# Configuring 802.1Q VLAN Interfaces

This module describes the configuration and management of 802.1Q VLAN interfaces on the Cisco NCS 6000 Series Router.

The IEEE 802.1Q specification establishes a standard method for tagging Ethernet frames with VLAN membership information, and defines the operation of VLAN bridges that permit the definition, operation, and administration of VLAN topologies within a bridged LAN infrastructure.

The 802.1Q standard is intended to address the problem of how to divide large networks into smaller parts so broadcast and multicast traffic does not use more bandwidth than necessary. The standard also helps provide a higher level of security between segments of internal networks.

**Feature History for Configuring 802.1Q VLAN Interfaces**

| Release | Modification |
|---|---|
| Release 5.0.0 | This feature was introduced. |


- Prerequisites for Configuring 802.1Q VLAN Interfaces, on page 1
- Information About Configuring 802.1Q VLAN Interfaces, on page 2
- How to Configure 802.1Q VLAN Interfaces, on page 3
- , on page 8
- Configuration Examples for VLAN Interfaces, on page 9
- Additional References, on page 11


# Prerequisites for Configuring 802.1Q VLAN Interfaces

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Before configuring 802.1Q VLAN interfaces, be sure that the following conditions are met:

- You must have configured a a 10-Gigabit Ethernet interface , a Fast Ethernet interface, or an Ethernet Bundle.

# Information About Configuring 802.1Q VLAN Interfaces

To configure 802.1Q VLAN interfaces, you must understand the following concepts:

## 802.1Q VLAN Overview

A VLAN is a group of devices on one or more LANs that are configured so that they can communicate as if they were attached to the same wire, when in fact they are located on a number of different LAN segments. Because VLANs are based on logical instead of physical connections, they are very flexible for user and host management, bandwidth allocation, and resource optimization.

The IEEE 802.1Q protocol standard addresses the problem of dividing large networks into smaller parts so broadcast and multicast traffic does not consume more bandwidth than necessary. The standard also helps provide a higher level of security between segments of internal networks.

The 802.1Q specification establishes a standard method for inserting VLAN membership information into Ethernet frames.

Cisco IOS XR software supports VLAN subinterface configuration on 10-Gigabit Ethernet, and Fast Ethernet interfaces.

## 802.1Q Tagged Frames

The IEEE 802.1Q tag-based VLAN uses an extra tag in the MAC header to identify the VLAN membership of a frame across bridges. This tag is used for VLAN and quality of service (QoS) priority identification. The VLANs can be created statically by manual entry or dynamically through Generic Attribute Registration Protocol (GARP) VLAN Registration Protocol (GVRP). The VLAN ID associates a frame with a specific VLAN and provides the information that switches must process the frame across the network. A tagged frame is four bytes longer than an untagged frame and contains two bytes of Tag Protocol Identifier (TPID) residing within the type and length field of the Ethernet frame and two bytes of Tag Control Information (TCI) which starts after the source address field of the Ethernet frame.

## Subinterfaces

Subinterfaces are logical interfaces created on a hardware interface. These software-defined interfaces allow for segregation of traffic into separate logical channels on a single hardware interface as well as allowing for better utilization of the available bandwidth on the physical interface.

Subinterfaces are distinguished from one another by adding an extension on the end of the interface name and designation. For instance, the Ethernet subinterface 23 on the physical interface designated TenGigE 0/1/0/0 would be indicated by TenGigE 0/1/0/0.23.

Before a subinterface is allowed to pass traffic it must have a valid tagging protocol encapsulation and VLAN identifier assigned. All Ethernet subinterfaces always default to the 802.1Q VLAN encapsulation. However, the VLAN identifier must be explicitly defined.

## Subinterface MTU

The subinterface maximum transmission unit (MTU) is inherited from the physical interface with an additional four bytes allowed for the 802.1Q VLAN tag.

## Native VLAN

Each physical port may have a native VLAN assigned. All untagged frames are assigned to the LAN specified in the PVID parameter. When received packet is tagged with the PVID, that packet is treated as if it was untagged. Therefore, the configuration associated with the native VLAN must be placed on the main interface. The native VLAN allows the coexistence of VLAN-aware bridge or stations with VLAN-unaware bridges or stations.

## VLAN Sub-interfaces on Ethernet Bundles

An Ethernet bundle is a group of one or more Ethernet ports that are aggregated together and treated as a single link. Multiple VLAN sub-interfaces can be added to a single Ethernet bundle.

The procedure for creating VLAN sub-interfaces on an Ethernet bundle is exactly the same as the procedure for creating VLAN sub-interfaces on a physical Ethernet interface. To create a VLAN subinterface on an Ethernet bundle, see the How to Configure 802.1Q VLAN Interfaces, on page 3 section later in this module.

# How to Configure 802.1Q VLAN Interfaces

This section contains the following procedures:

## Configuring 802.1Q VLAN Subinterfaces

This task explains how to configure 802.1Q VLAN sub-interfaces. To remove these sub-interfaces, see the Removing an 802.1Q VLAN Subinterface, on page 8 section of this module.

## Configuring 802.1Q VLAN Subinterfaces

### SUMMARY STEPS

1. **configure**
2. **interface** {**TenGigE** | **Bundle-Ether**} *interface-path-id.subinterface*
3. **encapsulation dot1q**
4. **ipv4 address** *ip-address mask*
5. **exit**
6. Repeat Step 2 through Step 5 to define the rest of the VLAN subinterfaces.
7. Do one of the following:
   - **end**
   -
   - **commit**
8. **show ethernet trunk bundle-ether** *instance*

**DETAILED STEPS**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters XR configuration mode. |
| **Step 2** | **interface** {**TenGigE** \| **Bundle-Ether**} *interface-path-id.subinterface*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# interface TenGigE 0/2/0/4.10` | Enters subinterface configuration mode and specifies the interface type, location, and subinterface number.<br>• Replace the *interface-path-id* argument with one of the following instances:<br>  • Physical Ethernet interface instance, or with an Ethernet bundle instance. Naming notation is *rack*/*slot*/*module*/*port*, and a slash between values is required as part of the notation.<br>  • Ethernet bundle instance. Range is from 1 through 65535.<br>• Replace the *subinterface* argument with the subinterface value. Range is from 0 through 4095.<br>• Naming notation is *interface-path-id.subinterface*, and a period between arguments is required as part of the notation. |
| **Step 3** | **encapsulation dot1q**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-subif)# encapsulation dot1q 100` | Sets the Layer 2 encapsulation of an interface. |
| **Step 4** | **ipv4 address** *ip-address mask*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-subif)# ipv4 address 178.18.169.23/24` | Assigns an IP address and subnet mask to the subinterface.<br>• Replace *ip-address* with the primary IPv4 address for an interface.<br>• Replace *mask* with the mask for the associated IP subnet. The network mask can be specified in either of two ways:<br>  • The network mask can be a four-part dotted decimal address. For example, 255.0.0.0 indicates that each bit equal to 1 means that the corresponding address bit belongs to the network address.<br>  • The network mask can be indicated as a slash (/) and number. For example, /8 indicates that the first 8 bits of the mask are ones, and the corresponding bits of the address are network address. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **exit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-subif)# exit` | (Optional) Exits the subinterface configuration mode.<br>• The exit command is not explicitly required. |
| **Step 6** | Repeat Step 2 through Step 5 to define the rest of the VLAN subinterfaces. | — |
| **Step 7** | Do one of the following:<br>• **end**<br>•<br>• **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# end`<br>**Example:**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# commit` | Saves configuration changes.<br>• When you issue the **end** command, the system prompts you to commit changes:<br>`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`<br>• • Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to XR EXEC mode.<br>• Entering **no** exits the configuration session and returns the router to XR EXEC mode without committing the configuration changes.<br>• Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |
| **Step 8** | **show ethernet trunk bundle-ether** *instance*<br>**Example:**<br>`RP/0/RP0/CPU0:router# show ethernet trunk bundle-ether 5` | (Optional) Displays the interface configuration.<br>The Ethernet bundle instance range is from 1 through 65535. |

# Configuring an Attachment Circuit on a VLAN

Use the following procedure to configure an attachment circuit on a VLAN.

### SUMMARY STEPS

1. **configure**
2. **interface** [**GigabitEthernet** | **TenGigE** | Bundle-Ether | **TenGigE**] *interface-path*] *id.subinterface* **l2transport**
3. **dot1q vlan** *vlan-id*
4. **l2protocol** {**cdp** | **pvst** | **stp** | **vtp**}{[**forward** | **tunnel**][**experimental** *bits*]|**drop**}
5. Do one of the following:

- **end**
- 
- **commit**

6. **show interfaces** [**GigabitEthernet** | **TenGigE**] *interface-path-id.subinterface*

**DETAILED STEPS**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure terminal` | Enters XR configuration mode. |
| **Step 2** | **interface** [**GigabitEthernet** \| **TenGigE** \| Bundle-Ether \| **TenGigE**] *interface-path*] *id.subinterface* **l2transport**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# interfaceTenGigE 0/1/0/0.1 l2transport` | Enters subinterface configuration and specifies the interface type, location, and subinterface number.<br>• Replace the *interface-path-id* argument with one of the following instances:<br>  • Physical Ethernet interface instance,or Ethernet bundle instance. Naming notation is *rack*/*slot*/*module*/*port*, and a slash between values is required as part of the notation.<br>  • Ethernet bundle instance. Range is from 1 through 65535.<br>• Replace the *subinterface* argument with the subinterface value. Range is from 0 through 4095.<br>• Naming notation is *instance.subinterface*, and a period between arguments is required as part of the notation.<br>**Note** You must include the **l2transport** keyword in the command string; otherwise, the configuration creates a Layer 3 subinterface rather that an AC. |
| **Step 3** | **dot1q vlan** *vlan-id*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-subif)# dot1q vlan 10 vlan any` | Assigns a VLAN AC to the subinterface.<br>• Replace the vlan-id argument with a subinterface identifier. Range is from 1 to 4094 inclusive (0 and 4095 are reserved). To configure a basic Dot1Q AC, use the following syntax:<br>**dot1q vlan**<br>vlan-id<br>• To configure a Q-in-Q AC, use the following syntax:<br>**dot1q vlan** |

| | Command or Action | Purpose |
|---|---|---|
| | | vlan-id<br>**vlan**<br>vlan-id<br>• To configure a Q-in-Any AC, use the following syntax:<br>**dot1q vlan**<br>vlan-id<br>**vlan**<br>**any** |
| **Step 4** | **l2protocol** {**cdp** \| **pvst** \| **stp** \| **vtp**} {[**forward** \| **tunnel**][**experimental** *bits*]\|**drop**}<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if-l2)# l2protocol stp tunnel`<br>**Example:** | Configures Layer 2 protocol tunneling and protocol data unit (PDU) filtering on an interface.<br>Possible protocols and options are:<br>• **cdp**—Cisco Discovery Protocol (CDP) tunneling and data unit parameters.<br>• **pvst**—Configures VLAN spanning tree protocol tunneling and data unit parameters.<br>• **stp**—spanning tree protocol tunneling and data unit parameters.<br>• **vtp**—VLAN trunk protocol tunneling and data unit parameters.<br>• **tunnel**—(Optional) Tunnels the packets associated with the specified protocol.<br>• **experimental** *bits*—(Optional) Modifies the MPLS experimental bits for the specified protocol.<br>• **drop**—(Optional) Drop packets associated with the specified protocol. |
| **Step 5** | Do one of the following:<br>• **end**<br>•<br>• **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if-l2)# end`<br>**Example:**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if-l2)# commit` | Saves configuration changes.<br>• When you issue the **end** command, the system prompts you to commit changes:<br>`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`<br>• • Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to XR EXEC mode.<br>• Entering **no** exits the configuration session and returns the router to XR EXEC mode without committing the configuration changes.<br>• Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. |

| | Command or Action | Purpose |
|---|---|---|
| | | • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |
| **Step 6** | **show interfaces** [**GigabitEthernet** \| **TenGigE**] *interface-path-id.subinterface*<br>**Example:**<br>`RP/0/RP0/CPU0:router# show interfaces TenGigE 0/3/0/0.1` | (Optional) Displays statistics for interfaces on the router. |

## What to Do Next

- To configure a Point-to Point pseudo-wire cross connect on the AC, see the "*Implementing MPLS Layer 2 VPNs"* module of the *Multiprotocol Label Switching Configuration Guide.*
- To attach Layer 3 service policies, such as Multiprotocol Label Switching (MPLS) or Quality of Service(QoS), to the VLAN, refer to the appropriate configuration guide.

# Removing an 802.1Q VLAN Subinterface

This task explains how to remove 802.1Q VLAN subinterfaces that have been previously configured using the "Configuring 802.1Q VLAN Subinterfaces, on page 3" section in this module.

### SUMMARY STEPS

1. **configure**
2. **no interface** {**TenGigE** | **Bundle-Ether**] *interface-path-id.subinterface*
3. Repeat Step 2 to remove other VLAN subinterfaces.
4. Do one of the following:
   - **end**
   -
   - **commit**
5. **show ethernet trunk bundle-ether** *instance*

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters XR configuration mode. |
| **Step 2** | **no interface** {**TenGigE** \| **Bundle-Ether**] *interface-path-id.subinterface* | Removes the subinterface, which also automatically deletes all the configuration applied to the subinterface. |

| | Command or Action | Purpose |
|---|---|---|
| | **Example:**<br>`RP/0/RP0/CPU0:router(config)# no interface TenGigE 0/2/0/4.10` | • Replace the *instance* argument with one of the following instances:<br>• Physical Ethernet interface instance, or with an Ethernet bundle instance. Naming notation is *rack*/*slot*/*module*/*port*, and a slash between values is required as part of the notation.<br>• Ethernet bundle instance. Range is from 1 through 65535.<br>• Replace the *subinterface* argument with the subinterface value. Range is from 0 through 4095.<br>Naming notation is *instance.subinterface*, and a period between arguments is required as part of the notation. |
| **Step 3** | Repeat Step 2 to remove other VLAN subinterfaces. | — |
| **Step 4** | Do one of the following:<br>• **end**<br>•<br>• **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# end`<br>**Example:**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# commit` | Saves configuration changes.<br>• When you issue the **end** command, the system prompts you to commit changes:<br>`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`<br>• • Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to XR EXEC mode.<br>• Entering **no** exits the configuration session and returns the router to XR EXEC mode without committing the configuration changes.<br>• Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |
| **Step 5** | **show ethernet trunk bundle-ether** *instance*<br>**Example:**<br>`RP/0/RP0/CPU0:router# show ethernet trunk bundle-ether 5` | (Optional) Displays the interface configuration.<br>The Ethernet bundle instance range is from 1 through 65535. |

# Configuration Examples for VLAN Interfaces

This section contains the following example:

# VLAN Subinterfaces: Example

The following example shows how to create three VLAN subinterfaces at one time:

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface TenGigE 0/2/0/4.1
RP/0/RP0/CPU0:router(config-subif)# dot1q vlan 10
RP/0/RP0/CPU0:router(config-subif)# ipv4 address 10.0.10.1/24
RP/0/RP0/CPU0:router(config-subif)# interface TenGigE0/2/0/4.2

RP/0/RP0/CPU0:router(config-subif)# dot1q vlan 20
RP/0/RP0/CPU0:router(config-subif)# ipv4 address 10.0.20.1/24

RP/0/RP0/CPU0:router(config-subif)# interface TenGigE0/2/0/4.3

RP/0/RP0/CPU0:router(config-subif)# dot1q vlan 30
RP/0/RP0/CPU0:router(config-subif)# ipv4 address 10.0.30.1/24
 RP/0/RP0/CPU0:router(config-subif)# commit
RP/0/RP0/CPU0:router(config-subif)# exit
RP/0/RP0/CPU0:router(config)# exit
RP/0/RP0/CPU0:router# show vlan trunks summary
VLAN trunks: 1,
  1 are 802.1Q (Ether).
Sub-interfaces: 3,
  3 are up.
802.1Q VLANs: 3,
  3 have VLAN Ids.
RP/0/RP0/CPU0:router# show vlan interface
          -
Interface               Encapsulation    Outer 2nd   Service  MTU    LineP
                                         VLAN  VLAN                    State
Te0/6/0/0.1             802.1Q              1       L3        9604  up
Te0/6/0/4.1             802.1Q              1       L3        9604  up
```

The following example shows how to create two VLAN subinterfaces on an Ethernet bundle:

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface bundle-ether 2
RP/0/RP0/CPU0:router(config-if)# ipv4 address 192.168.2.1/24
RP/0/RP0/CPU0:router(config-if)# exit
RP/0/RP0/CPU0:router(config)# interface bundle-ether 2.1
RP/0/RP0/CPU0:router(config-subif)#  dot1q vlan 10
RP/0/RP0/CPU0:router(config-subif)# ipv4 address 192.168.100.1/24
RP/0/RP0/CPU0:router(config-subif)# exit
RP/0/RP0/CPU0:router(config)# interface bundle-ether 2.2
RP/0/RP0/CPU0:router(config-subif)# dot1q vlan 20
RP/0/RP0/CPU0:router(config-subif)# ipv4 address 192.168.200.1/24
RP/0/RP0/CPU0:router(config-subif)# exit
RP/0/RP0/CPU0:router(config)# commit
```

The following example shows how to create a basic dot1Q AC:

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface GigabitEthernet 0/0/0/0.1
RP/0/RP0/CPU0:router(config-subif)# l2transport
RP/0/RP0/CPU0:router(config-subif)# dot1q vlan 20
RP/0/RP0/CPU0:router(config-subif)# commit
RP/0/RP0/CPU0:router(config-subif)# exit
RP/0/RP0/CPU0:router(config)# exit
```

# Additional References

The following sections provide references related to VLAN interface configuration.

### Related Documents

### Standards

| Standards | Title |
|---|---|
| No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature | — |

### MIBs

| MIBs | MIBs Link |
|---|---|
| There are no applicable MIBs for this module. | To locate and download MIBs for selected platforms using Cisco IOS XR Software, use the Cisco MIB Locator found at the following URL:http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml |

| RFCs | Title |
|---|---|
| No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature. | — |

### Technical Assistance

| Description | Link |
|---|---|
| The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content. | http://www.cisco.com/techsupport |