Cisco IOS XR Interface and Hardware Component Configuration Guide

Cisco IOS XR Software Release 3.5

Corporate Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 526-4100

Customer Order Number:
Text Part Number: OL-12288-01

Cisco IOS XR Interface and Hardware Component Configuration Guide

CONTENTS

Preface

This document describes the commands that support router interface and hardware configuration using Cisco IOS XR software.

The preface contains the following sections:

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-12282-01	June 2007	Initial release of this document.

Obtaining Documentation

Cisco documentation and additional literature are available on Cisco.com. This section explains the product documentation resources that Cisco offers.

Cisco.com

You can access the most current Cisco documentation at this URL:

http://www.cisco.com/techsupport

You can access the Cisco website at this URL:

http://www.cisco.com

You can access international Cisco websites at this URL:

http://www.cisco.com/public/countries_languages.shtml

Product Documentation DVD

The Product Documentation DVD is a library of technical product documentation on a portable medium. The DVD enables you to access installation, configuration, and command guides for Cisco hardware and software products. With the DVD, you have access to the HTML documentation and some of the PDF files found on the Cisco website at this URL:

http://www.cisco.com/univercd/home/home.htm

The Product Documentation DVD is created and released regularly. DVDs are available singly or by subscription. Registered Cisco.com users can order a Product Documentation DVD (product number DOC-DOCDVD= or DOC-DOCDVD=SUB) from Cisco Marketplace at the Product Documentation Store at this URL:

http://www.cisco.com/go/marketplace/docstore

Ordering Documentation

You must be a registered Cisco.com user to access Cisco Marketplace. Registered users may order Cisco documentation at the Product Documentation Store at this URL:

http://www.cisco.com/go/marketplace/docstore

If you do not have a user ID or password, you can register at this URL:

http://tools.cisco.com/RPF/register/register.do

Documentation Feedback

You can provide feedback about Cisco technical documentation on the Cisco Support site area by entering your comments in the feedback form available in every online document.

Cisco Product Security Overview

Cisco provides a free online Security Vulnerability Policy portal at this URL:

http://www.cisco.com/en/US/products/products_security_vulnerability_policy.html

From this site, you will find information about how to do the following:

- Report security vulnerabilities in Cisco products
- Obtain assistance with security incidents that involve Cisco products
- Register to receive security information from Cisco

A current list of security advisories, security notices, and security responses for Cisco products is available at this URL:

http://www.cisco.com/go/psirt

To see security advisories, security notices, and security responses as they are updated in real time, you can subscribe to the Product Security Incident Response Team Really Simple Syndication (PSIRT RSS) feed. Information about how to subscribe to the PSIRT RSS feed is found at this URL:

http://www.cisco.com/en/US/products/products_psirt_rss_feed.html

Reporting Security Problems in Cisco Products

Cisco is committed to delivering secure products. We test our products internally before we release them, and we strive to correct all vulnerabilities quickly. If you think that you have identified a vulnerability in a Cisco product, contact PSIRT:

- For emergencies only—security-alert@cisco.com

  An emergency is either a condition in which a system is under active attack or a condition for which a severe and urgent security vulnerability should be reported. All other conditions are considered nonemergencies.

- For nonemergencies—psirt@cisco.com

In an emergency, you can also reach PSIRT by telephone:

- 1 877 228-7302
- 1 408 525-6532

Tip We encourage you to use Pretty Good Privacy (PGP) or a compatible product (for example, GnuPG) to encrypt any sensitive information that you send to Cisco. PSIRT can work with information that has been encrypted with PGP versions 2.*x* through 9.*x*.

Never use a revoked encryption key or an expired encryption key. The correct public key to use in your correspondence with PSIRT is the one linked in the Contact Summary section of the Security Vulnerability Policy page at this URL:

http://www.cisco.com/en/US/products/products_security_vulnerability_policy.html

The link on this page has the current PGP key ID in use.

If you do not have or use PGP, contact PSIRT to find other means of encrypting the data before sending any sensitive material.

Product Alerts and Field Notices

Modifications to or updates about Cisco products are announced in Cisco Product Alerts and Cisco Field Notices. You can receive these announcements by using the Product Alert Tool on Cisco.com. This tool enables you to create a profile and choose those products for which you want to receive information.

To access the Product Alert Tool, you must be a registered Cisco.com user. Registered users can access the tool at this URL:

http://tools.cisco.com/Support/PAT/do/ViewMyProfiles.do?local=en

To register as a Cisco.com user, go to this URL:

http://tools.cisco.com/RPF/register/register.do

Obtaining Technical Assistance

Cisco Technical Support provides 24-hour-a-day award-winning technical assistance. The Cisco Support website on Cisco.com features extensive online support resources. In addition, if you have a valid Cisco service contract, Cisco Technical Assistance Center (TAC) engineers provide telephone support. If you do not have a valid Cisco service contract, contact your reseller.

Cisco Support Website

The Cisco Support website provides online documents and tools for troubleshooting and resolving technical issues with Cisco products and technologies. The website is available 24 hours a day at this URL:

http://www.cisco.com/en/US/support/index.html

Access to all tools on the Cisco Support website requires a Cisco.com user ID and password. If you have a valid service contract but do not have a user ID or password, you can register at this URL:

http://tools.cisco.com/RPF/register/register.do

Note Before you submit a request for service online or by phone, use the **Cisco Product Identification Tool** to locate your product serial number. You can access this tool from the Cisco Support website by clicking the **Get Tools & Resources** link, clicking the **All Tools (A-Z)** tab, and then choosing **Cisco Product Identification Tool** from the alphabetical list. This tool offers three search options: by product ID or model name; by tree view; or, for certain products, by copying and pasting **show** command output. Search results show an illustration of your product with the serial number label location highlighted. Locate the serial number label on your product and record the information before placing a service call.

Tip **Displaying and Searching on Cisco.com**

If you suspect that the browser is not refreshing a web page, force the browser to update the web page by holding down the Ctrl key while pressing **F5**.

To find technical information, narrow your search to look in technical documentation, not the entire Cisco.com website. After using the Search box on the Cisco.com home page, click the **Advanced Search** link next to the Search box on the resulting page and then click the **Technical Support & Documentation** radio button.

To provide feedback about the Cisco.com website or a particular technical document, click **Contacts & Feedback** at the top of any Cisco.com web page.

Submitting a Service Request

Using the online TAC Service Request Tool is the fastest way to open S3 and S4 service requests. (S3 and S4 service requests are those in which your network is minimally impaired or for which you require product information.) After you describe your situation, the TAC Service Request Tool provides recommended solutions. If your issue is not resolved using the recommended resources, your service request is assigned to a Cisco engineer. The TAC Service Request Tool is located at this URL:

http://www.cisco.com/techsupport/servicerequest

For S1 or S2 service requests, or if you do not have Internet access, contact the Cisco TAC by telephone. (S1 or S2 service requests are those in which your production network is down or severely degraded.) Cisco engineers are assigned immediately to S1 and S2 service requests to help keep your business operations running smoothly.

To open a service request by telephone, use one of the following numbers:

Asia-Pacific: +61 2 8446 7411
Australia: 1 800 805 227
EMEA: +32 2 704 55 55
USA: 1 800 553 2447

For a complete list of Cisco TAC contacts, go to this URL:

http://www.cisco.com/techsupport/contacts

Definitions of Service Request Severity

To ensure that all service requests are reported in a standard format, Cisco has established severity definitions.

Severity 1 (S1)—An existing network is “down” or there is a critical impact to your business operations. You and Cisco will commit all necessary resources around the clock to resolve the situation.

Severity 2 (S2)—Operation of an existing network is severely degraded, or significant aspects of your business operations are negatively affected by inadequate performance of Cisco products. You and Cisco will commit full-time resources during normal business hours to resolve the situation.

Severity 3 (S3)—Operational performance of the network is impaired while most business operations remain functional. You and Cisco will commit resources during normal business hours to restore service to satisfactory levels.

Severity 4 (S4)—You require information or assistance with Cisco product capabilities, installation, or configuration. There is little or no effect on your business operations.

Obtaining Additional Publications and Information

Information about Cisco products, technologies, and network solutions is available from various online and printed sources.

- The Cisco Online Subscription Center is the website where you can sign up for a variety of Cisco e-mail newsletters and other communications. Create a profile and then select the subscriptions that you would like to receive. To visit the Cisco Online Subscription Center, go to this URL:

  http://www.cisco.com/offer/subscribe

- The *Cisco Product Quick Reference Guide* is a handy, compact reference tool that includes brief product overviews, key features, sample part numbers, and abbreviated technical specifications for many Cisco products that are sold through channel partners. It is updated twice a year and includes the latest Cisco channel product offerings. To order and find out more about the *Cisco Product Quick Reference Guide*, go to this URL:

  http://www.cisco.com/go/guide

- Cisco Marketplace provides a variety of Cisco books, reference guides, documentation, and logo merchandise. Visit Cisco Marketplace, the company store, at this URL:

  http://www.cisco.com/go/marketplace/

- Cisco Press publishes a wide range of general networking, training, and certification titles. Both new and experienced users will benefit from these publications. For current Cisco Press titles and other information, go to Cisco Press at this URL:

  http://www.ciscopress.com

- *Internet Protocol Journal* is a quarterly journal published by Cisco for engineering professionals involved in designing, developing, and operating public and private internets and intranets. You can access the *Internet Protocol Journal* at this URL:

  http://www.cisco.com/ipj

- Networking products offered by Cisco, as well as customer support services, can be obtained at this URL:

  http://www.cisco.com/en/US/products/index.html

- Networking Professionals Connection is an interactive website where networking professionals share questions, suggestions, and information about networking products and technologies with Cisco experts and other networking professionals. Join a discussion at this URL:

  http://www.cisco.com/discuss/networking

- "What's New in Cisco Documentation" is an online publication that provides information about the latest documentation releases for Cisco products. Updated monthly, this online publication is organized by product category to direct you quickly to the documentation for your products. You can view the latest release of "What's New in Cisco Documentation" at this URL:

  http://www.cisco.com/univercd/cc/td/doc/abtunicd/136957.htm

- World-class networking training is available from Cisco. You can view current offerings at this URL:

  http://www.cisco.com/en/US/learning/index.html

Configuring ATM Interfaces on Cisco IOS XR Software

This chapter 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 the following ATM forum standards:

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

Feature History for Configuring Bidirectional Forwarding Detection on Cisco IOS XR

Release	Modification
Release 3.4.0	This feature was introduced on the Cisco XR 12000 Series Router on the following hardware: • Cisco XR 12000 Series 4-Port OC-3c/STM-1c ATM ISE Line Card, multimode • Cisco XR 12000 Series 4-Port OC-3c/STM-1c ATM ISE Line Card, single-mode • Cisco XR 12000 Series 4-port OC-12/STM-4 ATM multimode ISE line card with SC connector • 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.

Contents

Prerequisites for Implementing ATM

The following are required to implement ATM:

- You must be in a user group associated with a task group that includes the proper task IDs for the ATM commands. Task IDs for commands are listed in *Cisco IOS XR Interface and Hardware Component Command Reference*. For detailed information about user groups and task IDs, refer to the *Configuring AAA Services on Cisco IOS XR Software* module of *Cisco IOS XR System Security Configuration Guide*.
- A Cisco XR 12000 Series Router that runs Cisco IOS XR software.

Information About ATM

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:

- 4-port OC12
- 4-port OC3

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 affect 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 affect 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.

How to Bring Up and Configure ATM Interfaces

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

- Bringing Up an ATM Interface, page 6
- Configuring Optional ATM Interface Parameters, page 8

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** *instance*

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** *instance* **brief**

DETAILED STEPSs

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface atm** *instance* **Example:** `RP/0/0/CPU0:router (config)# interface atm 0/6/0/1`	Enters ATM interface configuration mode.
Step 3	**no shutdown** **Example:** `RP/0/0/CPU0:router (config-if)# no shutdown`	Removes the shutdown configuration. **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 4	**end** or **commit** **Example:** `RP/0/0/CPU0:router (config-if)# end` or `RP/0/0/CPU0:router(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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** **Example:** `RP/0/0/CPU0:router (config-if)# exit`	Exits interface configuration mode and enters global configuration mode.
Step 6	**exit** **Example:** `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. **Note** The configuration on both ends of the ATM connection must match.
Step 8	**show interfaces atm** *instance* **brief** **Example:** `RP/0/0/CPU0:router# show interfaces atm 0/6/0/1 brief`	(Optional) Verifies that the interface is active and properly configured. 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 8.
- 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 11.
- 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 16.
- To use the interface as a Layer 2 post mode AC, see the "How to Configure a Layer 2 Attachment Circuit" section on page 22.
- 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 34.
- To enable ILMI on the ATM interface you just brought up, see the "How to Configure ILMI on ATM Interfaces" section on page 42.

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 6.

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** *instance*
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** [*instance*]
10. **show interfaces atm** *instance* **brief**

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface atm** *instance* **Example:** `RP/0/0/CPU0:router (config)# interface atm 0/6/0/1`	Enters ATM interface configuration mode.
Step 3	**atm maxvpi-bits 12** **Example:** `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** **Example:** `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* **Example:** `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. **Note** The default value for each timer is 50 microseconds. **Note** The **atm mcpt-timers** command is applicable to Layer 2 ATM ACs only.

	Command or Action	Purpose
Step 6	**end** or **commit** **Example:** `RP/0/0/CPU0:router (config-if)# end` or `RP/0/0/CPU0:router(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 7	**exit** **Example:** `RP/0/0/CPU0:router (config-if)# exit`	Exits interface configuration mode and enters global configuration mode.
Step 8	**exit** **Example:** `RP/0/0/CPU0:router (config)# exit`	Exits global configuration mode and enters EXEC mode.
Step 9	**show atm interface atm** [*instance*] **Example:** `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** *instance* **Example:** `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 11.
- 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 16.
- To use the interface as a Layer 2 ATM AC, see the "How to Configure a Layer 2 Attachment Circuit" section on page 22.

- 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 34.
- To enable ILMI on the ATM interface you just brought up, see the "How to Configure ILMI on ATM Interfaces" section on page 42.

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 11
- Configuring Optional Point-to-Point ATM PVC Parameters, page 13

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 6.

Restrictions

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

SUMMARY STEPS

1. **configure**
2. **interface atm** *instance.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** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface atm** *instance.subinterface* **point-to-point** **Example:** `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* **Example:** `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* **Example:** `RP/0/0/CPU0:router (config-subif)# pvc 5/10`	(Optional) Creates an ATM permanent virtual circuit (PVC) and enters ATM PVC configuration submode. **Note** Only one PVC is allowed per subinterface.
Step 5	**end** or **commit** **Example:** `RP/0/0/CPU0:router (config-subif)# end` or `RP/0/0/CPU0:router(config-subif)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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. **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 13.
- 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 34.

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 11.

Restrictions

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

SUMMARY STEPS

1. **configure**
2. **interface atm** *instance.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** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface atm** *instance.subinterface* **point-to-point** **Example:** `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* **Example:** `RP/0/0/CPU0:router (config-subif)# pvc 5/10`	Enters subinterface configuration mode for the PVC.
Step 4	**encapsulation** {**aal5mux ipv4** | **aal5nlpid** | **aal5snap**} **Example:** `RP/0/0/CPU0:router (config-atm-vc)# encapsulation aal5snap`	Configures the ATM adaptation layer (AAL) and encapsulation type for a PVC. **Note** The default encapsulation type for a vc-class is AAL5/SNAP
Step 5	**oam-pvc manage** [*frequency*] [**disable**] [**keep-vc-up** [**seg-aisrdi-failure**] **Example:** `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). • Include the **disable** keyword to disable OAM management on the specified PVC. • Include the **keep-vc-up** keyword specify that PVC remains in the UP state when CC cells detect connectivity failure. • 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*]] **Example:** `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*]]] **Example:** `RP/0/0/CPU0:router (config-atm-vc)# oam retry 5 10 5`	Configures parameters related to OAM management for the PVC. If no OAM AIS/RDI cells are received within the specified interval, the PVC is brought up.

	Command or Action	Purpose
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*] **Example:** `RP/0/0/CPU0:router (config-atm-vc)# shape vbr-nrt 100000 100000 8000`	Configures ATM traffic shaping for the PVC. You must estimate how much bandwidth is required before you configure ATM traffic shaping. • *peak_output_rate*—Configures the maximum cell rate that is always available for the traffic. • *Sustained_output_rate*—Sustained output rate for the bit rate. • *burst size*—Burst cell size for the bit rate. Range is from 1 through 8192.
Step 9	**service-policy** [**input** | **output**] *policy_name* **Example:** `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. **Note** For information on creating and configuring service policies, see the *Cisco IOS XR Modular Quality of Service Configuration Guide.*
Step 10	**end** or **commit** **Example:** `RP/0/0/CPU0:router (config-subif)# end` or `RP/0/0/CPU0:router(config-subif)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. 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. **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 34.

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 16
- Creating and Configuring Subinterfaces with PVCs on a VP-tunnel, page 19

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 6.
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 16.
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 19.
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 19.

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 6.

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 19.
- 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** *instance*
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** [*instance*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface atm** *instance* **Example:** `RP/0/0/CPU0:router (config)# interface atm 0/6/0/1`	Enters ATM interface configuration mode.
Step 3	**vp-tunnel** *vpi* **Example:** `RP/0/0/CPU0:router (config)# vp-tunnel 10`	Configures a vp-tunnel on an ATM interface.
Step 4	**f4oam disable** **Example:** `RP/0/0/CPU0:router(config-atm-vp-tunnel)# f4oam disable`	(Optional) Disables the passing of OAM packets.

	Command or Action	Purpose
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*] **Example:** `RP/0/0/CPU0:router (config-if)# shape`	Configures ATM traffic shaping for the PVC. You must estimate how much bandwidth is required before you configure ATM traffic shaping. • *peak_output_rate*—Configures the maximum cell rate that is always available for the traffic. • *Sustained_output_rate*—Sustained output rate for the bit rate. • *burst size*—Burst cell size for the bit rate. Range is from 1 through 8192. **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.
Step 6	**end** or **commit** **Example:** `RP/0/0/CPU0:router (config-if)# end` or `RP/0/0/CPU0:router(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 7	**exit** **Example:** `RP/0/0/CPU0:router (config-if)# exit`	Exits interface configuration mode and enters global configuration mode.
Step 8	**exit** **Example:** `RP/0/0/CPU0:router (config)# exit`	Exits global configuration mode and enters EXEC mode.

	Command or Action	Purpose
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** [*instance*] **Example:** `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 16.

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** *instance.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** *instance.subinterface vpi/vci*
8. **show atm vp-tunnel** [**interface atm** *instance*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface atm** *instance.subinterface* **point-to-point** **Example:** `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* **Example:** `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* **Example:** `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 16. Replace *vpi* with the VPI of the vp-tunnel on which you are creating the PVC. **Note** The PVC VPI and vp-tunnel VCI must match or the connection will not be active. **Note** A vp-tunnel is not usable until you create PVCs under it.

	Command or Action	Purpose
Step 5	**end** or **commit** **Example:** `RP/0/0/CPU0:router (config-subif)# end` or `RP/0/0/CPU0:router(config-subif)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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** *instance.subinterface vpi/vci* **Example:** `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. • Replace *instance.subinterface* with the ATM subinterface that is configured on the vp-tunnel whose connectivity you want to verify. This is the same *instance.subinterface* you configured in Step 2. • 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. • 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** *instance*] **Example:** `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 19
- To configure a vc-class and apply it to an ATM interface, see the "Creating and Configuring a VC-Class" section on page 34.

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 6.

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** *instance*
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** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface atm** *instance* **Example:** `RP/0/0/CPU0:router (config)# interface atm 0/6/0/1`	Enters interface configuration mode for an ATM interface.
Step 3	**l2transport** **Example:** `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** or **commit** **Example:** `RP/0/0/CPU0:router (config-if-l2)# end` or `RP/0/0/CPU0:router(config-if-l2)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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. **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 24.

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 22.

SUMMARY STEPS

1. **configure**
2. **interface atm** *instance*
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** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface atm** *instance* **Example:** `RP/0/0/CPU0:router (config)# interface atm 0/6/0/1`	Enters interface configuration mode for an ATM interface.
Step 3	**atm mcpt-timers** *timer-1 timer-2 timer-3* **Example:** `RP/0/0/CPU0:router (config-if)# atm mcpt-timers 50 100 200`	Specifies the maximum cell packing timeout values for each of the three per-interface MCPT timers, in microseconds. **Note** The default value for each timer is 50 microseconds.
Step 4	**l2transport** **Example:** `RP/0/0/CPU0:router (config-if)# l2transport`	Enters ATM Layer 2 transport configuration mode.

	Command or Action	Purpose
Step 5	**cell-packing** *cells timer* **Example:** `RP/0/0/CPU0:router (config-if-l2)# cell-packing 6 1`	Sets the maximum number of cells allowed per packet, and specifies a maximum cell packing timeout (MCPT) timer to use for cell packing. • Replace *cells* with the maximum number of cells to use per packet. Range is from 2 through 86. • 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	**end** or **commit** **Example:** `RP/0/0/CPU0:router (config-if-l2)# end` or `RP/0/0/CPU0:router(config-if-l2)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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 AC at the other end of the connection.	Brings up the Layer 2 port mode AC. **Note** The configuration on both ends of the connection must match.

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 6.

Restrictions

Only one PVC can be configured for each ATM subinterface.

SUMMARY STEPS

1. **configure**
2. **interface atm** *instance.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** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface atm** *instance.subinterface* **l2transport** **Example:** `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* **Example:** `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. **Note** Only one PVC is allowed per subinterface.

	Command or Action	Purpose
Step 4	**end** or **commit** **Example:** `RP/0/0/CPU0:router(config-atm-l2transport-pvc)# end` or `RP/0/0/CPU0:router(config-atm-l2transport-pvc)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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. **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 27.
- 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 39.
- 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 25.

Restrictions

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

SUMMARY STEPS

1. **configure**
2. **interface atm** *instance.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** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface atm** *instance.subinterface* **l2transport** **Example:** `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* **Example:** `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**} **Example:** `RP/0/0/CPU0:router(config-atm-l2transport-pvc)# encapsulation aal5`	Configures the ATM adaptation layer (AAL) and encapsulation type for a PVC. **Note** The default encapsulation type for a PVC is AAL5.

	Command or Action	Purpose
Step 5	**cell-packing** *cells timer* **Example:** `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. • Replace *cells* with the maximum number of cells to use per packet. Range is from 2 through 86. • 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*] **Example:** `RP/0/0/CPU0:router(config-atm-l2transport-pvc)# shape vbr-nrt 100000 100000 8000`	Configures ATM traffic shaping for the PVC. You must estimate how much bandwidth is required before you configure ATM traffic shaping. • *peak_output_rate*—Configures the maximum cell rate that is always available for the traffic. • *Sustained_output_rate*—Sustained output rate for the bit rate. • *burst size*—Burst cell size for the bit rate. Range is from 1 through 8192.
Step 7	**end** or **commit** **Example:** `RP/0/0/CPU0:router(config-atm-l2transport-pvc)# end` or `RP/0/0/CPU0:router(config-atm-l2transport-pvc)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. 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. **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 39.

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 6.

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** *instance.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** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface atm** *instance.subinterface* **l2transport** **Example:** `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.
Step 3	**pvp** *vpi* **Example:** `RP/0/0/CPU0:router(config-if)# pvp 100`	(Optional) Creates an ATM PVP and enters ATM PVP configuration submode. **Note** Only one PVP is allowed per subinterface.

	Command or Action	Purpose
Step 4	**end** or **commit** **Example:** `RP/0/0/CPU0:router(config-atm-l2transport-pvp)# end` or `RP/0/0/CPU0:router(config-atm-l2transport-pvp)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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. **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 31.
- 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 30.

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** *instance.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** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface atm** *instance.subinterface* **l2transport** **Example:** `RP/0/0/CPU0:router(config)# interface atm 0/6/0/1.10 l2transport`	Enters ATM subinterface configuration mode.
Step 3	**pvp** *vpi* **Example:** `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* **Example:** `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. • Replace *cells* with the maximum number of cells to use per packet. Range is from 2 through 86. • 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*] **Example:** `RP/0/0/CPU0:router(config-atm-l2transport-pvp)# shape vbr-nrt 100000 100000 8000`	Configures ATM traffic shaping for the PVC. You must estimate how much bandwidth is required before you configure ATM traffic shaping. • *peak_output_rate*—Configures the maximum cell rate that is always available for the traffic. • *Sustained_output_rate*—Sustained output rate for the bit rate. • *burst size*—Burst cell size for the bit rate. Range is from 1 through 8192.
Step 6	**end** or **commit** **Example:** `RP/0/0/CPU0:router(config-atm-l2transport-pvp)# end` or `RP/0/0/CPU0:router(config-atm-l2transport-pvp)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. 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. **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 34
- Attaching a Vc-Class to a Point-to-Point ATM Main Interface, page 36
- Attaching a Vc-Class to a Point-to-Point ATM Subinterface, page 38
- Attaching a Vc-Class to a PVC on an ATM Subinterface, page 39

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** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**vc-class atm name** **Example:** `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**} **Example:** `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. **Note** The default encapsulation type for a vc-class is AAL5/SNAP **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*]] **Example:** `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. **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*]]] **Example:** `RP/0/0/CPU0:router (config-vc-class-atm)# oam retry 5 10 5`	Configures parameters related to OAM management. **Note** In Vc-classes, the **oam retry** command applies to Layer 3 Point-to-point configurations only.
Step 6	**oam-pvc manage** *seconds* **Example:** `RP/0/0/CPU0:router (config-vc-class-atm)# oam-pvc manage 300`	Configures the ATM OAM F5 loopback frequency. **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*] **Example:** `RP/0/0/CPU0:router (config-vc-class-atm)# shape vbr-nrt 100000 100000 8000`	Configures ATM traffic shaping for the PVC. You must estimate how much bandwidth is required before you configure ATM traffic shaping. • *peak_output_rate*—Configures the maximum cell rate that is always available for the traffic. • *Sustained_output_rate*—Sustained output rate for the bit rate. • *burst size*—Burst cell size for the bit rate. Range is from 1 through 8192.
Step 8	**end** or **commit** **Example:** `RP/0/0/CPU0:router (config-if)# end` or `RP/0/0/CPU0:router(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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 -36.
- 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 -38.
- 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 -39.

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** *instance* **point-to-point**
3. **class-int** *vc-class-name*
4. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface atm** *instance* **point-to-point** **Example:** `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* **Example:** `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 34.
Step 4	**end** or **commit** **Example:** `RP/0/0/CPU0:router (config-if)# end` or `RP/0/0/CPU0:router(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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** *instance.subinterface* **point-to-point**
3. **class-int** *vc-class-name*
4. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface atm** *instance.subinterface* **point-to-point** **Example:** `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* **Example:** `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 -34.
Step 4	**end** or **commit** **Example:** `RP/0/0/CPU0:router (config-subif)# end` or `RP/0/0/CPU0:router(config-subif)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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** *instance*[*.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** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface atm** *instance.subinterface* [**point-to-point** | **l2transport**] **Example:** `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. 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. **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 11.
Step 3	**pvc** *vpi*/*vci* **Example:** `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. **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 11.

	Command or Action	Purpose
Step 4	**class-vc** *vc-class-name* **Example:** `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** or **commit** **Example:** `RP/0/0/CPU0:router (config-if)# end` or `RP/0/0/CPU0:router(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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 42
- Disabling ILMI on an ATM Interface, page 44

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 6.

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** *instance*
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** *instance*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface atm** *instance* **Example:** `RP/0/0/CPU0:router (config)# interface atm 0/6/0/1`	Enters ATM interface configuration mode.
Step 3	**atm address-registration** **Example:** `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*] Example: `RP/0/0/CPU0:router (config-if)# atm ilmi-keepalive`	(Optional) Enables ILMI keepalives on an ATM interface.
Step 5	**pvc** *vpi/vci* **ilmi** **Example:** `RP/0/0/CPU0:router (config-if)# pvc 5/30 ilmi`	Creates an ATM permanent virtual circuit (PVC) with ILMI encapsulation.
Step 6	**end** or **commit** **Example:** `RP/0/0/CPU0:router (config-if)# end` or `RP/0/0/CPU0:router(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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** **Example:** `RP/0/0/CPU0:router (config-if)# exit`	Exits interface configuration mode and enters global configuration mode.
Step 8	**exit** **Example:** `RP/0/0/CPU0:router (config)# exit`	Exits global configuration mode and enters EXEC mode.
Step 9	**show atm ilmi-status** [**atm** *instance*] **Example:** `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** *instance*
3. **atm ilmi-config disable**
4. **end**
   or
   commit
5. **exit**
6. **exit**
7. **show atm ilmi-status** [**atm** *instance*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface atm** *instance* **Example:** `RP/0/0/CPU0:router (config)# interface atm 0/6/0/1`	Enters ATM interface configuration mode.

	Command or Action	Purpose
Step 3	**atm ilmi-config disable** **Example:** `RP/0/0/CPU0:router (config-if)# atm ilmi-config disable`	(Optional) Disables ILMI on the ATM interface. To re-enable ILMI on an ATM interface, use the **no atm ilmi-config disable** form of this command.
Step 4	**end** or **commit** **Example:** `RP/0/0/CPU0:router (config-if)# end` or `RP/0/0/CPU0:router(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 5	**exit** **Example:** `RP/0/0/CPU0:router (config-if)# exit`	Exits interface configuration mode and enters global configuration mode.
Step 6	**exit** **Example:** `RP/0/0/CPU0:router (config)# exit`	Exits global configuration mode and enters EXEC mode.
Step 7	**show atm ilmi-status** [**atm** *instance*] **Example:** `RP/0/0/CPU0:router (config)# show atm ilmi-status atm 0/6/0/1`	(Optional) Verifies the ILMI configuration for the specified interface.

Configuration Examples for Configuring ATM on Cisco IOS XR Software

This section provides the following configuration examples:

- ATM Interface Bring Up and Configuration: Example, page 46
- Point-To-Point ATM Subinterface Configuration: Example, page 46

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 0/2/0/2

Interface                          : ATM0/2/0/2
AAL Enabled                        : AAL5
Max-VP                             : 254
Max-VC                             : 2048
Configured VP-Tunnels              : 0
Configured PVCs                    : 4
VP-Tunnels in Down State           : 0
PVCs in Down State                 : 4

Received Side Statistics:
    Received Cells                 : 1691252892
    Received Bytes                 : 480532
    Received AAL Packets           : 9241

Receive Side Cells Dropped:
```

```
    Unrecognized VPI/VCI              : 64

Receive Side AAL5 Packets Dropped:
    Unavailable SAR Buffer            : 0
    Non-Resource Exhaustion           : 0
    Reassembly Timeout                : 0
    CRC-32 Errors                     : 0
    Zero Length                       : 0

RP/0/0/CPU0:router # show atm pvc 0/200

Detailed display of VC(s) with VPI/VCI = 0/200

ATM0/2/0/0.1: VPI: 0 VCI: 200
CBR, PeakRate: 100000 Kbps
AAL5-LLC/SNAP
OAM frequency: 300 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 verified,
VC is managed by OAM,

OAM cells received: 2290,
F5 InEndLoop: 2290, F5 InSegLoop: 0,
F5 InEndAIS: 0,  F5 InSegAIS: 0,
F5 InEndRDI: 0,  F5 InSegRDI: 0,
OAM cells sent: 2290,
F5 OutEndLoop: 2290, F5 OutSegLoop: 0,
F5 OutEndAIS: 0,  F5 OutSegAIS: 0,
F5 OutEndRDI: 0,  F5 OutSegRDI: 0,
OAM cells drops: 0

InPkts: 2290                 OutPkts: 2290
InBytes: 119080              OutBytes: 119080
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 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)
```

Additional References

The following 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*, Release 3.4

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://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

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/techsupport

Configuring Bidirectional Forwarding Detection on Cisco IOS XR

Bidirectional forwarding detection (BFD) provides low-overhead, short-duration detection of failures in the path between adjacent forwarding engines. BFD allows a single mechanism to be used for failure detection over any media and at any protocol layer, with a wide range of detection times and overhead. The fast detection of failures provides immediate reaction to failure in the event of a failed link or neighbor.

Feature History for Configuring Bidirectional Forwarding Detection on Cisco IOS XR

Release	Modification
Release 3.2	This feature was introduced on the Cisco CRS-1 with support for the following features: • IPv4 asynchronous and echo modes over physical POS and Gigabit Ethernet numbered links and VLANs • BFD IPv4 single-hop • Distribution on line cards • BFD Version 0 and Version 1
Release 3.3.0	• Support was added to BFD for the following features: – BFD over bundled VLANs using static routes – Minimum disruption restart (MDR), which allows for a node CPU restart while minimizing traffic loss and network churn. – Fast reroute/Traffic engineering (FRR/TE) using BFD on Ethernet interfaces. • Configuration procedure was added to support the **clear bfd counters packet** and **show bfd counters packet** commands.
Release 3.3.1	BFD support was added on the Cisco XR 12000 Series Router.
Release 3.3.2	• The **echo disable** command was added to enable users to disable echo mode on routers or interfaces where BFD is used in conjunction with Unicast Reverse Path Forwarding (uRPF). • A new BFD configuration mode was added, under which users can disable echo mode. The **bfd** command was added to allow users to enter the new BFD configuration mode.

Release 3.4.0	OSPF and IS-IS were supported on BFD over bundle VLANs.
Release 3.5.0	No modifications.

Contents

Prerequisites for Implementing BFD

The following prerequisites are required to implement BFD:

- You must be in a user group associated with a task group that includes the proper task IDs for the BFD commands. Task IDs for commands are listed in the *Cisco IOS XR Interface and Hardware Component Command Reference*.

  For detailed information about user groups and task IDs, refer to the *Configuring AAA Services on Cisco IOS XR Software* module of *Cisco IOS XR System Security Configuration Guide*.
- A Cisco CRS-1 or Cisco XR 12000 Series Router that runs Cisco IOS XR software.
- If enabling BFD on Multiprotocol Label Switching (MPLS), an installed composite PIE file including the MPLS package, or a composite-package image is required. For Border Gateway Protocol (BGP), Intermediate System-to-Intermediate System (IS-IS), Static, and Open Shortest Path First (OSPF), an installed Cisco IOS XR IP Unicast Routing Core Bundle image is required.
- Interior Gateway Protocol (IGP) is activated on the router if you are using IS-IS or OSPF.
- On the Cisco CRS-1 platform, each line card supporting BFD must be able to perform the following tasks:
  - Send echo packets every 15 ms (under normal conditions)
  - Send control packets every 15 ms (under stress conditions)
  - Send and receive more than 6700 User Datagram Protocol (UDP) pps. This sustains 100 sessions at a 15-ms interval (or 1024 sessions at a 150-ms interval)
- On the Cisco XR 12000 Series Router platform, each line card supporting BFD must be able to perform the following tasks:
  - Send echo packets every 50 ms (under normal conditions)
  - Send control packets every 250 ms (under normal conditions)
  - Send and receive more than 1000 User Datagram Protocol (UDP) pps on the 12000 Series platform. This sustains 50 sessions at a 50-ms interval.
- To enable BFD for a neighbor, the neighbor router must support BFD.

- We recommend configuring the local router ID with the **router-id** command in global configuration mode prior to setting up a BFD session. If you do not configure the local router ID, then the source address of the IP packet is IP address of the output interface.

Information About BFD

Cisco IOS XR software supports BFD for the verification of IPv4 single-hop connectivity. Cisco IOS XR software supports both asynchronous mode and echo mode over physical numbered Packet-over-SONET/SDH (POS) and Gigabit Ethernet links, as follows:

- Echo mode is initiated only after a session is established using BFD control packets. BFD echo packets are transmitted over UDP/IPv4 using source and destination port 3785. The source address of the IP packet is the local router ID, and the destination address is the local interface address.

Note To configure the local router ID, use the **router-id** command in global configuration mode. If you do not configure the local router ID prior to setting up a BFD session, then the source address of the IP Packet is the IP address of the output interface.

- BFD asynchronous packets are transmitted over UDP and IPv4 using source port 49152 and destination port 3784. For asynchronous mode, the source address of the IP packet is the local interface address, and the destination address is the remote interface address.

Note Echo mode is not supported on VLAN bundles.

Caution If you are using BFD with Unicast Reverse Path Forwarding (uRPF) on a particular interface, then you need to use the **echo disable** command to disable echo mode on that interface; otherwise, echo packets will be rejected. You can disable echo mode for the entire router, or for an individual interface.

To enable or disable IPv4 uRPF checking on an IPv4 interface, use the **[no] ipv4 verify unicast source reachable-via** command in interface configuration mode. To enable or disable loose IPv6 uRPF checking on an IPv6 interface, use the **[no] ipv6 verify unicast source reachable-via any** command in interface configuration mode.

Keep the following in mind when configuring BFD on Cisco IOS XR software:

- BFD is a fixed-length hello protocol, in which each end of a connection transmits packets periodically over a forwarding path. Cisco IOS XR software supports BFD adaptive detection times.
- BFD can be used with the following applications:
  - BGP
  - IS-IS
  - OSPF
  - MPLS traffic-engineering (MPLS-TE)
  - Static
  - Protocol Independent Multicast (PIM)
- BFD is supported for connections over the following interface types:
  - Packet-over-SONET/SDH (POS)
  - Gigabit Ethernet (GigE)
  - Ten Gigabit Ethernet (TenGigE)
  - Virtual LAN (VLAN).

- Cisco IOS XR software supports BFD Version 0 and Version 1. BFD sessions are established using either version, depending upon the neighbor. BFD Version 1 is the default version and is tried initially for session creation.
- BFD is supported on IPv4 directly connected external BGP peers.

The user can perform the following tasks on the router:

- Configure BFD parameters (desired interval and detect multiplier) in the application configuration space.
- Display BFD operational status (state, counters, tracing, and so forth)
- Clear BFD counters

BFD on Bundled VLANs

BFD is supported on bundled VLANS using static routing, IS-IS, and OSPF. When running a BFD session on a bundled VLAN interface, the BFD session is active as long as the VLAN bundle is up.

As long as the VLAN bundle is active, the following events do not cause the BFD session to fail:

- Failure of a component link
- Online insertion and removal (OIR) of a line card which hosts one or more of the component links
- Addition of a component link (by configuration) to the bundle
- Removal of a component link (by configuration) from the bundle
- Shutdown of a component link
- RP failover

Note For more information on configuring a VLAN bundle, see the *Configuring Link Bundling on Cisco IOS XR Software* module.

Keep the following in mind when configuring BFD over bundled VLANs:

- In the case of an RP failover, configured next hops are registered in the Routing Information Base (RIB).
- In the case of a BFD restart, static routes remain in the RIB. BFD sessions are reestablished when BFD restarts.

Note Static BFD sessions are supported on peers with address prefixes whose next hops are directly connected to the router.

BFD Packet Formats

BFD payload control packets are encapsulated in UDP packets, using destination port 3784 and source port 49152. Even on shared media, like Ethernet, BFD control packets are always sent as unicast packets to the BFD peer.

Echo packets are encapsulated in UDP packets, as well, using destination port 3785 and source port 3785.

Restrictions

- Echo mode is not supported on bundle VLANs.
- If you are using BFD with uRPF on a particular interface, then you need to use the **echo disable** command to disable echo mode on that interface; otherwise, echo packets will be rejected. You can disable echo mode for the entire router, or for an individual interface.
- Only the static, OSPF, and IS-IS applications are supported on BFD over bundle VLANs.

Configuring BFD

The following procedures describe how to configure BFD for BGP. BFD can be enabled per neighbor, or per interface. To enable BFD per neighbor, use the steps in the "Enabling BFD on a Neighbor" section on page 59. To enable BFD per interface, use the steps in the "Enabling BFD on an Interface" section on page 62.

Note FRR/TE using BFD is supported on POS interfaces and Ethernet interfaces.

Enabling BFD on a Neighbor

The following procedures describe how to configure BFD for BGP on a neighbor router.

Note BFD neighbor router configuration is supported for BGP only.

SUMMARY STEPS

1. **configure**
2. **router bgp** *autonomous-system-number*
3. **bfd minimum-interval** *milliseconds*
4. **bfd multiplier** *multiplier*
5. **neighbor** *ip-address*
6. **remote-as** *autonomous-system-number*
7. **bfd fast-detect**
8. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**router bgp** *autonomous-system-number* **Example:** `RP/0/RP0/CPU0:router(config)# router bgp 120`	Enters BGP configuration mode, allowing you to configure the BGP routing process. Use the **show bgp** command in EXEC mode to obtain the *autonomous-system-number* for the current router.
Step 3	**bfd minimum-interval** *milliseconds* **Example:** `RP/0/RP0/CPU0:router(config-bgp)# bfd minimum-interval 6500`	Sets the BFD minimum interval. Range is 15-30000 milliseconds. This example sets the BFD minimum interval to 6500 milliseconds.
Step 4	**bfd multiplier** *multiplier* **Example:** `RP/0/RP0/CPU0:router(config-bgp)# bfd multiplier 7`	Sets the BFD multiplier. This example sets the BFD multiplier to 7.
Step 5	**neighbor** *ip-address* **Example:** `RP/0/RP0/CPU0:router(config-bgp)# neighbor 172.168.40.24`	Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address as a BGP peer. This example configures the IP address 172.168.40.24 as a BGP peer.
Step 6	**remote-as** *autonomous-system-number* **Example:** `RP/0/RP0/CPU0:router(config-bgp-nbr)# remote-as 2002`	Creates a neighbor and assigns it a remote autonomous system. This example configures the remote autonomous system to be 2002.

	Command or Action	Purpose
Step 7	**bfd fast-detect** **Example:** `RP/0/RP0/CPU0:router(config-bgp-nbr)# bfd fast-detect`	Enables BFD between the local networking devices and the neighbor whose IP address you configured to be a BGP peer in Step 5. In the example in Step 5, the IP address 172.168.40.24 was set up as the BGP peer. In this example, BFD is enabled between the local networking devices and the neighbor 172.168.40.24.
Step 8	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router (config-bgp-nbr)# end` or `RP/0/RP0/CPU0:router(config-bgp-nbr)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Enabling BFD on an Interface

The following procedures describe how to configure BFD for Open Shortest Path First (OSPF) on an interface. The steps in the procedure are common to the steps for configuring BFD on IS-IS and MPLS-TE; only the command mode differs.

Note BFD per interface configuration is supported for OSPF, IS-IS, and MPLS-TE only.

SUMMARY STEPS

1. **configure**
2. **router ospf** *process-name*
3. **bfd minimum-interval** *milliseconds*
4. **bfd multiplier** *multiplier*
5. **area** *area-id*
6. **interface** *type instance*
7. **bfd fast-detect**
8. **end**
   or
   commit
9. **show run router ospf**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**router ospf** *process-name* **Example:** `RP/0/RP0/CPU0:router(config)# router ospf 0`	Enters OSPF configuration mode, allowing you to configure the OSPF routing process. Use the **show ospf** command in EXEC mode to obtain the process-name for the current router. **Note** To configure BFD for IS-IS or MPLS-TE, enter the corresponding configuration mode. For example, for MPLS-TE, enter MPLS-TE configuration mode.
Step 3	**bfd minimum-interval** *milliseconds* **Example:** `RP/0/RP0/CPU0:router(config-bgp)# bfd minimum-interval 6500`	Sets the BFD minimum interval. Range is 15-30000 milliseconds. This example sets the BFD minimum interval to 6500 milliseconds.

	Command or Action	Purpose
Step 4	**bfd multiplier** *multiplier* **Example:** `RP/0/RP0/CPU0:router(config-bgp)# bfd multiplier 7`	Sets the BFD multiplier. This example sets the BFD multiplier to 7.
Step 5	**area** *area-id* **Example:** `RP/0/RP0/CPU0:router(config-bgp)# area 0`	Configures an Open Shortest Path First (OSPF) area. Replace *area-id* with the OSPF area identifier.
Step 6	**interface** *type instance* **Example:** `RP/0/RP0/CPU0:router(config-ospf-ar)# interface gigabitEthernet 0/3/0/1`	Enters interface configuration mode and specifies the interface name and notation *rack/slot/module/port*. • The example indicates a Gigabit Ethernet interface in modular services card slot 3.
Step 7	**bfd fast-detect** **Example:** `RP/0/RP0/CPU0:router(config-ospf-ar-if)# bfd fast-detect`	Enables BFD to detect failures in the path between adjacent forwarding engines.
Step 8	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router (config-bgp-nbr)# end` or `RP/0/RP0/CPU0:router(config-bgp-nbr)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 9	**show run router ospf** **Example:** `RP/0/RP0/CPU0:router (config-bgp-nbr)# show run router ospf`	Verify that BFD is enabled on the appropriate interface.

Enabling BFD on a Static Route

The following procedure describes how to enable BFD on a static route.

Note Bundle VLAN sessions are restricted to an interval of 250 milliseconds and a multiplier of 3. More aggressive parameters are not allowed.

SUMMARY STEPS

1. **configure**
2. **router static**
3. **address-family ipv4 unicast** *address nexthop* **bfd fast-detect [minimum interval** *interval***] [multiplier** *multiplier*]
4. **vrf** *vrf-name*
5. **address-family ipv4 unicast** *address nexthop* **bfd fast-detect** [**minimum interval** *interval*] [**multiplier** *multiplier*]
6. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**router static** **Example:** `RP/0/RP0/CPU0:router(config)# router static`	Enters static route configuration mode, allowing you to configure static routing.

	Command or Action	Purpose
Step 3	**address-family ipv4 unicast** *address nexthop* **bfd fast-detect** [**minimum-interval** *interval*] [**multiplier** *multiplier*] ***Example:*** `RP/0/RP0/CPU0:router(config-static)# address-family ipv4 unicast 0.0.0.0/0 2.6.0.1 bfd fast-detect minimum-interval 1000 multiplier 5`	Enables BFD fast-detection on the specified IPV4 unicast destination address prefix and on the forwarding next-hop address. **Note** Include the optional **minimum-interval** keyword and argument to ensure that the next hop is assigned with the same hello interval. Replace the *interval* argument with a number that specifies the interval in milliseconds. Range is from 15 through 5000. **Note** Include the optional **multiplier** keyword argument to ensure that the next hop is assigned with the same detect multiplier. Replace the *multiplier* argument with a number that specifies the detect multiplier. Range is from 2 through 10. **Note** Bundle VLAN sessions are restricted to an interval of 250 milliseconds and a multiplier of 3. More aggressive parameters are not allowed.
Step 4	**vrf** *vrf-name* **Example:** `RP/0/RP0/CPU0:router(config-static)# vrf vrf1`	Specifies a VPN routing and forwarding (VRF) instance, and enters static route configuration mode for that VRF.
Step 5	**address-family ipv4 unicast** *address nexthop* **bfd fast-detect** **Example:** `RP/0/RP0/CPU0:router(config-static-vrf)# address-family ipv4 unicast 0.0.0.0/0 2.6.0.2`	Enables BFD fast-detection on the specified IPV4 unicast destination address prefix and on the forwarding next-hop address.
Step 6	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router (config-static-vrf)# end` or `RP/0/RP0/CPU0:router(config-static-vrf)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found. Commit them?` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the user in the same command mode without committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Disabling Echo mode on a Router

The following procedures describe how to disable echo mode on router that is using BFD in conjunction with uRPF.

Note To enable or disable IPv4 uRPF checking on an IPv4 interface, use the **[no] ipv4 verify unicast source reachable-via** command in interface configuration mode. To enable or disable loose IPv6 uRPF checking on an IPv6 interface, use the **[no] ipv6 verify unicast source reachable-via any** command in interface configuration mode.

SUMMARY STEPS

1. **configure**
2. **bgp**
3. **echo disable**
4. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**bfd** **Example:** `RP/0/RP0/CPU0:router(config)# bfd`	Enters BFD configuration mode.

	Command or Action	Purpose
Step 3	**echo disable** **Example:** `RP/0/RP0/CPU0:router(config-bfd)# echo disable`	Disables echo mode on the router.
Step 4	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router (config-bfd)# end` or `RP/0/RP0/CPU0:router(config-bfd)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Disabling Echo mode on an Individual Interface or Bundle

The following procedures describe how to disable echo mode on an interface or bundle that is using BFD in conjunction with uRPF.

Note To enable or disable IPv4 uRPF checking on an IPv4 interface, use the **[no] ipv4 verify unicast source reachable-via** command in interface configuration mode. To enable or disable loose IPv6 uRPF checking on an IPv6 interface, use the **[no] ipv6 verify unicast source reachable-via any** command in interface configuration mode.

SUMMARY STEPS

1. **configure**
2. **bgp**
3. **interface** *type instance*
4. **echo disable**
5. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**bfd** **Example:** `RP/0/RP0/CPU0:router(config)# bfd`	Enters BFD configuration mode.
Step 3	**interface** *type instance* **Example:** `RP/0/RP0/CPU0:router(config-bfd)# interface gigabitEthernet 0/1/5/0`	Enters BFD interface configuration mode for a specific interface or bundle. In BFD interface configuration mode, you can disable echo mode on an individual interface or bundle.
Step 4	**echo disable** **Example:** `RP/0/RP0/CPU0:router(config-bfd-if)# echo disable`	Disables echo mode on the router.
Step 5	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router (config-bfd-if)# end` or `RP/0/RP0/CPU0:router(config-bfd-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Clearing and Displaying BFD Counters

The following procedure describes how to display and clear BFD packet counters. You can clear packet counters for BFD sessions that are hosted on a specific node or on a specific interface.

SUMMARY STEPS

1. **show bfd counters packet** [**interface** *type instance*] **location** *node-id*
2. **clear bfd counters packet** [**interface** *type instance*] **location** *node-id*
3. **end**
   or
   commit
4. **show bfd counters packet** [**interface** *type instance*] **location** *node-id*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**show bfd counters packet** [**interface** *type instance*] **location** *node-id* **Example:** `RP/0/RP0/CPU0:router# show bfd counters packet location 0/3/cpu0`	Displays BFD counters.
Step 2	**clear bfd counters packet** [**interface** *type instance*] **location** *node-id* **Example:** `RP/0/RP0/CPU0:router# clear bfd counters packet interface POS 0/5/0/1 location 0/5/cpu0`	Clears Bidirectional Forwarding Detection (BFD) counters.
Step 3	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router (config-bgp-nbr)# end` or `RP/0/RP0/CPU0:router(config-bgp-nbr)# commit`	Saves configuration changes. When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the user in the same command mode without committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 4	**show bfd counters packet** [**interface** *type instance*] **location** *node-id* **Example:** `RP/0/RP0/CPU0:router# show bfd counters packet location 0/3/cpu0`	Verifies that all BFD counters are cleared.

Configuration Examples for Configuring Bidirectional Forwarding Detection on Cisco IOS XR

This section provides the following configuration example:

Bidirectional Forwarding Detection: Example

The following example shows how to configure BFD between autonomous system 65000 and neighbor 192.168.70.24:

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# router bgp 65000
RP/0/RP0/CPU0:router(config-bgp)# bfd multiplier 2
RP/0/RP0/CPU0:router(config-bgp)# bfd minimum-interval 20
RP/0/RP0/CPU0:router(config-bgp)# neighbor 192.168.70.24
RP/0/RP0/CPU0:router(config-bgp-nbr)# remote-as 2
RP/0/RP0/CPU0:router(config-bgp-nbr)# bfd fast-detect
```

The following example shows how to enable BFD on a Gigabit Ethernet interface:

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# router ospf 0
RP/0/0/CPU0:router(config-ospf)# area 0
RP/0/0/CPU0:router(config-ospf-ar)# interface gigabitEthernet 0/3/0/1
RP/0/0/CPU0:router(config-ospf-ar-if)# bfd fast-detect
RP/0/0/CPU0:router(config-ospf-ar-if)# commit
RP/0/0/CPU0:Dec 2 07:06:48.508 : config[65685]: %MGBL-LIBTARCFG-6-COMMIT : Configuration
committed by user 'xxx'. Use 'show configuration commit changes 1000001134' to view the
changes.
RP/0/0/CPU0:router(config-ospf-ar-if)# end
RP/0/0/CPU0:Dec 2 07:06:48.848 : config[65685]: %MGBL-SYS-5-CONFIG_I : Configured from
console by lab
RP/0/0/CPU0:router# show run router ospf
router ospf 0
area 0
interface GigabitEthernet0/3/0/1
bfd fast-detect
!
```

The following example shows how to enable BFD on a static route. In this example, BFD sessions are established with the next-hop 3.3.3.3 when it becomes reachable.

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# router static
RP/0/RP0/CPU0:router (config-static)# address-family ipv4 unicast 2.2.2.0/24 3.3.3.3 bfd
fast-detection
RP/0/RP0/CPU0:router(config-static)# end
```

The following example shows how to disable echo mode on a router:

```
RP/0/RP0/CPU0:router # configure

RP/0/RP0/CPU0:router(config)# bfd

RP/0/RP0/CPU0:router(config-bfd)# echo disable
```

The following example shows how to disable echo mode on an interface:

```
RP/0/RP0/CPU0:router # configure

RP/0/RP0/CPU0:router(config)# bfd

RP/0/RP0/CPU0:router(config-bfd)# interface pos 0/1/0/0

RP/0/RP0/CPU0:router(config-bfd-if)# echo disable
```

Where to Go Next

BFD is supported over multiple platforms; refer to the related command reference document for more detailed information about these commands.

- *BGP Commands on Cisco IOS XR Software*
- *IS-IS Commands on Cisco IOS XR Software*
- *OSPF Commands on Cisco IOS XR Software*
- *Static Routing Commands on Cisco IOS XR Software*
- *MPLS Traffic Engineering Commands on Cisco IOS XR Software*

Additional References

The following sections provide references related to implementing BFD for Cisco IOS XR software.

Related Documents

Related Topic	Document Title
BFD commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS XR Interface and Hardware Command Reference*, Release 3.2

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.	—

RFCs

RFCs	Title
draft-ietf-bfd-base-06	*Bidirectional Forwarding Detection,* March, 2007
draft-ietf-bfd-v4v6-1hop-06	*BFD for IPv4 and IPv6 (Single Hop),* March, 2007

Technical Assistance

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

Configuring Link Bundling on Cisco IOS XR Software

On the CRS-1 Series, a bundle is a group of one or more ports that are aggregated together and treated as a single link. The different links within a single bundle can have varying speeds, where the fastest link can be a maximum of four times greater than the slowest link. Each bundle has a single MAC, a single IP address, and a single configuration set (such as ACLs or QoS).

The CRS-1 Series supports bundling for the following types of interfaces:

- Packet-over-SONET/SDH (POS) interfaces
- Ethernet interfaces
- VLAN subinterfaces

Note Bundles do not have a one-to-one modular services card association.

Feature History for Configuring Link Bundling on Cisco IOS XR Software

Release	Modification
Release 3.2	This feature was introduced on the Cisco CRS-1.
Release 3.3.0	This feature was updated as follows: • To support the 1:N redundancy feature, users can configure the minimum number of active links using the **bundle minimum-active links** command. • To support the 1:N redundancy feature, users can configure the minimum bandwidth in kbps using the **bundle minimum-active links** command. • Support was added for VLAN subinterfaces on Ethernet link bundles. • Output for **show bundle bundle-Ether** command and **show bundle bundle-POS** command was modified. • The **reasons** keyword was added to the **show bundle bundle-Ether** command and the **show bundle bundle-POS** command. • The **bundle id** command was changed from **bundle-id**. • BFD over bundled VLANs using static routes.

Release 3.4.0	The configuration procedures in this chapter were modified with enhancements.
Release 3.5.0	No modifications.

Contents

This chapter includes the following sections:

Prerequisites for Configuring CRS-1 Series Link Bundling

Before configuring Link Bundling, be sure 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 for the bundle commands. Task IDs for commands are listed in the *Cisco IOS XR Interface and Hardware Component Command Reference*.
- You know the interface IP address.
- You know which links should be included in the bundle you are configuring.
- If you are configuring an Ethernet link bundle, you have at least one of the following Ethernet cards installed in the router:
  - 8-port 10-Gigabit Ethernet physical layer interface modules (PLIM) modular services card
  - 8-port Gigabit Ethernet shared port adapter (SPA)
- If you are configuring an POS link bundle, you have a POS line card or SPA installed in a router that is running Cisco IOS XR software.

Note For more information about physical interfaces, PLIMs, and modular services cards, refer to the *Cisco CRS-1 Carrier Routing System 8-Slot Line Card Chassis System Description*.

Information About Configuring CRS-1 Series Link Bundling

To implement Link Bundling, you must understand the following concepts:

Link Bundling Overview

A link bundle is simply a group of ports that are bundled together and act as a single link. The advantages of link bundles are as follows:

- Multiple links can span several line cards 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 OC48c PLIM modular services card to an OC192 PLIM modular services card without interrupting traffic.

Although the individual links within a single bundle can have varying speeds, 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.

Characteristics of CRS-1 Series Link Bundles

The following list describes the properties and limitations of link bundles on CRS-1 Series:

- Both POS and Ethernet interfaces can be bundled, with or without the use of LACP.
- Bundle membership can span across several modular services cards that are installed in a single router.
- A single bundle supports maximum of 32 physical links. If you add more than 32 links to a bundle, only 32 of the links function, and the remaining links are automatically disabled.
- A single CRS-1 Series router supports a maximum of 32 bundles.
- Different link speeds are allowed within a single bundle, with a maximum of four times the speed difference between the members of the bundle.
- 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.
- A bundle can be administratively enabled or disabled.
- Each individual link within a bundle can be administratively enabled or disabled.

- Ethernet and POS link bundles are created in the same way as Ethernet channels and POS channels, where the user enters the same configuration on both end systems.
- The MAC address that is set on the bundle becomes the MAC address of the links within that bundle.
- 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. Data is distributed to a link in proportion to the bandwidth of the link in relation to its bundle.
- HDLC is the only supported encapsulation type for POS link bundles in Cisco IOS XR software. POS links that are configured with any other encapsulation type cannot join a bundle. Keep in mind that all POS link bundle members must be running HDLC for HDLC to work on a bundle.
- QoS is supported and is applied proportionally on each bundle member.
- Link layer protocols, such as CDP and HDLC keepalives, work independently on each link within a bundle.
- 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 link must be in the up state before it can be added to a bundle.
- All links within a single bundle must be configured to run either POS Channel or 802.3ad. Mixed bundles are not supported.
- A bundle can contain physical links and VLAN subinterfaces only. Tunnels cannot be bundle members.
- Access Control List (ACL) configuration on link bundles is identical to ACL 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.

Link Aggregation Through LACP

Aggregating interfaces on different modular 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. For the CRS-1 Series router, 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.

QoS and Link Bundling

On the ingress direction, QoS is applied to the local instance of a bundle. Each bundle is associated with a set of queues. QoS is applied to the various network layer protocols that are configured on the bundle.

On the egress direction, QoS is applied on the bundle with a reference to the member links. QoS is applied based on the sum of the member bandwidths.

Link Bundling supports the following QoS features:

- hi priority /lo priority—Maximum bandwidth is calculated as a percentage of the bundle interface bandwidth. This percentage is then applied to every member link on the egress, or to the local bundle instance on ingress.
- guaranteed bandwidth—Provided in percentage and applied to every member link.
- traffic shaping—Provided in percentage and applied to every member link.
- WRED—Minimum and maximum parameters are converted to the right proportion per member link or bundle instance, and then are applied to the bundle.
- marking—Process of changing the packet QoS level according to a policy.
- tail drop— Packets are dropped when the queue is full.

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 128.
- The maximum number of bundled VLANs allowed per router is 4000.

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 *instance.subinterface*

After you create a VLAN on an Ethernet link bundle, all physical VLAN subinterface configuration is supported on that link bundle.

Link Bundle Configuration Overview

The following steps provide a general overview of the link bundle configuration process. 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 to be a member of a bundle from the interface configuration submode for that link.

Nonstop Forwarding During Card Failover

Cisco IOS XR software supports nonstop forwarding during failover between active and standby paired RP cards. Nonstop forwarding ensures that there is no change in the state of the link bundles when a failover occurs.

For example, if an active RP fails, the standby RP becomes operational. If a failover occurs on the active node, you must perform a restart on the redundant node to enable the failover. 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 Failover is always onto the standby RP. If there is no standby pair, the configuration manager restores the configuration to another RP, thereby recreating the bundles and other virtual interfaces.

Note You do not need to configure anything to guarantee that the standby interface configurations are maintained.

Link Failover

When one member link in a bundle fails, traffic is redirected to the remaining operational member links and traffic flow remains uninterrupted.

How to Configure Link Bundling

This section contains the following procedures:

- Configuring Ethernet Link Bundles, page 79
- Configuring VLAN Bundles, page 83
- Configuring POS Link Bundles, page 90

Configuring Ethernet Link Bundles

This section describes how to configure a 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*
7. **exit**
8. **interface** {**GigabitEthernet** | **TenGigE**} *instance*
9. **bundle id** *bundle-id* [**mode** {**active** | **on** | **passive**}
10. **no shutdown**

11. **exit**
12. Repeat Step 8 through Step 11 to add more links to the bundle you created in Step 2.
13. **end**
    or
    commit
14. **exit**
15. **exit**
16. Perform Step 1 through Step 15 on the remote end of the connection.
17. **show bundle Bundle-Ether** *bundle-id* [**reasons**]
18. **show lacp bundle Bundle-Ether** *bundle-id*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface Bundle-Ether** *bundle-id* **Example:** `RP/0/RP0/CPU0:router#(config)# interface Bundle-Ether 3`	Creates and names a new Ethernet link bundle. 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* **Example:** `RP/0/RP0/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* **Example:** `RP/0/RP0/CPU0:router(config-if)# bundle minimum-active bandwidth 580000`	Sets the minimum amount of bandwidth required before a user can bring up a bundle.
Step 5	**bundle minimum-active links** *links* **Example:** `RP/0/RP0/CPU0:router(config-if)# bundle minimum-active links 2`	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* **Example:** `RP/0/RP0/CPU0:router(config-if)# bundle maximum-active links 1`	(Optional) Limits the number of links that can be actively carrying traffic in a specific bundle. **Note** The default number of active links allowed in a single bundle is 32. **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** **Example:** `RP/0/RP0/CPU0:router(config-if)# exit`	Exits interface configuration submode for the Ethernet link bundle.
Step 8	**interface** {**GigabitEthernet** | **TenGigE**} *instance* **Example:** `RP/0/RP0/CPU0:router(config)# interface GigabitEthernet 1/0/0/0`	Enters the interface configuration mode for the specified interface. Enter the **GigabitEthernet** or **TenGigE** keyword to specify the interface type. Replace the *instance* argument with the node-id in the *rack*/*slot*/*module* format.
Step 9	**bundle id** *bundle-id* **[mode {active | on | passive}]** **Example:** `RP/0/RP0/CPU0:router(config-if)# bundle-id 3`	Adds the link to the specified bundle. To enable active or passive LACP on the bundle, include the optional **mode active** or **mode passive** keywords in the command string. To add the link to the bundle without LACP support, include the optional **mode on** keywords with the command string. **Note** If you do not specify the **mode** keyword, the default mode is **on** (LACP is not run over the port).
Step 10	**no shutdown** **Example:** `RP/0/RP0/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 11	**exit** **Example:** `RP/0/RP0/CPU0:router(config-if)# exit`	Exits interface configuration submode for the Ethernet interface.

	Command or Action	Purpose
Step 12	**interface** {**GigabitEthernet** | **TenGigE**} *number* **bundle id** *bundle-id* [**mode** {**active** | **passive** | **on**}] **no shutdown** **exit** **Example:** `RP/0/RP0/CPU0:router(config)# interface GigabitEthernet 1/0/2/1` `RP/0/RP0/CPU0:router(config-if)# bundle id 3` `RP/0/RP0/CPU0:router(config-if)# no shutdown` `RP/0/RP0/CPU0:router(config-if)# exit` `RP/0/RP0/CPU0:router(config)# interface GigabitEthernet 1/0/2/3` `RP/0/RP0/CPU0:router(config-if)# bundle id 3` `RP/0/RP0/CPU0:router(config-if)# no shutdown` `RP/0/RP0/CPU0:router(config-if)# exit`	(Optional) Repeat Step 8 through Step 11 to add more links to the bundle.
Step 13	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router(config-if)# end` or `RP/0/RP0/CPU0:router(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 14	**exit** **Example:** `RP/0/RP0/CPU0:router(config-if)# exit`	Exits interface configuration mode.

	Command or Action	Purpose
Step 15	**exit** **Example:** `RP/0/RP0/CPU0:router(config)# exit`	Exits global configuration mode.
Step 16	Perform Step 1 through Step 15 on the remote end of the connection.	Brings up the other end of the link bundle.
Step 17	**show bundle Bundle-Ether** *bundle-id* [**reasons**] **Example:** `RP/0/RP0/CPU0:router# show bundle Bundle-Ether 3 reasons`	(Optional) Shows information about the specified Ethernet link bundle.
Step 18	**show lacp bundle Bundle-Ether** *bundle-id* **Example:** `RP/0/RP0/CPU0:router# show lacp bundle Bundle-Ether 3`	(Optional) Shows detailed information about LACP ports and their peers.

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.

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*
7. **exit**
8. **interface Bundle-Ether** *bundle-id.vlan-id*
9. **dot1q vlan** *vlan-id*
10. **ipv4 address** *ipv4-address mask*

11. **no shutdown**
12. **exit**
13. Repeat Step 7 through Step 12 to add more VLANS to the bundle you created in Step 2.
14. **end**
    or
    commit
15. **exit**
16. **exit**
17. **show vlan interface**
18. **configure**
19. **interface** {**GigabitEthernet** | **TenGigE**} *instance*
20. **bundle id** *bundle-id* [**mode** {**active** | **on** | **passive**}
21. **no shutdown**
22. Repeat Step19 through Step 21 to add more Ethernet Interfaces to the bundle you created in Step 2.
23. **end**
    or
    commit
24. Perform Step 1 through Step 23 on the remote end of the connection.
25. **show bundle Bundle-Ether** *bundle-id* [**reasons**]
26. **show vlan trunks** [{**GigabitEthernet** | **TenGigE** | **Bundle-Ether**} *instance*] [**brief** | **summary**] [**location** *node-id*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface Bundle-Ether** *bundle-id* **Example:** `RP/0/RP0/CPU0:router#(config)# interface Bundle-Ether 3`	Creates and names a new Ethernet link bundle. 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* **Example:** `RP/0/RP0/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.

	Command or Action	Purpose
Step 4	**bundle minimum-active bandwidth** *kbps* **Example:** `RP/0/RP0/CPU0:router(config-if)# bundle minimum-active bandwidth 580000`	Sets the minimum amount of bandwidth required before a user can bring up a bundle.
Step 5	**bundle minimum-active links** *links* **Example:** `RP/0/RP0/CPU0:router(config-if)# bundle minimum-active links 2`	Sets the number of active links required before you can bring up a specific bundle.
Step 6	**bundle maximum-active links** *links* **Example:** `RP/0/RP0/CPU0:router(config-if)# bundle maximum-active links 1`	(Optional) Limits the number of links that can be actively carrying traffic in a specific bundle. **Note** The default number of active links allowed in a single bundle is 32. **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** **Example:** `RP/0/RP0/CPU0:router(config-if)# exit`	Exits the interface configuration submode.
Step 8	**interface Bundle-Ether** *bundle-id.vlan-id* **Example:** `RP/0/RP0/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. Replace the *bundle-id* argument with the *bundle-id* you created in Step 2. Replace the *vlan-id* with a subinterface identifier. Range is from 1 to 4094 inclusive (0 and 4095 are reserved). **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* **Example:** `RP/0/RP0/CPU0:router#(config-subif)# dot1q vlan 10`	Assigns a VLAN to the subinterface. Replace the *vlan-id* argument with a subinterface identifier. Range is from 1 to 4094 inclusive (0 and 4095 are reserved).

	Command or Action	Purpose
Step 10	**ipv4 address** *ipv4-address mask* **Example:** `RP/0/RP0/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** **Example:** `RP/0/RP0/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 12	**exit** **Example:** `RP/0/RP0/CPU0:router(config-subif)# exit`	Exits subinterface configuration mode for the VLAN subinterface.
Step 13	**interface Bundle-Ether** *bundle-id.vlan-id* **dot1q vlan** *vlan-id* **ipv4 address** *ipv4-address mask* **no shutdown** **exit** **Example:** `RP/0/RP0/CPU0:router(config-subif)# interface Bundle-Ether 3.1` `RP/0/RP0/CPU0:router(config-subif)# dot1q vlan 20` `RP/0/RP0/CPU0:router(config-subif)# ipv4 address 20.2.3.4/24` `RP/0/RP0/CPU0:router(config-subif)# no shutdown` `exit`	(Optional) Adds more subinterfaces to the bundle.

	Command or Action	Purpose
Step 14	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router(config-subif)# end` or `RP/0/RP0/CPU0:router(config-subif)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the commit command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 15	**exit** **Example:** `RP/0/RP0/CPU0:router(config-subif)# end`	Exits interface configuration mode.
Step 16	**exit** **Example:** `RP/0/RP0/CPU0:router(config)# exit`	Exits global configuration mode.
Step 17	**show vlan interface** **Example:** `RP/0/RP0/CPU0:router # show vlan interface`	Displays the current VLAN interface and status configuration.
Step 18	**configure** **Example:** `RP/0/RP0/CPU0:router # configure`	Enters global configuration mode.

	Command or Action	Purpose
Step 19	**interface** {**GigabitEthernet** | **TenGigE**} *instance* **Example:** `RP/0/RP0/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. Enter the **GigabitEthernet** or **TenGigE** keyword to specify the interface type. Replace the *instance* argument with the node-id in the rack/slot/module format. **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}]** **Example:** `RP/0/RP0/CPU0:router(config-if)# bundle-id 3`	Adds an Ethernet interface to the bundle you configured in Step 2 through Step 13. To enable active or passive LACP on the bundle, include the optional **mode active** or **mode passive** keywords in the command string. To add the interface to the bundle without LACP support, include the optional **mode on** keywords with the command string. **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** **Example:** `RP/0/RP0/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** or **commit** **Example:** `RP/0/RP0/CPU0:router(config-subif)# end` or `RP/0/RP0/CPU0:router(config-subif)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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.

	Command or Action	Purpose
Step 25	**show bundle Bundle-Ether** *bundle-id* [**reasons**] **Example:** `RP/0/RP0/CPU0:router# show bundle Bundle-Ether 3 reasons`	(Optional) Shows information about the specified Ethernet link bundle. 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 vlan trunks** [{**GigabitEthernet** | **TenGigE** | **Bundle-Ether**} *instance*] [**brief** | **summary**] [**location** *node-id*] **Example:** `RP/0/RP0/CPU0:router# show vlan trunk summary`	(Optional) Displays summary information about each of the VLAN trunk interfaces. • The keywords have the following meanings: – **brief—**Displays a brief summary. – **summary—**Displays a full summary. – **location—**Displays information about the VLAN trunk interface on the given slot. – **interface—**Displays information about the specified interface or subinterface. Use the **show vlan trunks** command to verify that all configured VLAN subinterfaces on an Ethernet bundle are "up."

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 the following 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*
7. **exit**
8. **interface POS** *number*

9. **bundle id** *bundle-id* [**mode** {**active** | **on** | **passive**}
10. **no shutdown**
11. **exit**
12. Repeat Step 8 through Step 11 to add more links to the bundle you created in Step 2.
13. **end**
    or
    commit
14. **exit**
15. **exit**
16. Perform Step 1 through Step 15 on the remote end of the connection.
17. **show bundle Bundle-POS** *bundle-id* [**reasons**]
18. **show lacp bundle Bundle-POS** *bundle-id*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface Bundle-POS** *bundle-id* **Example:** `RP/0/RP0/CPU0:router#(config)#interface Bundle-POS 2`	Configures and names the new bundled POS interface. This interface command will enter you into the interface configuration submode, from where 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* **Example:** `RP/0/RP0/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* **Example:** `RP/0/RP0/CPU0:router(config-if)# bundle minimum-active bandwidth 620000`	Sets the minimum amount of bandwidth required before a user can bring up a bundle.
Step 5	**bundle minimum-active links** *links* **Example:** `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.

	Command or Action	Purpose
Step 6	**bundle maximum-active links** *links* **Example:** `RP/0/RP0/CPU0:router(config-if)# bundle maximum-active links 1`	(Optional) Limits the number of links that can be actively carrying traffic in a specific bundle. **Note** The default number of active links allowed in a single bundle is 32. **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 **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**	Exits the interface configuration submode.
Step 8	**interface POS** *instance* **Example:** `RP/0/RP0/CPU0:router(config)# interface POS 0/1/0/0`	Enters the POS interface configuration mode and specifies the POS interface name and instance notation *rack/slot/module/port*.
Step 9	**bundle id** *bundle-id* [**mode** {**active** | **passive** | **on**}] **Example:** `RP/0/RP0/CPU0:router(config-if)# bundle-id 3`	Adds the link to the specified bundle. To enable active or passive LACP on the bundle, include the optional **mode active** or **mode passive** keywords in the command string. To add the link to the bundle without LACP support, include the optional **mode on** keywords with the command string. **Note** If you do not specify the **mode** keyword, the default mode is **on** (LACP is not run over the port).
Step 10	**no shutdown** **Example:** `RP/0/RP0/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 11	**exit** **Example:** `RP/0/RP0/CPU0:router# exit`	Exits the interface configuration submode for the POS interface.
Step 12	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 13	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router(config-if)# end` or `RP/0/RP0/CPU0:router(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the commit command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 14	**exit** **Example:** `RP/0/RP0/CPU0:router(config-if)# exit`	Exits interface configuration mode.
Step 15	**exit** **Example:** `RP/0/RP0/CPU0:router(config)# exit`	Exits global configuration mode.
Step 16	Perform Step 1 through Step 15 on the remote end of the connection.	Brings up the other end of the link bundle.
Step 17	**show bundle Bundle-POS** *number* [**reasons**] **Example:** `RP/0/RP0/CPU0:router# show bundle Bundle-POS 1 reasons`	(Optional) Shows information about the specified POS link bundle.
Step 18	**show lacp bundle Bundle-POS** *bundle-id* **Example:** `RP/0/RP0/CPU0:router# show lacp bundle Bundle-POS 3`	(Optional) Shows detailed information about LACP ports and their peers.

Configuration Examples for CRS-1 Series Link Bundling

The following example shows how to join two ports to form an EtherChannel bundle running LACP:

```
RP/0/RP0/CPU0:Router# config
RP/0/RP0/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/RP0/CPU0:Router(config-if)# exit
RP/0/RP0/CPU0:Router(config)# interface TenGigE 0/3/0/0
RP/0/RP0/CPU0:Router(config-if)# bundle id 3 mode active
RP/0/RP0/CPU0:Router(config-if)# no shutdown
RP/0/RP0/CPU0:Router(config)# exit
RP/0/RP0/CPU0:Router(config)# interface TenGigE 0/3/0/1
RP/0/RP0/CPU0:Router(config-if)# bundle id 3 mode active
RP/0/RP0/CPU0:Router(config-if)# no shutdown
RP/0/RP0/CPU0:Router(config-if)# exit
```

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/RP0/CPU0:Router(config-subif)# dot1q vlan 10
RP/0/RP0/CPU0:Router(config-subif)# ip addr 10.2.3.4/24
RP/0/RP0/CPU0:Router(config-subif)# no shutdown
RP/0/RP0/CPU0:Router(config-subif)# exit
RP/0/RP0/CPU0:Router(config)# interface Bundle-Ether 1.2
RP/0/RP0/CPU0:Router(config-subif)# dot1q vlan 20
RP/0/RP0/CPU0:Router(config-subif)# ip addr 20.2.3.4/24
RP/0/RP0/CPU0:Router(config-subif)# no shutdown
RP/0/RP0/CPU0:Router(config-subif)# exit
RP/0/RP0/CPU0:Router(config)#interface gig 0/1/5/7
RP/0/RP0/CPU0:Router(config-if)# bundle-id 1 mode act
RP/0/RP0/CPU0:Router(config-if)# commit
RP/0/RP0/CPU0:Router(config-if)# exit
RP/0/RP0/CPU0:Router(config)# exit
RP/0/RP0/CPU0:Router # show vlan trunks
```

The following example shows how to join two ports to form an POS link bundle:

```
Router# config
Router(config)# interface Bundle-POS 5
Router(config-if)# ipv4 address 1.2.3.4/24
Router(config-if)# bundle minimum-active bandwidth 620000
Router(config-if)# bundle minimum-active bandwidth 620000
Router(config-if)# exit
Router(config)# interface POS 0/0/1/0
Router(config-if)# bundle id 5
Router(config-if)# no shutdown
Router(config-if)# exit
Router(config)# interface POS 0/0/1/1
Router(config-if)# bundle id 5
Router(config-if)# no shutdown
Router(config-if)# exit
```

Additional References

The following 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,* Release 3.2
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*
Information about configuring interfaces and other components on the Cisco CRS-1 from a remote Craft Works Interface (CWI) client management application	*Cisco CRS-1 Series Carrier Routing System Craft Works Interface 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: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

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/techsupport

Configuring Channelized SONET on Cisco IOS XR Software

This module describes the configuration of Channelized SONET.

Feature History for Configuring Channelized SONET on Cisco IOS XR Software

Release	Modification
Release 3.5.0	This feature was introduced on the Cisco XR 12000 Series Router.

Contents

Prerequisites for Configuring Channelized SONET

Before configuring Channelized SONET, be sure 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 for SONET commands. Task IDs for commands are listed in *Cisco IOS XR Interface and Hardware Component Command Reference*.
- You have at least one of the following shared port adapters (SPAs) installed in your chassis:
  - Cisco 1-Port Channelized STM-1/OC-3 shared port adapter
  - Cisco 1-Port Channelized DS0/OC-12 shared port adapter
- You should know how to apply and 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 Channelized SONET

To configure Channelized SONET, you must understand the following concepts:

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 shared port adapters (SPAs):

- Cisco 1-Port Channelized STM-1/OC-3 shared port adapter
- Cisco 1-Port Channelized DS0/OC-12 shared port adapter

Channelized SDH is supported only on the following shared port adapter (SPA):

- Cisco 1-Port Channelized STM-1/OC-3 shared port adapter

SONET uses Synchronous Transport Signal (STS) framing. An STS is the electrical equivalent to an optical carrier 1 (OC1).

SDH uses Synchronous Transport Mode (STM) framing. An STM1 is the electrical equivalent to 3 optical carrier 1s (OC1s).

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 only)

The T3/E3 channels can be channelized further into T1s, and the T1s can be channelized into time slots (DS0s).

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. The mode can be set to T3, VT1.5-mapped T1, or POS.

Note POS is supported only on the STS-3c and STS-12c paths on the Cisco 1-Port Channelized DS0/OC-12 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 STM-1/OC-3 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 1 shows the SONET controller default configuration that is in place when the card is installed.

Figure 2 shows SONET controller configuration combinations.

Figure 3 shows the T3 paths that can be configured.

Figure 4 shows the VTG paths that can be configured.

Figure 1 SONET Controller Default Configuration

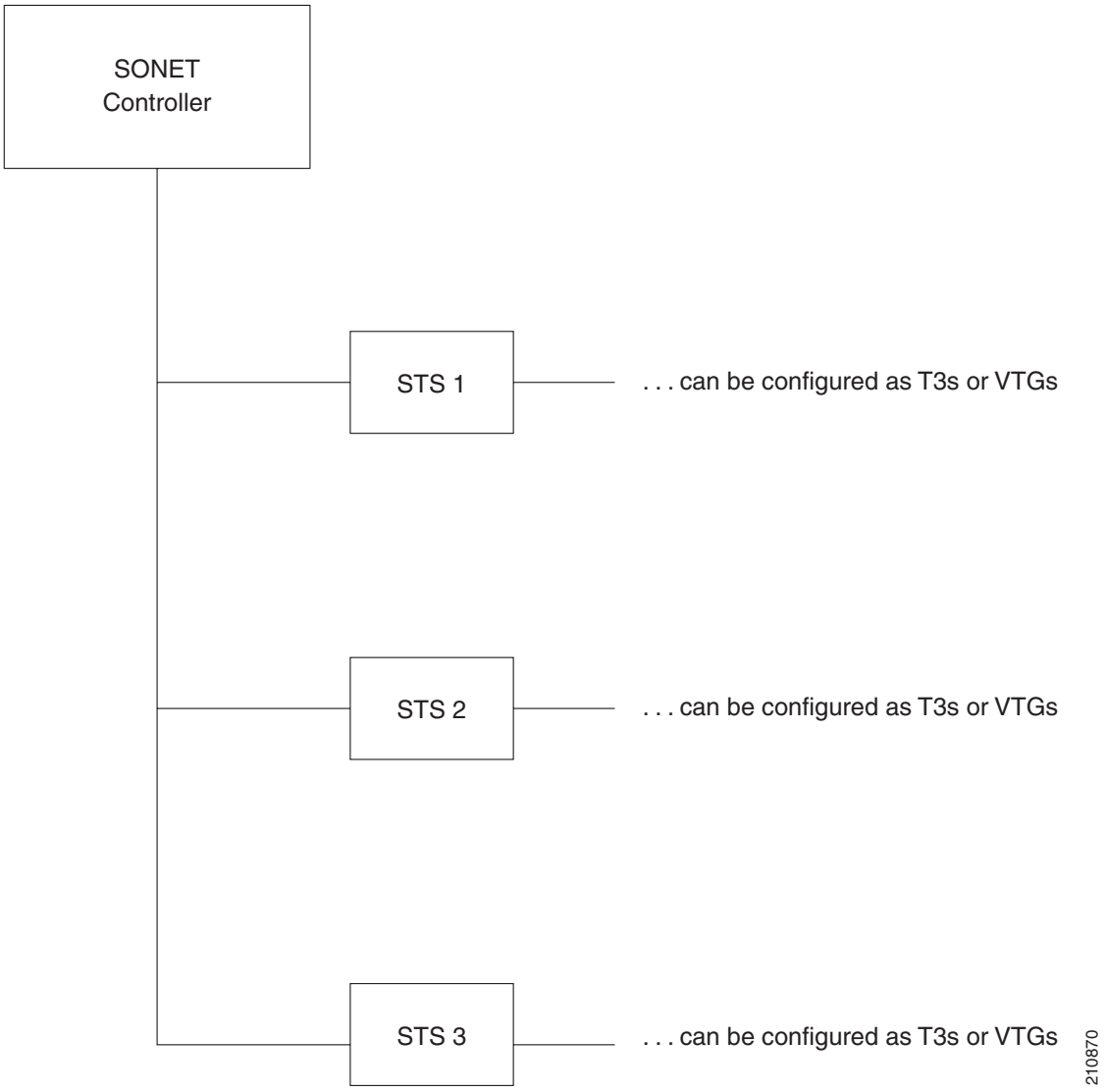


Figure 2 SONET Controller Configuration Combinations

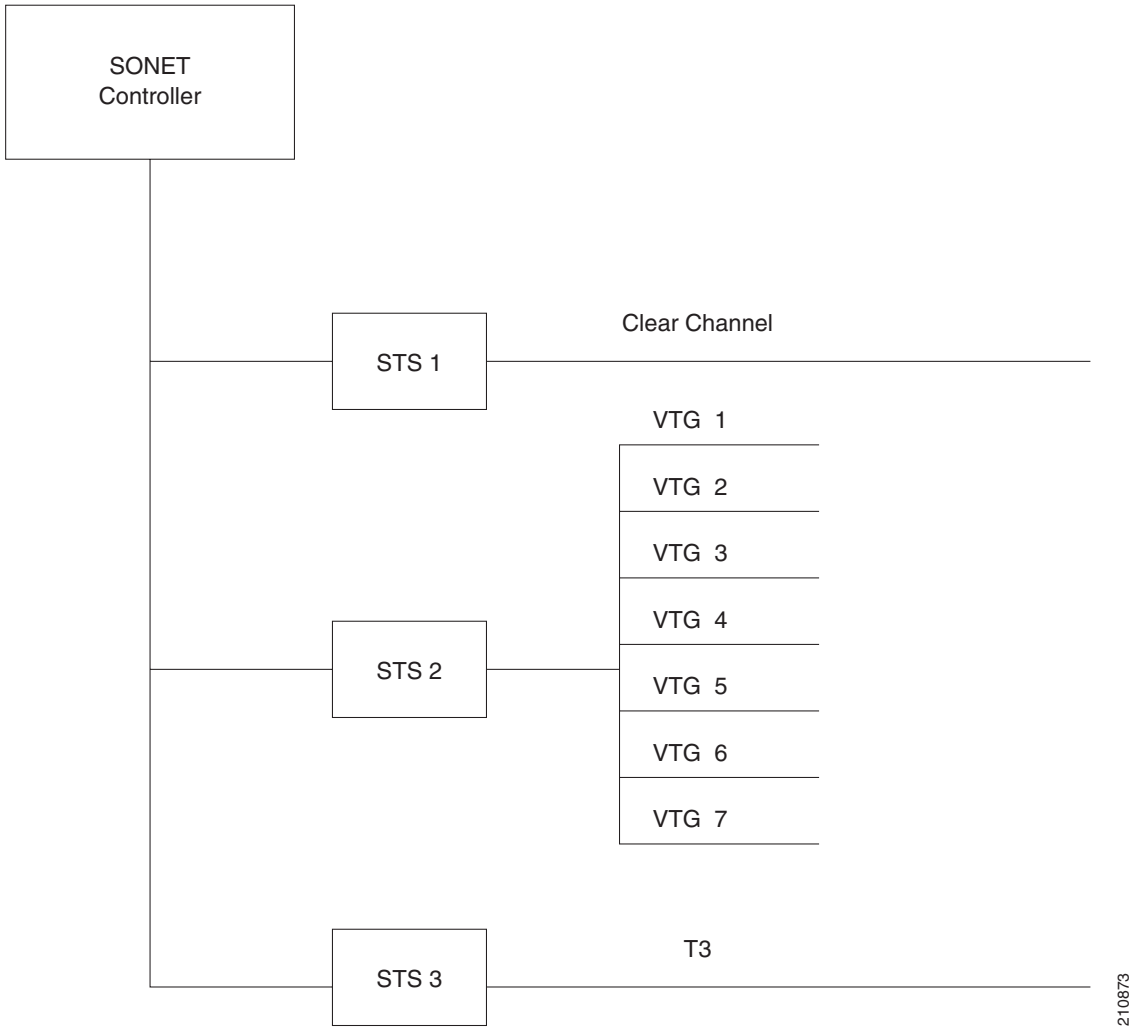


Figure 3 SONET T3 Channelized Paths

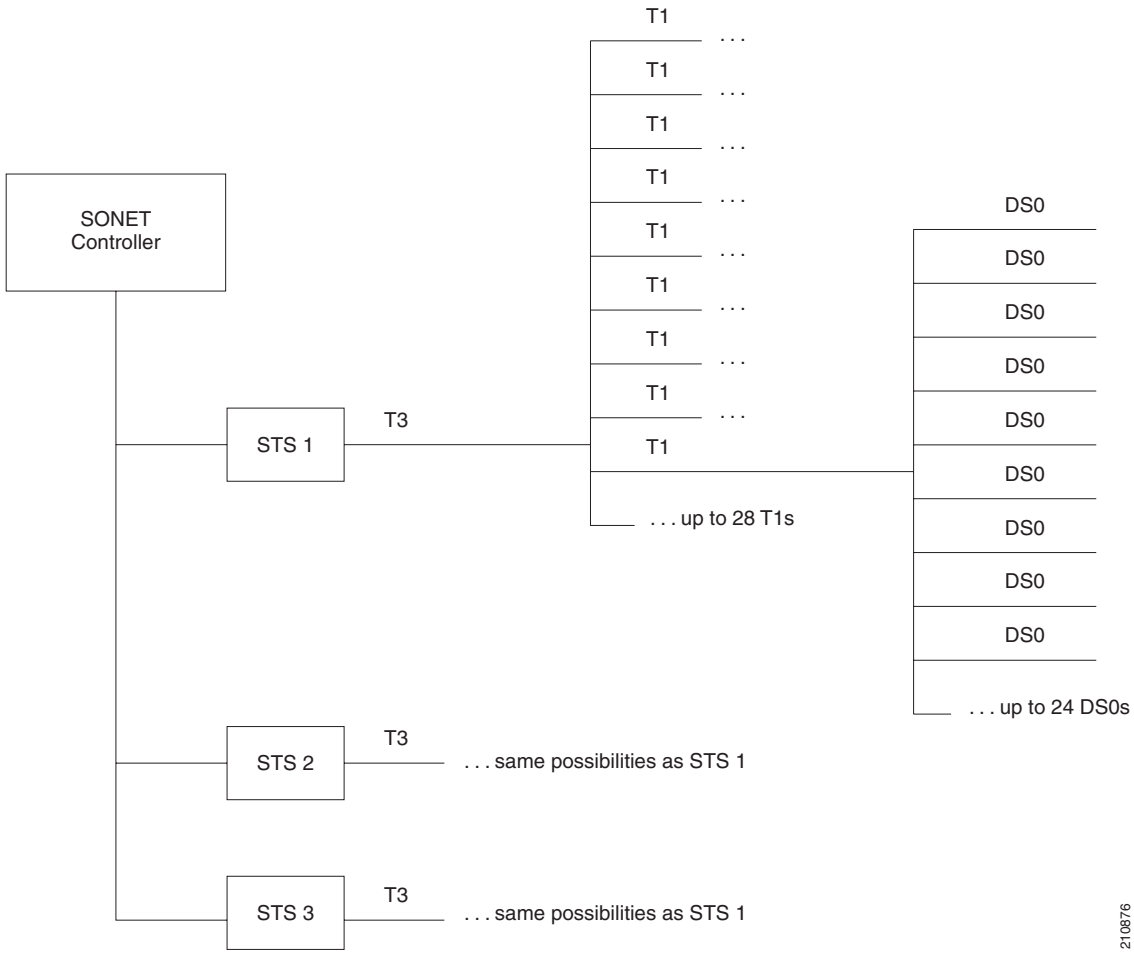
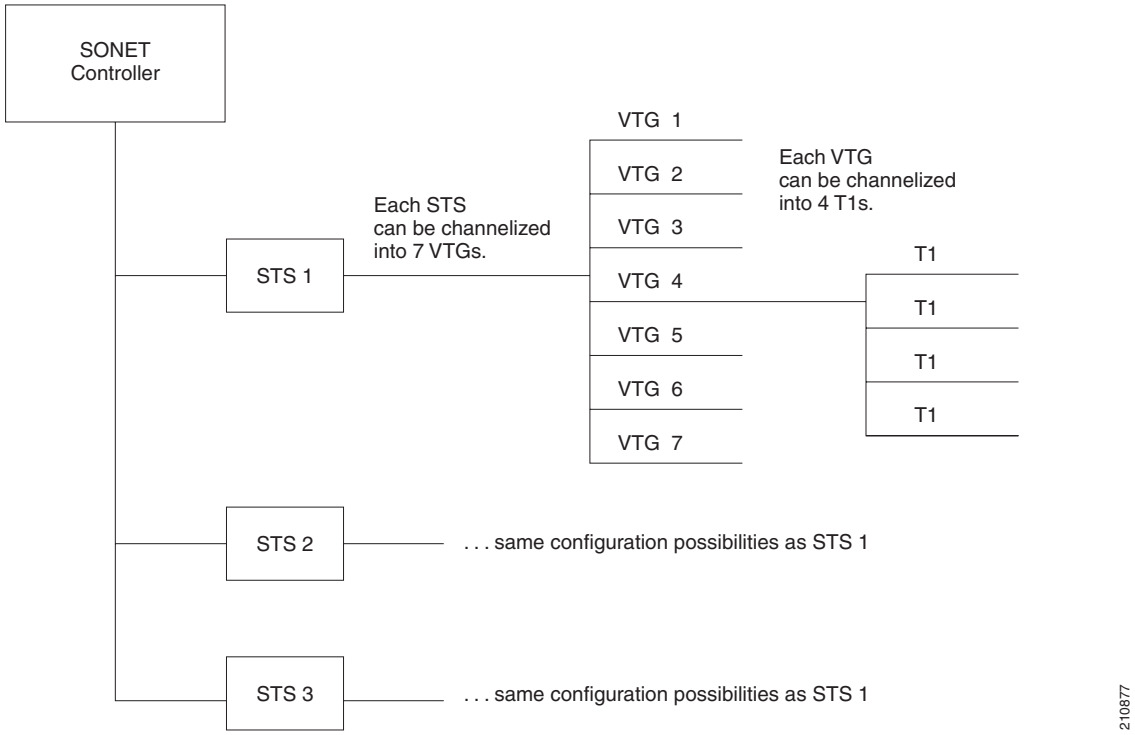


Figure 4 SONET VTG Channelized Paths



Channelized SDH Overview

Synchronous Digital Hierarchy (SDH) is the international equivalent of SONET.

Channelized SDH is supported only on the following shared port adapter (SPA):

- Cisco 1-Port Channelized STM-1/OC-3 shared port adapter

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 he 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 block are STM1 (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 1 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 5 shows the SDH AU3 paths that can be configured.

Figure 6 shows the SDH A4 paths that can be configured.

Figure 5 SDH AU3 Paths

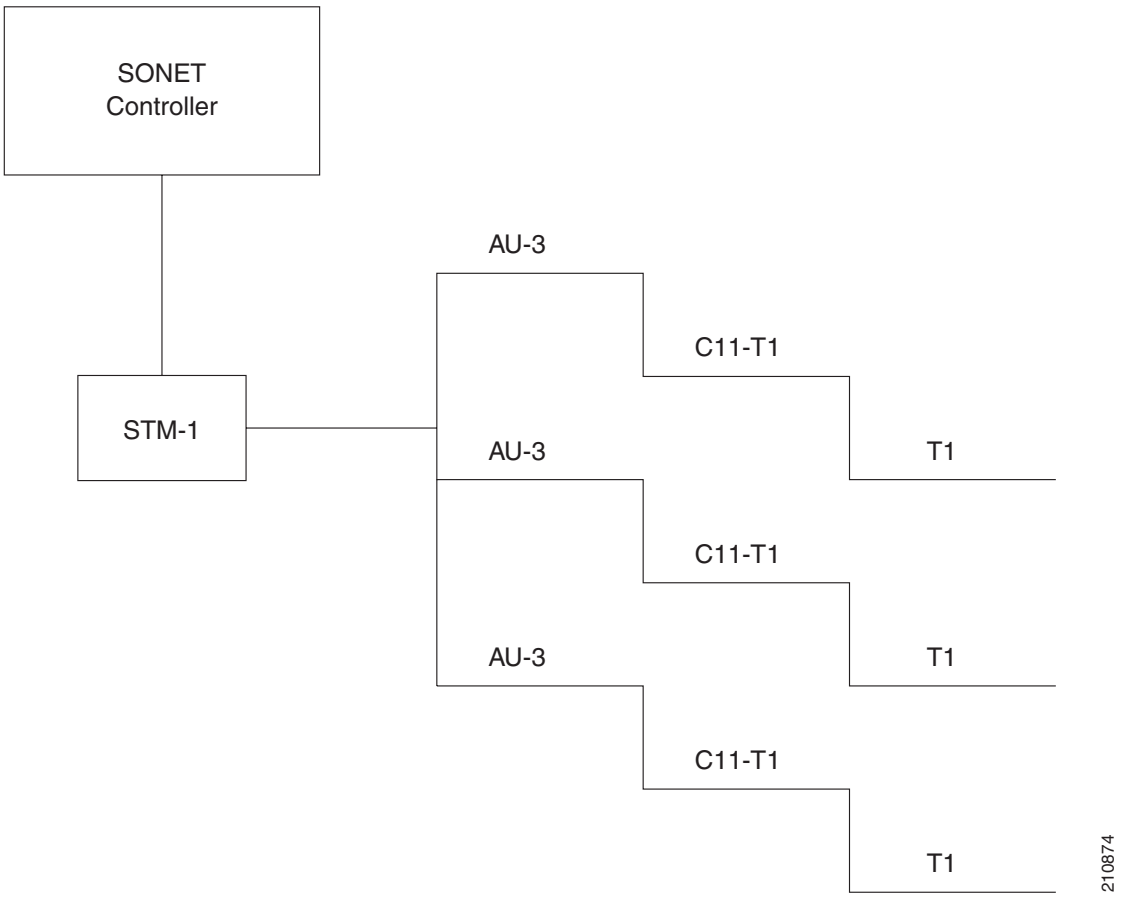
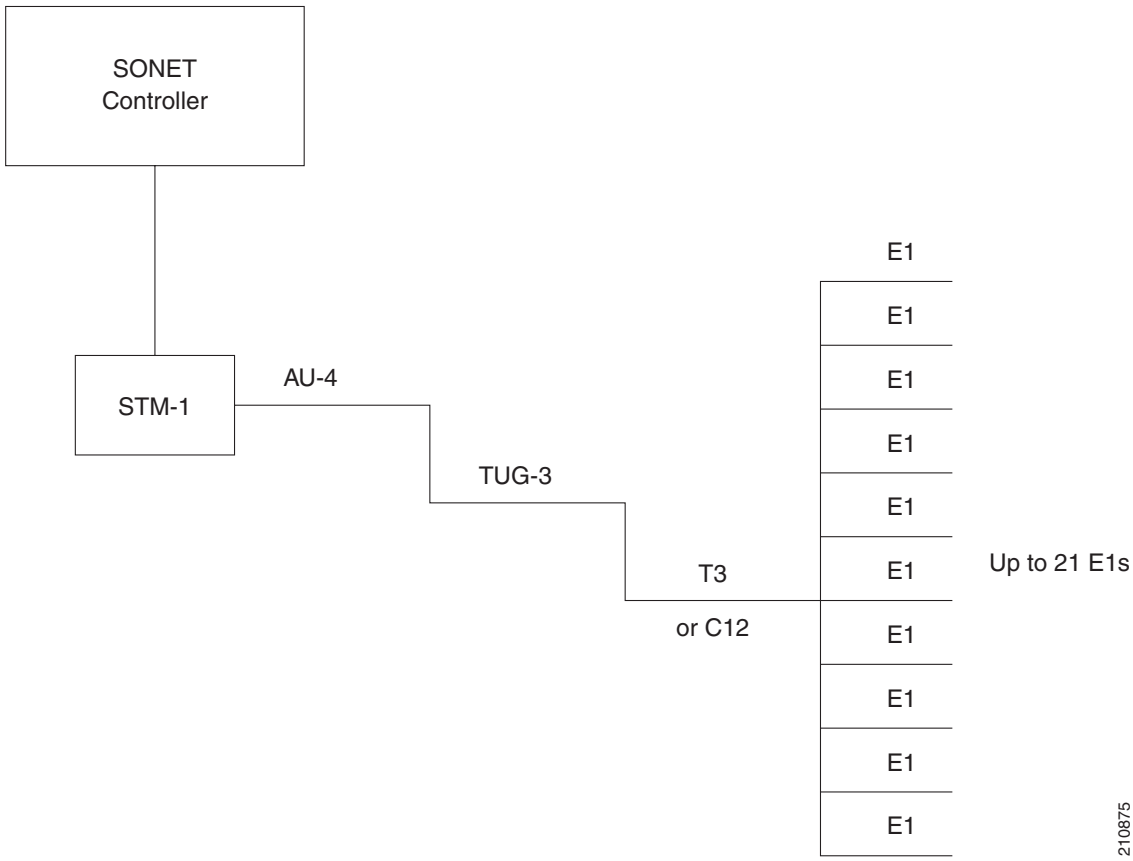


Figure 6 SDH AU4 Paths



Default Configuration Values for Channelized SONET/SDH

Table 2 describes the default configuration parameters that are present on the Channelized SONET/SDH.

Table 2 SONET/SDH Controller Default Configuration 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

This section contains the following procedures:

- Configuring SONET T3 and VT1.5-Mapped T1 Channels, page 106
- Configuring Packet over Sonet Channels, page 110
- Configuring Clear Channel T3, page 113
- Configuring Channelized SONET APS, page 117

- Configuring SDH AU-3, page 120
- Configuring SDH AU-4, page 123

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.

Restrictions

The following restrictions apply for Cisco IOS XR Release 3.5.0:

- STS paths can be channelized into T3s on:
  - Cisco 1-Port Channelized STM-1/OC-3 shared port adapter
  - Cisco 1-Port Channelized DS0/OC-12 shared port adapter
- STS paths can be channelized into VTG mapped T1s on:
  - Cisco 1-Port Channelized STM-1/OC-3 shared port adapter
  - Cisco 1-Port Channelized DS0/OC-12 shared port adapter
- T3 paths can be channelized into T1s or E1s on:
  - Cisco 1-Port Channelized STM-1/OC-3 shared port adapter
  - Cisco 1-Port Channelized DS0/OC-12 shared port adapter (No E1 support in this release)

SUMMARY STEPS

1. **configure**
2. **controller sonet** *instance*
3. **clock source** {**internal** | **line**}
4. **framing** {**sdh** | **sonet**}
5. **sts** *number*
6. **mode** *mode*
7. **width** *number*
8. **root**
9. **controller** *controllerName instance*
10. **mode** *mode*
11. **root**
12. **controller t1** *instance*
13. **channel-group** *number*

14. **timeslots** *num1:num2:num3:num4*
    or
    timeslots *range1-range2*
15. **show configuration**
16. **root**
17. **interface serial** *interfaceNumber*
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** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**controller sonet** *instance* **Example:** `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**} **Example:** `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. • 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. • **line** is the default keyword. **Note** Internal clocking is required for SRP interfaces.
Step 4	**framing** {**sdh** | **sonet**} **Example:** `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. SONET framing (**sonet**) is the default.
Step 5	**sts** *number* **Example:** `RP/0/0/CPU0:router(config-sonet)# sts 1`	Configures the STS stream specified by *number*. The ranges are: • 1 to 3 on the 1-Port OC-3/STM-1 SPA • 1 to 12 on the 1 Port Channelized OC12->DS0 SPA

	Command or Action	Purpose
Step 6	**mode** *mode* **Example:** `RP/0/0/CPU0:router(config-stsPath)# mode t3`	Sets the mode of interface at the STS level. The possible modes are: • t3—SONET path carrying T3 • vt15-t1—SONET path carrying virtual tributary 1.5 T1s (VT15 T1) • pos—Packet over SONET (OC12 only)
Step 7	**width** *number* **Example:** `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: • 1—Indicating one STS stream • 3—Indicating three STS streams (STS-3c) • 12—Indicating concatenation of 12 STS streams (STS-12c) Widths 3, 12, and 48 are configured on STS paths at natural boundaries, which coincide with the following path numbers: • 1, 4, 7, 10, and so on, for STS-3c • 1, 13, 25, and 37 for STS-12c • 1 for STS-48c
Step 8	**root** **Example:** `RP/0/0/CPU0:router(config-stsPath)# root`	Exits to global configuration mode.
Step 9	**controller** *controllerName instance* **Example:** `RP/0/0/CPU0:router(config)# controller t3 0/1/0/0/0`	Enters controller configuration submode and specifies the controller name and instance identifier with the *rack/slot/module/port/controllerName* notation. The controller names are: • t3—SONET path carrying T3 • vt15-t1—SONET path carrying virtual tributary 1.5 T1s (VT15 T1)
Step 10	**mode** *mode* **Example:** `RP/0/0/CPU0:router(config-t3)# mode t1`	Sets the mode of interface at this level. The possible modes are: • t1—Channelized into 28 T1s • e1—Channelized into 21 E1s • serial—Clear Channel carrying an HDLC-like payload
Step 11	**root** **Example:** `RP/0/0/CPU0:router(config-t3)# root`	Exits to global configuration mode.
Step 12	**controller t1** *instance* **Example:** `RP/0/0/CPU0:router(config)# controller t1 0/1/0/0/0/0`	Enters T1 controller configuration submode and specifies the T1 controller name and instance identifier with the *rack/slot/module/port/T3Num/T1num* notation.

	Command or Action	Purpose
Step 13	**channel-group** *number* **Example:** `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 24.
Step 14	**timeslots** *num1:num2:num3:num4* or **timeslots** *range1-range2* **Example:** `RP/0/0/CPU0:router(config-t1-channel_group)# timeslots 1:3:7:9` `RP/0/0/CPU0:router(config-t1-channel_group)# timeslots 1-12`	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** **Example:** `RP/0/0/CPU0:router(config-t1-channel_group)# show configuration`	Displays the contents of uncommitted configuration.
Step 16	**root** **Example:** `RP/0/0/CPU0:router(config-t3)# root`	Exits to global configuration mode.
Step 17	**interface serial** *interfaceNumber* **Example:** `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 18	**encapsulation frame-relay** | **hdlc** | **ppp** **Example:** `Router(config-if)# encapsulation frame-relay | hdlc | ppp`	Specifies the encapsulation type with the one of the following keywords: • **frame-relay**—Frame Relay network protocol • **hdlc**—High-level Data Link Control (HDLC) synchronous protocol • **ppp**—Point-to-Point Protocol
Step 19	**ipv4** *ip-address mask* **Example:** `Router(config-if)# ip address 10.10.10.10 255.255.255.255`	Assigns an IP address and subnet mask to the interface.
Step 20	**no shutdown** **Example:** `RP/0/0/CPU0:router (config-if)# no shutdown`	Removes the shutdown configuration. **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 21	**end** or **commit** **Example:** `RP/0/0/CPU0:router(config-sonet)# end` or `RP/0/0/CPU0:router(config-sonet)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 22	**show controllers sonet** *instance* **Example:** `RP/0/0/CPU0:router# show controllers sonet 0/1/0/0`	Verifies the SONET controller configuration.

Configuring Packet over Sonet Channels

This task explains how to configure Packet over SONET (POS) 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.

Restrictions

POS is only supported on the following SPAs:

- Cisco 1-Port Channelized DS0/OC-12 shared port adapter

SUMMARY STEPS

1. **configure**
2. **controller sonet** *instance*
3. **clock source** {**internal** | **line**}

4. **framing** {**sdh** | **sonet**}
5. **sts** *number*
6. **width** *number*
7. **mode** *mode*
8. **root**
9. **interface pos** *instance*
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** *instance*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**controller sonet** *instance* **Example:** `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**} **Example:** `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. • 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. • **line** is the default keyword. **Note** Internal clocking is required for SRP interfaces.
Step 4	**framing** {**sdh** | **sonet**} **Example:** `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. SONET framing (**sonet**) is the default.

	Command or Action	Purpose
Step 5	**sts** *number* **Example:** `RP/0/0/CPU0:router(config-sonet)# sts 1`	Configures the STS stream specified by *number*. The ranges are: • 1 to 3 on the 1-Port OC-3/STM-1 SPA • 1 to 12 on the 1 Port Channelized OC12->DS0 SPA
Step 6	**width** *number* **Example:** `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: • 3—Indicating three STS streams (STS-3c) • 12—Indicating concatenation of 12 STS streams (STS-12c) Widths 3, 12, and 48 are configured on STS paths at natural boundaries, which coincide with the following path numbers: • 1, 4, 7, 10, and so on, for STS-3c • 1, 13, 25, and 37 for STS-12c • 1 for STS-48c **Note** POS interfaces are not supported when width is 1.
Step 7	**mode** *mode* **Example:** `RP/0/0/CPU0:router(config-stsPath)# mode pos`	Sets the mode of interface at the STS level. Set the mode to pos to create POS interface (OC12 only).
Step 8	**root** **Example:** `RP/0/0/CPU0:router(config-stsPath)# root`	Exits to global configuration mode.
Step 9	**interface pos** *instance* **Example:** `RP/0/0/CPU0:router(config)# interface POS 0/1/0/0`	Specifies the POS interface name and notation *rack/slot/module/port,* and enters interface configuration mode.
Step 10	**encapsulation** [**hdlc** | **ppp** | **frame-relay** [**IETF**]] **Example:** `RP/0/0/CPU0:router(config-if)# encapsulation hdlc`	(Optional) Configures the interface encapsulation parameters and details such as HDLC or PPP. **Note** The default encapsulation is **hdlc**. **Note** The **frame-relay** option is available on the Cisco XR 12000 Series Router only.
Step 11	**pos crc** {**16** | **32**} **Example:** `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. The default CRC is **32**.
Step 12	**mtu** *value* **Example:** `RP/0/0/CPU0:router(config-if)# mtu 4474`	(Optional) Configures the POS MTU value. The range is 64–65535.

	Command or Action	Purpose
Step 13	**no shutdown** **Example:** `RP/0/0/CPU0:router (config-if)# no shutdown`	Removes the shutdown configuration. **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** or **commit** **Example:** `RP/0/0/CPU0:router(config-sonet)# end` or `RP/0/0/CPU0:router(config-sonet)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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** *instance* **Example:** `RP/0/0/CPU0:router# show interfaces pos 0/1/0/0`	(Optional) Displays the interface configuration.

Configuring Clear Channel T3

This task explains how to configure a SONET line into one 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 STM-1/OC-3 shared port adapter, Clear Channel is supported only on STS 1 and STS 2.

SUMMARY STEPS

1. **configure**
2. **controller sonet** *instance*
3. **clock source** {**internal** | **line**}
4. **framing** {**sdh** | **sonet**}
5. **sts** *number*
6. **mode** *mode*
7. **root**
8. **controller t3** *instance*
9. **mode** *mode*
10. **root**
11. **interface serial** *interfaceNumber*
12. **encapsulation frame-relay** | **hdlc** | **ppp**
13. **ipv4** *ip-address mask*
14. **no shutdown**
15. **end**
    or
    commit
16. **show controllers sonet** *instance*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**controller sonet** *instance* **Example:** `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**} **Example:** `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. • 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. • **line** is the default keyword. **Note** Internal clocking is required for SRP interfaces.

	Command or Action	Purpose
Step 4	**framing** {**sdh** | **sonet**} **Example:** `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. SONET framing (**sonet**) is the default.
Step 5	**sts** *number* **Example:** `RP/0/0/CPU0:router(config-sonet)# sts 1`	Configures the STS stream specified by *number*. The ranges are: • 1 to 3 on the 1-Port OC-3/STM-1 SPA • 1 to 12 on the 1 Port Channelized OC12->DS0 SPA
Step 6	**mode** *mode* **Example:** `RP/0/0/CPU0:router(config-stsPath)# mode t3`	Sets the mode of interface at the STS level. The possible modes are: • t3—SONET path carrying T3
Step 7	**root** **Example:** `RP/0/0/CPU0:router(config-stsPath)# root`	Exits to global configuration mode.
Step 8	**controller t3** *instance* **Example:** `RP/0/0/CPU0:router(config)# controller t3 0/1/0/0/0`	Enters T3 controller configuration submode and specifies the T3 controller name and instance identifier with the *rack/slot/module/port/T3Num* notation.
Step 9	**mode** *mode* **Example:** `RP/0/0/CPU0:router(config-t3)# mode serial`	Sets the mode of interface. To establish Clear Channel, set the mode to serial.
Step 10	**root** **Example:** `RP/0/0/CPU0:router(config-t3)# root`	Exits to global configuration mode.
Step 11	**interface serial** *interfaceNumber* **Example:** `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** | **hdlc** | **ppp** **Example:** `Router(config-if)# encapsulation frame-relay | hdlc | ppp`	Specifies the encapsulation type with the one of the following keywords: • **frame-relay**—Frame Relay network protocol • **hdlc**—High-level Data Link Control (HDLC) synchronous protocol • **ppp**—Point-to-Point Protocol

	Command or Action	Purpose
Step 13	**ipv4** *ip-address mask* **Example:** `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** **Example:** `RP/0/0/CPU0:router (config-if)# no shutdown`	Removes the shutdown configuration. **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 15	**end** or **commit** **Example:** `RP/0/0/CPU0:router(config-sonet)# end` or `RP/0/0/CPU0:router(config-sonet)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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** *instance* **Example:** `RP/0/0/CPU0:router# show controllers sonet 0/1/0/0`	Verifies the SONET controller configuration.

	Command or Action	Purpose
Step 17	**end** or **commit** **Example:** `RP/0/0/CPU0:router(config-sonet)# end` or `RP/0/0/CPU0:router(config-sonet)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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** *instance* **Example:** `RP/0/0/CPU0:router# show controllers sonet 0/1/0/0`	Verifies the SONET controller configuration.

Configuring Channelized SONET APS

This task explains how to configure APS for channelized SONET lines. Both local (one router) and remote (two routers) options are given.

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.

Restrictions

TBD

SUMMARY STEPS

1. **aps group** *number*

2. **channel** {**0** | **1**} **local sonet** *interface*
   or
   channel {**0** | **1**} **remote** *ip-address*
3. **channel** {**0** | **1**} **local sonet** *interface*
   or
   channel {**0** | **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* **Example:** `RP/0/0/CPU0:router(config)# aps group 1`	Adds an APS group with a specified number and enters APS group configuration mode. • Use the **aps group** command in global configuration mode. • 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. **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. **Note** The **aps group** command is used even when a single protect group is configured.
Step 2	**channel** {**0** | **1**} **local sonet** *interface* or **channel** {**0** | **1**} **remote** *ip-address* **Example:** `RP/0/0/CPU0:router(config-aps)# channel 0 local SONET 0/0/0/1` or `RP/0/0/CPU0:router(config-aps)# channel 0 remote 172.18.69.123`	Creates a channel for the APS group: • **0** designates a protect channel. • **1** designates an active channel. **Note** The protect channel must be assigned before the active channel can be assigned. **Note** To configure APS where both channels are on one router, use the **channel local** command for both the protect and active channels. 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** {**0** | **1**} **local sonet** *interface* or **channel** {**0** | **1**} **remote** *ip-address* **Example:** `RP/0/0/CPU0:router(config-aps)# channel 1 local SONET 0/0/0/2` or `RP/0/0/CPU0:router(config-aps)# channel 1 remote 172.18.69.123`	Creates a channel for the APS group: • **0** designates a protect channel. • **1** designates an active channel. **Note** The active channel must be assigned after the protect channel is assigned. **Note** To configure APS where both channels are on one router, use the **channel local** command for both the protect and active channels. 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**} **Example:** `RP/0/0/CPU0:router(config-aps)#` **signalling sonet**	Configures the K1K2 overhead byte signaling protocol used for automatic protection switching (APS). The keyword options are: • **sonet**—Sets signaling to SONET. • **sdh**—Sets signaling to Synchronous Digital Hierarchy (SDH).
Step 5	**end** or **commit** **Example:** `RP/0/0/CPU0:router(config-sonet)# end` or `RP/0/0/CPU0:router(config-sonet)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 6	**show aps** **Example:** `RP/0/0/CPU0:router# show aps`	(Optional) Displays the operational status for all configured SONET APS groups.
Step 7	**show aps group** [*number*] **Example:** `RP/0/0/CPU0:router# show aps group 3`	(Optional) Displays the operational status for configured SONET APS groups. **Note** The **show aps group** command is more useful than the **show aps** command when multiple groups are defined.

Configuring SDH AU-3

This task explains how to configure SDH AU-3 into 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.

Restrictions

Channelized SDH is supported only on the Cisco 1-Port Channelized STM-1/OC-3 shared port adapter (SPA).

In this release, AU-3 paths can be mapped only to C11-T1.

SUMMARY STEPS

1. **configure**
2. **controller sonet** *instance*
3. **clock source** {**internal** | **line**}
4. **framing** {**sdh** | **sonet**}
5. **au** *number*
6. **mode** *mode*
7. **root**
8. **controller t1** *instance*
9. **channel-group** *number*
10. **timeslots** *num1:num2:num3:num4*
    or
    timeslots *range1-range2*
11. **show configuration**
12. **root**
13. **interface serial** *interfaceNumber*
14. **encapsulation frame-relay** | **hdlc** | **ppp**
15. **ipv4** *ip-address mask*
16. **no shutdown**
17. **end**
    or
    commit
18. **show controllers sonet** *instance*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**controller sonet** *instance* **Example:** `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**} **Example:** `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. • 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. • **line** is the default keyword. **Note** Internal clocking is required for SRP interfaces.
Step 4	**framing** {**sdh** | **sonet**} **Example:** `RP/0/0/CPU0:router(config-sonet)# framing sdh`	(Optional) Configures the controller framing with either the **sdh** keyword for Synchronous Digital Hierarchy (SDH) framing or the **sonet** keyword for SONET framing. SONET framing (**sonet**) is the default.
Step 5	**au** *number* **Example:** `RP/0/0/CPU0:router(config-sonet)# au 1`	Specifies the administrative unit (AU) *number* and enters the config-auPath mode. The range is 1 to 3.
Step 6	**mode** *mode* **Example:** `RP/0/0/CPU0:router(config-auPath)# mode c11-t1`	Sets the mode of interface at the AU level. Currently only C11-T1 is supported.
Step 7	**root** **Example:** `RP/0/0/CPU0:router(config-auPath)# root`	Exits to global configuration mode.
Step 8	**controller t1** *instance* **Example:** `RP/0/0/CPU0:router(config)# controller T1 0/1/0/0/0/0/0`	Enters T1 controller configuration submode and specifies the T1 controller name and instance identifier with the *rack/slot/module/port/auNum/t1Num* notation.

	Command or Action	Purpose
Step 9	**channel-group** *number* **Example:** `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 24.
Step 10	**timeslots** *num1:num2:num3:num4* or **timeslots** *range1-range2* **Example:** `RP/0/0/CPU0:router(config-t1-channel_group)# timeslots 1:3:7:9` `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** **Example:** `RP/0/0/CPU0:router(config-t1-channel_group)# show configuration`	Displays the contents of uncommitted configuration.
Step 12	**root** **Example:** `RP/0/0/CPU0:router(config-t3)# root`	Exits to global configuration mode.
Step 13	**interface serial** *interfaceNumber* **Example:** `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 14	**encapsulation frame-relay** | **hdlc** | **ppp** **Example:** `Router(config-if)# encapsulation frame-relay | hdlc | ppp`	Specifies the encapsulation type with the one of the following keywords: • **frame-relay**—Frame Relay network protocol • **hdlc**—High-level Data Link Control (HDLC) synchronous protocol • **ppp**—Point-to-Point Protocol
Step 15	**ipv4** *ip-address mask* **Example:** `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** **Example:** `RP/0/0/CPU0:router (config-if)# no shutdown`	Removes the shutdown configuration. **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 17	**end** or **commit** **Example:** `RP/0/0/CPU0:router(config-sonet)# end` or `RP/0/0/CPU0:router(config-sonet)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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** *instance* **Example:** `RP/0/0/CPU0:router# show controllers sonet 0/1/0/0`	Verifies the SONET controller configuration.

Configuring SDH AU-4

This task explains how to configure an SDH AU-4 stream into a TUG3 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 only on the Cisco 1-Port Channelized STM-1/OC-3 shared port adapter (SPA).

In this release, AU-4 paths can only be channelized into TUG3s.

SUMMARY STEPS

1. **configure**
2. **controller sonet** *instance*
3. **clock source** {**internal** | **line**}

4. **framing** {**sdh** | **sonet**}
5. **au** *number*
6. **mode** *mode*
7. **width** *number*
8. **tug3** *number*
9. **mode** *mode*
10. **root**
11. **controller** *name instance*
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** *interfaceNumber*
20. **encapsulation frame-relay** | **hdlc** | **ppp**
21. **ipv4** *ip-address mask*
22. **no shutdown**
23. **end**
    or
    commit
24. **show controllers sonet** *instance*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**controller sonet** *instance* **Example:** `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.

	Command or Action	Purpose
Step 3	**clock source** {**internal** | **line**} **Example:** `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. • 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. • **line** is the default keyword. **Note** Internal clocking is required for SRP interfaces.
Step 4	**framing** {**sdh** | **sonet**} **Example:** `RP/0/0/CPU0:router(config-sonet)# framing sdh`	(Optional) Configures the controller framing with either the **sdh** keyword for Synchronous Digital Hierarchy (SDH) framing or the **sonet** keyword for SONET framing. SONET framing (**sonet**) is the default.
Step 5	**au** *number* **Example:** `RP/0/0/CPU0:router(config-sonet)# au 1`	Specifies the administrative unit (AU) *number* and enters the config-auPath mode. The range is 1 to 3.
Step 6	**mode** *mode* **Example:** `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* **Example:** `RP/0/0/CPU0:router(config-auPath)# width 3`	Configures the number of the AU streams. The range is 1 to 3.
Step 8	**tug3** *number* **Example:** `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* **Example:** `RP/0/0/CPU0:router(config-tug3Path)# mode e3`	Sets the mode of interface at the tug3 level. The modes are: • e3—TUG3 path carrying E3 • t3—TUG3 path carrying T3
Step 10	**root** **Example:** `RP/0/0/CPU0:router(config-tug3Path)# root`	Exits to global configuration mode.

	Command or Action	Purpose
Step 11	**controller** *name instance* **Example:** `RP/0/0/CPU0:router(config)# controller e3 0/1/0/0/0/0`	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: • e3—TUG3 path carrying E3 • t3—TUG3 path carrying T3
Step 12	**mode** *mode* **Example:** `RP/0/0/CPU0:router(config-e3)#mode e1`	Sets the mode of interface. The modes are: • e1—Channelized into 21 E1s • serial—Clear Channel carrying hdlc-like payload • t1—Channelized into 28 T1s
Step 13	**root** **Example:** `RP/0/0/CPU0:router(config-e3)# root`	Exits to global configuration mode.
Step 14	**controller** *name instance* **Example:** `RP/0/0/CPU0:router(config)# controller E1 0/1/0/0/0/0/0/0`	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: • e1—channelized into 21 E1s • serial—Clear Channel carrying hdlc-like payload • t1—Channelized into 28 T1s
Step 15	**channel-group** *number* **Example:** `RP/0/0/CPU0:router(config-e1)# channel-group 0`	Sets the channel-group number to which time slots are assigned. The range is from 1 to 24.
Step 16	**timeslots** *num1:num2:num3:num4* or **timeslots** *range1-range2* **Example:** `RP/0/0/CPU0:router(config-e1-channel_group)# timeslots 1:3:7:9` `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** **Example:** `RP/0/0/CPU0:router(config-e1-channel_group)# show configuration`	Displays the contents of uncommitted configuration.
Step 18	**root** **Example:** `RP/0/0/CPU0:router(config-e1-channel_group)# root`	Exits to global configuration mode.

	Command or Action	Purpose
Step 19	**interface serial** *interfaceNumber* **Example:** `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 20	**encapsulation frame-relay** | **hdlc** | **ppp** **Example:** `Router(config-if)# encapsulation frame-relay | hdlc | ppp`	Specifies the encapsulation type with the one of the following keywords: • **frame-relay**—Frame Relay network protocol • **hdlc**—High-level Data Link Control (HDLC) synchronous protocol • **ppp**—Point-to-Point Protocol
Step 21	**ipv4** *ip-address mask* **Example:** `Router(config-if)# ip address 10.10.10.10 255.255.255.255`	Assigns an IP address and subnet mask to the interface.
Step 22	**no shutdown** **Example:** `RP/0/0/CPU0:router (config-if)# no shutdown`	Removes the shutdown configuration. **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** or **commit** **Example:** `RP/0/0/CPU0:router(config-sonet)# end` or `RP/0/0/CPU0:router(config-sonet)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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** *instance* **Example:** `RP/0/0/CPU0:router# show controllers sonet 0/1/0/0`	Verifies the SONET controller configuration.

Configuration Examples for Channelized SONET

This section contains the following examples:

Channelized SONET T3 to T1 Configuration: Example

The following example shows SONET T3 to T1 configuration.

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# controller sonet 0/1/0/0
RP/0/0/CPU0:router(config-sonet)# clock source internal
RP/0/0/CPU0:router(config-sonet)# framing sonet
RP/0/0/CPU0:router(config-sonet)# sts 1
RP/0/0/CPU0:router(config-stsPath)# mode t3
RP/0/0/CPU0:router(config-stsPath)# width 3
RP/0/0/CPU0:router(config-stsPath)# root
RP/0/0/CPU0:router(config)# controller t3 0/1/0/0/0
RP/0/0/CPU0:router(config-t3)# mode t1
RP/0/0/CPU0:router(config-t3)# framing auto-detect
RP/0/0/CPU0:router(config-t3)# root
RP/0/0/CPU0:router(config)# controller t1 0/1/0/0/0/0
RP/0/0/CPU0:router(config-t1)# framing esf
RP/0/0/CPU0:router(config-t1)# channel-group 0
RP/0/0/CPU0:router(config-t1-channel_group)# timeslots 1:3:7:9
RP/0/0/CPU0:router(config-t1-channel_group)# show configuration
RP/0/0/CPU0:router(config-t3)# root
RP/0/0/CPU0:router(config)# interface serial 0/1/0/0/0/0:0
Router(config-if)# encapsulation frame-relay | hdlc | ppp
Router(config-if)# ip address 10.10.10.10 255.255.255.255
Router(config-if)# no shutdown
RP/0/0/CPU0:router(config-sonet)# commit
RP/0/0/CPU0:router# show controllers sonet 0/1/0/0
```

Channelized Packet over SONET Configuration: Example

The following example shows Channelized Packet over SONET configuration.

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# controller sonet 0/1/0/0
RP/0/0/CPU0:router(config-sonet)# clock source internal
RP/0/0/CPU0:router(config-sonet)# framing sonet
RP/0/0/CPU0:router(config-sonet)# sts 1
RP/0/0/CPU0:router(config-stsPath)# mode pos
RP/0/0/CPU0:router(config-stsPath)# width 3
RP/0/0/CPU0:router(config-stsPath)# root
RP/0/0/CPU0:router(config)# interface POS 0/1/0/0
RP/0/0/CPU0:router(config-if)# encapsulation hdlc
RP/0/0/CPU0:router(config-if)# pos crc 32
RP/0/0/CPU0:router(config-if)# mtu 4474
```

```
RP/0/0/CPU0:router (config-if)# no shutdown
RP/0/0/CPU0:router(config-sonet)# commit
RP/0/0/CPU0:router# show interfaces pos 0/1/0/0
```

Clear Channel T3 Configuration: Example

The following example shows SONET Clear Channel configuration.

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# controller sonet 0/1/0/0
RP/0/0/CPU0:router(config-sonet)# clock source internal
RP/0/0/CPU0:router(config-sonet)# framing sonet
RP/0/0/CPU0:router(config-sonet)# sts 1
RP/0/0/CPU0:router(config-stsPath)# mode t3
RP/0/0/CPU0:router(config-stsPath)# root
RP/0/0/CPU0:router(config)# controller t3 0/1/0/0/0
RP/0/0/CPU0:router(config-t3)# mode serial
RP/0/0/CPU0:router(config-t3)# root
RP/0/0/CPU0:router(config)# interface serial 0/1/0/0/0/0:0
Router(config-if)# encapsulation frame-relay | hdlc | ppp
Router(config-if)# ip address 10.10.10.10 255.255.255.255
Router(config-if)# no shutdown
RP/0/0/CPU0:router(config-sonet)# commit
RP/0/0/CPU0:router# show controllers sonet 0/1/0/0
```

Channelized SONET APS Configuration: Example

The following example shows SONET Local (one router) APS configuration.

```
RP/0/0/CPU0:router(config)# aps group 1
RP/0/0/CPU0:router(config-aps)# channel 0 local SONET 0/0/0/1
RP/0/0/CPU0:router(config-aps)# channel 1 local SONET 0/0/0/2
RP/0/0/CPU0:router(config-aps)# signalling sonet
RP/0/0/CPU0:router(config-sonet)# commit
RP/0/0/CPU0:router# show aps
RP/0/0/CPU0:router# show aps group 3
```

The following example shows SONET Remote (two routers) APS configuration.

```
RP/0/0/CPU0:router(config)# aps group 1
RP/0/0/CPU0:router(config-aps)# channel 0 local SONET 0/0/0/1
RP/0/0/CPU0:router(config-aps)# channel 1 remote 172.18.69.123
RP/0/0/CPU0:router(config-aps)# signalling sonet
RP/0/0/CPU0:router(config-sonet)# commit
RP/0/0/CPU0:router# show aps
RP/0/0/CPU0:router# show aps group 3
```

Channelized SDH AU-3 Configuration: Example

The following example shows SDH AU-3 configuration.

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# controller sonet 0/1/0/0
RP/0/0/CPU0:router(config-sonet)# clock source internal
RP/0/0/CPU0:router(config-sonet)# framing sdh
RP/0/0/CPU0:router(config-sonet)# au 1
RP/0/0/CPU0:router(config-auPath)# mode c11-t1
RP/0/0/CPU0:router(config-auPath)# root
RP/0/0/CPU0:router(config)# controller T1 0/1/0/0/0/0/0
```

```
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)# show configuration
RP/0/0/CPU0:router(config-t3)# root
RP/0/0/CPU0:router(config)# interface serial 0/1/0/0/0/0:0
Router(config-if)# encapsulation frame-relay | hdlc | ppp
Router(config-if)# ip address 10.10.10.10 255.255.255.255
Router(config-if)# no shutdown
RP/0/0/CPU0:router(config-sonet)# commit
RP/0/0/CPU0:router# show controllers sonet 0/1/0/0
```

Channelized SDH AU-4 Configuration: Example

The following example shows SDH AU-4 configuration.

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# controller sonet 0/1/0/0
RP/0/0/CPU0:router(config-sonet)# clock source internal
RP/0/0/CPU0:router(config-sonet)# framing sdh
RP/0/0/CPU0:router(config-sonet)# au 1
RP/0/0/CPU0:router(config-auPath)# mode tug3
RP/0/0/CPU0:router(config-auPath)# width 3
RP/0/0/CPU0:router(config-auPath)# tug3 1
RP/0/0/CPU0:router(config-tug3Path)# mode e3
RP/0/0/CPU0:router(config-tug3Path)# root
RP/0/0/CPU0:router(config)# controller e3 0/1/0/0/0/0
RP/0/0/CPU0:router(config-e3)# mode e1
RP/0/0/CPU0:router(config-e3)# root
RP/0/0/CPU0:router(config)# controller E1 0/1/0/0/0/0/0/0
RP/0/0/CPU0:router(config-e1)# channel-group 0
RP/0/0/CPU0:router(config-e1-channel_group)# timeslots 1-12
RP/0/0/CPU0:router(config-e1-channel_group)# show configuration
RP/0/0/CPU0:router(config-e1-channel_group)# root
RP/0/0/CPU0:router(config)# interface serial 0/1/0/0/0/0:0
Router(config-if)# encapsulation frame-relay | hdlc | ppp
Router(config-if)# ip address 10.10.10.10 255.255.255.255
Router(config-if)# no shutdown
RP/0/0/CPU0:router(config-sonet)# commit
RP/0/0/CPU0:router# show controllers sonet 0/1/0/0
```

Where to Go Next

After configuring SONET channels, you can configure other controllers and interfaces, such as clear channel T3/E3 or channelized T3 controllers and interfaces as described in:

- Chapter , “Configuring Clear Channel T3/E3 Controllers and Channelized T3 Controllers on Cisco IOS XR Software”

Additional References

The following 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*, Release 3.2
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*
Information about configuring router interfaces and other components from a remote Craft Works Interface (CWI) client management application	*Cisco Craft Works Interface 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: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

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/techsupport

Configuring Dense Wavelength Division Multiplexing Controllers on Cisco IOS XR Software

This module describes the configuration of dense wavelength division multiplexing (DWDM) controller on routers supporting Cisco IOS XR software. DWDM is an optical technology that is used to increase bandwidth over existing fiber-optic backbones. DWDM can be configured on supported 10-Gigabit Ethernet (GE) or Packet-over-SONET/SDH physical layer interface modules (PLIMs). After you configure the DWDM controller, you can configure an associated POS or 10-Gigabit Ethernet interface.

To configure a POS interface, refer to the *Configuring POS Interfaces on Cisco IOS XR Software* module, later in this document.

To configure a 10-Gigabit Ethernet interface, refer to the *Configuring Ethernet Interfaces on Cisco IOS XR Software* module, later in this document.

Feature History for Configuring DWDM Controller Interfaces

Release	Modification
Release 3.3.0	This feature was introduced on the Cisco CRS-1. Support was added for the OC-768c/STM-256c DWDM PLIM and 10-GE DWDM PLIM.
Release 3.4.0	Support was added for user configuration of the laser, TTI strings, and BDI insertion, as well as performance monitoring.
Release 3.5.0	

Contents

Prerequisites for Configuring DWDM Controller Interfaces

Before configuring a DWDM controller, be sure 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 for DWDM commands. Task IDs for commands are listed in *Cisco IOS XR Interface and Hardware Component Command Reference*.
- Your PLIM must support DWDM. The OC-768c/STM-256c DWDM PLIM and 10-GE DWDM PLIM support DWDM.

Information About the DWDM Controllers

DWDM support in Cisco IOS XR software is based on the Optical Transport Network (OTN) protocol that is specified in ITU-T G.709. This standard combines the benefits of SONET/SDH technology with the multiwavelength networks of DWDM. It also provides for forward error correction (FEC) that can allow a reduction in network costs by reducing the number of regenerators used.

To enable multiservice transport, OTN uses the concept of a wrapped overhead (OH). To illustrate this structure:

- Optical channel payload unit (OPU) OH information is added to the information payload to form the OPU. The OPU OH includes information to support the adaptation of client signals.
- Optical channel data unit (ODU) OH is added to the OPU to create the ODU. The ODU OH includes information for maintenance and operational functions to support optical channels.
- Optical channel transport unit (OTU) OH together with the FEC is added to form the OTU. The OTU OH includes information for operational functions to support the transport by way of one or more optical channel connections.
- Optical channel (OCh) OH is added to form the OCh. The OCh provides the OTN management functionality and contains four subparts: the OPU, ODU, OTU, and frame alignment signal (FAS). See Figure 7.

Figure 7 OTN Optical Channel Structure

How to Configure DWDM Controllers

The DWDM controllers are configured in the physical layer control element of the Cisco IOS XR software configuration space. This configuration is done using the **controller dwdm** command, and is described in the following tasks.

- Configuring the Optical Parameters, page 135
- Configuring G.709 Parameters, page 137

Note All interface configuration tasks for the POS or GE interfaces still must be performed in interface configuration mode. Refer to *Configuring POS Interfaces on Cisco IOS XR Software* and *Configuring Ethernet Interfaces on Cisco IOS XR Software* modules for more information.

Configuring the Optical Parameters

This task describes how to configure the receive power threshold and the wavelength parameters for the DWDM controller. You should verify that the optical parameters are configured correctly for your DWDM installation and if necessary, perform this task.

Prerequisites

The **rx-los-threshold**, **wavelength** and **transmit-power** commands can be used only when the controller is in the shutdown state. Use the **shutdown** command.

Restrictions

The transmit power level and receive LOS threshold are configurable only on the OC-768c/STM-256c DWDM PLIM.

SUMMARY STEPS

1. **configure**
2. **controller dwdm** *instance*
3. **shutdown**
4. **commit**
5. **rx-los-threshold** *power-level*
6. **wavelength** *channel-number*
7. **transmit-power** *power-level*
8. **no shutdown**
9. **laser** {**on** | **off**}
10. **end**
    or
    commit
11. **show controllers dwdm** *instance* [**optics** | **wavelength-map**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**controller dwdm** *instance* **Example:** `RP/0/RP0/CPU0:router(config)# controller dwdm 0/1/0/0`	Specifies the DWDM controller name in the notation *rack/slot/module/port* and enters DWDM configuration mode.
Step 3	**shutdown** **Example:** `RP/0/RP0/CPU0:router(config-dwdm)# shutdown`	Disables the DWDM controller. You must disable the controller before you can use the DWDM configuration commands.
Step 4	**commit** **Example:** `RP/0/RP0/CPU0:router(config-dwdm)# commit`	Saves configuration changes. This performs the shutdown from the previous step. When the controller has been shut down, you can proceed with the configuration.
Step 5	**rx-los-threshold** *power-level* **Example:** `RP/0/RP0/CPU0:router(config-dwdm)# rx-los-threshold -10`	Configures the transponder receive power threshold. Values are in units of 0.1 dBm and can range from -200 to 0. This corresponds to a range of -20 dBm to 0 dBm.
Step 6	**wavelength** *channel-number* **Example:** `RP/0/RP0/CPU0:router(config-dwdm)# wavelength 1`	Configures the channel number corresponding to the first wavelength. Values can range from 1 to 185, but not all channels are supported on all PLIMs. Use the **show controller dwdm** command with the **wavelength-map** keyword to determine which channels and wavelengths are supported on a specific controller. **Note** There is no cross-checking to determine if the chosen wavelength is being used on another port on the same PLIM or on another PLIM in the system.
Step 7	**transmit-power** *power-level* **Example:** `RP/0/RP0/CPU0:router(config-dwdm)# transmit-power 10`	Configures the transponder transmit power. Values are in units of 0.1 dBm and can range from -190 to +10. This corresponds to a range of -19 dBm to +1 dBm.
Step 8	**no shutdown** **Example:** `RP/0/RP0/CPU0:router(config-dwdm)# no shutdown`	Removes the shutdown configuration. • The removal of the shutdown configuration removes the forced administrative down on the controller, enabling the controller to move to an up or down state.

	Command or Action	Purpose
Step 9	**laser off** **Example:** `RP/0/RP0/CPU0:router(config-dwdm)# laser off`	Turns the laser off. The laser cannot be turned on if the controller is in the shutdown state in a non-G.709 mode or if the controller is in *internal loopback* mode.
Step 10	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router(config-dwdm)# end` or `RP/0/RP0/CPU0:router(config-dwdm)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 11	**show controllers dwdm** *instance* [**optics** | **wavelength-map**] **Example:** `RP/0/RP0/CPU0:router# show controller dwdm 0/1/0/0 optics`	Displays the output power level, input power level, wavelength, and laser bias current monitoring information.

Troubleshooting Tips

You will get an error message if you try to commit configuration changes to the controller when it is in the up state. You must shut down the controller before you can use the DWDM configuration commands.

Configuring G.709 Parameters

This task describes how to customize the alarm display and the thresholds for alerts and forward error correction (FEC). You need to use this task only if the default values are not correct for your installation.

Prerequisites

The **g709 disable**, **loopback**, and **g709 fec** commands can be used only when the controller is in the shutdown state. Use the **shutdown** command.

SUMMARY STEPS

1. **configure**
2. **controller dwdm** *instance*
3. **shutdown**
4. **commit**
5. **g709 disable**
6. **loopback internal**
7. **g709 fec** {**disable** | **enhanced** | **standard**}
8. **g709** {**odu** | **otu**} *alarm* **disable**
9. **g709 otu overhead tti** {**expected** | **sent**} {**ascii** | **hex**} *tti-string*
10. **no shutdown**
11. **end**
    or
    commit
12. **show controllers dwdm** *instance* **g709**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**controller dwdm** *instance* **Example:** `RP/0/RP0/CPU0:router(config)# controller dwdm 0/1/0/0`	Specifies the DWDM controller name in the notation *rack/slot/module/port* and enters DWDM configuration mode.
Step 3	**shutdown** **Example:** `RP/0/RP0/CPU0:router(config-dwdm)# shutdown`	Disables the DWDM controller. You must disable the controller before you can use the DWDM configuration commands.
Step 4	**commit** **Example:** `RP/0/RP0/CPU0:router(config-dwdm)# commit`	Saves configuration changes. This performs the shutdown from the previous step. When the controller has been shut down, you can proceed with the configuration.
Step 5	**g709 disable** **Example:** `RP/0/RP0/CPU0:router(config-dwdm)# g709 disable`	(Optional) Disables the G.709 wrapper. The wrapper is enabled by default.

	Command or Action	Purpose
Step 6	**loopback** {**internal** | **line**} **Example:** `RP/0/RP0/CPU0:router(config-dwdm)# loopback internal`	(Optional) Configures the DWDM controller for loopback mode.
Step 7	**g709 fec** {**disable** | **standard**} **Example:** `RP/0/RP0/CPU0:router(config-dwdm)# g709 disable`	(Optional) Configures the forward error correction mode (FEC) for the DWDM controller. By default, enhanced FEC is enabled. Only the OC-768c/STM-256c DWDM PLIM supports enhanced FEC.
Step 8	**g709** {**odu** | **otu**} *alarm* **disable** **Example:** `RP/0/RP0/CPU0:router(config-dwdm)# g709 odu bdi disable`	(Optional) Disables the logging of selected optical channel data unit (ODU) alarms or optical channel transport unit (OTU) alarms to the console for a DWDM controller. By default, all alarms are logged to the console.
Step 9	**g709 otu overhead tti** {**expected** | **sent**} {**ascii** | **hex**} *tti-string* `Example:` `RP/0/RP0/CPU0:router(config-dwdm)# g709 otu overhead tti expected ascii test OTU 5678`	Configures a transmit or expected Trail Trace Identifier (TTI) that is displayed in the **show controller dwdm** command.
Step 10	**no shutdown** **Example:** `RP/0/RP0/CPU0:router(config-dwdm)# no shutdown`	Removes the shutdown configuration. • The removal of the shutdown configuration removes the forced administrative down on the controller, enabling the controller to move to an up or down state.
Step 11	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router(config-dwdm)# end` or `RP/0/RP0/CPU0:router(config-dwdm)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 12	**show controllers dwdm** *instance* **g709** **Example:** `RP/0/RP0/CPU0:router# show controller dwdm 0/1/0/0 optics`	Displays the G.709 Optical Transport Network (OTN) protocol alarms and counters for Bit Errors, along with the FEC statistics and threshold-based alerts.

Examples

The following example shows how to bring the DWDM controller down before using the configuration commands:

```
RP/0/RP0/CPU0:Router# configure
RP/0/RP0/CPU0:Router(config)# controller dwdm 0/0/0/0
RP/0/RP0/CPU0:Router(config-dwdm)# shutdown
RP/0/RP0/CPU0:Router(config-dwdm)# commit
RP/0/RP0/CPU0:Router(config-dwdm)# rx-los-threshold 0
RP/0/RP0/CPU0:Router(config-dwdm)# wavelength 1
RP/0/RP0/CPU0:Router(config-dwdm)# transmit-power 0
RP/0/RP0/CPU0:Router(config-dwdm)# no shutdown
RP/0/RP0/CPU0:Router(config-dwdm)# end
Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]: y

RP/0/RP0/CPU0:Oct 15 12:35:54.299 : config[65732]: %MGBL-LIBTARCFG-6-COMMIT :
Configuration committed by user 'lab'.   Use 'show configuration commit changes
1000000312' to view the changes.
RP/0/RP0/CPU0:Oct 15 12:35:54.403 : config[65732]: %MGBL-SYS-5-CONFIG_I : Configured from
console by lab
```

The following example shows how to customize the alarm display and the thresholds for alerts and forward error correction (FEC):

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# controller dwdm 0/1/0/0
RP/0/RP0/CPU0:router(config-dwdm)# shutdown
RP/0/RP0/CPU0:router(config-dwdm)# commit
RP/0/RP0/CPU0:router(config-dwdm)# g709 disable
RP/0/RP0/CPU0:router(config-dwdm)# loopback internal
RP/0/RP0/CPU0:router(config-dwdm)# g709 fec standard
RP/0/RP0/CPU0:router(config-dwdm)# g709 odu bdi disable
RP/0/RP0/CPU0:router(config-dwdm)# no shutdown
RP/0/RP0/CPU0:router(config-dwdm)# commit
```

What to Do Next

All interface configuration tasks for the POS or GE interfaces still must be performed in interface configuration mode. Refer to *Configuring POS Interfaces on Cisco IOS XR Software* and *Configuring Ethernet Interfaces on Cisco IOS XR Software* modules for more information.

How to Perform Performance Monitoring on DWDM Controllers

Performance monitoring parameters are used to gather, store, set thresholds for, and report performance data for early detection of problems. Thresholds are used to set error levels for each performance monitoring parameter. During the accumulation cycle, if the current value of a performance monitoring parameter reaches or exceeds its corresponding threshold value, a threshold crossing alert (TCA) can be generated. The TCAs provide early detection of performance degradation.

Performance monitoring statistics are accumulated on a 15-minute basis, synchronized to the start of each quarter-hour. They are also accumulated on a daily basis starting at midnight. Historical counts are maintained for thirty-three 15-minute intervals and two daily intervals.

Performance monitoring is described in the following task:

- Configuring DWDM Controller Performance Monitoring, page 141

Configuring DWDM Controller Performance Monitoring

This task describes how to configure performance monitoring on DWDM controllers and how to display the performance parameters.

SUMMARY STEPS

1. **configure**
2. **controller dwdm** *instance*
3. **pm** {**15-min** | **24-hour**} **fec threshold** {**ec-bits** | **uc-words**} *threshold*
4. **pm** {**15-min** | **24-hour**} **optics threshold** {**lbc** | **opr** | **opt**} {**max** | **min**} *threshold*
5. **pm** {**15-min** | **24-hour**} **otn threshold** *otn-parameter threshold*
6. **pm** {**15-min** | **24-hour**} **fec report** {**ec-bits** | **uc-words**} **enable**
7. **pm** {**15-min** | **24-hour**} **optics report** {**lbc** | **opr** | **opt**} {**max-tca** | **min-tca**} **enable**
8. **pm** {**15-min** | **24-hour**} **otn report** *otn-parameter* **enable**
9. **end**
10. **show controllers dwdm** *instance* **pm history** [**15-min** | **24-hour** | **fec** | **optics** | **otn**]
11. **show controllers dwdm** *instance* **pm interval** {**15-min** | **24-hour**} [**fec** | **optics** | **otn**] *index*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**controller dwdm** *instance* **Example:** `RP/0/RP0/CPU0:router(config)# controller dwdm 0/1/0/0`	Specifies the DWDM controller name in the notation *rack/slot/module/port* and enters DWDM configuration mode.
Step 3	**pm** {**15-min** | **24-hour**} **fec threshold** {**ec-bits** | **uc-words**} *threshold* Example: `RP/0/RP0/CPU0:router(config-dwdm)# pm 15-min fec threshold ec-bits 49000000` `RP/0/RP0/CPU0:router(config-dwdm)# pm 15-min fec threshold uc-words xxxxxx`	Configures a performance monitoring threshold for specific parameters on the FEC layer.
Step 4	**pm** {**15-min** | **24-hour**} **optics threshold** {**lbc** | **opr** | **opt**} {**max** | **min**} *threshold* Example: `RP/0/RP0/CPU0:router(config-dwdm)# pm 15-min optics threshold opt max xxx` `RP/0/RP0/CPU0:router(config-dwdm)# pm 15-min optics threshold lbc min xxx`	Configures a performance monitoring threshold for specific parameters on the optics layer.

Command or Action	Purpose
Step 5 **pm** {**15-min** | **24-hour**} **otn threshold** *otn-parameter threshold* Example: RP/0/RP0/CPU0:router(config-dwdm)# pm 15-min otn threshold bbe-pm-ne xxx RP/0/RP0/CPU0:router(config-dwdm)# pm 15-min otn threshold es-sm-fe xxx	Configures a performance monitoring threshold for specific parameters on the optical transport network (OTN) layer. OTN parameters can be as follows: • **bbe-pm-fe**—Far-end path monitoring background block errors (BBE-PM) • **bbe-pm-ne**—Near-end path monitoring background block errors (BBE-PM) • **bbe-sm-fe**—Far-end section monitoring background block errors (BBE-SM) • **bbe-sm-ne**—Near-end section monitoring background block errors (BBE-SM) • **bber-pm-fe**—Far-end path monitoring background block errors ratio (BBER-PM) • **bber-pm-ne**—Near-end path monitoring background block errors ratio (BBER-PM) • **bber-sm-fe**—Far-end section monitoring background block errors ratio (BBER-SM) • **bber-sm-ne**—Near-end section monitoring background block errors ratio (BBER-SM) • **es-pm-fe**—Far-end path monitoring errored seconds (ES-PM) • **es-pm-ne**—Near-end path monitoring errored seconds (ES-PM) • **es-sm-fe**—Far-end section monitoring errored seconds (ES-SM) • **es-sm-ne**—Near-end section monitoring errored seconds (ES-SM) • **esr-pm-fe**—Far-end path monitoring errored seconds ratio (ESR-PM) • **esr-pm-ne**—Near-end path monitoring errored seconds ratio (ESR-PM) • **esr-sm-fe**—Far-end section monitoring errored seconds ratio (ESR-SM) • **esr-sm-ne**—Near-end section monitoring errored seconds ratio (ESR-SM) • **fc-pm-fe**—Far-end path monitoring failure counts (FC-PM) • **fc-pm-ne**—Near-end path monitoring failure counts (FC-PM) • **fc-sm-fe**—Far-end section monitoring failure counts (FC-SM) • **fc-sm-ne**—Near-end section monitoring failure counts (FC-SM)

Command or Action	Purpose
	• **ses-pm-fe**—Far-end path monitoring severely errored seconds (SES-PM) • **ses-pm-ne**—Near-end path monitoring severely errored seconds (SES-PM) • **ses-sm-fe**—Far-end section monitoring severely errored seconds (SES-SM) • **ses-sm-ne**—Near-end section monitoring severely errored seconds (SES-SM) • **sesr-pm-fe**—Far-end path monitoring severely errored seconds ratio (SESR-PM) • **sesr-pm-ne**—Near-end path monitoring severely errored seconds ratio (SESR-PM) • **sesr-sm-fe**—Far-end section monitoring severely errored seconds ratio (SESR-SM) • **sesr-sm-ne**—Near-end section monitoring severely errored seconds ratio (SESR-SM) • **uas-pm-fe**—Far-end path monitoring unavailable seconds (UAS-PM) • **uas-pm-ne**—Near-end path monitoring unavailable seconds (UAS-PM) • **uas-sm-fe**—Far-end section monitoring unavailable seconds (UAS-SM) • **uas-sm-ne**—Near-end section monitoring unavailable seconds (UAS-SM)
Step 6 **pm** {**15-min** | **24-hour**} **fec report** {**ec-bits** | **uc-words**} **enable** Example: `RP/0/RP0/CPU0:router(config-dwdm)# pm 15-min fec report ec-bits enable` `RP/0/RP0/CPU0:router(config-dwdm)# pm 15-min fec report uc-words enable`	Configures threshold crossing alert (TCA) generation for specific parameters on the FEC layer.
Step 7 **pm** {**15-min** | **24-hour**} **optics report** {**lbc** | **opr** | **opt**} {**max-tca** | **min-tca**} **enable** Example: `RP/0/RP0/CPU0:router(config-dwdm)# pm 15-min optics report opt enable` `RP/0/RP0/CPU0:router(config-dwdm)# pm 15-min optics report lbc enable`	Configures TCA generation for specific parameters on the optics layer.
Step 8 **pm** {**15-min** | **24-hour**} **otn report** *otn-parameter* **enable** Example: `RP/0/RP0/CPU0:router(config-dwdm)# pm 15-min otn report bbe-pm-ne enable` `RP/0/RP0/CPU0:router(config-dwdm)# pm 15-min otn report es-sm-fe enable`	Configures TCA generation for specific parameters on the optical transport network (OTN) layer. OTN parameters are as shown in Step 5 on page 142:

	Command or Action	Purpose
Step 9	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router(config-dwdm)# end` or `RP/0/RP0/CPU0:router(config-dwdm)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 10	**show controllers dwdm** *instance* **pm history** [**15-min** | **24-hour** | **fec** | **optics** | **otn**] **Example:** `RP/0/RP0/CPU0:router# show controllers dwdm 0/2/0/0 pm history 24-hour fec` `RP/0/RP0/CPU0:router# show controllers dwdm 0/2/0/0 pm history`	Displays all of the performance measurement and TCA generation information for the DWDM controller.
Step 11	**show controllers dwdm** *instance* **pm interval** {**15-min** | **24-hour**} [**fec** | **optics** | **otn**] *index* **Example:** `RP/0/RP0/CPU0:router# show controllers dwdm 0/2/0/0 pm interval 24-hour 0` `RP/0/RP0/CPU0:router# show controllers dwdm 0/2/0/0 pm interval 15-min optics 1`	Displays performance measurement and TCA generation information for a specific interval.

Examples

The following example shows how to configure performance monitoring for the optics parameters and how to display the configuration and current statistics:

```
RP/0/RP1/CPU0:roma# config
RP/0/RP1/CPU0:roma(config)# controller dwdm 0/2/0/0

RP/0/RP1/CPU0:roma(config-dwdm)# pm 15-min optics threshold opt max 2000000
RP/0/RP1/CPU0:roma(config-dwdm)# pm 15-min optics threshold opt min 200
RP/0/RP1/CPU0:roma(config-dwdm)# pm 15-min optics threshold lbc max 3000000
RP/0/RP1/CPU0:roma(config-dwdm)# pm 15-min optics threshold lbc min 300
RP/0/RP1/CPU0:roma(config-dwdm)# pm 15-min optics threshold opr max 4000000
RP/0/RP1/CPU0:roma(config-dwdm)# pm 15-min optics threshold opr min 400
RP/0/RP1/CPU0:roma(config-dwdm)# pm 15-min optics report opt max-tca enable
RP/0/RP1/CPU0:roma(config-dwdm)# pm 15-min optics report opt min-tca enable
RP/0/RP1/CPU0:roma(config-dwdm)# pm 15-min optics report opr max-tca enable
```

```
RP/0/RP1/CPU0:roma(config-dwdm)# pm 15-min optics report opr min-tca enable
RP/0/RP1/CPU0:roma(config-dwdm)# pm 15-min optics report lbc max-tca enable
RP/0/RP1/CPU0:roma(config-dwdm)# pm 15-min optics report lbc min-tca enable
RP/0/RP1/CPU0:roma(config-dwdm)# exit
RP/0/RP1/CPU0:roma(config)# exit

Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:y

LC/0/2/CPU0:Jul 12 04:10:47.252 : plim_4p_10ge_dwdm[194]: %L1-PMENGINE-4-TCA : Port DWDM
0/2/0/0 reports OPTICS TX-PWR-MIN(NE) PM TCA with current value 0, threshold 200 in
current 15-min interval window
LC/0/2/CPU0:Jul 12 04:10:47.255 : plim_4p_10ge_dwdm[194]: %L1-PMENGINE-4-TCA : Port DWDM
0/2/0/0 reports OPTICS RX-PWR-MIN(NE) PM TCA with current value 68, threshold 400 in
current 15-min interval window
RP/0/RP1/CPU0:Jul 12 04:09:05.443 : config[65678]: %MGBL-CONFIG-6-DB_COMMIT :
Configuration committed by user 'lab'. Use 'show configuration commit changes 1000000001'
to view the changes.
RP/0/RP1/CPU0:Jul 12 04:09:05.604 : config[65678]: %MGBL-SYS-5-CONFIG_I : Configured from
console by lab

RP/0/RP1/CPU0:roma# show controllers dwdm 0/2/0/0 pm interval 15-min optics 0

Optics in the current interval [ 4:15:00 - 04:26:02 Wed Jul 12 2006]
             MIN     AVG     MAX  Threshold  TCA  Threshold  TCA
                                  (min)  (enable) (max)  (enable)
LBC[mA ] :  3605    4948    6453    300        YES     3000000   YES
OPT[uW]  :  2593    2593    2593    200        YES     2000000   YES
OPR[uW]  :  69      69      70      400        YES     4000000   YES
```

Additional References

The following sections provide references related to DWDM controller configuration.

Related Documents

Related Topic	Document Title
Cisco IOS XR master command reference	*Cisco IOS XR Master Commands List,* Release 3.2
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*
Information about configuring interfaces and other components on the Cisco CRS-1 from a remote Craft Works Interface (CWI) client management application	*Cisco CRS-1 Series Carrier Routing System Craft Works Interface Configuration Guide*

Standards

Standards	Title
ITU-T G.709/Y.1331	Interfaces for the optical transport network (OTN)

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: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

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/techsupport

Upgrading FPD on Cisco IOS XR Software

In general terms, field-programmable devices (FPDs) are hardware devices implemented on router cards that support separate software upgrades. A field-programmable gate array (FPGA) is a type of programmable memory device that exists on most hardware components of a Cisco CRS-1. The term "FPD" has been introduced to collectively and generically describe any type of programmable hardware device on SIPs and shared port adapters (SPAs), including FPGAs. Cisco IOS XR software provides the Cisco FPD upgrade feature to manage the upgrade of FPD images on SIPs and SPAs.

This chapter describes the information that you must know to verify image versions and to perform an upgrade for SPA or SIP FPD images when incompatibilities arise.

Feature History for Upgrading FPD Software on Cisco IOS XR Software

Release	Modification
Release 3.2	SIPs and SPAs were released on the Cisco CRS-1. FPD images were introduced to support SIPs and SPAs.
Release 3.3.0	Support for FPD upgrades was added on the Cisco XR 12000 Series Router. Reload option was added to the **upgrade hw-module fpd** command, including a prompt to inform the user. Possibility for multiple FPD images on a card was added.
Release 3.4.0	No modification.
Release 3.5.0	No modification.

Contents

This chapter includes the following sections:

Overview of FPD Image Upgrade Support

An FPD image is used to upgrade the software on an FPD. Whenever a Cisco IOS XR software image is released that supports SPAs and SIPs, a companion SPA FPD image is companion SIP and SPA FPD image are bundled with the Cisco IOS XR software release. However, the FPD image is not automatically upgraded. You must manually upgrade the FPD image running on the SPA or SIP when you upgrade the Cisco IOS XR software image.

FPD versions must be compatible with the Cisco IOS XR software that is running on the router; if an incompatibility exists between an FPD version and the Cisco IOS XR software, the device with the FPGA may not operate properly until the incompatibility is resolved. An FPGA incompatibility on a SPA does not necessarily affect the running of the SPA interfaces; an FPD incompatibility on a SIP disables all interfaces for all SPAs in the SIP until the incompatibility is addressed.

The Cisco CRS-1 supports upgrades for FPGA devices on its SIPs and SPAs. FPGA software upgrades are part of an FPD image package that corresponds to a Cisco IOS XR software image. SIPs and SPAs support manual upgrades for FPGA devices using the Cisco FPD upgrade feature that is further described in this chapter.

How to Upgrade FPD Images

You must upgrade the FPD images in the following instances:

- Migrate to a later Cisco IOS XR software release
- Swap SPAs or SIPs from a system running s different Cisco IOS XR software release
- Insert a new SPA or SIP

In the event that there is an FPD incompatibility with your SPA, you may receive an error message. If you upgrade to a newer version of the Cisco IOS XR software and there is an FPD incompatibility, you receive the following message:

```
LC/0/1/CPU0:Dec 23 16:33:47.945 : spa_192_jacket_v2[203]: %PLATFORM-UPGRADE_FPD-4-DOWN_REV
: spa fpga2 instance 0 is down-rev (V0.6), upgrade to (V1.0). Use the "upgrade hw-module
fpd" CLI in admin mode.
```

If the FPD image on the SPA is newer then what is required by the currently running Cisco IOS XR software image on the router, you receive the following error message:

```
LC/0/1/CPU0:Dec 23 16:33:47.955 : spa_192_jacket_v2[203]: %PLATFORM-UPGRADE_FPD-4-UP_REV :
spa fpga instance 1 is severely up-rev (V2.1), downgrade to (V1.6). Use the "upgrade
hw-module fpd" CLI in admin mode.
```

You should perform the FPD upgrade procedure if you receive such messages. SPAs may not function properly if FPD incompatibilities are not resolved.

FPD Command Summary

Table 3 provides an alphabetical list of the related commands to configure, monitor, and upgrade FPD images for SPAs on the Cisco CRS-1. For more information about the commands, see *Cisco IOS XR Interface and Hardware Component Command Reference.*

Table 3 FPD Command Summary

Command	Purpose
`RP/0/RP0/CPU0:Router# show hw-module fpd location [all | node-id]`	Displays all current versions of FPD image files for all the active SPAs on a router.
`RP/0/RP0/CPU0:Router(admin)# show fpd package`	Displays the FPD image package requirements for the router to properly support the SPAs running on the Cisco IOS XR software release.
`RP/0/RP0/CPU0:Router(admin)# upgrade hw-module fpd {all | fpga-type} [force] location [all | node-id] [reload]`	Upgrades the current FPD image package on a SPA.

Restrictions

The FPD upgrade procedure is performed while the SIP or SPA is online. At the end of the procedure the SPA or SIP must be reloaded before the FPD upgrade is complete. This can be performed automatically, by using the **reload** keyword in the **upgrade hw-module fpd** command. Alternatively, you can use the **hw-module reload** command during your next maintenance window. The upgrade procedure is not complete until the SPA or SIP is reloaded.

Note Upgrading the FPD image on a SPA or SIP using the **reload** keyword temporarily places the card offline at the end of the upgrade procedure, and may interrupt traffic.

If you are not sure whether a SPA requires an FPD upgrade, you can install the SPA and use the **show hw-module fpd** command to determine if the FPD image on the SPA is compatible with the currently running Cisco IOS XR software release.

SUMMARY STEPS

1. **show hw-module fpd location all**
2. **admin**
3. **show fpd package**
4. **upgrade hw-module fpd** {**all** | *fpga-type*} **location** [**all** | *node-id*] [**reload**]
5. **exit**
6. **hw-module** {**node** *node-id* | **subslot** *subslot-id*} **reload**
7. **show platform**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**show hw-module fpd location all** **Example:** `RP/0/RP0/CPU0:Router# show hw-module fpd location all`	Displays the current FPD image versions for all cards installed in the router. Use this command to determine if you must upgrade the FPD image on your SPA or SIP.
Step 2	**admin** **Example:** `RP/0/RP0/CPU0:Router# admin`	Enters administration EXEC mode from EXEC mode.
Step 3	**show fpd package** **Example:** `RP/0/RP0/CPU0:Router(admin)# show fpd package`	(Optional) Displays which SPAs and SIPs are supported with your current Cisco IOS XR software release, which FPD image you need for each SPA and SIP, and what the minimum hardware requirements are for the SPA and SIP modules. If there are multiple FPD images for your card, use this command to determine which FPD image to use if you only want to upgrade a specific FPD type.
Step 4	**upgrade hw-module fpd all location** *node-id* **reload** **Example:** `RP/0/RP0/CPU0:Router(admin)# upgrade hw-module fpd fpga location 0/3/1 reload`	Upgrades all of the current FPD images that must be upgraded on the specified card with new images. **Note** The **reload** keyword causes the SPA or SIP to be reloaded after the FPD image has been updated. This interrupts traffic transmission. If you do not use the **reload** keyword, you must manually reload the SPA or SIP before the FPD upgrade is complete. Use the **hw-module subslot reload** command to reload a SPA and the **hw-module node reload** command to reload a SIP. **Note** If your SPA or SIP supports multiple FPD images, you can use the **show fpd package** admin command to determine what specific image to upgrade in the **upgrade hw-module fpd** command.
Step 5	**exit** **Example:** `RP/0/RP0/CPU0:Router(admin)# exit`	Exits administration EXEC mode and returns to EXEC mode.

	Command or Action	Purpose
Step 6	**hw-module {node** *node-id* | **subslot** *subslot-id***} reload** **Example:** `RP/0/RP0/CPU0:Router# hw-module subslot 0/3/1 reload` or `RP/0/RP0/CPU0:Router# hw-module node 0/3/cpu0 reload`	(Optional) Reloads the SIP or SPA and the Cisco IOS XR software. **Note** Only use this command if you do not use the **reload** keyword in the **upgrade hw-module fpd** command.
Step 7	**show platform** **Example:** `RP/0/RP0/CPU0:Router# show platform`	Verifies that the FPD image on the SPA or SIP has been successfully upgraded by displaying the status of all cards in the system.

FPD Image Upgrade Examples

The following sample output indicates the use of commands associated with the FPD image upgrade procedure:

- show hw-module fpd Command Output, page 151
- show fpd package Command Output, page 153
- upgrade hw-module fpd Command Output, page 155
- show platform Command Output, page 155

show hw-module fpd Command Output

Use the **show hw-module fpd** to display the current version of FPD images on the SPA and SIPs installed on your router.

The following sample output is from the **show hw-module fpd** command on a Cisco CRS-1. The output display in this example shows that FPD versions on the SPA in the system do not meet the minimum requirements. The output contains a “NOTES” section that states how to upgrade the SPA’s FPD image.

```
RP/0/RP0/CPU0:Router# show hw-module fpd location all

==================================== ========================================
                                     Existing Field Programmable Devices
                                     ========================================
                                       HW                        Current SW Upg/
Location     Card Type               Version Type Subtype Inst   Version   Dng?
============ ======================= ======= ==== ======= ==== =========== ====
0/1/CPU0     CRS1-SIP-800              0.96  lc   fpga    0       2.0       No
------------------------------------------------------------------------------
0/1/0        SPA-4XOC3-POS             1.0   spa  fpga    0       3.4       No
------------------------------------------------------------------------------
0/1/5        SPA-8X1GE                 2.2   spa  fpga    5       1.8       No
------------------------------------------------------------------------------
0/6/CPU0     CRS1-SIP-800              0.96  lc   fpga    0       2.0       No
------------------------------------------------------------------------------
0/6/0        SPA-4XOC3-POS             1.0   spa  fpga    0       3.4       No
------------------------------------------------------------------------------
```

```
0/6/4        SPA-8XOC3-OC12-POS         1.1   spa  fpga    4       0.5      Yes
--------------------------------------------------------------------------------
0/6/5        SPA-8X1GE                  2.2   spa  fpga    5       1.8      No
--------------------------------------------------------------------------------
NOTES:
1.  One or more FPD needs an upgrade or a downgrade.  This can be accomplished
    using the "admin upgrade hw-module fpd" CLI.
```

The following sample output is from the **show hw-module fpd** command on a Cisco XR 12000 Series Router.

```
RP/0/0/CPU0:Router# show hw-module fpd location all

==================================== ========================================
                                     Existing Field Programmable Devices
                                     ========================================
                                       HW                          Current SW Upg/
Location     Card Type                Version Type Subtype Inst   Version   Dng?
============ ======================== ======= ==== ======= ==== =========== ====
0/1/0        SPA-4XT3/E3                1.0   spa  fpga    0       0.24     No
                                              spa  rommon  0       2.12     No
                                              spa  fpga2   0       1.0      No
                                              spa  fpga3   0       1.0      No
--------------------------------------------------------------------------------
0/1/1        SPA-4XCT3/DS0              0.253 spa  fpga    1       2.1      No
                                              spa  rommon  1       2.12     No
                                              spa  fpga2   1       0.15     No
--------------------------------------------------------------------------------
0/3/0        SPA-2XOC48POS/RPR          1.0   spa  fpga    0       1.0      No
--------------------------------------------------------------------------------
0/3/1        SPA-1XTENGE-XFP            3.2   spa  fpga    1       1.7      No
--------------------------------------------------------------------------------
```

Table 4 describes the significant fields shown in the display.

Table 4 show hw-module fpd Field Descriptions

Field	Description
Location	Location of the module in the *rack*/*slot*/*module* notation.
Card Type	Card identifier.
Type	Hardware type. Possible types can be: • spa—Shared port adapter • lc—Line card
Subtype	FPD type. Possible types can be: • fabldr—Fabric downloader • fpga—Field-programmable gate array • rommon—Read-only memory monitor
Inst	Instance—A unique identifier that is used by the FPD process to register an FPD.
Current SW Version	Currently running FPD image version.

Table 4 show hw-module fpd Field Descriptions (continued)

Field	Description
HW Version	Hardware version of the SPA or SIP.
Upg/Dng	Specifies whether an FPD upgrade or downgrade is required. A downgrade is required in rare cases when the version of the FPD image has a higher major revision than the version of the FPD image in the current Cisco IOS XR software package.

Note This command can be used to identify information about FPDs on any SPA. If you enter the location of a line card that is not a SPA, the output displays information about any programmable devices on that line card.

show fpd package Command Output

Use the **show fpd package** command in administration EXEC mode to find out which SPAs and SIPs are supported with your current Cisco IOS XR software release, which FPD image package you need for each SPA or SIP, and what the minimum hardware requirements are for each module. If multiple FPD images are available for your card, they are listed as Subtype fpga2, fpga3, etc.

The following sample output is from the **show fpd package** command on a Cisco CRS-1:

```
RP/0/RP0/CPU0:Router# admin
RP/0/RP0/CPU0:Router(admin)# show fpd package

================================ ================================================
                                        Field Programmable Device Package
                                 ================================================
                                                                  SW      Min Req
Card Type            FPD Description            Type Subtype   Version   HW Vers
==================== ========================== ==== ======= =========== =======
Jacket Card          SPA FPGA swv13             lc   fpga         0.13       0.0
-------------------------------------------------------------------------------
SPA-4XOC3POS         SPA FPGA swv13             spa  fpga         0.13       0.0
                     SPA FPGA swv13 hwv2        spa  fpga         0.13       2.0
-------------------------------------------------------------------------------
SPA-OC192POS-XFP     SPA FPGA swv13             spa  fpga         0.13       0.0
                     SPA FPGA swv13 hwv2        spa  fpga         0.13       2.0
-------------------------------------------------------------------------------
SPA-8X1GE            SPA FPGA swv1.8            spa  fpga         1.8        0.0
-------------------------------------------------------------------------------
```

The following sample output is from the **show fpd package** command on a Cisco XR 12000 Series Router:

```
RP/0/0/CPU0:Router# admin
RP/0/0/CPU0:Router(admin)# show fpd package

================================ ================================================
                                        Field Programmable Device Package
                                 ================================================
                                                                  SW      Min Req
Card Type            FPD Description            Type Subtype   Version   HW Vers
==================== ========================== ==== ======= =========== =======
SPA-4XT3/E3          SPA E3 Subrate FPGA        spa  fpga2        0.6        0.0
                     SPA T3 Subrate FPGA        spa  fpga3        0.14       0.0
```

```
                    SPA I/O FPGA               spa  fpga        0.24      0.0
                    SPA ROMMON                 spa  rommon      2.12      0.0
-------------------------------------------------------------------------------
SPA-2XT3/E3         SPA E3 Subrate FPGA        spa  fpga2       0.6       0.0
                    SPA T3 Subrate FPGA        spa  fpga3       0.14      0.0
                    SPA I/O FPGA               spa  fpga        0.24      0.0
                    SPA ROMMON                 spa  rommon      2.12      0.0
-------------------------------------------------------------------------------
SPA-4XCT3/DS0       SPA T3 Subrate FPGA        spa  fpga2       0.11      0.100
                    SPA T3 Subrate FPGA        spa  fpga2       0.15      0.200
                    SPA I/O FPGA               spa  fpga        1.4       0.100
                    SPA ROMMON                 spa  rommon      2.12      0.100
-------------------------------------------------------------------------------
SPA-2XCT3/DS0       SPA T3 Subrate FPGA        spa  fpga2       0.11      0.100
                    SPA T3 Subrate FPGA        spa  fpga2       0.15      0.200
                    SPA I/O FPGA               spa  fpga        1.4       0.100
                    SPA ROMMON                 spa  rommon      2.12      0.100
-------------------------------------------------------------------------------
SPA-OC192RPR-XFP    SPA FPGA swv1.2            spa  fpga        1.2       0.0
-------------------------------------------------------------------------------
SPA-OC192POS-XFP    SPA FPGA swv1.2            spa  fpga        1.2       0.0
                    SPA FPGA swv1.2 hwv2       spa  fpga        1.2       2.0
-------------------------------------------------------------------------------
SPA-10X1GE          SPA FPGA swv1.7            spa  fpga        1.7       0.0
-------------------------------------------------------------------------------
SPA-5X1GE           SPA FPGA swv1.7            spa  fpga        1.7       0.0
-------------------------------------------------------------------------------
SPA-1XTENGE-XFP     SPA FPGA swv1.7            spa  fpga        1.7       0.0
-------------------------------------------------------------------------------
```

Table 5 describes the significant fields shown in the display.

Table 5 show fpd package Field Descriptions

Field	Description
Card Type	Module part number.
FPD Description	Description of all FPD images available for the SPA.
Type	Hardware type. Possible types can be: • spa—Shared port adapter • lc—Line card.
Subtype	FPD subtype. These values are used in the **upgrade hw-module fpd** command to indicate a specific FPD image type to upgrade.
SW Version	FPD software version required for the associated module running the current Cisco IOS XR software.
Min Req HW Vers	Minimum required hardware version for the associated FPD image.

Note In the **show fpd package** command output, the "subtype" column shows the FPDs that correspond with each SPA image. To upgrade a specific FPD with the **upgrade hw-module fpd** command, replace the *fpga-type* argument with the appropriate FPD from the "subtype" column, as shown in the following example:

```
RP/0/0/CPU0:Router(admin)# upgrade hw-module fpd fpga2 location 0/3/1 reload
```

upgrade hw-module fpd Command Output

Use **upgrade hw-module fpd** command to upgrade the FPD image on a SPA or SIP, as shown in the following example:

```
RP/0/RP0/CPU0:Router# admin
RP/0/RP0/CPU0:Router(admin)# upgrade hw-module fpd fpga force location 0/1/4

% RELOAD REMINDER:
  - The upgrade operation of the target module will not interrupt its normal
    operation. However, for the changes to take effect, the target module
    will need to be manually reloaded after the upgrade operation. This can
    be accomplished with the use of "hw-module <target> reload" command.
  - If automatic reload operation is desired after the upgrade, please use
    the "reload" option at the end of the upgrade command.
  - The output of "show hw-module fpd location" command will not display
    correct version information after the upgrade if the target module is
    not reloaded.
Continue? [confirm] y

SP/0/1/SP:Dec 22 05:41:17.920 : upgrade_daemon[125]: programming...with file /ne
t/node0_RP1_CPU0/hfr-lc-3.3.83/fpd/ucode/fpga_gladiator_sw0.6.xsvf
SP/0/1/SP:Dec 22 05:41:28.900 : upgrade_daemon[125]: ...programming...
SP/0/1/SP:Dec 22 05:41:28.906 : upgrade_daemon[125]: ...it will take a while...
SP/0/1/SP:Dec 22 05:41:29.004 : upgrade_daemon[125]: ...it will take a while...
SP/0/1/SP:Dec 22 05:43:03.432 : upgrade_daemon[125]: ...programming...
SP/0/1/SP:Dec 22 05:43:03.438 : upgrade_daemon[125]: ...it will take a while...
Successfully upgraded spa fpga instance 4 on location 0/1/4.
```

Use the **reload** option to force a manual reload after the FPD image is upgraded:

```
RP/0/RP0/CPU0:Router(admin)# upgrade hw-module fpd all location 0/1/1 reload

% RELOAD WARNING:
  - Option to automatically reload the target module after upgrade was
    selected. This action will interrupt normal operation of the module.
  - If necessary, ensure that appropriate actions have been taken to
    redirect target module's traffic before starting the upgrade operation.
  - Note that this reload option does not have effect on target module that
    has been placed in administrative shutdown state.
Continue? [confirm]
```

show platform Command Output

Use the **show platform** command to verify that the SPA is up and running.

The following **show platform** command output is from a Cisco CRS-1:

```
RP/0/RP0/CPU0:Router# show platform

Node            Type            PLIM            State           Config State
-------------------------------------------------------------------------------
0/1/SP          MSC(SP)         N/A             IOS XR RUN      PWR,NSHUT,MON
0/1/CPU0        MSC             Jacket Card     IOS XR RUN      PWR,NSHUT,MON
0/1/0           MSC(SPA)        4XOC3-POS       OK              PWR,NSHUT,MON
0/1/1           MSC(SPA)        OC192RPR-XFP    OK              PWR,NSHUT,MON
0/1/4           MSC(SPA)        8XOC3/OC12-POS  OK              PWR,NSHUT,MON
0/RP1/CPU0      RP(Active)      N/A             IOS XR RUN      PWR,NSHUT,MON
0/SM0/SP        FC/S(SP)        N/A             IOS XR RUN      PWR,NSHUT,MON
0/SM1/SP        FC/S(SP)        N/A             IOS XR RUN      PWR,NSHUT,MON
```

The following **show platform** command output is from a Cisco XR 12000 Series Router:

```
RP/0/0/CPU0:Router# show platform

Node            Type            PLIM            State           Config State
-------------------------------------------------------------------------------
0/0/CPU0        L3LC Eng 5+     Jacket Card     IOS-XR RUN      PWR,NSHUT,MON
0/0/0           SPA             SPA-2XOC48POS/R READY           PWR,NSHUT
0/0/2           SPA             SPA-8XFE-TX     READY           PWR,NSHUT
0/0/3           SPA             SPA-4XCT3/DS0   READY           PWR,NSHUT
0/1/CPU0        L3LC Eng 3      OC48-POS        IOS-XR RUN      PWR,NSHUT,MON
0/2/CPU0        L3LC Eng 3      OC3-POS-16      IOS-XR RUN      PWR,NSHUT,MON
0/3/CPU0        L3LC Eng 5      Jacket Card     IOS-XR RUN      PWR,NSHUT,MON
0/3/1           SPA             SPA-OC192POS-XF READY           PWR,NSHUT
0/7/CPU0        L3LC Eng 5      Jacket Card     IOS-XR RUN      PWR,NSHUT,MON
0/7/0           SPA             SPA-10X1GE      READY           PWR,NSHUT
0/7/1           SPA             SPA-1XTENGE-XFP READY           PWR,NSHUT
0/8/CPU0        PRP             N/A             Card Present    PWR,NSHUT,MON
0/9/CPU0        PRP(Active)     N/A             IOS-XR RUN      PWR,NSHUT,MON
```

Troubleshooting Problems with FPD Image Upgrades

This section contains information to help troubleshoot problems that can occur during the upgrade process.

Power Failure or Removal of a SPA During an FPD Image Upgrade

If the FPD upgrade operation is interrupted by a power failure or the removal of the SPA, it could corrupt the FPD image. This corruption of the FPD image file makes the SPA unusable by the router and the system displays the following messages when it tries to power up the SPA. When it cannot successfully power up the SPA, it places it in the failed state, as shown in the following example:

```
LC/0/3/CPU0:Feb  4 08:23:16.672 : spa_192_jacket[188]: %L2-SPA-5-OIR_INSERTED : SPA
discovered in bay 0
LC/0/3/CPU0:Feb  4 08:23:23.349 : spa_192_jacket[188]: %L2-SPA-5-OIR_ERROR : SPA (0): An
error occurred (0x1002), error recovery action: reset SPA
LC/0/3/CPU0:Feb  4 08:23:26.431 : spa_192_jacket[188]: %L2-SPA-5-OIR_INSERTED : SPA
discovered in bay 0
LC/0/3/CPU0:Feb  4 08:23:32.593 : spa_192_jacket[188]: %L2-SPA-5-OIR_ERROR : SPA (0): Too
many retries, error recovery stopped
LC/0/3/CPU0:Feb  4 08:23:32.593 : spa_192_jacket[188]: %L2-SPA-5-OIR_ERROR : SPA (0): An
error occurred (0x1002), error recovery action: hold SPA in reset
```

When a SPA is in the failed state, it may not register itself with the FPD upgrade mechanism. In this case, you do not see the SPA listed when you use the **show hw-module fpd** command. To verify the state of a SPA, use the **show hw-module subslot error** command and the **show hw-module subslot status** command.

Performing a SPA FPD Recovery Upgrade

To recover a SPA from the failed state due to a corrupted FPD image, you must manually shut down the SPA. Use the **hw-module subslot** *subslot-id* **shutdown** command in global configuration mode to administratively shutdown the SPA. After the SPA is shut down, you can use the **upgrade hw-module fpd** command in administration EXEC mode, with the **force** option, to restart the FPD upgrade process, as shown in the following example:

```
RP/0/0/CPU0:Router# admin
RP/0/0/CPU0:Router(admin)# upgrade hw-module fpd fpga force location 0/3/0
```

Performing a SIP FPD Recovery Upgrade

If a SIP upgrade fails for whatever reason, do not reload the SIP. Try to perform the upgrade procedure again. You can perform the upgrade procedure multiple times, as long as you do not reload the SIP. The FPD upgrade procedure takes several minutes to complete; do not interrupt the procedure. If you reload the SIP when the FPD image is corrupted, the SIP malfunctions and you must contact Cisco technical support for assistance.

To recover a SIP from the failed state due to a corrupted FPD image, you must contact Cisco technical support.

To recover a SIP from the failed state due to a corrupted FPD image, you must turn off the automatic reset of the SIP card. Use the **hw-module reset auto disable** command in administration configuration mode, as shown in the following example:

```
RP/0/RP0/CPU0:Router(admin-config)# hw-module reset auto disable location 0/1/4
```

Additional References

The following sections provide references related to interface configuration.

Related Documents

Related Topic	Document Title
Cisco IOS XR master command reference	*Cisco IOS XR Master Commands List,* Release 3.2
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*
Information about configuring interfaces and other components on the Cisco CRS-1 from a remote Craft Works Interface (CWI) client management application	*Cisco CRS-1 Series Carrier Routing System Craft Works Interface 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: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

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/techsupport

Modifying the Default Frame Relay Configuration on Cisco IOS XR Software

This module describes the optional configurable Frame Relay parameters available on Packet-over-SONET/SDH (POS) 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. Layer 2 Tunnel Protocol Version 3 (L2TPv3) was supported on serial interfaces with Frame Relay encapsulation.

Contents

Prerequisites for Configuring Frame Relay

Before configuring Frame Relay, be sure that the following conditions are met:

- You must be in a user group associated with a task group that includes the proper task IDs for Frame Relay commands. Task IDs for commands are listed in the *Cisco IOS XR Interface and Hardware Component Command Reference*.
- 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 chapter:
  - 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 Software" module in this manual.

Information About Configuring 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 **encap** command in Frame Relay PVC configuration mode to configure Cisco or IETF encapsulation on the main interface.

Note If the encapsulation type is not configured explicitly for a PVC with the **encap** command, then that PVC inherits the encapsulation type from the main interface.

The **encapsulation frame relay** and **encap** commands are described in the following chapters:

– 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 Software" 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 6 are present. These default settings can be changed by configuration as described in this module.

g
Table 6 Frame Relay Encapsulation Default Settings

Parameter	Configuration File Entry	Default Settings	Command Mode
PVC Encapsulation	**encap** [**cisco** | **ietf**]	**cisco** **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

Table 6 Frame Relay Encapsulation Default Settings

Parameter	Configuration File Entry	Default Settings	Command Mode
LMI type supported on the interface	**frame-relay lmi-type** {**ansi** | **cisco** | **q933a**}	For a DCE, the default setting is **cisco**. For a DTE, the default setting is synchronized to match the LMI type supported on the DCE. **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. To reenable LMI on an interface after it has been disabled, use the **no frame-relay lmi disable** command.	interface configuration

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 must match on both ends of a Frame Relay connection. We recommend using the default LMI type.

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** command in EXEC mode to display statistical information for the Frame Relay interfaces in your system. 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 with **cisco** LMI is 557.

For non-Cisco LMI types, 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 7 describes the commands that can be used to modify LMI polling options on PVCs configured for a DCE.

Table 7 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 8 describes the commands that can be used to modify LMI polling options on PVCs configured for a DTE.

Table 8 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**

Modifying the Default Frame Relay Configuration on an Interface

Perform this task to modify the default Frame Relay parameters on POS or serial interfaces that have Frame Relay encapsulation enabled.

Prerequisites

Before you can modify the default Frame Relay configuration, you need to enable Frame Relay on the interface, as described in the following chapters:

- 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 Software” module in this manual.

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 cHDLC encapsulation, you need remove all interfaces, subinterface, ILMI, and Frame Relay configuration from that interface.

SUMMARY STEPS

1. **configure**
2. **interface** *type instance*
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** *type* [*instance*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface** *type instance* **Example:** `RP/0/0/CPU0:router(config)# interface pos 0/4/0/1`	Enters the interface configuration mode.
Step 3	**frame-relay intf-type** [**dce** | **dte**] **Example:** `RP/0/0/CPU0:router(config-if)# frame-relay intf-type dce`	Configures the type of support provided by the interface. • 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). • 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** The default interface type is DTE.

	Command or Action	Purpose
Step 4	**frame-relay lmi-type** {**ansi** | **q933a** | **cisco**} **Example:** `RP/0/0/CPU0:router(config-if)# frame-relay lmi-type ansi`	Selects the LMI type supported on the interface. • Enter the **frame-relay lmi-type ansi** command to use LMI as defined by ANSI T1.617a-1994 Annex D. • Enter the **frame-relay lmi-type cisco** command to use LMI as defined by Cisco (not standard). • Enter the **frame-relay lmi-type q933a** command to use LMI as defined by ITU-T Q.933 (02/2003) Annex A. **Note** The default LMI type is Cisco.
Step 5	**encap** [**cisco** | **ietf**] **Example:** `RP/0/0/CPU0:router (config-subif)# encap ietf`	Configures the encapsulation for a Frame Relay PVC. **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 6	**end** or **commit** **Example:** `RP/0/0/CPU0:router(config-if)# end` or `RP/0/0/CPU0:router(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 7	**show interfaces** *type* [*instance*] **Example:** `RP/0/RP0/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 instance*
3. **frame-relay lmi disable**
4. **end**
   or
   commit
5. **show interfaces** *type* [*instance*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface** *type instance* **Example:** `RP/0/0/CPU0:router(config)# interface POS 0/4/0/1`	Enters the interface configuration mode.
Step 3	**frame-relay lmi disable** **Example:** `RP/0/0/CPU0:router(config-if)# frame-relay lmi disable`	Disables LMI on the specified interface.

	Command or Action	Purpose
Step 4	**end** or **commit** **Example:** `RP/0/0/CPU0:router(config-if)# end` or `RP/0/0/CPU0:router(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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 instance* **Example:** `RP/0/0/CPU0:router# show interfaces POS 0/1/0/0`	(Optional) Verifies that LMI is disabled on the specified interface.

Configuration Examples for Frame Relay

This section provides the following configuration examples:

• "Configuring Optional Frame Relay Parameters" section on page 166

Configuring Optional Frame Relay Parameters

The following example shows how to bring up a 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/RP0/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 serial interface that has Frame Relay encapsulation configured:

```
RP/0/0/CPU0:router# configure

RP/0/0/CPU0:router(config)# interface

RP/0/RP0/CPU0:router(config)# interface serial 0/3/0/0/0:0

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/RP0/CPU0:router(config)# interface serial 0/3/0/0/0:0

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 about the LMI on a POS interface:

```
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/RP0/CPU0:router# configure

RP/0/RP0/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** command to display information for monitoring and troubleshooting the interface:

```
RP/0/RP0/CPU0:router# configure

RP/0/RP0/CPU0:router(config)# interface serial 0/3/0/0/0

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

LMI Statistics for interface serial 0/3/0/0/0 (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
```

Additional References

The following 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,* Release 3.4
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*
Information about configuring interfaces and other components on the Cisco CRS-1 from a remote Craft Works Interface (CWI) client management application	*Cisco CRS-1 Series Carrier Routing System Craft Works Interface Configuration Guide*

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://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

RFCs	Title
RFC 1294	*Multiprotocol Interconnect Over Frame Relay*
RFC 1490	*Multiprotocol Interconnect Over Frame Relay*
RFC 2427	*Multiprotocol Interconnect Over Frame Relay*
RFC 1586	*Guidelines for Running OSPF Over Frame Relay Networks*
RFC 1315	*Management Information Base for Frame Relay DTEs*
RFC 2115	*Management Information Base for Frame Relay DTEs Using SMIv2*
RFC 1604	*Definitions of Managed Objects for Frame Relay Service*
RFC 2954	*Definitions of Managed Objects for Frame Relay Service*
RFC 2390	*Inverse Address Resolution Protocol*

Technical Assistance

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

Configuring Ethernet Interfaces on Cisco IOS XR Software

This module describes the configuration of Ethernet interfaces on routers supporting Cisco IOS XR software.

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 and 3 switches.

Note This module does not include configuration information for Management Ethernet interfaces. To set up a Management Ethernet interface and enable telnet servers, see the *Cisco IOS XR Getting Started Guide*. To configure a Management Ethernet interface for routing or modify the configuration of a Management Ethernet interface, see the *Advanced Configuration and Modification of the Management Ethernet Interface on Cisco IOS XR Software* module later in this document.

Feature History for Configuring Ethernet Interfaces on Cisco IOS XR Software

Release	Modification
Release 3.0	This feature was introduced on the Cisco CRS-1.
Release 3.2	Support was added for the Cisco XR 12000 Series Router. Support was added for the Cisco CRS-1 SIP-800. Support was added on the Cisco XR 12000 Series Router for the following SPAs: • 1-Port 10-Gigabit Ethernet SPA • 5-Port Gigabit Ethernet SPA • 10-Port Gigabit Ethernet SPA Support for the 8-Port Gigabit Ethernet single SPA was introduced on the Cisco CRS-1.

Release 3.3.0	Support was added for egress MAC accounting on the 8-port 10-Gigabit Ethernet PLIM. Support was added on the Cisco XR 12000 Series Router for the following SIPs: • Cisco XR 12000 SIP-401 • Cisco XR 12000 SIP-501 • Cisco XR 12000 SIP-601 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 on the Cisco CRS-1 and Cisco XR 12000 Series Router. Support was added on Cisco CRS-1 and Cisco XR 12000 Series Router 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.
Release 3.5.0	Support was added on Cisco CRS-1 for the 1-port 10-Gigabit Ethernet WAN SPA.

Contents

Prerequisites for Configuring Ethernet Interfaces

Before configuring Ethernet interfaces, be sure that the following tasks and conditions are met:

- To use the Ethernet commands, you must be in a user group associated with a task group that includes the proper task IDs for interface commands.

  Task IDs for commands are listed in the *Cisco IOS XR Interface and Hardware Component Command Reference*.
- For the Cisco CRS-1, confirm that at least one of the following cards is installed on the router:
  - 8-port 10-Gigabit Ethernet (8 x 10 GE) physical layer interface modules (PLIM)
  - 8-port 1-Gigabit Ethernet shared port adapter (SPA)
- For the Cisco XR 12000 Series Router, confirm that at least one of the following cards is installed on the router:
  - 4-port 1-Gigabit Ethernet PLIM

  - 2-Port Gigabit Ethernet SPA
  - 5-port 1-Gigabit Ethernet SPA
  - 10-port 1-Gigabit Ethernet SPA
  - 1-port 10-Gigabit Ethernet SPA
  - 1-port 10-Gigabit Ethernet WAN SPA
  - 8-Port 1-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, as described in the Configuring Dense Wavelength Division Multiplexing Controllers on Cisco IOS XR Software module earlier in this document.

Information About Configuring Ethernet Interfaces

To configure 10-Gigabit Ethernet interfaces, you must understand the following concepts:

Ethernet Technology Overview

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 CRS-1 supports Gigabit Ethernet (1000 Mbps) and 10-Gigabit Ethernet (10 Gbps) interfaces.

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

Default Configuration Values for Gigabit Ethernet and 10-Gigabit Ethernet

Table 9 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 9 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 ingress off
MTU	**mtu**	1514 bytes for normal frames 1518 bytes for 802.1Q tagged frames. 1522 bytes for Q-in-Q frames
MAC address	**mac address**	Hardware burned-in address (BIA[1])

1. burned-in address

Default Configuration Values for Fast Ethernet

Table 10 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 10 Fast Ethernet Default Configuration Values

Parameter	Configuration File Entry	Default Value
MAC accounting	**mac-accounting**	**off**
Duplex operation	**duplex full** **duplex half**	Autonegotiates duplex operation

Table 10 Fast Ethernet Default Configuration Values

Parameter	Configuration File Entry	Default Value
MTU	**mtu**	1500 bytes
Interface speed	**speed**	100 Mbps
Autonegotiation	**negotiation auto**	**disable**

Layer 2 VPN on Ethernet Interfaces

Layer 2 Virtual Private Network (L2VPN) connections emulate the behavior of a LAN across an 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 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 a pseudowire over the SP core network to another edge router. The edge router sends the traffic down another AC to the customer's remote site.

The L2VPN feature enables the connection between different types of Layer 2 attachment circuits and pseudo-wires, 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 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 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 or access network 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 802.1Q VLAN Interfaces on Cisco IOS XR Software" chapter later in this manual. 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 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 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 and MTU configuration.
- If your network requires that packets are transported transparently, you may need to modify the packet's destination MAC 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 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.*

Gigabit Ethernet Protocol Standards Overview

The Gigabit Ethernet interfaces support the following protocol standards:

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

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 and fiber-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.

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.

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.

VLAN configuration is described in this document in the *Configuring 802.1Q VLAN Interfaces on Cisco IOS XR Software* module.

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 failover 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.

Note Duplex operation is autonegotiated by default.

Duplex mode is configurable 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 autonegotiation is enabled on an interface, then the default speed negotiated.
- If autonegotiation is disabled on an interface, then 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 autonegotiated 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.

Table 11 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.

Table 11 Relationship Between duplex and speed Commands

duplex Command	speed Command	Resulting System Action
no duplex	**no speed**	Autonegotiates both speed and duplex modes.
no duplex	**speed 100**	Autonegotiates for duplex mode and forces 100 Mbps.
no duplex	**speed 10**	Autonegotiates for duplex mode and forces 10 Mbps.
full-duplex	**no speed**	Forces full duplex and autonegotiates for speed.
full-duplex	**speed 100**	Forces 100 Mbps and full duplex.
full-duplex	**speed 10**	Forces 10 Mbps and full duplex.

Table 11 Relationship Between duplex and speed Commands (continued)

duplex Command	speed Command	Resulting System Action
half-duplex	**no speed**	Forces half duplex and autonegotiates for speed (10 or 100 Mbps.)
half-duplex	**speed 100**	Forces 100 Mbps and half duplex.
half-duplex	**speed 10**	Forces 10 Mbps and half duplex.

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.

How to Configure Ethernet Interfaces

The following tasks are described in this section:

- Configuring a Gigabit Ethernet or 10-Gigabit Ethernet Interface, page 182
- Configuring a Fast Ethernet Interface, page 185
- Configuring MAC Accounting on an Ethernet Interface, page 188

Configuring a Gigabit Ethernet or 10-Gigabit Ethernet Interface

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**] *instance*
3. **configure**
4. **interface** [**GigabitEthernet | TenGigE**] *instance*
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 interfaces only)
10. **no shutdown**

11. **end**
    or
    commit
12. **show interfaces** [**GigabitEthernet** | **TenGigE**] *instance*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**show version** **Example:** `RP/0/RP0/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 interface** [**GigabitEthernet** | **TenGigE**] *instance* **Example:** `RP/0/RP0/CPU0:router# show interface TenGigE 0/1/0/0`	(Optional) Displays the configured interface and checks the status of each interface port. Possible interface types for this procedure are: • GigabitEthernet • TenGigE
Step 3	**configure** **Example:** `RP/0/RP0/CPU0:router# configure terminal`	Enters global configuration mode.
Step 4	**interface** [**GigabitEthernet** | **TenGigE**] *instance* **Example:** `RP/0/RP0/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: • GigabitEthernet • TenGigE **Note** The example indicates an 8-port 10-Gigabit Ethernet interface in modular services card slot 1.
Step 5	**ipv4 address** *ip-address mask* **Example:** `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. • Replace *ip-address* with the primary IPv4 address for the interface. • Replace *mask* with the mask for the associated IP subnet. The network mask can be specified in either of two ways: – 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. – The network mask can be indicated as a slash (/) and number. For example, /8 indicates that the first 8 bits of the mask are ones, and the corresponding bits of the address are network address.

	Command or Action	Purpose
Step 6	**flow-control** {**bidirectional**| **egress** | **ingress**} **Example:** `RP/0/RP0/CPU0:router(config-if)# flow control ingress`	(Optional) Enables the sending and processing of flow control pause frames. • **egress**—enables the sending of flow control pause frames in egress. • **ingress**—enables the processing of received pause frames on ingress. • **bidirectional**—enables the sending of flow control pause frames in egress and the processing of received pause frames on ingress.
Step 7	**mtu** *bytes* **Example:** `RP/0/RP0/CPU0:router(config-if)# mtu 1448`	(Optional) Sets the MTU value for the interface. • The default is 1514 bytes for normal frames and 1518 bytes for 802.1Q tagged frames. • The range for Gigabit Ethernet and 10-Gigabit Ethernet mtu values is 64 bytes to 65535 bytes.
Step 8	**mac-address** *value1.value2.value3* **Example:** `RP/0/RP0/CPU0:router(config-if)# mac address 0001.2468.ABCD`	(Optional) Sets the MAC layer address of the Management Ethernet interface. • 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** **Example:** `RP/0/0/CPU0:router(config-if)# negotiation auto`	(Optional) Enables autonegotiation on a Gigabit Ethernet interface. • 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. • If autonegotiation is enabled, any manually-configured speed or duplex settings take precedence. **Note** The **negotiation auto** command is available on Gigabit Ethernet and Fast Ethernet interfaces only.
Step 10	**no shutdown** **Example:** `RP/0/RP0/CPU0:router(config-if)# no shutdown`	Removes the shutdown configuration, which forces an interface administratively down.

	Command or Action	Purpose
Step 11	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router(config-if)# end` or `RP/0/RP0/CPU0:router(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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**] *instance* **Example:** `RP/0/RP0/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 802.1Q VLAN subinterface on the Ethernet interface, see the "Configuring 802.1Q VLAN Interfaces on Cisco IOS XR Software" module later in this manual.
- To configure an AC on the Ethernet port for Layer 2 VPN implementation, see the "Configuring an Attachment Circuit on an 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** *instance*
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** *instance*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure terminal`	Enters global configuration mode.
Step 2	**interface fastethernet** *instance* **Example:** `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*. • The example indicates the second interface on an 8-port Fast Ethernet SPA in slot 0, and SPA subslot 2.
Step 3	**ipv4 address** *ip-address mask* **Example:** `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. • Replace *ip-address* with the primary IPv4 address for the interface. • Replace *mask* with the mask for the associated IP subnet. The network mask can be specified in either of two ways: – 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. – The network mask can be indicated as a slash (/) and number. For example, /8 indicates that the first 8 bits of the mask are ones, and the corresponding bits of the address are network address.

	Command or Action	Purpose
Step 4	**mtu** *bytes* **Example:** `RP/0/0/CPU0:router(config-if# mtu 1448`	(Optional) Sets the MTU value for the interface. • The default is 1500 bytes. • The range for Fast Ethernet mtu values is 0 to 10240 bytes.
Step 5	**duplex full** **Example:** `RP/0/0/CPU0:router(config-if)# duplex full`	(Optional) Enables full-duplex mode. Use the **duplex half** command to enable half-duplex mode. • Duplex operation is autonegotiated by default. • If autonegotiation is enabled, any user-defined duplex value is ignored.
Step 6	**speed** *speed* **Example:** `RP/0/0/CPU0:router(config-if)# speed 100`	(Optional) Sets the speed of the interface. Valid options are 10 Mbps or 100 Mbps. • The default is 100 Mbps. • If autonegotiation is enabled, any user-defined speed value is ignored.
Step 7	**negotiation auto** **Example:** `RP/0/0/CPU0:router(config-if)# negotiation auto`	(Optional) Enables autonegotiation on the interface. • 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. • If autonegotiation is enabled, any manually-configured speed or duplex settings take precedence.
Step 8	**no shutdown** **Example:** `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** or **commit** **Example:** `RP/0/0/CPU0:router(config-if)# end` or `RP/0/0/CPU0:router(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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** *instance* **Example:** `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 MAC Accounting on the Fast Ethernet interface, see the "Configuring MAC Accounting on an Ethernet Interface" section later in this module.
- To configure an 802.1Q VLAN subinterface on the Fast Ethernet interface, see the "Configuring 802.1Q VLAN Interfaces on Cisco IOS XR Software" module later in this manual.
- To configure an AC on the Fast Ethernet port for Layer 2 VPN implementation, see the "Configuring an Attachment Circuit on an 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 a Gigabit Ethernet or 10-Gigabit Ethernet Interface" in this module for information about configuring the other common parameters for Ethernet interfaces.

SUMMARY STEPS

1. **configure**
2. **interface** [**GigabitEthernet** | **TenGigE** | **fastethernet**] *instance*
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** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface** [**GigabitEthernet** | **TenGigE** | **fastethernet**] *instance* **Example:** `RP/0/RP0/CPU0:router(config)# interface TenGigE 0/1/0/0`	Enters the interface configuration mode and specifies the Ethernet interface name and instance in the *rack/slot/module/port* notation. • The example indicates an 8-PORT 10-Gigabit Ethernet interface in modular services card slot 1.
Step 3	**ipv4 address** *ip-address mask* **Example:** `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. • Replace *ip-address* with the primary IPv4 address for the interface. • Replace *mask* with the mask for the associated IP subnet. The network mask can be specified in either of two ways: – 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. – 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**} **Example:** `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. • To disable MAC accounting, use the **no** form of this command.

	Command or Action	Purpose
Step 5	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router(config-if)# end` or `RP/0/RP0/CPU0:router(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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* **Example:** `RP/0/RP0/CPU0:router# show mac-accounting TenGigE location 0/2/0/4`	Displays MAC accounting statistics for an interface.

Configuring an Attachment Circuit on an Ethernet Port

Use the following procedure to configure an attachment circuit on a Gigabit Ethernet, 10-Gigabit Ethernet, or Fast 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**] *instance*
3. **l2transport**
4. **l2protocol** {**cdp** | **pvst** | **stp** | **vtp**} {[**tunnel**] **experimental** *bits* | **drop**}
5. **end**
   or
   commit
6. **show interfaces** [**GigabitEthernet** | **TenGigE**] *instance*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure terminal`	Enters global configuration mode.
Step 2	**interface** [**GigabitEthernet** | **TenGigE**] *instance* **Example:** `RP/0/RP0/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: • GigabitEthernet • TenGigE
Step 3	**l2transport** **Example:** `RP/0/RP0/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**} {[**tunnel**] **experimental** *bits* | **drop**} **Example:** `RP/0/RP0/CPU0:router(config-if-l2)#`	Configures Layer 2 protocol tunneling and data units parameters on an interface. Possible protocols are: • **cdp**—Cisco Discovery Protocol (CDP) tunneling and data unit parameters. • **pvst**—Configures VLAN spanning tree protocol tunneling and data unit parameters. • **stp—**spanning tree protocol tunneling and data unit parameters for the • **vtp—**VLAN trunk protocol tunneling and data unit parameters. Include the **tunnel** option if you want to tunnel the packets associated with the specified protocol. Include the **experimental** *bits* keyword argument to modify the MPLS experimental bits for the specified protocol. Include the **drop** keyword to drop packets associated with the specified protocol.

	Command or Action	Purpose
Step 5	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router(config-if-l2)# end` or `RP/0/RP0/CPU0:router(config-if-l2)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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**] *instance* **Example:** `RP/0/RP0/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 pseudo-wire XConnect on the 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 Interfaces

This section provides the following configuration examples:

- Configuring an Ethernet Interface: Example
- Configuring a Fast Ethernet Interface: Example
- Configuring MAC-accounting: Example
- Configuring a Layer 2 VPN AC: Example

Configuring an Ethernet Interface: Example

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

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface TenGigE 0/0/0/1
RP/0/RP0/CPU0:router(config-if)# ipv4 address 172.18.189.38 255.255.255.224
RP/0/RP0/CPU0:router(config-if)# flow-control ingress
RP/0/RP0/CPU0:router(config-if)# mtu 1448
RP/0/RP0/CPU0:router(config-if)# mac-address 0001.2468.ABCD
RP/0/RP0/CPU0:router(config-if)# no shutdown
RP/0/RP0/CPU0:router(config-if)# end
Uncommitted changes found, commit them? [yes]: yes

RP/0/RP0/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/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface TenGigE 0/0/0/2
RP/0/RP0/CPU0:router(config-if)# ipv4 address 172.18.189.38 255.255.255.224
RP/0/RP0/CPU0:router(config-if)# mac-accounting egress
RP/0/RP0/CPU0:router(config-if)# commit
RP/0/RP0/CPU0:router(config-if)# exit
RP/0/RP0/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/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface TenGigE 0/0/0/2
RP/0/RP0/CPU0:router(config-if)# ipv4 address 172.18.189.38 255.255.255.224
RP/0/RP0/CPU0:router(config-if)# l2transport
RP/0/RP0/CPU0:router(config-if-l2)# l2protocol cdp drop
RP/0/RP0/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 configuring VLAN subinterfaces, see the Configuring 802.1Q VLAN Interfaces on Cisco IOS XR Software module later in this document.

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

The following 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*, Release 3.2
Cisco IOS XR interface configuration commands	*Cisco CRS-1 Series Interface and Hardware Component Command Reference*
Information about user groups and task IDs	*Cisco IOS XR Interface and Hardware Component Command Reference*
Information about configuring interfaces and other components on the Cisco CRS-1 from a remote Craft Works Interface (CWI) client management application	*Cisco CRS-1 Series Carrier Routing System Craft Works Interface 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: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

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/techsupport

Configuring Virtual Loopback and Null Interfaces on Cisco IOS XR Software

This module describes the configuration of loopback and null interfaces on routers supporting Cisco IOS XR Software. 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 failover, 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 2.0	This feature was introduced on the Cisco CRS-1.
Release 3.0	No modifications.
Release 3.2	Support was added for the Cisco XR 12000 Series Router.
Release 3.3.0	No modifications.
Release 3.4.0	This chapter was updated to include information on configuring virtual IPV4 management interfaces.
Release 3.5.0	No modifications.

Contents

Prerequisites for Configuring Virtual Interfaces

To use the **ipv4 unreachables** command in conjunction with configuring a null interface, you must be in a user group associated with a task group that includes the proper task IDs for network commands.

Task IDs for commands are listed in *Cisco IOS XR Interface and Hardware Component Command Reference*.

Information About Configuring Virtual Interfaces

To configure virtual interfaces, you must understand the following concepts:

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) failover 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 failover, 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 failover. The routers use nonstop forwarding (NSF) over bundles and tunnels through the failover of the host RP.

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

How to Configure Virtual Interfaces

This section contains the following procedures:

- Configuring Virtual Loopback Interfaces, page 200 (Required)
- Configuring Null Interfaces, page 201 (Required)
- Configuring Virtual IPV4 Interfaces, page 203 (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** *instance*
3. **ipv4 address** *ip-address*
4. **end**
   or
   commit
5. **show interfaces** *type instance*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface loopback** *instance* **Example:** `RP/0/RP0/CPU0:router#(config)# interface Loopback 3`	Enters interface configuration mode and names the new loopback interface.
Step 3	**ipv4 address** *ip-address* **Example:** `RP/0/RP0/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** or **commit** **Example:** `RP/0/RP0/CPU0:router(config-if)# end` or `RP/0/RP0/CPU0:router(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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 instance* **Example:** `RP/0/RP0/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. **ipv4 unreachables**
4. **end**
   or
   commit
5. **show interfaces** *type instance*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface null 0** **Example:** `RP/0/RP0/CPU0:router#(config)# interface null 0`	Enters the null0 interface configuration mode.
Step 3	**ipv4 unreachables** **Example:** `RP/0/RP0/CPU0:router#(config-null0)# ipv4 unreachables`	Generates IPv4 ICMP unreachables messages. • This command has no arguments or keywords.
Step 4	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router(config-null0)# end` or `RP/0/RP0/CPU0:router(config-null0)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 5	**show interfaces null 0** **Example:** `RP/0/RP0/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** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**ipv4 address virtual address** *ipv4-address/mask* **Example:** `RP/0/RP0/CPU0:router(config)# ipv4 virtual address 10.3.32.154/8`	Defines an IPv4 virtual address for the management Ethernet interface.
Step 3	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router(config-null0)# end` or `RP/0/RP0/CPU0:router(config-null0)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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

The following example indicates how to configure a loopback interface:

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface Loopback 3
RP/0/RP0/CPU0:router(config-if)# ipv4 address 172.18.189.38/32
RP/0/RP0/CPU0:router(config-if)# end
Uncommitted changes found, commit them? [yes]: yes
RP/0/RP0/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/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface Null 0
RP/0/RP0/CPU0:router(config-null0)# ipv4 unreachables
RP/0/RP0/CPU0:router(config-null0)# end
Uncommitted changes found, commit them? [yes]: yes
RP/0/RP0/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/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# ipv4 virtual address 10.3.32.154/8
RP/0/RP0/CPU0:router(config-null0)# commit
```

Additional References

The following 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,* Release 3.2
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*
Information about configuring interfaces and other components on the Cisco CRS-1 from a remote Craft Works Interface (CWI) client management application	*Cisco CRS-1 Series Carrier Routing System Craft Works Interface 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: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

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/techsupport

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

Before you can telnet into the switch 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 XR Getting 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 on Cisco IOS XR Software

Release	Modification
Release 2.0	This feature was introduced on the Cisco CRS-1.
Release 3.0	No 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.
Release 3.4.0	No modification.
Release 3.5.0	No modifications.

Contents

Prerequisites for Configuring Management Ethernet Interfaces

Before performing the Management Ethernet interface configuration procedures that are described in this chapter, 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.*
- To use the **speed**, **duplex**, **mac-address**, and **show interfaces** commands, you must be in a user group associated with a task group that includes the proper task IDs for interface commands.

  Task IDs for **speed**, **duplex**, **mac-address**, and **show interfaces** commands are listed in *Cisco IOS XR Interface and Hardware Component 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 209

Default Interface Settings

Table 12 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 12 Management Ethernet Interface Default Settings

Parameter	Default Value	Configuration File Entry
Speed in Mbps	Speed is autonegotiated.	**speed** [**10** | **100** | **1000**] To return the system to autonegotiated speed, use the **no speed** [**10** | **100** | **1000**] command.
Duplex mode	Duplex mode is autonegotiated.	**duplex** {**full** | **half**} 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* 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 210 (required)
- Configuring the Duplex Mode for a Management Ethernet Interface, page 213 (optional)
- Configuring the Speed for a Management Ethernet Interface, page 215 (optional)
- Modifying the MAC Address for a Management Ethernet Interface, page 217 (optional)
- Verifying Management Ethernet Interface Configuration, page 218 (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 must 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 *Cisco IOS XR Getting Started Guide*.

SUMMARY STEPS

1. **configure**
2. **interface MgmtEth** *instance*
3. **ipv4 address** *ip-address mask*
4. **mtu** *bytes*
5. **no shutdown**
6. **end**
   or
   commit
7. **show interfaces MgmtEth** *instance*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure terminal`	Enters global configuration mode.
Step 2	**interface MgmtEth** *instance* **Example:** `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*. The example indicates port 0 on the RP card that is installed in slot 0.

	Command or Action	Purpose
Step 3	**ipv4 address** *ip-address mask* **Example:** `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. • Replace *ip-address* with the primary IPv4 address for the interface. • Replace *mask* with the mask for the associated IP subnet. The network mask can be specified in either of two ways: – 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. – 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* **Example:** `RP/0/RP0/CPU0:router(config-if# mtu 1448`	(Optional) Sets the maximum transmission unit (MTU) value for the interface. • The default is 1514 bytes. • The range for the Management Ethernet Interface **mtu** values is 64 to 1514 bytes.
Step 5	**no shutdown** **Example:** `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** or **commit** **Example:** `RP/0/RP0/CPU0:router(config-if)# end` or `RP/0/RP0/CPU0:router(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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** *instance* **Example:** `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** *instance*
3. **duplex** [**full** | **half**]
4. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface MgmtEth** *instance* **Example:** `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	**duplex** [**full** | **half**] **Example:** `RP/0/RP0/CPU0:router(config-if)# duplex full`	Configures the interface duplex mode. Valid options are **full** or **half**. **Note** To return the system to autonegotiated duplex operation, use the **no duplex** command.
Step 4	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router(config-if)# end` or `RP/0/RP0/CPU0:router(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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** *instance*
3. **speed** {**10** | **100** | **1000**}
4. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface MgmtEth** *instance* **Example:** `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	**speed** {**10** | **100** | **1000**} **Example:** `RP/0/RP0/CPU0:router(config-if)# speed 100`	Configures the interface speed parameter. • On a Cisco CRS-1 router, valid **speed** options are **10**, **100** or **1000** Mbps. • On a Cisco XR 12000 Series Router, valid **speed** options are **10** or **100** Mbps. **Note** The default Management Ethernet interface speed is autonegotiated. **Note** To return the system to the default autonegotiated speed, use the **no speed** command.
Step 4	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router(config-if)# end` or `RP/0/RP0/CPU0:router(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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** *instance*
3. **mac-address** *address*
4. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface MgmtEth** *instance* **Example:** `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* **Example:** `RP/0/RP0/CPU0:router(config-if)# mac-address 0001.2468.ABCD`	Configures the MAC layer address of the Management Ethernet interface. **Note** To return the device to its default MAC address, use the **no mac-address** *address* command.
Step 4	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router(config-if)# end` or `RP/0/RP0/CPU0:router(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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** *instance*
2. **show running-config**

Step 1	**show interfaces MgmtEth** *instance* **Example:** `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** *instance* **Example:** `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

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

The following 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,* Release 3.2
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*
Information about configuring interfaces and other components on the Cisco CRS-1 from a remote Craft Works Interface (CWI) client management application	*Cisco CRS-1 Series Carrier Routing System Craft Works Interface 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 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: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

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/techsupport

Configuring NetFlow on Cisco IOS XR Software

A NetFlow flow is a unidirectional sequence of packets that arrive on a single interface (or subinterface), and have the same values for key fields.

NetFlow is useful for the following:

- Accounting/Billing—NetFlow data provides fine grained metering for highly flexible and detailed resource utilization accounting.
- Network Planning and Analysis—NetFlow data provides key information for strategic network planning.
- Network Monitoring—NetFlow data enables near real-time network monitoring capabilities.

Feature History for Configuring NetFlow on Cisco IOS XR Software

Release	Modification
Release 3.2	This feature was introduced on the Cisco CRS-1.
Release 3.3.0	• This feature was first supported on the Cisco XR 12000 Series Router. • NetFlow support for sub-interfaces was added on the CRS-1 platform. • Added information on using the **bgp attribute-download** command to enable the NetFlow BGP data export function. • NetFlow support for subinterfaces was introduced on the Cisco CRS-1.
Release 3.3.1	• Support for Multiprotocol Label Switching (MPLS) aware NetFlow was introduced on the Cisco CRS-1. • The **mpls** keyword was added to the **flow** command to support MPLS aware NetFlow. • The **mpls** keyword was added to the **record** command to support MPLS aware NetFlow.

Release 3.4.0	• The following commands were moved to flow exporter map configuration mode: – **destination** – **dscp** – **source** – **transport udp** • NetFlow was updated so that a single flow monitor map supports up to 8 exporters. • NetFlow was updated so that users could specify the number of MPLS labels to use as keys.
Release 3.4.1	• The record ipv4 and record mpls fields were introduced in the Cisco CRS-1 **show** command output.
Release 3.5.0	• The **record ipv6** command was introduced on the Cisco CRS-1. • The **record mpls** command was introduced on the Cisco XR 12000 Series Router. • MPLS aware NetFlow was supported on the Cisco XR 12000 Series Router.. • Support for IPv6 aware NetFlow was introduced on the Cisco CRS-1.

Contents

This chapter includes the following sections:

Prerequisites for Configuring NetFlow

Before configuring NetFlow, be sure that the following conditions are met:

- You must be in a user group associated with a task group that includes the proper task IDs for the NetFlow commands. Task IDs for commands are listed in *Cisco IOS XR Interface and Hardware Component Command Reference.*

Restrictions for Configuring NetFlow

Consider the following restrictions when configuring NetFlow in Cisco IOS XR software:

- You must configure a source interface. If you do not configure a source interface, the exporter will remain in a disabled state.
- Cisco IOS XR software supports export format Version 9 only.
- You must configure a valid record map name for every flow monitor map.

Tip We recommend that you do not use the management interface to export NetFlow packets. Exporting the management interface does not work on the Cisco XR 12000 Series Router and is not efficient on the Cisco CRS-1.

Information About Configuring NetFlow

To implement NetFlow, you must understand the following concepts:

- NetFlow Overview, page 225
- Monitor Map Overview, page 225
- Sampler Map Overview, page 226
- Exporter Map Overview, page 226
- NetFlow Configuration Submodes, page 227

NetFlow Overview

A flow is exported as part of a NetFlow export User Datagram Protocol (UDP) datagram under the following circumstances:

- The flow has been inactive or active for too long.
- The flow cache is getting full.
- One of the counters (packets and or bytes) has wrapped.

NetFlow export UDP datagrams are sent to an external flow collector device that provides NetFlow export data filtering and aggregation. The export of data consists of expired flows and control information.

The NetFlow infrastructure is based on the configuration and use of the following maps:

- Monitor map
- Sampler map
- Exporter map

These maps are described in the sections that follow.

Monitor Map Overview

A monitor map contains name references to the flow record map and flow exporter map. Monitor maps are applied to an interface. You can configure the following monitor map attributes:

- Number of entries in the flow cache

- Type of cache (permanent or normal). Permanent caches do not have their entries removed from the cache unless they are explicitly cleared by the user
- Active flow timeout
- Inactive flow timeout
- Update timeout
- Default timeouts
- Record type of packets sampled and collected.

Note On Cisco CRS-1, NetFlow supports the sampling of IPv4, IPv6, and MPLS packets. On Cisco XR 12000 Series Router, NetFlow supports the sampling of IPv4 and MPLS packets. The record name specifies the type of packets NetFlow samples as they pass through the router.

Note The active flow and inactive flow timeouts are associated with a normal cache type. The update timeout is associated with the permanent cache type.

Sampler Map Overview

The sampler map specifies the rate at which packets (one out of *n* packets) are sampled. On high bandwidth interfaces, applying NetFlow processing to every single packet can result in significant CPU utilization. Sampler map configuration is typically geared towards such high speed interfaces. If NetFlow is applied in both directions, then the flow record packets are policed at the rate of 35,000 packets per second per direction on the Cisco CRS-1 and 25,000 packets per second per direction on the Cisco XR 12000 Series Router. If NetFlow is applied in one direction only, then the flow record packets are policed at the rate of 70,000 packets per second per direction on the Cisco CRS-1 and 50,000 packets per second per direction on the Cisco XR 12000 Series Router.

Exporter Map Overview

An exporter map contains user network specification and transport layer details for the NetFlow export packet. The **flow exporter-map** command allows you to configure collector and version attributes. You can configure the following collector information:

- Export destination IP address
- DSCP value for export packet
- Source interface
- UDP port number (This is where the collector is listening for NetFlow packets.)
- Transport protocol for export packets

Note In Cisco IOS XR software, UDP is the only supported transport protocol for export packets.

Note NetFlow export packets use the IP address that is assigned to the source interface. If the source interface does not have an IP address assigned to it, the exporter will be inactive.

You can also configure the following export version attributes:

- Template timeout
- Template data timeout
- Template options timeout
- Interface table timeout
- Sampler table timeout

Note A single flow monitor map can support up to eight exporters.

NetFlow Configuration Submodes

In Cisco IOS XR, NetFlow map configuration takes place in map-specific submodes. Cisco IOS XR supports the following NetFlow map configuration submodes:

- Flow Exporter Map Configuration Submode, page 227
- Flow Exporter Map Version Configuration Submode, page 228
- Flow Monitor Map Configuration Submode, page 228
- Sampler Map Configuration Submode, page 229

Tip The Cisco IOS XR software allows you to issue most commands available under submodes as one single command string from global configuration mode. For example, you can issue the **record ipv4** command from the flow monitor map configuration submode as follows:

```
RP/0/RP0/CPU0:router(config)# flow monitor-map fmm
RP/0/RP0/CPU0:router(config-fmm)# record ipv4
```

Alternatively, you can issue the same command from global configuration mode, as shown in the following example:

```
RP/0/RP0/CPU0:router(config)# flow monitor-map fmm record ipv4
```

Flow Exporter Map Configuration Submode

When you issue the **flow exporter-map** *fem-name* command in global configuration mode, the command-line interface (CLI) prompt changes to "config-fem," indicating that you have entered the flow exporter map configuration submode.

In the following sample output, the question mark (**?**) online help function displays all the commands available under the flow exporter map configuration submode:

```
RP/0/RP0/CPU0:router(config)# flow exporter-map fem

RP/0/RP0/CPU0:router(config-fem)# ?

  commit       Commit the configuration changes to running
  describe     Describe a command without taking real actions
  destination  Export destination configuration
  do           Run an exec command
```

```
  dscp         Specify DSCP value for export packets
  exit         Exit from this submode
  no           Negate a command or set its defaults
  root         Exit to the global configuration mode
  show         Show contents of configuration
  source       Source interface
  transport    Specify the transport protocol for export packets
  version      Specify export version parameters

RP/0/RP0/CP0:router(config-fem)#
```

Note If you enter the **version** command, you enter the flow exporter map version configuration submode.

Note A single flow monitor map can support up to eight exporters.

Flow Exporter Map Version Configuration Submode

When you issue the **version v9** command in the flow exporter map configuration submode, the CLI prompt changes to "config-fem-ver," indicating that you have entered the flow exporter map version configuration submode.

In the following sample output, the question mark (**?**) online help function displays all the commands available under the flow exporter map version configuration submode:

```
RP/0/RP0/CPU0:router(config-fem)# version v9

RP/0/RP0/CPU0:router(config-fem-ver)# ?

  commit    Commit the configuration changes to running
  describe  Describe a command without taking real actions
  do        Run an exec command
  exit      Exit from this submode
  no        Negate a command or set its defaults
  options   Specify export of options template
  show      Show contents of configuration
  template  Specify template export parameters

RP/0/RP0/CPU0:router(config-fem-ver)#
```

Flow Monitor Map Configuration Submode

When you issue the **flow monitor-map** *map_name* command in global configuration mode, the CLI prompt changes to "config-fmm," indicating that you have entered the flow monitor map configuration submode.

In the following sample output, the question mark (**?**) online help function displays all the commands available under the flow monitor map configuration submode:

```
RP/0/RP0/CPU0:router(config)# flow monitor-map fmm

RP/0/RP0/CPU0:router(config-fmm)# ?

  cache     Specify flow cache attributes
  commit    Commit the configuration changes to running
  describe  Describe a command without taking real actions
  do        Run an exec command
  exit      Exit from this submode
```

```
  exporter  Specify flow exporter map name
  no        Negate a command or set its defaults
  record    Specify a flow record map name
  show      Show contents of configuration

RP/0/RP0/CP0:router(config-fmm)#
```

Sampler Map Configuration Submode

When you issue the **sampler-map** *map_name* command in global configuration mode, the CLI prompt changes to "config-sm," indicating that you have entered the sampler map configuration submode.

In the following sample output, the question mark (**?**) online help function displays all the commands available under the sampler map configuration submode:

```
RP/0/RP0/CPU0:router(config)# sampler-map fmm

RP/0/RP0/CPU0:router(config-sm)# ?

  commit    Commit the configuration changes to running
  describe  Describe a command without taking real actions
  do        Run an exec command
  exit      Exit from this submode
  no        Negate a command or set its defaults
  random    Use random mode for sampling packets
  show      Show contents of configuration

RP/0/RP0/CP0:router(config-sm)#
```

Enabling the NetFlow BGP Data Export Function

Use the **bgp attribute-download** command to enable the NetFlow BGP data export function. When attribute download is enabled, BGP downloads the attribute information for prefixes (community, extended community, and as-path) to the Routing Information Base (RIB) and Forwarding Information Base (FIB). This enables FIB to associate the prefixes with attributes and send the NetFlow statistics along with the associated attributes.

How to Configure NetFlow on Cisco IOS XR Software

The steps that follow provide a general overview of NetFlow configuration:

Step 1 Create and configure an exporter map.

Step 2 Create and configure a monitor map and a sampler map.

Note The monitor map must reference the exporter map you created in Step 1. If you do not apply an exporter-map to the monitor-map, the flow records are not exported, and aging is done according to the cache parameters specified in the monitor-map.

Step 3 Apply the monitor map and sampler map to an interface.

These steps are described in detail in the following sections:

- Configuring an Exporter Map, page 230
- Configuring a Sampler Map, page 232
- Configuring a Monitor Map, page 234
- Applying a Monitor Map and a Sampler Map to an Interface, page 237

Configuring an Exporter Map

Configure an exporter map and apply it to the monitor map with the **flow monitor-map** *map_name* **exporter** *map_name* command. You can configure the exporter map prior to configuring the monitor map, or you can configure the monitor map first and then configure and apply an exporter map later on.

Note Cisco IOS XR software supports the configuration of a single collector only in the exporter map.

The steps that follow describe how to create and configure an exporter map.

SUMMARY STEPS

1. **configure**
2. **flow exporter-map** *map_name*
3. **destination** *hostname_or_IP_address*
4. **dscp** *dscp_value*
5. **source** *type instance*
6. **transport udp** *port*
7. **exit**
8. **version v9**
9. **options** {**interface-table** | **sampler-table**} [**timeout** *seconds*]
10. **template** [**data** | **options**] **timeout** *seconds*
11. **end**
    or
    commit
12. **exit**
13. **exit**
14. **exit**
15. **show flow exporter-map** *map_name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**flow exporter-map** *map_name* **Example:** `RP/0/RP0/CPU0:router(config)# flow exporter-map fem`	Creates an exporter map, configures the exporter map name, and enters flow exporter map configuration mode.
Step 3	**destination** *hostname_or_IP_address* **Example:** `RP/0/RP0/CPU0:router(config-fem)# destination nnn.nnn.nnn.nnn`	(Optional) Configures the export destination for the flow exporter map. The destination can be a hostname or an IP address.
Step 4	**dscp** *dscp_value* **Example:** `RP/0/RP0/CPU0:router(config-fem)# dscp 55`	(Optional) Specifies the differentiated services codepoint (DSCP) value for export packets. Replace the *dscp_value* argument with a value in the range from 0 through 63.
Step 5	**source** *type instance* **Example:** `RP/0/RP0/CPU0:router(config-fem)# source pos 0/1/0/1`	(Optional) Specifies a source interface, in the format *type instance*. For example: **POS** `0/1/0/1`
Step 6	**transport udp** *port* **Example:** `RP/0/RP0/CPU0:router(config-fem)# transport udp 9991`	(Optional) Specifies the destination port for UDP packets. Replace *port* with the destination UDP port value, in the range from 1024 through 65535.
Step 7	**exit** **Example:** `RP/0/RP0/CPU0:router(config-fem)# exit`	Exits flow exporter map configuration mode, and enters flow exporter map configuration mode.
Step 8	**version v9** **Example:** `RP/0/RP0/CPU0:router(config-fem-ver)# version v9`	(Optional) Enters flow exporter map version configuration submode.
Step 9	**options** {**interface-table** | **sampler-table**} [**timeout** seconds] **Example:** `RP/0/RP0/CPU0:router(config-fem-ver)# options sampler-table timeout 2000`	(Optional) Configures the export timeout value for the sampler table. Replace *seconds* with the export timeout value, in the range from 1 through 604800 seconds. Default is 1800 seconds.

	Command or Action	Purpose
Step 10	**template** [**data** | **options**] **timeout** *seconds* **Example:** `RP/0/RP0/CPU0:router(config-fem-ver)# template data timeout 10000`	(Optional) Configures the export period for data packets. Replace *seconds* with the export timeout value, in the range from 1 through 604800 seconds.
Step 11	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router (config-fem-ver)# end` or `RP/0/RP0/CPU0:router(config-fem-ver)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 12	**exit** **Example:** `RP/0/RP0/CPU0:router(config-fem-ver)# exit`	Exits flow exporter map version configuration mode, and enters flow exporter map configuration mode.
Step 13	**exit** **Example:** `RP/0/RP0/CPU0:router(config-fem)# exit`	Exits flow exporter map configuration mode, and enters global configuration mode.
Step 14	**exit** **Example:** `RP/0/RP0/CPU0:router(config)# exit`	Exits global configuration mode, and enters EXEC mode.
Step 15	**show flow exporter-map** *map_name* **Example:** `RP/0/RP0/CPU0:router# show flow exporter-map fem`	Displays exporter map data.

Configuring a Sampler Map

The steps that follow describe how to create and configure a sampler map.

SUMMARY STEPS

1. **configure**
2. **sampler-map** *map_name*
3. **random 1 out-of** *sampling_interval*
4. **end**
   or
   commit
5. **exit**
6. **exit**
7. **show sampler-map** *map_name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**sampler-map** *map_name* **Example:** `RP/0/RP0/CPU0:router(config)# sampler-map fsm` `RP/0/RP0/CPU0:router(config-sm)#`	Creates a sampler map and enters sampler map configuration mode. Keep the following in mind when configuring a sampler map: • On the Cisco CRS-1, NetFlow supports policing at a rate of 35,000 packets per second per direction for each individual line card. • On the Cisco XR 12000 Series Router, NetFlow supports policing at a rate of 50,0000 packets per second per direction for each individual line card if SNF is enabled in one direction (ingress or egress). Note that this limit does not apply if SNF is enabled in both directions. If SNF is enabled in both directions, then NetFlow supports 25,000 packets per second per direction for each individual line card.
Step 3	**random 1 out-of** *sampling_interval* **Example:** `RP/0/RP0/CPU0:router(config-sm)# random 1 out-of 10`	Configures the sampling interval to use random mode for sampling packets. Replace the *sampling_interval* argument with a number, in the range from1 through 65535 units.

	Command or Action	Purpose
Step 4	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router (config-sm)# end` or `RP/0/RP0/CPU0:router (config-sm)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 5	**exit** **Example:** `RP/0/RP0/CPU0:router(config-sm)# exit`	Exits sampler map configuration mode and enters global configuration mode.
Step 6	**exit** **Example:** `RP/0/RP0/CPU0:router(config)# exit`	Exits global configuration mode and enters EXEC mode.
Step 7	**show sampler-map** *map_name* **Example:** `RP/0/RP0/CPU0:router# show sampler-map fsm`	Displays sampler map data.

Configuring a Monitor Map

The steps that follow describe how to create and configure a monitor map.

Note The **record ipv6** command is supported on the Cisco CRS-1 only.

SUMMARY STEPS

1. **configure**
2. **flow monitor-map** *map_name*

3. **record ipv4**
   or
   record ipv6
   or
   record mpls [**ipv4-fields**] [**labels** *number*]
4. **cache entries** *number*
5. **cache permanent**
6. **cache timeout** {**active** | **inactive** | **upate**} *timeout_value*
7. **exporter** *map_name*
8. **end**
   or
   commit
9. **exit**
10. **exit**
11. **show flow monitor-map** *map_name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**flow monitor-map** *map_name* **Example:** `RP/0/RP0/CPU0:router(config)# flow monitor-map fmm` `RP/0/RP0/CPU0:router(config-fmm)#`	Creates a monitor map and configures a monitor map name and enters flow monitor map configuration submode.
Step 3	**record ipv4** or **record ipv6** or **record mpls** [**ipv4-fields**] [**labels** *number*] **Example:** `RP/0/RP0/CPU0:router(config-fmm)# record ipv4`	Configures the flow record map name for IPv4, IPv6, or MPLS. • Use the **record ipv4** command to configure the flow record map name for IPv4. • Use the **record ipv6** command to configure the flow record map name for IPv6. • Use the **record mpls ipv4-fields** command to collect IPv4 fields in the MPLS aware NetFlow. • Use the **record mpls labels** command with the *number* argument to specify the number of labels that you want to aggregate. By default, MPLS aware NetFlow aggregates the top six labels of the MPLS label stack. The maximum value is 6. **Note** The **record ipv6** command is supported on the Cisco CRS-1 only.

	Command or Action	Purpose
Step 4	**cache entries** *number* **Example:** `RP/0/RP0/CPU0:router(config-fmm)# cache entries 10000`	(Optional) Configures the number of entries in the flow cache. Replace the *number* argument with the number of flow entries allowed in the flow cache, in the range from 4096 through 1000000. The default number of cache entries is 65535.
Step 5	**cache permanent** **Example:** `RP/0/RP0/CPU0:router(config-fmm)# flow monitor-map fmm cache permanent`	(Optional) Disables removal of entries from flow cache.
Step 6	**cache timeout** {**active** *timeout_value* | **inactive** *timeout_value* | **upate** *timeout_value*} **Example:** `RP/0/RP0/CPU0:router(config-fmm)# cache timeout inactive 1000`	(Optional) Configures the active, inactive, or update flow cache timeout value. • The default timeout value for the inactive flow cache is 15 seconds. • The default timeout value for the active flow cache is 1800 seconds. • The default timeout value for the update flow cache is 1800 seconds. **Note** The **update** *timeout_value* keyword argument is used for permanent caches only. It specifies the timeout value that is used to export entries from permanent caches. In this case, the entries are exported but remain the cache.
Step 7	**exporter** *map_name* **Example:** `RP/0/RP0/CPU0:router(config-fmm)# exporter fem`	Associates an exporter map with a monitor map. **Note** A single flow monitor map can support up to eight exporters.

	Command or Action	Purpose
Step 8	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router(config-fmm)# end` or `RP/0/RP0/CPU0:router(config-fmm)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 9	**exit** **Example:** `RP/0/RP0/CPU0:router(config-fmm)# exit`	Exits flow monitor map configuration submode.
Step 10	**exit** **Example:** `RP/0/RP0/CPU0:router(config)# exit`	Exits global configuration mode.
Step 11	**show flow monitor-map** *map_name* **Example:** `RP/0/RP0/CPU0:router# show flow monitor-map fmm`	Displays flow monitor map data.

Applying a Monitor Map and a Sampler Map to an Interface

SUMMARY STEPS

The steps that follow describe how to apply a monitor map and a sampler map to an interface.

1. **configure**
2. **interface** *type number*
3. **flow** [**ipv4** | **ipv6** | **mpls**] **monitor** *monitor_map* **sampler** *sampler_map* {**egress** | **ingress**}
4. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface** *type number* **Example:** `RP/0/RP0/CPU0:router(config)# interface POS 0/0/0/0` `RP/0/RP0/CPU0:router(config-if)#`	Enters interface configuration mode.
Step 3	**flow** [**ipv4** | **ipv6** | **mpls**] **monitor** *monitor_map* **sampler** *sampler_map* {**egress** | **ingress**} **Example:** `RP/0/RP0/CPU0:router(config-if)# flow ipv4 monitor fmm sampler fsm egress`	Associates a monitor map and a sampler map with an interface. Enter **ipv4** to enable IPV4 NetFlow on the specified interface. Enter **ipv6** to enable IPV6 NetFlow on the specified interface. Enter **mpls** to enable MPLS aware NetFlow on the specified interface. **Note** The **ipv6** keyword is supported on the Cisco CRS-1 only.
Step 4	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router(config-if)# end` or `RP/0/RP0/CPU0:router(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)?` `[cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Clearing NetFlow Data

The steps that follow describe how to clear flow exporter map and flow monitor map data.

SUMMARY STEPS

1. **clear flow exporter** [*map_name*] {**restart** | **statistics**} **location** *node-id*
2. **clear flow monitor** [*map_name*] **cache** [**force-export** | **statistics**] **location** *node-id*}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**clear flow exporter** [*exporter_name*] {**restart** | **statistics**} **location** *node-id* **Example:** `RP/0/RP0/CPU0:router# clear flow exporter statistics location 0/0/CPU0`	Clears the flow exporter data. Specify the **statistics** option to clear exporter statistics. Specify the **restart** option to export all of the templates that are currently configured on the specified node.
Step 2	**clear flow monitor** [*monitor_name*] **cache** [**force-export** | **statistics**] **location** *node-id*} **Example:** `RP/0/RP0/CPU0:router# clear flow monitor cache force-export location 0/0/CPU0`	Clears the flow monitor data. Specify the **statistics** option to clear cache statistics. Specify the **force-export** option to force an export of deleted cache entries.

Configuration Examples for CRS-1 Series NetFlow

The following example shows how to create a new sampler map called “fsm1,” which samples 1 out of 1000 packets:

```
RP/0/RP0/CPU0:router(config)# sampler-map fsm1
RP/0/RP0/CPU0:router(config-sm)# random 1 out-of 1000
RP/0/RP0/CPU0:router(config)# exit
```

The following example shows how to create a new flow exporter map called “fem1,” which uses the version 9 (V9) export format for NetFlow export packets. The data template flow-set is inserted into the V9 export packets once every 10 minutes, and the options interface table flow-set is inserted into the V9 export packet. The export packets are sent to the flow collector destination 10.1.1.1, where the source address is identical to the interface IP address of Loopback 0. The UDP destination port is 1024, and the DSCP value is 10.

```
RP/0/RP0/CPU0:router(config)# flow exporter-map fem1
RP/0/RP0/CPU0:router(config-fem)# destination 10.1.1.1
RP/0/RP0/CPU0:router(config-fem)# source Loopback 0
RP/0/RP0/CPU0:router(config-fem)# transport udp 1024
RP/0/RP0/CPU0:router(config-fem)# dscp 10
RP/0/RP0/CPU0:router(config-fem)# exit
RP/0/RP0/CPU0:router(config-fem)# version v9
RP/0/RP0/CPU0:router(config-fem-ver)# template data timeout 600
RP/0/RP0/CPU0:router(config-fem-ver)# options interface-table
RP/0/RP0/CPU0:router(config-fem-ver)# exit
```

The following example shows how to create a new flow monitor map with name “fmm1”. This flow monitor map references the flow exporter map “fem1,” and sets the flow cache attributes to 10000 cache entries. The active entries from the cache are aged every 30 seconds, while the inactive entries from the cache are aged every 15 seconds. The record map for this monitor map is IPv4.

```
RP/0/RP0/CPU0:router(config)# flow monitor-map fmm1
RP/0/RP0/CPU0:router(config-fmm)# record ipv4
RP/0/RP0/CPU0:router(config-fmm)# exporter fem1
RP/0/RP0/CPU0:router(config-fmm)# cache entries 10000
RP/0/RP0/CPU0:router(config-fmm)# cache timeout active 15
RP/0/RP0/CPU0:router(config-fmm)# cache timeout inactive 30
RP/0/RP0/CPU0:router(config-fmm)# exit
```

The following example shows how to apply the flow monitor “fmm1”and the sampler “fsm1” to the TenGigE 0/0/0/0 interface in the ingress direction.

```
RP/0/RP0/CPU0:router(config)# interface TenGigE 0/0/0/0
RP/0/RP0/CPU0:router(config-if)# flow ipv4 monitor fmm1 sampler fsm1 ingress
RP/0/RP0/CPU0:router(config-if)# exit
```

Additional References

The following sections provide references related to 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*
Information about configuring interfaces and other components on the Cisco CRS-1 from a remote Craft Works Interface (CWI) client management application	*Cisco Craft Works Interface User 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
—	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://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

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/techsupport

Configuring POS Interfaces on Cisco IOS XR Software

This module describes the configuration of Packet-over-SONET/SDH (POS) interfaces. 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 Routers support 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 2.0	This feature was introduced on the Cisco CRS-1.
Release 3.0	No modification.
Release 3.2	This feature was supported on the Cisco XR 12000 Series Router. Support for the following hardware was introduced on the Cisco CRS-1: • 1 Port OC-192c/STM-64 POS/RPR XFP SPA • 4 Port OC-3c/STM-1 POS SPA • Cisco CRS-1 SIP-800.
Release 3.3.0	Support for the 8-Port OC-12c/STM-4 POS SPA was introduced on the Cisco CRS-1. Support was added on the Cisco XR 12000 Series Router for the 2-Port OC-48 POS/RPR SPA.

Release 3.4.0	Support was added on the Cisco CRS-1 for the following hardware: • 2 port OC-48c/STM16c POS SPA • 4 port OC-48c/STM16c POS SPA Support for the following features was introduced on the Cisco XR 12000 Series Router: • Subinterfaces with permanent virtual circuits (PVCs) • Frame Relay encapsulation on POS main interfaces and PVCs on the following hardware: – 1 port 192c/STM-64c POS/SDH SPA – 2 port OC48/STM16 POS/SDH SPA – 4 port OC-3c/STM-1 POS/SDH Line Card – 8 port OC-3c/STM-1c POS/SDH Line Card – 16 port OC-3c/STM-1c POS/SDH Line Card – 4 port OC-12c/STM4 POS/SDH ISE Line Card – 1 port OC-48c/STM16c POS/SDH ISE Line Card
Release 3.4.1	Support was added on the Cisco CRS-1 for the 1-Port OC-192c/STM-64 POS/RPR VSR Optics SPA.
Release 3.5.0	Support was added on the Cisco XR 12000 Series Router for the following SPAs: • 1-Port Channelized OC3 SPA • 1-Port Channelized OC48 SPA • 1-Port Channelized OC12 SPA • 2-Port OC12 POS • 4-Port OC12 POS • 8-Port OC12 POS • 4-Port OC3 POS • 8-Port OC3 POS On the Cisco XR 12000 Series Router, L2TPv3-based L2VPN support was added on Frame Relay encapsulated POS interfaces. On the Cisco XR 12000 Series Router, the **l2transport** keyword was added to the **interface** command.

Contents

Prerequisites for Configuring POS Interfaces

Before configuring POS interfaces, be sure that the following conditions are met:

- You must be in a user group associated with a task group that includes the proper task IDs for POS commands. To perform Cisco HDLC, PPP, and Frame Relay tasks, you must be in a user group associated with a task group that includes the proper task IDs for HDLC, PPP, and Frame Relay commands.

  Task IDs for commands are listed in the *Cisco IOS XR Interface and Hardware Component Command Reference*.
- You know the IP address of the interface you will assign to the new POS interface configuration.
- You must have configured one of the following controller types:
  - a SONET controller, as described in the module earlier in this document.
  - a DWDM controller, as described in the Configuring Dense Wavelength Division Multiplexing Controllers on Cisco IOS XR Software module earlier in this document.

Note POS DWDM controller configuration is supported on the OC-768c/STM-256c DWDM PLIM only.

Information About Configuring POS Interfaces

To configure POS interfaces, you must understand the following concepts:

- Cisco HDLC Encapsulation, page 246
- PPP Encapsulation, page 246
- Keepalive Timer, page 247
- Frame Relay Encapsulation, page 248
- Default Settings for POS Interfaces, page 245

On the Cisco XR 12000 Series Router, a single POS interface carries data using PPP, Cisco HDLC, or Frame Relay encapsulation.

On the Cisco CRS-1, a single POS interface carries data using PPP or Cisco HDLC encapsulation. Frame Relay is not supported on the Cisco CRS-1.

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 instance.

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 13 are present. These default settings can be changed by configuration.

Table 13 POS Modular Services Card and PLIM Default Interface Settings

Parameter	Configuration File Entry	Default Settings
Keepalive	**keepalive** [**disable**] **no keepalive**	keepalive 10 seconds
Encapsulation	On the Cisco XR 12000 Series Router: **encapsulation** [**hdlc** | **ppp** | **frame-relay**] On the Cisco CRS-1: **encapsulation** [**hdlc** | **ppp**]	**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 247.

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 is 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 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 247.

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 affect 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).

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. Layer 3 configuration typically takes place on the subinterface.
3. Point-to-point PVCs are configured under a POS 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. Connections on the POS PVC support Frame Relay encapsulation only.
4. Layer 2 PVC ACs are configured under a POS subinterface. You cannot configure a PVC directly under a main interface. A single Layer 2 PVC AC is allowed per subinterface. Like point-to-point PVCs, Layer 2 PVC ACs 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.

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:

- 1 port 192c/STM-64c POS/SDH SPA
- 2 port OC48/STM16 POS/SDH SPA
- 4 port OC-3c/STM-1 POS/SDH Line Card
- 8 port OC-3c/STM-1c POS/SDH Line Card
- 16 port OC-3c/STM-1c POS/SDH Line Card
- 4 port OC-12c/STM4 POS/SDH ISE Line Card
- 1 port OC-48c/STM16c POS/SDH ISE Line Card

Note Frame Relay encapsulation is supported on the Cisco XR 12000 Series Router only.

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 Cisco IOS XR Software" chapter 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 number of PVCs supported on POS PVCs with **cisco** LMI is 557.
For non-Cisco LMI types, up to 992 PVCs are supported under a single main interface.

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, one end mst be DTE and other end must be DCE for LMI to come up. For more information about configuring the LMI interface type on Frame Relay interface, see the "Modifying the Default Frame Relay Configuration on Cisco IOS XR Software"module later in this manual.

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 ACs are connected together.

L2TPv3 connection setup requires the following tasks:.

- configuring an attachment circuit (AC) on each Provider Edge (PE) router
- configuring an L2TPv3 encapsulated pseudowire between two PE routers.

This chapter 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 *Cisco IOS XR Multiprotocol Label Switching Configuration Guide.*

Note In Release 3.5.0, Frame Relay interfaces support DLCI mode layer 2 ACs only; Layer 2 port mode ACs are not supported on Frame Relay interfaces.

How to Configure a POS Interface

This section contains the following procedures:

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** *instance*
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** *instance*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**show interfaces** **Example:** `RP/0/0/CPU0:router# show interfaces`	(Optional) Displays configured interfaces. • Use this command to also confirm that the router recognizes the PLIM card.
Step 2	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 3	**interface pos** *instance* **Example:** `RP/0/0/CPU0:router(config)# interface POS 0/1/0/0`	Specifies the POS interface name and notation *rack/slot/module/port,* and enters interface configuration mode.

	Command or Action	Purpose
Step 4	**ipv4 address** *ipv4_address/prefix* **Example:** `RP/0/0/CPU0:router (config)#ipv4 address 10.46.8.6/24`	Assigns an IP address and subnet mask to the interface. **Caution** 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 PVC.
Step 5	**no shutdown** **Example:** `RP/0/0/CPU0:router (config-if)# no shutdown`	Removes the shutdown configuration. **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** or **commit** **Example:** `RP/0/0/CPU0:router (config-if)# end` or `RP/0/0/CPU0:router(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 7	**exit** **Example:** `RP/0/0/CPU0:router (config-if)# exit`	Exits interface configuration mode and enters global configuration mode.
Step 8	**exit** **Example:** `RP/0/0/CPU0:router (config)# exit`	Exits global configuration mode and enters EXEC mode.

	Command or Action	Purpose
Step 9	**show interfaces** **configure** **interface pos** *instance* **no shut** **exit** **exit** **commit** **Example:** `RP/0/0/CPU0:router# show interfaces` `RP/0/0/CPU0:router# configure` `RP/0/0/CPU0:router (config)# interface pos 0/1/0/0` `RP/0/0/CPU0:router (config-if)# no shutdown` `RP/0/0/CPU0:router (config-if)# commit` `RP/0/0/CPU0:router (config-if)# exit` `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. **Note** The configuration on both ends of the POS connection must match.
Step 10	**show ipv4 interface brief** **Example:** `RP/0/0/CPU0:router # show ipv4 interface brief`	Verifies that the interface is active and properly configured. 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** *instance* **Example:** `RP/0/0/CPU0:router# show interfaces pos 0/6/0/1`	(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 253.

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 250.

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** *instance*
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** [*instance*]

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface pos** *instance* **Example:** `RP/0/RP0/CPU0:router(config)# interface POS 0/1/0/0`	Specifies the POS interface name and notation *rack/slot/module/port,* and enters interface configuration mode.
Step 3	**encapsulation** [**hdlc** | **ppp** | **frame-relay** [**IETF**]] **Example:** `RP/0/RP0/CPU0:router(config-if)# encapsulation hdlc`	(Optional) Configures the interface encapsulation parameters and details such as HDLC or PPP. **Note** The default encapsulation is **hdlc**. **Note** The **frame-relay** option is available on the Cisco XR 12000 Series Router only.
Step 4	**pos crc** {**16** | **32**} **Example:** `RP/0/RP0/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. **Note** The default CRC is **32**.
Step 5	**mtu** *value* **Example:** `RP/0/RP0/CPU0:router(config-if)# mtu 4474`	(Optional) Configures the MTU value. • The default value is 4474. • The POS MTU range is 64–9216 on the Cisco CRS-1.

	Command or Action	Purpose
Step 6	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router (config-if)# end` or `RP/0/RP0/CPU0:router(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 7	**exit** **Example:** `RP/0/RP0/CPU0:router (config-if)# exit`	Exits interface configuration mode and enters global configuration mode.
Step 8	**exit** **Example:** `RP/0/RP0/CPU0:router (config)# exit`	Exits global configuration mode and enters EXEC mode.
Step 9	**show interfaces pos** [*instance*] **Example:** `RP/0/RP0/CPU0:router# show interface pos 0/1/0/0`	(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 256.
- 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 260.
- 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 Cisco IOS XR Software" module later 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 Subinterfaces with PVCs are supported on the Cisco XR 12000 Series Router only.

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 250.

Restrictions

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

SUMMARY STEPS

1. **configure**
2. **interface pos** *instance.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** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface pos** *instance.subinterface* **point-to-point** **Example:** `RP/0/0/CPU0:router (config)# interface pos 0/6/0/1.10 point-to-point`	Enters POS subinterface configuration mode. Replace *subinterface* with a subinterface ID, in the range from 1 through 4294967295.

	Command or Action	Purpose
Step 3	**ipv4 address** *ipv4_address/prefix* **Example:** `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* **Example:** `RP/0/0/CPU0:router (config-subif)# pvc 20`	Creates a POS permanent virtual circuit (PVC) and enters Frame Relay PVC configuration submode. Replace *dlci* with a PVC identifier, in the range from 16 to 1007. **Note** Only one PVC is allowed per subinterface.
Step 5	**end** or **commit** **Example:** `RP/0/0/CPU0:router (config-fr-vc)# end` or `RP/0/0/CPU0:router(config-fr-vc)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 6	**configure** **interface pos** *instance.subinterface* **pvc** *dlci* **commit** **Example:** `RP/0/0/CPU0:router# configure` `RP/0/0/CPU0:router (config)# interface pos 0/6/0/1.10` `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)# commit`	Repeat Step 1 through Step 5 to bring up the POS subinterface and any associated PVC at the other end of the connection. **Note** The DLCI (or PVC identifier) must match on both ends of the subinterface connection.

What to do Next

- To configure optional PVC parameters, see the "Configuring Optional PVC Parameters" section on page 258.
- 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 Cisco IOS XR Software" module later in this manual.
- 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 256.

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** *instance.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*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface pos** *instance.subinterface* **Example:** `RP/0/0/CPU0:router (config)# interface pos 0/6/0/1.10`	Enters POS subinterface configuration mode.
Step 3	**pvc** *dlci* **Example:** `RP/0/0/CPU0:router (config-subif)# pvc 20`	Enters subinterface configuration mode for the PVC. Replace *dlci* with the DLCI number used to identify the PVC. Range is from 16 to 1007.
Step 4	**encap** [**cisco** | **ietf**] **Example:** `RP/0/0/CPU0:router (config-subif)# encap ietf`	(Optional) Configures the encapsulation for a Frame Relay PVC. **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* **Example:** `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. **Note** For information on creating and configuring policy maps, refer to the *Cisco IOS XR Modular Quality of Service Configuration Guide*,
Step 6	**end** or **commit** **Example:** `RP/0/0/CPU0:router (config-subif)# end` or `RP/0/0/CPU0:router(config-subif)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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** **interface pos** *instance.subinterface* **pvc** *dlci* **encap** [**cisco** | **ietf**] **commit** **Example:** `RP/0/0/CPU0:router# configure` `RP/0/0/CPU0:router (config)# interface pos 0/6/0/1.10` `RP/0/0/CPU0:router (config-subif)# pvc dlci` `RP/0/0/CPU0:router (config-fr-vc)# encap cisco` `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. **Note** The configuration on both ends of the subinterface connection must match.
Step 8	**show frame-relay pvc** *dlci-number* **Example:** `RP/0/RP0/CPU0:router# show frame-relay pvc 20`	(Optional) Verifies the configuration of specified POS interface.

What to do Next

- 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.
- 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 Cisco IOS XR Software" module later in this manual.

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 253.

Restrictions

During MDR, the keepalive interval must be 10 seconds or more.

SUMMARY STEPS

1. **configure**
2. **interface pos** *instance*
3. **keepalive** {*seconds* | **disable**}
   or
   no keepalive
4. **end**
   or
   commit
5. **show interfaces** *type instance*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface pos** *instance* **Example:** `RP/0/RP0/CPU0:router(config)# interface POS 0/1/0/0`	Specifies the POS interface name and notation *rack/slot/module/port* and enters interface configuration mode.
Step 3	**keepalive** {*seconds* | **disable**} or **no keepalive** **Example:** `RP/0/RP0/CPU0:router(config-if)# keepalive 3` or `RP/0/RP0/CPU0:router(config-if)# no keepalive`	Specifies the number of seconds between keepalive messages. • Use the **keepalive disable** command, the **no keepalive**, or the **keepalive** command with an argument of 0 to disable the keepalive feature entirely. • 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** or **commit** **Example:** `RP/0/RP0/CPU0:router(config-if)# end` or `RP/0/RP0/CPU0:router(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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** *instance* **Example:** `RP/0/RP0/CPU0:router# show interfaces POS 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 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 At present, 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 250.

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.
- Layer 2 configuration is available on the Cisco XR 12000 Series Router router only; Layer 2 configuration is not available on the CRS-1 Series.

SUMMARY STEPS

1. **configure**
2. **interface pos** *instance.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

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface pos** *instance.subinterface* **l2transport** **Example:** `RP/0/0/CPU0:router(config)# interface pos 0/6/0/1.10 l2transport`	Creates a subinterface and enters POS subinterface configuration mode for that subinterface. **Note** The *subinterface* must be unique to any other subinterfaces configured under a single main interface.

	Command or Action	Purpose
Step 3	**pvc** dlci **Example:** `RP/0/0/CPU0:router(config-if)# pvc 100`	Creates a Frame Relay permanent virtual circuit (PVC) and enters Layer 2 transport PVC configuration mode. Replace *dlci* with the DLCI number used to identify the PVC. Range is from 16 to 1007. **Note** Only one PVC is allowed per subinterface.
Step 4	**end** or **commit** **Example:** `RP/0/0/CPU0:router(config-fr-vc)# end` or `RP/0/0/CPU0:router(config-fr-vc)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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 subinterface and any associated PVC at the other end of the AC.	Brings up the AC. **Note** The configuration on both ends of the AC must match.

What to do Next

- To configure optional subinterface parameters, see the "Configuring Optional Layer 2 Subinterface Parameters" section on page 266.
- To configure optional PVC parameters, see the "Configuring Optional Layer 2 PVC Parameters" section on page 264.
- To configure a point-to-point pseudowire XConnect on the AC you just created, see the "Layer 2 Tunnel Protocol Version 3 on Cisco IOS XR Software" module of *Cisco IOS XR Multiprotocol Label Switching Configuration Guide*.

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 262.

Restrictions

QOS is not supported on Layer 2 subinterfaces.

SUMMARY STEPS

1. **configure**
2. **interface pos** *instance.subinterface* **l2transport**
3. **pvc** *dlci*
4. **encap** [**cisco** | **ietf**]
5. **end**
   or
   commit
6. Repeat Step 1 through Step 5 to configure the PVC at the other end of the AC.

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface pos** *instance.subinterface* **l2transport** **Example:** `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* **Example:** `RP/0/0/CPU0:router(config-if)# pvc 100`	Enters Frame Relay PVC configuration mode for the specified PVC. Replace *dlci* with the DLCI number used to identify the PVC. Range is from 16 to 1007.
Step 4	**encap** {**cisco** | **ietf**} **Example:** `RP/0/0/CPU0:router(config-fr-vc)# encap ietf`	Configures the encapsulation for a Frame Relay PVC. The encapsulation type must match on both ends of the PVC.

	Command or Action	Purpose
Step 5	**end** or **commit** **Example:** `RP/0/0/CPU0:router(config-pos-l2transport-pvc)# end` or `RP/0/0/CPU0:router(config-pos-l2transport-pvc)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. 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 configure the PVC at the other end of the AC.	Brings up the AC. **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 AC you just created, see the "Layer 2 Tunnel Protocol Version 3 on Cisco IOS XR Software" module of *Cisco IOS XR Multiprotocol Label Switching Configuration Guide*.

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 262.

Restrictions

- QOS is not supported on Layer 2 subinterfaces.
- 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 will display 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** *instance.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** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface pos** *instance.subinterface* **Example:** `RP/0/0/CPU0:router(config)# interface pos 0/6/0/1.10`	Enters POS subinterface configuration mode for a Layer 2 Frame Relay subinterface.
Step 3	**mtu** *value* **Example:** `RP/0/RP0/CPU0:router(config-if)# mtu 5000`	(Optional) Configures the MTU value. Range is from 64 through 65535.

	Command or Action	Purpose
Step 4	**end** or **commit** **Example:** `RP/0/0/CPU0:router(config-pos-l2transport-pvc)# end` or `RP/0/0/CPU0:router(config-pos-l2transport-pvc)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. 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. **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

The following example shows how to bring up a basic POS interface with Cisco HDLC encapsulation:

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface POS 0/3/0/0
RP/0/RP0/CPU0:router(config-if)# ipv4 address 172.18.189.38 255.255.255.224
RP/0/RP0/CPU0:router(config-if)# no shutdown
RP/0/RP0/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/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface POS 0/3/0/0
RP/0/RP0/CPU0:router(config-if)# keepalive 10
RP/0/RP0/CPU0:router(config-if)# commit
```

Configuring a POS Interface with Frame Relay Encapsulation: Example

The following example shows how to bring up a POS interface with Frame Relay 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)# 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
```

The following example shows how create a point-to-point POS subinterface with a PVC on the main POS interface:

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router (config)# interface pos 0/3/0/1.10 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-subif)# commit

RP/0/0/CPU0:router# show interfaces pos 0/3/0/1

POS0/1/0/0 is up, line protocol is up

  Hardware is Packet over SONET/SDH

  Internet address is Unknown

  MTU 4474 bytes, BW 622080 Kbit
```

```
   reliability 255/255, txload 1/255, rxload 1/255
Encapsulation FRAME-RELAY, crc 32, controller loopback not set,
LMI enq sent  0, LMI stat recvd 0, LMI upd recvd 0
LMI enq recvd 9463, LMI stat sent  9463, LMI upd sent  0 , DCE LMI up
LMI DLCI 0  LMI type is ANSI Annex D  frame relay DCE
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
   20934 packets input, 1508069 bytes, 1151 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
   1151 input errors, 1058 CRC, 0 frame, 0 overrun, 93 ignored, 0 abort
   19590 packets output, 990924 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
```

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 0/3/0/1.10 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/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface POS 0/3/0/0
RP/0/RP0/CPU0:router(config-if)# ipv4 address 172.18.189.38 255.255.255.224
RP/0/RP0/CPU0:router(config-if)# encapsulation ppp
RP/0/RP0/CPU0:router(config-if)# no shutdown
RP/0/RP0/CPU0:router(config-if)# end
Uncommitted changes found, commit them? [yes]: yes

RP/0/RP0/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

The following 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,* Release 3.2
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*
Information about configuring interfaces and other components on the Cisco CRS-1 from a remote Craft Works Interface (CWI) client management application	*Cisco CRS-1 Series Carrier Routing System Craft Works Interface Configuration Guide*

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://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

RFCs	Title
RFC 1294	*Multiprotocol Interconnect Over Frame Relay*
RFC 1490	*Multiprotocol Interconnect Over Frame Relay*
RFC 2427	*Multiprotocol Interconnect Over Frame Relay*
RFC 1586	*Guidelines for Running OSPF Over Frame Relay Networks*
RFC 1315	*Management Information Base for Frame Relay DTEs*
RFC 2115	*Management Information Base for Frame Relay DTEs Using SMIv2*
RFC 1604	*Definitions of Managed Objects for Frame Relay Service*
RFC 2954	*Definitions of Managed Objects for Frame Relay Service*
RFC 2390	*Inverse Address Resolution Protocol*

Technical Assistance

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

Configuring PPP on Cisco IOS XR Software

This module describes how to perform the following Point-to-Point Protocol (PPP) related tasks on POS and serial interfaces in Cisco IOS XR software:

- Enable and configure PPP authentication protocols
- Disable PPP authentication
- Modify optional PPP timeout and retry parameters
- Configure Multilink PPP (MLPPP)

Feature History for Configuring PPP Interfaces

Release	Modification
Release 2.0	PPP authentication was introduced on the Cisco CRS-1.
Release 3.0	No modification.
Release 3.3.0	Support for serial interfaces with PPP encapsulation was introduced on the Cisco XR 12000 Series Router.
Release 3.4.0	No modification.
Release 3.4.1	Support for Multilink PPP was introduced on the Cisco XR 12000 Series Router.
Release 3.5.0	

Contents

Prerequisites for Configuring PPP Authentication

Before you can configure PPP authentication on a POS or serial interface, be sure 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 for serial Interface commands. Task IDs for commands are listed in *Cisco IOS XR Interface and Hardware Component Command Reference*.
- 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 chapter:
  - 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 Software" module in this manual.

Information About 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 affect 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.

How to Configure PPP Authentication

This section contains the following procedures:

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 chapters:

- 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 Software" module in this manual.

SUMMARY STEPS

1. **configure**
2. **interface** *type instance*
3. **ppp authentication** *protocol* [*protocol* [*protocol*]] [*list-name* | **default**]
4. **end**
   or
   commit
5. **show ppp interfaces** {*type interface_instance* | **all** | **brief** {*type interface_instance* | **all** | **location** *node-id*} | **detail** {*type interface_instance* | **all** | **location** *node-id*} | **location** *node-id*}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.

	Command or Action	Purpose
Step 2	**interface** *type instance* **Example:** `RP/0/RP0/CPU0:router(config)# interface serial 0/4/0/1`	Enters the interface configuration mode.
Step 3	**ppp authentication** *protocol* [*protocol* [*protocol*]] [*list-name* | **default**] `Example:` `RP/0/RP0/CPU0:router(config-if)# ppp authentication chap pap MIS-access`	(Optional) 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. • Replace the *protocol* argument with **pap**, **chap**, **or ms-chap**. • 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" chapter of the *Cisco IOS XR System Security Command Reference*. • 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" chapter of the *Cisco IOS XR System Security Command Reference*.

	Command or Action	Purpose
Step 4	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router(config-if)# end` or `RP/0/RP0/CPU0:router(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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_instance* | **all** | **brief** {*type interface_instance* | **all** | **location** *node-id*} | **detail** {*type interface_instance* | **all** | **location** *node-id*} | **location** *node-id*} **Example:** `RP/0/RP0/CPU0:router# show ppp interfaces serial 0/2/0/0`	Displays PPP state information for an interface. • Enter the *type interface_instance* argument to display PPP information for a specific interface. • 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. • Enter the **all** keyword to display detailed PPP information for all nodes installed in the router. • Enter the **location** *node-id* keyword argument to display detailed PPP information for the designated node. 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 281.
- If you enabled CHAP on an interface, configure a CHAP authentication password, as described in the "Configuring a CHAP Authentication Password" section on page 283
- 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 285

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 278.

SUMMARY STEPS

1. **configure**
2. **interface** *type instance*
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` **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.

	Command or Action	Purpose
Step 2	**interface** *type instance* **Example:** `RP/0/RP0/CPU0:router(config)# interface serial 0/4/0/1`	Enters the interface configuration mode.
Step 3	**ppp pap sent-username** *username* **password** [**clear** | **encrypted**] *password* **Example:** `RP/0/RP0/CPU0:router(config-if)# ppp pap sent-username xxxx password notified`	(Optional) 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. • Replace the *username* argument with the username sent in the PAP authentication request. • Enter **password 0** to select cleartext encryption for the password, or enter **password 7** if the password is already encrypted. • 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. • You must configure the **ppp pap sent-username** command for each interface. • Remote PAP support is disabled by default.
Step 4	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router(config-if)# end` or `RP/0/RP0/CPU0:router(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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** **Example:** `RP/0/RP0/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 278.

Restrictions

The same CHAP password must be configured on both host endpoints.

SUMMARY STEPS

1. **configure**
2. **interface** *type instance*
3. **ppp chap password** [**0** | **7**] *password*
4. **end**
   or
   commit
5. **show running-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	`configure` **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.

	Command or Action	Purpose
Step 2	**interface** *type instance* **Example:** `RP/0/RP0/CPU0:router(config)# interface serial 0/4/0/1`	Enters the interface configuration mode.
Step 3	**ppp chap password** [**0** | **7**] *password* **Example:** `RP/0/RP0/CPU0:router(config-if)# ppp chap password 0 xxxx`	(Optional) Enables CHAP authentication on the specified interface, and defines an interface-specific CHAP password. • Enter **0** to select cleartext encryption, or **7** if the password is already encrypted. • Replace the *password* argument with a cleartext or already-encrypted password. This password is used to authenticate secure communications among a collection of routers. • The **ppp chap password** command is used for remote CHAP authentication only (when routers authenticate to the peer) and does not affect 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). • The CHAP secret password is used by the routers in response to challenges from an unknown peer.
Step 4	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router(config-if)# end` or `RP/0/RP0/CPU0:router(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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** **Example:** `RP/0/RP0/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 278.

Restrictions

The same MS-CHAP password must be configured on both host endpoints.

SUMMARY STEPS

1. **configure**
2. **interface** *type instance*
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** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface** *type instance* **Example:** `RP/0/RP0/CPU0:router(config)# interface serial 0/4/0/1`	Enters the interface configuration mode.
Step 3	**ppp ms-chap password** [**clear** | **encrypted**] *password* **Example:** `RP/0/RP0/CPU0:router(config-if)# ppp ms-chap password clear xxxx`	(Optional) Enables a router calling a collection of routers to configure a common Microsoft Challenge Handshake Authentication (MS-CHAP) secret password. • 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** or **commit** **Example:** `RP/0/RP0/CPU0:router(config-if)# end` or `RP/0/RP0/CPU0:router(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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** **Example:** `RP/0/RP0/CPU0:router# show running-config`	Verifies PPP authentication information for interfaces that have PPP encapsulation enabled.

How to Modify 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 Software" module in this manual.

SUMMARY STEPS

1. **configure**
2. **interface** *type instance*
3. **ppp max-bad-auth** *retries*
4. **ppp max-configure** *retries*
5. **ppp max-failure** *retries*
6. **ppp max-terminate** *number*
7. **ppp pap refuse**
8. **ppp pap sent-username** *username* **password** [**clear** | **encrypted**] *password*
9. **ppp timeout authentication** *seconds*
10. **ppp timeout retry** *seconds*
11. **end**
    or
    commit
12. **show ppp interfaces** {*type interface_instance* | **all** | **brief** {*type interface_instance* | **all** | **location** *node-id*} | **detail** {*type interface_instance* | **all** | **location** *node-id*} | **location** *node-id*}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.

	Command or Action	Purpose
Step 2	**interface** *type instance* **Example:** `RP/0/RP0/CPU0:router(config)# interface serial 0/4/0/1`	Enters the interface configuration mode.
Step 3	**ppp max-bad-auth** *retries* **Example:** `RP/0/RP0/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. • If you do not specify the number of authentication retries allowed, the router resets itself immediately after an authentication failure. • Replace the *retries* argument with number of retries after which the interface is to reset itself, in the range from 0 through 10. • The default is 0 retries. • The **ppp max-bad-auth** command applies to any interface on which PPP encapsulation is enabled.
Step 4	**ppp max-configure** *retries* **Example:** `RP/0/RP0/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. • Replace the *retries* argument with the maximum number of configure requests retries, in the range from 4 through 20. • The default maximum number of configure requests is 10. • 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* **Example:** `RP/0/RP0/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. • Replace the *retries* argument with the maximum number of CONFNAKs to permit before terminating a negotiation, in the range from 2 through 10. • The default maximum number of CONFNAKs is 5.
Step 6	**ppp max-terminate** *number* **Example:** `RP/0/RP0/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. • 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. • The default maximum number of TermReqs is 2.

	Command or Action	Purpose
Step 7	**ppp pap refuse** **Example:** `RP/0/RP0/CPU0:router(config-if)# ppp pap refuse`	(Optional) Refuses MS-CHAP authentication from peers requesting it.
Step 8	**ppp pap sent-username** *username password* [**clear** | **encrypted**] *password* **Example:** `RP/0/RP0/CPU0:router(config-if)# ppp pap sent-username xxxx password notified`	(Optional) Enables remote PAP support for an interface, and uses the values specified for username and password in the PAP authentication request.
Step 9	**ppp timeout authentication** *seconds* **Example:** `RP/0/RP0/CPU0:router(config-if)# ppp timeout authentication 20`	(Optional) Sets PPP authentication timeout parameters. • 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. • 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 10	**ppp timeout retry** *seconds* **Example:** `RP/0/RP0/CPU0:router(config-if)# ppp timeout retry 8`	(Optional) Sets PPP timeout retry parameters. • 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. • The default is 3 seconds.

	Command or Action	Purpose
Step 11	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router(config-if)# end` or `RP/0/RP0/CPU0:router(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 12	**show ppp interfaces** {*type interface_instance* | **all** | **brief** {*type interface_instance* | **all** | **location** *node-id*} | **detail** {*type interface_instance* | **all** | **location** *node-id*} | **location** *node-id*} **Example:** `RP/0/RP0/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.

How to Disable an Authentication Protocol

This section contains the following procedures:

- Disabling PAP Authentication on an Interface, page 290
- Disabling CHAP Authentication on an Interface, page 292
- Disabling MS-CHAP Authentication on an Interface, page 294

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 instance*
3. **ppp pap refuse**

4. **end**
   or
   commit
5. **show running-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface** *type instance* **Example:** `RP/0/RP0/CPU0:router(config)# interface serial 0/4/0/1`	Enters the interface configuration mode.
Step 3	**ppp pap refuse** **Example:** `RP/0/RP0/CPU0:router(config-if)# ppp pap refuse`	(Optional) Refuses Password Authentication Protocol (PAP) authentication from peers requesting it. • 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. • PAP authentication is disabled by default.

	Command or Action	Purpose
Step 4	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router(config-if)# end` or `RP/0/RP0/CPU0:router(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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** **Example:** `RP/0/RP0/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 instance*
3. **ppp chap refuse**
4. **end**
   or
   commit
5. **show running-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface** *type instance* **Example:** `RP/0/RP0/CPU0:router(config)# interface serial 0/4/0/1`	Enters the interface configuration mode.
Step 3	**ppp chap refuse** **Example:** `RP/0/RP0/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. • CHAP authentication is disabled by default. • 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** or **commit** **Example:** `RP/0/RP0/CPU0:router(config-if)# end` or `RP/0/RP0/CPU0:router(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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** **Example:** `RP/0/RP0/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 instance*
3. **ppp ms-chap refuse**
4. **end**
   or
   commit
5. **show running-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface** *type instance* **Example:** `RP/0/RP0/CPU0:router(config)# interface serial 0/4/0/1`	Enters the interface configuration mode.
Step 3	**ppp ms-chap refuse** **Example:** `RP/0/RP0/CPU0:router(config-if)# ppp ms-chap refuse`	(Optional) 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. • MS-CHAP authentication is disabled by default. • 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** or **commit** **Example:** `RP/0/RP0/CPU0:router(config-if)# end` or `RP/0/RP0/CPU0:router(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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** **Example:** `RP/0/RP0/CPU0:router# show running-config`	Verifies PPP authentication information for interfaces that have PPP encapsulation enabled.

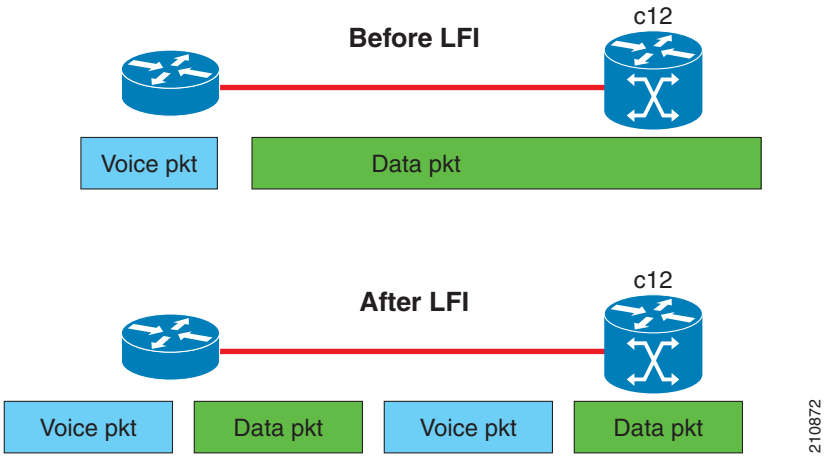
Information about Multilink PPP

Multilink Point-to-Point Protocol (MLPPP) provides a method for combining multiple physical links into one logical link. The implementation on Cisco IOS XR on Cisco XR 12000 Series Routers 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 that run at less than 768 Kbps.

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 that run at less than 768 Kbps. LFI reduces delay and jitter by fragmenting large datagrams and interleaving them with low-delay traffic packets.

Figure 8 Link Fragmentation Interleave



Supported Cards

MLPPP is supported on the following line cards and SPAs:

- Cisco XR 12000 multiservice line cards
- 2-Port and 4-Port Channelized T3 SPAs (SPA-2XCT3/DS0, SPA-4XCT3/DS0)

LFI is supported on:

- Cisco 1-Port Channelized STM-1/OC-3 Shared Port Adapter

Feature Summary

MLPPP in Cisco IOS XR Release 3.4.1 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

Limitations

MLPPP for Cisco IOS XR software has the following limitations:

- Limited QoS support: Dynamic adjustments are not possible due to the bandwidth changes of a multilink interface
- 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.
- Maximum of 12 links per bundle.
- Maximum of 28 bundles on the 2-Port Channelized T3 SPA.
- Maximum of 56 bundles on the 4-Port Channelized T3 SPA.
- Maximum of 224 bundles per line card.

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 Multilink Controller makes the multilink interfaces behave like the serial interfaces of channelized SPAs.

How to Configure Multilink PPP

This section contains the following procedures:

- Configuring the Controller, page 297
- Configuring the Interfaces, page 299
- Configuring MLPPP Optional Features, page 301

Configuring the Controller

Perform this task to configure the controller.

SUMMARY STEPS

1. **configure**
2. **controller** *type instance*
3. **mode** *type*
4. **clock source** {**internal** | **line**}
5. **exit**
6. **controller t1** *instance*
7. **channel-group** *channel-group-number*
8. **timeslots** *range*
9. **exit**
10. **exit**
11. **controller mgmtmultilink** *instance*
12. **bundle** *bundle-id*
13. **end**
    or
    commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**controller** *type instance* **Example:** `RP/0/0/CPU0:router(config)# controller t3 0/1/0/0`	Enters the controller configuration submode and specifies the controller name and instance identifier in *rack/slot/module/port* notation.
Step 3	**mode** *type* **Example:** `RP/0/0/CPU0:router# mode t1`	Configures the type of multilinks to channelize; for example, 28 T1s.
Step 4	**clock source** {**internal** | **line**} **Example:** `RP/0/0/CPU0:router(config-t3)# clock source internal`	(Optional) Configures the clocking for the port. **Note** The default clock source is **internal**.
Step 5	**exit** **Example:** `RP/0/0/CPU0:router(config-t3)# exit`	Exits controller configuration mode.
Step 6	**controller t1** *instance* **Example:** `RP/0/0/CPU0:router(config)# controller t1 0/1/0/0/0`	Enters T1 configuration mode.
Step 7	**channel-group** *channel-group-number* **Example:** `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* **Example:** `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. • Range is from 1 to 24 time slots. • You can assign all 24 time slots to a single channel group, or you can divide the time slots among several channel groups. **Note** The time slot range must be from 1 to 24 for the resulting serial interface to be accepted into a MLPPP bundle.

	Command or Action	Purpose
Step 9	**exit** **Example:** `RP/0/0/CPU0:router(config-t1-channel_group)# exit`	Exits channel group configuration mode.
Step 10	**exit** **Example:** `RP/0/0/CPU0:router(config-t1)# exit`	Exits T1 configuration mode and enters global configuration mode.
Step 11	**controller mgmtmultilink** *instance* **Example:** `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. **Note** For Cisco IOS XR Release 3.4.1, the value of the *port* argument must always be zero (0).
Step 12	**bundle** *bundle-id* **Example:** `RP/0/0/CPU0:router(config-mgmtmultilink)# bundle 20`	Creates a multililnk interface with the specified bundle ID.
Step 13	**end** or **commit** **Example:** `RP/0/0/CPU0:router(config-t3)# end` or `RP/0/0/CPU0:router(config-t3)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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.

SUMMARY STEPS

1. **configure**
2. **interface multilink** *instance*
3. **ipv4 address** *address/mask*
4. **multilink fragment-size** *size*
5. **keepalive** {*interval* | **disable**}
6. **exit**
7. **interface** *type instance*
8. **encapsulation** *type*
9. **multilink group** *group-id*
10. **end**
    or
    commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface multilink** *instance* **Example:** `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	**ipv4 address** *ip-address* **Example:** `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: `A.B.C.D/prefix or A.B.C.D/mask`
Step 4	**multilink fragment-size** *size* **Example:** `RP/0/0/CPU0:router(config-if)# multilink fragment-size 128`	(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** {*seconds* | **disable**} **Example:** `RP/0/0/CPU0:router(config-if)# keepalive disable`	Enables keepalive messages by specifying the number of seconds between keepalive messages in the range of 1 to 30. The default value is 10. Use the **disable** keyword to disable keepalives.

	Command or Action	Purpose
Step 6	**exit** **Example:** `RP/0/0/CPU0:router(config-if)# exit`	Exits interface configuration mode and enters global configuration mode.
Step 7	**interface** *type instance* **Example:** `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* **Example:** `RP/0/0/CPU0:router(config-if)# encapsulation ppp`	Specifies the type of encapsulation; in this case, PPP. **Note** PPP is supported only in Cisco IOS XR Release 3.4.1 and later releases.
Step 9	**multilink group** *group-id* **Example:** `RP/0/0/CPU0:router(config-if)# group 1`	Specifies the multilink group ID for this interface.
Step 10	**end** or **commit** **Example:** `RP/0/0/CPU0:router(config-t3)# end` or `RP/0/0/CPU0:router(config-t3)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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** *instance*
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** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface multilink** *instance* **Example:** `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** **Example:** `RP/0/0/CPU0:router(config-if)# multilink`	Enters interface multilink configuration mode.
Step 4	**ppp multilink minimum-active links** *value* Example: `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.
Step 5	**multilink interleave** Example: `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** **Example:** `RP/0/0/CPU0:router(config-if-mutlilink)# no shutdown`	Removes the shutdown configuration. • 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** or **commit** **Example:** `RP/0/0/CPU0:router(config-t3)# end` or `RP/0/0/CPU0:router(config-t3)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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 303
- Configuring a Serial Interface with PPP Encapsulation: Example, page 304
- Verifying Multilink PPP Configurations, page 305

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/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface POS 0/3/0/0
RP/0/RP0/CPU0:router(config-if)# ipv4 address 172.18.189.38 255.255.255.224
RP/0/RP0/CPU0:router(config-if)# encapsulation ppp
RP/0/RP0/CPU0:router(config-if)# no shutdown
RP/0/RP0/CPU0:router(config-if)# ppp pap sent-username P1_CRS-8 password xxxx
RP/0/RP0/CPU0:router(config-if)# ppp authentication chap pap MIS-access
RP/0/RP0/CPU0:router(config-if)# ppp chap password encrypted xxxx
RP/0/RP0/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/RP0/CPU0:router# configuration
RP/0/RP0/CPU0:router(config)# interface POS 0/3/0/1
RP/0/RP0/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/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface serial 0/3/0/0/0:0
RP/0/RP0/CPU0:router(config-if)# ipv4 address 172.18.189.38 255.255.255.224
RP/0/RP0/CPU0:router(config-if)# encapsulation ppp
RP/0/RP0/CPU0:router(config-if)# no shutdown
RP/0/RP0/CPU0:router(config-if)# ppp authentication ms-chap MIS-access
RP/0/RP0/CPU0:router(config-if)# ppp ms-chap password encrypted xxxx
RP/0/RP0/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
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 0/1/0/0/0
RP/0/0/CPU0:router(config-t1)# channel-group 0
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)# 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 0/1/0/0/1
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 0/1/0/0/1:0
RP/0/0/CPU0:router(config-if)# encapsulation ppp
RP/0/0/CPU0:router(config-if)# group 1
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 12
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: Example, page 305
- show ppp interfaces multilink: Example, page 305
- show ppp interface serial: Example, page 306
- show imds interface multilink: Example, page 306

show multilink interfaces: Example

```
RP/0/0/CPU0:Router# show multilink interfaces multilink 0/3/1/0/301

Multilink0/3/1/0/301 is up, line protocol is up

     Fragmentation: disabled
     Member Links: 2 active, 0 inactive
      - Serial0/3/1/0/0:0: ACTIVE
      - Serial0/3/1/0/1:0: ACTIVE

RRP/0/0/CPU0:Router# show multilink interfaces

Multilink0/3/1/0/301 is up, line protocol is up

     Fragmentation: disabled
     Member Links: 2 active, 0 inactive
      - Serial0/3/1/0/0:0: ACTIVE
      - Serial0/3/1/0/1:0: ACTIVE

Multilink0/3/1/0/302 is up, line protocol is up

     Fragmentation: disabled
     Member Links: 2 active, 0 inactive
      - Serial0/3/1/1/1:0: ACTIVE
      - Serial0/3/1/1/0:0: ACTIVE

Serial0/3/1/0/0:0 is up, line protocol is up
     Multilink group id: 301
     Member status: ACTIVE

Serial0/3/1/1/0:0 is up, line protocol is up
     Multilink group id: 302
     Member status: ACTIVE

Serial0/3/1/0/1:0 is up, line protocol is up
     Multilink group id: 301
     Member status: ACTIVE

Serial0/3/1/1/1:0 is up, line protocol is up
     Multilink group id: 302
     Member status: ACTIVE
```

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

The following 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,* Release 3.4
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*
Information about configuring interfaces and other components on the Cisco CRS-1 from a remote Craft Works Interface (CWI) client management application	*Cisco CRS-1 Series Carrier Routing System Craft Works Interface 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
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: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

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/techsupport

Preconfiguring Physical Interfaces on Cisco IOS XR Software

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

Preconfiguration is supported for the following interfaces on the Cisco CRS-1:

- Management Ethernet
- Gigabit Ethernet
- 10-Gigabit Ethernet
- Packet-over-SONET/SDH (POS)
- Spatial Reuse Protocol (SRP)

Preconfiguration is supported for the following interfaces on the Cisco XR 12000 Series Router:

- Management Ethernet
- GigabitEthernet
- 10-Gigabit Ethernet
- Fast Ethernet
- POS
- Serial
- ATM

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 on Cisco IOS XR Software

Release	Modification
Release 2.0	POS preconfiguration was introduced on the Cisco CRS-1.
Release 3.0	Ethernet preconfiguration was introduced on the Cisco CRS-1.
Release 3.2	Support was added for the Cisco XR 12000 Series Router.
Release 3.3.0	Management Ethernet interface preconfiguration was introduced on the Cisco CRS-1 and Cisco XR 12000 Series Router. Fast Ethernet and serial interface preconfiguration was introduced on Cisco XR 12000 Series Router. SRP interface preconfiguration was introduced on the Cisco CRS-1 for the 4-port OC-192c/STM-64c POS/DPT PLIM.
Release 3.4.0	The configuration procedures in this chapter were modified with enhancements. ATM interface preconfiguration was introduced on the Cisco XR 12000 Series Router.
Release 3.5.0	No modifications.

Contents

Prerequisites for Preconfiguring Physical Interfaces

Before preconfiguring physical interfaces, be sure that the following conditions are met:

- You must be in a user group associated with a task group that includes the proper task IDs for preconfigure commands.

  Task IDs for commands are listed in *Cisco IOS XR Interface and Hardware Component Command Reference*.

- 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

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 failover, 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 failover. The CRS-1 system uses nonstop forwarding (NSF) over tunnels through the failover 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 instance*
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** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface preconfigure** *type instance* **Example:** `RP/0/RP0/CPU0:router(config)# interface preconfigure MgmtEth 0/RP0/CPU0/0`	Enters interface preconfiguration mode for an interface. Replace *type* with the type of interface you want to configure. Replace *instance* with the interface identifier in the *rack/slot/module/port* notation.
Step 3	**ipv4 address** *ip-address subnet-mask* **Example:** `RP/0/RP0/CPU0:router(config-if-pre)# ip address 192.168.255.255/32`	Assigns an IP address and subnet 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 you are configuring.
Step 5	**exit**	Exits interface preconfiguration mode and enters global configuration mode.
Step 6	**exit**	Exits global configuration and enters EXEC mode.

	Command or Action	Purpose
Step 7	**end** or **commit best-effort** **Example:** `RP/0/RP0/CPU0:router(config-if-pre)# end` or `RP/0/RP0/CPU0:router(config-if-pre)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting (yes/no/cancel)?` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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 8	**show running-config** **Example:** `RP/0/RP0/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 314

Preconfiguring an Interface: Example

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

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface preconfigure TenGigE 0/1/0/0.1
RP/0/RP0/CPU0:router(config-if)# ipv4 address 192.168.255.255/32
RP/0/RP0/CPU0:router(config-if)# commit
```

Where to Go Next

- For information about configuring management Ethernet interfaces, see the *Configuring General Router Features* module of the *Cisco IOS XR Getting Started Guide*. Advance configuration of the management Ethernet interfaces is described in the *Advanced Configuration and Modification of the Management Ethernet Interface on Cisco IOS XR Software* module later in this document.
- For information about configuring Gigabit Ethernet and Fast Ethernet interfaces, see the *Configuring Ethernet Interfaces on Cisco IOS XR Software* module earlier in this document.
- For information about configuring POS interfaces, see the *Configuring POS Interfaces on Cisco IOS XR Software* module earlier in this document.
- For information about configuring serial interfaces, see the *Configuring Serial Interfaces on Cisco IOS XR Software* module earlier in this document.
- For information about configuring ATM interfaces, see the *Configuring ATM Interfaces on Cisco IOS XR Software* module earlier in this document.

Additional References

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

Related Documents

Related Topic	Document Title
Cisco IOS XR master command reference	*Cisco IOS XR Master Commands List*, Release 3.2
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*
Information about configuring interfaces and other components on the Cisco CRS-1 from a remote Craft Works Interface (CWI) client management application	*Cisco CRS-1 Series Carrier Routing System Craft Works Interface 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: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

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/techsupport

Configuring 802.1Q VLAN Interfaces on Cisco IOS XR Software

This module describes the configuration and management of 802.1Q VLAN interfaces on routers supporting Cisco IOS XR software.

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 CRS-1 and Cisco XR 12000 Series Router.
Release 3.3.0	• Support was added for VLAN commands on bundled Ethernet interfaces. • Support was added for the **dot1q native vlan** command on Cisco CRS-1 shared port adapters (SPAs).
Release 3.4.0	• The Layer 2 Virtual Private Network (L2VPN) feature was first supported on Ethernet interfaces on the Cisco CRS-1 and Cisco XR 12000 Series Router. • Support was added on Cisco CRS-1 and Cisco XR 12000 Series Router for the 8-Port 1-Gigabit Ethernet SPA.
Release 3.5.0	No modification.

Contents

Prerequisites for Configuring 802.1Q VLAN Interfaces

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

- You must be in a user group associated with a task group that includes the proper task IDs for VLAN commands.

  Task IDs for commands are listed in *Cisco IOS XR Interface and Hardware Component Command Reference*.

- You must have configured a Gigabit Ethernet interface, a 10-Gigabit Ethernet interface, a Fast Ethernet interface, or an Ethernet Bundle.
  - To configure a Gigabit Ethernet, 10-Gigabit Ethernet, or Fast Ethernet interface, see the *Configuring Ethernet Interfaces on Cisco IOS XR Software* module earlier in this document.
  - To configure an Ethernet bundle, see the *Configuring Link Bundling on Cisco IOS XR Software* module later in this document.

Information About Configuring 802.1Q VLAN Interfaces

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

802.1Q VLAN Overview

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

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

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

Cisco IOS XR software supports VLAN subinterface configuration on 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

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

VLAN Subinterfaces on Ethernet Bundles

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

For more information about configuring Ethernet bundles, see the *Configuring Link Bundling on Cisco IOS XR Software* module later in this document. The procedure for creating VLAN subinterfaces on an Ethernet bundle is exactly the same as the procedure for creating VLAN subinterfaces on a physical Ethernet interface.

To create a VLAN subinterface on an Ethernet bundle, see the *How to Configure 802.1Q VLAN Interfaces* section later in this module.

Note Ethernet bundles are supported on the Cisco CRS-1 only. They are not supported on the Cisco XR 12000 Series Router.

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, as described in the "Layer 2 VPN on Ethernet Interfaces" section of the "Configuring Ethernet Interfaces on Cisco IOS XR Software" module earlier in this manual.

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 325.

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

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**} *instance.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**} *instance*] [**location** *instance*]
9. **show vlan trunks** [**brief**] [**location** *instance*] [{**GigabitEthernet** | **TenGigE** | **Bundle-Ether** | **fastethernet**} *instance*] [**summary**]

DETAILED STEPS

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

	Command or Action	Purpose
Step 5	**exit** **Example:** `RP/0/RP0/CPU0:router(config-subif)# exit`	(Optional) Exits the subinterface configuration mode. • 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** or **commit** **Example:** `RP/0/RP0/CPU0:router(config)# end` or `RP/0/RP0/CPU0:router(config)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 8	**show vlan interface** [*type instance*][**location** *instance*] **Example:** `RP/0/RP0/CPU0:router# show vlan interface 5`	(Optional) Displays the interface configuration. • To display the configuration for a particular port, use the **location** keyword. • 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**} *instance*] [**summary**] **Example:** `RP/0/RP0/CPU0:router# show vlan trunk summary`	(Optional) Displays summary information about each of the VLAN trunk interfaces. • The keywords have the following meanings: – **brief—**Displays a brief summary. – **summary—**Displays a full summary. – **location—**Displays information about the VLAN trunk interface on the given port. – **interface—**Displays information about the specified interface or subinterface.

Configuring Native VLAN

This task explains how to configure the native, or default, VLAN on an interface.

Note This task is not available on the Cisco XR 12000 Series Router.

SUMMARY STEPS

1. **configure**
2. **interface** {**GigabitEthernet** | **TenGigE** | **fastethernet** | **Bundle-Ether**} *instance*
3. **dot1q native vlan** *number*
4. **end**
   or
   commit
5. **show vlan trunks** [**brief**] [**location** *instance*] [{**GigabitEthernet** | **TenGigE** | **Bundle-Ether** | **fastethernet**} *instance*] [**summary**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface** {**GigabitEthernet** | **TenGigE** | **fastethernet** | **Bundle-Ether**} *instance* **Example:** `RP/0/RP0/CPU0:router(config)# interface TenGigE 0/2/0/4`	Enters interface configuration mode and specifies the Ethernet interface name and designation. Replace the *instance* argument with one of the following instances: • Physical Ethernet interface instance, or with an Ethernet bundle instance. Naming notation is *rack*/*slot*/*module*/*port*, and a slash between values is required as part of the notation. • Ethernet bundle instance. Range is from 1 through 65535.
Step 3	**dot1q native vlan** *number* **Example:** `RP/0/RP0/CPU0:router(config-if)# dot1q native vlan 1`	Defines the default, or Native VLAN, associated with an 802.1Q trunk interface. • The *number* argument is the ID of the trunk interface. • Range is from 1 through 4094 inclusive (0 and 4095 are reserved).

	Command or Action	Purpose
Step 4	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router(config-if)# end` or `RP/0/RP0/CPU0:router(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 5	**show vlan trunks** [**brief**] [**location** *instance*] [{**GigabitEthernet** | **TenGigE** | **Bundle-Ether** | **fastethernet**} *instance*] [**summary**] **Example:** `RP/0/RP0/CPU0:router# show vlan trunk summary`	(Optional) Displays summary information about each of the VLAN trunk interfaces. • The keywords have the following meanings: – **brief**—Displays a brief summary. – **summary**—Displays a full summary. – **location**— Displays information about the VLAN trunk interface on the given port. – **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**] *instance.subinterface* **l2transport**
3. **dot1q vlan** *vlan-id* [**vlan** {*vlan id* | **any**}]
4. **l2protocol** {**cdp** | **pvst** | **stp** | **vtp**} {[**tunnel**] **experimental** *bits* | **drop**}
5. **end**
   or
   commit
6. **show interfaces** [**GigabitEthernet** | **TenGigE**] *instance*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure terminal`	Enters global configuration mode.
Step 2	**interface** [**GigabitEthernet** | **TenGigE**] *instance.subinterface* **l2transport** **Example:** `RP/0/RP0/CPU0:router(config)# interface TenGigE 0/1/0/0.1 l2transport`	Enters subinterface configuration and specifies the interface type, location, and subinterface number. • Replace the *instance* argument with one of the following instances: – Physical Ethernet interface instance, or with an Ethernet bundle instance. Naming notation is *rack*/*slot*/*module*/*port*, and a slash between values is required as part of the notation. – Ethernet bundle instance. Range is from 1 through 65535. • Replace the *subinterface* argument with the subinterface value. Range is from 0 through 4095. • Naming notation is *instance.subinterface*, and a period between arguments is required as part of the notation. **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* [**vlan** {*vlan-id*|**any**}] **Example:** `RP/0/RP0/CPU0:router(config-subif)# dot1q vlan 10 vlan any`	Assigns a VLAN AC to the subinterface. • 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: **dot1q vlan** *vlan-id* • To configure a Q-in-Q AC, use the following syntax: **dot1q vlan** *vlan-id* **vlan** *vlan-id* • To configure a Q-in-Any AC, use the following syntax: **dot1q vlan** *vlan-id* **vlan any**

	Command or Action	Purpose
Step 4	**l2protocol** {**cdp** | **pvst** | **stp** | **vtp**} {[**tunnel**] **experimental** *bits* | **drop**} **Example:** `RP/0/RP0/CPU0:router(config-subif)# l2protocol pvst tunnel`	Configures Layer 2 protocol tunneling and data units parameters on an interface. Possible protocols are: • **cdp**—Cisco Discovery Protocol (CDP) tunneling and data unit parameters. • **pvst**—Configures VLAN spanning tree protocol tunneling and data unit parameters. • **stp—**spanning tree protocol tunneling and data unit parameters for the • **vtp—**VLAN trunk protocol tunneling and data unit parameters. Include the **tunnel** option if you want to tunnel the packets associated with the specified protocol. Include the **experimental** *bits* keyword argument to modify the experimental bits for the specified protocol. Include the **drop** keyword to drop packets associated with the specified protocol.
Step 5	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router(config-if-l2)# end` or `RP/0/RP0/CPU0:router(config-if-l2)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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**] *instance.subinterface* **Example:** `RP/0/RP0/CPU0:router# show interfaces TenGigE 0/3/0/0.1`	(Optional) Displays statistics for interfaces on the router.

What to do Next

- To configure a Point-to-Point pseudo-wire cross connect on the AC, see the "Implementing MPLS Layer 2 VPNs" module of the *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" task in this module.

SUMMARY STEPS

1. **configure**
2. **no interface** {**GigabitEthernet | TenGigE | fastethernet | Bundle-Ether**] *instance.subinterface*
3. Repeat Step 2 to remove other VLAN subinterfaces.
4. **end**
   or
   commit
5. **show vlan interface** [*type instance*] [**location** *instance*]
6. **show vlan trunks** [**brief**] **[location** *instance*] **[**{**GigabitEthernet | TenGigE | Bundle-Ether | fastethernet**} *instance*] [**summary]**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**no interface** {**GigabitEthernet** | **TenGigE** | **fastethernet** | **Bundle-Ether**] *instance.subinterface* **Example:** `RP/0/RP0/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. • Replace the *instance* argument with one of the following instances: – Physical Ethernet interface instance, or with an Ethernet bundle instance. Naming notation is *rack*/*slot*/*module*/*port*, and a slash between values is required as part of the notation. – Ethernet bundle instance. Range is from 1 through 65535. • Replace the *subinterface* argument with the subinterface value. Range is from 0 through 4095. Naming notation is *instance.subinterface*, and a period between arguments is required as part of the notation.

	Command or Action	Purpose
Step 3	Repeat Step 2 to remove other VLAN subinterfaces.	—
Step 4	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router(config)# end` or `RP/0/RP0/CPU0:router(config)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 5	**show vlan interface** [{**GigabitEthernet** | **TenGigE** | **Bundle-Ether**} *instance* | **location** *instance*] **Example:** `RP/0/RP0/CPU0:router# show vlan trunk summary`	(Optional) Displays the interface configuration. • To display the configuration for a port, use the **location** keyword. • 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**} *instance*] [**summary**] **Example:** `RP/0/RP0/CPU0:router# show vlan trunk summary`	(Optional) Displays summary information about each of the VLAN trunk interfaces. • The keywords have the following meanings: – **brief**—Displays a brief summary. – **summary**—Displays a full summary. – **location**—Displays information about the VLAN trunk interface on the given port. – **interface**—Displays information about the specified interface or subinterface.

Configuration Examples for VLAN Interfaces

This section contains the following example:

VLAN Subinterfaces: Example, page 330

VLAN Subinterfaces: Example

Following is a comprehensive example for creating three VLAN subinterfaces at one time:

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface TenGigE 0/2/0/4.1
RP/0/RP0/CPU0:router(config-subif)# dot1q vlan 10
RP/0/RP0/CPU0:router(config-subif)# ipv4 address 10.0.10.1/24
RP/0/RP0/CPU0:router(config-subif)# interface TenGigE0/2/0/4.2
RP/0/RP0/CPU0:router(config-subif)# dot1q vlan 20
RP/0/RP0/CPU0:router(config-subif)# ipv4 address 10.0.20.1/24
RP/0/RP0/CPU0:router(config-subif)# interface TenGigE0/2/0/4.3
RP/0/RP0/CPU0:router(config-subif)# dot1q vlan 30
RP/0/RP0/CPU0:router(config-subif)# ipv4 address 10.0.30.1/24
RP/0/RP0/CPU0:router(config-subif)# commit
RP/0/RP0/CPU0:router(config-subif)# exit
RP/0/RP0/CPU0:router(config)# exit

RP/0/RP0/CPU0:router# show vlan trunks summary
VLAN trunks: 1,
  1 are 802.1Q (Ether).
Sub-interfaces: 3,
  3 are up.
802.1Q VLANs: 3,
  3 have VLAN Ids.

RP/0/RP0/CPU0:router# show vlan interface
interface           encapsulation  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/RP0/CPU0:router# show vlan trunks brief
interface           encapsulations        intf-state
Te0/2/0/4           802.1Q (Ether)        up
```

Following is a comprehensive example for creating two VLAN subinterfaces on an Ethernet bundle, one at one time:

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface Bundle-Ether 1
RP/0/RP0/CPU0:router(config-if)# ip address 1.0.0.1/24
RP/0/RP0/CPU0:router(config-if)# exit
RP/0/RP0/CPU0:router(config)# interface Bundle-Ether  1.1
RP/0/RP0/CPU0:router(config-subif)#  dot1q vlan 10
RP/0/RP0/CPU0:router(config-subif)# ip address 10.0.0.1/24
RP/0/RP0/CPU0:router(config-subif)# exit
RP/0/RP0/CPU0:router(config)# interface Bundle-Ether 1.2
RP/0/RP0/CPU0:router(config-subif)# dot1q vlan 20
RP/0/RP0/CPU0:router(config-subif)# ip address 20.0.0.1/24
RP/0/RP0/CPU0:router(config-subif)# exit
```

Following is an example for creating a basic Dot1Q AC:

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface TenGigE 0/2/0/4.1
RP/0/RP0/CPU0:router(config-subif)# dot1q vlan 20
RP/0/RP0/CPU0:router(config-subif)# l2protocol pvst tunnel
RP/0/RP0/CPU0:router(config-subif)# commit
RP/0/RP0/CPU0:router(config-subif)# exit
RP/0/RP0/CPU0:router(config)# exit
```

Following is an example for creating a Q-in-Q AC:

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface TenGigE 0/2/0/4.2
RP/0/RP0/CPU0:router(config-subif)# dot1q vlan 20 vlan 10
RP/0/RP0/CPU0:router(config-subif)# l2protocol cdp drop
RP/0/RP0/CPU0:router(config-subif)# commit
RP/0/RP0/CPU0:router(config-subif)# exit
RP/0/RP0/CPU0:router(config)# exit
```

Following is an example for creating a Q-in-Any AC:

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# interface TenGigE 0/2/0/4.3
RP/0/0/CPU0:router(config-subif)# dot1q vlan 30 vlan any
RP/0/0/CPU0:router(config-subif)# l2protocol vtp experimental 7
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

The following 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,* Release 3.2
Cisco IOS XR interface configuration commands	*Cisco CRS-1 Series 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*
Information about configuring interfaces and other components on the Cisco CRS-1 from a remote Craft Works Interface (CWI) client management application	*Cisco CRS-1 Series Carrier Routing System Craft Works Interface 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: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

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/techsupport

Configuring Clear Channel SONET Controllers on Cisco IOS XR Software

This module describes the configuration of clear channel SONET controllers on routers supporting Cisco IOS XR software. 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.

Note The Cisco CRS-1 supports clear channel SONET controllers only. The Cisco XR 12000 Series Router supports both clear channel and channelized SONET controllers on the 1-Port Channelized OC-3 and OC-12 SPAs. For more information about configuring a channelized SONET controller, see the Configuring Channelized SONET 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 2.0	This feature was introduced on the Cisco CRS-1.
Release 3.0	No modifications.
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: • Cisco XR 12000 SIP-401 • Cisco XR 12000 SIP-501 • Cisco XR 12000 SIP-601 Support was added on the Cisco XR 12000 Series Router for the 2-Port OC-48 POS/RPR SPA. Support was added on the Cisco CRS-1 for the 1 port OC-768c/STM-256c POS PLIM.

Release 3.4.0	Support was added on the Cisco CRS-1 for the following hardware: • 2 port OC-48/STM-16 POS SPA • 4 port OC-48/STM-16 POS SPA
Release 3.5.0	Support was added on the Cisco XR 12000 Series Router for the following SPAs: • 1-Port Channelized OC-3 SPA • 1-Port Channelized OC-12 SPA • 2-Port OC-12 POS • 4-Port OC-12 POS • 8-Port OC-12 POS • 4-Port OC-3 POS • 8-Port OC-3 POS

Contents

Prerequisites for Configuring Clear Channel SONET Controllers

Before configuring SONET controllers, be sure 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 for SONET commands.

  Task IDs for commands are listed in *Cisco IOS XR Interface and Hardware Component Command Reference*.

- You have at least one of the following physical layer interface module (PLIM) cards installed in your chassis:
  - 4 port OC-3c/STM-1 POS SPA
  - 8 port OC-12c/STM-4 POS SPA
  - 2 port OC-48/STM-16 POS SPA (Cisco XR 12000 Series Router only)
  - 4 port OC-48/STM-16 POS SPA (Cisco XR 12000 Series Router only)
  - 16 port OC-48c/STM-16c POS
  - 4 port OC-192c/STM-64c POS
  - 1 port OC-192c/STM-64 POS/RPR XFP SPA
  - 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 335
- Default Configuration Values for SONET Controllers, page 335
- SONET APS, page 336

SONET Controller Overview

In routers supporting Cisco IOS XR software, the physical ports on the line cards are called controllers. Before you can configure a POS or SRP 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.

The router uses SONET controllers for Layer 1 and Layer 2 processing.

Note Path UNEQ is not supported on the OC-768 card. Therefore, UNEQ-P and PPLM alarms are not reported for any unequipped C2 byte that is received on an OC-768 interface. Cisco supports all error codes from the ERDI-P standard except for the UNEQ-P code.

Default Configuration Values for SONET Controllers

Table 14 describes the default configuration parameters that are present on the SONET controllers.

Table 14 SONET Controller Default Configuration Values

Parameter	Default Value	Configuration File Entry
B3 BER threshold crossing alert (TCA) reporting status	**enabled**	To disable B3 BER TCA reporting on the SONET controller, enter the **path report b3-tca disable** command in SONET controller configuration submode.
Path loss of pointer reporting status	**enabled**	To disable path loss of pointer reporting status on a SONET controller, enter the **path report plop disable** command in SONET controller configuration submode.

Table 14 SONET Controller Default Configuration Values

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 packet-over-SONET (POS) 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" POS interface in the SONET network as the backup for "working" POS 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 may 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

This task explains how to configure SONET controllers as a prerequisite to configuring POS and SRP interfaces.

Prerequisites

- You need to have a POS line card or SPA installed in a router that is running Cisco IOS XR software.
- 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 -340.

Restrictions

SUMMARY STEPS

1. **show version**
2. **show interfaces** [**pos** | **srp**] *instance*
3. **configure**
4. **controller sonet** *instance*
5. **clock source** {**internal** | **line**}
6. **delay trigger line** *value*
7. **framing** {**sdh** | **sonet**}
8. **loopback** {**internal** | **line**}
9. **overhead** {**j0** | **s1s0**} *byte-value*
10. **path** *keyword* [*values*]
11. **end**
    or
    commit
12. **show controllers sonet** *instance*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**show version** **Example:** `RP/0/RP0/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** [**pos**| **srp**] *instance* **Example:** `RP/0/RP0/CPU0:router# show interface pos 0/1/0/0`	(Optional) Displays the configured interface and checks the status of each interface port.
Step 3	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 4	**controller sonet** *instance* **Example:** `RP/0/RP0/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 5	**clock source** {**internal** | **line**} **Example:** `RP/0/RP0/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. • 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. • The **line** clock is the default. **Note** Internal clocking is required for SRP interfaces.
Step 6	**delay trigger line** *value* **Example:** `RP/0/RP0/CPU0:router(config-sonet)# delay trigger line 0`	(Optional) Configures the SONET port delay trigger line values, where the trigger values are in the range from 0 through 511 milliseconds, and the default delay trigger value is 0 milliseconds.
Step 7	**framing** {**sdh** | **sonet**} **Example:** `RP/0/RP0/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. SONET framing (**sonet**) is the default.
Step 8	**loopback** {**internal** | **line**} **Example:** `RP/0/RP0/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.

	Command or Action	Purpose
Step 9	**overhead** {**j0** | **s1s0**} *byte-value* **Example:** `RP/0/RP0/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. • The default byte value for the **j0** keyword is 0xcc, and the default byte value for the **s1s0** keyword is 0. • The range of valid values for **j0** and **s1s0** is 0 through 255.
Step 10	**path** *keyword* [*values*] **Example:** `RP/0/RP0/CPU0:router(config-sonet)# path delay trigger 25`	(Optional) Configures SONET controller path values. Keyword definitions are as follows: • **ais-shut**—Set sending path alarm indication signal (PAIS) when shut down. • **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 511 milliseconds. The default value is 0. • **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). • **report [b3-tca disable | pais | plop disable | prdi**]—Set SONET path alarm reporting. Specifies which alarms are reported and which bit error rate (BER) thresholds will signal an alarm, where the **b3-tca disable** keyword disables B3 BER threshold crossing alert (TCA) reporting status; **pais** sets PAIS reporting status; **plop disable** disables path loss of pointer reporting status; and **prdi** sets path remote defect indication reporting status. • **scrambling disable**—Disable SPE scrambling. Note that SPE scrambling is enabled by default. • **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. • **uneq-shut—**Sets sending Unequipped (UNEQ) when shut down.

	Command or Action	Purpose
Step 11	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router(config-sonet)# end` or `RP/0/RP0/CPU0:router(config-sonet)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 12	**show controllers sonet** *instance* **Example:** `RP/0/RP0/CPU0:router# show controllers sonet 0/1/0/0`	Verifies the SONET controller configuration.

What to do Next

- To configure an associated POS interface, see the *Configuring POS Interfaces on Cisco IOS XR Software* module later in this document.
- To configure an associated SRP interface, see the *Configuring SRP Interfaces on Cisco IOS XR Software* module later in this document.

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

You need to have a POS line card or SPA installed in a router that is running Cisco IOS XR software.

Restrictions

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** *instance*
10. **ipv4 address** *ip-address mask*
11. **pos crc** {**16** | **32**}
12. **keepalive** {*seconds* | **disable**}
13. **no shutdown**
14. Repeat Step 9 to Step 13 for each channel in the group.
15. **exit**
16. **controller sonet** *instance*
17. **ais-shut**
18. **path scrambling disable**
19. **clock source** {**internal** | **line**}
20. Repeat Step 16 to Step 19 for each channel of the group.
21. **end**
    or
    commit
22. **exit**
23. **exit**
24. **show aps**
25. **show aps group** [*number*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**aps group** *number* **Example:** `RP/0/RP0/CPU0:router(config)# aps group 1`	Adds an APS group with a specified number and enters APS group configuration mode. • Use the **aps group** command in global configuration mode. • 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. **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. **Note** The **aps group** command is used even when a single protect group is configured.
Step 3	**channel** {**0** | **1**} **local sonet** *interface* **Example:** `RP/0/RP0/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. **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* **Example:** `RP/0/RP0/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. **Note** In this example, the loopback interface is used as the interconnect.
Step 7	**ipv4 address** *ip-address mask* **Example:** `RP/0/RP0/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.
Step 9	**interface pos** *instance* **Example:** `RP/0/RP0/CPU0:router(config)# interface POS 0/2/0/0`	Connects the interface for the channel selected in Step 3, and enters interface configuration mode for a POS interface.

	Command or Action	Purpose
Step 10	**ipv4 address** *ip-address mask* **Example:** `RP/0/RP0/CPU0:router(config-if)# ipv4 address 172.18.0.1 255.255.255.224`	Assigns an IPv4 address and subnet mask to the POS interface.
Step 11	**pos crc** {**16** | **32**} **Example:** `RP/0/RP0/CPU0:router(config-if)# pos 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. **Note** The default CRC is **32**.
Step 12	**keepalive** {*seconds* | **disable**} **Example:** `RP/0/RP0/CPU0:router(config-if)# keepalive disable`	Sets the keepalive timer for the channel. Replace the *seconds* argument with the Number of seconds that define the keepalive interval. Default is 10 seconds. Enter the **keepalive disable** command to turn off the keepalive timer.
Step 13	**no shutdown** **Example:** `RP/0/RP0/CPU0:router(config-if)# no shutdown`	Removes the shutdown configuration. • 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 14	Repeat Step 9 through Step 13 for each channel in the group.	—
Step 15	**exit**	Exits interface configuration mode for a POS interface, and enters global configuration mode.
Step 16	**controller sonet** *instance* **Example:** `RP/0/RP0/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 17	**ais-shut** **Example:** `RP/0/RP0/CPU0:router(config-sonet)# ais-shut`	Configures POS path values such as alarm indication signal (AIS) at shut down.
Step 18	**path scrambling disable** **Example:** `RP/0/RP0/CPU0:router(config-sonet)# path scrambling disable`	(Optional) Disables synchronous payload envelope (SPE) scrambling. **Note** SPE scrambling is enabled by default.

	Command or Action	Purpose
Step 19	**clock source** {**internal** | **line**} **Example:** `RP/0/RP0/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. • 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. • The line clock (**line**) is the default.
Step 20	Repeat Step 16 through Step 19 for each channel in the group.	—
Step 21	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router(config-sonet)# end` or `RP/0/RP0/CPU0:router(config-sonet)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 22	**exit**	Exits SONET controller configuration mode, and enters global configuration mode.
Step 23	**exit**	Exits global configuration mode, and enters EXEC mode.
Step 24	**show aps** **Example:** `RP/0/RP0/CPU0:router# show aps`	(Optional) Displays the operational status for all configured SONET APS groups.
Step 25	**show aps group** [*number*] **Example:** `RP/0/RP0/CPU0:router# show aps group 3`	(Optional) Displays the operational status for configured SONET APS groups. **Note** The **show aps group** command is more useful than the **show aps** command when multiple groups are defined.

What to do Next

- To configure the POS interfaces in the APS group, see the *Configuring POS Interfaces on Cisco IOS XR Software* module later in this document.
- To configure an a hold-off timer to prevent FRR from being triggered while the CORE network is doing a restoration, see the "Configuring a Hold-off Timer to Prevent Fast Reroute from being Triggered" section on page 345 module later in this document.

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 340.

SUMMARY STEPS

1. **configure**
2. **controller sonet** *instance*
3. **delay trigger line** *value*
   or
   path delay trigger *value*
4. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**controller sonet** *instance* **Example:** `RP/0/RP0/CPU0:router(config)# controller sonet 0/6/0/0`	Enters SONET configuration mode.

	Command or Action	Purpose
Step 3	**delay trigger line** *value* or **path delay trigger** *value* **Example:** `RP/0/RP0/CPU0:router(config-sonet)# delay trigger line 250` `or` `RP/0/RP0/CPU0:router(config-sonet)# path delay trigger 300`	Configures SONET port delay trigger values in milliseconds. **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* **delay trigger line** or **controller sonet** *r/s/m/p* **path delay trigger**.
Step 4	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router(config-sonet)# end` or `RP/0/RP0/CPU0:router(config-sonet)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)?` `[cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

What to do Next

- To configure the POS interfaces in the APS group, see the *Configuring POS Interfaces on Cisco IOS XR Software* module later in this document.

Configuration Examples for SONET Controllers

This section contains the following examples:

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 337. 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.

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# controller sonet 0/1/0/0
RP/0/RP0/CPU0:router(config-sonet)# ais-shut
RP/0/RP0/CPU0:router(config-sonet)# clock source internal
RP/0/RP0/CPU0:router(config-sonet)# framing sonet
RP/0/RP0/CPU0:router(config-sonet)# loopback internal
Loopback is a traffic-affecting operation
RP/0/RP0/CPU0:router(config-sonet)# overhead s1s0 1
RP/0/RP0/CPU0:router(config-sonet)# path ais-shut
RP/0/RP0/CPU0:router(config-sonet)# path delay trigger 0
RP/0/RP0/CPU0:router(config-sonet)# path overhead j1 line 11
RP/0/RP0/CPU0:router(config-sonet)# path report b3-tca disable
RP/0/RP0/CPU0:router(config-sonet)# path scrambling disable
RP/0/RP0/CPU0:router(config-sonet)# path threshold b3-tca 6
RP/0/RP0/CPU0:router(config-sonet)# path uneq-shut
RP/0/RP0/CPU0:router(config-sonet)# report b2-tca disable
RP/0/RP0/CPU0:router(config-sonet)# threshold b2-tca 4
RP/0/RP0/CPU0:router(config-sonet)# commit
```

SONET APS Group Configuration: Example

The following example shows how to configure a two-router SONET APS group:

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# aps group 1
RP/0/RP0/CPU0:router(config-aps)# channel 0 local SONET 0/0/0/1
RP/0/RP0/CPU0:router(config-aps)# channel 1 local SONET 0/0/0/2
RP/0/RP0/CPU0:router(config-aps)# interface loopback 0
RP/0/RP0/CPU0:router(config-if)# ipv4 address 172.18.23.169 255.255.255.0
RP/0/RP0/CPU0:router(config-if)# exit
RP/0/RP0/CPU0:router(config-if)# interface POS 0/0/0/2
RP/0/RP0/CPU0:router(config-if)# ipv4 address 172.18.69.123 255.255.255.0
RP/0/RP0/CPU0:router(config-if)# pos crc 32
RP/0/RP0/CPU0:router(config-if)# keepalive 20
RP/0/RP0/CPU0:router(config-if)# no shutdown
RP/0/RP0/CPU0:router(config-if)# interface POS 0/0/0/1
RP/0/RP0/CPU0:router(config-if)# ipv4 address 172.18.69.123 255.255.255.0
RP/0/RP0/CPU0:router(config-if)# keepalive disable
RP/0/RP0/CPU0:router(config-if)# no shutdown
RP/0/RP0/CPU0:router(config-if)# exit
RP/0/RP0/CPU0:router(config)# controller SONET 0/0/0/2
RP/0/RP0/CPU0:router(config-sonet)# ais-shut
RP/0/RP0/CPU0:router(config-sonet)# path
```

```
RP/0/RP0/CPU0:router(config-sonet-path)# scrambling disable
RP/0/RP0/CPU0:router(config-sonet-path)# clock source internal
RP/0/RP0/CPU0:router(config-sonet)# controller SONET 0/0/0/1
RP/0/RP0/CPU0:router(config-sonet)# ais-shut
RP/0/RP0/CPU0:router(config-sonet)# path
RP/0/RP0/CPU0:router(config-sonet-path)# scrambling disable
RP/0/RP0/CPU0:router(config-sonet-path)# clock source internal
RP/0/RP0/CPU0:router(config-sonet)# commit
RP/0/RP0/CPU0:router(config-sonet)# end
RP/0/RP0/CPU0:router# show aps

APS Group 1
  Protect ch 0 (SONET0_0_0_1): Disabled
    SONET framing, SONET signalling, bidirectional, non-revertive
    Rx K1: 0x00 (No Request - Null)
       K2: 0x00 (bridging Null, 1+1, non-aps)
    Tx K1: 0x00 (No Request - Null)
       K2: 0x05 (bridging Null, 1+1, bidirectional)
  Working ch 1 (0_0_0_2): Enabled


RP/0/RP0/CPU0:router# show aps group 1

APS Group 1
  Protect ch 0 (SONET0_0_0_1): Disabled
    SONET framing, SONET signalling, bidirectional, non-revertive
    Rx K1: 0x00 (No Request - Null)
       K2: 0x00 (bridging Null, 1+1, non-aps)
    Tx K1: 0x00 (No Request - Null)
       K2: 0x05 (bridging Null, 1+1, bidirectional)
  Working ch 1 (SONET0_0_0_2): Enabled

RP/0/RP0/CPU0:router#
```

Where to Go Next

After configuring the SONET controller, you can configure the POS or SRP interface that is associated with that controller.

- To configure a POS interface, see the *Configuring POS Interfaces on Cisco IOS XR Software* module in this document.
- To configure an SRP interface, see the *Configuring SRP Interfaces on Cisco IOS XR Software* module in this document.

Additional References

The following 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*, Release 3.2
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*
Information about configuring router interfaces and other components from a remote Craft Works Interface (CWI) client management application	*Cisco Craft Works Interface 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: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

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/techsupport

Configuring Serial Interfaces on Cisco IOS XR Software

This module describes the serial interfaces on routers supporting Cisco IOS XR software. 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. Support was added on the Cisco XR 12000 Series Router for the following hardware: • Cisco XR 12000 SIP-401 • Cisco XR 12000 SIP-501 • Cisco XR 12000 SIP-601 Support was added on the Cisco XR 12000 Series Router for the following SPAs: • 2-Port and 4-Port Channelized T3 Serial SPA • 2-Port and 4-Port T3/E3 Serial SPA
Release 3.4.0	Support for the following features was introduced: • Subinterfaces with permanent virtual circuits (PVCs) • Frame Relay encapsulation on serial main interfaces and PVCs.
Release 3.4.1	Multilink PPP was supported on serial interfaces.
Release 3.5.0	No modification.

Contents

Prerequisites for Configuring Serial Interfaces

Before configuring serial interfaces, be sure 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 for serial Interface commands. Task IDs for commands are listed in *Cisco IOS XR Interface and Hardware Component Command Reference*.
- Your hardware must support T3/E3 controllers and serial interfaces. The following hardware supports T3/E3 controllers and serial interfaces in Cisco IOS XR Software Release 3.3:
  - 2-Port and 4-Port Clear Channel T3/E3 SPAs
  - 2-Port and 4-Port Channelized T3 SPAs

  Note The 2-Port and 4-Port Channelized T3 SPAs support T1/E1 controllers and DS0 channels.

- You have already configured the clear channel T3/E3 controller or channelized T3-to-T1/E1controller that is associated with the serial interface you want to configure, as described in the Configuring Clear Channel T3/E3 Controllers and Channelized T3 Controllers on Cisco IOS XR Software module earlier in this document.

Information About Serial Interfaces

Serial interfaces are supported on the following Cisco XR 12000 Series Router shared port adapter (SPA) cards:

- 2-Port and 4-Port Clear Channel T3/E3 SPAs
- 2-Port and 4-Port Channelized T3 SPAs

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.

T3/E3 serial interfaces are automatically created on clear channel T3/E3 controllers. On channelized T3-to-T1/E1 controllers, serial interfaces are automatically created when users configure individual DS0 channel groups the T1/E1 controllers.

To configure serial interfaces, you must understand the following concepts:

High Level Over-View: Serial Interface Configuration on Clear-Channel SPAs

Table 15 provides a high-level overview of the tasks required to configure a T3 serial interface on a 2-Port and 4-Port Clear Channel T3/E3 SPA.

Table 15 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. **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 Controllers and Channelized T3 Controllers on Cisco IOS XR Software"	"Setting the Card Type for the Clear Channel SPAs"
2.	Configure the T3 controller.	"Configuring Clear Channel T3/E3 Controllers and Channelized T3 Controllers on Cisco IOS XR Software"	"Setting the Card Type for the Clear Channel SPAs"
3.	Configure the serial interface that is associated with the T3 controller you configured in Step 2.	"Configuring Serial Interfaces on Cisco IOS XR Software"	"How to Configure Serial Interfaces"

Table 16 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 16 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 Controllers and Channelized T3 Controllers on Cisco IOS XR Software"	*Setting the Card Type for the Clear Channel SPAs*
2.	Configure the E3 controller.	"Configuring Clear Channel T3/E3 Controllers and Channelized T3 Controllers on Cisco IOS XR Software"	*Setting the Card Type for the Clear Channel SPAs*
3.	Configure the serial interface that is associated with the E3 controller you configured in Step 2.	Configuring Serial Interfaces on Cisco IOS XR Software	*How to Configure Serial Interfaces*

High Level Over-View: Serial Interface Configuration on Channelized SPAs

Table 17 provides a high-level overview of the tasks required to configure a T1 serial interface on the 2-Port and 4-Port Channelized T3 SPA.

Table 17 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. 28 T1 controllers are automatically created.	"Configuring Clear Channel T3/E3 Controllers and Channelized T3 Controllers on Cisco IOS XR Software"	"Configuring a Channelized T3 Controller"
2.	Create and configure DS0 channel groups on the T1 controllers you created in Step 1.	"Configuring Clear Channel T3/E3 Controllers and Channelized T3 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 Software"	"How to Configure Serial Interfaces"

Table 18 provides a high-level overview of the tasks required to configure an E1 serial interface on the 2-Port and 4-Port Channelized T3 SPA.

Table 18 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. 21 E1 controllers are automatically created.	Configuring Clear Channel T3/E3 Controllers and Channelized T3 Controllers on Cisco IOS XR Software	Configuring a Channelized T3 Controller
2.	Create and configure DS0 channel groups on the E1 controllers you created in Step 1.	Configuring Clear Channel T3/E3 Controllers and Channelized T3 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 Software	"How to Configure Serial Interfaces"

Serial Interface Concepts

To configure serial interfaces, you must understand the following concepts:

On the Cisco XR 12000 Series Router, a single serial interface carries data over a single interface using PPP, Cisco HDLC, or Frame Relay encapsulation.

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 357.

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 three 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 357.

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 serial interface.

Note Enabling or disabling PPP authentication does not affect the local router's willingness to authenticate itself to the remote device.

Multilink PPP

Multilink Point-to-Point Protocol (MLPPP) provides a method for combining multiple physical links into one logical link. The implementation of MLPPP on Cisco XR 12000 Series Routers combines multiple PPP serial interfaces into one multilink interface. MLPPP performs the fragmenting, reassembling, and sequencing of datagrams across multiple PPP links. MLPPP is supported on the 2-Port and 4-Port Channelized T3 SPAs.

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 later 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.

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).

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**.

Note During Minimal Disruptive Restart (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; 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.

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 Cisco IOS XR Software" chapter 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 number of PVCs supported on POS PVCs with **cisco** LMI is 557.
For non-Cisco LMI types, up to 992 PVCs are supported under a single main interface.

To configure Frame Relay encapsulation on serial 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 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.

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 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 chapter describes how to configure a Layer 2 AC on a Frame Relay encapsilated serial interface.

Note In Release 3.5.0, serial interfaces support DLCI mode layer 2 ACs only; layer 2 port mode ACs are not supported on serial interfaces.

Note For detailed information about configuring L2TPv3 in your network, see the "Layer 2 Tunnel Protocol Version 3 on Cisco IOS XR Software" module of *Cisco IOS XR Multiprotocol Label Switching Configuration Guide.*

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 19 are present. These default settings can be changed by configuration.

Table 19 Serial Interface Default Settings

Parameter	Configuration File Entry	Default Settings
Keepalive	**keepalive** [**disable**] **no keepalive**	keepalive 10 seconds
Encapsulation	**encapsulation** [**hdlc** | **ppp** | **frame-relay**]	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 1 to 24; E1 channels range from 1 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.

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 Controllers and Channelized T3 Controllers on Cisco IOS XR Software module earlier in this document, you can configure the serial interfaces associated with that controller. The following task describes how to configure a serial interface.

- Bringing Up a Serial Interface, page 361
- Configuring Optional Serial Interface Parameters, page 364
- Creating a Point-to-Point Serial Subinterface with a PVC, page 367

- Modifying the Keepalive Interval on Serial Interfaces, page 372

Bringing Up a Serial Interface

This task describes the commands you can use to bring up a serial 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:

- Cisco XR 12000 SIP-401
- Cisco XR 12000 SIP-501
- Cisco XR 12000 SIP-601
- 2-Port and 4-Port Channelized T3 Serial SPA
- 2-Port and 4-Port T3/E3 Serial 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** *instance*
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** *instance*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**show interfaces** **Example:** `RP/0/0/CPU0:router# show interfaces`	(Optional) Displays configured interfaces. • Use this command to also confirm that the router recognizes the PLIM card.
Step 2	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 3	**interface serial** *instance* **Example:** `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* **Example:** `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. **Caution** 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 PVC.
Step 5	**no shutdown** **Example:** `RP/0/0/CPU0:router (config-if)# no shutdown`	Removes the shutdown configuration. **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).

	Command or Action	Purpose
Step 6	**end** or **commit** **Example:** `RP/0/0/CPU0:router (config-if)# end` or `RP/0/0/CPU0:router(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 7	**exit** **Example:** `RP/0/0/CPU0:router (config-if)# exit`	Exits interface configuration mode and enters global configuration mode.
Step 8	**exit** **Example:** `RP/0/0/CPU0:router (config)# exit`	Exits global configuration mode and enters EXEC mode.
Step 9	**show interfaces** **configure** **interface serial** *instance* **no shut** **exit** **exit** **commit** **Example:** `RP/0/0/CPU0:router# show interfaces` `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)# ipv4 address 10.1.2.1 255.255.255.224` `RP/0/0/CPU0:router (config-if)# no shutdown` `RP/0/0/CPU0:router (config-if)# commit` `RP/0/0/CPU0:router (config-if)# exit` `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. **Note** The configuration on both ends of the serial connection must match.

	Command or Action	Purpose
Step 10	**show ipv4 interface brief** **Example:** `RP/0/0/CPU0:router # show ipv4 interface brief`	Verifies that the interface is active and properly configured. If you have brought up a serial interface properly, the "Status" field for that interface in the **show ipv4 interface brief** command output shows "Up."
Step 11	**show interfaces serial** *instance* **Example:** `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 364.

Configuring Optional Serial Interface Parameters

This task describes the commands you can use 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 361.

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** *instance*
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** [*instance*]

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface serial** *instance* **Example:** `RP/0/RP0/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**]] **Example:** `RP/0/RP0/CPU0:router(config-if)# encapsulation hdlc`	(Optional) Configures the interface encapsulation parameters and details such as HDLC, PPP or Frame Relay. **Note** The default encapsulation is **hdlc**.
Step 4	**serial** **Example:** `RP/0/0/CPU0:router(config-if)# serial` `RP/0/0/CPU0:ios(config-if-serial)#`	(Optional) Enters serial submode to configure the serial parameters.
Step 5	**crc** *length* **Example:** `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. **Note** The default is CRC length is 16.
Step 6	**invert** **Example:** `RP/0/0/CPU0:ios(config-if-serial)# inverts`	(Optional) Inverts the data stream.
Step 7	**scramble** **Example:** `RP/0/0/CPU0:ios(config-if-serial)# scramble`	(Optional) Enables payload scrambling on the interface. **Note** Payload scrambling is disabled on the interface.
Step 8	**transmit-delay** *hdlc-flags* **Example:** `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. **Note** Transmit delay is disabled by default (the transmit delay is set to **0**).

	Command or Action	Purpose
Step 9	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router (config-if)# end` or `RP/0/RP0/CPU0:router(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 10	**exit** **Example:** `RP/0/0/CPU0:router(config-if-serial)# exit` `RP/0/0/CPU0:router(config-if)#`	Exits serial configuration mode.
Step 11	**exit** **Example:** `RP/0/RP0/CPU0:router (config-if)# exit`	Exits interface configuration mode and enters global configuration mode.
Step 12	**exit** **Example:** `RP/0/RP0/CPU0:router (config)# exit`	Exits global configuration mode and enters EXEC mode.
Step 13	**show interfaces serial** [*instance*] **Example:** `RP/0/RP0/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 367.
- 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 372.

- 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 Cisco IOS XR Software" module later in this manual.

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 361.

Restrictions

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

SUMMARY STEPS

1. **configure**
2. **interface serial** *instance.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** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface serial** *instance.subinterface* **point-to-point** **Example:** `RP/0/0/CPU0:router (config)# interface serial 0/6/0/1.10`	Enters serial subinterface configuration mode.
Step 3	**ipv4 address** *ipv4_address/prefix* **Example:** `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* **Example:** `RP/0/0/CPU0:router (config-subif)# pvc 20`	Creates a serial permanent virtual circuit (PVC) and enters Frame Relay PVC configuration submode. Replace *dlci* with a PVC identifier, in the range from 16 to 1007. **Note** Only one PVC is allowed per subinterface.

	Command or Action	Purpose
Step 5	**end** or **commit** **Example:** `RP/0/0/CPU0:router (config-subif)# end` or `RP/0/0/CPU0:router(config-subif)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 6	**configure** **interface serial** *instance* **pvc** *dlci* **commit** **Example:** `RP/0/0/CPU0:router# configure` `RP/0/0/CPU0:router (config)# interface serial 0/6/0/1.10` `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)# commit`	Repeat Step 1 through Step 5 to bring up the serial subinterface and any associated PVC at the other end of the connection. **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 Serial Interface Parameters" section on page 364.
- 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 Cisco IOS XR Software" module later in this manual.
- 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.

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 367.

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** *instance.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*

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface serial** *instance.subinterface* **Example:** `RP/0/0/CPU0:router (config)# interface serial 0/6/0/1.10`	Enters serial subinterface configuration mode.
Step 3	**pvc** *dlci* **Example:** `RP/0/0/CPU0:router (config-subif)# pvc 20`	Enters subinterface configuration mode for the PVC.

	Command or Action	Purpose
Step 4	**encap** [**cisco** | **ietf**] **Example:** `RP/0/0/CPU0:router (config-subif)# encap ietf`	(Optional) Configures the encapsulation for a Frame Relay PVC. **Note** If the encapsulation type is not configured explicitly for a PVC, then that PVC inherits the encapsulation type from the main serial interface.
Step 5	**service-policy** {**input** | **output**} *policy-map* **Example:** `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	**end** or **commit** **Example:** `RP/0/0/CPU0:router (config-subif)# end` or `RP/0/0/CPU0:router(config-subif)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)?` `[cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 7	**configure** **interface serial** *instance.subinterface* **pvc** *dlci* **encap** [**cisco** | **ietf**] **commit** **Example:** `RP/0/0/CPU0:router# configure` `RP/0/0/CPU0:router (config)# interface serial 0/6/0/1.10` `RP/0/0/CPU0:router (config-subif)# pvc dlci` `RP/0/0/CPU0:router (config-fr-vc)# encap cisco` `RP/0/0/CPU0:router (config-fr-vc)# commit`	Repeat Step 1 through Step 6 to bring up the serial subinterface and any associated PVC at the other end of the connection. **Note** The configuration on both ends of the subinterface connection must match.
Step 8	**show frame-relay pvc** *dlci-number* **Example:** `RP/0/RP0/CPU0:router# show frame-relay pvc 20`	(Optional) Verifies the configuration of specified serial interface.

What to do Next

- 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.
- 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 Cisco IOS XR Software" module later 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 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 Serial Interface Parameters" section on page 364.

Restrictions

During MDR, the keepalive interval must be 10 seconds or more.

SUMMARY STEPS

1. **configure**
2. **interface serial** *instance*
3. **keepalive** {*seconds* | **disable**}
   or
   no keepalive
4. **end**
   or
   commit
5. **show interfaces** *type instance*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface serial** *instance* **Example:** `RP/0/RP0/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**} or **no keepalive** **Example:** `RP/0/RP0/CPU0:router(config-if)# keepalive 3` or `RP/0/RP0/CPU0:router(config-if)# no keepalive`	Specifies the number of seconds between keepalive messages. • Use the **keepalive disable** command, the **no keepalive**, or the **keepalive** command with an argument of 0 to disable the keepalive feature. • Range is from 1 through 30 seconds. • Default is 10 seconds. • 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** or **commit** **Example:** `RP/0/RP0/CPU0:router(config-if)# end` or `RP/0/RP0/CPU0:router(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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** *instance* **Example:** `RP/0/RP0/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 an serial interface, you must bring up an serial interface, as described in the “Bringing Up a Serial Interface” section on page 361.

Restrictions

Only one PVC can be configured for each serial subinterface.

SUMMARY STEPS

1. **configure**
2. **interface serial** *instance.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** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface serial** *instance.subinterface* **l2transport** **Example:** `RP/0/0/CPU0:router(config)# interface serial 0/6/0/1.10 l2transport`	Creates a subinterface and enters serial subinterface configuration mode for that subinterface.
Step 3	**pvc** *vpi/vci* **Example:** `RP/0/0/CPU0:router(config-if)# pvc 5/20`	Creates an serial permanent virtual circuit (PVC) and enters serial Layer 2 transport PVC configuration mode. **Note** Only one PVC is allowed per subinterface.

	Command or Action	Purpose
Step 4	**end** or **commit** **Example:** `RP/0/0/CPU0:router(config-fr-vc)# end` or `RP/0/0/CPU0:router(config-fr-vc)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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. **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 376.
- To configure a point-to-point pseudowire XConnect on the AC you just created, see the "Layer 2 Tunnel Protocol Version 3 on Cisco IOS XR Software" module of *Cisco IOS XR Multiprotocol Label Switching Configuration Guide*.

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 374.

Restrictions

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

SUMMARY STEPS

1. **configure**
2. **interface serial** *instance.subinterface* **l2transport**
3. **pvc** *dlci*
4. **encap** [**cisco** | **ietf**]
5. **service-policy** {**input** | **output**} *policy-map*
6. **fragment end-to-end** *fragment-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** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface serial** *instance.subinterface* **l2transport** **Example:** `RP/0/0/CPU0:router(config)# interface serial 0/6/0/1.10 l2transport`	Enters serial subinterface configuration mode for a Layer 2 serial subinterface.
Step 3	**pvc** *dlci* **Example:** `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**} **Example:** `RP/0/0/CPU0:router(config-fr-vc)# encapsulation aal5`	Configures the encapsulation for a Frame Relay PVC.
Step 5	**fragment end-to-end** fragment-size Example: `RP/0/0/CPU0:router(config-fr-vc)# fragment end-to-end 100`	Enables fragmentation of Frame Relay frames on an interface. 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. Valid values are from 16 to 1600; the default is 53.

	Command or Action	Purpose
Step 6	**service-policy** {**input** | **output**} *policy-map* **Example:** `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 7	**end** or **commit** **Example:** `RP/0/0/CPU0:router(config-serial-l2transport-pvc)# end` or `RP/0/0/CPU0:router(config-serial-l2transport-pvc)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. 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. **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 AC you just created, see the "Layer 2 Tunnel Protocol Version 3 on Cisco IOS XR Software" module of *Cisco IOS XR Multiprotocol Label Switching Configuration Guide*.

Configuration Examples for Serial Interfaces

This section provides the following configuration examples:

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/RP0/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/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface serial 0/3/0/0/0:0
RP/0/RP0/CPU0:router(config-if)# keepalive 10
RP/0/RP0/CPU0:router(config-if)# commit
```

The following example shows how to modify the optional serial interface parameters:

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/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 bring up a serial interface with Frame Relay encapsulation:

```
RP/0/0/CPU0:router# configure
RP/0/0/CPU0:router(config)# interface
RP/0/RP0/CPU0:router(config)# interface serial 0/3/0/0/0:0
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
```

The following example shows how create a serial subinterface with a PVC on the main serial interface:

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/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
```

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/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface serial 0/3/0/0/0:0
RP/0/RP0/CPU0:router(config-if)# ipv4 address 172.18.189.38 255.255.255.224
RP/0/RP0/CPU0:router(config-if)# encapsulation ppp
RP/0/RP0/CPU0:router(config-if)# no shutdown
RP/0/RP0/CPU0:router(config-if)# ppp authentication chap MIS-access
RP/0/RP0/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/RP0/CPU0:router# configuration
RP/0/RP0/CPU0:router(config)# interface serial 0/3/0/0/0:0
RP/0/RP0/CPU0:router(config-if)# encapsulation ppp
RP/0/RP0/CPU0:router(config-if)# ppp authentication chap
RP/0/RP0/CPU0:router(config-if)# ppp max-bad-auth 3
RP/0/RP0/CPU0:router(config-if)# end
Uncommitted changes found, commit them? [yes]: yes
```

Additional References

The following 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,* Release 3.2
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*
Information about configuring interfaces and other components on the Cisco CRS-1 from a remote Craft Works Interface (CWI) client management application	*Cisco CRS-1 Series Carrier Routing System Craft Works Interface Configuration Guide*

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://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

RFCs	Title
RFC 1294	*Multiprotocol Interconnect Over Frame Relay*
RFC 1490	*Multiprotocol Interconnect Over Frame Relay*
RFC 2427	*Multiprotocol Interconnect Over Frame Relay*
RFC 1586	*Guidelines for Running OSPF Over Frame Relay Networks*
RFC 1315	*Management Information Base for Frame Relay DTEs*
RFC 2115	*Management Information Base for Frame Relay DTEs Using SMIv2*
RFC 1604	*Definitions of Managed Objects for Frame Relay Service*
RFC 2954	*Definitions of Managed Objects for Frame Relay Service*
RFC 2390	*Inverse Address Resolution Protocol*

Technical Assistance

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

Configuring SRP Interfaces on Cisco IOS XR Software

This module describes how to configure the Spatial Reuse Protocol (SRP) on supported Cisco Dynamic Packet Transport (DPT) interfaces in routers running Cisco IOS XR software.

SRP is a MAC-layer protocol developed by Cisco and is used in conjunction with Cisco DPT products. DPT products deliver scalable Internet service, reliable IP-aware optical transport, and simplified network operations. These solutions allow you to scale and distribute your IP services across a reliable optical packet ring infrastructure.

Note Throughout the remainder of this publication, the term SRP is used to describe features related to DPT products.

Feature History for Configuring SRP Interfaces on Cisco IOS XR Software

Release	Modification
Release 3.2.2	This feature was introduced on the Cisco CRS-1 and is supported only on the 4-port OC-192c/STM-64c POS/DPT PLIM.
Release 3.4.0	This command was first supported on the 16-port OC-48c/STM-16c POS/DPT PLIM.
Release 3.5.0	No modifications.

Contents

Prerequisites for Configuring SRP Interfaces

Before configuring SRP interfaces, be sure that the following conditions are met:

- You must be in a user group associated with a task group that includes the proper task IDs for SRP commands. Task IDs for commands are listed in the *SRP Commands on Cisco IOS XR Software*.
- You know the interface IP address you will assign to the new SRP interface configuration.
- The hardware that you are using supports SRP. As of Cisco IOS XR Software Release 3.5, SRP is supported on the 4-port OC-192c/STM-64c POS/DPT PLIM and the 16-port OC-48c/STM-16c POS/DPT PLIM.

Information About Configuring SRP Interfaces

Spatial bandwidth reuse is possible due to the packet destination-stripping property of SRP. Older technologies incorporate source stripping, where packets traverse the entire ring until they are removed by the source. Even if the source and destination nodes are next to each other on the ring, packets continue to traverse the entire ring until they return to the source to be removed. SRP provides more efficient use of available bandwidth by having the destination node remove the packet after it is read. This provides more bandwidth for other nodes on the SRP ring.

SRP rings consists of two counter rotating fibers, known as outer and inner rings, both concurrently used to carry data and control packets. SRP uses both explicit control packets and control information piggybacked inside data packets (control packets handle tasks such as keepalives, protection switching, and bandwidth control propagation). Control packets propagate in the opposite direction from the corresponding data packets, ensuring that the data takes the shortest path to its destination. The use of dual fiber-optic rings provides a high level of packet survivability. In the event of a failed node or a fiber cut, data is transmitted over the alternate ring.

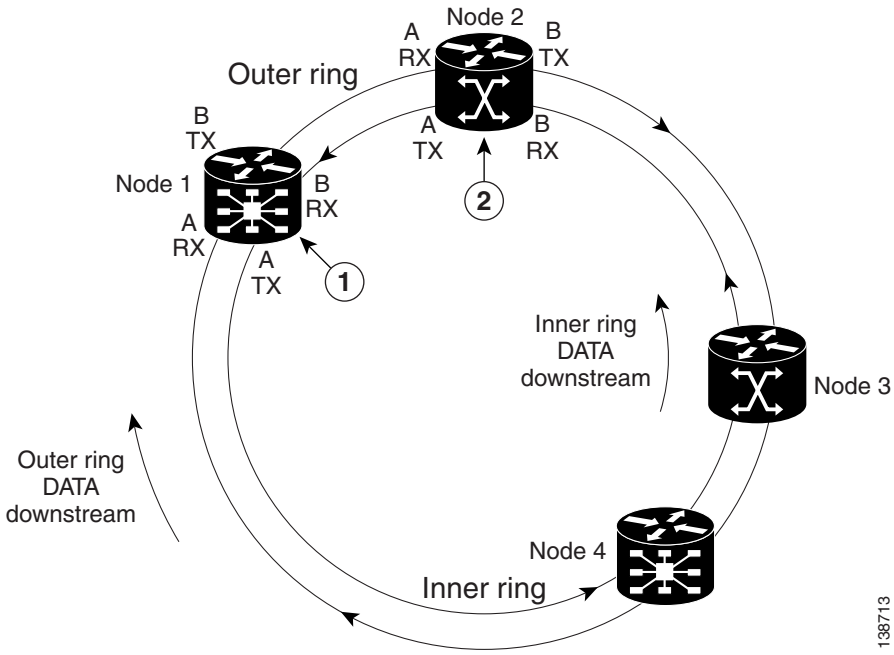
SRP rings are media independent and can operate over a variety of underlying technologies, including SONET/SDH, wavelength division multiplexing (WDM), and dark fiber. This ability to run SRP rings over any embedded fiber transport infrastructure provides a path to packet-optimized transport for high-bandwidth IP networks. Figure 9 shows an SRP ring created with the Cisco CRS-1 and Cisco XR 12000 Series Routers.

To distinguish between the two rings, one is referred to as the “inner” ring and the other as the “outer” ring. SRP operates by sending data packets in one direction (downstream) and sending the corresponding control packets in the opposite direction (upstream) on the other fiber. This allows SRP to use both fibers concurrently to maximize bandwidth for packet transport and to accelerate control signal propagation for adaptive bandwidth utilization, and for self-healing purposes.

As shown in Figure 9, an SRP node uses SRP Side A to receive (RX) outer ring data and transmit (TX) inner ring data. The node uses SRP Side B to receive (RX) inner ring data and transmit (TX) outer ring data. Side A on one node connects to Side B on an adjacent SRP node.

The commands for configuring SRP interfaces are provided in the *Cisco IOS XR Interface and Hardware Component Command Reference*.

Figure 9 SRP Ring Example



1	Cisco CRS-1	**2**	Cisco XR 12000 or 12000 Series Router

How to Configure an SRP Interface

This section contains the following procedures:

- Enabling SRP on a Port, page 385 (required)
- Creating a Basic SRP Configuration, page 388 (required)
- Configuring Intelligent Protection Switching (IPS), page 390 (optional)
- Configuring Modular Quality of Service CLI (MQC) with SRP, page 393 (optional)
- Adding a Node to the Ring, page 396 (optional)

Enabling SRP on a Port

To enable the use of SRP, you must perform this task. By default, POS/DPT PLIMs support only POS.

Restrictions

On the 4-port OC-192c/STM-64c POS/DPT PLIM, each port pair (0 and 1 or 2 and 3) must be configured the same. If you configure port 0 to be SRP and do not configure port 1 to be SRP, the configuration does not work.

On the 16-port OC-48c/STM-16c POS/DPT PLIM, each group of four ports must be configured the same. If you want to use ports 0 and 1 as a single SRP interface, you must configure all four ports: 0, 1, 2, and 3 to be SRP. Similarly, ports 4-7, 8-11, and 12-15 must be configured the same, as either SRP or POS, for the configuration to work.

SUMMARY STEPS

1. **configure**
2. **hw-module port** *port-number-1* **srp location** *instance*
3. **hw-module port** *port-number-2* **srp location** *instance*
4. **hw-module port** *port-number-3* **srp location** *instance*
5. **hw-module port** *port-number-4* **srp location** *instance*
6. **end**
   or
   commit
7. **hw-module location** *node-id* **reload**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**hw-module port** *port-number-1* **srp location** *instance* **Example:** `RP/0/RP0/CPU0:router(config)# hw-module port 0 srp location 0/5/cpu0`	Enables SRP functionality on the first port. **Note** An SRP interface requires two consecutive physical ports for proper configuration. The first, lower-numbered port must be even, for example 0 or 2. **Note** On the 16-port OC-48c/STM-16c POS/DPT PLIM, groups of four consecutive ports must be configured the same: ports 0-3, 4-7, 8-11 and 12-15.
Step 3	**hw-module port** *port-number-2* **srp location** *instance* **Example:** `RP/0/RP0/CPU0:router(config)# hw-module port 1 srp location 0/5/cpu0`	Enables SRP functionality on the second port. **Note** An SRP interface requires two consecutive physical ports for proper configuration. The second, higher-numbered port must be odd. **Note** On the 16-port OC-48c/STM-16c POS/DPT PLIM, groups of four consecutive ports must be configured the same: ports 0-3, 4-7, 8-11, and 12-15.

	Command or Action	Purpose
Step 4	**hw-module port** *port-number-3* **srp location** *instance* **Example:** `RP/0/RP0/CPU0:router(config)# hw-module port 2 srp location 0/5/cpu0`	Enables SRP functionality on the third port, for 16-port OC-48c/STM-16c POS/DPT PLIMs. **Note** An SRP interface requires two consecutive physical ports for proper configuration. The first, lower-numbered port must be even; for example, 0 or 2. **Note** On the 16-port OC-48c/STM-16c POS/DPT PLIM, groups of four consecutive ports must be configured the same: ports 0-3, 4-7, 8-11, and 12-15.
Step 5	**hw-module port** *port-number-4* **srp location** *instance* **Example:** `RP/0/RP0/CPU0:router(config)# hw-module port 3 srp location 0/5/cpu0`	Enables SRP functionality on the fourth port, for 16-port OC-48c/STM-16c POS/DPT PLIMs. **Note** An SRP interface requires two consecutive physical ports for proper configuration. The second, higher-numbered port must be odd. **Note** On the 16-port OC-48c/STM-16c POS/DPT PLIM, groups of four consecutive ports must be configured the same: ports 0-3, 4-7, 8-11, and 12-15.
Step 6	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router(config)# end` or `RP/0/RP0/CPU0:router(config)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 7	**hw-module location** *node-id* **reload** **Example:** `RP/0/RP0/CPU0:router# hw-module location 0/5/cpu0 reload`	Reloads the PLIM and makes the **hw-module port** command become effective.

Note You must reload the PLIM to enable this configuration change and create the SRP interface.

After you complete this procedure, the following SRP interfaces are available to be configured on the PLIM in slot 5:

- 0/5/0/0, which comprises ports 0/5/0/0 and 0/5/0/1
- 0/5/0/2, which comprises ports 0/5/0/2 and 0/5/0/3

Creating a Basic SRP Configuration

This task explains how to create a basic SRP configuration. There are many other possible parameters that can be set and only the most basic are illustrated in this task.

Note You must enable SRP on the interface before you can perform this task. See Enabling SRP on a Port, page 385.

SUMMARY STEPS

1. **show interfaces**
2. **configure**
3. **controller sonet** *instance* **clock source internal**
4. **interface srp** *instance*
5. **ipv4 address** *ip-address mask*
6. **srp topology-timer** *value*
7. **no shutdown**
8. **end**
   or
   commit
9. **show interfaces srp** *instance*
10. **show running-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	`show interfaces` **Example:** `RP/0/RP0/CPU0:router# show interfaces`	(Optional) Displays configured interfaces. • Also use this command to confirm that the router recognizes the PLIM card.
Step 2	`configure` **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**controller sonet** *instance* **clock source internal** **Example:** `RP/0/RP0/CPU0:router(config)# controller sonet 0/1/0/0 clock source internal` `RP/0/RP0/CPU0:router(config)# controller sonet 0/1/0/1 clock source internal`	Configures the SONET port transmit clock source for each port comprising the SRP interface. The controller instance is in the notation *rack/slot/module/port*, and the **internal** keyword specifies internal clock. **Note** Internal clocking is required for SRP interfaces. **Note** Refer to *Configuring Clear Channel SONET Controllers on Cisco IOS XR Software* for more detailed information on the SONET controller configuration.
Step 4	**interface srp** *instance* **Example:** `RP/0/RP0/CPU0:router(config)# interface srp 0/1/0/0`	Specifies the SRP interface name and notation *rack/slot/module/port,* and enters interface configuration mode.
Step 5	**ipv4 address** *ip-address* **Example:** `RP/0/RP0/CPU0:router(config-if)# ipv4 address 10.1.2.1 255.255.255.224`	Assigns an IP address and subnet mask to the interface.
Step 6	**srp topology-timer** *value* **Example:** `RP/0/RP0/CPU0:router(config-if)# srp topology-timer 1`	(Optional) Specifies how frequently topology discovery messages are sent around the ring to identify the current nodes on the SRP ring.
Step 7	**no shutdown** **Example:** `RP/0/RP0/CPU0:router(config-if)# no shutdown`	Removes the shutdown configuration. • The removal of the shutdown configuration removes the forced administrative down state on the interface, enabling it to move to an up or down state.
Step 8	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router(config-if)# end` or `RP/0/RP0/CPU0:router(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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	**show interfaces srp** *instance* **Example:** `RP/0/RP0/CPU0:router# show interfaces srp 0/1/0/0`	(Optional) Displays the SRP interface configuration.
Step 10	**show running-config** **Example:** `RP/0/RP0/CPU0:router# show running-config`	(Optional) Displays the configuration information currently running on the router.

Configuring Intelligent Protection Switching (IPS)

Perform this task to configure IPS on an SRP interface. This is an optional task.

Intelligent Protection Switching (IPS) provides IP self-healing and restoration, and performance monitoring after a link or node failure. There are two SRP IPS modes:

- Automatic SRP IPS mode takes effect when the SRP ring detects an event, a fiber cut, or a node failure, and remains in effect until the trigger condition is cleared. Once the trigger is cleared, the SRP IPS mode remains in effect until the wait-to-restore (WTR) value expires.
- User-configured SRP IPS mode takes effect as soon as you enter the command and remains in effect until it is removed by a user command or overridden by an SRP IPS command with higher priority. You can use the **no srp ips request forced-switch** global configuration command or the **srp remove manual-switch** EXEC command to negate a user-configured command.

A a user-configured, forced-switch adds a high-priority protection switch wrap on each end of a specified span by entering the user-configured **srp ips request forced-switch** command. For example, you can enter an **srp ips request forced-switch** command to force data traffic to one side of the ring before a DPT PLIM is removed from a router slot, or in response to an event.

Table 20 describes the IPS requests in the order of priority, from higher to lower.

Table 20 Explanation of SRP IPS User Requests

SRP IPS Request	Description
Forced-switch	Adds a high-priority protection switch wrap on each end of a specified span by entering the user-configured **srp ips request forced-switch** command.
Manual-switch	Adds a low-priority protection switch wrap on each end of a specified span by entering the user-configured **srp request manual-switch** command.

Note Before removing the DPT PLIM, you can use the **srp ips request forced-switch** command on both sides of the interface that is to be removed.

If an automatic or user-configured protection switch is requested for a given span, the node that receives the protection request issues a protection request to the node on the other end of the span using both the short path over the failed span, because the failure may be unidirectional, and the long path around the ring.

As the protection requests travel around the ring, the protection hierarchy is applied. For example, if a high-priority Signal Fail (SF) request enters the ring, it overrides a preexisting lower-priority request. If an event or a user-configured command enters a low-priority request, it is not allowed if a high-priority request is present on the ring.

Note An exception is that multiple signal-fail and forced-switch requests can coexist on the SRP ring and will bisect the ring if they occur on separate fiber links.

All protection switches are performed bidirectionally and enter wraps at both ends of a span for transmit and receive directions, even if a failure is only unidirectional.

SUMMARY STEPS

1. **configure**
2. **interface srp** instance
3. **srp ips wtr-timer** *seconds*
4. **srp ips timer** *seconds*
5. **srp ips request forced-switch** {**a** | **b**}
6. **end**
   or
   commit
7. **srp** {**request** | **remove**} **manual-switch** {**a** | **b**} **interface srp** *instance*
8. **show srp ips interface srp** *instance*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface srp** *instance* **Example:** `RP/0/RP0/CPU0:router(config)# interface srp 0/1/0/0`	Specifies the SRP interface name in the notation *rack/slot/module/port* and enters interface configuration mode.
Step 3	**srp ips wtr-timer** *seconds* **Example:** `RP/0/RP0/CPU0:router(config-if)# srp ips wtr-timer 60`	(Optional) Configures the amount of time in seconds that a wrap remains in place after the cause of the wrap is removed.

	Command or Action	Purpose
Step 4	**srp ips timer** *seconds* **Example:** `RP/0/RP0/CPU0:router(config-if)# srp ips timer 60 a`	(Optional) Specifies the frequency of the transmission of IPS requests. The default is 1 second. **Note** We recommend that the IPS timer value be the same for all nodes on a ring. Therefore, if the IPS timer value is changed on one node, you should change it for all nodes on the ring using **srp ips timer** command.
Step 5	**srp ips request forced-switch** {**a** | **b**} **Example:** `RP/0/RP0/CPU0:router(config-if)# srp ips request forced-switch a`	(Optional) Adds a high-priority protection switch wrap on each end of a specified span. **Note** Use this command only as necessary, as it disables the node.
Step 6	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router(config-if)# end` or `RP/0/RP0/CPU0:router(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 7	**srp** {**request** | **remove**} **manual-switch** {**a** | **b**} **interface srp** *instance* **Example:** `RP/0/RP0/CPU0:router# srp remove manual-switch a interface srp 0/1/0/0`	(Optional) Adds or removes a low-priority protection switch wrap on each end of a specified span. **Note** Use this command only as necessary.
Step 8	**show srp ips interface srp** *instance* **Example:** `RP/0/RP0/CPU0:router# show srp ips interface srp 0/1/0/0`	(Optional) Displays the IPS configuration on the SRP interface.

Configuring Modular Quality of Service CLI (MQC) with SRP

Perform this task to configure quality-of-service (QoS) classifications with SRP using the Modular QoS command-line interface (MQC) feature. This is an optional task.

Note For more information regarding MQC, refer to *Configuring Modular Quality of Service Packet Classification on Cisco IOS XR Software* and *Cisco IOS XR Modular Quality of Service Command Reference*.

SUMMARY STEPS

1. **configure**
2. **class-map match-any** *access-group-name*
3. **match mpls experimental topmost** *exp-value*
4. **exit**
5. **class-map match-any** *access-group-name*
6. **match precedence** *precedence-value*
7. **exit**
8. **policy-map** *policy-name*
9. **class** *class-name*
10. **police cir** *kbps*
11. **set cos** *cos-value*
12. **priority**
13. **exit**
14. **class** *class-name*
15. **priority**
16. **set cos** *cos-value*
17. **exit**
18. **exit**
19. **interface srp** *instance*
20. **service-policy output** *policy-map*
21. **end**
    or
    commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**class-map match-any** *class-map-name* **Example:** `RP/0/RP0/CPU0:router(config)# class-map match-any voice`	Enters class map configuration mode. • Creates a class map to be used for matching packets to the class whose name you specify. • If you specify **match-any**, one of the match criteria must be met for traffic entering the traffic class to be classified as part of the traffic class.
Step 3	**match mpls experimental topmost** *exp-value* **Example:** `RP/0/RP0/CPU0:router(config-cmap)# match mpls experimental topmost 4`	Configures a class map so that the three-bit experimental (EXP) field in the topmost Multiprotocol Label Switching (MPLS) labels are examined for EXP field values. • The EXP value argument is specified as the exact value from 0 to 7.
Step 4	**exit** **Example:** `RP/0/RP0/CPU0:router(config-cmap)# exit`	Exits the current submode.
Step 5	**class-map match-any** *class-map-name* **Example:** `RP/0/RP0/CPU0:router(config)# class-map match-any ctrl`	Enters class map configuration mode. • Creates a class map to be used for matching packets to the class whose name you specify. • If you specific **match-any**, one of the match criteria must be met for traffic entering the traffic class to be classified as part of the traffic class.
Step 6	**match precedence** *precedence-value* **Example:** `RP/0/RP0/CPU0:router(config-cmap)# match precedence internet`	(Optional) Identifies IP precedence values as match criteria. • The range is from 0 to 63. • Reserved keywords can be specified instead of numeric values.
Step 7	**exit** **Example:** `RP/0/RP0/CPU0:router(config-cmap)# exit`	Exits the current submode.
Step 8	**policy-map** *policy-name* **Example:** `RP/0/RP0/CPU0:router(config)# policy-map srp-policy`	Enters policy map configuration mode. • Creates or modifies a policy map that can be attached to one or more interfaces to specify a service policy.

	Command or Action	Purpose
Step 9	**class** *class-name* **Example:** `RP/0/RP0/CPU0:router(config-pmap)# class voice`	Specifies the name of the class whose policy you want to create or change.
Step 10	**police cir** *kbps* **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# police cir 2000000`	Configures traffic policing. **Note** 2000000 represents 10 percent of the interface line rate.
Step 11	**set cos** *cos-value* **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# set cos 4`	Sets the Layer 2 class of service (CoS) value of an outgoing packet.
Step 12	**priority** **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# priority`	Gives priority to a class of traffic belonging to a policy map. **Note** The **priority** command should only be used if the **set cos** command is also used and specifies a cos value greater then or equal to 2.
Step 13	**exit** **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# exit`	Exits the current submode.
Step 14	**class** *class-name* **Example:** `RP/0/RP0/CPU0:router(config-pmap)# class ctrl`	Specifies the name of the class whose policy you want to create or change.
Step 15	**priority** **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# priority`	Gives priority to a class of traffic belonging to a policy map. **Note** The **priority** command should only be used if the **set cos** command is also used and specifies a cos value greater than or equal to 2.
Step 16	**set cos** *cos-value* **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# set cos 6`	Sets the Layer 2 CoS value of an outgoing packet.
Step 17	**exit** **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# exit`	Exits the current submode.
Step 18	**exit** **Example:** `RP/0/RP0/CPU0:router(config-pmap)# exit`	Exits the current submode.

	Command or Action	Purpose
Step 19	**interface srp** *instance* **Example:** `RP/0/RP0/CPU0:router(config)# interface srp 0/1/0/0`	Specifies the SRP interface in the notation *rack/slot/module/port* and enters interface configuration mode.
Step 20	**service-policy output** *policy-map* **Example:** `RP/0/RP0/CPU0:router(config-if)# service-policy output srp-policy`	Attaches a policy map to an input or output interface to be used as the service policy for that interface. • The traffic policy evaluates all traffic leaving that interface.
Step 21	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router(config-if)# end` or `RP/0/RP0/CPU0:router(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Adding a Node to the Ring

This task describes how to add a node to an existing SRP ring, using Cisco IOS XR commands that insert forced-switch wraps away from the area on the fiber where the node is being added, to ensure a minimal loss of data traffic.

For the purpose of this example, a fifth node is added to a four-node ring. Node 5 is added between Node 1 and Node 4. Figure 10 and Figure 11 show the physical configuration using a single DPT PLIM. Figure 12 and Figure 13 show the logical configuration.

SUMMARY STEPS

1. **configure**
2. **interface srp** *instance*
3. **srp ips request forced-switch** {**a** | **b**}
4. **end**
   or
   commit

5. **interface srp** *instance*
6. **no srp ips request forced-switch** {**a** | **b**}
7. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router1# configure`	Enters global configuration mode.
Step 2	**interface srp** *instance* **Example:** `RP/0/RP0/CPU0:router1(config)# interface srp 0/1/0/0`	Specifies the SRP interface in the notation *rack/slot/module/port* for Node 1 and enters interface configuration mode.
Step 3	**srp ips request forced-switch** {**a** | **b**} **Example:** `RP/0/RP0/CPU0:router1(config-if)# srp ips request forced-switch a`	(Optional) Adds a high-priority protection switch wrap on each end of the specified span. This stops traffic flowing from Node 1 on the fiber that will be disconnected and creates a wrap next to Node 1 on Side A. **Note** If you choose not to use the **srp ips request forced-switch** command, as soon as you perform Step 5, a signal failure is detected by Node 1 and Node 4, and they automatically insert two signal-fail wraps away from the failure between the nodes. We recommend that you use the **srp ips request forced-switch** command to minimize data loss.

	Command or Action	Purpose
Step 4	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router1(config-if)# end` or `RP/0/RP0/CPU0:router1(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 5	Disconnect the fiber-optic cables connecting Node 1 to Node 4.	
Step 6	Connect the cables to add the new node while observing the receive (RX) and transmit (TX) cabling relationship.	See Figure 13.
Step 7	**interface srp** *instance* **Example:** `RP/0/RP0/CPU0:router1(config)# interface srp 0/1/0/0`	Specifies the SRP interface in the notation *rack/slot/module/port* for Node 1 and enters interface configuration mode.
Step 8	**no srp ips request forced-switch** {**a** | **b**} **Example:** `RP/0/RP0/CPU0:router1(config-if)# no srp ips request forced-switch a`	Removes the high-priority protection switch wrap on each end of the specified span. This allows traffic to flow again from Node 1. (See Figure 13.) **Note** If you performed Step 3, then you must use the **no srp ips request forced-switch** command to remove the wraps. If you did not perform Step 3, the wraps are removed automatically when the WTR timer has expired.

	Command or Action	Purpose
Step 9	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router1(config-if)# end` or `RP/0/RP0/CPU0:router1(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 10	**show srp ips** **Example:** `RP/0/RP0/CPU0:router5# show srp ips`	Confirms that the wraps have disappeared and the new node is part of the ring. (See Figure 13.)
Step 11	**show srp errors** **Example:** `RP/0/RP0/CPU0:router5# show srp errors`	Confirms that there are no problems with the new ring configuration. If there are failures, note the status on the LEDs to determine what the problem might be.

Figure 10 Four Routers on the SRP Ring

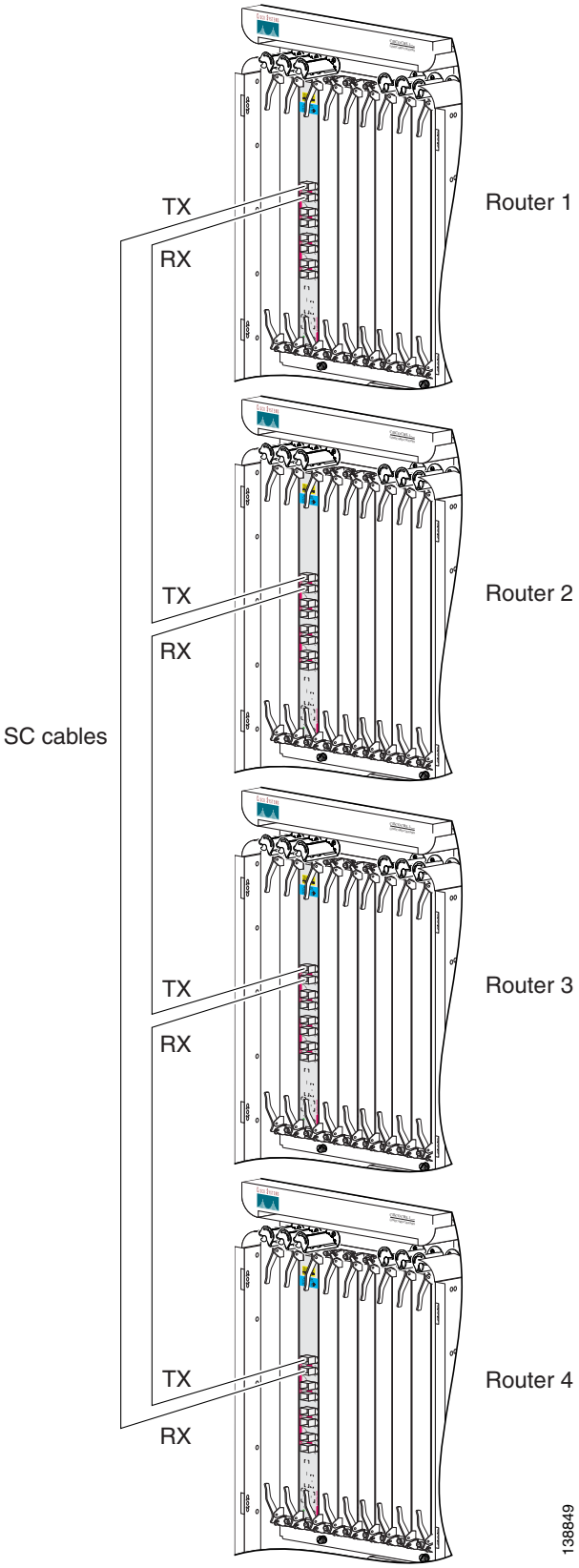


Figure 11 Adding a Router to an SRP Ring

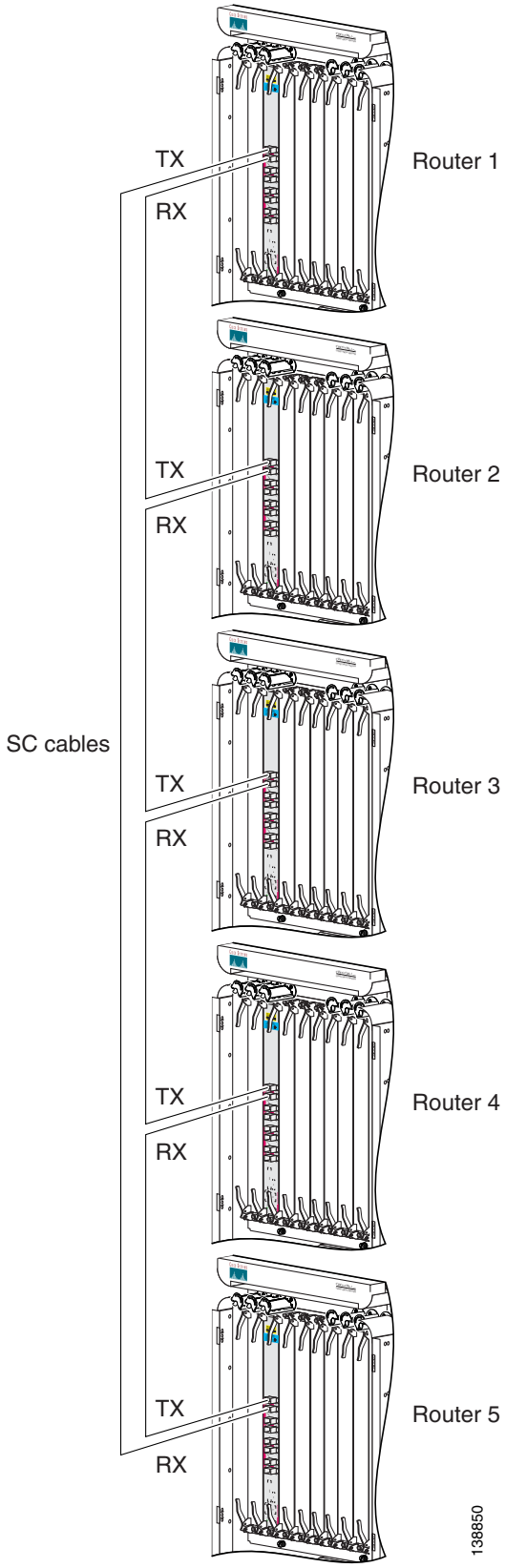
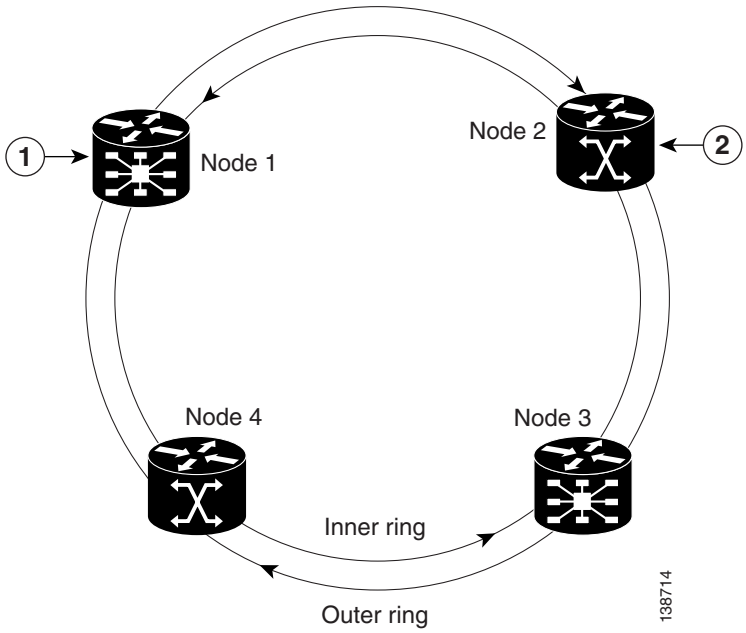
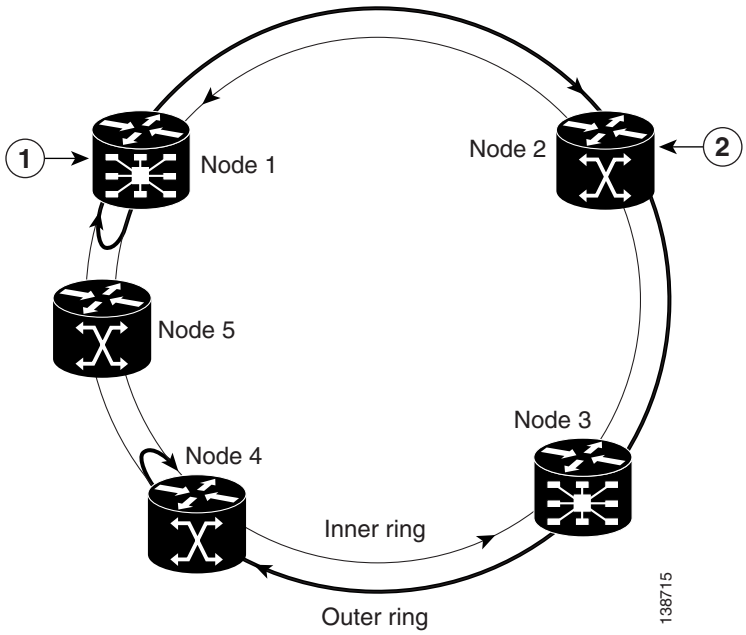


Figure 12 SRP Ring Topology with Four Nodes



1	Cisco CRS-1	**2**	Cisco XR 12000 or 12000 Series Router

Figure 13 SRP Ring Topology with a Fifth Node Added to a Wrapped Ring



1	Cisco CRS-1	**2**	Cisco XR 12000 or 12000 Series Router

Configuration Examples for SRP Interfaces

This section provides the following configuration examples:

Configuring SRP: Example

This example shows how to configure SRP. First the ports are set to SRP mode using the **hw-module port** command, then the PLIM is reloaded, and then the basic interface configuration is performed.

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# hw-module port 0 srp location 0/3/CPU0
RP/0/RP0/CPU0:router(config)# hw-module port 1 srp location 0/3/CPU0
RP/0/RP0/CPU0:router(config)# controller SONET 0/3/0/0 clock source internal
RP/0/RP0/CPU0:router(config)# controller SONET 0/3/0/1 clock source internal
RP/0/RP0/CPU0:router(config)# commit
RP/0/RP0/CPU0:router(config)# end
RP/0/RP0/CPU0:router# hw-module node 0/3/CPU0 reload

<Wait for LC to be reloaded, and interface created.  Or can use 'preconfigure'...>

RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# interface SRP 0/3/0/0
RP/0/RP0/CPU0:router(config-if)# ipv4 address 172.18.189.38 255.255.255.224
RP/0/RP0/CPU0:router(config-if)# no shutdown
RP/0/RP0/CPU0:router(config-if)# commit
RP/0/RP0/CPU0:router(config)# end
```

Configuring Modular QoS with SRP: Example

This example shows how to configure two quality-of-service (QoS) classes. One is for voice traffic and is identified by an MPLS experimental bit value of 4; the second is control traffic that is identified by an IP precedence value of 6. Both classes of traffic are sent to the SRP high priority queue and are marked with high SRP priority (4 and 6).

```
Last configuration change at 04:56:06 UTC Tue Sep 06 2005 by lab
!
hostname router
class-map match-any ctrl
 match precedence internet
!
class-map match-any voice
 match mpls experimental topmost 4
!
policy-map srp-policy
 class voice
  police cir 2000000
  set cos 4
  priority
 !
 class ctrl
  priority
  set cos 6
 !
!
```

```
interface SRP0/7/0/0
 description "Connected to 3-nodes ring"
 service-policy output srp-policy
 ipv4 address 30.30.30.2 255.255.255.0
```

Additional References

The following sections provide references related to SRP interface configuration.

Related Documents

Related Topic	Document Title
Cisco IOS XR master command reference	*Cisco IOS XR Master Commands List,* Release 3.2
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 Task ID Reference Guide*
Information about configuring interfaces and other components on the Cisco CRS-1 from a remote Craft Works Interface (CWI) client management application	*Cisco CRS-1 Series Carrier Routing System Craft Works Interface 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
—	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://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

RFCs	Title
RFC-2892	*The Cisco SRP MAC Layer Protocol*

Technical Assistance

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

Configuring Clear Channel T3/E3 Controllers and Channelized T3 Controllers on Cisco IOS XR Software

This module describes the configuration of clear channel T3/E3 controllers and channelized T3 controllers on routers supporting Cisco IOS XR software. 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. Support was added on the Cisco XR 12000 Series Router for the following SIPS: • Cisco XR 12000 SIP-401 • Cisco XR 12000 SIP-501 • Cisco XR 12000 SIP-601 Support was added on the Cisco XR 12000 Series Router for the following SPAs: • 2-Port and 4-Port Channelized T3 SPA • 2-Port Clear Channel T3/E3 SPA
Release 3.4.0	No modifications.
Release 3.4.1	This feature was introduced on the Cisco CRS-1 for the 4-Port Clear Channel T3/E3 SPA.
Release 3.5.0	This feature was introduced on the Cisco XR 12000 Series Router for the Cisco 1-Port Channelized DS0/OC-12 shared port adapter.

Contents

Prerequisites for Configuring T3/E3 Controllers

Before configuring T3/E3 controllers, be sure 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 for T3/E3 commands. Task IDs for commands are listed in *Cisco IOS XR Interface and Hardware Component Command Reference*.
- If you are configuring a channelized T3 controller, you must be in a user group associated with a task group that includes the proper task IDs for T1/E1 commands. Task IDs for commands are listed in *Cisco IOS XR Interface and Hardware Component Command Reference*.
- Your hardware must support T3/E3 controllers and serial interfaces. The following hardware supports T3/E3 controllers and serial interfaces in Cisco IOS XR Software Release 3.3:
  - 2-Port and 4-Port Clear Channel T3/E3 SPAs
  - 2-Port and 4-Port Channelized T3 SPAs

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 34010 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.

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 Software* module later in this document.

Default Configuration Values for T3 and E3 Controllers

Table 21 describes the default configuration parameters that are present on the T3 and E3 controllers.

Table 21 T3 and E3 Controller Default Configuration Values

Parameter	Default Value	Configuration File Entry
Frame type for the data line	For T3: C-bit framing For E3: G.751	**framing** {**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 (T1 only)	**disable**	**mdl transmit** {**idle-signal** | **path** | **test-signal**} {**enable** | **disable**}
National reserved bits for an E3 port (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 Controllers

Table 22 describes the default configuration parameters that are present on the T1 controllers.

Table 22 T1 Controller Default Configuration Values

Parameter	Default Value	Configuration File Entry
Frame type for the data line	Extended superframe (**esf**)	**framing** {**sf** | **esf**}
Detection and generation of T1 yellow alarms.	Yellow alarms are detected and generated on the T1 channel.	**yellow** {**detection** | **generation**} {**disable** | **enable**}
Clocking for individual T1 links	**internal**	**clock source** {**internal** | **line**}
Transmission of ANSI T1.403 once-per-second remote performance reports through Facility Data Link (FDL) for a T1 channel	**enable**	**fdl ansi** {**enable** | **disable**}

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 E1 Controllers

Table 23 describes the default configuration parameters that are present on the E1 controllers.

Table 23 E1 Controller Default Configuration Values

Parameter	Default Value	Configuration File Entry
Frame type for the data line	Framing with CRC-4 error monitoring capabilities (**crc**).	**framing** {**crc** | **no-crc** | **unframed**
Clocking for individual T1 links	internal	**clock source** {**internal** | **line**}
National reserved bits for an E1 port	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 for the Clear Channel SPAs

By default, the 2-Port and 4-Port Clear Channel T3/E3 SPAs boot in T3 mode. If you want to use the card in E3 mode you must set the card type to be E3, as described in this procedure that follows.

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 You must set the card type on the 2-Port and 4-Port Clear Channel T3/E3 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, and now you want to change the card type, you must delete any previously defined T3/E3 controller and serial interface configurations. Use the **no controller** [**t3**|**e3**] 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 only.

SUMMARY STEPS

1. **configure**
2. **hw-module subslot** *subslot-id* **cardtype** {**t3** | **e3**}
3. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**hw-module subslot** *subslot-id* **cardtype** {**t3** | **e3**} **Example:** `RP/0/0/CPU0:router(config)# hw-module subslot 0/1/0 cardtype e3`	Sets the serial mode for the SPA. • **t3**—Specifies T3 connectivity of 44,210 kbps through the network, using B3ZS coding. This is the default setting. • **e3**—Specifies a wide-area digital transmission scheme used predominantly in Europe that carries data at a rate of 34,010 kbps.
Step 3	**end** or **commit** **Example:** `RP/0/0/CPU0:router(config)# end` or `RP/0/0/CPU0:router(config)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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** *instance*
3. **mode serial**
4. **no shutdown**
5. **end**
   or
   commit
6. **show controllers e3** *instance*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**controller e3** *instance* **Example:** `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** **Example:** `RP/0/0/CPU0:router(config-e3)# mode serial`	Configures the mode of the port to be clear channel serial. **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.

	Command or Action	Purpose
Step 4	**no shutdown** **Example:** `RP/0/0/CPU0:router(config-e3)# no shutdown`	Removes the shutdown configuration. • 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** or **commit** **Example:** `RP/0/0/CPU0:router(config-e3)# end` or `RP/0/0/CPU0:router(config-e3)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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** *instance* **Example:** `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 Software* 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 chapter.

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** *instance*
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** *instance*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**controller e3** *instance* **Example:** `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**} **Example:** `RP/0/0/CPU0:router(config-e3)# clock source internal`	(Optional) Sets the clocking for individual E3 links. **Note** The default clock source is **internal**. **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.

	Command or Action	Purpose
Step 4	**cablelength** *feet* **Example:** `RP/0/0/CPU0:router(config-e3)# cablelength 250`	(Optional) Specifies the distance of the cable from the router to the network equipment. **Note** The default cable length is 224 feet.
Step 5	**framing** {**g751** | **g832**} **Example:** `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. **Note** The default framing for E3 is G.751.
Step 6	**national bits** {**disable** | **enable**} **Example:** `RP/0/0/CPU0:router(config-e3)# national bits enable`	(Optional) Enables or disables the 0x1F national reserved bit pattern on the E3 port. **Note** The E3 national bit is enabled by default, and the bit pattern value is 1.
Step 7	**no shutdown** **Example:** `RP/0/0/CPU0:router(config-e3)# no shutdown`	Removes the shutdown configuration. • 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** or **commit** **Example:** `RP/0/0/CPU0:router(config-e3)# end` or `RP/0/0/CPU0:router(config-e3)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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** *instance* **Example:** `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 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 Software* 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 for the Clear Channel SPAs 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** *instance*
3. **mode serial**
4. **no shutdown**
5. **end**
   or
   commit
6. **show controllers t3** *instance*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**controller t3** *instance* **Example:** `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** **Example:** `RP/0/0/CPU0:router(config-t3)# mode serial`	Configures the mode of the port to be clear channel serial. **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** **Example:** `RP/0/0/CPU0:router(config-t3)# no shutdown`	Removes the shutdown configuration. • 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** or **commit** **Example:** `RP/0/0/CPU0:router(config-t3)# end` or `RP/0/0/CPU0:router(config-t3)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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** *instance* **Example:** `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 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 Software* module later in this document.

Configuring a Channelized T3 Controller

The 2-Port and 4-Port Channelized T3 SPAs 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.

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** *instance*
3. **mode** [**t1** | **e1**]
4. **no shutdown**
5. **end**
   or
   commit
6. **show controllers t3** *instance*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**controller T3** *instance* **Example:** `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** **Example:** `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** **Example:** `RP/0/0/CPU0:router(config-t3)# no shutdown`	Removes the shutdown configuration. • 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** or **commit** **Example:** `RP/0/0/CPU0:router(config-t3)# end` or `RP/0/0/CPU0:router(config-t3)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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** *instance* **Example:** `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.
- 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 module later in this document.
- 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 module later in this document.

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 earlier in this chapter.

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** *instance*
3. **clock source** {**internal** | **line**}
4. **cablelength** *feet*
5. **framing** {**c-bit** | **m23**}
6. **mdl transmit** {**idle-signal** | **path** | **test-signal**} {**enable** | **disable**}
7. **mdl string** {**eic** | **fi** | **fic** | **gen-number** | **lic** | **port-number** | **unit**} *string*
8. **no shutdown**
9. **end**
   or
   commit
10. **show controllers t3** *instance*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**controller T3** *instance* **Example:** `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**} **Example:** `RP/0/0/CPU0:router(config-t3)# clock source internal`	(Optional) Sets the clocking for the T3 port. **Note** The default clock source is **internal**. **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* **Example:** `RP/0/0/CPU0:router(config-t3)# cablelength 250`	(Optional) Specifies the distance of the cable from the router to the network equipment. **Note** The default cable length is 224 feet.
Step 5	**framing** {**c-bit** | **m23**} **Example:** `RP/0/0/CPU0:router(config-t3)# framing c-bit`	(Optional) Selects the frame type for the T3 port. **Note** The default frame type for T3 is C-bit.
Step 6	**mdl transmit** {**idle-signal** | **path** | **test-signal**} {**enable** | **disable**} **Example:** `RP/0/0/CPU0:router(config-t3)# mdl transmit path enable`	(Optional) Enables Maintenance Data Link (MDL) messages on the T3 port. **Note** MDL messages are supported only when the T3 framing is C-bit parity. **Note** MDL message are disabled by default.
Step 7	**mdl string** {**eic** | **fi** | **fic** | **gen-number** | **lic** | **port-number** | **unit**} *string* **Example:** `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** **Example:** `RP/0/0/CPU0:router(config-t3)# no shutdown`	Removes the shutdown configuration. • 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 9	**end** or **commit** **Example:** `RP/0/0/CPU0:router(config-t3)# end` or `RP/0/0/CPU0:router(config-t3)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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** *instance* **Example:** `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 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 Software* module later in this document.
- 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 module later in this document.
- 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 module later in this document.

Configuring a T1 Controller

A single T3 controller can be channelized into 28 T1 controllers, and each T1 controller can be further channelized into up to 24 DS0 time slots. This task describes how to configure an individual T1 controller and channelize it into individual DS0 timeslots. Perform this task after you channelize a T3 controller into 28 individual T1 controllers.

Prerequisites

- You must have a 2-Port or 4-Port Channelized T3 SPA installed in your router.
- You must configure a channelized T3 controller to run in T1 mode, as described in Configuring a Channelized T3 Controller section earlier in this module.

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** *instance*
2. **configure**
3. **controller t1** *instance*
4. **framing** {**sf** | **esf**}
5. **yellow** {**detection** | **generation**} {**disable** | **enable**}
6. **clock source** {**internal** | **line**}
7. **fdl ansi** {**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** *instance* **Example:** `RP/0/0/CPU0:router# show controllers t3 0/1/0/0`	(Optional) Displays information about the T1controllers you created in Step 3.
Step 2	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 3	**controller t1** *instance* **Example:** `RP/0/0/CPU0:router(config)# controller t1 0/3/0/0/0`	Enters T1 configuration mode.
Step 4	**framing** {**sf** | **esf**} **Example:** `RP/0/0/CPU0:router(config-t1)# framing esf`	(Optional) Selects the frame type for the T1 data line: • **sf**—Superframe • **esf**—Extended super frame **Note** The default frame type for T1 is Extended superframe (**esf**).
Step 5	**yellow** {**detection** | **generation**} {**disable** | **enable**} **Example:** `RP/0/0/CPU0:router(config-t1e1)# yellow detection enable`	(Optional) Enables or disables the detection and generation of T1 yellow alarms. **Note** Yellow alarms are detected and generated on the T1 channel by default.
Step 6	**clock source** {**internal** | **line**} **Example:** `RP/0/0/CPU0:router(config-t1e1)# clock source internal`	(Optional) Sets the clocking for individual T1 links. **Note** The default clock source is **internal**. **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** {**enable** | **disable**} **Example:** `RP/0/0/CPU0:router(config-t1e1)# fdl ansi enable`	(Optional) Enables the transmission of ANSI T1.403 once-per-second remote performance reports through Facility Data Link (FDL). **Note** FDL ansi is enabled by default.
Step 8	**no shutdown** **Example:** `RP/0/0/CPU0:router(config-t1e1)# no shutdown`	Removes the shutdown configuration. • 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* **Example:** `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* **Example:** `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. • Range is from 1 to 24 time slots. • You can assign all 24 time slots to a single channel group, or you can divide the time slots among several channel groups. **Note** Each individual T1 controller supports a total of 24 DS0 time slots.
Step 11	**speed** *kbps* **Example:** `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. **Note** The default speed is 64 kbps.
Step 12	**exit** **Example:** `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** **Example:** `RP/0/0/CPU0:router(config-t1)#`	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** or **commit** **Example:** `RP/0/0/CPU0:router(config-t3)# end` or `RP/0/0/CPU0:router(config-t3)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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 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 Software* module later in this document.

Configuring an E1 Controller

The 2-Port and 4-Port Channelized T3 SPAs support channelization to E1 controllers. A single T3 controller can be channelized into 21 E1 controllers, and each E1 controller can be further channelized into up to 31 DS0 timeslots.

Prerequisites

You must configure a channelized T3 controller to run in E1 mode, as described in Configuring a Channelized T3 Controller section earlier in this module.

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** *instance*
2. **configure**
3. **controller e1** *instance*
4. **clock source** {**internal** | **line**}
5. **framing** {**crc** | **no-crc** | **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** *instance* **Example:** `RP/0/0/CPU0:router# show controllers e1 0/1/0/0`	(Optional) Displays information about the E1controllers.
Step 2	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 3	**controller e1** *instance* **Example:** `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**} **Example:** `RP/0/0/CPU0:router(config-e1)# clock source internal`	(Optional) Sets the clocking for individual E1 links. **Note** The default clock source is **internal**. **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** {**crc** | **no-crc** | **unframed**} **Example:** `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: • **crc**—framing with CRC-4 error monitoring capabilities • **no-crc**—framing without CRC • **unframed**—unframed E1 **Note** The default frame type for E1 is **crc**.
Step 6	**national bits** *bits* **Example:** `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. **Note** The default bit pattern is 0, which corresponds to the hexadecimal value *0x1f*.
Step 7	**no shutdown** **Example:** `RP/0/0/CPU0:router(config-e1)# no shutdown`	Removes the shutdown configuration. • 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* **Example:** `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* **Example:** `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. • Range is from 1 to 31 time slots. • You can assign all 31 time slots to a single channel group, or you can divide the time slots among several channel groups. **Note** Each E1 controller supports a total of 31 DS0 time slots.
Step 10	**speed** *kbps* **Example:** `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. **Note** The default speed is 64 kbps.

	Command or Action	Purpose
Step 11	**exit** **Example:** `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** **Example:** `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** or **commit** **Example:** `RP/0/0/CPU0:router(config-e3)# end` or `RP/0/0/CPU0:router(config-e3)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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 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 Software* module later in this document.

Configuring BERT

Bit error rate testing (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.

SUMMARY STEPS

1. **configure**
2. **controller** [**t3** | **e3** | **t1** | **e1**] *instance*
3. **bert pattern** *pattern*
4. **bert interval** *time*
5. **bert error** [*number*]
6. **end**
   or
   commit
7. **exit**
8. **exit**
9. **bert** [**t3** | **e3** | **t1** | **e1**] *instance* [**channel-group** *channel-group-number*] [**error**] **start**
10. **bert** [**t3** | **e3** | **t1** | **e1**] *instance* [**channel-group** *channel-group-number*] **stop**
11. **show controllers** [**t3** | **e3** | **t1** | **e1**] *instance*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**controller** [**t3** | **e3** | **t1** | **e1**] *instance* **Example:** `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* **Example:** `RP/0/0/CPU0:router(config-t3)# bert pattern 2^15`	Enables a specific bit error rate test (BERT) pattern on a controller. Valid patterns include: **0s**, **1s**, **2^15 2^20**, **2^20-QRSS**, **2^23**, **alt-0-1**. and **none**. **Note** You must use the **bert** command in EXEC mode to start the BER test.
Step 4	**bert interval** *time* **Example:** `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*] **Example:** `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** or **commit** **Example:** `RP/0/0/CPU0:router(config-t3)# end` or `RP/0/0/CPU0:router(config-t3)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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** **Example:** `RP/0/0/CPU0:router(config-t3)# exit`	Exits T3/E3 or T1/E1 controller configuration mode.
Step 8	**exit** **Example:** `RP/0/0/CPU0:router(config)# exit`	Exits global configuration mode.
Step 9	**bert** [**t3** | **e3** | **t1** | **e1**] *instance* [**channel-group** *channel-group-number*] [**error**] **start** **Example:** `RP/0/0/CPU0:router# bert t3 0/3/0/0 error start`	Starts the configured BERT test on the specified T3/E3 or T1/E1 controller. **Note** For T3/E3 controllers, you can include the optional **error** keyword to inject errors into the running BERT stream. The **error** keyword is not available on T1/E1 interfaces.
Step 10	**bert** [**t3** | **e3** | **t1** | **e1**] *instance* [**channel-group** *channel-group-number*] **stop** **Example:** `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**] *instance* **Example:** `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 Software* module later in this document.

Configuring BERT on a DS0 Channel Group

This task explains how to enable a bit error rate test (BERT) pattern on a an individual DS0 channel group.

Prerequisites

You must have configured a clear channel T3/E3 controller, or a channelized T3-to-T1/E1 controller.

SUMMARY STEPS

1. **configure**
2. **controller** [**t1**| **e1**] *instance*
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**] *instance* **start**
11. **bert** [**t1**| **e1**] *instance* **stop**
12. **show controllers** [**t1**| **e1**] *instance*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**controller** [**t1** | **e1**] *instance* **Example:** `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* **Example:** `RP/0/0/CPU0:router(config-t1)# channel-group 1` `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* **Example:** `RP/0/0/CPU0:router(config-t1-channel_group)# bert pattern 2^15`	Enables a specific bit error rate test (BERT) pattern on a T3 line. Valid patterns include: **0s**, **1s**, **2^15 2^20**, **2^20-QRSS**, **2^23**, **alt-0-1**. and **none**. **Note** You must use the **bert** command in EXEC mode to start the BER test.
Step 5	**bert interval** *time* **Example:** `RP/0/0/CPU0:router(config-t1-channel_group)# 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.

	Command or Action	Purpose
Step 6	**end** or **commit** **Example:** `RP/0/0/CPU0:router(config-t1-channel_group)# end` or `RP/0/0/CPU0:router(config-t1-channel_group)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 7	**exit** **Example:** `RP/0/0/CPU0:router(config-t1-channel_group)# exit`	Exits channel group configuration mode.
Step 8	**exit** **Example:** `RP/0/0/CPU0:router(config-t1)# exit`	Exits T1 or E1 configuration mode.
Step 9	**exit** **Example:** `RP/0/0/CPU0:router(config)# exit`	Exits global configuration mode.
Step 10	**bert** [**t1** | **e1**] *instance* **channel-group** *channel-group-number* **start** **Example:** `RP/0/0/CPU0:router# bert t1 0/3/0/0/0 error start`	Starts the configured BERT test on the specified channel group.

	Command or Action	Purpose
Step 11	**bert** [**t1** | **e1**] *instance* **channel-group** *channel-group-number* **stop** **Example:** `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** | **e3**] *instance* **Example:** `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 Software* module later in this document.

Examples

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
RP/0/0/CPU0:router(config-t3)#
```

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(config)#
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
```

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

The following 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,* Release 3.2
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*
Information about configuring interfaces and other components on the Cisco CRS-1 from a remote Craft Works Interface (CWI) client management application	*Cisco CRS-1 Series Carrier Routing System Craft Works Interface 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
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: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

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/techsupport

Configuring Tunnel Interfaces on Cisco IOS XR Software

This module describes the configuration of Tunnel-IPSec interfaces on routers supporting Cisco IOS XR Software. 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 failover, 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 2.0	This feature was introduced on the Cisco CRS-1.
Release 3.0	No modification.
Release 3.2	No modification.
Release 3.3.0	No modification.
Release 3.4.0	No modification.
Release 3.5.0	No modifications.

Contents

Prerequisites for Configuring Tunnel Interfaces

Before configuring tunnel interfaces, be sure that the following condition is met:

- You must be in a user group associated with a task group that includes the proper task IDs for tunnel commands.
  - The task ID for the **interface tunnel-ipsec** command is listed in the *Cisco IOS XR Multiprotocol Label Switching Command Reference*.
  - Task IDs for the **tunnel source**, **tunnel destination**, **profile** and commands are listed *Cisco IOS XR System Security Command Reference.*
  - The task ID for the **ping** command is listed in the *Cisco IOS XR IP Addresses and Services Command Reference.*

Information About Configuring Tunnel Interfaces

To implement tunnel interfaces, you must understand the following concepts:

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-te** *number* or **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*}.

Release 2.0 of the Cisco IOS XR Software supports MPLS traffic-engineering. Release 3.0 supports IPSec tunnels for secure communications over public lines.

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 448. 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.

MPLS Traffic Engineering Overview

MPLS traffic engineering software enables an MPLS backbone to replicate and expand upon 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. Thus, you can offer in a one-tier network what has been achieved only by overlaying a Layer 3 network on a Layer 2 network.

Traffic engineering is essential for service provider and Internet service provider (ISP) backbones. Such backbones must support a high use of transmission capacity, and the networks must be very resilient so that they can withstand link or node failures. MPLS traffic engineering provides an integrated approach to traffic engineering. With MPLS, traffic engineering capabilities are integrated into Layer 3, which optimizes the routing of IP traffic, given the constraints imposed by backbone capacity and topology

For more information on MPLS TE, see the *Implementing MPLS Traffic Engineering on Cisco IOS XR Software* module of *Cisco IOS XR Multiprotocol Label Switching Configuration Guide*.

How to Configure Tunnel Interfaces

This section contains the following procedures:

- Configuring Tunnel-IPSec Interfaces, page 445 (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. **ping** *ip-address*
8. **show ip route**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface tunnel-ipsec** *identifier* **Example:** `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* **Example:** `RP/0/RP0/CPU0:router(config-if)# profile user1`	Assigns the crypto profile name to be applied to the tunnel for IPSec processing. • The same crypto profile cannot be shared in different IPSec modes.
Step 4	**tunnel source** (*ip-address* | *interface-id*) **Example:** `RP/0/RP0/CPU0:router(config-if)# tunnel source Ethernet0/1/1/2`	Specifies the tunnel source IP address or interface ID. • This command is required for both static and dynamic profiles.
Step 5	**tunnel destination** {*ip-address* | *tunnel-id*} **Example:** `RP/0/RP0/CPU0:router(config-if)# tunnel destination 192.168.164.19`	(Optional) Specifies the tunnel destination IP address. • This command is not required if the profile is dynamic.
Step 6	**end** or **commit** **Example:** `RP/0/RP0/CPU0:router(config-if)# end` or `RP/0/RP0/CPU0:router(config-if)# commit`	Saves configuration changes. • When you issue the **end** command, the system prompts you to commit changes: `Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:` – Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. – Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. – Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. • 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	**ping** *ip-address* **Example:** `RP/0/RP0/CPU0:router(config)# ping 192.168.164.19`	Checks for connectivity to a particular IP address.
Step 8	**show ip route** **Example:** `RP/0/RP0/CPU0:router# show ip route`	Displays forwarding information for the tunnel. • 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 447

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 esp-sha-hmac
RP/0/RP0/CPU0:router(config)# end
Uncommitted changes found, commit them? [yes]: 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 discover
RP/0/RP0/CPU0:router(config-user1)# isakmp authorization list list1
RP/0/RP0/CPU0:router(config-user1)# client authentication list list2
RP/0/RP0/CPU0:router(config-user1)# exit
RP/0/RP0/CPU0:router(config)# crypto isakmp client configuration group group1
RP/0/RP0/CPU0:router(config-group)# end
Uncommitted changes found, commit them? [yes]: yes


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? [yes]: 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.*

For information on MPLS-TE, see the *Implementing MPLS Traffic Engineering on Cisco IOS XR Software* module of *Cisco IOS XR Multiprotocol Label Switching Configuration Guide*.

Additional References

The following 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,* Release 3.2
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	*Cisco IOS XR Multiprotocol Label Switching Configuration Guide*
Information about user groups and task IDs	*Cisco IOS XR Interface and Hardware Component Command Reference*
Information about configuring interfaces and other components on the Cisco CRS-1 from a remote Craft Works Interface (CWI) client management application	*Cisco CRS-1 Series Carrier Routing System Craft Works Interface 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: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

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/techsupport

Symbols

A

B

C

D

E

F

H

I

K

L

M

N

O

P

R

S

T

U

V

W

Y