# Cisco IOS XR Interface and Hardware Component Configuration Guide for the Cisco XR 12000 Series Router

Cisco IOS XR Software Release 4.3.x

**Cisco Systems, Inc.**
www.cisco.com

Cisco has more than 200 offices worldwide.
Addresses, phone numbers, and fax numbers
are listed on the Cisco website at
www.cisco.com/go/offices.

Text Part Number: OL-28390-02

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB's public domain version of the UNIX operating system. All rights reserved. Copyright © 1981, Regents of the University of California.

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED "AS IS" WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Cisco and the Cisco logo are trademarks or registered trademarks of Cisco and/or its affiliates in the U.S. and other countries. To view a list of Cisco trademarks, go to this URL: www.cisco.com/go/trademarks. Third-party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1110R)

Any Internet Protocol (IP) addresses used in this document are not intended to be actual addresses. Any examples, command display output, and figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses in illustrative content is unintentional and coincidental.

*Cisco IOS XR Interface and Hardware Component Configuration Guide for the Cisco XR 12000 Series Router*
© 2010-2013 Cisco Systems, Inc. All rights reserved.



**Preface** HC-xvii

Changes to This Document HC-xvii

Obtaining Documentation and Submitting a Service Request HC-xvii

**New and Changed Information in Release 4.3.x** HC-1

**Preconfiguring Physical Interfaces on Cisco IOS XR Software** HC-3

Contents HC-4

Prerequisites for Preconfiguring Physical Interfaces HC-4

Information About Preconfiguring Physical Interfaces HC-4

- Physical Interface Preconfiguration Overview HC-5
- Benefits of Interface Preconfiguration HC-5
- Use of the Interface Preconfigure Command HC-5
- Active and Standby RPs and Virtual Interface Configuration HC-6

How to Preconfigure Physical Interfaces HC-6

Configuration Examples for Preconfiguring Physical Interfaces HC-8

- Preconfiguring an Interface: Example HC-8

Additional References HC-9

- Related Documents HC-9
- Standards HC-9
- MIBs HC-9
- RFCs HC-9
- Technical Assistance HC-10

**Configuring ATM Interfaces on Cisco IOS XR Software** HC-11

Contents HC-12

Prerequisites for Implementing ATM HC-12

Information About ATM HC-13

- VC-Class Mapping HC-14
- VP-Tunnels HC-14
- F5 OAM on ATM Interfaces HC-15
- ILMI on ATM Interfaces HC-15
- Layer 2 VPN on ATM Interfaces HC-15

Circuit Emulation over Packet on the Cisco 2-Port Channelized T3/E3 ATM and Circuit Emulation Shared Port Adapter **HC-17**

ATM Layer 2 QoS **HC-18**

Configuring ATM Interfaces **HC-21**

Bringing Up an ATM Interface **HC-22**

Configuring Optional ATM Interface Parameters **HC-24**

How to Create and Configure a Point-to-Point ATM Subinterface with a PVC **HC-26**

Creating a Point-to-Point ATM Subinterface with a PVC **HC-26**

Configuring Optional Point-to-Point ATM PVC Parameters **HC-28**

How to Create and Configure a VP-Tunnel **HC-31**

Creating and Configuring a VP-Tunnel on an ATM Interface **HC-32**

Creating and Configuring Subinterfaces with PVCs on a VP-tunnel **HC-35**

How to Configure a Layer 2 Attachment Circuit **HC-38**

Creating a Layer 2 Port Mode AC **HC-38**

Configuring Optional Parameters on a Layer 2 Port Mode AC **HC-40**

Creating an ATM Layer 2 Subinterface with a PVC **HC-42**

Configuring Optional ATM Layer 2 PVC Parameters **HC-43**

Creating an ATM Layer 2 Subinterface with a PVP **HC-46**

Configuring Optional ATM Layer 2 PVP Parameters **HC-47**

How to Create and Configure a VC-Class **HC-50**

Creating and Configuring a VC-Class **HC-50**

Attaching a VC-Class to a Point-to-Point ATM Main Interface **HC-53**

Attaching a VC-Class to a Point-to-Point ATM Subinterface **HC-54**

Attaching a VC-Class to a PVC on an ATM Subinterface **HC-55**

How to Configure ILMI on ATM Interfaces **HC-57**

Enabling ILMI on an ATM Interface **HC-57**

Disabling ILMI on an ATM Interface **HC-59**

How to Configure Channelized ATM **HC-61**

How to Configure Clear Channel ATM with Virtual Path (VP) Tunnels **HC-64**

Attaching a Service-Policy to an Attachment Circuit **HC-67**

ATM Configuration: Examples **HC-70**

ATM Interface Bring Up and Configuration: Example **HC-70**

Point-To-Point ATM Subinterface Configuration: Example **HC-70**

VP-Tunnel Configuration: Example **HC-72**

Layer 2 AC Creation and Configuration: Example **HC-73**

VC-Class Creation and Configuration: Example **HC-73**

Channelized ATM Configuration: Example **HC-74**

Clear Channel ATM with Virtual Path (VP) Tunnels Configuration: Example **HC-75**

ATM Layer 2 QoS Configuration: Examples **HC-75**

Verifying ATM Layer 2 QoS Configuration: Examples **HC-77**

Additional References HC-79
Related Documents HC-79
Standards HC-79
MIBs HC-79
RFCs HC-79
Technical Assistance HC-80

**Advanced Configuration and Modification of the Management Ethernet Interface on Cisco IOS XR Software** HC-81

Contents HC-81
Prerequisites for Configuring Management Ethernet Interfaces HC-82
Information About Configuring Management Ethernet Interfaces HC-82
Default Interface Settings HC-82
How to Perform Advanced Management Ethernet Interface Configuration HC-83
Configuring a Management Ethernet Interface HC-83
Configuring the Duplex Mode for a Management Ethernet Interface HC-85
Configuring the Speed for a Management Ethernet Interface HC-86
Modifying the MAC Address for a Management Ethernet Interface HC-88
Verifying Management Ethernet Interface Configuration HC-89
Configuration Examples for Management Ethernet Interfaces HC-90
Configuring a Management Ethernet Interface: Example HC-90
Additional References HC-91
Related Documents HC-91
Standards HC-91
MIBs HC-91
RFCs HC-91
Technical Assistance HC-92

**Configuring Ethernet Interfaces on Cisco IOS XR Software** HC-93

Contents HC-94
Prerequisites for Configuring Ethernet Interfaces HC-94
Information About Configuring Ethernet HC-95
Default Configuration Values for Gigabit Ethernet and 10-Gigabit Ethernet HC-95
Default Configuration Values for Fast Ethernet HC-96
Layer 2 VPN on Ethernet Interfaces HC-96
Gigabit Ethernet Protocol Standards Overview HC-97
MAC Address HC-98
MAC Accounting HC-99
Ethernet MTU HC-99
Flow Control on Ethernet Interfaces HC-99

802.1Q VLAN HC-100
VRRP HC-100
HSRP HC-100
Duplex Mode on Fast Ethernet Interfaces HC-101
Fast Ethernet Interface Speed HC-101
Link Autonegotiation on Ethernet Interfaces HC-101
Carrier Delay on Ethernet Interfaces HC-102
How to Configure Ethernet HC-102
Configuring Ethernet Interfaces HC-103
Configuration Examples for Ethernet HC-112
Configuring an Ethernet Interface: Example HC-113
Configuring a Fast Ethernet Interface: Example HC-113
Configuring MAC-Accounting: Example HC-114
Configuring a Layer 2 VPN AC: Example HC-114
Where to Go Next HC-114
Additional References HC-115
Related Documents HC-115
Standards HC-115
MIBs HC-115
RFCs HC-115
Technical Assistance HC-116

**Configuring Ethernet OAM on Cisco IOS XR Software** HC-117

Contents HC-118
Prerequisites for Configuring Ethernet OAM HC-118
Restrictions for Configuring Ethernet OAM HC-118
Information About Configuring Ethernet OAM HC-119
Ethernet Link OAM HC-119
Ethernet CFM HC-120
Ethernet SLA (Y.1731 Performance Monitoring) HC-135
How to Configure Ethernet OAM HC-140
Configuring Ethernet Link OAM HC-140
Configuring Ethernet CFM HC-149
Configuring Ethernet SLA HC-169
Configuration Examples for Ethernet OAM HC-181
Configuration Examples for EOAM Interfaces HC-181
Configuration Examples for Ethernet CFM HC-183
Configuration Examples for Ethernet SLA HC-192
Where to Go Next HC-199

Additional References HC-200
Related Documents HC-200
Standards HC-200
MIBs HC-200
RFCs HC-200
Technical Assistance HC-201

**Configuring Link Bundling on Cisco IOS XR Software HC-203**

Contents HC-205
Prerequisites for Configuring Link Bundling HC-205
Prerequisites for Configuring Link Bundling on a Cisco XR 12000 Series Router HC-205
Information About Configuring Link Bundling HC-207
Link Bundling Overview HC-208
Features and Compatible Characteristics of Link Bundles HC-209
Link Aggregation Through LACP HC-210
LACP Short Period Time Intervals HC-211
Load Balancing HC-212
Unequal Bandwidth on Link Bundles HC-213
QoS and Link Bundling HC-213
MPLS-TE and FRR over Link Bundles HC-213
VLANs on an Ethernet Link Bundle HC-215
Link Bundle Configuration Overview HC-215
Nonstop Forwarding During RP Switchover HC-215
Link Switchover HC-216
How to Configure Link Bundling HC-216
Configuring Ethernet Link Bundles HC-216
Configuring EFP Load Balancing on an Ethernet Link Bundle HC-221
Configuring VLAN Bundles HC-223
Configuring POS Link Bundles HC-229
Configuring the Default LACP Short Period Time Interval HC-234
Configuring Custom LACP Short Period Time Intervals HC-236
Configuration Examples for Link Bundling HC-245
Example: Configuring an Ethernet Link Bundle HC-246
Example: Configuring a VLAN Link Bundle HC-246
Example: Configuring a POS Link Bundle HC-247
Example: Configuring EFP Load Balancing on an Ethernet Link Bundle HC-247
Examples: Configuring LACP Short Periods HC-248
Example: Configuring MPLS-TE and FRR over Link Bundles HC-249
Additional References HC-250

Related Documents HC-250

Standards HC-250

MIBs HC-250

RFCs HC-250

Technical Assistance HC-251

**Configuring Virtual Loopback and Null Interfaces on Cisco IOS XR Software HC-253**

Contents HC-253

Prerequisites for Configuring Virtual Interfaces HC-254

Information About Configuring Virtual Interfaces HC-254

Virtual Loopback Interface Overview HC-254

Null Interface Overview HC-254

Virtual Management Interface Overview HC-255

Active and Standby RPs and Virtual Interface Configuration HC-255

How to Configure Virtual Interfaces HC-255

Configuring Virtual Loopback Interfaces HC-256

Configuring Null Interfaces HC-257

Configuring Virtual IPV4 Interfaces HC-258

Configuration Examples for Virtual Interfaces HC-259

Configuring a Loopback Interface: Example HC-259

Configuring a Null Interface: Example HC-260

Configuring a Virtual IPv4 Interface: Example HC-260

Additional References HC-261

Related Documents HC-261

Standards HC-261

MIBs HC-261

RFCs HC-261

Technical Assistance HC-262

**Configuring Channelized SONET/SDH on Cisco IOS XR Software HC-263**

Contents HC-263

Prerequisites for Configuring Channelized SONET/SDH HC-263

Information About Configuring Channelized SONET/SDH HC-264

Channelized SONET Overview HC-264

Channelized SDH Overview HC-269

Default Configuration Values for Channelized SONET/SDH HC-272

How to Configure Channelized SONET/SDH HC-273

Configuring SONET T3 and VT1.5-Mapped T1 Channels HC-273

Configuring Packet over SONET Channels HC-278

Configuring a Clear Channel SONET Controller for T3 HC-281
Configuring Channelized SONET APS HC-284
Configuring SDH AU-3 HC-287
Configuring SDH AU-4 HC-294
Configuration Examples for Channelized SONET HC-299
Channelized SONET Examples HC-299
Channelized SDH Examples HC-302
Additional References HC-306
Related Documents HC-306
Standards HC-306
MIBs HC-306
RFCs HC-306
Technical Assistance HC-307

**Configuring Circuit Emulation over Packet on Cisco IOS XR Software HC-309**

Contents HC-309
Prerequisites for Configuration HC-310
Overview of Circuit Emulation over Packet Service HC-310
Information About Configuring CEoP Channelized SONET/SDH HC-311
Channelized SONET and SDH Overview HC-311
Default Configuration Values for Channelized SONET/SDH HC-316
Clock Distribution HC-317
Mode Change for CEoP SPA HC-319
How to implement CEM HC-320
Configuring SONET VT1.5-Mapped T1 Channels and Creating CEM Interface HC-320
Configuring SDH AU-3 Mapped to C11-T1 or C12-E1 HC-323
Configuring the Cisco 24-Port Channelized T1/E1 Circuit Emulation and Channelized ATM SPA and Creating CEM Interface HC-329
Configuring the Cisco 2-Port Channelized T3/E3 Circuit Emulation and Channelized ATM SPA and Creating CEM Interface HC-332
Configuring CEM Interface HC-337
Configuring Clocking HC-345
Show Commands for CEM HC-347
Configuration Examples for CEM HC-348
Circuit Emulation Interface Configuration: Examples HC-348
Clock Recovery : Example HC-351
Additional References HC-352
Related Documents HC-352
Standards HC-352

MIBs HC-353

RFCs HC-353

Technical Assistance HC-353

**Configuring Clear Channel SONET Controllers on Cisco IOS XR Software** HC-355

Contents HC-356

Prerequisites for Configuring Clear Channel SONET Controllers HC-356

Information About Configuring SONET Controllers HC-357

SONET Controller Overview HC-357

Default Configuration Values for SONET Controllers HC-358

SONET APS HC-359

How to Configure Clear Channel SONET Controllers HC-360

Configuring a Clear Channel SONET Controller HC-360

Configuring SONET APS HC-364

Configuring a Hold-off Timer to Prevent Fast Reroute from Being Triggered HC-369

Configuration Examples for SONET Controllers HC-371

SONET Controller Configuration: Example HC-371

SONET APS Group Configuration: Example HC-371

Additional References HC-372

Related Documents HC-372

Standards HC-372

MIBs HC-372

RFCs HC-372

Technical Assistance HC-373

**Configuring Clear Channel T3/E3 and Channelized T3 and T1/E1 Controllers on Cisco IOS XR Software** HC-375

Contents HC-375

Prerequisites for Configuring T3/E3 Controllers HC-375

Information About T3/E3 Controllers and Serial Interfaces HC-376

Configuration Overview HC-378

Default Configuration Values for T3 and E3 Controllers HC-379

Default Configuration Values for T1 and E1 Controllers HC-379

How to Configure Clear Channel T3/E3 Controllers and Channelized T1/E1 Controllers HC-380

Setting the Card Type HC-381

Configuring a Clear Channel E3 Controller HC-383

Modifying the Default E3 Controller Configuration HC-385

Configuring a Clear Channel T3 Controller HC-388

Configuring a Channelized T3 Controller HC-389

Modifying the Default T3 Controller Configuration HC-391

Configuring a T1 Controller HC-394
Configuring an E1 Controller HC-397
Configuring BERT HC-400
Configuration Examples HC-406
Configuring a Clear Channel T3 Controller: Example HC-406
Configuring a T3 Controller with Channelized T1 Controllers: Example HC-406
Configuring BERT on a T3 Controller: Example HC-408
Additional References HC-409
Related Documents HC-409
Standards HC-409
MIBs HC-409
RFCs HC-410
Technical Assistance HC-410

**Configuring POS Interfaces on Cisco IOS XR Software HC-411**

Contents HC-413
Prerequisites for Configuring POS Interfaces HC-413
Information About Configuring POS Interfaces HC-413
Default Settings for POS Interfaces HC-413
Cisco HDLC Encapsulation HC-414
PPP Encapsulation HC-414
Keepalive Timer HC-415
Frame Relay Encapsulation HC-416
How to Configure a POS Interface HC-418
Bringing Up a POS Interface HC-418
Configuring Optional POS Interface Parameters HC-421
Creating a Point-to-Point POS Subinterface with a PVC HC-424
Configuring Optional PVC Parameters HC-426
Modifying the Keepalive Interval on POS Interfaces HC-428
How to Configure a Layer 2 Attachment Circuit HC-430
Creating a Layer 2 Frame Relay Subinterface with a PVC HC-431
Configuring Optional Layer 2 PVC Parameters HC-433
Configuring Optional Layer 2 Subinterface Parameters HC-435
Configuration Examples for POS Interfaces HC-437
Bringing Up and Configuring a POS Interface with Cisco HDLC Encapsulation: Example HC-437
Configuring a POS Interface with Frame Relay Encapsulation: Example HC-438
Configuring a POS Interface with PPP Encapsulation: Example HC-439
Additional References HC-440
Related Documents HC-440

Standards HC-440
MIBs HC-441
RFCs HC-441
Technical Assistance HC-441

**Configuring Serial Interfaces on Cisco IOS XR Sotware HC-443**
Contents HC-444
Prerequisites for Configuring Serial Interfaces HC-444
Information About Configuring Serial Interfaces HC-445
High-Level Overview: Serial Interface Configuration on Clear-Channel SPAs HC-446
High-Level Overview: Serial Interface Configuration on Channelized SPAs HC-446
Cisco HDLC Encapsulation HC-447
PPP Encapsulation HC-448
Keepalive Timer HC-449
Frame Relay Encapsulation HC-450
Layer 2 Tunnel Protocol Version 3-Based Layer 2 VPN on Frame Relay HC-451
Default Settings for Serial Interface Configurations HC-452
Serial Interface Naming Notation HC-452
IPHC Overview HC-453
How to Configure Serial Interfaces HC-455
Bringing Up a Serial Interface HC-455
Configuring Optional Serial Interface Parameters HC-458
Creating a Point-to-Point Serial Subinterface with a PVC HC-461
Configuring Optional PVC Parameters HC-464
Modifying the Keepalive Interval on Serial Interfaces HC-466
How to Configure a Layer 2 Attachment Circuit HC-468
Creating a Serial Layer 2 Subinterface with a PVC HC-468
Configuring Optional Serial Layer 2 PVC Parameters HC-470
Configuring IPHC HC-473
Configuration Examples for Serial Interfaces HC-479
Bringing Up and Configuring a Serial Interface with Cisco HDLC Encapsulation: Example HC-479
Configuring a Serial Interface with Frame Relay Encapsulation: Example HC-480
Configuring a Serial Interface with PPP Encapsulation: Example HC-482
IPHC Configuration: Examples HC-482
Additional References HC-484
Related Documents HC-484
Standards HC-484
MIBs HC-485
RFCs HC-485
Technical Assistance HC-485

**Configuring Frame Relay Cisco IOS XR Software** HC-487

Contents HC-488

Prerequisites for Configuring Frame Relay HC-488

Information About Frame Relay Interfaces HC-488

Frame Relay Encapsulation HC-488

Multilink Frame Relay (FRF.16) HC-491

End-to-End Fragmentation (FRF.12) HC-495

Configuring Frame Relay HC-495

Modifying the Default Frame Relay Configuration on an Interface HC-495

Disabling LMI on an Interface with Frame Relay Encapsulation HC-498

Configuring Multilink Frame Relay Bundle Interfaces HC-500

Configuring FRF.12 End-to-End Fragmentation on a Channelized Frame Relay Serial Interface HC-506

Configuration Examples for Frame Relay HC-511

Optional Frame Relay Parameters: Example HC-512

Multilink Frame Relay: Example HC-514

End-to-End Fragmentation: Example HC-515

Additional References HC-515

Related Documents HC-516

Standards HC-516

MIBs HC-516

RFCs HC-516

Technical Assistance HC-517

**Configuring PPP on Cisco IOS XR Software** HC-519

Contents HC-519

Prerequisites for Configuring PPP HC-519

Information About PPP HC-520

PPP Authentication HC-520

Multilink PPP HC-522

T3 SONET Channels HC-523

How to Configure PPP HC-523

Modifying the Default PPP Configuration HC-523

Configuring PPP Authentication HC-527

Disabling an Authentication Protocol HC-535

Configuring Multilink PPP HC-540

Configuration Examples for PPP HC-547

Configuring a POS Interface with PPP Encapsulation: Example HC-547

Configuring a Serial Interface with PPP Encapsulation: Example HC-548

Configuring MLPPP: Example **HC-548**

Verifying Multilink PPP Configurations **HC-549**

Additional References **HC-553**

Related Documents **HC-553**

Standards **HC-553**

MIBs **HC-553**

RFCs **HC-553**

Technical Assistance **HC-554**

**Configuring 802.1Q VLAN Interfaces on Cisco IOS XR Software** **HC-555**

Contents **HC-555**

Prerequisites for Configuring 802.1Q VLAN Interfaces **HC-556**

Information About Configuring 802.1Q VLAN Interfaces **HC-556**

802.1Q VLAN Overview **HC-556**

802.1Q Tagged Frames **HC-556**

Subinterfaces **HC-557**

Subinterface MTU **HC-557**

Native VLAN **HC-557**

Layer 2 VPN on VLANs **HC-557**

How to Configure 802.1Q VLAN Interfaces **HC-558**

Configuring 802.1Q VLAN Subinterfaces **HC-558**

Configuring an Attachment Circuit on a VLAN **HC-561**

Removing an 802.1Q VLAN Subinterface **HC-564**

Configuration Examples for VLAN Interfaces **HC-566**

VLAN Subinterfaces: Example **HC-566**

Additional References **HC-568**

Related Documents **HC-568**

Standards **HC-568**

MIBs **HC-568**

RFCs **HC-568**

Technical Assistance **HC-569**

**Configuring Tunnel Interfaces on Cisco IOS XR Software** **RC-571**

Contents **RC-571**

Prerequisites for Configuring Tunnel Interfaces **RC-572**

Information About Configuring Tunnel Interfaces **RC-572**

Tunnel Interfaces Overview **RC-572**

Virtual Interface Naming Convention **RC-572**

Tunnel-IPSec Overview **RC-573**

Tunnel-IPSec Naming Convention RC-573
Crypto Profile Sets RC-573
How to Configure Tunnel Interfaces RC-574
Configuring Tunnel-IPSec Interfaces RC-574
Configuration Examples for Tunnel Interfaces RC-576
Tunnel-IPSec: Example RC-576
Where to Go Next RC-577
Additional References RC-577
Related Documents RC-577
Standards RC-577
MIBs RC-578
RFCs RC-578
Technical Assistance RC-578

# Preface

The *Cisco IOS XR Interface and Hardware Component Configuration Guide for the Cisco XR 12000 Series Router* provides information and procedures related to router interface and hardware configuration.

The preface contains the following sections:

- Changes to This Document
- Obtaining Documentation and Submitting a Service Request

# Changes to This Document

Table 1 lists the technical changes made to this document since it was first printed.

***Table 1 Changes to This Document***

| Revision | Date | Change Summary |
|---|---|---|
| OL-28390-02 | May 2013 | Republished with documentation updates for Cisco IOS XR Release 4.3.1 features. |
| OL-28390-01 | December 2012 | Initial release of this document. |

# Obtaining Documentation and Submitting a Service Request

For information on obtaining documentation, submitting a service request, and gathering additional information, see the monthly *What's New in Cisco Product Documentation*, which also lists all new and revised Cisco technical documentation, at:

http://www.cisco.com/en/US/docs/general/whatsnew/whatsnew.html

Subscribe to the *What's New in Cisco Product Documentation* as a Really Simple Syndication (RSS) feed and set content to be delivered directly to your desktop using a reader application. The RSS feeds are a free service and Cisco currently supports RSS version 2.0.

# New and Changed Information in Release 4.3.x

This table summarizes the new and changed feature information for the Cisco IOS XR Interface and Hardware Component Configuration Guide for the Cisco XR 12000 Series Router, and tells you where they are documented.

*Table 2 New and Changed Features*

| Feature | Description | Introduced/Changed in Release | Where Documented |
|---|---|---|---|
| No new features | — | — | — |

# Preconfiguring Physical Interfaces on Cisco IOS XR Software

This module describes the preconfiguration of physical interfaces on the Cisco IOS XR Software.

Preconfiguration is supported for the following types of interfaces and controllers:

- ATM
- Fast Ethernet
- Gigabit Ethernet
- 10-Gigabit Ethernet
- Management Ethernet
- Packet-over-SONET/SDH (POS)
- Serial
- SONET controllers and channelized SONET controllers

Preconfiguration allows you to configure modular services cards before they are inserted into the router. When the cards are inserted, they are instantly configured.

The preconfiguration information is created in a different system database tree (known as the *preconfiguration directory* on the route processor [RP]), rather than with the regularly configured interfaces.

There may be some preconfiguration data that cannot be verified unless the modular services card is present, because the verifiers themselves run only on the modular services card. Such preconfiguration data is verified when the modular services card is inserted and the verifiers are initiated. A configuration is rejected if errors are found when the configuration is copied from the preconfiguration area to the active area.

**Note** Only physical interfaces can be preconfigured.

**Feature History for Preconfiguring Physical Interfaces**

| Release | Modification |
|---|---|
| Release 3.2 | Preconfiguration support was introduced. |
| Release 3.3.0 | The following interface preconfiguration support was introduced:<br>• Management Ethernet interface<br>• Fast Ethernet<br>• Serial Interface |
| Release 3.4.0 | ATM interface preconfiguration was introduced. |

# Contents


- Prerequisites for Preconfiguring Physical Interfaces, page 4
- Information About Preconfiguring Physical Interfaces, page 4
- How to Preconfigure Physical Interfaces, page 6
- Configuration Examples for Preconfiguring Physical Interfaces, page 8
- Additional References, page 9


# Prerequisites for Preconfiguring Physical 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 preconfiguring physical interfaces, be sure that the following conditions are met:

- Preconfiguration drivers and files are installed. Although it may be possible to preconfigure physical interfaces without a preconfiguration driver installed, the preconfiguration files are required to set the interface definition file on the router that supplies the strings for valid interface names.

# Information About Preconfiguring Physical Interfaces

To preconfigure interfaces, you must understand the following concepts:


- Physical Interface Preconfiguration Overview, page 5
- Benefits of Interface Preconfiguration, page 5
- Use of the Interface Preconfigure Command, page 5
- Active and Standby RPs and Virtual Interface Configuration, page 6

## Physical Interface Preconfiguration Overview

Preconfiguration is the process of configuring interfaces before they are present in the system. Preconfigured interfaces are not verified or applied until the actual interface with the matching location (rack/slot/module) is inserted into the router. When the anticipated modular services card is inserted and the interfaces are created, the precreated configuration information is verified and, if successful, immediately applied to the router's running configuration.

**Note** When you plug the anticipated modular services card in, make sure to verify any preconfiguration with the appropriate **show** commands.

Use the **show run** command to see interfaces that are in the preconfigured state.

**Note** We recommend filling out preconfiguration information in your site planning guide, so that you can compare that anticipated configuration with the actual preconfigured interfaces when that card is installed and the interfaces are up.

**Tip** Use the **commit best-effort** command to save the preconfiguration to the running configuration file. The **commit best-effort** command merges the target configuration with the running configuration and commits only valid configuration (best effort). Some configuration might fail due to semantic errors, but the valid configuration still comes up.

## Benefits of Interface Preconfiguration

Preconfigurations reduce downtime when you add new cards to the system. With preconfiguration, the new modular services card can be instantly configured and actively running during modular services card bootup.

Another advantage of performing a preconfiguration is that during a card replacement, when the modular services card is removed, you can still see the previous configuration and make modifications.

## Use of the Interface Preconfigure Command

Interfaces that are not yet present in the system can be preconfigured with the **interface preconfigure** command in global configuration mode.

The **interface preconfigure** command places the router in interface configuration mode. Users should be able to add any possible interface commands. The verifiers registered for the preconfigured interfaces verify the configuration. The preconfiguration is complete when the user enters the **end** command, or any matching **exit** or global configuration mode command.

**Note** It is possible that some configurations cannot be verified until the modular services card is inserted.

**Note** Do not enter the **no shutdown** command for new preconfigured interfaces, because the **no** form of this command removes the existing configuration, and there is no existing configuration.

Users are expected to provide names during preconfiguration that will match the name of the interface that will be created. If the interface names do not match, the preconfiguration cannot be applied when the interface is created. The interface names must begin with the interface type that is supported by the router and for which drivers have been installed. However, the slot, port, subinterface number, and channel interface number information cannot be validated.

**Note** Specifying an interface name that already exists and is configured (or an abbreviated name like e0/3/0/0) is not permitted.

## Active and Standby RPs and Virtual Interface Configuration

The standby RP is available and in a state in which it can take over the work from the active RP should that prove necessary. Conditions that necessitate the standby RP to become the active RP and assume the active RP's duties include:

- Failure detection by a watchdog
- Standby RP is administratively commanded to take over
- Removal of the active RP from the chassis

If a second RP is not present in the chassis while the first is in operation, a second RP may be inserted and will automatically become the standby RP. The standby RP may also be removed from the chassis with no effect on the system other than loss of RP redundancy.

After switchover, the virtual interfaces will all be present on the standby (now active) RP. Their state and configuration will be unchanged, and there will have been no loss of forwarding (in the case of tunnels) over the interfaces during the switchover. The Cisco XR 12000 Series Router uses nonstop forwarding (NSF) over tunnels through the switchover of the host RP.

**Note** The user does not need to configure anything to guarantee that the standby interface configurations are maintained.

# How to Preconfigure Physical Interfaces

This task describes only the most basic preconfiguration of an interface.

### SUMMARY STEPS

1. **configure**
2. **interface preconfigure** *type interface-path-id*
3. **ipv4 address** *ip-address subnet-mask*
4. Configure additional interface parameters.

5. **end**
   or
   **commit**
6. **exit**
7. **exit**
8. **show running-config**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| Step 1 | **`configure`**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| Step 2 | **`interface preconfigure`** *`type interface-path-id`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface preconfigure GigabitEthernet 0/1/0/0` | Enters interface preconfiguration mode for an interface, where *type* specifies the supported interface type that you want to configure and *interface-path-id* specifies the location where the interface will be located in *rack*/*slot*/*module*/*port* notation. |
| Step 3 | **`ipv4 address`** *`ip-address subnet-mask`*<br>`or`<br>**`ipv4 address`** *`ip-address`*`/`*`prefix`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if-pre)# ipv4 address 192.168.1.2/32` | Assigns an IP address and mask to the interface. |
| Step 4 | Configure additional interface parameters, as described in this manual in the configuration chapter that applies to the type of interface that you are configuring. | |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **end**<br>or<br>**commit** best-effort<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if-pre)# end`<br>or<br>`RP/0/0/CPU0:router(config-if-pre)# 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)?`<br>– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 best-effort** command to save the configuration changes to the running configuration file and remain within the configuration session. The **commit best-effort** command merges the target configuration with the running configuration and commits only valid changes (best effort). Some configuration changes might fail due to semantic errors. |
| **Step 6** | **show running-config**<br><br>**Example:**<br>`RP/0/0/CPU0:router# show running-config` | (Optional) Displays the configuration information currently running on the router. |

# Configuration Examples for Preconfiguring Physical Interfaces

This section contains the following example:

Preconfiguring an Interface: Example, page 8

## Preconfiguring an Interface: Example

The following example shows how to preconfigure a basic Ethernet interface:

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# interface preconfigure GigabitEthernet 0/1/0/0
RP/0/0/CPU0:router(config-if)# ipv4 address 192.168.1.2/32
RP/0/0/CPU0:router(config-if)# commit
```

# Additional References

The sections that follow provide references related to the preconfiguration of physical interfaces.

## Related Documents

| Related Topic | Document Title |
|---|---|
| Master command reference | *Cisco XR 12000 Series Router Master Command Listing* |
| Interface configuration commands | *Cisco XR 12000 Series Router Interface and Hardware Component Command Reference* |
| Initial system bootup and configuration information | *Cisco IOS XR Getting Started Guide for the Cisco XR 12000 Series Router* |
| Information about user groups and task IDs | *Cisco IOS XR Task ID Reference Guide* |

## 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:<br>http://www.cisco.com/go/mibs |

## RFCs

| 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/support |

# Configuring ATM Interfaces on Cisco IOS XR Software

This module describes how to configure ATM on the Cisco XR 12000 Series Router using Cisco IOS XR software. ATM is a cell-switching and multiplexing technology that is widely used in Wide Area Networks (WANs). ATM protocol standards enable point-to-point, point-to-multipoint, and broadcast services connections using various slow- and high-speed network media. Connectivity between two ATM permanent virtual circuits (PVCs) is established using ATM signaling mechanisms. Various ATM signaling standards are defined by these ATM forum standards:

- UNI Version 3.0, Version 3.1, and Version 4.0
- ITU
- IETF

**Feature History for Configuring ATM Interfaces on Cisco IOS XR Software**

| Release | Modification |
|---|---|
| Release 3.4.0 | This feature was introduced on the Cisco XR 12000 Series Router on the following hardware:<br>• Cisco XR 12000 Series 4-Port OC-3c/STM-1c ATM ISE Line Card, multimode<br>• Cisco XR 12000 Series 4-Port OC-3c/STM-1c ATM ISE Line Card, single-mode<br>• Cisco XR 12000 Series 4-port OC-12/STM-4 ATM multimode ISE line card with SC connector<br>• Cisco XR 12000 Series 4-port OC-12/STM-4 ATM single-mode, intermediate-reach ISE line card with SC Connector |
| Release 3.4.1 | The Layer 2 Virtual Private Network (L2VPN) feature was first supported on ATM interfaces on the Cisco XR 12000 Series Router. |
| Release 3.5.0 | OAM configuration was first supported on L2VPN ATM interfaces. |

| | |
|---|---|
| Release 3.7.0 | Support for the following cards was added:<br>• 1-port Clear Channel OC-3 ATM SPA<br>• 1-port Clear Channel OC-12 ATM SPA<br>• 3-port Clear Channel OC-3 ATM SPA<br>The following new information was added:<br>• Circuit Emulation over Packet (CEoP) on the OC3/OC-12-ATM-V2 Shared Port Adapter on the Cisco XR 12000 Series Router.<br>• Channelized ATM<br>• Clear Channel ATM with Virtual Path (VP) Tunnels |
| Release 3.8.0 | The display outputs for the examples in these sections were updated:<br>• How to Configure Clear Channel SONET Controllers<br>• Configuring a Hold-off Timer to Prevent Fast Reroute from being Triggered |
| Release 3.9.0 | ATM Layer 2 QoS was added. |
| Release 4.0.1 | The following support was added for Circuit Emulation over Packet (CEoP) Shared Port Adapters (SPAs):<br>• ATM Layer 2 QoS<br>• Inverse Multiplexing (IMA) |

# Contents

- Prerequisites for Implementing ATM, page 12
- Information About ATM, page 13
- Configuring ATM Interfaces, page 21
- ATM Configuration: Examples, page 70
- Additional References, page 79

# Prerequisites for Implementing ATM

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.

# Information About ATM

This section provides overviews of these features:


- VC-Class Mapping, page 14
- VP-Tunnels, page 14
- F5 OAM on ATM Interfaces, page 15
- ILMI on ATM Interfaces, page 15
- Layer 2 VPN on ATM Interfaces, page 15
- Circuit Emulation over Packet on the Cisco 2-Port Channelized T3/E3 ATM and Circuit Emulation Shared Port Adapter, page 17
- ATM Layer 2 QoS, page 18


Network nodes use ATM connections to transfer bits of data organized as 53-byte ATM cells. User information (such as voice, video, and data) is segmented into ATM cells on one end of the connection, and then reassembled on the other end of the connection. ATM Adaptation Layer (AAL) defines the conversion of user information into ATM cells. AAL1 and AAL2 handle isochronous traffic (such as voice and video), and are relevant to the ATM node only when it is equipped with either a CES (Circuit Emulation Service) ATM interface card, or when it has voice over AAL2 capabilities. AAL3/4 and AAL5 support data communications; that is, they segment and reassemble data packets.

The two types of devices in an ATM network are switches and routers. Typically, ATM switches do packet switching at Layer 2, while ATM routers do packet switching using Layer 3 addresses, such as IPv4 network addresses, IPv6 network addresses, and MPLS labels.

ATM is supported on the following line cards:

- Cisco 1-port Clear Channel OC-3 SPA
- Cisco 3-port Clear Channel OC-3 SPA
- Cisco 1-port Clear Channel OC-12 SPA
- Cisco 4-Port OC-3c/STM-1c ATM ISE
- Cisco 4-port OC-12/STM-4 ATM ISE
- Cisco 2-Port Channelized T3/E3 ATM and Circuit Emulation (CEoPs) Shared Port Adapter (SPA-2CHT3-CE-ATM)

Cisco IOS XR software ATM interfaces can operate in the following modes:

- Point-to-point
- Layer 2 port mode

**Note** A single ATM interface can simultaneously support point-to-point and L2VPN subinterfaces.

In Cisco IOS XR software, ATM interface configuration is hierarchical and comprises the following elements:

1. The ATM main interface, which is the physical interface. ATM main interfaces can be configured with point-to-point subinterfaces, vp-tunnels, ILMI interfaces, or as Layer 2 port mode attachment circuits (ACs) or Layer 2 subinterface ACs.
2. ATM subinterfaces, which are configured under the ATM main interface. An ATM subinterface does not actively carry traffic until you configure a PVC or PVP under the ATM subinterface.

3. PVCs, which are configured under an ATM subinterface. A single PVC is allowed per subinterface. PVCs are supported under point-to-point and Layer 2 subinterfaces.
4. Permanent virtual paths (PVPs), which are configured under a Layer 2 ATM subinterface. A single PVP is allowed per subinterface.

# VC-Class Mapping

A virtual circuit (VC) class enables the configuration of VC parameters that are then mapped to a main interface, subinterface, or PVC. Without vc-classes, you must perform considerable manual configuration on each ATM main interface, subinterface, and PVC and on the router. This configuration can be time consuming and error prone. After you have created vc-class, you can apply that vc-class to as many ATM interfaces, subinterfaces, or PVCs as you want.

Vc-classes include the following types of configuration data:

- ATM encapsulation for the VC
- OAM management
- traffic shaping

The order of configuration precedence is hierarchical, as demonstrated in the following list, where configuration on the PVC takes the highest precedence, and configuration on a vc-class that is attached to the ATM main interface takes the lowest precedence:

1. Configuration on the PVC
2. Configuration on a vc-class that is attached to the PVC
3. Configuration on the subinterface
4. Configuration on a vc-class that is attached to the subinterface
5. Configuration on the ATM main interface
6. Configuration on a vc-class that is attached to the ATM main interface

For example, if the a PVC has unspecified bit rate (UBR) traffic shaping configured, but it is attached to a class map that is configure with CBR traffic shaping, the PVC maintains the UBR traffic shaping.

**Note** Vc-classes are not applicable to Layer 2 port mode ACs and Layer 2 PVPs. For Layer 2 VPN configurations, Vc-classes are applicable to the PVC only.

# VP-Tunnels

ATM interfaces support vp-tunnels. vp-tunnels are typically used to shape PVCs into a bundle and manage F4 Operation, Administration, and Maintenance (OAM). A vp-tunnel is configured under the main ATM interface, and then subinterfaces and PVCs can be added to the vp-tunnel. vp-tunnels and the PVCs that are configured under them share the same VPI. When a vp-tunnel goes down, all PVCs configured under that vp-tunnel go down, too.

By default, two F4 OAM connections are automatically opened for each vp-tunnel. Use the **f4oam disable** command in ATM vp-tunnel configuration mode to disable the F4 OAM packets for a vp-tunnel.

## F5 OAM on ATM Interfaces

The F5 Operation, Administration, and Maintenance (OAM) feature performs fault-management and performance-management functions on PVCs. If the F5 OAM feature is not enabled on a PVC, then that PVC remains up on the end device in the event of a service disruption where network connectivity is lost. The result is that routing entries that point to the connection remain in the routing table and, therefore, packets are lost. The F5 OAM feature detects such failures and brings the PVC down if there is a disruption along its path.

Use the **oam-pvc manage** command to enable the F5OAM feature on a PVC. After OAM is enabled on a PVC, the PVC can generate F5 loopback cells and you can configure continuity check (CC) management for the PVC. Use the **oam ais-rdi** and **oam retry** commands to configure continuity checking on a PVC.

To drop all current and future OAM cells received on an ATM interface, use the **atm oam flush** command in interface configuration mode.

**Note** The **oam ais-rdi** and **oam retry** commands take effect only after OAM management is enabled on a PVC with the **oam-pvc manage** command.

## ILMI on ATM Interfaces

The ILMI protocol is defined by the ATM Forum for setting and capturing physical layer, ATM layer, virtual path, and virtual circuit parameters on ATM interfaces. When two ATM interfaces run the ILMI protocol, they exchange ILMI packets across the physical connection. These packets consist of SNMP messages as large as 484 octets. ATM interfaces encapsulate these messages in an ATM adaptation layer 5 (AAL5) trailer, segment the packet into cells, and schedule the cells for transmission.

You must enable ILMI on ATM interfaces that communicate with end devices that are configured for ILMI. To enable ILMI, create a PVC with ILMI encapsulation directly under the main ATM interface by using the **pvc** *vpi*/*vci* **ilmi** command in interface configuration mode.

PVCs use ILMI encapsulation to carry ILMI messages. Use the **pvc** *vpi*/*vci* **ilmi** command in interface configuration mode to create an ILMI PVC on an ATM main interface.

**Note** You must use the same VPI and VCI values on both ends of the PVC that connects the end device and the router.

**Note** The ILMI configuration commands are available only after an ILMI PVC is created under the ATM main interface. The ILMI configuration takes effect on the ATM main interface.

**Note** ILMI configuration is not supported on Layer 2 port mode ACs.

## Layer 2 VPN on ATM Interfaces

The Layer 2 VPN (L2VPN) feature enables the connection between different types of Layer 2 attachment circuits and pseudowires, allowing users to implement different types of end-to-end services.

Cisco IOS XR software supports a point-to-point, end-to-end service, where two ATM ACs are connected together.

Switching can take place in two ways:

- AC-to-PW—Traffic reaching the PE is tunneled over a pseudowire (and conversely, traffic arriving over the PW is sent out over the AC). This is the most common scenario.
- Local switching—Traffic arriving on one AC is immediately sent out another AC without passing through a pseudowire.

Keep the following in mind when configuring L2VPN on an ATM interface:

- Cisco IOS XR software supports up to 2000 ACs per line card.
- ATM-over-MPLS supports two types of cell encapsulation:
  - AAL5 CPCS mode—Unsegmented ATM cells are transported across an MPLS backbone.
  - ATM cell (AAL0) mode—Cells are segmented and then reassembled, or packed. AAL0 is supported on ATM main ports, PVCs, and PVPs. The benefits of using AAL0 mode is that groups of ATM cells share a label that maximizes bandwidth efficiencies.

**Note** AAL5 mode is supported on PVCs only.

Use the following commands to display AC and pseudowire information:

- **show interfaces**
- **show l2vpn xconnect**
- **show atm pvp**
- **show atm pvc**

**Note** For detailed information about configuring an L2VPN network, see the *Implementing MPLS Layer 2 VPNs* module of *Cisco IOS XR Multiprotocol Label Switching Configuration Guide.*

## Cell Packing on L2VPN ACs with AAL0 Mode Encapsulation

Cell packing is supported on L2VPN ATM interfaces that are configured with AAL0 mode encapsulation. Cell packing relates to the delay variations that are defined in the ATM standards. Users can specify the number of cells that can be processed by the pseudowire, and configure the maximum cell packing timeout (MCPT) timers to use in conjunction with cell packing.

The **cell-packing** command allows the user to perform the following tasks:

- Configure the maximum number of cells that can be transmitted in a single packet
- Attach one of the three MCPT timers to an individual Layer 2 port mode AC, PVC, or PVP.

The three MCPT timers are defined under the main ATM interface with the **atm mcpt-timer** command, which lets the user specify the maximum number of microseconds to wait to complete cell packing on a single packet before that packet is transmitted. If the associated MCPT timer expires before the maximum number of cells that can be packed is reached, then the packet is transmitted with the number of cells that have been packed thus far.

We recommend configuring a low, medium, and high value for the three MCPT timers to accommodate the different ATM traffic classes. Low- latency constant bit rate (CBR) traffic typically uses a low MCPT timer value, while high-latency Unspecified bit rate (UBR) traffic typically requires a high MCPT timer value. Variable bit rate real-time (VBR-rt) and variable bit rate non-real-time (VBR-nrt) traffic typically use a median MCPT timer value.

# Circuit Emulation over Packet on the Cisco 2-Port Channelized T3/E3 ATM and Circuit Emulation Shared Port Adapter

The Circuit Emulation over Packet (CEoP) SPA is a low-speed ATM SPA that enables service providers and enterprises to migrate to a one-packet network capable of efficiently delivering both data and circuit services.

CEoP is a virtual imitation of a physical connection. CEoP does ATM conversion and transparent transfer of ATM data to the router engine. CEoP first performs cell delineation, then segmentation and reassembly (SAR), and then sends the cells to the router engine.

CEoP benefits service providers who operate both packet-switched networks and time-division multiplexed (TDM) networks, who wish to move their data services from a TDM network to a packet network for scalability and efficiency.

Cisco provides routing and switching solutions capable of transporting Layer 2 and Layer 3 protocols such as Ethernet, IP, and Frame Relay. Although most applications and services have been migrated to the packet-based network, some, including voice and legacy applications, still rely on a circuit or leased line for transport. The CEoP SPAs implement Circuit-Emulation-over-Packet by transporting circuits over a packet-based network. CEoP SPAs help service providers migrate to one packet network capable of efficiently delivering both data and circuit services.

CEoP is supported on the following card on the Cisco XR 12000 Series Router:

- Cisco 2-Port Channelized T3/E3 ATM and Circuit Emulation (CEoPs) Shared Port Adapter (SPA-2CHT3-CE-ATM)

No new or specific configurations or CLIs are necessary to enable CEoP on the card.

## Features Supported in CEoP

CEoP supports these features:

- T3 channelized to T1
- Clear channel T3
- ATM encapsulation only
- Packet forwarding
- Scalability - 2K L3 interfaces per CEoP SPA
- Scalability - 2K L2 connections per CEoP SPA
- On-Line Insertion and Removal (OIR)
- Quack authentication
- Environment Monitoring
- FPD

- Quality of Service (QoS)
- Inverse Multiplexing (IMA)

# ATM Layer 2 QoS

QoS is configured on ATM interfaces primarily in the same way that it is configured on other interfaces. No new CLIs are added in this release.

For complete information on configuring QoS and QoS commands, refer to these documents:

- *Cisco IOS XR Modular Quality of Service Configuration Guide for the Cisco XR 12000 Series Router*
- *Cisco IOS XR Modular Quality of Service Command Reference for the Cisco XR 12000 Series Router*

**Note** Layer 2 QoS is supported on all Engine 5 ATM SPAs, including all CEoP SPAs.

This section describes the features and restrictions that apply to ATM Layer 2 QoS.

## Features

These QoS features are supported:

- Layer 2 Ingress QoS – policing, marking, and queueing are supported
- Layer 2 Egress Main Interface QoS – shaping, policing, and queueing are supported. Marking is not supported. This feature works on both Layer 2 and Layer 3 PVCs independent of any subinterface QoS policies.
- The Modular QoS CLI (MQC) actions are supported for ATM traffic in the ingress direction only.
  - match atm clp
  - match atm oam
  - set atm clp
  - set mpls exp imp
  - set prec tunnel (L2TPv3 only)
  - set dscp tunnel (L2TPv3 only)
- Traffic is classified based on Cell Loss Priority–CLP1, CLP0, or OAM.
- OAM traffic can be excluded from policing by using the match-oam classification in a hierarchical policy map
- The following set actions are supported:
  - set mpls exp imp
  - set prec tunnel
  - set dscp tunnel
  - set qos-group
  - set disc-class
  - set atm-clp (exceed action only)

- Policy map counters are supported.

## Matching

The following match criteria is supported on Layer 2 ATM interfaces in the ingress direction only:

- match atm clp0
- match atm clp1
- match atm oam

The following match criteria is supported on Layer 2 ATM interfaces in the egress direction only:

- match mpls exp topmost (egress only)
- match qos-group (egress only)

**Note** The **match-all** command does not support the above match criteria.

## Marking

The following marking actions are supported on Layer 2 ATM interfaces:

- set mpls exp imposition (AToM only)
- set qos-group (AToM and local switching)
- set discard-class (AToM and local switching)
- set mpls exp imposition and set atm-clp (AToM only)
- set tunnel-dscp (L2TPv3 only)
- set tunnel-prec (L2TPv3 only)

**Note** Packets can be matched and remarked for CLP0, CLP1, and OAM.

## Policing

Policing is supported on Layer 2 ATM interfaces in the ingress direction only.

Policing is performed during segmentation and reassembly (SAR) for the following ATM traffic classes:

- CBR.1
- VBR.1
- VBR.2
- VBR.3
- UBR.1
- UBR.2

Policing is supported for VC and VP modes, but not for Port mode L2 ATM interfaces.

OAM cells are policed along with the user cells unless the QOS policy is explicitly configured to exclude OAM cells from being policed. This can be achieved using different match criteria in the policy map with class-default matching all the traffic including OAM cells.

Policing is supported for ATM AAL5SNAP, AAL5MUX and AAL5NLPID encapsulated packets.

Policing is done on AAL0 packets with the same conditions as AAL5 packets as follows:

- AAL5 packet is conforming if all the cells in the packet conform to PCR and SCR buckets.
- AAL5 packet is exceeding if at least one cell does not conform to the SCR bucket.
- AAL5 packet is violating if at least one cell does not conform to the PCR bucket.

**Note** The Martini Control Word C bit is set for all exceeding AAL5 packets. All violating AAL5 packets are dropped.

The following policing options are supported for ATM TM4.0 GCRA policing:

- Rate in cellsps and percent
- Peak rate in cellsps and percent
- Delay tolerance in us
- Maximum burst size in cells

The following conform and exceed actions are supported for Layer 2 ATM interfaces in the ingress direction:

- transmit
- drop
- set mpls exp imposition (AToM only)
- set qos-group (AToM and Local switching)
- set discard-class (AToM and Local switching)
- set atm-clp (exceed action only, AToM and Local switching)
- set tunnel-prec (L2TPv3 only)
- set tunnel-dscp (L2TPv3 only)

**Note** The only violate action that is supported is the drop action.

The following combination of multiple policing actions is supported:

- set mpls exp imposition and set atm-clp (exceed action only, AToM only)

## Hierarchical Policy Maps

For VBR.2 and VBR.3 traffic classes, 2-level hierarchical policy maps are supported in the ingress direction only. Attempts to attach hierarchical policy maps in the egress direction are denied.

The parent policy contains the policing configuration for the PCR bucket and matches on all traffic. The parent policy may exclude OAM traffic.

The child policy contains the policing configuration for the SCR bucket and typically matches on CLP0 cells.

Marking actions are supported only in child policy maps. All other policing actions are allowed in parent policy maps.

Only two policing buckets per Layer 2 circuit are allowed; one in the parent policy that defines the peak rate, and one in the child policy that defines the SCR.

Typically CLP0 cells are sent to the SCR bucket, but it is possible to send both CLP0 and CLP1 cells to the SCR bucket, using the classification criteria in the child policy.

**Note** For ATM Layer 2 QoS, in policy maps, the **set atm-clp** command is supported only as a police exceed action. It is not supported as a standalone set action.

# Configuring ATM Interfaces

The ATM interface configuration tasks are described in the following procedures:

- Bringing Up an ATM Interface, page 22
- Configuring Optional ATM Interface Parameters, page 24
- How to Create and Configure a Point-to-Point ATM Subinterface with a PVC, page 26
  - Creating a Point-to-Point ATM Subinterface with a PVC, page 26
  - Configuring Optional Point-to-Point ATM PVC Parameters, page 28
- How to Create and Configure a VP-Tunnel, page 31
  - Creating and Configuring a VP-Tunnel on an ATM Interface, page 32
  - Creating and Configuring Subinterfaces with PVCs on a VP-tunnel, page 35
- How to Configure a Layer 2 Attachment Circuit, page 38
  - Creating a Layer 2 Port Mode AC, page 38
  - Configuring Optional Parameters on a Layer 2 Port Mode AC, page 40
  - Creating an ATM Layer 2 Subinterface with a PVC, page 42
  - Configuring Optional ATM Layer 2 PVC Parameters, page 43
  - Creating an ATM Layer 2 Subinterface with a PVP, page 46
  - Configuring Optional ATM Layer 2 PVP Parameters, page 47
- How to Create and Configure a VC-Class, page 50
  - Creating and Configuring a VC-Class, page 50
  - Attaching a VC-Class to a Point-to-Point ATM Main Interface, page 53
  - Attaching a VC-Class to a Point-to-Point ATM Subinterface, page 54
  - Attaching a VC-Class to a PVC on an ATM Subinterface, page 55
- How to Configure ILMI on ATM Interfaces, page 57
  - Enabling ILMI on an ATM Interface, page 57
  - Disabling ILMI on an ATM Interface, page 59
- How to Configure Channelized ATM, page 61
- How to Configure Clear Channel ATM with Virtual Path (VP) Tunnels, page 64
- Attaching a Service-Policy to an Attachment Circuit, page 67

# Bringing Up an ATM Interface

This task describes the commands used to bring up an ATM interface.

## Prerequisites

You must have one of the following line cards installed in a Cisco XR 12000 Series Router that is running Cisco IOS XR software:

- 4-Port OC12
- 4-Port OC3

## Restrictions

The configuration on both ends of the ATM connection must match for the interface to be active.

### SUMMARY STEPS

1. **configure**
2. **interface atm** *interface-path-id*
3. **no shutdown**
4. **end**
   or
   **commit**
5. **exit**
6. **exit**
7. Repeat Step 1 through Step 6 to bring up the interface at the other end of the connection.
8. **show interfaces atm** *interface-path-id* **brief**

### DETAILED STEPSs

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | `configure`<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **`interface atm`** *`interface-path-id`*<br><br>**Example:**<br>`RP/0/0/CPU0:router (config)# interface atm 0/6/0/1` | Enters ATM interface configuration mode. |
| **Step 3** | `no shutdown`<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-if)# no shutdown` | Removes the shutdown configuration.<br>**Note** Removal of the shutdown configuration eliminates the forced administrative down on the interface, enabling it to move to an up or down state. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-if)# end`<br>or<br>`RP/0/0/CPU0:router(config-if)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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** | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-if)# exit` | Exits interface configuration mode and enters global configuration mode. |
| **Step 6** | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config)# exit` | Exits global configuration mode and enters EXEC mode. |
| **Step 7** | Repeat Step 1 through Step 6 to bring up the interface at the other end of the connection. | Brings up the connection.<br>**Note** The configuration on both ends of the ATM connection must match. |
| **Step 8** | **show interfaces atm** *interface-path-id* **brief**<br><br>**Example:**<br>`RP/0/0/CPU0:router# show interfaces atm 0/6/0/1 brief` | (Optional) Verifies that the interface is active and properly configured.<br>If you have brought up an ATM interface properly, the "Intf State" field for that interface in the **show interfaces atm** command output shows "up." |

## What to Do Next

- To modify the default configuration of the ATM interface you just brought up, see the "Configuring Optional ATM Interface Parameters" section on page 24.
- To configure a point-to-point subinterface on the ATM interface you just brought up, see the "How to Create and Configure a Point-to-Point ATM Subinterface with a PVC" section on page 26.
- To create a vp-tunnel on the ATM interface you just brought up, see the "How to Create and Configure a VP-Tunnel" section on page 31.

- To use the interface as a Layer 2 post mode AC, see the "How to Configure a Layer 2 Attachment Circuit" section on page 38.
- To attach a Vc-class to the ATM interface you just brought up, see the "How to Create and Configure a VC-Class" section on page 50.
- To enable ILMI on the ATM interface you just brought up, see the "How to Configure ILMI on ATM Interfaces" section on page 57.

# Configuring Optional ATM Interface Parameters

This task describes the commands you can use to modify the default configuration on an ATM interface.

## Prerequisites

Before you modify the default ATM interface configuration, we recommend that you bring up the ATM interface and remove the shutdown configuration, as described in the "Bringing Up an ATM Interface" section on page 22.

## Restrictions

The configuration on both ends of the ATM connection must match for the interface to be active.

### SUMMARY STEPS

1. **configure**
2. **interface atm** *interface-path-id*
3. **atm maxvpi-bits 12**
4. **atm oam flush**
5. **atm mcpt-timers** *timer-1 timer-2 timer-3*
6. **end**
   or
   **commit**
7. **exit**
8. **exit**
9. **show atm interface atm** [*interface-path-id*]
10. **show interfaces atm** *interface-path-id* **brief**

| | Command or Action | Purpose |
|---|---|---|
| Step 1 | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| Step 2 | **interface atm** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router (config)# interface atm 0/6/0/1` | Enters ATM interface configuration mode. |
| Step 3 | **atm maxvpi-bits 12**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-if)# atm maxvpi-bits 12` | (Optional) Enables support for the 12-bit VPI NNI cell format. |
| Step 4 | **atm oam flush**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-if)# atm oam flush` | (Optional) Drops all current and future OAM cells received on an ATM interface. |
| Step 5 | **atm mcpt-timers** *timer-1 timer-2 timer-3*<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-if)#` **`atm mcpt-timers 50 100 200`** | (Optional) Specifies the maximum cell packing timeout values for each of the three per-interface MCPT timers, in microseconds.<br><br>**Note** The default value for each timer is 50 microseconds.<br><br>**Note** The **atm mcpt-timers** command is applicable to Layer 2 ATM ACs only. |
| Step 6 | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-if)# end`<br>or<br>`RP/0/0/CPU0:router(config-if)# commit` | Saves configuration changes.<br><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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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><br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 7** | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-if)# exit` | Exits interface configuration mode and enters global configuration mode. |
| **Step 8** | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config)# exit` | Exits global configuration mode and enters EXEC mode. |
| **Step 9** | **show atm interface atm** [*interface-path-id*]<br><br>**Example:**<br>`RP/0/0/CPU0:router# show atm interface atm 0/6/0/1` | (Optional) Displays ATM-specific data for the specified ATM interface. |
| **Step 10** | **show interfaces atm** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router# show interfaces atm 0/6/0/1` | (Optional) Displays general information for the specified ATM interface. |

## What to Do Next

- To configure a point-to-point subinterface on the ATM interface you just brought up, see the "How to Create and Configure a Point-to-Point ATM Subinterface with a PVC" section on page 26.
- To create a vp-tunnel on the ATM interface you just brought up, see the "How to Create and Configure a VP-Tunnel" section on page 31.
- To use the interface as a Layer 2 ATM AC, see the "How to Configure a Layer 2 Attachment Circuit" section on page 38.
- To attach a Vc-class to the ATM interface you just brought up, see the "How to Create and Configure a VC-Class" section on page 50.
- To enable ILMI on the ATM interface you just brought up, see the "How to Configure ILMI on ATM Interfaces" section on page 57.

# How to Create and Configure a Point-to-Point ATM Subinterface with a PVC

The configuration tasks for creating and configuring a point-to-point ATM subinterface with a PVC are described in the following procedures:

- Creating a Point-to-Point ATM Subinterface with a PVC, page 26
- Configuring Optional Point-to-Point ATM PVC Parameters, page 28

# Creating a Point-to-Point ATM Subinterface with a PVC

The procedure in this section creates a point-to-point ATM subinterface and configures a permanent virtual circuit (PVC) on that ATM subinterface.

## Prerequisites

Before you can create an ATM subinterface on an ATM interface, you must bring up an ATM interface, as described in the "Bringing Up an ATM Interface" section on page 22.

## Restrictions

Only one PVC can be configured for each point-to-point ATM subinterface.

### SUMMARY STEPS

1. **configure**
2. **interface atm** *interface-path-id.subinterface* **point-to-point**
3. **ipv4 address** *ipv4_address/prefix*
4. **pvc** *vpi/vci*
5. **end**
   or
   **commit**
6. Repeat Step 1 through Step 5 to bring up the ATM subinterface and any associated PVC at the other end of the connection.

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| Step 1 | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| Step 2 | **interface atm** *interface-path-id.subinterface* **point-to-point**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config)# interface atm 0/6/0/1.10` | Enters ATM subinterface configuration mode. |
| Step 3 | **ipv4 address** *ipv4_address/prefix*<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-subif)#`**`ipv4 address 10.46.8.6/24`** | Assigns an IP address and subnet mask to the subinterface. |
| Step 4 | **pvc** *vpi/vci*<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-subif)# pvc 5/10` | (Optional) Creates an ATM permanent virtual circuit (PVC) and enters ATM PVC configuration submode.<br>**Note** Only one PVC is allowed per subinterface. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-subif)# end`<br>or<br>`RP/0/0/CPU0:router(config-subif)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 6** | Repeat Step 1 through Step 5 to bring up the ATM subinterface and any associated PVC at the other end of the connection. | Brings up the ATM connection.<br>**Note** The configuration on both ends of the subinterface connection must match. |

## What to Do Next

- To configure optional PVC parameters, see the "Configuring Optional Point-to-Point ATM PVC Parameters" section on page 28.
- To attach Layer 3 service policies, such as Multiprotocol Label Switching (MPLS) or quality of service (QoS), to the PVC under the PVC submode, refer to the appropriate Cisco IOS XR software configuration guide.
- To configure a vc-class and apply it to an ATM subinterface or PVC, see the "Creating and Configuring a VC-Class" section on page 50.

# Configuring Optional Point-to-Point ATM PVC Parameters

This task describes the commands you can use to modify the default configuration on an ATM PVC.

## Prerequisites

Before you can modify the default PVC configuration, you must create the PVC on an ATM subinterface, as described in the "Creating a Point-to-Point ATM Subinterface with a PVC" section on page 26.

## Restrictions

The configuration on both ends of the PVC must match for the connection to be active.

## SUMMARY STEPS

1. **configure**
2. **interface atm** *interface-path-id.subinterface* **point-to-point**
3. **pvc** *vpi*/*vci*
4. **encapsulation** {**aal5mux ipv4** | **aal5nlpid** | **aal5snap**}
5. **oam-pvc manage** [*frequency*] [**disable**] [**keep-vc-up** [**seg-aisrdi-failure**]]
6. **oam ais-rdi** [*down-count* [*up-count*]]
7. **oam retry**
8. **shape** [**cbr** *peak_output_rate* | **ubr** *peak_output_rate* | **vbr-nrt** *peak_output_rate sustained_output_rate burst_size*| **vbr-rt** *peak_output_rate sustained_output_rate burst_size*]
9. **service-policy** [**input** | **output**] *policy_name*
10. **end**
    or
    **commit**
11. Repeat Step 1 through Step 10 to configure the PVC at the other end of the connection.

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| Step 1 | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| Step 2 | **interface atm** *interface-path-id.subinterface* **point-to-point**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config)# interface atm 0/6/0/1.10 point-to-point` | Enters ATM subinterface configuration mode. |
| Step 3 | **pvc** *vpi*/*vci*<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-subif)# pvc 5/10` | Enters subinterface configuration mode for the PVC. |
| Step 4 | **encapsulation** {**aal5mux ipv4** \| **aal5nlpid** \| **aal5snap**}<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-atm-vc)# encapsulation aal5snap` | Configures the ATM adaptation layer (AAL) and encapsulation type for a PVC.<br><br>**Note** The default encapsulation type for a vc-class is AAL5/SNAP |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **oam-pvc manage** [*frequency*] [**disable**] [**keep-vc-up** [**seg-aisrdi-failure**]<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-atm-vc)# oam-pvc manage 200 keep-vc-up` | Enable ATM OAM F5 loopback cell generation and configures continuity check (CC) management for the ATM permanent virtual circuit (PVC).<br>• Include the **disable** keyword to disable OAM management on the specified PVC.<br>• Include the **keep-vc-up** keyword specify that PVC remains in the UP state when CC cells detect connectivity failure.<br>• Include the **seg-aisrdi-failure** keyword to specify that, if segment AIS/RDI cells are received, the VC will not be brought down because of end CC failure or loopback failure. |
| **Step 6** | **oam ais-rdi** [*down-count* [*up-count*]]<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-atm-vc)# oam ais-rdi 25 5` | Configures the PVC so that it is brought down after a specified number of OAM alarm indication signal/remote defect indication (AIS/RDI) cells are received on the associated PVC. |
| **Step 7** | **oam retry** [*up-count* [*down-count* [*retry-frequency*]]]<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-atm-vc)# oam retry 5 10 5` | Configures parameters related to OAM management for the PVC.<br>If no OAM AIS/RDI cells are received within the specified interval, the PVC is brought up. |
| **Step 8** | **shape** [**cbr** *peak_output_rate* \| **ubr** *peak_output_rate*\| **vbr-nrt** *peak_output_rate sustained_output_rate burst_size*\| **vbr-rt** *peak_output_rate sustained_output_rate burst_size*]<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-atm-vc)# shape vbr-nrt 100000 100000 8000` | Configures ATM traffic shaping for the PVC.<br>You must estimate how much bandwidth is required before you configure ATM traffic shaping.<br>• *peak_output_rate*—Configures the maximum cell rate that is always available for the traffic.<br>• *Sustained_output_rate*—Sustained output rate for the bit rate.<br>• *burst size*—Burst cell size for the bit rate. Range is from 1 through 8192. |
| **Step 9** | **service-policy** [**input** \| **output**] *policy_name*<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-atm-vc)# service-policy input policyA` | Attaches a QoS policy to an input or output PVC. Replace *policy_name* with the name of the service policy you want to attach to the PVC.<br>**Note** For information on creating and configuring service policies, see the *Cisco IOS XR Modular Quality of Service Configuration Guide.* |

| | Command or Action | Purpose |
|---|---|---|
| **Step 10** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-subif)# end`<br>or<br>`RP/0/0/CPU0:router(config-subif)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 11** | Repeat Step 1 through Step 10 to configure the PVC at the other end of the connection. | Brings up the connection.<br>**Note** The configuration on both ends of the connection must match. |

## What to Do Next

- To attach Layer 3 service policies, such as MPLS or QoS, to the PVC under the PVC submode, refer to the appropriate Cisco IOS XR software configuration guide.
- To configure a vc-class and apply it to an ATM subinterface or PVC, see the "Creating and Configuring a VC-Class" section on page 50.

# How to Create and Configure a VP-Tunnel

The configuration tasks for creating and configuring an ATM vp-tunnel are described in the following procedures:

- Creating and Configuring a VP-Tunnel on an ATM Interface, page 32
- Creating and Configuring Subinterfaces with PVCs on a VP-tunnel, page 35

**Note** VP-tunnels are specific to point-to-point ATM interfaces and cannot be configured on ATM ACs.

# Creating and Configuring a VP-Tunnel on an ATM Interface

The procedures in this section create a vp-tunnel on a point-to-point ATM main interface. The creation and configuration of a vp-tunnel is a four-step process:

1. Bring up an ATM interface, as described in the "Bringing Up an ATM Interface" section on page 22.
2. Create and configure a vp-tunnel on the ATM interface, as describe in the "Creating and Configuring a VP-Tunnel on an ATM Interface" section on page 32.
3. Create subinterfaces with PVCs on the vp-tunnel, as described in the "Creating and Configuring Subinterfaces with PVCs on a VP-tunnel" section on page 35.
4. Ping the other side of the connection through the vp-tunnel to verify the vp-tunnel configuration, as described in "Creating and Configuring Subinterfaces with PVCs on a VP-tunnel" section on page 35.

The procedure in this section creates a vp-tunnel on an ATM main interface.

## Prerequisites

Before you can create a vp-tunnel on an ATM main interface, you must bring up an ATM interface, as described in the "Bringing Up an ATM Interface" section on page 22.

## Restrictions

- A vp-tunnel is not truly active until a PVC is created with the same VPI value as the vp-tunnel, as described in the "Creating and Configuring Subinterfaces with PVCs on a VP-tunnel" section on page 35.
- When a vp-tunnel goes down, all VCs that are configured under that vp-tunnel go down.
- The following cards do not support vp-tunnels with a VPI of 0:
  - 4-Port OC-3c/STM-1c ATM ISE Line Card, multimode
  - 4-Port OC-3c/STM-1c ATM ISE Line Card, single-mode
  - 4-port OC-12/STM-4 ATM multimode ISE line card with SC connector
  - Series 4-port OC-12/STM-4 ATM single-mode, intermediate-reach ISE line card with SC Connector

### SUMMARY STEPS

1. **configure**
2. **interface atm** *interface-path-id*
3. **vp-tunnel** *vpi*
4. **f4oam disable**
5. **shape** [**cbr** *peak_output_rate* | **vbr-nrt** *peak_output_rate sustained_output_rate burst_size* | **vbr-rt** *peak_output_rate sustained_output_rate burst_size*]
6. **end**
   or
   **commit**
7. **exit**

8. **exit**
9. Repeat Step 1 through Step 8 to bring up the vp-tunnel at the other end of the connection.
10. **show atm vp-tunnel interface atm** [*interface-path-id*]

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| Step 1 | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| Step 2 | **interface atm** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router (config)# interface atm 0/6/0/1` | Enters ATM interface configuration mode. |
| Step 3 | **vp-tunnel** *vpi*<br><br>**Example:**<br>`RP/0/0/CPU0:router (config)# vp-tunnel 10` | Configures a vp-tunnel on an ATM interface. |
| Step 4 | **f4oam disable**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-atm-vp-tunnel)# f4oam disable` | (Optional) Disables the passing of OAM packets. |
| Step 5 | **shape** [**cbr** *peak_output_rate* \| **vbr-nrt** *peak_output_rate sustained_output_rate burst_size* \| **vbr-rt** *peak_output_rate sustained_output_rate burst_size*]<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-if)# shape` | Configures ATM traffic shaping for the PVC.<br><br>You must estimate how much bandwidth is required before you configure ATM traffic shaping.<br><br>• *peak_output_rate*—Configures the maximum cell rate that is always available for the traffic.<br>• *Sustained_output_rate*—Sustained output rate for the bit rate.<br>• *burst size*—Burst cell size for the bit rate. Range is from 1 through 8192.<br><br>**Note** After you configure traffic shaping on the vp-tunnel, you cannot configure traffic shaping directly on the PVCs configured under that vp-tunnel. If you attempt to use the **shape** command on a PVC that is configured under a tunnel, the command is rejected. |

| | Command or Action | Purpose |
|---|---|---|
| Step 6 | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-if)# end`<br>or<br>`RP/0/0/CPU0:router(config-if)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 7 | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-if)# exit` | Exits interface configuration mode and enters global configuration mode. |
| Step 8 | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config)# exit` | Exits global configuration mode and enters EXEC mode. |
| Step 9 | Repeat Step 1 through Step 8 to bring up the vp-tunnel at the other end of the connection. | Brings up the vp-tunnel. |
| Step 10 | **show atm vp-tunnel interface atm** [*interface-path-id*]<br><br>**Example:**<br>`RP/0/0/CPU0:router (config)# show atm vp-tunnel interface atm 0/6/0/1` | Displays vp-tunnel information for the entire router or a specific ATM interface. |

## What to Do Next

To attach Layer 3 service policies, such as MPLS or QoS, to the vp-tunnel or its PVCs, refer to the appropriate Cisco IOS XR software configuration guide.

# Creating and Configuring Subinterfaces with PVCs on a VP-tunnel

The procedure in this section creates and configures a subinterface with a PVC on a vp-tunnel.

**Note** A vp-tunnel is not truly active until a PVC is created with the same VPI value as the vp-tunnel.

## Prerequisites

Before you can create a subinterface with a PVC on an ATM vp-tunnel, you must create a vp-tunnel on the ATM main interface, as described in the "Creating and Configuring a VP-Tunnel on an ATM Interface" section on page 32.

## Restrictions

- A PVC and its host vp-tunnel must share the same VPI for the connection to be active.
- The following cards do not support vp-tunnels with a VPI of 0:
  - 4-Port OC-3c/STM-1c ATM ISE Line Card, multimode
  - 4-Port OC-3c/STM-1c ATM ISE Line Card, single-mode
  - 4-port OC-12/STM-4 ATM multimode ISE line card with SC connector
  - Series 4-port OC-12/STM-4 ATM single-mode, intermediate-reach ISE line card with SC Connector

### SUMMARY STEPS

1. **configure**
2. **interface atm** *interface-path-id.subinterface* **point-to-point**
3. **ipv4 address** *ipv4_address/prefix*
4. **pvc** *vpi/vci*
5. **end**
   or
   **commit**
6. Repeat Step 1 through Step 5 to bring up the ATM subinterface and PVC at the other end of the connection.
7. **ping atm interface atm** *interface-path-id.subinterface vpi/vci*
8. **show atm vp-tunnel** [**interface atm** *interface-path-id*]

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **interface atm** *interface-path-id.subinterface* **point-to-point**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config)# interface atm 0/6/0/1.10 point-to-point` | Creates a new subinterface and enters ATM subinterface configuration mode for that subinterface. |
| **Step 3** | **ipv4 address** *ipv4_address/prefix*<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-subif)#ipv4 address 10.46.8.6/24` | Assigns an IP address and subnet mask to the subinterface. |
| **Step 4** | **pvc** *vpi/vci*<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-subif)# pvc 5/10` | Creates an ATM permanent virtual circuit (PVC) on the subinterface and attaches it to the vp-tunnel you created in the "Creating and Configuring a VP-Tunnel on an ATM Interface" section on page 32.<br><br>Replace *vpi* with the VPI of the vp-tunnel on which you are creating the PVC.<br><br>**Note** The PVC VPI and vp-tunnel VCI must match or the connection will not be active.<br><br>**Note** A vp-tunnel is not usable until you create PVCs under it. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **`end`**<br>or<br>**`commit`**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-subif)# end`<br>or<br>`RP/0/0/CPU0:router(config-subif)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 6** | Repeat Step 1 through Step5 to bring up the subinterface and PVC at the other end of the vp-tunnel. | Brings up the subinterface and PVC. |
| **Step 7** | **`ping atm interface atm`** *`interface-path-id.subinterface vpi/vci`*<br><br>**Example:**<br>`RP/0/0/CPU0:router # ping atm interface atm 0/2/0/0.10 10/100` | Verifies connectivity between two ATM connection endpoints through the vp-tunnel you configured in Step 1 through Step 6.<br>• Replace *interface-path-id.subinterface* with the ATM subinterface that is configured on the vp-tunnel whose connectivity you want to verify. This is the same *interface-path-id.subinterface* you configured in Step 2.<br>• Replace *vci* with the VCI of the PVC configured on the vp-tunnel whose connectivity you want to verify. This is the same *vci* you configured in Step 4.<br>• Replace *vpi* with the VPI of the PVC that is configured on the vp-tunnel whose connectivity you want to verify. This is the same *vpi* you configured in Step 4. |
| **Step 8** | **`show atm vp-tunnel`** [**`interface atm`** *`interface-path-id`*]<br><br>**Example:**<br>`RP/0/0/CPU0:router (config)# show atm vp-tunnel interface atm 0/6/0/1` | Displays vp-tunnel information for the entire router or a specific ATM interface. |

## What to Do Next

- To create and configure ATM subinterfaces and PVCs on a vp-tunnel, see the "Creating and Configuring Subinterfaces with PVCs on a VP-tunnel" section on page 35
- To configure a vc-class and apply it to an ATM interface, see the "Creating and Configuring a VC-Class" section on page 50.

# How to Configure a Layer 2 Attachment Circuit

The Layer 2 AC configuration tasks are described in the following procedures:

- Creating a Layer 2 Port Mode AC
- Configuring Optional Parameters on a Layer 2 Port Mode AC
- Creating an ATM Layer 2 Subinterface with a PVC
- Configuring Optional ATM Layer 2 PVC Parameters
- Creating an ATM Layer 2 Subinterface with a PVP
- Configuring Optional ATM Layer 2 PVP Parameters

**Note** After you configure an interface for Layer 2 switching, no routing commands such as **ipv4 address** are permissible. If any routing commands are configured on the interface, then the **l2transport** command is rejected.

# Creating a Layer 2 Port Mode AC

The procedure in this section creates a Layer 2 port mode AC.

## Prerequisites

Before you can create a Layer 2 port mode AC, you must bring up an ATM main interface, as described in the "Bringing Up an ATM Interface" section on page 22.

## Restrictions

ILMI configuration is not supported on Layer 2 port mode ACs.

## Restrictions

Before you can configure an Layer 2 port mode AC, you must ensure that no configuration, such as subinterfaces, already exists on that port. If any preconfiguration does exist, you must remove it.

### SUMMARY STEPS

1. **configure**
2. **interface atm** *interface-path-id*
3. **l2transport**

4. **end**
   or
   **commit**
5. Repeat Step 1 through Step 4 to bring up the ATM AC at the other end of the connection.

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| Step 1 | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| Step 2 | **interface atm** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router (config)# interface atm 0/6/0/1` | Enters interface configuration mode for an ATM interface. |
| Step 3 | **l2transport**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-if)# l2transport` | Enters ATM Layer 2 transport configuration mode, and enables Layer 2 port mode on this ATM interface. |
| Step 4 | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-if-l2)# end`<br>or<br>`RP/0/0/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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 | Repeat Step 1 through Step 4 to bring up the Layer 2 port mode AC at the other end of the connection. | Brings up the Layer 2 port mode AC.<br>**Note** The configuration on both ends of the connection must match. |

## What to Do Next

- To configure a point-to-point pseudowire XConnect on the Layer 2 port mode AC you just created, see the *Implementing MPLS Layer 2 VPNs* module of *Cisco IOS XR Multiprotocol Label Switching Configuration Guide*.
- To configure optional Layer 2 VPN parameters for the ATM AC, see the "Configuring Optional Parameters on a Layer 2 Port Mode AC" section on page 40.

# Configuring Optional Parameters on a Layer 2 Port Mode AC

The procedure in this section configures optional Layer 2 VPN transport parameters on a Layer 2 port mode AC.

## Prerequisites

Before you can configure Layer 2 VPN parameters on a Layer 2 port mode AC, you must create a Layer 2 port mode AC, as described in the "Creating a Layer 2 Port Mode AC" section on page 38.

### SUMMARY STEPS

1. **configure**
2. **interface atm** *interface-path-id*
3. **atm mcpt-timers** *timer-1 timer-2 timer-3*
4. **l2transport**
5. **cell-packing** *cells timer*
6. **end**
   or
   **commit**
7. Repeat Step 1 through Step 6 to configure the Layer 2 port mode AC at the other end of the connection.

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **interface atm** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router (config)# interface atm 0/6/0/1` | Enters interface configuration mode for an ATM interface. |

<table>
<tr><th></th><th>Command or Action</th><th>Purpose</th></tr>
<tr><td>Step 3</td><td><code>atm mcpt-timers timer-1 timer-2 timer-3</code><br><br>Example:<br><code>RP/0/0/CPU0:router (config-if)# atm mcpt-timers 50 100 200</code></td><td>Specifies the maximum cell packing timeout values for each of the three per-interface MCPT timers, in microseconds.<br>Note The default value for each timer is 50 microseconds.</td></tr>
<tr><td>Step 4</td><td><code>l2transport</code><br><br>Example:<br><code>RP/0/0/CPU0:router (config-if)# l2transport</code></td><td>Enters ATM Layer 2 transport configuration mode.</td></tr>
<tr><td>Step 5</td><td><code>cell-packing cells timer</code><br><br>Example:<br><code>RP/0/0/CPU0:router (config-if-l2)# cell-packing 6 1</code></td><td>Sets the maximum number of cells allowed per packet, and specifies a maximum cell packing timeout (MCPT) timer to use for cell packing.<ul><li>Replace cells with the maximum number of cells to use per packet. Range is from 2 through 86.</li><li>Replace timer with the number that indicates the appropriate MCPT timer to use for cell packing. Can be 1, 2, or 3. You can configure up to three different MCPT values for a single main interface.</li></ul></td></tr>
<tr><td>Step 6</td><td><code>end</code><br>or<br><code>commit</code><br><br>Example:<br><code>RP/0/0/CPU0:router (config-if-l2)# end</code><br>or<br><code>RP/0/0/CPU0:router(config-if-l2)# commit</code></td><td>Saves configuration changes.<ul><li>When you issue the end command, the system prompts you to commit changes:<br><code>Uncommitted changes found, commit them before exiting(yes/no/cancel)?<br>[cancel]:</code><ul><li>Entering yes saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.</li><li>Entering no exits the configuration session and returns the router to EXEC mode without committing the configuration changes.</li><li>Entering cancel leaves the router in the current configuration session without exiting or committing the configuration changes.</li></ul></li><li>Use the commit command to save the configuration changes to the running configuration file and remain within the configuration session.</li></ul></td></tr>
<tr><td>Step 7</td><td>Repeat Step 1 through Step 6 to configure the AC at the other end of the connection.</td><td>Brings up the Layer 2 port mode AC.<br>Note The configuration on both ends of the connection must match.</td></tr>
</table>

# Creating an ATM Layer 2 Subinterface with a PVC

The procedure in this section creates a Layer 2 subinterface with a PVC.

## Prerequisites

Before you can create a subinterface on an ATM interface, you must bring up an ATM interface, as described in the "Bringing Up an ATM Interface" section on page 22.

## Restrictions

Only one PVC can be configured for each ATM subinterface.

### SUMMARY STEPS

1. **configure**
2. **interface atm** *interface-path-id.subinterface* **l2transport**
3. **pvc** *vpi/vci*
4. **end**
   or
   **commit**
5. Repeat Step 1 through Step 4 to bring up the ATM subinterface and any associated PVC at the other end of the AC.

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | `configure`<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | `interface atm` *`interface-path-id.subinterface`* `l2transport`<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface atm 0/6/0/1.10 l2transport` | Creates a subinterface and enters ATM subinterface configuration mode for that subinterface. |
| **Step 3** | `pvc` *`vpi/vci`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# pvc 5/20` | Creates an ATM permanent virtual circuit (PVC) and enters ATM Layer 2 transport PVC configuration mode.<br><br>**Note** Only one PVC is allowed per subinterface. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-atm-l2transport-pvc)# end`<br>or<br>`RP/0/0/CPU0:router(config-atm-l2transport-pvc)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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** | Repeat Step 1 through Step 4 to bring up the ATM subinterface and any associated PVC at the other end of the AC. | Brings up the AC.<br>**Note** The configuration on both ends of the AC must match. |

## What to Do Next

- To configure optional PVC parameters, see the "Configuring Optional ATM Layer 2 PVC Parameters" section on page 43.
- To configure a vc-class and apply it to the PVC, see the"Attaching a VC-Class to a PVC on an ATM Subinterface" section on page 55.
- To configure a point-to-point pseudowire XConnect on the AC you just created, see the *Implementing MPLS Layer 2 VPNs* module of *Cisco IOS XR Multiprotocol Label Switching Configuration Guide*.

# Configuring Optional ATM Layer 2 PVC Parameters

This task describes the commands you can use to modify the default configuration on an ATM Layer 2 PVC.

## Prerequisites

Before you can modify the default PVC configuration, you must create the PVC on a Layer 2 ATM subinterface, as described in the "Creating an ATM Layer 2 Subinterface with a PVC" section on page 42.

## Restrictions

The configuration on both ends of the PVC must match for the connection to be active.

### SUMMARY STEPS

1. **configure**
2. **interface atm** *interface-path-id.subinterface* **l2transport**
3. **pvc** *vpi/vci*
4. **encapsulation** {**aal0** | **aal5**}
5. **cell-packing** *cells timer*
6. **shape** [**cbr** *peak_output_rate* | **ubr** *peak_output_rate* | **vbr-nrt** *peak_output_rate sustained_output_rate burst_size*| **vbr-rt** *peak_output_rate sustained_output_rate burst_size*]
7. **end**
   or
   **commit**
8. Repeat Step 1 through Step 7 to configure the PVC at the other end of the AC.

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | `configure`<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | `interface atm` *`interface-path-id.subinterface`* `l2transport`<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# interface atm 0/6/0/1.10 l2transport` | Enters ATM subinterface configuration mode for a Layer 2 ATM subinterface. |
| **Step 3** | `pvc` *`vpi/vci`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-atm-l2transport-pvc)# pvc 5/20` | Enters ATM Layer 2 transport PVC configuration mode for the specified PVC. |
| **Step 4** | `encapsulation {aal0 \| aal5}`<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-atm-l2transport-pvc)# encapsulation aal5` | Configures the ATM adaptation layer (AAL) and encapsulation type for a PVC.<br><br>**Note** The default encapsulation type for a PVC is AAL5. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **cell-packing** *cells timer*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-atm-l2transport-pvc)# cell-packing 5 2` | Sets the maximum number of cells allowed per packet, and specifies a maximum cell packing timeout (MCPT) timer to use for cell packing.<br>• Replace *cells* with the maximum number of cells to use per packet. Range is from 2 through 86.<br>• Replace *timer* with the number that indicates the appropriate MCPT timer to use for cell packing. Can be **1**, **2**, or **3**. You can configure up to three different MCPT values for a single main interface. |
| **Step 6** | **shape** [**cbr** *peak_output_rate* \| **ubr** *peak_output_rate*\| **vbr-nrt** *peak_output_rate sustained_output_rate burst_size*\| **vbr-rt** *peak_output_rate sustained_output_rate burst_size*]<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-atm-l2transport-pvc)# shape vbr-nrt 100000 100000 8000` | Configures ATM traffic shaping for the PVC.<br>You must estimate how much bandwidth is required before you configure ATM traffic shaping.<br>• *peak_output_rate*—Configures the maximum cell rate that is always available for the traffic.<br>• *Sustained_output_rate*—Sustained output rate for the bit rate.<br>• *burst size*—Burst cell size for the bit rate. Range is from 1 through 8192. |
| **Step 7** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-atm-l2transport-pvc)# end`<br>or<br>`RP/0/0/CPU0:router(config-atm-l2transport-pvc)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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** | Repeat Step 1 through Step 7 to configure the PVC at the other end of the AC. | Brings up the AC.<br>**Note** The configuration on both ends of the connection must match. |

## What to Do Next

- To configure a pseudo-wire XConnect on the AC you just created, see the *Implementing MPLS Layer 2 VPNs* module of *Cisco IOS XR Multiprotocol Label Switching Configuration Guide*.
- To configure a vc-class and apply it to the PVC, see the"Attaching a VC-Class to a PVC on an ATM Subinterface" section on page 55.

# Creating an ATM Layer 2 Subinterface with a PVP

The procedure in this section creates an ATM Layer 2 subinterface with a permanent virtual path (PVP) on that ATM subinterface.

## Prerequisites

Before you can create a subinterface with a PVP on an ATM interface, you must bring up an ATM interface, as described in the "Bringing Up an ATM Interface" section on page 22.

## Restrictions

- Only one PVP can be configured for each L2VPN ATM AC.
- F4 OAM emulation is not supported on Layer 2 PVPs.

### SUMMARY STEPS

1. **configure**
2. **interface atm** *interface-path-id.subinterface* **l2transport**
3. **pvp** *vpi*
4. **end**
   or
   **commit**
5. Repeat Step 1 through Step 4 to bring up the ATM subinterface and any associated PVP at the other end of the AC.

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **`configure`**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **`interface atm`** *`interface-path-id.subinterface`* **`l2transport`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface atm 0/6/0/1.10 l2transport` | Creates an ATM subinterface and enters ATM layer2 transport configuration mode for that subinterface. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **pvp** *vpi*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# pvp 100` | (Optional) Creates an ATM PVP and enters ATM PVP configuration submode.<br><br>**Note** Only one PVP is allowed per subinterface. |
| **Step 4** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-atm-l2transport-pvp)# end`<br>or<br>`RP/0/0/CPU0:router(config-atm-l2transport-pvp)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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** | Repeat Step 1 through Step 4 to bring up the ATM subinterface and any associated PVP at the other end of the AC. | Brings up the ATM AC.<br><br>**Note** The configuration on both ends of the AC connection must match. |

## What to Do Next

- To configure optional PVP parameters, see the "Configuring Optional ATM Layer 2 PVP Parameters" section on page 47.
- To configure a point-to-point pseudowire XConnect on the AC you just created, see the *Implementing MPLS Layer 2 VPNs* module of *Cisco IOS XR Multiprotocol Label Switching Configuration Guide*.

# Configuring Optional ATM Layer 2 PVP Parameters

This task describes the commands you can use to modify the default configuration on an ATM Layer 2 PVP.

## Prerequisites

Before you can modify the default PVP configuration, you must create the PVP on an ATM subinterface, as described in the "Creating an ATM Layer 2 Subinterface with a PVP" section on page 46.

## Restrictions

- The following cards do not support vp-tunnels with a VPI of 0:
  - 4-Port OC-3c/STM-1c ATM ISE Line Card, multimode
  - 4-Port OC-3c/STM-1c ATM ISE Line Card, single-mode
  - 4-port OC-12/STM-4 ATM multimode ISE line card with SC connector
  - Series 4-port OC-12/STM-4 ATM single-mode, intermediate-reach ISE line card with SC Connector

### SUMMARY STEPS

1. **configure**
2. **interface atm** *interface-path-id.subinterface* **l2transport**
3. **pvp** *vpi*
4. **cell-packing** *cells timer*
5. **shape** [**cbr** *peak_output_rate* | **ubr** *peak_output_rate* | **vbr-nrt** *peak_output_rate sustained_output_rate burst_size*| **vbr-rt** *peak_output_rate sustained_output_rate burst_size*]
6. **end**
   or
   **commit**
7. Repeat Step 1 through Step 6 to configure the PVP at the other end of the connection.

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| Step 1 | `configure`<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| Step 2 | **interface atm** *interface-path-id.subinterface* **l2transport**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface atm 0/6/0/1.10 l2transport` | Enters ATM subinterface configuration mode. |
| Step 3 | **pvp** *vpi*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# pvp 10` | Enters subinterface configuration mode for the PVP. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **cell-packing** *cells timer*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-atm-l2transport-pvp)# cell-packing 5 2` | Sets the maximum number of cells allowed per packet, and specifies a maximum cell packing timeout (MCPT) timer to use for cell packing.<br>• Replace *cells* with the maximum number of cells to use per packet. Range is from 2 through 86.<br>• Replace *timer* with the number that indicates the appropriate MCPT timer to use for cell packing. Can be **1**, **2**, or **3**. You can configure up to three different MCPT values for a single main interface. |
| **Step 5** | **shape** [**cbr** *peak_output_rate* \| **ubr** *peak_output_rate*\| **vbr-nrt** *peak_output_rate sustained_output_rate burst_size*\| **vbr-rt** *peak_output_rate sustained_output_rate burst_size*]<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-atm-l2transport-pvp)# shape vbr-nrt 100000 100000 8000` | Configures ATM traffic shaping for the PVC.<br>You must estimate how much bandwidth is required before you configure ATM traffic shaping.<br>• *peak_output_rate*—Configures the maximum cell rate that is always available for the traffic.<br>• *Sustained_output_rate*—Sustained output rate for the bit rate.<br>• *burst size*—Burst cell size for the bit rate. Range is from 1 through 8192. |
| **Step 6** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-atm-l2transport-pvp)# end`<br>or<br>`RP/0/0/CPU0:router(config-atm-l2transport-pvp)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 7** | Repeat Step 1 through Step 6 to configure the PVP at the other end of the AC. | Brings up the AC.<br>**Note** The configuration on both ends of the AC connection must match. |

### What to Do Next

- To configure a point-to-point pseudowire XConnect on the AC you just created, see the *Implementing MPLS Layer 2 VPNs* module of *Cisco IOS XR Multiprotocol Label Switching Configuration Guide*.

# How to Create and Configure a VC-Class

The configuration tasks for creating and configuring an ATM VC-Class are described in the following procedures:


- Creating and Configuring a VC-Class, page 50
- Attaching a VC-Class to a Point-to-Point ATM Main Interface, page 53
- Attaching a VC-Class to a Point-to-Point ATM Subinterface, page 54
- Attaching a VC-Class to a PVC on an ATM Subinterface, page 55


# Creating and Configuring a VC-Class

This section describes the tasks and commands required to create a virtual circuit (VC) class and attach it to an ATM main interface, subinterface, or permanent virtual circuit (PVC).

## Restrictions

For Layer 2 VPN AC configurations, VC-classes can be applied to PVCs only. VC-classes are not supported on Layer 2 port mode interfaces or PVPs.

### SUMMARY STEPS

1. **configure**
2. **vc-class atm** *name*
3. **encapsulation** {**aal5mux ipv4** | **aal5nlpid** | **aal5snap**}
4. **oam ais-rdi** [*down-count* [*up-count*]]
5. **oam retry** [*up-count* [*down-count* [*retry-frequency*]]]
6. **oam-pvc manage** *seconds*
7. **shape** [**cbr** *peak_output_rate* | **ubr** *peak_output_rate* | **vbr-nrt** *peak_output_rate sustained_output_rate burst_size*| **vbr-rt** *peak_output_rate sustained_output_rate burst_size*]
8. **end**
   or
   **commit**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| Step 1 | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| Step 2 | **vc-class atm name**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config)# vc-class atm class1` | Creates a vc-class for an ATM interface and enters vc-class configuration mode. |
| Step 3 | **encapsulation** {**aal5mux ipv4** \| **aal5nlpid** \| **aal5snap**}<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-vc-class-atm)# encapsulation aal5snap` | Configures the ATM adaptation layer (AAL) and encapsulation type for an ATM vc-class.<br><br>**Note** The default encapsulation type for a vc-class is AAL5/SNAP<br><br>**Note** In Vc-classes, the **encapsulation** command applies to Layer 3 Point-to-point configurations only. |
| Step 4 | **oam ais-rdi** [*down-count* [*up-count*]]<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-vc-class-atm)# oam ais-rdi 25 5` | Configures the vc-class so that it is brought down after a specified number of OAM alarm indication signal/remote defect indication (AIS/RDI) cells are received on the associated PVC.<br><br>**Note** In vc-classes, the **oam ais-rdi** command applies to Layer 3 Point-to-point configurations only. |
| Step 5 | **oam retry** [*up-count* [*down-count* [*retry-frequency*]]]<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-vc-class-atm)# oam retry 5 10 5` | Configures parameters related to OAM management.<br><br>**Note** In vc-classes, the **oam retry** command applies to Layer 3 Point-to-point configurations only. |
| Step 6 | **oam-pvc manage** *seconds*<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-vc-class-atm)# oam-pvc manage 300` | Configures the ATM OAM F5 loopback frequency.<br><br>**Note** In vc-classes, the **oam-pvc manage** command applies to Layer 3 Point-to-point configurations only. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 7** | **shape** [**cbr** *peak_output_rate* \| **ubr** *peak_output_rate* \| **vbr-nrt** *peak_output_rate sustained_output_rate burst_size*\| **vbr-rt** *peak_output_rate sustained_output_rate burst_size*]<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-vc-class-atm)# shape vbr-nrt 100000 100000 8000` | Configures ATM traffic shaping for the PVC.<br><br>You must estimate how much bandwidth is required before you configure ATM traffic shaping.<br><br>• *peak_output_rate*—Configures the maximum cell rate that is always available for the traffic.<br>• *Sustained_output_rate*—Sustained output rate for the bit rate.<br>• *burst size*—Burst cell size for the bit rate. Range is from 1 through 8192. |
| **Step 8** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-if)# end`<br>or<br>`RP/0/0/CPU0:router(config-if)# commit` | Saves configuration changes.<br><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)?`<br>`[cancel]:`<br>– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

## What to Do Next

Attach the vc-class to an ATM main interface, subinterface, or PVC.

- To attach a vc-class to an ATM main interface, see the "Attaching a VC-Class to a Point-to-Point ATM Main Interface" section on page -53.
- To attach a vc-class to an ATM subinterface, see the "Attaching a VC-Class to a Point-to-Point ATM Subinterface" section on page -54.
- To attach a vc-class to an ATM PVC, see the "Attaching a VC-Class to a PVC on an ATM Subinterface" section on page -55.

# Attaching a VC-Class to a Point-to-Point ATM Main Interface

This section describes the tasks and commands required to attach a vc-class to a point-to-point ATM main interface.

## Restrictions

VC-classes are not applicable to Layer 2 port mode ACs. For Layer 2 VPN configurations, Vc-classes are applicable to the PVC only.

### SUMMARY STEPS

1. **configure**
2. **interface atm** *interface-path-id* **point-to-point**
3. **class-int** *vc-class-name*
4. **end**
   or
   **commit**

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| Step 1 | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| Step 2 | **interface atm** *interface-path-id* **point-to-point**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config)# interface atm 0/6/0/1 point-to-point` | Enters ATM interface configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | `class-int vc-class-name`<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-if)# class-int classA` | Attaches the vc-class to an ATM main interface. Replace the *vc-class-name* argument with the name of the vc-class you configured in the "Creating and Configuring a VC-Class" section on page 50. |
| **Step 4** | `end`<br>or<br>`commit`<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-if)# end`<br>or<br>`RP/0/0/CPU0:router(config-if)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

# Attaching a VC-Class to a Point-to-Point ATM Subinterface

This section describes the tasks and commands required to attach a vc-class to an ATM subinterface.

## SUMMARY STEPS

1. **configure**
2. **interface atm** *interface-path-id.subinterface* **point-to-point**
3. **class-int** *vc-class-name*
4. **end**
   or
   **commit**

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| Step 1 | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| Step 2 | **interface atm** *interface-path-id.subinterface* **point-to-point**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config)# interface atm 0/6/0/1.10 point-to-point` | Enters ATM subinterface configuration mode. |
| Step 3 | **class-int** *vc-class-name*<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-subif)# class-int classA` | Assigns the vc-class to an ATM subinterface. Replace the *vc-class-name* argument with the name of the vc-class you configured in the "Creating and Configuring a VC-Class" section on page -50. |
| Step 4 | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-subif)# end`<br>or<br>`RP/0/0/CPU0:router(config-subif)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

# Attaching a VC-Class to a PVC on an ATM Subinterface

This section describes the tasks and commands required to attach a vc-class to a PVC on an ATM subinterface.

**Note** VC-classes are supported on point-to-point and Layer 2 PVCs.

## SUMMARY STEPS

1. **configure**
2. **interface atm** *interface-path-id*[.*subinterface*] [**point-to-point** | **l2transport**]
3. **pvc** *vpi*/*vci*
4. **class vc** *vc-class-name*
5. **end**
   or
   **commit**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **`configure`**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **`interface atm`** *`interface-path-id.subinterface`* `[`**`point-to-point`** `\|` **`l2transport`**`]`<br><br>**Example:**<br>`RP/0/0/CPU0:router (config)# interface atm 0/6/0/1.10` | Enters subinterface configuration mode and creates the ATM subinterface if it does not already exist.<br><br>Use the **point-to-point** keyword if you are attaching the vc-class to a point-to-point subinterface. Use the **l2transport** keyword if you are attaching the vc-class to a Layer 2 transport subinterface.<br><br>**Note** For more information on creating and configuring ATM subinterfaces, see the "Creating a Point-to-Point ATM Subinterface with a PVC" section on page 26. |
| **Step 3** | **`pvc`** *`vpi/vci`*<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-if)# pvc 5/50` | Enters ATM PVC configuration mode and creates the PVC if it does not already exist.<br><br>**Note** For more information on creating and configuring PVCs on ATM subinterfaces, see the "Creating a Point-to-Point ATM Subinterface with a PVC" section on page 26. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **class-vc** *vc-class-name*<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-atm-vc)# class-vc classA` | Assigns a vc-class to an ATM PVC. Replace the *vc-class-name* argument with the name of the vc-class you want to attach to the PVC. |
| **Step 5** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-if)# end`<br>or<br>`RP/0/0/CPU0:router(config-if)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

# How to Configure ILMI on ATM Interfaces

The configuration tasks for managing ILMI on ATM interfaces are described in the following procedures:

- Enabling ILMI on an ATM Interface, page 57
- Disabling ILMI on an ATM Interface, page 59

# Enabling ILMI on an ATM Interface

This task describes the commands you can use to configure an ATM interface for ILMI.

**Note** For ILMI, a PVC is configured directly on the ATM main interface. Subinterface configuration is not necessary for ATM interfaces that are used for ILMI.

## Prerequisites

Bring up the ATM interface and remove the shutdown configuration, as described in the "Bringing Up an ATM Interface" section on page 22.

## Restrictions

- The configuration on both ends of the ATM ILMI connection must match for the interface to be active.
- ILMI configuration is not supported on Layer 2 port mode ACs.

### SUMMARY STEPS

1. **configure**
2. **interface atm** *interface-path-id*
3. **atm address-registration**
4. **atm ilmi-keepalive** [**act-poll-freq** *frequency*] [**retries** *count*] [**inact-poll-freq** *frequency*]
5. **pvc** *vpi*/*vci* **ilmi**
6. **end**
   or
   **commit**
7. **exit**
8. **exit**
9. **show atm ilmi-status** [**atm** *interface-path-id*]

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| Step 1 | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| Step 2 | **interface atm** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router (config)# interface atm 0/6/0/1` | Enters ATM interface configuration mode. |
| Step 3 | **atm address-registration**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-if)# atm address-registration` | (Optional) Enables the router to engage in address registration and callback functions with the Interim Local Management Interface (ILMI). |
| Step 4 | **atm ilmi-keepalive** [**act-poll-freq** *frequency*] [**retries** *count*] [**inact-poll-freq** *frequency*]<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-if)# atm ilmi-keepalive` | (Optional) Enables ILMI keepalives on an ATM interface. |

| | Command or Action | Purpose |
|---|---|---|
| Step 5 | **pvc** *vpi*/*vci* **ilmi**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-if)# pvc 5/30 ilmi` | Creates an ATM permanent virtual circuit (PVC) with ILMI encapsulation. |
| Step 6 | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-if)# end`<br>or<br>`RP/0/0/CPU0:router(config-if)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 7 | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-if)# exit` | Exits interface configuration mode and enters global configuration mode. |
| Step 8 | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config)# exit` | Exits global configuration mode and enters EXEC mode. |
| Step 9 | **show atm ilmi-status** [**atm** *interface-path-id*]<br><br>**Example:**<br>`RP/0/0/CPU0:router (config)# show atm ilmi-status atm 0/6/0/1` | (Optional) Verifies the ILMI configuration for the specified interface. |

# Disabling ILMI on an ATM Interface

This task describes the commands you can use to disable ILMI on an ATM interface.

## SUMMARY STEPS

1. **configure**
2. **interface atm** *interface-path-id*

3. **atm ilmi-config disable**
4. **end**
   or
   **commit**
5. **exit**
6. **exit**
7. **show atm ilmi-status** [**atm** *interface-path-id*]

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **`configure`**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **`interface atm`** *`interface-path-id`*<br><br>**Example:**<br>`RP/0/0/CPU0:router (config)# interface atm 0/6/0/1` | Enters ATM interface configuration mode. |
| **Step 3** | **`atm ilmi-config disable`**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-if)# atm ilmi-config disable` | (Optional) Disables ILMI on the ATM interface.<br><br>To re-enable ILMI on an ATM interface, use the **no atm ilmi-config disable** form of this command. |
| **Step 4** | **`end`**<br>or<br>**`commit`**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-if)# end`<br>or<br>`RP/0/0/CPU0:router(config-if)# 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)?`<br>`[cancel]:`<br>– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-if)# exit` | Exits interface configuration mode and enters global configuration mode. |
| **Step 6** | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config)# exit` | Exits global configuration mode and enters EXEC mode. |
| **Step 7** | **show atm ilmi-status** [**atm** *interface-path-id*]<br><br>**Example:**<br>`RP/0/0/CPU0:router (config)# show atm ilmi-status atm 0/6/0/1` | (Optional) Verifies the ILMI configuration for the specified interface. |

# How to Configure Channelized ATM

This task describes how to configure a T3 path into multiple T1 channels carrying ATM traffic.

## Prerequisites

The following prerequisites apply channelized ATM in Release 3.7.0:

- You must have a Cisco 2-Port Channelized T3/E3 ATM and Circuit Emulation Shared Port Adapter installed on your router.
- You should know how to configure the SONET controller as specified in the "How to Configure Clear Channel SONET Controllers" section of the *Configuring Clear Channel SONET Controllers on Cisco IOS XR Software* module.

## Restrictions

The following restrictions apply to channelized ATM in Release 3.7.0:

- Channelized ATM is supported only on the Cisco 2-Port Channelized T3/E3 ATM and Circuit Emulation Shared Port Adapter.
- An ATM T3 path can be channelized only into T1 ATM channels or virtual path (VP) tunnels.
- DS0s are not supported.

### SUMMARY STEPS

1. **configure**
2. **hw-module subslot** *subslot-id* **cardtype {t3 | e3}**
3. **controller t3** *interface-path-id*
4. **interface atm** *interface-path-id*
5. **mode** *mode*
6. **controller t1** *interface-path-id*

7. **mode** *mode*
8. **interface atm** *interface-path-id.subinterface* **point-to-point**
9. **pvc** *vpi/vci*
10. **ipv4 address** *ipv4_address/prefix*
11. **end**
    or
    **commit**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| Step 1 | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| Step 2 | **hw-module subslot** *subslot-id* **cardtype** {**t3** \| **e3**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# hw-module subslot 0/1/0 cardtype t3` | Sets the card type for the SPA.<br>• **t3**—Specifies T3 connectivity of 44,210 kbps through the network, using B3ZS coding. This is the default setting.<br>• **e3**—Specifies a wide-area digital transmission scheme used predominantly in Europe that carries data at a rate of 34,010 kbps. |
| Step 3 | **controller t3** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller t3 0/1/0/0` | Creates a T3 controller and enters the T3 controller configuration mode. Specifies the T3 controller interface-path-id identifier with the *rack/slot/module/port* notation. |
| Step 4 | **mode** *mode*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# mode t1` | Sets the mode of interface. The possible modes are:<br>• atm—clear channel carrying atm<br>• e1—channelize into 21 E1s<br>• serial—clear channel carrying hdlc like payload<br>• t1—channelized into 28 T1s |
| Step 5 | **controller t1** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# controller t1 0/1/0/0` | Creates a T1 controller and enters the T1 controller configuration submode. Specifies the T1 controller *interface-path-id* with the *rack/slot/module/port* notation. |
| Step 6 | **mode** *mode*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1)# mode atm` | Sets the mode of interface. The possible modes are:<br>• atm—clear channel carrying atm<br>• e1—channelize into 21 E1s<br>• serial—clear channel carrying hdlc like payload<br>• t1—channelized into 28 T1s |

| | Command or Action | Purpose |
|---|---|---|
| **Step 7** | **interface atm** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1)# interface atm 0/1/0/0` | Creates an ATM interface and enters ATM interface configuration mode. Specifies the ATM interface with the *rack/slot/module/port* notation. |
| **Step 8** | **interface atm** *interface-path-id.subinterface* **point-to-point**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# interface atm 0/1/0/1.1 point-to-point` | Creates an ATM subinterface as one endpoint of a point-to-point link and enters ATM subinterface configuration mode. Specifies the ATM interface with the *rack/slot/module/port.subinterface* notation. |
| **Step 9** | **pvc** *vpi/vci*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-subif)# pvc 10/100` | Creates an ATM permanent virtual circuit (PVC) and enters ATM PVC configuration submode.<br>**Note** Only one PVC is allowed per subinterface. |
| **Step 10** | **ipv4 address** *ipv4_address/prefix*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-atm-vc)#ipv4 address 10.212.4.22 255.255.255.0` | Assigns an IP address and subnet mask to the subinterface. |
| **Step 11** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# end`<br>or<br>`RP/0/0/CPU0:router(config-sonet)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

# How to Configure Clear Channel ATM with Virtual Path (VP) Tunnels

This task describes how to configure a T3 ATM path with multiple VP tunnels.

## Prerequisites

The following prerequisites apply channelized ATM in Release 3.7.0:

- You must have a Cisco 2-Port Channelized T3/E3 ATM and Circuit Emulation Shared Port Adapter installed on your router.
- You should know how to configure the SONET controller as specified in the "How to Configure Clear Channel SONET Controllers" section of the *Configuring Clear Channel SONET Controllers on Cisco IOS XR Software* module.

## Restrictions

The following restrictions apply channelized ATM in Release 3.7.0:

- Channelized ATM is supported only on the Cisco 2-Port Channelized T3/E3 ATM and Circuit Emulation Shared Port Adapter.
- An ATM T3 path can be channelized only into T1 ATM channels or virtual path (VP) tunnels.
- DS0s are not supported.

### SUMMARY STEPS

1. **configure**
2. **hw-module subslot** *subslot-id* **cardtype {t3 | e3}**
3. **controller t3** *interface-path-id*
4. **mode** *mode*
5. **interface atm** *interface-path-id*
6. **vp-tunnel** *vpi*
7. **interface atm** *interface-path-id.subinterface* **point-to-point**
8. **pvc** *vpi/vci*
9. **ipv4 address** *ipv4_address/prefix*
10. **interface atm** *interface-path-id.subinterface* **point-to-point**
11. **pvc** *vpi/vci*
12. **ipv4 address** *ipv4_address/prefix*
13. **end**
    or
    **commit**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **hw-module subslot** *subslot-id* **cardtype** {**t3** \| **e3**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# hw-module subslot 0/1/0 cardtype t3` | Sets the card type for the SPA.<br>• **t3**—Specifies T3 connectivity of 44,210 kbps through the network, using B3ZS coding. This is the default setting.<br>• **e3**—Specifies a wide-area digital transmission scheme used predominantly in Europe that carries data at a rate of 34,010 kbps. |
| **Step 3** | **controller t3** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller t3 0/1/0/0` | Creates a T3 controller and enters the T3 controller configuration mode. Specifies the T3 controller *interface-path-id* with the *rack/slot/module/port* notation. |
| **Step 4** | **mode** *mode*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# mode t1` | Sets the mode of interface. The possible modes are:<br>• atm—clear channel carrying atm<br>• e1—channelize into 21 E1s<br>• serial—clear channel carrying hdlc like payload<br>• t1—channelized into 28 T1s |
| **Step 5** | **interface atm** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1)# interface atm 0/1/0/0` | Creates an ATM interface and enters ATM interface configuration mode. Specifies the ATM interface with the *rack/slot/module/port* notation. |
| **Step 6** | **vp-tunnel** *vpi*<br><br>**Example:**<br>`RP/0/0/CPU0:router (config)# vp-tunnel 10` | Configures a vp-tunnel on an ATM interface. |
| **Step 7** | **interface atm** *interface-path-id.subinterface* **point-to-point**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# interface atm 0/1/0/1.1 point-to-point` | Creates an ATM subinterface as one endpoint of a point-to-point link and enters ATM subinterface configuration mode. Specifies the ATM interface with the *rack/slot/module/port.subinterface* notation. |
| **Step 8** | **pvc** *vpi/vci*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-subif)# pvc 10/100` | Creates an ATM permanent virtual circuit (PVC) and enters ATM PVC configuration submode.<br>**Note** Only one PVC is allowed per subinterface. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 9** | **ipv4 address** *ipv4_address/prefix*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-atm-vc)#ipv4 address 10.212.8.22 255.255.255.0` | Assigns an IP address and subnet mask to the subinterface. |
| **Step 10** | **interface atm** *interface-path-id.subinterface* **point-to-point**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# interface atm 0/1/0/1.2 point-to-point` | Creates an ATM subinterface as one endpoint of a point-to-point link and enters ATM subinterface configuration mode. Specifies the ATM interface with the *rack/slot/module/port.subinterface* notation. |
| **Step 11** | **pvc** *vpi/vci*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-subif)# pvc 10/200` | Creates an ATM permanent virtual circuit (PVC) and enters ATM PVC configuration submode.<br><br>**Note** Only one PVC is allowed per subinterface. |
| **Step 12** | **ipv4 address** *ipv4_address/prefix*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-atm-vc)#ipv4 address 10.212.12.22 255.255.255.0` | Assigns an IP address and subnet mask to the subinterface. |
| **Step 13** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# end`<br>or<br>`RP/0/0/CPU0:router(config-sonet)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

# Attaching a Service-Policy to an Attachment Circuit

The QoS **service-policy** command can be configured for an attachment circuit in the following modes:

- PVC mode
- PVP mode
- Port mode
- Main Interface (non-port mode)

In PVC mode, PVP mode, and Port mode, the service policy is attached in the l2transport sub-interface mode. In non-port mode, the service policy is attached to the main interface.

Use the following procedures to attach a service-policy to an attachment circuit.

## SUMMARY STEPS

### PVC Mode

1. **config**
2. **interface atm** *interface-path-id.subinterface* **l2transport**
3. **pvc** *vpi/vci*
4. **service-policy input | output** *policy_name*
5. **commit**

### PVP Mode

1. **config**
2. **interface atm** *interface-path-id.subinterface* **l2transport**
3. **pvp** *vpi*
4. **service-policy input | output** *policy_name*
5. **commit**

### Port Mode

1. **config**
2. **interface atm** *interface-path-id*
3. **l2transport**
4. **service-policy input | output** *policy_name*
5. **commit**

### Main Interface (non-port mode)

1. **config**
2. **interface atm** *interface-path-id*
3. **service-policy input | output** *policy_name*
4. **commit**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **PVC Mode** | | |
| **Step 1** | **config**<br><br>**Example:**<br>`RP/0/0/CPU0:router# config terminal` | Enters global configuration mode. |
| **Step 2** | **interface atm** *interface-path-id.subinterface* **l2transport**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface atm 0/1/0/0.2 l2transport` | Creates a subinterface and enters ATM subinterface configuration mode for that subinterface. |
| **Step 3** | **interface atm** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router (config)# interface atm 0/1/0/1` | Enters interface configuration mode for an ATM interface. |
| **Step 4** | **service-policy input \| output** *policy_name*<br><br>**Example:**<br>`RP/0/0/CPU0(config-atm-l2transport-pvc)#service -policy input \| output atm_policy_1` | Attaches the specified service policy to the ATM PVC subinterface. |
| **Step 5** | **commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# commit` | Saves configuration changes. |
| **PVP Mode** | | |
| **Step 1** | **config**<br><br>**Example:**<br>`RP/0/0/CPU0:router# config terminal` | Enters global configuration mode. |
| **Step 2** | **interface atm** *interface-path-id.subinterface* **l2transport**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface atm 0/1/0/0.2 l2transport` | Creates a subinterface and enters ATM subinterface configuration mode for that subinterface. |
| **Step 3** | **pvp** *vpi*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-subif)# pvp 30` | (Optional) Creates an ATM PVP and enters ATM PVP configuration submode.<br><br>**Note** Only one PVP is allowed per subinterface. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **service-policy input \| output** *policy_name*<br><br>**Example:**<br>`RP/0/0/CPU0(config-atm-l2transport-pvp)#service -policy input \| output atm_policy_2` | Attaches the specified service policy to the ATM PVP subinterface. |
| **Step 5** | **commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# commit` | Saves configuration changes. |
| **Port Mode** | | |
| **Step 1** | **config**<br><br>**Example:**<br>`RP/0/0/CPU0:router# config terminal` | Enters global configuration mode. |
| **Step 2** | **interface atm** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router (config)# interface atm 0/1/0/1` | Enters interface configuration mode for an ATM interface. |
| **Step 3** | **l2transport**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-if)# l2transport` | Enters ATM Layer 2 transport configuration mode, and enables Layer 2 port mode on this ATM interface. |
| **Step 4** | **service-policy input \| output** *policy_name*<br><br>**Example:**<br>`RP/0/0/CPU0(config-if-l2)#service-policy input \| output atm_policy_3` | Attaches the specified service policy to the ATM Layer 2 subinterface. |
| **Step 5** | **commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# commit` | Saves configuration changes. |
| **Main Interface (non-port mode)** | | |
| **Step 1** | **config**<br><br>**Example:**<br>`RP/0/0/CPU0:router# config terminal` | Enters global configuration mode. |
| **Step 2** | **interface atm** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router (config)# interface atm 0/1/0/1` | Enters interface configuration mode for an ATM interface. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **service-policy input \| output** *policy_name*<br><br>**Example:**<br>`RP/0/0/CPU0(config-if)#service-policy input \| output atm_policy_4` | Attaches the specified service policy to the main ATM interface. |
| **Step 4** | **commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# commit` | Saves configuration changes. |

# ATM Configuration: Examples

This section provides the following configuration examples:


- ATM Interface Bring Up and Configuration: Example, page 70
- Point-To-Point ATM Subinterface Configuration: Example, page 70
- VP-Tunnel Configuration: Example, page 72
- Layer 2 AC Creation and Configuration: Example, page 73
- VC-Class Creation and Configuration: Example, page 73
- Channelized ATM Configuration: Example, page 74
- Clear Channel ATM with Virtual Path (VP) Tunnels Configuration: Example, page 75
- ATM Layer 2 QoS Configuration: Examples, page 75
- Verifying ATM Layer 2 QoS Configuration: Examples, page 77


## ATM Interface Bring Up and Configuration: Example

The following example shows how to bring up and configure an ATM interface:

```
RP/0/0/CPU0:router # configure
RP/0/0/CPU0:router(config)# interface atm 0/6/0/0
RP/0/0/CPU0:router(config-if)# atm address-registration
RP/0/0/CPU0:router(config-if)# no shutdown
RP/0/0/CPU0:router(config-if)# commit
```

## Point-To-Point ATM Subinterface Configuration: Example

The following example shows how to configure a point-to-point ATM subinterface on an ATM main interface:

```
RP/0/0/CPU0:router # configure
RP/0/0/CPU0:router (config)# interface atm 0/2/0/2.1 point-to-point
RP/0/0/CPU0:router (config-if)# ipv4 address 10.46.8.6/24
RP/0/0/CPU0:router (config-if)# pvc 0/200
RP/0/0/CPU0:router (config-atm-vc)# commit
RP/0/0/CPU0:router (config-atm-vc)# exit
RP/0/0/CPU0:router (config-if)# exit
RP/0/0/CPU0:router (config)# exit
```

```
RP/0/0/CPU0:router # show interfaces atm 0/2/0/2.1

ATM0/2/0/2.1 is up, line protocol is up
  Hardware is ATM network sub-interface(s)
  Description: Connect to P4_C12810 ATM 1/2.1
  Internet address is 10.46.8.6/24
  MTU 4470 bytes, BW 155000 Kbit
     reliability Unknown, txload Unknown, rxload Unknown
  Encapsulation AAL5/SNAP, controller loopback not set,
  Last clearing of "show interface" counters Unknown
  Datarate information unavailable.
  Interface counters unavailable.

RP/0/0/CPU0:router # show atm interface atm 0/2/0/3

Interface                               : ATM0/2/0/3
AAL Enabled                             : AAL5
Max-VP                                  : 254
Max-VC                                  : 2046
Configured L2 PVPs                      : 0
Configured L2 PVCs                      : 0
Configured L3 VP-Tunnels                : 0
Configured L3 PVCs                      : 1
L2 PVPs in Down State                   : 0
L2 PVCs in Down State                   : 0
L3 VP-Tunnels in Down State             : 0
L3 PVCs in Down State                   : 0
Cell packing count                      : 0

Received Side Statistics:
    Received Cells                      : 0
    Received Bytes                      : 0
    Received AAL Packets                : 0

Receive Side Cells Dropped:
    Unrecognized VPI/VCI                : 0

Receive Side AAL5 Packets Dropped:
    Unavailable SAR Buffer              : 0
    Non-Resource Exhaustion             : 0
    Reassembly Timeout                  : 0
    Zero Length                         : 0
    Unavailable Host Buffer             : 0
    Packet size exceeds MPS             : 0
    AAL5 Trailer Length Errors          : 0

Transmit Side Statistics:
    Transmitted Cells                   : 1899716067
    Transmitted Bytes                   : 0
    Transmitted AAL Packets             : 0

Transmit Side Cells Dropped:
    Unrecognized VPI/VCI                : 0

Transmit Side AAL5 Packets Dropped:
    Unavailable SAR Buffer              : 0
    Non-Resource Exhaustion             : 0
    WRED Threshold                      : 0
    WRED Random                         : 0

RP/0/0/CPU0:router # show atm pvc 10/100

Detailed display of VC(s) with VPI/VCI = 10/100
```

```
ATM0/2/0/3.100: VPI: 10 VCI: 100
UBR, PeakRate: 622000 Kbps
AAL5-LLC/SNAP
OAM frequency: 10 second(s), OAM retry frequency: 1 second(s),
OAM up retry count: 3, OAM down retry count: 5,
OAM Keep-vc-up: False, OAM AIS-RDI failure: None,
OAM AIS-RDI down count: 1, OAM AIS-RDI up time: 3 second(s),
OAM Loopback status: No loopback enabled,
OAM VC state: Loopback Not verified,
VC is not managed by OAM,

OAM cells received: 0,
F5 InEndLoop: 0, F5 InSegLoop: 0,
F5 InEndAIS: 0,  F5 InSegAIS: 0,
F5 InEndRDI: 0,  F5 InSegRDI: 0,
OAM cells sent: 0,
F5 OutEndLoop: 0, F5 OutSegLoop: 0,
F5 OutEndAIS: 0,  F5 OutSegAIS: 0,
F5 OutEndRDI: 0,  F5 OutSegRDI: 0,
OAM cells drops: 0

InPkts: 0                    OutPkts: 0
InBytes: 0                   OutBytes: 0
WRED pkt drop: 0
Non WRED pkt drop: 0

Internal state: READY
Status: UP
```

# VP-Tunnel Configuration: Example

The following example shows how to configure one endpoint of a vp-tunnel on an ATM main interface:

```
RP/0/0/CPU0:router # configure
RP/0/0/CPU0:router(config)# interface atm 0/6/0/0
RP/0/0/CPU0:router(config-if)# vp-tunnel 10
RP/0/0/CPU0:router(config-atm-vp-tunnel)# shape cbr 150000
RP/0/0/CPU0:router(config-atm-vp-tunnel)# f4oam disable
RP/0/0/CPU0:router(config-atm-vp-tunnel)# commit
RP/0/0/CPU0:router(config-atm-vp-tunnel)# exit
RP/0/0/CPU0:router(config-if)# exit
RP/0/0/CPU0:router(config)# exit
RP/0/0/CPU0:router# show atm vp-tunnel
                                  Data       Peak    Avg/Min      Burst
Interface         VPI       SC     VCs       Kbps      Kbps       Cells    Status

ATM0/2/0/3          30      UBR       2     155000        N/A        N/A        UP
```

The following example shows how to create and configure an ATM subinterface and PVC on one endpoint of a vp-tunnel, and then verify connectivity through that vp-tunnel:

```
RP/0/0/CPU0:router # configure
RP/0/0/CPU0:router(config)# interface atm 0/6/0/0.16 point-to-point
RP/0/0/CPU0:router(config-subif)# pvc 10/100
RP/0/0/CPU0:router (config-atm-vc)# commit
RP/0/0/CPU0:router (config-subif)# exit
RP/0/0/CPU0:router (config)# exit
RP/0/0/CPU0:router # ping atm interface atm 0/6/0/0.16 10/100

Sending 5, 53-byte end-to-end OAM echos, timeout is 2 seconds:
!!!!!
```

```
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/3/4 ms
```

# Layer 2 AC Creation and Configuration: Example

The following example shows how to create and configure one endpoint of a Layer 2 port mode AC:

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router (config)# interface atm 0/6/0/1
RP/0/0/CPU0:router (config-if)# l2transport
RP/0/0/CPU0:router (config-if-l2)# cell-packing 6 1
RP/0/0/CPU0:router(config-if-l2)# commit
```

The following example shows how to create and configure an AC on a Layer 2 subinterface with a PVC:

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# interface atm 0/1/0/0.230 l2transport
RP/0/0/CPU0:router(config-if)# pvc 15/230
RP/0/0/CPU0:router(config-atm-l2transport-pvc)# encapsulation aal0
RP/0/0/CPU0:router(config-atm-l2transport-pvc)# cell-packing 5 2
RP/0/0/CPU0:router(config-atm-l2transport-pvc)# shape cbr 622000
RP/0/0/CPU0:router(config-atm-l2transport-pvc)# commit
RP/0/0/CPU0:router(config-atm-l2transport-pvc)#
RP/0/0/CPU0:router(config-if)# exit
RP/0/0/CPU0:router(config)# exit
RP/0/0/CPU0:router# show atm pvc

                                              Peak  Avg/Min  Burst
Interface          VPI  VCI Type  Encaps  SC   Kbps   Kbps    Cells  Sts
ATM0/1/0/0.230      15  230 PVC   AAL0   UBR  622000    N/A      N/A   UP
ATM0/1/0/3.19       17   19 PVC   SNAP   UBR  622000    N/A      N/A   UP
```

The following example shows how to create and configure an AC on an ATM subinterface with a PVP:

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# interface atm 0/6/0/1.10 l2transport
RP/0/0/CPU0:router(config-if)# pvp 100
RP/0/0/CPU0:router(config-atm-l2transport-pvp)# cell-packing 5 2
RP/0/0/CPU0:router(config-atm-l2transport-pvp)# shape ubr 155000
RP/0/0/CPU0:router(config-atm-l2transport-pvp)# commit

RP/0/0/CPU0:router# show atm pvp interface atm 0/6/0/1

                                   Peak  Avg/Min   Burst
Interface          VPI  SC          Kbps     Kbps   Cells       Sts
ATM0/6/0/1.10      100  UBR       155000    N/A       N/A         UP
```

# VC-Class Creation and Configuration: Example

The following example shows how to configure a vc-class:

```
RP/0/0/CPU0:router # configure
RP/0/0/CPU0:router(config)# vc-class atm atm-class-1
RP/0/0/CPU0:router(config-vc-class-atm)# encapsulation aal5snap
RP/0/0/CPU0:router(config-vc-class-atm)# oam ais-rdi 25 5
RP/0/0/CPU0:router(config-vc-class-atm)# oam retry 5 10 5
RP/0/0/CPU0:router(config-vc-class-atm)# oam-pvc manage 300
RP/0/0/CPU0:router(config-vc-class-atm)# shape cbr 100000
RP/0/0/CPU0:router(config-vc-class-atm)# commit
```

The following example shows how to attach a vc-class to an ATM main interface:

```
RP/0/0/CPU0:router # configure
RP/0/0/CPU0:router(config)# interface ATM0/2/0/0.1 point-to-point
RP/0/0/CPU0:router (config-if)# class-int atm-class-1
RP/0/0/CPU0:router (config-if)# commit
```

The following example shows how to attach a vc-class to an ATM subinterface:

```
RP/0/0/CPU0:router # configure
RP/0/0/CPU0:router(config)# interface ATM0/2/0/0.1 point-to-point
RP/0/0/CPU0:router(config-if)# pvc 10/100
RP/0/0/CPU0:router (config-atm-vc)# class-vc atm-class-1
RP/0/0/CPU0:router (config-atm-vc)# commit
```

The following example shows how to display information about a specific ATM vc-class:

```
RP/0/0/CPU0:router # show atm vc-class atm-class-1
ATM vc-class atm-class-1

  encapsulation          -    aal5snap
  shape                  -    cbr 100000
  oam ais-rdi            -    not configured
  oam retry              -    not configured
  oam-pvc                -    manage 300
```

The following example shows how to display configuration information for the parameters on a virtual circuit (VC) class that is associated with a particular PVC:

```
RP/0/0/CPU0:router # show atm class-link 10/100

Detailed display of VC(s) with VPI/VCI = 10/100

Class link for VC 10/100
ATM0/2/0/0.1: VPI: 10 VCI: 100
shape : cbr 100000 (VC-class configured on VC)
encapsulation : aal5snap (VC-class configured on VC)
oam-pvc : manage 300   (VC-class configured on VC)
oam retry : 3 5 1 (Default value)
oam ais-rdi : 1 3 (Default value)
```

# Channelized ATM Configuration: Example

The following example shows how to configure a T3 path into mutliple T1 channels carrying ATM traffic.

```
RP/0/0/CPU0:router# config
RP/0/0/CPU0:router(config)# hw-module subslot 0/4/0 cardtype t3
RP/0/0/CPU0:router(config)# controller T3 0/4/0/0
RP/0/0/CPU0:router(config-t3)# mode t1
RP/0/0/CPU0:router(config-t3)# controller T1 0/4/0/0/1
RP/0/0/CPU0:router(config-t1)# mode atm
RP/0/0/CPU0:router(config-t1)# interface ATM 0/4/0/0/1
RP/0/0/CPU0:router(config-if)# interface ATM 0/4/0/0/1.1 point-to-point
RP/0/0/CPU0:router(config-subif)# pvc 10/100
RP/0/0/CPU0:router(config-atm-vc)# ipv4 address 10.212.4.22 255.255.255.0
RP/0/0/CPU0:router(config-subif)# commit
```

# Clear Channel ATM with Virtual Path (VP) Tunnels Configuration: Example

The following example shows how to configure a T3 ATM path with multiple VP tunnels.

```
RP/0/0/CPU0:router# config
RP/0/0/CPU0:router(config)# hw-module subslot 0/4/0 cardtype t3
RP/0/0/CPU0:router(config)# controller T3 0/4/0/1
RP/0/0/CPU0:router(config-t3)# mode atm
RP/0/0/CPU0:router(config-t3)# interface ATM 0/4/0/1
RP/0/0/CPU0:router(config-if)# vp-tunnel 10
RP/0/0/CPU0:router(config-atm-vp-tunnel)# interface ATM 0/4/0/1.1 point-to$
RP/0/0/CPU0:router(config-subif)# pvc 10/100
RP/0/0/CPU0:router(config-atm-vc)# ipv4 address 10.212.8.22 255.255.255.0
RP/0/0/CPU0:router(config-subif)# interface ATM 0/4/0/1.2 point-to-point
RP/0/0/CPU0:router(config-subif)# pvc 10/200
RP/0/0/CPU0:router(config-atm-vc)# ipv4 address 10.212.12.22 255.255.255.0
RP/0/0/CPU0:router(config-subif)# commit
```

# ATM Layer 2 QoS Configuration: Examples

The following examples show how to configure QoS for ATM. For complete information on configuring QoS and QoS commands, refer to the Cisco XR 12000 Series Router Modular Quality of Service Configuration Guide and the Cisco XR 12000 Series Router Modular Quality of Service Command Reference.

## Attaching a Service-Policy to an Attachment Circuit Configuration: Example

### PVC Mode

```
config
   interface ATM 0/1/0/0.2 l2transport
      pvc 10/2
         service-policy input | output atm_policy_o
```

### PVP Mode

```
config
   interface ATM 0/1/0/0.3 l2transport
      pvp 30
         service-policy input atm_policy_i
```

### Port Mode

```
config
   interface ATM 0/1/0/0
      l2transport
         service-policy input atm_policy_i
```

### Main Interface (non-port mode)

```
config
   interface ATM 0/1/0/0
      service-policy input | output atm_policy_o
```

## Policy Map Configuration for CBR/UBR: Example

For CBR.1 (real-time traffic) and UBR (best effort, non-real time traffic) you must specify the PCR and delay tolerance parameters for policing. The main difference between the configurations for UBR.1 and UBR.2 traffic is that for UBR.2 traffic, the exceed action includes the **set-clp-transmit** option to tag non-conforming cells. The police rate can also be expressed as a percentage.

The following example shows how to configure a QoS policy map for CBR/UBR:

```
policy-map CBR1
    class class-default
          police rate pcr cellsps delay-tolerance cdvt us
               conform-action action
               exceed-action action
```

## Policy Map Configuration for VBR.1: Example

For VBR.1 real-time and non-real time traffic you must specify the PCR, SCR, and delay tolerance parameters for for policing. The **atm-mbs** parameter can be specified to define the burst allowed on the SCR bucket. The police rates can also be expressed as percentages. Class atm_clp1 is allowed with police actions.

The following example shows how to configure a QoS policy map for VBR.1:

```
policy-map VBR1
    class class-default
          police rate scr cellsps atm-mbs mbs cells peak-rate pcr cellsps delay-tolerance
cdvt us
               conform-action action
               exceed-action action
```

## Policy Map Configuration for VBR.2 and VBR.3: Example

For VBR.2 and VBR.3 real-time and non-real time traffic you must specify the PCR, SCR, and delay tolerance parameters for policing. The **atm-mbs** parameter can be specified to define the burst allowed on the SCR bucket. The main difference between VBR.1 and VBR.2/VBR.3 is that the SCR bucket is for CLP0 cells only. The police rates can be expressed as percentages. The child policy can have other set actions and can match on ATM CLP1.

The following example shows how to configure a hierarchical policy for VBR.2:

```
policy-map child
    class atm_clp0
          police rate scr cellsps atm-mbs mbs cells
               conform-action action
               exceed-action action

policy-map VBR2
    class class-default
          police rate pcr cellsps delay-tolerance cdvt us
               conform-action action
               exceed-action action
      service-policy child
```

### Policy Map Configuration to Exclude OAM Cells: Example

OAM cells can be excluded from being policed by configuring the classification criteria. Since **match not** is not supported, the different classes must be explicitly configured:.

The following example shows how to configure a policy map to exclude OAM cells:

```
class-map clp-0-1
 match clp 0
 match clp 1

policy-map child
    class atm-oam
        set
    class class-default
          police rate scr cellsps atm-mbs mbs cells
               conform-action action
               exceed-action action

policy-map VBR2
    class clp-0-1
          police rate pcr cellsps delay-tolerance cdvt us
               conform-action action
               exceed-action action
      service-policy child
```

### Policy Map Configuration for Dual Queue Limit: Example

Dual Queue limit configuration is supported on egress L2 ATM interfaces to differentiate between CLP0 and CLP1 cells.

**Note** For dual queue, only output service policies are supported. Input service policies are not supported.

The following example shows how to configure a policy map for Dual Queue Limit:

```
policy-map q-limit
    class class-default
        queue-limit atm-clp Threshold {[ms|us|cells]} Tail-drop-threshold {[ms|us|cells]}
```

## Verifying ATM Layer 2 QoS Configuration: Examples

The following examples show how to display policing results for an ATM interface policy map:

```
show policy-map interface ATM 0/3/0/0.12 input

ATM 0/3/0/0.12 input: pvc1

Class class-default
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :                 0/0             0
    Transmitted         :                 0/0             0
    Total Dropped       :                 0/0             0
```

```
show policy-map interface ATM 0/3/0/0.12 output

ATM 0/3/0/0.12 output: pvc1

Class class-default
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :                  0/0               0
    Transmitted         :                  0/0               0
    Total Dropped       :                  0/0               0
```

The following examples show how to display the configured QoS properties for an ATM interface policy map:

```
show qos interface atm 0/3/0/0.12 input

Interface ATM0_3_0_0.12  --   Direction: input
Policy                  :   pvc1
Total number of classes:   1
Cell Packing Criteria = CELL_PACK_TIMER_MTU
---------------------------------------------------
LEVEL1 class: classid    =   0x1
class name               =   class-default
new exp                  =   6
```

```
show qos interface atm 0/3/0/0.12 output

Interface ATM0_3_0_0.12  --   Direction: output
Policy                  :   pvc1
Total number of classes:   1
Cell Packing Criteria = CELL_PACK_TIMER_MTU
---------------------------------------------------
LEVEL1 class: classid    =   0x1
class name               =   class-default
new exp                  =   6
```

# Additional References

These sections provide references related to implementing ATM for Cisco IOS XR software.

## Related Documents

| Related Topic | Document Title |
|---|---|
| ATM commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples | *Cisco IOS XR Interface and Hardware Command Reference* |

## 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 |
|---|---|
| — | To locate and download MIBs using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: http://www.cisco.com/go/mibs |

## RFCs

| RFCs | Title |
|---|---|
| RFC 1483 | *Multiprotocol Encapsulation over ATM Adaptation Layer 5* |
| RFC 1577 | *Classical IP and ARP over ATM.* |
| RFC 2225 | *Classical IP and ARP over ATM* |
| RFC 2255 | *The LDAP URL Format* |
| RFC 2684 | *Multiprotocol Encapsulation over ATM Adaptation Layer 5.* |
| RFC 4385 | *Pseudowire Emulation Edge-to-Edge (PWE3) Control Word for Use over an MPLS PSN* |
| RFC 4717 | *Encapsulation Methods for Transport of Asynchronous Transfer Mode (ATM) over MPLS Networks* |
| RFC 4816 | *Pseudowire Emulation Edge-to-Edge (PWE3) Asynchronous Transfer Mode (ATM) Transparent Cell Transport Service* |

## 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/support |

# Advanced Configuration and Modification of the Management Ethernet Interface on Cisco IOS XR Software

This module describes the configuration of Management Ethernet interfaces on the Cisco IOS XR Software.

Before you can use Telnet to access the router through the LAN IP address, you must set up a Management Ethernet interface and enable Telnet servers, as described in the *Configuring General Router Features* module of the *Cisco IOS XRGetting Started Guide*. This module describes how to modify the default configuration of the Management Ethernet interface after it has been configured, as described in *Cisco IOS XR Getting Started Guide.*

**Note** Forwarding between physical layer interface modules (PLIM) ports and Management Ethernet interface ports is disabled by default. To enable forwarding between PLIM ports and Management Ethernet interface ports, use the **rp mgmtethernet forwarding** command.

**Note** Although the Management Ethernet interfaces on the system are present by default, the user must configure these interfaces to use them for accessing the router, using protocols and applications such as Simple Network Management Protocol (SNMP), Common Object Request Broker Architecture (CORBA), HTTP, extensible markup language (XML), TFTP, Telnet, and command-line interface (CLI).

**Feature History for Configuring Management Ethernet Interfaces**

| Release | Modification |
|---|---|
| Release 3.2 | This feature was first supported on the Cisco XR 12000 Series Router. |
| Release 3.3.0 | Manual configuration of the Management Ethernet interface is the only option. The initial prompts that originally walked the user through Management Ethernet interface configuration upon software installation were removed. |

# Contents


- Prerequisites for Configuring Management Ethernet Interfaces, page 82
- Information About Configuring Management Ethernet Interfaces, page 82

- How to Perform Advanced Management Ethernet Interface Configuration, page 83
- Configuration Examples for Management Ethernet Interfaces, page 90
- Additional References, page 91


# Prerequisites for Configuring Management Ethernet 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 performing the Management Ethernet interface configuration procedures that are described in this module, be sure that the following tasks and conditions are met:

- You have performed the initial configuration of the Management Ethernet interface, as described in the *Configuring General Router Features* module of *Cisco IOS XR Getting Started Guide*.
- To use the **show running-config** command, you must be in a user group associated with a task group that includes the proper task IDs for configuration management commands. The Task ID for the **show running-config** command is listed in *Cisco IOS XR System Management Command Reference.*
- You know how to apply the generalized interface name specification *rack/slot/module/port*.

  For further information on interface naming conventions, refer to *Cisco IOS XR Getting Started Guide*.

**Note** Note that, for transparent switchover, both active and standby Management Ethernet interfaces are expected to be physically connected to the same LAN or switch.

# Information About Configuring Management Ethernet Interfaces

To configure Management Ethernet interfaces, you must understand the following concept:


- Default Interface Settings, page 82


## Default Interface Settings

Table 1 describes the default Management Ethernet interface settings that can be changed by manual configuration. Default settings are not displayed in the **show running-config** command output.

*Table 1 Management Ethernet Interface Default Settings*

| Parameter | Default Value | Configuration File Entry |
|---|---|---|
| Speed in Mbps | Speed is autonegotiated. | **speed** [**10** \| **100** \| **1000**]<br>To return the system to autonegotiate speed, use the **no speed** [**10** \| **100** \| **1000**] command. |
| Duplex mode | Duplex mode is autonegotiated. | **duplex** {**full** \| **half**}<br>To return the system to autonegotiated duplex operation, use the **no duplex {full \| half}** command, as appropriate. |
| MAC address | MAC address is read from the hardware burned-in address (BIA). | **mac-address** *address*<br>To return the device to its default MAC address, use the **no mac-address** *address* command. |

# How to Perform Advanced Management Ethernet Interface Configuration

This section contains the following procedures:

- Configuring a Management Ethernet Interface, page 83 (required)
- Configuring the Duplex Mode for a Management Ethernet Interface, page 85 (optional)
- Configuring the Speed for a Management Ethernet Interface, page 86 (optional)
- Modifying the MAC Address for a Management Ethernet Interface, page 88 (optional)
- Verifying Management Ethernet Interface Configuration, page 89 (optional)

## Configuring a Management Ethernet Interface

Perform this task to configure a Management Ethernet interface. This procedure provides the minimal configuration required for the Management Ethernet interface.

**Note** You do not need to perform this task if you have already set up the Management Ethernet interface to enable telnet servers, as described in the *Configuring General Router Features* module of the *Getting Started Guide*.

### SUMMARY STEPS

1. **configure**
2. **interface MgmtEth** *interface-path-id*
3. **ipv4 address** *ip-address mask*
4. **mtu** *bytes*
5. **no shutdown**

6. **end**
   or
   **commit**
7. **show interfaces MgmtEth** *interface-path-id*

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br>`RP/0/RP0/CPU0:router# configure terminal` | Enters global configuration mode. |
| **Step 2** | **interface MgmtEth** *interface-path-id*<br><br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# interface MgmtEth 0/RP0/CPU0/0` | Enters interface configuration mode and specifies the Ethernet interface name and notation *rack/slot/module/port*.<br>The example indicates port 0 on the RP card that is installed in slot 0. |
| **Step 3** | **ipv4 address** *ip-address mask*<br><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if)# ipv4 address 172.18.189.38 255.255.255.224` | Assigns an IP address and subnet mask to the interface.<br>• Replace *ip-address* with the primary IPv4 address for the 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. |
| **Step 4** | **mtu** *bytes*<br><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if# mtu 1448` | (Optional) Sets the maximum transmission unit (MTU) byte value for the interface. The default is 1514.<br>• The default is 1514 bytes.<br>• The range for the Management Ethernet interface Interface **mtu** values is 64 to 1514 bytes. |
| **Step 5** | **no shutdown**<br><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if)# no shutdown` | Removes the shutdown configuration, which removes the forced administrative down on the interface, enabling it to move to an up or down state. |

| | Command or Action | Purpose |
|---|---|---|
| Step 6 | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if)# end`<br>or<br>`RP/0/RP0/CPU0:router(config-if)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 7 | **show interfaces MgmtEth** *interface-path-id*<br><br>**Example:**<br>`RP/0/RP0/CPU0:router# show interfaces MgmtEth 0/RP0/CPU0/0` | (Optional) Displays statistics for interfaces on the router. |

# Configuring the Duplex Mode for a Management Ethernet Interface

Perform this task to configure the duplex mode of the Management Ethernet interfaces for the RPs.

## SUMMARY STEPS

1. **configure**
2. **interface MgmtEth** *interface-path-id*
3. **duplex** [**full** | **half**]
4. **end**
   or
   **commit**

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **interface MgmtEth** *interface-path-id*<br><br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# interface MgmtEth 0/RP0/CPU0/0` | Enters interface configuration mode and specifies the Management Ethernet interface name and instance. |
| **Step 3** | **duplex** [**full** \| **half**]<br><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if)# duplex full` | Configures the interface duplex mode. Valid options are **full** or **half**.<br><br>**Note** To return the system to autonegotiated duplex operation, use the **no duplex** command. |
| **Step 4** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if)# end`<br>or<br>`RP/0/RP0/CPU0:router(config-if)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

# Configuring the Speed for a Management Ethernet Interface

Perform this task to configure the speed of the Management Ethernet interfaces for the RPs.

### SUMMARY STEPS

1. **configure**
2. **interface MgmtEth** *interface-path-id*
3. **speed** {**10** | **100** | **1000**}

4. **end**
   or
   **commit**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **interface MgmtEth** *interface-path-id*<br><br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# interface MgmtEth 0/RP0/CPU0/0` | Enters interface configuration mode and specifies the Management Ethernet interface name and instance. |
| **Step 3** | **speed** {**10** \| **100** \| **1000**}<br><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if)# speed 100` | Configures the interface speed parameter.<br>Valid **speed** options are **10**, **100** or **1000** Mbps.<br>**Note** The default Management Ethernet interface speed is autonegotiated.<br>**Note** To return the system to the default autonegotiated speed, use the **no speed** command. |
| **Step 4** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if)# end`<br>or<br>`RP/0/RP0/CPU0:router(config-if)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

# Modifying the MAC Address for a Management Ethernet Interface

Perform this task to configure the MAC layer address of the Management Ethernet interfaces for the RPs.

## SUMMARY STEPS

1. **configure**
2. **interface MgmtEth** *interface-path-id*
3. **mac-address** *address*
4. **end**
   or
   **commit**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | `configure`<br><br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | `interface MgmtEth` *interface-path-id*<br><br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# interface MgmtEth 0/RP0/CPU0/0` | Enters interface configuration mode and specifies the Management Ethernet interface name and instance. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **`mac-address`** *`address`*<br><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if)# mac-address 0001.2468.ABCD` | Configures the MAC layer address of the Management Ethernet interface.<br><br>**Note** To return the device to its default MAC address, use the **no mac-address** *address* command. |
| **Step 4** | **`end`**<br>or<br>**`commit`**<br><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if)# end`<br>or<br>`RP/0/RP0/CPU0:router(config-if)# commit` | Saves configuration changes.<br><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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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><br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

# Verifying Management Ethernet Interface Configuration

Perform this task to verify configuration modifications on the Management Ethernet interfaces for the RPs.

## SUMMARY STEPS

1. **show interfaces MgmtEth** *interface-path-id*
2. **show running-config**

| | | |
|---|---|---|
| **Step 1** | **`show interfaces MgmtEth`** *`interface-path-id`*<br><br>**Example:**<br>`RP/0/RP0/CPU0:router# show interfaces MgmtEth 0/RP0/CPU0/0` | Displays the Management Ethernet interface configuration. |
| **Step 2** | **`show running-config interface MgmtEth`** *`interface-path-id`*<br><br>**Example:**<br>`RP/0/RP0/CPU0:router# show running-config interface MgmtEth 0/RP0/CPU0/0` | Displays the running configuration. |

# Configuration Examples for Management Ethernet Interfaces

This section provides the following configuration examples:

- Configuring a Management Ethernet Interface: Example, page 90

## Configuring a Management Ethernet Interface: Example

This example displays advanced configuration and verification of the Management Ethernet interface on the RP:

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface MgmtEth 0/RP0/CPU0/0
RP/0/RP0/CPU0:router(config)# ipv4 address 172.29.52.70 255.255.255.0
RP/0/RP0/CPU0:router(config-if)# speed 100
RP/0/RP0/CPU0:router(config-if)# duplex full
RP/0/RP0/CPU0:router(config-if)# no shutdown
RP/0/RP0/CPU0:router(config-if)# commit
RP/0/RP0/CPU0:Mar 26 01:09:28.685 :ifmgr[190]:%LINK-3-UPDOWN :Interface
MgmtEth0/RP0/CPU0/0, changed state to Up
RP/0/RP0/CPU0:router(config-if)# end

RP/0/RP0/CPU0:router# show interfaces MgmtEth 0/RP0/CPU0/0

MMgmtEth0/RP0/CPU0/0 is up, line protocol is up
  Hardware is Management Ethernet, address is 0011.93ef.e8ea (bia 0011.93ef.e8ea
)
  Description: Connected to Lab LAN
  Internet address is 172.29.52.70/24
  MTU 1514 bytes, BW 100000 Kbit
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA,  loopback not set,
  ARP type ARPA, ARP timeout 04:00:00
  Last clearing of "show interface" counters never
  5 minute input rate 3000 bits/sec, 7 packets/sec
  5 minute output rate 0 bits/sec, 1 packets/sec
     30445 packets input, 1839328 bytes, 64 total input drops
     0 drops for unrecognized upper-level protocol
     Received 23564 broadcast packets, 0 multicast packets
              0 runts, 0 giants, 0 throttles, 0 parity
     57 input errors, 40 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     171672 packets output, 8029024 bytes, 0 total output drops
     Output 16 broadcast packets, 0 multicast packets
     0 output errors, 0 underruns, 0 applique, 0 resets
     0 output buffer failures, 0 output buffers swapped out
     1 carrier transitions

RP/0/RP0/CPU0:router# show running-config interface MgmtEth 0/RP0/CPU0/0

interface MgmtEth0/RP0/CPU0/0
 description Connected to Lab LAN
 ipv4 address 172.29.52.70 255.255.255.0
!
```

# Additional References

These sections provide references related to Management Ethernet interface configuration.

## Related Documents

| Related Topic | Document Title |
|---|---|
| Cisco IOS XR master command reference | *Cisco IOS XR Master Commands List* |
| Cisco IOS XR interface configuration commands | *Cisco IOS XR Interface and Hardware Component Command Reference* |
| Initial system bootup and configuration information for a router using the Cisco IOS XR software. | *Cisco IOS XR Getting Started Guide* |
| Information about user groups and task IDs | *Cisco IOS XR Interface and Hardware Component Command Reference* |

## Standards

| Standards | Title |
|---|---|
| No new or modified standards are supported by this feature, and support for existing standards has not been modified by the 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:<br>http://www.cisco.com/go/mibs |

## RFCs

| 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/support |

# Configuring Ethernet Interfaces on Cisco IOS XR Software

This module describes the configuration of Ethernet interfaces on the Cisco XR 12000 Series Router.

The distributed Gigabit Ethernet, 10-Gigabit Ethernet, and Fast Ethernet architecture and features deliver network scalability and performance, while enabling service providers to offer high-density, high-bandwidth networking solutions designed to interconnect the router with other systems in POPs, including core and edge routers and Layer 2 switches.

**Feature History for Configuring Ethernet Interfaces on Cisco IOS XR Software**

| Release | Modification |
|---|---|
| Release 3.2 | Support was added for the Cisco XR 12000 Series Router.<br>Support was added on the Cisco XR 12000 Series Router for the following SPAs:<br>• 1-Port 10-Gigabit Ethernet SPA<br>• 5-Port Gigabit Ethernet SPA<br>• 10-Port Gigabit Ethernet SPA |
| Release 3.3.0 | Support was added for egress MAC accounting on the 8-port 10-Gigabit Ethernet PLIM.<br>Support was added on the Cisco XR 12000 Series Router for the following SIPs:<br>• Cisco XR 12000 SIP-401<br>• Cisco XR 12000 SIP-501<br>• Cisco XR 12000 SIP-601<br>Support was added on the Cisco XR 12000 Series Router for the 8-Port FastEthernet SPA. |
| Release 3.4.0 | The Layer 2 Virtual Private Network (L2VPN) feature was first supported on Ethernet interfaces.<br>Support was added for the 8-Port 1-Gigabit Ethernet SPA. |
| Release 3.4.1 | Support was added on the Cisco XR 12000 Series Router for the 2-Port Gigabit Ethernet SPA. |

# Contents


- Prerequisites for Configuring Ethernet Interfaces, page 94
- Information About Configuring Ethernet, page 95
- Configuring Ethernet Interfaces, page 103
- Configuration Examples for Ethernet, page 112
- Where to Go Next, page 114
- Additional References, page 115


# Prerequisites for Configuring Ethernet 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 Ethernet interfaces, be sure that the following tasks and conditions are met:

- Confirm that at least one of the following line cards supported on the router is installed:
  - 4-Port 1-Gigabit Ethernet physical layer interface module (PLIM)
  - 2-Port Gigabit Ethernet SPA
  - 5-Port Gigabit Ethernet SPA
  - 10-Port Gigabit Ethernet SPA
  - 1-Port 10-Gigabit Ethernet SPA
  - 1-Port 10-Gigabit Ethernet WAN SPA
  - 8-Port Gigabit Ethernet SPA
  - 8-Port Fast Ethernet SPA
- Know the interface IP address.
- You know how to apply the specify the generalized interface name with the generalized notation *rack/slot/module/port*.
- If you are configuring a 10-Gigabit Ethernet interface on a 10-GE DWDM PLIM, you must have configured the DWDM controller.

# Information About Configuring Ethernet

Ethernet is defined by the IEEE 802.3 international standard. It enables the connection of up to 1024 nodes over coaxial, twisted-pair, or fiber-optic cable.

The Cisco XR 12000 Series Router supports Fast Ethernet (100 Mbps), Gigabit Ethernet (1000 Mbps), and 10-Gigabit Ethernet (10 Gbps) interfaces.

This section provides the following information sections:


- Default Configuration Values for Gigabit Ethernet and 10-Gigabit Ethernet, page 95
- Gigabit Ethernet Protocol Standards Overview, page 97
- MAC Address, page 98
- MAC Accounting, page 99
- Ethernet MTU, page 99
- Flow Control on Ethernet Interfaces, page 99
- 802.1Q VLAN, page 100
- VRRP, page 100
- HSRP, page 100
- Duplex Mode on Fast Ethernet Interfaces, page 101
- Fast Ethernet Interface Speed, page 101
- Link Autonegotiation on Ethernet Interfaces, page 101
- Carrier Delay on Ethernet Interfaces, page 102


## Default Configuration Values for Gigabit Ethernet and 10-Gigabit Ethernet

Table 2 describes the default interface configuration parameters that are present when an interface is enabled on a Gigabit Ethernet or 10-Gigabit Ethernet modular services card and its associated PLIM.

**Note** You must use the **shutdown** command to bring an interface administratively down. The interface default is **no shutdown**. When a modular services card is first inserted into the router, if there is no established preconfiguration for it, the configuration manager adds a shutdown item to its configuration. This shutdown can be removed only be entering the **no shutdown** command.

*Table 2 Gigabit Ethernet and 10-Gigabit Ethernet Modular Services Card Default Configuration Values*

| Parameter | Configuration File Entry | Default Value |
|---|---|---|
| MAC accounting | **mac-accounting** | **off** |
| Flow control | **flow-control** | egress on<br>ingress off |

*Table 2 Gigabit Ethernet and 10-Gigabit Ethernet Modular Services Card Default Configuration Values*

| Parameter | Configuration File Entry | Default Value |
|---|---|---|
| MTU | **mtu** | • 1514 bytes for normal frames<br>• 1518 bytes for 802.1Q tagged frames.<br>• 1522 bytes for Q-in-Q frames. |
| MAC address | **mac address** | Hardware burned-in address (BIA) |

## Default Configuration Values for Fast Ethernet

Table 3 describes the default interface configuration parameters that are present when an interface is enabled on the Fast Ethernet SPA card and its associated PLIM.

**Note** You must specifically configure the **shutdown** command to bring an interface administratively down. The interface default is **no shutdown**. When a modular services card is first inserted into the router, if there is no established preconfiguration for it, the configuration manager adds a shutdown item to its configuration. This shutdown can be removed only be entering the **no shutdown** command.

*Table 3 Fast Ethernet Default Configuration Values*

| Parameter | Configuration File Entry | Default Value |
|---|---|---|
| MAC accounting | **mac-accounting** | **off** |
| Duplex operation | **duplex full**<br>**duplex half** | Auto-negotiates duplex operation |
| MTU | **mtu** | 1500 bytes |
| Interface speed | **speed** | 100 Mbps |
| Auto-negotiation | **negotiation auto** | **disable** |

## Layer 2 VPN on Ethernet Interfaces

Layer 2 Virtual Private Network (L2VPN) connections emulate the behavior of a LAN across an L2 switched, IP or MPLS-enabled IP network, allowing Ethernet devices to communicate with each other as if they were connected to a common LAN segment.

The L2VPN feature enables service providers (SPs) to provide Layer 2 services to geographically disparate customer sites. Typically, an SP uses an access network to connect the customer to the core network. This access network is may use a mixture of Layer 2 technologies, such as Ethernet, ATM and Frame Relay. The connection between the customer site and the nearby SP edge router is known as an Attachment Circuit (AC).

Traffic from the customer travels over this link to the edge of the SP core network. The traffic then tunnels through an L2VPN over the SP core network to another edge router. The edge router sends the traffic down another attachment circuit (AC) to the customer's remote site.

The L2VPN feature enables users to implement different types of end-to-end services.

Cisco IOS XR software supports a point-to-point end-to-end service, where two Ethernet circuits are connected together. An L2VPN Ethernet port can operate in one of two modes:

- Port Mode—In this mode, all packets reaching the port are sent over the PW (pseudowire), regardless of any VLAN tags that are present on the packets. In VLAN mode, the configuration is performed under the l2transport configuration mode.
- VLAN Mode—Each VLAN on a CE (customer edge) or access network to PE (provider edge) link can be configured as a separate L2VPN connection (using either VC type 4 or VC type 5). In VLAN mode, the configuration is performed under the individual subinterface.

Switching can take place in three ways:

- AC-to-PW—Traffic reaching the PE is tunneled over a PW (and conversely, traffic arriving over the PW is sent out over the AC). This is the most common scenario.
- Local switching—Traffic arriving on one AC is immediately sent out of another AC without passing through a pseudowire.
- PW stitching—Traffic arriving on a PW is not sent to an AC, but is sent back into the core over another PW.

Keep the following in mind when configuring L2VPN on an Ethernet interface:

- L2VPN links support QoS (Quality of Service) and MTU (maximum transmission unit) configuration.
- If your network requires that packets are transported transparently, you may need to modify the packet's destination MAC (Media Access Control) address at the edge of the Service Provider (SP) network. This prevents the packet from being consumed by the devices in the ST network.
- Cisco IOS XR software supports up to 4,000 ACs per line card. Note that not all line cards can support as many as 4,000 ACs. Refer to the specifications of the individual line card for details on the maximum number of ACs supported.

Use the **show interfaces** command to display AC and PW information.

To configure a point-to-point pseudowire xconnect on an AC, see the *Implementing MPLS Layer 2 VPNs* module of the *Cisco IOS XR Multiprotocol Label Switching Configuration Guide.*

To attach Layer 2 service policies, such as QoS, to the Ethernet interface, refer to the appropriate Cisco IOS XR software configuration guide.

# Gigabit Ethernet Protocol Standards Overview

The Gigabit Ethernet interfaces support the following protocol standards:

- IEEE 802.3 Physical Ethernet Infrastructure, page 98
- IEEE 802.3ab 1000BASE-T Gigabit Ethernet, page 98
- IEEE 802.3z 1000 Mbps Gigabit Ethernet, page 98
- IEEE 802.3ae 10 Gbps Ethernet, page 98

These standards are further described in the sections that follow.

### IEEE 802.3 Physical Ethernet Infrastructure

The IEEE 802.3 protocol standards define the physical layer and MAC sublayer of the data link layer of wired Ethernet. IEEE 802.3 uses Carrier Sense Multiple Access with Collision Detection (CSMA/CD) access at a variety of speeds over a variety of physical media. The IEEE 802.3 standard covers 10 Mbps Ethernet. Extensions to the IEEE 802.3 standard specify implementations for Gigabit Ethernet, 10-Gigabit Ethernet, and Fast Ethernet.

### IEEE 802.3ab 1000BASE-T Gigabit Ethernet

The IEEE 802.3ab protocol standards, or Gigabit Ethernet over copper (also known as 1000BaseT) is an extension of the existing Fast Ethernet standard. It specifies Gigabit Ethernet operation over the Category 5e/6 cabling systems already installed, making it a highly cost-effective solution. As a result, most copper-based environments that run Fast Ethernet can also run Gigabit Ethernet over the existing network infrastructure to dramatically boost network performance for demanding applications.

### IEEE 802.3z 1000 Mbps Gigabit Ethernet

Gigabit Ethernet builds on top of the Ethernet protocol, but increases speed tenfold over Fast Ethernet to 1000 Mbps, or 1 Gbps. Gigabit Ethernet allows Ethernet to scale from 10 or 100 Mbps at the desktop to 100 Mbps up to 1000 Mbps in the data center. Gigabit Ethernet conforms to the IEEE 802.3z protocol standard.

By leveraging the current Ethernet standard and the installed base of Ethernet and Fast Ethernet switches and routers, network managers do not need to retrain and relearn a new technology in order to provide support for Gigabit Ethernet.

### IEEE 802.3ae 10 Gbps Ethernet

Under the International Standards Organization's Open Systems Interconnection (OSI) model, Ethernet is fundamentally a Layer 2 protocol. 10-Gigabit Ethernet uses the IEEE 802.3 Ethernet MAC protocol, the IEEE 802.3 Ethernet frame format, and the minimum and maximum IEEE 802.3 frame size. 10 Gbps Ethernet conforms to the IEEE 802.3ae protocol standards.

Just as 1000BASE-X and 1000BASE-T (Gigabit Ethernet) remained true to the Ethernet model, 10-Gigabit Ethernet continues the natural evolution of Ethernet in speed and distance. Because it is a full-duplex only technology, it does not need the carrier-sensing multiple-access with the CSMA/CD protocol that defines slower, half-duplex Ethernet technologies. In every other respect, 10-Gigabit Ethernet remains true to the original Ethernet model.

### IEEE 802.3ba 100 Gbps Ethernet

IEEE 802.3ba is supported on the Cisco 1-Port 100-Gigabit Ethernet PLIM beginning in Cisco IOS XR 4.0.1.

## MAC Address

A MAC address is a unique 6-byte address that identifies the interface at Layer 2.

## MAC Accounting

The MAC address accounting feature provides accounting information for IP traffic based on the source and destination MAC addresses on LAN interfaces. This feature calculates the total packet and byte counts for a LAN interface that receives or sends IP packets to or from a unique MAC address. It also records a time stamp for the last packet received or sent.

These statistics are used for traffic monitoring, debugging and billing. For example, with this feature you can determine the volume of traffic that is being sent to and/or received from various peers at NAPS/peering points. This feature is currently supported on Ethernet, FastEthernet, and bundle interfaces and supports Cisco Express Forwarding (CEF), distributed CEF (dCEF), flow, and optimum switching.

**Note** A maximum of 512 MAC addresses per trunk interface are supported for MAC address accounting.

## Ethernet MTU

The Ethernet maximum transmission unit (MTU) is the size of the largest frame, minus the 4-byte frame check sequence (FCS), that can be transmitted on the Ethernet network. Every physical network along the destination of a packet can have a different MTU.

Cisco IOS XR software supports two types of frame forwarding processes:

- Fragmentation for IPV4 packets–In this process, IPv4 packets are fragmented as necessary to fit within the MTU of the next-hop physical network.

  **Note** IPv6 does not support fragmentation.

- MTU discovery process determines largest packet size–This process is available for all IPV6 devices, and for originating IPv4 devices. In this process, the originating IP device determines the size of the largest IPv6 or IPV4 packet that can be sent without being fragmented. The largest packet is equal to the smallest MTU of any network between the IP source and the IP destination devices. If a packet is larger than the smallest MTU of all the networks in its path, that packet will be fragmented as necessary. This process ensures that the originating device does not send an IP packet that is too large.

Jumbo frame support is automatically enable for frames that exceed the standard frame size. The default value is 1514 for standard frames and 1518 for 802.1Q tagged frames. These numbers exclude the 4-byte frame check sequence (FCS).

## Flow Control on Ethernet Interfaces

The flow control used on 10-Gigabit Ethernet interfaces consists of periodically sending flow control pause frames. It is fundamentally different from the usual full- and half-duplex flow control used on standard management interfaces. Flow control can be activated or deactivated for ingress traffic only. It is automatically implemented for egress traffic.

## 802.1Q VLAN

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, it is very flexible for user and host management, bandwidth allocation, and resource optimization.

The IEEE's 802.1Q protocol standard addresses the problem of breaking 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.

## VRRP

The Virtual Router Redundancy Protocol (VRRP) eliminates the single point of failure inherent in the static default routed environment. VRRP specifies an election protocol that dynamically assigns responsibility for a virtual router to one of the VPN concentrators on a LAN. The VRRP VPN concentrator controlling the IP addresses associated with a virtual router is called the master, and forwards packets sent to those IP addresses. When the master becomes unavailable, a backup VPN concentrator takes the place of the master.

For more information on VRRP, see the *Implementing VRRP on Cisco IOS XR Software* module of *Cisco IOS XR IP Addresses and Services Configuration Guide.*

## HSRP

Hot Standby Routing Protocol (HSRP) is a proprietary protocol from Cisco. HSRP is a routing protocol that provides backup to a router in the event of failure. Several routers are connected to the same segment of an Ethernet, FDDI, or token-ring network and work together to present the appearance of a single virtual router on the LAN. The routers share the same IP and MAC addresses and therefore, in the event of failure of one router, the hosts on the LAN are able to continue forwarding packets to a consistent IP and MAC address. The transfer of routing responsibilities from one device to another is transparent to the user.

HSRP is designed to support non disruptive switchover of IP traffic in certain circumstances and to allow hosts to appear to use a single router and to maintain connectivity even if the actual first hop router they are using fails. In other words, HSRP protects against the failure of the first hop router when the source host cannot learn the IP address of the first hop router dynamically. Multiple routers participate in HSRP and in concert create the illusion of a single virtual router. HSRP ensures that one and only one of the routers is forwarding packets on behalf of the virtual router. End hosts forward their packets to the virtual router.

The router forwarding packets is known as the *active router*. A standby router is selected to replace the active router should it fail. HSRP provides a mechanism for determining active and standby routers, using the IP addresses on the participating routers. If an active router fails a standby router can take over without a major interruption in the host's connectivity.

HSRP runs on top of User Datagram Protocol (UDP), and uses port number 1985. Routers use their actual IP address as the source address for protocol packets, not the virtual IP address, so that the HSRP routers can identify each other.

For more information on HSRP, see the *Implementing HSRP on Cisco IOS XR Software* module of *Cisco IOS XR IP Addresses and Services Configuration Guide.*

## Duplex Mode on Fast Ethernet Interfaces

Fast Ethernet ports support the duplex transmission type. Full-duplex mode enables the simultaneous data transmission between a sending station and a receiving station, while half-duplex mode enables data transmission in only one direction at a time.

When configuring duplex mode on a Fast Ethernet interface, keep the following in mind:

- If auto-negotiation is enabled on the interface, the default is duplex negotiated.
- If auto-negotiation is disabled on the interface, the default is full-duplex.

**Note** You can configure duplex mode on Fast Ethernet interfaces only. Gigabit Ethernet and 10-Gigabit Ethernet interfaces always run in full-duplex mode.

## Fast Ethernet Interface Speed

You can configure the interface speed on Fast Ethernet interfaces. Keep the following in mind when configuring the speed for a Fast Ethernet interface:

- If auto-negotiation is enabled on an interface, the default is speed negotiated.
- If auto-negotiation is disabled on an interface, the default speed is the maximum speed allowed on the interface.

**Note** Both ends of a link must have the same interface speed. A manually configured interface speed overrides any auto-negotiated speed, which can prevent a link from coming up if the configured interface speed at one end of a link is different from the interface speed on the other end.

## Link Autonegotiation on Ethernet Interfaces

Link autonegotiation ensures that devices that share a link segment are automatically configured with the highest performance mode of interoperation. Use the **negotiation auto** command in interface configuration mode to enable link autonegotiation on an Ethernet interface. On line card Ethernet interfaces, link autonegotiation is disabled by default.

**Note** The **negotiation auto** command is available on Gigabit Ethernet and Fast Ethernet interfaces only.

Table 4 describes the performance of the system for different combinations of the duplex and speed modes. The specified **duplex** command configured with the specified **speed** command produces the resulting system action, provided that you have configured autonegotiation on the interface.

*Table 4 Relationship Between duplex and speed Commands*

| duplex Command | speed Command | Resulting System Action |
|---|---|---|
| **no duplex** | **no speed** | Auto-negotiates both speed and duplex modes. |
| **no duplex** | **speed 1000** | Auto-negotiates for duplex mode and forces 1000 Mbps. |
| **no duplex** | **speed 100** | Auto-negotiates for duplex mode and forces 100 Mbps. |
| **no duplex** | **speed 10** | Auto-negotiates for duplex mode and forces 10 Mbps. |
| **full-duplex** | **no speed** | Forces full duplex and auto-negotiates for speed. |
| **full-duplex** | **speed 1000** | Forces full duplex and 1000 Mbps. |
| **full-duplex** | **speed 100** | Forces full duplex and 100 Mbps. |
| **full-duplex** | **speed 10** | Forces full duplex and 10 Mbps. |
| **half-duplex** | **no speed** | Forces half duplex and auto-negotiates for speed. |
| **half-duplex** | **speed 1000** | Forces half duplex and 1000 Mbps. |
| **half-duplex** | **speed 100** | Forces half duplex and 100 Mbps. |
| **half-duplex** | **speed 10** | Forces half duplex and 10 Mbps. |

## Carrier Delay on Ethernet Interfaces

When enabled on an Ethernet interface, the Carrier Delay feature slows the response of the system to line-up or line-down events. You can configure both Carrier Delay up and Carrier Delay down on an interface at the same time.

Carrier Delay up suppresses short line flaps where the line is down, then comes up, then goes down again. A line that was previously down must be up longer than the duration specified for the **carrier-delay up** command before the system is informed that the interface has come up. All flaps that are shorter than the duration specified for the **carrier-delay up** command are suppressed.

Configuring Carrier Delay up helps to ensure that a line is reasonably stable before the system is informed that the interface is up and ready to forward traffic.

Carrier Delay down suppresses short line flaps where the line is up, then goes down, then comes up again. A line that was previously up must be down longer than the duration specified for the **carrier-delay down** command before the system is informed that the interface has gone down. All flaps that are shorter than the value specified for the **carrier-delay down** command are suppressed.

Configuring Carrier Delay down can be beneficial in suppressing very short link flaps, thereby preventing interface flaps. Alternatively, configuring this feature can be beneficial in allowing other line protection equipment to have enough time to intervene.

# How to Configure Ethernet

This section provides the following configuration procedures:

- Configuring Ethernet Interfaces, page 103

# Configuring Ethernet Interfaces

This section provides the following configuration procedures:

- Configuring Gigabit Ethernet Interfaces, page 103
- Configuring a Fast Ethernet Interface, page 106
- Configuring a L2VPN Ethernet Port, page 110
- Configuring MAC Accounting on an Ethernet Interface, page 108

## Configuring Gigabit Ethernet Interfaces

Use the following procedure to create a basic Gigabit Ethernet or 10-Gigabit Ethernet interface configuration.

### SUMMARY STEPS

1. **show version**
2. **show interfaces** [**GigabitEthernet** | **TenGigE** ] *interface-path-id*
3. **configure**
4. **interface** [**GigabitEthernet** | **TenGigE** ] *interface-path-id*
5. **ipv4 address** *ip-address mask*
6. **flow-control** {**bidirectional** | **egress** | **ingress**}
7. **mtu** *bytes*
8. **mac-address** *value1.value2.value3*
9. **negotiation auto** (on Gigabit Ethernet and Fast Ethernet interfaces only)
10. **no shutdown**
11. **end**
    or
    **commit**
12. **show interfaces** [**GigabitEthernet** | **TenGigE** ] *interface-path-id*

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **show version**<br><br>**Example:**<br>`RP/0/0/CPU0:router# show version` | (Optional) Displays the current software version, and can also be used to confirm that the router recognizes the modular services card. |
| **Step 2** | **show interfaces** [**GigabitEthernet** \| **TenGigE** ] *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router# show interfaces TenGigE 0/1/0/0` | (Optional) Displays the configured interface and checks the status of each interface port.<br><br>Possible interface types for this procedure are:<br>• GigabitEthernet<br>• TenGigE |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure terminal` | Enters global configuration mode. |
| **Step 4** | **interface** [**GigabitEthernet** \| **TenGigE** ] *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface TenGigE 0/1/0/0` | Enters interface configuration mode and specifies the Ethernet interface name and notation *rack/slot/module/port*. Possible interface types for this procedure are:<br>• GigabitEthernet<br>• TenGigE<br>**Note** The example indicates an 8-port 10-Gigabit Ethernet interface in modular services card slot 1. |
| **Step 5** | **ipv4 address** *ip-address mask*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# ipv4 address 172.18.189.38 255.255.255.224` | Assigns an IP address and subnet mask to the interface.<br>• Replace *ip-address* with the primary IPv4 address for the 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. |
| **Step 6** | **flow-control** {**bidirectional**\| **egress** \| **ingress**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# flow control ingress` | (Optional) Enables the sending and processing of flow control pause frames.<br>• **egress**—Enables the sending of flow control pause frames in egress.<br>• **ingress**—Enables the processing of received pause frames on ingress.<br>• **bidirectional**—Enables the sending of flow control pause frames in egress and the processing of received pause frames on ingress. |
| **Step 7** | **mtu** *bytes*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# mtu 1448` | (Optional) Sets the MTU value for the interface.<br>• The default is 1514 bytes for normal frames and 1518 bytes for 802.1Q tagged frames.<br>• The range for Gigabit Ethernet and 10-Gigabit Ethernet mtu values is 64 bytes to 65535 bytes. |

| | Command or Action | Purpose |
|---|---|---|
| Step 8 | **mac-address** *value1.value2.value3*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# mac address 0001.2468.ABCD` | (Optional) Sets the MAC layer address of the Management Ethernet interface.<br>• The values are the high, middle, and low 2 bytes, respectively, of the MAC address in hexadecimal. The range of each 2-byte value is 0 to ffff. |
| Step 9 | **negotiation auto**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# negotiation auto` | (Optional) Enables autonegotiation on a Gigabit Ethernet interface.<br>• Autonegotiation must be explicitly enabled on both ends of the connection, or speed and duplex settings must be configured manually on both ends of the connection.<br>• If autonegotiation is enabled, any speed or duplex settings that you configure manually take precedence.<br>**Note** The **negotiation auto** command is available on Gigabit Ethernet and Fast Ethernet interfaces only. |
| Step 10 | **no shutdown**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# no shutdown` | Removes the shutdown configuration, which forces an interface administratively down. |
| Step 11 | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# end`<br>or<br>`RP/0/0/CPU0:router(config-if)# 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)?`<br>`[cancel]:`<br>– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 12 | **show interfaces** [**GigabitEthernet** \| **TenGigE**] *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router# show interfaces TenGigE 0/3/0/0` | (Optional) Displays statistics for interfaces on the router. |

## What to Do Next

- To configure MAC Accounting on the Ethernet interface, see the "Configuring MAC Accounting on an Ethernet Interface" section later in this module.
- To configure an AC on the Ethernet port for Layer 2 VPN implementation, see the "Configuring a L2VPN Ethernet Port" section later in this module.
- To attach Layer 3 service policies, such as Multiprotocol Label Switching (MPLS) or Quality of Service (QoS), to the Ethernet interface, refer to the appropriate Cisco IOS XR software configuration guide.

## Configuring a Fast Ethernet Interface

Use the following procedure to create a basic Fast Ethernet interface configuration.

**Note** Fast Ethernet is supported on the Cisco XR 12000 Series Router only.

### SUMMARY STEPS

1. **configure**
2. **interface fastethernet** *interface-path-id*
3. **ipv4 address** *ip-address mask*
4. **mtu** *bytes*
5. **duplex full**
6. **speed** *speed*
7. **negotiation auto**
8. **no shutdown**
9. **end**
   or
   **commit**
10. **show interfaces fastethernet** *interface-path-id*

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | `configure`<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure terminal` | Enters global configuration mode. |
| **Step 2** | `interface fastethernet` *`interface-path-id`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface fastethernet 0/0/2/1` | Enters interface configuration mode and specifies the Ethernet interface name and notation *rack/slot/module/port*.<br>• The example indicates the second interface on an 8-port Fast Ethernet SPA in slot 0, and SPA subslot 2. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **`ipv4 address`** *`ip-address mask`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# ipv4 address 172.18.189.38 255.255.255.224` | Assigns an IP address and subnet mask to the interface.<br>• Replace *ip-address* with the primary IPv4 address for the 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. |
| **Step 4** | **`mtu`** *`bytes`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if# mtu 1448` | (Optional) Sets the MTU value for the interface.<br>• The default is 1500 bytes.<br>• The range for Fast Ethernet mtu values is 0 to 10240 bytes. |
| **Step 5** | `duplex full`<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# duplex full` | (Optional) Enables full-duplex mode. Use the **duplex half** command to enable half-duplex mode.<br>• Duplex operation is auto-negotiated by default.<br>• If auto-negotiation is enabled, any user-defined duplex value is ignored. |
| **Step 6** | **`speed`** *`speed`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# speed 100` | (Optional) Sets the speed of the interface. Valid options are 10 Mbps or 100 Mbps.<br>• The default is 100 Mbps.<br>• If auto-negotiation is enabled, any user-defined speed value is ignored. |
| **Step 7** | **`negotiation auto`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# negotiation auto` | (Optional) Enables auto-negotiation on the interface.<br>• Auto-negotiation must be explicitly enabled on both ends of the connection, or speed and duplex settings must be configured manually on both ends of the connection.<br>• If auto-negotiation is enabled, any manually-configured speed or duplex settings take precedence. |
| **Step 8** | **`no shutdown`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# no shutdown` | Removes the shutdown configuration, which removes the forced administrative down on the interface, enabling it to move to an up or down state. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 9** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# end`<br>or<br>`RP/0/0/CPU0:router(config-if)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 10** | **show interfaces fastethernet** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router# show interfaces fastethernet 0/0/1/1` | (Optional) Displays statistics for interfaces on the router. |

## What to Do Next

- To configure an AC on the Fast Ethernet port for Layer 2 VPN implementation, see the "Configuring a L2VPN Ethernet Port" section later in this module.
- To attach Layer 3 service policies, such as Multiprotocol Label Switching (MPLS) or Quality of Service (QoS), to the Fast Ethernet interface, refer to the appropriate Cisco IOS XR software configuration guide.

## Configuring MAC Accounting on an Ethernet Interface

This task explains how to configure MAC accounting on an Ethernet interface. MAC accounting has special **show** commands, which are illustrated in this procedure. Otherwise, the configuration is the same as configuring a basic Ethernet interface, and the steps can be combined in one configuration session. See "Configuring Gigabit Ethernet Interfaces" in this module for information about configuring the other common parameters for Ethernet interfaces.

### SUMMARY STEPS

1. **configure**
2. **interface** [**GigabitEthernet** | **TenGigE** | **fastethernet**] *interface-path-id*
3. **ipv4 address** *ip-address mask*
4. **mac-accounting** {**egress** | **ingress**}

5. **end**
   or
   **commit**
6. **show mac-accounting** *type* **location** *instance*

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | `configure`<br><br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **`interface`** `[`**`GigabitEthernet`** `\|` **`TenGigE`** `\|` **`fastethernet`**`]` *`interface-path-id`*<br><br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# interface TenGigE 0/1/0/0` | Physical interface or virtual interface.<br>**Note** Use the **show interfaces** command to see a list of all interfaces currently configured on the router.<br><br>For more information about the syntax for the router, use the question mark (**?**) online help function. |
| **Step 3** | **`ipv4 address`** *`ip-address mask`*<br><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if)# ipv4 address 172.18.189.38 255.255.255.224` | Assigns an IP address and subnet mask to the interface.<br>• Replace *ip-address* with the primary IPv4 address for the 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. |
| **Step 4** | **`mac-accounting`** `{`**`egress`** `\|` **`ingress`**`}`<br><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if)# mac-accounting egress` | Generates accounting information for IP traffic based on the source and destination MAC addresses on LAN interfaces.<br>• To disable MAC accounting, use the **no** form of this command. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if)# end`<br>or<br>`RP/0/RP0/CPU0:router(config-if)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 6** | **show mac-accounting** *type* **location** *instance*<br><br>**Example:**<br>`RP/0/RP0/CPU0:router# show mac-accounting TenGigE location 0/2/0/4` | Displays MAC accounting statistics for an interface. |

## Configuring a L2VPN Ethernet Port

Use the following procedure to configure an L2VPN Ethernet port.

**Note** The steps in this procedure configure the L2VPN Ethernet port to operate in port mode.

### SUMMARY STEPS

1. **configure**
2. **interface** [**GigabitEthernet** | **TenGigE**] *interface-path-id*
3. **l2transport**
4. **l2protocol** {**cdp** | **pvst** | **stp** | **vtp**} {[**forward** | **tunnel**] [**experimental** *bits*] | **drop**}
5. **end**
   or
   **commit**
6. **show interfaces** [**GigabitEthernet** | **TenGigE**] *interface-path-id*

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure terminal` | Enters global configuration mode. |
| **Step 2** | **interface** [**GigabitEthernet** \| **TenGigE**] *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface TenGigE 0/1/0/0` | Enters interface configuration mode and specifies the Ethernet interface name and notation *rack/slot/module/port*. Possible interface types for this procedure are:<br>• GigabitEthernet<br>• TenGigE |
| **Step 3** | **l2transport**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# l2transport` | Enables Layer 2 transport mode on a port and enter Layer 2 transport configuration mode. |
| **Step 4** | **l2protocol** {**cdp** \| **pvst** \| **stp** \| **vtp**}{[**forward** \| **tunnel**][**experimental** *bits*]\|**drop**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if-l2)# l2protocol stp tunnel` | 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. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if-l2)# end`<br>or<br>`RP/0/0/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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 6** | **show interfaces** [**GigabitEthernet** \| **TenGigE**] *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router# show interfaces TenGigE 0/3/0/0` | (Optional) Displays statistics for interfaces on the router. |

## What to Do Next

To configure a point-to-point pseudowire xconnect on an AC, see the *Implementing MPLS Layer 2 VPNs* module of the *Cisco IOS XR Multiprotocol Label Switching Configuration Guide.*

To attach Layer 2 service policies, such as quality of service (QoS), to the Ethernet interface, refer to the appropriate Cisco IOS XR software configuration guide.

# Configuration Examples for Ethernet

This section provides the following configuration examples:


- Configuring an Ethernet Interface: Example, page 113
- Configuring a Fast Ethernet Interface: Example, page 113
- Configuring MAC-Accounting: Example, page 114
- Configuring a Layer 2 VPN AC: Example, page 114

## Configuring an Ethernet Interface: Example

The following example shows how to configure an interface for a 10-Gigabit Ethernet modular services card:

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# interface TenGigE 0/0/0/1
RP/0/0/CPU0:router(config-if)# ipv4 address 172.18.189.38 255.255.255.224
RP/0/0/CPU0:router(config-if)# flow-control ingress
RP/0/0/CPU0:router(config-if)# mtu 1448
RP/0/0/CPU0:router(config-if)# mac-address 0001.2468.ABCD
RP/0/0/CPU0:router(config-if)# no shutdown
RP/0/0/CPU0:router(config-if)# end
Uncommitted changes found, commit them? [yes]: yes

RP/0/0/CPU0:router# show interfaces TenGigE 0/0/0/1

TenGigE0/0/0/1 is down, line protocol is down
  Hardware is TenGigE, address is 0001.2468.abcd (bia 0001.81a1.6b23)
  Internet address is 172.18.189.38/27
  MTU 1448 bytes, BW 10000000 Kbit
     reliability 0/255, txload Unknown, rxload Unknown
  Encapsulation ARPA,
  Full-duplex, 10000Mb/s, LR
  output flow control is on, input flow control is on
  loopback not set
  ARP type ARPA, ARP timeout 01:00:00
  Last clearing of "show interface" counters never
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     0 packets input, 0 bytes, 0 total input drops
     0 drops for unrecognized upper-level protocol
     Received 0 broadcast packets, 0 multicast packets
              0 runts, 0 giants, 0 throttles, 0 parity
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     0 packets output, 0 bytes, 0 total output drops
     Output 0 broadcast packets, 0 multicast packets
     0 output errors, 0 underruns, 0 applique, 0 resets
     0 output buffer failures, 0 output buffers swapped out
     0 carrier transitions
```

## Configuring a Fast Ethernet Interface: Example

The following example indicates how to configure an interface for a Fast Ethernet SPA:

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# interface fastethernet 0/0/2/0
RP/0/0/CPU0:router(config-if)# ipv4 address 172.30.1.2 255.255.255.224
RP/0/0/CPU0:router(config-if)# duplex full
RP/0/0/CPU0:router(config-if)# mtu 1514
RP/0/0/CPU0:router(config-if)# speed 100
RP/0/0/CPU0:router(config-if)# no shutdown
RP/0/0/CPU0:router(config-if)# end
Uncommitted changes found, commit them? [yes]: yes

RP/0/0/CPU0:router# show interfaces fastethernet 0/0/2/0

FastEthernet0/0/2/0 is up, line protocol is up
  Hardware is FastEthernet, address is 000f.f83b.30c8 (bia 000f.f83b.30c8)
  Internet address is 172.30.1.2/24
  MTU 1514 bytes, BW 1000000 Kbit
```

```
   reliability 255/255, txload 1/255, rxload 1/255
Encapsulation ARPA,
Duplex unknown, 100Mb/s, TX, link type is force-up
output flow control is off, input flow control is off
loopback not set
ARP type ARPA, ARP timeout 04:00:00
Last clearing of "show interface" counters never
5 minute input rate 0 bits/sec, 0 packets/sec
5 minute output rate 0 bits/sec, 0 packets/sec
   0 packets input, 0 bytes, 0 total input drops
   0 drops for unrecognized upper-level protocol
   Received 0 broadcast packets, 0 multicast packets
            0 runts, 0 giants, 0 throttles, 0 parity
   0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
   0 packets output, 0 bytes, 0 total output drops
   Output 0 broadcast packets, 0 multicast packets
   0 output errors, 0 underruns, 0 applique, 0 resets
   0 output buffer failures, 0 output buffers swapped out
   0 carrier transitions
```

## Configuring MAC-Accounting: Example

The following example indicates how to configure MAC-accounting on an Ethernet interface:

```
RP/0/RSP0RP00/CPU0:router# configure
RP/0/RSP0RP00/CPU0:router(config)# interface TenGigE 0/0/0/2
RP/0/RSP0RP00/CPU0:router(config-if)# ipv4 address 172.18.189.38 255.255.255.224
RP/0/RSP0RP00/CPU0:router(config-if)# mac-accounting egress
RP/0/RSP0RP00/CPU0:router(config-if)# commit
RP/0/RSP0RP00/CPU0:router(config-if)# exit
RP/0/RSP00/CPU0:router(config)# exit
```

## Configuring a Layer 2 VPN AC: Example

The following example indicates how to configure a Layer 2 VPN AC on an Ethernet interface:

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# interface TenGigE 0/0/0/2
RP/0/0/CPU0:router(config-if)# l2transport
RP/0/0/CPU0:router(config-if-l2)# l2protocol pvst tunnel
RP/0/0/CPU0:router(config-if-l2)# commit
```

# Where to Go Next

When you have configured an Ethernet interface, you can configure individual VLAN subinterfaces on that Ethernet interface.

For information about modifying Ethernet management interfaces for the shelf controller (SC), route processor (RP), and distributed RP, see the *Advanced Configuration and Modification of the Management Ethernet Interface on Cisco IOS XR Software* module later in this document.

For information about IPv6 see the *Implementing Access Lists and Prefix Lists on Cisco IOS XR Software* module in the *Cisco IOS XR IP Addresses and Services Configuration Guide.*

# Additional References

These sections provide references related to implementing Gigabit, 10-Gigabit, and Fast Ethernet interfaces.

## Related Documents

| Related Topic | Document Title |
|---|---|
| Cisco IOS XR master command reference | *Cisco IOS XR Master Commands List* |
| Cisco IOS XR interface configuration commands | *Cisco IOS XR Interface and Hardware Component Command Reference* |
| Information about user groups and task IDs | *Cisco IOS XR Interface and Hardware Component Command Reference* |

## Standards

| Standards | Title |
|---|---|
| IEEE 802.1ag<br>ITU-T Y.1731 | — |

## MIBs

| MIBs | MIBs Link |
|---|---|
| IEEE CFM MIB | To locate and download MIBs for selected platforms using Cisco IOS XR Software, use the Cisco MIB Locator found at the following URL:<br>http://www.cisco.com/go/mibs |

## RFCs

| 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/support |

# Configuring Ethernet OAM on Cisco IOS XR Software

This module describes the configuration of Ethernet Operations, Administration, and Maintenance (OAM) on the Cisco XR 12000 Series Router.

**Feature History for Configuring Ethernet OAM**

| Release | Modification |
|---|---|
| Release 4.0.0 | Support for the following features was introduced:<br>• Ethernet CFM on AToM core<br>• Ethernet Link OAM |
| Release 4.1.0 | Support for the following features was introduced:<br>• AIS<br>• CFM Y.1731 ITU Carrier Code (ICC)-based MEG ID (MAID) format.<br>• EFD<br>• Ethernet CFM on Layer 2 Tunneling Protocol Version 3 (L2TPv3) core. Up MEPs and MIPs are now supported on Virtual Private Wire Service (VPWS) cross-connects over L2TPv3.<br>• Ethernet SLA |

# Contents


- Prerequisites for Configuring Ethernet OAM, page 118
- Restrictions for Configuring Ethernet OAM, page 118
- Information About Configuring Ethernet OAM, page 119
- How to Configure Ethernet OAM, page 140
- Configuration Examples for Ethernet OAM, page 181
- Where to Go Next, page 199
- Additional References, page 200


# Prerequisites for Configuring Ethernet OAM

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 Ethernet OAM, confirm that at least one of the Gigabit Ethernet line cards supported on the router is installed:

- 8-Port Fast Ethernet SPA
- 2-Port Gigabit Ethernet SPA
- 5-Port Gigabit Ethernet SPA
- 8-Port Gigabit Ethernet SPA
- 10-Port Gigabit Ethernet SPA
- 1-Port 10-Gigabit Ethernet SPA

# Restrictions for Configuring Ethernet OAM

The following functional areas of Ethernet OAM are not supported on the Cisco XR 12000 Series Router in Cisco IOS XR Release 4.1:

- Remote Loopback
- Symbol period thresholds and window for link monitoring
- Unidirectional link-fault detection

# Information About Configuring Ethernet OAM

To configure Ethernet OAM, you should understand the following concepts:

- Ethernet Link OAM, page 119
- Ethernet CFM, page 120
- Ethernet SLA (Y.1731 Performance Monitoring), page 135

## Ethernet Link OAM

Ethernet as a Metro Area Network (MAN) or a Wide Area Network (WAN) technology benefits greatly from the implementation of Operations, Administration and Maintenance (OAM) features. Ethernet link OAM features allow Service Providers to monitor the quality of the connections on a MAN or WAN. Service providers can monitor specific events, and take actions on events. Ethernet link OAM operates on a single, physical link and it can be configured to monitor either side or both sides of that link.

Ethernet link OAM can be configured in the following ways:

- A Link OAM profile can be configured, and this profile can be used to set the parameters for multiple interfaces.
- Link OAM can be configured directly on an interface.

  When an interface is also using a link OAM profile, specific parameters that are set in the profile can be overridden by configuring a different value directly on the interface.

An EOAM profile simplifies the process of configuring EOAM features on multiple interfaces. An Ethernet OAM profile, and all of its features, can be referenced by other interfaces, allowing other interfaces to inherit the features of that Ethernet OAM profile.

Individual Ethernet link OAM features can be configured on individual interfaces without being part of a profile. In these cases, the individually configured features always override the features in the profile.

The preferred method of configuring custom EOAM settings is to create an EOAM profile in Ethernet configuration mode and then attach it to an individual interface or to multiple interfaces.

The following standard Ethernet Link OAM features are supported on the router:

- Neighbor Discovery, page 119
- Link Monitoring, page 120
- MIB Retrieval, page 120
- Miswiring Detection (Cisco-Proprietary), page 120
- SNMP Traps, page 120

### Neighbor Discovery

Neighbor discovery enables each end of a link to learn the OAM capabilities of the other end and establish an OAM peer relationship. Each end also can require that the peer have certain capabilities before it will establish a session. You can configure certain actions to be taken if there is a capabilities conflict or if a discovery process times out, using the **action capabilities-conflict** or **action discovery-timeout** commands.

### Link Monitoring

Link monitoring enables an OAM peer to monitor faults that cause the quality of a link to deteriorate over time. When link monitoring is enabled, an OAM peer can be configured to take action when the configured thresholds are exceeded.

### MIB Retrieval

MIB retrieval enables an OAM peer on one side of an interface to get the MIB variables from the remote side of the link. The MIB variables that are retrieved from the remote OAM peer are READ ONLY.

### Miswiring Detection (Cisco-Proprietary)

Miswiring Detection is a Cisco-proprietary feature that uses the 32-bit vendor field in every Information OAMPDU to identify potential miswiring cases.

### SNMP Traps

SNMP traps can be enabled or disabled on an Ethernet OAM interface.

## Ethernet CFM

Ethernet Connectivity Fault Management (CFM) is a service-level OAM protocol that provides tools for monitoring and troubleshooting end-to-end Ethernet services per VLAN.  This includes proactive connectivity monitoring, fault verification, and fault isolation. CFM uses standard Ethernet frames and can be run on any physical media that is capable of transporting Ethernet service frames. Unlike most other Ethernet protocols which are restricted to a single physical link, CFM frames can transmit across the entire end-to-end Ethernet network.

CFM is defined in two standards:

- IEEE 802.1ag—Defines the core features of the CFM protocol.
- ITU-T Y.1731—Redefines, but maintains compatibility with the features of IEEE 802.1ag, and defines some additional features.

Ethernet CFM on the Cisco XR 12000 Series Router supports the following functions of ITU-T Y.1731:

- ETH-CC, ETH-RDI, ETH-LB, ETH-LT—These are equivalent to the corresponding features defined in IEEE 802.1ag.

**Note** The Linktrace responder procedures defined in IEEE 802.1ag are used rather than the procedures defined in Y.1731; however, these are interoperable.

- ETH-AIS—The reception of ETH-LCK messages is also supported.
- ETH-DM—This is supported with the Ethernet SLA feature. For more information about Ethernet SLA, see the “Ethernet SLA (Y.1731 Performance Monitoring)” section on page 135.

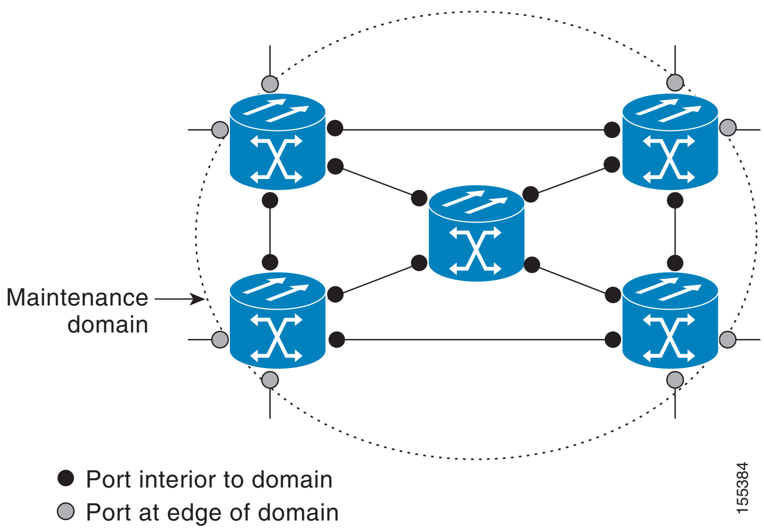
To understand how the CFM maintenance model works, you need to understand the following concepts and features:

- Maintenance Domains, page 121
- Services, page 123
- Maintenance Points, page 123
- CFM Protocol Messages, page 126
- MEP Cross-Check, page 133
- Configurable Logging, page 134
- EFD, page 134

## Maintenance Domains

A *maintenance domain* describes a management space for the purpose of managing and administering a network. A domain is owned and operated by a single entity and defined by the set of interfaces internal to it and at its boundary, as shown in Figure 1.

***Figure 1 CFM Maintenance Domain***




A maintenance domain is defined by the bridge ports that are provisioned within it. Domains are assigned maintenance levels, in the range of 0 to 7, by the administrator. The level of the domain is useful in defining the hierarchical relationships of multiple domains.

CFM maintenance domains allow different organizations to use CFM in the same network, but independently. For example, consider a service provider who offers a service to a customer, and to provide that service, they use two other operators in segments of the network. In this environment, CFM can be used in the following ways:

- The customer can use CFM between their CE devices, to verify and manage connectivity across the whole network.
- The service provider can use CFM between their PE devices, to verify and manage the services they are providing.

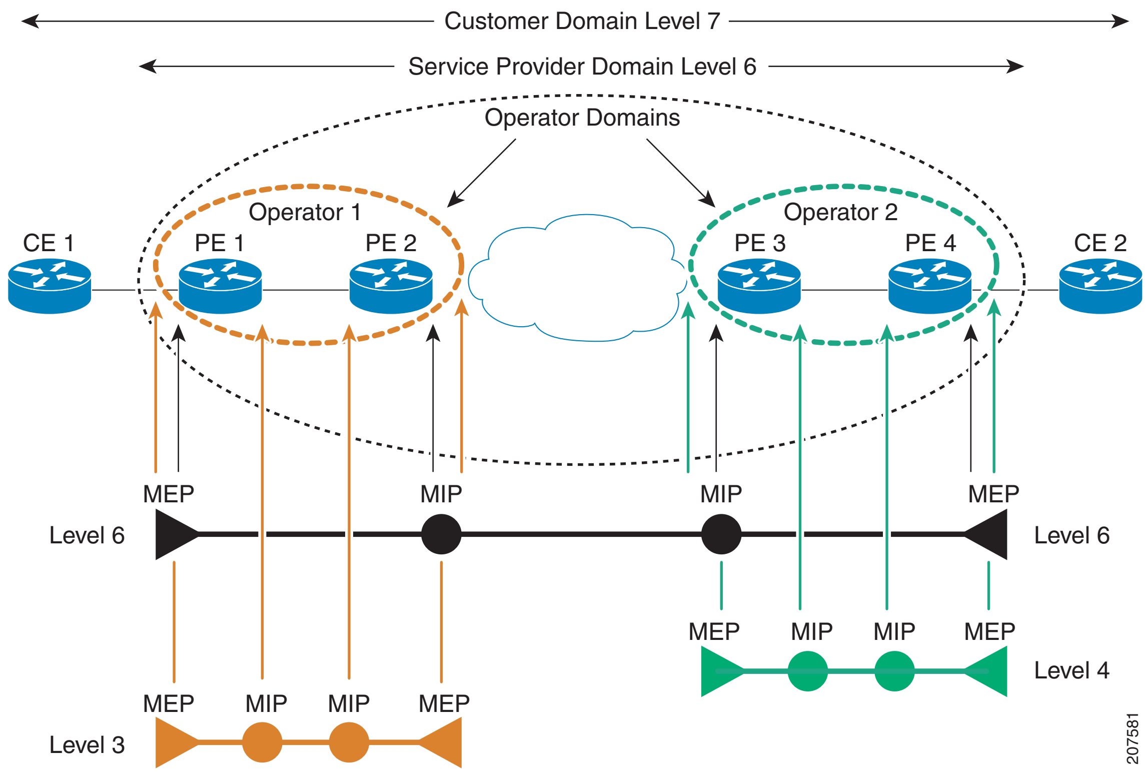
- Each operator can use CFM within their operator network, to verify and manage connectivity within their network.

Each organization uses a different CFM maintenance domain.

Figure 2 shows an example of the different levels of maintenance domains in a network.

**Note** In CFM diagrams, the conventions are that triangles represent MEPs, pointing in the direction that the MEP sends CFM frames, and circles represent MIPs. For more information about MEPs and MIPs, see the "Maintenance Points" section on page 123.

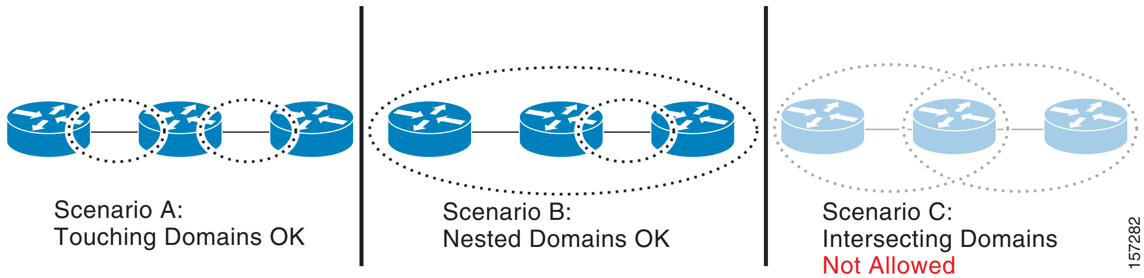
***Figure 2 Different CFM Maintenance Domains Across a Network***




To ensure that the CFM frames for each domain do not interfere with each other, each domain is assigned a maintenance level, between 0 and 7. Where domains are nested, as in this example, the encompassing domain must have a higher level than the domain it encloses. In this case, the domain levels must be negotiated between the organizations involved. The maintenance level is carried in all CFM frames that relate to that domain.

CFM maintenance domains may touch or nest, but cannot intersect. Figure 3 illustrates the supported structure for touching and nested domains, and the unsupported intersection of domains.

***Figure 3 Supported CFM Maintenance Domain Structure***




## Services

A CFM service allows an organization to partition its CFM maintenance domain, according to the connectivity within the network. For example, if the network is divided into a number of virtual LANs (VLANs), a CFM service is created for each of these. CFM can then operate independently in each service. It is important that the CFM services match the network topology, so that CFM frames relating to one service cannot be received in a different service. For example, a service provider may use a separate CFM service for each of their customers, to verify and manage connectivity between that customer's end points.

A CFM service is always associated with the maintenance domain that it operates within, and therefore with that domain's maintenance level. All CFM frames relating to the service carry the maintenance level of the corresponding domain.

**Note** CFM Services are referred to as *Maintenance Associations* in IEEE 802.1ag and as *Maintenance Entity Groups* in ITU-T Y.1731.

## Maintenance Points

A CFM *Maintenance Point* (MP) is an instance of a particular CFM service on a specific interface. CFM only operates on an interface if there is a CFM maintenance point on the interface; otherwise, CFM frames are forwarded transparently through the interface.

A maintenance point is always associated with a particular CFM service, and therefore with a particular maintenance domain at a particular level. Maintenance points generally only process CFM frames at the same level as their associated maintenance domain. Frames in the higher or lower maintenance levels are forwarded transparently. This helps enforce the maintenance domain hierarchy described in the "Maintenance Domains" section on page 121, and ensures that CFM frames for a particular domain cannot leak out beyond the boundary of the domain.

There are two types of MP:

- Maintenance End Points (MEPs)—Created at the edge of the domain. Maintenance end points (MEPs) are members of a particular service within a domain and are responsible for sourcing and sinking CFM frames. They periodically transmit continuity check messages and receive similar messages from other MEPs within their domain. They also transmit traceroute and loopback messages at the request of the administrator. MEPs are responsible for confining CFM messages within the domain.

- Maintenance Intermediate Points (MIPs)—Created in the middle of the domain. MIPs allow CFM frames to be forwarded at either lower, higher, or their own maintenance levels.

### MIP Creation

Unlike MEPs, MIPs are not explicitly configured on each interface. MIPs are created automatically according to the algorithm specified in the CFM 802.1ag standard. The algorithm, in brief, operates as follows for each interface:

- The cross-connect for the interface is found, and all services associated with that cross-connect are considered for MIP auto-creation.
- The level of the highest-level MEP on the interface is found. From among the services considered above, the service in the domain with the lowest level that is higher than the highest MEP level is selected. If there are no MEPs on the interface, the service in the domain with the lowest level is selected.
- The MIP auto-creation configuration (**mip auto-create** command) for the selected service is examined to determine whether a MIP should be created.

**Note** Configuring a MIP auto-creation policy for a service does not guarantee that a MIP will automatically be created for that service. The policy is only considered if that service is selected by the algorithm first.

### MEP and CFM Processing Overview

The boundary of a domain is an interface, rather than a bridge or host. Therefore, MEPs can be sub-divided into two categories:

- Down MEPs—Send CFM frames from the interface where they are configured, and process CFM frames received on that interface. Down MEPs transmit AIS messages upward (toward the cross-connect).
- Up MEPs—Send frames into the bridge relay function, as if they had been received on the interface where the MEP is configured. They process CFM frames that have been received on other interfaces, and have been switched through the bridge relay function as if they are going to be sent out of the interface where the MEP is configured. Up MEPs transmit AIS messages downward (toward the wire). However, AIS packets are only sent when there is a MIP configured on the same interface as the MEP and at the level of the MIP.

**Note** The terms *Down MEP* and *Up MEP* are defined in the IEEE 802.1ag and ITU-T Y.1731 standards, and refer to the direction that CFM frames are sent from the MEP. The terms should not be confused with the operational status of the MEP.

Figure 4 illustrates the monitored areas for Down and Up MEPs.

***Figure 4 Monitored Areas for Down and Up MEPs***

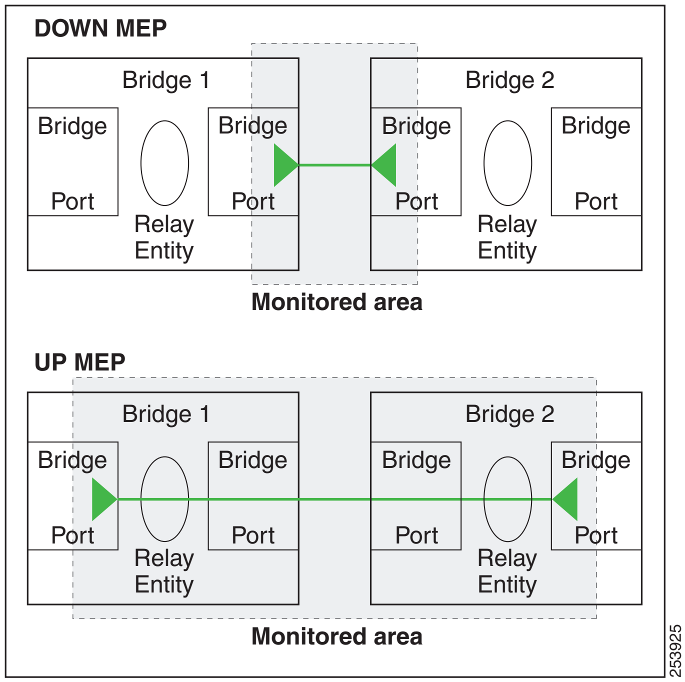
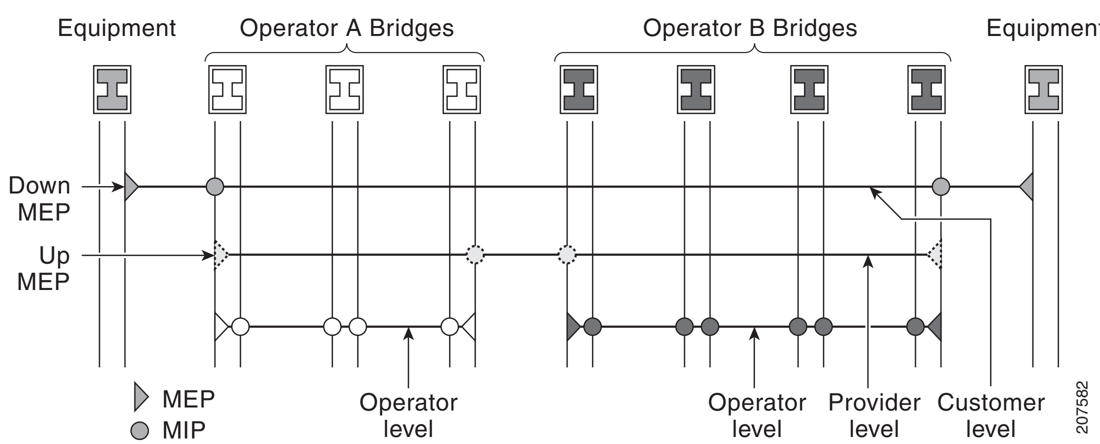



Figure 5 shows maintenance points at different levels. Because domains are allowed to nest but not intersect (see Figure 3), a MEP at a low level always corresponds with a MEP or MIP at a higher level. In addition, only a single MIP is allowed on any interface—this is generally created in the lowest domain that exists at the interface and that does not have a MEP.

***Figure 5 CFM Maintenance Points at Different Levels***




MIPs and Up MEPs can only exist on switched (Layer 2) interfaces, because they send and receive frames from the bridge relay function. Down MEPs can be created on switched (Layer 2) or routed (Layer 3) interfaces.

MEPs continue to operate normally if the interface they are created on is blocked by the Spanning Tree Protocol (STP); that is, CFM frames at the level of the MEP continue to be sent and received, according to the direction of the MEP. MEPs never allow CFM frames at the level of the MEP to be forwarded, so the STP block is maintained.

MIPs also continue to receive CFM frames at their level if the interface is STP blocked, and can respond to any received frames. However, MIPs do not allow CFM frames at the level of the MIP to be forwarded if the interface is blocked.

**Note** A separate set of CFM maintenance levels is created every time a VLAN tag is pushed onto the frame. Therefore, if CFM frames are received on an interface which pushes an additional tag, so as to "tunnel" the frames over part of the network, the CFM frames will not be processed by any MPs within the tunnel, even if they are at the same level. For example, if a CFM MP is created on an interface with an encapsulation that matches a single VLAN tag, any CFM frames that are received at the interface that have two VLAN tags will be forwarded transparently, regardless of the CFM level.

## CFM Protocol Messages

The CFM protocol consists of a number of different message types, with different purposes. All CFM messages use the CFM EtherType, and carry the CFM maintenance level for the domain to which they apply.

This section describes the following CFM messages:

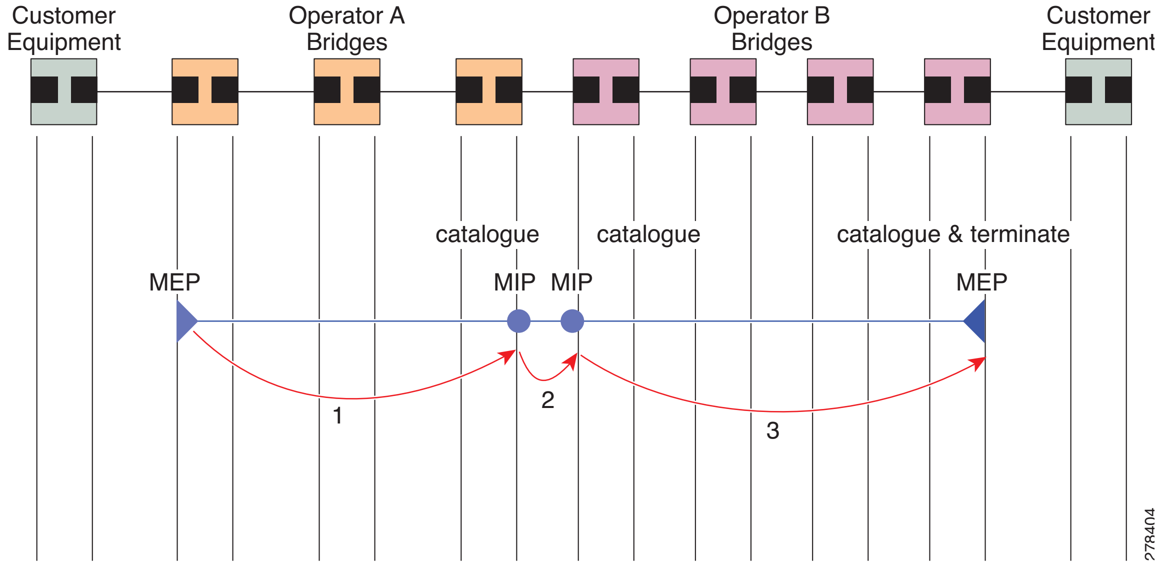

- Continuity Check (IEEE 802.1ag and ITU-T Y.1731), page 126
- Loopback (IEEE 802.1ag and ITU-T Y.1731), page 128
- Linktrace (IEEE 802.1ag and ITU-T Y.1731), page 129
- Exploratory Linktrace (Cisco), page 131
- Alarm Indication Signal (ITU-T Y.1731), page 132
- Delay and Jitter Measurement (ITU-T Y.1731), page 133


### Continuity Check (IEEE 802.1ag and ITU-T Y.1731)

Continuity Check Messages (CCMs) are "heartbeat" messages exchanged periodically between all the MEPs in a service. Each MEP sends out multicast CCMs, and receives CCMs from all the other MEPs in the service—these are referred to as *peer MEPs*. This allows each MEP to discover its peer MEPs, and to verify that there is connectivity between them.

MIPs also receive CCMs. MIPs use the information to build a MAC learning database that is used when responding to Linktrace. For more information about Linktrace, see the "Linktrace (IEEE 802.1ag and ITU-T Y.1731)" section on page 129.

***Figure 6 Continuity Check Message Flow***




All the MEPs in a service must transmit CCMs at the same interval. IEEE 802.1ag defines 7 possible intervals that can be used:

- 3.3ms
- 10ms
- 100ms
- 1s
- 10s
- 1 minute
- 10 minutes

A MEP detects a loss of connectivity with one of its peer MEPs when some number of CCMs have been missed. This occurs when sufficient time has passed during which a certain number of CCMs were expected, given the CCM interval. This number is called the *loss threshold*, and is usually set to 3.

CCM messages carry a variety of information that allows different defects to be detected in the service. This information includes:

- A configured identifier for the domain of the transmitting MEP. This is referred to as the Maintenance Domain Identifier (MDID).
- A configured identifier for the service of the transmitting MEP. This is referred to as the Short MA Name (SMAN). Together, the MDID and the SMAN make up the Maintenance Association Identifier (MAID). The MAID must be configured identically on every MEP in the service.
- A configured numeric identifier for the MEP (the MEP ID). Each MEP in the service must be configured with a different MEP ID.
- A sequence number.
- A Remote Defect Indication (RDI). Each MEP includes this in the CCMs it is sending, if it has detected a defect relating to the CCMs it is receiving. This notifies all the MEPs in the service that a defect has been detected somewhere in the service.
- The interval at which CCMs are being transmitted.

- The status of the interface where the MEP is operating—for example, whether the interface is up, down, STP blocked, and so on.

**Note** The status of the interface (up/down) should not be confused with the direction of any MEPs on the interface (Up MEPs/Down MEPs).

The following defects can be detected from received CCMs:

- Interval mismatch—The CCM interval in the received CCM does not match the interval that the MEP is sending CCMs.
- Level mismatch—A MEP has received a CCM carrying a lower maintenance level than the MEPs own level.
- Loop—A CCM is received with the source MAC address equal to the MAC address of the interface where the MEP is operating.
- Configuration error—A CCM is received with the same MEP ID as the MEP ID configured for the receiving MEP.
- Cross-connect—A CCM is received with an MAID that does not match the locally configured MAID. This generally indicates a VLAN misconfiguration within the network, such that CCMs from one service are leaking into a different service.
- Peer interface down—A CCM is received that indicates the interface on the peer is down.
- Remote defect indication—A CCM is received carrying a remote defect indication.

**Note** This defect does not cause the MEP to include a remote defect indication in the CCMs that it is sending.

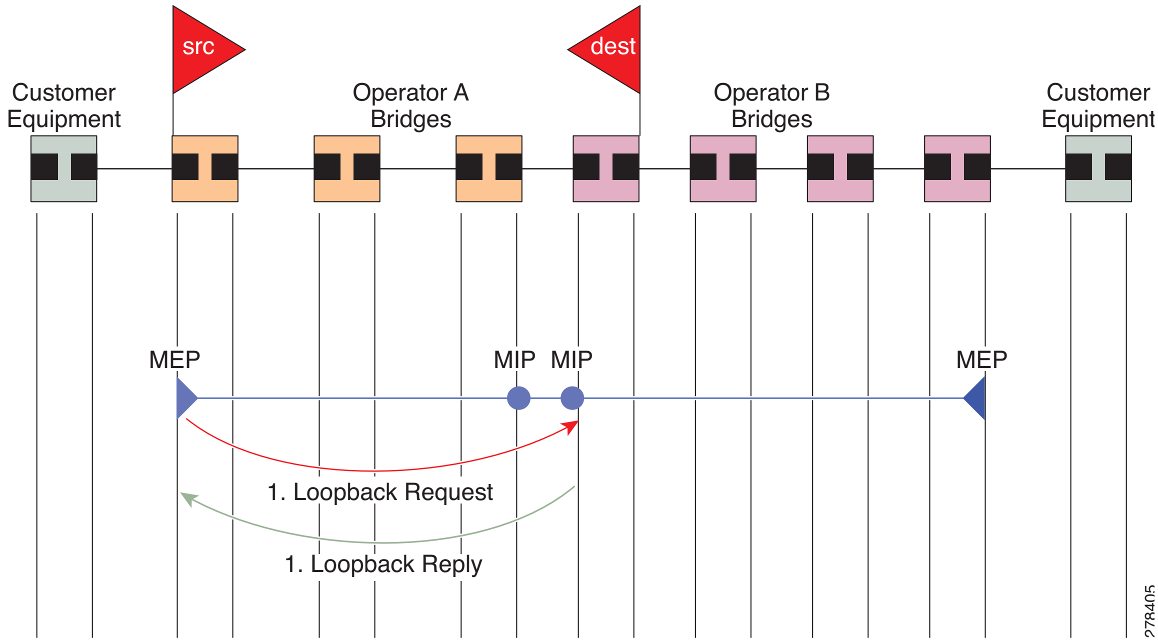
Out-of-sequence CCMs can also be detected by monitoring the sequence number in the received CCMs from each peer MEP. However, this is not considered a CCM defect.

### Loopback (IEEE 802.1ag and ITU-T Y.1731)

Loopback Messages (LBM) and Loopback Replies (LBR) are used to verify connectivity between a local MEP and a particular remote MP. At the request of the administrator, a local MEP sends unicast LBMs to the remote MP. On receiving each LBM, the target maintenance point sends an LBR back to the originating MEP. Loopback indicates whether the destination is reachable or not—it does not allow hop-by-hop discovery of the path. It is similar in concept to an ICMP Echo (ping). Since loopback messages are destined for unicast addresses, they are forwarded like normal data traffic, while observing the maintenance levels. At each device that the loopback reaches, if the outgoing interface is known (in the bridge's forwarding database), then the frame is sent out on that interface. If the outgoing interface is not known, then the message is flooded on all interfaces.

Figure 7 shows an example of CFM loopback message flow between a MEP and MIP.

***Figure 7 Loopback Messages***



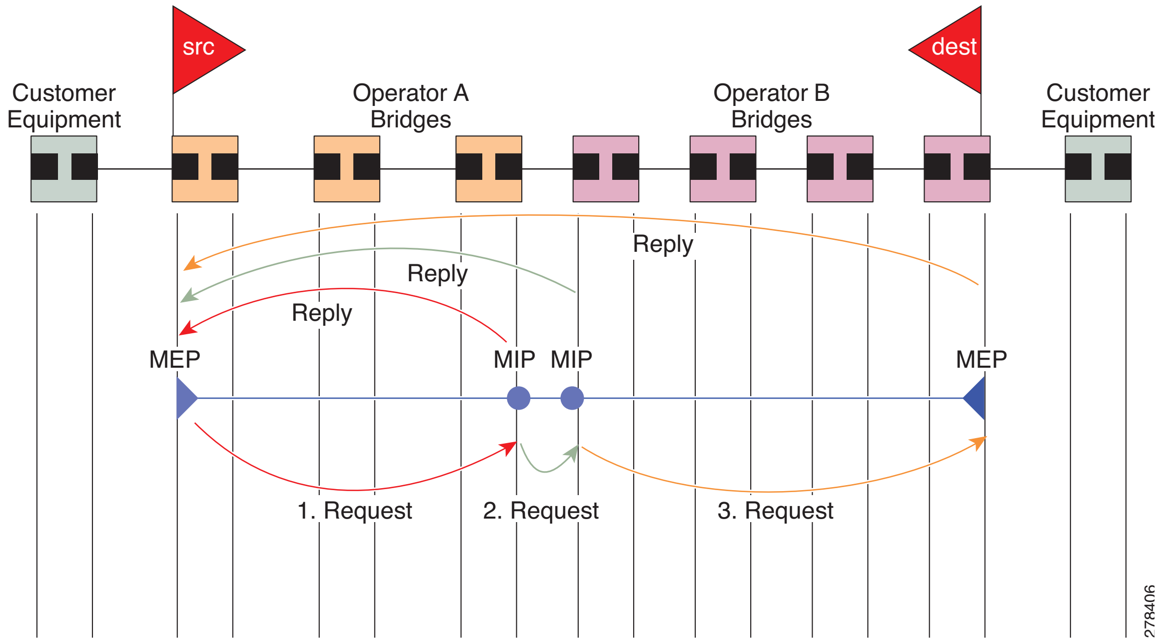

Loopback messages can be padded with user-specified data. This allows data corruption to be detected in the network. They also carry a sequence number which allows for out-of-order frames to be detected.

### Linktrace (IEEE 802.1ag and ITU-T Y.1731)

Linktrace Messages (LTM) and Linktrace Replies (LTR) are used to track the path (hop-by-hop) to a unicast destination MAC address. At the request of the operator, a local MEP sends an LTM. Each hop where there is a maintenance point sends an LTR back to the originating MEP. This allows the administrator to discover connectivity data about the path. It is similar in concept to IP traceroute, although the mechanism is different. In IP traceroute, successive probes are sent, whereas CFM Linktrace uses a single LTM which is forwarded by each MP in the path. LTMs are multicast, and carry the unicast target MAC address as data within the frame. They are intercepted at each hop where there is a maintenance point, and either retransmitted or dropped to discover the unicast path to the target MAC address.

Figure 8 shows an example of CFM linktrace message flow between MEPs and MIPs.

***Figure 8 Linktrace Message Flow***




The linktrace mechanism is designed to provide useful information even after a network failure. This allows it to be used to locate failures, for example after a loss of continuity is detected. To achieve this, each MP maintains a CCM Learning Database. This maps the source MAC address for each received CCM to the interface through which the CCM was received. It is similar to a typical bridge MAC learning database, except that it is based only on CCMs and it times out much more slowly—on the order of days rather than minutes.

**Note** In IEEE 802.1ag, the CCM Learning Database is referred to as the MIP CCM Database. However, it applies to both MIPs and MEPs.

In IEEE 802.1ag, when an MP receives an LTM message, it determines whether to send a reply using the following steps:

1. The target MAC address in the LTM is looked up in the bridge MAC learning table. If the MAC address is known, and therefore the egress interface is known, then an LTR is sent.
2. If the MAC address is not found in the bridge MAC learning table, then it is looked up in the CCM learning database. If it is found, then an LTR is sent.
3. If the MAC address is not found, then no LTR is sent (and the LTM is not forwarded).

If the target MAC has never been seen previously in the network, the linktrace operation will not produce any results.

**Note** IEEE 802.1ag and ITU-T Y.1731 define slightly different linktrace mechanisms. In particular, the use of the CCM learning database and the algorithm described above for responding to LTM messages are specific to IEEE 802.1ag. IEEE 802.1ag also specifies additional information that can be included in LTRs. Regardless of the differences, the two mechanisms are interoperable.

### Exploratory Linktrace (Cisco)

Exploratory Linktrace is a Cisco extension to the standard linktrace mechanism described above. It has two primary purposes:

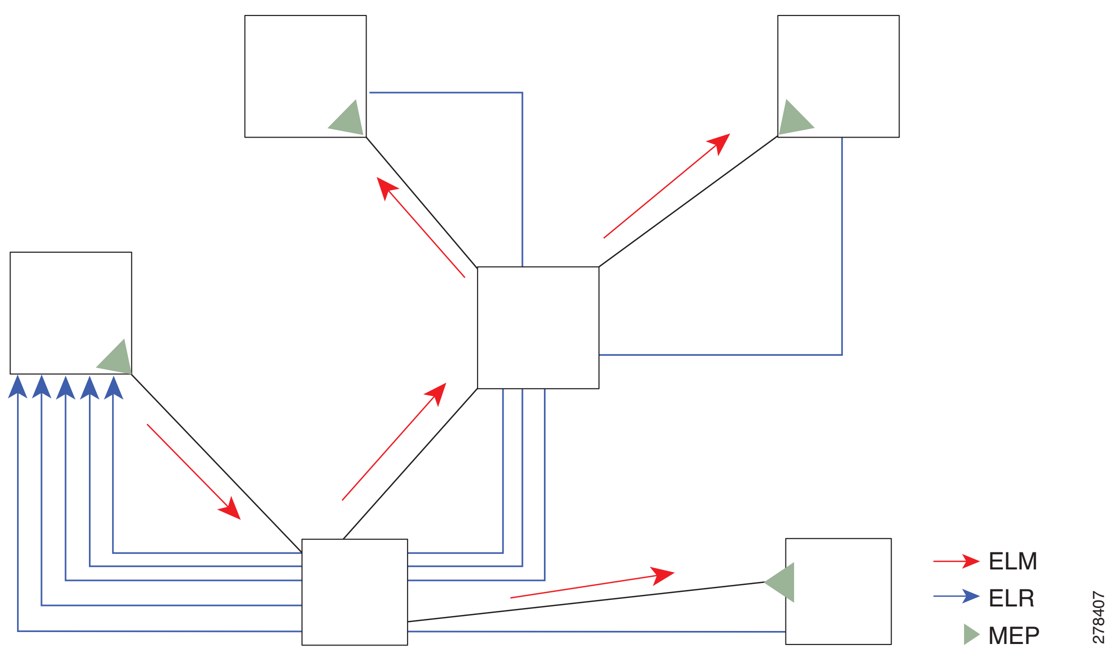
- Provide a mechanism to locate faults in cases where standard linktrace does not work, such as when a MAC address has never been seen previously in the network. For example, if a new MEP has been provisioned but is not working, standard linktrace does not help isolate a problem because no frames will ever have been received from the new MEP. Exploratory Linktrace overcomes this problem.
- Provide a mechanism to map the complete active network topology from a single node. This can only be done currently by examining the topology (for example, the STP blocking state) on each node in the network individually, and manually combining this information to create the overall active topology map. Exploratory linktrace allows this to be done automatically from a single node.

Exploratory Linktrace is implemented using the Vendor Specific Message (VSM) and Vendor Specific Reply (VSR) frames defined in ITU-T Y.1731. These allow vendor-specific extensions to be implemented without degrading interoperability. Exploratory Linktrace can safely be deployed in a network that includes other CFM implementations because those implementations will simply ignore the Exploratory Linktrace messages.

Exploratory Linktrace is initiated at the request of the administrator, and results in the local MEP sending a multicast Exploratory Linktrace message. Each MP in the network that receives the message sends an Exploratory Linktrace reply. MIPs that receive the message also forward it on. The initiating MEP uses all the replies to create a tree of the overall network topology.

Figure 9 show an example of the Exploratory Linktrace message flow between MEPs.

*Figure 9 Exploratory Linktrace Messages and Replies*




To avoid overloading the originating MEP with replies in a large network, responding MPs delay sending their replies for a random amount of time, and that time increases as the size of the network increases.

In a large network, there will be a corresponding large number of replies and the resulting topology map will be equally large. If only a part of the network is of interest, for example, because a problem has already been narrowed down to a small area, then the Exploratory Linktrace can be "directed" to start at a particular MP. Replies will thus only be received from MPs beyond that point in the network. The replies are still sent back to the originating MEP.

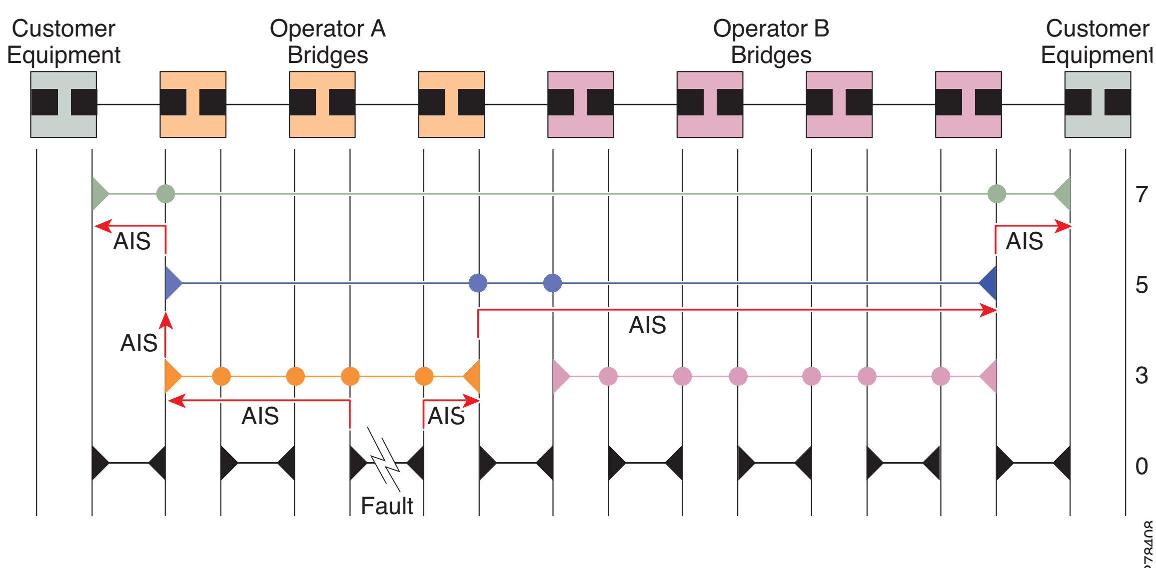
## Alarm Indication Signal (ITU-T Y.1731)

Alarm Indication Signal (AIS) messages are used to rapidly notify MEPs when a fault is detected in the middle of a domain, in an event driven way. MEPs thereby learn of the fault much sooner than if they relied on detecting a loss of continuity, for example, failure to receive some number of consecutive CCMs.

Unlike all other CFM messages, AIS messages are injected into the middle of a domain, and sent outward toward the MEPs at the edge of the domain. Typically, AIS messages are injected by a MEP in a lower level domain. To put it another way, when a MEP sends AIS messages, they are sent in the opposite direction to other CFM messages sent by the MEP, and at a level above the MEP's own level. The AIS messages are received by the MEPs in the higher level domain, not by the peer MEPs in the same domain as the MEP sending the AIS. When a MEP receives an AIS message, it may itself send another AIS message at an even higher level.

Figure 10 show an example of AIS message flow. The maintenance domain levels are numbered at the right side of the diagram.

***Figure 10 AIS Message Flow***




AIS is only applicable in point-to-point networks. In multipoint networks with redundant paths, a failure at a low level does not necessarily result in a failure at a higher level, as the network may reconverge so as to route around the failed link.

AIS messages are typically sent by a MEP. However, AIS messages can also be sent when there is no MEP present, if a fault is detected in the underlying transport, such as if an interface goes down. In ITU-T Y.1731 these are referred to as *server MEPs*.

AIS messages are sent in response to a number of failure conditions:

- Detection of CCM defects, as described "Continuity Check (IEEE 802.1ag and ITU-T Y.1731)" section on page 126.

- Loss of continuity.
- Receipt of AIS messages.
- Failure in the underlying transport, such as when an interface is down.

Received AIS messages can be used to detect and act on failures more quickly than waiting for a loss of continuity. They can also be used to suppress any failure action, on the basis that the failure has already been detected at a lower level and will be handled there. This is described in ITU-T Y.1731; however, the former is often more useful.

### Delay and Jitter Measurement (ITU-T Y.1731)

The router supports one-way and two-way delay measurement using two packet types:

- Delay Measurement Message (DMM)
- Delay Measurement Response (DMR)

These packets are unicast similar to loopback messages. The packets carry timestamps generated by the system time-of-day clock to support more accurate delay measurement, and also support an SLA manageability front-end.

However, unlike loopback messages, these message types can also measure one-way delay and jitter either from destination to source, or from source to destination.

For more information about SLA, see the "Ethernet SLA (Y.1731 Performance Monitoring)" section on page 135.

## MEP Cross-Check

MEP cross-check supports configuration of a set of expected peer MEPs so that errors can be detected when any of the known MEPs are missing, or if any additional peer MEPs are detected that are not in the expected group.

The set of expected MEP IDs in the service is user-defined. Optionally, the corresponding MAC addresses can also be specified. CFM monitors the set of peer MEPs from which CCMs are being received. If no CCMs are ever received from one of the specified expected peer MEPs, or if a loss of continuity is detected, then a cross-check "missing" defect is detected. Similarly, if CCMs are received from a matching MEP ID but with the wrong source MAC address, a cross-check "missing" defect is detected. If CCMs are subsequently received that match the expected MEP ID, and if specified, the expected MAC address, then the defect is cleared.

**Note** While loss of continuity can be detected for any peer MEP, it is only treated as a defect condition if cross-check is configured.

If cross-check is configured and CCMs are received from a peer MEP with a MEP ID that is not expected, this is detected as a cross-check "unexpected" condition. However, this is not treated as a defect condition.

## Configurable Logging

CFM supports logging of various conditions to syslog. Logging can be enabled independently for each service, and when the following conditions occur:

- New peer MEPs are detected, or loss of continuity with a peer MEP occurs.
- Changes to the CCM defect conditions are detected.
- Cross-check “missing” or “unexpected” conditions are detected.
- AIS condition detected (AIS messages received) or cleared (AIS messages no longer received).
- EFD used to shut down an interface, or bring it back up.

## EFD

Ethernet Fault Detection (EFD) is a mechanism that allows Ethernet OAM protocols, such as CFM, to control the “line protocol” state of an interface.

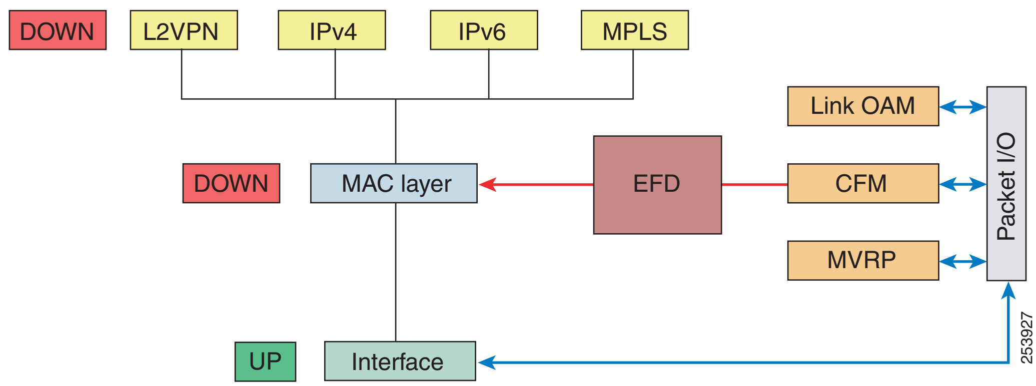
Unlike many other interface types, Ethernet interfaces do not have a line protocol, whose state is independent from that of the interface. For Ethernet interfaces, this role is handled by the physical-layer Ethernet protocol itself, and therefore if the interface is physically up, then it is available and traffic can flow.

EFD changes this to allow CFM to act as the line protocol for Ethernet interfaces. This allows CFM to control the interface state so that if a CFM defect (such as AIS or loss of continuity) is detected with an expected peer MEP, the interface can be shut down. This not only stops any traffic flowing, but also triggers actions in any higher-level protocols to route around the problem. For example, in the case of Layer 2 interfaces, the MAC table would be cleared and MSTP would reconverge. For Layer 3 interfaces, the ARP cache would be cleared and potentially the IGP would reconverge.

**Note** EFD can only be used for down MEPs. When EFD is used to shut down the interface, the CFM frames continue to flow. This allows CFM to detect when the problem has been resolved, and thus bring the interface backup automatically.

Figure 11 shows CFM detection of an error on one of its sessions EFD signaling an error to the corresponding MAC layer for the interface. This triggers the MAC to go to a down state, which further triggers all higher level protocols (Layer 2 pseudowires, IP protocols, and so on) to go down and also trigger a reconvergence where possible. As soon as CFM detects there is no longer any error, it can signal to EFD and all protocols will once again go active.

***Figure 11 CFM Error Detection and EFD Trigger***




# Ethernet SLA (Y.1731 Performance Monitoring)

Customers require their service providers to conform to a Service Level Agreement (SLA). Consequently, service providers must be able to monitor the performance characteristics of their networks. Likewise, customers also want to monitor the performance characteristics of their networks. Cisco provides Y.1731 performance monitoring using the Cisco Ethernet SLA feature.

An SLA defines a set of criteria that guarantees a minimum level of service for customers using a service provider network. The criteria can cover many different areas, including latency, jitter, frame loss, and availability.

The Cisco Ethernet SLA feature conforms to the following standards:

- IEEE 802.1ag
- ITU-T Y.1731

The Cisco Ethernet SLA feature provides the architecture to monitor a network at Layer 2. This architecture provides functions such as collecting, storing, displaying, and analyzing SLA statistics. These SLA statistics can be stored and displayed in various ways, so that statistical analysis can be performed.

Ethernet SLA provides the framework for performing the following major functions of performance monitoring:

- Sending probes consisting of one or more packets to measure performance

  Ethernet SLA provides a flexible mechanism for sending SLA probes to measure performance. Probes can consist of either CFM loopback or CFM delay measurement packets. Options are available to modify how often the packets are sent, and to specify the attributes of the probe packets such as the size and priority.

- Scheduling of operations consisting of periodic probes.

  A flexible mechanism is provided by Ethernet SLA to specify how often each probe should be executed, how long it should last, and when the first probe should start. Probes can be scheduled to run back-to-back to provide continuous measurements, or at a defined interval ranging from once a minute to once a week.

- Collecting and storing results.

  Ethernet SLA provides flexibility to specify which performance parameters should be collected and stored for each measurement probe. Performance parameters include frame delay and jitter (inter-frame delay variation). For each performance parameter, either each individual result can be stored, or the results can be aggregated by storing a counter of the number of results that fall within a particular range. A configurable amount of historical data can also be stored as well as the latest results.

- Analyzing and displaying results.

  Ethernet SLA performs some basic statistical analysis on the collected results, such as calculating the minimum, maximum, mean and standard deviation. It also records whether any of the probe packets were lost or misordered, or if there is any reason why the results may not be a true reflection of the performance (for example if a big jump in the local time-of-day clock was detected during the time when the measurements were being made).

## Ethernet SLA Concepts

To successfully configure the Cisco Ethernet SLA feature, you should understand the following concepts:

- Ethernet SLA Statistic, page 136
- Ethernet SLA Measurement Packet, page 137
- Ethernet SLA Sample, page 137
- Ethernet SLA Probe, page 137
- Ethernet SLA Burst, page 137
- Ethernet SLA Schedule, page 137
- Ethernet SLA Bucket, page 138
- Ethernet SLA Aggregation Bin, page 138
- Ethernet SLA Operation Profile, page 138
- Ethernet SLA Operation, page 138
- Ethernet SLA On-Demand Operation, page 138

### Ethernet SLA Statistic

A *statistic* in Ethernet SLA is a single performance parameter. The following statistics can be measured by Ethernet SLA:

- Round-trip delay
- Round-trip jitter
- One-way delay from source to destination
- One-way jitter from source to destination
- One-way delay from destination to source

- One-way jitter from destination to source

**Note** Not all statistics can be measured by all types of packet. For example, one-way statistics cannot be measured when using CFM loopback packets.

### Ethernet SLA Measurement Packet

An Ethernet SLA *measurement packet* is a single protocol message and corresponding reply that is sent on the network for the purpose of making SLA measurements. The following types of measurement packet are supported:

- CFM Delay Measurement (Y.1731 DMM/DMR packets)—CFM delay measurement packets contain timestamps within the packet data that can be used for accurate measurement of frame delay and jitter. These packets can be used to measure round-trip or one-way statistics; however, the size of the DMM/DMR packets cannot be modified.
- CFM loopback (LBM/LBR)—CFM loopback packets are less accurate, but can be used if the peer device does not support DMM/DMR packets. Only round-trip statistics can be measured because these packets do not contain timestamps. However, loopback packets can be padded, so measurements can be made using frames of a specific size.

### Ethernet SLA Sample

A *sample* is a single result—a number—that relates to a given statistic. For some statistics such as round-trip delay, a sample can be measured using a single measurement packet. For other statistics such as jitter, obtaining a sample requires two measurement packets.

### Ethernet SLA Probe

A *probe* is a sequence of measurement packets used to gather SLA samples for a specific set of statistics. The measurement packets in a probe are of a specific type (for example, CFM delay measurement or CFM loopback) and have specific attributes, such as the frame size and priority.

**Note** A single probe can collect data for different statistics at the same time, using the same measurement packets (for example, one-way delay and round-trip jitter).

### Ethernet SLA Burst

Within a probe, measurement packets can either be sent individually, or in bursts. A *burst* contains two or more packets sent within a short interval apart. Each burst can last up to one minute, and bursts can follow each other immediately to provide continuous measurement within the probe.

For statistics that require two measurement packets for each sample (such as jitter), samples are only calculated based on measurement packets in the same burst. For all statistics, it is more efficient to use bursts than to send individual packets.

### Ethernet SLA Schedule

An Ethernet SLA *schedule* describes how often probes are sent, how long each probe lasts, and at what time the first probe starts.

### Ethernet SLA Bucket

For a particular statistic, a *bucket* is a collection of results that were gathered during a particular period of time. All of the samples for measurements that were initiated during the period of time represented by a bucket are stored in that bucket. Buckets allow results from different periods of time to be compared (for example, peak traffic to off-peak traffic).

By default, a separate bucket is created for each probe; that is, the bucket represents the period of time starting at the same time as the probe started, and continuing for the duration of the probe. The bucket will therefore contain all the results relating to measurements made by that probe.

### Ethernet SLA Aggregation Bin

Rather than storing each sample separately within a bucket, an alternative is to aggregate the samples into bins. An *aggregation bin* is a range of sample values, and contains a counter of the number of samples that were received that fall within that range. The set of bins forms a histogram. When aggregation is enabled, each bucket contains a separate set of bins. See Figure 12 on page 196.

### Ethernet SLA Operation Profile

An *operation profile* is a configuration entity that defines the following aspects of an operation:

- What packet types to send and in what quantities (probe and burst configuration)
- What statistics to measure, and how to aggregate them
- When to schedule the probes

An operation profile by itself does not cause any packets to be sent or statistics collected, but is used to create operation instances.

### Ethernet SLA Operation

An *operation* is an instance of a given operation profile that is actively collecting performance data. Operation instances are created by associating an operation profile with a given source (an interface and MEP) and with a given destination (a MEP ID or MAC address). Operation instances exist for as long as the configuration is applied, and they run for an indefinite duration on an ongoing basis.

### Ethernet SLA On-Demand Operation

An *on-demand operation* is a method of Ethernet SLA operation that can be run on an as-needed basis for a specific and finite period of time. This can be useful in situations such as when you are starting a new service or modifying the parameters for a service to verify the impact of the changes, or if you want to run a more detailed probe when a problem is detected by an ongoing scheduled operation.

On-demand operations do not use profiles and have a finite duration. The statistics that are collected are discarded after a finite time after the operation completes (two weeks), or when you manually clear them.

On-demand operations are not persistent so they are lost during certain events such as a card reload or Minimal Disruptive Restart (MDR).

## Statistics Measurement and Ethernet SLA Operations Overview

Ethernet SLA statistics measurement for network performance is performed by sending packets and storing data metrics such as:

- Round-trip delay time—The time for a packet to travel from source to destination and back to source again.
- Round-trip jitter—The variance in round-trip delay time (latency).
- One-way delay and jitter—The router also supports measurement of one-way delay or jitter from source to destination, or from destination to source.

In addition to these metrics, the following statistics are also kept for SLA probe packets:

- Packet loss count
- Packet corruption event
- Out-of-order event

Counters for packet loss, corruption and out-of-order packets are kept for each bucket, and in each case, a percentage of the total number of samples for that bucket is reported (for example, 4% packet corruption). For delay and jitter statistics, the minimum, maximum, mean and standard deviation for the whole bucket are reported, as well as the individual samples or aggregated bins.

When aggregation is enabled using the **aggregate** command, bins are created to store a count of the samples that fall within a certain value range, which is set by the **width** keyword. Only a counter of the number of results that fall within the range for each bin is stored. This uses less memory than storing individual results. When aggregation is not used, each sample is stored separately, which can provide a more accurate statistics analysis for the operation, but it is highly memory-intensive due to the independent storage of each sample.

A bucket represents a time period during which statistics are collected. All the results received during that time period are recorded in the corresponding bucket. If aggregation is enabled, each bucket has its own set of bins and counters, and only results relating to the measurements initiated during the time period represented by the bucket are included in those counters.

By default, there is a separate bucket for each probe. The time period is determined by how long the probe lasts (configured by the **probe**, **send (SLA)**, and **schedule (SLA)** commands).You can modify the size of buckets so that you can have more buckets per probe or fewer buckets per probe (less buckets allows the results from multiple probes to be included in the same bucket). Changing the size of the buckets for a given metric clears all stored data for that metric. All existing buckets are deleted and new buckets are created.

Scheduled SLA operation profiles run indefinitely, according to a configured schedule, and the statistics that are collected are stored in a rolling buffer, where data in the oldest bucket is discarded when a new bucket needs to be recorded.

## Configuration Overview of Scheduled Ethernet SLA Operations

When you configure a scheduled Ethernet SLA operation, you perform the following basic steps:

1. Configure global profiles to define how packets are sent in each probe, how the probes are scheduled, and how the results are stored.
2. Configure operations from a specific local MEP to a specific peer MEP using these profiles.

**Note** Certain Ethernet SLA configurations use large amounts of memory which can affect the performance of other features on the system. For more information, see the "Configuring Ethernet SLA" section on page 169.

# How to Configure Ethernet OAM

This section provides the following configuration procedures:


- Configuring Ethernet Link OAM, page 140
- Configuring Ethernet CFM, page 149
- Configuring Ethernet SLA, page 169


## Configuring Ethernet Link OAM

Custom EOAM settings can be configured and shared on multiple interfaces by creating an EOAM profile in Ethernet configuration mode and then attaching the profile to individual interfaces. The profile configuration does not take effect until the profile is attached to an interface. After an EOAM profile is attached to an interface, individual EOAM features can be configured separately on the interface to override the profile settings when desired.

This section describes how to configure an EOAM profile and attach it to an interface in the following procedures:


- Configuring an Ethernet OAM Profile, page 140
- Attaching an Ethernet OAM Profile to an Interface, page 145
- Configuring Ethernet OAM at an Interface and Overriding the Profile Configuration, page 146
- Verifying the Ethernet OAM Configuration, page 148


### Configuring an Ethernet OAM Profile

Perform the following steps to configure an Ethernet OAM profile.

#### SUMMARY STEPS

1. **configure**
2. **ethernet oam profile** *profile-name*
3. **link-monitor**
4. **frame window** *window*
5. **frame threshold low** *threshold*
6. **frame-period window** *window*
7. **frame-period threshold low** *threshold*
8. **frame-seconds window** *window*

9. **frame-seconds threshold low** *threshold*
10. **exit**
11. **mib-retrieval**
12. **connection timeout** *seconds*
13. **hello-interval** {**100ms** | **1s**}
14. **mode** {**active** | **passive**}
15. **require-remote mode** {**active** | **passive**}
16. **require-remote link-monitoring**
17. **require-remote mib-retrieval**
18. **action capabilities-conflict** {**disable** | **efd** | **error-disable-interface**}
19. **action critical-event** {**disable** | **error-disable-interface**}
20. **action discovery-timeout** {**disable** | **efd** | **error-disable-interface** }
21. **action dying-gasp** {**disable** | **error-disable-interface**}
22. **action high-threshold** {**error-disable-interface** | **log**}
23. **action remote-loopback disable**
24. **action session-down** {**disable** | **efd** | **error-disable-interface**}
25. **action session-up disable**
26. **action uni-directional link-fault** {**disable** | **efd** | **error-disable-interface**}
27. **action wiring-conflict** {**disable** | **efd** | **log**}
28. **commit**
29. **end**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| Step 1 | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure terminal` | Enters global configuration mode. |
| Step 2 | **ethernet oam profile** *profile-name*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# ethernet oam profile Profile_1` | Creates a new Ethernet Operations, Administration and Maintenance (OAM) profile and enters Ethernet OAM configuration mode. |
| Step 3 | **link-monitor**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-eoam)# link-monitor` | Enters the Ethernet OAM link monitor configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| Step 4 | **frame window** *window*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-eoam-lm)# frame window 60` | (Optional) Configures the frame window size (in milliseconds) of an OAM frame error event.<br>The range is 1000 to 60000.<br>The default value is 1000. |
| Step 5 | **frame threshold low** *threshold* **high** *threshold*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-eoam-lm)# frame threshold low 10000000 high 60000000` | (Optional) Configures the thresholds (in symbols) that triggers an Ethernet OAM frame error event. The high threshold is optional and is configurable only in conjunction with the low threshold.<br>The range is 0 to 60000000.<br>The default low threshold is 1. |
| Step 6 | **frame-period window** *window*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-eoam-lm)# frame-period window 60000` | (Optional) Configures the window size (in milliseconds) for an Ethernet OAM frame-period error event.<br>The range is 100 to 60000.<br>The default value is 1000. |
| Step 7 | **frame-period threshold low** *threshold* **high** *threshold*<br><br>`RP/0/0/CPU0:router(config-eoam-lm)# frame-period threshold low 100 high 1000000` | (Optional) Configures the thresholds (in frames) that trigger an Ethernet OAM frame-period error event. The high threshold is optional and is configurable only in conjunction with the low threshold.<br>The range is 0 to 1000000.<br>The default low threshold is 60000. |
| Step 8 | **frame-seconds window** *window*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-eoam-lm)# frame-seconds window 900000` | (Optional) Configures the window size (in milliseconds) for the OAM frame-seconds error event.<br>The range is 10000 to 900000.<br>The default value is 6000. |
| Step 9 | **frame-seconds threshold low** *threshold* **high** *threshold*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-eoam-lm)# frame-seconds threshold 3 threshold 900` | (Optional) Configures the thresholds (in seconds) that trigger a frame-seconds error event. The high threshold value can be configured only in conjunction with the low threshold value.<br>The range is 1 to 900<br>The default value is 1. |
| Step 10 | **exit**<br><br>**Example:**<br>`RP/0//CPU0:router(config-eoam-lm)# exit` | Exits back to Ethernet OAM mode. |
| Step 11 | **mib-retrieval**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-eoam)# mib-retrieval` | Enables MIB retrieval in an Ethernet OAM profile or on an Ethernet OAM interface. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 12** | **connection timeout** *seconds*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-eoam)# connection timeout 30` | Configures the timeout value (in seconds) for an Ethernet OAM session.<br>The range is 2 to 30.<br>The default value is 5. |
| **Step 13** | **hello-interval** {**100ms**\|**1s**}<br><br>Example:<br>`RP/0/0/CPU0:router(config-eoam)# hello-interval 100ms` | Configures the time interval between hello packets for an Ethernet OAM session. The default is 1 second (**1s**). |
| **Step 14** | **mode** {**active**\|**passive**}<br><br>Example:<br>`RP/0/0/CPU0:router(config-eoam)# mode passive` | Configures the Ethernet OAM mode. The default is active. |
| **Step 15** | **require-remote mode** {**active**\|**passive**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-eoam)# require-remote mode active` | Requires that active mode or passive mode is configured on the remote end before the OAM session becomes active. |
| **Step 16** | **require-remote link-monitoring**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-eoam)# require-remote link-monitoring` | Requires that link-monitoring is configured on the remote end before the OAM session becomes active. |
| **Step 17** | **require-remote mib-retrieval**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-eoam)# require-remote mib-retrieval` | Requires that MIB-retrieval is configured on the remote end before the OAM session becomes active. |
| **Step 18** | **action capabilities-conflict** {**disable** \| **efd** \| **error-disable-interface**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-eoam)# action capabilities-conflict efd` | Specifies the action that is taken on an interface when a capabilities-conflict event occurs. The default action is to create a syslog entry.<br>**Note** If you change the default, the **log** keyword option is available in Interface Ethernet OAM configuration mode to override the profile setting and log the event for the interface when it occurs. |
| **Step 19** | **action critical-event** {**disable** \| **error-disable-interface**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-eoam)# action critical-event error-disable-interface` | Specifies the action that is taken on an interface when a critical-event notification is received from the remote Ethernet OAM peer. The default action is to create a syslog entry.<br>**Note** If you change the default, the **log** keyword option is available in Interface Ethernet OAM configuration mode to override the profile setting and log the event for the interface when it occurs. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 20** | **action discovery-timeout** {**disable** \| **efd** \| **error-disable-interface**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-eoam)# action discovery-timeout efd` | Specifies the action that is taken on an interface when a connection timeout occurs. The default action is to create a syslog entry.<br><br>**Note** If you change the default, the **log** keyword option is available in Interface Ethernet OAM configuration mode to override the profile setting and log the event for the interface when it occurs. |
| **Step 21** | **action dying-gasp** {**disable** \| **error-disable-interface**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-eoam)# action dying-gasp error-disable-interface` | Specifies the action that is taken on an interface when a dying-gasp notification is received from the remote Ethernet OAM peer. The default action is to create a syslog entry.<br><br>**Note** If you change the default, the **log** keyword option is available in Interface Ethernet OAM configuration mode to override the profile setting and log the event for the interface when it occurs. |
| **Step 22** | **action high-threshold** {**error-disable-interface** \| **log**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-eoam)# action high-threshold error-disable-interface` | Specifies the action that is taken on an interface when a high threshold is exceeded. The default is to take no action when a high threshold is exceeded.<br><br>**Note** If you change the default, the **disable** keyword option is available in Interface Ethernet OAM configuration mode to override the profile setting and take no action at the interface when the event occurs. |
| **Step 23** | **action remote-loopback disable**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-eoam)# action remote-loopback disable` | Specifies that no action is taken on an interface when a remote-loopback event occurs. The default action is to create a syslog entry.<br><br>**Note** If you change the default, the **log** keyword option is available in Interface Ethernet OAM configuration mode to override the profile setting and log the event for the interface when it occurs. |
| **Step 24** | **action session-down** {**disable** \| **efd** \| **error-disable-interface**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-eoam)# action session-down efd` | Specifies the action that is taken on an interface when an Ethernet OAM session goes down.<br><br>**Note** If you change the default, the **log** keyword option is available in Interface Ethernet OAM configuration mode to override the profile setting and log the event for the interface when it occurs. |
| **Step 25** | **action session-up disable**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-eoam)# action session-up disable` | Specifies that no action is taken on an interface when an Ethernet OAM session is established. The default action is to create a syslog entry.<br><br>**Note** If you change the default, the **log** keyword option is available in Interface Ethernet OAM configuration mode to override the profile setting and log the event for the interface when it occurs. |

| | Command or Action | Purpose |
|---|---|---|
| Step 26 | **action uni-directional link-fault** {**disable** \| **efd** \| **error-disable-interface**} | Specifies the action that is taken on an interface when a link-fault notification is received from the remote Ethernet OAM peer. The default action is to create a syslog entry.<br>**Note** If you change the default, the **log** keyword option is available in Interface Ethernet OAM configuration mode to override the profile setting and log the event for the interface when it occurs.<br>**Note** |
| Step 27 | **action wiring-conflict** {**disable** \| **efd** \| **log**}<br>**Example:**<br>`RP/0/0/CPU0:router(config-eoam)# action session-down efd` | Specifies the action that is taken on an interface when a wiring-conflict event occurs. The default is to put the interface into error-disable state.<br>**Note** If you change the default, the **error-disable-interface** keyword option is available in Interface Ethernet OAM configuration mode to override the profile setting and put the interface into error-disable state when the event occurs. |
| Step 28 | **commit**<br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# commit` | Saves the configuration changes to the running configuration file and remains within the configuration session. |
| Step 29 | **end**<br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# end` | Ends the configuration session and exits to the EXEC mode. |

## Attaching an Ethernet OAM Profile to an Interface

Perform the following steps to attach an Ethernet OAM profile to an interface:

### SUMMARY STEPS

1. **configure**
2. **interface** [**FastEthernet** | **GigabitEthernet** | **TenGigE**] *interface-path-id*
3. **ethernet oam**
4. **profile** *profile-name*
5. **commit**
6. **end**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure terminal` | Enters global configuration mode. |
| **Step 2** | **interface** [**FastEthernet** \| **GigabitEthernet** \| **TenGigE**] *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface TenGigE 0/1/0/0` | Enters interface configuration mode and specifies the Ethernet interface name and notation *rack/slot/module/port*.<br><br>**Note** The example indicates an 8-port 10-Gigabit Ethernet interface in modular services card slot 1. |
| **Step 3** | **ethernet oam**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# ethernet oam` | Enables Ethernet OAM and enters interface Ethernet OAM configuration mode. |
| **Step 4** | **profile** *profile-name*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if-eoam)# profile Profile_1` | Attaches the specified Ethernet OAM profile (*profile-name*), and all of its configuration, to the interface. |
| **Step 5** | **commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# commit` | Saves the configuration changes to the running configuration file and remains within the configuration session. |
| **Step 6** | **end**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# end` | Ends the configuration session and exits to the EXEC mode. |

## Configuring Ethernet OAM at an Interface and Overriding the Profile Configuration

Using an EOAM profile is an efficient way of configuring multiple interfaces with a common EOAM configuration. However, if you want to use a profile but also change the behavior of certain functions for a particular interface, then you can override the profile configuration. To override certain profile settings that are applied to an interface, you can configure that command in interface Ethernet OAM configuration mode to change the behavior for that interface.

In some cases, only certain keyword options are available in interface Ethernet OAM configuration due to the default settings for the command. For example, without any configuration of the **action** commands, several forms of the command have a default behavior of creating a syslog entry when a profile is created and applied to an interface. Therefore, the **log** keyword is not available in Ethernet OAM configuration for these commands in the profile because it is the default behavior. However, the **log** keyword is available in Interface Ethernet OAM configuration if the default is changed in the profile configuration so you can retain the action of creating a syslog entry for a particular interface.

To see all of the default Ethernet OAM configuration settings, see the "Verifying the Ethernet OAM Configuration" section on page 148.

To configure Ethernet OAM settings at an interface and override the profile configuration, perform the following steps:

## SUMMARY STEPS

1. **configure**
2. **interface** [**FastEthernet** | **GigabitEthernet** | **TenGigE**] *interface-path-id*
3. **ethernet oam**
4. *interface-Ethernet-OAM-command*
5. **commit**
6. **end**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **`configure`**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure terminal` | Enters global configuration mode. |
| **Step 2** | **`interface`** [**`FastEthernet`** \| **`GigabitEthernet`** \| **`TenGigE`**] *`interface-path-id`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface TenGigE 0/1/0/0` | Enters interface configuration mode and specifies the Ethernet interface name and notation *rack/slot/module/port*.<br><br>**Note** The example indicates an 8-port 10-Gigabit Ethernet interface in modular services card slot 1. |
| **Step 3** | **`ethernet oam`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# ethernet oam` | Enables Ethernet OAM and enters interface Ethernet OAM configuration mode. |
| **Step 4** | *`interface-Ethernet-OAM-command`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if-eoam)# action capabilities-conflict error-disable-interface` | Configures a setting for an Ethernet OAM configuration command and overrides the setting for the profile configuration, where *interface-Ethernet-OAM-command* is one of the supported commands on the platform in interface Ethernet OAM configuration mode. |
| **Step 5** | **`commit`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# commit` | Saves the configuration changes to the running configuration file and remains within the configuration session. |
| **Step 6** | **`end`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# end` | Ends the configuration session and exits to the EXEC mode. |

## Verifying the Ethernet OAM Configuration

Use the **show ethernet oam configuration** command to display the values for the Ethernet OAM configuration for a particular interface, or for all interfaces. The following example shows the default values for Ethernet OAM settings:

**Note** Some of these settings are not supported on certain platforms, but the defaults are still reported. On the Cisco XR 12000 Series Router, the following areas are unsupported:

- Remote loopback
- Symbol period window
- Symbol period thresholds
- Uni-directional link-fault detection

```
RP/0/0/CPU0:router# show ethernet oam configuration
Thu Aug  5 22:07:06.870 DST
GigabitEthernet0/4/0/0:
  Hello interval:                                            1s
  Link monitoring enabled:                                    Y
  Remote loopback enabled:                                    N
  Mib retrieval enabled:                                      N
  Uni-directional link-fault detection enabled:               N
  Configured mode:                                       Active
  Connection timeout:                                         5
  Symbol period window:                                       0
  Symbol period low threshold:                                1
  Symbol period high threshold:                            None
  Frame window:                                            1000
  Frame low threshold:                                        1
  Frame high threshold:                                    None
  Frame period window:                                     1000
  Frame period low threshold:                                 1
  Frame period high threshold:                             None
  Frame seconds window:                                   60000
  Frame seconds low threshold:                                1
  Frame seconds high threshold:                            None
  High threshold action:                                   None
  Link fault action:                                        Log
  Dying gasp action:                                        Log
  Critical event action:                                    Log
  Discovery timeout action:                                 Log
  Capabilities conflict action:                             Log
  Wiring conflict action:                         Error-Disable
  Session up action:                                        Log
  Session down action:                                      Log
  Remote loopback action:                                   Log
  Require remote mode:                                   Ignore
  Require remote MIB retrieval:                               N
  Require remote loopback support:                            N
  Require remote link monitoring:                             N
```

# Configuring Ethernet CFM

To configure Ethernet CFM, perform the following tasks:

- Configuring a CFM Maintenance Domain, page 149 (required)
- Configuring Services for a CFM Maintenance Domain, page 150 (required)
- Enabling and Configuring Continuity Check for a CFM Service, page 152 (optional)
- Configuring Automatic MIP Creation for a CFM Service, page 154 (optional)
- Configuring Cross-Check on a MEP for a CFM Service, page 156 (optional)
- Configuring Other Options for a CFM Service, page 158 (optional)
- Configuring CFM MEPs, page 160 (required)
- Configuring Y.1731 AIS, page 162 (optional)
- Configuring EFD for a CFM Service, page 166 (optional)
- Verifying the CFM Configuration, page 168
- Troubleshooting Tips, page 168

## Configuring a CFM Maintenance Domain

To configure a CFM maintenance domain, perform the following steps:

### SUMMARY STEPS

1. **configure**
2. **ethernet cfm**
3. **domain** *domain-name* **level** *level-value* [**id** [**null**] [**dns** *DNS-name*] [**mac** *H.H.H*] [**string** *string*] ]
4. **traceroute cache hold-time** *minutes* **size** *entries*
5. **end**
   or
   **commit**

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | `configure`<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | `ethernet cfm`<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# ethernet cfm` | Enters Ethernet Connectivity Fault Management (CFM) configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **domain** *domain-name* **level** *level-value* [**id** [**null**] [**dns** *DNS-name*] [**mac** *H.H.H*] [**string** *string*] ]<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cfm)# domain Domain_One level 1 id string D1` | Creates and names a container for all domain configurations and enters CFM domain configuration mode.<br><br>The level must be specified.<br><br>The **id** is the maintenance domain identifier (MDID) and is used as the first part of the maintenance association identifier (MAID) in CFM frames. If the MDID is not specified, the domain name is used as the MDID by default. |
| **Step 4** | **traceroute cache hold-time** *minutes* **size** *entries*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cfm)# traceroute cache hold-time 1 size 3000` | (Optional) Sets the maximum limit of traceroute cache entries or the maximum time limit to hold the traceroute cache entries. The default is 100 minutes and 100 entries. |
| **Step 5** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cfm-dmn)# commit` | Saves configuration changes.<br><br>• When you use 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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><br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

## Configuring Services for a CFM Maintenance Domain

You can configure up to 32000 CFM services for a maintenance domain.

### Restrictions

When you configure services for a CFM maintenance domain, consider the following restrictions:

- VPLS configuration (L2VPN bridge groups and bridge-domains) is supported with CFM down MEPs only.
- Policy-Based Tunnel Selection (PBTS) in the core network is not supported.

To configure services for a CFM maintenance domain, perform the following steps:

## SUMMARY STEPS

1. **configure**
2. **ethernet cfm**
3. **domain** *domain-name* **level** *level-value* [**id** [**null**] [**dns** *DNS-name*] [**mac** *H.H.H*] [**string** *string*]]
4. **service** *service-name* {**down-meps | xconnect group** *xconnect-group-name* **p2p** *xconnect-name*}[**id** [**string** *text*] | [**number** *number*] | [**vlan-id** *id-number*] | [**vpn-id** *oui-vpnid*]]
5. **end**
   or
   **commit**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **ethernet cfm**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# ethernet cfm` | Enters Ethernet CFM configuration mode. |
| **Step 3** | **domain** *domain-name* **level** *level-value* [**id** [**null**] [**dns** *DNS-name*] [**mac** *H.H.H*] [**string** *string*] ]<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cfm)# domain Domain_One level 1 id string D1` | Creates and names a container for all domain configurations at a specified maintenance level, and enters CFM domain configuration mode.<br><br>The **id** is the maintenance domain identifier (MDID) and is used as the first part of the maintenance association identifier (MAID) in CFM frames. If the MDID is not specified, the domain name is used as the MDID by default. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **service** *service-name* {**down-meps** \| **xconnect group** *xconnect-group-name* **p2p** *xconnect-name*}[**id** [**string** *text*] \| [**number** *number*] \| [**vlan-id** *id-number*] \| [**vpn-id** *oui-vpnid*]]<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cfm-dmn)# service Bridge_Service bridge group BD1 bridge-domain B1` | Configures and associates a service with the domain and enters CFM domain service configuration mode. You can specify that the service is used only for down MEPs, or associate the service with an xconnect where MIPs and up MEPs will be created.<br>The **id** sets the short MA name. |
| **Step 5** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cfm-dmn-svc)# commit` | Saves configuration changes.<br>• When you use 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

## Enabling and Configuring Continuity Check for a CFM Service

The Cisco XR 12000 Series Router supports Continuity Check as defined in the IEEE 802.1ag specification, and supports CCMs intervals of 100 ms and longer. The overall packet rates for CCM messages are up to 2000 CCMs-per-second sent, and up to 2000 CCMs-per-second received, per card.

**Note** If Ethernet SLA is configured, the overall combined packet rate for CCMs and SLA frames is 2000 frames-per-second in each direction, per card.

To configure Continuity Check for a CFM service, complete the following steps:

### SUMMARY STEPS

1. **configure**
2. **ethernet cfm**
3. **domain** *domain-name* **level** *level-value* [**id** [**null**] [**dns** *DNS-name*] [**mac** *H.H.H*] [**string** *string*]]
4. **service** *service-name* {**down-meps** | **xconnect group** *xconnect-group-name* **p2p** *xconnect-name*}[**id** [**string** *text*] | [**number** *number*] | [**vlan-id** *id-number*] | [**vpn-id** *oui-vpnid*]]

5. **continuity-check interval** *time* [**loss-threshold** *threshold* ]
6. **continuity-check archive hold-time** *minutes*
7. **continuity-check loss auto-traceroute**
8. **end**
or
**commit**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| Step 1 | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| Step 2 | **ethernet cfm**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# ethernet cfm` | Enters Ethernet Connectivity Fault Management (CFM) configuration mode. |
| Step 3 | **domain** *domain-name* **level** *level-value* [**id** [**null**] [**dns** *DNS-name*] [**mac** *H.H.H*] [**string** *string*] ]<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cfm)# domain Domain_One level 1 id string D1` | Creates and names a container for all domain configurations and enters the CFM domain configuration mode.<br>The level must be specified.<br>The **id** is the maintenance domain identifier (MDID) and is used as the first part of the maintenance association identifier (MAID) in CFM frames. If the MDID is not specified, the domain name is used as the MDID by default. |
| Step 4 | **service** *service-name* {**down-meps** \| **xconnect group** *xconnect-group-name* **p2p** *xconnect-name*}[**id** [**string** *text*] \| [**number** *number*] \| [**vlan-id** *id-number*] \| [**vpn-id** *oui-vpnid*]]<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cfm-dmn)# service Bridge_Service bridge group BD1 bridge-domain B1` | Configures and associates a service with the domain and enters CFM domain service configuration mode. You can specify that the service is used only for down MEPs, or associate the service with an xconnect where MIPs and up MEPs will be created.<br>The **id** sets the short MA name. |
| Step 5 | **continuity-check interval** *time* [**loss-threshold** *threshold*]<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cfm-dmn-svc)# continuity-check interval 100m loss-threshold 10` | (Optional) Enables Continuity Check and specifies the time interval at which CCMs are transmitted or to set the threshold limit for when a MEP is declared down. |
| Step 6 | **continuity-check archive hold-time** *minutes*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cfm-dmn-svc)# continuity-check archive hold-time 100` | (Optional) Configures how long information about peer MEPs is stored after they have timed out. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 7** | **continuity-check loss auto-traceroute**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cfm-dmn-svc)# continuity-check loss auto-traceroute` | (Optional) Configures automatic triggering of a traceroute when a MEP is declared down. |
| **Step 8** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cfm-dmn-svc)# commit` | Saves configuration changes.<br>• When you use 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

## Configuring Automatic MIP Creation for a CFM Service

For more information about the algorithm for creating MIPs, see the "MIP Creation" section on page 124.

To configure automatic MIP creation for a CFM service, complete the following steps:

### SUMMARY STEPS

1. **configure**
2. **ethernet cfm**
3. **domain** *domain-name* **level** *level-value* [**id** [**null**] [**dns** *DNS-name*] [**mac** *H.H.H*] [**string** *string*]]
4. **service** *service-name* {**down-meps | xconnect group** *xconnect-group-name* **p2p** *xconnect-name*}[**id** [**string** *text*] | [**number** *number*] | [**vlan-id** *id-number*] | [**vpn-id** *oui-vpnid*]]
5. **mip auto-create {all | lower-mep-only}**
6. **end**
   or
   **commit**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| Step 1 | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| Step 2 | **ethernet cfm**<br><br>**Example:**<br>`RP/0/0/CPU0:router# ethernet cfm` | Enters the Ethernet Connectivity Fault Management (CFM) configuration mode. |
| Step 3 | **domain** *domain-name* **level** *level-value* [**id** [**null**] [**dns** *DNS-name*] [**mac** *H.H.H*] [**string** *string*] ]<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cfm)# domain Domain_One level 1 id string D1` | Creates and names a container for all domain configurations and enters the CFM domain configuration mode.<br><br>The level must be specified.<br><br>The **id** is the maintenance domain identifier (MDID) and is used as the first part of the maintenance association identifier (MAID) in CFM frames. If the MDID is not specified, the domain name is used as the MDID by default. |
| Step 4 | **service** *service-name* {**down-meps** \| **xconnect group** *xconnect-group-name* **p2p** *xconnect-name*}[**id** [**string** *text*] \| [**number** *number*] \| [**vlan-id** *id-number*] \| [**vpn-id** *oui-vpnid*]]<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cfm-dmn)# service Bridge_Service bridge group BD1 bridge-domain B1` | Configures and associates a service with the domain and enters CFM domain service configuration mode. You can specify that the service is used only for down MEPs, or associate the service with an xconnect where MIPs and up MEPs will be created.<br><br>The **id** sets the short MA name. |

| | Command or Action | Purpose |
|---|---|---|
| Step 5 | **mip auto-create** {**all** \| **lower-mep-only**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cfm-dmn-svc)# mip auto-create all` | (Optional) Enables the automatic creation of MIPs in an xconnect. |
| Step 6 | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cfm-dmn-svc)# commit` | Saves configuration changes.<br>• When you use 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

## Configuring Cross-Check on a MEP for a CFM Service

To configure cross-check on a MEP for a CFM service and specify the expected set of MEPs, complete the following steps:

### SUMMARY STEPS

1. **configure**
2. **ethernet cfm**
3. **domain** *domain-name* **level** *level-value* [**id** [**null**] [**dns** *DNS-name*] [**mac** *H.H.H*] [**string** *string*]]
4. **service** *service-name* {**down-meps | xconnect group** *xconnect-group-name* **p2p** *xconnect-name*}[**id** [**string** *text*] | [**number** *number*] | [**vlan-id** *id-number*] | [**vpn-id** *oui-vpnid*]]
5. **mep crosscheck**
6. **mep-id** *mep-id-number* [**mac-address** *mac-address*]
7. **end**
   or
   **commit**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| Step 1 | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| Step 2 | **ethernet cfm**<br><br>**Example:**<br>`RP/0//CPU0:router# ethernet cfm` | Enters the Ethernet Connectivity Fault Management (CFM) configuration mode. |
| Step 3 | **domain** *domain-name* **level** *level-value* [**id** [**null**] [**dns** *DNS-name*] [**mac** *H.H.H*] [**string** *string*] ]<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cfm)# domain Domain_One level 1 id string D1` | Creates and names a container for all domain configurations and enters the CFM domain configuration mode.<br><br>The level must be specified.<br><br>The **id** is the maintenance domain identifier (MDID) and is used as the first part of the maintenance association identifier (MAID) in CFM frames. If the MDID is not specified, the domain name is used as the MDID by default. |
| Step 4 | **service** *service-name* {**down-meps \| xconnect group** *xconnect-group-name* **p2p** *xconnect-name*}[**id** [**string** *text*] \| [**number** *number*] \| [**vlan-id** *id-number*] \| [**vpn-id** *oui-vpnid*]]<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cfm-dmn)# service Bridge_Service bridge group BD1 bridge-domain B1` | Configures and associates a service with the domain and enters CFM domain service configuration mode. You can specify that the service is used only for down MEPs, or associate the service with an xconnect where MIPs and up MEPs will be created.<br><br>The **id** sets the short MA name. |
| Step 5 | **mep crosscheck**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cfm-dmn-svc)# mep crosscheck mep-id 10` | Enters CFM MEP crosscheck configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| Step 6 | **mep-id** *mep-id-number* [**mac-address** *mac-address*]<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cfm-xcheck)# mep-id 10` | Enables cross-check on a MEP.<br>**Note** Repeat this command for every MEP that you want included in the expected set of MEPs for cross-check. |
| Step 7 | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cfm-xcheck)# commit` | Saves configuration changes.<br>• When you use 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

## Configuring Other Options for a CFM Service

To configure other options for a CFM service, complete the following steps:

### SUMMARY STEPS

1. **configure**
2. **ethernet cfm**
3. **domain** *domain-name* **level** *level-value* [**id** [**null**] [**dns** *DNS-name*] [**mac** *H.H.H*] [**string** *string*]]
4. **service** *service-name* {**down-meps** | **xconnect group** *xconnect-group-name* **p2p** *xconnect-name*}[**id** [**string** *text*] | [**number** *number*] | [**vlan-id** *id-number*] | [**vpn-id** *oui-vpnid*]]
5. **maximum meps** *number*
6. **log** {**ais** | **continuity-check errors** | **continuity-check mep changes** | **crosscheck errors** | **efd**}
7. **end**
   or
   **commit**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **`configure`**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **`ethernet cfm`**<br><br>**Example:**<br>`RP/0/0/CPU0:router# ethernet cfm` | Enters the Ethernet Connectivity Fault Management (CFM) configuration mode. |
| **Step 3** | **`domain`** *`domain-name`* **`level`** *`level-value`* `[`**`id`** `[`**`null`**`] [`**`dns`** *`DNS-name`*`] [`**`mac`** *`H.H.H`*`] [`**`string`** *`string`*`] ]`<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cfm)# domain Domain_One level 1 id string D1` | Creates and names a container for all domain configurations and enters the CFM domain configuration mode.<br><br>The level must be specified.<br><br>The **id** is the maintenance domain identifier (MDID) and is used as the first part of the maintenance association identifier (MAID) in CFM frames. If the MDID is not specified, the domain name is used as the MDID by default. |
| **Step 4** | **`service`** *`service-name`* `{`**`down-meps`** `\|` **`xconnect group`** *`xconnect-group-name`* **`p2p`** *`xconnect-name`*`}[`**`id`** `[`**`string`** *`text`*`] \| [`**`number`** *`number`*`] \| [`**`vlan-id`** *`id-number`*`] \| [`**`vpn-id`** *`oui-vpnid`*`]]`<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cfm-dmn)# service Bridge_Service bridge group BD1 bridge-domain B1` | Configures and associates a service with the domain and enters CFM domain service configuration mode. You can specify that the service is used only for down MEPs, or associate the service with an xconnect where MIPs and up MEPs will be created.<br><br>The **id** sets the short MA name. |
| **Step 5** | **`maximum-meps`** *`number`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cfm-dmn-svc)# maximum-meps 1000` | (Optional) Configures the maximum number (2 to 8190) of MEPs across the network, which limits the number of peer MEPs recorded in the database. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 6** | **log** {**ais**\|**continuity-check errors**\|**continuity-check mep changes**\|**crosscheck errors**\|**efd**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cfm-dmn-svc)# log continuity-check errors` | (Optional) Enables logging of certain types of events. |
| **Step 7** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cfm-dmn-svc)# commit` | Saves configuration changes.<br>• When you use 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

## Configuring CFM MEPs

When you configure CFM MEPs, consider the following guidelines:

- Up to 16 MEPs are supported per interface (8 up MEPs and 8 down MEPs) or up to 15 MPs (7 up MEPs, 7 down MEPs, and 1 MIP).
- Up to 2000 maintenance points are supported per card.
- Up to 3000 maintenance points are supported per system.
- CFM maintenance points can be created on the following interface types:
  - Attachment circuit (AC) Layer 2 interfaces and Layer 3 interfaces.
  - Up MEPs can be configured on an AC interface, receiving messages to and from a pseudowire.
  - Down MEPs can be configured on an AC or L3 interface, receiving and sending messages to and from an Ethernet interface.
  - L3 interfaces can only support down MEPs.
  - MIPs are only supported on an AC interface.
  - Both up and down MEPs (and MIPs) can be configured on the same interface. They can be at the same or different levels.

### Restrictions

When you configure MEPs, consider the following restrictions:

- Up MEPs are not supported on Layer 3 interfaces.
- MEPs are not supported on Layer 2 bundle interfaces or bundle member interfaces.

### SUMMARY STEPS

1. **configure**
2. **interface** {**FastEthernet** | **GigabitEthernet** | **TenGigE**} *interface-path-id*
3. **ethernet cfm**
4. **mep domain** *domain-name* **service** *service-name* **mep-id** *id-number*
5. **cos** *cos*
6. **end**
   or
   **commit**

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| Step 1 | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| Step 2 | **interface** {**FastEthernet** \| **GigabitEthernet** \| **TenGigE**} *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface gigabitethernet 0/1/0/1` | Type of Ethernet interface on which you want to create a MEP. Enter **FastEthernet**, **GigabitEthernet** or **TenGigE** and the physical interface or virtual interface.<br><br>**Note** Use the **show interfaces** command to see a list of all interfaces currently configured on the router.<br><br>For more information about the syntax for the router, use the question mark (**?**) online help function. |
| Step 3 | **ethernet cfm**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# ethernet cfm` | Enters interface Ethernet CFM configuration mode. |
| Step 4 | **mep domain** *domain-name* **service** *service-name* **mep-id** *id-number*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if-cfm)# mep domain Dm1 service Sv1 mep-id 1` | Creates a maintenance end point (MEP) on an interface and enters interface CFM MEP configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **cos** *cos*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if-cfm-mep)# cos 7` | (Optional) Configures the class of service (CoS) (from 0 to 7) for all CFM packets generated by the MEP on an interface. If not configured, the CoS is inherited from the Ethernet interface. |
| **Step 6** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if-cfm-mep)# commit` | Saves configuration changes.<br>• When you use 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

# Configuring Y.1731 AIS

This section has the following step procedures:

- Configuring AIS in a CFM Domain Service
- Configuring AIS on a CFM Interface

## Configuring AIS in a CFM Domain Service

Use the following procedure to configure Alarm Indication Signal (AIS) transmission for a CFM domain service and configure AIS logging.

### SUMMARY STEPS

1. **configure**
2. **ethernet cfm**
3. **domain** *name* **level** *level*
4. **service** *name* **xconnect group** *xconnect-group-name* **p2p** *xconnect-name*
5. **ais transmission** [**interval** {**1s** | **1m**}][**cos** *cos*]
6. **log ais**

7. **end**
or
**commit**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| Step 1 | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| Step 2 | **ethernet cfm**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# ethernet cfm` | Enters Ethernet CFM global configuration mode. |
| Step 3 | **domain** *name* **level** *level*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cfm)# domain D1 level 1` | Specifies the domain and domain level. |
| Step 4 | **service** *name* **xconnect group** *xconnect-group-name* **p2p** *xconnect-name*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cfm-dmn)# service S1 bridge group BG1 bridge-domain BD2` | Specifies the service and cross-connect group and name. |
| Step 5 | **ais transmission** [**interval** {**1s**\|**1m**}][**cos** *cos*]<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cfm-dmn-svc)# ais transmission interval 1m cos 7` | Configures Alarm Indication Signal (AIS) transmission for a Connectivity Fault Management (CFM) domain service. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 6** | **`log ais`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cfm-dmn-svc)# log ais` | Configures AIS logging for a Connectivity Fault Management (CFM) domain service to indicate when AIS or LCK packets are received. |
| **Step 7** | **`end`**<br>or<br>**`commit`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sla-prof-stat-cfg)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

### Configuring AIS on a CFM Interface

To configure AIS on a CFM interface, perform the following steps:

#### SUMMARY STEPS

1. **configure**
2. **interface gigabitethernet** *interface-path-id*
3. **ethernet cfm**
4. **ais transmission up interval 1m cos** *cos*
5. **end**
   or
   **commit**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| Step 1 | **`configure`**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| Step 2 | **`interface gigabitethernet`** *`interface-path-id`*<br><br>**Example:**<br>`RP/0/0/CPU0:router# interface gigabitethernet 0/1/0/2` | Enters interface configuration mode. |
| Step 3 | **`ethernet cfm`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# ethernet cfm` | Enters Ethernet CFM interface configuration mode. |
| Step 4 | **`ais transmission up interval 1m cos`** *`cos`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if-cfm)# ais transmission up interval 1m cos 7` | Configures Alarm Indication Signal (AIS) transmission on a Connectivity Fault Management (CFM) interface. |
| Step 5 | **`end`**<br>or<br>**`commit`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sla-prof-stat-cfg)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

## Configuring EFD for a CFM Service

To configure EFD for a CFM service, complete the following steps.

### Restrictions

EFD is not supported on up MEPs. It can only be configured on down MEPs, within a particular service.

### SUMMARY STEPS

1. **configure**
2. **ethernet cfm**
3. **domain** *domain-name* **level** *level-value*
4. **service** *service-name* **down-meps**
5. **efd**
6. **log efd**
7. **end**
   or
   **commit**

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **`configure`**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **`ethernet cfm`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# ethernet cfm` | Enters CFM configuration mode. |
| **Step 3** | **`domain`** *`domain-name`* **`level`** *`level-value`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cfm-dmn)# domain D1 level 1` | Specifies or creates the CFM domain and enters CFM domain configuration mode. |
| **Step 4** | **`service`** *`service-name`* **`down-meps`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cfm-dmn)# service S1 down-meps` | Specifies or creates the CFM service for down MEPS and enters CFM domain service configuration mode. |
| **Step 5** | **`efd`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cfm-dmn-svc)# efd` | Enables EFD on all down MEPs in the down MEPS service. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 6** | **`log efd`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cfm-dmn-svc)# log efd` | (Optional) Enables logging of EFD state changes on an interface. |
| **Step 7** | **`end`**<br>or<br>**`commit`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cfm-dmn-svc)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

### Verifying the EFD Configuration

The following example shows how to display all interfaces that are shut down because of Ethernet Fault Detection (EFD):

```
RP/0/0/CPU0:router# show efd interfaces

Server VLAN MA
==============
Interface         Clients
-------------------------
GigE0/0/0/0.0     CFM
```

## Verifying the CFM Configuration

To verify the CFM configuration, use one or more of the following commands:

| Command | Purpose |
|---|---|
| **show ethernet cfm configuration-errors** [**domain** *domain-name*] [**interface** *interface-path-id* ] | Displays information about errors that are preventing configured CFM operations from becoming active, as well as any warnings that have occurred. |
| **show ethernet cfm local maintenance-points domain** *name* [**service** *name*] \| **interface** *type interface-path-id*] [**mep** \| **mip**] | Displays a list of local maintenance points. |

## Troubleshooting Tips

To troubleshoot problems within the CFM network, perform the following steps:

**Step 1** To verify connectivity to a problematic MEP, use the **ping ethernet cfm** command as shown in the following example:

```
RP/0/0/CPU0:router# ping ethernet cfm domain D1 service S1 mep-id 16 source
interface GigabitEthernet 0/0/0/0

Type escape sequence to abort.
Sending 5 CFM Loopbacks, timeout is 2 seconds -
Domain foo (level 2), Service foo
Source: MEP ID 1, interface GigabitEthernet0/0/0/0
Target: 0001.0002.0003 (MEP ID 16):
  Running (5s) ...
Success rate is 60.0 percent (3/5), round-trip min/avg/max = 1251/1349/1402 ms
Out-of-sequence: 0.0 percent (0/3)
Bad data: 0.0 percent (0/3)
Received packet rate: 1.4 pps
```

**Step 2** If the results of the **ping ethernet cfm** command show a problem with connectivity to the peer MEP, use the **traceroute ethernet cfm** command to help further isolate the location of the problem as shown in the following example:

```
RP/0/0/CPU0:router# traceroute ethernet cfm domain D1 service S1 mep-id 16 source
interface gigabitethernet 0/0/0/0

Traceroutes in domain D1 (level 4), service S1
Source: MEP-ID 1, interface GigabitEthernet0/0/0/0
```

```
================================================================================
Traceroute at 2009-05-18 12:09:10 to 0001.0203.0402,
TTL 64, Trans ID 2:

Hop Hostname/Last            Ingress MAC/name       Egress MAC/Name        Relay
--- ------------------------ ---------------------- ---------------------- -----
  1 ios                      0001.0203.0400 [Down]                         FDB
      0000-0001.0203.0400    Gi0/0/0/0
  2 abc                                             0001.0203.0401 [Ok]    FDB
      ios                                           Not present
  3 bcd                      0001.0203.0402 [Ok]                           Hit
      abc                    GigE0/0
Replies dropped: 0
```

If the target was a MEP, verify that the last hop shows "Hit" in the Relay field to confirm connectivity to the peer MEP.

If the Relay field contains "MPDB" for any of the hops, then the target MAC address was not found in the bridge MAC learning table at that hop, and the result is relying on CCM learning. This result can occur under normal conditions, but it can also indicate a problem. If you used the **ping ethernet cfm** command before using the **traceroute ethernet cfm** command, then the MAC address should have been learned. If "MPDB" is appearing in that case, then this indicates a problem at that point in the network.

# Configuring Ethernet SLA

This section describes how to configure Ethernet SLA.

## Ethernet SLA Configuration Guidelines

**Caution** Certain SLA configurations can use a large amount of memory which can affect the performance of other features on the router.

Before you configure Ethernet SLA, consider the following guidelines:

- Aggregation—Use of the **aggregate none** command significantly increases the amount of memory required because each individual measurement is recorded, rather than just counts for each aggregation bin. When you configure aggregation, consider that more bins will require more memory.
- Buckets archive—When you configure the **buckets archive** command, consider that the more history that is kept, the more memory will be used.
- Measuring two statistics (such as both delay and jitter) will use approximately twice as much memory as measuring one.
- Separate statistics are stored for one-way source-to-destination and destination-to-source measurements, which consumes twice as much memory as storing a single set of round-trip statistics.
- The Cisco XR 12000 Series Router supports SLA packet of 100 ms and longer. The overall packet rates for SLA is up to 2000 CCMs-per-second sent, and up to 2000 CCMs-per-second received, per card.

The following procedure provides the steps to configure Ethernet Service Level Agreement (SLA) monitoring at Layer 2.

To configure SLA, perform the following tasks:

- Configuring an SLA Operation Profile, page 170
- Configuring SLA Probe Parameters in a Profile, page 171
- Configuring SLA Statistics Measurement in a Profile, page 173
- Configuring a Schedule for an SLA Operation Probe in a Profile, page 175
- Configuring an SLA Operation, page 177
- Configuring an On-Demand SLA Operation, page 178
- Verifying SLA Configuration, page 180

## Configuring an SLA Operation Profile

To configure a profile, perform the following steps:

### SUMMARY STEPS

1. **configure**
2. **ethernet sla**
3. **profile** *profile-name* **type** {**cfm-delay-measurement** | **cfm-loopback**}
4. **end**
   or
   **commit**

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **ethernet sla**<br><br>**Example:**<br>`RP/0/0/CPU0:router# ethernet sla` | Enters the SLA configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **profile** *profile-name* **type** {**cfm-delay-measurement** \| **cfm-loopback**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sla)# profile  Prof1 type cfm-loopback` | Creates an SLA operation profile and enters the SLA profile configuration mode. |
| **Step 4** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sla)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

## Configuring SLA Probe Parameters in a Profile

To configure SLA probe parameters in a profile, perform the following steps beginning in SLA profile configuration mode:

### SUMMARY STEPS

1. **probe**
2. **send burst** {**every** *number* {**seconds** | **minutes** | **hours**}| **once**} **packet count** *packets* **interval** *number* {**seconds** | **milliseconds**}
   or
   **send packet** {**every** *number* {**milliseconds** | **seconds** | **minutes** | **hours**} | **once**}
3. **packet size** *bytes* [**test pattern** {**hex 0x***HHHHHHHH* | **pseudo-random**}]
4. **priority** *priority*
5. **end**
   or
   **commit**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **probe**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sla-prof)# probe` | Enters the SLA profile probe configuration mode. |
| **Step 2** | **send burst** {**every** *number* {**seconds** \| **minutes** \| **hours**} \| **once**} **packet count** *packets* **interval** *number* {**seconds** \| **milliseconds**}<br>or<br>**send packet** {**every** *number* {**milliseconds** \| **seconds** \| **minutes** \| **hours**} \| **once**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sla-prof-pb)# send burst every 60 seconds packet count 100 interval 100 milliseconds`<br>or<br>`RP/0/0/CPU0:router(config-sla-prof-pb)# send burst once packet count 2 interval 1 second`<br>or<br>`RP/0/0/CPU0:router(config-sla-prof-pb)# send packet every 100 milliseconds` | Configures the number and timing of packets sent by a probe in an operations profile. |
| **Step 3** | **packet size** *bytes* [**test pattern** {**hex 0x***HHHHHHHH* \| **pseudo-random**}]<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sla-prof-pb)# packet size 9000` | (CFM loopback probe types only) Configures the minimum size (in bytes) for outgoing probe packets, including padding when necessary. Use the test pattern keyword to specify a hexadecimal string to use as the padding characters, or a pseudo-random bit sequence. The default padding is 0's. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **priority** *priority*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sla-prof-pb)# priority 7` | Configures the priority of outgoing SLA probe packets. |
| **Step 5** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sla-prof-pb)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

## Configuring SLA Statistics Measurement in a Profile

The Ethernet SLA feature supports measurement of one-way and two-way delay and jitter statistics.

### Prerequisites

To configure one-way measurements, you must first configure the **profile (SLA)** command using the **type cfm-delay-measurement** form of the command.

### Restrictions

One-way delay and jitter measurements are not supported by cfm-loopback profile types.

To configure SLA statistics measurement in a profile, perform the following steps beginning in SLA profile configuration mode:

### SUMMARY STEPS

1. **statistics measure** {**one-way-delay-ds** | **one-way-delay-sd** | **one-way-jitter-ds** | **one-way-jitter-sd** | **round-trip-delay** | **round-trip-jitter**}
2. **aggregate** {**bins** *count* **width** *width* | **none**}
3. **buckets size** *number* {**per-probe** | **probes**}
4. **buckets archive** *number*

5. **end**
   or
   **commit**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **statistics measure** {**one-way-delay-ds** \| **one-way-delay-sd** \| **one-way-jitter-ds** \| **one-way-jitter-sd** \| **round-trip-delay** \| **round-trip-jitter**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sla-prof)# statistics measure round-trip-delay` | Enables the collection of SLA statistics, and enters SLA profile statistics configuration mode. |
| **Step 2** | **aggregate** {**bins** *count* **width** *width* \| **none**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sla-prof-stat-cfg)# aggregate bins 100 width 10000` | Configures the size and number of bins into which to aggregate the results of statistics collection. |
| **Step 3** | **buckets size** *number* {**per-probe** \| **probes}**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sla-prof-stat-cfg)# buckets size 100 per-probe` | Configures the size of the buckets in which statistics are collected. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **buckets archive** *number*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sla-prof-stat-cfg)# buckets archive 50` | Configures the number of buckets to store in memory. |
| **Step 5** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sla-prof-stat-cfg)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

## Configuring a Schedule for an SLA Operation Probe in a Profile

This section describes how to configure a schedule for an SLA operation probe on an ongoing basis within an SLA profile. For information about how to configure a schedule for a limited, on-demand SLA operation, see the "Configuring an On-Demand SLA Operation" section on page 178.

To configure a schedule for an SLA operation probe, perform the following steps beginning in SLA profile configuration mode:

### SUMMARY STEPS

1. **schedule every week on** *day* [**at** *hh*:*mm*] [**for** *duration* {**seconds** | **minutes** | **hours** | **days** | **week**}]
   or
   **schedule every day** [**at** *hh*:*mm*] [**for** *duration* {**seconds** | **minutes** | **hours** | **days** | **week**}]
   or
   **schedule every** *number* {**hours** | **minutes**}[**first at** *hh*:*mm*[.*ss*]] [**for** *duration* {**seconds** | **minutes** | **hours** | **days** | **week**}]
2. **end**
   or
   **commit**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **schedule every week on** *day* [**at** *hh*:*mm*] [**for** *duration* {**seconds** \| **minutes** \| **hours** \| **days** \| **week**}]<br>or<br>**schedule every day** [**at** *hh*:*mm*] [**for** *duration* {**seconds** \| **minutes** \| **hours** \| **days** \| **week**}]<br>**or**<br>**schedule every** *number* {**hours** \| **minutes**}[**first at** *hh*:*mm*[.*ss*]] [**for** *duration* {**seconds** \| **minutes** \| **hours** \| **days** \| **week**}]<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sla-prof)# schedule every week on Monday at 23:30 for 1 hour`<br>`or`<br>`RP/0/0/CPU0:router(config-sla-prof)# schedule every day at 11:30 for 5 minutes`<br>`or`<br>`RP/0/0/CPU0:router(config-sla-prof)# schedule every 2 hours first at 13:45:01`<br>`or`<br>`RP/0/0/CPU0:router(config-sla-prof)# schedule every 6 hours for 2 hours` | Schedules an operation probe in a profile. A profile may contain only one schedule. |
| **Step 2** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sla-prof-stat-cfg)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

## Configuring an SLA Operation

This section describes how to configure an ongoing SLA operation on a MEP using an SLA profile.

### SUMMARY STEPS

1. **interface** [**FastEthernet** | **GigabitEthernet** | **TenGigE**] *interface-path-id*
2. **ethernet cfm**
3. **mep domain** *domain-name* **service** *service-name* **mep-id** *id-number*
4. **sla operation profile** *profile-name* **target** {**mep-id** *id* | **mac-address** *mac-address*}
5. 
6. **end**
   or
   **commit**

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **interface** [**FastEthernet** \| **GigabitEthernet** \| **TenGigE**] *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# interface gigabitethernet 0/1/0/1` | Physical interface or virtual interface.<br>**Note** Use the **show interfaces** command to see a list of all interfaces currently configured on the router.<br><br>For more information about the syntax for the router, use the question mark (**?**) online help function. |
| **Step 2** | **ethernet cfm**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# ethernet cfm` | Enters interface CFM configuration mode. |
| **Step 3** | **mep domain** *domain-name* **service** *service-name* **mep-id** *id-number*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if-cfm)# mep domain Dm1 service Sv1 mep-id 1` | Creates a MEP on an interface and enters interface CFM MEP configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **sla operation profile** *profile-name* **target** {**mep-id** *id* \| **mac-address** *mac-address*}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if-cfm-mep)# sla operation profile Profile_1 target mac-address 01:23:45:67:89:ab` | Creates an operation instance from a MEP to a specified destination. |
| **Step 5** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sla-prof-stat-cfg)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

## Configuring an On-Demand SLA Operation

You can configure an on-demand SLA operation to run on an as-needed basis for a finite period of time.

This section includes the following topics:


- Configuring Guidelines, page 178
- Configuring an On-Demand Ethernet SLA Operation for CFM Delay Measurement, page 179
- Configuring an On-Demand Ethernet SLA Operation for CFM Loopback, page 180


### Configuration Guidelines

When you configure on-demand SLA operations, consider the following guidelines:

- Each MEP supports up to 50 on-demand operations.
- Each card supports up to 250 on-demand operations.
- On-demand Ethernet SLA operations can be run in addition to any other ongoing scheduled SLA operations that you might have configured, and use similar amounts of CPU and router memory. When configuring an on-demand Ethernet SLA operation, you should consider your existing SLA operation configuration and the potential impact of additional packet processing to your normal operations.

- If you do not specify a schedule for the on-demand operation, the probe defaults to running one time beginning two seconds from the execution of the command, and runs for a ten-second duration.
- If you do not specify the statistics for the probe to measure, it defaults to measuring all statistics, inlcuding the following statistics by probe type:
  - CFM loopback—Two-way delay and jitter is measured by default.
  - CFM delay measurement—One-way delay and jitter in both directions, in addition to two-way delay and jitter is measured by default.
- The default operation mode is synchronous, where progress of the operation is reported to the console and the output of the statistics collection is displayed.

### Configuring an On-Demand Ethernet SLA Operation for CFM Delay Measurement

To configure an on-demand Ethernet SLA operation for CFM delay measurement, use the following command in privileged EXEC configuration mode:

| Command | Purpose |
|---|---|
| **ethernet sla on-demand operation type cfm-delay-measurement probe** [**priority** *number*] [**send** {**packet** {**once** \| **every** *number* {**milliseconds** \| **seconds** \| **minutes** \| **hours**}} \| **burst** {**once** \| **every** *number* {**seconds** \| **minutes** \| **hours**}} **packet count** *number* **interval** *number* {**milliseconds** \| **seconds**}] **domain** *domain-name* **source interface** *type interface-path-id* **target** {**mac-address** *H.H.H.H* \| **mep-id** *id-number*} [**statistics measure** {**one-way-delay-ds** \| **one-way-delay-sd** \| **one-way-jitter-ds** \| **one-way-jitter-sd** \| **round-trip-delay** \| **round-trip-jitter**}][**aggregate** {**none** \| **bins** *number* **width** *milliseconds*}] [**buckets** {**archive** *number* \| **size** *number* {**per-probe** \| **probes**}}] [**schedule** {**now** \| **at** *hh*:*mm*[.*ss*] [*day* [*month* [*year*]]] \| **in** *number* {**seconds** \| **minutes** \| **hours**}}[**for** *duration* {**seconds** \| **minutes** \| **hours**}][**repeat every** *number* {**seconds** \| **minutes** \| **hours**} **count** *probes*]] [**asynchronous**]<br><br>**Example:**<br>`RP/0/0/CPU0:router# ethernet sla on-demand operation type cfm-delay-measurement probe domain D1 source interface TenGigE 0/6/1/0 target mep-id 100` | Configures an on-demand Ethernet SLA operation for CFM delay measurement.<br>The example shows a minimum configuration, that specifies the local domain and source interface and target MEP, using the following defaults:<br>• Send a burst once for a packet count of 10 and interval of 1 second (10-second probe).<br>• Use default class of service (CoS) for the egress interface.<br>• Measure all statistics, including both one-way and round-trip delay and jitter statistics.<br>• Aggregate statistics into one bin.<br>• Schedule now.<br>• Display results on the console. |

### Configuring an On-Demand Ethernet SLA Operation for CFM Loopback

To configure an on-demand Ethernet SLA operation for CFM loopback, use the following command in privileged EXEC configuration mode:

| Command | Purpose |
|---|---|
| **ethernet sla on-demand operation type cfm-loopback probe** [**packet size** *bytes* [**test pattern** {**hex 0x***HHHHHHHH* \| **pseudo-random**}]] [**priority** *number*] [**send** {**packet** {**once** \| **every** *number* {**milliseconds** \| **seconds** \| **minutes** \| **hours**}} \| **burst** {**once** \| **every** *number* {**seconds** \| **minutes** \| **hours**}} **packet count** *number* **interval** *number* {**milliseconds** \| **seconds**}] **domain** *domain-name* **source interface** *type interface-path-id* **target** {**mac-address** *H.H.H.H* \| **mep-id** *id-number*} [**statistics measure** {**round-trip-delay** \| **round-trip-jitter**}][**aggregate** {**none** \| **bins** *number* **width** *milliseconds*}][**buckets** {**archive** *number* \| **size** *number* {**per-probe** \| **probes**}}] [**schedule** {**now** \| **at** *hh***:***mm*[**.***ss*] [*day* [*month* [*year*]]] \| **in** *number* {**seconds** \| **minutes** \| **hours**}}[**for** *duration* {**seconds** \| **minutes** \| **hours**}][**repeat every** *number* {**seconds** \| **minutes** \| **hours**} **count** *probes*]] [**asynchronous**]<br><br>**Example:**<br>`RP/0/0/CPU0:router# ethernet sla on-demand operation type cfm-loopback probe packet size 1500 domain D1 source interface TenGigE 0/6/1/0 target mep-id 100` | Configures an on-demand Ethernet SLA operation for CFM loopback.<br><br>The example shows a minimum configuration, but specifies the option of a minimum packet size, and specifies the local domain and source interface and target MEP, using the following defaults:<br>• Send a burst once for a packet count of 10 and interval of 1 second (10-second probe).<br>• Use default test pattern of 0's for padding.<br>• Use default class of service (CoS) for the egress interface.<br>• Measure all statistics.<br>• Aggregate statistics into one bin.<br>• Schedule now.<br>• Display results on the console. |

## Verifying SLA Configuration

To verify SLA configuration, use one or more of the following commands:

| Command | Purpose |
|---|---|
| **show ethernet sla configuration-errors** [**domain** *domain-name*] [**interface** *interface-path-id*] [**profile** *profile-name*] | Displays information about errors that are preventing configured SLA operations from becoming active, as well as any warnings that have occurred. |
| **show ethernet sla operations** [**detail**] [**domain** *domain-name*] [**interface** *interface-path-id*] [**profile** *profile-name*] | Displays information about configured SLA operations. |

# Configuration Examples for Ethernet OAM

This section provides the following configuration examples:


- Configuration Examples for EOAM Interfaces, page 181
- Configuration Examples for Ethernet CFM, page 183
- Configuration Examples for Ethernet SLA, page 192


## Configuration Examples for EOAM Interfaces

This section provides the following configuration examples:


- Configuring an Ethernet OAM Profile Globally: Example, page 181
- Configuring Ethernet OAM Features on an Individual Interface: Example, page 181
- Configuring Ethernet OAM Features to Override the Profile on an Individual Interface: Example, page 182
- Clearing Ethernet OAM Statistics on an Interface: Example, page 183
- Enabling SNMP Server Traps on a Router: Example, page 183


### Configuring an Ethernet OAM Profile Globally: Example

The following example shows how to configure an Ethernet OAM profile globally:

```
configure terminal
 ethernet oam profile Profile_1
  link-monitor
   frame window 60
   frame threshold low 10000000 high 60000000
   frame-period window 60000
   frame-period threshold low 100 high 12000000
   frame-seconds window 900000
   frame-seconds threshold 3 threshold 900
   exit
  mib-retrieval
  connection timeout 30
  require-remote mode active
  require-remote link-monitoring
  require-remote mib-retrieval
  action dying-gasp error-disable-interface
  action critical-event error-disable-interface
  action discovery-timeout error-disable-interface
  action session-down error-disable-interface
  action capabilities-conflict error-disable-interface
  action wiring-conflict error-disable-interface
  action remote-loopback error-disable-interface
  commit
```

### Configuring Ethernet OAM Features on an Individual Interface: Example

The following example shows how to configure Ethernet OAM features on an individual interface:

```
configure terminal
 interface TenGigE 0/1/0/0
  ethernet oam
```

```
   link-monitor
    frame window 60
    frame threshold low 10000000 high 60000000
    frame-period window 60000
    frame-period threshold low 100 high 12000000
    frame-seconds window 900000
    frame-seconds threshold 3 threshold 900
    exit
   mib-retrieval
   connection timeout 30
   require-remote mode active
   require-remote link-monitoring
   require-remote mib-retrieval
   action link-fault  error-disable-interface
   action dying-gasp error-disable-interface
   action critical-event error-disable-interface
   action discovery-timeout error-disable-interface
   action session-down error-disable-interface
   action capabilities-conflict error-disable-interface
   action wiring-conflict error-disable-interface
   action remote-loopback error-disable-interface
   commit
```

## Configuring Ethernet OAM Features to Override the Profile on an Individual Interface: Example

The following example shows the configuration of Ethernet OAM features in a profile followed by an override of that configuration on an interface:

```
configure terminal
 ethernet oam profile Profile_1
  mode passive
  action dying-gasp disable
  action critical-event disable
  action discovery-timeout disable
  action session-up disable
  action session-down disable
  action capabilities-conflict disable
  action wiring-conflict disable
  action remote-loopback disable
  action uni-directional link-fault error-disable-interface
  commit

configure terminal
 interface TenGigE 0/1/0/0
  ethernet oam
   profile Profile_1
    mode active
    action dying-gasp log
    action critical-event log
    action discovery-timeout log
    action session-up log
    action session-down log
    action capabilities-conflict log
    action wiring-conflict log
    action remote-loopback log
    action uni-directional link-fault log
    uni-directional link-fault detection
    commit
```

### Clearing Ethernet OAM Statistics on an Interface: Example

The following example shows how to clear Ethernet OAM statistics on an interface:

```
RP/0/RP0/CPU0:router# clear ethernet oam statistics interface gigabitethernet 0/1/5/1
```

### Enabling SNMP Server Traps on a Router: Example

The following example shows how to enable SNMP server traps on a router:

```
configure terminal
  ethernet oam profile Profile_1
  snmp-server traps ethernet oam events
```

## Configuration Examples for Ethernet CFM

This section includes the following examples:


- Ethernet CFM Domain Configuration: Example, page 183
- Ethernet CFM Service Configuration: Example, page 183
- Continuity Check for an Ethernet CFM Service Configuration: Example, page 184
- MIP Creation for an Ethernet CFM Service Configuration: Example, page 184
- Cross-check for an Ethernet CFM Service Configuration: Example, page 184
- Other Ethernet CFM Service Parameter Configuration: Example, page 184
- MEP Configuration: Example, page 184
- Ethernet CFM Show Command: Examples, page 184
- AIS for CFM Configuration: Examples, page 187
- AIS for CFM Show Commands: Examples, page 187
- EFD Configuration: Examples, page 191
- Displaying EFD Information: Examples, page 191


### Ethernet CFM Domain Configuration: Example

This example shows how to configure a basic domain for Ethernet CFM:

```
configure
 ethernet cfm
  traceroute cache hold-time 1 size 3000
  domain Domain_One level 1 id string D1
  commit
```

### Ethernet CFM Service Configuration: Example

The following example shows how to create a service for an Ethernet CFM domain:

```
  service Cross_Connect_1 xconnect group XG1 p2p X1
  commit
```

## Continuity Check for an Ethernet CFM Service Configuration: Example

The following example shows how to configure continuity-check options for an Ethernet CFM service:

```
continuity-check archive hold-time 100
continuity-check loss auto-traceroute
continuity-check interval 100ms loss-threshold 10
commit
```

## MIP Creation for an Ethernet CFM Service Configuration: Example

The following example shows how to enable MIP auto-creation for an Ethernet CFM service:

```
mip auto-create all
commit
```

## Cross-check for an Ethernet CFM Service Configuration: Example

The following example shows how to configure cross-check for MEPs in an Ethernet CFM service:

```
mep crosscheck
 mep-id 10
 mep-id 20
 commit
```

## Other Ethernet CFM Service Parameter Configuration: Example

The following example shows how to configure other Ethernet CFM service options:

```
   maximum-meps 4000
   log continuity-check errors
   commit
   exit
  exit
 exit
```

## MEP Configuration: Example

The following example shows how to configure a MEP for Ethernet CFM on an interface:

```
interface gigabitethernet 0/1/0/1
 ethernet cfm
 mep domain Dm1 service Sv1 mep-id 1
 commit
```

## Ethernet CFM Show Command: Examples

The following examples show how to verify the configuration of Ethernet Connectivity Fault Management (CFM):

### Example 1

The following example shows how to display all the maintenance points that have been created on an interface:

```
RP/0/0/CPU0:router# show ethernet cfm local maintenance-points

Domain/Level         Service             Interface          Type   ID   MAC
```

```
-------------------- ------------------- ----------------- ------ ---- --------
fig/5                bay                 Gi0/10/0/12.23456 Dn MEP    2 44:55:66
fig/5                bay                 Gi0/0/1/0.1       MIP         55:66:77
fred/3               barney              Gi0/1/0/0.1       Up MEP    5 66:77:88!
```

### Example 2

The following example shows how to display all the CFM configuration errors on all domains:

```
RP/0/0/CPU0:router# show ethernet cfm configuration-errors

Domain fig (level 5), Service bay
 * An Up MEP is configured for this domain on interface GigabitEthernet0/1/2/3.234 and an
Up MEP is also configured for domain blort, which is at the same level (5).
 * A MEP is configured on interface GigabitEthernet0/3/2/1.1 for this domain/service,
which has CC interval 100ms, but the lowest interval supported on that interface is 1s
```

### Example 3

The following example shows how to display operational state for local maintenance end points (MEPs):

```
RP/0/0/CPU0:router# show ethernet cfm local meps

 A - AIS received               I - Wrong interval
 R - Remote Defect received     V - Wrong Level
 L - Loop (our MAC received)    T - Timed out (archived)
 C - Config (our ID received)   M - Missing (cross-check)
 X - Cross-connect (wrong MAID) U - Unexpected (cross-check)
 P - Peer port down

Domain foo (level 6), Service bar
   ID Interface (State)        Dir MEPs/Err RD Defects AIS
----- ----------------------- --- -------- -- ------- ---
  100 Gi1/1/0/1.234 (Up)       Up     0/0   N  A

Domain fred (level 5), Service barney
   ID Interface (State)        Dir MEPs/Err RD Defects AIS
----- ----------------------- --- -------- -- ------- ---
    2 Gi0/1/0/0.234 (Up)       Up     3/2   Y  RPC
```

### Example 4

The following example shows how to display operational state of other maintenance end points (MEPs) detected by a local MEP:

```
RP/0/0/CPU0:router# show ethernet cfm peer meps

Flags:
 > - Ok                          I - Wrong interval
 R - Remote Defect received      V - Wrong level
 L - Loop (our MAC received)     T - Timed out
 C - Config (our ID received)    M - Missing (cross-check)
 X - Cross-connect (wrong MAID)  U - Unexpected (cross-check)

Domain fred (level 7), Service barney
Up MEP on GigabitEthernet0/1/0/0.234, MEP-ID 2
================================================================================
St    ID MAC address    Port    Up/Downtime   CcmRcvd SeqErr   RDI Error
-- ----- -------------- ------- ----------- --------- ------ ----- -----
 >     1 0011.2233.4455 Up      00:00:01         1234      0     0     0
R>     4 4455.6677.8899 Up      1d 03:04         3456      0   234     0
L      2 1122.3344.5566 Up      3w 1d 6h         3254      0     0  3254
C      2 7788.9900.1122 Test    00:13            2345      6    20  2345
X      3 2233.4455.6677 Up      00:23              30      0     0    30
I      3 3344.5566.7788 Down    00:34           12345      0   300  1234
```

```
V      3 8899.0011.2233 Blocked 00:35            45      0     0    45
 T     5 5566.7788.9900         00:56            20      0     0     0
M      6                                          0      0     0     0
U>     7 6677.8899.0011 Up      00:02           456      0     0     0

Domain fred (level 7), Service fig
Down MEP on GigabitEthernet0/10/0/12.123, MEP-ID 3
================================================================================
St    ID MAC address    Port    Up/Downtime  CcmRcvd SeqErr   RDI Error
-- ----- -------------- ------- ----------- -------- ------ ----- -----
 >     1 9900.1122.3344 Up      03:45           4321      0     0     0
```

### Example 5

The following example shows how to display operational state of other maintenance end points (MEPs) detected by a local MEP with details:

```
RP/0/0/CPU0:router# show ethernet cfm peer meps detail
Domain dom3 (level 5), Service ser3
Down MEP on GigabitEthernet0/0/0/0 MEP-ID 1
================================================================================
Peer MEP-ID 10, MAC 0001.0203.0403
   CFM state: Wrong level, for 00:01:34
   Port state: Up
   CCM defects detected:    V - Wrong Level
   CCMs received: 5
     Out-of-sequence:             0
     Remote Defect received:      5
     Wrong Level:                 0
     Cross-connect (wrong MAID):  0
     Wrong Interval:              5
     Loop (our MAC received):     0
     Config (our ID received):    0
Last CCM received 00:00:06 ago:
     Level: 4, Version: 0, Interval: 1min
     Sequence number: 5, MEP-ID: 10
     MAID: String: dom3, String: ser3
     Port status: Up, Interface status: Up


Domain dom4 (level 2), Service ser4
Down MEP on GigabitEthernet0/0/0/0 MEP-ID 1
================================================================================
Peer MEP-ID 20, MAC 0001.0203.0402
   CFM state: Ok, for 00:00:04
   Port state: Up
   CCMs received: 7
     Out-of-sequence:             1
     Remote Defect received:      0
     Wrong Level:                 0
     Cross-connect (wrong MAID):  0
     Wrong Interval:              0
     Loop (our MAC received):     0
     Config (our ID received):    0
Last CCM received 00:00:04 ago:
     Level: 2, Version: 0, Interval: 10s
     Sequence number: 1, MEP-ID: 20
     MAID: String: dom4, String: ser4
     Chassis ID: Local: ios; Management address: 'Not specified'
     Port status: Up, Interface status: Up

Peer MEP-ID 21, MAC 0001.0203.0403
   CFM state: Ok, for 00:00:05
   Port state: Up
```

```
   CCMs received: 6
     Out-of-sequence:             0
     Remote Defect received:      0
     Wrong Level:                 0
     Cross-connect (wrong MAID):  0
     Wrong Interval:              0
     Loop (our MAC received):     0
     Config (our ID received):    0
Last CCM received 00:00:05 ago:
     Level: 2, Version: 0, Interval: 10s
     Sequence number: 1, MEP-ID: 21
     MAID: String: dom4, String: ser4
     Port status: Up, Interface status: Up
```

## AIS for CFM Configuration: Examples

### Example 1

The following example shows how to configure Alarm Indication Signal (AIS) transmission for a CFM domain service:

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# ethernet cfm
RP/0/0/CPU0:router(config-cfm)# domain D1 level 1
RP/0/0/CPU0:router(config-cfm-dmn)# service Cross_Connect_1 xconnect group XG1 p2p
RP/0/0/CPU0:router(config-cfm-dmn-svc)# ais transmission interval 1m cos 7
```

### Example 2

The following example shows how to configure AIS logging for a Connectivity Fault Management (CFM) domain service to indicate when AIS or LCK packets are received:

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# ethernet cfm
RP/0/0/CPU0:router(config-cfm)# domain D1 level 1
RP/0/0/CPU0:router(config-cfm-dmn)# service Cross_Connect_1 xconnect group XG1 p2p
RP/0/0/CPU0:router(config-cfm-dmn-svc)# log ais
```

The following example shows how to configure AIS transmission on a CFM interface.

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# interface gigabitethernet 0/1/0/2
RP/0/0/CPU0:router(config-if)# ethernet cfm
RP/0/0RP0RSP0/CPU0:router(config-if-cfm)# ais transmission up interval 1m cos 7
```

## AIS for CFM Show Commands: Examples

This section includes the following examples:

- show ethernet cfm interfaces ais Command: Example, page 188
- show ethernet cfm local meps Command: Examples, page 188

## show ethernet cfm interfaces ais Command: Example

The following example shows how to display the information published in the Interface AIS table:

```
RP/0/0/CPU0:router# show ethernet cfm interfaces ais

Defects (from at least one peer MEP):
 A - AIS received               I - Wrong interval
 R - Remote Defect received     V - Wrong Level
 L - Loop (our MAC received)    T - Timed out (archived)
 C - Config (our ID received)   M - Missing (cross-check)
 X - Cross-connect (wrong MAID)  U - Unexpected (cross-check)
 P - Peer port down             D - Local port down

                              Trigger                  Transmission
                         AIS  ---------    Via    ---------------------------
Interface (State)        Dir  L Defects  Levels   L Int Last started  Packets
------------------------ ---  - -------  -------  - --- ------------ --------
Gi0/1/0/0.234 (Up)       Dn   5 RPC      6        7 1s  01:32:56 ago     5576
Gi0/1/0/0.567 (Up)       Up   0 M        2,3      5 1s  00:16:23 ago      983
Gi0/1/0/1.1 (Dn)         Up     D                 7 60s 01:02:44 ago     3764
Gi0/1/0/2 (Up)           Dn   0 RX       1!
```

## show ethernet cfm local meps Command: Examples

### Example 1: Default

The following example shows how to display statistics for local maintenance end points (MEPs):

```
RP/0/0/CPU0:router# show ethernet cfm local meps

 A - AIS received               I - Wrong interval
 R - Remote Defect received     V - Wrong Level
 L - Loop (our MAC received)    T - Timed out (archived)
 C - Config (our ID received)   M - Missing (cross-check)
 X - Cross-connect (wrong MAID)  U - Unexpected (cross-check)
 P - Peer port down

Domain foo (level 6), Service bar
   ID Interface (State)        Dir MEPs/Err RD Defects AIS
----- ------------------------ --- -------- -- ------- ---
  100 Gi1/1/0/1.234 (Up)       Up     0/0   N  A       7

Domain fred (level 5), Service barney
   ID Interface (State)        Dir MEPs/Err RD Defects AIS
----- ------------------------ --- -------- -- ------- ---
    2 Gi0/1/0/0.234 (Up)       Up     3/2   Y  RPC     6
```

### Example 2: Domain Service

The following example shows how to display statistics for MEPs in a domain service:

```
RP/0/RSP0RP0/CPU0:router# show ethernet cfm local meps domain foo service bar detail

Domain foo (level 6), Service bar
Up MEP on GigabitEthernet0/1/0/0.234, MEP-ID 100
================================================================================
  Interface state: Up      MAC address: 1122.3344.5566
  Peer MEPs: 0 up, 0 with errors, 0 timed out (archived)

  CCM generation enabled:  No
  AIS generation enabled:  Yes (level: 7, interval: 1s)
  Sending AIS:             Yes (started 01:32:56 ago)
  Receiving AIS:           Yes (from lower MEP, started 01:32:56 ago)
```

```
Domain fred (level 5), Service barney
Up MEP on GigabitEthernet0/1/0/0.234, MEP-ID 2
================================================================================
  Interface state: Up      MAC address: 1122.3344.5566
  Peer MEPs: 3 up, 2 with errors, 0 timed out (archived)
  Cross-check defects: 0 missing, 0 unexpected

  CCM generation enabled:  Yes (Remote Defect detected: Yes)
  CCM defects detected:    R - Remote Defect received
                           P - Peer port down
                           C - Config (our ID received)
  AIS generation enabled:  Yes (level: 6, interval: 1s)
  Sending AIS:             Yes (to higher MEP, started 01:32:56 ago)
  Receiving AIS:           No
```

#### Example 3: Verbose

The following example shows how to display verbose statistics for MEPs in a domain service:

**Note** The Discarded CCMs field is not displayed when the number is zero (0). It is unusual for the count of discarded CCMs to be any thing other than zero, since CCMs are only discarded when the limit on the number of peer MEPs is reached.

```
RP/0/RSP0RP0/CPU0:router# show ethernet cfm local meps domain foo service bar verbose

Domain foo (level 6), Service bar
Up MEP on GigabitEthernet0/1/0/0.234, MEP-ID 100
================================================================================
  Interface state: Up      MAC address: 1122.3344.5566
  Peer MEPs: 0 up, 0 with errors, 0 timed out (archived)

  CCM generation enabled:  No
  AIS generation enabled:  Yes (level: 7, interval: 1s)
  Sending AIS:             Yes (started 01:32:56 ago)
  Receiving AIS:           Yes (from lower MEP, started 01:32:56 ago)

  Packet        Sent      Received
  ------  ----------  -----------------------------------------------------
  CCM            0             0  (out of seq: 0)
  LBM            0             0
  LBR            0             0  (out of seq: 0, with bad data: 0)
  AIS         5576             0
  LCK            -             0

Domain fred (level 5), Service barney
Up MEP on GigabitEthernet0/1/0/0.234, MEP-ID 2
================================================================================
  Interface state: Up      MAC address: 1122.3344.5566
  Peer MEPs: 3 up, 2 with errors, 0 timed out (archived)
  Cross-check defects: 0 missing, 0 unexpected

  CCM generation enabled:  Yes (Remote Defect detected: Yes)
  CCM defects detected:    R - Remote Defect received
                           P - Peer port down
                           C - Config (our ID received)
  AIS generation enabled:  Yes (level: 6, interval: 1s)
  Sending AIS:             Yes (to higher MEP, started 01:32:56 ago)
  Receiving AIS:           No
```

```
Packet        Sent      Received
------  ----------  --------------------------------------------------------
CCM          12345         67890  (out of seq: 6, discarded: 10)
LBM              5             0
LBR              0             5  (out of seq: 0, with bad data: 0)
AIS              0         46910
LCK              -             0
```

#### Example 4: Detail

The following example shows how to display detailed statistics for MEPs in a domain service:

```
RP/0/0/CPU0:router# show ethernet cfm local meps detail

Domain foo (level 6), Service bar
Up MEP on GigabitEthernet0/1/0/0.234, MEP-ID 100
================================================================================
  Interface state: Up      MAC address: 1122.3344.5566
  Peer MEPs: 0 up, 0 with errors, 0 timed out (archived)

  CCM generation enabled:  No
  AIS generation enabled:  Yes (level: 7, interval: 1s)
  Sending AIS:             Yes (started 01:32:56 ago)
  Receiving AIS:           Yes (from lower MEP, started 01:32:56 ago)

Domain fred (level 5), Service barney
Up MEP on GigabitEthernet0/1/0/0.234, MEP-ID 2
================================================================================
  Interface state: Up      MAC address: 1122.3344.5566
  Peer MEPs: 3 up, 2 with errors, 0 timed out (archived)
  Cross-check defects: 0 missing, 0 unexpected

  CCM generation enabled:  Yes (Remote Defect detected: Yes)
  CCM defects detected:    R - Remote Defect received
                           P - Peer port down
                           C - Config (our ID received)
  AIS generation enabled:  Yes (level: 6, interval: 1s)
  Sending AIS:             Yes (to higher MEP, started 01:32:56 ago)
  Receiving AIS:           No
```

## EFD Configuration: Examples

The following example shows how to enable EFD:

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# ethernet cfm
RP/0/0/CPU0:router(config-cfm)# domain D1 level 1
RP/0/0/CPU0:router(config-cfm-dmn)# service S1 down-meps
RP/0/0/CPU0:router(config-cfm-dmn-svc)# efd
```

The following example shows how to enable EFD logging:

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# ethernet cfm
RP/0/0/CPU0:router(config-cfm)# domain D1 level 1
RP/0/0/CPU0:router(config-cfm-dmn)# service S1 down-meps
RP/0/0/CPU0:router(config-cfm-dmn-svc)# log efd
```

## Displaying EFD Information: Examples

The following examples show how to display information about EFD:

- show efd interfaces Command: Example, page 192
- show ethernet cfm local meps detail Command: Example, page 192

### show efd interfaces Command: Example

The following example shows how to display all interfaces that are shut down in response to an EFD action:

```
RP/0/0/CPU0:router# show efd interfaces

Server VLAN MA
==============
Interface         Clients
-------------------------
GigE0/0/0/0.0     CFM
```

### show ethernet cfm local meps detail Command: Example

Use the **show ethernet cfm local meps detail** command to display MEP-related EFD status information. The following example shows that EFD is triggered for MEP-ID 100:

```
RP/0/0/CPU0:router# show ethernet cfm local meps detail

Domain foo (level 6), Service bar
Up MEP on GigabitEthernet0/1/0/0.234, MEP-ID 100
================================================================================
  Interface state: Up      MAC address: 1122.3344.5566
  Peer MEPs: 0 up, 0 with errors, 0 timed out (archived)
  Cross-check errors: 2 missing, 0 unexpected

  CCM generation enabled:  No
  AIS generation enabled:  Yes (level: 7, interval: 1s)
  Sending AIS:             Yes (started 01:32:56 ago)
  Receiving AIS:           Yes (from lower MEP, started 01:32:56 ago)
  EFD triggered:           Yes

Domain fred (level 5), Service barney
Up MEP on GigabitEthernet0/1/0/0.234, MEP-ID 2
================================================================================
  Interface state: Up      MAC address: 1122.3344.5566
  Peer MEPs: 3 up, 0 with errors, 0 timed out (archived)
  Cross-check errors: 0 missing, 0 unexpected

  CCM generation enabled:  Yes (Remote Defect detected: No)
  AIS generation enabled:  Yes (level: 6, interval: 1s)
  Sending AIS:             No
  Receiving AIS:           No
  EFD triggered:           No
```

**Note** You can also verify that EFD has been triggered on an interface using the **show interfaces** and s**how interfaces brief** commands. When an EFD trigger has occurred, these commands will show the interface status as *up* and the line protocol state as *down*.

# Configuration Examples for Ethernet SLA

This section includes the following examples:


- Ethernet SLA Profile Type Configuration: Examples, page 193
- Ethernet SLA Probe Configuration: Examples, page 193
- Profile Statistics Measurement Configuration: Examples, page 194

- Scheduled SLA Operation Probe Configuration: Examples, page 195
- Ethernet SLA Operation Probe Scheduling and Aggregation Configuration: Example, page 195
- Ongoing Ethernet SLA Operation Configuration: Example, page 196
- On-Demand Ethernet SLA Operation Basic Configuration: Examples, page 197
- Ethernet SLA Show Commands: Examples, page 197

## Ethernet SLA Profile Type Configuration: Examples

The following examples show how to configure the different profile types supported by Ethernet SLA.

### Example 1

This example configures a profile named "Prof1" for CFM loopback measurements:

```
configure
 ethernet sla
  profile  Prof1 type cfm-loopback
   commit
```

### Example 2

This example configures a profile named "Prof1" for CFM delay measurements. Setting this type allows you to configure the probe to measure additional one-way delay and jitter statistics:

```
configure
 ethernet sla
  profile  Prof1 type cfm-delay-measurement
   commit
```

## Ethernet SLA Probe Configuration: Examples

The following examples show how to configure some of the packet options for an Ethernet CFM loopback probe.

### Example 1

This example shows how to configure sending a group of 100 packets in 100 ms intervals and repeat that burst every 60 seconds. Packets are padded to a size of 9000 bytes as needed using a hexadecimal test pattern of "abcdabcd," and with a class of service value of 7:

**Note** The total length of a burst (packet count multiplied by the interval) must not exceed 1 minute.

```
configure
 ethernet sla
  profile  Prof1 type cfm-loopback
   probe
    send burst every 60 seconds packet count 100 interval 100 milliseconds
    packet size 9000 test pattern hex 0xabcdabcd
    priority 7
    commit
```

### Example 2

This example has the same characteristics as the configuration in Example 1, but sends a single burst of 50 packets, one second apart:

```
configure
 ethernet sla
  profile  Prof1 type cfm-loopback
   probe
    send burst once packet count 50 interval 1 second
    packet size 9000 test pattern hex 0xabcdabcd
    priority 7
    commit
```

### Example 3

This example shows how to configure a continuous stream of packets at 100 ms intervals for the duration of the probe. Packets are padded to a size of 9000 bytes as needed using a pseudo-random test pattern, and with a class of service value of 7:

```
configure
 ethernet sla
  profile Prof1 type cfm-loopback
   probe
    send burst every 60 seconds packet count 600 interval 100 milliseconds
    packet size 9000 test pattern pseudo-random
    priority 7
    commit
```

## Profile Statistics Measurement Configuration: Examples

The following examples show how to configure the different types of statistics measurement.

### Example 1

This example shows the two available types of statistics that can be measured by a CFM loopback SLA profile type:

```
configure
 ethernet sla
  profile  Prof1 type cfm-loopback
  statistics measure round-trip-delay
  statistics measure round-trip-jitter
   commit
```

### Example 2

This example shows how to configure measurement of round-trip delay and one-way jitter (from destination to source) for a CFM delay measurement SLA profile type:

**Note** The CFM delay measurement profile type supports measurement of all round-trip and one-way delay and jitter statistics.

```
configure
 ethernet sla
  profile  Prof1 type cfm-delay-measurement
  statistics measure round-trip-delay
  statistics measure one-way-jitter-ds
   commit
```

## Scheduled SLA Operation Probe Configuration: Examples

The following examples show how to configure different schedules for an SLA operation probe.

### Example 1

This example shows how to configure a probe to run hourly for a specified duration:

```
configure
 ethernet sla
  profile  Prof1 type cfm-delay-measurement
  schedule every 1 hours for 15 minutes
  commit
```

### Example 2

This example shows how to configure a probe to run daily for a specified period of time:

```
configure
 ethernet sla
  profile  Prof1 type cfm-delay-measurement
  schedule every day at 11:30 for 5 minutes
  commit
```

### Example 3

This example shows how to configure a probe to run weekly beginning at a specified time and for a specified duration:

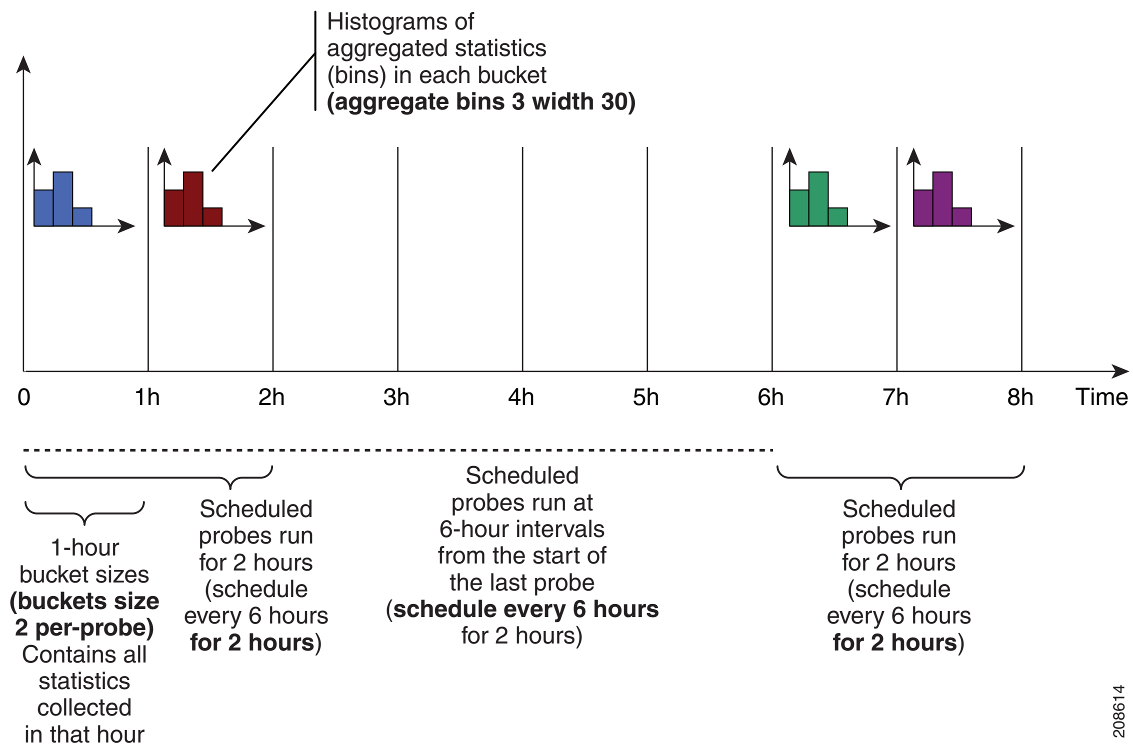
```
configure
 ethernet sla
  profile  Prof1 type cfm-delay-measurement
  schedule every week on Monday at 23:30 for 1 hour
  commit
```

## Ethernet SLA Operation Probe Scheduling and Aggregation Configuration: Example

Figure 12 shows a more comprehensive example of how some of the probe scheduling and measurement configuration works using aggregation. The following configuration supports some of the concepts shown in the figure:

```
configure
 ethernet sla profile Prof1 type cfm-loopback
  probe
   send packet every 60 seconds
   schedule every 6 hours for 2 hours
   statistics measure round-trip-delay
    aggregate bins 3 width 30
    buckets size 2 per-probe
    buckets archive 4
    commit
```

*Figure 12 SLA Probe Scheduling Operation With Bin Aggregation*




This example schedules a probe with the following characteristics:

- Sends packets 60 seconds apart (for a 2-hour probe, this results in sending 120 individual packets).
- Probe runs every 6 hours for 2 hours duration.
- Collects data into 2 buckets for every probe, so each bucket covers 1 hour of the 2-hour probe duration.
- Aggregates statistics within the buckets into 3 bins each in the following ranges:
  - Bin 1 contains samples in the range 0 to < 30 ms.
  - Bin 2 contains samples in the range 30 ms to < 60 ms.
  - Bin 3 contains samples in the range 60 ms or greater (unbounded).
- The last 4 buckets are saved in memory.

## Ongoing Ethernet SLA Operation Configuration: Example

The following example shows how to configure an ongoing Ethernet SLA operation on a MEP:

```
interface gigabitethernet 0/1/0/1
 ethernet cfm
 mep domain Dm1 service Sv1 mep-id 1
 sla operation profile Profile_1 target mac-address 01:23:45:67:89:ab s
 commit
 end
```

## On-Demand Ethernet SLA Operation Basic Configuration: Examples

The following examples show how to configure on-demand Ethernet SLA operations.

### Example 1

The following example shows how to configure a basic on-demand Ethernet SLA operation for a CFM loopback probe that by default will measure round-trip delay and round-trip jitter for a one-time, 10-second operation to the target MEP:

```
RP/0/0/CPU0:router# ethernet sla on-demand operation type cfm-loopback probe domain D1
source interface TenGigE 0/6/1/0 target mep-id 1
```

### Example 2

The following example shows how to configure a basic on-demand Ethernet SLA operation for a CFM delay measurement probe that by default will measure one-way delay and jitter in both directions, as well as round-trip delay and round-trip jitter for a one-time, 10-second operation to the target MEP:

```
RP/0/0/CPU0:router# ethernet sla on-demand operation type cfm-delay-measurement probe
domain D1 source interface TenGigE 0/6/1/0 target mep-id 1
```

## Ethernet SLA Show Commands: Examples

The following examples show how to display information about configured SLA operations:

### show ethernet sla operations Command: Example 1

```
RP/0/0/CPU0:router# show ethernet sla operations interface gigabitethernet 0/1/0/1.1

Interface GigabitEthernet0/1/0/1.1
Domain mydom Service myser to 00AB.CDEF.1234
-----------------------------------------------------------------------------
Profile 'business-gold'
Probe type CFM-delay-measurement:
    bursts sent every 1min, each of 20 packets sent every 100ms
    packets padded to 1500 bytes with zeroes
    packets use priority value of 7
Measures RTT: 5 bins 20ms wide; 2 buckets/ probe; 75/100 archived
Measures Jitter (interval 1): 3 bins 40ms wide; 2 buckets/probe; 50 archived
Scheduled to run every Sunday at 4am for 2 hours:
    last run at 04:00 25/05/2008
```

### show ethernet sla configuration-errors Command: Example 2

```
RP/0/0/CPU0:router# show ethernet sla configuration-errors

Errors:
-------
  Profile 'gold' is not defined but is used on Gi0/0/0/0.0
  Profile 'red' defines a test-pattern, which is not supported by the type
```

The following examples show how to display the contents of buckets containing SLA metrics collected by probes:

### show ethernet sla statistics current Command: Example 3

```
RP/0/0/CPU0:router# show ethernet sla statistics current interface GigabitEthernet
0/0/0/0.0

Interface GigabitEthernet 0/0/0/0.0
Domain mydom Service myser to 00AB.CDEF.1234
```

```
==============================================================================
Profile 'business-gold', packet type 'cfm-loopback'
Scheduled to run every Sunday at 4am for 2 hours

Round Trip Delay
~~~~~~~~~~~~~~~~
2 buckets per probe

Bucket started at 04:00 Sun 17 Feb 2008 lasting 1 hour:
    Pkts sent: 2342; Lost 2 (0%); Corrupt: 0 (0%); Misordered: 0 (0%)
    Min: 13ms; Max: 154ms; Mean: 28ms; StdDev: 11ms

Round Trip Jitter
~~~~~~~~~~~~~~~~~
2 buckets per probe

Bucket started at 04:00 Sun 17 Feb 2008 lasting 1 hour:
    Pkts sent: 2342; Lost: 2 (0%); Corrupt: 0 (0%); Misordered: 0 (0%)
    Min: -5ms; Max: 8ms; Mean: 0ms; StdDev: 3.6ms

Bucket started at 05:00 Sun 17 Feb 2008 lasting 1 hour:
    Pkts sent: 2342; Lost: 2 (0%); Corrupt: 0 (0%); Misordered: 0 (0%)
    Min: 0; Max: 4; Mean: 1.4; StdDev: 1
```

### show ethernet sla statistics history detail Command: Example 4

```
RP/0/0/CPU0:router# show ethernet sla history detail GigabitEthernet 0/0/0/0.0

Interface GigabitEthernet 0/0/0/0.0
Domain mydom Service myser to 00AB.CDEF.1234
================================================================================
Profile 'business-gold', packet type 'cfm-loopback'
Scheduled to run every Sunday at 4am for 2 hours

Round Trip Delay
~~~~~~~~~~~~~~~~
2 buckets per probe

Bucket started at 04:00 Sun 17 Feb 2008 lasting 1 hour:
    Pkts sent: 2342; Lost: 2 (0%); Corrupt: 0 (0%); Misordered: 0 (0%)
    Min: 13ms, occurred at 04:43:29 on Sun 22 Aug 2010 UTC
    Max: 154ms, occurred at 05:10:32 on Sun 22 Aug 2010 UTC
    Mean: 28ms; StdDev: 11ms

    Results suspect as more than 10 seconds time drift detected
    Results suspect as scheduling latency prevented some packets being sent

    Samples:
    Time sent       Result  Notes
    ------------  --------  ----------
    04:00:01.324      23ms
    04:00:01.425      36ms
    04:00:01.525         -  Timed Out
    ...

Round Trip Jitter
~~~~~~~~~~~~~~~~~
2 buckets per probe

Bucket started at 04:00 Sun 17 Feb 2008, lasting 1 hour:
    Pkts sent: 2342; Lost: 2 (0%); Corrupt: 0 (0%); Misordered: 0 (0%)
    Min: -5ms; Max: 10ms; Mean: 0ms; StdDev: 3.6ms

    Samples:
```

```
    Time sent       Result  Notes
    ------------  --------  ----------
    04:00:01.324         -
    04:00:01.425      13ms
    04:00:01.525         -  Timed out
    ...
```

### show ethernet sla statistics history detail on-demand: Example 5

The following example shows how to display statistics for all full buckets for on-demand operations in detail:

```
RP/0/0/CPU0/router #show ethernet sla statistics history detail on-demand

Interface GigabitEthernet0/0/0/0.1
Domain mydom Service myser to 0123.4567.890A
=============================================================================
On-demand operation ID #1, packet type 'cfm-delay-measurement'
Started at 15:38 on 06 July 2010 UTC, runs every 1 hour for 1 hour

Round Trip Delay
~~~~~~~~~~~~~~~~
1 bucket per probe

Bucket started at 15:38 on Tue 06 Jul 2010 UTC, lasting 1 hour:
    Pkts sent: 1200; Lost: 4 (0%); Corrupt: 600 (50%); Misordered: 0 (0%)
    Min: 13ms, occurred at 15:43:29 on Tue 06 Jul 2010 UTC
    Max: 154ms, occurred at 16:15:34 on Tue 06 Jul 2010 UTC
    Mean: 28ms; StdDev: 11ms

    Bins:
    Range              Samples    Cum. Count      Mean
    ------------  ------------  ------------  --------
     0 - 20 ms       194 (16%)      194 (16%)      17ms
    20 - 40 ms       735 (61%)      929 (77%)      27ms
    40 - 60 ms       212 (18%)     1141 (95%)      45ms
    > 60    ms        55  (5%)     1196            70ms

Bucket started at 16:38 on Tue 01 Jul 2008 UTC, lasting 1 hour:
    Pkts sent: 3600; Lost: 12 (0%); Corrupt: 1800 (50%); Misordered: 0 (0%)
    Min: 19ms, occurred at 17:04:08 on Tue 06 Jul 2010 UTC
    Max: 70ms, occurred at 16:38:00 on Tue 06 Jul 2010 UTC
    Mean: 28ms; StdDev: 11ms

    Bins:
    Range              Samples    Cum. Count      Mean
    ------------  ------------  ------------  --------
     0 - 20 ms       194 (16%)      194 (16%)      19ms
    20 - 40 ms       735 (61%)      929 (77%)      27ms
    40 - 60 ms       212 (18%)     1141 (95%)      45ms
    > 60    ms        55  (5%)     1196            64ms
```

# Where to Go Next

When you have configured an Ethernet interface, you can configure individual VLAN subinterfaces on that Ethernet interface.

For information about modifying Ethernet management interfaces for the shelf controller (SC), route processor (RP), and distributed RP, see the "Advanced Configuration and Modification of the Management Ethernet Interface on Cisco IOS XR Software" module later in this document.

For information about IPv6 see the *Implementing Access Lists and Prefix Lists on Cisco IOS XR Software* module in the *Cisco IOS XR IP Addresses and Services Configuration Guide.*

# Additional References

These sections provide references related to implementing Gigabit, 10-Gigabit, and Fast Ethernet interfaces.

## Related Documents

| Related Topic | Document Title |
|---|---|
| Cisco IOS XR master command reference | *Cisco IOS XR Master Commands List* |
| Cisco IOS XR interface configuration commands | *Cisco IOS XR Interface and Hardware Component Command Reference* |
| Information about user groups and task IDs | *Cisco IOS XR Interface and Hardware Component Command Reference* |

## Standards

| Standards | Title |
|---|---|
| IEEE 802.1ag | *Connectivity Fault Management* |
| ITU-T Y.1731 | *OAM Functions and Mechansims for Ethernet Based Networks* |
| MEF 16 | *Metro Ethernet Forum, Technical Specification MEF 16, Ethernet Local Management Interface (E-LMI), January 2006* |

## MIBs

| MIBs | MIBs Link |
|---|---|
| IEEE8021-CFM-MIB | To locate and download MIBs for selected platforms using Cisco IOS XR Software, use the Cisco MIB Locator found at the following URL:<br>http://www.cisco.com/go/mibs |

## RFCs

| 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/support |

# Configuring Link Bundling on Cisco IOS XR Software

This module describes the configuration of link bundle interfaces on the Cisco XR 12000 Series Router.

A link bundle is a group of one or more ports that are aggregated together and treated as a single link.

Each bundle has a single MAC, a single IP address, and a single configuration set, such as Quality of Service (QoS).

**Note** Link bundles do not have a one-to-one modular services card association. Member links can terminate on different cards.

**Feature History for Configuring Link Bundling**

| Release | Modification |
|---|---|
| Release 3.6.0 | This feature was first supported on the Cisco XR 12000 Series Router. |
| Release 3.8.0 | This feature was updated as follows:<br>• The **reasons** keyword was removed from the **show bundle bundle-Ether** command and the **show bundle bundle-POS** command. Now, if a port is in a state other than the distributing state, the output of both commands displays the reason.<br>• The **hot-standby** keyword was added to the **bundle maximum-active links** command.<br>• The **lacp fast-switchover** command was added. |
| Release 3.8.4 | Bundle member links are put into new err-disable link interface status and admin-down protocol state when a bundle interface is shut down. |

| | |
|---|---|
| Release 3.9.0 | Support for short LACP was added.<br>Support for load balancing was added.<br>Support for POS link bundles, QoS on link bundles, and MPLS TE with FRR on link bundles was added. |
| Release 4.1.0 | Support for the following link bundling features was added:<br>• ACLs<br>• IPv6<br>• L2TPv3 with core-facing link bundles<br>• MAC accounting<br>• mVPNv4 with edge-facing link bundles<br>• 6PE with link bundles on the edge (with MPLS core)<br>• PIM IPv6<br>• Unequal bandwidth<br>• uRPF<br>• VPLS with core-facing link bundles (Ethernet and POS SPAs only as supported by the Cisco XR 12000 SIP-600, SIP-401, SIP-501 and SIP-601)<br>• VPLS with link bundles as core (Ethernet and POS SPAs only as supported by the Cisco XR 12000 SIP-600, SIP-401, SIP-501 and SIP-601) |

# Contents

This module includes the following sections:


- Prerequisites for Configuring Link Bundling, page 205
- Information About Configuring Link Bundling, page 207
- How to Configure Link Bundling, page 216
- Configuration Examples for Link Bundling, page 245
- Additional References, page 250


# Prerequisites for Configuring Link Bundling

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.

The prerequisites for link bundling depend on the platform on which you are configuring this feature. This section includes the following information:


- Prerequisites for Configuring Link Bundling on a Cisco XR 12000 Series Router, page 205


## Prerequisites for Configuring Link Bundling on a Cisco XR 12000 Series Router

Before configuring link bundling on a Cisco XR 12000 Series Router, be sure that the following tasks and conditions are met:

- You know the interface IP address.
- You know which links should be included in the bundle you are configuring.
- You have at least one of the following SIPs installed in the router:
  - Cisco XR 12000 SIP-401
  - Cisco XR 12000 SIP-501
  - Cisco XR 12000 SIP-601
- You have at least one of the following Ethernet cards installed in the router:
  - 1-port 10-Gigabit Ethernet SPA
  - 10-port Gigabit Ethernet SPA
  - 8-port Gigabit Ethernet SPA
  - 5-port Gigabit Ethernet SPA
  - 4-port ISE Gigabit Ethernet Line Card
  - 4-port OC-3 POS/SDH SPA
  - 8-port OC-3 POS/SDH SPA
  - 2-port OC-12 POS/SDH SPA
  - 4-port OC-12 POS/SDH SPA
  - 8-port OC-12 POS/SDH SPA

- 4-port OC-3 POS ISE Line Card
- 8-port OC-3 POS ISE Line Card
- 12-port OC-3 POS ISE Line Card
- 16-port OC-3 POS ISE Line Card
- 1-port OC-48 POS ISE Line Card
- 4 port OC-12 POS ISE Line Card

## Restrictions

After a system upgrade or reload, current link bundling design prevents a bundled interface from remaining in the up state if there are errors encountered on features, such as QOS, configured under it. This behavior applies to bundled interfaces only. Physical interfaces are not effected.

Media Access Control (MAC) Accounting is *not* supported on Ethernet link bundles.

## Features Supported

On the Cisco XR 12000 Series Router, link bundling supports the following features:

- 802.3ad LACP support
- Access Control Lists (ACLs)
- IPv4 unicast and multicast forwarding
- IPv6 unicast and multicast forwarding
- SNF, Routing protocols support over LB interface
- L2TPv3 with core-facing link bundles
- MPLS
- MPLS forwarding
- MPLS traffic engineering (TE)
- MPLS TE for Fast Re-route (FRR)
- mMVPN4 for both core- and edge-facing bundles
- PIM IPv6
- QoS on Link Bundles
- 6PE with link bundles on the edge (with MPLS core)
- Unequal bandwidth
- uRPF
- VPLS with core-facing link bundles (Ethernet and POS SPAs only as supported by the Cisco XR 12000 SIP-600, SIP-401, SIP-501 and SIP-601)
- VPLS with link bundles as core (Ethernet and POS SPAs only as supported by the Cisco XR 12000 SIP-600, SIP-401, SIP-501 and SIP-601)

## Features Not Supported

On the Cisco XR 12000 Series Router, link bundling does *not* support the following features:

- QinQ encapsulation
- Multicast VPNs
- 1:1 link protection
- Channelized POS interfaces
- X-Blade with link-bundle interfaces in the core
- Interfaces on the 4-port ISE Gigabit Ethernet line card.
  Link bundling is not supported on the 4-port ISE Gigabit Ethernet line card. No interfaces on this card can be members of a link bundle. However, traffic entering through interfaces on this card can be sent out through link bundle interfaces.
- All member links must be configured to run at the same speed. If members of a bundle are configured at different speeds, only members that run at the speed of the highest priority link will be active links in that bundle.
- The maximum number of link bundling interfaces allowed on a line card is 16:
  - 16 Ethernet bundles
  - 16 Packet-over-SONET (POS) bundles
- The maximum number of VLAN link bundle interfaces allowed on a line card is 100.
- The maximum number of members allowed in a link bundle is 8. Users can configure more than 8 members, but the additional links are created in the detached state and become active only when currently active members are detached from the bundle.
- You cannot configure Virtual Firewall (VFW) on a link bundle interface. Traffic from VFW cannot egress through a link bundle interface.
- A link bundle with LACP enabled will flap during Minimum Disruption Restart (MDR).
- For Packet-over-SONET (POS) link bundles, only HDLC encapsulation is supported. No other encapsulations are supported.

# Information About Configuring Link Bundling

To configure link bundling, you must understand the following concepts:

- Link Bundling Overview, page 208
- Features and Compatible Characteristics of Link Bundles, page 209
- Link Aggregation Through LACP, page 210
- LACP Short Period Time Intervals, page 211
- Load Balancing, page 212
- Unequal Bandwidth on Link Bundles, page 213
- QoS and Link Bundling, page 213
- MPLS-TE and FRR over Link Bundles, page 213
- Link Bundle Configuration Overview, page 215
- Nonstop Forwarding During RP Switchover, page 215

- Link Switchover, page 216

## Link Bundling Overview

The Link Bundling feature allows you to group multiple point-to-point links together into one logical link and provide higher bidirectional bandwidth, redundancy, and load balancing between two routers. A virtual interface is assigned to the bundled link. The component links can be dynamically added and deleted from the virtual interface.

The virtual interface is treated as a single interface on which one can configure an IP address and other software features used by the link bundle. Packets sent to the link bundle are forwarded to one of the links in the bundle.

The advantages of link bundles are as follows:

- Multiple links can span several line cards and SPAs to form a single interface. Thus, the failure of a single link does not cause a loss of connectivity.
- Bundled interfaces increase bandwidth availability, because traffic is forwarded over all available members of the bundle. Therefore, traffic can move onto another link if one of the links within a bundle fails. You can add or remove bandwidth without interrupting packet flow. For example, you can upgrade from an OC-48c PLIM modular services card to an OC-192 PLIM modular services card without interrupting traffic.

All links within a bundle must be of the same type. For example, a bundle can contain all Ethernet interfaces, or it can contain all POS interfaces, but it cannot contain Ethernet and POS interfaces at the same time.

Cisco IOS XR software supports the following methods of forming bundles of Ethernet and POS interfaces:

- IEEE 802.3ad—Standard technology that employs a Link Aggregation Control Protocol (LACP) to ensure that all the member links in a bundle are compatible. Links that are incompatible or have failed are automatically removed from a bundle.
- EtherChannel or POS Channel—Cisco proprietary technology that allows the user to configure links to join a bundle, but has no mechanisms to check whether the links in a bundle are compatible. (EtherChannel applies to Ethernet interfaces, and POS Channel applies to POS interfaces.)

The following types of link bundling are supported on Cisco XR 12000 Series Routers:

- EtherChannel is used bundle multiple Gigabit Ethernet (GE) interfaces.
- Gigabit EtherChannel link bundles are at Layer 2 and use one MAC address and one IP address for all GigabitEthernet interfaces in the bundle.
- POS Channel is used to bundle multiple Packet-over-SONET (POS) interfaces.

On Cisco XR 12000 Series Routers, Gigabit EtherChannel and POS Channel link bundling provide the following benefits:

- Flexible, incremental bandwidth
- Transparency to network applications
- Support for IP unicast and MPLS traffic
- Load balancing (equal cost) across all active links on the bundle
- Redundancy: if there is a failure of an individual GE or POS link, the traffic flow through the channel is evenly distributed across the available links.
- Interoperability with link bundling implementations in other Cisco and OEM routers and switches.

- Out-of-service support: a Gigabit EtherChannel or POS Channel is brought down if the minimum number of Gigabit Ethernet or POS links are not up.
- Bandwidth propagation support: bandwidth changes in a Gigabit EtherChannel or POS Channel can be (optionally) propagated to the upper-layer protocols until the amount of bandwidth required in the link bundle exceeds a specified threshold.

## Features and Compatible Characteristics of Link Bundles

Link bundles support the following features:

- ACL
- Basic IP
- Basic MPLS
- MPLS VPN
- Sampled Netflow
- BGP Policy Accounting
- HSRP/VRRP
- VLAN Bundling (Ethernet only)
- Basic IP
- Basic MPLS
- MPLS VPN
- Inter-AS
- WRED/MDRR per member interface.

The following list describes the properties and limitations of link bundles:

- A bundle contains links, each of which has LACP enabled or disabled. If a bundle contains links, some that have LACP enabled and some that have LACP disabled, the links with LACP disabled are not aggregated in the bundle.
- Bundle membership can span across several modular services cards that are installed in a single router and across SPAS in the same service card.
- The Cisco XR 12000 Series Router supports a maximum of 16 bundles: 16 Etherbundles or 16 POS bundles.
- Physical layer and link layer configuration are performed on individual member links of a bundle.
- Configuration of network layer protocols and higher layer applications is performed on the bundle itself.
- IPv4 and IPv6 addressing is supported on ethernet link bundles.
- A bundle can be administratively enabled or disabled. Beginning in Cisco IOS XR Release 3.8.4, when you shut down a bundle interface, the member links are put into err-disable link interface status and admin-down line protocol state. You can show the status of a bundle interface and its members using the **show interfaces** command.
- Each individual link within a bundle can be administratively enabled or disabled.
- Each individual member link within a bundle has unique MAC address.

- If a MAC address is not set on the bundle, the bundle and bundle members inherit the address of the first member.
- Each link within a bundle can be configured to allow different keepalive periods on different members.
- Load balancing (the distribution of data between member links) is done by flow instead of by packet.
- Upper layer protocols, such as routing updates and hellos, are sent over any member link of an interface bundle.
- All links within a single bundle must terminate on the same two systems. Both systems must be directly connected.
- Bundled interfaces are point-to-point.
- A bundle can contain physical links only. Tunnels and VLAN subinterfaces cannot be bundle members. However, you can create VLANs as subinterfaces of bundles.
- An IPv4 address configuration on link bundles is identical to an IPv4 address configuration on regular interfaces.
- Multicast traffic is load balanced over the members of a bundle. For a given flow, internal processes select the member link, and all traffic for that flow is sent over that member.

### Characteristics of Cisco XR 12000 Series Router Link Bundles

The following list describes additional properties and limitations of link bundles that are specific to Cisco XR 12000 Series Routers:

- A single bundle supports a maximum of 8 physical links. If you add more than 8 links to a bundle, only 8 of the links function, and the remaining links are automatically disabled.
- A Cisco XR 12000 Series Router supports a maximum of 16 bundles.
- Ethernet link bundles are created in the same way as Ethernet channels, where the user enters the same configuration on both end systems.

## Link Aggregation Through LACP

Aggregating interfaces on different modular services cards and on SPAs within the same services cards provides redundancy, allowing traffic to be quickly redirected to other member links when an interface or modular services card failure occurs.

The optional Link Aggregation Control Protocol (LACP) is defined in the IEEE 802 standard. LACP communicates between two directly connected systems (or peers) to verify the compatibility of bundle members. The peer can be either another router or a switch. LACP monitors the operational state of link bundles to ensure the following:

- All links terminate on the same two systems.
- Both systems consider the links to be part of the same bundle.
- All links have the appropriate settings on the peer.

LACP transmits frames containing the local port state and the local view of the partner system’s state. These frames are analyzed to ensure both systems are in agreement.

## IEEE 802.3ad Standard

The IEEE 802.3ad standard typically defines a method of forming Ethernet link bundles. In Cisco IOS XR software, the IEEE 802.3ad standard is used on both Ethernet and POS link bundles.

For each link configured as bundle member, the following information is exchanged between the systems that host each end of the link bundle:

- A globally unique local system identifier
- An identifier (operational key) for the bundle of which the link is a member
- An identifier (port ID) for the link
- The current aggregation status of the link

This information is used to form the link aggregation group identifier (LAG ID). Links that share a common LAG ID can be aggregated. Individual links have unique LAG IDs.

The system identifier distinguishes one router from another, and its uniqueness is guaranteed through the use of a MAC address from the system. The bundle and link identifiers have significance only to the router assigning them, which must guarantee that no two links have the same identifier, and that no two bundles have the same identifier.

The information from the peer system is combined with the information from the local system to determine the compatibility of the links configured to be members of a bundle.

The MAC address of the first link attached to a bundle becomes the MAC address of the bundle itself. The bundle uses this MAC address until that link (the first link attached to the bundle) is detached from the bundle, or until the user configures a different MAC address. The bundle MAC address is used by all member links when passing bundle traffic. Any unicast or multicast addresses set on the bundle are also set on all the member links.

**Note** We recommend that you avoid modifying the MAC address, because changes in the MAC address can affect packet forwarding.

# LACP Short Period Time Intervals

As packets are exchanged across member links of a bundled interface, some member links may slow down or time-out and fail. LACP packets are exchanged periodically across these links to verify the stability and reliability of the links over which they pass. The configuration of short period time intervals, in which LACP packets are sent, enables faster detection and recovery from link failures.

**Note** On the Cisco XR 12000 Series Router, only the default short period (1000 milliseconds) is supported.

Short period time intervals are configured as follows:

- In milliseconds
- In increments of 100 milliseconds
- In the range 100 to 1000 milliseconds
- The default is 1000 milliseconds (1 second)
- Up to 64 member links
- Up to 1280 packets per second (pps)

After 6 missed packets, the link is detached from the bundle.

When the short period time interval is *not* configured, LACP packets are transmitted over a member link every 30 seconds by default.

When the short period time interval is configured, LACP packets are transmitted over a member link once every 1000 milliseconds (1 second) by default. Optionally, both the transmit and receive intervals can be configured to less than 1000 milliseconds, independently or together, in increments of 100 milliseconds (100, 200, 300, and so on).

When you configure a custom LACP short period *transmit* interval at one end of a link, you must configure the same time period for the *receive* interval at the other end of the link.

**Note** You must always configure the *transmit* interval at both ends of the connection before you configure the *receive* interval at either end of the connection. Failure to configure the *transmit* interval at both ends first results in route flapping (a route going up and down continuously). When you remove a custom LACP short period, you must do it in reverse order. You must remove the *receive* intervals first and then the *transmit* intervals.

# Load Balancing

Load balancing is a forwarding mechanism which distributes traffic over multiple links, based on Layer 3 routing information in the router. Per-flow load balancing is supported on all links in the bundle. This scheme achieves load sharing by allowing the router to distribute packets over one of the links in the bundle, that is determined through a hash calculation. The hash calculation is an algorithm for link selection based on certain parameters.

The standard hash calculation is a 3-tuple hashing, using the following parameters:

- IP source address
- IP destination address
- Router ID

7-tuple hashing can also be configured. based on Layer 3 and Layer 4 parameters:

- IP source address
- IP destination address
- Router ID
- Input interface
- IP protocol
- Layer 4 source port
- Layer 4 destination port

When per-flow load balancing and 3-tuple hashing is enabled, all packets for a certain source-destination pair will go through the same link, though there are multiple links available. Per-flow load balancing ensures that packets for a certain source-destination pair arrive in order.

**Note** For multicast traffic, ingress forwarding is based on the Fabric Multicast Group Identifier (FGID). Egress forwarding over the bundle is based on the bundle load balancing.

## Unequal Bandwidth on Link Bundles

In Cisco IOS XR Releases prior to 4.1, all of the interfaces in a link bundle are required to have the same bandwidth characteristic. For example, an OC-3 POS link could be part of a bundle which has other OC-3 links, but it could not be part of a bundle with either OC-12 or OC-48 links.

Unequal bandwidth on link bundles allows links with different bandwidth characteristics to be part of the same bundle. This feature allows a 4:1 ratio in bandwidth characteristic of participating links in the same bundle, which means the following combination of links are supported:

- Bundles with OC-3 and OC-12 links
- Bundles with OC-12 and OC-48 links

The following combination of links are unsupported:

- Bundles with OC-3 and OC-48 links
- Bundles with Gigabit links and 10 Gigabit links
- Bundle with Gigabit and POS links

## QoS and Link Bundling

On the Cisco XR 12000 Series Router, QoS features currently supported on Ethernet and Packet-over-SONET (POS) interfaces are also supported on link bundle interfaces.

For complete information on configuring QoS and important restrictions for link bundles, refer to the *Cisco XR 12000 Series Router Modular Quality of Service Configuration Guide* and the *Cisco XR 12000 Series Router Modular Quality of Service Command Reference*.

## MPLS-TE and FRR over Link Bundles

Beginning with Cisco IOS XR Release 3.9.0, MPLS Traffic Engineering (TE) tunnels and Fast Re-Route (FRR) are supported on Link Bundle interfaces.

MPLS-TE and FRR are supported on the following types of Link Bundle interfaces:

- Packet-over-SONET (POS)
- Ethernet

MPLS-TE is supported, but FRR is not supported on the following types of Link Bundle interfaces:

- VLANs

The following example shows the configuration for FRR:

```
config
   mpls traffic-eng
        interface Bundle-Ether1
             backup-path tunnel-te 2
!
```

For complete information on MPLS-TE and FRR, refer to the Cisco XR 12000 Series Router MPLS Configuration Guide and theCisco XR 12000 Series Router MPLS Command Reference.

## Restrictions

The following restrictions apply to MPLS-TE and FRR over Link Bundles in Cisco IOS XR Release 3.9.0:

- The maximum number of links for MPLS-TE and FRR is 100.
- Backup assignments are void if both the primary assignment and the backup assignment are over the same physical Link Bundle interface.
- Packet loss greater than 50ms can happen if a single member link goes down, but the number of currently active members is still above the configured threshold.
- BFD is not supported on bundle interfaces.

## MPLS-TE

MPLS-TE software enables an MPLS backbone to replicate and expand the traffic engineering capabilities of Layer 2 ATM and Frame Relay networks. MPLS is an integration of Layer 2 and Layer 3 technologies. By making traditional Layer 2 features available to Layer 3, MPLS enables traffic engineering (TE). With MPLS, TE capabilities are integrated into Layer 3, which optimizes the routing of IP traffic, given the constraints imposed by backbone capacity and topology.

## FRR

Fast ReRoute (FRR) is used by MPLS-TE. FRR guarantees that if a TE tunnel fails, traffic is switched to a backup tunnel. FRR provides link protection to LSPs by rerouting traffic carried by LSPs to other links. The ability to configure FRR on a per-LSP basis makes it possible to provide different levels of FRR to tunnels with different bandwidths.

FRR is triggered on Link Bundles in the following ways:

- When the minimum links threshold is reached, FRR is triggered over a Link Bundle interface.
- When the minimum available bandwidth threshold is reached, FRR is triggered over a Link Bundle interface.

## CLI

No new CLI commands are introduced in Cisco IOS XR Release 3.9.0.

See Example: Configuring MPLS-TE and FRR over Link Bundles, page 249 for examples of how to configure MPLS-TE and FRR on Link Bundles.

To verify MPLS-TE and FRR over Link Bundles, use any of the following MPLS commands that are documented in the Cisco XR 12000 Series Router MPLS Command Reference:

- **show [l cef [ipv4 | mpls] adjacency tunnel-te hardware [ingress | egress] location**
- **show int tunnel-te * accounting**
- **sshowh mpls traffic-eng fast-reroute database**
- **show mpls traffic-eng tunnels 1**
- **show mpls forwarding**
- **show cef ipv4 adjacency hardware egress location**
- **show cef ipv4 adjacency bundle-ether 22 hardware egress location**

- **show cef ipv4 address hardware ingress location**
- **show cef ipv4 address hardware egress location**
- **show cef ipv4 address hardware ingress location**

## VLANs on an Ethernet Link Bundle

802.1Q VLAN subinterfaces can be configured on 802.3ad Ethernet link bundles. Keep the following information in mind when adding VLANs on an Ethernet link bundle:

- The maximum number of VLANs allowed per bundle is 100.
- The maximum number of bundled VLANs allowed per router is 1600.

**Note** The memory requirement for bundle VLANs is slightly higher than standard physical interfaces.

To create a VLAN subinterface on a bundle, include the VLAN subinterface instance with the **interface Bundle-Ether** command, as follows:

**interface Bundle-Ether** *interface-bundle-id.subinterface*

After you create a VLAN on an Ethernet link bundle, all VLAN subinterface configuration is supported on that link bundle.

VLAN subinterfaces can support multiple Layer 2 frame types and services, such as Ethernet Flow Points - EFPs) and Layer 3 services.

## Link Bundle Configuration Overview

The following steps provide a general overview of the link bundle configuration. Keep in mind that a link must be cleared of all previous network layer configuration before it can be added to a bundle:

1. In global configuration mode, create a link bundle. To create an Ethernet link bundle, enter the **interface Bundle-Ether** command. To create a POS link bundle, enter the **interface Bundle-POS** command.
2. Assign an IP address and subnet mask to the virtual interface using the **ipv4 address** command.
3. Add interfaces to the bundle you created in Step 1 with the **bundle id** command in the interface configuration submode. You can add up to 32 links to a single bundle.

**Note** A link is configured as a member of a bundle from the interface configuration submode for that link.

## Nonstop Forwarding During RP Switchover

Cisco IOS XR software supports nonstop forwarding during switchover between active and standby paired RP cards. Nonstop forwarding ensures that there is no change in the state of the link bundles when a switchover occurs.

For example, if an active RP fails, the standby RP becomes operational. The configuration, node state, and checkpoint data of the failed RP are replicated to the standby RP. The bundled interfaces will all be present when the standby RP becomes the active RP.

**Note** You do not need to configure anything to guarantee that the standby interface configurations are maintained.

## Link Switchover

By default, a maximum of 8 links can actively carry traffic on a Cisco XR 12000 Series Router. If one member link in a bundle fails, traffic is redirected to the remaining operational member links.

If the active and standby links are running LACP, you can choose between an IEEE standard-based switchover (the default) or a faster proprietary optimized switchover. If the active and standby links are not running LACP, the proprietary optimized switchover option is used.

Regardless of the type of switchover you are using, you can disable the wait-while timer, which expedites the state negotiations of the standby link and causes a faster switchover from a failed active link to the standby link.

# How to Configure Link Bundling

This section contains the following procedures:

- Configuring Ethernet Link Bundles, page 216
- Configuring EFP Load Balancing on an Ethernet Link Bundle, page 221
- Configuring VLAN Bundles, page 223
- Configuring POS Link Bundles, page 229
- Configuring the Default LACP Short Period Time Interval, page 234
- Configuring Custom LACP Short Period Time Intervals, page 236

## Configuring Ethernet Link Bundles

This section describes how to configure an Ethernet link bundle.

**Note** MAC accounting is not supported on Ethernet link bundles.

**Note** In order for an Ethernet bundle to be active, you must perform the same configuration on both connection endpoints of the bundle.

### SUMMARY STEPS

The creation of an Ethernet link bundle involves creating a bundle and adding member interfaces to that bundle, as shown in the steps that follow.

1. **configure**
2. **interface Bundle-Ether** *bundle-id*
3. **ipv4 address** *ipv4-address mask*

4. **bundle minimum-active bandwidth** *kbps*
5. **bundle minimum-active links** *links*
6. **bundle maximum-active links** *links* [**hot-standby**]
7. **lacp fast-switchover**
8. **exit**
9. **interface** {**GigabitEthernet** | **TenGigE**} *interface-path-id*
10. **bundle id** *bundle-id* [**mode** {**active** | **on** | **passive**}
11. **bundle port-priority** *priority*
12. **no shutdown**
13. **exit**
14. Repeat Step 8 through Step 11 to add more links to the bundle you created in Step 2.
15. **end**
    or
    **commit**
16. **exit**
17. **exit**
18. Perform Step 1 through Step 15 on the remote end of the connection.
19. **show bundle Bundle-Ether** *bundle-id*
20. **show lacp Bundle-Ether** *bundle-id*

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| Step 1 | `configure`<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| Step 2 | **`interface Bundle-Ether`** *`bundle-id`*<br><br>**Example:**<br>`RP/0/0/CPU0:router#(config)# interface Bundle-Ether 3` | Creates a new Ethernet link bundle with the specified bundle-id. The range is 1 to 65535.<br><br>This **interface Bundle-Ether** command enters you into the interface configuration submode, where you can enter interface specific configuration commands are entered. Use the **exit** command to exit from the interface configuration submode back to the normal global configuration mode. |
| Step 3 | **`ipv4 address`** *`ipv4-address mask`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# ipv4 address 10.1.2.3 255.0.0.0` | Assigns an IP address and subnet mask to the virtual interface using the **ipv4 address** configuration subcommand.<br><br>**Note** |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **bundle minimum-active bandwidth** *kbps*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# bundle minimum-active bandwidth 580000` | (Optional) Sets the minimum amount of bandwidth required before a user can bring up a bundle. |
| **Step 5** | **bundle minimum-active links** *links*<br><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if)# bundle minimum-active links 2` | (Optional) Sets the number of active links required before you can bring up a specific bundle. |
| **Step 6** | **bundle maximum-active links** *links* [**hot-standby**]<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# bundle maximum-active links 1 hot-standby` | (Optional) Implements 1:1 link protection for the bundle, which causes the highest-priority link in the bundle to become active and the second-highest-priority link to become the standby. Also, specifies that a switchover between active and standby LACP-enabled links is implemented per a proprietary optimization.<br><br>**Note** The Cisco XR 12000 Series Router does not currently support 1:1 link protection. Therefore, the **bundle maximum-active links** command is not supported on the Cisco XR 12000 Series Router.<br><br>**Note** The priority of the active and standby links is based on the value of the **bundle port-priority** command. |
| **Step 7** | **lacp fast-switchover**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# lacp fast-switchover` | (Optional) If you enabled 1:1 link protection (you set the value of the **bundle maximum-active links** command to 1) on a bundle with member links running LACP, you can optionally disable the wait-while timer in the LACP state machine. Disabling this timer causes a bundle member link in standby mode to expedite its normal state negotiations, thereby enabling a faster switchover from a failed active link to the standby link.<br><br>**Note** The Cisco XR 12000 Series Router does not currently support 1:1 link protection. Therefore, the **lacp fast-switchover** command is not supported on the Cisco XR 12000 Series Router. |
| **Step 8** | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# exit` | Exits interface configuration submode for the Ethernet link bundle. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 9** | **interface** {**GigabitEthernet** \| **TenGigE**} *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface GigabitEthernet 1/0/0/0` | Enters interface configuration mode for the specified interface.<br><br>Enter the **GigabitEthernet** or **TenGigE** keyword to specify the interface type. Replace the *interface-path-id* argument with the node-id in the *rack*/*slot*/*module* format. |
| **Step 10** | **bundle id** *bundle-id* [**mode** {**active** \| **on** \| **passive**}]<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# bundle-id 3` | Adds the link to the specified bundle.<br><br>To enable active or passive LACP on the bundle, include the optional **mode active** or **mode passive** keywords in the command string.<br><br>To add the link to the bundle without LACP support, include the optional **mode on** keywords with the command string.<br><br>**Note** If you do not specify the **mode** keyword, the default mode is **on** (LACP is not run over the port). |
| **Step 11** | **bundle port-priority** *priority*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# bundle port-priority 1` | (Optional) If you set the **bundle maximum-active links** command to 1, you must also set the priority of the active link to the highest priority (lowest value) and the standby link to the second-highest priority (next lowest value). For example, you can set the priority of the active link to 1 and the standby link to 2. |
| **Step 12** | **no shutdown**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# no shutdown` | (Optional) If a link is in the down state, bring it up. The **no shutdown** command returns the link to an up or down state depending on the configuration and state of the link. |
| **Step 13** | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# exit` | Exits interface configuration submode for the Ethernet interface. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 14** | **interface** {**GigabitEthernet** \| **TenGigE**} *number*<br>**bundle id** *bundle-id* [**mode** {**active** \| **passive** \| **on**}]<br>**no shutdown**<br>**exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface GigabitEthernet 1/0/2/1`<br><br>`RP/0/0/CPU0:router(config-if)# bundle id 3`<br><br>`RP/0/0/CPU0:router(config-if)# bundle port-priority 2`<br><br>`RP/0/0/CPU0:router(config-if)# no shutdown`<br><br>`RP/0/0/CPU0:router(config-if)# exit`<br><br>`RP/0/0/CPU0:router(config)# interface GigabitEthernet 1/0/2/3`<br><br>`RP/0/0/CPU0:router(config-if)# bundle id 3`<br><br>`RP/0/0/CPU0:router(config-if)# no shutdown`<br><br>`RP/0/0/CPU0:router(config-if)# exit` | (Optional) Repeat Step 8 through Step 11 to add more links to the bundle. |
| **Step 15** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# end`<br>or<br>`RP/0/0/CPU0:router(config-if)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 16** | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# exit` | Exits interface configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 17** | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# exit` | Exits global configuration mode. |
| **Step 18** | Perform Step 1 through Step 15 on the remote end of the connection. | Brings up the other end of the link bundle. |
| **Step 19** | **show bundle Bundle-Ether** *bundle-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router# show bundle Bundle-Ether 3` | (Optional) Shows information about the specified Ethernet link bundle. |
| **Step 20** | **show lacp bundle Bundle-Ether** *bundle-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router# show lacp bundle Bundle-Ether 3` | (Optional) Shows detailed information about LACP ports and their peers. |

# Configuring EFP Load Balancing on an Ethernet Link Bundle

This section describes how to configure Ethernet flow point (EFP) Load Balancing on an Ethernet link bundle.

By default, Ethernet flow point (EFP) load balancing is enabled. However, the user can choose to configure all egressing traffic on the fixed members of a bundle to flow through the same physical member link. This configuration is available only on an Ethernet Bundle subinterface with Layer 2 transport (**l2transport**) enabled.

**Note** If the active members of the bundle change, the traffic for the bundle may get mapped to a different physical link that has a hash value that matches the configured value.

## SUMMARY STEPS

Perform the following steps to configure EFP Load Balancing on an Ethernet link bundle:

1. **configure**
2. **interface Bundle-Ether** *bundle-id* **l2transport**
3. **bundle load-balance hash** *hash-value* [**auto**]
4. **end**
   or
   **commit**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| Step 1 | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| Step 2 | **interface Bundle-Ether** *bundle-id* **l2transport**<br><br>**Example:**<br>`RP/0/0/CPU0:router#(config)# interface Bundle-Ether 3 l2transport` | Creates a new Ethernet link bundle with the specified *bundle-id* and with Layer 2 transport enabled.<br>The range is 1 to 65535. |
| Step 3 | **bundle load-balance hash** *hash-value* [**auto**]<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-subif)# bundle load-balancing hash 1`<br>or<br>`RP/0/0/CPU0:router(config-subif)# bundle load-balancing hash auto` | Configures all egressing traffic on the fixed members of a bundle to flow through the same physical member link.<br>• *hash-value*—Numeric value that specifies the physical member link through which all egressing traffic in this bundle will flow. The values are 1 through 8.<br>• **auto**—The physical member link through which all egressing traffic on this bundle will flow is automatically chosen. |
| Step 4 | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# end`<br>or<br>`RP/0/0/CPU0:router(config-if)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

# Configuring VLAN Bundles

This section describes how to configure a VLAN bundle. The creation of a VLAN bundle involves three main tasks:

1. Create an Ethernet bundle
2. Create VLAN subinterfaces and assign them to the Ethernet bundle.
3. Assign Ethernet links to the Ethernet bundle.

These tasks are describe in detail in the procedure that follows.

**Note** In order for a VLAN bundle to be active, you must perform the same configuration on both ends of the bundle connection.

## Restrictions

The Cisco XR 12000 Series Router does not currently support 1:1 link protection. Therefore, the **bundle maximum-active links** and **lacp fast-switchover** commands are not supported on the Cisco XR 12000 Series Router.

### SUMMARY STEPS

The creation of a VLAN link bundle is described in the steps that follow.

1. **configure**
2. **interface Bundle-Ether** *bundle-id*
3. **ipv4 address** *ipv4-address mask*
4. **bundle minimum-active bandwidth** *kbps*
5. **bundle minimum-active links** *links*
6. **bundle maximum-active links** *links* [**hot-standby**]
7. **lacp fast-switchover**
8. **exit**
9. **interface Bundle-Ether** *bundle-id.vlan-id*
10. **dot1q vlan** *vlan-id*
11. **ipv4 address** *ipv4-address mask*
12. **no shutdown**
13. **exit**
14. Repeat Step 9 through Step 12 to add more VLANS to the bundle you created in Step 2.
15. **end**
    or
    **commit**
16. **exit**
17. **exit**
18. **configure**

19. **interface** {**GigabitEthernet** | **TenGigE**} *interface-path-id*
20. **bundle id** *bundle-id* [**mode** {**active** | **on** | **passive**}
21. **bundle port-priority** *priority*
22. **no shutdown**
23. Repeat Step19 through Step 21 to add more Ethernet Interfaces to the bundle you created in Step 2.
24. **end**
    or
    **commit**
25. Perform Step 1 through Step 23 on the remote end of the connection.
26. **show bundle Bundle-Ether** *bundle-id*
27. **show vlan interface**
28. **show vlan trunks** [{**GigabitEthernet** | **TenGigE** | **Bundle-Ether**} *interface-path-id*] *[***brief** | **summary**] [**location** *node-id]*

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **`configure`**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **`interface Bundle-Ether`** *`bundle-id`*<br><br>**Example:**<br>`RP/0/0/CPU0:router#(config)# interface Bundle-Ether 3` | Creates and names a new Ethernet link bundle.<br>This **interface Bundle-Ether** command enters you into the interface configuration submode, where you can enter interface-specific configuration commands. Use the **exit** command to exit from the interface configuration submode back to the normal global configuration mode. |
| **Step 3** | **`ipv4 address`** *`ipv4-address mask`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# ipv4 address 10.1.2.3 255.0.0.0` | Assigns an IP address and subnet mask to the virtual interface using the **ipv4 address** configuration subcommand. |
| **Step 4** | **`bundle minimum-active bandwidth`** *`kbps`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# bundle minimum-active bandwidth 580000` | (Optional) Sets the minimum amount of bandwidth required before a user can bring up a bundle. |
| **Step 5** | **`bundle minimum-active links`** *`links`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# bundle minimum-active links 2` | (Optional) Sets the number of active links required before you can bring up a specific bundle. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 6** | **bundle maximum-active links** *links* [**hot-standby**]<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# bundle maximum-active links 1 hot-standby` | (Optional) Implements 1:1 link protection for the bundle, which causes the highest-priority link in the bundle to become active and the second-highest-priority link to become the standby. Also, specifies that a switchover between active and standby LACP-enabled links is implemented per a proprietary optimization.<br><br>**Note** The Cisco XR 12000 Series Router does not currently support 1:1 link protection. Therefore, the **bundle maximum-active links** command is not supported on the Cisco XR 12000 Series Router.<br><br>**Note** The priority of the active and standby links is based on the value of the **bundle port-priority** command. |
| **Step 7** | **lacp fast-switchover**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# lacp fast-switchover` | (Optional) If you enabled 1:1 link protection (you set the value of the **bundle maximum-active links** command to 1) on a bundle with member links running LACP, you can optionally disable the wait-while timer in the LACP state machine. Disabling this timer causes a bundle member link in standby mode to expedite its normal state negotiations, thereby enabling a faster switchover from a failed active link to the standby link.<br><br>**Note** The Cisco XR 12000 Series Router does not currently support 1:1 link protection. Therefore, the **lacp fast-switchover** command is not supported on the Cisco XR 12000 Series Router. |
| **Step 8** | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# exit` | Exits the interface configuration submode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 9** | **interface Bundle-Ether** *bundle-id.vlan-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router#(config)# interface Bundle-Ether 3.1` | Creates a new VLAN, and assigns the VLAN to the Ethernet bundle you created in Step 2.<br><br>Replace the *bundle-id* argument with the *bundle-id* you created in Step 2.<br><br>Replace the *vlan-id* with a subinterface identifier. Range is from 1 to 4094 inclusive (0 and 4095 are reserved).<br><br>**Note** When you include the *.vlan-id* argument with the **interface Bundle-Ether** *bundle-id* command, you enter subinterface configuration mode. |
| **Step 10** | **dot1q vlan** *vlan-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router#(config-subif)# dot1q vlan 10` | Assigns a VLAN to the subinterface.<br><br>Replace the *vlan-id* argument with a subinterface identifier. Range is from 1 to 4094 inclusive (0 and 4095 are reserved). |
| **Step 11** | **ipv4 address** *ipv4-address mask*<br><br>**Example:**<br>`RP/0/0/CPU0:router#(config-subif)# ipv4 address 10.1.2.3/24` | Assigns an IP address and subnet mask to the subinterface. |
| **Step 12** | **no shutdown**<br><br>**Example:**<br>`RP/0/0/CPU0:router#(config-subif)# no shutdown` | (Optional) If a link is in the down state, bring it up. The **no shutdown** command returns the link to an up or down state depending on the configuration and state of the link. |
| **Step 13** | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-subif)# exit` | Exits subinterface configuration mode for the VLAN subinterface. |
| **Step 14** | Repeat Step 9 through Step 12 to add more VLANS to the bundle you created in Step 2.<br><br>**interface Bundle-Ether** *bundle-id.vlan-id*<br>**dot1q vlan** *vlan-id*<br><br>**ipv4 address** *ipv4-address mask*<br>**no shutdown**<br>**exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-subif)# interface Bundle-Ether` *3.1*<br>`RP/0/0/CPU0:router(config-subif)# dot1q vlan 20`<br>`RP/0/0/CPU0:router(config-subif)# ipv4 address 20.2.3.4/24`<br>`RP/0/0/CPU0:router(config-subif)# no shutdown`<br>`exit` | (Optional) Adds more subinterfaces to the bundle. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 15** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-subif)# end`<br>or<br>`RP/0/0/CPU0:router(config-subif)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 16** | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-subif)# end` | Exits interface configuration mode. |
| **Step 17** | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# exit` | Exits global configuration mode. |
| **Step 18** | **configure**<br><br>**Example:**<br>`RP/0/RP0/CPU0:router # configure` | Enters global configuration mode. |
| **Step 19** | **interface** {**GigabitEthernet** \| **TenGigE**} *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface GigabitEthernet 1/0/0/0` | Enters interface configuration mode for the Ethernet interface you want to add to the Bundle.<br>Enter the **GigabitEthernet** or **TenGigE** keyword to specify the interface type. Replace the *interface-path-id* argument with the node-id in the rack/slot/module format.<br>**Note** A VLAN bundle is not active until you add an Ethernet interface on both ends of the link bundle. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 20** | **bundle id** *bundle-id* [**mode** {**active** \| **on** \| **passive**}]<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# bundle-id 3` | Adds an Ethernet interface to the bundle you configured in Step 2 through Step 13.<br><br>To enable active or passive LACP on the bundle, include the optional **mode active** or **mode passive** keywords in the command string.<br><br>To add the interface to the bundle without LACP support, include the optional **mode on** keywords with the command string.<br><br>**Note** If you do not specify the **mode** keyword, the default mode is **on** (LACP is not run over the port). |
| **Step 21** | **bundle port-priority** *priority*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# bundle port-priority 1` | (Optional) If you set the **bundle maximum-active links** command to 1, you must also set the priority of the active link to the highest priority (lowest value) and the standby link to the second-highest priority (next lowest value). For example, you can set the priority of the active link to 1 and the standby link to 2. |
| **Step 22** | **no shutdown**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# no shutdown` | (Optional) If a link is in the down state, bring it up. The **no shutdown** command returns the link to an up or down state depending on the configuration and state of the link. |
| **Step 23** | — | Repeat Step 19 through Step 21 to add more Ethernet interfaces to the VLAN bundle. |
| **Step 24** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-subif)# end`<br>or<br>`RP/0/0/CPU0:router(config-subif)# commit` | Saves configuration changes.<br><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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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><br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 25** | Perform Step 1 through Step 23 on the remote end of the VLAN bundle connection. | Brings up the other end of the link bundle. |
| **Step 26** | **`show bundle Bundle-Ether`** *`bundle-id`*<br><br>**Example:**<br>`RP/0/0/CPU0:router# show bundle Bundle-Ether 3` | (Optional) Shows information about the specified Ethernet link bundle.<br><br>The **show bundle Bundle-Ether** command displays information about the specified bundle. If your bundle has been configured properly and is carrying traffic, the State field in the **show bundle Bundle-Ether** command output shows the number "4," which means the specified VLAN bundle port is "distributing." |
| **Step 27** | **`show vlan interface`**<br><br>**Example:**<br>`RP/0/0/CPU0:router # show vlan interface` | Displays the current VLAN interface and status configuration. |
| **Step 28** | **`show vlan trunks`** `[{`**`GigabitEthernet`** `\|` **`TenGigE`** `\|` **`Bundle-Ether`**`}` *`interface-path-id`*`] [`**`brief`** `\|` **`summary`**`] [`**`location`** *`node-id`*`]`<br><br>**Example:**<br>`RP/0/0/CPU0:router# show vlan trunk summary` | (Optional) Displays summary information about each of the VLAN trunk interfaces.<br><br>• The keywords have the following meanings:<br>– **brief—**Displays a brief summary.<br>– **summary—**Displays a full summary.<br>– **location—**Displays information about the VLAN trunk interface on the given slot.<br>– **interface—**Displays information about the specified interface or subinterface.<br><br>Use the **show vlan trunks** command to verify that all configured VLAN subinterfaces on an Ethernet bundle are "up." |
| **Step 29** | **`lacp fast-switchover`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# lacp fast-switchover` | (Optional) If you enabled 1:1 link protection (you set the value of the **bundle maximum-active links** command to 1) on a bundle with member links running LACP, you can optionally disable the wait-while timer in the LACP state machine. Disabling this timer causes a bundle member link in standby mode to expedite its normal state negotiations, thereby enabling a faster switchover from a failed active link to the standby link. |

# Configuring POS Link Bundles

This section describes how to configure a POS link bundle.

**Note** In order for a POS bundle to be active, you must perform the same configuration on both connection endpoints of the POS bundle.

## SUMMARY STEPS

The creation of a bundled POS interface involves configuring both the bundle and the member interfaces, as shown in these steps:

1. **configure**
2. **interface Bundle-POS** *bundle-id*
3. **ipv4 address** *ipv4-address mask*
4. **bundle minimum-active bandwidth** *kbps*
5. **bundle minimum-active links** *links*
6. **bundle maximum-active links** *links* [**hot-standby**]
7. **lacp fast-switchover**
8. **exit**
9. **interface POS** *interface-path-id*
10. **bundle id** *bundle-id* [**mode** {**active** | **on** | **passive**}]
11. **bundle port-priority** *priority*
12. **no shutdown**
13. **exit**
14. Repeat Step 8 through Step 11 to add more links to the bundle you created in Step 2.
15. **end**
    or
    **commit**
16. **exit**
17. **exit**
18. Perform Step 1 through Step 15 on the remote end of the connection.
19. **show bundle Bundle-POS** *bundle-id*
20. **show lacp bundle bundle-POS** *bundle-id*

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | `configure`<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | **`interface Bundle-POS`** *`bundle-id`*<br><br>**Example:**<br>`RP/0/0/CPU0:router#(config)#interface Bundle-POS 2` | Configures and names the new bundled POS interface.<br><br>Enters the interface configuration submode, from where interface specific configuration commands are executed. Use the **exit** command to exit from the interface configuration submode, and get back to the normal global configuration mode. |
| **Step 3** | **`ipv4 address`** *`ipv4-address mask`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# ipv4 address 10.1.2.3 255.0.0.0` | Assigns an IP address and subnet mask to the virtual interface using the ip address configuration subcommand. |
| **Step 4** | **`bundle minimum-active bandwidth`** *`kbps`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# bundle minimum-active bandwidth 620000` | (Optional) Sets the minimum amount of bandwidth required before a user can bring up a bundle. |
| **Step 5** | **`bundle minimum-active links`** *`links`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# bundle minimum-active links 2` | (Optional) Sets the number of active links required before you can bring up a specific bundle. |
| **Step 6** | **`bundle maximum-active links`** *`links`* [**`hot-standby`**]<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# bundle maximum-active links 1 hot-standby` | (Optional) Implements 1:1 link protection for the bundle, which causes the highest-priority link in the bundle to become active and the second-highest-priority link to become the standby. Also, specifies that a switchover between active and standby LACP-enabled links is implemented according to a proprietary optimization.<br><br>**Note** The priority of the active and standby links is based on the value of the **bundle port-priority** command. |
| **Step 7** | **`lacp fast-switchover`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# lacp fast-switchover` | (Optional) If you enabled 1:1 link protection (you set the value of the **bundle maximum-active links** command to 1) on a bundle with member links running LACP, you can optionally disable the wait-while timer in the LACP state machine. Disabling this timer causes a bundle member link in standby mode to expedite its normal state negotiations, thereby enabling a faster switchover from a failed active link to the standby link. |
| **Step 8** | **`exit`** | Exits the interface configuration submode. |
| **Step 9** | **`interface POS`** *`interface-path-id`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface POS 0/1/0/0` | Enters POS interface configuration mode and specifies the POS interface name and interface-path-id notation *rack/slot/module/port*. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 10** | **bundle id** *bundle-id* [**mode** {**active** \| **on** \| **passive**}]<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# bundle-id 3` | Adds the link to the specified bundle.<br><br>To enable active or passive LACP on the bundle, include the optional **mode active** or **mode passive** keywords in the command string.<br><br>To add the link to the bundle without LACP support, include the optional **mode on** keywords with the command string.<br><br>**Note** If you do not specify the **mode** keyword, the default mode is **on** (LACP is not run over the port). |
| **Step 11** | **bundle port-priority** *priority*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# bundle port-priority 1` | (Optional) If you set the **bundle maximum-active links** command to 1, you must also set the priority of the active link to the highest priority (lowest value) and the standby link to the second-highest priority (next lowest value). For example, you can set the priority of the active link to 1 and the standby link to 2. |
| **Step 12** | **no shutdown**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# no shutdown` | Removes the shutdown configuration which forces the interface administratively down. The **no shutdown** command then returns the link to an up or down state, depending on the configuration and state of the link. |
| **Step 13** | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router# exit` | Exits the interface configuration submode for the POS interface. |
| **Step 14** | Repeat Step 8 through Step 11 to add more links to a bundle | (Optional) Adds more links to the bundle you created in Step 2. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 15** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# end`<br>or<br>`RP/0/0/CPU0:router(config-if)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 16** | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# exit` | Exits interface configuration mode. |
| **Step 17** | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# exit` | Exits global configuration mode. |
| **Step 18** | Perform Step 1 through Step 15 on the remote end of the connection. | Brings up the other end of the link bundle. |
| **Step 19** | **show bundle Bundle-POS** *number*<br><br>**Example:**<br>`RP/0/0/CPU0:router# show bundle Bundle-POS 1` | (Optional) Shows information about the specified POS link bundle. |
| **Step 20** | **show lacp bundle Bundle-POS** *bundle-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router# show lacp bundle Bundle-POS 3` | (Optional) Shows detailed information about LACP ports and their peers. |

# Configuring the Default LACP Short Period Time Interval

This section describes how to configure the default short period time interval for sending and receiving LACP packets on a Gigabit Ethernet interface. This procedure also enables the LACP short period.

## SUMMARY STEPS

To enable an LACP short period time interval, using the default time of 1 second, perform the following steps.

1. **configure**
2. **interface GigabitEthernet** *interface-path*
3. **bundle id** *number* **mode active**
4. **lacp period short**
5. **commit**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **`configure`**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **`interface GigabitEthernet`** *`interface-path`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface GigabitEthernet 0/0/0/1` | Creates a Gigabit Ethernet interface and enters interface configuration mode. |
| **Step 3** | **`bundle id`** *`number`* **`mode active`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# bundle id 1 mode active` | Specifies the bundle interface and puts the member interface in active mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **lacp period short**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# lacp period short` | Configures a short period time interval for the sending and receiving of LACP packets, using the default time period of 1000 milliseconds or 1 second. |
| **Step 5** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# end`<br>or<br>`RP/0/0/CPU0:router(config-if)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

## Configuring Custom LACP Short Period Time Intervals

This section describes how to configure custom short period time intervals (less than 1000 milliseconds) for sending and receiving LACP packets on a Gigabit Ethernet interface.

**Note** You must always configure the *transmit* interval at both ends of the connection before you configure the *receive* interval at either end of the connection. Failure to configure the *transmit* interval at both ends first results in route flapping (a route going up and down continuously). When you remove a custom LACP short period, you must do it in reverse order. You must remove the *receive* intervals first and then the *transmit* intervals.

### SUMMARY STEPS

To configure custom receive and transmit intervals for LACP packets, perform the following steps.

**Router A**

1. **configure**
2. **interface GigabitEthernet** *interface-path*
3. **bundle id** *number* **mode active**
4. **lacp period short**
5. **commit**

**Router B**

6. **configure**
7. **interface GigabitEthernet** *interface-path*
8. **bundle id** *number* **mode active**
9. **lacp period short**
10. **commit**

**Router A**

11. **configure**
12. **interface GigabitEthernet** *interface-path*
13. **lacp period short transmit** *interval*
14. **commit**

**Router B**

15. **configure**
16. **interface GigabitEthernet** *interface-path*
17. **lacp period short transmit** *interval*
18. **commit**

**Router A**

19. **configure**
20. **interface GigabitEthernet** *interface-path*

21. **lacp period short receive** *interval*
22. **commit**

**Router B**

23. **configure**
24. **interface GigabitEthernet** *interface-path*
25. **lacp period short receive** *interval*
26. **commit** or **end**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **`configure`**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **`interface Bundle-Ether`** *`bundle-id`*<br><br>**Example:**<br>`RP/0/0/CPU0:router#(config)# interface Bundle-Ether 3` | Creates and names a new Ethernet link bundle.<br><br>This **interface Bundle-Ether** command enters you into the interface configuration submode, where you can enter interface-specific configuration commands. Use the **exit** command to exit from the interface configuration submode back to the normal global configuration mode. |
| **Step 3** | **`ipv4 address`** *`ipv4-address mask`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# ipv4 address 10.1.2.3 255.0.0.0` | Assigns an IP address and subnet mask to the virtual interface using the **ipv4 address** configuration subcommand. |
| **Step 4** | **`bundle minimum-active bandwidth`** *`kbps`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# bundle minimum-active bandwidth 580000` | (Optional) Sets the minimum amount of bandwidth required before a user can bring up a bundle. |
| **Step 5** | **`bundle minimum-active links`** *`links`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# bundle minimum-active links 2` | (Optional) Sets the number of active links required before you can bring up a specific bundle. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 6** | **bundle maximum-active links** *links*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# bundle maximum-active links 1` | (Optional) Designates one active link and one link in standby mode that can take over immediately for a bundle if the active link fails (1:1 protection).<br>**Note** The default number of active links allowed in a single bundle is 8.<br>**Note** If the **bundle maximum-active** command is issued, then only the highest-priority link within the bundle is active. The priority is based on the value from the **bundle port-priority** command, where a lower value is a higher priority. Therefore, we recommend that you configure a higher priority on the link that you want to be the active link. |
| **Step 7** | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# exit` | Exits the interface configuration submode. |
| **Step 8** | **interface Bundle-Ether** *bundle-id.vlan-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router#(config)# interface Bundle-Ether 3.1` | Creates a new VLAN, and assigns the VLAN to the Ethernet bundle you created in Step 2.<br>Replace the *bundle-id* argument with the *bundle-id* you created in Step 2.<br>Replace the *vlan-id* with a subinterface identifier. Range is from 1 to 4094 inclusive (0 and 4095 are reserved).<br>**Note** When you include the *.vlan-id* argument with the **interface Bundle-Ether** *bundle-id* command, you enter subinterface configuration mode. |
| **Step 9** | **dot1q vlan** *vlan-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router#(config-subif)# dot1q vlan 10` | Assigns a VLAN 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). |
| **Step 10** | **ipv4 address** *ipv4-address mask*<br><br>**Example:**<br>`RP/0/0/CPU0:router#(config-subif)# ipv4 address 10.1.2.3/24` | Assigns an IP address and subnet mask to the subinterface. |
| **Step 11** | **no shutdown**<br><br>**Example:**<br>`RP/0/0/CPU0:router#(config-subif)# no shutdown` | (Optional) If a link is in the down state, bring it up. The **no shutdown** command returns the link to an up or down state depending on the configuration and state of the link. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 12** | **`exit`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-subif)# exit` | Exits subinterface configuration mode for the VLAN subinterface. |
| **Step 13** | Repeat Step 7 through Step 12 to add more VLANs to the bundle you created in Step 2. | (Optional) Adds more subinterfaces to the bundle. |
| **Step 14** | **`end`**<br>or<br>**`commit`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-subif)# end`<br>or<br>`RP/0/0/CPU0:router(config-subif)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 15** | **`exit`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-subif)# exit` | Exits interface configuration mode. |
| **Step 16** | **`exit`**<br><br>**Example:**<br>`RP/0/RSP0/CPU0:router(config)# exit` | Exits global configuration mode. |
| **Step 17** | **`show ethernet trunk bundle-ether`** *`instance`*<br><br>**Example:**<br>`RP/0/0/CPU0:router# show ethernet trunk bundle-ether 5` | (Optional) Displays the interface configuration.<br>The Ethernet bundle instance range is from 1 through 65535. |
| **Step 18** | **`configure`**<br><br>**Example:**<br>`RP/0/0/CPU0:router # configure` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| Step 19 | **interface** {**GigabitEthernet** \| **TenGigE**} *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface GigabitEthernet 1/0/0/0` | Enters the interface configuration mode for the Ethernet interface you want to add to the Bundle.<br><br>Enter the **GigabitEthernet** or **TenGigE** keyword to specify the interface type. Replace the *interface-path-id* argument with the node-id in the *rack/slot/module* format.<br><br>**Note** A VLAN bundle is not active until you add an Ethernet interface on both ends of the link bundle. |
| Step 20 | **bundle id** *bundle-id* [**mode** {**active** \| **on** \| **passive**}]<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# bundle-id 3` | Adds an Ethernet interface to the bundle you configured in Step 2 through Step 13.<br><br>To enable active or passive LACP on the bundle, include the optional **mode active** or **mode passive** keywords in the command string.<br><br>To add the interface to the bundle without LACP support, include the optional **mode on** keywords with the command string.<br><br>**Note** If you do not specify the **mode** keyword, the default mode is **on** (LACP is not run over the port). |
| Step 21 | **no shutdown**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# no shutdown` | (Optional) If a link is in the down state, bring it up. The **no shutdown** command returns the link to an up or down state depending on the configuration and state of the link. |
| Step 22 | Repeat Step 19 through Step 21 to add more Ethernet interfaces to the VLAN bundle. | — |

| | Command or Action | Purpose |
|---|---|---|
| **Step 23** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-subif)# end`<br>or<br>`RP/0/0/CPU0:router(config-subif)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 24** | Perform Step 1 through Step 23 on the remote end of the VLAN bundle connection. | Brings up the other end of the link bundle. |
| **Step 25** | **show bundle Bundle-Ether** *bundle-id* [**reasons**]<br><br>**Example:**<br>`RP/0/0/CPU0:router# show bundle Bundle-Ether 3 reasons` | (Optional) Shows information about the specified Ethernet link bundle.<br>The **show bundle Bundle-Ether** command displays information about the specified bundle. If your bundle has been configured properly and is carrying traffic, the State field in the **show bundle Bundle-Ether** command output will show the number "4," which means the specified VLAN bundle port is "distributing." |
| **Step 26** | **show ethernet trunk bundle-ether** *instance*<br><br>**Example:**<br>`RP/0/0/CPU0:router# show ethernet trunk bundle-ether 5` | (Optional) Displays the interface configuration.<br>The Ethernet bundle instance range is from 1 through 65535. |

| | Command or Action | Purpose |
|---|---|---|
| | **Router A** | |
| **Step 1** | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **interface GigabitEthernet** *interface-path*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface GigabitEthernet 0/0/0/1` | Creates a Gigabit Ethernet interface and enters interface configuration mode. |
| **Step 3** | **bundle id** *number* **mode active**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# bundle id 1 mode active` | Specifies the bundle interface and puts the member interface in active mode. |
| **Step 4** | **lacp period short**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# lacp period short` | Enables the short period time interval. |
| **Step 5** | **commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# commit` | Saves configuration changes and exits to EXEC mode. |
| | **Router B** | |
| **Step 6** | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 7** | **interface GigabitEthernet** *interface-path*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface GigabitEthernet 0/0/0/1` | Creates a Gigabit Ethernet interface and enters interface configuration mode. |
| **Step 8** | **bundle id** *number* **mode active**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# bundle id 1 mode active` | Specifies the bundle interface and puts the member interface in active mode. |
| **Step 9** | **lacp period short**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# lacp period short` | Enables the short period time interval. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 10** | **commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# commit` | Saves configuration changes and exits to EXEC mode. |
| | **Router A** | |
| **Step 11** | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 12** | **interface GigabitEthernet** *interface-path*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface GigabitEthernet 0/0/0/1` | Creates a Gigabit Ethernet interface and enters interface configuration mode at one end of the connection. |
| **Step 13** | **lacp period short transmit** *interval*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# lacp period short transmit 500` | Configures the short period transmit time interval for LACP packets at one end of the connection.<br><br>Valid values are 100 to 1000 milliseconds in multiples of 100, such as 100, 200, 300, and so on. |
| **Step 14** | **commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# commit` | Saves configuration changes and exits to EXEC mode. |
| | **Router B** | |
| **Step 15** | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 16** | **interface GigabitEthernet** *interface-path*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface GigabitEthernet 0/0/0/1` | Creates a Gigabit Ethernet interface and enters interface configuration mode at one end of the connection. |
| **Step 17** | **lacp period short transmit** *interval*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# lacp period short transmit 500` | Configures the short period transmit time interval for LACP packets at one end of the connection.<br><br>Valid values are 100 to 1000 milliseconds in multiples of 100, such as 100, 200, 300, and so on. |
| **Step 18** | **commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# commit` | Saves configuration changes and exits to EXEC mode. |
| | **Router A** | |

| | Command or Action | Purpose |
|---|---|---|
| Step 19 | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| Step 20 | **interface GigabitEthernet** *interface-path*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface GigabitEthernet 0/0/0/1` | Creates a Gigabit Ethernet interface and enters interface configuration mode at one end of the connection. |
| Step 21 | **lacp period short receive** *interval*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# lacp period short receive 500` | Configures the short period receive time interval for LACP packets at one end of the connection.<br>Valid values are 100 to 1000 milliseconds in multiples of 100, such as 100, 200, 300, and so on. |
| Step 22 | **commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# commit` | Saves configuration changes and exits to EXEC mode. |
| | **Router B** | |
| Step 23 | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 24** | **interface GigabitEthernet** *interface-path*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface GigabitEthernet 0/0/0/1` | Creates a Gigabit Ethernet interface and enters interface configuration mode at one end of the connection. |
| **Step 25** | **lacp period short receive** *interval*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# lacp period short receive 500` | Configures the short period receive time interval for LACP packets at one end of the connection.<br><br>Valid values are 100 to 1000 milliseconds in multiples of 100, such as 100, 200, 300, and so on. |
| **Step 26** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# end`<br>or<br>`RP/0/0/CPU0:router(config-if)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

# Configuration Examples for Link Bundling

This section contains the following examples:


- Example: Configuring an Ethernet Link Bundle, page 246
- Example: Configuring a VLAN Link Bundle, page 246
- Example: Configuring a POS Link Bundle, page 247
- Example: Configuring EFP Load Balancing on an Ethernet Link Bundle, page 247
- Examples: Configuring LACP Short Periods, page 248
- Example: Configuring MPLS-TE and FRR over Link Bundles, page 249

# Example: Configuring an Ethernet Link Bundle

The following example shows how to join two ports to form an EtherChannel bundle running LACP:

**Note** The Cisco XR 12000 Series Router does not currently support 1:1 link protection. Therefore, the **bundle maximum-active links** and **lacp fast-switchover** commands are not supported on the Cisco XR 12000 Series Router.

```
RP/0/RP0/CPU0:Router# config
RP/0/0/CPU0:Router(config)# interface Bundle-Ether 3
RP/0/RP0/CPU0:Router(config-if)# ipv4 address 1.2.3.4/24
RP/0/RP0/CPU0:Router(config-if)# bundle minimum-active bandwidth 620000
RP/0/RP0/CPU0:Router(config-if)# bundle minimum-active links 1
RP/0/0/CPU0:Router(config-if)# bundle maximum-active links 1 hot-standby
RP/0/0/CPU0:Router(config-if)# lacp fast-switchover
RP/0/0/CPU0:Router(config-if)# exit
RP/0/0/CPU0:Router(config)# interface TenGigE 0/3/0/0
RP/0/0/CPU0:Router(config-if)# bundle id 3 mode active
RP/0/0/CPU0:Router(config-if)# bundle port-priority 1
RP/0/0/CPU0:Router(config-if)# no shutdown
RP/0/0/CPU0:Router(config)# exit
RP/0/0/CPU0:Router(config)# interface TenGigE 0/3/0/1
RP/0/0/CPU0:Router(config-if)# bundle id 3 mode active
RP/0/0/CPU0:Router(config-if)# bundle port-priority 2
RP/0/0/CPU0:Router(config-if)# no shutdown
RP/0/0/CPU0:Router(config-if)# exit
```

# Example: Configuring a VLAN Link Bundle

The following example shows how to create and bring up two VLANS on an Ethernet bundle:

```
RP/0/RP0/CPU0:Router# config
RP/0/RP0/CPU0:Router(config)# interface Bundle-Ether 1
RP/0/RP0/CPU0:Router(config-if)# ipv4 address 1.2.3.4/24
RP/0/RP0/CPU0:Router(config-if)# bundle minimum-active bandwidth 620000
RP/0/RP0/CPU0:Router(config-if)# bundle minimum-active links 1
RP/0/RP0/CPU0:Router(config-if)# exit
RP/0/RP0/CPU0:Router(config)# interface Bundle-Ether 1.1
RP/0/0/CPU0:Router(config-subif)# dot1q vlan 10
RP/0/0/CPU0:Router(config-subif)# ip addr 10.2.3.4/24
RP/0/0/CPU0:Router(config-subif)# no shutdown
RP/0/0/CPU0:Router(config-subif)# exit
RP/0/0/CPU0:Router(config)# interface Bundle-Ether 1.2
RP/0/0/CPU0:Router(config-subif)# dot1q vlan 20
RP/0/RP0/CPU0:Router(config-subif)# ip addr 20.2.3.4/24
RP/0/0/CPU0:Router(config-subif)# no shutdown
RP/0/0/CPU0:Router(config-subif)# exit
RP/0/0/CPU0:Router(config)#interface gig 0/1/5/7
RP/0/0/CPU0:Router(config-if)# bundle-id 1 mode act
RP/0/0/CPU0:Router(config-if)# commit
RP/0/0/CPU0:Router(config-if)# exit
RP/0/0/CPU0:Router(config)# exit
RP/0/0/CPU0:Router # show vlan trunks
```

# Example: Configuring a POS Link Bundle

The following example shows how to join two ports to form a Packet-over-SONET (POS) link bundle:

```
RP/0/0/CPU0:Router# config
RP/0/0/CPU0:Router(config)# interface Bundle-POS 5
RP/0/0/CPU0:Router(config-if)# ipv4 address 1.2.3.4/24
RP/0/0/CPU0:Router(config-if)# bundle minimum-active bandwidth 620000
RP/0/0/CPU0:Router(config-if)# bundle minimum-active bandwidth 620000
RP/0/0/CPU0:Router(config-if)# exit
RP/0/0/CPU0:Router(config)# interface POS 0/0/1/1
RP/0/0/CPU0:Router(config-if)# bundle id 5
RP/0/0/CPU0:Router(config-if)# no shutdown
RP/0/0/CPU0:Router(config-if)# exit
```

# Example: Configuring EFP Load Balancing on an Ethernet Link Bundle

The following example shows how to configure all egressing traffic on the fixed members of a bundle to flow through the same physical member link automatically.

```
RP/0/RP0/CPU0:router# configuration terminal
RP/0/RP0/CPU0:router(config)# interface bundle-ether 1.1 l2transport
RP/0/RP0/CPU0:router(config-subif)# bundle load-balancing hash auto
RP/0/RP0/CPU0:router(config-subif)#
```

The following example shows how to configure all egressing traffic on the fixed members of a bundle to flow through a specified physical member link.

```
RP/0/RP0/CPU0:router# configuration terminal
RP/0/RP0/CPU0:router(config)# interface bundle-ether 1.1 l2transport
RP/0/RP0/CPU0:router(config-subif)# bundle load-balancing hash 1
RP/0/RP0/CPU0:router(config-subif)#
```

# Examples: Configuring LACP Short Periods

The following example shows how to configure the LACP short period time interval to the default time of 1000 milliseconds (1 second):

```
config
interface gigabitethernet 0/0/0/1
   bundle id 1 mode active
   lacp period short
   commit
```

The following example shows how to configure custom LACP short period transmit and receive intervals to *less than* the default of 1000 milliseconds (1 second):

### Router A

```
config
interface gigabitethernet 0/0/0/1
   bundle id 1 mode active
   lacp period short
   commit
```

### Router B

```
config
interface gigabitethernet 0/0/0/1
   bundle id 1 mode active
   lacp period short
   commit
```

### Router A

```
config
interface gigabitethernet 0/0/0/1
    lacp period short transmit 100
   commit
```

### Router B

```
config
interface gigabitethernet 0/0/0/1
    lacp period short transmit 100
   commit
```

### Router A

```
config
interface gigabitethernet 0/0/0/1
   lacp period short receive 100
   commit
```

### Router B

```
config
interface gigabitethernet 0/0/0/1
   lacp period short receive 100
   commit
```

# Example: Configuring MPLS-TE and FRR over Link Bundles

The following example shows how to configure MPLS-TE and FRR on a TE1 tunnel:

```
config
   interface tunnel-te1
      ipv4 unnumbered Loopback0
      autoroute announce
      destination 10.10.10.10
      ! For PBTS: All traffic which has exp set to 2, will take this tunnel.
      policy-class 2
      ! For FRR
      fast-reroute
      record-route
      path-option 1 explicit name PRIMARY
     !
```

The following example shows how to configure MPLS-TE and FRR on an Ether bundle:

```
config
   interface Bundle-Ether2
      mtu 1500
      ipv4 address 11.1.1.1 255.255.255.0
      bundle minimum-active links 2
      bundle minimum-active bandwidth 2000000
     !
  interface GigabitEthernet0/3/0/2
      bundle id 2 mode active
     !
     interface GigabitEthernet0/3/0/3
      bundle id 2 mode active
     !
```

**Note** To trigger FRR using the minimum link threshold, use the **bundle minimum-active links** command and the **bundle minimum-active bandwidth** command in interface submode.

The following example shows how to configure a backup tunnel configuration on PLR:

```
config
    interface tunnel-te2
! backup tunnel configuration on PLR
        ipv4 unnumbered Loopback0
        autoroute announce
        destination 31.0.0.1
!
    mpls traffic-eng
        interface Bundle-Ether1   <=Protected interface.
            backup-path tunnel-te 2  <= Backup tunnel
!
```

# Additional References

These sections provide references related to link bundle configuration.

## Related Documents

| Related Topic | Document Title |
|---|---|
| Cisco IOS XR master command reference | *Cisco IOS XR Master Commands List* |
| Cisco IOS XR interface configuration commands | *Cisco IOS XR Interface and Hardware Component Command Reference* |
| Initial system bootup and configuration information for a router using the Cisco IOS XR software. | *Cisco IOS XR Getting Started Guide* |
| Information about user groups and task IDs | *Cisco IOS XR Interface and Hardware Component Command Reference* |

## Standards

| Standards | Title |
|---|---|
| IEEE 802.3ad (incorporated as Annex 43 into 802.3-2002) | — |

## MIBs

| MIBs | MIBs Link |
|---|---|
| The IEEE-defined MIB for Link Aggregation (defined in 802.3 Annex 30C) | To locate and download MIBs for selected platforms using Cisco IOS XR Software, use the Cisco MIB Locator found at the following URL:<br>http://www.cisco.com/go/mibs |
| MPLS TE MIB | — |

## RFCs

| RFCs | Title |
|---|---|
| No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature. | — |
| RFC 3812 | — |

# 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/support |

# Configuring Virtual Loopback and Null Interfaces on Cisco IOS XR Software

This module describes the configuration of loopback and null interfaces on the Cisco XR 12000 Series Router.

Loopback and null interfaces are considered virtual interfaces.

A virtual interface represents a logical packet switching entity within the router. Virtual Interfaces have a global scope and do not have an associated location. Virtual interfaces have instead a globally unique numerical ID after their names. Examples are Loopback 0, Loopback1, and Loopback 99999. The ID is unique per virtual interface type to make the entire name string unique such that you can have both Loopback 0 and Null 0.

Loopback and null interfaces have their control plane presence on the active route processor (RP). The configuration and control plane are mirrored onto the standby RP and, in the event of a switchover, the virtual interfaces move to the ex-standby, which then becomes the newly active RP.

**Feature History for Configuring Loopback and Null Interfaces on Cisco IOS XR Software**

| Release | Modification |
|---|---|
| Release 3.3.0 | This feature was introduced on the Cisco XR 12000 Series Router. |
| Release 3.4.0 | This module was updated to include information on configuring virtual IPV4 management interfaces. |

# Contents


- Prerequisites for Configuring Virtual Interfaces, page 254
- Information About Configuring Virtual Interfaces, page 254
- How to Configure Virtual Interfaces, page 255
- Configuration Examples for Virtual Interfaces, page 259
- Additional References, page 261

# Prerequisites for Configuring Virtual 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.

# Information About Configuring Virtual Interfaces

To configure virtual interfaces, you must understand the following concepts:


- Virtual Loopback Interface Overview, page 254
- Null Interface Overview, page 254
- Virtual Management Interface Overview, page 255
- Active and Standby RPs and Virtual Interface Configuration, page 255


## Virtual Loopback Interface Overview

A virtual loopback interface is a virtual interface with a single endpoint that is always up. Any packet transmitted over a virtual loopback interface is immediately received by the selfsame interface. Loopback interfaces emulate a physical interface.

In Cisco IOS XR software,virtual loopback interfaces perform the following functions:

- loopback interfaces can act as a termination address for routing protocol sessions. This allows routing protocol sessions to stay up even if the outbound interface is down.
- you can ping the loopback interface to verify that the router IP stack is working properly.

In applications where other routers or access servers attempt to reach a virtual loopback interface, you must configure a routing protocol to distribute the subnet assigned to the loopback address.

Packets routed to the loopback interface are rerouted back to the router or access server and processed locally. IP packets routed out the loopback interface but not destined to the loopback interface are dropped. Under these two conditions, the loopback interface can behave like a null interface.

## Null Interface Overview

A null interface functions similarly to the null devices available on most operating systems. This interface is always up and can never forward or receive traffic; encapsulation always fails. The null interface provides an alternative method of filtering traffic. You can avoid the overhead involved with using access lists by directing undesired network traffic to the null interface.

The only interface configuration command that you can specify for the null interface is the **ipv4 unreachables** command. With the **ipv4 unreachables** command, if the software receives a nonbroadcast packet destined for itself that uses a protocol it does not recognize, it sends an Internet Control Message Protocol (ICMP) protocol unreachable message to the source. If the software receives a datagram that it cannot deliver to its ultimate destination because it knows of no route to the destination address, it replies to the originator of that datagram with an ICMP host unreachable message.

The Null0 interface is created by default on the RP during boot and cannot be removed. The **ipv4 unreachables** command can be configured for this interface, but most configuration is unnecessary because this interface just discards all the packets sent to it.

The Null0 interface can be displayed with the **show interfaces null0** command.

## Virtual Management Interface Overview

Configuring an IPv4 virtual address enables you to access the router from a single virtual address with a management network without prior knowledge of which RP is active. An IPv4 virtual address persists across route processor (RP) switchover situations. For this to happen, the virtual IPv4 address must share a common IPv4 subnet with a management Ethernet interface on both RPs.

On a Cisco XR 12000 Series Router where each RP has multiple management Ethernet interfaces, the virtual IPv4 address maps to the management Ethernet interface on the active RP that shares the same IP subnet.

## Active and Standby RPs and Virtual Interface Configuration

The standby RP is available and in a state in which it can take over the work from the active RP should that prove necessary. Conditions that necessitate the standby RP to become the active RP and assume the active RP's duties include:

- Failure detection by a watchdog
- Administrative command to take over
- Removal of the active RP from the chassis

If a second RP is not present in the chassis while the first is in operation, a second RP may be inserted and automatically becomes the standby RP. The standby RP may also be removed from the chassis with no effect on the system other than loss of RP redundancy.

After switchover, the virtual interfaces  all are present on the standby (now active) RP. Their state and configuration are unchanged and there has been no loss of forwarding (in the case of tunnels) over the interfaces during the switchover. The routers use nonstop forwarding (NSF) over bundles and tunnels through the switchover of the host RP.

**Note** The user need not configure anything to guarantee that the standby interface configurations are maintained.

**Note** Protocol configuration such as tacacs source-interface, snmp-server trap-source, ntp source, logging source-interface do not use the virtual management IP address as their source by default. Use the **ipv4 virtual address use-as-src-addr** command to ensure that the protocol uses the virtual IPv4 address as its source address. Alternatively, you can also configure a loopback address with the designated or desired IPv4 address and set that as the source for protocols such as TACACS+ using the **tacacs source-interface** command.

# How to Configure Virtual Interfaces

This section contains the following procedures:

- Configuring Virtual Loopback Interfaces, page 256 (Required)
- Configuring Null Interfaces, page 257 (Required)
- Configuring Virtual IPV4 Interfaces, page 258 (Required)

## Configuring Virtual Loopback Interfaces

This task explains how to configure a basic loopback interface.

### Restrictions

The IP address of a loopback interface must be unique across all routers on the network. It must not be used by another interface on the router, and it must not be used by an interface on any other router on the network.

#### SUMMARY STEPS

1. **configure**
2. **interface loopback** *interface-path-id*
3. **ipv4 address** *ip-address*
4. **end**
   or
   **commit**
5. **show interfaces** *type interface-path-id*

#### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **interface loopback** *interface-path-id*<br><br>Example:<br>`RP/0/0/CPU0:router#(config)# interface Loopback 3` | Enters interface configuration mode and names the new loopback interface. |
| **Step 3** | **ipv4 address** *ip-address*<br><br>**Example:**<br>`RP/0/00/CPU0:router(config-if)# ipv4 address 172.18.189.38/32` | Assigns an IP address and subnet mask to the virtual loopback interface using the **ipv4 address** configuration command. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# end`<br>or<br>`RP/0/0/CPU0:router(config-if)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 interfaces** *type interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router# show interfaces Loopback 3` | (Optional) Displays the configuration of the loopback interface. |

## Configuring Null Interfaces

This task explains how to configure a basic Null interface.

### SUMMARY STEPS

1. **configure**
2. **interface null 0**
3. **end**
   or
   **commit**
4. **show interfaces** *type interface-path-id*

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **interface null 0**<br><br>**Example:**<br>`RP/0/0/CPU0:router#(config)# interface null 0` | Enters null0 interface configuration mode. |
| **Step 3** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-null0)# end`<br>or<br>`RP/0/0/CPU0:router(config-null0)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 4** | **show interfaces null 0**<br><br>**Example:**<br>`RP/0/0/CPU0:router# show interfaces null0` | Verifies the configuration of the null interface. |

# Configuring Virtual IPV4 Interfaces

This task explains how to configure an IPv4 virtual interface.

### SUMMARY STEPS

1. **configure**
2. **ipv4 address virtual address** *ip-address subnet mask*
3. **end**
   or
   **commit**

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **ipv4 address virtual address** *ipv4-address/mask*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# ipv4 virtual address 10.3.32.154/8` | Defines an IPv4 virtual address for the management Ethernet interface. |
| **Step 3** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-null0)# end`<br>or<br>`RP/0/0/CPU0:router(config-null0)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

# Configuration Examples for Virtual Interfaces

This section provides the following configuration examples:

- Configuring a Loopback Interface: Example, page 259
- Configuring a Null Interface: Example, page 260

## Configuring a Loopback Interface: Example

The following example indicates how to configure a loopback interface:

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# interface Loopback 3
RP/0/0/CPU0:router(config-if)# ipv4 address 172.18.189.38/32
RP/0/0/CPU0:router(config-if)# end
Uncommitted changes found, commit them? [yes]: yes
RP/0/0/CPU0:router# show interfaces Loopback 3
```

```
Loopback3 is up, line protocol is up
  Hardware is Loopback interface(s)
  Internet address is 172.18.189.38/32
  MTU 1514 bytes, BW Unknown
     reliability 0/255, txload Unknown, rxload Unknown
  Encapsulation Loopback,  loopback not set
  Last clearing of "show interface" counters never
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     0 packets input, 0 bytes, 0 total input drops
     0 drops for unrecognized upper-level protocol
     Received 0 broadcast packets, 0 multicast packets
     0 packets output, 0 bytes, 0 total output drops
     Output 0 broadcast packets, 0 multicast packets
```

# Configuring a Null Interface: Example

The following example indicates how to configure a null interface:

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# interface Null 0
RP/0/0/CPU0:router(config-null0)# ipv4 unreachables
RP/0/0/CPU0:router(config-null0)# end
Uncommitted changes found, commit them? [yes]: yes
RP/0/0/CPU0:router# show interfaces Null 0

Null0 is up, line protocol is up
  Hardware is Null interface
  Internet address is Unknown
  MTU 1500 bytes, BW Unknown
     reliability 0/255, txload Unknown, rxload Unknown
  Encapsulation Null,  loopback not set
  Last clearing of "show interface" counters never
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     0 packets input, 0 bytes, 0 total input drops
     0 drops for unrecognized upper-level protocol
     Received 0 broadcast packets, 0 multicast packets
     0 packets output, 0 bytes, 0 total output drops
     Output 0 broadcast packets, 0 multicast packets
```

# Configuring a Virtual IPv4 Interface: Example

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# ipv4 virtual address 10.3.32.154/8
RP/0/0/CPU0:router(config-null0)# commit
```

# Additional References

These sections provide references related to loopback and null interface configuration.

## Related Documents

| Related Topic | Document Title |
|---|---|
| Cisco IOS XR master command reference | *Cisco IOS XR Master Commands List* |
| Cisco IOS XR interface configuration commands | *Cisco IOS XR Interface and Hardware Component Command Reference* |
| Initial system bootup and configuration information for a router using the Cisco IOS XR Software. | *Cisco IOS XR Getting Started Guide* |
| Information about user groups and task IDs | *Cisco IOS XR Interface and Hardware Component Command Reference* |

## 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://www.cisco.com/go/mibs |

## RFCs

| 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/support |

# Configuring Channelized SONET/SDH on Cisco IOS XR Software

This module describes the configuration of Channelized SONET/SDH on the Cisco XR 12000 Series Router.

**Feature History for Configuring Channelized SONET/SDH on Cisco IOS XR Software**

| Release | Modification |
|---|---|
| Release 3.5.0 | This feature was introduced on the Cisco XR 12000 Series Router. |
| Release 3.7.0 | Channelized ATM and clear channel ATM procedures and examples was added. |
| Release 3.9.0 | Support for the following SPAs was introduced on the Cisco XR 12000 Series Router:<br>• Cisco 1-Port Channelized OC-12/DS0 SPA<br>• Cisco 1-Port Channelized OC-48/STM-16 SPA |

# Contents


- Prerequisites for Configuring Channelized SONET/SDH, page 263
- Information About Configuring Channelized SONET/SDH, page 264
- How to Configure Channelized SONET/SDH, page 273
- Configuration Examples for Channelized SONET, page 299
- Additional References, page 306


# Prerequisites for Configuring Channelized SONET/SDH

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 Channelized SONET/SDH, be sure that the following tasks and conditions are met:

- You have at least one of the following SPAs installed in your chassis:
  - Cisco 1-Port Channelized OC-3/STM-1 SPA

- Cisco 1-Port Channelized OC-12/DS0 SPA
- Cisco 1-Port Channelized OC-48/STM-16 SPA
- Cisco 2-Port Channelized T3/E3 ATM and Circuit Emulation SPA

- You should know how to apply and specify the SONET controller name and *interface-path-id* with the generalized notation *rack/slot/module/port*. The SONET controller name and *interface-path-id* are required with the **controller sonet** command.

# Information About Configuring Channelized SONET/SDH

To configure Channelized SONET/SDH, you must understand the following concepts:

- Channelized SONET Overview, page 264
- Channelized SDH Overview, page 269
- Default Configuration Values for Channelized SONET/SDH, page 272
- How to Configure Channelized SONET/SDH, page 273

## Channelized SONET Overview

Synchronous Optical Network (SONET) is an American National Standards Institute (ANSI) specification format used in transporting digital telecommunications services over optical fiber.

Synchronous Digital Hierarchy (SDH) is the international equivalent of SONET.

Channelized SONET provides the ability to transport SONET frames across multiplexed T3/E3 and virtual tributary group (VTG) channels.

Channelized SONET is supported on the following SPAs:

- Cisco 1-Port Channelized OC-3/STM-1 SPA
- Cisco 1-Port Channelized OC-12/DS0 SPA
- Cisco 1-Port Channelized OC-48/STM-16 SPA

Channelized SDH is supported on the following SPAs:

- Cisco 1-Port Channelized OC-3/STM-1 SPA
- Cisco 1-Port Channelized OC-12/DS0 SPA
- Cisco 1-Port Channelized OC-48/STM-16 SPA

Channelized T3/E3 ATM is supported only on the following SPA:

- Cisco 2-Port Channelized T3/E3 ATM and Circuit Emulation SPA

SONET uses Synchronous Transport Signal (STS) framing. An STS is the electrical equivalent to an optical carrier 1 (OC-1).

SDH uses Synchronous Transport Mode (STM) framing. An STM-1 is the electrical equivalent to 3 optical carrier 1s (OC-1s).

A channelized SONET interface is a composite of STS streams, which are maintained as independent frames with unique payload pointers. The frames are multiplexed before transmission.

When a line is channelized, it is logically divided into smaller bandwidth channels called *paths*. These paths carry the SONET payload. The sum of the bandwidth on all paths cannot exceed the line bandwidth.

When a line is not channelized, it is called *clear channel*, and the full bandwidth of the line is dedicated to a single channel that carries broadband services.

An STS stream can be channelized into the following types of channels:

- T3/E3
- VT1.5 mapped T1
- Packet over SONET/SDH (POS) (OC12 and OC48 only)

The T3/E3 channels can be channelized further into T1s, and the T1s can be channelized into time slots (DS0s), except on the ATM Cisco 2-Port Channelized T3/E3 ATM and Circuit Emulation Shared Port Adapter, which does not support DS0s.

, except on the 1-Port Channelized OC-48/STM-16 SPA, which does not support T1 or DS0sChannelizing a SONET line consists of two primary processes:

- Configuring the controller
- Configuring the interface into channelized paths

You configure the controller first by setting the mode of the STS path. The mode can be set to T3, or VT1.5-mapped T1, or POS, depending on your hardware support.

**Note** POS is supported only on the STS-3c and STS-12c paths on the Cisco 1-Port Channelized OC-12/DS0 SPA and on the STS-3c, STS-12c, and STS-48c paths on the Cisco 1-Port Channelized OC-48/STM-16 SPA.

When the mode is specified, the respective controller is created, and the remainder of the configuration is applied on that controller. For example, mode T3 creates a T3 controller. The T3 controller can then be configured to a serial channel, or it can be further channelized to carry T1s, and those T1s can be configured to serial interfaces.

On a Cisco 1-Port Channelized OC-3/STM-1 SPA, the default configuration consists of the following paths that are already configured when the SONET card is installed.

- STS 1
- STS 2
- STS 3

Each STS path can be independently configured into T3s, E3s, or VTGs, and so on.

Figure 13 shows the SONET controller default configuration that is in place when the Cisco 1-Port Channelized OC-3/STM-1 SPA is installed.

***Figure 13 SONET Controller Default Configuration***

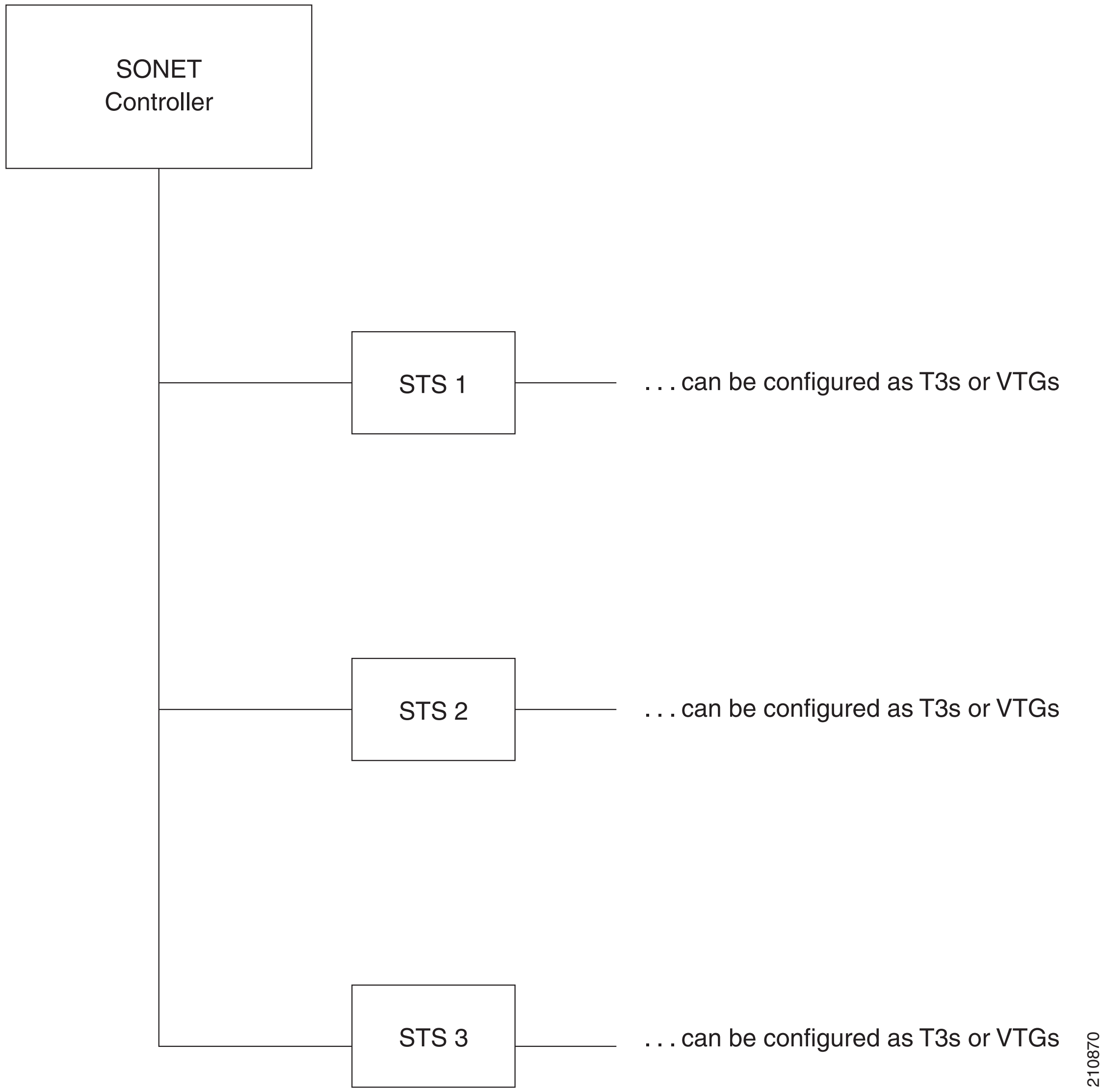


Figure 14 shows an example of some SONET controller configuration combinations.

***Figure 14 SONET Controller Configuration Combinations***

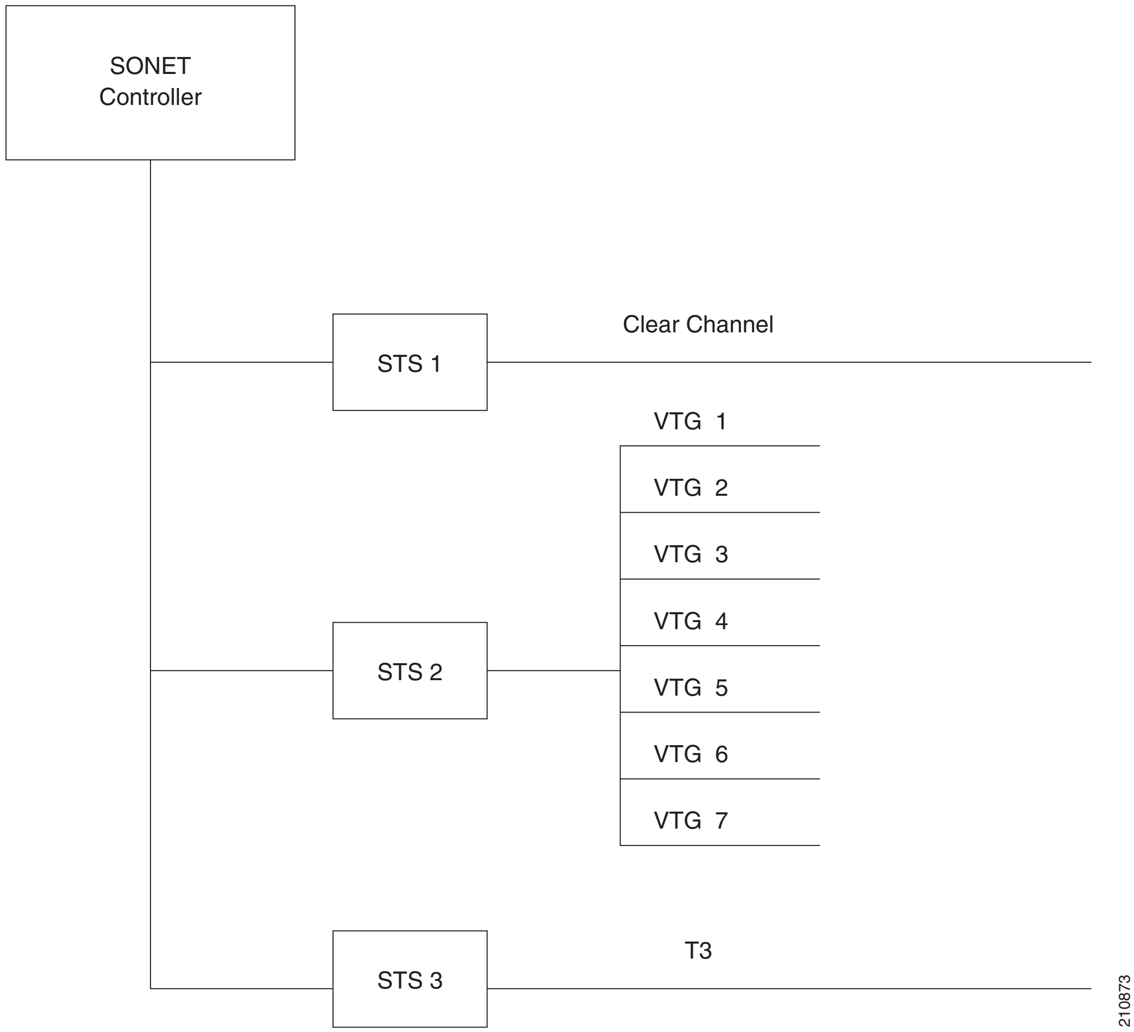


Figure 15 shows the T3 paths that can be configured.

***Figure 15 SONET T3 Channelized Paths***

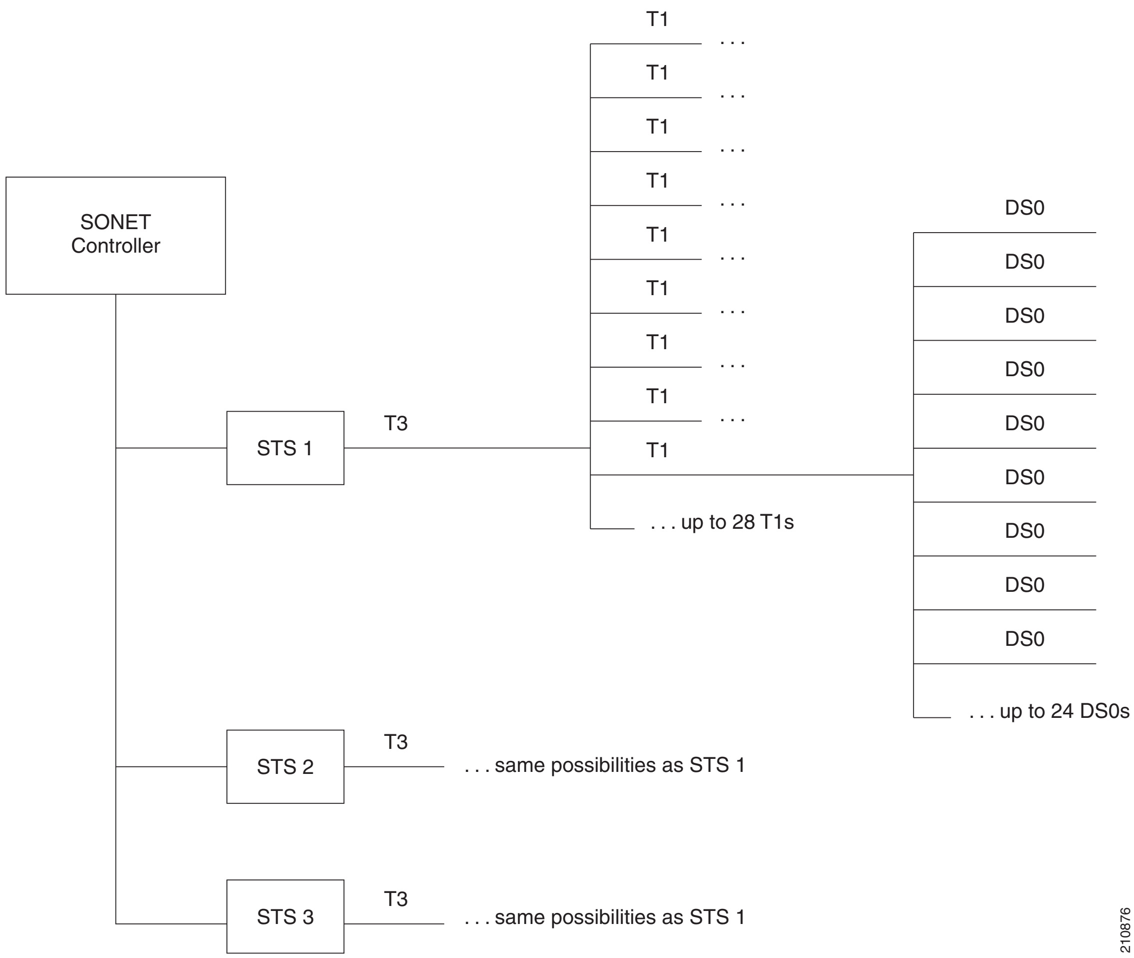
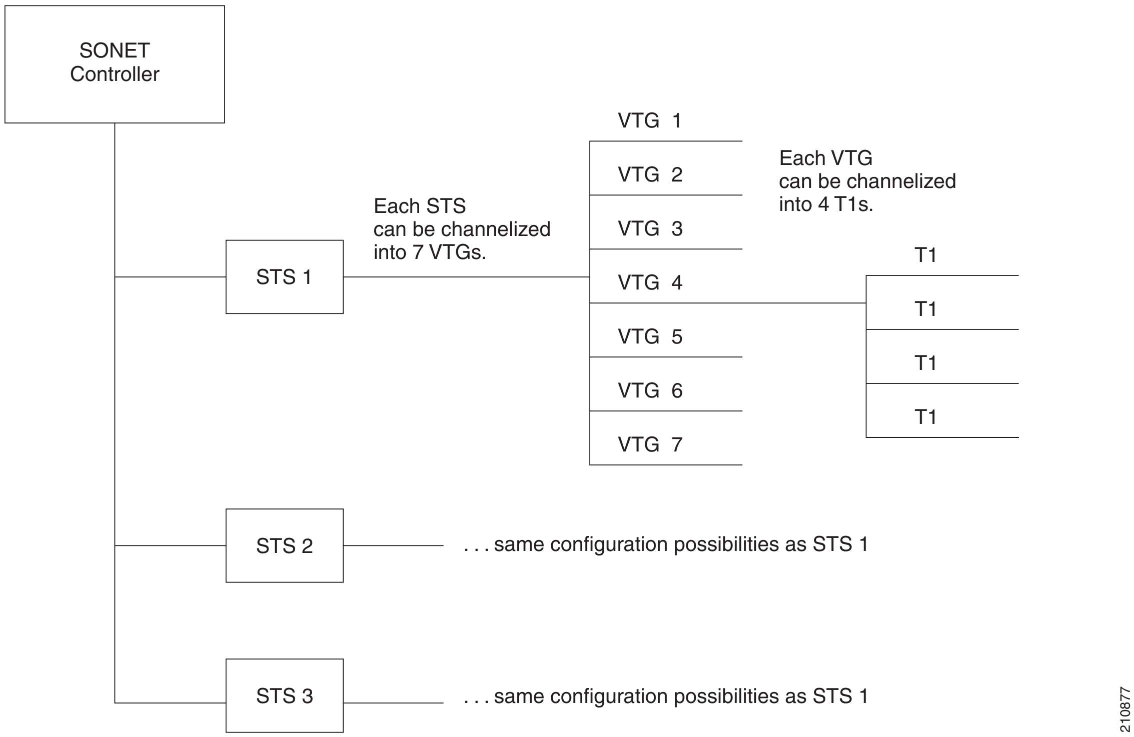


Figure 16 shows the VTG paths that can be configured.

***Figure 16 SONET VTG Channelized Paths***




# Channelized SDH Overview

Synchronous Digital Hierarchy (SDH) is the international equivalent of SONET.

Channelized SDH is supported on the following SPAs:

- Cisco 1-Port Channelized OC-3/STM-1 SPA
- Cisco 1-Port Channelized OC-12/DS0 SPA
- Cisco 1-Port Channelized OC-48/STM-16 SPA

A Synchronous Transport Module (STM) signal is the Synchronous Digital Hierarchy (SDH) equivalent of the SONET STS, but the numbers are different for each bandwidth. In this guide, the STM term refers to both path widths and optical line rates. The paths within an STM signals are called administrative units (AUs).

A summary of the basic terminology differences between SONET and SDH is as follows:

- SONET STS is equivalent to SDH administrative unit (AU)
- SONET VT is equivalent to SDH tributary unit (TU)
- SDH basic building blocks are STM-1 (equivalent to STS-3) and STM-0 (equivalent to STS-1)

An administrative unit (AU) is the information structure that provides adaptation between the higher-order path layer and the multiplex section layer. It consists of an information payload (the higher-order virtual container) and an administrative unit pointer, which indicates the offset of the payload frame start relative to the multiplex section frame start.

An AU can be channelized into tributary units (TUs) and tributary unit groups (TUGs).

An administrative unit 4 (AU-4) consists of three STM-1s or an STM-3.

An administrative unit 3 (AU-3) consists of one STM-1.

An administrative unit group (AUG) consists of one or more administrative units occupying fixed, defined positions in an STM payload.

***Table 5 SONET and SDH Terminology Equivalencies***

| SONET Term | SDH Term |
|---|---|
| SONET | SDH |
| STS-3c | AU-4 |
| STS-1 | AU-3 |
| VT | TU |
| SPE | VC |
| Section | Regenerator Section |
| Line | Multiplex Section |
| Path | Path |

Figure 17 shows an example of SDH AU-3 paths that can be configured on certain supported SPAs.

***Figure 17 SDH AU3 Paths***

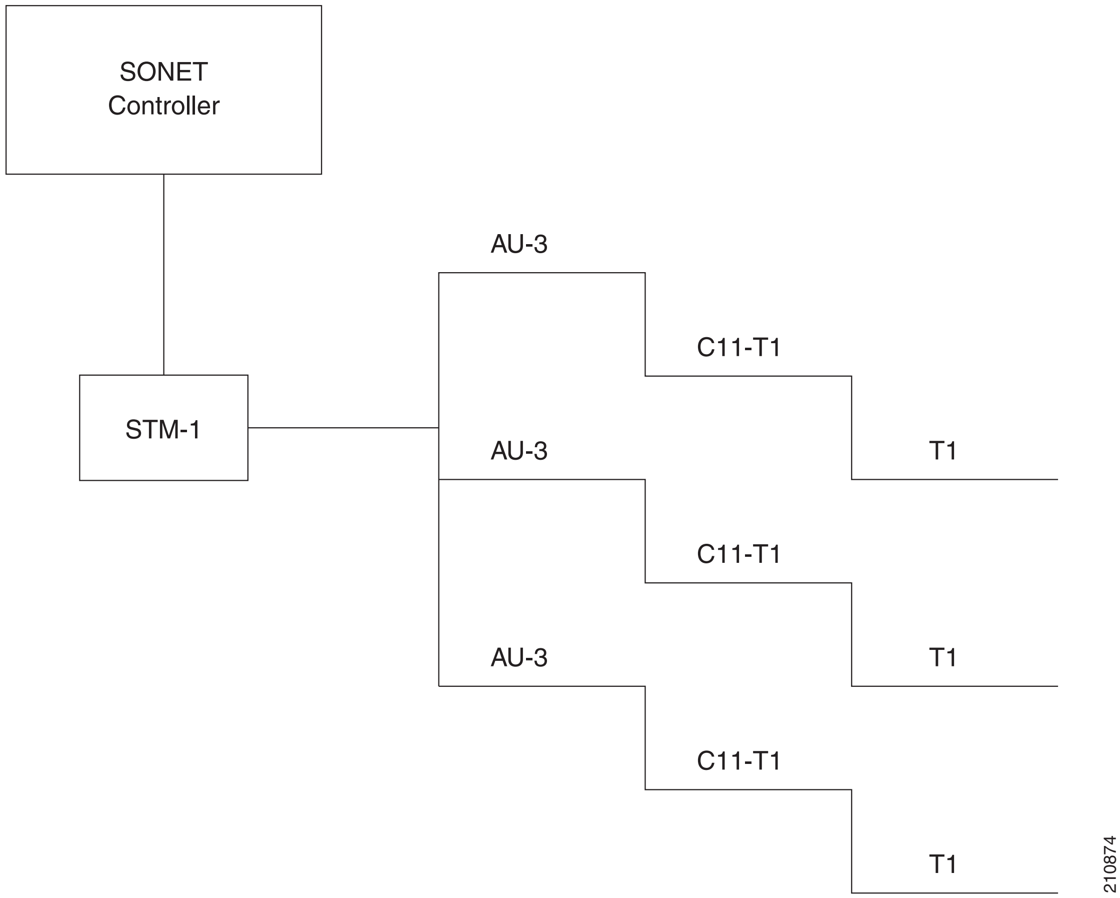
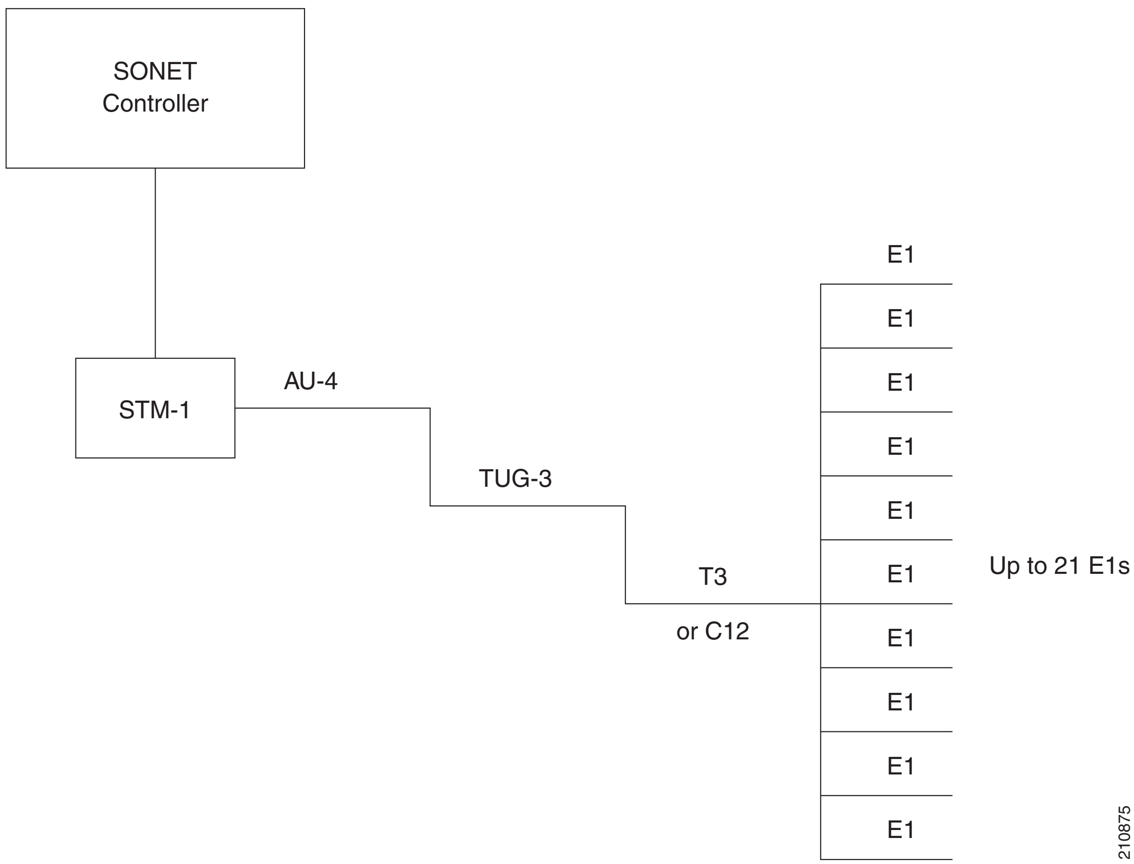


Figure 18 shows the SDH AU4 paths that can be configured on supported SPAs.

***Figure 18 SDH AU4 Paths***




# Default Configuration Values for Channelized SONET/SDH

Table 6 describes the default configuration parameters that are present on the Channelized SONET/SDH.

***Table 6 SONET/SDH Controller Default Cit onfiguration Values***

| Parameter | Default Value | Configuration File Entry |
|---|---|---|
| Clock source | line | **clock source** {**internal** \| **line**} |
| SONET framing | sonet | **framing** {**sdh** \| **sonet**} |

# How to Configure Channelized SONET/SDH

This section contains the following procedures:


- Configuring SONET T3 and VT1.5-Mapped T1 Channels, page 273
- Configuring Packet over SONET Channels, page 278
- Configuring a Clear Channel SONET Controller for T3, page 281
- Configuring Channelized SONET APS, page 284
- Configuring SDH AU-3, page 287
- Configuring SDH AU-4, page 294


## Configuring SONET T3 and VT1.5-Mapped T1 Channels

This task explains how to configure a SONET line into T3 and VT-mapped T1 Channels.

### Prerequisites

- You should know how to configure the SONET controller as specified in the "How to Configure Clear Channel SONET Controllers" section of the *Configuring Clear Channel SONET Controllers on Cisco IOS XR Software* module.
- STS paths can be channelized into T3s on the following SPAs:
  - Cisco 1-Port Channelized OC-3/STM-1 SPA
  - Cisco 1-Port Channelized OC-12/DS0 SPA
  - Cisco 1-Port Channelized OC-48/STM-16 SPA
- STS paths can be channelized into VTG mapped T1s on the following SPAs:
  - Cisco 1-Port Channelized OC-3/STM-1 SPA
  - Cisco 1-Port Channelized OC-12/DS0 SPA
- T3 paths can be channelized into T1s or E1s on the following SPAs:
  - Cisco 1-Port Channelized OC-3/STM-1 SPA
  - Cisco 1-Port Channelized OC-12/DS0 SPA (No E1 support in this release)

### Restrictions

T1s and E1s are not supported on the Cisco 1-Port Channelized OC-48/STM-16 SPA.

#### SUMMARY STEPS

1. **configure**
2. **controller sonet** *interface-path-id*
3. **clock source** {**internal** | **line**}
4. **framing sonet**
5. **sts** *number*

6. **mode** *mode*
7. **width** *number*
8. **root**
9. **controller** *controllerName instance*
10. **mode** *mode*
11. **root**
12. **controller t1** *interface-path-id*
13. **channel-group** *number*
14. **timeslots** *num1:num2:num3:num4*
    or
    **timeslots** *range1-range2*
15. **show configuration**
16. **root**
17. **interface serial** *interface-path-id*
18. **encapsulation** {**frame-relay** | **hdlc** | **ppp**}
19. **ipv4** *ip-address mask*
20. **no shutdown**
21. **end**
    or
    **commit**
22. **show**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | `configure`<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **`controller sonet`** *`interface-path-id`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller sonet 0/1/1/0` | Enters SONET controller configuration submode and specifies the SONET controller name and *interface-path-id* with the *rack/slot/module/port* notation. |

| | Command or Action | Purpose |
|---|---|---|
| Step 3 | **clock source** {**internal** \| **line**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# clock source internal` | Configures the SONET port transmit clock source, where the **internal** keyword sets the internal clock and the **line** keyword sets the clock recovered from the line.<br>• Use the **line** keyword whenever clocking is derived from the network. Use the **internal** keyword when two routers are connected back to back or over fiber for which no clocking is available.<br>• **line** is the default keyword.<br>**Note** Internal clocking is required for SRP interfaces. |
| Step 4 | **framing sonet**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# framing sonet` | Configures the controller for SONET framing.<br>SONET framing (**sonet**) is the default. |
| Step 5 | **sts** *number*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# sts 1` | Configures the STS stream specified by *number*. The ranges are:<br>• 1 to 3—1-Port Channelized OC-3/STM-1 SPA<br>• 1 to 12—1 Port Channelized OC-12/DS0 SPA<br>• 1 to 48—1 Port Channelized OC-48/STM-16 SPA |
| Step 6 | **mode** *mode*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-stsPath)# mode t3` | Sets the mode of interface at the STS level. The possible modes are:<br>• t3—SONET path carrying T3<br>• vt15-t1—SONET path carrying virtual tributary 1.5 T1s (VT15 T1)<br>• pos—Packet over SONET (OC12 and OC48 only) |
| Step 7 | **width** *number*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-stsPath)# width 3` | Configures the number of the STS streams that are concatenated. The possible values for *number* are:<br>• 1—Indicating one STS stream<br>• 3—Indicating three STS streams (STS-3c)<br>• 12—Indicating concatenation of 12 STS streams (STS-12c)<br>• 48—Indicating concatenation of 48 STS streams (STS-48c). This is the default on the 1-Port Channelized OC-48/STM-16 SPA.<br>Widths 3, and 12, and 48 are configured on STS paths at natural boundaries, which coincide with the following path numbers:<br>• 1, 4, 7, 10, and so on, for STS-3c<br>• 1, 13, 25, and 37 for STS-12c<br>• 1 for STS-48c |
| Step 8 | **root**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-stsPath)# root` | Exits to global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| Step 9 | **controller** *controllerName instance*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller t3 0/1/1/0/0` | Enters controller configuration submode and specifies the controller name and instance identifier with the *rack/slot/module/port/controllerName* notation. The controller names are:<br>• t3—SONET path carrying T3<br>• vt15-t1—SONET path carrying virtual tributary 1.5 T1s (VT15 T1) |
| Step 10 | **mode** *mode*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# mode t1` | Sets the mode of interface at this level. The possible modes are:<br>• t1—Channelized into 28 T1s<br>• e1—Channelized into 21 E1s<br>• serial—Clear channel carrying an HDLC-like payload |
| Step 11 | **root**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# root` | Exits to global configuration mode. |
| Step 12 | **controller t1** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller t1 0/1/1/0/0/0` | Enters T1 controller configuration submode and specifies the T1 controller name and *interface-path-id* with the *rack/slot/module/port/T3Num/T1num* notation. |
| Step 13 | **channel-group** *number*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1)# channel-group 1` | Sets the channel group number to which time slots are assigned. The range is from 1 to 24. |
| Step 14 | **timeslots** *num1:num2:num3:num4*<br>or<br>**timeslots** *range1-range2*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1-channel_group)# timeslots 1:3:7:9`<br>`RP/0/0/CPU0:router(config-t1-channel_group)# timeslots 1-24` | Specifies the time slots for the interface by number with the *num1:num2:num3:num4* notation, or by range with the *range1-range2* notation. |
| Step 15 | **show configuration**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1-channel_group)# show configuration` | Displays the contents of uncommitted configuration. |
| Step 16 | **root**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# root` | Exits to global configuration mode. |

<table>
<tr><th></th><th>Command or Action</th><th>Purpose</th></tr>
<tr><td>Step 17</td><td><code>interface serial interface-path-id</code><br><br>Example:<br><code>RP/0/0/CPU0:router(config)# interface serial 0/1/1/0/0/0:0</code></td><td>Specifies the complete interface number with the rack/slot/module/port/T3Num/T1num:instance notation.</td></tr>
<tr><td>Step 18</td><td><code>encapsulation {frame-relay | hdlc | ppp}</code><br><br>Example:<br><code>RP/0/0/CPU0:router(config-if)# encapsulation ppp</code></td><td>Specifies the encapsulation type with the one of the following keywords:<ul><li>frame-relay—Frame Relay network protocol</li><li>hdlc—High-level Data Link Control (HDLC) synchronous protocol</li><li>ppp—Point-to-Point Protocol</li></ul></td></tr>
<tr><td>Step 19</td><td><code>ipv4 ip-address mask</code><br><br>Example:<br><code>RP/0/0/CPU0:router(config-if)# ip address 10.10.10.10 255.255.255.255</code></td><td>Assigns an IP address and subnet mask to the interface.</td></tr>
<tr><td>Step 20</td><td><code>no shutdown</code><br><br>Example:<br><code>RP/0/0/CPU0:router(config-if)# no shutdown</code></td><td>Removes the shutdown configuration.<br><br>Note Removal of the shutdown configuration eliminates the forced administrative down on the interface, enabling it to move to an up or down state (assuming that the parent SONET layer is not configured administratively down).</td></tr>
<tr><td>Step 21</td><td><code>end</code><br>or<br><code>commit</code><br><br>Example:<br><code>RP/0/00/CPU0:router(config-sonet)# end</code><br>or<br><code>RP/0/0/CPU0:router(config-sonet)# commit</code></td><td>Saves configuration changes.<ul><li>When you issue the end command, the system prompts you to commit changes:<br><code>Uncommitted changes found, commit them before exiting(yes/no/cancel)?<br>[cancel]:</code><ul><li>Entering yes saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.</li><li>Entering no exits the configuration session and returns the router to EXEC mode without committing the configuration changes.</li><li>Entering cancel leaves the router in the current configuration session without exiting or committing the configuration changes.</li></ul></li><li>Use the commit command to save the configuration changes to the running configuration file and remain within the configuration session.</li></ul></td></tr>
<tr><td>Step 22</td><td><code>show controllers sonet interface-path-id</code><br><br>Example:<br><code>RP/0/0/CPU0:router# show controllers sonet 0/1/1/0</code></td><td>Verifies the SONET controller configuration.</td></tr>
</table>

# Configuring Packet over SONET Channels

This task explains how to configure Packet over SONET (POS) channels on SPAs supporting channelized SONET.

## Prerequisites

You have one of the following SPAs installed:

- Cisco 1-Port Channelized OC-12/DS0 SPA
- Cisco 1-Port Channelized OC-48/STM-16 SPA

### SUMMARY STEPS

1. **configure**
2. **controller sonet** *interface-path-id*
3. **clock source** {**internal** | **line**}
4. **framing** {**sdh** | **sonet**}
5. **sts** *number*
6. **width** *number*
7. **mode** *mode* **scramble**
8. **root**
9. **interface pos** *interface-path-id*
10. **encapsulation** [**hdlc** | **ppp** | **frame-relay** [**IETF**]]
11. **pos crc** {**16** | **32**}
12. **mtu** *value*
13. **no shutdown**
14. **end**
    or
    **commit**
15. **show interfaces pos** *interface-path-id*

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | `configure`<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | `controller sonet` *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller sonet 0/1/1/0` | Enters SONET controller configuration submode and specifies the SONET controller name and *interface-path-id* with the *rack/slot/module/port* notation. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **clock source** {**internal** \| **line**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# clock source internal` | Configures the SONET port transmit clock source, where the **internal** keyword sets the internal clock and the **line** keyword sets the clock recovered from the line.<br>• Use the **line** keyword whenever clocking is derived from the network. Use the **internal** keyword when two routers are connected back to back or over fiber for which no clocking is available.<br>• **line** is the default keyword.<br>**Note** Internal clocking is required for SRP interfaces. |
| **Step 4** | **framing** {**sdh** \| **sonet**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# framing sonet` | (Optional) Configures the controller framing with either the **sdh** keyword for Synchronous Digital Hierarchy (SDH) framing or the **sonet** keyword for SONET framing.<br>SONET framing (**sonet**) is the default. |
| **Step 5** | **sts** *number*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# sts 1` | Configures the STS stream specified by *number.* The ranges are:<br>• 1 to 12 on the 1-Port Channelized OC12/DS0 SPA<br>• 1 to 48 on the 1 Port Channelized OC48/DS3 SPA |
| **Step 6** | **width** *number*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-stsPath)# width 3` | Configures the number of the STS streams that are concatenated. The possible values for *number* are:<br>• 3—Indicating three STS streams (STS-3c)<br>• 12—Indicating concatenation of 12 STS streams (STS-12c)<br>• 48—Indicating concatenation of 48 STS streams (STS-48c)<br>Widths 3, 12, and 48 are configured on STS paths at natural boundaries, which coincide with the following path numbers:<br>• 1, 4, 7, 10, and so on, for STS-3c<br>• 1, 13, 25, and 37 for STS-12c<br>• 1 for STS-48c<br>**Note** POS interfaces are not supported when width is 1. |
| **Step 7** | **mode** *mode* **scramble**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-stsPath)# mode pos scramble` | Sets the mode of interface at the STS level. Set the mode to pos to create POS interface (OC12 and OC48 only). |
| **Step 8** | **root**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-stsPath)# root` | Exits to global configuration mode. |
| **Step 9** | **interface pos** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface POS 0/1/1/0` | Specifies the POS interface name and notation *rack/slot/module/port,* and enters interface configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| Step 10 | **encapsulation** [**hdlc** \| **ppp** \| **frame-relay** [**IETF**]]<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# encapsulation hdlc` | (Optional) Configures the interface encapsulation parameters and details such as HDLC or PPP. The default is HDLC. |
| Step 11 | **pos crc** {**16** \| **32**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# pos crc 32` | (Optional) Configures the CRC value for the interface. Enter the **16** keyword to specify 16-bit CRC mode, or enter the **32** keyword to specify 32-bit CRC mode.<br>The default CRC is **32**. |
| Step 12 | **mtu** *value*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# mtu 4474` | (Optional) Configures the POS MTU value.<br>The range is 64–65535. |
| Step 13 | **no shutdown**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-if)# no shutdown` | Removes the shutdown configuration.<br>**Note** Removal of the shutdown configuration eliminates the forced administrative down on the interface, enabling it to move to an up or down state (assuming that the parent SONET layer is not configured administratively down). |
| Step 14 | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# end`<br>or<br>`RP/0/0/CPU0:router(config-sonet)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 15 | **show interfaces pos** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router# show interfaces pos 0/1/1/0` | (Optional) Displays the interface configuration. |

# Configuring a Clear Channel SONET Controller for T3

This task explains how to configure a SONET line into a single T3 serial channel called *clear channel*. Clear channel is established by setting the T3 controller mode to serial.

## Prerequisites

- You should know how to configure the SONET controller as specified in the "How to Configure Clear Channel SONET Controllers" section of the *Configuring Clear Channel SONET Controllers on Cisco IOS XR Software* module.

## Restrictions

On the Cisco 1-Port Channelized OC-3/STM-1 SPA, clear channel is supported only on STS-1 and STS-2.

### SUMMARY STEPS

1. **configure**
2. **controller sonet** *interface-path-id*
3. **clock source** {**internal** | **line**}
4. **framing sonet**
5. **sts** *number*
6. **mode t3**
7. **root**
8. **controller t3** *interface-path-id*
9. **mode serial**
10. **root**
11. **interface serial** *interface-path-id*
12. **encapsulation** {**frame-relay** | **hdlc** | **ppp**}
13. **ipv4** *ip-address mask*
14. **no shutdown**
15. **end**
    or
    **commit**
16. **show controllers sonet** *interface-path-id*

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| Step 1 | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| Step 2 | **controller sonet** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller sonet 0/1/1/0` | Enters SONET controller configuration submode and specifies the SONET controller name and *interface-path-id* with the *rack/slot/module/port* notation. |
| Step 3 | **clock source** {**internal** \| **line**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# clock source internal` | Configures the SONET port transmit clock source, where the **internal** keyword sets the internal clock and the **line** keyword sets the clock recovered from the line.<br>• Use the **line** keyword whenever clocking is derived from the network. Use the **internal** keyword when two routers are connected back to back or over fiber for which no clocking is available.<br>• **line** is the default keyword.<br>**Note** Internal clocking is required for SRP interfaces. |
| Step 4 | **framing sonet**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# framing sonet` | Configures the controller for SONET framing. SONET framing (**sonet**) is the default. |
| Step 5 | **sts** *number*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# sts 1` | Configures the STS stream specified by *number*. The ranges are:<br>• 1 to 3—1-Port Channelized OC-3/STM-1 SPA<br>• 1 to 12—1-Port Channelized OC-12/DS0 SPA<br>• 1 to 48—1-Port Channelized OC-48/DS3 SPA |
| Step 6 | **mode t3**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-stsPath)# mode t3` | Sets the mode of the interface at the STS level for T3. |
| Step 7 | **root**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-stsPath)# root` | Exits to global configuration mode. |
| Step 8 | **controller t3** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller t3 0/1/1/0/0` | Enters T3 controller configuration submode and specifies the T3 controller name and interface-path-id identifier with the *rack/slot/module/port/T3Num* notation. |

| | Command or Action | Purpose |
|---|---|---|
| Step 9 | **mode serial**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# mode serial` | Sets the mode of the interface to serial to establish a clear channel. |
| Step 10 | **root**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# root` | Exits to global configuration mode. |
| Step 11 | **interface serial** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface serial 0/1/1/0/0/0:0` | Specifies the complete interface number with the *rack/slot/module/port/T3Num/T1num:instance* notation. |
| Step 12 | **encapsulation** {**frame-relay** \| **hdlc** \| **ppp**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# encapsulation ppp` | Specifies the encapsulation type with the one of the following keywords:<br>• **frame-relay**—Frame Relay network protocol<br>• **hdlc**—High-level Data Link Control (HDLC) synchronous protocol<br>• **ppp**—Point-to-Point Protocol |
| Step 13 | **ipv4** *ip-address mask*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# ip address 10.10.10.10 255.255.255.255` | Assigns an IP address and subnet mask to the interface. |
| Step 14 | **no shutdown**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# no shutdown` | Removes the shutdown configuration.<br>**Note** Removal of the shutdown configuration eliminates the forced administrative down on the interface, enabling it to move to an up or down state (assuming that the parent SONET layer is not configured administratively down). |

| | Command or Action | Purpose |
|---|---|---|
| **Step 15** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# end`<br>or<br>`RP/0/0/CPU0:router(config-sonet)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 16** | **show controllers sonet** *interface-path-id*<br><br>**Example:**<br>`RP/0//0/CPU0:router# show controllers sonet 0/1/1/0` | Verifies the SONET controller configuration. |

# Configuring Channelized SONET APS

This task explains how to configure APS for channelized SONET lines. The Cisco XR 12000 Series Router supports both single router and multirouter APS.

## Prerequisites

- You should know how to configure the SONET controller as specified in the "How to Configure Clear Channel SONET Controllers" section of the *Configuring Clear Channel SONET Controllers on Cisco IOS XR Software* module.
- You should know how to configure the SONET APS as specified in the "Configuring SONET APS" section of the *Configuring Clear Channel SONET Controllers on Cisco IOS XR Software* module.

### SUMMARY STEPS

1. **aps group** *number*
2. **channel 0 local sonet** *interface*
   or
   **channel 0 remote** *ip-address*

3. **channel 1 local sonet** *interface*
   or
   **channel 1 remote** *ip-address*
4. **signalling** {**sonet** | **sdh**}
5. **end**
   or
   **commit**
6. **show aps**
7. **show aps group** [*number*]

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| Step 1 | `aps group number`<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# aps group 1` | Adds an APS group with a specified number and enters APS group configuration mode.<br>• Use the **aps group** command in global configuration mode.<br>• To remove a group, use the **no** form of this command, as in: **no aps group** *number,* where the value range is from 1 to 255.<br>**Note** To use the **aps group** command, you must be a member of a user group associated with the proper task IDs for APS commands.<br>**Note** The **aps group** command is used even when a single protect group is configured. |
| Step 2 | `channel 0 local sonet interface`<br>`or`<br>`channel 0 remote ip-address`<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-aps)# channel 0 local SONET 0/0/0/1`<br>`or`<br>`RP/0/0/CPU0:router(config-aps)# channel 0 remote 172.18.69.123` | Creates a protect channel for the APS group, where **0** designates a protect channel.<br>**Note** The protect channel must be assigned before the active channel can be assigned.<br>**Note** To configure APS where both channels are on one router, use the **channel local** command for both the protect and active channels.<br>To configure APS using two different routers where the active channel is on one router and the protect channel is on another router, use the **channel local** command for either the protect or the active channel, but use the **channel remote** command for the other channel. |

| | Command or Action | Purpose |
|---|---|---|
| Step 3 | **channel 1 local sonet** *interface*<br>or<br>**channel 1 remote** *ip-address*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-aps)# channel 1 local SONET 0/0/0/2`<br>or<br>`RP/0/0/CPU0:router(config-aps)# channel 1 remote 172.18.69.123` | Creates an active channel for the APS group, where **1** designates an active channel.<br>**Note** The active channel must be assigned after the protect channel is assigned.<br>**Note** To configure APS where both channels are on one router, use the **channel local** command for both the protect and active channels.<br>To configure APS using two different routers where the active channel is on one router and the protect channel is on another router, use the **channel local** command for either the protect or the active channel, but use the **channel remote** command for the other channel. |
| Step 4 | **signalling** {**sonet** \| **sdh**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-aps)#` **`signalling sonet`** | Configures the K1K2 overhead byte used for automatic protection switching (APS). The keyword options are:<br>• **sonet**—Sets signaling to SONET.<br>• **sdh**—Sets signaling to Synchronous Digital Hierarchy (SDH). |
| Step 5 | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# end`<br>or<br>`RP/0/0/CPU0:router(config-sonet)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 6 | **show aps**<br><br>**Example:**<br>`RP/0/0/CPU0:router# show aps` | (Optional) Displays the operational status for all configured SONET APS groups. |
| Step 7 | **show aps group** [*number*]<br><br>**Example:**<br>`RP/0/0/CPU0:router# show aps group 3` | (Optional) Displays the operational status for configured SONET APS groups.<br>**Note** The **show aps group** command is more useful than the **show aps** command when multiple groups are defined. |

# Configuring SDH AU-3

This section includes the following tasks:

- Configuring SDH AU-3 Mapped to C11-T1 or C12-E1, page 287
- Configuring SDH AU-3 Mapped to T3 or E3, page 290

## Configuring SDH AU-3 Mapped to C11-T1 or C12-E1

This task explains how to configure SDH AU-3 with c11-t1 or c12-e1 mapping.

### Prerequisites

- You should know how to configure the SONET controller as specified in the "How to Configure Clear Channel SONET Controllers" section of the *Configuring Clear Channel SONET Controllers on Cisco IOS XR Software* module.

### Restrictions

Channelized SDH AU-3 with c11-t1 or c12-e1 mapping is supported on the following SPAs:

- Cisco 1-Port Channelized OC-3/STM-1 SPA
- Cisco 1-Port Channelized OC-12/DS0 SPA

### SUMMARY STEPS

1. **configure**
2. **controller sonet** *interface-path-id*
3. **clock source** {**internal** | **line**}
4. **framing sdh**
5. **au** *number*
6. **mode** *mode*
7. **root**
8. **controller t1** *interface-path-id*
9. **channel-group** *number*
10. **timeslots** *num1:num2:num3:num4*
    or
    **timeslots** *range1-range2*
11. **show configuration**
12. **root**
13. **interface serial** *interface-path-id*
14. **encapsulation** {**frame-relay** | **hdlc** | **ppp**}
15. **ipv4** *ip-address mask*
16. **no shutdown**

17. **end**
or
**commit**

18. **show controllers sonet** *interface-path-id*

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **`configure`**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **`controller sonet`** *`interface-path-id`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller sonet 0/1/1/0` | Enters SONET controller configuration submode and specifies the SONET controller name and interface-path-id identifier with the *rack/slot/module/port* notation. |
| **Step 3** | **`clock source`** {**`internal`** \| **`line`**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# clock source internal` | Configures the SONET port transmit clock source, where the **internal** keyword sets the internal clock and the **line** keyword sets the clock recovered from the line.<br>• Use the **line** keyword whenever clocking is derived from the network. Use the **internal** keyword when two routers are connected back to back or over fiber for which no clocking is available.<br>• **line** is the default keyword.<br>**Note** Internal clocking is required for SRP interfaces. |
| **Step 4** | **`framing sdh`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# framing sdh` | Configures the controller framing for Synchronous Digital Hierarchy (SDH) framing.<br>SONET framing (**sonet**) is the default. |
| **Step 5** | **`au`** *`number`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# au 1` | Specifies the administrative unit (AU) group and enters AU path configuration mode. For AU-3, the valid range is:<br>• 1 to 3—1-Port Channelized OC-3/STM-1 SPA<br>• 1 to 12—1-Port Channelized OC-12/DS0 SPA<br>**Note** The **au** command does not specify the AU type. It specifies the number of the AU group for the AU type that you want to configure. The range for the AU command varies based on whether you are configuring AU-3 or AU-4. |
| **Step 6** | **`mode`** *`mode`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-auPath)# mode c11-t1` | Sets the mode of interface at the AU level. AU-3 paths can be mapped to c11-t1 or c12-e1 on supported SPAs. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 7** | **root**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-auPath)# root` | Exits to global configuration mode. |
| **Step 8** | **controller t1** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller T1 0/1/1/0/0/0/0` | Enters T1 controller configuration submode and specifies the T1 controller name and *interface-path-id* with the *rack/slot/module/port/auNum/t1Num* notation. |
| **Step 9** | **channel-group** *number*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1)# channel-group 0` | Sets the channel-group number to which time slots are assigned. The range is from 1 to 28. |
| **Step 10** | **timeslots** *num1:num2:num3:num4*<br>or<br>**timeslots** *range1-range2*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1-channel_group)# timeslots 1:3:7:9`<br>`RP/0/0/CPU0:router(config-t1-channel_group)# timeslots 1-12` | Specifies time slots for the interface by number with the *num1:num2:num3:num4* notation, or by range with the *range1-range2* notation. |
| **Step 11** | **show configuration**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1-channel_group)# show configuration` | Displays the contents of uncommitted configuration. |
| **Step 12** | **root**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# root` | Exits to global configuration mode. |
| **Step 13** | **interface serial** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface serial 0/1/1/0/0/0:0` | Specifies the complete interface number with the *rack/slot/module/port/T3Num/T1num:instance* notation. |
| **Step 14** | **encapsulation** {**frame-relay** \| **hdlc** \| **ppp**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# encapsulation frame-relay` | Specifies the encapsulation type with the one of the following keywords:<br>• **frame-relay**—Frame Relay network protocol<br>• **hdlc**—High-level Data Link Control (HDLC) synchronous protocol<br>• **ppp**—Point-to-Point Protocol |

| | Command or Action | Purpose |
|---|---|---|
| **Step 15** | **ipv4** *ip-address mask*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# ip address 10.10.10.10 255.255.255.255` | Assigns an IP address and subnet mask to the interface. |
| **Step 16** | **no shutdown**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# no shutdown` | Removes the shutdown configuration.<br>**Note** Removal of the shutdown configuration eliminates the forced administrative down on the interface, enabling it to move to an up or down state (assuming that the parent SONET layer is not configured administratively down). |
| **Step 17** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# end`<br>or<br>`RP/0/0/CPU0:router(config-sonet)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 18** | **show controllers sonet** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router# show controllers sonet 0/1/1/0` | Verifies the SONET controller configuration. |

## Configuring SDH AU-3 Mapped to T3 or E3

This task explains how to configure SDH AU-3 mapped to T3 or E3.

### Prerequisites

- You should know how to configure the SONET controller as specified in the "How to Configure Clear Channel SONET Controllers" section of the *Configuring Clear Channel SONET Controllers on Cisco IOS XR Software* module.

### Restrictions

Channelized SDH AU-3 with T3 or E3 mapping is supported on the following SPAs:

- Cisco 1-Port Channelized OC-3/STM-1 SPA
- Cisco 1-Port Channelized OC-12/DS0 SPA
- Cisco 1-Port Channelized OC-48/STM-16 SPA

## SUMMARY STEPS

1. **configure**
2. **controller sonet** *interface-path-id*
3. **clock source** {**internal** | **line**}
4. **framing sdh**
5. **au** *number*
6. **mode t3**
   or
   **mode e3**
7. **root**
8. **controller** {**t3** | **e3**} *interface-path-id*
9. **mode serial**
10. **show configuration**
11. **root**
12. **interface serial** *interface-path-id*
13. **encapsulation** {**frame-relay** | **hdlc** | **ppp**}
14. **ipv4** *ip-address mask*
15. **no shutdown**
16. **end**
    or
    **commit**
17. **show controllers sonet** *interface-path-id*

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| Step 1 | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| Step 2 | **controller sonet** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller sonet 0/1/1/0` | Enters SONET controller configuration submode and specifies the SONET controller name and interface-path-id identifier with the *rack/slot/module/port* notation. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **clock source** {**internal** \| **line**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# clock source internal` | Configures the SONET port transmit clock source, where the **internal** keyword sets the internal clock and the **line** keyword sets the clock recovered from the line.<br>• Use the **line** keyword whenever clocking is derived from the network. Use the **internal** keyword when two routers are connected back to back or over fiber for which no clocking is available.<br>• **line** is the default keyword.<br>**Note** Internal clocking is required for SRP interfaces. |
| **Step 4** | **framing sdh**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# framing sdh` | Configures the controller framing for Synchronous Digital Hierarchy (SDH) framing.<br>SONET framing (**sonet**) is the default. |
| **Step 5** | **au** *number*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# au 1` | Specifies the administrative unit (AU) group and enters AU path configuration mode. For AU-3, the valid range is:<br>• 1 to 3—1-Port Channelized OC-3/STM-1 SPA<br>• 1 to 12—1-Port Channelized OC-12/DS0 SPA<br>• 1 to 48—1-Port Channelized OC-48/DS3 SPA<br>**Note** The **au** command does not specify the AU type. It specifies the number of the AU group for the AU type that you want to configure. The range for the AU command varies based on whether you are configuring AU-3 or AU-4. |
| **Step 6** | **mode t3**<br>or<br>**mode e3**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-auPath)# mode t3` | Sets the mode of interface at the AU level to T3 or E3. |
| **Step 7** | **root**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-auPath)# root` | Exits to global configuration mode. |
| **Step 8** | **controller** {**t3** \| **e3**} *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller T3 0/1/1/0/0` | Enters T3 or E3 controller configuration submode and specifies the T3 or E3 controller name and *interface-path-id* with the *rack/slot/module/port/auNum* notation. |
| **Step 9** | **mode serial**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# mode serial` | Configures the mode of the port to be clear channel serial. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 10** | **show configuration**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# show configuration` | Displays the contents of uncommitted configuration. |
| **Step 11** | **root**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# root` | Exits to global configuration mode. |
| **Step 12** | **interface serial** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface serial 0/1/1/0/0/0:0` | Specifies the complete interface number with the *rack/slot/module/port/T3Num/T1num:instance* notation. |
| **Step 13** | **encapsulation frame-relay** \| **hdlc** \| **ppp**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# encapsulation frame-relay | hdlc | ppp` | Specifies the encapsulation type with the one of the following keywords:<br>• **frame-relay**—Frame Relay network protocol<br>• **hdlc**—High-level Data Link Control (HDLC) synchronous protocol<br>• **ppp**—Point-to-Point Protocol |
| **Step 14** | **ipv4** *ip-address mask*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# ip address 10.10.10.10 255.255.255.255` | Assigns an IP address and subnet mask to the interface. |
| **Step 15** | **no shutdown**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# no shutdown` | Removes the shutdown configuration.<br>**Note** Removal of the shutdown configuration eliminates the forced administrative down on the interface, enabling it to move to an up or down state (assuming that the parent SONET layer is not configured administratively down). |

| | Command or Action | Purpose |
|---|---|---|
| **Step 16** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# end`<br>or<br>`RP/0/0/CPU0:router(config-sonet)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 17** | **show controllers sonet** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router# show controllers sonet 0/1/1/0` | Verifies the SONET controller configuration. |

# Configuring SDH AU-4

This task explains how to configure an SDH AU-4 stream into a TUG-3 channel mapped to E3s.

## Prerequisites

- You should know how to configure the SONET controller as specified in the "How to Configure Clear Channel SONET Controllers" section of the *Configuring Clear Channel SONET Controllers on Cisco IOS XR Software* module.

## Restrictions

- Channelized SDH is supported on the following SPAs:
  - Cisco 1-Port Channelized OC-3/STM-1 SPA
  - Cisco 1-Port Channelized OC-12/DS0 SPA
  - Cisco 1-Port Channelized OC-48/STM-16 SPA
- In this release, AU-4 paths can only be channelized into TUG-3s.
- The 1-Port Channelized OC-48/STM-16 SPA does not support T1 or E1 channelization.

## SUMMARY STEPS

1. **configure**
2. **controller sonet** *interface-path-id*
3. **clock source** {**internal** | **line**}
4. **framing sdh**
5. **au** *number*
6. **mode tug3**
7. **width** *number*
8. **tug3** *number*
9. **mode** *mode*
10. **root**
11. **controller** *name interface-path-id*
12. **mode** *mode*
13. **root**
14. **controller** *name instance*
15. **channel-group** *number*
16. **timeslots** *num1:num2:num3:num4*
    or
    **timeslots** *range1-range2*
17. **show configuration**
18. **root**
19. **interface serial** *interface-path-id*
20. **encapsulation** {**frame-relay** | **hdlc** | **ppp**}
21. **ipv4** *ip-address mask*
22. **no shutdown**
23. **end**
    or
    **commit**
24. **show controllers sonet** *interface-path-id*

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **controller sonet** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller sonet 0/1/1/0` | Enters SONET controller configuration submode and specifies the SONET controller name and *interface-path-id* with the *rack/slot/module/port* notation. |
| **Step 3** | **clock source** {**internal** \| **line**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# clock source internal` | Configures the SONET port transmit clock source, where the **internal** keyword sets the internal clock and the **line** keyword sets the clock recovered from the line.<br>• Use the **line** keyword whenever clocking is derived from the network. Use the **internal** keyword when two routers are connected back to back or over fiber for which no clocking is available.<br>• **line** is the default keyword.<br>**Note** Internal clocking is required for SRP interfaces. |
| **Step 4** | **framing sdh**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# framing sdh` | Configures the controller for Synchronous Digital Hierarchy (SDH) framing.<br>SONET framing (**sonet**) is the default. |
| **Step 5** | **au** *number*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# au 1` | Specifies the administrative unit (AU) group and enters AU path configuration mode. For AU-4, the valid range is:<br>• 1 to 3—1-Port Channelized OC-3/STM-1 SPA<br>• 1 to 4—1-Port Channelized OC-12/DS0 SPA<br>• 1 to 16—1-Port Channelized OC-48/DS3 SPA<br>**Note** The **au** command does not specify the AU type. It specifies the number of the AU group for the AU type that you want to configure. The range for the AU command varies based on whether you are configuring AU-3 or AU-4. |
| **Step 6** | **mode tug3**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-auPath)# mode tug3` | Sets the mode of interface at the AU level. Currently only TUG3 is supported. |
| **Step 7** | **width** *number*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-auPath)# width 3` | Configures the number of the AU streams. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 8** | **tug3** *number*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-auPath)#tug3 1` | Specifies the Tributary Unit Group (TUG) *number* and enters the config-tug3Path mode. The range is 1 to 3. |
| **Step 9** | **mode** *mode*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-tug3Path)# mode e3` | Sets the mode of interface at the tug3 level. The modes are:<br>• **c11**—TUG-3 path carrying TU-11<br>• **c11-t1**—TUG-3 path carrying TU-11 to T1<br>• **c12**—TUG-3 path carrying TU-12<br>• **c12-e1**—TUG-3 path carrying TU-12 to E1<br>• **e3**—TUG-3 path carrying E3<br>• **t3**—TUG-3 path carrying T3<br>**Note** The 1-Port Channelized OC-48/STM-16 SPA only supports the **e3** and **t3** options. |
| **Step 10** | **root**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-tug3Path)# root` | Exits to global configuration mode. |
| **Step 11** | **controller** *name instance*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller e3 0/1/1/0/0/0` | Enters controller configuration submode and specifies the controller name and instance identifier with the *rack/slot/module/port/name/instance* notation. The controller names are:<br>• **e3**—TUG3 path carrying E3<br>• **t3**—TUG3 path carrying T3<br>• **e1**—channelized E1 port<br>**Note** In this step, you can create an E3 or T3 controller and add T1 channels under the T3 controller as shown in Step 14, or you can create a channelized E1 port at this point.<br>**Note** E1 is not supported on the 1-Port Channelized OC-48/STM-16 SPA. |
| **Step 12** | **mode** *mode*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-e3)#mode e1` | Sets the mode of interface. The modes are:<br>• **e1**—Channelized into 21 E1s<br>• **serial**—Clear Channel carrying HDLC-like payload<br>• **t1**—Channelized into 28 T1s<br>**Note** T1 and E1 are not supported on the 1-Port Channelized OC-48/STM-16 SPA. |
| **Step 13** | **root**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-e3)# root` | Exits to global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| Step 14 | **controller** *name instance*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller E1 0/1/1/0/0/0/0/0` | Enters controller configuration submode and specifies the controller name and instance identifier with the *rack/slot/module/port/name/instance1/instance2* notation. The controller names are:<br>• serial—Clear Channel carrying HDLC-like payload.<br>• t1—Channelized into 24 T1s. |
| Step 15 | **channel-group** *number*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-e1)# channel-group 0` | Sets the channel-group number to which time slots are assigned.<br>• For t1, the range is from 1 to 24.<br>• For e1, the range is from 1 to 32. |
| Step 16 | **timeslots** *num1:num2:num3:num4*<br>or<br>**timeslots** *range1-range2*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-e1-channel_group)# timeslots 1:3:7:9`<br>`RP/0/0/CPU0:router(config-e1-channel_group)# timeslots 1-12` | Specifies time slots for the interface by number with the *num1:num2:num3:num4* notation, or by range with the *range1-range2* notation. |
| Step 17 | **show configuration**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-e1-channel_group)# show configuration` | Displays the contents of uncommitted configuration. |
| Step 18 | **root**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-e1-channel_group)# root` | Exits to global configuration mode. |
| Step 19 | **interface serial** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface serial 0/1/1/0/0/0:0` | Specifies the complete interface number with the *rack/slot/module/port/T3Num/T1num:instance* notation. |
| Step 20 | **encapsulation** {**frame-relay** \| **hdlc** \| **ppp**}<br><br>**Example:**<br>`Router(config-if)# encapsulation frame-relay | hdlc | ppp` | Specifies the encapsulation type with the one of the following keywords:<br>• **frame-relay**—Frame Relay network protocol<br>• **hdlc**—High-level Data Link Control (HDLC) synchronous protocol<br>• **ppp**—Point-to-Point Protocol |
| Step 21 | **ipv4** *ip-address mask*<br><br>**Example:**<br>`Router(config-if)# ip address 10.10.10.10 255.255.255.255` | Assigns an IP address and subnet mask to the interface. |

| | Command or Action | Purpose |
|---|---|---|
| Step 22 | **no shutdown**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-if)# no shutdown` | Removes the shutdown configuration.<br><br>**Note** Removal of the shutdown configuration eliminates the forced administrative down on the interface, enabling it to move to an up or down state (assuming that the parent SONET layer is not configured administratively down). |
| Step 23 | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# end`<br>or<br>`RP/0/0/CPU0:router(config-sonet)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 24 | **show controllers sonet** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router# show controllers sonet 0/1/1/0` | Verifies the SONET controller configuration. |

# Configuration Examples for Channelized SONET

This section contains the following examples:


- Channelized SONET Examples, page 299
- Channelized SDH Examples, page 302


## Channelized SONET Examples


- Channelized SONET T3 to T1 Configuration: Example, page 300
- Channelized Packet over SONET Configuration: Example, page 300
- SONET Clear Channel T3 Configuration: Example, page 301
- Channelized SONET APS Single Router Configuration: Example, page 301
- Channelized SONET APS Multirouter Configuration: Example, page 301

## Channelized SONET T3 to T1 Configuration: Example

The following example shows SONET T3 to T1 configuration.

```
configure
 controller sonet 0/1/1/0
  clock source internal
  framing sonet
  sts 1
  mode t3
  width 3
  root
 controller t3 0/1/1/0/0
  mode t1
  framing auto-detect
  root
 controller t1 0/1/1/0/0/0
  framing esf
  channel-group 0
   timeslots 1:3:7:9
   show configuration
  root
 interface serial 0/1/1/0/0/0:0
 encapsulation hdlc
 ip address 10.10.10.10 255.255.255.255
 no shutdown
  commit
show controllers sonet 0/1/1/0
```

## Channelized SONET in VT1.5 Mode and T1 Channelization to NxDS0

**Note** This example is not supported on the 1-Port Channelized OC-48/STM-16 SPA.

The following example shows how to configure SONET channelized to NxDS0s through SONET VT1.5 mode:

```
configure
 controller sonet 0/1/1/0
  clock source internal
  framing sonet
  sts 1
  mode vt15-t1
  root
controller t1 0/1/1/0/0/0
  channel-group 0 timeslots 1
  channel-group 1 timeslots 2-3
  commit
```

## Channelized Packet over SONET Configuration: Example

The following example shows Channelized Packet over SONET configuration.

```
configure
 controller sonet 0/1/1/0
  clock source internal
  framing sonet
  sts 1
  mode pos scramble
  width 3
```

```
  root
 interface POS 0/1/1/0
  encapsulation hdlc
  pos crc 32
  mtu 4474
  no shutdown
  commit
show interfaces pos 0/1/1/0
```

## SONET Clear Channel T3 Configuration: Example

The following example shows SONET clear channel configuration for T3:

```
configure
 controller sonet 0/1/1/0
  clock source internal
  framing sonet
  sts 1
  mode t3
  root
 controller t3 0/1/1/0/0
  mode serial
  root
 interface serial 0/1/1/0/0/0:0
 encapsulation ppp
 ip address 10.10.10.10 255.255.255.255
 no shutdown
  commit
show controllers sonet 0/1/1/0
```

## Channelized SONET APS Single Router Configuration: Example

The following example shows SONET APS configuration for a single router.

```
 aps group 1
  channel 0 local SONET 0/0/0/1
  channel 1 local SONET 0/0/0/2
  signalling sonet
  commit
show aps
show aps group 3
```

## Channelized SONET APS Multirouter Configuration: Example

The following example shows SONET APS multirouter configuration.

```
 aps group 1
  channel 0 local SONET 0/0/0/1
  channel 1 remote 172.18.69.123
  signalling sonet
  commit
show aps
show aps group 3
```

# Channelized SDH Examples


- Channelized SDH AU-3 Configuration: Examples, page 302
- Channelized SDH AU-4 Configuration: Examples, page 303


## Channelized SDH AU-3 Configuration: Examples

This section includes the following configuration examples:


- Channelized SDH AU-3 to VC-3 and Clear Channel T3/E3: Examples, page 302
- Channelized SDH AU-3 to TUG-2, VC-11, T1 and NxDS0s: Example, page 302
- Channelized SDH AU-3 to TUG-2, VC-12, E1 and NxDS0s: Example, page 303


### Channelized SDH AU-3 to VC-3 and Clear Channel T3/E3: Examples

The following example shows how to configure SDH AU-3 to VC-3 and clear channel T3:

```
configure
 controller sonet 0/1/1/0
  clock source internal
  framing sdh
  au 1
  width 1
  mode t3
  root
 controller t3 0/1/1/0/1
  mode serial
  commit
```

The following example shows how to configure SDH AU-3 to VC-3 and clear channel E3:

```
configure
 controller sonet 0/1/1/0
  clock source internal
  framing sdh
  au 1
  width 1
  mode e3
  root
 controller e3 0/1/1/0/1
  mode serial
  commit
```

### Channelized SDH AU-3 to TUG-2, VC-11, T1 and NxDS0s: Example

**Note** This example is not supported on the 1-Port Channelized OC-48/STM-16 SPA.

The following example shows how to configure SDH AU-3 to TUG-2, VC-11 and channelized T1 to NxDS0s:

```
configure
 controller sonet 0/1/1/0
  clock source internal
  framing sdh
  au 1
```

```
  mode c11-t1
  width 1
  root
 controller T1 0/1/1/0/0/1/1
  channel-group 0
   timeslots 1-12
   show configuration
  root
 interface serial 0/1/1/0/1/1:0
 encapsulation ppp
 ip address 10.10.10.10 255.255.255.255
 no shutdown
  commit
show controllers sonet 0/1/1/0
```

### Channelized SDH AU-3 to TUG-2, VC-12, E1 and NxDS0s: Example

**Note** This example is not supported on the 1-Port Channelized OC-48/STM-16 SPA.

The following example shows how to configure SDH AU-3 to TUG-2, VC-12 and channelized E1 to NxDS0s:

```
configure
 controller sonet 0/1/1/0
  clock source internal
  framing sdh
  au 1
  mode c12-e1
  width 1
  root
 controller e1 0/1/1/0/0/1/1
  channel-group 0
   timeslots 1-12
   show configuration
  root
 interface serial 0/1/1/0/1/1:0
 encapsulation ppp
 ip address 10.10.10.10 255.255.255.255
 no shutdown
  commit
show controllers sonet 0/1/1/0
```

## Channelized SDH AU-4 Configuration: Examples

This section includes the following configuration examples:


- Channelized SDH AU-4 to TUG-3 and Clear Channel T3/E3: Examples, page 303
- Channelized SDH AU-4 to TUG-3, TUG-2, and T1/E1 and NxDS0: Examples, page 304


### Channelized SDH AU-4 to TUG-3 and Clear Channel T3/E3: Examples

The following exampe shows SDH AU-4 channelization to TUG-3 and clear channel T3:

```
configure
 controller sonet 0/4/0/0
  framing sdh
  au 1
  width 3
```

```
  mode tug3
  tug3 1
   mode t3
   root
 controller t3 0/4/0/0/1/1
  mode serial
  commit
```

The following exampe shows SDH AU-4 channelization to TUG-3 and clear channel E3:

```
configure
 controller sonet 0/4/0/0
  framing sdh
  au 1
  width 3
  mode tug3
  tug3 1
   mode e3
   root
 controller e3 0/4/0/0/1/1
  mode serial
  commit
```

### Channelized SDH AU-4 to TUG-3, TUG-2, and T1/E1 and NxDS0: Examples

**Note** Channelization to T1/E1 and NxDS0s is not supported on the 1-Port Channelized OC-48/STM-16 SPA.

The following example shows SDH AU-4 configuration with unframed E1 controllers and serial interfaces:

```
configure
 controller sonet 0/1/2/0
  framing sdh
  au 1
  width 3
  mode tug3
  tug3 1
   mode c12-e1
!
  tug3 2
   mode c12-e1
!
  tug3 3
   mode c12-e1
!
controller E1 0/1/2/0/1/1/1/1
framing unframed
!
controller E1 0/1/2/0/1/1/1/2
framing unframed
!
controller E1 0/1/2/0/1/1/1/3
framing unframed
!
interface Serial0/1/2/0/1/1/1/1:0
encapsulation ppp
multilink
  group 1
!
interface Serial0/1/2/0/1/1/1/2:0
encapsulation ppp
```

```
multilink
  group 1
!
!
interface Serial0/1/2/0/1/1/1/3:0
encapsulation ppp
multilink
  group 1
!
```

This example shows SDH AU-4 configuration with E1 controller channel groups and serial interfaces:

```
configure
 controller SONET0/3/2/0
   framing sdh
   au 1
   width 3
   mode tug3
   tug3 1
   mode c12-e1
!
   tug3 2
   mode c12-e1
!
   tug3 3
   mode c12-e1
!
controller E1 0/3/2/0/1/1/1/1
 framing crc4
 channel-group 0
  timeslots 1-4
!
controller E1 0/3/2/0/1/1/3/1
 framing crc4
 channel-group 0
  timeslots 1-31
!
controller E1 0/3/2/0/1/1/1/2
 framing crc4
 channel-group 0
  timeslots 1-31
!
controller E1 0/3/2/0/1/2/7/3
 framing crc4
 channel-group 0
  timeslots 1-5
!
 channel-group 1
  timeslots 6-31
!
interface Serial0/3/2/0/1/1/1/1:0
 encapsulation frame-relay IETF
 frame-relay lmi-type ansi
 frame-relay intf-type dce
!
interface Serial0/3/2/0/1/1/1/1:0.1 point-to-point
 ipv4 address 192.168.200.2 255.255.255.252
 ipv4 verify unicast source reachable-via rx
 pvc 100
  encap ietf
!
interface Serial0/3/2/0/1/1/3/1:0
 encapsulation ppp
 multilink
  group 1
```

```
!
interface Serial0/3/2/0/1/1/1/2:0
 encapsulation ppp
 multilink
  group 1
```

# Additional References

These sections provide references related to channelized SONET configuration.

## Related Documents

| Related Topic | Document Title |
|---|---|
| Cisco IOS XR master command reference | *Cisco IOS XR Master Commands List* |
| Cisco IOS XR interface configuration commands | *Cisco IOS XR Interface and Hardware Component Command Reference* |
| Initial system bootup and configuration information for a router using the Cisco IOS XR software | *Cisco IOS XR Getting Started Guide* |
| Information about user groups and task IDs | *Configuring AAA Services on Cisco IOS XR Software* module of *Cisco IOS XR System Security Configuration Guide* |

## 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:<br>http://www.cisco.com/go/mibs |

## RFCs

| 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/support |

# Configuring Circuit Emulation over Packet on Cisco IOS XR Software

This module describes the configuration of Circuit Emulation over Packet (CEoP) shared port adapters (SPAs) on the Cisco XR 12000 Series Router.

**Feature History for Configuring CEoP on Cisco XR 12000 Series Router**

| Release | Modification |
|---|---|
| Release 4.2.0 | • Support for Circuit Emulation Service over Packet Switched Network was added in these SPAs:<br>– Cisco 1-port Channelized OC3/STM-1 Circuit Emulation and Channelized ATM SPA<br>– Cisco 2-Port Channelized T3/E3 Circuit Emulation and Channelized ATM SPA<br>– Cisco 24-Port Channelized T1/E1 Circuit Emulation and Channelized ATM SPA |

# Contents


- Prerequisites for Configuration, page 310
- Overview of Circuit Emulation over Packet Service, page 310
- Information About Configuring CEoP Channelized SONET/SDH, page 311
- Clock Distribution, page 317
- Mode Change for CEoP SPA, page 319
- How to implement CEM, page 320
- Configuring Clocking, page 345
- Configuration Examples for CEM, page 348
- Additional References, page 352

# Prerequisites for Configuration

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 the Circuit Emulation over Packet (CEoP) service on your router, ensure that these conditions are met:

- You must have one of these SPAs installed in your chassis:
  - Cisco 1-port Channelized OC3/STM-1 Circuit Emulation and Channelized ATM SPA
  - Cisco 2-Port Channelized T3/E3 Circuit Emulation and Channelized ATM SPA
  - Cisco 24-Port Channelized T1/E1 Circuit Emulation and Channelized ATM SPA
- You should know how to apply and specify the SONET controller name and *interface-path-id* with the generalized notation *rack/slot/module/port*. The SONET controller name and *interface-path-id* are required with the **controller sonet** command.
- You should know how to apply and specify the T3/E3 and T1/E1 controller name and *interface-path-id* with the generalized notation *rack/slot/module/port*. The T3/E3, T1/E1 controller name and *interface-path-id* are required with the **controller {T3|E3|T1|E1}** command.

# Overview of Circuit Emulation over Packet Service

Circuit Emulation over Packet (CEoP) is a way to carry TDM circuits over packet switched network. Circuit Emulation over Packet is the imitation of a physical connection. The goal of CEoP is to replace leased lines and legacy TDM networks. This feature allows network administrators to use their existing IP/MPLS network to provide leased-line emulation services or to carry data streams or protocols that do not meet the format requirements of other multiservice platform interfaces. CEoP puts TDM bits into the packets, encapsulates them into appropriate header and then sends through PSN. The receiver side of CEoP restores the TDM bit stream from packets.

CEoP SPAs are half-height (HH) Shared Port Adapters (SPA) and the CEoP SPA family consists of 24xT1/E1/J1, 2xT3/E3, and 1xOC3/STM1 unstructured and structured (NxDS0) quarter rate, half height SPAs. The CEoP SPAs provide bit-transparent data transport that is completely protocol independent.

CEoP has two major modes:

- Unstructured mode is called SAToP (Structure Agnostic TDM over Packet) — SAToP does not look what is inside the incoming data and considers it as a pure bit stream.
- Structured mode is named CESoPSN (Circuit Emulation Service over Packet Switched Network) — CESoPSN is aware of the structure of the incoming TDM bit stream at DS0 level.

CESoPSN and SAToP can use MPLS, UDP/IP, and L2TPv3 for the underlying transport mechanism.

**Note** The Cisco IOS XR Release 4.2.0 supports only MPLS transport mechanism.

These SPAs are the first Cisco router interfaces designed to meet the emerging standards for Circuit Emulation Services over Packet Switched Network (CESoPSN) and Structure-Agnostic Transport over Packet (SAToP) transport.

In Cisco IOS XR Release 4.2.0, the CEM functionality is supported on the following CEoP SPAs:

- Cisco 1-port Channelized OC3/STM-1 Circuit Emulation and Channelized ATM SPA

- Cisco 2-Port Channelized T3/E3 Circuit Emulation and Channelized ATM SPA
- Cisco 24-Port Channelized T1/E1 Circuit Emulation and Channelized ATM SPA

In SAToP mode, these SPAs do not assume that data has any predefined format or structure. They simply regard the data as an arbitrary bit stream. All data bits are simply transported to a defined destination encapsulated in IP/MPLS packets. In CESoPSN mode the carrier has defined format. The SPAs support a full range of E1 and T1 framing. CESoPSN applications can save utilized bandwidth by selecting only valid timeslots for transmission. Some primary applications include:

- Transporting 2G and 3G network traffic over packet networks, for mobile operators. Mobile service providers are implementing high-speed data networks with HSDPA to support new revenue-generating services. The SPA is uniquely positioned for multigenerational migration of mobile networks (2G and 3G), simultaneously carrying TDM and ATM traffic over IP/MPLS networks. This technology provides a mechanism to enable IP/MPLS to the cell site, which can eventually be in place to transport the mobile traffic over IP from end to end.
- T3/E3 circuit emulation for leased-line replacement.
- T1/E1 circuit emulation for leased-line replacement.
- PBX to PBX connectivity over PSN.
- High density SS7 backhaul over IP/MPLS.
- Inter-MSC connectivity.
- Preencrypted data for government, defense, or other high-security applications.
- Proprietary synchronous or asynchronous data protocols used in transportation, utilities, and other industries.
- Leased-line emulation service offerings in metropolitan (metro) Ethernet or WAN service provider environments.

For more information on Circuit Emulation service concepts, configuration, and example, see the *Implementing MPLS Layer 2 VPNs* module in the *Cisco IOS XR Virtual Private Network Configuration Guide for the Cisco XR 12000 Series Router.*

# Information About Configuring CEoP Channelized SONET/SDH

To configure the Circuit Emulation over Packet Channelized SONET/SDH, you must understand the following concepts:

- Channelized SONET and SDH Overview, page 311
- Default Configuration Values for Channelized SONET/SDH, page 316

## Channelized SONET and SDH Overview

Synchronous Optical Network (SONET) is an American National Standards Institute (ANSI) specification format used in transporting digital telecommunications services over optical fiber.

Channelized SONET provides the ability to transport SONET frames across multiplexed T3/E3 and virtual tributary group (VTG) channels.

The Cisco 1-port Channelized OC3/STM-1 Circuit Emulation and Channelized ATM SPA does not support the following modes:

- Clear Channel OC3

- Clear Channel E3
- Clear Channel T3

SONET uses Synchronous Transport Signal (STS) framing. An STS is the electrical equivalent to an optical carrier 1 (OC-1).

A channelized SONET interface is a composite of STS streams, which are maintained as independent frames with unique payload pointers. The frames are multiplexed before transmission.

When a line is channelized, it is logically divided into smaller bandwidth channels called *paths*. These paths carry the SONET payload. The sum of the bandwidth on all paths cannot exceed the line bandwidth.

When a line is not channelized, it is called *clear channel*, and the full bandwidth of the line is dedicated to a single channel that carries broadband services.

Channelizing a SONET line consists of two primary processes:

- Configuring the controller
- Configuring the interface into channelized paths

You configure the controller first by setting the mode of the STS path.

When the mode is specified, the respective controller is created, and the remainder of the configuration is applied on that controller. For example, mode T3 creates a T3 controller. The T3 controller can then be configured to a serial channel, or it can be further channelized to carry T1s, and those T1s can be configured to serial interfaces.

The Cisco 1-Port Channelized OC-3/STM-1 Circuit Emulation and Channelized ATM SPA does not support the following modes:

- Clear Channel OC3
- Clear Channel E3
- Clear Channel T3

On a Cisco 1-Port Channelized OC-3/STM-1 SPA, the default configuration consists of the following paths that are already configured when the SONET card is installed.

- STS 1
- STS 2
- STS 3

Each STS path can be independently configured into T3s, E3s, or VTGs, and so on.

Depending on the support for your installed SPA, each STS path can be independently configured into T3s, E3s, or VTGs, and so on.The following level of SONET channelization modes are supported in CEoP SPA:

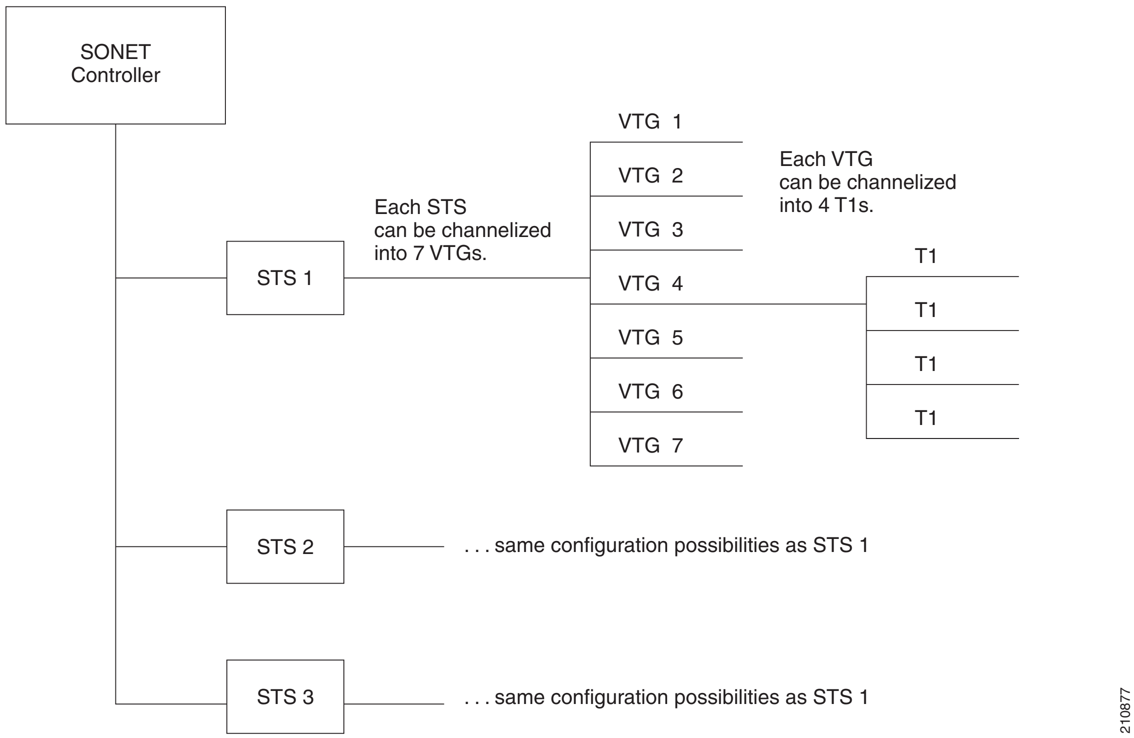
OC3->STS-1->VTG-> VT1.5 -> Unframed T1

OC3->STS-1->VTG-> VT1.5 -> T1 -> DS0

Figure 19 shows the VTG paths that can be configured.

**Note** Only VTG paths are supported on the Cisco 1-Port Channelized OC-3/STM-1 SPA on the Cisco XR12000 Series Router.

***Figure 19 SONET VTG Channelized Paths***




Synchronous Digital Hierarchy (SDH) is the international equivalent of SONET.

SDH uses Synchronous Transport Mode (STM) framing. An STM-1 is the electrical equivalent to 3 optical carrier 1s (OC-1s). A Synchronous Transport Module (STM) signal is the Synchronous Digital Hierarchy (SDH) equivalent of the SONET STS, but the numbers are different for each bandwidth. In this guide, the STM term refers to both path widths and optical line rates. The paths within an STM signals are called administrative units (AUs).

A summary of the basic terminology differences between SONET and SDH is as follows:

- SONET STS is equivalent to SDH administrative unit (AU)
- SONET VT is equivalent to SDH tributary unit (TU)
- SDH basic building blocks are STM-1 (equivalent to STS-3) and STM-0 (equivalent to STS-1)

An administrative unit (AU) is the information structure that provides adaptation between the higher-order path layer and the multiplex section layer. It consists of an information payload (the higher-order virtual container) and an administrative unit pointer, which indicates the offset of the payload frame start relative to the multiplex section frame start.

An AU can be channelized into tributary units (TUs) and tributary unit groups (TUGs).

An administrative unit 4 (AU-4) consists of three STM-1s or an STM-3.

An administrative unit 3 (AU-3) consists of one STM-1.

An administrative unit group (AUG) consists of one or more administrative units occupying fixed, defined positions in an STM payload.

The Table 7 shows the commonly used notations and terms in SONET standards and their SDH equivalents.

*Table 7 SONET and SDH Terminology Equivalencies*

| SONET Term | SDH Term |
|---|---|
| SONET | SDH |
| STS-3c | AU-4 |
| STS-1 | AU-3 |
| VT | TU |
| SPE | VC |
| Section | Regenerator Section |
| Line | Multiplex Section |
| Path | Path |

These are the levels of SDH channelization that are supported on the CEoP SPA:

- For E1 :
  - STM1-> AU-4 -> TUG-3 -> TUG-2 ->VC12-> Unframed E1
  - STM1-> AU-4 -> TUG-3 -> TUG-2 ->VC12-> E1 -> DS0
- For T1 :
  - STM1-> AU-3-> TUG-2 -> VC11->Unframed T1
  - STM1-> AU-3-> TUG-2 -> VC11->T1 -> DS0

Figure 20 shows an example of SDH AU-3 paths that can be configured on the CEoP SPA.

***Figure 20 SDH AU3 Paths***

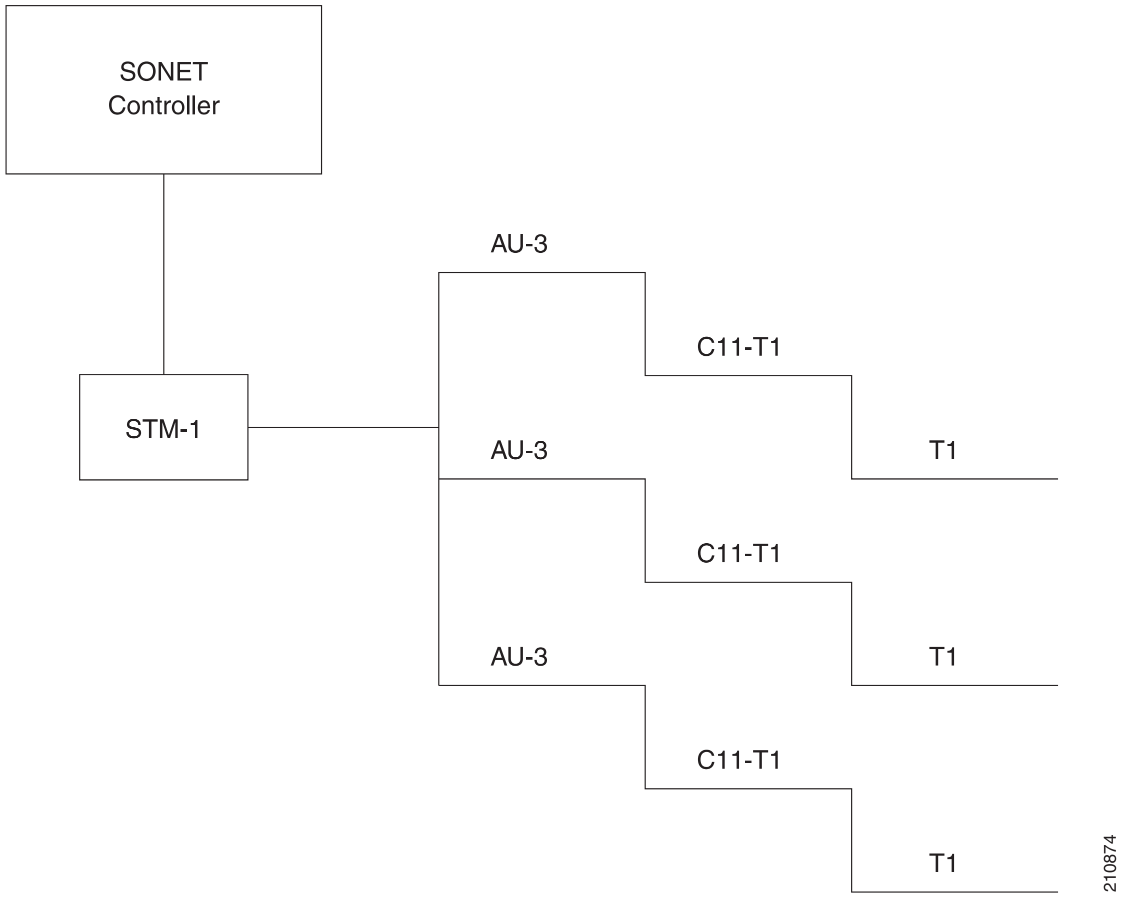
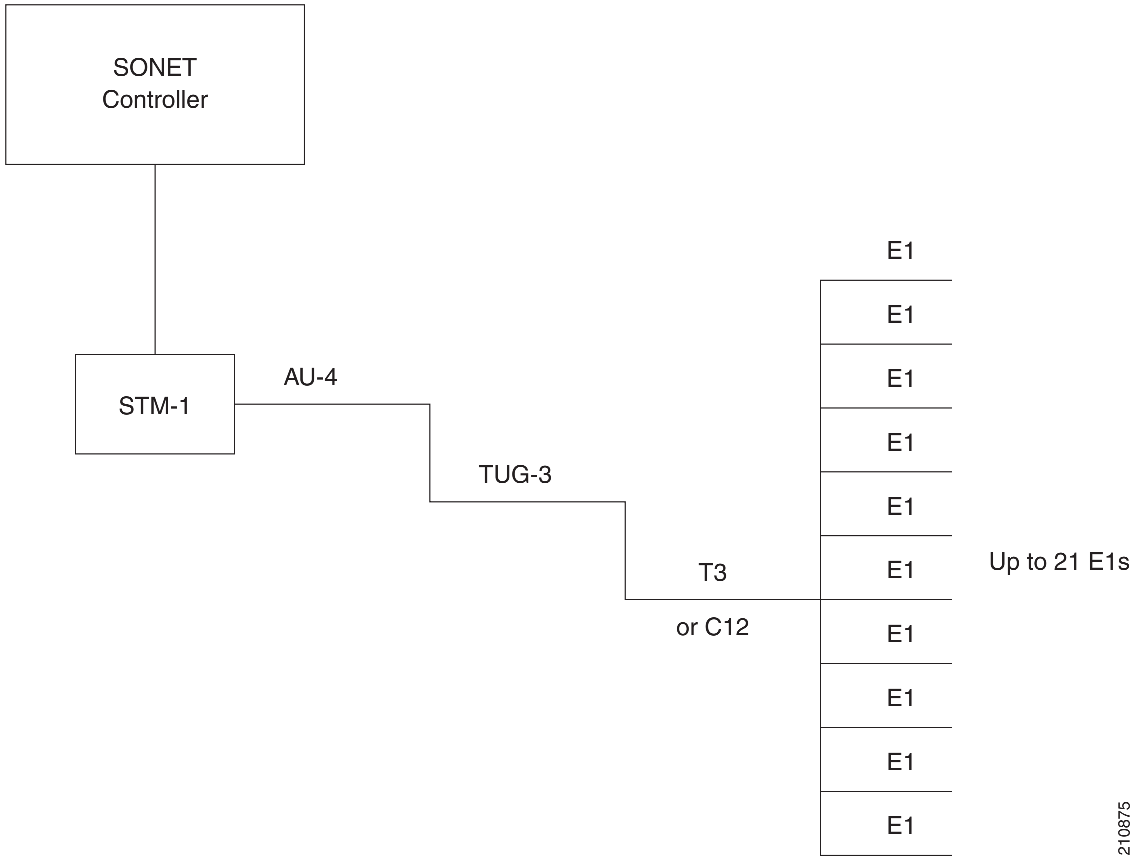


Figure 21 shows the SDH AU4 paths that can be configured on the CEoP SPA.

***Figure 21 SDH AU4 Paths***




# Default Configuration Values for Channelized SONET/SDH

Table 8 describes the default configuration parameters that are present on the Channelized SONET/SDH.

***Table 8 SONET/SDH Controller Default Cit onfiguration Values***

| Parameter | Default Value | Configuration File Entry |
|---|---|---|
| Clock source | line | **clock source** {**internal** \| **line**} |
| SONET framing | sonet | **hw-module sub-slot** ***node-id*** **cardtype** {**sonet** \| **sdh**} |

# Clock Distribution

Clocking distribution in the CEoP SPA can be done in these ways:

- Synchronous Clocking — With synchronous clocking, TDM lines on source and destination are synchronized to the same clock delivered by some means of physical clock distribution (SONET/SDH, BITS, GPS, and so on). The clock to the particular TDM line can be delivered from
  - Line: the transmit clock is from the receiver of the same physical line
  - Internal: the transmit clock is taken from line card and can be derived either from an internal free running oscillator or from another physical line
  - Recovered: In-band pseudowire-based activeclock recovery on a CEM interface which is used to drive the transmit clock

The number of recovered clocks that can be configured for CEoP SPA are:

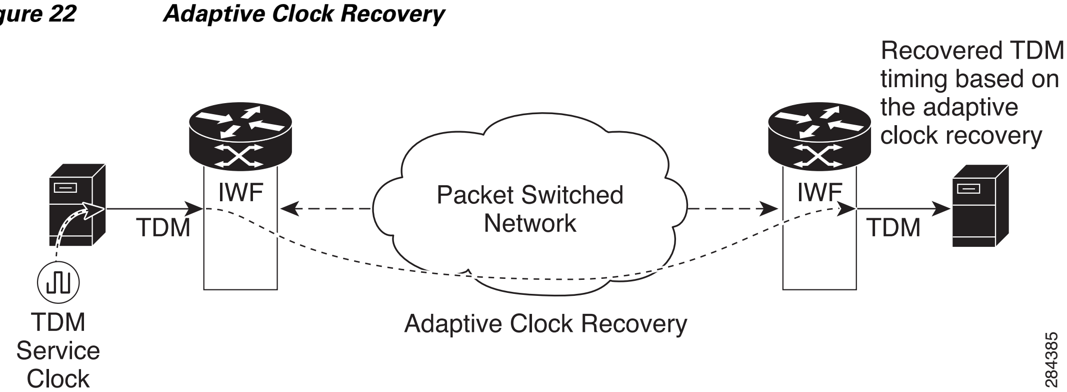
  - Cisco 24-Port Channelized T1/E1 Circuit Emulation and Channelized ATM SPA : 24 clocks for each SPA.
  - Cisco 2-Port Channelized T3/E3 Circuit Emulation and Channelized ATM SPA : 10 clocks for each SPA in the T1/E1 mode and 2 clocks for each SPA in the T3/E3 mode.
  - Cisco 1-port Channelized OC3/STM-1 Circuit Emulation and Channelized ATM SPA : 10 clocks per SPA in the T1/E1 mode.

- Adaptive Clocking — Adaptive clocking is used when the routers do not have a common clock source. See Figure 22. The clock is derived based on packet arrival rates. Two major types of adaptive clock recovery algorithms are:
  - Based on dejitter buffer fill level
  - Based on packet arrival rate

The clock quality depends on packet size, has less tolerance to packet loss/corruption and introduces unnecessary delay in order to have sufficient number of packets in the buffer for clock recovery. The dejitter buffer size determines the ability of the emulated circuit to tolerate network jitter. The dejitter buffer in CEoP software is configurable up to a maximum of 500 milliseconds.

**Note** The CEoP SPA hardware supports only the packet arrival rate algorithm.

***Figure 22 Adaptive Clock Recovery***



**Note** CEM supports only adaptive clocking in the Cisco XR 12000 Series Router.

For a sample CEM interface configuration, refer Circuit Emulation Interface Configuration: Examples, page 348.

# Mode Change for CEoP SPA

This section explains about the command to change the mode of operation of CEoP SPA. At any specific instance, only one of this feature combination is possible on all the CEoP SPAs:

- Combination 1 : ATM + IMA + L3QoS
- Combination 2 : ATM + IMA + CEM

The **hw-module subslot** *nodeid* **mode CEM** command is introduced in Cisco XR 12000 Series Router to configure the modes. The ATM + IMA + L3QoS is the default mode. These sections describe two scenarios for both the mode combinations.

## Host is configured and SPA is running in ATM + IMA + CEM mode

When L3QoS is configured on an ATM / IMA interface and committed, the commit operation fails, as the mode is CEM mode.

When the **hw-module subslot** *nodeid* **mode CEM** command is executed to switch the mode to ATM + IMA + L3QoS mode, with CEM configuration already present in running configuration, the commit fails with the message that CEM needs to be unconfigured before the change of mode to ATM + IMA + L3QoS. You must manually remove the CEM configuration before the SPA can be reloaded and booted in this new mode.

## Host is configured and SPA is running in ATM + IMA + L3QoS mode

When CEM is configured and committed, the commit operation fails, as the mode is L3QoS mode.

When the **hw-module subslot** *nodeid* **mode CEM** command is executed to switch the mode to ATM + IMA + CEM mode, with L3QoS configuration already present in running configuration, the commit fails with the message that L3QoS needs to be unconfigured before the change of mode to ATM + IMA + CEM. You must manually remove the L3QoS configuration before the SPA can be reloaded and booted in this new mode.

The Table 9 shows the support for L3QoS functionality on the three CEoP SPAs.

*Table 9 Support for L3QoS functionality*

| CEoP SPA Variant | (ATM + IMA + CEM) | (ATM + IMA + L3QoS) |
|---|---|---|
| Cisco 24-Port Channelized T1/E1 Circuit Emulation and Channelized ATM SPA | Yes | Yes |
| Cisco 2-Port Channelized T3/E3 Circuit Emulation and Channelized ATM SPA | Yes | Yes |
| Cisco 1-port Channelized OC3/STM-1 Circuit Emulation and Channelized ATM SPA | Yes | No |

# How to implement CEM

This section contains the following procedures:


- Configuring SONET VT1.5-Mapped T1 Channels and Creating CEM Interface, page 320
- Configuring SDH AU-3 Mapped to C11-T1 or C12-E1, page 323
- Configuring the Cisco 24-Port Channelized T1/E1 Circuit Emulation and Channelized ATM SPA and Creating CEM Interface, page 329
- Configuring the Cisco 2-Port Channelized T3/E3 Circuit Emulation and Channelized ATM SPA and Creating CEM Interface, page 332
- Configuring CEM Interface, page 337
- Configuring Clocking, page 345
- Show Commands for CEM, page 347


## Configuring SONET VT1.5-Mapped T1 Channels and Creating CEM Interface

In the case of Cisco 1-port Channelized OC3/STM-1 CEoP SPA, the STS stream can be channelized into the VT1.5 mapped T1 channel.

This task explains how to configure a SONET line into VT-mapped T1 Channels.

### Prerequisites

None.

#### Restrictions

Channelized SONET STS stream with VT1.5-T1 mapping is supported on the following SPA:

- Cisco 1-Port Channelized OC-3/STM-1 SPA

#### SUMMARY STEPS

1. **configure**
2. **hw-module subslot** *node-id* **cardtype** *type*
3. **hw-module subslot** *node-id* **mode** *CEM*
4. **commit**
5. **controller sonet** *interface-path-id*
6. **sts** *number*
7. **mode** *mode*
8. **root**
9. **controller t1** *interface-path-id*
10. **cem-group unframed**
11. **controller t1** *interface-path-id*
12. **cem-group framed** *group-number* **timeslots** *range1-range2*

13. **no shutdown**
14. **end**
    or
    **commit**
15. **show runn interface cem** *interface-path-id*

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| Step 1 | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| Step 2 | **hw-module subslot** *node-id* **cardtype {sonet\| sdh}**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# hw-module subslot 0/3/0 sonet` | The **hw-module subslot** *node-id* **cardtype** *type* command configures the SPA to function in sonet/sdh mode.<br><br>SONET framing (**sonet**) is the default. Whenever there is a change in framing mode (sonet/sdh), the SPA will be reloaded automatically. Reload will happen only when all the CEM Interface, T1 Controller and Sonet Controller configurations are removed completely. This is not applicable when you configure the first time because T1 controller and interface configurations would not exist.<br><br>This configuration is mandatory for CEoP SPA to work normally in one of the framing modes. When you configure for the first time, it will not cause a SPA reload, if the cardtype is set to Sonet. |
| Step 3 | **hw-module subslot** *node-id* **mode CEM**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# hw-module subslot 0/3/0 mode CEM` | The **hw-module subslot** *node-id* **mode** *CEM* command configures the SPA to function in CEM mode. |
| Step 4 | **commit** | Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |
| Step 5 | **controller sonet** *interface-path-id* | Enters controller configuration submode and specifies the SONET controller name and instance identifier with the *rack/slot/module/port/controllerName* notation. |
| Step 6 | **sts** *number*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# sts 1` | Configures the STS stream specified by *number*. The range is from1 to 3. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 7** | **mode** *mode*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-stsPath)# mode t1` | Sets the mode of interface at the STS level. The possible modes are:<br>• vt15-t1—SONET path carrying virtual tributary 1.5 T1s (VT15 T1) |
| **Step 8** | **root**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-stsPath)# root` | Exits to global configuration mode. Go to step 7, if you want to create an structure agnostic CEM interface. Go to step 9, if you want to create a structure aware CEM interface. |
| **Step 9** | **controller t1** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller t1 0/0/1/0/1/4/1` | Enters T1 controller configuration submode and specifies the T1 controller name and *interface-path-id* with the *rack/slot/module/port/sts-num/vtg-num/T1-num* notation. |
| **Step 10** | **cem-group unframed**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# cem-group unframed` | Creates an structure agnostic CEM interface. |
| **Step 11** | **controller t1** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller t1 0/0/1/0/1/5/1` | Enters T1 controller configuration submode and specifies the T1 controller name and *interface-path-id* with the *rack/slot/module/port/sts-num/vtg-num/T1-num* notation. |
| **Step 12** | **cem-group framed** *group-number* **timeslots** *range1-range2*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# cem-group framed 0 timeslots 1` | Creates an structure aware CEM interface. The **timeslots** keyword specifies the time slots for the interface by range with the *range1-range2* notation. |
| **Step 13** | **no shutdown**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# no shutdown` | Removes the shutdown configuration.<br>**Note** Removal of the shutdown configuration eliminates the forced administrative down on the interface, enabling it to move to an up or down state (assuming that the parent SONET layer is not configured administratively down). |

| | Command or Action | Purpose |
|---|---|---|
| Step 14 | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/00/CPU0:router(config-sonet)# end`<br>or<br>`RP/0/0/CPU0:router(config-sonet)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 15 | **show runn interface cem** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router# show runn interface cem 0/0/2/0/1/1/1/1:1` | Verifies the CEM interface configuration. |

# Configuring SDH AU-3 Mapped to C11-T1 or C12-E1

This section includes the following tasks:

- Configuring SDH AU-3 Mapped to C11-T1 and Creating CEM Interface, page 323
- Configuring SDH AU-3 Mapped to C12-E1 and Creating CEM Interface, page 326

## Configuring SDH AU-3 Mapped to C11-T1 and Creating CEM Interface

This task explains how to configure SDH AU-3 with c11-t1 mapping.

### Prerequisites

- You should know how to configure the SONET/SDH controller.

### Restrictions

Channelized SDH AU-3 with c11-t1 mapping is supported on the following SPA:

- Cisco 1-Port Channelized OC-3/STM-1 SPA

## SUMMARY STEPS

1. **configure**
2. **hw-module subslot** *node-id* **cardtype** *type*
3. **hw-module subslot** *node-id* **mode** *CEM*
4. **commit**
5. **controller sonet** *interface-path-id*
6. **au** *number*
7. **mode** *mode*
8. **root**
9. **controller t1** *interface-path-id*
10. **cem-group unframed**
11. **controller t1** *interface-path-id*
12. **cem-group framed** *group***-***number* **timeslots** *range1-range2*
13. **no shutdown**
14. **end**
    or
    **commit**
15. **show runn interface cem** *interface-path-id*

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **hw-module sub-slot** *node-id* **cardtype** *type*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# hw-module sub-slot <> cardtype sdh` | Configures the controller for Synchronous Digital Hierarchy (SDH) framing. The **hw-module subslot** *node-id* **cardtype** *type* command configures the SPA to function in sonet/sdh mode.<br>This command when committed results in automatic reload of SPA. Reload happens only when all the CEM interface, T1 Controller and Sonet Controller configurations are removed completely. This is not applicable when you configure the first time because T1 controller and interface configurations would not exist.<br>This configuration is mandatory for CEoP SPA to work normally in one of the framing modes. SONET framing (**sonet**) is the default. |
| **Step 3** | **hw-module subslot** *node-id* **mode CEM**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# hw-module subslot 0/3/0 mode CEM` | The **hw-module subslot** *node-id* **mode** *CEM* command configures the SPA to function in CEM mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **commit** | Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |
| **Step 5** | **controller sonet** *interface-path-id* | Enters controller configuration submode and specifies the SDH controller name and instance identifier with the *rack/slot/module/port/controllerName* notation. |
| **Step 6** | **au** *number*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# au 1` | Specifies the administrative unit (AU) group and enters AU path configuration mode. For AU-3, the valid range is:<br>• 1 to 3—1-Port Channelized OC-3/STM-1 SPA<br>**Note** The **au** command does not specify the AU type. It specifies the number of the AU group for the AU type that you want to configure. The range for the AU command varies based on whether you are configuring AU-3 or AU-4. |
| **Step 7** | **mode** *mode*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-auPath)# mode c11-t1` | Sets the mode of interface at the AU level. AU-3 paths can be mapped to c11-t1 on supported SPAs. |
| **Step 8** | **root**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-auPath)# root` | Exits to global configuration mode. |
| **Step 9** | **controller t1** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller T1 0/0/2/0/1/1/4` | Enters T1 controller configuration submode and specifies the T1 controller name and *interface-path-id* with the *rack/slot/module/port/auNum/t1Num* notation. |
| **Step 10** | **cem-group unframed**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# cem-group unframed` | Creates an structure agnostic CEM interface. |
| **Step 11** | **controller t1** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller t1 0/0/2/0/1/1/7` | Enters T1 controller configuration submode and specifies the T1 controller name and *interface-path-id* with the *rack/slot/module/port/auNum/t1Num* notation. |
| **Step 12** | **cem-group framed** *group-number* **timeslots** *range1-range2*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# cem-group framed 1 timeslots 2-3` | Creates an structure aware CEM interface. The **timeslots** keyword specifies the time slots for the interface by range with the *range1-range2* notation. |

| | Command or Action | Purpose |
|---|---|---|
| Step 13 | **no shutdown**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# no shutdown` | Removes the shutdown configuration.<br>**Note** Removal of the shutdown configuration eliminates the forced administrative down on the interface, enabling it to move to an up or down state (assuming that the parent SONET layer is not configured administratively down). |
| Step 14 | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# end`<br>or<br>`RP/0/0/CPU0:router(config-sonet)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 15 | **show runn interface cem** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router# show runn interface cem 0/0/2/0/1/1/1/1:1` | Verifies the CEM interface configuration. |

## Configuring SDH AU-3 Mapped to C12-E1 and Creating CEM Interface

This task explains how to configure SDH AU-3 with c12-e1 mapping.

### Prerequisites

- You should know how to configure the SONET/SDH controller.

### Restrictions

Channelized SDH AU-3 with c12-e1 mapping is supported on the following SPAs:

- Cisco 1-Port Channelized OC-3/STM-1 SPA

### SUMMARY STEPS

1. **configure**
2. **hw-module subslot** *node-id* **cardtype** *type*

3. **hw-module subslot** *node-id* **mode** *CEM*
4. **commit**
5. **controller sonet** *interface-path-id*
6. **au** *number*
7. **mode tug3**
8. **width** *number*
9. **tug3** *number*
10. **mode** *mode*
11. **root**
12. **controller e1** *interface-path-id*
13. **cem-group unframed**
14. **controller e1** *interface-path-id*
15. **cem-group framed** *group***-***number* **timeslots** *range1-range2*
16. **no shutdown**
17. **end**
    or
    **commit**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **hw-module sub-slot** *node-id* **cardtype** *type*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# hw-module sub-slot <> cardtype sdh` | The **hw-module subslot** *node-id* **cardtype** *type* command configures the SPA to function in sonet/sdh mode. This command when committed results in automatic reload of SPA. Reload happens only when all the CEM interface, E1 Controller and Sonet Controller configurations are removed completely. |
| **Step 3** | **hw-module subslot** *node-id* **mode CEM**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# hw-module subslot 0/3/0 mode CEM` | The **hw-module subslot** *node-id* **mode** *CEM* command configures the SPA to function in CEM mode. |
| **Step 4** | **commit** | Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

| | Command or Action | Purpose |
|---|---|---|
| Step 5 | **controller sonet** *interface-path-id* | Enters controller configuration submode and specifies the SDH controller name and instance identifier with the *rack/slot/module/port/controllerName* notation. |
| Step 6 | **au** *number*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# au 1` | Specifies the administrative unit (AU) group and enters AU path configuration mode. For AU-3, the valid range is:<br>• 1 to 3—1-Port Channelized OC-3/STM-1 SPA<br>**Note** The **au** command does not specify the AU type. It specifies the number of the AU group for the AU type that you want to configure. The range for the AU command varies based on whether you are configuring AU-3 or AU-4. |
| Step 7 | **mode tug3**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-auPath)# mode tug3` | Sets the mode of interface at the AU level. Currently only TUG3 is supported. |
| Step 8 | **width** *number*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-auPath)# width 3` | Configures the number of the AU streams. |
| Step 9 | **tug3** *number*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-auPath)#tug3 1` | Specifies the Tributary Unit Group (TUG) *number* and enters the config-tug3Path mode. The range is 1 to 3. |
| Step 10 | **mode** *mode*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-tug3Path)# mode c12-e1` | Sets the mode of interface at the tug3 level. The modes are:<br>• **c12-e1**—TUG-3 path carrying TU-12 to E1 |
| Step 11 | **root**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-auPath)# root` | Exits to global configuration mode. |
| Step 12 | **controller e1** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller E1 0/0/2/0/1/1/1` | Enters E1 controller configuration submode and specifies the E1 controller name and *interface-path-id* with the *rack/slot/module/port/auNum/tugNum/t1Num* notation. |
| Step 13 | **cem-group unframed**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# cem-group unframed` | Creates an structure agnostic CEM interface. |

| | Command or Action | Purpose |
|---|---|---|
| Step 14 | **controller e1** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller E1 0/0/2/0/1/1/7` | Enters E1 controller configuration submode and specifies the E1 controller name and *interface-path-id* with the *rack/slot/module/port/auNum/tugNum/t1Num* notation. |
| Step 15 | **cem-group framed** *group-number* **timeslots** *range1-range2*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# cem-group framed 0 timeslots 1` | Creates an structure aware CEM interface. |
| Step 16 | **no shutdown**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# no shutdown` | Removes the shutdown configuration.<br>**Note** Removal of the shutdown configuration eliminates the forced administrative down on the interface, enabling it to move to an up or down state (assuming that the parent SONET layer is not configured administratively down). |
| Step 17 | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# end`<br>or<br>`RP/0/0/CPU0:router(config-sonet)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

# Configuring the Cisco 24-Port Channelized T1/E1 Circuit Emulation and Channelized ATM SPA and Creating CEM Interface

This task explains how to configure the Cisco 24-Port Channelized T1/E1 Circuit Emulation and Channelized ATM SPA.

## SUMMARY STEPS

1. **configure**
2. **hw-module subslot** *node-id* **cardtype** *type*
3. **hw-module subslot** *node-id* **mode** *CEM*
4. **commit**
5. **controller t1** *interface-path-id*
6. **cem-group unframed**
7. **controller t1** *interface-path-id*
8. **cem-group framed** *group-number* **timeslots** *range1-range2*
9. **no shutdown**
10. **end**
    or
    **commit**
11. **show runn interface cem** *interface-path-id*

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| Step 1 | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| Step 2 | **hw-module subslot** *node-id* **cardtype {t1\| e1}**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1)# hw-module subslot 0/3/0 cardtype t1` | The **hw-module subslot** *node-id* **cardtype** *type* command configures the SPA to function in t1/e1 mode.<br><br>Reload will happen only when all the CEM interface, T1 Controller configurations are removed completely. This is not applicable when you configure the first time because T1 controller and interface configurations would not exist. |
| Step 3 | **hw-module subslot** *node-id* **mode CEM**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1)# hw-module subslot 0/3/0 mode CEM` | The **hw-module subslot** *node-id* **mode** *CEM* command configures the SPA to function in CEM mode. |
| Step 4 | **commit** | Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **`controller t1`** *`interface-path-id`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller t1 0/0/1/0/1` | Enters T1 controller configuration submode and specifies the T1 controller name and *interface-path-id* with the *rack/slot/module/port/T1num* notation. |
| **Step 6** | **`cem-group unframed`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1)# cem-group unframed` | Creates an structure agnostic CEM interface. |
| **Step 7** | **`controller t1`** *`interface-path-id`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller t1 0/0/1/0/1` | Enters T1 controller configuration submode and specifies the T1 controller name and *interface-path-id* with the *rack/slot/module/port/T1num* notation. |
| **Step 8** | **`cem-group framed`** *`group-number`* **`timeslots`** *`range1-range2`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1)# cem-group framed 0 timeslots 1` | Creates an structure aware CEM interface. The **timeslots** keyword specifies the time slots for the interface by range with the *range1-range2* notation. |
| **Step 9** | **`no shutdown`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# no shutdown` | Removes the shutdown configuration.<br><br>**Note** Removal of the shutdown configuration eliminates the forced administrative down on the interface, enabling it to move to an up or down state. |

| | Command or Action | Purpose |
|---|---|---|
| Step 10 | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1)# end`<br>or<br>`RP/0/0/CPU0:router(config-t1)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 11 | **show runn interface cem** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router# show runn interface cem 0/0/2/0/1:1` | Verifies the CEM interface configuration. |

# Configuring the Cisco 2-Port Channelized T3/E3 Circuit Emulation and Channelized ATM SPA and Creating CEM Interface

## T3/E3 Channelization Mode

This task explains how to configure the Cisco 2-Port Channelized T3/E3 Circuit Emulation and Channelized ATM SPA using T3/E3 channelization.

**Note** The T3/E3 channels can be channelized further into T1s or E1s, and the T1s or E1s can be channelized into time slots (DS0s), on the Cisco 2-Port Channelized T3/E3 Circuit Emulation and Channelized ATM SPA.

### SUMMARY STEPS

1. **configure**
2. **hw-module subslot** *node-id* **cardtype** *type*

3. **hw-module subslot** *node-id* **mode** *CEM*
4. **commit**
5. **controller {t3 | e3}** *interface-path-id*
6. **cem-group unframed**
7. **no shutdown**
8. **end**
   or
   **commit**
9. **show runn interface cem** *interface-path-id*

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **hw-module subslot** *node-id* **cardtype {t3\| e3}**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# hw-module subslot 0/3/0 cardtype t3` | The **hw-module subslot** *node-id* **cardtype** *type* command configures the SPA to function in t3/e3 mode.<br>Whenever there is a change in framing mode (t3/e3), the SPA will be reloaded automatically. Reload will happen only when all the CEM Interface, T3 Controller configurations are removed completely. This is not applicable when you configure the first time because T3 controller and interface configurations would not exist. |
| **Step 3** | **hw-module subslot** *node-id* **mode CEM**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# hw-module subslot 0/3/0 mode CEM` | The **hw-module subslot** *node-id* **mode** *CEM* command configures the SPA to function in CEM mode. |
| **Step 4** | **commit** | Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |
| **Step 5** | **controller {t3\|e3}** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller t3 0/0/1/0/4` | Enters T3/E3 controller configuration submode and specifies the T3/E3 controller name and *interface-path-id* with the *rack/slot/module/port/T3Num* notation. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 6** | **`cem-group unframed`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# cem-group unframed` | Creates an structure agnostic CEM interface. Only the unframed CEM interface is supported in this mode. |
| **Step 7** | **`no shutdown`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# no shutdown` | Removes the shutdown configuration.<br><br>**Note** Removal of the shutdown configuration eliminates the forced administrative down on the interface, enabling it to move to an up or down state. |
| **Step 8** | **`end`**<br>or<br>**`commit`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# end`<br>or<br>`RP/0/0/CPU0:router(config-t3)# commit` | Saves configuration changes.<br><br>• When you issue the **end** command, the system prompts you to commit changes:<br><br>`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`<br><br>– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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><br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |
| **Step 9** | **`show runn interface cem`** *`interface-path-id`*<br><br>**Example:**<br>`RP/0/0/CPU0:router# show runn interface cem 0/0/1/0/4:1` | Verifies the CEM interface configuration. |

## T1/E1 Channelization Mode

This task explains how to configure the Cisco 2-Port Channelized T3/E3 Circuit Emulation and Channelized ATM SPA using T1/E1 channelization.

### SUMMARY STEPS

1. **configure**
2. **hw-module subslot** *node-id* **cardtype** *type*
3. **hw-module subslot** *node-id* **mode** *CEM*

4. **commit**
5. **controller t3** *interface-path-id*
6. **mode {t1|e1}**
7. **controller t1** *interface-path-id*
8. **cem-group unframed**
9. **controller t1** *interface-path-id*
10. **cem-group framed** *group-number* **timeslots** *range1-range2*
11. **no shutdown**
12. **end**
    or
    **commit**
13. **show runn interface cem** *interface-path-id*

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **`configure`**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **`hw-module subslot`** *`node-id`* **`cardtype {t3|e3}`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# hw-module subslot 0/3/0 t3` | The **hw-module subslot** *node-id* **cardtype** *type* command configures the SPA to function in t3/e3 mode.<br><br>Whenever there is a change in framing mode (t3/e3), the SPA will be reloaded automatically. Reload will happen only when all the CEM Interface, T3 Controller configurations are removed completely. This is not applicable when you configure the first time because T3 controller and interface configurations would not exist. |
| **Step 3** | **`hw-module subslot`** *`node-id`* **`mode CEM`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# hw-module subslot 0/3/0 mode CEM` | The **hw-module subslot** *node-id* **mode** *CEM* command configures the SPA to function in CEM mode. |
| **Step 4** | **`commit`** | Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |
| **Step 5** | **`controller t3`** *`interface-path-id`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller t3 0/0/1/0/4` | Enters T3 controller configuration submode and specifies the T3 controller name and *interface-path-id* with the *rack/slot/module/port/T3Num* notation. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 6** | **mode {t1\|e1}** | Sets the mode of interface. The possible modes are T1 and E1 channelization mode. |
| **Step 7** | **controller t1** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller t1 0/0/1/0/4/1` | Enters T1 controller configuration submode and specifies the T1 controller name and *interface-path-id* with the *rack/slot/module/port/T3Num/T1num* notation. |
| **Step 8** | **cem-group unframed**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1)# cem-group unframed` | Creates an structure agnostic CEM interface. |
| **Step 9** | **controller t1** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller t1 0/0/1/0/4/1` | Enters T1 controller configuration submode and specifies the T1 controller name and *interface-path-id* with the *rack/slot/module/port/T3Num/T1num* notation. |
| **Step 10** | **cem-group framed** *group-number* **timeslots** *range1-range2*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1)# cem-group framed 0 timeslots 1` | Creates an structure aware CEM interface. The **timeslots** keyword specifies the time slots for the interface by range with the *range1-range2* notation. |
| **Step 11** | **no shutdown**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# no shutdown` | Removes the shutdown configuration.<br><br>**Note** Removal of the shutdown configuration eliminates the forced administrative down on the interface, enabling it to move to an up or down state. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 12** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1)# end`<br>or<br>`RP/0/0/CPU0:router(config-t1)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 13** | **show runn interface cem** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router# show runn interface cem 0/0/2/0/1/1/1/1:1` | Verifies the CEM interface configuration. |

# Configuring CEM Interface

This section provides information about how to configure CEM. CEM provides a bridge between a time-division multiplexing (TDM) network and a packet network using Multiprotocol Label Switching (MPLS). The router encapsulates the TDM data in the MPLS packets and sends the data over a CEM pseudowire to the remote provider edge (PE) router.

The following sections describe how to configure CEM:

- Configuration Guidelines and Restrictions
- Configuring a Global CEM Class
- Attaching a CEM Class
- Configuring Payload Size
- Setting the Dejitter Buffer Size
- Setting an Idle Pattern
- Enabling Dummy Mode
- Setting a Dummy Pattern

## Configuration Guidelines and Restrictions

Not all combinations of payload size and dejitter buffer size are supported. If you apply an incompatible payload size or dejitter buffer configuration, the router rejects it and reverts to the previous configuration.

## Configuring a Global CEM Class

This task explains how to configure a global CEM class.

**Note** Any interface configuration would have higher precedence over configuration applied through attaching a CEM class. Also, CEM class attached to an interface would have higher precedence than CEM class attached to the parent controller. For example, if the dummy pattern value of *0xcf* is applied directly to an interface and then a CEM class which contains dummy pattern value of *0xaa* is attached to the same interface, then the dummy pattern value would be *0xcf*. The new configuration would not be applied until the dummy pattern value applied directly to the interface is removed.

### SUMMARY STEPS

1. **configure**
2. **cem class** *class-name*
3. **payload** *value*
4. **dejitter** *value*
5. **idle pattern** *value*
6. **dummy mode {last-frame | user-defined}**
7. **dummy pattern** *value*
8. **end**
   or
   **commit**

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| Step 1 | `configure`<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| Step 2 | `cem class` *class-name*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# cem class Default` | Creates a new CEM class. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **payload** *value*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cem-class)# payload 512` | Enter the payload size for the CEM class. |
| **Step 4** | **dejitter** *value*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cem-class)# dejitter 10` | Enter the dejitter buffer size for the CEM class. |
| **Step 5** | **idle pattern** *value*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cem-class)# idle pattern 0x55` | Enter the idle pattern value for the CEM class. |
| **Step 6** | **dummy mode**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cem-class)# dummy mode last-frame` | Enter the dummy mode for the CEM class. The options are last-frame or user-defined. |
| **Step 7** | **dummy pattern** *value*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cem-class)# dummy pattern` | Enter the dummy pattern value for the CEM class. This value is applied only when the dummy mode is user-defined. |
| **Step 8** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cem-class)# end`<br>or<br>`RP/0/0/CPU0:router(config-cem-class)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

## Attaching a CEM Class

This task explains how to attach a global CEM class.

**Note** You can attach a CEM class either to a CEM interface or to a T1/E1 controller.

### SUMMARY STEPS

1. **configure**
2. **interface cem** *interface-path-id* **(or) controller {t1|e1}** *rack/slot/subslot/port*
3. **cem class-attach** *class-name*
4. **end**
   or
   **commit**

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **interface cem** *interface-path-id*<br>(or)<br>**controller {t1\|e1}** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller t1 0/0/2/0/1/1` | Specifies the CEM interface or the T1/E1 controller. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | `cem class-attach` *class-name*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# cem class-attach Default` | Attaches the CEM class to an interface or controller. |
| **Step 4** | `end`<br>or<br>`commit`<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-cem-class)# end`<br>or<br>`RP/0/0/CPU0:router(config-cem-class)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

## Configuring Payload Size

To specify the number of bytes encapsulated into a single IP packet, use the **cem payload** command. The size argument specifies the number of bytes in the payload of each packet. The range is from 32 to 1312 bytes.

Default payload sizes for an unstructured CEM channel are as follows:

- E1 = 256 bytes
- T1 = 192 bytes
- E3 = 1024 bytes
- T3 = 1024 bytes

Default payload sizes for a structured CEM channel depend on the number of time slots that constitute the channel. Payload (L in bytes), number of time slots (N), and packetization delay (D in milliseconds) have the following relationship: L = 8*N*D.

The default payload size is calculated using the packetization latency depending on the number of time slots the cem interface represents. The relationship between the number of time slots and the packetization latency is provided below:

- For N = 1, D is 8 milliseconds (with the corresponding packet payloadsize of 64 bytes)
- For 2 <=N <= 4, D is 4 milliseconds (with the corresponding packetpayload size of 32*N bytes)
- For N >= 5, D is 1 millisecond (with the corresponding packet payloadsize of 8*N octets).

Support of 5 ms packetization latency for N = 1 is recommended.

## Setting the Dejitter Buffer Size

To specify the size of the dejitter buffer used to compensate for the network filter, use the **cem dejitter** command. The configured dejitter buffer size is converted from milliseconds to packets and rounded up to the next integral number of packets. Use the size argument to specify the size of the buffer, in milliseconds. The range is from 1 to 500 ms. The following is an example:

```
Router(config-cem)# cem dejitter 5
```

The default dejitter buffer for a CEM channel, irrespective of CESoPSN or SAToP, is as follows:

- E1 = 16 milliseconds
- T1 = 16 milliseconds
- E3 = 5 milliseconds
- T3 = 5 milliseconds

**Note** Refer Table 10, Table 11, and Table 12 for the relationship between payload and dejitter buffer on SAToP T1/E1, T3/E3, and CESoPSN lines. Configuration of payload and dejitter should be in accordance with the minimum and maximum values as mentioned in the table.

**Note** The maximum and minimum dejitter buffer value, that is the range is fixed for a given payload value.

## Setting an Idle Pattern

To specify an idle pattern, use the [no] **cem idle pattern pattern** command. The payload of each lost CESoPSN data packet must be replaced with the equivalent amount of the replacement data. The range for pattern is from 0x0 to 0xff; the default idle pattern is 0xff. This is an example:

```
Router(config-cem)# cem idle pattern 0xff
```

If the expected CEM packets are not received for a given CEM interface and are considered as being lost, then the CEoP SPA will play out the idle pattern towards the TDM attachment circuit in the respective timeslots configured in the CEM group.

## Enabling Dummy Mode

Dummy mode enables a bit pattern for filling in for lost or corrupted frames. To enable dummy mode, use the **cem dummy mode** [**last-frame** | **user-defined**] command. The default is **last-frame**. This is an example:

```
Router(config-cem)# cem dummy mode last-frame
```

When packets are lost due to misordering or where reordering of packets is not successful, the CEoP SPA will play out the Dummy pattern towards the TDM attachment circuit in respective timeslots configured in the CEM group.

## Setting a Dummy Pattern

If dummy mode is set to user-defined, you can use the **cem dummy-pattern** command to configure the dummy pattern. The range for pattern is from 0x0 to 0xff. The default dummy pattern is 0xff. This is an example:

```
Router(config-cem)# cem dummy-pattern 0xff
```

The Table 10 shows the relationship between payload and dejitter for T1/E1 SATOP lines.

*Table 10 T1/E1 SAToP lines: Payload and Jitter Limits*

| T1/E1 | Maximum Payload | Maximum Jitter | Minimum Jitter | Minimum Payload | Maximum Jitter | Minimun Jitter |
|---|---|---|---|---|---|---|
| T1 | 960 | 320 | 10 | 192 | 64 | 2 |
| E1 | 1280 | 320 | 10 | 256 | 64 | 2 |

The Table 11 shows the relationship between payload and dejitter for T3/E3 SAToP lines.

*Table 11 T3/E3 SAToP lines: Payload and Jitter Limits*

| T3/E3 | Maximum Payload | Maximum Jitter | Minimum Jitter | Minimum Payload | Maximum Jitter | Minimun Jitter |
|---|---|---|---|---|---|---|
| T3 | 1312 | 8 | 2 | 672 | 8 | 2 |
| E3 | 1312 | 16 | 2 | 512 | 8 | 2 |

The Table 12 shows the relationship between payload and dejitter for DS0 lines.

*Table 12 CESoPSN DS0 Lines: Payload and Jitter Limits*

| DS0 | Maximum Payload | Maximum Jitter | Minimum Jitter | Minimum Payload | Maximum Jitter | Minimun Jitter |
|---|---|---|---|---|---|---|
| 1 | 40 | 320 | 10 | 32 | 256 | 8 |
| 2 | 80 | 320 | 10 | 32 | 128 | 4 |
| 3 | 120 | 320 | 10 | 33 | 128 | 4 |
| 4 | 160 | 320 | 10 | 32 | 64 | 2 |
| 5 | 200 | 320 | 10 | 40 | 64 | 2 |
| 6 | 240 | 320 | 10 | 48 | 64 | 2 |
| 7 | 280 | 320 | 10 | 56 | 64 | 2 |
| 8 | 320 | 320 | 10 | 64 | 64 | 2 |
| 9 | 360 | 320 | 10 | 72 | 64 | 2 |
| 10 | 400 | 320 | 10 | 80 | 64 | 2 |
| 11 | 440 | 320 | 10 | 88 | 64 | 2 |
| 12 | 480 | 320 | 10 | 96 | 64 | 2 |
| 13 | 520 | 320 | 10 | 104 | 64 | 2 |
| 14 | 560 | 320 | 10 | 112 | 64 | 2 |
| 15 | 600 | 320 | 10 | 120 | 64 | 2 |
| 16 | 640 | 320 | 10 | 128 | 64 | 2 |
| 17 | 680 | 320 | 10 | 136 | 64 | 2 |
| 18 | 720 | 320 | 10 | 144 | 64 | 2 |
| 19 | 760 | 320 | 10 | 152 | 64 | 2 |
| 20 | 800 | 320 | 10 | 160 | 64 | 2 |
| 21 | 840 | 320 | 10 | 168 | 64 | 2 |
| 22 | 880 | 320 | 10 | 176 | 64 | 2 |
| 23 | 920 | 320 | 10 | 184 | 64 | 2 |
| 24 | 960 | 320 | 10 | 192 | 64 | 2 |
| 25 | 1000 | 320 | 10 | 200 | 64 | 2 |
| 26 | 1040 | 320 | 10 | 208 | 64 | 2 |
| 27 | 1080 | 320 | 10 | 216 | 64 | 2 |
| 28 | 1120 | 320 | 10 | 224 | 64 | 2 |
| 29 | 1160 | 320 | 10 | 232 | 64 | 2 |
| 30 | 1200 | 320 | 10 | 240 | 64 | 2 |
| 31 | 1240 | 320 | 10 | 248 | 64 | 2 |
| 32 | 1280 | 320 | 10 | 256 | 64 | 2 |

# Configuring Clocking

Each SPA port shall be configured either to use system clock from the host card or loop timed independently. Each SPA also supplies a reference clock to the host which can be selected among the received port clocks. This section provides information about how to configure clocking on the 1xOC3, 24xT1/E1 and 2xT3/E3 SPA.

This section describes the following topics:

- Configuring Clock Recovery
- Verifying Clock recovery

## Configuring Clock Recovery

When configuring clock recovery, consider the following guidelines:

### Adaptive Clock Recovery

- Clock source:
  - In Cisco IOS XR Release 4.2.0 and later, recovered clock from a CEM interface on the 1-Port Channelized OC-3/STM1 CEoP SPA, 24xT1/E1 CEoP SPA and 2xT3/E3 CEoP SPA can be used as a clock source on the SPA itself.
- Number of clock sources allowed:
  - Refer the section Clock Distribution, page 317 for more information.
- The minimum bundle size of CEM pseudowires on the network that delivers robust clock recovery is 4 DS0s.
- The minimum packet size of CEM pseudowires on the network that delivers robust clock recovery is 64 bytes.

To configure clock recovery on the CEoP SPA and to apply the recovered clock to the controller, use the following procedure:

### SUMMARY STEPS

1. **configure**
2. **interface cem** *rack/slot/subslot/port:cem-group*
3. **recover-clock** *clock-id* **{adaptive}**
4. **controller {t1|e1|t3|e3}** *rack/slot/subslot/port*
5. **clock source recovered** *clock-id*

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **`configure`**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **`interface cem`** `rack/slot/subslot/port:cem-group`<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface cem 0/1/0/0:2` | Specifies the complete CEM interface instance. |
| **Step 3** | **`recover-clock`** *`clock-id`* **`{adaptive}`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# recover-clock clock-id <> adaptive` | Specifies the recovered clock number and the clock recovery type. This is typically configured at the node acting as Slave that recovers the clock from incoming CEM packets from core. |
| **Step 4** | **`controller`** *`name instance`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller t1 0/1/1/0/0/0` | Enters controller configuration submode and specifies the controller name and instance identifier with the *rack/slot/module/port/name/instance1/instance2* notation. |
| **Step 5** | **`clock source recovered`** *`clock-id`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1)# clock source recovered 3` | Specifies the recovered clock number. This applies the recovered clock from a CEM interface on a T1/E1 or T3/E3 Controller. |

## Verifying Clock recovery

To verify clock recovery, use the **show recovered-clock** command.

```
Router# show recovered-clock sublsot 0/3/0
Recovered clock status for subslot 0/3/0
----------------------------------------
Clock    Mode         Port CEM  Status     Frequency Offset(ppb)
1        ADAPTIVE     0    1    HOLDOVER   0
Router# show recovered-clock
Recovered clock status for subslot 3/0
----------------------------------------
Clock    Mode         Port CEM  Status     Frequency Offset(ppb)
1        ADAPTIVE     0    1    ACQUIRING  -694
```

# Show Commands for CEM

You can use the command **show controller cem** <*forward interface instance*> to verify the CEM parameter information. The following example provides a sample output for the command.

### Ouput of show controller cem forward interface instance command

```
RP/0/0/CPU0:Router# show controllers cem 0/4/1/0:0

Interface                 : CEM0/4/1/0:0
Admin state               : Up
Driver link state         : Up
Port bandwidth(kbps)      : 1984
Dejitter buffer           : 16
Payload size              : 248
Dummy mode                : last-frame
Dummy pattern             : 0xff
Idle pattern              : 0xff
Signalling                : No CAS
RTP                       : Enabled
Ingress packets           : 1638960097, Ingress packets drop     : 0
Egress packets            : 1207954294, Egress packets drop      : 287140468
Missing packets           : 160475876, Reordered packets        : 50092
Malformed packets         : 73, Misorder drops           : 7
Jitter buffer underrun    : 28, Error seconds            : 79673
Severely error seconds    : 25721, Unavailable seconds      : 160361
Failure counts            : 2
```

# Configuration Examples for CEM

This section contains the following examples:


- Circuit Emulation Interface Configuration: Examples, page 348
  - Channelized Sonet / SDH Configurations and CEM Interface Creation, page 348
  - SAToP CEM interface creation on T3 / E3 on Cisco 2-Port Channelized T3/E3 Circuit Emulation and Channelized ATM SPA, page 350
  - SAToP CEM interface creation on T1 / E1 on Cisco 2-Port Channelized T3/E3 Circuit Emulation and Channelized ATM SPA, page 350
  - CESoPSN CEM interface creation on T1/E1 on Cisco 2-Port Channelized T3/E3 Circuit Emulation and Channelized ATM SPA, page 350
  - SAToP CEM interface creation on T1 / E1 on Cisco 24-Port Channelized T1/E1 Circuit Emulation and Channelized ATM SPA, page 351
  - CESoPSN CEM interface creation on T1 / E1 on Cisco 24-Port Channelized T1/E1 Circuit Emulation and Channelized ATM SPA, page 351
- Clock Recovery : Example, page 351
  - Adaptive Clock Recovery Configuration:, page 351


## Circuit Emulation Interface Configuration: Examples

The following example shows a sample CEM interface configuration on the Cisco 1-port Channelized OC3/STM-1 SPA.

### Channelized Sonet / SDH Configurations and CEM Interface Creation

#### Sonet - T1 Channelization and CEM Interface Creation

```
hw-module subslot  <loc> cardtype sonet
controller SONET 0/0/1/0
 sts 1
  mode vt15-t1
sts 2
  mode vt15-t1
sts 3
  mode vt15-t1
commit
```

**In case of structure agnostic cem interface**:

```
controller T1 0/0/1/0/1/4/1
 cem-group unframed
```

**In case of structure aware cem interface**:

```
controller T1 0/0/1/0/1/5/1
 cem-group framed 0 timeslots 1
 cem-group framed 1 timeslots 2-3
 cem-group framed 2 timeslots 4-6
 cem-group framed 3 timeslots 7-10
 cem-group framed 4 timeslots 11-15
 cem-group framed 5 timeslots 16-21
```

```
 cem-group framed 6 timeslots 22-24
```

### SDH - T1 Channelization and CEM Interface Creation

```
hw-module subslot  <loc> cardtype sdh
controller SONET0/0/2/0
 au 1
  mode c11-t1
 au 2
  mode c11-t1
 au 3
  mode c11-t1
commit
```

**In case of structure agnostic cem interface**:

```
controller T1 0/0/2/0/1/1/4
 cem-group unframed
```

**In case of structure aware cem interface**:

```
controller T1 0/0/2/0/1/7/1
 cem-group framed 0 timeslots 1
 cem-group framed 1 timeslots 2-3
 cem-group framed 2 timeslots 4-6
 cem-group framed 3 timeslots 7-10
 cem-group framed 4 timeslots 11-15
 cem-group framed 5 timeslots 16-21
 cem-group framed 6 timeslots 22-24
```

### SDH - E1 Channelization and CEM Interface Creation

```
hw-module subslot  <loc> cardtype sdh
controller SONET 0/0/2/0
 au 1
  mode tug3
  width 3
  tug3 1
   mode c12-e1
tug3 2
   mode c12-e1
 tug3 3
   mode c12-e1
commit
```

**In case of structure agnostic cem interface**:

```
controller E1 0/0/2/0/1/1/1/1
 cem-group unframed
```

**In case of structure aware cem interface**:

```
controller E1 0/0/2/0/1/1/7/1
 cem-group framed 0 timeslots 1
 cem-group framed 1 timeslots 2-3
 cem-group framed 2 timeslots 4-6
 cem-group framed 3 timeslots 7-10
 cem-group framed 4 timeslots 11-15
 cem-group framed 5 timeslots 16-21
 cem-group framed 6 timeslots 22-31
```

### CEM Interface Configuration

```
RP/0/RSP0/CPU0:CEOP-01#show runn interface cem 0/0/2/0/1/1/1/1:1

interface CEM0/0/2/0/1/1/1/1:1
 l2transport
 !
CEM Interface Config Options :

RP/0/RSP0/CPU0:CEOP-01(config)#interface cem 0/0/2/0/1/1/1/1:1
RP/0/RSP0/CPU0:CEOP-01(config-if)#cem ?
  class-attach  Attach a CEM class to this interface
  clock         Configure clocks on this CEM interface
  dejitter      Configure dejitter buffer
  dummy         Configure dummy frame parameters
  idle          Configure idle frame parameters
  payload       Configure payload size of CEM frames
```

## SAToP CEM interface creation on T3 / E3 on Cisco 2-Port Channelized T3/E3 Circuit Emulation and Channelized ATM SPA

```
RP/0/0/CPU0:router(config)#controller t3 0/4/2/0
RP/0/0/CPU0:router(config-t3)#cem-group ?
  unframed  clear channel carrying CEM
RP/0/0/CPU0:router(config-t3)#cem-group unframed
RP/0/0/CPU0:router(config-t3)#commit
RP/0/0/CPU0:router(config-t3)#
```

## SAToP CEM interface creation on T1 / E1 on Cisco 2-Port Channelized T3/E3 Circuit Emulation and Channelized ATM SPA

```
RP/0/0/CPU0:router(config)#controller t3 0/4/2/0
RP/0/0/CPU0:router(config-t3)#mode ?
  atm     clear channel carrying atm
  e1      channelize into 21 E1s
  serial  clear channel carrying hdlc like payload
  t1      channelized into 28 T1s
RP/0/0/CPU0:router(config-t3)#mode e1
RP/0/0/CPU0:router(config-t3)#commit

RP/0/0/CPU0:router(config)#controller e1 0/4/2/0/1
RP/0/0/CPU0:router(config-e1)#cem-group ?
  framed    Configure a framed CEM interface on T1/E1
  unframed  Configure a unframed CEM interface on T1/E1
RP/0/0/CPU0:router(config-e1)#cem-group unframed ?
  <cr>
RP/0/0/CPU0:router(config-e1)#cem-group unframed
RP/0/0/CPU0:router(config-e1)#commit
```

## CESoPSN CEM interface creation on T1/E1 on Cisco 2-Port Channelized T3/E3 Circuit Emulation and Channelized ATM SPA

```
RP/0/0/CPU0:router(config)#controller t3 0/4/2/1
RP/0/0/CPU0:router(config-t3)#mode ?
  atm     clear channel carrying atm
  e1      channelize into 21 E1s
```

```
  serial  clear channel carrying hdlc like payload
  t1      channelized into 28 T1s
RP/0/0/CPU0:router(config-t3)#mode t1
RP/0/0/CPU0:router(config-t3)#commit

RP/0/0/CPU0:router(config)#controller t1 0/4/2/1/1
RP/0/0/CPU0:router(config-t1)#cem-group ?
  framed    Configure a framed CEM interface on T1/E1
  unframed  Configure a unframed CEM interface on T1/E1
RP/0/0/CPU0:router(config-t1)#cem-group framed ?
  <0-23>  CEM group number
RP/0/0/CPU0:router(config-t1)#cem-group framed 0 ?
  timeslots  List of timeslots in the CEM group
RP/0/0/CPU0:router(config-t1)#cem-group framed 0 timeslots ?
  WORD  timeslot string seprated by (:) or (-) from 1 to 24. (:) indicates individual
timeslot and (-) represent range
RP/0/0/CPU0:router(config-t1)#cem-group framed 0 timeslots 1:23
RP/0/0/CPU0:router(config-t1)#commit
```

### SAToP CEM interface creation on T1 / E1 on Cisco 24-Port Channelized T1/E1 Circuit Emulation and Channelized ATM SPA

```
RP/0/0/CPU0:router(config)#controller e1 0/4/1/2
RP/0/0/CPU0:router(config-e1)#cem-group ?
  framed    Configure a framed CEM interface on T1/E1
  unframed  Configure a unframed CEM interface on T1/E1
RP/0/0/CPU0:router(config-e1)#cem-group unframed ?
  <cr>
RP/0/0/CPU0:router(config-e1)#cem-group unframed
RP/0/0/CPU0:router(config-e1)#commit
```

### CESoPSN CEM interface creation on T1 / E1 on Cisco 24-Port Channelized T1/E1 Circuit Emulation and Channelized ATM SPA

```
RP/0/0/CPU0:router(config)#controller e1 0/4/1/1
RP/0/0/CPU0:router(config-e1)#cem-group framed ?
  <0-30>  CEM group number
RP/0/0/CPU0:router(config-e1)#cem-group framed 1 ?
  timeslots  List of timeslots in the CEM group
RP/0/0/CPU0:router(config-e1)#cem-group framed 1 timeslots ?
  WORD  timeslot string seprated by (:) or (-) from 1 to 31. (:) indicates individual
timeslot and (-) represent range
RP/0/0/CPU0:router(config-e1)#cem-group framed 1 timeslots 1:20
RP/0/0/CPU0:router(config-e1)#commit
```

# Clock Recovery : Example

## Adaptive Clock Recovery Configuration:

(E1 configurations are similar to T1s given below)

```
CE1
----
Router (config)#controller t1 0/0/2/0/1/1/4
Router (config-t1)#clock source internal
```

```
PE1 (Acts as source of clock, but no specific configuration under CEM Interface is needed
here)
-----------------------------------------------------------------------------------------
Router (config)#controller t1 0/0/2/0/1/1/4
Router (config-t1)#clock source line

PE2 (On PE node where clock recovery is done):
----------------------------------------
```

To recover the adaptive clock:

```
Router(config)# interface cem 0/0/2/0/1/1/4:0
Router(config-if)#cem clock recover <clock-id> adaptive
```

To apply the recovered clock,

```
Router (config)#controller t1 0/0/2/0/1/1/4
Router (config-t1)#clock source recovered <clock-id>

CE2
----
Router (config)#controller t1 0/0/2/0/1/1/4
Router (config-t1)#clock source line
```

# Additional References

These sections provide references to related documents.

## Related Documents

| Related Topic | Document Title |
|---|---|
| Cisco IOS XR master command reference | *Cisco IOS XR Master Commands List* |
| Cisco IOS XR interface configuration commands | *Cisco IOS XR Interface and Hardware Component Command Reference for the Cisco XR 12000 Series Router* |
| Initial system bootup and configuration information for a router using the Cisco IOS XR software | *Cisco IOS XR Getting Started Guide for the Cisco XR 12000 Series Router* |
| Information about user groups and task IDs | *Configuring AAA Services on Cisco IOS XR Software* module of *Cisco IOS XR System Security Configuration Guide* |

## 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:<br>http://www.cisco.com/go/mibs |

## RFCs

| RFCs | Title |
|---|---|
| RFC 5086, RFC 4553. RFC 4197, RFC 5287 | • *Structure-Aware Time Division Multiplexed (TDM) Circuit Emulation Service over Packet Switched Network (CESoPSN)*<br>• *Structure-Agnostic Time Division Multiplexing (TDM) over Packet (SAToP)*<br>• *Requirements for Edge-to-Edge Emulation of Time Division Multiplexed (TDM) Circuits over Packet Switching Networks*<br>• *Control Protocol Extensions for the Setup of Time-Division Multiplexing (TDM) Pseudowires in MPLS Networks* |

## 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/support |

# Configuring Clear Channel SONET Controllers on Cisco IOS XR Software

This module describes the configuration of clear channel SONET controllers on the Cisco XR 12000 Series Router.

SONET controller configuration is a preliminary step toward Packet-over-SONET/SDH (POS) configuration on routers using Cisco IOS XR software.

SONET allows you to define optical signals and a synchronous frame structure for multiplexed digital traffic. It is a set of standards defining the rates and formats for optical networks specified in American National Standards Institute (ANSI) T1.105, ANSI T1.106, and ANSI T1.117.

For more information about configuring a channelized SONET controller, see the "Configuring Channelized SONET/SDH on Cisco IOS XR Software" module.

The commands for configuring the Layer 1 SONET controllers are provided in the *Cisco IOS XR Interface and Hardware Component Command Reference*.

**Feature History for Configuring SONET Controllers on Cisco IOS XR Software**

| Release | Modification |
|---|---|
| Release 3.2 | Support was added for the Cisco XR 12000 Series Router. |
| Release 3.3.0 | Support was added on the Cisco XR 12000 Series Router for the following hardware:<br>• Cisco XR 12000 SIP-401<br>• Cisco XR 12000 SIP-501<br>• Cisco XR 12000 SIP-601<br>Support was added on the Cisco XR 12000 Series Router for the Cisco 2-Port OC-48 POS/RPR SPA. |

| | |
|---|---|
| Release 3.5.0 | Support was added on the Cisco XR 12000 Series Router for the following SPAs:<br>• Cisco 1-Port Channelized OC-3/STM-1 SPA<br>• Cisco 1-Port Channelized OC-12/DS0 SPA<br>• Cisco 2-Port OC-12 POS<br>• Cisco 4-Port OC-12 POS<br>• Cisco 8-Port OC-12 POS<br>• Cisco 4-Port OC-3 POS<br>• Cisco 8-Port OC-3 POS |
| Release 3.8.0 | The **delay trigger line** command was updated to **line delay trigger** in the following sections:<br>• How to Configure Clear Channel SONET Controllers<br>• Configuring a Hold-off Timer to Prevent Fast Reroute from being Triggered |

# Contents


- Prerequisites for Configuring Clear Channel SONET Controllers, page 356
- Information About Configuring SONET Controllers, page 357
- How to Configure Clear Channel SONET Controllers, page 360
- Configuration Examples for SONET Controllers, page 371
- Additional References, page 372


# Prerequisites for Configuring Clear Channel SONET Controllers

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 SONET controllers, be sure that the following tasks and conditions are met:

- You have at least one of the following physical layer interface module (PLIM) cards installed in your chassis:
  - Cisco 4-Port OC-3c/STM-1 POS SPA
  - Cisco 8-Port OC-12c/STM-4 POS SPA
  - Cisco 2-Port OC-48/STM-16 POS SPA
  - Cisco 4-Port OC-48/STM-16 POS SPA
  - Cisco 16-Port OC-48c/STM-16c POS
  - Cisco 4-Port OC-192c/STM-64c POS
  - Cisco 1-Port OC-192c/STM-64 POS/RPR XFP SPA
  - Cisco 1-Port OC-768c/STM-256c POS PLIM

- You know how to apply the specify the SONET controller name and instance identifier with the generalized notation *rack/slot/module/port*. The SONET controller name and instance identifier are required with the **controller sonet** command.

# Information About Configuring SONET Controllers

To configure SONET controllers, you must understand the following concepts:

- SONET Controller Overview, page 357
- Default Configuration Values for SONET Controllers, page 358
- SONET APS, page 359

## SONET Controller Overview

In routers supporting Cisco IOS XR software, the physical ports on certain line cards are called controllers. Before you can configure a POS, SRP, or serial interface, you need to configure the SONET controller.

The commands used to configure the physical SONET port are grouped under the SONET controller configuration mode. To get to the SONET controller configuration mode, enter the **controller sonet** command in global configuration mode. You can also preconfigure a SONET controller using the **controller preconfigure sonet** global configuration command.

The router uses SONET controllers for Layer 1 and Layer 2 processing.

# Default Configuration Values for SONET Controllers

Table 13 describes some default configuration parameters that are present on SONET controllers.

*Table 13 SONET Controller Default Configuration Values*

| Parameter | Default Value | Configuration File Entry |
|---|---|---|
| Reporting of the following alarms for a SONET controller:<br>• Bit 1 (B1) bit error rate (BER) threshold crossing alert (TCA) errors<br>• Bit 2 (B2) BER TCA errors<br>• Signal failure BER errors<br>• Section loss of frame (SLOF) errors<br>• Section loss of signal (SLOS) errors | **enabled** | To disable reporting of any alarms enabled by default, use the **no report** [**b1-tca** \| **b2-tca** \| **sf-ber** \| **slof** \| **slos**] command in SONET/SDH configuration mode.<br>To enable reporting of line alarm indication signal (LAIS), line remote defect indication (LRDI), or signal degradation BER errors, use the **report** [**lais** \| **lrdi** \| **sd-ber**] command in SONET/SDH configuration mode. |
| Reporting of the following alarms for a SONET path controller:<br>• Bit 3 (B3) BER TCA errors<br>• Path loss of pointer (PLOP) errors | **enabled** | To disable B3 BER TCA or PLOP reporting on the SONET path controller, enter the **no report b3-tca** or **no report plop** command in SONET/SDH path configuration submode.<br>To enable reporting of path alarm indication signal (PAIS), path payload mismatch (PPLM), path remote defect indication (PRDI), or path trace identity mismatch (PTIM) errors, use the **report** [ **pais** \| **pplm** \| **prdi** \| **ptim** command in SONET/SDH path configuration submode. |

*Table 13 SONET Controller Default Configuration Values (continued)*

| Parameter | Default Value | Configuration File Entry |
|---|---|---|
| Synchronous payload envelope (SPE) scrambling | **enabled** | To disable SPE scrambling on a SONET controller, enter the **path scrambling disable** command in SONET controller configuration submode. |
| Keepalive timer | **enabled** | To turn off the keepalive timer, enter the **keepalive disable** command in interface configuration mode. |

# SONET APS

The automatic protection switching (APS) feature allows switchover of interfaces in the event of failure, and is often required when connecting SONET equipment to telco equipment. APS refers to the mechanism of using a *protect* interface in the SONET network as the backup for *working* interface. When the working interface fails, the protect interface quickly assumes its traffic load. The working interfaces and their protect interfaces make up an *APS group*.

In Cisco IOS XR software, SONET APS configuration defines a working line and a protection line for each redundant line pair. The working line is the primary or preferred line, and communications take place over that line as long as the line remains operative. If a failure occurs on the working line, APS initiates a switchover to the protection line. For proper APS operation between two routers, a working line on one router must also be the working line on the other router, and the same applies to the protection line.

In a SONET APS group, each connection may be bidirectional or unidirectional, and revertive or non-revertive. The same signal payload is sent to the working and protect interfaces. The working and protect interfaces can terminate in two ports of the same card, or in different cards in the same router, or in two different routers.

The protect interface directs the working interface to activate or deactivate in the case of degradation, loss of channel signal, or manual intervention. If communication between the working and protect interfaces is lost, the working router assumes full control of the working interface as if no protect circuit existed.

In an APS group, each line is called a *channel*. In bidirectional mode, the receive and transmit channels are switched as a pair. In unidirectional mode, the transmit and receive channels are switched independently. For example, in bidirectional mode, if the receive channel on the working interface has a loss of channel signal, both the receive and transmit channels are switched.

# How to Configure Clear Channel SONET Controllers

This section contains the following procedures:


- Configuring a Clear Channel SONET Controller, page 360
- Configuring SONET APS, page 364
- Configuring a Hold-off Timer to Prevent Fast Reroute from Being Triggered, page 369


## Configuring a Clear Channel SONET Controller

This task explains how to configure SONET controllers as a prerequisite to configuring POS and SRP or serial interfaces.

### Prerequisites

- You need to have a supported POS line card or channelized SPA installed in a router that is running the corresponding supported Cisco IOS XR software release.
- If you want to ensure recovery from fiber or equipment failures, then configure SONET APS on the router as describe in the "Configuring SONET APS" section on page -364.

#### SUMMARY STEPS

1. **configure**
2. **controller sonet** *interface-path-id*
3. **clock source** {**internal** | **line**}
4. **line delay trigger** *value*
5. **line delay clear** *value*
6. **framing** {**sdh** | **sonet**}
7. **loopback** {**internal** | **line**}
8. **overhead** {**j0** | **s1s0**} *byte-value*
9. **path** *keyword* [*values*]
10. **end**
    or
    **commit**
11. **show controllers sonet** *interface-path-id*

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **`configure`**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **`controller sonet`** *`interface-path-id`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller sonet 0/1/0/0` | Enters SONET controller configuration submode and specifies the SONET controller name and instance identifier with the *rack/slot/module/port* notation. |
| **Step 3** | **`clock source`** {**`internal`** \| **`line`**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# clock source internal` | Configures the SONET port transmit clock source, where the **internal** keyword sets the internal clock and **line** keyword sets the clock recovered from the line.<br>• Use the **line** keyword whenever clocking is derived from the network. Use the **internal** keyword when two routers are connected back-to-back or over fiber for which no clocking is available.<br>• The **line** clock is the default.<br>**Note** Internal clocking is required for SRP interfaces. |
| **Step 4** | **`line delay trigger`** *`value`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# line delay trigger 3000` | (Optional) Configures the SONET line delay trigger values, where the trigger values are in the range from 0 through 60000 milliseconds, and the default delay trigger value is 0 milliseconds. |
| **Step 5** | **`line delay clear`** *`value`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# line delay clear 4000` | (Optional) Configures the amount of time before a SONET line delay trigger alarm is cleared. The range is from 1000 through 180000 milliseconds, and the default is 10 seconds. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 6** | **framing** {**sdh** \| **sonet**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# framing sonet` | (Optional) Configures the controller framing with either the **sdh** keyword for Synchronous Digital Hierarchy (SDH) framing or the **sonet** keyword for SONET framing.<br><br>SONET framing (**sonet**) is the default. |
| **Step 7** | **loopback** {**internal** \| **line**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# loopback internal` | (Optional) Configures the SONET controller for loopback, where the **internal** keyword selects internal (terminal) loopback, or the **line** keyword selects line (facility) loopback. |
| **Step 8** | **overhead** {**j0** \| **s1s0**} *byte-value*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# overhead s1s0` | (Optional) Configures the controller's overhead, where the **j0** keyword specifies the STS identifier (J0/C1) byte, and the **s1s0** keyword specifies bits s1 and s0 of H1 byte.<br><br>• The default byte value for the **j0** keyword is 0xcc, and the default byte value for the **s1s0** keyword is 0.<br><br>• The range of valid values for **j0** and **s1s0** is 0 through 255. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 9** | **`path`** *`keyword`* `[`*`values`*`]`<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# path delay trigger 25` | (Optional) Configures SONET controller path values.<br><br>Keyword definitions are as follows:<br><br>• **ais-shut**—Set sending path alarm indication signal (PAIS) when shut down.<br><br>• **b3-ber-prdi**—Enable sending of a path-level remote defect indication (PRDI) when the bit error rate (BER) bit interleaved parity (BIP) threshold is exceeded.<br><br>• **delay clear** *value*—Set the amount of time before a Synchronous Transport Signal (STS) path delay trigger alarm is cleared. Replace the *value* argument with a number in the range from 0 through 180000 milliseconds. The default value is 10 seconds.<br><br>• **delay trigger** *value* —Set SONET path delay values or delay trigger value. Replace the *value* argument with a number in the range from 0 through 60000 milliseconds. The default value is 0 milliseconds.<br><br>• **overhead** [**c2** *byte-value* \| **j1** *line*] —Set SONET POH byte or bit values. Enter the **c2** keyword to specify STS SPE content (C2) byte, and replace the *byte-value* argument with a number in the range from 0 through 255. Enter the **j1** keyword to configure the SONET path trace (J1) buffer, and replace the *line* argument with the path trace buffer identifier (in ASCII text).<br><br>• **report [b3-tca \| pais \| plop \| pplm \| prdi \| ptim**]—Set SONET path alarm reporting. Specifies which alarms are reported and which bit error rate (BER) thresholds will signal an alarm. By default, B3 BER threshold crossing alert (TCA) and path loss of pointer (PLOP) reporting are enabled. Specifying the **pais** keyword sets PAIS reporting status; **pplm** sets path payload mismatch (PPLM) defect reporting status; **prdi** sets path remote defect indication reporting status; and **ptim** sets path trace identity mismatch (PTIM) defect reporting status.<br><br>The **no report b3-tca** and **no report plop** commands in SONET/SDH path configuration submode disable B3 BER TCA and PLOP reporting status, respectively.<br><br>• **scrambling disable**—Disable SPE scrambling. Note that SPE scrambling is enabled by default.<br><br>• **threshold b3-tca** *BER*—Set SONET path BER threshold value. Replace the *BER* argument with a number in the range from 3 through 9. The threshold value is interpreted as a negative exponent of 10 when determining the bit error rate. For example, a value of 5 implies a bit error rate of 10 to the minus 5. The default BER threshold value is 6.<br><br>• **uneq-shut—**Sets sending Unequipped (UNEQ) when shut down. |

| | Command or Action | Purpose |
|---|---|---|
| Step 10 | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# end`<br>or<br>`RP/0/0/CPU0:router(config-sonet)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 11 | **show controllers sonet** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router# show controllers sonet 0/1/0/0` | Verifies the SONET controller configuration. |

# Configuring SONET APS

SONET APS offers recovery from fiber (external) or equipment (interface and internal) failures at the SONET line layer. This task explains how to configure basic automatic protection switching (APS) on the router and how to configure more than one protect or working interface on a router by using the **aps group** command.

To verify the configuration or to determine if a switchover has occurred, use the **show aps** command.

## Prerequisites

Before you configure SONET APS, be sure that you have a supported POS line card or channelized SPA installed in a router that is running Cisco IOS XR software.

## Restrictions

Before you configure SONET APS, consider the following restictions:

- For proper APS operation between two routers, a working line on one router must also be the working line on the other router, and the same applies to the protection line.

## SUMMARY STEPS

1. **configure**
2. **aps group** *number*
3. **channel** {**0** | **1**} **local sonet** *interface*
4. Repeat Step 3 for each channel in the APS group.
5. **exit**
6. **interface loopback** *number*
7. **ipv4 address** *ip-address mask*
8. **exit**
9. **interface pos** *interface-path-id*
   or
   **interface serial** *interface-path-id*
10. **ipv4 address** *ip-address mask*
11. **pos crc** {**16** | **32**}
    or
    **crc** {**16** | **32**}
12. **encapsulation** {**frame-relay** | **hdlc** | **ppp**} (Serial interfaces only)
13. **keepalive** {*interval* | **disable**}[*retry*]
14. **no shutdown**
15. Repeat Step 9 to Step 13 for each channel in the group.
16. **exit**
17. **controller sonet** *interface-path-id*
18. **ais-shut**
19. **path scrambling disable**
20. **clock source** {**internal** | **line**}
21. Repeat Step 16 to Step 19 for each channel of the group.
22. **end**
    or
    **commit**
23. **exit**
24. **exit**
25. **show aps**
26. **show aps group** [*number*]

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **aps group** *number*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# aps group 1` | Adds an APS group with a specified number and enters APS group configuration mode.<br>• Use the **aps group** command in global configuration mode.<br>• To remove a group, use the **no** form of this command, as in: **no aps group** *number,* where the value range is from 1–255.<br>**Note** To use the **aps group** command, you must be a member of a user group associated with the proper task IDs for aps commands.<br>**Note** The **aps group** command is used even when a single protect group is configured. |
| **Step 3** | **channel** {**0** \| **1**} **local sonet** *interface*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-aps)# channel 0 local SONET 0/0/0/1` | Creates a channel for the APS group. **0** designates a protect channel, and **1** designates a working channel.<br>**Note** If the protect channel is local, it must be assigned using the **channel** command *before* any of the working channels is assigned. |
| **Step 4** | Repeat Step 3 for each channel in the group. | — |
| **Step 5** | **exit** | Exits APS group configuration mode and enters global configuration mode. |
| **Step 6** | **interface loopback** *number*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface loopback 1` | (Optional) Configures a loopback interface if a two-router APS is desired and enters interface configuration mode for a loopback interface.<br>**Note** In this example, the loopback interface is used as the interconnect. |
| **Step 7** | **ipv4 address** *ip-address mask*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# ipv4 address 172.18.0.1 255.255.255.224` | Assigns an IPV4 address and subnet mask to the loopback interface. |
| **Step 8** | **exit** | Exits interface configuration mode for a loopback interface, and enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 9** | **interface pos** *interface-path-id*<br>or<br>**interface serial** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface POS 0/2/0/0`<br>or<br>`RP/0/0/CPU0:router(config)# interface serial 0/1/1/0/0/0:0` | Connects the interface for the channel selected in Step 3, and enters interface configuration mode.<br><br>For serial interfaces, specifies the complete interface number with the *rack/slot/module/port/T3Num/T1num:instance* notation. |
| **Step 10** | **ipv4 address** *ip-address mask*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# ipv4 address 172.18.0.1 255.255.255.224` | Assigns an IPv4 address and subnet mask to the interface. |
| **Step 11** | **pos crc** {**16** \| **32**}<br>or<br>**crc** {**16** \| **32**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# pos crc 32`<br>or<br>`RP/0/0/CPU0:router(config-if)# crc 32` | Selects a CRC value for the channel. Enter the **16** keyword to specify 16-bit CRC mode, or enter the **32** keyword to specify 32-bit CRC mode. For POS interfaces, the default CRC is 32. For serial interfaces, the default is 16. |
| **Step 12** | **encapsulation** {**frame-relay** \| **hdlc** \| **ppp**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# encapsulation ppp` | (Serial interfaces only) Set the Layer 2 encapsulation of an interface. |
| **Step 13** | **keepalive** {*interval* \| **disable**}[*retry*]<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# keepalive disable` | Sets the keepalive timer for the channel, where:<br>• *interval*—Number of seconds (from 1 to 30) between keepalive messages. The default is 10.<br>• **disable**—Turns off the keepalive timer.<br>• *retry*—(Optional) Number of keepalive messages (from 1 to 255) that can be sent to a peer without a response before transitioning the link to the down state. The default is 5 for interfaces with PPP encapsulation, and 3 for interfaces with HDLC encapsulation.<br><br>The **keepalive** command does not apply to interfaces using Frame Relay encapsulation. |
| **Step 14** | **no shutdown**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# no shutdown` | Removes the shutdown configuration.<br>• The removal of the shutdown configuration removes the forced administrative down on the interface, enabling that interface to move to an up or down state (assuming the parent SONET layer is not configured administratively down). |
| **Step 15** | Repeat Step 9 through Step 13 for each channel in the group. | — |

| | Command or Action | Purpose |
|---|---|---|
| **Step 16** | **exit** | Exits interface configuration mode, and enters global configuration mode. |
| **Step 17** | **controller sonet** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller sonet 0/1/0/0` | Enters SONET controller configuration mode and specifies the SONET controller name and instance identifier with the *rack/slot/module/port* notation. |
| **Step 18** | **ais-shut**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# ais-shut` | Configures SONET path values such as alarm indication signal (AIS) at shut down. |
| **Step 19** | **path scrambling disable**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# path scrambling disable` | (Optional) Disables synchronous payload envelope (SPE) scrambling.<br><br>**Note** SPE scrambling is enabled by default. |
| **Step 20** | **clock source** {**internal** \| **line**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# clock source internal` | Configures the SONET port TX clock source, where the **internal** keyword sets the internal clock and the **line** keyword sets the clock recovered from the line.<br>• Use the **line** keyword whenever clocking is derived from the network; use the **internal** keyword when two routers are connected back-to-back or over fiber for which no clocking is available.<br>• The line clock (**line**) is the default. |
| **Step 21** | Repeat Step 16 through Step 19 for each channel in the group. | — |
| **Step 22** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# end`<br>or<br>`RP/0/0/CPU0:router(config-sonet)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

| | Command or Action | Purpose |
|---|---|---|
| Step 23 | `exit` | Exits SONET controller configuration mode, and enters global configuration mode. |
| Step 24 | `exit` | Exits global configuration mode, and enters EXEC mode. |
| Step 25 | `show aps`<br><br>**Example:**<br>`RP/0/0/CPU0:router# show aps` | (Optional) Displays the operational status for all configured SONET APS groups. |
| Step 26 | **show aps group** [*number*]<br><br>**Example:**<br>`RP/0/0/CPU0:router# show aps group 3` | (Optional) Displays the operational status for configured SONET APS groups.<br><br>**Note** The **show aps group** command is more useful than the **show aps** command when multiple groups are defined. |

# Configuring a Hold-off Timer to Prevent Fast Reroute from Being Triggered

When APS is configured on a router, it does not offer protection for tunnels; because of this limitation, fast reroute (FRR) still remains the protection mechanism for Multiprotocol Label Switching (MPLS) traffic-engineering.

When APS is configured in a SONET core network, an alarm might be generated toward a router downstream. If the router downstream is configured with FRR, you may want to configure a hold-off timer at the SONET level to prevent FRR from being triggered while the CORE network is doing a restoration. Perform this task to configure the delay.

## Prerequisites

Configure SONET APS, as describe in the "Configuring SONET APS" section on page 364.

### SUMMARY STEPS

1. **configure**
2. **controller sonet** *interface-path-id*
3. **line delay trigger** *value*
   or
   **path delay trigger** *value*
4. **end**
   or
   **commit**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **controller sonet** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller sonet 0/6/0/0` | Enters SONET configuration mode. |
| **Step 3** | **line delay trigger** *value*<br>or<br>**path delay trigger** *value*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# line delay trigger 250`<br>`or`<br>`RP/0/0/CPU0:router(config-sonet)# path delay trigger 300` | Configures SONET port delay trigger values in milliseconds.<br><br>**Tip** The commands in Step 2 and Step 3 can be combined in one command string and entered from global configuration mode like this: **controller sonet** *r/s/m/p* **line delay trigger** or **controller sonet** *r/s/m/p* **path delay trigger**. |
| **Step 4** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-sonet)# end`<br>or<br>`RP/0/0/CPU0:router(config-sonet)# 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)?`<br>`[cancel]:`<br>– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

# Configuration Examples for SONET Controllers

This section contains the following examples:

- SONET Controller Configuration: Example, page 371
- SONET APS Group Configuration: Example, page 371

## SONET Controller Configuration: Example

The following example shows the commands and output generated when you are performing the configuration of a SONET controllers following the steps outlined in the "Configuring a Clear Channel SONET Controller" section on page 360. This example shows the usage of every optional command, along with listings of options within commands where relevant. An actual configuration may or may not include all these commands.

```
configure
controller sonet 0/1/0/0
    ais-shut
    clock source internal
    framing sonet
    loopback internal
Loopback is a traffic-effecting operation
    overhead s1s0 1
    path ais-shut
    path delay trigger 0
    path overhead j1 line 11
    path report pais
    path scrambling disable
    path threshold b3-tca 6
    path uneq-shut
    report pais
    threshold b2-tca 4
    commit
```

## SONET APS Group Configuration: Example

The following example shows SONET Local (one router) APS configuration.

```
aps group 1
    channel 0 local SONET 0/0/0/1
    channel 1 local SONET 0/0/0/2
    signalling sonet
    commit
show aps
show aps group 3
```

The following example shows SONET Remote (two routers) APS configuration.

```
RP/0/0/CPU0:router(config)# aps group 1
    channel 0 local SONET 0/0/0/1
    channel 1 remote 172.18.69.123
    signalling sonet
    commit
show aps
show aps group 3
```

# Additional References

These sections provide references related to SONET controller configuration.

## Related Documents

| Related Topic | Document Title |
|---|---|
| Cisco IOS XR master command reference | *Cisco IOS XR Master Commands List* |
| Cisco IOS XR interface configuration commands | *Cisco IOS XR Interface and Hardware Component Command Reference* |
| Initial system bootup and configuration information for a router using the Cisco IOS XR Software | *Cisco IOS XR Getting Started Guide* |
| Information about user groups and task IDs | *Configuring AAA Services on Cisco IOS XR Software* module of *Cisco IOS XR System Security Configuration Guide* |

## 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:<br>http://www.cisco.com/go/mibs |

## RFCs

| 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/support |

# Configuring Clear Channel T3/E3 and Channelized T3 and T1/E1 Controllers on Cisco IOS XR Software

This module describes the configuration of clear channel T3/E3 controllers and channelized T3 and T1/E1 controllers on the Cisco XR 12000 Series Router.

You must configure the T3/E3 controller before you can configure an associated serial interface.

**Feature History for Configuring T3/E3 Controller Interfaces**

| Release | Modification |
|---|---|
| Release 3.3.0 | This feature was introduced on the Cisco XR 12000 Series Router. |
| Release 3.5.0 | This feature was introduced on the Cisco XR 12000 Series Router for the Cisco 1-Port Channelized OC-12/DS0 SPA. |
| Release 3.6.0 | This feature was introduced on the Cisco XR 12000 Series Router for the 8-Port Channelized T1/E1 SPA. |

# Contents


- Prerequisites for Configuring T3/E3 Controllers, page 375
- Information About T3/E3 Controllers and Serial Interfaces, page 376
- How to Configure Clear Channel T3/E3 Controllers and Channelized T1/E1 Controllers, page 380
- Configuration Examples, page 406
- Additional References, page 409


# Prerequisites for Configuring T3/E3 Controllers

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 T3/E3 controllers, be sure that you have one of the following supported SPAs installed in the router:

- The Cisco XR 12000 Series Router supports the following SIPs:

- Cisco XR 12000 SIP-401
- Cisco XR 12000 SIP-501
- Cisco XR 12000 SIP-601

- The Cisco XR 12000 Series Router supports the following SPAs:
  - Cisco 2-Port and 4-Port Channelized T3 SPA
  - Cisco 2-Port and 4-Port Clear Channel T3/E3 SPA
  - Cisco 8-Port Channelized T1/E1 SPA

**Note** The 2-Port and 4-Port Channelized T3 SPAs can run in clear channel mode, or they can be channelized into 28 T1 or 21 E1 controllers.

# Information About T3/E3 Controllers and Serial Interfaces

The 2-Port and 4-Port Clear Channel T3/E3 SPAs support clear channel services over serial lines only. The 2-Port and 4-Port Channelized T3 SPAs supports clear channel services and channelized serial lines.

If a controller is not channelized, then it is a clear channel controller, and the full bandwidth of its associated serial line is dedicated to a single channel that carries serial services.

**Note** In this release, only T3-to-T1/E1 channelization is supported.

When a T3 controller is channelized, it is logically divided into smaller bandwidth T1 or E1 controllers, depending on which mode of channelization you select. The sum of the bandwidth of the serial interfaces on the T1 or E1 controllers cannot exceed the bandwidth of the T3 controller that contains those channelized T1 or E1 controllers.

When you channelize a T3 controller, each individual T1 or E1 controller is automatically further channelized into DS0 time slots. A single T1 controller carries 24 DS0 time slots, and a single E1 controller carries 31 DS0 time slots. Users can divide these DS0 time slots up into individual channel groups. Each channel group can support a single serial interface.

When a controller is channelized, and channel groups have been created, services are provisioned on the associated serial interfaces.

The channelization feature in this release allows the following types of channelization:

- A single T3 controller into 28 T1 controllers, for a total controller size of 44210 kbps.
- A single T3 controller into 21 E1 controllers, for a total controller size of 43008 kbps.
- A single T1 controller supports up to 1.536 MB.
- A single E1 controller supports up to 2.048 MB.

**Note** A single shared port adapter (SPA) can support up to 448 channel groups.

This section includes the following additional topics:

- Configuration Overview, page 378
- Default Configuration Values for T3 and E3 Controllers, page 379

- Default Configuration Values for T1 and E1 Controllers, page 379

## Loopback Support

### Cisco 1-Port Channelized OC-3/STM-1 SPA

This section describes the types of loopback supported on the 1-Port Channelized OC-3/STM-1 SPA:

- For SONET controller:
  - Local loopback
  - Network line loopback
- For T3:
  - Local loopback
  - Network loopback
  - Remote loopback line (Use FEAC in C-Bit mode for T3)
  - Remote loopback payload (Use FEAC in C-Bit mode for T3)
- For E3:
  - Local loopback
  - Network loopback
- For T1:
  - Local loopback
  - Network line loopback
  - Remote line FDL ANSI loopback (also known as Remote CSU loopback - ESF mode)
  - Remote line FDL Bellcore loopback (also known as Remote SmartJack loopback - ESF mode)
  - Remote line inband loopback (SF inband loopback)
  - Remote payload FDL ANSI loopback (ESF remote payload loopback)
- For E1:
  - Local loopback
  - Network line loopback

### Cisco 4-Port Channelized T3/DS0 SPA

This section describes the types of loopback supported on the 4-Port Channelized T3/DS0 SPA:

- For T3:
  - Local loopback
  - Network loopback
  - Remote loopback line
- For T1:
  - Local loopback

   - Network line loopback
   - Remote line FDL ANSI loopback (also known as Remote CSU loopback - ESF mode)
   - Remote line FDL Bellcore loopback (also known as Remote SmartJack loopback - ESF mode)
- For E1:
   - Local loopback
   - Network line loopback

### Cisco 8-Port Channelized T1/E1 SPA

This section describes the types of loopback supported on the 8-Port Channelized T1/E1 SPA:

- For T1:
   - Local loopback
   - Networkl line loopback
   - Remote line FDL ANSI loopback (also known as Remote CSU loopback - ESF mode)
   - Remote line FDL Bellcore loopback (also known as Remote SmartJack loopback - ESF mode)
- For E1:
   - Local loopback

### Cisco 2-Port and 4-Port Clear Channel T3/E3 SPA

This section describes the types of loopback supported on the 2-Port and 4-Port Clear Channel T3/E3 SPA:

- Local loopback
- Network payload loopback (Configure the local framer to send all data received from the remote side back to the remote side.)
- Network line loopback (Configure the local LIU to send all data received from the remote side back to the remote side.)
- Remote line loopback (Use FEAC to request the remote interface to loop back to SPA—T3 only)

## Configuration Overview

Configuring a channelized T3 controller and its associated serial interfaces is a 4-step process:

**Step 1** Configure the T3 controller, and set the mode for that controller to T1 or E1.

**Step 2** Configure the T1 or E1 controller.

**Step 3** Create channel groups and assign DS0 time slots to these channel groups as desired.

**Step 4** Configure the serial interfaces that are associated with the individual channel groups, as described in the *Configuring Serial Interfaces on Cisco IOS XR Sotware* module later in this document.

## Default Configuration Values for T3 and E3 Controllers

Table 14 describes the default configuration parameters that are present on the T3 and E3 controllers.

*Table 14 T3 and E3 Controller Default Configuration Values*

| Parameter | Default Value | Configuration File Entry |
|---|---|---|
| Frame type for the data line | For T3: C-bit framing<br>For E3: G.751 | **framing** {**auto-detect** \| **c-bit** \| **m23**} |
| Clocking for individual T3/E3 links | **internal** | **clock source** {**internal** \| **line**} |
| Cable length | 224 feet | **cablelength** *feet* |
| Maintenance data link (MDL) messages<br>(T3 only) | **disable** | **mdl transmit** {**idle-signal** \| **path** \| **test-signal**} {**disable** \| **enable**} |
| National reserved bits for an E3 port<br>(E3 only) | **enable**, and the bit pattern value is 1. | **national bits** {**disable** \| **enable**} |

**Note** When configuring clocking on a serial link, you must configure one end to be **internal**, and the other end to be **line**. If you configure **internal** clocking on both ends of a connection, framing slips occur. If you configure **line** clocking on both ends of a connection, the line does not come up.

## Default Configuration Values for T1 and E1 Controllers

Table 15 describes the default configuration parameters that are present on the T1 and E1 controllers.

*Table 15 T1 and E1 Controller Default Configuration Values*

| Parameter | Default Value | Configuration File Entry |
|---|---|---|
| Frame type for the data line | For T1: extended superframe (**esf**)For E1: framing with CRC-4 error monitoring capabilities (**crc4**). | For T1: **framing** {**sf** \| **esf**}For E1: **framing** {**crc4** \| **no-crc4** \| **unframed** |
| Detection and generation of T1 yellow alarms.<br>(T1 only) | Yellow alarms are detected and generated on the T1 channel. | **yellow** {**detection** \| **generation**} {**disable** \| **enable**} |
| Clocking for individual T1 and E1 links | **internal** | **clock source** {**internal** \| **line**} |

*Table 15 T1 and E1 Controller Default Configuration Values*

| Parameter | Default Value | Configuration File Entry |
|---|---|---|
| Cable length<br>(T1 only) | For **cablelength long** command: *db-gain-value*: gain26; *db-loss-value*: 0db.<br>For **cablelength short** command: 533 feet. | To set a cable length of longer than 655 feet: **cablelength long** *db-gain-value db-loss-value*<br>To set a cable length of 655 feet or shorter: **cablelength short** *length* |
| Transmission of ANSI T1.403 or AT&T TR54016 once-per-second performance reports through Facility Data Link (FDL) for a T1 channel<br>(T1 only) | **disable** | **fdl** {**ansi** \| **att**} {**enable** \| **disable**} |
| National reserved bits for an E1 port<br>(E1 only) | 0 (which corresponds to *0x1f* in hexadecimal format) | **national bits** *bits* |

**Note** When configuring clocking on a serial link, you must configure one end to be **internal**, and the other end to be **line**. If you configure **internal** clocking on both ends of a connection, framing slips occur. If you configure **line** clocking on both ends of a connection, the line does not come up.

# How to Configure Clear Channel T3/E3 Controllers and Channelized T1/E1 Controllers

The T3/E3 controllers are configured in the physical layer control element of the Cisco IOS XR software configuration space. This configuration is described in the following tasks:


- Setting the Card Type, page 381
- Configuring a Clear Channel E3 Controller, page 383
- Modifying the Default E3 Controller Configuration, page 385
- Configuring a Clear Channel T3 Controller, page 388
- Configuring a Channelized T3 Controller, page 389
- Modifying the Default T3 Controller Configuration, page 391
- Configuring a T1 Controller, page 394
- Configuring an E1 Controller, page 397
- Configuring BERT, page 400

## Setting the Card Type

By default, the 2-Port and 4-Port Clear Channel T3/E3 SPAs boot in T3 mode, and the 8-Port Channelized T1/E1 SPA boots in T1 mode. If you want to use the 2-Port or 4-Port Clear Channel T3/E3 SPA in E3 mode or the 8-Port Channelized T1/E1 SPA in E1 mode, you must change the default setting of the **hw-module subslot card type** command as described in this section.

**Note** The **hw-module subslot card type** command configures all ports on the SPA to be the same type.

**Caution** The SPA is automatically reset when the **hw-module subslot card type** command is committed.

**Note** The **hw-module subslot card type** command applies to the 2-Port and 4-Port Clear Channel T3/E3 SPAs and the 8-Port Channelized T1/E1 SPA only. The 2-Port and 4-Port Channelized T3 SPA runs in T3 mode only.

### Prerequisites

If you have previously configured the interfaces on the 2-Port or 4-Port Clear Channel T3/E3 SPA or 8-Port Channelized T1/E1 SPA and now you want to change the card type, you must delete any previously defined T3/E3 or T1/E1 controller and serial interface configurations. Use the **no controller** [**e1** | **e3** | **t1** | **t3**] and **no interface serial** commands to revert the controller and interface configurations to their defaults.

### Restrictions

This task is applicable to 2-Port and 4-Port Clear Channel T3/E3 SPAs and the 8-Port Channelized T1/E1 SPA only.

#### SUMMARY STEPS

1. **configure**
2. **hw-module subslot** *subslot-id* **cardtype** {**e1** | **e3** | **t1** | **t3**}
3. **end**
   or
   **commit**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **`configure`**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **`hw-module subslot`** *`subslot-id`* **`cardtype`** `{`**`e1`** `\|` **`e3`** `\|` **`t1`** `\|` **`t3`**`}`<br><br>**Example:s**<br>`RP/0/0/CPU0:router(config)# hw-module subslot 0/1/0 cardtype e3`<br>or<br>`RP/0/0/CPU0:router(config)# hw-module subslot 0/2/0 cardtype e1` | Sets the serial mode for the SPA.<br>• **t3**—Specifies T3 connectivity of 44,210 kbps through the network, using B3ZS coding. This is the default setting.<br>• **e3**—Specifies a wide-area digital transmission scheme used predominantly in Europe that carries data at a rate of 34,010 kbps.<br>• **t1**—Specifies 24 DS0 time slots, which support up to 1.536 MB.<br>• **e1**—Specifies 31 DS0 time slots, which support up to 2.048 MB. |

| | Command or Action | Purpose |
|---|---|---|
| Step 3 | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# end`<br>or<br>`RP/0/0/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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

# Configuring a Clear Channel E3 Controller

When an E3 controller is in clear channel mode, it carries a single serial interface.

The E3 controllers are configured using the E3 configuration mode.

## Prerequisites

You must first use the **hw-module subslot cardtype** command to set the card to support E3.

## Restrictions

- If you configure an option that is not valid for your controller type, you receive an error when you commit the configuration.
- A single SPA cannot support a mixture of T3 and E3 interfaces.
- This task is applicable to 2-Port and 4-Port Clear Channel T3/E3 SPAs only.

### SUMMARY STEPS

1. **configure**
2. **controller e3** *interface-path-id*
3. **mode serial**
4. **no shutdown**

5. **end**
   or
   **commit**
6. **show controllers e3** *interface-path-id*

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **`configure`**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **`controller e3`** *`interface-path-id`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller t3 0/1/0/0` | Specifies the E3 controller name in the notation *rack/slot/module/port* and enters E3 configuration mode. |
| **Step 3** | **`mode serial`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-e3)# mode serial` | Configures the mode of the port to be clear channel serial.<br><br>**Note** This step is required for the 2-Port and 4-Port Channelized T3 SPA only. The 2-Port and 4-Port Clear Channel T3/E3 SPA run in serial mode by default. |
| **Step 4** | **`no shutdown`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-e3)# no shutdown` | Removes the shutdown configuration.<br><br>• The removal of the shutdown configuration removes the forced administrative down on the controller, enabling the controller to move to an up or a down state. |

| | Command or Action | Purpose |
|---|---|---|
| Step 5 | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-e3)# end`<br>or<br>`RP/0/0/CPU0:router(config-e3)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 6 | **show controllers e3** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router# show controllers e3 0/1/0/0` | (Optional) Displays information about the E3 controllers. |

## What to Do Next

- Modify the default configuration that is running on the E3 controller you just configured, as described in the "Modifying the Default E3 Controller Configuration" section later in this module.
- Configure a bit error rate test (BERT) on the controller to test its integrity, as described in the "Configuring BERT" section later in this module.
- Configure the associated serial interface, as described in the *Configuring Serial Interfaces on Cisco IOS XR Sotware* module later in this document.

# Modifying the Default E3 Controller Configuration

This task explains how to modify the default E3 controller configuration, which is described in the "Default Configuration Values for T3 and E3 Controllers" section earlier in this module.

## Prerequisites

You must configure a clear channel E3 controller, as described in the "Configuring a Clear Channel E3 Controller" section earlier in this module.

## SUMMARY STEPS

1. **configure**
2. **controller e3** *interface-path-id*
3. **clock source** {**internal** | **line**}
4. **cablelength** *feet*
5. **framing** {**g751** | **g832**}
6. **national bits** {**disable** | **enable**}
7. **no shutdown**
8. **end**
   or
   **commit**
9. **show controllers e3** *interface-path-id*

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **controller e3** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller t3 0/1/0/0` | Specifies the E3 controller name in the notation *rack/slot/module/port* and enters E3 configuration mode. |
| **Step 3** | **clock source** {**internal** \| **line**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-e3)# clock source internal` | (Optional) Sets the clocking for individual E3 links.<br><br>**Note** The default clock source is **internal**.<br><br>**Note** When configuring clocking on a serial link, you must configure one end to be **internal**, and the other end to be **line**. If you configure **internal** clocking on both ends of a connection, framing slips occur. If you configure **line** clocking on both ends of a connection, the line does not come up. |
| **Step 4** | **cablelength** *feet*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-e3)# cablelength 250` | (Optional) Specifies the distance of the cable from the router to the network equipment.<br><br>**Note** The default cable length is 224 feet. |
| **Step 5** | **framing** {**g751** \| **g832**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-e3)# framing g832` | (Optional) Selects the frame type for the E3 port. Possible E3 frame types are G.751 and G.832.<br><br>**Note** The default framing for E3 is G.751. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 6** | **national bits** {**disable** \| **enable**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-e3)# national bits enable` | (Optional) Enables or disables the 0x1F national reserved bit pattern on the E3 port.<br>**Note** The E3 national bit is enabled by default, and the bit pattern value is 1. |
| **Step 7** | **no shutdown**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-e3)# no shutdown` | Removes the shutdown configuration.<br>• The removal of the shutdown configuration removes the forced administrative down on the controller, enabling the controller to move to an up or a down state. |
| **Step 8** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-e3)# end`<br>or<br>`RP/0/0/CPU0:router(config-e3)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 9** | **show controllers e3** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router# show controllers e3 0/1/0/0` | (Optional) Displays information about the E3 controllers. |

## What to Do Next

- Modify the default configuration that is running on the T3 controller you just configured, as described in the "Modifying the Default T3 Controller Configuration" section later in this module.
- Configure BERT on the controller to test its integrity, as described in the "Configuring BERT" section later in this module.
- Configure the associated serial interface, as described in the *Configuring Serial Interfaces on Cisco IOS XR Sotware* module later in this document.

# Configuring a Clear Channel T3 Controller

When a T3 controller is in clear channel mode, it carries a single serial interface.

The T3 controllers are configured in the T3 configuration mode.

## Prerequisites

You must use the **hw-module subslot cardtype** command to set the card to support T3, as described in the "Setting the Card Type"section on earlier in this module.

## Restrictions

- This task is applicable to 2-Port and 4-Port Clear Channel T3/E3 SPAs only.
- If you configure an option that is not valid for your controller type, you receive an error when you commit the configuration.
- A single SPA cannot support a mixture of T3 and E3 interfaces.

### SUMMARY STEPS

1. **configure**
2. **controller t3** *interface-path-id*
3. **mode serial**
4. **no shutdown**
5. **end**
   or
   **commit**
6. **show controllers t3** *interface-path-id*

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **controller t3** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller t3 0/1/0/0` | Specifies the T3 controller name in the *rack/slot/module/port* notation and enters T3 configuration mode. |
| **Step 3** | **mode serial**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# mode serial` | Configures the mode of the port to be clear channel serial.<br>**Note** This step is required for the 2-Port and 4-Port Channelized T3 SPA only. The 2-Port and 4-Port Clear Channel T3/E3 SPA runs in serial mode by default. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **no shutdown**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# no shutdown` | Removes the shutdown configuration.<br>• The removal of the shutdown configuration removes the forced administrative down on the controller, enabling the controller to move to an up or a down state. |
| **Step 5** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# end`<br>or<br>`RP/0/0/CPU0:router(config-t3)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 6** | **show controllers t3** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router# show controllers t3 0/1/0/0` | (Optional) Displays information about the T3 controllers. |

## What to Do Next

- Modify the default configuration that is running on the T3 controller you just configured, as described in the "Modifying the Default T3 Controller Configuration" section later in this module.
- Configure BERT on the controller to test its integrity, as described in the "Configuring BERT" section later in this module.
- Configure the associated serial interface, as described in the *Configuring Serial Interfaces on Cisco IOS XR Sotware* module.

# Configuring a Channelized T3 Controller

The SPAs that support channelized T3 support channelization to T1, E1, and DS0. The steps in this section describe how to channelize a single T3 controller into 28 T1 controllers or 21 E1 controllers. Once you have created T1 or E1 controllers, you can further channelize those controllers into DS0 time slots, as described in the following sections:

- Configuring a T1 Controller
- Configuring an E1 Controller

Each individual T1 controller supports a total of 24 DS0 time slots, and each individual E1 controller supports a total of 31 DS0 time slots.

## Prerequisites

Before you configure a channelized T3 controller, be sure that the following requirements are met:

- You have one of the following SPAs installed:
  - 2-Port Channelized T3 SPA
  - 4-Port Channelized T3/DS0 SPA

**Note** If you configure an option that is not valid for your controller type, you receive an error when you commit the configuration.

### SUMMARY STEPS

1. **configure**
2. **controller t3** *interface-path-id*
3. **mode** [**t1** | **e1**]
4. **no shutdown**
5. **end**
   or
   **commit**
6. **show controllers t3** *interface-path-id*

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | `configure`<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | `controller T3` *`interface-path-id`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller t3 0/1/0/0` | Specifies the T3 controller name in the notation *rack/slot/module/port* and enters T3 configuration mode. |
| **Step 3** | `mode t1`<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# mode t1` | Sets the mode of the channelized controllers to be T1, and creates 28 T1 controllers. |
| **Step 4** | `no shutdown`<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# no shutdown` | Removes the shutdown configuration.<br>• The removal of the shutdown configuration removes the forced administrative down on the controller, enabling the controller to move to an up or a down state. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# end`<br>or<br>`RP/0/0/CPU0:router(config-t3)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 6** | **show controllers t3** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router# show controllers t3 0/1/0/0` | (Optional) Displays information about the T3 controllers. |

## What to Do Next

- Modify the default configuration that is running on the T3 controller you just configured, as described in the "Modifying the Default T3 Controller Configuration" section on page 391.
- If you channelized your T3 controller into 28 T1 controllers, configure the T1 controllers and assign DS0 time slots to them, as described in the *"Configuring a T1 Controller" section on page 394*.
- If you channelized your T3 controller into 21 E1 controllers, configure the E1 controllers and assign DS0 time slots to them, as described in the *"Configuring an E1 Controller" section on page 397*.

# Modifying the Default T3 Controller Configuration

This task explains how to modify the default T3 controller configuration, which is described in the "Default Configuration Values for T3 and E3 Controllers" section on page 379.

## Prerequisites

You must configure a clear channel or channelized T3 controller, as described in one of the following sections:

- Configuring a Clear Channel T3 Controller
- Configuring a Channelized T3 Controller

## SUMMARY STEPS

1. **configure**
2. **controller t3** *interface-path-id*
3. **clock source** {**internal** | **line**}
4. **cablelength** *feet*
5. **framing** {**auto-detect** | **c-bit** | **m23**}
6. **mdl transmit** {**idle-signal** | **path** | **test-signal**} {**disable** | **enable**}
7. **mdl string** {**eic** | **fi** | **fic** | **gen-number** | **lic** | **port-number** | **unit**} *string*
8. **no shutdown**
9. **end**<br>or<br>**commit**
10. **show controllers t3** *interface-path-id*

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | `configure`<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **`controller T3`** *`interface-path-id`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller t3 0/1/0/0` | Specifies the T3 controller name in the notation *rack/slot/module/port* and enters T3 configuration mode. |
| **Step 3** | **`clock source`** {**`internal`** \| **`line`**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# clock source internal` | (Optional) Sets the clocking for the T3 port.<br>**Note** The default clock source is **internal**.<br>**Note** When configuring clocking on a serial link, you must configure one end to be **internal**, and the other end to be **line**. If you configure **internal** clocking on both ends of a connection, framing slips occur. If you configure **line** clocking on both ends of a connection, the line does not come up. |
| **Step 4** | **`cablelength`** *`feet`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# cablelength 250` | (Optional) Specifies the distance of the cable from the router to the network equipment.<br>**Note** The default cable length is 224 feet. |
| **Step 5** | **`framing`** {**`auto-detect`** \| **`c-bit`** \| **`m23`**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# framing c-bit` | (Optional) Selects the frame type for the T3 port.<br>**Note** The default frame type for T3 is C-bit. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 6** | **mdl transmit** {**idle-signal** \| **path** \| **test-signal**} {**disable** \| **enable**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# mdl transmit path enable` | (Optional) Enables Maintenance Data Link (MDL) messages on the T3 port.<br>**Note** MDL messages are supported only when the T3 framing is C-bit parity.<br>**Note** MDL message are disabled by default. |
| **Step 7** | **mdl string** {**eic** \| **fi** \| **fic** \| **gen-number** \| **lic** \| **port-number** \| **unit**} *string*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# mdl fi facility identification code` | (Optional) Specifies the values of the strings sent in the MDL messages. |
| **Step 8** | **no shutdown**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# no shutdown` | Removes the shutdown configuration.<br>• The removal of the shutdown configuration removes the forced administrative down on the controller, enabling the controller to move to an up or a down state. |
| **Step 9** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# end`<br>or<br>`RP/0/0/CPU0:router(config-t3)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 10** | **show controllers t3** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router# show controllers t3 0/1/0/0` | (Optional) Displays information about the T3 controllers. |

## What to Do Next

- If you configured a clear channel T3 controller, perform the following tasks:
  - Configure BERT on the controller to test its integrity, as described in the "Configuring BERT" section on page 400 later in this module.

  - Configure the associated serial interface, as described in the *Configuring Serial Interfaces on Cisco IOS XR Sotware* module.
- If you channelized your T3 controller into 28 T1 controllers, configure the T1 controllers and assign DS0 time slots to them, as described in the *"Configuring a T1 Controller" section on page 394*.
- If you channelized your T3 controller into 21 E1 controllers, configure the E1 controllers and assign DS0 time slots to them, as described in the *"Configuring an E1 Controller" section on page 397*.

# Configuring a T1 Controller

This task describes how to configure an individual T1 controller and channelize it into 24 individual DS0 timeslots.

## Prerequisites

Before you configure a T1 controller, be sure that the following requirements are met:

- You have one of the following SPAs installed:
  - 2-Port Channelized T3 SPA
  - 4-Port Channelized T3/DS0 SPA
  - 8-Port Channelized T1/E1 SPA
- If you have a 2-Port or 4-Port Channelized T3/DS0 SPA, you must configure a channelized T3 controller to run in T1 mode, as described in the "Configuring a Channelized T3 Controller" section on page 389.

## Restrictions

If you configure an option that is not valid for your controller type, you receive an error when you commit the configuration.

### SUMMARY STEPS

1. **show controllers t1** *interface-path-id*
2. **configure**
3. **controller t1** *interface-path-id*
4. **framing** {**sf** | **esf**}
5. **yellow** {**detection** | **generation**} {**disable** | **enable**}
6. **clock source** {**internal** | **line**}
7. **fdl** {**ansi** | **att**} {**enable** | **disable**}
8. **no shutdown**
9. **channel-group** *channel-group-number*
10. **timeslots** *range*
11. **speed** *kbps*
12. **exit**

13. Repeat Step 9 through Step 12 to assign time slots to a channel group. Each controller can contain up to 24 time slots.
14. **exit**
15. Repeat Step 2 through Step 14 to assign more channel groups to a controller.
16. **end**
    or
    **commit**

## DETAILED STEPS

| | | |
|---|---|---|
| **Step 1** | **`show controllers t1`** *`interface-path-id`*<br><br>**Example:**<br>`RP/0/0/CPU0:router# show controllers t3 0/1/0/0` | (Optional) Displays information about the T1 controllers you created in Step 3. |
| **Step 2** | **`configure`**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 3** | **`controller t1`** *`interface-path-id`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller t1 0/3/0/0/0` | Enters T1 configuration mode. |
| **Step 4** | **`framing`** {**`sf`** \| **`esf`**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1)# framing esf` | (Optional) Selects the frame type for the T1 data line:<br>• **sf**—Superframe<br>• **esf**—Extended super frame<br>**Note** The default frame type for T1 is Extended superframe (**esf**). |
| **Step 5** | **`yellow`** {**`detection`** \| **`generation`**} {**`disable`** \| **`enable`**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1e1)# yellow detection enable` | (Optional) Enables or disables the detection and generation of T1 yellow alarms.<br>**Note** Yellow alarms are detected and generated on the T1 channel by default. |
| **Step 6** | **`clock source`** {**`internal`** \| **`line`**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1e1)# clock source internal` | (Optional) Sets the clocking for individual T1 links.<br>**Note** The default clock source is **internal**.<br>**Note** When configuring clocking on a serial link, you must configure one end to be **internal**, and the other end to be **line**. If you configure **internal** clocking on both ends of a connection, framing slips occur. If you configure **line** clocking on both ends of a connection, the line does not come up. |

| | | |
|---|---|---|
| **Step 7** | **fdl** {**ansi** \| **att**} {**enable** \| **disable**}<br><br>Example:<br>`RP/0/0/CPU0:router(config-t1e1)# fdl ansi enable` | (Optional) Enables the transmission of ANSI T1.403 or AT&T TR54016 once-per-second performance reports through Facility Data Link (FDL).<br>**Note** FDL ansi and att are disabled by default. |
| **Step 8** | **no shutdown**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1e1)# no shutdown` | Removes the shutdown configuration.<br>• The removal of the shutdown configuration removes the forced administrative down on the controller, enabling the controller to move to an up or a down state. |
| **Step 9** | **channel-group** *channel-group-number*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1)# channel-group 0` | Creates a T1 channel group and enters channel group configuration mode for that channel group. |
| **Step 10** | **timeslots** *range*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1-channel_group)# timeslots 7-12` | Associates one or more DS0 time slots to a channel group and creates an associated serial subinterface on that channel group.<br>• Range is from 1 to 24 time slots.<br>• You can assign all 24 time slots to a single channel group, or you can divide the time slots among several channel groups.<br>**Note** Each individual T1 controller supports a total of 24 DS0 time slots. |
| **Step 11** | **speed** *kbps*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1e1-channel_group)# speed 64` | (Optional) Specifies the speed of the DS0s in kilobits per second. Valid values are 56 and 64.<br>**Note** The default speed is 64 kbps. |
| **Step 12** | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1-channel_group)# exit` | Exits channel group configuration mode. |
| **Step 13** | Repeat Step 9 through Step 12 to assign time slots to a channel group. Each controller can contain up to 24 time slots. | — |
| **Step 14** | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1)# exit` | Exits T1 configuration mode and enters global configuration mode. |

| | | |
|---|---|---|
| **Step 15** | Repeat Step 2 through Step 14 to assign more channel groups to a controller as desired. | — |
| **Step 16** | **`end`**<br>or<br>**`commit`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# end`<br>or<br>`RP/0/0/CPU0:router(config-t3)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

## What to Do Next

- Configure BERT on the controller to test its integrity, as described in the "Configuring BERT" section on page 400.
- Configure the associated serial interface, as described in the *Configuring Serial Interfaces on Cisco IOS XR Sotware* module.

# Configuring an E1 Controller

This task describes how to configure an individual E1 controller and channelize it into 31 individual DS0 timeslots.

## Prerequisites

Before you configure an E1 controller, be sure that the following requirements are met:

- You have one of the following SPAs installed:
  - 2-Port Channelized T3 SPA
  - 4-Port Channelized T3/DS0 SPA
  - 8-Port Channelized T1/E1 SPA
- If you have a 2-Port or 4-Port Channelized T3/DS0 SPA, you must configure a channelized T3 controller to run in E1 mode, as described in the "Configuring a Channelized T3 Controller" section on page 389.

## Restrictions

If you configure an option that is not valid for your controller type, you receive an error when you commit the configuration.

### SUMMARY STEPS

1. **show controllers e1** *interface-path-id*
2. **configure**
3. **controller e1** *interface-path-id*
4. **clock source** {**internal** | **line**}
5. **framing** {**crc4** | **no-crc4** | **unframed**}
6. **national bits** *bits*
7. **no shutdown**
8. **channel-group** *channel-group-number*
9. **timeslots** *range*
10. **speed** *kbps*
11. **exit**
12. Repeat Step 8 through Step 11 to assign time slots to a channel group. Each controller can contain up to 24 time slots.
13. **exit**
14. Repeat Step 2 through Step 13 to assign more channel groups to a controller as desired.
15. **end**
    or
    **commit**

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| Step 1 | `show controllers e1 interface-path-id`<br><br>**Example:**<br>`RP/0/0/CPU0:router# show controllers e1 0/1/0/0` | (Optional) Displays information about the E1 controllers. |
| Step 2 | `configure`<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| Step 3 | `controller e1 interface-path-id`<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller e1 0/3/0/0/0` | Enters E1 configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| Step 4 | **clock source** {**internal** \| **line**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-e1)# clock source internal` | (Optional) Sets the clocking for individual E1 links.<br>**Note** The default clock source is **internal**.<br>**Note** When configuring clocking on a serial link, you must configure one end to be **internal**, and the other end to be **line**. If you configure **internal** clocking on both ends of a connection, framing slips occur. If you configure **line** clocking on both ends of a connection, the line does not come up. |
| Step 5 | **framing** {**crc4** \| **no-crc4** \| **unframed**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-e1)# framing unframed` | (Optional) Selects the frame type for the E1 data line. The following frame types are valid for E1:<br>• **crc4**—Framing with CRC-4 error monitoring capabilities<br>• **no-crc4**—Framing without CRC-4 error monitoring capabilities<br>• **unframed**—Unframed E1<br>**Note** The default frame type for E1 is **crc4**. |
| Step 6 | **national bits** *bits*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-e1)# national bits 10` | (Optional) Specifies the national reserved bits for an E1 port. Range is from 0 to 31.<br>**Note** The default bit pattern is 0, which corresponds to the hexadecimal value *0x1f*. |
| Step 7 | **no shutdown**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-e1)# no shutdown` | Removes the shutdown configuration.<br>• The removal of the shutdown configuration removes the forced administrative down on the controller, enabling the controller to move to an up or a down state. |
| Step 8 | **channel-group** *channel-group-number*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-e1)# channel-group 0` | Creates an E1 channel group and enters channel group configuration mode for that channel group. |
| Step 9 | **timeslots** *range*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-e1-channel_group)# timeslots 1-16` | Associates one or more time slots to a channel group and creates an associated serial subinterface on that channel group.<br>• Range is from 1 to 31 time slots.<br>• You can assign all 31 time slots to a single channel group, or you can divide the time slots among several channel groups.<br>**Note** Each E1 controller supports a total of 31 DS0 time slots. |
| Step 10 | **speed** *kbps*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-e1-channel_group)# speed 100` | (Optional) Specifies the speed of the DS0s in kilobits per second. Valid values are 56 and 64.<br>**Note** The default speed is 64 kbps. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 11** | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-e1-channel_group)# exit` | Exits channel group configuration mode |
| **Step 12** | Repeat Step 8 through Step 11 to assign time slots to a channel group. | — |
| **Step 13** | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-e1)# exit` | Exits E1 configuration mode |
| **Step 14** | Repeat Step 2 through Step 13 to assign more channel groups to a controller as desired. | — |
| **Step 15** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-e3)# end`<br>or<br>`RP/0/0/CPU0:router(config-e3)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

## What to Do Next

- Configure BERT on the controller to test its integrity, as described in the "Configuring BERT" section on page 400 in this module.
- Configure the associated serial interface, as described in the *Configuring Serial Interfaces on Cisco IOS XR Sotware* module later in this document.

# Configuring BERT

Depending on your hardware support, BERT is supported on each of the T3/E3 or T1/E1 controllers, and on the DS0 channel groups. It is done only over an unframed T3/E3 or T1/E1 signal and is run on only one port at a time. It is also supported on individual channel groups.

To view the BERT results, use the **show controllers t1** or **show controllers t3** command in EXEC mode. The BERT results include the following information:

- Type of test pattern selected
- Status of the test
- Interval selected
- Time remaining on the BER test
- Total bit errors
- Total bits received

BERT is data intrusive. Regular data cannot flow on a line while the test is in progress. The line is put in an alarm state when BERT is in progress and restored to a normal state after BERT has been terminated.

## Configuring BERT on T3/E3 and T1/E1 Controllers

This task explains how to enable a bit error rate test (BERT) pattern on a T3/E3 or T1/E1 line or an individual channel group.

## Prerequisites

You must have configured a clear channel T3/E3 controller, or a channelized T3-to-T1/E1 controller.

## Restrictions

Valid patterns for all controllers and channel groups include: 0s, 1s, 2^15, 2^20, 2^20-QRSS, 2^23, and alt-0-1.

Additional valid patterns for T1 and E1 controllers include: 1in8, 3in24, 55Daly, and 55Octet. Additional valid patterns for channel groups include: 2^11, 2^9, ds0-1, ds0-2, ds0-3, and ds0-4.

### SUMMARY STEPS

1. **configure**
2. **controller** [**t3** | **e3** | **t1** | **e1**] *interface-path-id*
3. **pattern** *pattern*
4. **bert interval** *time*
5. **bert error** [*number*]
6. **end**
   or
   **commit**
7. **exit**
8. **exit**
9. **bert** [**t3** | **e3** | **t1** | **e1**] *interface-path-id* [**channel-group** *channel-group-number*] [**error**] **start**
10. **bert** [**t3** | **e3** | **t1** | **e1**] *interface-path-id* [**channel-group** *channel-group-number*] **stop**
11. **show controllers** [**t3** | **e3** | **t1** | **e1**] *interface-path-id*

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **controller** [**t3** \| **e3** \| **t1** \| **e1**] *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller t3 0/1/0/0` | Specifies the controller name and instance in the notation *rack/slot/module/por*t, and enters T3, E3, T1, or E1 controller configuration mode. |
| **Step 3** | **bert pattern** *pattern*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# bert pattern 2^15` | Enables a specific bit error rate test (BERT) pattern on a controller.<br>**Note** You must use the **bert** command in EXEC mode to start the BER test. |
| **Step 4** | **bert interval** *time*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# bert pattern 2^15` | (Optional) Specifies the duration of a bit error rate test (BERT) pattern on a T3/E3 or T1/E1 line. The interval can be a value from 1 to 14400. |
| **Step 5** | **bert error** [*number*]<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# bert error 10` | Specifies the number of BERT errors to introduce into the bit stream. Range is from 1 to 255. |
| **Step 6** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# end`<br>or<br>`RP/0/0/CPU0:router(config-t3)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 7** | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# exit` | Exits T3/E3 or T1/E1 controller configuration mode. |
| **Step 8** | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# exit` | Exits global configuration mode. |
| **Step 9** | **bert** [**t3** \| **e3** \| **t1** \| **e1**] *interface-path-id* [**channel-group** *channel-group-number*] [**error**] **start**<br><br>**Example:**<br>`RP/0/0/CPU0:router# bert t3 0/3/0/0 start`<br>`RP/0/0/CPU0:router# bert t3 0/3/0/0 error` | Starts the configured BERT test on the specified T3/E3 or T1/E1 controller.<br><br>**Note** You can include the optional **error** keyword to inject errors into the running BERT stream. |
| **Step 10** | **bert** [**t3** \| **e3** \| **t1** \| **e1**] *interface-path-id* [**channel-group** *channel-group-number*] **stop**<br><br>**Example:**<br>`RP/0/0/CPU0:router# bert t3 0/3/0/0 stop` | Stops the configured BERT test on the specified T3/E3 or T1/E1 controller. |
| **Step 11** | **show controllers** [**t3** \| **e3** \| **t1** \| **e1**] *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router# show controllers t3 0/3/0/0` | Displays the results of the configured BERT. |

### What to Do Next

Configure the serial interfaces that are associate with the controllers you tested, as described in the *Configuring Serial Interfaces on Cisco IOS XR Sotware* module.

## Configuring BERT on a DS0 Channel Group

This task explains how to enable a bit error rate test (BERT) pattern on an individual DS0 channel group.

## Prerequisites

You must have configured a clear channel T1/E1 controller, or a channelized T3-to-T1/E1 controller.

### SUMMARY STEPS

1. **configure**
2. **controller** {**t1** | **e1**} *interface-path-id*
3. **channel-group** *channel-group-number*
4. **bert pattern** *pattern*
5. **bert interval** *time*

6. **end**
   or
   **commit**
7. **exit**
8. **exit**
9. **exit**
10. **bert** [**t1** | **e1**] *interface-path-id* [**channel-group** *channel-group-number*][**error**] **start**
11. **bert** [**t1** | **e1**] *interface-path-id* [**channel-group** *channel-group-number*] **stop**
12. **show controllers** [**t1** | **e1**] *interface-path-id*

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| Step 1 | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| Step 2 | **controller** {**t1** \| **e1**} *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller t3 0/1/0/0` | Specifies the controller name and instance in the notation *rack/slot/module/por*t, and enters T1 or E1 controller configuration mode. |
| Step 3 | **channel-group** *channel-group-number*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1)# channel-group 1`<br>`RP/0/0/CPU0:router(config-t1-channel_group)#` | Enters channel group configuration mode for a specific channel group. Replace *channel-group-number* with the number that identifies the channel group on which you want to configure a BERT. |
| Step 4 | **bert pattern** *pattern*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1-channel_group)# bert pattern 2^15` | Enables a specific bit error rate test (BERT) pattern on a T1 line. Valid patterns for all controllers and channel groups include: **0s**, **1s**, **2^15**, **2^20**, **2^20-QRSS**, **2^23**, and **alt-0-1**. Additional valid patterns for T1 and E1 controllers include: **1in8**, **3in24**, **55Daly**, and **55Octet**. Additional valid patterns for channel groups include: **2^11**, **2^9**, **ds0-1**, **ds0-2**, **ds0-3**, and **ds0-4.**<br><br>**Note** You must use the **bert** command in EXEC mode to start the BER test. |
| Step 5 | **bert interval** *time*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1-channel_group)# bert interval 5` | (Optional) Specifies the duration, in minutes, of a bit error rate test (BERT) pattern on a T1/E1 line. The interval can be a value from 1 to 14400. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 6** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1-channel_group)# end`<br>or<br>`RP/0/0/CPU0:router(config-t1-channel_group)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 7** | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1-channel_group)# exit` | Exits channel group configuration mode. |
| **Step 8** | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1)# exit` | Exits T1 or E1 configuration mode. |
| **Step 9** | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# exit` | Exits global configuration mode. |
| **Step 10** | **bert** [**t1** \| **e1**] *interface-path-id* [**channel-group** *channel-group-number*] [**error**] **start**<br><br>**Example:**<br>`RP/0/0/CPU0:router# bert t1 0/3/0/0/0 start`<br>`RP/0/0/CPU0:router# bert t1 0/3/0/0/0 error` | Starts the configured BERT test on the specified channel group.<br>**Note** You can include the optional **error** keyword to inject errors into the running BERT stream. |

| | Command or Action | Purpose |
|---|---|---|
| Step 11 | **bert** [**t1** \| **e1**] *interface-path-id* [**channel-group** *channel-group-number*] **stop**<br><br>**Example:**<br>`RP/0/0/CPU0:router# bert t1 0/3/0/0/0 stop` | Stops the configured BERT test on the specified channel group. |
| Step 12 | **show controllers** [**t1** \| **e1**] *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router# show controllers t3 0/3/0/0` | Displays the results of the configured BERT. |

### What to Do Next

Configure the serial interfaces that are associate with the controllers you tested, as described in the *Configuring Serial Interfaces on Cisco IOS XR Sotware* module later in this document.

# Configuration Examples

This section contains the following examples:


- Configuring a Clear Channel T3 Controller: Example, page 406
- Configuring a T3 Controller with Channelized T1 Controllers: Example, page 406
- Configuring BERT on a T3 Controller: Example, page 408


## Configuring a Clear Channel T3 Controller: Example

The following example shows configuration for a clear channel T3 controller:

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)#controller T3 0/3/2/0
RP/0/0/CPU0:router(config-t3)#clock source internal
RP/0/0/CPU0:router(config-t3)#mode serial
RP/0/0/CPU0:router(config-t3)#cablelength 4
RP/0/0/CPU0:router(config-t3)#framing c-bit
RP/0/0/CPU0:router(config-t3)#commit
```

## Configuring a T3 Controller with Channelized T1 Controllers: Example

The following example shows how to configure a T3 controller that has been channelized 28 T1 controllers:

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# controller T3 0/3/0/0
RP/0/0/CPU0:router(config-t3)# mode t1
RP/0/0/CPU0:router(config-t3)# framing m23
RP/0/0/CPU0:router(config-t3)# cablelength 11
RP/0/0/CPU0:router(config-t3)# clock source line
RP/0/0/CPU0:router(config-t3)#commit
RP/0/0/CPU0:router(config-t3)#exit
RP/0/0/CPU0:router(config)# exit
RP/0/0/CPU0:router# show controllers T1 ?
```

```
  0/3/0/0/0   T1 Interface Instance
  0/3/0/0/1   T1 Interface Instance
  0/3/0/0/10  T1 Interface Instance
  0/3/0/0/11  T1 Interface Instance
  0/3/0/0/12  T1 Interface Instance
  0/3/0/0/13  T1 Interface Instance
  0/3/0/0/14  T1 Interface Instance
  0/3/0/0/15  T1 Interface Instance
  0/3/0/0/16  T1 Interface Instance
  0/3/0/0/17  T1 Interface Instance
  0/3/0/0/18  T1 Interface Instance
  0/3/0/0/19  T1 Interface Instance
  0/3/0/0/2   T1 Interface Instance
  0/3/0/0/20  T1 Interface Instance
  0/3/0/0/21  T1 Interface Instance
  0/3/0/0/22  T1 Interface Instance
  0/3/0/0/23  T1 Interface Instance
  0/3/0/0/24  T1 Interface Instance
  0/3/0/0/25  T1 Interface Instance
  0/3/0/0/26  T1 Interface Instance
  0/3/0/0/27  T1 Interface Instance
  0/3/0/0/3   T1 Interface Instance
  0/3/0/0/4   T1 Interface Instance
  0/3/0/0/5   T1 Interface Instance
 --More--
!
RP/0/0/CPU0:router#
RP/0/0/CPU0:router(config)#configure
RP/0/0/CPU0:router(config)# controller t1 0/3/0/0/0
RP/0/0/CPU0:router(config-t1)# channel-group 0
RP/0/0/CPU0:router(config-t1-channel_group)# timeslots 1-24
RP/0/0/CPU0:router(config-t1-channel_group)# exit
RP/0/0/CPU0:router(config-t1)# exit
RP/0/0/CPU0:router(config)# controller t1 0/3/0/0/1
RP/0/0/CPU0:router(config-t1)# channel-group 0
RP/0/0/CPU0:router(config-t1-channel_group)# timeslots 1-24
RP/0/0/CPU0:router(config-t1-channel_group)# exit
RP/0/0/CPU0:router(config-t1)# exit
RP/0/0/CPU0:router(config)# controller t1 0/3/0/0/2
RP/0/0/CPU0:router(config-t1)# channel-group 0
RP/0/0/CPU0:router(config-t1-channel_group)# timeslots 1-12
RP/0/0/CPU0:router(config-t1-channel_group)# exit
RP/0/0/CPU0:router(config-t1)# channel-group 1
RP/0/0/CPU0:router(config-t1-channel_group)# timeslots 13-24
RP/0/0/CPU0:router(config-t1-channel_group)# exit
RP/0/0/CPU0:router(config-t1)# exit
RP/0/0/CPU0:router(config)# controller t1 0/3/0/0/3
RP/0/0/CPU0:router(config-t1)# channel-group 0
RP/0/0/CPU0:router(config-t1-channel_group)# timeslots 1-6
RP/0/0/CPU0:router(config-t1-channel_group)# exit
RP/0/0/CPU0:router(config-t1)# channel-group 1
RP/0/0/CPU0:router(config-t1-channel_group)# timeslots 7-12
RP/0/0/CPU0:router(config-t1-channel_group)# exit
RP/0/0/CPU0:router(config-t1)# channel-group 2
RP/0/0/CPU0:router(config-t1-channel_group)# timeslots 13-18
RP/0/0/CPU0:router(config-t1-channel_group)# exit
RP/0/0/CPU0:router(config-t1)# channel-group 3
RP/0/0/CPU0:router(config-t1-channel_group)# timeslots 19-24
RP/0/0/CPU0:router(config-t1-channel_group)# exit
RP/0/0/CPU0:router(config-t1-channel_group)#commit
```

# Configuring BERT on a T3 Controller: Example

The following example shows how to configure a BERT on a T3 controller, and then display the results of the BERT:

```
RP/0/0/CPU0:router# config
RP/0/0/CPU0:router(config)# controller t3 0/3/0/1
RP/0/0/CPU0:router(config-t3)# bert pattern 0s

Run bert from exec mode for the bert config to take effect

RP/0/0/CPU0:router(config-t3)#exit
RP/0/0/CPU0:router(config)# exit

Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]
RP/0/0/CPU0:router# bert t3 0/3/0/1 start


RP/0/0/CPU0:router# bert t3 0/3/0/1 stop


RP/0/0/CPU0:router# show controllers t3 0/3/0/1

  T30/3/0/1 is up
  No alarms detected.
  MDL transmission is disabled
    EIC: , LIC: , FIC: , UNIT:
     Path FI:
     Idle Signal PORT_NO:
     Test Signal GEN_NO:
  FEAC code received: No code is being received
  Framing is C-BIT Parity, Line Code is B3ZS, Clock Source is Internal
  Data in current interval (108 seconds elapsed):
     0 Line Code Violations, 0 P-bit Coding Violation
     0 C-bit Coding Violation, 0 P-bit Err Secs
     0 P-bit Severely Err Secs, 0 Severely Err Framing Secs
     0 Unavailable Secs, 0 Line Errored Secs
     0 C-bit Errored Secs, 0 C-bit Severely Errored Secs
  Data in Interval 1:
     0 Line Code Violations, 0 P-bit Coding Violation
     0 C-bit Coding Violation, 0 P-bit Err Secs
     0 P-bit Severely Err Secs, 0 Severely Err Framing Secs
     0 Unavailable Secs, 0 Line Errored Secs
     0 C-bit Errored Secs, 0 C-bit Severely Errored Secs
  Data in Interval 2:
     0 Line Code Violations, 0 P-bit Coding Violation
     0 C-bit Coding Violation, 0 P-bit Err Secs
     0 P-bit Severely Err Secs, 0 Severely Err Framing Secs
     0 Unavailable Secs, 0 Line Errored Secs
     0 C-bit Errored Secs, 0 C-bit Severely Errored Secs
  Data in Interval 3:
     0 Line Code Violations, 0 P-bit Coding Violation
     0 C-bit Coding Violation, 0 P-bit Err Secs
     0 P-bit Severely Err Secs, 0 Severely Err Framing Secs
     0 Unavailable Secs, 0 Line Errored Secs
     0 C-bit Errored Secs, 0 C-bit Severely Errored Secs
```

# Additional References

These sections provide references related to T3/E3 and T1/E1 controllers.

## Related Documents

| Related Topic | Document Title |
|---|---|
| Cisco IOS XR master command reference | *Cisco IOS XR Master Commands List* |
| Cisco IOS XR interface configuration commands | *Cisco IOS XR Interface and Hardware Component Command Reference* |
| Initial system bootup and configuration information for a router using Cisco IOS XR software | *Cisco IOS XR Getting Started Guide* |
| Cisco IOS XR AAA services configuration information | *Cisco IOS XR System Security Configuration Guide* and *Cisco IOS XR System Security Command Reference* |

## 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 |
|---|---|
| • IF-MIB<br>• DS3-MIB<br>• CISCO-DS3-MIB<br>• DS1-MIB<br>**Note** Not supported on the 4-Port Clear Channel T3/E3 SPA.<br>• Entity MIBs | To locate and download MIBs for selected platforms using Cisco IOS XR software, use the Cisco MIB Locator found at this URL:<br>http://www.cisco.com/go/mibs |

## RFCs

| 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/support |

# Configuring POS Interfaces on Cisco IOS XR Software

This module describes the configuration of Packet-over-SONET/SDH (POS) interfaces on the Cisco XR 12000 Series Routers.

POS interfaces provide secure and reliable data transmission over SONET and Synchronous Digital Hierarchy (SDH) frames using Cisco High-Level Data Link Control (HDLC) protocol or Point-to-Point Protocol (PPP) encapsulation. In addition to Cisco HDLC and PPP encapsulation, the Cisco XR 12000 Series Routersupports Frame Relay encapsulation.

The commands for configuring Layer 1 POS interfaces are provided in the *Cisco IOS XR Interface and Hardware Component Command Reference*.

**Feature History for Configuring POS Interfaces on Cisco IOS XR Software**

| Release | Modification |
|---|---|
| Release 3.2 | This feature was introduced on the Cisco XR 12000 Series Router. |
| Release 3.3.0 | Support was added on the Cisco XR 12000 Series Router for the 2-Port OC-48 POS/RPR SPA. |

| | |
|---|---|
| Release 3.4.0 | Support for the following features was introduced on the Cisco XR 12000 Series Router:<br>• Subinterfaces with permanent virtual circuits (PVCs)<br>• Frame Relay encapsulation on POS main interfaces and PVCs on the following hardware:<br>– Cisco 4-Port OC-3 POS/SDH SPA<br>– Cisco 8-Port OC-3 POS/SDH SPA<br>– Cisco 2-Port OC-12 POS/SDH SPA<br>– Cisco 4-Port OC-12 POS/SDH SPA<br>– Cisco 8-Port OC-12 POS/SDH SPA<br>– Cisco 1-Port OC-48/STM-16 POS/SDH SPA<br>– Cisco 2-Port OC-48/STM-16 POS/SDH SPA<br>– Cisco 1-Port OC-192c/STM-64c POS/SDH SPA<br>– Cisco 4-Port OC-3c/STM-1 POS/SDH Line Card<br>– Cisco 8-Port OC-3c/STM-1c POS/SDH Line Card<br>– Cisco 16-Port OC-3c/STM-1c POS/SDH Line Card<br>– Cisco 4-Port Channelized OC-12/STM-4 POS ISE Line Card<br>– Cisco 4-Port OC-12c/STM-4 POS/SDH ISE Line Card<br>– Cisco 1-Port Channelized OC-48/STM-16 POS ISE Line Card<br>– Cisco 1-Port OC-48c/STM-16c POS/SDH ISE Line Card |
| Release 3.5.0 | Support was added on the Cisco XR 12000 Series Router for the following SPAs:<br>• Cisco 1-Port Channelized OC-3/STM-1 SPA<br>• Cisco 1-Port Channelized OC-48/STM-16 SPA<br>• Cisco 1-Port Channelized OC-12/DS0 SPA<br>• Cisco 2-Port OC12 POS<br>• Cisco 4-Port OC12 POS<br>• Cisco 8-Port OC12 POS<br>• Cisco 4-Port OC3 POS<br>• Cisco 8-Port OC3 POS<br>On the Cisco XR 12000 Series Router, L2TPv3-based L2VPN support was added on Frame Relay encapsulated POS interfaces.<br>On the Cisco XR 12000 Series Router, the **l2transport** keyword was added to the **interface** command. |
| Release 3.8.0 | Support was added on the Cisco XR 12000 Series Router for quality of service (QoS) on Layer 2 subinterfaces. |

# Contents


- Prerequisites for Configuring POS Interfaces, page 413
- Information About Configuring POS Interfaces, page 413
- How to Configure a POS Interface, page 418
- Configuration Examples for POS Interfaces, page 437
- Additional References, page 440


# Prerequisites for Configuring POS 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 POS interfaces, be sure that the following conditions are met:

- You know the IP address of the interface you will assign to the new POS interface configuration.
- You have configured a clear channel or channelized SONET controller, as described in the "Configuring Clear Channel SONET Controllers on Cisco IOS XR Software" or "Configuring Channelized SONET/SDH on Cisco IOS XR Software" modules.

# Information About Configuring POS Interfaces

To configure POS interfaces, you must understand the following concepts:


- Cisco HDLC Encapsulation, page 414
- PPP Encapsulation, page 414
- Keepalive Timer, page 415
- Frame Relay Encapsulation, page 416
- Default Settings for POS Interfaces, page 413


On the Cisco XR 12000 Series Router, a single POS interface carries data using PPP, Cisco HDLC, or Frame Relay encapsulation.

The router identifies the POS interface address by the physical layer interface module (PLIM) card rack number, slot number, bay number, and port number that are associated with that interface. If a subinterface and permanent virtual circuits (PVCs) are configured under the POS interface, then the router includes the subinterface number in the POS interface path ID.

## Default Settings for POS Interfaces

When a POS interface is brought up and no additional configuration commands are applied, the default interface settings shown in Table 16 are present. These default settings can be changed by configuration.

*Table 16 POS Modular Services Card and PLIM Default Interface Settings*

| Parameter | Configuration File Entry | Default Settings |
|---|---|---|
| Keepalive<br>**Note** The **keepalive** command applies to POS interfaces using HDLC or PPP encapsulation. It does not apply to POS interfaces using Frame Relay encapsulation. | **keepalive** {*interval* [*retry*] \| **disable**}<br>**no keepalive** | Interval of 10 seconds<br>Retry of:<br>• 5 (with PPP encapsulation)<br>• 3 (with HDLC encapsulation) |
| Encapsulation | **encapsulation** [**hdlc** \| **ppp** \| **frame-relay** [**IETF**]] | **hdlc** |
| Maximum transmission unit (MTU) | **mtu** *bytes* | 4474 bytes |
| Cyclic redundancy check (CRC) | **crc** [**16** \| **32**] | **32** |

**Note** Default settings do not appear in the output of the **show running-config** command.

# Cisco HDLC Encapsulation

Cisco High-Level Data Link Controller (HDLC) is the Cisco proprietary protocol for sending data over synchronous serial links using HDLC. Cisco HDLC also provides a simple control protocol called Serial Line Address Resolution Protocol (SLARP) to maintain serial link keepalives. HDLC is the default encapsulation type for POS interfaces under Cisco IOS XR software. Cisco HDLC is the default for data encapsulation at Layer 2 (data link) of the Open System Interconnection (OSI) stack for efficient packet delineation and error control.

**Note** Cisco HDLC is enabled by default for POS interfaces.

Cisco HDLC uses keepalives to monitor the link state, as described in the "Keepalive Timer" section on page 415.

# PPP Encapsulation

PPP is a standard protocol used to send data over synchronous serial links. PPP also provides a Link Control Protocol (LCP) for negotiating properties of the link. LCP uses echo requests and responses to monitor the continuing availability of the link.

**Note** When an interface is configured with PPP encapsulation, a link is declared down, and full LCP negotiation is re-initiated after three ECHOREQ packets are sent without receiving an ECHOREP response.

PPP provides the following Network Control Protocols (NCPs) for negotiating the properties of data protocols that run on the link:

- IP Control Protocol (IPCP)—negotiates IP properties
- Multiprotocol Label Switching control processor (MPLSCP)—negotiates MPLS properties
- Cisco Discovery Protocol control processor (CDPCP)—negotiates CDP properties
- IPv6CP—negotiates IP Version 6 (IPv6) properties
- Open Systems Interconnection control processor (OSICP)—negotiates OSI properties

PPP uses keepalives to monitor the link state, as described in the "Keepalive Timer" section on page 415.

PPP supports the following authentication protocols, which require a remote device to prove its identity before allowing data traffic to flow over a connection:

- Challenge Handshake Authentication Protocol (CHAP)—CHAP authentication sends a challenge message to the remote device. The remote device encrypts the challenge value with a shared secret and returns the encrypted value and its name to the local router in a response message. The local router attempts to match the remote device's name with an associated secret stored in the local username or remote security server database; it uses the stored secret to encrypt the original challenge and verify that the encrypted values match.
- Microsoft Challenge Handshake Authentication Protocol (MS-CHAP)—MS-CHAP is the Microsoft version of CHAP. Like the standard version of CHAP, MS-CHAP is used for PPP authentication; in this case, authentication occurs between a personal computer using Microsoft Windows NT or Microsoft Windows 95 and a Cisco router or access server acting as a network access server.
- Password Authentication Protocol (PAP)—PAP authentication requires the remote device to send a name and a password, which are checked against a matching entry in the local username database or in the remote security server database.

**Note** For more information on enabling and configuring PPP authentication protocols, see the "Configuring PPP on Cisco IOS XR Software" module later in this manual.

Use the **ppp authentication** command in interface configuration mode to enable CHAP, MS-CHAP, and PAP on a POS interface.

**Note** Enabling or disabling PPP authentication does not effect the local router's willingness to authenticate itself to the remote device.

## Keepalive Timer

Cisco keepalives are useful for monitoring the link state. Periodic keepalives are sent to and received from the peer at a frequency determined by the value of the keepalive timer. If an acceptable keepalive response is not received from the peer, the link makes the transition to the down state. As soon as an acceptable keepalive response is obtained from the peer or if keepalives are disabled, the link makes the transition to the up state.

If three keepalives are sent to the peer and no response is received from peer, then the link makes the transition to the down state. ECHOREQ packets are sent out only when LCP negotiation is complete (for example, when LCP is open).

**Note** The **keepalive** command applies to POS interfaces using HDLC or PPP encapsulation. It does not apply to POS interfaces using Frame Relay encapsulation.

Use the **keepalive** command in interface configuration mode to set the frequency at which LCP sends ECHOREQ packets to its peer. To restore the system to the default keepalive interval of 10 seconds, use the **keepalive** command with **no** argument. To disable keepalives, use the **keepalive disable** command. For both PPP and Cisco HDLC, a keepalive of 0 disables keepalives and is reported in the **show running-config** command output as **keepalive disable**.

To remove the **keepalive** command from the configuration entirely, use the **no keepalive** command. You must remove the **keepalive** command from an interface configuration before you can configure Frame Relay encapsulation on that interface. Frame Relay interfaces do not support keepalives.

**Note** During MDR, the keepalive interval must be 10 seconds or more.

When LCP is running on the peer and receives an ECHOREQ packet, it responds with an echo reply (ECHOREP) packet, regardless of whether keepalives are enabled on the peer.

Keepalives are independent between the two peers. One peer end can have keepalives enabled while the other end has them disabled. Even if keepalives are disabled locally, LCP still responds with ECHOREP packets to the ECHOREQ packets it receives. Similarly, LCP also works if the period of keepalives at each end is different.

**Note** Use the **debug chdlc slarp packet** command and other Cisco HDLC **debug** commands to display information about the Serial Line Address Resolution Protocol (SLARP) packets that are sent to the peer after the keepalive timer has been configured.

## Frame Relay Encapsulation

On the Cisco XR 12000 Series Router, Frame Relay encapsulated POS interface configuration is hierarchical and comprises the following elements:

1. The POS main interface is comprised of the physical interface and port. If you are not using the POS interface to support Cisco HDLC and PPP encapsulated connections, then you must configure subinterfaces with PVCs under the POS main interface. Frame Relay connections are supported on PVCs only.
2. POS subinterfaces are configured under the POS main interface. A POS subinterface does not actively carry traffic until you configure a PVC under the POS subinterface.
3. Point-to-point and Layer 2 attachment circut (AC) PVCs are configured under a POS subinterface. You cannot configure a PVC directly under a main interface. A single point-to-point or L2 AC PVC is allowed per subinterface. PVCs use a predefined circuit path and fail if the path is interrupted. PVCs remain active until the circuit is removed. Connections on the POS PVC support Frame Relay encapsulation only.
4. Layer 3 configuration typically takes place on the subinterface.

**Note** The administrative state of a parent interface drives the state of the subinterface and its PVC. When the administrative state of a parent interface or subinterface changes, so does the administrative state of any child PVC configured under that parent interface or subinterface.

On the Cisco XR 12000 Series Router, the following hardware supports Frame Relay encapsulation:

- Cisco 1-Port 192c/STM-64c POS/SDH SPA
- Cisco 2-Port OC48/STM16 POS/SDH SPA
- Cisco 4-Port OC-3c/STM-1 POS/SDH Line Card
- Cisco 8-Port OC-3c/STM-1c POS/SDH Line Card
- Cisco 16-Port OC-3c/STM-1c POS/SDH Line Card
- Cisco 4-Port OC-12c/STM4 POS/SDH ISE Line Card
- Cisco 1-Port OC-48c/STM16c POS/SDH ISE Line Card

To configure Frame Relay encapsulation on POS interfaces, use the **encapsulation frame-relay** command.

Frame Relay interfaces support two types of encapsulated frames:

- Cisco (this is the default)
- IETF

Use the **encap** command in PVC configuration mode to configure Cisco or IETF encapsulation on a PVC. If the encapsulation type is not configured explicitly for a PVC, then that PVC inherits the encapsulation type from the main POS interface.

**Note** Cisco encapsulation is required on POS main interfaces that are configured for MPLS. IETF encapsulation is not supported for MPLS.

Before you configure Frame Relay encapsulation on an interface, you must verify that all prior Layer 3 configuration is removed from that interface. For example, you must ensure that there is no IP address configured directly under the main interface; otherwise, any Frame Relay configuration done under the main interface will not be viable.

## LMI on Frame Relay Interfaces

The Local Management Interface (LMI) protocol monitors the addition, deletion, and status of PVCs. LMI also verifies the integrity of the link that forms a Frame Relay UNI interface. By default, **cisco** LMI is enabled on all PVCs. However, you can modify the default LMI type to be ANSI or Q.933, as described in the "Modifying the Default Frame Relay Configuration on an Interface" module later in this manual.

If the LMI type is **cisco** (the default LMI type), the maximum number of PVCs that can be supported under a single interface is related to the MTU size of the main interface. Use the following formula to calculate the maximum number of PVCs supported on a card or SPA:

```
(MTU - 13)/8 = maximum number of PVCs
```

**Note** The default setting of the **mtu** command for a POS interface is 4474 bytes. Therefore, the default numbers of PVCs supported on a POS interface configured with **cisco** LMI is 557.

**Note** You must configure the LMI interface type on Frame Relay interfaces; otherwise, the POS interface does not come up. For connections between Provider Edge (PE) and Customer Edge (CE) routers, the PE end must be DCE and the CE end must be DTE for LMI to come up. For more information about configuring the LMI interface type on a Frame Relay interface, see the "Configuring Frame Relay Cisco IOS XR Software" module.

### Layer 2 Tunnel Protocol Version 3-Based Layer 2 VPN for Frame Relay

Layer 2 Tunnel Protocol Version 3 (L2TPv3) is a protocol used for tunneling Layer 2 payloads over an IP core network. L2TPv3 defines the signaling and formatting of packets for L2VPN on an IP Network.

Cisco IOS XR software supports a point-to-point, end-to-end service, where two attachment circuits (ACs) are connected together.

L2TPv3 connection setup requires the following tasks:

- Configuring an AC on each Provider Edge (PE) router
- Configuring an L2TPv3 encapsulated pseudowire between two PE routers.

This module describes how to configure a Layer 2 AC on a Frame Relay encapsulated POS interface.

For detailed information about configuring L2TPv3 pseudowires in your network, see the *Layer 2 Tunnel Protocol Version 3 on Cisco IOS XR Software* module of the *Cisco IOS XR MPLS Configuration Guide.*

For detailed information about configuring L2VPNs, see the "Implementing MPLS Layer 2 VPNs on Cisco IOS XR Software" module of the *Cisco IOS XR MPLS Configuration Guide.*

# How to Configure a POS Interface

This section contains the following procedures:


- Bringing Up a POS Interface, page 418
- Configuring Optional POS Interface Parameters, page 421
- Creating a Point-to-Point POS Subinterface with a PVC, page 424
- Configuring Optional PVC Parameters, page 426
- Modifying the Keepalive Interval on POS Interfaces, page 428
- Creating a Layer 2 Frame Relay Subinterface with a PVC, page 431


## Bringing Up a POS Interface

This task describes the commands you can use to bring up a POS interface.

### Prerequisites

You must have a POS line card or SPA installed in a router that is running Cisco IOS XR software.

## Restrictions

The configuration on both ends of the POS connection must match for the interface to be active.

### SUMMARY STEPS

1. **show interfaces**
2. **configure**
3. **interface pos** *interface-path-id*
4. **ipv4 address** *ipv4_address/prefix*
5. **no shutdown**
6. **end**
   or
   **commit**
7. **exit**
8. **exit**
9. Repeat Step 1 through Step 8 to bring up the interface at the other end of the connection.
10. **show ipv4 interface brief**
11. **show interfaces pos** *interface-path-id*

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| Step 1 | **`show interfaces`**<br><br>**Example:**<br>`RP/0/0/CPU0:router# show interfaces` | (Optional) Displays configured interfaces.<br>• Use this command to also confirm that the router recognizes the PLIM card. |
| Step 2 | **`configure`**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| Step 3 | **`interface pos`** *`interface-path-id`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface POS 0/3/0/0` | Specifies the POS interface name and notation *rack/slot/module/port,* and enters interface configuration mode. |
| Step 4 | **`ipv4 address`** *`ipv4_address/prefix`*<br><br>**Example:**<br>`RP/0/0/CPU0:router (config)#ipv4 address 10.46.8.6/24` | Assigns an IP address and subnet mask to the interface.<br>**Note** Skip this step if you are configuring Frame Relay encapsulation on this interface. For Frame Relay, the IP address and subnet mask are configured under the subinterface. |

| | Command or Action | Purpose |
|---|---|---|
| Step 5 | **no shutdown**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-if)# no shutdown` | Removes the shutdown configuration.<br><br>**Note** Removal of the shutdown configuration eliminates the forced administrative down on the interface, enabling it to move to an up or down state (assuming the parent SONET layer is not configured administratively down). |
| Step 6 | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-if)# end`<br>or<br>`RP/0/0/CPU0:router(config-if)# commit` | Saves configuration changes.<br><br>• When you issue the **end** command, the system prompts you to commit changes:<br><br>`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`<br><br>– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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><br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |
| Step 7 | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-if)# exit` | Exits interface configuration mode and enters global configuration mode. |
| Step 8 | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config)# exit` | Exits global configuration mode and enters EXEC mode. |

| | Command or Action | Purpose |
|---|---|---|
| Step 9 | **show interfaces**<br>**configure**<br>**interface pos** *interface-path-id*<br>**no shut**<br>**exit**<br>**exit**<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router# show interfaces`<br>`RP/0/0/CPU0:router# configure`<br>`RP/0/0/CPU0:router (config)# interface pos 0/3/0/0`<br>`RP/0/0/CPU0:router (config-if)# no shutdown`<br>`RP/0/0/CPU0:router (config-if)# commit`<br>`RP/0/0/CPU0:router (config-if)# exit`<br>`RP/0/0/CPU0:router (config)# exit` | Repeat Step 1 through Step 8 to bring up the interface at the other end of the connection.<br><br>**Note** The configuration on both ends of the POS connection must match. |
| Step 10 | **show ipv4 interface brief**<br><br>**Example:**<br>`RP/0/0/CPU0:router # show ipv4 interface brief` | Verifies that the interface is active and properly configured.<br><br>If you have brought up a POS interface properly, the "Status" field for that interface in the **show ipv4 interface brief** command output shows "Up." |
| Step 11 | **show interfaces pos** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router# show interfaces pos 0/3/0/0` | (Optional) Displays the interface configuration. |

## What to Do Next

To modify the default configuration of the POS interface you just brought up, see the "Configuring Optional POS Interface Parameters" section on page 421.

# Configuring Optional POS Interface Parameters

This task describes the commands you can use to modify the default configuration on a POS interface.

## Prerequisites

Before you modify the default POS interface configuration, you must bring up the POS interface and remove the shutdown configuration, as described in the "Bringing Up a POS Interface" section on page 418.

## Restrictions

The configuration on both ends of the POS connection must match for the interface to be active.

## SUMMARY STEPS

1. **configure**
2. **interface pos** *interface-path-id*
3. **encapsulation** [**hdlc** | **ppp** | **frame-relay** [**IETF**]]
4. **pos crc** {**16** | **32**}
5. **mtu** *value*
6. **end**
   or
   **commit**
7. **exit**
8. **exit**
9. **show interfaces pos** [*interface-path-id*]

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **interface pos** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface POS 0/3/0/0` | Specifies the POS interface name and notation *rack/slot/module/port,* and enters interface configuration mode. |
| **Step 3** | **encapsulation** [**hdlc** \| **ppp** \| **frame-relay** [**IETF**]]<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# encapsulation hdlc` | (Optional) Configures the interface encapsulation parameters and details such as HDLC or PPP.<br><br>**Note** The default encapsulation is **hdlc**. |
| **Step 4** | **pos crc** {**16** \| **32**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# pos crc 32` | (Optional) Configures the CRC value for the interface. Enter the **16** keyword to specify 16-bit CRC mode, or enter the **32** keyword to specify 32-bit CRC mode.<br><br>**Note** The default CRC is **32**. |
| **Step 5** | **mtu** *value*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# mtu 4474` | (Optional) Configures the MTU value.<br>• The default value is 4474.<br>• The POS MTU range is 64–9216. |

| | Command or Action | Purpose |
|---|---|---|
| Step 6 | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-if)# end`<br>or<br>`RP/0/0/CPU0:router(config-if)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 7 | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-if)# exit` | Exits interface configuration mode and enters global configuration mode. |
| Step 8 | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config)# exit` | Exits global configuration mode and enters EXEC mode. |
| Step 9 | **show interfaces pos** [*interface-path-id*]<br><br>**Example:**<br>`RP/0/0/CPU0:router# show interface pos 0/3/0/0` | (Optional) Displays general information for the specified POS interface. |

## What to Do Next

- To create a point-to-point Frame Relay subinterface with a PVC on the POS interface you just brought up, see the "Creating a Point-to-Point POS Subinterface with a PVC" section on page 424.
- To configure PPP authentication on POS interfaces where PPP encapsulation is enabled, see the *Configuring PPP on Cisco IOS XR Software* module later in this manual.
- To modify the keepalive interval on POS interfaces that have Cisco HDLC or PPP encapsulation enabled, see the "Modifying the Keepalive Interval on POS Interfaces" section on page 428.
- To modify the default Frame Relay configuration on POS interfaces that have Frame Relay encapsulation enabled, see the "Modifying the Default Frame Relay Configuration on an Interface" of the *Configuring Frame Relay Cisco IOS XR Software* module in this manual.

# Creating a Point-to-Point POS Subinterface with a PVC

The procedure in this section creates a point-to-point POS subinterface and configures a permanent virtual circuit (PVC) on that POS subinterface.

**Note** Subinterface and PVC creation is supported on interfaces with Frame Relay encapsulation only.

## Prerequisites

Before you can create a subinterface on a POS interface, you must bring up the main POS interface with Frame Relay encapsulation, as described in the "Bringing Up a POS Interface" section on page 418.

## Restrictions

Only one PVC can be configured for each point-to-point POS subinterface.

### SUMMARY STEPS

1. **configure**
2. **interface pos** *interface-path-id.subinterface* **point-to-point**
3. **ipv4 address** *ipv4_address/prefix*
4. **pvc** *dlci*
5. **end**
   or
   **commit**
6. Repeat Step 1 through Step 5 to bring up the POS subinterface and any associated PVC at the other end of the connection.

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | `configure`<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **`interface pos`** *`interface-path-id.subinterface`* **`point-to-point`**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config)# interface pos 0/3/0/0.1 point-to-point` | Enters POS subinterface configuration mode.<br><br>Replace *subinterface* with a subinterface ID, in the range from 1 through 4294967295. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **ipv4 address** *ipv4_address/prefix*<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-subif)#ipv4 address 10.46.8.6/24` | Assigns an IP address and subnet mask to the subinterface. |
| **Step 4** | **pvc** *dlci*<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-subif)# pvc 20` | Creates a POS permanent virtual circuit (PVC) and enters Frame Relay PVC configuration submode.<br><br>Replace *dlci* with a PVC identifier, in the range from 16 to 1007.<br><br>**Note** Only one PVC is allowed per subinterface. |
| **Step 5** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-fr-vc)# end`<br>or<br>`RP/0/0/CPU0:router(config-fr-vc)# commit` | Saves configuration changes.<br><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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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><br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |
| **Step 6** | **configure**<br>**interface pos** *interface-path-id.subinterface*<br>**pvc** *dlci*<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure`<br>`RP/0/0/CPU0:router (config)# interface pos 0/3/0/1.1`<br>`RP/0/0/CPU0:router (config-subif)#ipv4 address 10.46.8.5/24`<br>`RP/0/0/CPU0:router (config-subif)# pvc 20`<br>`RP/0/0/CPU0:router (config-fr-vc)# commit` | Repeat Step 1 through Step 5 to bring up the POS subinterface and any associated PVC at the other end of the connection.<br><br>**Note** The DLCI (or PVC identifier) must match on both ends of the subinterface connection.<br><br>**Note** When assigning an IP address and subnet mask to the subinterface at the other end of the connection, keep in mind that the addresses at both ends of the connection must be in the same subnet. |

## What to Do Next

- To configure optional PVC parameters, see the "Configuring Optional PVC Parameters" section on page 426.
- To modify the default Frame Relay configuration on POS interfaces that have Frame Relay encapsulation enabled, see the "Modifying the Default Frame Relay Configuration on an Interface" of the *"Configuring Frame Relay Cisco IOS XR Software"* module.
- To attach a Layer 3 QOS service policy to the PVC under the PVC submode, refer to the appropriate Cisco IOS XR software configuration guide.

# Configuring Optional PVC Parameters

This task describes the commands you can use to modify the default configuration on a POS PVC.

## Prerequisites

Before you can modify the default PVC configuration, you must create the PVC on a POS subinterface, as described in the "Creating a Point-to-Point POS Subinterface with a PVC" section on page 424.

## Restrictions

- The DLCI (or PVC identifier) must match on both ends of the PVC for the connection to be active.
- To change the PVC DLCI, you must delete the PVC and then add it back with the new DLCI.

### SUMMARY STEPS

1. **configure**
2. **interface pos** *interface-path-id.subinterface*
3. **pvc** *dlci*
4. **encap** [**cisco** | **ietf**]
5. **service-policy** {**input** | **output**} *policy-map*
6. **end**
   or
   **commit**
7. Repeat Step 1 through Step 6 to configure the PVC at the other end of the connection.
8. **show frame-relay pvc** *dlci-number*
9. **show policy-map interface pos** *interface-path-id.subinterface* {**input** | **output**}
   or
   **show policy-map type qos interface pos** *interface-path-id.subinterface* {**input** | **output**}

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **interface pos** *interface-path-id.subinterface*<br><br>**Example:**<br>`RP/0/0/CPU0:router (config)# interface pos 0/3/0/0.1` | Enters POS subinterface configuration mode. |
| **Step 3** | **pvc** *dlci*<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-subif)# pvc 20` | Enters subinterface configuration mode for the PVC.<br><br>Replace *dlci* with the DLCI number used to identify the PVC. Range is from 16 to 1007. |
| **Step 4** | **encap** [**cisco** \| **ietf**]<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-fr-vc)# encap ietf` | (Optional) Configures the encapsulation for a Frame Relay PVC.<br><br>**Note** If the encapsulation type is not configured explicitly for a PVC, then that PVC inherits the encapsulation type from the main POS interface. |
| **Step 5** | **service-policy** {**input** \| **output**} *policy-map*<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-fr-vc)# service-policy output policy1` | Attaches a policy map to an input subinterface or output subinterface. Once attached, the policy map is used as the service policy for the subinterface.<br><br>**Note** For information on creating and configuring policy maps, refer to the *Cisco IOS XR Modular Quality of Service Configuration Guide*, |
| **Step 6** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-fr-vc)# end`<br>or<br>`RP/0/0/CPU0:router(config-fr-vc)# commit` | Saves configuration changes.<br><br>• When you issue the **end** command, the system prompts you to commit changes:<br><br>`Uncommitted changes found, commit them before exiting(yes/no/cancel)?`<br>`[cancel]:`<br><br>– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br><br>– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br><br>– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br><br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 7** | **configure**<br>**interface pos** *interface-path-id.subinterface*<br>**pvc** *dlci*<br>**encap** [**cisco** \| **ietf**]<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure`<br>`RP/0/0/CPU0:router (config)# interface pos 0/3/0/1.1`<br>`RP/0/0/CPU0:router (config-subif)# pvc 20`<br>`RP/0/0/CPU0:router (config-fr-vc)# encap cisco`<br>`RP/0/0/CPU0:router (config-fr-vc)# commit` | Repeat Step 1 through Step 6 to bring up the POS subinterface and any associated PVC at the other end of the connection.<br><br>**Note** The configuration on both ends of the subinterface connection must match. |
| **Step 8** | **show frame-relay pvc** *dlci-number*<br><br>**Example:**<br>`RP/0/0/CPU0:router# show frame-relay pvc 20` | (Optional) Verifies the configuration of specified POS interface. |
| **Step 9** | **show policy-map interface pos** *interface-path-id.subinterface* {**input** \| **output**}<br>or<br>**show policy-map type qos interface pos** *interface-path-id.subinterface* {**input** \| **output**}<br><br>**Example:**<br>`RP/0/0/CPU0:router# show policy-map interface pos 0/3/0/0.1 output`<br>or<br>`RP/0/0/CPU0:router# show policy-map type qos interface pos 0/3/0/0.1 output` | (Optional) Displays the statistics and the configurations of the input and output policies that are attached to a subinterface. |

## What to Do Next

To modify the default Frame Relay configuration on POS interfaces that have Frame Relay encapsulation enabled, see the "Modifying the Default Frame Relay Configuration on an Interface" of the *"Configuring Frame Relay Cisco IOS XR Software"* module.

# Modifying the Keepalive Interval on POS Interfaces

Perform this task to modify the keepalive interval on POS interfaces that have Cisco HDLC or PPP encapsulation enabled.

**Note** When you enable Cisco HDLC or PPP encapsulation on a POS interface, the default keepalive interval is 10 seconds. Use this procedure to modify that default keepalive interval.

**Note** Cisco HDLC is enabled by default on POS interfaces.

## Prerequisites

Before you can modify the keepalive timer configuration, you must ensure that Cisco HDLC or PPP encapsulation is enabled on the interface. Use the **encapsulation** command to enable Cisco HDLC or PPP encapsulation on the interface, as described in the "Configuring Optional POS Interface Parameters" section on page 421.

## Restrictions

During MDR, the keepalive interval must be 10 seconds or more.

### SUMMARY STEPS

1. **configure**
2. **interface pos** *interface-path-id*
3. **keepalive** {*seconds* [*retry-count*] | **disable**}
   or
   **no keepalive**
4. **end**
   or
   **commit**
5. **show interfaces** *type interface-path-id*

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| Step 1 | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| Step 2 | **interface pos** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface POS 0/3/0/0` | Specifies the POS interface name and notation *rack/slot/module/port* and enters interface configuration mode. |
| Step 3 | **keepalive** {*seconds* [*retry-count*] \| **disable**}<br>or<br>**no keepalive**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# keepalive 3`<br>or<br>`RP/0/0/CPU0:router(config-if)# no keepalive` | Specifies the number of seconds between keepalive messages, and optionally the number of keepalive messages that can be sent to a peer without a response before transitioning the link to the down state.<br>• Use the **keepalive disable** command, the **no keepalive**, or the **keepalive** command with an argument of 0 to disable the keepalive feature entirely.<br>• If keepalives are configured on an interface, use the **no keepalive** command to disable the keepalive feature before you configure Frame Relay encapsulation on that interface. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | `end`<br>or<br>`commit`<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# end`<br>or<br>`RP/0/0/CPU0:router(config-if)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 interfaces pos** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router# show interfaces POS 0/3/0/0` | (Optional) Verifies the interface configuration. |

# How to Configure a Layer 2 Attachment Circuit

The Layer 2 AC configuration tasks are described in the following procedures:

- Creating a Layer 2 Frame Relay Subinterface with a PVC
- Configuring Optional Layer 2 PVC Parameters

**Note** After you configure an interface for Layer 2 switching, no routing commands such as **ipv4 address** are permissible.

**Note** Layer 2 ACs are not supported on interfaces configured with HDLC or PPP encapsulation.

# Creating a Layer 2 Frame Relay Subinterface with a PVC

The procedure in this section creates a Layer 2 Frame Relay subinterface with a PVC.

## Prerequisites

Before you can create a subinterface on a POS interface, you must bring up a POS interface, as described in the "Bringing Up a POS Interface" section on page 418.

**Note** You must skip Step 4 of the "Bringing Up a POS Interface" configuration steps when configuring an interface for Layer 2 switching. The **ipv4 address** command is not permissible on Frame Relay encapsulated interface.

## Restrictions

- Only one PVC can be configured for each subinterface.
- The configuration on both ends of the PVC must match for the connection to operate properly.
- The **ipv4 address** command is not permissible on Frame Relay encapsulated interface. Any previous configuration of an IP address must be removed before you can configure an interface for Layer 2 transport mode.
- Layer 2 configuration is supported on Frame Relay PVCs only. Layer 2 Port mode, where Layer 2 configuration is applied directly under the main POS interface, is not supported.

### SUMMARY STEPS

1. **configure**
2. **interface pos** *interface-path-id.subinterface* **l2transport**
3. **pvc** *dlci*
4. **end**
   or
   **commit**
5. Repeat Step 1 through Step 4 to bring up the subinterface and any associated PVC at the other end of the AC.

## DETAILED STEPS

<table>
<tr><th></th><th>Command or Action</th><th>Purpose</th></tr>
<tr><td>Step 1</td><td><code>configure</code><br><br>Example:<br><code>RP/0/0/CPU0:router# configure</code></td><td>Enters global configuration mode.</td></tr>
<tr><td>Step 2</td><td><code>interface pos</code> <code>interface-path-id.subinterface</code> <code>l2transport</code><br><br>Example:<br><code>RP/0/0/CPU0:router(config)# interface pos 0/3/0/0.1 l2transport</code></td><td>Creates a subinterface and enters POS subinterface configuration mode for that subinterface.<br><br>Note The subinterface must be unique to any other subinterfaces configured under a single main interface.</td></tr>
<tr><td>Step 3</td><td><code>pvc</code> <code>dlci</code><br><br>Example:<br><code>RP/0/0/CPU0:router(config-if)# pvc 100</code></td><td>Creates a Frame Relay permanent virtual circuit (PVC) and enters Layer 2 transport PVC configuration mode.<br><br>Replace dlci with the DLCI number used to identify the PVC. Range is from 16 to 1007.<br><br>Note Only one PVC is allowed per subinterface.</td></tr>
<tr><td>Step 4</td><td><code>end</code><br>or<br><code>commit</code><br><br>Example:<br><code>RP/0/0/CPU0:router(config-fr-vc)# end</code><br>or<br><code>RP/0/0/CPU0:router(config-fr-vc)# commit</code></td><td>Saves configuration changes.<ul><li>When you issue the end command, the system prompts you to commit changes:<br><code>Uncommitted changes found, commit them before exiting(yes/no/cancel)?<br>[cancel]:</code><ul><li>Entering yes saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.</li><li>Entering no exits the configuration session and returns the router to EXEC mode without committing the configuration changes.</li><li>Entering cancel leaves the router in the current configuration session without exiting or committing the configuration changes.</li></ul></li><li>Use the commit command to save the configuration changes to the running configuration file and remain within the configuration session.</li></ul></td></tr>
<tr><td>Step 5</td><td>Repeat Step 1 through Step 4 to bring up the subinterface and any associated PVC at the other end of the AC.</td><td>Brings up the AC.<br><br>Note The configuration on both ends of the AC must match.</td></tr>
</table>

## What to Do Next

- To configure optional subinterface parameters, see the "Configuring Optional Layer 2 Subinterface Parameters" section on page 435.

- To configure optional PVC parameters, see the "Configuring Optional Layer 2 PVC Parameters" section on page 433.
- To configure a point-to-point pseudowire XConnect on the AC you just created, see the "*Implementing Layer 2 Tunnel Protocol Version 3 on Cisco IOS XR Software*" module of the *Cisco IOS XR Virtual Private Network Configuration Guide for the Cisco XR 12000 Series Router.*
- To configure an L2VPN, see the "*Implementing MPLS Layer 2 VPNs on Cisco IOS XR Software*" module of the *Cisco IOS XR Virtual Private Network Configuration Guide for the Cisco XR 12000 Series Router.*

# Configuring Optional Layer 2 PVC Parameters

This task describes the commands you can use to modify the default configuration on a Frame Relay Layer 2 PVC.

## Prerequisites

You must create the PVC on a Layer 2 subinterface, as described in the "Creating a Layer 2 Frame Relay Subinterface with a PVC" section on page 431.

### SUMMARY STEPS

1. **configure**
2. **interface pos** *interface-path-id.subinterface* **l2transport**
3. **pvc** *dlci*
4. **encap** [**cisco** | **ietf**]
5. **service-policy** {**input** | **output**} *policy-map*
6. **end**
   or
   **commit**
7. Repeat Step 1 through Step 5 to configure the PVC at the other end of the AC.
8. **show policy-map interface pos** *interface-path-id.subinterface* {**input** | **output**}
   or
   **show policy-map type qos interface pos** *interface-path-id.subinterface* {**input** | **output**}

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **interface pos** *interface-path-id.subinterface* **l2transport**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface pos 0/6/0/1.10 l2transport` | Enters POS subinterface configuration mode for a Layer 2 Frame Relay subinterface. |
| **Step 3** | **pvc** *dlci*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# pvc 100` | Enters Frame Relay PVC configuration mode for the specified PVC.<br><br>Replace *dlci* with the DLCI number used to identify the PVC. Range is from 16 to 1007. |
| **Step 4** | **encap** {**cisco** \| **ietf**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-fr-vc)# encap ietf` | Configures the encapsulation for a Frame Relay PVC.<br><br>The encapsulation type must match on both ends of the PVC. |
| **Step 5** | **service-policy** {**input** \| **output**} *policy-map*<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-fr-vc)# service-policy output policy1` | Attaches a policy map to an input subinterface or output subinterface. Once attached, the policy map is used as the service policy for the subinterface.<br><br>**Note** For information on creating and configuring policy maps, refer to the *Cisco IOS XR Modular Quality of Service Configuration Guide*, |
| **Step 6** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-pos-l2transport-pvc)# end`<br>or<br>`RP/0/0/CPU0:router(config-pos-l2transport-pvc)# commit` | Saves configuration changes.<br><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><br>– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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><br>Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 7** | Repeat Step 1 through Step 5 to configure the PVC at the other end of the AC. | Brings up the AC.<br>**Note** The configuration on both ends of the connection must match. |
| **Step 8** | **show policy-map interface pos** *interface-path-id.subinterface* {**input** \| **output**}<br>or<br>**show policy-map type qos interface pos** *interface-path-id.subinterface* {**input** \| **output**}<br><br>**Example:**<br>`RP/0/0/CPU0:router# show policy-map interface pos 0/6/0/1.10 output`<br>or<br>`RP/0/0/CPU0:router# show policy-map type qos interface pos 0/6/0/1.10 output` | (Optional) Displays the statistics and the configurations of the input and output policies that are attached to a subinterface. |

## What to Do Next

- To configure a point-to-point pseudowire XConnect on the AC you just created, see the "*Implementing Layer 2 Tunnel Protocol Version 3 on Cisco IOS XR Software"* module of the *Cisco IOS XR Virtual Private Network Configuration Guide for the Cisco XR 12000 Series Router.*
- To configure an L2VPN, see the "*Implementing MPLS Layer 2 VPNs on Cisco IOS XR Software"* module of the *Cisco IOS XR Virtual Private Network Configuration Guide for the Cisco XR 12000 Series Router.*

# Configuring Optional Layer 2 Subinterface Parameters

This task describes the commands you can use to modify the default configuration on a Frame Relay Layer 2 subinterface.

## Prerequisites

Before you can modify the default PVC configuration, you must create the PVC on a Layer 2 subinterface, as described in the "Creating a Layer 2 Frame Relay Subinterface with a PVC" section on page 431.

## Restrictions

In most cases, the MTU that is configured under the subinterface has priority over the MTU that is configured under the main interface. The exception to this rule is when the subinterface MTU is higher than main interface MTU. In such cases, the subinterface MTU displays the configured value in the CLI output, but the actual operational MTU is the value that is configured under the main interface value. To avoid confusion when troubleshooting and optimizing your Layer 2 connections, we recommend always configuring a higher MTU on main interface.

## SUMMARY STEPS

1. **configure**
2. **interface pos** *interface-path-id.subinterface*
3. **mtu** *value*
4. **end**
   or
   **commit**
5. Repeat Step 1 through Step 4 to configure the subinterface at the other end of the AC.

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **`configure`**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **`interface pos`** *`interface-path-id.subinterface`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface pos 0/3/0/1.1` | Enters POS subinterface configuration mode for a Layer 2 Frame Relay subinterface. |
| **Step 3** | **`mtu`** *`value`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# mtu 5000` | (Optional) Configures the MTU value. Range is from 64 through 65535. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-pos-l2transport-pvc)# end`<br>or<br>`RP/0/0/CPU0:router(config-pos-l2transport-pvc)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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** | Repeat Step 1 through Step 4 to configure the PVC at the other end of the AC. | Brings up the AC.<br>**Note** The configuration on both ends of the connection must match. |

# Configuration Examples for POS Interfaces

This section provides the following configuration examples:


- Bringing Up and Configuring a POS Interface with Cisco HDLC Encapsulation: Example, page 437
- Configuring a POS Interface with Frame Relay Encapsulation: Example, page 438
- Configuring a POS Interface with PPP Encapsulation: Example, page 439


## Bringing Up and Configuring a POS Interface with Cisco HDLC Encapsulation: Example

The following example shows how to bring up a basic POS interface with Cisco HDLC encapsulation:

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# interface POS 0/3/0/0
RP/0/0/CPU0:router(config-if)# ipv4 address 172.18.189.38 255.255.255.224
RP/0/0/CPU0:router(config-if)# no shutdown
RP/0/0/CPU0:router(config-if)# end
Uncommitted changes found, commit them? [yes]: yes
```

The following example shows how to configure the interval between keepalive messages to be 10 seconds:

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# interface POS 0/3/0/0
RP/0/0/CPU0:router(config-if)# keepalive 10
RP/0/0/CPU0:router(config-if)# commit
```

## Configuring a POS Interface with Frame Relay Encapsulation: Example

The following example shows how to create a POS interface with Frame Relay encapsulation and a point-to-point POS subinterface with a PVC on router 1:

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# interface POS 0/3/0/0
RP/0/0/CPU0:router(config-if)# encapsulation frame-relay
RP/0/0/CPU0:router(config-if)# no shutdown
RP/0/0/CPU0:router(config-if)# end
Uncommitted changes found, commit them? [yes]: yes

RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router (config)# interface pos 0/3/0/0.1 point-to-point
RP/0/0/CPU0:router (config-subif)#ipv4 address 10.20.3.1/24
RP/0/0/CPU0:router (config-subif)# pvc 100
RP/0/0/CPU0:router(config-if)# end
Uncommitted changes found, commit them? [yes]: yes

RP/0/0/CPU0:router# show interface POS 0/3/0/0

Wed Oct  8 04:20:30.248 PST DST
POS0/3/0/0 is up, line protocol is up
  Interface state transitions: 1
  Hardware is Packet over SONET/SDH
  Internet address is 10.20.3.1/24
  MTU 4474 bytes, BW 155520 Kbit
     reliability 255/255, txload 0/255, rxload 0/255
  Encapsulation FRAME-RELAY, crc 32, controller loopback not set,
  LMI enq sent  116, LMI stat recvd 76, LMI upd recvd 0, DTE LMI up
  LMI enq recvd 0, LMI stat sent  0, LMI upd sent  0
  LMI DLCI 1023  LMI type is CISCO  frame relay DTE
  Last clearing of "show interface" counters 00:00:06
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     1 packets input, 13 bytes, 0 total input drops
     0 drops for unrecognized upper-level protocol
     Received 0 runts, 0 giants, 0 throttles, 0 parity
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     1 packets output, 13 bytes, 0 total output drops
     0 output errors, 0 underruns, 0 applique, 0 resets
     0 output buffer failures, 0 output buffers swapped out
```

The following example shows how to create a POS interface with Frame Relay encapsulation and a point-to-point POS subinterface with a PVC on router 2, which is connected to router 1:

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# interface POS 0/3/0/1
RP/0/0/CPU0:router(config-if)# encapsulation frame-relay
RP/0/0/CPU0:router(config-if)# frame-relay intf-type dce
RP/0/0/CPU0:router(config-if)# no shutdown
RP/0/0/CPU0:router(config-if)# end
Uncommitted changes found, commit them? [yes]: yes
```

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router (config)# interface pos 0/3/0/1.1 point-to-point
RP/0/0/CPU0:router (config-subif)#ipv4 address 10.20.3.2/24
RP/0/0/CPU0:router (config-subif)# pvc 100
RP/0/0/CPU0:router(config-if)# end
Uncommitted changes found, commit them? [yes]: yes

RP/0/0/CPU0:router# show interface POS 0/3/0/1

Wed Oct  8 04:20:38.037 PST DST
POS0/3/0/1 is up, line protocol is up
  Interface state transitions: 1
  Hardware is Packet over SONET/SDH
  Internet address is 10.20.3.2/24
  MTU 4474 bytes, BW 155520 Kbit
     reliability 255/255, txload 0/255, rxload 0/255
  Encapsulation FRAME-RELAY, crc 32, controller loopback not set,
  LMI enq sent  0, LMI stat recvd 0, LMI upd recvd 0
  LMI enq recvd 77, LMI stat sent  77, LMI upd sent  0 , DCE LMI up
  LMI DLCI 1023  LMI type is CISCO  frame relay DCE
  Last clearing of "show interface" counters 00:00:14
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     2 packets input, 26 bytes, 0 total input drops
     0 drops for unrecognized upper-level protocol
     Received 0 runts, 0 giants, 0 throttles, 0 parity
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     2 packets output, 26 bytes, 0 total output drops
     0 output errors, 0 underruns, 0 applique, 0 resets
     0 output buffer failures, 0 output buffers swapped out
```

The following example shows how create a Layer 2 POS subinterface with a PVC on the main POS interface:

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router (config)# interface pos 0/3/0/0.1 l2transport
RP/0/0/CPU0:router (config-subif)# pvc 100
RP/0/0/CPU0:router(config-subif)# commit
```

# Configuring a POS Interface with PPP Encapsulation: Example

The following example shows how to create and configure a POS interface with PPP encapsulation:

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# interface POS 0/3/0/0
RP/0/0/CPU0:router(config-if)# ipv4 address 172.18.189.38 255.255.255.224
RP/0/0/CPU0:router(config-if)# encapsulation ppp
RP/0/0/CPU0:router(config-if)# no shutdown
RP/0/0/CPU0:router(config-if)# end
Uncommitted changes found, commit them? [yes]: yes

RP/0/0/CPU0:router# show interfaces POS 0/3/0/0

POS0/3/0/0 is down, line protocol is down
  Hardware is Packet over SONET
  Internet address is 172.18.189.38/27
  MTU 4474 bytes, BW 2488320 Kbit
     reliability 0/255, txload Unknown, rxload Unknown
  Encapsulation PPP, crc 32, controller loopback not set, keepalive set (
10 sec)
  LCP Closed
  Closed: IPCP
```

```
Last clearing of "show interface" counters never
5 minute input rate 0 bits/sec, 0 packets/sec
5 minute output rate 0 bits/sec, 0 packets/sec
   0 packets input, 0 bytes, 0 total input drops
   0 drops for unrecognized upper-level protocol
   Received 0 broadcast packets, 0 multicast packets
            0 runts, 0 giants, 0 throttles, 0 parity
   0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
   0 packets output, 0 bytes, 0 total output drops
   Output 0 broadcast packets, 0 multicast packets
   0 output errors, 0 underruns, 0 applique, 0 resets
   0 output buffer failures, 0 output buffers swapped out
   0 carrier transitions
```

# Additional References

These sections provide references related to POS interface configuration.

## Related Documents

| Related Topic | Document Title |
|---|---|
| Cisco IOS XR master command reference | *Cisco IOS XR Master Commands List* |
| Cisco IOS XR interface configuration commands | *Cisco IOS XR Interface and Hardware Component Command Reference* |
| Initial system bootup and configuration information for a router using the Cisco IOS XR software. | *Cisco IOS XR Getting Started Guide* |
| Cisco IOS XR AAA services configuration information | *Cisco IOS XR System Security Configuration Guide* and *Cisco IOS XR System Security Command Reference* |
| Information about user groups and task IDs | *Cisco IOS XR Interface and Hardware Component Command Reference* |

## Standards

| Standards | Title |
|---|---|
| FRF.1.2 | *PVC User-to-Network Interface (UNI) Implementation Agreement - July 2000* |
| ANSI T1.617 Annex D | — |
| ITU Q.933 Annex A | — |

## MIBs

| MIBs | MIBs Link |
|---|---|
| — | To locate and download MIBs using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: http://www.cisco.com/go/mibs |

## RFCs

| RFCs | Title |
|---|---|
| RFC 1294 | *Multiprotocol Interconnect Over Frame Relay* |
| RFC 1315 | *Management Information Base for Frame Relay DTEs* |
| RFC 1490 | *Multiprotocol Interconnect Over Frame Relay* |
| RFC 1586 | *Guidelines for Running OSPF Over Frame Relay Networks* |
| RFC 1604 | *Definitions of Managed Objects for Frame Relay Service* |
| RFC 2115 | *Management Information Base for Frame Relay DTEs Using SMIv2* |
| RFC 2390 | *Inverse Address Resolution Protocol* |
| RFC 2427 | *Multiprotocol Interconnect Over Frame Relay* |
| RFC 2954 | *Definitions of Managed Objects for Frame Relay Service* |

## 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/support |

# Configuring Serial Interfaces on  Cisco IOS XR Sotware

This module describes the configuration of serial interfaces on the Cisco XR 12000 Series Router.

Before you configure a serial interface, you must configure the clear channel T3/E3 controller or channelized T1/E1controller (DS0 channel) that is associated with that interface.

**Feature History for Configuring Serial Controller Interfaces**

| Release | Modification |
|---|---|
| Release 3.3.0 | This feature was introduced on the Cisco XR 12000 Series Router.<br>Support was added on the Cisco XR 12000 Series Router for the following hardware:<br>• Cisco XR 12000 SIP-401<br>• Cisco XR 12000 SIP-501<br>• Cisco XR 12000 SIP-601<br>Support was added on the Cisco XR 12000 Series Router for the following SPAs:<br>• Cisco 2-Port and 4-Port Channelized T3/DS0 SPA<br>• Cisco 2-Port and 4-Port T3/E3 Serial SPA |
| Release 3.4.0 | Support for the following features was introduced:<br>• Subinterfaces with permanent virtual circuits (PVCs)<br>• Frame Relay encapsulation on serial main interfaces and PVCs on the following hardware:<br>– Cisco 8-port Channelized T1/E1 SPA<br>– Cisco 2-Port and 4-Port Channelized T3/DS0 SPA<br>– Cisco 2-Port and 4-Port T3/E3 Serial SPA<br>– Cisco 1-Port Channelized OC-3 SPA<br>– Cisco 1-Port Channelized OC-12 SPA<br>– Cisco 1-Port Channelized OC-48 SPA<br>– Cisco 1-Port Channelized OC-12/STM-4 ISE Line Card |
| Release 3.4.1 | Multilink PPP was supported on serial interfaces on the Cisco XR 12000 Series Router. |

| | |
|---|---|
| Release 3.5.0 | Support was added on the Cisco XR 12000 Series Router for the following SPAs:<br>• Cisco 1-Port Channelized OC-12/DS0 SPA<br>• Cisco 1-Port Channelized OC-48/STM-16 SPA |
| Release 3.7.0 | Support was added on the Cisco XR 12000 Series Router for the 1-Port Channelized OC-48/DS3 Line Card. |
| Release 3.8.0 | Support was added on the Cisco XR 12000 Series Router for quality of service (QoS) on Layer 2 subinterfaces and the following line cards:<br>• Cisco 1-Port Channelized OC-12/DS0 Line Card<br>• Cisco 4-Port Channelized OC-12/DS3 Line Card |
| Release 4.0.0 | Support for fragmentation counters using the **fragment-counter** command was added for the following SPAs:<br>• Cisco 1-Port Channelized OC-3/STM-1 SPA<br>• Cisco 4-Port Channelized T3/DS0 SPA<br>• Cisco 8-Port Channelized T1/E1 SPA |

## Contents


- Prerequisites for Configuring Serial Interfaces, page 444
- Information About Configuring Serial Interfaces, page 445
- How to Configure Serial Interfaces, page 455
- Configuration Examples for Serial Interfaces, page 479
- Additional References, page 484


# Prerequisites for Configuring Serial Interfaces

Before configuring serial interfaces, ensure that the following tasks and conditions are met:

- You must be in a user group associated with a task group that includes the proper task IDs. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.
- Your hardware must support T3/E3 or T1/E1 controllers and serial interfaces.

  The following hardware supports T3/E3 controllers and serial interfaces:
  - Cisco 2-Port and 4-Port Clear Channel T3/E3 SPAs
  - Cisco 2-Port and 4-Port Channelized T3 SPAs
  - Cisco 4-Port Channelized OC-12/DS3 line cards
  - Cisco 1-Port Channelized OC-48/STM-16 SPA and line cards

  The following hardware supports T1/E1 controllers and DS0 channels:
  - Cisco 2-Port and 4-Port Channelized T3 SPAs
  - Cisco 4-Port Channelized OC-12/DS3 line cards
  - Cisco 1-Port Channelized OC-12/DS0 SPAs and line cards

- Cisco 1-Port Channelized OC-48/DS3 SPAs and line cards
- Cisco 1-Port Channelized OC3/STM-1 SPA
- Cisco 8-Port Channelized T1/E1 SPA

The following hardware supports serial interfaces:

- Cisco 2-Port and 4-Port Clear Channel T3/E3 SPAs
- Cisco 2-Port and 4-Port Channelized T3 SPAs
- Cisco 4-Port Channelized OC-12/DS3 line cards
- Cisco 1-Port Channelized OC-12/DS0 SPAs and line cards
- Cisco 1-Port Channelized OC-48/DS3 SPAs and line cards
- Cisco 1-Port Channelized OC3/STM-1 SPA
- Cisco 8-Port Channelized T1/E1 SPA

**Note** The Cisco 2-Port and 4-Port Channelized T3 SPAs can run in clear channel mode, or they can be channelized into 28 T1 or 21 E1 controllers.

- You should have configured the clear channel T3/E3 controller or channelized T3 to T1/E1 controller that is associated with the serial interface you want to configure, as described in the *"Configuring Clear Channel T3/E3 and Channelized T3 and T1/E1 Controllers on Cisco IOS XR Software"* module in this manual.

**Note** On channelized T3 to T1/E1 controllers, serial interfaces are automatically created when users configure individual DS0 channel groups on the T1/E1 controllers.

# Information About Configuring Serial Interfaces

To configure serial interfaces, study the following concepts:


- High-Level Overview: Serial Interface Configuration on Clear-Channel SPAs, page 446
- High-Level Overview: Serial Interface Configuration on Channelized SPAs, page 446
- Cisco HDLC Encapsulation, page 447
- PPP Encapsulation, page 448
- Keepalive Timer, page 449
- Frame Relay Encapsulation, page 450
- Layer 2 Tunnel Protocol Version 3-Based Layer 2 VPN on Frame Relay, page 451
- Default Settings for Serial Interface Configurations, page 452
- Serial Interface Naming Notation, page 452
- IPHC Overview, page 453


On the , a single serial interface carries data over a single interface using PPP, Cisco HDLC, or Frame Relay encapsulation.

## High-Level Overview: Serial Interface Configuration on Clear-Channel SPAs

Table 17 provides a high-level overview of the tasks required to configure a T3 serial interface on the Cisco 2-Port and 4-Port Clear Channel T3/E3 SPA.

*Table 17 Overview: Configuring a T3 Serial Interface on a Clear Channel SPA*

| Step | Task | Module | Section |
|---|---|---|---|
| 1. | Use the **hw-module subslot** command to set serial mode for the SPA to be T3, if necessary.<br>**Note** By default, the 2-Port and 4-Port Clear Channel T3/E3 SPA is set to run in T3 mode. | "Configuring Clear Channel T3/E3 and Channelized T3 and T1/E1 Controllers on Cisco IOS XR Software" | Setting the Card Type |
| 2. | Configure the T3 controller. | "Configuring Clear Channel T3/E3 and Channelized T3 and T1/E1 Controllers on Cisco IOS XR Software" | Setting the Card Type |
| 3. | Configure the serial interface that is associated with the T3 controller you configured in Step 2. | "Configuring Serial Interfaces on Cisco IOS XR Sotware" | "How to Configure Serial Interfaces" |

Table 18 provides a high-level overview of the tasks required to configure an E3 serial interface on a 2-Port and 4-Port Clear Channel T3/E3 SPA.

*Table 18 Overview: Configuring an E3 Serial Interface on a Clear Channel SPA*

| Step | Task | Module | Section |
|---|---|---|---|
| 1. | Use the **hw-module subslot** command to set serial mode for the SPA to be E3. | "Configuring Clear Channel T3/E3 and Channelized T3 and T1/E1 Controllers on Cisco IOS XR Software" | Setting the Card Type |
| 2. | Configure the E3 controller. | "Configuring Clear Channel T3/E3 and Channelized T3 and T1/E1 Controllers on Cisco IOS XR Software" | Setting the Card Type |
| 3. | Configure the serial interface that is associated with the E3 controller you configured in Step 2. | Configuring Serial Interfaces on Cisco IOS XR Sotware | *How to Configure Serial Interfaces* |

## High-Level Overview: Serial Interface Configuration on Channelized SPAs

Table 19 provides a high-level overview of the tasks required to configure a T1 serial interface on the following SPAs and line cards.

- Cisco 2-Port and 4-Port Channelized T3 SPA
- Cisco 4-Port Channelized OC-12/DS3 line cards
- Cisco 1-Port Channelized OC-12/DS0 SPAs and line cards
- Cisco 1-Port Channelized OC-48/STM-16 SPA and line cards

*Table 19 Overview: Configuring a Serial Interface on a T1 DS0 Channel*

| Step | Task | Module | Section |
|---|---|---|---|
| **1.** | Configure the T3 controller parameters and set the SPA mode to be T3.<br>28 T1 controllers are automatically created. | "Configuring Clear Channel T3/E3 and Channelized T3 and T1/E1 Controllers on Cisco IOS XR Software" | Setting the Card Type<br>Configuring a Channelized T3 Controller |
| **2.** | Create and configure DS0 channel groups on the T1 controllers. | "Configuring Clear Channel T3/E3 and Channelized T3 and T1/E1 Controllers on Cisco IOS XR Software" | Configuring a T1 Controller |
| **3.** | Configure the Serial interfaces that are associated channel groups you created in Step 2. | "Configuring Serial Interfaces on Cisco IOS XR Sotware" | How to Configure Serial Interfaces |

Table 20 provides a high-level overview of the tasks required to configure an E1 serial interface on the following SPAs and line cards.

- 2-Port and 4-Port Channelized T3 SPA
- 4-Port Channelized OC-12/DS3 line cards
- 1-Port Channelized OC-12/DS0 SPAs and line cards
- 1-Port Channelized OC-48/DS3 SPAs and line cards

*Table 20 Overview: Configuring a Serial Interface on an E1 DS0 Channel*

| Step | Task | Module | Section |
|---|---|---|---|
| **1.** | Configure the T3 controller parameters and set the SPA mode to be E3.<br>21 E1 controllers are automatically created. | "Configuring Clear Channel T3/E3 and Channelized T3 and T1/E1 Controllers on Cisco IOS XR Software" | Configuring a Channelized T3 Controller |
| **2.** | Create and configure DS0 channel groups on the E1 controllers. | "Configuring Clear Channel T3/E3 and Channelized T3 and T1/E1 Controllers on Cisco IOS XR Software" | Configuring an E1 Controller |
| **3.** | Configure the Serial interfaces that are associated channel groups you created in Step 2. | Configuring Serial Interfaces on Cisco IOS XR Sotware | How to Configure Serial Interfaces |

# Cisco HDLC Encapsulation

*Cisco High-Level Data Link Controller* (HDLC) is the Cisco proprietary protocol for sending data over synchronous serial links using HDLC. Cisco HDLC also provides a simple control protocol called Serial Line Address Resolution Protocol (SLARP) to maintain serial link keepalives. HDLC is the default encapsulation type for serial interfaces under Cisco IOS XR software. Cisco HDLC is the default for data encapsulation at Layer 2 (data link) of the Open System Interconnection (OSI) stack for efficient packet delineation and error control.

**Note** Cisco HDLC is the default encapsulation type for the serial interfaces.

Cisco HDLC uses keepalives to monitor the link state, as described in the "Keepalive Timer" section on page 449.

**Note** Use the **debug chdlc slarp packet** command to display information about the Serial Line Address Resolution Protocol (SLARP) packets that are sent to the peer after the keepalive timer has been configured.

# PPP Encapsulation

PPP is a standard protocol used to send data over synchronous serial links. PPP also provides a Link Control Protocol (LCP) for negotiating properties of the link. LCP uses echo requests and responses to monitor the continuing availability of the link.

**Note** When an interface is configured with PPP encapsulation, a link is declared down, and full LCP negotiation is re-initiated after five ECHOREQ packets are sent without receiving an ECHOREP response.

PPP provides the following Network Control Protocols (NCPs) for negotiating properties of data protocols that will run on the link:

- IP Control Protocol (IPCP) to negotiate IP properties
- Multiprotocol Label Switching control processor (MPLSCP) to negotiate MPLS properties
- Cisco Discovery Protocol control processor (CDPCP) to negotiate CDP properties
- IPv6CP to negotiate IP Version 6 (IPv6) properties
- Open Systems Interconnection control processor (OSICP) to negotiate OSI properties

PPP uses keepalives to monitor the link state, as described in the "Keepalive Timer" section on page 449.

PPP supports the following authentication protocols, which require a remote device to prove its identity before allowing data traffic to flow over a connection:

- Challenge Handshake Authentication Protocol (CHAP)—CHAP authentication sends a challenge message to the remote device. The remote device encrypts the challenge value with a shared secret and returns the encrypted value and its name to the local router in a response message. The local router attempts to match the name of the remote device with an associated secret stored in the local username or remote security server database; it uses the stored secret to encrypt the original challenge and verify that the encrypted values match.
- Microsoft Challenge Handshake Authentication Protocol (MS-CHAP)—MS-CHAP is the Microsoft version of CHAP. Like the standard version of CHAP, MS-CHAP is used for PPP authentication; in this case, authentication occurs between a personal computer using Microsoft Windows NT or Microsoft Windows 95 and a Cisco router or access server acting as a network access server.

- Password Authentication Protocol (PAP)—PAP authentication requires the remote device to send a name and a password, which are checked against a matching entry in the local username database or in the remote security server database.

**Note** For more information on enabling and configuring PPP authentication protocols, see the *"Configuring PPP on Cisco IOS XR Software"* module in this manual.

Use the **ppp authentication** command in interface configuration mode to enable CHAP, MS-CHAP, and PAP on a serial interface.

**Note** Enabling or disabling PPP authentication does not effect the local router's willingness to authenticate itself to the remote device.

## Multilink PPP

Multilink Point-to-Point Protocol (MLPPP) is supported on the following SPAs:

- 1-Port Channelized OC-12/DS0 SPAs and line cards
- 2-Port and 4-Port Channelized T3 SPAs
- 8-Port Channelized T1/E1 SPA

MLPPP provides a method for combining multiple physical links into one logical link. The implementation of MLPPP combines multiple PPP serial interfaces into one multilink interface. MLPPP performs the fragmenting, reassembling, and sequencing of datagrams across multiple PPP links.

MLPPP provides the same features that are supported on PPP Serial interfaces with the exception of QoS. It also provides the following additional features:

- Fragment sizes of 128, 256, and 512 bytes
- Long sequence numbers (24-bit)
- Lost fragment detection timeout period of 80 ms
- Minimum-active-links configuration option
- LCP echo request/reply support over multilink interface
- Full T1 and E1 framed and unframed links

For more information about configuring MLPPP on a serial interface, see the *"Configuring PPP on Cisco IOS XR Software"* module in this document.

# Keepalive Timer

Cisco keepalives are useful for monitoring the link state. Periodic keepalives are sent to and received from the peer at a frequency determined by the value of the keepalive timer. If an acceptable keepalive response is not received from the peer, the link makes the transition to the down state. As soon as an acceptable keepalive response is obtained from the peer or if keepalives are disabled, the link makes the transition to the up state.

**Note** The **keepalive** command applies to serial interfaces using HDLC or PPP encapsulation. It does not apply to serial interfaces using Frame Relay encapsulation.

For each encapsulation type, a certain number of keepalives ignored by a peer triggers the serial interface to transition to the down state. For HDLC encapsulation, three ignored keepalives causes the interface to be brought down. For PPP encapsulation, five ignored keepalives causes the interface to be brought down. ECHOREQ packets are sent out only when LCP negotiation is complete (for example, when LCP is open).

**Note** Use the **keepalive** command in interface configuration mode to set the frequency at which LCP sends ECHOREQ packets to its peer. To restore the system to the default keepalive interval of 10 seconds, use the **keepalive** command with **no** argument. To disable keepalives, use the **keepalive disable** command. For both PPP and Cisco HDLC, a keepalive of 0 disables keepalives and is reported in the **show running-config** command output as **keepalive disable**.Before performing a Minimal Disruptive Restart (MDR) upgrade, we recommend disabling keepalives on a Cisco XR 12000 Series Router.

When LCP is running on the peer and receives an ECHOREQ packet, it responds with an echo reply (ECHOREP) packet, regardless of whether keepalives are enabled on the peer.

Keepalives are independent between the two peers. One peer end can have keepalives enabled; the other end can have them disabled. Even if keepalives are disabled locally, LCP still responds with ECHOREP packets to the ECHOREQ packets it receives. Similarly, LCP also works if the period of keepalives at each end is different.

**Note** Use the **debug chdlc slarp packet** command and other Cisco HDLC **debug** commands to display information about the Serial Line Address Resolution Protocol (SLARP) packets that are sent to the peer after the keepalive timer has been configured.

## Frame Relay Encapsulation

When Frame Relay encapsulation is enabled on a serial interface, the interface configuration is hierarchical and comprises the following elements:

1. The serial main interface comprises the physical interface and port. If you are not using the serial interface to support Cisco HDLC and PPP encapsulated connections, then you must configure subinterfaces with permanent virtual circuits (PVCs) under the serial main interface. Frame Relay connections are supported on PVCs only.
2. Serial subinterfaces are configured under the serial main interface. A serial subinterface does not actively carry traffic until you configure a PVC under the serial subinterface. Layer 3 configuration typically takes place on the subinterface.
3. Point-to-point PVCs are configured under a serial subinterface. You cannot configure a PVC directly under a main interface. A single point-to-point PVC is allowed per subinterface. PVCs use a predefined circuit path and fail if the path is interrupted. PVCs remain active until the circuit is removed from either configuration. Connections on the serial PVC support Frame Relay encapsulation only.

**Note** The administrative state of a parent interface drives the state of the subinterface and its PVC. When the administrative state of a parent interface or subinterface changes, so does the administrative state of any child PVC configured under that parent interface or subinterface.

To configure Frame Relay encapsulation on serial interfaces, use the **encapsulation frame-relay** command.

Frame Relay interfaces support two types of encapsulated frames:

- Cisco (default)
- IETF

Use the **encap** command in PVC configuration mode to configure Cisco or IETF encapsulation on a PVC. If the encapsulation type is not configured explicitly for a PVC, then that PVC inherits the encapsulation type from the main serial interface.

**Note** Cisco encapsulation is required on serial main interfaces that are configured for MPLS. IETF encapsulation is not supported for MPLS.

Before you configure Frame Relay encapsulation on an interface, you must verify that all prior Layer 3 configuration is removed from that interface. For example, you must ensure that there is no IP address configured directly under the main interface; otherwise, any Frame Relay configuration done under the main interface will not be viable.

## LMI on Frame Relay Interfaces

The Local Management Interface (LMI) protocol monitors the addition, deletion, and status of PVCs. LMI also verifies the integrity of the link that forms a Frame Relay UNI interface. By default, **cisco** LMI is enabled on all PVCs. However, you can modify the default LMI type to be ANSI or Q.933, as described in the *"Modifying the Default Frame Relay Configuration on an Interface"* section of the *"Configuring Frame Relay Cisco IOS XR Software"* module in this manual.

If the LMI type is **cisco** (the default LMI type), the maximum number of PVCs that can be supported under a single interface is related to the MTU size of the main interface. Use the following formula to calculate the maximum number of PVCs supported on a card or SPA:

```
(MTU - 13)/8 = maximum number of PVCs
```

**Note** The default setting of the **mtu** command for a serial interface is 1504 bytes. Therefore, the default numbers of PVCs supported on a serial interface configured with **cisco** LMI is 186.

# Layer 2 Tunnel Protocol Version 3-Based Layer 2 VPN on Frame Relay

The Layer 2 Tunnel Protocol Version 3 (L2TPv3) feature defines the L2TP protocol for tunneling Layer 2 payloads over an IP core network using Layer 2 virtual private networks (VPNs).

L2TPv3 is a tunneling protocol used for transporting Layer 2 protocols. It can operate in a number of different configurations and tunnel a number of different Layer 2 protocols and connections over a packet-switched network.

Before you can configure L2TPv3, you need to configure a connection between the two attachment circuits (ACs) that will host the L2TPv3 psuedowire. Cisco IOS XR software supports a point-to-point, end-to-end service, where two ACs are connected together.

This module describes how to configure a Layer 2 AC on a Frame Relay encapsulated serial interface.

**Note** Serial interfaces support DLCI mode layer 2 ACs only; layer 2 port mode ACs are not supported on serial interfaces.

For detailed information about configuring L2TPv3 in your network, see the "*Implementing Layer 2 Tunnel Protocol Version 3 on Cisco IOS XR Software*" module of the *Cisco IOS XR Virtual Private Network Configuration Guide for the Cisco XR 12000 Series Router.* For detailed information about configuring L2VPNs, see the "*Implementing MPLS Layer 2 VPNs on Cisco IOS XR Software*" module of the *Cisco IOS XR Virtual Private Network Configuration Guide for the Cisco XR 12000 Series Router.*

# Default Settings for Serial Interface Configurations

When an interface is enabled on a T3/E3 SPA, and no additional configuration commands are applied, the default interface settings shown in Table 21 are present. These default settings can be changed by configuration.

***Table 21 Serial Interface Default Settings***

| Parameter | Configuration File Entry | Default Settings |
|---|---|---|
| Keepalive<br>**Note** The **keepalive** command applies to serial interfaces using HDLC or PPP encapsulation. It does not apply to serial interfaces using Frame Relay encapsulation. | **keepalive** [**disable**]<br>**no keepalive** | keepalive 10 seconds |
| Encapsulation | **encapsulation** [**hdlc** \| **ppp** \| **frame-relay** [**ietf**]] | hdlc |
| Maximum transmission unit (MTU) | **mtu** *bytes* | 1504 bytes |
| Cyclic redundancy check (CRC) | **crc** [**16** \| **32**] | 16 |
| Data stream inversion on a serial interface | **invert** | Data stream is not inverted |
| Payload scrambling (encryption) | **scramble** | Scrambling is disabled. |
| Number of High-Level Data Link Control (HDLC) flag sequences to be inserted between the packets | **transmit-delay** | Default is 0 (disabled). |

**Note** Default settings do not appear in the output of the **show running-config** command.

# Serial Interface Naming Notation

The naming notation for serial interfaces on a clear channel SPA is *rack/slot/module/port*, as shown in the following example:

```
interface serial 0/0/1/2
```

The naming notation for T1, E1, and DS0 interfaces on a channelized SPA is *rack*/*slot*/*module*/*port*/*channel-num:channel-group-number*, as shown in the following example:

```
interface serial 0/0/1/2/4:3
```

If a subinterface and PVC are configured under the serial interface, then the router includes the subinterface number at the end of the serial interface address. In this case, the naming notation is *rack*/*slot*/*module*/*port*[/*channel-num:channel-group-number*].*subinterface,* as shown in the following examples:

```
interface serial 0/0/1/2.1
interface serial 0/0/1/2/4:3.1
```

**Note** A slash between values is required as part of the notation.

The naming notation syntax for serial interfaces is as follows:

- *rack*: Chassis number of the rack.
- *slot*: Physical slot number of the modular services card or line card.
- *module*: Module number. Shared port adapters (SPAs) are referenced by their subslot number.
- *port*: Physical port number of the controller.
- *channel-num*: T1 or E1 channel number. T1 channels range from 0 to 23; E1 channels range from 0 to 31.
- *channel-group-number*: Time slot number. T1 time slots range from 1 to 24; E1 time slots range from 1 to 31. The *channel-group-number* is preceded by a colon and not a slash.
- *subinterface*: Subinterface number.

Use the question mark (**?**) online help function following the **serial** keyword to view a list of all valid interface choices.

# IPHC Overview

IP header compression (IPHC) is based on the premise that most of the headers in the packets of a particular transmission remain constant throughout the flow. Only a few fields in the headers of related packets change during a flow.

IPHC compresses these headers so that the compressed header contains only the fields that change from packet to packet. All fields that remain the same from packet to packet are eliminated in the compressed headers. Full headers are sent between compressed headers.

Full headers are uncompressed headers that contain all the original header fields along with additional information (context ID) to identify the flow. The interval at which full headers are sent between compressed packets is configurable using the **refresh max-period** and command.

IPHC contexts are used by the compressor (sender) and decompressor (receiver) of compressed packets to encode and decode the packets in a flow. A context is stored on the compressor and decompressor and is used in the delta calculation at both ends. The number of contexts allowed on a particular interface is configurable. The maximum size of the header that can be compressed is also configurable.

IPHC supports the compression and decompression of RTP and UDP traffic and the decompression of CN on TCP and CTCP traffic.

Users may choose one of the following types of compression formats:

- Internet Engineering Task Force (IETF) standard format.
  Uses RFC2507 and RFC2508 compression schemes.
- IPHC format.
  Provides options similar to IETF.

Table 22 shows the IPHC features, the values of the features, and their defaults:

*Table 22 IPHC features and default setttings*

| IPHC Feature | Values | Defaults |
|---|---|---|
| TCP contexts | 0 to 255 | 1 |
| Non-TCP contexts | 1 to 6000 | 16 |
| Compression Format Options | IETF or IPHC | — |
| Feedback Messages | Enable or Disable | Enabled |
| Maximum Refresh Period Size | 1 to 65535 packets | 256 |
| Maximum Header Size | 20 to 40 bytes | 40 |
| Real Time Protocol (RTP) | Enable or Disable | Enabled |
| Refresh RTP | Enable or Disable | Disable |

Currently, only IPv4 unicast packets with UDP in the protocol field of the IP header are compressed.

IPHC is configured on an interface as follows:

- Configure the IPHC slot level command
- Create an IPHC profile
- Configure IPHC attributes in the profile
- Attach the profile to an interface

IPHC profiles must contain the **rtp** command to enable Real Time Protocol (RTP) on the interface, or the profile is not enabled. The **refresh rtp** command must be used to enable the configured refresh settings for RTP packets. By default, refresh RTP is disabled and only the first packet in the flow is sent as a 'full-header' packet.

If some attributes, such as feedback messages, maximum refresh period size and maximum header size, are not configured in the profile, the default values for those attributes apply when the profile is enabled on the interface.

Currently, IPHC is supported only on serial interfaces with PPP encapsulation and on multilink with PPP encapsulation interfaces.

IPHC is typically configured between the Customer Edge (CE) and Provide Edge (PE) ends of an interface and must be configured at both ends of the interface to work. The PPP protocol negotiates the IPHC specific parameters between the two ends of the interface and settles on the lowest value configured between the two ends.

### QoS and IPHC

An IPHC profile can be enabled on an interface so that the IPHC profile applies only to packets that match a Quality of Service (QoS) service policy. In this case, the QoS service-policy class attributes determine which packets are compressed. This allows users to fine tune IPHC with greater granularity.

Policy maps are attached to an interface using the **service-policy** command. IPHC action applies only to output service policies. IPHC is not supported on input service policies.

The user can configure IPHC using QoS as follows:

- Create a QoS **policy-map** with the **compress header ip** action.
- Attach the IPHC profile to the interface using the **ipv4 iphc profile** *profile_name* **mode service-policy** command.
- Attach the QoS **policy-map** with **compress header ip** action using the **service-policy output** command.

See "IPHC on a Serial Interface with MLPPP/LFI and QoS Configuration: Example" section on page 484 for an example of how to configure IPHC using QoS.

For complete information on configuring QoS, refer to the *Cisco XR 12000 Series Router Modular Quality of Service Configuration Guide* and the *Cisco XR 12000 Series Router Modular Quality of Service Command Reference*.

# How to Configure Serial Interfaces

After you have configured a channelized or clear channel T3/E3 controller, as described in the "Configuring Clear Channel T3/E3 and Channelized T3 and T1/E1 Controllers on Cisco IOS XR Software" module in this document, you can configure the serial interfaces associated with that controller.

The following tasks describe how to configure a serial interface:


- Bringing Up a Serial Interface, page 455
- Configuring Optional Serial Interface Parameters, page 458
- Creating a Point-to-Point Serial Subinterface with a PVC, page 461
- Configuring Optional PVC Parameters, page 464
- Modifying the Keepalive Interval on Serial Interfaces, page 466
- How to Configure a Layer 2 Attachment Circuit, page 468
  - Creating a Serial Layer 2 Subinterface with a PVC, page 468
  - Configuring Optional Serial Layer 2 PVC Parameters, page 470
- Configuring IPHC, page 473
  - Configuring the IPHC Slot Level Command, page 474
  - Configuring an IPHC Profile, page 475
  - Enabling an IPHC Profile on an Interface, page 478


## Bringing Up a Serial Interface

This task describes the commands used to bring up a serial interface.

## Prerequisites

The Cisco XR 12000 Series Router must have at least one of the SIPs and one of the SPAs or line cards installed and be running Cisco IOS XR software:

- Cisco XR 12000 SIP-401
- Cisco XR 12000 SIP-501
- Cisco XR 12000 SIP-601
- 2-Port and 4-Port T3/E3 Serial SPA
- 2-Port and 4-Port Channelized T3/DS0 Serial SPA
- 4-Port Channelized OC-12/DS3 line cards
- 1-Port Channelized OC-12/DS0 SPAs and line cards
- 1-Port Channelized OC-48/DS3 SPAs and line cards
- 1-Port Channelized OC3/STM-1 SPA
- 8-Port Channelized T1/E1 SPA

## Restrictions

The configuration on both ends of the serial connection must match for the interface to be active.

### SUMMARY STEPS

1. **show interfaces**
2. **configure**
3. **interface serial** *interface-path-id*
4. **ipv4 address** *ip-address*
5. **no shutdown**
6. **end**
   or
   **commit**
7. **exit**
8. **exit**
9. Repeat Step 1 through Step 8 to bring up the interface at the other end of the connection.
10. **show ipv4 interface brief**
11. **show interfaces serial** *interface-path-id*

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| Step 1 | **show interfaces**<br><br>**Example:**<br>`RP/0/0/CPU0:router# show interfaces` | (Optional) Displays configured interfaces.<br>• Use this command to also confirm that the router recognizes the PLIM card. |
| Step 2 | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| Step 3 | **interface serial** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface serial 0/1/0/0` | Specifies the serial interface name and notation *rack/slot/module/port,* and enters interface configuration mode. |
| Step 4 | **ipv4 address** *ip-address*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# ipv4 address 10.1.2.1 255.255.255.224` | Assigns an IP address and subnet mask to the interface.<br>**Note** Skip this step if you are configuring Frame Relay encapsulation on this interface. For Frame Relay, the IP address and subnet mask are configured under the subinterface. |
| Step 5 | **no shutdown**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-if)# no shutdown` | Removes the shutdown configuration.<br>**Note** Removal of the shutdown configuration eliminates the forced administrative down on the interface, enabling it to move to an up or down state (assuming the parent SONET layer is not configured administratively down). |
| Step 6 | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-if)# end`<br>or<br>`RP/0/0/CPU0:router(config-if)# 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)?`<br>`[cancel]:`<br>– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

| | Command or Action | Purpose |
|---|---|---|
| Step 7 | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-if)# exit` | Exits interface configuration mode and enters global configuration mode. |
| Step 8 | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config)# exit` | Exits global configuration mode and enters EXEC mode. |
| Step 9 | **show interfaces**<br>**configure**<br>**interface serial** *interface-path-id*<br>**no shut**<br>**exit**<br>**exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router# show interfaces`<br>`RP/0/0/CPU0:router# configure`<br>`RP/0/0/CPU0:router (config)# interface serial 0/1/0/1`<br>`RP/0/0/CPU0:router(config-if)# ipv4 address 10.1.2.2 255.255.255.224`<br>`RP/0/0/CPU0:router (config-if)# no shutdown`<br>`RP/0/0/CPU0:router (config-if)# commit`<br>`RP/0/0/CPU0:router (config-if)# exit`<br>`RP/0/0/CPU0:router (config)# exit` | Repeat Step 1 through Step 8 to bring up the interface at the other end of the connection.<br>**Note** The configuration on both ends of the serial connection must match. |
| Step 10 | **show ipv4 interface brief**<br><br>**Example:**<br>`RP/0/0/CPU0:router # show ipv4 interface brief` | Verifies that the interface is active and properly configured.<br>If you have brought up a serial interface properly, the "Status" field for that interface in the **show ipv4 interface brief** command output displays "Up." |
| Step 11 | **show interfaces serial** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router# show interfaces serial 0/1/0/0` | (Optional) Displays the interface configuration. |

## What to Do Next

To modify the default configuration of the serial interface you just brought up, see the "Configuring Optional Serial Interface Parameters" section on page 458.

# Configuring Optional Serial Interface Parameters

This task describes the commands used to modify the default configuration on a serial interface.

## Prerequisites

Before you modify the default serial interface configuration, you must bring up the serial interface and remove the shutdown configuration, as described in the "Bringing Up a Serial Interface" section on page 455.

## Restrictions

The configuration on both ends of the serial connection must match for the interface to be active.

### SUMMARY STEPS

1. **configure**
2. **interface serial** *interface-path-id*
3. **encapsulation** [**hdlc** | **ppp** | **frame-relay** [**IETF**]
4. **serial**
5. **crc** *length*
6. **invert**
7. **scramble**
8. **transmit-delay** *hdlc-flags*
9. **end**
   or
   **commit**
10. **exit**
11. **exit**
12. **exit**
13. **show interfaces serial** [*interface-path-id*]

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | `configure`<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **`interface serial`** *`interface-path-id`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface serial 0/1/0/0` | Specifies the serial interface name and notation *rack/slot/module/port,* and enters interface configuration mode. |
| **Step 3** | **`encapsulation`** [**`hdlc`** \| **`ppp`** \| **`frame-relay`** [**`IETF`**]<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# encapsulation hdlc` | (Optional) Configures the interface encapsulation parameters and details such as HDLC, PPP or Frame Relay.<br><br>**Note** The default encapsulation is **hdlc**. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **serial**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# serial` | (Optional) Enters serial submode to configure the serial parameters. |
| **Step 5** | **crc** *length*<br><br>**Example:**<br>`RP/0/0/CPU0:ios(config-if-serial)# crc 32` | (Optional) Specifies the length of the cyclic redundancy check (CRC) for the interface. Enter the **16** keyword to specify 16-bit CRC mode, or enter the **32** keyword to specify 32-bit CRC mode.<br><br>**Note** The default is CRC length is 16. |
| **Step 6** | **invert**<br><br>**Example:**<br>`RP/0/0/CPU0:ios(config-if-serial)# inverts` | (Optional) Inverts the data stream. |
| **Step 7** | **scramble**<br><br>**Example:**<br>`RP/0/0/CPU0:ios(config-if-serial)# scramble` | (Optional) Enables payload scrambling on the interface.<br><br>**Note** Payload scrambling is disabled on the interface. |
| **Step 8** | **transmit-delay** *hdlc-flags*<br><br>**Example:**<br>`RP/0/0/CPU0:ios(config-if-serial)# transmit-delay 10` | (Optional) Specifies a transmit delay on the interface. Values can be from 0 to 128.<br><br>**Note** Transmit delay is disabled by default (the transmit delay is set to **0**). |
| **Step 9** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-if)# end`<br>or<br>`RP/0/0/CPU0:router(config-if)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 10** | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if-serial)# exit` | Exits serial configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 11** | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-if)# exit` | Exits interface configuration mode and enters global configuration mode. |
| **Step 12** | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config)# exit` | Exits global configuration mode and enters EXEC mode. |
| **Step 13** | **show interfaces serial** [*interface-path-id*]<br><br>**Example:**<br>`RP/0/0/CPU0:router# show interface serial 0/1/0/0` | (Optional) Displays general information for the specified serial interface. |

## What to Do Next

- To create a point-to-point Frame Relay subinterface with a PVC on the serial interface you just brought up, see the "Creating a Point-to-Point Serial Subinterface with a PVC" section on page 461.
- To configure PPP authentication on serial interfaces with PPP encapsulation, see the *"Configuring PPP on Cisco IOS XR Software"* module later in this manual.
- To modify the default keepalive configuration, see the "Modifying the Keepalive Interval on Serial Interfaces" section on page 466.
- To modify the default Frame Relay configuration on serial interfaces that have Frame Relay encapsulation enabled, see the "Modifying the Default Frame Relay Configuration on an Interface" section of the *"Configuring Frame Relay Cisco IOS XR Software"* module.

# Creating a Point-to-Point Serial Subinterface with a PVC

The procedure in this section creates a point-to-point serial subinterface and configures a permanent virtual circuit (PVC) on that serial subinterface.

**Note** Subinterface and PVC creation is supported on interfaces with Frame Relay encapsulation only.

## Prerequisites

Before you can create a subinterface on a serial interface, you must bring up the main serial interface with Frame Relay encapsulation, as described in the "Bringing Up a Serial Interface" section on page 455.

## Restrictions

Only one PVC can be configured for each point-to-point serial subinterface.

## SUMMARY STEPS

1. **configure**
2. **interface serial** *interface-path-id.subinterface* **point-to-point**
3. **ipv4 address** *ipv4_address/prefix*
4. **pvc** *dlci*
5. **end**
   or
   **commit**
6. Repeat Step 1 through Step 5 to bring up the serial subinterface and any associated PVC at the other end of the connection.

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **`configure`**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **`interface serial`** *`interface-path-id.subinterface`* **`point-to-point`**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config)# interface serial 0/1/0/0.1` | Enters serial subinterface configuration mode. |
| **Step 3** | **`ipv4 address`** *`ipv4_address/prefix`*<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-subif)#ipv4 address 10.46.8.6/24` | Assigns an IP address and subnet mask to the subinterface. |
| **Step 4** | **`pvc`** *`dlci`*<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-subif)# pvc 20` | Creates a serial permanent virtual circuit (PVC) and enters Frame Relay PVC configuration submode.<br><br>Replace *dlci* with a PVC identifier, in the range from 16 to 1007.<br><br>**Note** Only one PVC is allowed per subinterface. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-subif)# end`<br>or<br>`RP/0/0/CPU0:router(config-subif)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 6** | **configure**<br>**interface serial** *interface-path-id*<br>**pvc** *dlci*<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure`<br>`RP/0/0/CPU0:router (config)# interface serial 0/1/0/1.1`<br>`RP/0/0/CPU0:router (config-subif)#ipv4 address 10.46.8.5/24`<br>`RP/0/0/CPU0:router (config-subif)# pvc 20`<br>`RP/0/0/CPU0:router (config-fr-vc)# commit` | Repeat Step 1 through Step 5 to bring up the serial subinterface and any associated PVC at the other end of the connection.<br>**Note** The DLCI (or PVC identifier) must match on both ends of the subinterface connection.<br>**Note** When assigning an IP address and subnet mask to the subinterface at the other end of the connection, keep in mind that the addresses at both ends of the connection must be in the same subnet. |

## What to Do Next

- To configure optional PVC parameters, see the "Configuring Optional Serial Interface Parameters" section on page 458.
- To modify the default Frame Relay configuration on serial interfaces that have Frame Relay encapsulation enabled, see the"Modifying the Default Frame Relay Configuration on an Interface" section of the *"Configuring Frame Relay Cisco IOS XR Software"* module.
- To attach a Layer 3 QOS service policy to the PVC under the PVC submode, refer to the appropriate Cisco IOS XR software configuration guide.

# Configuring Optional PVC Parameters

This task describes the commands you can use to modify the default configuration on a serial PVC.

For additional information about Frame Relay options, see the "Configuring Frame Relay on Cisco IOS XR Software" module in the *Cisco IOS XR Interface and Hardware Component Configuration Guide for the Cisco XR 12000 Series Router.*

## Prerequisites

Before you can modify the default PVC configuration, you must create the PVC on a serial subinterface, as described in the "Creating a Point-to-Point Serial Subinterface with a PVC" section on page 461.

## Restrictions

- The DLCI (or PVI identifier) must match on both ends of the PVC for the connection to be active.
- To change the PVC DLCI, you must delete the PVC and then add it back with the new DLCI.

### SUMMARY STEPS

1. **configure**
2. **interface serial** *interface-path-id.subinterface*
3. **pvc** *dlci*
4. **encap** [**cisco** | **ietf**]
5. **service-policy** {**input** | **output**} *policy-map*
6. **end**
   or
   **commit**
7. Repeat Step 1 through Step 6 to configure the PVC at the other end of the connection.
8. **show frame-relay pvc** *dlci-number*
9. **show policy-map interface serial** *interface-path-id.subinterface* {**input** | **output**}
   or
   **show policy-map type qos interface serial** *interface-path-id.subinterface* {**input** | **output**}

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | `configure`<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | `interface serial` *`interface-path-id.subinterface`*<br><br>**Example:**<br>`RP/0/0/CPU0:router (config)# interface serial 0/1/0/0.1` | Enters serial subinterface configuration mode. |

<table>
<tr><th></th><th>Command or Action</th><th>Purpose</th></tr>
<tr><td>Step 3</td><td>pvc dlci<br><br>Example:<br><code>RP/0/0/CPU0:router (config-subif)# pvc 20</code></td><td>Enters subinterface configuration mode for the PVC.</td></tr>
<tr><td>Step 4</td><td>encap [cisco | ietf]<br><br>Example:<br><code>RP/0/0/CPU0:router (config-fr-vc)# encap ietf</code></td><td>(Optional) Configures the encapsulation for a Frame Relay PVC.<br><br>Note If the encapsulation type is not configured explicitly for a PVC, then that PVC inherits the encapsulation type from the main serial interface.</td></tr>
<tr><td>Step 5</td><td>service-policy {input | output} policy-map<br><br>Example:<br><code>RP/0/0/CPU0:router (config-fr-vc)#<br>service-policy output policy1</code></td><td>Attaches a policy map to an input subinterface or output subinterface. Once attached, the policy map is used as the service policy for the subinterface.</td></tr>
<tr><td>Step 6</td><td>end<br>or<br>commit<br><br>Example:<br><code>RP/0/0/CPU0:router (config-fr-vc)# end</code><br>or<br><code>RP/0/0/CPU0:router(config-fr-vc)# commit</code></td><td>Saves configuration changes.<ul><li>When you issue the end command, the system prompts you to commit changes:<br><code>Uncommitted changes found, commit them before<br>exiting(yes/no/cancel)?<br>[cancel]:</code><ul><li>Entering yes saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.</li><li>Entering no exits the configuration session and returns the router to EXEC mode without committing the configuration changes.</li><li>Entering cancel leaves the router in the current configuration session without exiting or committing the configuration changes.</li></ul></li><li>Use the commit command to save the configuration changes to the running configuration file and remain within the configuration session.</li></ul></td></tr>
<tr><td>Step 7</td><td>configure<br>interface serial interface-path-id.subinterface<br>pvc dlci<br>encap [cisco | ietf]<br>commit<br><br>Example:<br><code>RP/0/0/CPU0:router# configure<br>RP/0/0/CPU0:router (config)# interface serial<br>0/1/0/1.1<br>RP/0/0/CPU0:router (config-subif)# pvc 20<br>RP/0/0/CPU0:router (config-fr-vc)# encap cisco<br>RP/0/0/CPU0:router (config-fr-vc)# commit</code></td><td>Repeat Step 1 through Step 6 to bring up the serial subinterface and any associated PVC at the other end of the connection.<br><br>Note The configuration on both ends of the subinterface connection must match.</td></tr>
</table>

| | Command or Action | Purpose |
|---|---|---|
| **Step 8** | **show frame-relay pvc** *dlci-number*<br><br>**Example:**<br>`RP/0/0/CPU0:router# show frame-relay pvc 20` | (Optional) Verifies the configuration of specified serial interface. |
| **Step 9** | **show policy-map interface serial** *interface-path-id.subinterface* {**input** \| **output**}<br>or<br>**show policy-map type qos interface serial** *interface-path-id.subinterface* {**input** \| **output**}<br><br>**Example:**<br>`RP/0/0/CPU0:router# show policy-map interface serial 0/1/0/0.1 output`<br>or<br>`RP/0/0/CPU0:router# show policy-map type qos interface serial 0/1/0/0.1 output` | (Optional) Displays the statistics and the configurations of the input and output policies that are attached to a subinterface. |

## What to Do Next

- To modify the default Frame Relay configuration on serial interfaces that have Frame Relay encapsulation enabled, see the "Modifying the Default Frame Relay Configuration on an Interface" section of the *"Configuring Frame Relay Cisco IOS XR Software"* module in this manual.

# Modifying the Keepalive Interval on Serial Interfaces

Perform this task to modify the keepalive interval on serial interfaces that have Cisco HDLC or PPP encapsulation enabled.

**Note** When you enable Cisco HDLC or PPP encapsulation on a serial interface, the default keepalive interval is 10 seconds. Use this procedure to modify that default keepalive interval.

**Note** Cisco HDLC is enabled by default on serial interfaces.

## Prerequisites

Before modifying the keepalive timer configuration, ensure that Cisco HDLC or PPP encapsulation is enabled on the interface. Use the **encapsulation** command to enable Cisco HDLC or PPP encapsulation on the interface, as described in the "Configuring Optional Serial Interface Parameters" section on page 458.

## Restrictions

- Before performing a Minimal Disruptive Restart (MDR) upgrade, we recommend disabling keepalives on a Cisco XR 12000 Series Router.

## SUMMARY STEPS

1. **configure**
2. **interface serial** *interface-path-id*
3. **keepalive** {*seconds* | **disable**}
   or
   **no keepalive**
4. **end**
   or
   **commit**
5. **show interfaces** *type interface-path-id*

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **interface serial** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface serial 0/1/0/0` | Specifies the serial interface name and notation *rack/slot/module/port* and enters interface configuration mode. |
| **Step 3** | **keepalive** {*seconds* \| **disable**}<br>or<br>**no keepalive**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# keepalive 3`<br>or<br>`RP/0/0/CPU0:router(config-if)# no keepalive` | Specifies the number of seconds between keepalive messages.<br>• Use the **keepalive disable** command, the **no keepalive**, or the **keepalive** command with an argument of 0 to disable the keepalive feature.<br>• The range is from 1 to 30 seconds. The default is 10 seconds.<br>• If keepalives are configured on an interface, use the **no keepalive** command to disable the keepalive feature before configuring Frame Relay encapsulation on that interface. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# end`<br>or<br>`RP/0/0/CPU0:router(config-if)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 interfaces serial** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router# show interfaces serial 0/1/0/0` | (Optional) Verifies the interface configuration. |

# How to Configure a Layer 2 Attachment Circuit

The Layer 2 AC configuration tasks are described in the following procedures:

- Creating a Serial Layer 2 Subinterface with a PVC
- Configuring Optional Serial Layer 2 PVC Parameters

**Note** After you configure an interface for Layer 2 switching, no routing commands such as **ipv4 address** are permissible. If any routing commands are configured on the interface, then the **l2transport** command is rejected.

# Creating a Serial Layer 2 Subinterface with a PVC

The procedure in this section creates a Layer 2 subinterface with a PVC.

## Prerequisites

Before you can create a subinterface on a serial interface, you must bring up a serial interface, as described in the "Bringing Up a Serial Interface" section on page 455.

## Restrictions

Only one PVC can be configured for each serial subinterface.

### SUMMARY STEPS

1. **configure**
2. **interface serial** *interface-path-id.subinterface* **l2transport**
3. **pvc** *vpi/vci*
4. **end**
   or
   **commit**
5. Repeat Step 1 through Step 4 to bring up the serial subinterface and any associated PVC at the other end of the AC.

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **`configure`**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **`interface serial`** *`interface-path-id.subinterface`* **`l2transport`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface serial 0/1/0/0.1 l2transport` | Creates a subinterface and enters serial subinterface configuration mode for that subinterface. |
| **Step 3** | **`pvc`** *`vpi/vci`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# pvc 5/20` | Creates a serial permanent virtual circuit (PVC) and enters serial Layer 2 transport PVC configuration mode.<br><br>**Note** Only one PVC is allowed per subinterface. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-fr-vc)# end`<br>or<br>`RP/0/0/CPU0:router(config-fr-vc)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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** | Repeat Step 1 through Step 4 to bring up the serial subinterface and any associated PVC at the other end of the AC. | Brings up the AC.<br>**Note** The configuration on both ends of the AC must match. |

## What to Do Next

- To configure optional PVC parameters, see the "Configuring Optional Serial Layer 2 PVC Parameters" section on page 470.
- For detailed information about configuring L2TPv3 in your network, see the "*Implementing Layer 2 Tunnel Protocol Version 3 on Cisco IOS XR Software*" module of the *Cisco IOS XR Virtual Private Network Configuration Guide for the Cisco XR 12000 Series Router*. For detailed information about configuring L2VPNs, see the "*Implementing MPLS Layer 2 VPNs on Cisco IOS XR Software*" module of the *Cisco IOS XR Virtual Private Network Configuration Guide for the Cisco XR 12000 Series Router*.

# Configuring Optional Serial Layer 2 PVC Parameters

This task describes the commands you can use to modify the default configuration on a serial Layer 2 PVC.

## Prerequisites

Before you can modify the default PVC configuration, you must create the PVC on a Layer 2 subinterface, as described in the "Creating a Serial Layer 2 Subinterface with a PVC" section on page 468.

## Restrictions

The configuration on both ends of the PVC must match for the connection to be active.

### SUMMARY STEPS

1. **configure**
2. **interface serial** *interface-path-id.subinterface* **l2transport**
3. **pvc** *dlci*
4. **encap** [**cisco** | **ietf**]
5. **service-policy** {**input** | **output**} *policy-map*
6. **fragment end-to-end** *fragment-size*
7. fragment-counter
8. **end**
   or
   **commit**
9. Repeat Step 1 through Step 7 to configure the PVC at the other end of the AC.
10. **show policy-map interface serial** *interface-path-id.subinterface* {**input** | **output**}
    or
    **show policy-map type qos interface serial** *interface-path-id.subinterface* {**input** | **output**}

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | `configure`<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **interface serial** *interface-path-id.subinterface* **l2transport**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface serial 0/1/0/0.1 l2transport` | Enters serial subinterface configuration mode for a Layer 2 serial subinterface. |
| **Step 3** | **pvc** *dlci*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# pvc 100` | Enters serial Frame Relay PVC configuration mode for the specified PVC. |
| **Step 4** | **encap** {**cisco** \| **ietf**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-fr-vc)# encapsulation aal5` | Configures the encapsulation for a Frame Relay PVC. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **`service-policy`** `{`**`input`** `|` **`output`**`}` *`policy-map`*<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-subif)# service-policy output policy1` | Attaches a policy map to an input subinterface or output subinterface. Once attached, the policy map is used as the service policy for the subinterface. |
| **Step 6** | **`fragment end-to-end`** *`fragment-size`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-fr-vc)# fragment end-to-end 100` | Enables fragmentation of Frame Relay frames on an interface.<br><br>Replace *fragment-size* with the number of payload bytes from the original Frame Relay frame that will go into each fragment. This number excludes the Frame Relay header of the original frame.<br><br>On the Cisco 8-Port Channelized T1/E1 SPA, valid values are 128, 256, and 512. |
| **Step 7** | **`fragment-counter`**<br><br>**`Example:`**<br>`RP/0/0/CPU0:router(config-fr-vc)# fragment-counter` | Enables fragmentation counters for a Frame Relay subinterface and PVC. |
| **Step 8** | **`end`**<br>or<br>**`commit`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-serial-l2transport-pvc)# end`<br>or<br>`RP/0/0/CPU0:router(config-serial-l2transport-pvc)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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><br>Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 9** | Repeat Step 1 through Step 7 to configure the PVC at the other end of the AC. | Brings up the AC.<br>**Note** The configuration on both ends of the connection must match. |
| **Step 10** | **show policy-map interface serial** *interface-path-id.subinterface* {**input** \| **output**}<br>or<br>**show policy-map type qos interface serial** *interface-path-id.subinterface* {**input** \| **output**}<br><br>**Example:**<br>`RP/0/0/CPU0:router# show policy-map interface pos 0/1/0/0.1 output`<br>or<br>`RP/0/0/CPU0:router# show policy-map type qos interface pos 0/1/0/0.1 output` | (Optional) Displays the statistics and the configurations of the input and output policies that are attached to a subinterface. |

## What to Do Next

- To configure a point-to-point pseudowire XConnect on the AC you just created, see the "*Implementing Layer 2 Tunnel Protocol Version 3 on Cisco IOS XR Software*" module of the *Cisco IOS XR Virtual Private Network Configuration Guide for the Cisco XR 12000 Series Router.*
- To configure an L2VPN, see the "*Implementing MPLS Layer 2 VPNs on Cisco IOS XR Software*" module of the *Cisco IOS XR Virtual Private Network Configuration Guide for the Cisco XR 12000 Series Router.*

# Configuring IPHC

This section contains the following step procedures:


- Prerequisites for Configuring IPHC, page 473
- Configuring the IPHC Slot Level Command, page 474
- Configuring an IPHC Profile, page 475
- Enabling an IPHC Profile on an Interface, page 478


## Prerequisites for Configuring IPHC

IP header compression (IPHC) is supported on the following cards:

- Cisco 1-Port Channelized OC-12/STM-4
- Cisco 1-Port Channelized OC-48/STM-16
- Cisco 1-Port Channelized STM-1/OC-3
- Cisco 8-Port Channelized T1/E1 SPA
- Cisco 2-Port and 4-Port Clear Channel T3/E3 SPA
- Cisco 2-Port and 4-Port Channelized T3 SPA

- Cisco Multirate 10G IP Services Engines SIPs
  - Cisco 12000-SIP-600
  - Cisco 12000-SIP-401
  - Cisco 12000-SIP-501
  - Cisco 12000-SIP-601

## Configuring the IPHC Slot Level Command

This section describes how to configure the IP header compression (IPHC) slot level command, which reserves the IPHC resources, enables IPHC on the line card, and defines the maximum number of TCP and non-TCP connections for the nodes. This configuration must be done before an IPHC profile can be created.

**Note** IPHC slot level configuration is required on both the peer routers.

### SUMMARY STEPS

To configure the IP header compression (IPHC) slot level, perform the following steps.

1. **config**
2. **iphc tcp connections** *max-number* **location** *node-id*
3. **iphc non-tcp connections** *max-number* **location** *node-id*
4. **commit**

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | `config`<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | **iphc tcp connections** *max-number* **location** *node-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# iphc tcp connections 2000 location 0/1/cpu0` | Sets the maximum number of TCP connections that may be configured for IPHC on a line card.<br>The range is 1 to 2000. |
| **Step 3** | **iphc non-tcp connections** *max-number* **location** *node-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# iphc non-tcp connections 20000 location 0/1/cpu0` | Sets the maximum number of non-TCP connections that may be configured for IPHC on a line card.<br>The range is 1 to 20000. |
| **Step 4** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# end`<br>or<br>`RP/0/0/CPU0:router(config-if)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

## Configuring an IPHC Profile

This section describes how to create and configure an IP header compression (IPHC) profile. This procedure is for TCP and non-TCP compression.

### SUMMARY STEPS

To configure an IP header compression (IPHC) profile, perform the following steps.

1. **configure**
2. **iphc profile** *profile-name* **type** {**ietf** | **iphc**}
3. **tcp compression**
4. **tcp context absolute** *number-of-contexts*
5. **non-tcp compression**

6. **non-tcp context absolute** *number-of-contexts*
7. **rtp**
8. **refresh max-period** {*max-number* | ***infinite***}
9. **refresh rtp**
10. **feedback disable**
11. **max-header** *number-of-bytes*
12. **end**
   or
   **commit**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **`configure`**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **`iphc profile`** *`profile-name`* **`type`** `{`**`ietf`** `\|` **`iphc`**`}`<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# iphc profile Profile_1 type iphc` | Creates an IPHC profile, sets the compression format type. and enters the IPHC profile configuration mode. |
| **Step 3** | **`tcp compression`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-iphc-profile)# tcp compression` | Enables TCP compression in an IPHC profile. |
| **Step 4** | **`tcp context absolute`** `number-of-contexts`<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-iphc-profile)# tcp context absolute 255` | Configures the maximum number of TCP contexts that are allowed for IPHC on a line card. |
| **Step 5** | **`non-tcp compression`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-iphc-profile)# non-tcp compression` | Enables non-TCP compression in an IPHC profile. |
| **Step 6** | **`non-tcp context absolute`** `number-of-contexts`<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-iphc-profile)# non-tcp context absolute 255` | Configures the maximum number of non-TCP contexts that are allowed for IPHC on a line card. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 7** | **rtp**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-iphc-profile)# rtp` | Configures Real Time Protocol (RTP) on the interface. |
| **Step 8** | **refresh max-period** {*max-number* \| **infinite**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-iphc-profile)# refresh max-period 50` | Configures the maximum number of compressed IP header packets that are exchanged on a link before the IPHC context is refreshed. |
| **Step 9** | **refresh rtp**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-iphc-profile)# refresh rtp` | Enables the configured context refresh settings for RTP packets. |
| **Step 10** | **feedback disable**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-iphc-profile)# feedback disable` | Disables the IPHC context status feedback messages on an interface. |
| **Step 11** | **max-header** *number-of-bytes*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-iphc-profile)# max-header 20` | Configures the maximum size (in bytes) of a compressed IP header. |
| **Step 12** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# end`<br>or<br>`RP/0/0/CPU0:router(config-if)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

## Enabling an IPHC Profile on an Interface

This section describes how to enable an IP header compression (IPHC) profile on an interface by attaching the profile directly to the interface.

### SUMMARY STEPS

To configure to enable an IPHC profile on an interface, perform the following steps.

1. **config**
2. **interface** *type interface-path-id*
3. **encapsulation ppp**
4. **ipv4 iphc profile** *profile-name* [**mode service-policy**]
5. **service policy input | output | type** *service-policy-name*
6. **commit**

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| Step 1 | `config`<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| Step 2 | `interface type interface-path-id`<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface serial 0/1/0/1` | Specifies the interface.<br><br>**Note** Use the **show interfaces** command to see a list of all interfaces currently configured on the router.<br><br>For more information about the syntax for the router, use the question mark (**?**) online help function. |
| Step 3 | `encapsulation {hdlc \| ppp \| frame-relay \| mfr}`<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# encapsulation ppp` | Specifies Layer 2 encapsulation for the interface. |
| Step 4 | `ipv4 iphc profile profile-name [mode service-policy]`<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# ipv4 iphc profile Profile_1`<br>`or`<br>`RP/0/0/CPU0:router(config-if)# ipv4 iphc profile Profile_1 mode service-policy` | Attaches an IPHC profile to the interface:<br>• *profile-name*—Text name of the IPHC profile to attach to the interface.<br>• **mode service-policy**—Specifies that the IPHC profile applies only to a QoS service policy. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **service policy output** *service-policy-name*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# service policy input \| output \| type service-policy-name` | (Optional) Specifies the name of the QoS service policy to which the IPHC profile applies. Only output service policies are allowed.<br>Used only when **mode service-policy** is specified in Step 2. |
| **Step 6** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# end`<br>or<br>`RP/0/0/CPU0:router(config-if)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

# Configuration Examples for Serial Interfaces

This section provides the following configuration examples:

- Bringing Up and Configuring a Serial Interface with Cisco HDLC Encapsulation: Example, page 479
- Configuring a Serial Interface with Frame Relay Encapsulation: Example, page 480
- Configuring a Serial Interface with PPP Encapsulation: Example, page 482
- IPHC Configuration: Examples, page 482

## Bringing Up and Configuring a Serial Interface with Cisco HDLC Encapsulation: Example

The following example shows how to bring up a basic serial interface with Cisco HDLC encapsulation:

```
RP/0/0/CPU0:Router#config
RP/0/0/CPU0:Router(config)# interface serial 0/3/0/0/0:0
RP/0/0/CPU0:Router(config-if)# ipv4 address 192.0.2.2 255.255.255.252
RP/0/0/CPU0:Router(config-if)# no shutdown
```

```
RP/0/0/CPU0:router(config-if)# end
Uncommitted changes found, commit them? [yes]: yes
```

The following example shows how to configure the interval between keepalive messages to be 10 seconds:

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# interface serial 0/3/0/0/0:0
RP/0/0/CPU0:router(config-if)# keepalive 10
RP/0/0/CPU0:router(config-if)# commit
```

The following example shows how to modify the optional serial interface parameters:

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# interface serial 0/3/0/0/0:0
RP/0/0/CPU0:Router(config-if)# serial
RP/0/0/CPU0:Router(config-if-serial)# crc 16
RP/0/0/CPU0:Router(config-if-serial)# invert
RP/0/0/CPU0:Router(config-if-serial)# scramble
RP/0/0/CPU0:Router(config-if-serial)# transmit-delay 3
RP/0/0/CPU0:Router(config-if-serial)# commit
```

The following is sample output from the **show interfaces serial** command:

```
RP/0/0/CPU0:Router# show interfaces serial 0/0/3/0/5:23
Serial0/0/3/0/5:23 is down, line protocol is down
  Hardware is Serial network interface(s)
  Internet address is Unknown
  MTU 1504 bytes, BW 64 Kbit
     reliability 143/255, txload 1/255, rxload 1/255
  Encapsulation HDLC, crc 16,  loopback not set, keepalive set (10 sec)
  Last clearing of "show interface" counters 18:11:15
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     2764 packets input, 2816 bytes, 3046 total input drops
     0 drops for unrecognized upper-level protocol
     Received 0 broadcast packets, 0 multicast packets
              0 runts, 0 giants, 0 throttles, 0 parity
     3046 input errors, 1 CRC, 0 frame, 0 overrun, 2764 ignored, 281 abort
     2764 packets output, 60804 bytes, 0 total output drops
     Output 0 broadcast packets, 0 multicast packets
     0 output errors, 0 underruns, 0 applique, 0 resets
     0 output buffer failures, 0 output buffers swapped out
     0 carrier transitions
```

# Configuring a Serial Interface with Frame Relay Encapsulation: Example

The following example shows how to create a serial interface on a SPA with Frame Relay encapsulation and a serial subinterface with a PVC on router 1:

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# interface serial 0/1/0/0
RP/0/0/CPU0:router(config-if)# encapsulation frame-relay
RP/0/0/CPU0:router(config-if)#frame-relay intf-type dce
RP/0/0/CPU0:router(config-if)# no shutdown
RP/0/0/CPU0:router(config-if)# end
Uncommitted changes found, commit them? [yes]: yes

RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# interface serial 0/1/0/0.1 point-to-point
```

```
RP/0/0/CPU0:router (config-subif)#ipv4 address 10.20.3.1/24
RP/0/0/CPU0:router (config-subif)# pvc 16
RP/0/0/CPU0:router (config-fr-vc)# encapsulation ietf
RP/0/0/CPU0:router (config-fr-vc)# commit
RP/0/0/CPU0:router(config-fr-vc)# exit
RP/0/0/CPU0:router(config-subif)# exit
RP/0/0/CPU0:router(config)# exit

RP/0/0/CPU0:router# show interface serial 0/1/0/0
Wed Oct  8 04:14:39.946 PST DST
Serial0/1/0/0 is up, line protocol is up
  Interface state transitions: 5
  Hardware is Serial network interface(s)
  Internet address is 10.20.3.1/24
  MTU 4474 bytes, BW 44210 Kbit
     reliability 255/255, txload 0/255, rxload 0/255
  Encapsulation FRAME-RELAY, crc 16,
  Scrambling is disabled, Invert data is disabled
  LMI enq sent  0, LMI stat recvd 0, LMI upd recvd 0
  LMI enq recvd 880, LMI stat sent  880, LMI upd sent  0 , DCE LMI up
  LMI DLCI 1023  LMI type is CISCO  frame relay DCE
  Last clearing of "show interface" counters 02:23:04
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     858 packets input, 11154 bytes, 0 total input drops
     0 drops for unrecognized upper-level protocol
     Received 0 runts, 0 giants, 0 throttles, 0 parity
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     858 packets output, 12226 bytes, 0 total output drops
     0 output errors, 0 underruns, 0 applique, 0 resets
     0 output buffer failures, 0 output buffers swapped out
```

The following example shows how to create a serial interface on a SPA with Frame Relay encapsulation and a serial subinterface with a PVC on router 2, which is connected to router 1:

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# interface serial 0/1/0/1
RP/0/0/CPU0:router(config-if)# encapsulation frame-relay
RP/0/0/CPU0:router(config-if)# no shutdown
RP/0/0/CPU0:router(config-if)# end
Uncommitted changes found, commit them? [yes]: yes

RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# interface serial 0/1/0/1.1 point-to-point
RP/0/0/CPU0:router (config-subif)#ipv4 address 10.20.3.2/24
RP/0/0/CPU0:router (config-subif)# pvc 16
RP/0/0/CPU0:router (config-fr-vc)# encapsulation ietf
RP/0/0/CPU0:router (config-fr-vc)# commit
```

```
RP/0/0/CPU0:router(config-fr-vc)# exit
RP/0/0/CPU0:router(config-subif)# exit
RP/0/0/CPU0:router(config)# exit

RP/0/0/CPU0:router# show interface serial 0/1/0/1
Wed Oct  8 04:13:45.046 PST DST
Serial0/1/0/1 is up, line protocol is up
  Interface state transitions: 7
  Hardware is Serial network interface(s)
  Internet address is Unknown
  MTU 4474 bytes, BW 44210 Kbit
     reliability 255/255, txload 0/255, rxload 0/255
  Encapsulation FRAME-RELAY, crc 16,
  Scrambling is disabled, Invert data is disabled
  LMI enq sent  1110, LMI stat recvd 875, LMI upd recvd 0, DTE LMI up
  LMI enq recvd 0, LMI stat sent  0, LMI upd sent  0
  LMI DLCI 1023  LMI type is CISCO  frame relay DTE
  Last clearing of "show interface" counters 02:22:09
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     853 packets input, 12153 bytes, 0 total input drops
     0 drops for unrecognized upper-level protocol
     Received 0 runts, 0 giants, 0 throttles, 0 parity
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     853 packets output, 11089 bytes, 0 total output drops
     0 output errors, 0 underruns, 0 applique, 0 resets
     0 output buffer failures, 0 output buffers swapped out
```

# Configuring a Serial Interface with PPP Encapsulation: Example

The following example shows how to create and configure a serial interface with PPP encapsulation:

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# interface serial 0/3/0/0/0:0
RP/0/0/CPU0:router(config-if)# ipv4 address 172.18.189.38 255.255.255.224
RP/0/0/CPU0:router(config-if)# encapsulation ppp
RP/0/0/CPU0:router(config-if)# no shutdown
RP/0/0/CPU0:router(config-if)# ppp authentication chap MIS-access
RP/0/0/CPU0:router(config-if)# end
Uncommitted changes found, commit them? [yes]: yes
```

The following example shows how to configure serial interface 0/3/0/0/0:0 to allow two additional retries after an initial authentication failure (for a total of three failed authentication attempts):

```
RP/0/0/CPU0:router# configuration
RP/0/0/CPU0:router(config)# interface serial 0/3/0/0/0:0
RP/0/0/CPU0:router(config-if)# encapsulation ppp
RP/0/0/CPU0:router(config-if)# ppp authentication chap
RP/0/0/CPU0:router(config-if)# ppp max-bad-auth 3
RP/0/0/CPU0:router(config-if)# end
Uncommitted changes found, commit them? [yes]: yes
```

# IPHC Configuration: Examples

This section provides the following examples:


- IPHC Profile Configuration: Example, page 483
- IPHC on a Serial Interface Configuration: Examples, page 483
- IPHC on Multilink Configuration: Example, page 483

- IPHC on a Serial Interface with MLPPP/LFI and QoS Configuration: Example, page 484

## IPHC Profile Configuration: Example

The following example shows how to configure an IPHC Profile:

```
config
  iphc tcp connections 6000 location 0/2/1
  iphc non-tcp connections 6000 location 0/2/1
  iphc profile Profile_1 type iphc
    tcp compression
    tcp context absolute 255
    non-tcp compression
    non-tcp context absolute 255
    rtp
    refresh max-period 50
    refresh rtp
    feedback disable
    max-header 20
commit
```

## IPHC on a Serial Interface Configuration: Examples

### Example 1

The following example shows how to enable an IP header compression (IPHC) profile on a serial interface by attaching the profile directly to the interface:

```
config
  interface serial 0/1/0/1
    encapsulation ppp
    ipv4 iphc profile Profile_1
commit
```

### Example 2

The following example shows how to enable an IP header compression (IPHC) profile on an interface by specifying a QoS service policy that contains an IPHC profile:

```
config
  interface serial 0/1/0/1:1
    encapsulation ppp
    ipv4 iphc profile Profile_2 mode service-policy
    service-policy output ip_header_compression_policy_map
commit
```

## IPHC on Multilink Configuration: Example

The following example shows how to configure an IP header compression (IPHC) on a multilink interface:

```
config
  interface multilink 0/4/3/0/4
    ipv4 address 10.10.10.10
    encapsulation ppp
    ipv4 iphc profile Profile_1
    commit
  interface serial 0/1/0/1:1
    encapsulation ppp
    multilink group 4
    commit
```

### IPHC on a Serial Interface with MLPPP/LFI and QoS Configuration: Example

The following example shows how to configure IP header compression (IPHC) on a serial interface with LFI and by specifying a QoS service policy that contains an IPHC profile:

```
config
  interface multilink 0/4/3/0/4
    ipv4 address 10.10.10.10
      multilink
        fragment-size 128
        interleave
    ipv4 iphc profile Profile_2 mode service-policy
    service-policy output SP_2
    commit
  interface serial 0/1/0/1:2
    encapsulation ppp
    multilink group 4
    commit
```

# Additional References

These sections provide references related to T3/E3 and T1/E1 controllers and serial interfaces.

## Related Documents

| Related Topic | Document Title |
|---|---|
| Cisco IOS XR master command reference | *Cisco IOS XR Master Commands List* |
| Cisco IOS XR interface configuration commands | *Cisco IOS XR Interface and Hardware Component Command Reference* |
| Initial system bootup and configuration information for a router using Cisco IOS XR software | *Cisco IOS XR Getting Started Guide* |
| Cisco IOS XR AAA services configuration information | *Cisco IOS XR System Security Configuration Guide* and *Cisco IOS XR System Security Command Reference* |

## Standards

| Standards | Title |
|---|---|
| FRF.1.2 | *PVC User-to-Network Interface (UNI) Implementation Agreement - July 2000* |
| ANSI T1.617 Annex D | — |
| ITU Q.933 Annex A | — |

## MIBs

| MIBs | MIBs Link |
|---|---|
| — | To locate and download MIBs using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: http://www.cisco.com/go/mibs |

## RFCs

| RFCs | Title |
|---|---|
| RFC 1294 | *Multiprotocol Interconnect Over Frame Relay* |
| RFC 1315 | *Management Information Base for Frame Relay DTEs* |
| RFC 1490 | *Multiprotocol Interconnect Over Frame Relay* |
| RFC 1586 | *Guidelines for Running OSPF Over Frame Relay Networks* |
| RFC 1604 | *Definitions of Managed Objects for Frame Relay Service* |
| RFC 2115 | *Management Information Base for Frame Relay DTEs Using SMIv2* |
| RFC 2390 | *Inverse Address Resolution Protocol* |
| RFC 2427 | *Multiprotocol Interconnect Over Frame Relay* |
| RFC 2954 | *Definitions of Managed Objects for Frame Relay Service* |

## 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/support |

# Configuring Frame Relay Cisco IOS XR Software

This module describes the optional configurable Frame Relay parameters available on Packet-over-SONET/SDH (POS), multilink, and serial interfaces configured with Frame Relay encapsulation.

**Feature History for Configuring Frame Relay Interfaces on Cisco IOS XR Software**

| Release | Modification |
|---|---|
| Release 3.4.0 | This feature was introduced on the Cisco XR 12000 Series Router. |
| Release 3.5.0 | This feature was updated to support IPv6.<br>Layer 2 Tunnel Protocol Version 3 (L2TPv3) was supported on serial and POS interfaces with Frame Relay encapsulation. |
| Release 3.6.0 | Multilink Frame Relay (FRF.16) and End-to-End Fragmentation (FRF.12) were introduced on the Cisco 1-Port Channelized STM-1/OC-3 shared port adapter (SPA) and the 2-Port and 4-Port Channelized T3 SPAs on the Cisco XR 12000 Series Routers. |
| Release 3.8.0 | FRF.16 and FRF.12 were introduced on the Cisco 1-Port Channelized OC-12 SPA on the Cisco XR 12000 Series Routers.<br>Support for Frame Relay was introduced on the following line cards on the Cisco XR 12000 Series Router:<br>• Cisco 1-port Channelized OC-12/STM-1 Line Card<br>• Cisco 4-port Channelized OC-12/STM-4 Line Card |
| Release 4.0.0 | Support for the following frame relay features was added for the Cisco 8-Port Channelized T1/E1 SPA on the Cisco XR 12000 Series Router:<br>• Multilink Frame Relay (FRF.16)<br>• End-to-End Fragmentation (FRF.12)<br>Support for fragmentation counters using the **fragment-counter** command was added for the following SPAs:<br>• Cisco 1-Port Channelized OC-3/STM-1 SPA<br>• Cisco 4-Port Channelized T3/DS0 SPA<br>• Cisco 8-Port Channelized T1/E1 SPA |

# Contents


- Prerequisites for Configuring Frame Relay, page 488
- Information About Frame Relay Interfaces, page 488
- Configuring Frame Relay, page 495
- Configuration Examples for Frame Relay, page 511
- Additional References, page 515


# Prerequisites for Configuring Frame Relay

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 Frame Relay, be sure that the following conditions are met:

- Your hardware must support POS or serial interfaces.
- You have enabled Frame Relay encapsulation on your interface with the **encapsulation frame relay** command, as described in the appropriate module:
  - To enable Frame Relay encapsulation on a multilink bundle interface, see the "Configuring Multilink Frame Relay Bundle Interfaces" section on page 500.
  - To enable Frame Relay encapsulation on a POS interface, see the *"Configuring POS Interfaces on Cisco IOS XR Software"* module in this manual.
  - To enable Frame Relay encapsulation on a serial interface, see the *Configuring Serial Interfaces on Cisco IOS XR Sotware* module in this manual.

# Information About Frame Relay Interfaces

The following sections explain the various aspects of configuring Frame Relay interfaces:


- Frame Relay Encapsulation, page 488
- Multilink Frame Relay (FRF.16), page 491
- End-to-End Fragmentation (FRF.12), page 495


## Frame Relay Encapsulation

On the Cisco XR 12000 Series Router, Frame Relay is supported on POS and serial main interfaces, and on PVCs that are configured under those interfaces. To enable Frame Relay encapsulation on an interface, use the **encapsulation frame-relay** command in interface configuration mode.

Frame Relay interfaces support two types of encapsulated frames:

- Cisco (this is the default)
- IETF

Use the **encapsulation frame-relay** command in interface configuration mode to configure Cisco or IETF encapsulation on a PVC.

**Note** If the encapsulation type is not configured explicitly for a PVC with the **encapsulation** command, then that PVC inherits the encapsulation type from the main interface.

The **encapsulation frame relay** and **encap (PVC)** commands are described in the following modules:

- To enable Frame Relay encapsulation on a POS interface, see the *"Configuring POS Interfaces on Cisco IOS XR Software"* module in this manual.
- To enable Frame Relay encapsulation on a serial interface, see the *Configuring Serial Interfaces on Cisco IOS XR Sotware* module in this manual.

When an interface is configured with Frame Relay encapsulation and no additional configuration commands are applied, the default interface settings shown in Table 23 are present. These default settings can be changed by configuration as described in this module.

*Table 23 Frame Relay Encapsulation Default Settings*

| Parameter | Configuration File Entry | Default Settings | Command Mode |
|---|---|---|---|
| PVC Encapsulation | **encap** {**cisco** \| **ietf**} | **cisco**<br>**Note** When the **encap** command is not configured, the PVC encapsulation type is inherited from the Frame Relay main interface. | PVC configuration |
| Type of support provided by the interface | **frame-relay intf-type** {**dce** \| **dte**} | **dte** | Interface configuration |
| LMI type supported on the interface | **frame-relay lmi-type** [**ansi** \| **cisco** \| **q933a**] | For a DCE, the default setting is **cisco**.<br>For a DTE, the default setting is synchronized to match the LMI type supported on the DCE.<br>**Note** To return an interface to its default LMI type, use the **no frame-relay lmi-type** [**ansi** \| **cisco** \| **q933a**] command. | Interface configuration |
| Disable or enable LMI | **frame-relay lmi disable** | LMI is enabled by default on Frame Relay interfaces.<br>To reenable LMI on an interface after it has been disabled, use the **no frame-relay lmi disable** command. | Interface configuration |

**Note** The default settings of LMI polling-related commands appear in Table 24 on page 490 and Table 25 on page 491.

## LMI

The Local Management Interface (LMI) protocol monitors the addition, deletion, and status of PVCs. LMI also verifies the integrity of the link that forms a Frame Relay User-Network Interface (UNI).

Frame Relay interfaces supports the following types of LMI on UNI interfaces:

- ANSI—ANSI T1.617 Annex D
- Q.933—ITU-T Q.933 Annex A
- Cisco

Use the **frame-relay lmi-type** command to configure the LMI type to be used on an interface.

**Note** The LMI type that you use must correspond to the PVCs configured on the main interface. The LMI type must match on both ends of a Frame Relay connection.

If your router functions as a switch connected to another non-Frame Relay router, use the **frame-relay intf-type dce** command to configure the LMI type to support data communication equipment (DCE).

If your router is connected to a Frame Relay network, use the **frame-relay intf-type dte** command to configure the LMI type to support data terminal equipment (DTE).

**Note** LMI type auto-sensing is supported on DTE interfaces by default.

Use the **show frame-relay lmi** and **show frame-relay lmi-info** commands in EXEC mode to display information and statistics for the Frame Relay interfaces in your system. (When specifying the *type* and *interface-path-id* arguments, you must specify information for the main interface.) You can modify the error threshold, event count, and polling verification timer and then use the **show frame-relay lmi** command to gather information that can help you monitor and troubleshoot Frame Relay interfaces.

If the LMI type is **cisco** (the default LMI type), the maximum number of PVCs that can be supported under a single interface is related to the MTU size of the main interface. Use the following formula to calculate the maximum number of PVCs supported on a card or SPA :

```
(MTU - 13)/8 = maximum number of PVCs
```

The default number of PVCs supported on POS PVCs configured with **cisco** LMI is 557, while the default number of PVCs supported on serial PVCs configured with **cisco** LMI is 186.

For LMI types that are not from Cisco, up to 992 PVCs are supported under a single main interface.

**Note** If a specific LMI type is configured on an interface, use the **no frame-relay lmi-type** [**ansi** | **cisco** | **q933a**] command to bring the interface back to the default LMI type.

Table 24 describes the commands that can be used to modify LMI polling options on PVCs configured for a DCE.

*Table 24 LMI Polling Configuration Commands for DCE*

| Parameter | Configuration File Entry | Default Settings |
|---|---|---|
| Sets the error threshold on a DCE interface. | **lmi-n392dce** *threshold* | **3** |
| Sets the monitored event count. | **lmi-n393dce** *events* | **4** |
| Sets the polling verification timer on the DCE end. | **lmi-t392dce** *seconds* | **15** |

Table 25 describes the commands that can be used to modify LMI polling options on PVCs configured for a DTE.

*Table 25 LMI Polling Configuration Commands for DTE*

| Parameter | Configuration File Entry | Default Settings |
|---|---|---|
| Set the number of Line Integrity Verification (LIV) exchanges performed before requesting a full status message. | **lmi-n391dte** *polling-cycles* | **6** |
| Sets the error threshold. | **lmi-n392dte** *threshold* | **3** |
| Sets the monitored event count. | **lmi-n393dte** *events* | **4** |
| Sets the polling interval (in seconds) between each status inquiry from the DTE end. | **frame-relay lmi-t391dte** *seconds* | **10** |

# Multilink Frame Relay (FRF.16)

Multilink Frame Relay (MFR) is supported only on the following shared port adapters (SPAs):

- Cisco 1-Port Channelized STM-1/OC-3 SPA
- 2-Port and 4-Port Channelized T3 SPA
- Cisco 1-port Channelized OC-12 SPA
- Cisco 8-Port Channelized T1/E1 SPA

## Multilink Frame Relay High Availability

MFR supports the following levels of high availability support:

- MFR supports a process restart, but some statistics will be reset during a restart of certain processes.
- MFR member links remain operational during a route processor (RP) switchover.

## Multilink Frame Relay Configuration Overview

A multilink Frame Relay interface is part of a multilink bundle that allows Frame Relay encapsulation on its interfaces. You create a multilink Frame Relay interface by configuring the following components:

- MgmtMultilink controller
- Multilink bundle interface that allows Frame Relay encapsulation
- Bundle identifier name
- Multilink Frame Relay subinterfaces
- Bundle interface bandwidth class
- Serial interfaces

### MgmtMultilink Controller

You configure a multilink bundle under a controller, using the following commands:

**controller MgmtMultilink** *rack/slot/bay/controller-id*

**bundle** *bundleId*

This configuration creates the controller for a generic multilink bundle. The controller ID number is the zero-based index of the controller chip. Currently, the SPAs that support multilink Frame Relay have only one controller per bay; therefore, the controller ID number is always zero (0).

#### Multilink Bundle Interface

After you create the multilink bundle, you create a multilink bundle interface that allows Frame Relay encapsulation, using the following commands:

**interface multilink** *interface-path-id*

**encapsulation frame-relay**

This configuration allows you to create multilink Frame Relay subinterfaces under the multilink bundle interface.

---

**Note** After you set the encapsulation on a multilink bundle interface to Frame Relay, you cannot change the encapsulation if the interface has member links or any member links associated with a multilink bundle.

---

#### Bundle Identifier Name

---

**Note** Bundle identifier name is configurable only under Frame Relay Forum 16.1 (FRF 16.1).

---

The bundle identifier (**bid**) name value identifies the bundle interface at both endpoints of the interface. The bundle identifier name is exchanged in the information elements to ensure consistent link assignments.

By default, the interface name, for example, Multilink 0/4/1/0/1, is used as the bundle identifier name. However, you can optionally create a name using the **frame-relay multilink bid** command.

---

**Note** Regardless of whether you use the default name or create a name using the **frame-relay multilink bid** command, each bundle must have a unique name. If the same name is used by different bundles, the bundles will flap indefinitely.

---

The bundle identifier name can be up to 50 characters including the null termination character. The bundle identifier name is configured at the bundle interface level and is applied to each member link.

You configure the bundle identifier name using the following commands:

**interface multilink** *interface-path-id*

**frame-relay multilink bid** *bundle-id-name*

#### Multilink Frame Relay Subinterfaces

You configure a multilink Frame Relay subinterface, using the following command:

**interface multilink** *interface-path-id*[*.subinterface* {**l2transport** | **point-to-point**}]

You can configure up to 992 subinterfaces on a multilink bundle interface.

---

**Note** You configure specific Frame Relay interface features at the subinterface level.

---

#### Multilink Frame-Relay Subinterface Features

The following commands are available to set specific features on a multilink Frame Relay bundle subinterface:

- **mtu** *MTU size*
- **description**
- **shutdown**
- **bandwidth** *bandwidth*
- **service-policy** {**input** | **output**} *policymap-name*

**Note** When entering the **service-policy** command, which enables you to attach a policy map to a multilink Frame Relay bundle subinterface, you must do so while in Frame Relay PVC configuration mode. For more information, see Configuring Multilink Frame Relay Bundle Interfaces, page 500.

#### Bundle Interface Bandwidth Class

**Note** Bandwidth class is configurable only under a multilink bundle interface.

You can configure one of three types of bandwidth classes on a multilink Frame Relay interface:

- a—Bandwidth Class A
- b—Bandwidth Class B
- c—Bandwidth Class C

When Bandwidth Class A is configured and one or more member links are up (PH_ACTIVE), the bundle interface is also up and BL_ACTIVATE is signaled to the Frame Relay connections. When all the member links are down, the bundle interface is down and BL_DEACTIVATE is signaled to the Frame Relay connections.

When Bandwidth Class B is configured and all the member links are up (PH_ACTIVE), the bundle interface is up and BL_ACTIVATE is signaled to the Frame Relay connections. When any member link is down, the bundle interface is down and BL_ACTIVATE is signaled to the Frame Relay connections.

When Bandwidth Class C is configured, you must also set the bundle link threshold to a value between 1 and 255. The threshold value is the minimum number of links that must be up (PH_ACTIVE) for the bundle interface to be up and for BL_ACTIVATE to be signaled to the Frame Relay connections. When the number of links that are up falls below this threshold, the bundle interface goes down and BL_DEACTIVATE is signaled to the Frame Relay connections. When 1 is entered as the threshold value, the behavior is identical to Bandwidth Class A. If you enter a threshold value that is greater than the number of member links that are up, the bundle remains down.

You configure the bandwidth class for a Frame Relay multilink bundle interface using the following commands:

**interface multilink** *interface-path-id*

**frame-relay multilink bandwidth-class** {**a** | **b** | **c** [*threshold*]}

The default is a (Bandwidth Class A).

#### Serial Interfaces

After the T3 and T1 controllers are configured, you can add serial interfaces to the multilink Frame Relay bundle subinterface by configuring the serial interface, encapsulating it as multilink Frame Relay (mfr), assigning it to the bundle interface (specified by the multilink group number), and configuring a name for the link. You may also configure MFR acknowledge timeout value, retry count for retransmissions and hello interval, for the bundle link.

You configure a multilink Frame Relay serial interface using the following commands:

**interface serial** *rack/slot/module/port/t1-num:channel-group-number*

**encapsulation mfr**

**multilink group** *group number*

**frame-relay multilink lid** *link-id name*

**frame-relay multilink ack** *ack-timeout*

**frame-relay multilink hello** *hello-interval*

**frame-relay multilink retry** *retry-count*

**Note** All serial links in an MFR bundle inherit the value of the **mtu** command from the multilink interface. Therefore, you should not configure the **mtu** command on a serial interface before configuring it as a member of an MFR bundle. The Cisco IOS XR software blocks attempts to configure a serial interface as a member of an MFR bundle if the interface is configured with a nondefault MTU value as well as attempts to change the **mtu** command value for a serial interface that is configured as a member of an MFR bundle.

#### Show Commands

You can verify a multilink Frame Relay serial interface configuration using the following **show** commands:

**show frame-relay multilink location** *node id*

**show frame-relay multilink interface serial** *interface-path-id* [**detail** | **verbose**]

The following example shows the display output of the **show frame-relay multilink location** command:

```
RP/0/0/CPU0:router# show frame-relay multilink  location  0/4/cpu0
Member interface: Serial0/4/2/0/9:0, ifhandle 0x05007b00
HW state = Up, link state = Up
Member of bundle interface Multilink0/4/2/0/2 with ifhandle 0x05007800

Bundle interface: Multilink0/4/2/0/2, ifhandle 0x05007800
    Member Links: 4 active, 0 inactive
    State = Up,   BW Class = C (threshold   3)
    Member Links:
    Serial0/4/2/0/12:0, HW state = Up, link state = Up
    Serial0/4/2/0/11:0, HW state = Up, link state = Up
    Serial0/4/2/0/10:0, HW state = Up, link state = Up
    Serial0/4/2/0/9:0, HW state = Up, link state = Up

Member interface: Serial0/4/2/0/10:0, ifhandle 0x05007c00
HW state = Up, link state = Up
Member of bundle interface Multilink0/4/2/0/2 with ifhandle 0x05007800

Member interface: Serial0/4/2/0/11:0, ifhandle 0x05007d00
HW state = Up, link state = Up
Member of bundle interface Multilink0/4/2/0/2 with ifhandle 0x05007800
```

```
Member interface: Serial0/4/2/0/12:0, ifhandle 0x05007e00
HW state = Up, link state = Up
Member of bundle interface Multilink0/4/2/0/2 with ifhandle 0x05007800
```

The following example shows the display output of

```
RP/0/0/CPU0:router# show frame-relay multilink interface serial 0/4/2/0/10:0

Member interface: Serial0/4/2/0/10:0, ifhandle 0x05007c00
HW state = Up, link state = Up
Member of bundle interface Multilink0/4/2/0/2 with ifhandle 0x05007800
```

## End-to-End Fragmentation (FRF.12)

You can configure an FRF.12 end-to-end fragmentation connection using the data-link connection identifier (DLCI). However, it must be done on a channelized Frame Relay serial interface.

**Note** The **fragment end-to-end** command is not allowed on Packet-over-SONET/SDH (POS) interfaces or under the DLCI of a multilink Frame Relay bundle interface.

You configure FRF.12 end-to-end fragmentation on a DLCI connection using the following command:

**fragment end-to-end** *fragment-size*

The *fragment-size* argument defines the size of the fragments, in bytes, for the serial interface.

**Note** On a DLCI connection, we highly recommend that you configure an egress service policy that classifies packets into high and low priorities, so that interleaving of high-priority and low-priority fragments occurs.

# Configuring Frame Relay

The following sections describe how to configure Frame Relay interfaces.


- Modifying the Default Frame Relay Configuration on an Interface, page 495
- Disabling LMI on an Interface with Frame Relay Encapsulation, page 498
- Configuring Multilink Frame Relay Bundle Interfaces, page 500
- Configuring FRF.12 End-to-End Fragmentation on a Channelized Frame Relay Serial Interface, page 506


## Modifying the Default Frame Relay Configuration on an Interface

Perform this task to modify the default Frame Relay parameters on a Packet-over-SONET/SDH (POS), multilink, or serial interface with Frame Relay encapsulation.

### Prerequisites

Before you can modify the default Frame Relay configuration, you need to enable Frame Relay on the interface, as described in the following modules:

- To enable Frame Relay encapsulation on a POS interface, see the *"Configuring POS Interfaces on Cisco IOS XR Software"* module in this manual.
- To enable Frame Relay encapsulation on a serial interface, see the *Configuring Serial Interfaces on Cisco IOS XR Sotware* module in this manual.

**Note** Before enabling Frame Relay encapsulation on a POS or serial interface, make certain that you have not previously assigned an IP address to the interface. If an IP address is assigned to the interface, you will not be able to enable Frame Relay encapsulation. For Frame Relay, the IP address and subnet mask are configured on the subinterface.

## Restrictions

- The LMI type must match on both ends of the connection for the connection to be active.
- Before you can remove Frame Relay encapsulation on an interface and reconfigure that interface with PPP or HDLC encapsulation, you must remove all interfaces, subinterface, LMI, and Frame Relay configuration from that interface.

### SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. **frame-relay intf-type** {**dce** | **dte**}
4. **frame-relay lmi-type** [**ansi** | **cisco** | **q933a**]
5. **encap** {**cisco** | **ietf**}
6. **end**
   or
   **commit**
7. **show interfaces** [**summary** | [*type interface-path-id*] [**brief** | **description** | **detail** | **accounting** [**rates**]]] [**location** *node-id*]

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | `configure`<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | **interface** *type interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface pos 0/4/0/1` | Enters interface configuration mode. |
| **Step 3** | **frame-relay intf-type** {**dce** \| **dte**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# frame-relay intf-type dce` | Configures the type of support provided by the interface.<br>• If your router functions as a switch connected to another router, use the **frame-relay intf-type dce** command to configure the LMI type to support data communication equipment (DCE).<br>• If your router is connected to a Frame Relay network, use the **frame-relay intf-type dte** command to configure the LMI type to support data terminal equipment (DTE).<br>**Note** The default interface type is DTE. |
| **Step 4** | **frame-relay lmi-type** [**ansi** \| **q933a** \| **cisco**]<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# frame-relay lmi-type ansi` | Selects the LMI type supported on the interface.<br>• Enter the **frame-relay lmi-type ansi** command to use LMI as defined by ANSI T1.617a-1994 Annex D.<br>• Enter the **frame-relay lmi-type cisco** command to use LMI as defined by Cisco (not standard).<br>• Enter the **frame-relay lmi-type q933a** command to use LMI as defined by ITU-T Q.933 (02/2003) Annex A.<br>**Note** The default LMI type is Cisco. |
| **Step 5** | **encap** {**cisco** \| **ietf**}<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-fr-vc)# encap ietf` | Configures the encapsulation for a Frame Relay PVC.<br>**Note** If the encapsulation type is not configured explicitly for a PVC, then that PVC inherits the encapsulation type from the main interface. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 6** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# end`<br>or<br>`RP/0/0/CPU0:router(config-if)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 7** | **show interfaces** [**summary** \| [*type interface-path-id*] [**brief** \| **description** \| **detail** \| **accounting** [**rates**]]] [**location** *node-id*]<br><br>**Example:**<br>`RP/0/0/CPU0:router# show interface pos 0/4/0/1` | (Optional) Verifies the configuration for the specified interface. |

# Disabling LMI on an Interface with Frame Relay Encapsulation

Perform this task to disable LMI on interfaces that have Frame Relay encapsulation.

**Note** LMI is enabled by default on interfaces that have Frame Relay encapsulation enabled. To reenable LMI on an interface after it has been disabled, use the **no frame-relay lmi disable** command in interface configuration mode.

## SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. **frame-relay lmi disable**
4. **end**
   or
   **commit**
5. **show interfaces** [**summary** | [*type interface-path-id*] [**brief** | **description** | **detail** | **accounting** [**rates**]]] [**location** *node-id*]

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **interface** *type interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface POS 0/4/0/1` | Enters interface configuration mode. |
| **Step 3** | **frame-relay lmi disable**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# frame-relay lmi disable` | Disables LMI on the specified interface. |
| **Step 4** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# end`<br>or<br>`RP/0/0/CPU0:router(config-if)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 interfaces** [**summary** \| [*type interface-path-id*] [**brief** \| **description** \| **detail** \| **accounting** [**rates**]]] [**location** *node-id*]<br><br>**Example:**<br>`RP/0/0/CPU0:router# show interfaces POS 0/1/0/0` | (Optional) Verifies that LMI is disabled on the specified interface. |

# Configuring Multilink Frame Relay Bundle Interfaces

Perform these steps to configure a multilink Frame Relay (MFR) bundle interface and its subinterfaces.

## Restrictions

- MFR does not support the FRF.16.1 Vendor Extension Information element.
- All member links in a multilink Frame Relay bundle interface must be of the same type (for example, T1s or E1s). The member links must have the same framing type, such as point-to-point, and they must have the same bandwidth class.
- All member links must be full T1s or E1s. Fractional links, such as DS0s, are not supported.
- All member links must reside on the same SPA; otherwise, they are considered to be unrelated bundles.
- All member links must be connected to the same line card or SPA at the far end.
- A maximum of 992 MFR subinterfaces is supported on each main interface, based on the supported DLCI range 16–1007.
- A maximum of 4096 subinterfaces are supported per line card or SPA.
- The Cisco 8-Port Channelized T1/E1 SPA has the following additional guidelines:
  - A maximum of 8 T1/E1 links per SPA is supported.
  - A maximum of 8 links per bundle is supported.
  - A maximum of 8 MFR bundles per SPA is supported, with a minimum of one link per bundle.
- All serial links in an MFR bundle inherit the value of the **mtu** command from the multilink interface. Therefore, you should not configure the **mtu** command on a serial interface before configuring it as a member of an MFR bundle. The Cisco IOS XR software blocks the following:
  - Attempts to configure a serial interface as a member of an MFR bundle if the interface is configured with a nondefault MTU value.
  - Attempts to change the **mtu** command value for a serial interface that is configured as a member of an MFR bundle.

### SUMMARY STEPS

1. **configure**
2. **controller MgmtMultilink** *rack/slot/bay/controller-id*
3. **exit**
4. **controller t3** *interface-path-id*
5. **mode** *type*
6. **clock source** {**internal** | **line**}
7. **exit**
8. **controller** {**t1** | **e1**} *interface-path-id*
9. **channel-group** *channel-group-number*
10. **timeslots** *range*
11. **exit**

12. **exit**
13. **interface multilink** *interface-path-id*[.*subinterface* {**l2transport** | **point-to-point**}]
14. **encapsulation frame-relay**
15. **frame-relay multilink bid** *bundle-id-name*
16. **frame-relay multilink bandwidth-class** {**a** | **b** | **c** [*threshold*]}
17. **multilink fragment-size** *size* [**fragment-counter**]
18. **exit**
19. **interface multilink** *interface-path-id*[.*subinterface* {**l2transport** | **point-to-point**}]
20. **ipv4 address** *ip-address*
21. **pvc** *dlci*
22. **service-policy** {**input** | **output**} *policy-map*
23. **exit**
24. **exit**
25. **interface serial** *interface-path-id*
26. **encapsulation mfr**
27. **multilink group** *group-id*
28. **frame-relay multilink lid** *link-id name*
29. **frame-relay multilink ack** *ack-timeout*
30. **frame-relay multilink hello** *hello-interval*
31. **frame-relay multilink retry** *retry-count*
32. **exit**
33. **end**
    or
    **commit**
34. **exit**
35. **show frame-relay multilink interface** *type interface-path-id* [**detail** | **verbose**]

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# config` | Enters global configuration mode. |
| **Step 2** | **controller MgmtMultilink** *rack/slot/bay/controller-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller MgmtMultilink 0/1/0/0` | Creates the controller for a generic multilink bundle in the *rack/slot/bay/controller-id* notation and enters the multilink management configuration mode. The controller ID number is the zero-based index of the controller chip. Currently, the SPAs that support multilink Frame Relay have only one controller per bay; therefore, the controller ID number is always zero (0). |
| **Step 3** | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-mgmtmultilink)# exit` | Exits the multilink management configuration mode. |
| **Step 4** | **controller t3** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller t3 0/1/0/0` | Specifies the T3 controller name in the *rack/slot/module/port* notation and enters T3 configuration mode. |
| **Step 5** | **mode** *type*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# mode t1` | Configures the type of multilinks to channelize; for example, 28 T1s. |
| **Step 6** | **clock source** {**internal** \| **line**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# clock source internal` | (Optional) Sets the clocking for individual E3 links.<br><br>**Note** The default clock source is **internal**.<br><br>**Note** When configuring clocking on a serial link, you must configure one end to be **internal**, and the other end to be **line**. If you configure **internal** clocking on both ends of a connection, framing slips occur. If you configure **line** clocking on both ends of a connection, the line does not come up. |
| **Step 7** | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# exit` | Exits T3/E3 controller configuration mode. |
| **Step 8** | **controller** {**t1** \| **e1**} *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller t1 0/1/0/0/0` | Enters T1 or E1 configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| Step 9 | **channel-group** *channel-group-number*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1)# channel-group 0` | Creates a T1 channel group and enters channel group configuration mode for that channel group. |
| Step 10 | **timeslots** *range*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1-channel_group)# timeslots 1-24` | Associates one or more DS0 time slots to a channel group and creates an associated serial subinterface on that channel group.<br>• For T1 controllers—Range is from 1 to 24 time slots.<br>• For E1 controllers—Range is from 1 to 31 time slots.<br>• You can assign all time slots to a single channel group, or you can divide the time slots among several channel groups. |
| Step 11 | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1-channel_group)# exit` | Exits channel group configuration mode. |
| Step 12 | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1)# exit` | Exits T1 configuration mode. |
| Step 13 | **interface multilink** *interface-path-id*[.*subinterface* {**l2transport** \| **point-to-point**}]<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface Multilink 0/1/0/0/100` | Creates a multilink bundle interface where you can specify Frame Relay encapsulation for the bundle. You create multilink Frame Relay subinterfaces under the multilink bundle interface. |
| Step 14 | **encapsulation frame-relay**<br><br>**Example:**<br>`Router(config-if)# encapsulation frame-relay` | Specifies the Frame Relay encapsulation type. |
| Step 15 | **frame-relay multilink bid** *bundle-id-name*<br><br>**Example:**<br>`Router(config-if)# frame-relay multilink bid MFRBundle` | (Optional) By default, the interface name, for example, Multilink 0/4/1/0/1, is used as the bundle identifier name. However, you can optionally create a name using the **frame-relay multilink bid** command.<br>**Note** Regardless of if you use the default name or create a name using the **frame-relay multilink bid** command, each bundle must have a unique name. If the same name is used by different bundles, the bundles will flap indefinitely. |

| | Command or Action | Purpose |
|---|---|---|
| Step 16 | **frame-relay multilink bandwidth-class** {**a** \| **b** \| **c** [*threshold*]}<br><br>**Example:**<br>`Router(config-if)# frame-relay multilink bandwidth-class a` | Configures one of three types of bandwidth classes on a multilink Frame Relay interface:<br>• a—Bandwidth Class A<br>• b—Bandwidth Class B<br>• c—Bandwidth Class C<br>The default is **a** (Bandwidth Class A). |
| Step 17 | **multilink fragment-size** *size* [**fragment-counter**]<br><br>Example:<br>`RP/0/0/CPU0:router(config-if)# multilink fragment-size 256 fragment-counter` | (Optional) Specifies the size of the multilink fragments, and optionally enables counting of the fragmented packets. The default is no fragments. |
| Step 18 | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# exit` | Exits interface configuration mode. |
| Step 19 | **interface multilink** *interface-path-id*[.*subinterface* {**l2transport** / **point-to-point**}]<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface Multilink 0/1/0/0/100.16 point-to-point` | Creates a multilink subinterface in the *rack/slot/bay/controller-id bundleId.subinterace* [**point-to-point** \| **l2transport** ] notation and enters the subinterface configuration mode.<br>• **l2transport**—Treat as an attachment circuit<br>• **point-to-point**—Treat as a point-to-point link<br>You can configure up to 992 subinterfaces on a multilink bundle interface. The DLCIs are 16 to 1007. |
| Step 20 | **ipv4 address** *ip-address*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-subif)# ipv4 address 3.1.100.16 255.255.255.0` | Assigns an IP address and subnet mask to the interface in the format:<br>*A.B.C.D/prefix* or *A.B.C.D/mask* |
| Step 21 | **pvc** *dlci*<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-subif)# pvc 16` | Creates a POS permanent virtual circuit (PVC) and enters Frame Relay PVC configuration submode.<br>Replace *dlci* with a PVC identifier, in the range from 16 to 1007.<br>**Note** Only one PVC is allowed per subinterface. |
| Step 22 | **service-policy** {**input** \| **output**} *policy-map*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-fr-vc)# service-policy output policy-mapA` | Attaches a policy map to an input subinterface or output subinterface. When attached, the policy map is used as the service policy for the subinterface.<br>**Note** For information on creating and configuring policy maps, refer to *Cisco IOS XR Modular Quality of Service Configuration Guide*. |
| Step 23 | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-fr-vc)# exit` | Exits the Frame-Relay virtual circuit mode. |

| | Command or Action | Purpose |
|---|---|---|
| Step 24 | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-subif)# exit` | Exits the subinterface configuration mode. |
| Step 25 | **interface serial** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface serial 0/1/0/0/0/0:0` | Specifies the complete interface number with the *rack/slot/module/port/T3Num/T1num:instance* notation. |
| Step 26 | **encapsulation mfr**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# encapsulation mfr` | Enables multilink Frame Relay on the serial interface. |
| Step 27 | **multilink group** *group-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# multilink group 100` | Specifies the multilink group ID for this interface. |
| Step 28 | **frame-relay multilink lid** *link-id name*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# frame-relay multilink lid sj1` | Configures a name for the Frame Relay multilink bundle link.<br><br>**Note** Each link within a bundle must have a unique name. If the same name is used by different links in the same bundle, the bundles will flap indefinitely. |
| Step 29 | **frame-relay multilink ack** *ack-timeout*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# frame-relay multilink ack 5` | Configures the acknowledge timeout value for the Frame Relay multilink bundle link. |
| Step 30 | **frame-relay multilink hello** *hello-interval*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# frame-relay multilink hello 60` | Configures the hello interval for the Frame Relay multilink bundle link. |
| Step 31 | **frame-relay multilink retry** *retry-count*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# frame-relay multilink retry 2` | Configures the retry count for retransmissions for the Frame Relay multilink bundle link. |
| Step 32 | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# exit` | Exits interface configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| Step 33 | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# end`<br>or<br>`RP/0/0/CPU0:router(config-if)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 34 | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# exit` | Exits global configuration mode. |
| Step 35 | **show frame-relay multilink interface** *type interface-path-id* [**detail** \| **verbose**]<br><br>**Example:**<br>`RP/0/0/CPU0:router# show frame-relay multilink interface Multilink 0/5/1/0/1` | Shows the information retrieved from the interface description block (IDB), including bundle-specific information and Frame Relay information. |

# Configuring FRF.12 End-to-End Fragmentation on a Channelized Frame Relay Serial Interface

Perform the following steps to configure FRF.12 end-to-end fragmentation on a channelized Frame Relay serial interface.

## Restrictions

The Cisco 8-Port Channelized T1/E1 SPA on the Cisco XR 12000 Series Router supports fragment sizes of 128, 512, and 256 bytes only.

## SUMMARY STEPS

1. **config**
2. **controller t3** *interface-path-id*
3. **mode** *type*
4. **clock source** {**internal** | **line**}
5. **exit**
6. **controller t1** *interface-path-id*
7. **channel-group** *channel-group-number*
8. **timeslots** *range*
9. **exit**
10. **exit**
11. **interface serial** *interface-path-id*
12. **encapsulation frame-relay**
13. **exit**
14. **interface serial** *interface-path-id*
15. **ipv4 address** *ip-address*
16. **pvc** *dlci*
17. **service-policy** {**input** | **output**} *policy-map*
18. **fragment end-to-end** *fragment-size*
19. **fragment-counter**
20. **exit**
21. **exit**
22. **exit**
23. **end**
    or
    **commit**
24. **exit**
25. **show frame-relay pvc** [ **dlci** | **interface** | **location** ]

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| Step 1 | **config**<br><br>**Example:**<br>`RP/0/0/CPU0:router# config` | Enters global configuration mode. |
| Step 2 | **controller t3** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller t3 0/1/0/0` | Specifies the T3 controller name in the *rack/slot/module/port* notation and enters T3 configuration mode. |
| Step 3 | **mode** *type*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# mode t1` | Configures the type of multilinks to channelize; for example, 28 T1s. |
| Step 4 | **clock source** {**internal** \| **line**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# clock source internal` | (Optional) Sets the clocking for individual E3 links.<br><br>**Note** The default clock source is **internal**.<br><br>**Note** When configuring clocking on a serial link, you must configure one end to be **internal**, and the other end to be **line**. If you configure **internal** clocking on both ends of a connection, framing slips occur. If you configure **line** clocking on both ends of a connection, the line does not come up. |
| Step 5 | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# exit` | Exits T3/E3 or T1/E1 controller configuration mode. |
| Step 6 | **controller t1** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller t1 0/1/0/0/0` | Enters T1 configuration mode. |
| Step 7 | **channel-group** *channel-group-number*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1)# channel-group 0` | Creates a T1 channel group and enters channel group configuration mode for that channel group. |

| | Command or Action | Purpose |
|---|---|---|
| Step 8 | **timeslots** *range*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1-channel_group)# timeslots 1-24` | Associates one or more DS0 time slots to a channel group and creates an associated serial subinterface on that channel group.<br>• Range is from 1 to 24 time slots.<br>• You can assign all 24 time slots to a single channel group, or you can divide the time slots among several channel groups.<br>**Note** Each individual T1 controller supports a total of 24 DS0 time slots. |
| Step 9 | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1-channel_group)# exit` | Exits channel group configuration mode. |
| Step 10 | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1)# exit` | Exits T1 configuration mode. |
| Step 11 | **interface serial** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface serial 0/1/0/0/0/0:0` | Specifies the complete interface number with the *rack/slot/module/port/T3Num/T1num:instance* notation. |
| Step 12 | **encapsulation frame-relay**<br><br>**Example:**<br>`RP/0/0/CPU0:Router(config-if)# encapsulation frame-relay` | Specifies the Frame Relay encapsulation type. |
| Step 13 | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# exit` | Exits interface configuration mode. |
| Step 14 | **interface serial** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface serial 1/0/0/0/0:0.1` | Specifies the complete subinterface number with the *rack/slot/module/port*[*/channel-num:channel-group-number*].*subinterface* notation. |
| Step 15 | **ipv4 address** *ip-address*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-subif)# ipv4 address 3.1.100.16 255.255.255.0` | Assigns an IP address and subnet mask to the interface in the format:<br>*A.B.C.D/prefix* or *A.B.C.D/mask* |

| | Command or Action | Purpose |
|---|---|---|
| **Step 16** | **pvc** *dlci*<br><br>**Example:**<br>`RP/0/0/CPU0:router (config-subif)# pvc 100` | Creates a POS permanent virtual circuit (PVC) and enters Frame Relay PVC configuration submode.<br><br>Replace *dlci* with a PVC identifier, in the range from 16 to 1007.<br><br>**Note** Only one PVC is allowed per subinterface. |
| **Step 17** | **service-policy** {**input** \| **output**} *policy-map*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-fr-vc)# service-policy output policy-mapA` | Attaches a policy map to an input subinterface or output subinterface. When attached, the policy map is used as the service policy for the subinterface.<br><br>**Note** For effective FRF.12 functionality (interleave specifically), you should configure an egress service policy with priority.<br><br>**Note** For information on creating and configuring policy maps, refer to *Cisco IOS XR Modular Quality of Service Configuration Guide*, |
| **Step 18** | **fragment end-to-end** *fragment-size*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-fr-vc)# fragment end-to-end 100` | (Optional) Enables fragmentation of Frame Relay frames on an interface and specifies the size (in bytes) of the payload from the original frame that will go into each fragment. This number excludes the Frame Relay header of the original frame.<br><br>Valid values are from 64 to 512, depending on your hardware. |
| **Step 19** | **fragment-counter**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-fr-vc)# fragment-counter` | (Optional) Enables fragmentation counters for a Frame Relay subinterface and PVC. |
| **Step 20** | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-fr-vc)# exit` | Exits the Frame-Relay virtual circuit mode. |
| **Step 21** | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-subif)# exit` | Exits the subinterface configuration mode. |
| **Step 22** | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# exit` | Exits interface configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 23** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# end`<br>or<br>`RP/0/0/CPU0:router(config-if)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 24** | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# exit` | Exits global configuration mode. |
| **Step 25** | **show frame-relay pvc** [ **dlci** \| **interface** \| **location** ]<br><br>**Example:**<br>`RP/0/0/CPU0:router# show frame-relay pvc 100` | Displays the information for the specified PVC DLCI, interface, or location. |

# Configuration Examples for Frame Relay

This section provides the following configuration examples:


- Optional Frame Relay Parameters: Example, page 512
- Multilink Frame Relay: Example, page 514
- End-to-End Fragmentation: Example, page 515

## Optional Frame Relay Parameters: Example

The following example shows how to bring up and configure a POS interface with Frame Relay encapsulation. In this example, the user modifies the default Frame Relay configuration so that the interface supports ANSI T1.617a-1994 Annex D LMI on DCE.

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# interface POS 0/3/0/0
RP/0/0/CPU0:router(config-if)# encapsulation frame-relay IETF
RP/0/0/CPU0:router(config-if)# frame-relay intf-type dce
RP/0/0/CPU0:router(config-if)# frame-relay lmi-type ansi
RP/0/0/CPU0:router(config-if)# no shutdown
RP/0/0/CPU0:router(config-if)# end

Uncommitted changes found, commit them? [yes]: yes

RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router (config)# interface pos 0/3/0/0.10 point-to-point
RP/0/0/CPU0:router (config-subif)#ipv4 address 10.46.8.6/24
RP/0/0/CPU0:router (config-subif)# pvc 20
RP/0/0/CPU0:router (config-fr-vc)# encap ietf
RP/0/0/CPU0:router(config-subif)# commit
```

The following example shows how to disable LMI on a POS interface that has Frame Relay encapsulation configured:

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# interface
RP/0/0/CPU0:router(config)# interface pos 0/3/0/0
RP/0/0/CPU0:router(config-if)# frame-relay lmi disable
RP/0/0/CPU0:router(config-if)# end

Uncommitted changes found, commit them? [yes]: yes
```

The following example shows how to reenable LMI on a serial interface:

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# interface
RP/0/0/CPU0:router(config)# interface serial 0/3/0/0
RP/0/0/CPU0:router(config-if)# no frame-relay lmi disable
RP/0/0/CPU0:router(config-if)# end

Uncommitted changes found, commit them? [yes]: yes
```

The following example shows how to display Frame Relay statistics for LMI on all interfaces:

```
RP/0/0/CPU0:router# show frame-relay lmi

LMI Statistics for interface POS0/1/0/0/ (Frame Relay DCE) LMI TYPE = ANSI
Invalid Unnumbered Info 0            Invalid Prot Disc 0
Invalid Dummy Call Ref 0             Invalid Msg Type 0
Invalid Status Message 0             Invalid Lock Shift 9
Invalid Information ID 0             Invalid Report IE Len 0
Invalid Report Request 0             Invalid Keep IE Len 0
Num Status Enq. Rcvd 9444            Num Status Msgs Sent 9444
Num Full Status Sent 1578            Num St Enq. Timeouts 41
Num Link Timeouts 7

LMI Statistics for interface POS0/1/0/1/ (Frame Relay DCE) LMI TYPE = CISCO
Invalid Unnumbered Info 0            Invalid Prot Disc 0
Invalid Dummy Call Ref 0             Invalid Msg Type 0
Invalid Status Message 0             Invalid Lock Shift 0
Invalid Information ID 0             Invalid Report IE Len 0
```

```
Invalid Report Request 0             Invalid Keep IE Len 0
Num Status Enq. Rcvd 9481            Num Status Msgs Sent 9481
Num Full Status Sent 1588            Num St Enq. Timeouts 16
Num Link Timeouts 4
```

The following example shows how to create a serial subinterface with a PVC on the main serial interface:

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# interface serial 0/3/0/0/0:0.10 point-to-point
RP/0/0/CPU0:router (config-subif)#ipv4 address 10.46.8.6/24
RP/0/0/CPU0:router (config-subif)# pvc 20
RP/0/0/CPU0:router (config-fr-vc)# encapsulation ietf
RP/0/0/CPU0:router(config-subif)# commit
```

The following example shows how to display information about all PVCs configured on your system:

```
RP/0/0/CPU0router# show frame-relay pvc

PVC Statistics for interface Serial0/3/2/0 (Frame Relay DCE)

              Active     Inactive      Deleted       Static
  Local          4           0             0            0
  Switched       0           0             0            0
  Dynamic        0           0             0            0

DLCI = 612, DLCI USAGE = LOCAL, ENCAP = CISCO, INHERIT = TRUE, PVC STATUS = ACTI
VE, INTERFACE = Serial0/3/2/0.1
  input pkts 0          output pkts 0           in bytes 0
  out bytes 0           dropped pkts 0          in FECN packets 0
  in BECN pkts 0        out FECN pkts 0         out BECN pkts 0
  in DE pkts 0          out DE pkts 0
  out bcast pkts 0      out bcast bytes 0
  pvc create time 00:00:00      last time pvc status changed 00:00:00

DLCI = 613, DLCI USAGE = LOCAL, ENCAP = CISCO, INHERIT = TRUE, PVC STATUS = ACTI
VE, INTERFACE = Serial0/3/2/0.2
  input pkts 0          output pkts 0           in bytes 0
  out bytes 0           dropped pkts 0          in FECN packets 0
  in BECN pkts 0        out FECN pkts 0         out BECN pkts 0
  in DE pkts 0          out DE pkts 0
  out bcast pkts 0      out bcast bytes 0
  pvc create time 00:00:00      last time pvc status changed 00:00:00

DLCI = 614, DLCI USAGE = LOCAL, ENCAP = CISCO, INHERIT = TRUE, PVC STATUS = ACTI
VE, INTERFACE = Serial0/3/2/0.3
  input pkts 0          output pkts 0           in bytes 0
  out bytes 0           dropped pkts 0          in FECN packets 0
  in BECN pkts 0        out FECN pkts 0         out BECN pkts 0
  in DE pkts 0          out DE pkts 0
  out bcast pkts 0      out bcast bytes 0
  pvc create time 00:00:00      last time pvc status changed 00:00:00

DLCI = 615, DLCI USAGE = LOCAL, ENCAP = CISCO, INHERIT = TRUE, PVC STATUS = ACTI
VE, INTERFACE = Serial0/3/2/0.4
  input pkts 0          output pkts 0           in bytes 0
  out bytes 0           dropped pkts 0          in FECN packets 0
  in BECN pkts 0        out FECN pkts 0         out BECN pkts 0
  in DE pkts 0          out DE pkts 0
  out bcast pkts 0      out bcast bytes 0
  pvc create time 00:00:00      last time pvc status changed 00:00:00
```

The following example shows how to modify LMI polling options on PVCs configured for a DTE, and then use the **show frame-relay lmi** and **show frame-relay lmi-info** commands to display information for monitoring and troublehooting the interface:

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# interface pos 0/3/0/0
RP/0/0/CPU0:router(config-if)# frame-relay lmi-n391dte 10
RP/0/0/CPU0:router(config-if)# frame-relay lmi-n391dte 5
RP/0/0/CPU0:router(config-if)# frame-relay lmi-t391dte 15
RP/0/0/CPU0:router(config-subif)# commit

RP/0/0/CPU0:router# show frame-relay lmi interface pos 0/3/0/0

LMI Statistics for interface pos 0/3/0/0 (Frame Relay DTE) LMI TYPE = ANSI
Invalid Unnumbered Info 0            Invalid Prot Disc 0
Invalid Dummy Call Ref 0             Invalid Msg Type 0
Invalid Status Message 0             Invalid Lock Shift 9
Invalid Information ID 0             Invalid Report IE Len 0
Invalid Report Request 0             Invalid Keep IE Len 0
Num Status Enq. Rcvd 9444            Num Status Msgs Sent 9444
Num Full Status Sent 1578            Num St Enq. Timeouts 41
Num Link Timeouts 7

RP/0/0/CPU0:router# show frame-relay lmi-info interface pos 0/3/0/0

LMI IDB Info for interface POS0/3/0/0
  ifhandle:             0x6176840
  Interface type:       DTE
  Interface state:      UP
  Line Protocol:        UP
  LMI type (cnf/oper):  AUTO/CISCO
  LMI type autosense:   OFF
  Interface MTU:        1504
  -------------- DTE -------------
  T391:                 15s
  N391: (cnf/oper):     5/5
  N392: (cnf/oper):     3/0
  N393:                 4
  My seq#:              83
  My seq# seen:         83
  Your seq# seen:       82
  -------------- DCE -------------
  T392:                 15s
  N392: (cnf/oper):     3/0
  N393:                 4
  My seq#:              0
  My seq# seen:         0
  Your seq# seen:       0
```

# Multilink Frame Relay: Example

The following example shows how to configure multilink Frame Relay with serial interfaces:

```
RP/0/0/CPU0:router# config
RP/0/0/CPU0:router(config)# controller MgmtMultilink 0/3/1/0
RP/0/0/CPU0:router(config-mgmtmultilink)# bundle 100
RP/0/0/CPU0:router(config-mgmtmultilink)# exit

RP/0/0/CPU0:router(config)# controller T3 0/3/1/0
RP/0/0/CPU0:router(config-t3)# mode t1
RP/0/0/CPU0:router(config-t3)# clock source internal
RP/0/0/CPU0:router(config-t3)# exit
```

```
RP/0/0/CPU0:router(config)# controller T1 0/3/1/0/0
RP/0/0/CPU0:router(config-t1)# channel-group 0
RP/0/0/CPU0:router(config-t1-channel_group)# timeslots 1-24
RP/0/0/CPU0:router(config-t1-channel_group)#  exit
RP/0/0/CPU0:router(config-t1-channel_group)# exit
RP/0/0/CPU0:router(config-t1)# exit

RP/0/0/CPU0:router(config)# interface Multilink 0/3/1/0/100
RP/0/0/CPU0:router(config-if)# encapsulation frame-relay
RP/0/0/CPU0:router(config-if)# exit

RP/0/0/CPU0:router(config)# interface Multilink 0/3/1/0/100.16 point-to-point
RP/0/0/CPU0:router(config-subif)# ipv4 address 3.1.100.16 255.255.255.0
RP/0/0/CPU0:router(config-subif)# pvc 16
RP/0/0/CPU0:router(config-fr-vc)# service-policy output policy-mapA
RP/0/0/CPU0:router(config-fr-vc)# exit
RP/0/0/CPU0:router(config-subif)# exit

RP/0/0/CPU0:router(config)# interface Serial 0/3/1/0/0:0
RP/0/0/CPU0:router(config-if)# encapsulation mfr
RP/0/0/CPU0:router(config-if)# multilink group 100
RP/0/0/CPU0:router(config-if)# frame-relay multilink lid sj1
RP/0/0/CPU0:router(config-if)# frame-relay multilink ack 5
RP/0/0/CPU0:router(config-if)# frame-relay multilink hello 60
RP/0/0/CPU0:router(config-if)# frame-relay multilink retry 2
RP/0/0/CPU0:router(config-if)# exit
RP/0/0/CPU0:router(config)#
```

## End-to-End Fragmentation: Example

The following example shows how to configure FRF.12 end-to-end fragmentation on a channelized Frame Relay serial interface:

```
RP/0/0/CPU0:router# config
RP/0/0/CPU0:router(config)# controller T30/3/1/0
RP/0/0/CPU0:router(config-t3)# mode t1
RP/0/0/CPU0:router(config-t3)# clock source internal
RP/0/0/CPU0:router(config-t3)# exit
RP/0/0/CPU0:router(config-t3)# controller T10/3/1/0/0
RP/0/0/CPU0:router(config-t1)# channel-group 0
RP/0/0/CPU0:router(config-t1-channel_group)#  timeslots 1-24
RP/0/0/CPU0:router(config-t1-channel_group)# exit
RP/0/0/CPU0:router(config-t1-channel_group)# interface Serial 0/3/1/0/0:0
RP/0/0/CPU0:router(config-if)# encapsulation frame-relay
RP/0/0/CPU0:router(config-if)# exit
RP/0/0/CPU0:router(config-if)# interface Serial 0/3/1/0/0:0.100 point-to-point
RP/0/0/CPU0:router(config-subif)# ipv4 address 3.1.1.1 255.255.255.0
RP/0/0/CPU0:router(config-subif)#  pvc 100
RP/0/0/CPU0:router(config-fr-vc)#  service-policy output LFI
RP/0/0/CPU0:router(config-fr-vc)# fragment end-to-end 256
RP/0/0/CPU0:router(config-fr-vc-frag)# fragment-counter
```

# Additional References

These sections provide references related to Frame Relay.

## Related Documents

| Related Topic | Document Title |
|---|---|
| Cisco IOS XR master command reference | *Cisco IOS XR Master Commands List* |
| Cisco IOS XR interface configuration commands | *Cisco IOS XR Interface and Hardware Component Command Reference* |
| Initial system bootup and configuration information for a router using Cisco IOS XR software | *Cisco IOS XR Getting Started Guide* |
| Cisco IOS XR AAA services configuration information | *Cisco IOS XR System Security Configuration Guide* and *Cisco IOS XR System Security Command Reference* |

## Standards

| Standards | Title |
|---|---|
| FRF.12 | *Frame Relay Forum .12* |
| FRF.16 | *Frame Relay Forum .16* |
| ANSI T1.617 Annex D | *American National Standards Institute T1.617 Annex D* |
| ITU Q.933 Annex A | *International Telecommunication Union Q.933 Annex A* |

## MIBs

| MIBs | MIBs Link |
|---|---|
| FRF.16 MIB<br>Cisco Frame Relay MIB<br>IF-MIB<br>Management Information Base for Frame Relay DTEs<br>Management Information Base for Frame Relay DTEs Using SMIv2 | To locate and download MIBs using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: http://www.cisco.com/go/mibs |

## RFCs

| RFCs | Title |
|---|---|
| RFC 1294 | *Multiprotocol Interconnect Over Frame Relay* |
| RFC 1315 | *Management Information Base for Frame Relay DTEs* |
| RFC 1490 | *Multiprotocol Interconnect Over Frame Relay* |
| RFC 1586 | *Guidelines for Running OSPF Over Frame Relay Networks* |
| RFC 1604 | *Definitions of Managed Objects for Frame Relay Service* |
| RFC 2115 | *Management Information Base for Frame Relay DTEs Using SMIv2* |
| RFC 2390 | *Inverse Address Resolution Protocol* |

| RFCs | Title |
|---|---|
| RFC 2427 | *Multiprotocol Interconnect Over Frame Relay* |
| RFC 2954 | *Definitions of Managed Objects for Frame Relay Service* |
| RFC 3020 | *RFC for FRF.16 MIB* |

## 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/support |

# Configuring PPP on Cisco IOS XR Software

This module describes the configuration of Point-to-Point Protocol (PPP) on POS and serial interfaces on the Cisco XR 12000 Series Router.

**Feature History for Configuring PPP Interfaces on Cisco IOS XR Software**

| Release | Modification |
|---|---|
| Release 3.3.0 | Support for serial interfaces with PPP encapsulation was introduced on the Cisco XR 12000 Series Router. |
| Release 3.4.1 | Support for Multilink PPP was introduced on the Cisco XR 12000 Series Router. |
| Release 4.0.0 | Support for the following features was added for the 8-Port Channelized T1/E1 SPA on the Cisco XR 12000 Series Router:<br>• IPHC over MLPPP<br>• MLPPP<br>• MLPPP/LFI |

# Contents


- Prerequisites for Configuring PPP, page 519
- Information About PPP, page 520
- How to Configure PPP, page 523
- Configuration Examples for PPP, page 547
- Additional References, page 553


# Prerequisites for Configuring PPP

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 you can configure PPP authentication on a POS or serial interface, be sure that the following tasks and conditions are met:

- Your hardware must support POS or serial interfaces.

- You have enabled PPP encapsulation on your interface with the **encap ppp** command, as described in the appropriate module:
  - To enable PPP encapsulation on a POS interface, see the *Configuring POS Interfaces on Cisco IOS XR Software* module in this manual.
  - To enable PPP encapsulation on a serial interface, see the *Configuring Serial Interfaces on Cisco IOS XR Sotware* module in this manual.

# Information About PPP

To configure PPP and related features, you should understand the information in this section:

- PPP Authentication, page 520
- Multilink PPP, page 522
- T3 SONET Channels, page 523

## PPP Authentication

When PPP authentication is configured on an interface, a host requires that the other host uniquely identify itself with a secure password before establishing a PPP connection. The password is unique and is known to both hosts.

PPP supports the following authentication protocols:

- Challenge-Handshake Authentication Protocol (CHAP)
- Microsoft extension to the CHAP protocol (MS-CHAP)
- Password Authentication Protocol (PAP).

When you first enable PPP on a POS or serial interface, no authentication is enabled on the interface until you configure a CHAP, MS-CHAP, or PAP secret password under that interface. Keep the following information in mind when configuring PPP on an interface:

- CHAP, MS-CHAP, and PAP can be configured on a single interface; however, only one authentication method is used at any one time. The order in which the authentication protocols are used is determined by the peer during the LCP negotiations. The first authentication method used is the one that is also supported by the peer.
- PAP is the least secure authentication protocol available on POS and serial interfaces. To ensure higher security for information that is sent over POS and serial interfaces, we recommend configuring CHAP or MS-CHAP authentication in addition to PAP authentication.
- Enabling or disabling PPP authentication does not effect the local router's willingness to authenticate itself to the remote device.
- The **ppp authentication** command is also used to specify the order in which CHAP, MS-CHAP, and PAP authentication is selected on the interface. You can enable CHAP, MS-CHAP, or PAP in any order. If you enable all three methods, the first method specified is requested during link negotiation. If the peer suggests using the second method, or refuses the first method, the second method is tried. Some remote devices support only one method. Base the order in which you specify methods on the remote device's ability to correctly negotiate the appropriate method and on the level of data line security you require. PAP usernames and passwords are sent as clear text strings, which can be intercepted and reused.

**Caution** If you use a *list-name* value that was not configured with the **aaa authentication ppp** command, your interface cannot authenticate the peer. For details on implementing the **aaa authentication** command with the **ppp** keyword, see the *Authentication, Authorization, and Accounting Commands on Cisco IOS XR Software* module of *Cisco IOS XR System Security Command Reference* and *Configuring AAA Services on Cisco IOS XR Software* module of the *Cisco IOS XR System Security Configuration Guide*.

## PAP Authentication

PAP provides a simple method for a remote node to establish its identity using a two-way handshake. After a PPP link is established between two hosts, a username and password pair is repeatedly sent by the remote node across the link (in clear text) until authentication is acknowledged, or until the connection is terminated.

PAP is not a secure authentication protocol. Passwords are sent across the link in clear text and there is no protection from playback or trial-and-error attacks. The remote node is in control of the frequency and timing of the login attempts.

## CHAP Authentication

CHAP is defined in RFC 1994, and it verifies the identity of the peer by means of a three-way handshake. The steps that follow provide a general overview of the CHAP process:

**Step 1** The CHAP authenticator sends a challenge message to the peer.

**Step 2** The peer responds with a value calculated through a one-way hash function.

**Step 3** The authenticator checks the response against its own calculation of the expected hash value. If the values match, then the authentication is successful. If the values do not match, then the connection is terminated.

This authentication method depends on a CHAP password known only to the authenticator and the peer. The CHAP password is not sent over the link. Although the authentication is only one-way, you can negotiate CHAP in both directions, with the help of the same CHAP password set for mutual authentication.

**Note** For CHAP authentication to be valid, the CHAP password must be identical on both hosts.

## MS-CHAP Authentication

Microsoft Challenge Handshake Authentication Protocol (MS-CHAP) is the Microsoft version of CHAP and is an extension to RFC 1994. MS-CHAP follows the same authentication process used by CHAP. In this case, however, authentication occurs between a PC using Microsoft Windows NT or Microsoft Windows 95 and a Cisco router or access server acting as a network access server (NAS).

**Note** For MS-CHAP authentication to be valid, the MS-CHAP password must be identical on both hosts.

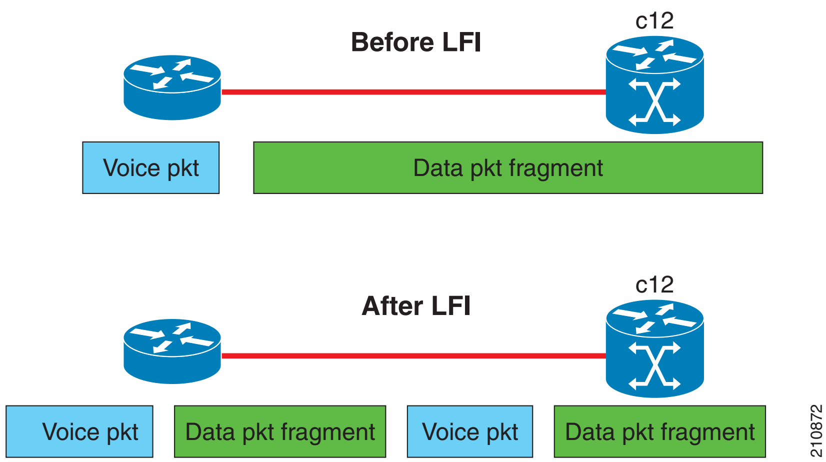
# Multilink PPP

Multilink Point-to-Point Protocol (MLPPP) provides a method for combining multiple physical links into one logical link. The implementation combines multiple PPP interfaces into one multilink interface. MLPPP performs the fragmenting, reassembling, and sequencing of datagrams across multiple PPP links.

Link Fragmentation and Interleaving (LFI) is designed for MLPPP interfaces and is required when integrating voice and data on low-speed interfaces.

Link Fragmentation and Interleaving (LFI) provides stability for delay-sensitive traffic, such as voice or video, traveling on the same circuit as data. Voice is susceptible to increased latency and jitter when the network processes large packets on low-speed interfaces. LFI reduces delay and jitter by fragmenting large datagrams and interleaving them with low-delay traffic packets.

***Figure 23 Link Fragmentation Interleave***




## MLPPP Feature Summary

MLPPP in Cisco IOS XR provides the same features that are supported on PPP Serial interfaces with the exception of QoS. It also provides the following additional features:

- Fragment sizes of 128, 256, and 512 bytes.
- Long sequence numbers (24-bit).
- Lost fragment detection timeout period of 80 milliseconds.
- Minimum-active-links configuration option.
- LCP echo request/reply support over multilink interface.
- Full T1 and E1 framed and unframed links.

## IPHC Over MLPPP

The 8-Port Channelized T1/E1 SPA supports IPHC over MLPPP. For more information about IPHC and how to configure it, see the “Configuring Serial Interfaces on Cisco IOS XR Software” module in the *Cisco IOS XR Interface and Hardware Component Configuration Guide for the Cisco XR 12000 Series Router.*

## T3 SONET Channels

The Cisco XR 12000 Series Router supports T3 channelized SONET on the following hardware:

- Cisco 1-Port Channelized OC-3/STM-1 SPA
- Cisco 1-Port Channelized OC-12/DS0 SPA
- Cisco 1-Port Channelized OC-48/STM-16 SPA

Channelized SONET provides the ability to transport multiple T3 channels over the same physical link.

For more detailed information about configuring channelized SONET, T3 and T1 controllers, serial interfaces, and SONET APS, see the following related modules:

- "Configuring Channelized SONET/SDH on Cisco IOS XR Software"
- "Configuring Clear Channel SONET Controllers on Cisco IOS XR Software"
- "Configuring Clear Channel T3/E3 and Channelized T3 and T1/E1 Controllers on Cisco IOS XR Software"
- "Configuring Serial Interfaces on Cisco IOS XR Sotware"

# How to Configure PPP

This section includes the following procedures:

- Modifying the Default PPP Configuration, page 523
- Configuring PPP Authentication, page 527
- Disabling an Authentication Protocol, page 535
- Configuring Multilink PPP, page 540

## Modifying the Default PPP Configuration

When you first enable PPP on an interface, the following default configuration applies:

- The interface resets itself immediately after an authentication failure.
- The maximum number of configuration requests without response permitted before all requests are stopped is 10.
- The maximum number of consecutive Configure Negative Acknowledgments (CONFNAKs) permitted before terminating a negotiation is 5.
- The maximum number of terminate requests (TermReqs) without response permitted before the Link Control Protocol (LCP) or Network Control Protocol (NCP) is closed is 2.
- Maximum time to wait for a response to an authentication packet is 10 seconds.
- Maximum time to wait for a response during PPP negotiation is 3 seconds.

This task explains how to modify the basic PPP configuration on serial and POS interfaces that have PPP encapsulation enabled. The commands in this task apply to all authentication types supported by PPP (CHAP, MS-CHAP, and PAP).

## Prerequisites

You must enable PPP encapsulation on the interface with the **encapsulation ppp** command.

- To enable PPP encapsulation on a POS interface, see the *Configuring POS Interfaces on Cisco IOS XR Software* module in this manual.
- To enable PPP encapsulation on an interface, see the *Configuring Serial Interfaces on Cisco IOS XR Sotware* module in this manual.

### SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. **ppp max-bad-auth** *retries*
4. **ppp max-configure** *retries*
5. **ppp max-failure** *retries*
6. **ppp max-terminate** *number*
7. **ppp timeout authentication** *seconds*
8. **ppp timeout retry** *seconds*
9. **end**
   or
   **commit**
10. **show ppp interfaces** {*type interface-path-id* | **all** | **brief** {*type interface-path-id* | **all** | **location** *node-id*} | **detail** {*type interface-path-id* | **all** | **location** *node-id*} | **location** *node-id*}

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | `configure`<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | **interface** *type interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface serial 0/4/0/1` | Enters interface configuration mode. |
| **Step 3** | **ppp max-bad-auth** *retries*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# ppp max-bad-auth 3` | (Optional) Configures the number of authentication retries allowed on an interface after a PPP authentication failure.<br>• If you do not specify the number of authentication retries allowed, the router resets itself immediately after an authentication failure.<br>• Replace the *retries* argument with number of retries after which the interface is to reset itself, in the range from 0 through 10.<br>• The default is 0 retries.<br>• The **ppp max-bad-auth** command applies to any interface on which PPP encapsulation is enabled. |
| **Step 4** | **ppp max-configure** *retries*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# ppp max-configure 4` | (Optional) Specifies the maximum number of configure requests to attempt (without response) before the requests are stopped.<br>• Replace the *retries* argument with the maximum number of configure requests retries, in the range from 4 through 20.<br>• The default maximum number of configure requests is 10.<br>• If a configure request message receives a reply before the maximum number of configure requests are sent, further configure requests are abandoned. |
| **Step 5** | **ppp max-failure** *retries*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# ppp max-failure 3` | (Optional) Configures the maximum number of consecutive Configure Negative Acknowledgments (CONFNAKs) permitted before a negotiation is terminated.<br>• Replace the *retries* argument with the maximum number of CONFNAKs to permit before terminating a negotiation, in the range from 2 through 10.<br>• The default maximum number of CONFNAKs is 5. |
| **Step 6** | **ppp max-terminate** *number*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# ppp max-terminate 5` | (Optional) Configures the maximum number of terminate requests (TermReqs) to send without reply before the Link Control Protocol (LCP) or Network Control Protocol (NCP) is closed.<br>• Replace the *number* argument with the maximum number of TermReqs to send without reply before closing down the LCP or NCP. Range is from 2 to 10.<br>• The default maximum number of TermReqs is 2. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 7** | **ppp timeout authentication** *seconds*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# ppp timeout authentication 20` | (Optional) Sets PPP authentication timeout parameters.<br>• Replace the *seconds* argument with the maximum time, in seconds, to wait for a response to an authentication packet. Range is from 3 to 30 seconds.<br>• The default authentication time is 10 seconds, which should allow time for a remote router to authenticate and authorize the connection and provide a response. However, it is also possible that it will take much less time than 10 seconds. In such cases, use the **ppp timeout authentication** command to lower the timeout period to improve connection times in the event that an authentication response is lost. |
| **Step 8** | **ppp timeout retry** *seconds*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# ppp timeout retry 8` | (Optional) Sets PPP timeout retry parameters.<br>• Replace the *seconds* argument with the maximum time, in seconds, to wait for a response during PPP negotiation. Range is from 1 to 10 seconds.<br>• The default is 3 seconds. |
| **Step 9** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# end`<br>or<br>`RP/0/0/CPU0:router(config-if)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 10** | **show ppp interfaces** {*type interface-path-id* \| **all** \| **brief** {*type interface-path-id* \| **all** \| **location** *node-id*} \| **detail** {*type interface-path-id* \| **all** \| **location** *node-id*} \| **location** *node-id*}<br><br>**Example:**<br>`RP/0/0/CPU0:router# show ppp interfaces serial 0/2/0/0` | Verifies the PPP configuration for an interface or for all interfaces that have PPP encapsulation enabled. |

# Configuring PPP Authentication

This section contains the following procedures:

- Enabling PAP, CHAP, and MS-CHAP Authentication, page 527
- Configuring a PAP Authentication Password, page 530
- Configuring a CHAP Authentication Password, page 532
- Configuring an MS-CHAP Authentication Password, page 534

## Enabling PAP, CHAP, and MS-CHAP Authentication

This task explains how to enable PAP, CHAP, and MS-CHAP authentication on a serial or POS interface.

## Prerequisites

You must enable PPP encapsulation on the interface with the **encapsulation ppp** command, as described in the following modules:

- To enable PPP encapsulation on a POS interface, see the *Configuring POS Interfaces on Cisco IOS XR Software* module in this manual.
- To enable PPP encapsulation on an interface, see the *Configuring Serial Interfaces on Cisco IOS XR Sotware* module in this manual.

### SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. **ppp authentication** *protocol* [*protocol* [*protocol*]] [*list-name* | **default**]
4. **end**
   or
   **commit**
5. **show ppp interfaces** {*type interface-path-id* | **all** | **brief** {*type interface-path-id* | **all** | **location** *node-id*} | **detail** {*type interface-path-id* | **all** | **location** *node-id*} | **location** *node-id*}

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| Step 1 | `configure`<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | **interface** *type interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface serial 0/4/0/1` | Enters interface configuration mode. |
| **Step 3** | **ppp authentication** *protocol* [*protocol* [*protocol*]] [*list-name* \| **default**]<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# ppp authentication chap pap MIS-access` | Enables CHAP, MS-CHAP, or PAP on an interface, and specifies the order in which CHAP, MS-CHAP, and PAP authentication is selected on the interface.<br>• Replace the *protocol* argument with **pap**, **chap**, **or ms-chap**.<br>• Replace the *list name* argument with the name of a list of methods of authentication to use. To create a list, use the **aaa authentication ppp** command, as described in the *Authentication, Authorization, and Accounting Commands on Cisco IOS XR Software* module of the *Cisco IOS XR System Security Command Reference*.<br>• If no list name is specified, the system uses the default. The default list is designated with the **aaa authentication ppp** command, as described in the *Authentication, Authorization, and Accounting Commands on Cisco IOS XR Software* module of the *Cisco IOS XR System Security Command Reference*. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# end`<br>or<br>`RP/0/0/CPU0:router(config-if)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 ppp interfaces** {*type interface-path-id* \| **all** \| **brief** {*type interface-path-id* \| **all** \| **location** *node-id*} \| **detail** {*type interface-path-id* \| **all** \| **location** *node-id*} \| **location** *node-id*}<br><br>**Example:**<br>`RP/0/0/CPU0:router# show ppp interfaces serial 0/2/0/0` | Displays PPP state information for an interface.<br>• Enter the *type interface-path-id* argument to display PPP information for a specific interface.<br>• Enter the **brief** keyword to display brief output for all interfaces on the router, for a specific interface instance, or for all interfaces on a specific node.<br>• Enter the **all** keyword to display detailed PPP information for all nodes installed in the router.<br>• Enter the **location** *node-id* keyword argument to display detailed PPP information for the designated node.<br>There are seven possible PPP states applicable for either the Link Control Protocol (LCP) or the Network Control Protocol (NCP). |

## Where To Go Next

Configure a PAP, CHAP, or MS-CHAP authentication password, as described in the appropriate section:

- If you enabled PAP on an interface, configure a PAP authentication username and password, as described in the "Configuring a PAP Authentication Password" section on page 530.
- If you enabled CHAP on an interface, configure a CHAP authentication password, as described in the "Configuring a CHAP Authentication Password" section on page 532
- If you enabled MS-CHAP on an interface, configure an MS-CHAP authentication password, as described in the "Configuring an MS-CHAP Authentication Password" section on page 534

## Configuring a PAP Authentication Password

This task explains how to enable and configure PAP authentication on a serial or POS interface.

**Note** PAP is the least secure authentication protocol available on POS and interfaces. To ensure higher security for information that is sent over POS and interfaces, we recommend configuring CHAP or MS-CHAP authentication in addition to PAP authentication.

### Prerequisites

You must enable PAP authentication on the interface with the **ppp authentication** command, as described in the "Enabling PAP, CHAP, and MS-CHAP Authentication" section on page 527.

### SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. **ppp pap sent-username** *username* **password** [**clear** | **encrypted**] *password*
4. **end**
   or
   **commit**
5. **show running-config**

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | `configure`<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | **`interface`** *`type interface-path-id`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface serial 0/4/0/1` | Enters interface configuration mode. |
| **Step 3** | **`ppp pap sent-username`** *`username`* **`password`** `[`**`clear`** `\|` **`encrypted`**`]` *`password`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# ppp pap sent-username xxxx password notified` | Enables remote Password Authentication Protocol (PAP) support for an interface, and includes the **sent-username** and **password** commands in the PAP authentication request packet to the peer.<br>• Replace the *username* argument with the username sent in the PAP authentication request.<br>• Enter **password clear** to select cleartext encryption for the password, or enter **password encrypted** if the password is already encrypted.<br>• The **ppp pap sent-username** command allows you to replace several username and password configuration commands with a single copy of this command on interfaces.<br>• You must configure the **ppp pap sent-username** command for each interface.<br>• Remote PAP support is disabled by default. |
| **Step 4** | **`end`**<br>or<br>**`commit`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# end`<br>or<br>`RP/0/0/CPU0:router(config-if)# 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)?`<br>`[cancel]:`<br>– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 running-config`**<br><br>**Example:**<br>`RP/0/0/CPU0:router# show running-config` | Verifies PPP authentication information for interfaces that have PPP encapsulation enabled. |

## Configuring a CHAP Authentication Password

This task explains how to enable CHAP authentication and configure a CHAP password on a serial or POS interface.

### Prerequisites

You must enable CHAP authentication on the interface with the **ppp authentication** command, as described in the "Enabling PAP, CHAP, and MS-CHAP Authentication" section on page 527.

### Restrictions

The same CHAP password must be configured on both host endpoints.

### SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. **ppp chap password** [**clear** | **encrypted**] *password*
4. **end**
   or
   **commit**
5. **show running-config**

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | `configure`<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | **interface** *type interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface serial 0/4/0/1` | Enters interface configuration mode. |
| **Step 3** | **ppp chap password** [**clear** \| **encrypted**] *password*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# ppp chap password clear xxxx` | Enables CHAP authentication on the specified interface, and defines an interface-specific CHAP password.<br>• Enter **clear** to select cleartext encryption, or **encrypted** if the password is already encrypted.<br>• Replace the *password* argument with a cleartext or already-encrypted password. This password is used to authenticate secure communications among a collection of routers.<br>• The **ppp chap password** command is used for remote CHAP authentication only (when routers authenticate to the peer) and does not effect local CHAP authentication.This command is useful when you are trying to authenticate a peer that does not support this command (such as a router running an older Cisco IOS XR software image).<br>• The CHAP secret password is used by the routers in response to challenges from an unknown peer. |
| **Step 4** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# end`<br>or<br>`RP/0/0/CPU0:router(config-if)# 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)?`<br>`[cancel]:`<br>– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 running-config**<br><br>**Example:**<br>`RP/0/0/CPU0:router# show running-config` | Verifies PPP authentication information for interfaces that have PPP encapsulation enabled. |

## Configuring an MS-CHAP Authentication Password

This task explains how to enable MS-CHAP authentication and configure an MS-CHAP password on a serial or POS interface.

### Prerequisites

You must enable MS-CHAP authentication on the interface with the **ppp authentication** command, as described in the "Enabling PAP, CHAP, and MS-CHAP Authentication" section on page 527.

### Restrictions

The same MS-CHAP password must be configured on both host endpoints.

### SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. **ppp ms-chap password** [**clear** | **encrypted**] *password*
4. **end**
   or
   **commit**
5. **show running-config**

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| Step 1 | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| Step 2 | **interface** *type interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface serial 0/4/0/1` | Enters interface configuration mode. |
| Step 3 | **ppp ms-chap password** [**clear** \| **encrypted**] *password*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# ppp ms-chap password clear xxxx` | Enables a router calling a collection of routers to configure a common Microsoft Challenge Handshake Authentication (MS-CHAP) secret password.<br><br>The MS-CHAP secret password is used by the routers in response to challenges from an unknown peer. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# end`<br>or<br>`RP/0/0/CPU0:router(config-if)# 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)?`<br>`[cancel]:`<br>– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 running-config**<br><br>**Example:**<br>`RP/0/0/CPU0:router# show running-config` | Verifies PPP authentication information for interfaces that have PPP encapsulation enabled. |

# Disabling an Authentication Protocol

This section contains the following procedures:

- Disabling PAP Authentication on an Interface, page 535
- Disabling CHAP Authentication on an Interface, page 537
- Disabling MS-CHAP Authentication on an Interface, page 538

## Disabling PAP Authentication on an Interface

This task explains how to disable PAP authentication on a serial or POS interface.

### SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. **ppp pap refuse**
4. **end**
   or
   **commit**
5. **show running-config**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **interface** *type interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface serial 0/4/0/1` | Enters interface configuration mode. |
| **Step 3** | **ppp pap refuse**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# ppp pap refuse` | Refuses Password Authentication Protocol (PAP) authentication from peers requesting it.<br>• If outbound Challenge Handshake Authentication Protocol (CHAP) has been configured (using the **ppp authentication** command), CHAP will be suggested as the authentication method in the refusal packet.<br>• PAP authentication is disabled by default. |
| **Step 4** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# end`<br>or<br>`RP/0/0/CPU0:router(config-if)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 running-config**<br><br>**Example:**<br>`RP/0/0/CPU0:router# show running-config` | Verifies PPP authentication information for interfaces that have PPP encapsulation enabled. |

## Disabling CHAP Authentication on an Interface

This task explains how to disable CHAP authentication on a serial or POS interface.

### SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. **ppp chap refuse**
4. **end**
   or
   **commit**
5. **show running-config**

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **`configure`**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **`interface`** *`type interface-path-id`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface serial 0/4/0/1` | Enters interface configuration mode. |
| **Step 3** | **`ppp chap refuse`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# ppp chap refuse` | Refuses CHAP authentication from peers requesting it. After you enter the **ppp chap refuse** command under the specified interface, all attempts by the peer to force the user to authenticate with the help of CHAP are refused.<br>• CHAP authentication is disabled by default.<br>• If outbound Password Authentication Protocol (PAP) has been configured (using the **ppp authentication** command), PAP will be suggested as the authentication method in the refusal packet. |

| | Command or Action | Purpose |
|---|---|---|
| Step 4 | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# end`<br>or<br>`RP/0/0/CPU0:router(config-if)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 running-config**<br><br>**Example:**<br>`RP/0/0/CPU0:router# show running-config` | Verifies PPP authentication information for interfaces that have PPP encapsulation enabled. |

## Disabling MS-CHAP Authentication on an Interface

This task explains how to disable MS-CHAP authentication on a serial or POS interface.

### SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. **ppp ms-chap refuse**
4. **end**
   or
   **commit**
5. **show running-config**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **`configure`**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **`interface`** *`type interface-path-id`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface serial 0/4/0/1` | Enters interface configuration mode. |
| **Step 3** | **`ppp ms-chap refuse`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# ppp ms-chap refuse` | Refuses MS-CHAP authentication from peers requesting it. After you enter the **ppp ms-chap refuse** command under the specified interface, all attempts by the peer to force the user to authenticate with the help of MS-CHAP are refused.<br>• MS-CHAP authentication is disabled by default.<br>• If outbound Password Authentication Protocol (PAP) has been configured (using the **ppp authentication** command), PAP will be suggested as the authentication method in the refusal packet. |
| **Step 4** | **`end`**<br>or<br>**`commit`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# end`<br>or<br>`RP/0/0/CPU0:router(config-if)# 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)?`<br>`[cancel]:`<br>– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 running-config`**<br><br>**Example:**<br>`RP/0/0/CPU0:router# show running-config` | Verifies PPP authentication information for interfaces that have PPP encapsulation enabled. |

# Configuring Multilink PPP

This section contains the following procedures:


- Prerequisites, page 540
- Restrictions, page 540
- Configuring the Controller, page 541
- Configuring the Interfaces, page 543
- Configuring MLPPP Optional Features, page 545


## Prerequisites

Before configuring MLPPP, be sure you have one of the SPAs listed here installed. Both MLPPP and LFI are supported on the following SIPs and SPAs:

- Cisco XR 12000 SIP-401
- Cisco XR 12000 SIP-501
- Cisco XR 12000 SIP-601
- Cisco 1-Port Channelized STM-1/OC-3 SPA
- Cisco 1-Port Channelized OC-12/STM-4 SPA
- Cisco 2-Port and 4-Port Channelized T3 SPAs (SPA-2XCT3/DS0, SPA-4XCT3/DS0)
- Cisco 8-Port Channelized T1/E1 SPA
- Cisco 1-Port Channelized OC-12 Line Card (Supports MLPPP only)

## Restrictions

MLPPP for Cisco IOS XR software has the following restrictions:

- Only full rate T1s are supported.
- All links in a bundle must belong to the same SPA.
- All links in a bundle must operate at the same speed.
- A maximum of 12 links per bundle is supported.
- A maximum of 28 bundles is supported on the 2-Port Channelized T3 SPA.
- A maximum of 56 bundles is supported on the 4-Port Channelized T3 SPA.
- A maximum of 224 bundles is supported per line card.
- All serial links in an MLPPP bundle inherit the value of the **mtu** command from the multilink interface. Therefore, you should not configure the **mtu** command on a serial interface before configuring it as a member of an MLPPP bundle. The Cisco IOS XR software blocks the following:
  - Attempts to configure a serial interface as a member of an MLPPP bundle if the interface is configured with a nondefault MTU value.
  - Attempts to change the **mtu** command value for a serial interface that is configured as a member of an MLPPP bundle.

In Cisco IOS XR software, multilink processing is controlled by a hardware module called the Multilink Controller, which consists of an ASIC, network processor, and CPU working in conjunction. The MgmtMultilink Controller makes the multilink interfaces behave like the serial interfaces of channelized SPAs.

## Configuring the Controller

Perform this task to configure the controller.

### SUMMARY STEPS

1. **configure**
2. **controller** *type interface-path-id*
3. **mode** *type*
4. **clock source** {**internal** | **line**}
5. **exit**
6. **controller t1** *interface-path-id*
7. **channel-group** *channel-group-number*
8. **timeslots** *range*
9. **exit**
10. **exit**
11. **controller mgmtmultilink** *interface-path-id*
12. **bundle** *bundle-id*
13. **end**
    or
    **commit**

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| Step 1 | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| Step 2 | **controller** *type interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller t3 0/1/0/0` | Enters controller configuration submode and specifies the controller name and instance identifier in *rack/slot/module/port* notation. |
| Step 3 | **mode** *type*<br><br>**Example:**<br>`RP/0/0/CPU0:router# mode t1` | Configures the type of multilinks to channelize; for example, 28 T1s. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **clock source** {**internal** \| **line**}<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# clock source internal` | (Optional) Configures the clocking for the port.<br><br>**Note** The default clock source is **internal**. |
| **Step 5** | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# exit` | Exits controller configuration mode. |
| **Step 6** | **controller t1** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller t1 0/1/0/0/1` | Enters T1 configuration mode. |
| **Step 7** | **channel-group** *channel-group-number*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1)# channel-group 0` | Creates a T1 channel group and enters channel group configuration mode for that channel group. Channel group numbers can range from 0 to 23. |
| **Step 8** | **timeslots** *range*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1-channel_group)# timeslots 1-24` | Associates one or more DS0 time slots to a channel group and creates an associated serial subinterface on that channel group.<br><br>• Range is from 1 to 24 time slots.<br><br>**Note** The time slot range must be from 1 to 24 for the resulting serial interface to be accepted into a MLPPP bundle. |
| **Step 9** | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1-channel_group)# exit` | Exits channel group configuration mode. |
| **Step 10** | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t1)# exit` | Exits T1 configuration mode and enters global configuration mode. |
| **Step 11** | **controller mgmtmultilink** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# controller mgmtmultilink 0/1/0/0` | Enters controller configuration submode for the management of multilink interfaces. Specify the controller name and instance identifier in *rack/slot/module/port* notation. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 12** | **bundle** *bundle-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-mgmtmultilink)# bundle 20` | Creates a multilink interface with the specified bundle ID. |
| **Step 13** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# end`<br>or<br>`RP/0/0/CPU0:router(config-t3)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

## Configuring the Interfaces

Perform this task to configure the interfaces.

### Restrictions

- All serial links in an MLPPP bundle inherit the value of the **mtu** command from the multilink interface. Therefore, you should not configure the **mtu** command on a serial interface before configuring it as a member of an MLPPP bundle. The Cisco IOS XR software blocks the following:
  - Attempts to configure a serial interface as a member of an MLPPP bundle if the interface is configured with a nondefault MTU value.
  - Attempts to change the **mtu** command value for a serial interface that is configured as a member of an MLPPP bundle.

### SUMMARY STEPS

1. **configure**
2. **interface multilink** *interface-path-id*
3. **ipv4 address** *address/mask*
4. **multilink fragment-size** *bytes*
5. **keepalive** {*interval* | **disable**}[*retry*]

6. **exit**
7. **interface** *type interface-path-id*
8. **encapsulation** *type*
9. **multilink group** *group-id*
10. **end**
or
**commit**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **interface multilink** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface multilink 0/1/0/0/20` | Specifies the multilink interface name and instance identifier in *rack/slot/module/port/bundle-id* notation, and enters interface configuration mode. |
| **Step 3** | **ipv4 address** *ip-address*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# ipv4 address 80.170.0.1/24` | Assigns an IP address and subnet mask to the interface in the format:<br>*A.B.C.D/prefix* or *A.B.C.D/mask* |
| **Step 4** | **multilink fragment-size** *bytes*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# multilink fragment-size 350` | (Optional) Specifies the size of the multilink fragments, such as 128 bytes. Some fragment sizes may not be supported. The default is no fragments. |
| **Step 5** | **keepalive** {*interval* \| **disable**}[*retry*]<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# keepalive disable` | Sets the keepalive timer for the channel, where:<br>• *interval*—Number of seconds (from 1 to 30) between keepalive messages. The default is 10.<br>• **disable**—Turns off the keepalive timer.<br>• *retry*—(Optional) Number of keepalive messages (from 1 to 255) that can be sent to a peer without a response before transitioning the link to the down state. The default is 3.<br>**Note** To connect with some Cisco IOS devices, multilink keepalives need to be disabled on both devices. |
| **Step 6** | **exit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# exit` | Exits interface configuration mode and enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 7** | **interface** *type interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface serial 0/1/0/0/1:0` | Specifies the interface name and instance identifier in *rack/slot/module/port/t1-number:channel-group* notation, and enters interface configuration mode. |
| **Step 8** | **encapsulation** *type*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# encapsulation ppp` | Specifies the type of encapsulation; in this case, PPP. |
| **Step 9** | **multilink group** *group-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# multilink group 20` | Specifies the multilink group ID for this interface. |
| **Step 10** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# end`<br>or<br>`RP/0/0/CPU0:router(config-t3)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

## Configuring MLPPP Optional Features

Perform this task to configure either of the following optional features:

- Minimum number of active links
- Multilink interleave

**Note** Minimum number active links must be configured at both endpoints.

## SUMMARY STEPS

1. **configure**
2. **interface multilink** *interface-path-id*
3. **multilink**
4. **ppp multilink minimum-active links** *value*
5. **multilink interleave**
6. **no shutdown**
7. **end**
   or
   **commit**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **interface multilink** *interface-path-id*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface multilink 0/1/0/0/1` | Specifies the multilink interface name and instance identifier in *rack/slot/module/port/bundle-id* notation, and enters interface configuration mode. |
| **Step 3** | **multilink**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if)# multilink` | Enters interface multilink configuration mode. |
| **Step 4** | **ppp multilink minimum-active links** *value*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if-multilink)# ppp multilink minimum-active links 12` | (Optional) Specifies the minimum number of active links for the multilink interface.<br><br>**Note** |
| **Step 5** | **multilink interleave**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if-multilink)# multilink interleave` | (Optional) Enables interleave on a multilink interface. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 6** | **no shutdown**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if-mutlilink)# no shutdown` | Removes the shutdown configuration.<br>• The removal of the shutdown configuration removes the forced administrative down on the controller, enabling the controller to move to an up or a down state. |
| **Step 7** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-t3)# end`<br>or<br>`RP/0/0/CPU0:router(config-t3)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

# Configuration Examples for PPP

This section provides the following configuration examples:

- Configuring a POS Interface with PPP Encapsulation: Example, page 547
- Configuring a Serial Interface with PPP Encapsulation: Example, page 548
- Verifying Multilink PPP Configurations, page 549

## Configuring a POS Interface with PPP Encapsulation: Example

The following example shows how to create and configure a POS interface with PPP encapsulation:

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# interface POS 0/3/0/0
RP/0/0/CPU0:router(config-if)# ipv4 address 172.18.189.38 255.255.255.224
RP/0/0/CPU0:router(config-if)# encapsulation ppp
RP/0/0/CPU0:router(config-if)# no shutdown
RP/0/0/CPU0:router(config-if)# ppp pap sent-username P1_TEST-8 password xxxx
RP/0/0/CPU0:router(config-if)# ppp authentication chap pap MIS-access
RP/0/0/CPU0:router(config-if)# ppp chap password encrypted xxxx
RP/0/0/CPU0:router(config-if)# end
Uncommitted changes found, commit them? [yes]: yes
```

The following example shows how to configure POS interface 0/3/0/1 to allow two additional retries after an initial authentication failure (for a total of three failed authentication attempts):

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# interface POS 0/3/0/1
RP/0/0/CPU0:router(config-if)# ppp max-bad-auth 3
```

## Configuring a Serial Interface with PPP Encapsulation: Example

The following example shows how to create and configure a serial interface with PPP MS-CHAP encapsulation:

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# interface serial 0/3/0/0/0:0
RP/0/0/CPU0:router(config-if)# ipv4 address 172.18.189.38 255.255.255.224
RP/0/0/CPU0:router(config-if)# encapsulation ppp
RP/0/0/CPU0:router(config-if)# no shutdown
RP/0/0/CPU0:router(config-if)# ppp authentication ms-chap MIS-access
RP/0/0/CPU0:router(config-if)# ppp ms-chap password encrypted xxxx
RP/0/0/CPU0:router(config-if)# end
Uncommitted changes found, commit them? [yes]: yes
```

## Configuring MLPPP: Example

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# controller t3 0/1/0/0/1
RP/0/0/CPU0:router# mode t1
RP/0/0/CPU0:router(config-t3)# clock source internal
RP/0/0/CPU0:router(config-t3)# exit
RP/0/0/CPU0:router(config)# controller t1 0/1/0/0/1/1
RP/0/0/CPU0:router(config-t1)# channel-group 0
RP/0/0/CPU0:router(config-t1-channel_group)# timeslots 1-24
RP/0/0/CPU0:router(config-t1-channel_group)# exit
RP/0/0/CPU0:router(config-t1)# exit
RP/0/0/CPU0:router(config)# controller mgmtmultilink 0/1/0/0
RP/0/0/CPU0:router(config-mgmtmultilink)# bundle 20
RP/0/0/CPU0:router(config-t3)# commit
RP/0/0/CPU0:router(config-t3)# exit

RP/0/0/CPU0:router(config)# interface multilink 0/1/0/0/20
RP/0/0/CPU0:router(config-if)# ipv4 address 80.170.0.1/24
RP/0/0/CPU0:router(config-if)# multilink fragment-size 128
RP/0/0/CPU0:router(config-if)# keepalive disable
RP/0/0/CPU0:router(config-if)# exit
RP/0/0/CPU0:router(config)# interface serial 0/1/0/0/1/1:0
RP/0/0/CPU0:router(config-if)# encapsulation ppp
RP/0/0/CPU0:router(config-if)# multilink group 20
RP/0/0/CPU0:router(config-t3)# commit
RP/0/0/CPU0:router(config-t3)# exit

RP/0/0/CPU0:router(config)# interface multilink 0/1/0/0/1
RP/0/0/CPU0:router(config-if)# multilink
RP/0/0/CPU0:router(config-if-multilink)# ppp multilink minimum-active links 10
RP/0/0/CPU0:router(config-if-multilink)# multilink interleave
RP/0/0/CPU0:router(config-if-mutlilink)# no shutdown
RP/0/0/CPU0:router(config-t3)# commit
```

# Verifying Multilink PPP Configurations

Use the following show commands to verify and troubleshoot your multilink configurations:

- show multilink interfaces: Examples, page 549
- show ppp interfaces multilink: Example, page 551
- show ppp interface serial: Example, page 552
- show imds interface multilink: Example, page 552

## show multilink interfaces: Examples

```
RP/0/0/CPU0:Router# show multilink interfaces Serial 0/4/3/1/10:0
Mon Sep 21 09:24:19.604 UTC

Serial0/4/3/1/10:0 is up, line protocol is up
      Encapsulation: PPP
      Multilink group id: 6
      Member status: ACTIVE

RP/0/0/CPU0:Router# show multilink interfaces Multilink 0/4/3/0/3
Mon Sep 21 09:17:12.131 UTC

Multilink0/4/3/0/3 is up, line protocol is up
  Fragmentation: disabled
  Interleave: disabled
  Encapsulation: PPP
  Member Links: 1 active, 1 inactive
    - Serial0/4/3/1/5:0 is up, line protocol is up
      Encapsulation: PPP
      Multilink group id: 3
      Member status: ACTIVE

  - Serial0/4/3/1/6:0 is administratively down, line protocol is administratively down
Encapsulation: PPP
      Multilink group id: 3
      Member status: INACTIVE : LCP has not been negotiated

  Fragmentation Statistics
  Input Fragmented packets 0            Input Fragmented bytes 0
  Output Fragmented packets 0           Output Fragmented bytes 0
  Input Unfragmented packets 0          Input Unfragmented bytes 0
  Output Unfragmented packets 0         Output Unfragmented bytes 0
  Input Reassembled packets 0           Input Reassembled bytes 0

RP/0/5/CPU0:Mav-IOX-Rahul#sho multilink interfaces Serial 0/4/3/1/10:0
Mon Sep 21 09:24:19.604 UTC

Serial0/4/3/1/10:0 is up, line protocol is up
      Encapsulation: PPP
      Multilink group id: 6
      Member status: ACTIVE

RP/0/0/CPU0:Router# show multilink interfaces
Mon Sep 21 09:15:10.679 UTC

Multilink0/4/3/0/1 is up, line protocol is up
  Fragmentation: disabled
  Interleave: disabled
  Encapsulation: FR
```

```
  Member Links: 1 active, 1 inactive
   - Serial0/4/3/1/2:0: INACTIVE : Down  (Member link idle)
   - Serial0/4/3/1/1:0: ACTIVE : Up


Multilink0/4/3/0/10 is up, line protocol is down
  Fragmentation: disabled
  Interleave: disabled
  Encapsulation: PPP
  Member Links: 0 active, 0 inactive
  Fragmentation Statistics
  Input Fragmented packets 0            Input Fragmented bytes 0
  Output Fragmented packets 0           Output Fragmented bytes 0
  Input Unfragmented packets 0          Input Unfragmented bytes 0
  Output Unfragmented packets 0         Output Unfragmented bytes 0
  Input Reassembled packets 0           Input Reassembled bytes 0

Multilink0/4/3/0/100 is administratively down, line protocol is administratively down
  Fragmentation: disabled
  Interleave: disabled
  Encapsulation: PPP
  Member Links: 0 active, 0 inactive
  Fragmentation Statistics
  Input Fragmented packets 0            Input Fragmented bytes 0
  Output Fragmented packets 0           Output Fragmented bytes 0
  Input Unfragmented packets 0          Input Unfragmented bytes 0
  Output Unfragmented packets 0         Output Unfragmented bytes 0
  Input Reassembled packets 0           Input Reassembled bytes 0

Multilink0/4/3/0/2 is up, line protocol is up
  Fragmentation: disabled
  Interleave: disabled
  Encapsulation: FR
  Member Links: 2 active, 0 inactive
   - Serial0/4/3/1/4:0: ACTIVE : Up
   - Serial0/4/3/1/3:0: ACTIVE : Up


Multilink0/4/3/0/3 is up, line protocol is up
  Fragmentation: disabled
  Interleave: disabled
  Encapsulation: PPP
  Member Links: 1 active, 1 inactive
   - Serial0/4/3/1/5:0: ACTIVE
   - Serial0/4/3/1/6:0: INACTIVE : LCP has not been negotiated
  Fragmentation Statistics
  Input Fragmented packets 0            Input Fragmented bytes 0
  Output Fragmented packets 0           Output Fragmented bytes 0
  Input Unfragmented packets 0          Input Unfragmented bytes 0
  Output Unfragmented packets 0         Output Unfragmented bytes 0
  Input Reassembled packets 0           Input Reassembled bytes 0

Multilink0/4/3/0/4 is up, line protocol is up
  Fragmentation: disabled
  Interleave: disabled
  Encapsulation: PPP
  Member Links: 2 active, 0 inactive
   - Serial0/4/3/1/8:0: ACTIVE
   - Serial0/4/3/1/7:0: ACTIVE
  Fragmentation Statistics
  Input Fragmented packets 0            Input Fragmented bytes 0
  Output Fragmented packets 0           Output Fragmented bytes 0
  Input Unfragmented packets 0          Input Unfragmented bytes 0
  Output Unfragmented packets 0         Output Unfragmented bytes 0
```

```
  Input Reassembled packets 0           Input Reassembled bytes 0

Multilink0/4/3/0/5 is up, line protocol is up
  Fragmentation: disabled
  Interleave: enabled
  Encapsulation: PPP
  Member Links: 1 active, 0 inactive
   - Serial0/4/3/1/9:0: ACTIVE
  Fragmentation Statistics
  Input Fragmented packets 0            Input Fragmented bytes 0
  Output Fragmented packets 0           Output Fragmented bytes 0
  Input Unfragmented packets 0          Input Unfragmented bytes 0
  Output Unfragmented packets 0         Output Unfragmented bytes 0
  Input Reassembled packets 0           Input Reassembled bytes 0

Multilink0/4/3/0/6 is up, line protocol is up
  Fragmentation: disabled
  Interleave: enabled
  Encapsulation: PPP
  Member Links: 1 active, 0 inactive
   - Serial0/4/3/1/10:0: ACTIVE
  Fragmentation Statistics
  Input Fragmented packets 0            Input Fragmented bytes 0
  Output Fragmented packets 0           Output Fragmented bytes 0
  Input Unfragmented packets 0          Input Unfragmented bytes 0
  Output Unfragmented packets 0         Output Unfragmented bytes 0
  Input Reassembled packets 0           Input Reassembled bytes 0

Multilink0/4/3/0/7 is up, line protocol is down
  Fragmentation: disabled
  Interleave: enabled
  Encapsulation: PPP
  Member Links: 0 active, 1 inactive
   - Serial0/4/3/1/11:0: INACTIVE : LCP has not been negotiated
  Fragmentation Statistics
  Input Fragmented packets 0            Input Fragmented bytes 0
  Output Fragmented packets 0           Output Fragmented bytes 0
  Input Unfragmented packets 0          Input Unfragmented bytes 0
  Output Unfragmented packets 0         Output Unfragmented bytes 0
  Input Reassembled packets 0           Input Reassembled bytes 0

Multilink0/4/3/0/8 is up, line protocol is down
  Fragmentation: disabled
  Interleave: enabled
  Encapsulation: PPP
  Member Links: 0 active, 1 inactive
   - Serial0/4/3/1/12:0: INACTIVE : LCP has not been negotiated
  Fragmentation Statistics
  Input Fragmented packets 0            Input Fragmented bytes 0
  Output Fragmented packets 0           Output Fragmented bytes 0
  Input Unfragmented packets 0          Input Unfragmented bytes 0
  Output Unfragmented packets 0         Output Unfragmented bytes 0
  Input Reassembled packets 0           Input Reassembled bytes 0
```

## show ppp interfaces multilink: Example

```
RP/0/0/CPU0:Router# show ppp interfaces multilink 0/3/1/0/1

Multilink 0/3/1/0/1 is up, line protocol is up
LCP: Open
     Keepalives disabled
```

```
  IPCP: Open
     Local IPv4 address: 1.1.1.2
     Peer IPv4 address:  1.1.1.1
  Multilink
     Member Links: 2 active, 1 inactive (min-active 1)
      - Serial0/3/1/0/0:0: ACTIVE
      - Serial0/3/1/0/1:0: ACTIVE
      - Serial0/3/1/0/2:0: INACTIVE : LCP has not been negotiated
```

## show ppp interface serial: Example

```
RP/0/0/CPU0:Router# show ppp interface Serial 0/3/1/0/0:0

Serial 0/3/1/0/0:0 is up, line protocol is up
  LCP: Open
     Keepalives disabled
     Local MRU: 1500 bytes
     Peer  MRU: 1500 bytes
     Local Bundle MRRU: 1596 bytes
     Peer  Bundle MRRU: 1500 bytes
     Local Endpoint Discriminator: 1b61950e3e9ce8172c8289df0000003900000001
     Peer  Endpoint Discriminator: 7d046cd8390a4519087aefb90000003900000001
  Authentication
     Of Peer: <None>
     Of Us:   <None>
  Multilink
     Multilink group id: 1
     Member status: ACTIVE
```

## show imds interface multilink: Example

```
RP/0/0/CPU0:Router# show imds interface Multilink 0/3/1/0/1

IMDS INTERFACE DATA (Node 0x0)

Multilink0_3_1_0_1 (0x04001200)
-----------------------
flags: 0x0001002f    type: 55 (IFT_MULTILINK)    encap: 52 (ppp)
state: 3 (up)     mtu: 1600    protocol count: 3
control parent: 0x04000800     data parent: 0x00000000
       protocol         capsulation           state        mtu
    --------------- -------------------- --------------- --------
    12 (ipv4)
                    26 (ipv4)            3 (up)          1500
                    47 (ipcp)            3 (up)          1500
    16 (ppp_ctrl)
                    53 (ppp_ctrl)        3 (up)          1500
    0 (Unknown)
                    139 (c_shim)         3 (up)          1600
                    52 (ppp)             3 (up)          1504
                    56 (queue_fifo)      3 (up)          1600
                    60 (txm_nopull)      3 (up)          1600
```

# Additional References

These sections provide references related to PPP encapsulation.

## Related Documents

| Related Topic | Document Title |
|---|---|
| Cisco IOS XR master command reference | *Cisco IOS XR Master Commands List* |
| Cisco IOS XR interface configuration commands | *Cisco IOS XR Interface and Hardware Component Command Reference* |
| Initial system bootup and configuration information for a router using Cisco IOS XR software | *Cisco IOS XR Getting Started Guide* |
| Cisco IOS XR AAA services configuration information | *Cisco IOS XR System Security Configuration Guide* and *Cisco IOS XR System Security Command Reference* |

## 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 |
|---|---|
| No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature | To locate and download MIBs for selected platforms using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL:<br>http://www.cisco.com/go/mibs |

## RFCs

| RFCs | Title |
|---|---|
| RFC-1661 | *The Point-to-Point Protocol (PPP)* |
| RFC- 1994 | *PPP Challenge Handshake Authentication Protocol (CHAP)* |

# 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/support |

# Configuring 802.1Q VLAN Interfaces on Cisco IOS XR Software

This module describes the configuration and management of 802.1Q VLAN interfaces on the Cisco XR 12000 Series Routers.

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 on Cisco IOS XR Software**

| Release | Modification |
|---|---|
| Release 3.2 | This feature was introduced on the Cisco XR 12000 Series Router. |
| Release 3.3.0 | • Support was added for VLAN commands on bundled Ethernet interfaces.<br>• |
| Release 3.4.0 | • The Layer 2 Virtual Private Network (L2VPN) feature was first supported on Ethernet interfaces on the Cisco XR 12000 Series Router.<br>• Support was added on for the 8-Port 1-Gigabit Ethernet SPA. |

# Contents


- Prerequisites for Configuring 802.1Q VLAN Interfaces, page 556
- Information About Configuring 802.1Q VLAN Interfaces, page 556
- How to Configure 802.1Q VLAN Interfaces, page 558
- Configuration Examples for VLAN Interfaces, page 566
- Additional References, page 568

# 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 Gigabit Ethernet interface, 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, page 556
- 802.1Q Tagged Frames, page 556
- Subinterfaces, page 557
- Subinterface MTU, page 557
- Native VLAN, page 557
- Layer 2 VPN on VLANs, page 557

## 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 Gigabit Ethernet, 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

The Cisco XR 12000 Series Router does not support a native VLAN.

## Layer 2 VPN on VLANs

The Layer 2 Virtual Private Network (L2VPN) feature enables Service Providers (SPs) to provide layer 2 services to geographically disparate customer sites.

The configuration model for configuring VLAN attachment circuits (ACs) is similar to the model used for configuring basic VLANs, where the user first creates a VLAN subinterface, and then configures that VLAN in subinterface configuration mode. To create an AC, you need to include the **l2transport** keyword in the **interface** command string to specify that the interface is a Layer 2 interface.

VLAN ACs support three modes of L2VPN operation:

- Basic Dot1Q AC—The AC covers all frames that are received and sent with a specific VLAN tag.
- Q-in-Q AC—The AC covers all frames received and sent with a specific outer VLAN tag and a specific inner VLAN tag. Q-in-Q is an extension to Dot1Q that uses a stack of two tags.
- Q-in-Any AC—The AC covers all frames received and sent with a specific outer VLAN tag and any inner VLAN tag, as long as that inner VLAN tag is not L3 terminated. Q-in-Any is an extension to Q-in-Q that uses wildcarding to match any second tag.

**Note** The Q-in-Any mode is a variation of the basic Dot1Q mode. In Q-in-Any mode, the frames have a basic Q-in-Q encapsulation; however, in Q-in-Any mode the inner tag is not relevant, except for the fact that a few specific inner VLAN tags are siphoned for specific services. For example, a tag may be used to provide L3 services for general internet access.

Each VLAN on a CE-to-PE link can be configured as a separate L2VPN connection (using either VC type 4 or VC type 5). To configure L2VPN on VLANs, see the "Configuring an Attachment Circuit on a VLAN" section on page 561.

Keep the following in mind when configuring L2VPN on a VLAN:

- Cisco IOS XR software supports 4k ACs per LC.
- In a point-to-point connection, the two ACs do not have to be of the same type. For example, a port mode Ethernet AC can be connected to a Dot1Q Ethernet AC.
- Pseudo-wires can run in VLAN mode or in port mode. A pseudo-wire running in VLAN mode has a single Dot1Q tag, while a pseudo-wire running in port mode has no tags. Some interworking is required to connect these different types of circuits together. This interworking takes the form of popping, pushing and rewriting tags. The advantage of Layer 2 VPN is that is simplifies the interworking required to connect completely different media types together.
- The ACs on either side of an MPLS pseudo-wire can be different types. In this case, the appropriate conversion is carried out at one or both ends of the AC to pseudo-wire connection.

Use the **show interfaces** command to display AC and pseudo-wire information.

**Note** For detailed information about configuring an L2VPN network, see the *Implementing MPLS Layer 2 VPNs* module of the *Cisco IOS XR Multiprotocol Label Switching Configuration Guide.*

# How to Configure 802.1Q VLAN Interfaces

This section contains the following procedures:


- Configuring 802.1Q VLAN Subinterfaces, page 558
- Configuring an Attachment Circuit on a VLAN, page 561
- Configuring an Attachment Circuit on a VLAN, page 561
- Removing an 802.1Q VLAN Subinterface, page 564


## Configuring 802.1Q VLAN Subinterfaces

This task explains how to configure 802.1Q VLAN subinterfaces. To remove these subinterfaces, see the "Removing an 802.1Q VLAN Subinterface" section of this module.

### SUMMARY STEPS

1. **configure**
2. **interface** {**GigabitEthernet** | **TenGigE** | **fastethernet** | **Bundle-Ether**} *interface-path-id.subinterface*
3. **dot1q vlan** *vlan-id*
4. **ipv4 address** *ip-address mask*
5. **exit**
6. Repeat Step 2 through Step 5 to define the rest of the VLAN subinterfaces.
7. **end**
   or
   **commit**

8. **show vlan interface** [{**GigabitEthernet** | **TenGigE** | **Bundle-Ether** | **fastethernet**} *interface-path-id*] [**location** *instance*]
9. **show vlan trunks** [**brief**] [**location** *instance*] [{**GigabitEthernet** | **TenGigE** | **Bundle-Ether** | **fastethernet**} *interface-path-id*] [**summary**]

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | `configure`<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **`interface`** {**`GigabitEthernet`** \| **`TenGigE`** \| **`Bundle-Ether`** \| **`fastethernet`**} *`interface-path-id.subinterface`*<br><br>**Example:**<br>`RP/0/0/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 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** | **`dot1q vlan`** *`vlan-id`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-subif)# dot1q vlan 100` | 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`** *`vlan-id`*<br>• To configure a Q-in-Q AC, use the following syntax:<br>**`dot1q vlan`** *`vlan-id`* **`vlan`** *`vlan-id`* |

| | Command or Action | Purpose |
|---|---|---|
| Step 4 | **ipv4 address** *ip-address mask*<br><br>**Example:**<br>`RP/0/0/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. |
| Step 5 | **exit**<br><br>**Example:**<br>`RP/0/0/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 | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# end`<br>or<br>`RP/0/0/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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

| | Command or Action | Purpose |
|---|---|---|
| Step 8 | **show vlan interface** [*type interface-path-id*][**location** *instance*]<br><br>**Example:**<br>`RP/0/0/CPU0:router# show vlan interface 5` | (Optional) Displays the interface configuration.<br>• To display the configuration for a particular port, use the **location** keyword.<br>• To display the configuration for the specified interface or subinterface, use the **interface** keyword. |
| Step 9 | **show vlan trunks** [**brief**] [**location** *instance*] [{**GigabitEthernet** \| **TenGigE** \| **Bundle-Ether** \| **fastethernet**} *interface-path-id*] [**summary**]<br><br>**Example:**<br>`RP/0/0/CPU0:router# show vlan trunk summary` | (Optional) Displays summary information about each of the VLAN trunk interfaces.<br>• The keywords have the following meanings:<br>– **brief—**Displays a brief summary.<br>– **summary—**Displays a full summary.<br>– **location—**Displays information about the VLAN trunk interface on the given port.<br>– **interface—**Displays information about the specified interface or subinterface. |

# 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** | **fastethernet** | **Bundle-Ether**] *interface-path-id.subinterface* **l2transport**
3. **dot1q vlan** *vlan-id*
4. **l2protocol** {**cdp** | **pvst** | **stp** | **vtp**} {[**forward** | **tunnel**] [**experimental** *bits*] | **drop**}
5. **end**
   or
   **commit**
6. **show interfaces** [**GigabitEthernet** | **TenGigE**] *interface-path-id*

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure terminal` | Enters global configuration mode. |
| **Step 2** | **interface** [**GigabitEthernet** \| **TenGigE** \| **Bundle-Ether** \| **TenGigE**] *interface-path*] *id.subinterface* **l2transport**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# interface TenGigE 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><br>**Example:**<br>`RP/0/0/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** *vlan-id*<br>• To configure a Q-in-Q AC, use the following syntax:<br>**dot1q vlan** *vlan-id* **vlan** *vlan-id*<br>• To configure a Q-in-Any AC, use the following syntax:<br>**dot1q vlan** *vlan-id* **vlan any**<br>**Note** |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | `l2protocol {cdp \| pvst \| stp \| vtp}{[forward \| tunnel][experimental bits]\|drop}`<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if-l2)# l2protocol stp tunnel` | 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** | `end`<br>or<br>`commit`<br><br>**Example:**<br>`RP/0/0/CPU0:router(config-if-l2)# end`<br>or<br>`RP/0/0/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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 6** | `show interfaces [GigabitEthernet \| TenGigE]` *interface-path-id.subinterface*<br><br>**Example:**<br>`RP/0/0/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 *Cisco IOS XR 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 Cisco IOS XR software 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" section in this module.

### SUMMARY STEPS

1. **configure**
2. **no interface** {**GigabitEthernet** | **TenGigE** | **fastethernet** | **Bundle-Ether**] *interface-path-id.subinterface*
3. Repeat Step 2 to remove other VLAN subinterfaces.
4. **end**
   or
   **commit**
5. **show vlan interface** [*type interface-path-id*] [**location** *instance*]
6. **show vlan trunks** [**brief**] [**location** *instance*] [{**GigabitEthernet** | **TenGigE** | **Bundle-Ether** | **fastethernet**} *interface-path-id*] [**summary**]

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **`configure`**<br><br>**Example:**<br>`RP/0/0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **`no interface`** {**`GigabitEthernet`** \| **`TenGigE`** \| **`fastethernet`** \| **`Bundle-Ether`**} *`interface-path-id.subinterface`*<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# no interface TenGigE 0/2/0/4.10` | Removes the subinterface, which also automatically deletes all the configuration applied to the subinterface.<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. |
| **Step 3** | Repeat Step 2 to remove other VLAN subinterfaces. | — |
| **Step 4** | **`end`**<br>or<br>**`commit`**<br><br>**Example:**<br>`RP/0/0/CPU0:router(config)# end`<br>or<br>`RP/0/0/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)?`<br>`[cancel]:`<br>– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **show vlan interface** [{**GigabitEthernet** \| **TenGigE** \| **Bundle-Ether**} *interface-path-id* \| **location** *instance*]<br><br>**Example:**<br>`RP/0/0/CPU0:router# show vlan trunk summary` | (Optional) Displays the interface configuration.<br>• To display the configuration for a port, use the **location** keyword.<br>• To display the configuration for the specified interface or subinterface, use the **interface** keyword. |
| **Step 6** | **show vlan trunks** [**brief**] [**location** *instance*] [{**GigabitEthernet** \| **TenGigE** \| **Bundle-Ether** \| **fastethernet**} *interface-path-id*] [**summary**]<br><br>**Example:**<br>`RP/0/0/CPU0:router# show vlan trunk summary` | (Optional) Displays summary information about each of the VLAN trunk interfaces.<br>• The keywords have the following meanings:<br>– **brief**—Displays a brief summary.<br>– **summary**—Displays a full summary.<br>– **location**—Displays information about the VLAN trunk interface on the given port.<br>– **interface**—Displays information about the specified interface or subinterface. |

# Configuration Examples for VLAN Interfaces

This section contains the following example:

VLAN Subinterfaces: Example, page 566

## VLAN Subinterfaces: Example

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

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# interface TenGigE 0/2/0/4.1
RP/0/0/CPU0:router(config-subif)# dot1q vlan 10
RP/0/0/CPU0:router(config-subif)# ipv4 address 10.0.10.1/24
RP/0/0/CPU0:router(config-subif)# interface TenGigE0/2/0/4.2
RP/0/0/CPU0:router(config-subif)# dot1q vlan 20
RP/0/0/CPU0:router(config-subif)# ipv4 address 10.0.20.1/24
RP/0/0/CPU0:router(config-subif)# interface TenGigE0/2/0/4.3
RP/0/0/CPU0:router(config-subif)# dot1q vlan 30
RP/0/0/CPU0:router(config-subif)# ipv4 address 10.0.30.1/24
RP/0/0/CPU0:router(config-subif)# commit
RP/0/0/CPU0:router(config-subif)# exit
RP/0/0/CPU0:router(config)# exit

RP/0/0/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/0/0/CPU0:router# show vlan interface
interface           encapsulation  vlan-id  intf-state
Te0/2/0/4.1         802.1Q              10  up
```

```
Te0/2/0/4.2         802.1Q              20  up
Te0/2/0/4.3         802.1Q              30  up


RP/0/0/CPU0:router# show vlan trunks brief
interface           encapsulations        intf-state
Te0/2/0/4           802.1Q (Ether)        up
```

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

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

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

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

The following example shows how to create a Q-in-Q AC:

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# interface GigabitEthernet 0/0/0/0.2
RP/0/0/CPU0:router(config-subif)# l2transport
RP/0/0/CPU0:router(config-subif)# dot1q vlan 20 vlan 10
RP/0/0/CPU0:router(config-subif)# commit
RP/0/0/CPU0:router(config-subif)# exit
RP/0/0/CPU0:router(config)# exit
```

The following example shows how to create a Q-in-Any AC:

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# interface GigabitEthernet 0/0/0/0.3
RP/0/0/CPU0:router(config-subif)# l2transport
RP/0/0/CPU0:router(config-subif)# dot1q vlan 30 vlan any
RP/0/0/CPU0:router(config-subif)# commit
RP/0/0/CPU0:router(config-subif)# exit
RP/0/0/CPU0:router(config)# exit
```

# Additional References

These sections provide references related to VLAN interface configuration.

## Related Documents

| Related Topic | Document Title |
|---|---|
| Cisco IOS XR master command reference | *Cisco IOS XR Master Commands List* |
| Cisco IOS XR interface configuration commands | *Cisco IOS XR Interface and Hardware Component Command Reference* |
| Initial system bootup and configuration information for a router using the Cisco IOS XR Software | *Cisco IOS XR Getting Started Guide* |
| Information about user groups and task IDs | *Cisco IOS XR Interface and Hardware Component Command Reference* |

## 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 this URL:<br>http://www.cisco.com/go/mibs |

## RFCs

a

| 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/support |

# Configuring Tunnel Interfaces on Cisco IOS XR Software

This module describes the configuration of Tunnel-IPSec interfaces on the Cisco XR 12000 Series Routers.

Tunnel interfaces are virtual interfaces that provide encapsulation of arbitrary packets within another transport protocol. The Tunnel-IPSec interface provides secure communications over otherwise unprotected public routes.

A virtual interface represents a logical packet switching entity within the router. Virtual Interfaces have a global scope and do not have an associated location. The Cisco IOS XR Software uses the *rack/slot/module/port* notation for identifying physical interfaces, but uses a globally unique numerical ID after the interface name to identify virtual interfaces. Examples of this numerical ID are Loopback 0, Loopback 1, and Null99999. The ID is unique for each virtual interface type so you may simultaneously have a Loopback 0 and a Null 0.

Virtual interfaces have their control plane presence on the active route processor (RP). The configuration and control plane are mirrored onto the standby RP and, in the event of a switchover, the virtual interfaces will move to the standby, which then becomes the newly active RP.

**Note** Subinterfaces can be physical or virtual, depending on their parent interface.

Virtual tunnels are *configured* on any RP  or distributed RP (DRP), but they are created and operate only from the RP.

**Note** Tunnels do not have a one-to-one modular services card association.

**Feature History for Configuring Tunnel Interfaces on Cisco IOS XR Software**

| Release | Modification |
|---|---|
| Release 3.2 | This feature was introduced on the Cisco XR 12000 Series Router. |

# Contents


- Prerequisites for Configuring Tunnel Interfaces, page 572
- Information About Configuring Tunnel Interfaces, page 572

- How to Configure Tunnel Interfaces, page 574
- Configuration Examples for Tunnel Interfaces, page 576
- Where to Go Next, page 577
- Additional References, page 577


# Prerequisites for Configuring Tunnel 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.

# Information About Configuring Tunnel Interfaces

To implement tunnel interfaces, you must understand the following concepts:


- Tunnel Interfaces Overview, page 572
- Virtual Interface Naming Convention, page 572
- Tunnel-IPSec Overview, page 573
- Tunnel-IPSec Naming Convention, page 573
- Crypto Profile Sets, page 573
- How to Configure Tunnel Interfaces, page 574


## Tunnel Interfaces Overview

Tunneling provides a way to encapsulate arbitrary packets inside of a transport protocol. This feature is implemented as a virtual interface to provide a simple interface for configuration. The tunnel interfaces are not tied to specific “passenger” or “transport” protocols, but, rather, they represent an architecture that is designed to provide the services necessary to implement any standard point-to-point encapsulation scheme. Because supported tunnels are point-to-point links, you must configure a separate tunnel for each link.

There are three necessary steps in configuring a tunnel interface:

1. Specify the tunnel interface—**interface tunnel-ipsec** *identifier*.
2. Configure the tunnel source—**tunnel source** {*ip-address* | *interface-id*}.
3. Configure the tunnel destination—**tunnel destination** {*ip-address* | *tunnel-id*}.

## Virtual Interface Naming Convention

Virtual interface names never use the physical interface naming notation *rack/slot/module/port* for identifying an interface’s rack, slot, module, and port, because they are not tied to any physical interface or subinterface.

Virtual interfaces use a globally unique numerical identifier (per virtual interface type).

Examples of naming notation for virtual interfaces:

```
Interface                        IP-Address      Status  Protocol
Loopback0                        10.9.0.0         Up             Up
Loopback10                       10.7.0.0         Up             Up
Tunnel-TE5000                    172.18.189.38    Down           Down
Null10                           10.8.0.0          Up             Up
```

# Tunnel-IPSec Overview

IPSec (IP security) is a framework of open standards for ensuring secure private communications over the Internet. It can be used to support Virtual Private Network (VPN), firewalls, and other applications that must transfer data across a public or insecure network. The router IPSec protocol suite provides a set of standards that are used to provide privacy, integrity, and authentication service at the IP layer. The IPSec protocol suite also includes cryptographic techniques to support the key management requirements of the network-layer security.

When IPSec is used, there is no need to use Secure Shell (SSH) or Secure Socket Layer (SSL). Their use causes the same data to be encrypted or decrypted twice, which creates unnecessary overhead. The IPSec daemon is running on both the RPs and the DRPs. IPSec is an optional feature on the router. IPSec is a good choice for a user who has multiple applications that require secure transport. On the client side, customers can use “Cisco VPN 3000 Client” or any other third-party IPSec client software to build IPSec VPN.

**Note** IPSec tunnel  exists in the control plane, so you do not have to bring up or bring down the tunnel. Entry into the IPSec tunnel is only for locally sourced traffic from the RP or DRP, and is dictated by the access control lists (ACL) configured as a part of the profile that is applied to the Tunnel-IPSec.

# Tunnel-IPSec Naming Convention

A profile is entered from interface configuration submode for interface tunnel-ipsec. For example:

```
interface tunnel-ipsec 30
     profile <profile name>
```

# Crypto Profile Sets

Crypto profile sets must be configured and applied to tunnel interfaces (or to the crypto IPSec transport). For details on using the crypto IPSec transport, refer to the link provided in the “Additional References” section on page 577. For IPSec to succeed between two IPSec peers, the crypto profile entries of both peers must contain compatible configuration statements.

Two peers that try to establish a security association must each have at least one crypto profile entry that is compatible with one of the other peer's crypto profile entries. For two crypto profile entries to be compatible, they must at least meet the following criteria:

- They must contain compatible crypto access lists. In the case where the responding peer is using dynamic crypto profiles, the entries in the local crypto access list must be “permitted” by the peer's crypto access list.
- They must each identify the other peer (unless the responding peer is using dynamic crypto profiles).
- They must have at least one transform set in common.

**Note** Crypto profiles cannot be shared; that is, the same profile cannot be attached to multiple interfaces.

# How to Configure Tunnel Interfaces

This section contains the following procedures:

- Configuring Tunnel-IPSec Interfaces, page 574 (Required)

## Configuring Tunnel-IPSec Interfaces

This task explains how to configure Tunnel-IPSec interfaces.

### Prerequisites

To use the **profile** command, you must be in a user group associated with a task group that includes the proper task IDs for crypto commands. To use the **tunnel destination** command, you must be in a user group associated with a task group that includes the proper task IDs for interface commands.

For detailed information about user groups and task IDs, see the *Configuring AAA Services on Cisco IOS XR Software* module of *Cisco IOS XR System Security Configuration Guide*.

The following tasks are required for creating Tunnel-IPSec interfaces:

- Setting Global Lifetimes for IPSec Security Associations
- Configuring Checkpointing
- Configuring Crypto Profiles

For detailed information on configuring the prerequisite checkpointing and crypto profiles, and setting the global lifetimes for IPSec security associations, refer to the *Implementing IPSec Network Security on Cisco IOS XR Software* module in *Cisco IOS XR System Security Configuration Guide*.

After configuring crypto profiles, you must apply a crypto profile to each tunnel interface through which IPSec traffic will flow. Applying the crypto profile set to a tunnel interface instructs the router to evaluate all the interface's traffic against the crypto profile set and to use the specified policy during connection or security association negotiation on behalf of traffic to be protected by crypto.

#### SUMMARY STEPS

1. **configure**
2. **interface tunnel-ipsec** *identifier*
3. **profile** *profile-name*
4. **tunnel source** {*ip-address* | *interface-id*}
5. **tunnel destination** {*ip-address* | *tunnel-id*}
6. **end**
   or
   **commit**
7. **show ip route**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| Step 1 | **configure**<br><br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| Step 2 | **interface tunnel-ipsec** *identifier*<br><br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# interface tunnel-ipsec 30` | Identifies the IPSec interface to which the crypto profile will be attached and enters interface configuration mode. |
| Step 3 | **profile** *profile-name*<br><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if)# profile user1` | Assigns the crypto profile name to be applied to the tunnel for IPSec processing.<br>• The same crypto profile cannot be shared in different IPSec modes. |
| Step 4 | **tunnel source** (*ip-address* \| *interface-id*)<br><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if)# tunnel source Ethernet0/1/1/2` | Specifies the tunnel source IP address or interface ID.<br>• This command is required for both static and dynamic profiles. |
| Step 5 | **tunnel destination** {*ip-address* \| *tunnel-id*}<br><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if)# tunnel destination 192.168.164.19` | (Optional) Specifies the tunnel destination IP address.<br>• This command is not required if the profile is dynamic. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 6** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if)# end`<br>or<br>`RP/0/RP0/CPU0:router(config-if)# 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 EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to 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 7** | **show ip route**<br><br>**Example:**<br>`RP/0/RP0/CPU0:router# show ip route` | Displays forwarding information for the tunnel.<br>• The command **show ip route** displays what was advertised and shows the routes for static and autoroute. |

# Configuration Examples for Tunnel Interfaces

This section contains the following example:

Tunnel-IPSec: Example, page 576

## Tunnel-IPSec: Example

This example shows the process of creating and applying a profile to an IPSec tunnel. The necessary preliminary steps are also shown. You must first define a transform set and then create a profile before configuring the IPSec tunnel.

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# crypto ipsec transform-set tset1
RP/0/RP0/CPU0:router(config-transform-set tset1)# transform esp-sha-hmac
RP/0/RP0/CPU0:router(config-transform-set tset1)# end
Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]: yes

RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# crypto ipsec profile user1
RP/0/RP0/CPU0:router(config-user1)# match sampleac1 transform-set tset1
RP/0/RP0/CPU0:router(config-user1)# set pfs group5
RP/0/RP0/CPU0:router(config-user1)# set type dynamic
RP/0/RP0/CPU0:router(config-user1)# exit
```

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface tunnel-ipsec 30
RP/0/RP0/CPU0:router(config-if)# profile user1
RP/0/RP0/CPU0:router(config-if)# tunnel source MgmtEth 0/RP0/CPU0/0
RP/0/RP0/CPU0:router(config-if)# tunnel destination 192.168.164.19
RP/0/RP0/CPU0:router(config-if)# end
Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]: yes
```

# Where to Go Next

You now must apply a crypto profile to each transport. Applying the crypto profile set to a transport instructs the router to evaluate all the interface's traffic against the crypto profile set and to use the specified policy during connection or security association negotiation on behalf of traffic to be protected by crypto.

For information on applying a crypto profile to each transport, see the *Implementing IPSec Network Security on Cisco IOS XR Software* module of *Cisco IOS XR System Security Configuration Guide.*

# Additional References

These sections provide references related to tunnel interface configuration.

## Related Documents

| Related Topic | Document Title |
|---|---|
| Cisco IOS XR master command reference | *Cisco IOS XR Master Commands List* |
| Cisco IOS XR interface configuration commands | *Cisco IOS XR Interface and Hardware Component Command Reference* |
| Information about IPSec and crypto profiles | *Cisco IOS XR System Security Configuration Guide* |
| Information about MPLS Traffic Engineering, including configuring a tunnel interface for MPLS-TE | *Cisco IOS XR Multiprotocol Label Switching Configuration Guide* |
| Information about user groups and task IDs | *Cisco IOS XR Interface and Hardware Component Command Reference* |

## 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:<br>http://www.cisco.com/go/mibs |

## RFCs

| 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/support |

# INDEX

| | |
|---|---|
| **HC** | Cisco IOS XR Interface and Hardware Component Configuration Guide |
| **IC** | Cisco IOS XR IP Addresses and Services Configuration Guide |
| **MCC** | Cisco IOS XR Multicast Configuration Guide |
| **MNC** | Cisco IOS XR System Monitoring Configuration Guide |
| **MPC** | Cisco IOS XR MPLS Configuration Guide |
| **QC** | Cisco IOS XR Modular Quality of Service Configuration Guide |
| **RC** | Cisco IOS XR Routing Configuration Guide |
| **SBC** | Cisco IOS XR Session Border Controller Configuration Guide |
| **SC** | Cisco IOS XR System Security Configuration Guide |
| **SMC** | Cisco IOS XR System Management Configuration Guide |
| **VFC** | Cisco IOS XR Virtual Firewall Configuration Guide |

## A


action capabilities-conflict command **HC-119**
action discovery-timeout command **HC-119**
aggregate none command **HC-169**
ais-shut command **HC-368**
aps group command **HC-285, HC-366**
aps submode
    channel command **HC-285, HC-286, HC-366**
    interface loopback command **HC-366**
    *See* aps group command
atm address-registration command **HC-58**
atm ilmi-config disable command **HC-60**
atm ilmi-keepalive command **HC-58**
ATM interface
    address registration and callback functions **HC-58**
    ATM overview **HC-13**
    attaching a vc-class **HC-54, HC-55, HC-57**
    bringing up **HC-22, HC-70**
    configuring
        IP address and subnet mask **HC-36**
        OAM AIS/RDI **HC-51**
        OAM F5 loopback **HC-51**
        OAM management **HC-51**
        optional parameters **HC-24, HC-70**
        PVC **HC-28**
        traffic shaping **HC-33, HC-52**
        VC class **HC-50, HC-73**
        VP tunnel **HC-32, HC-35, HC-72**
    creating
        interface **HC-27**
        PVC **HC-27**
        subinterface **HC-26, HC-27**
        VP tunnel **HC-33**
    F4 OAM overview **HC-15**
    ILMI
        disabling **HC-59, HC-60**
        enabling **HC-57, HC-58**
        overview **HC-15**
    OAM packets **HC-33**
    prerequisites **HC-12**
    vc-class, attaching **HC-53**
    VC class-mapping overview **HC-14**
    verifying
        connectivity **HC-37**
        ILMI configuration **HC-59, HC-61**
    VP tunnel
        displaying **HC-34, HC-37**
        overview **HC-14**


## B


bert pattern command **HC-402, HC-404**
buckets archive command **HC-169**
bundle command **HC-543**
Bundle-Ether command **HC-215**

bundle id command HC-215, HC-219
bundle-POS HC-240

## C

cablelength command HC-379, HC-386, HC-392
card type
changing to E1 mode HC-381
changing to E3 mode HC-381
changing to T1 mode HC-381
changing to T3 mode HC-381
channel command HC-285, HC-286, HC-366
channel-group command HC-396, HC-503, HC-508, HC-542
Channelized ATM HC-61, HC-74
Channelized SONET, configuring HC-273, HC-278, HC-320, HC-323
CHAP
defined HC-415
enabling HC-527
password, configuring HC-532
ppp HC-448, HC-521
refusing HC-537
circuit emulation HC-17
class-int command HC-54
class-vc command HC-57
clear channel HC-75
clear channel ATM HC-64, HC-75
clock source command HC-379
controller e1 command HC-404
controller mgmtmultilink command HC-542
crc command HC-414, HC-452

## D

default settings
flow control, Management Ethernet HC-83
mac-address (Management Ethernet) HC-83
speed (Management Ethernet) HC-83
delay trigger command HC-361
dot1q vlan command HC-559, HC-562
duplex command HC-86
duplex full command HC-107
duplex half command HC-107

## E

E1 controller
card type, setting HC-381
default configuration values HC-379
frame type HC-379, HC-399
E1 mode, changing to T1 mode HC-381
E3 controller
cable length, setting HC-379
card type, setting HC-381
clock source, default value HC-379
configuring
clear channel HC-383
clear channel serial HC-384
national reserved bits HC-379
E3 configuration mode HC-384
frame type HC-379, HC-386
E3 mode, changing to T3 mode HC-381
encap command (Frame Relay) HC-489, HC-497
encapsulation command HC-51, HC-414, HC-452
encapsulation frame relay command HC-489
encapsulation frame-relay command HC-488
Ethernet interface
configuring
flow control HC-95
IP address and subnet mask HC-104
MAC accounting HC-95, HC-109
MAC address HC-96, HC-105
MTU HC-96, HC-104
configuring MAC accounting HC-95
configuring the MAC address HC-96
default settings
flow control HC-95
MAC accounting HC-95

MAC address HC-96
mtu HC-96
displaying
Ethernet interfaces HC-105
MAC accounting statistics HC-110
displaying Ethernet interfaces HC-105
enabling
flow-control HC-104
Layer 2 transport mode HC-111
Gigabit Ethernet standards HC-97
IEEE 802.3ab 1000BASE-T Gigabit Ethernet HC-98
IEEE 802.3ae 10 Gbps Ethernet HC-98
IEEE 802.3 Physical Ethernet Infrastructure HC-98
IEEE 802.3z 1000 Mbps Gigabit Ethernet HC-98
Layer 2 VPN
configuring Layer 2 protocol tunneling and data units HC-111, HC-563
Layer 2 protocol tunneling and data units, configuring HC-111, HC-563
overview HC-96
VLAN support HC-557
using the l2protocol command HC-111, HC-563
using the mac-accounting command HC-95
using the mac address command HC-96
VLANs
802.1Q frames tagging HC-556
assigning a VLAN AC HC-559, HC-562
displaying VLAN interfaces HC-239, HC-241
Layer 2 VPN attachment circuit, configuring HC-561
MTU inheritance HC-557
native VLAN description HC-557
overview HC-556
subinterface, configuring HC-558
subinterface, removing HC-564
subinterface overview HC-557
using the dot1q vlan command HC-559, HC-562
using the show vlan interfaces command HC-239, HC-241
VLAN interfaces, displaying HC-561
VLAN trunks, displaying HC-561


## F


f4oam disable command HC-33
Fast Ethernet interface
auto-negotiation HC-96, HC-107
configuring
duplex operation HC-96, HC-107
interface speed HC-107
IP address and subnet mask HC-107
MAC accounting HC-96
MTU HC-96, HC-107
default settings
auto-negotiation HC-96
duplex operation HC-96
interface speed HC-96
MAC accounting HC-96
mtu HC-96
shutdown configuration, removing HC-107
fdl HC-380
fdl ansi HC-380
fdl command HC-380, HC-396
flow-control command HC-95, HC-104
Frame Relay
configuration examples HC-512
configuring
PVC encapsulation HC-497
support type HC-489, HC-497
default configuration, modifying HC-495, HC-496
default settings HC-489
LMI
configuring HC-489, HC-497
disabling HC-489, HC-498, HC-499
enabling HC-489
overview HC-489

polling HC-490, HC-491
overview HC-488
POS interfaces HC-411
prerequisites HC-488
PVCs HC-489
serial interfaces HC-450
frame-relay intf-type command HC-489, HC-497
frame-relay intf-type dce command HC-490
frame-relay intf-type dte command HC-490
frame-relay lmi command HC-489
frame-relay lmi disable command HC-499
frame-relay lmi-t391dte command HC-491
frame-relay lmi-type command HC-489, HC-497
framing command HC-379


## H


HDLC HC-414
hw-module subslot cardtype command HC-381, HC-382


## I


IEEE 802.3ad standard HC-211
if preconfiguration submode, ipv4 address command HC-7
if submode
bundle id command HC-219, HC-220, HC-228, HC-232, HC-240
controller sonet command HC-368
duplex command HC-86
interface command HC-367
ip address command HC-237, HC-238
ipv4 address command HC-84, HC-504
keepalive command HC-367, HC-544
mtu command HC-84
no shutdown command HC-84, HC-219, HC-238, HC-240
pos crc command HC-367
*See* interface preconfigure command
speed command HC-87
tunnel destination command RC-575
tunnel source command RC-575
ILMI, on ATM interfaces HC-57
interface Bundle-Ether command HC-217, HC-224, HC-237
interface Bundle-POS command HC-215
interface command HC-27
for Ethernet interfaces HC-562
forVLAN subinterfaces HC-562
Link Bundling HC-240
interface loopback command HC-366
interface POS command HC-231
interface preconfigure command HC-7
interfaces
Link Bundling HC-203
configuring HC-216
link switchover HC-216
prerequisites HC-205
tunnel RC-572
invert command HC-460
ip address command
bundle-POS HC-237, HC-238
ipv4 address command HC-7, HC-27, HC-215, HC-237
Fast Ethernet HC-84


## K


keepalive command HC-414, HC-429
keepalive timer
description HC-415
monitoring
POS link state HC-415


## L


l2protocol command HC-111, HC-563
l2transport command HC-111
L2VPN
See Layer 2 VPN HC-96

*See* Layer 2 VPN HC-96

LACP (Link Aggregation Control Protocol) HC-210

Layer 2 VPN

configuring

Layer 2 protocol tunneling and data units HC-111, HC-563

Layer 2 transport mode HC-111

configuring Layer 2 protocol tunneling and data units HC-111, HC-563

overview HC-96

LCP (Link Control Protocol) HC-414, HC-448

link switchover HC-216

LMI HC-489, HC-498, HC-499

lmi-n391dte command HC-491

lmi-n392dce command HC-490

lmi-n392dte command HC-491

lmi-n393dce command HC-490

lmi-n393dte command HC-491

lmi-t392dce command HC-490

Local Management Interface (LMI) HC-451

loopback, naming convention HC-253

loopback command HC-362

## M

mac accounting command HC-109

mac-accounting command HC-95, HC-96

mac address command HC-96, HC-105

management ethernet interface, configuring HC-82

mdl string command HC-393

mdl transmit command HC-379, HC-393

mip auto-create command HC-124

mode command HC-384

MS-CHAP authentication

disabling HC-538

enabling HC-527, HC-528

password, configuring HC-534

ppp HC-415, HC-521, HC-534, HC-539

viewing HC-529

mtu command HC-96, HC-414, HC-452

multilink command HC-546

multilink fragment-size command HC-504, HC-544

multilink Frame Relay bundle interface HC-504

multilink interleave command HC-546

Multiprotocol Label Switching control processor (MPLSCP) HC-415, HC-448

## N

naming conventions

loopback HC-253

null interface HC-253

preconfigure HC-6

national bits command HC-379, HC-380, HC-387, HC-399

negotiation auto command HC-96, HC-105, HC-107

Network Control Protocols (NCPs) HC-415, HC-448

no interface command HC-565

nonstop forwarding HC-215

no shutdown command

(warning) HC-6

bundle-POS HC-238, HC-240

Fast Ethernet HC-84

null interface

configuring HC-254

displaying HC-255

naming convention HC-253

## O

oam ais-rdi command HC-51

oam-pvc manage command HC-51

oam retry command HC-51

overhead command HC-362

## P

PAP authentication

defined HC-521

disabling HC-535
enabling HC-527, HC-528, HC-530, HC-531
ppp HC-415, HC-530
refusing HC-536
viewing HC-529
path command HC-363, HC-370
path scrambling command HC-368
ping command HC-37
ping ethernet cfm command HC-168
Point-to-Point protocol
*See* PPP
POS (Packet-over-SONET)
*See* POS interface
pos crc command HC-280
POS interface
bringing up HC-418
configuring
CRC value HC-280, HC-414, HC-422
encapsulation type HC-414
interface encapsulation HC-280, HC-422, HC-545
keepalive timer HC-414, HC-428, HC-429, HC-544
MTU HC-280, HC-414, HC-422, HC-436
optional parameters HC-421
PPP authentication HC-415
default settings
CRC HC-414
encapsulation HC-414
keepalive HC-414
mtu HC-414
Frame Relay encapsulation HC-411
HDLC encapsulation
description
overview HC-414
interface configuration mode
interface command HC-545
interface multilink command HC-544, HC-546
interface pos command HC-279, HC-419, HC-422
PPP encapsulation
description HC-411
overview HC-414
subinterfaces, creating with PVCs HC-424
PPP
CHAP
authentication HC-521
enabling HC-527, HC-528
password, configuring HC-532, HC-533
refusing HC-537
viewing HC-529
default configuration, modfying HC-523, HC-525
interfaces, displaying HC-526
MS-CHAP
disabling HC-538, HC-539
enabling HC-528
authenticaion HC-527
password, configuring HC-534
ppp HC-448, HC-521
viewing HC-529
overview HC-520
PAP
authentication HC-449, HC-530
disabling HC-535
enabling HC-527, HC-528, HC-530, HC-531
refusing HC-536
viewing HC-529
POS configuration example HC-547
POS interface HC-411, HC-414
prerequisites HC-519
serial configuration example HC-548
serial interface HC-445, HC-448
ppp authentication command HC-415, HC-449, HC-520, HC-528
ppp chap password command HC-533
ppp chap refuse command HC-537
ppp max-bad-auth command HC-525
ppp max-configure command HC-525
ppp max-failure command HC-525
ppp max-terminate command HC-525
ppp ms-chap password command HC-534

ppp ms-chap refuse command HC-539

ppp multilink minimum-active links command HC-546

ppp pap refuse command HC-536

ppp pap sent-username command HC-530, HC-531

ppp timeout authentication command HC-526

ppp timeout retry command HC-526

preconfiguration

- advantages HC-5
- directory HC-3
- naming conventions HC-6
- restriction to physical interface HC-3

profile command RC-575

PVC

- ATM interface HC-28
- ATM subinterface HC-26, HC-56
- POS subinterface HC-424, HC-426
- VP tunnel HC-35

pvc command HC-27


## R


RP, preconfiguration directory HC-3


## S


scramble command HC-460

serial interface

- configuring
  - CRC HC-460
  - interface encapsulation HC-459
  - IP address and subnet mask HC-457
  - keepalive timer HC-467
  - transmit delay HC-460
- data stream, inverting HC-460
- default settings
  - CRC HC-452
  - encapsulation HC-452
  - keepalive HC-452
  - mtu HC-452
- link state HC-448, HC-449
- payload scrambling, enabling HC-460
- PPP encapsulation HC-448
- prerequisites HC-444

shape command HC-33

show aps command HC-286, HC-369

show atm ilmi-status command HC-59, HC-61

show atm vp-tunnel command HC-37

show atm vp-tunnel interface atm command HC-34

show bundle Bundle-Ether command HC-221, HC-229, HC-241

show bundle Bundle-POS command HC-233

show controllers command HC-385

show controller sonet command HC-364

show frame-relay lmi command HC-490

show interfaces command HC-103, HC-480

- for Ethernet interfaces HC-105, HC-112

show ip route command RC-576

show lacp bundle Bundle-Ether command HC-221

show lacp bundle Bundle-POS command HC-233

show mac accounting command HC-110

show ppp interfaces command HC-526, HC-529

show version command HC-103

show vlan command HC-239, HC-241, HC-561

show vlan interface command HC-566

show vlan trunks command HC-229

SLARP (Serial Line Address Resolution Protocol) HC-416, HC-448, HC-450

SONET (Synchronous Optical Network)

- APS HC-364
- description HC-355
- fast reroute (FFR) HC-369

SONET APS (SONET Automatic Protection Switching) HC-364

SONET controller

- configuring HC-360
- frame type HC-362

sonet submode

- ais-shut command HC-368

clock source command HC-361, HC-368
delay trigger command HC-361, HC-370
framing command HC-362
loopback command HC-362
overhead command HC-362
path command HC-363, HC-370
path scrambling command HC-368
*See* controller sonet command
speed command HC-96, HC-107
SSH (Secure Shell) RC-573
SSH (Secure Shell), not needed with IPSec RC-573
SSL (Secure Socket Layer) RC-573
SSL (Secure Socket Layer), not needed with IPSec RC-573
switchover HC-215


## T


T1 controller
ANSI T1.403 or AT&T TR54016 performance reports HC-380, HC-396
BERT, configuring HC-404
card type, setting HC-381
clock source HC-379, HC-395
default configuration values HC-379
DS0 timeslots, associating HC-396, HC-503, HC-509, HC-542
frame type HC-379, HC-395
T1 channel group, creating HC-396, HC-503, HC-508, HC-542
T1 configuration mode HC-395, HC-502, HC-508, HC-542
yellow alarms HC-379
T1 mode, changing to E1 mode HC-381
T3 controller
cable length, setting HC-379
card type, setting HC-381
clock source
configuring HC-386, HC-392, HC-542
default value HC-379
configuring HC-388
BERT HC-402
channelized T3 controller HC-390
clear channel E3 controller HC-384
clear channel T3 controller HC-388
FRF.12 end-to-end fragmentation HC-508
MDL messages HC-379
multilink Frame Relay bundle interfaces HC-502
frame type HC-379, HC-392
modifying
default E3 controller HC-386
default T3 controller HC-392
T3 mode, changing to E3 mode HC-381
timeslots command HC-396, HC-503, HC-509, HC-542
traffic filtering HC-254
transmit-delay command HC-460
transparent switchover HC-82
tunnel destination command RC-575
tunnel source command RC-575


## V


vc-class
ATM interfaces HC-50, HC-51, HC-53
ATM PVCs HC-55
ATM subinterfaces HC-54
configuring HC-73
creating HC-73
vc-class command HC-51
virtual interface
active/standby RPs HC-6, HC-255
naming convention HC-253, RC-572
null interface definition HC-254
switchover HC-253
VLANs
802.1Q frames tagging HC-556
assigning a VLAN AC HC-559, HC-562
configuring
IP address and subnet mask HC-560
subinterfaces HC-558

displaying
VLAN interfaces HC-561, HC-566
VLAN trunks HC-561, HC-566
displaying VLAN interfaces HC-239, HC-241
Layer 2 VPN
attachment circuit, configuring HC-561
Layer 2 VPN support HC-557
MTU inheritance HC-557
native VLAN description HC-557
overview HC-556
subinterface overview HC-557
using the dot1q vlan command HC-559, HC-562
using the show vlan interfaces command HC-239, HC-241
VLAN subinterface, removing HC-564
VP tunnel
configuring HC-32, HC-64
creating HC-32
example HC-75
on ATM interfaces HC-72
vp-tunnel command HC-33

## Y

yellow command HC-379, HC-395