)# dmac 00-00-00-00-00-02`
`Switch-A-MAC(config-stream)# exit`
`Switch-A-MAC(config)# interface GigabitEthernet 1/2`
`Switch-A-MAC(config-if)# switchport hybrid allowed vlan none`
`Switch-A-MAC(config-if)# switchport mode hybrid`
`Switch-A-MAC(config-if)# stream-id 17`
`Switch-A-MAC(config-if)# interface GigabitEthernet 1/4,5`
`Switch-A-MAC(config-if)# switchport hybrid allowed vlan 1,35`
`Switch-A-MAC(config-if)# switchport mode hybrid`
`Switch-A-MAC(config-if)# exit`
	This is the same and with the same explanations as in Example 1-1.
	The switch name, however, is suffixed by 'MAC' to emphasize that this is the MAC address table based approach.

`Switch-A-MAC(config)# mac address-table static 00-00-00-00-00-02 vlan 35 interface GigabitEthernet 1/4,5` `Switch-A-MAC(config)# no mac address-table learning vlan 35`	When using MAC table forwarding, we tell the switch that the Listener can be reached on two egress ports (Gi 1/4 and Gi 1/5) on the FRER VLAN. This ensures that frames destined to the Listener only egresses those two ports, no matter which other ports are members of the FRER VLAN, making it possible for several FRER instances to use the same FRER VLAN - possibly on other egress ports. We also - as in the previous example - disable learning in the FRER VLAN - for safety.

`Switch-A-MAC(config)# vlan 35`
`Switch-A-MAC(config-vlan)# no flooding`
`Switch-A-MAC(config-vlan)# exit`
	This is the second half of using MAC table forwarding. Disabling flooding in the FRER VLAN ensures that incoming frames that get classified to the FRER VLAN for one or the other reason, get dropped on ingress **unless** the frame's DMAC is in the MAC address table.
	Also, frames **with** the DMAC in the MAC address table are only sent towards the statically learned destination ports.

Switch-A-MAC(config)# **tsn frer 10** Switch-A-MAC(config-frer)# **mode generation** Switch-A-MAC(config-frer)# **ingress stream-id-list 17** Switch-A-MAC(config-frer)# **frer-vlan 35** Switch-A-MAC(config-frer)# **egress interface GigabitEthernet 1/4,5** Switch-A-MAC(config-frer)# **admin-state enable** Switch-A-MAC(config-frer)# **end** Switch-A-MAC#	This is the same and with the same explanations as in Example 1-1.

To summarize, here’s a list of commands, where the default commands are omitted.

```
prompt Switch-A-MAC

vlan 1
vlan 35
 no flooding

no mac address-table learning vlan 35
mac address-table static 00-00-00-00-00-02 vlan 35 interface GigabitEthernet 1/4,5
spanning-tree mst te vlan 35

stream 17
 dmac 00-00-00-00-00-02

interface GigabitEthernet 1/2
 switchport hybrid allowed vlan none
 switchport mode hybrid
 stream-id 17

interface GigabitEthernet 1/4,5
 switchport hybrid allowed vlan 1,35
 switchport mode hybrid

tsn frer 10
 mode generation
 ingress stream-id-list 17
 frer-vlan 35
 egress interface GigabitEthernet 1/4,5
 admin-state enable
```

Switch B: Recovery using MAC Table Forwarding

Command	Purpose
# **configure terminal**	Enter configuration mode.

`(config)#` **`prompt Switch-B-MAC`** `Switch-B-MAC(config)#` **`spanning-tree mst te vlan 100`** `Switch-B-MAC(config-stream)#` **`stream 20`** `Switch-B-MAC(config-stream)#` **`outer-tag vid 35`** `Switch-B-MAC(config-stream)#` **`dmac 00-00-00-00-00-02`** `Switch-B-MAC(config-stream)#` **`interface GigabitEthernet 1/2,3`** `Switch-B-MAC(config-if)#` **`switchport hybrid allowed vlan 1`** `Switch-B-MAC(config-if)#` **`switchport mode hybrid`** `Switch-B-MAC(config-if)#` **`stream-id 20`** `Switch-B-MAC(config-if)#` **`interface GigabitEthernet 1/4`** `Switch-B-MAC(config-if)#` **`switchport access vlan 100`** `Switch-B-MAC(config-if)#` **`vlan 100`**	This is the same and with the same explanations as in Example 1-1, except for one thing shown in the first NOTE below.
`Switch-B-MAC(config-vlan)#` **`no flooding`** `Switch-B-MAC(config)#` **`no mac address-table learning vlan 100`** `Switch-B-MAC(config)#` **`mac address-table static 00-00-00-00-00-02 vlan 100 interface GigabitEthernet 1/4`**	Here, we make sure that frames that get classified to VLAN 100 are discarded unless they are present in the MAC table. We also ensure that the Listener's MAC address is in the MAC table on the correct port.
`Switch-B-MAC(config)#` **`tsn frer 20`** `Switch-B-MAC(config-frer)#` **`mode recovery`** `Switch-B-MAC(config-frer)#` **`ingress stream-id-list 20`** `Switch-B-MAC(config-frer)#` **`frer-vlan 100`** `Switch-B-MAC(config-frer)#` **`egress interface GigabitEthernet 1/4`** `Switch-B-MAC(config-frer)#` **`recovery terminate`** `Switch-B-MAC(config-frer)#` **`admin-state enable`** `Switch-B-MAC(config-frer)#` **`end`** `Switch-B-MAC#`	This is the same and with the same explanations as in Example 1-1.

NOTE

This is very important: LAN9668 only supports stream matching beyond two tags, where the R-tag is one of them. So if the frames contain both a FRER VLAN tag, an R-tag **and** an inner VLAN tag, it is only possible to match on DMAC and SMAC, and not other, higher layer frame properties. If the frames ingressing the generator end are untagged, it is still possible to match on higher layer frame features in recovering switches.
SparX-5i and LAN969x, on the other hand, support stream matching on **three** tags, which makes it possible to match on frame features beyond L2, also on recovering switches in the FRER network.

NOTE

> If the DMAC is added to the MAC table on another egress port but it's not specified in the `egress interface …` of the FRER instance, frames will egress that other port unrecovered and therefore in multiple copies. On SparX-5i, they will in addition still contain the R-tag, whereas on LAN9668, the R-tag will be stripped.

To summarize, here's a list of commands, where the default commands are omitted.

```
prompt Switch-B-MAC

vlan 1
vlan 100
 no flooding

no mac address-table learning vlan 100
mac address-table static 00-00-00-00-00-02 vlan 100 interface GigabitEthernet 1/4
spanning-tree mst te vlan 100

stream 20
 dmac 00-00-00-00-00-02
 outer-tag vid 35

interface GigabitEthernet 1/2,3
 switchport hybrid allowed vlan 1
 switchport mode hybrid
 stream-id 20

interface GigabitEthernet 1/4
 switchport access vlan 100

tsn frer 20
 mode recovery
 ingress stream-id-list 20
 frer-vlan 100
 egress interface GigabitEthernet 1/4
 recovery terminate
 admin-state enable
```

9.4. Individual Recovery

The following figure shows an example of how to configure Switch B to perform individual recovery.

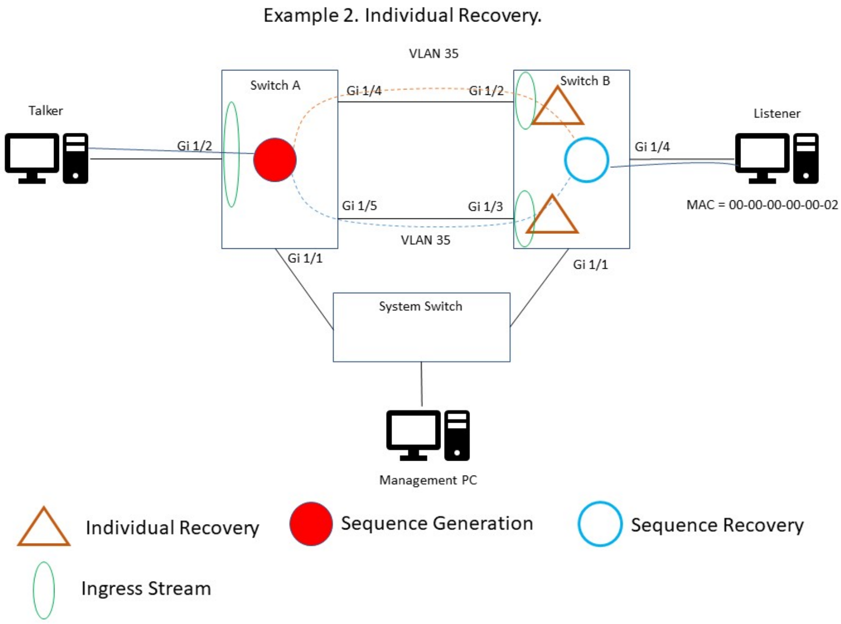


Figure 19. FRER Generation and Individual Recovery.

The first thing to note compared to the figure of Example 1 is that the two ingress ports of Switch B have its single ingress stream split in two. This allows the switch to perform individual recovery on each of the two member streams.

Individual recovery means that a member stream undergoes recovery before it reaches the compound recovery function. The compound recovery function sits on each and every egress port in the FRER instance, and is what we have used until now.

So what is the purpose of individual recovery?
The one and only thing that individual recovery can do that compound recovery can't is to filter out member streams that keep presenting the same R-tag sequence number because of a defect transmitter. It goes like this:

Suppose the transmitter of member stream 1 is working perfectly. It will send out frames with an increasing sequence number and wrap back to 0 after 65536 frames.
Suppose the transmitter of member stream 2 is sending out the same frame with the same sequence number, X, over and over again.
If we only had a compound recovery function, that function would at times be presented with frames with sequence number X from stream 1 and sequence number X from stream 2, and the first of these two frames would be sent to the egress port.
So - depending on timing - sometimes the frame with sequence number X would come from stream 1 and sometimes it would come from the erroneous stream 2.
The effect of enabling individual recovery is to have the individual recovery function for stream 2 filter out all identically numbered frames before they are presented to the compound recovery function.

Individual recovery is very expensive in terms of hardware resources: Every ingress stream needs an individual recovery function per egress port. So if a FRER instance defines 8 ingress streams and 8 egress ports, the switch needs 64 individual recovery instances - just for this one FRER instance.

NOTE SparX-5i and LAN969x support up to 8 ingress streams and 8 egress ports, whereas LAN9668 only supports 4 of each.

9.4.1. Example 2: Configuring Individual Recovery

Here is how you configure individual recovery on Switch B using the same methodology as was used in Example 1-1. Only changed commands are shown.

Command	Purpose
`Switch-B-Flood(config)#` **`stream 20`** `Switch-B-Flood(config-stream)#` **`outer-tag vid 35`** `Switch-B-Flood(config-stream)#` **`stream 21`** `Switch-B-Flood(config-stream)#` **`outer-tag vid 35`**	We need to create two separate streams with the same properties.
`Switch-B-Flood(config-stream)#` **`interface GigabitEthernet 1/2`** `Switch-B-Flood(config-if)#` **`stream-id 20`** `Switch-B-Flood(config-if)#` **`interface GigabitEthernet 1/3`** `Switch-B-Flood(config-if)#` **`stream-id 21`**	Instantiate stream 20 on Gi 1/2 and instantiate stream 21 on Gi 1/3.
`Switch-B-Flood(config-if)#` **`tsn frer 20`** `Switch-B-Flood(config-frer)#` **`ingress stream-id-list 20,21`** `Switch-B-Flood(config-frer)#` **`recovery individual`** `Switch-B-Flood(config-frer)#` **`end`** `Switch-B#`	Map both ingress streams to this FRER instance and enable individual recovery.

This boils down to this configuration:

```
prompt Switch-B-Flood

vlan 1,100

no mac address-table learning vlan 100
spanning-tree mst te vlan 100

stream 20
 outer-tag vid 35

stream 21
 outer-tag vid 35

interface GigabitEthernet 1/2
 switchport hybrid allowed vlan 1
 switchport mode hybrid
 stream-id 20

interface GigabitEthernet 1/3
 switchport hybrid allowed vlan 1
 switchport mode hybrid
 stream-id 21

interface GigabitEthernet 1/4
 switchport access vlan 100

tsn frer 20
 mode recovery
 ingress stream-id-list 20,21
 frer-vlan 100
 egress interface GigabitEthernet 1/4
 recovery individual
 recovery terminate
 admin-state enable
```

NOTE It is not possible to use stream collections with individual recovery.

9.5. Recovery Algorithm

IEEE 802.1CB-2017 requires implementations to provide two different recovery function algorithms, `match` and `vector`.

`match` is the simplest algorithm: It basically says: Discard all packets with a sequence number equal to the last sequence number seen. Accept all others. The algorithm also comes with a reset timer that - when it expires - causes the algorithm to accept any sequence number - even the same as the previous. The reset timer is restarted every time a packet is accepted.

The `match` algorithm counts the number of times the reset timer has expired and the number of passed, discarded, and out-of-order packets. Out-of-order happens when the sequence number of a given packet is not one higher than the previous (and the timer hasn't expired).

`vector` is somewhat more complicated. When a packet with a given sequence number arrives, it must be within the previous accepted packet's sequence number +/- a configurable history length, or it will be discarded. If the packet is already seen (within the history length window), it is also discarded.
Also this algorithm comes with a reset timer that - upon expiration - causes the algorithm to accept any sequence number next time a packet arrives. The reset timer is restarted every time a packet is accepted.

The `vector` algorithm counts the number of times the reset timer has expired and the number of passed, discarded, out-of-order, and so-called rogue packets. Out-of-order happens when the sequence number of a given packet is "older" than a previous packet's (taking wrap-around into account), and the packet hasn't been accepted before. Out-of-order packets are accepted.
Rogue packets are packets with a sequence number beyond the history length window. Rogue packets are also counted as discarded.
Furthermore, the `vector` algorithm counts lost packets, that is, the number of unreceived sequence numbers when the history window moves.

Both algorithms also count the number of packets arriving without an R-tag. This is done with the tagless counter. By default, such packets will be discarded. A per-FRER instance parameter `recovery take-no-sequence`, however, allows such frames to pass through.
Notice: The 802.1CB standard utilizes the `frerSeqRcvyTakeNoSequence` only in the `vector` algorithm, but the switch chips that the present guide is meant for also utilize it in the `match` algorithm.
Notice: This feature should only be used on terminating switches, because such tagless packets will be R-tagged (with sequence number 0) on their way out on non-terminating switches.

The selected algorithm on a given FRER instance will be used in both compound and individual recovery functions.

Default is the `vector` algorithm with a history-length of 2 and a reset timeout of 1000 milliseconds and tagless packets are discarded.

9.5.1. Recovery Algorithm Examples

Command	Purpose
`Switch-B#` **`configure terminal`**	Enter configuration mode.
`Switch-B(config)#` **`tsn frer 20`** `Switch-B(config-frer)#` **`recovery algorithm match`** `Switch-B(config-frer)#` **`recovery reset-timeout 4000`**	This selects the `match` algorithm on this FRER instance. It times out after 4000 milliseconds.

Command	Purpose
`Switch-B#` **`configure terminal`**	Enter configuration mode.

`Switch-B(config)# tsn frer 20`
`Switch-B(config-frer)# recovery algorithm vector history-length 32`
`Switch-B(config-frer)# recovery reset-timeout 500`
`Switch-B(config-frer)# recovery take-no-sequence`
	This selects the `vector` algorithm with a history length of 32 on this FRER instance.
	Here, we have configured the algorithm to reset after 500 milliseconds, and we allow it to pass frames arriving without an R-tag (default is *not* to allow this).

9.6. Latent Error Detection

The purpose of latent error detection is to raise a flag if the number of discarded packets is "relatively few" compared to the number of passed packets.

The algorithm relies on four user inputs:

1. `period`: The number of milliseconds between invoking the `test function`. Default is 2000.
2. `reset-period`: The number of milliseconds between invoking the `reset function`. Default is 30000.
3. `paths`: The number of member streams expected to ingress this FRER instance. Default is 2.
4. `difference`: The number of packets "allowed" to be in difference without raising the flag. Default is 100.

The `reset function` algorithm is as follows: Every `reset-period` milliseconds, read number of passed and discarded packet counters, and set a per-FRER instance variable, `CurDiff`, as follows:

```
CurDiff = passed_packets * (paths - 1) - discarded_packets;
```

The `test function` algorithm is as follows: Every `timeout` milliseconds, read the discarded and passed packet counters, and perform the following:

```
diff = Abs(CurDiff - (passed_packets * (paths - 1) - discarded_packets));

if (diff > difference) {
    raise_flag();
}
```

Basically, it says: If you expect N member streams to ingress this FRER instance, N-1 of these member streams are expected to be discarded, and only one is expected to pass.

To allow for some slack due to random packet losses and the fact that counters are not neccessarily read simultaneously, set the `difference` to account for that.

The `reset function` makes sure that `CurDiff` is updated to avoid that occassional packet losses don’t accumulate forever.

Once the flag is raised, it will become sticky in the sense that it requires a network administrator to clear it. The value of the sticky flag can be read with `show tsn frer <instance>` and it may give rise to a JSON notification and an SNMP trap. To clear it, use the `tsn frer <instance> reset latent-error` CLI command.

9.6.1. Latent Error Detection Example

This example shows how to configure latent error detection on switch B from example 1-1.

Command	Purpose
`Switch-B#` **`configure terminal`**	Enter configuration mode.
`Switch-B(config)#` **`tsn frer 20`**	Enter configuration of our FRER instance.
`Switch-B(config-frer)#` **`recovery latent-error-detection`** or `Switch-B(config-frer)#` **`recovery latent-error-detection paths 2 difference 1000 period 1000 reset-period 10000`** or `Switch-B(config-frer)#` **`no recovery latent-error-detection`**	First example shows how to enable latent error detection with default parameters. Second example shows how to enable it with custom parameters. Last example shows how to disable latent error detection.
`Switch-B(config-frer)#` **`end`**	End configuration mode.

The following shows how to see the current latent error detection status and how to clear the raised flag.

```
Switch-B# show tsn frer 20
Inst Operational State Mode       Latent Errors
---- ----------------- ---------- -------------
  20 Active            Recovery   Yes

Switch-B# tsn frer 20 reset latent-error

Switch-B# show tsn frer 20
Inst Operational State Mode       Latent Errors
---- ----------------- ---------- -------------
  20 Active            Recovery   No
```

9.7. Advanced Configuration Example

Before diving into status and statistics, let’s have a look at an advanced configuration example.

The following figure is very much alike Figure C-4 from IEEE-802.1CB-2017.

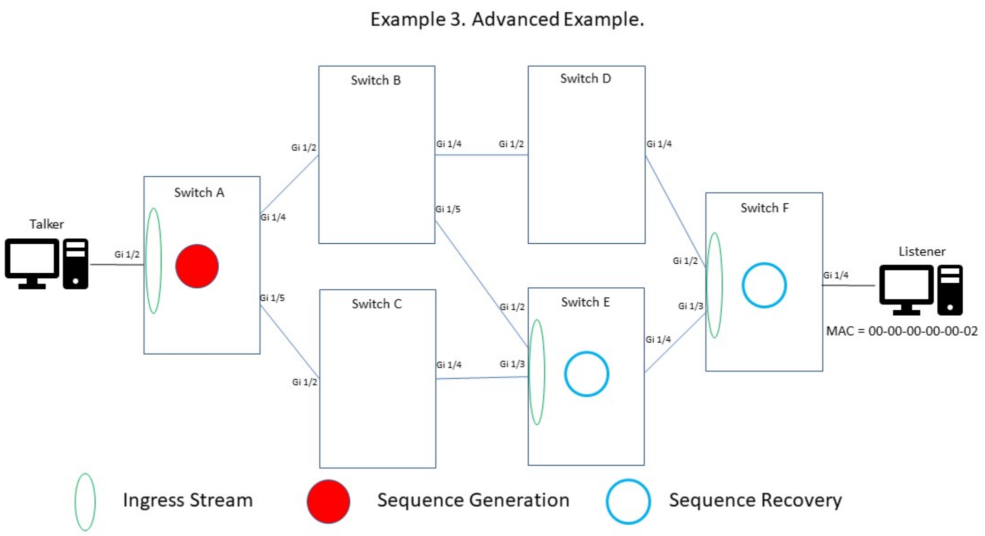


Figure 20. Advanced Configuration Example.

Here, the Talker is transmitting a stream, but has no FRER functions. Switch A transforms the ingress stream into a compound stream by sequencing the packets and splitting them into two streams.
These two streams go into switch B and C, where B splits one of the member streams further and D simply relays them from ingress to egress.
The - now - three member streams go into switch D and E. Switch D makes a simple relay of one of the member streams and switch E performs a recovery on the two others before presenting one recovered stream on egress.
The - now - two member streams then go into switch F, which performs recovery before egressing the restored stream to the Listener, which also doesn't have any FRER functions.

Switch A, E, and F are required to have FRER functions, whereas switch B, C, and D can be simple layer 2 (yet managed) forwarding switches.

Let's see how this plays out when using the MAC Table Forwarding method for ensuring that frames come from ingress to egress on any particular switch.

We assume that the Listener is having MAC address 00-00-00-00-00-02.

For simplicity, configuration of the management VLAN is left out.

9.7.1. Switch A

The configuration is identical to that of Switch A from Example 1-2 and boils down to:

```
vlan 35
 no flooding

no mac address-table learning vlan 35
mac address-table static 00-00-00-00-00-02 vlan 35 interface GigabitEthernet 1/4,5
spanning-tree mst te vlan 35

stream 17
 dmac 00-00-00-00-00-02

interface GigabitEthernet 1/2
 switchport hybrid allowed vlan none
 switchport mode hybrid
 stream-id 17

interface GigabitEthernet 1/4,5
 switchport hybrid allowed vlan 1,35
 switchport mode hybrid

tsn frer 10
 mode generation
 ingress stream-id-list 17
 frer-vlan 35
 egress interface GigabitEthernet 1/4,5
 admin-state enable
```

9.7.2. Switch B

A simple split of one frame into two can be done in many ways. Here, let’s provision the egress ports with the Listener’s MAC address.

Since the ingress stream is C-tagged with the FRER VLAN and since we carefully manage that VLAN, we can simply let both ingress and egress ports carry all VLANs (trunk mode) and disable flooding and learning in the FRER VLAN (VLAN 35) while using the MAC address table to forward.

We must also prevent MSTP from blocking VLAN 35.

This gives the following configuration.

```
vlan 35
 no flooding

no mac address-table learning vlan 35
mac address-table static 00-00-00-00-00-02 vlan 35 interface GigabitEthernet 1/4,5
spanning-tree mst te vlan 35

interface GigabitEthernet 1/2,4,5
 switchport mode trunk
```

9.7.3. Switch C and Switch D

These switches are also FRER-unaware and we use the same type of configuration as for Switch B, but with only one egress port.

```
vlan 35
 no flooding

no mac address-table learning vlan 35
mac address-table static 00-00-00-00-00-02 vlan 35 interface GigabitEthernet 1/4
spanning-tree mst te vlan 35

interface GigabitEthernet 1/2,4
 switchport mode trunk
```

9.7.4. Switch E

This switch performs recovery as does Switch B from Example 1-2. The only difference is that frames must be tagged with the FRER VLAN on egress and we are not terminating the recovery, so we don't strip the R-tag (`no recovery terminate`).

```
vlan 35
 no flooding

no mac address-table learning vlan 35
mac address-table static 00-00-00-00-00-02 vlan 35 interface GigabitEthernet 1/4
spanning-tree mst te vlan 35

stream 20
 dmac 00-00-00-00-00-02
 outer-tag vid 35

interface GigabitEthernet 1/2,3
 switchport mode trunk
 stream-id 20

interface GigabitEthernet 1/4
 switchport mode trunk

tsn frer 20
 mode recovery
 ingress stream-id-list 20
 frer-vlan 35
 egress interface GigabitEthernet 1/4
 admin-state enable
```

9.7.5. Switch F

This switch performs recovery exactly the same way as does Switch B from Example 1-2.

Whether to configure the egress port that connects to the Listener as an access port or as a trunk/hybrid port is up to the network administrator. Here, we have re-used that from Example 1-2.

```
vlan 1
vlan 100
 no flooding

no mac address-table learning vlan 100
mac address-table static 00-00-00-00-00-02 vlan 100 interface GigabitEthernet 1/4
spanning-tree mst te vlan 100

stream 20
 dmac 00-00-00-00-00-02
 outer-tag vid 35

interface GigabitEthernet 1/2,3
 switchport hybrid allowed vlan 1
 switchport mode hybrid
 stream-id 20

interface GigabitEthernet 1/4
 switchport access vlan 100

tsn frer 20
 mode recovery
 ingress stream-id-list 20
 frer-vlan 100
 egress interface GigabitEthernet 1/4
 recovery terminate
 admin-state enable
```

9.8. Bidirectional FRER

FRER is unidirectional, so by nature, there is no such thing as bidirectional FRER, but it is often desired to protect flows in both directions. The configuration can be tricky, so let’s go through an example.

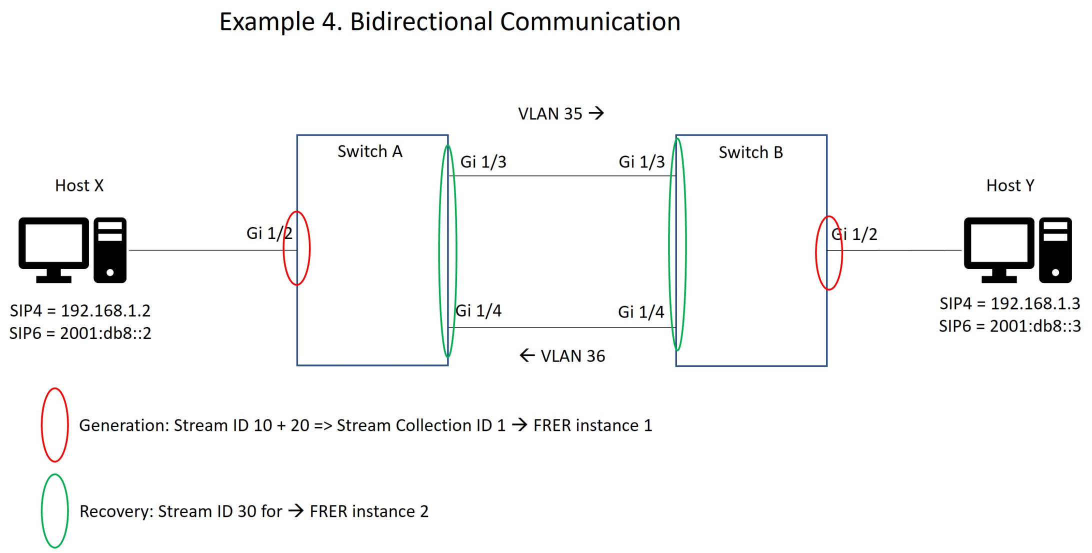


Figure 21. Bidirectional FRER.

The picture shows two hosts, X and Y, with individual IPv4 and IPv6 addresses. We wish to let all IP communication (both IPv4 and IPv6) between the two hosts be protected by FRER.

The stream concept does not allow for matching IPv4 ARP and IPv6 Neighbor Solicitation for particular IP addresses, so such traffic needs to go unprotected via the management VLAN. One could set up a stream that matches all ARP, but that would prevent the two hosts from communicating with other hosts (provided they need ARP/ Neighbor Solicitation).

In the following, we use the VLAN flooding model to configure the two switches, A and B. We let VLAN 1 be the management VLAN.

FRER VLAN 35 is used in direction from Host X to Host Y.
FRER VLAN 36 is used in direction from Host Y to Host X.

Each switch needs two FRER instances - one for generation (#1) and one for terminating recovery (#2).

The port numbers are changed compared to the previous examples, so that they now are symmetric. This eases the comparison of the two switches' running-config.

Most of the details as to why a particular port is configured as it is are already explained in Example 1 above, so only new things are shown below.

To separate things, we first configure Switch A’s FRER generator end (FRER VLAN 35).

Command (Switch-A Generator)	**Purpose**
`# `**`configure terminal`**	Enter configuration mode.
`(config)# `**`prompt Switch-A-Bidir`**	Give this switch a name. Use 'Bidir' to distinguish this setup from other setups in this guide.

`Switch-A-Bidir(config)#` **`spanning-tree mst te vlan 35`** `Switch-A-Bidir(config)#` **`no mac address-table learning vlan 35`**	The FRER VLAN must be member of the TE MSTI to prevent it from being blocked by MSTP. In the port configurations below, we make sure not to create loops in the FRER VLAN by only having egress ports as members of it. Also disable learning in the FRER VLAN to make sure frames flood.
`Switch-A-Bidir(config)#` **`stream 10`** `Switch-A-Bidir(config-stream)#` **`ipv4 dip 192.168.1.3/32`** `Switch-A-Bidir(config-stream)#` **`stream 20`** `Switch-A-Bidir(config-stream)#` **`ipv6 dip 2001:db8::3/128`** `Switch-A-Bidir(config-stream)#` **`exit`**	Create one stream that matches all 32 bits of Host Y's IPv4 destination address and another stream that matches all 128 bits of Host Y's IPv6 destination address. Notice that this requires two streams, since a single stream can only match either IPv4 or IPv6.
`Switch-A-Bidir(config)#` **`stream-collection 1`** `Switch-A-Bidir(config-stream-collection)#` **`stream-id-list 10,20`** `Switch-A-Bidir(config-stream-collection)#` **`exit`**	We wish to match both streams in the same FRER instance. For FRER generator instances, this is **only** possible with stream collections (FRER recovery instances may use a stream-id-list, which also enables the use of individual recovery).
`Switch-A-Bidir(config)#` **`interface GigabitEthernet 1/2`** `Switch-A-Bidir(config-if)#` **`stream-id 10,20`**	Assign the two streams to the ingress port.
`Switch-A-Bidir(config-if)#` **`interface GigabitEthernet 1/3,4`** `Switch-A-Bidir(config-if)#` **`switchport hybrid allowed vlan 1,35`** `Switch-A-Bidir(config-if)#` **`switchport mode hybrid`** `Switch-A-Bidir(config-if)#` **`exit`**	On the egress ports, allow the management VLAN and the FRER VLAN.
`Switch-A-Bidir(config)#` **`tsn frer 1`** `Switch-A-Bidir(config-frer)#` **`mode generation`** `Switch-A-Bidir(config-frer)#` **`ingress stream-collection-id 1`** `Switch-A-Bidir(config-frer)#` **`frer-vlan 35`** `Switch-A-Bidir(config-frer)#` **`egress interface GigabitEthernet 1/3,4`** `Switch-A-Bidir(config-frer)#` **`admin-state enable`**	Create the FRER generator instance and let all frames that hit the stream collection get classified to the FRER VLAN, and put R-tags on frames egressing Gi 1/3 and Gi 1/4.
`Switch-A-Bidir(config-frer)#` **`end`** `Switch-A-Bidir#`	We are now done with configuring the FRER generator of Switch A.

Then configure Switch A's FRER recovery end (FRER VLAN 36).

Command (Switch-A Recovery)	Purpose
`Switch-A-Bidir#` **`configure terminal`**	Enter configuration mode.
`Switch-A-Bidir(config)#` **`spanning-tree mst te vlan 36`** `Switch-A-Bidir(config)#` **`no mac address-table learning vlan 36`**	Same explanation as before.

`Switch-A-Bidir(config)# stream 30`
`Switch-A-Bidir(config-stream)# outer-tag vid 36`
`Switch-A-Bidir(config-stream)# exit`
	Create a stream that matches the FRER traffic coming from Switch B.

`Switch-A-Bidir(config)# interface GigabitEthernet 1/3,4` `Switch-A-Bidir(config-if)# stream-id 30`	Assign the stream to the ingress ports. We are happy with the VLAN memberships already configured on the two ports.

`Switch-A-Bidir(config-if)# interface GigabitEthernet 1/2`
`Switch-A-Bidir(config-if)# switchport hybrid allowed vlan 1,36`
`Switch-A-Bidir(config-if)# switchport hybrid egress-tag none`
`Switch-A-Bidir(config-if)# switchport mode hybrid`
`Switch-A-Bidir(config-if)# exit`
	On the egress ports, allow the management VLAN and the FRER VLAN.
	Since ARP/Neighbor Solication traffic is on VLAN 1 and IP traffic is on VLAN 36, and Host X isn’t VLAN aware, we need to make sure that both VLANs get untagged before frames egress Gi 1/2. This is the `egress-tag none` line.

`Switch-A-Bidir(config)# tsn frer 2`
`Switch-A-Bidir(config-frer)# mode recovery`
`Switch-A-Bidir(config-frer)# ingress stream-id-list 30`
`Switch-A-Bidir(config-frer)# frer-vlan 36`
`Switch-A-Bidir(config-frer)# egress interface GigabitEthernet 1/2`
`Switch-A-Bidir(config-frer)# recovery terminate`
`Switch-A-Bidir(config-frer)# admin-state enable`
	Create the FRER recovery instance and let all frames that hit the stream get classified to the FRER VLAN, and remove R-tags from the frames before they egress Gi 1/2.

`Switch-A-Bidir(config-frer)# end` `Switch-A-Bidir#`	We are now all done with configuration of Switch A.

Configuration of Switch B is more or less identical, but let’s summarize the two switches' configuration side-by-side below.

prompt Switch-A-Bidir	prompt Switch-B-Bidir

```
prompt Switch-A-Bidir

spanning-tree mst te vlan 35-36
no mac address-table learning vlan 35,36

stream 10
 ipv4 dip 192.168.1.3/32

stream 20
 ipv6 dip 2001:db8::3/128

stream 30
 outer-tag vid 36

stream-collection 1
 stream-id-list 10,20

interface GigabitEthernet 1/2
 switchport hybrid allowed vlan 1,36
 switchport hybrid egress-tag none
 switchport mode hybrid
 stream-id 10
 stream-id 20

interface GigabitEthernet 1/3,4
 switchport hybrid allowed vlan 1,35
 switchport mode hybrid
 stream-id 30

tsn frer 1
 mode generation
 ingress stream-collection-id 1
 frer-vlan 35
 egress interface GigabitEthernet 1/3-4
 admin-state enable

tsn frer 2
 mode recovery
 ingress stream-id-list 30
 frer-vlan 36
 egress interface GigabitEthernet 1/2
 recovery terminate
 admin-state enable
```

```
prompt Switch-B-Bidir

spanning-tree mst te vlan 35-36
no mac address-table learning vlan 35,36

stream 10
 ipv4 dip 192.168.1.2/32

stream 20
 ipv6 dip 2001:db8::2/128

stream 30
 outer-tag vid 35

stream-collection 1
 stream-id-list 10,20

interface GigabitEthernet 1/2
 switchport hybrid allowed vlan 1,35
 switchport hybrid egress-tag none
 switchport mode hybrid
 stream-id 10
 stream-id 20

interface GigabitEthernet 1/3,4
 switchport hybrid allowed vlan 1,36
 switchport mode hybrid
 stream-id 30

tsn frer 1
 mode generation
 ingress stream-collection-id 1
 frer-vlan 36
 egress interface GigabitEthernet 1/3-4
 admin-state enable

tsn frer 2
 mode recovery
 ingress stream-id-list 30
 frer-vlan 35
 egress interface GigabitEthernet 1/2
 recovery terminate
 admin-state enable
```

Let’s ping Host Y from Host X:

```
Host-X# ping -c 5 192.168.1.3
PING 192.168.1.3 (192.168.1.3): 56 data bytes
64 bytes from 192.168.1.3: seq=0 ttl=64 time=8.424 ms
64 bytes from 192.168.1.3: seq=1 ttl=64 time=2.014 ms
64 bytes from 192.168.1.3: seq=2 ttl=64 time=2.019 ms
64 bytes from 192.168.1.3: seq=3 ttl=64 time=2.036 ms
64 bytes from 192.168.1.3: seq=4 ttl=64 time=3.515 ms

--- 192.168.1.3 ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max = 2.014/3.601/8.424 ms

Host-X# ping -c 5 2001:db8::3
PING 2001:db8::3 (2001:db8::3): 56 data bytes
64 bytes from 2001:db8::3: seq=0 ttl=64 time=16.069 ms
64 bytes from 2001:db8::3: seq=1 ttl=64 time=2.262 ms
64 bytes from 2001:db8::3: seq=2 ttl=64 time=2.277 ms
64 bytes from 2001:db8::3: seq=3 ttl=64 time=2.280 ms
64 bytes from 2001:db8::3: seq=4 ttl=64 time=2.385 ms

--- 2001:db8::3 ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max = 2.262/5.054/16.069 ms
```

The first packet of the two pings take a little longer due to ARP/Neighbor Solicitation.

The statistics look like:

```
Switch-A-Bidir# show tsn frer stati
Inst Oper. State    Mode       Egress I/F Resets       Passed       Discarded
---- -------------- ---------- ---------- ------------ ------------ ------------
   1 Active         Generation                       0           10
   2 Active         Recovery   Gi 1/2               11           12           12
```

The ten passed frames of instance 1 are ping requests.
The ten passed frames of instance 2 are ping replies.
The last two passed frames of instance 2 are a Neighbor Advertisement from Host Y directly to Host X and a subsequent Neighbor Solicitation from Host Y directly to Host X.
The ARP Request and Neighbor Solicitation from Host X to Host Y go through the management VLAN, as does the ARP Reply from Host Y to Host X.

Finally, we can see that MSTP indeed has blocked one of the Gi 1/3,4 ports:

```
Switch-B# show spanning-tree
CIST Bridge STP Status
Bridge ID    : 32768.C2-1D-15-06-ED-00
Root ID      : 32768.00-01-C1-00-00-00
Root Port    : 3
Root PathCost: 20000
Regional Root: 32768.C2-1D-15-06-ED-00
Int. PathCost: 0
Max Hops     : 20
TC Flag      : Steady
TC Count     : 158
TC Last      : 0d 00:02:07
Port       Port Role       State       Pri  PathCost  Edge  P2P         Uptime
---------  --------------  ----------  ---  --------  ----  ---  -------------
Gi 1/2     DesignatedPort  Forwarding  128     20000  No    Yes  0d 03:27:04
Gi 1/3     RootPort        Forwarding  128     20000  No    Yes  0d 03:03:16
Gi 1/4     AlternatePort   Discarding  128     20000  No    Yes  0d 03:03:06
```

9.9. Status

FRER instances' status is displayed with the `show tsn frer [instance(s)] [details]` ICLI command in EXEC mode.

An overview is displayed if omitting `details`, like the following:

```
# show tsn frer
Inst Operational State Mode       Latent Errors
---- ----------------- ---------- --------------
   3 Admin disabled    Generation
  10 Active            Generation
  14 Admin disabled    Recovery
 101 Active (warnings) Recovery   No
#
```

In this particular case, we have four FRER instances, two of which are in generation mode, and two in recovery. Two are administratively disabled and two are active, one of which (FRER instance #101) has operational warnings.

The active recovery instances also have a column with `Latent Errors`, which may show `Yes` or `No` if latent error discovery is enabled. Otherwise, it shows `Check disabled`.

It is interesting to know why FRER instance #101 has warnings. To see them, ask for details:

```
# show tsn frer 101 details
Instance:                101
Operational state:       Active
Mode:                    Recovery
Ingress Stream IDs:      20
FRER VLAN:               35
Egress interfaces:       GigabitEthernet 1/4
Recovery algorithm:      Vector (with history-length 2)
Reset timeout:           1000 ms
Take-no-sequence:        No
Terminating FRER:        No
Latent error detection: Enabled (with difference 100 packets, period 2000 ms, paths
= 2, and reset period 30000 ms)
Latent errors:           No
Operational warnings:    At least one of the ingress streams doesn't exist
#
```

The operational warnings are shown at the bottom of the output. Here, it says that at least one of the ingress streams doesn't exist, meaning that the stream with ID 20 has not yet been created (or has been deleted), so the FRER instance is active but no frames will get mapped to it. Once stream ID 20 has been created and installed on at least one ingress port, will the status become updated automatically.

Besides the operational warnings, the details consist of various configuration properties and whether any latent errors have been discovered.

It is easy to make configurational mistakes. The IStaX software attempts to disclose the most obvious, such as stream configuration errors. It is important to note that the detected warnings do not constitute the complete list of possible configuration mistakes.

The `Operational warnings` both include warnings that are due to configurational mistakes and warnings that may happen due to operational issues, such as link down on an egress port.

At the time of writing, the possible operational warnings are:

- `At least one of the ingress streams doesn't exist`: At least one of the stream IDs You have specified in the stream-id-list doesn't exist. Create the stream of change the stream-id-list.
- `The specified stream collection ID does not exist`: You have specified a stream collection ID that is non-existent. Create the stream collection or change the stream-collection-id of the FRER instance.
- `Unable to attach to at least one of the ingress streams, possibly because it is part of a stream collection`: At least one of the stream IDs exists, but it is part of a stream collection, in which case it cannot be attached to directly. Either change the FRER instance to attach to the stream collection or change the stream collection to not include the stream ID.
- `Unable to attach to the specified stream collection`: There is no good reason for this one.

- `At least one of the ingress streams has configurational warnings`: One or more of the specified stream IDs has configurational warnings. Use `show stream status <stream-id> details` to see them.
- `The specified stream collection has configurational warnings`: The stream collection this FRER instance is attached to has configurational warnings. Use `show stream-collection status <stream-collection-id> details` to see them.
- `There is an overlap between ingress and egress ports`: The ingress ports are given by the ports specified in the stream-id-list or the streams that are members of the stream collection. If at least one of these ports is also specified as an egress port for this FRER instance, you will get this warnings.
- `In generation mode, at leaast two egress ports should be configured`: The FRER instance is in generation mode, but only one egress port is specified. Change it to two or more.
- `At least one of the ingress ports doesn’t have link`: Check that the cable is correctly inserted into all ingress ports.
- `At least one of the egress ports doesn’t have link`: Check that the cable is correctly inserted into all egress ports.
- `At least one of the egress ports is not member of the FRER VLAN`: You have specified an egress port that is not member of the FRER VLAN.
- `At least one of the egress ports is blocked by STP`: The RSTP protocol has blocked one or more of the egress ports. This may happen if there is not link on one of the egress ports or if one of the egress ports use RSTP instead of MSTP, or if the FRER VLAN is not a member of the TE MSTI.
- `At least one of the egress ports is blocked by MSTP`: The FRER VLAN is blocked on one or more of the egress ports. This may happen if there is not link on one of the egress ports or if the FRER VLAN is not a member of the TE MSTI.

9.10. Statistics

The counters mentioned throughout this configuration guide can be shown with `show tsn frer statistics [instance(s)] [details]` ICLI command in EXEC mode. Counters are only shown if an instance is active.

An overview, containing number of resets, number of passed packets, and number of discarded packets, is displayed if omitting `details`:

```
# show tsn frer statistics
Inst Oper. State    Mode       Egress I/F Resets       Passed       Discarded
---- -------------- ---------- ---------- ------------ ------------ ------------
   3 Admin disabled Generation                       0          112
  10 Active         Generation                       1
  14 Admin disabled Recovery
  20 Active         Recovery   Gi 1/4                1         1004         1004
                               Gi 1/5                1         1004         1004
```

To see detailed counters, add `details` to the command. In the following example, only details for one instance is shown:

```
# show tsn frer statistics 20 details
Instance:            20
Operational state:   Active
Mode:                Recovery

Egress interface:    GigabitEthernet 1/4
Passed:              1004
Discarded:           1004
Out-of-order:        0
Rogue:               0
Lost:                1
Tagless:             0
Resets:              6
Latent error resets: 1863

Egress interface:    GigabitEthernet 1/5
Passed:              1004
Discarded:           1004
Out-of-order:        0
Rogue:               0
Lost:                1
Tagless:             0
Resets:              6
Latent error resets: 1863
```

This FRER instance is in non-individual recovery mode, so it only has compound counters per egress interface.

If the ingress stream is split into two (one per ingress port) and individual recovery is enabled, the output of the detailed statistics could look like the following:

```
# show tsn frer statistics 20 details
Instance:            20
Operational state:   Active
Mode:                Recovery

Egress interface:    GigabitEthernet 1/4
Ingress Stream IDs:  Compound               20          21
                     ------------ ------------ ------------
Passed:                      1004         1004         1004
Discarded:                   1004            0            0
Out-of-order:                   0            0            0
Rogue:                          0            0            0
Lost:                           2            2            2
Tagless:                        0            0            0
Resets:                         4            4            4
Latent error resets:            0            0            0

Egress interface:    GigabitEthernet 1/5
Ingress Stream IDs:  Compound               20          21
                     ------------ ------------ ------------
Passed:                      1004         1004         1004
Discarded:                   1004            0            0
Out-of-order:                   0            0            0
Rogue:                          0            0            0
Lost:                           2            2            2
Tagless:                        0            0            0
Resets:                         4            4            4
Latent error resets:            0            0            0
```

Now, the statistics is presented in a table per egress interface. The compound counters are shown in the first column. Subsequent columns hold individual counters per ingress stream ID.

In generation mode, there are two counters: The number of frames that has matched the configured stream and the number of times the sequence number generator has been reset.

Counters may be cleared with the `clear tsn frer [instance(s)] statistics` ICLI command in EXEC mode.

NOTE In generation mode, if FRER and PSFP share the same ingress stream, clearing FRER statistics also clears the PSFP statistics and vice versa.

9.11. Reset Control

9.11.1. Reset Recovery Functions

Although of doubtable use, it is possible to force the recovery function(s) to reset for a particular FRER instance. This corresponds to setting `frerSeqRcvyReset` (802.1CB-2017, clause 10.4.1.4). It is not possible to control this per egress port or per individual recovery function; all functions get reset. Here’s how:

```
# show tsn frer 20 statistics
Inst Oper. State    Mode       Egress I/F Resets       Passed       Discarded
---- ------------- ---------- ---------- ----------- ------------ ------------
  20 Active         Recovery   Gi 1/4               6         1004         1004
                               Gi 1/5               6         1004         1004
# tsn frer 20 reset
# show tsn frer 20 statistics
Inst Oper. State    Mode       Egress I/F Resets       Passed       Discarded
---- ------------- ---------- ---------- ----------- ------------ ------------
  20 Active         Recovery   Gi 1/4               7         1004         1004
                               Gi 1/5               7         1004         1004
```

The example above first shows counters for the FRER instance, then performs the reset, and then shows counters again. This shows that the number of resets has increased from 6 to 7.

9.11.2. Reset Sequence Number Generator

If the FRER instance is in generation mode, the same command has another effect: It resets the sequence number generator to provide sequence number 0 in the next transmitted packet. Example:

```
# show tsn frer statistics 10
Inst Oper. State    Mode       Egress I/F Resets       Passed       Discarded
---- ------------- ---------- ---------- ----------- ------------ ------------
  10 Active         Generation                      1          112
# tsn frer 10 reset
# show tsn frer statistics 10
Inst Oper. State    Mode       Egress I/F Resets       Passed       Discarded
---- ------------- ---------- ---------- ----------- ------------ ------------
  10 Active         Generation                      2          112
```

Here, we also show the counters for the FRER instance before and after, just to emphasize that it has an effect to issue the reset command.

One thing worth noting is that once a FRER instance - whether in recovery or generation mode - is getting enabled (`admin-state enable`), an automatic reset happens, causing the counters to count to 1.

9.12. Pitfalls

There are a couple of pitfalls that are important to note:

9.12.1. PSFP and FRER Generator

A stream or a stream collection may be used by PSFP and FRER simultaneously. However, if FRER is configured as a generator, and PSFP discards traffic, there will be holes in the sequence numbers of the R-tags. The receiving switches that attempt to perform recovery will have a hard time doing so, depending on the recovery algorithm and the vector length (in case of vector algorithm).

9.12.2. Frames Absorbed by the Generator Switch

Suppose you have a FRER instance in generator mode and attach a stream (or stream collection) that matches frames that will be absorbed by the switch. Such frames will still draw a sequence number, but will never be transmitted, thereby causing holes in the sequence numbers of the R-tags, and the recovery function on subsequent switches will have a hard time.

9.12.3. FRER Generator: Multiple Ingress Ports

In the FRER generator case, it is recommended only to add streams to one single ingress port. Suppose a stream gets added to two ports, X and Y, and suppose a frame arriving on port X gets sequence number 17, and a frame arriving shortly after on port Y gets sequence number 18. If the frame arriving on Port X is much longer than the frame arriving on port Y, the frame on port Y may leave the egress port(s) before the frame on Port X, causing the two sequence numbers to be swapped as seen by the receiving switch(es).